Analysis of influencing factors of the number of short video likes and establishment of clustering, generative, and classification models

2021/12/29

Group members:

Zhengqian Cui 11912505; Hanbin Liu 11912410; Haoyv Liao 11911921

Data Source: Bear Club

Background:

Short video refers to high-frequency video content that is played on a variety of new media, suitable for viewing in the mobile state and short-term leisure state, and the duration is generally less than 5 minutes. According to data from iResearch Consulting, in 2018, the number of short video users in China reached 501 million. It is expected that the user scale of China's short video industry will continue to grow steadily in the future. The content value, popularity, and spread of short videos can be reflected by key indicators such as the number of likes. This case studies the influencing factors of the number of short video likes, so as to deepen people's understanding of the factors affecting short video preferences, and summarizes a set of feasible methods that can effectively increase the number of short video likes, helping operators to better account and Video management.

Data:

Each column corresponds to: serial number, author number, number of likes, number of comments, number of shares, BGM, duration, release date, release time, type, title number.

Contents:

1 Data Pre-processing & Exploratory Data Analysis(EDA)

1.1 Data clean

* Sorry, we cannot display Chinese in the report. Therefore, we had to switch to pictures (black pictures). But don't worry, we only need to do this in this section(1.1 Data clean).

1.1.1 Data import

```
video <-
  read.csv("       /data.csv",
           header = TRUE,
           encoding = "UTF-8")
names(video)
```

```
video <-
  read.csv("短视频点赞量影响因素分析/data.csv",
           header = TRUE,
           encoding = "UTF-8")
names(video)
```

```
##  [1] " "     "  " " "  "   " "   "  "   "BGM"
##  [7] " "     "  " " "  " " " "     "  "
```

```
##  [1] "序号"     "作者编号" "点赞数"   "评论数"   "分享数"   "BGM"
##  [7] "时长"     "发布日期" "发布时间" "类别"     "标题字数"
```

- Data Overview (Data cannot be provided due to copyright reasons)

```
names(video)[1:5] <-
  c("index", "author", "like", "comment", "share")
names(video)[7:11] <- c("duration", "  ", "  ", "type", "title")
```

```
names(video)[1:5] <-
  c("index", "author", "like", "comment", "share")
names(video)[7:11] <- c("duration", "发布日期", "发布时间", "type", "title")
```

1.1.2 Missing values and duplicated rows

```
## missing values
any(is.na(video))
```

```
## [1] FALSE
```

```
## duplicated rows
sel_video <- video[, c("index", "like", "comment", "share")]
dup <-
  sel_video[duplicated(sel_video[,-1]) |
              duplicated(sel_video[,-1], fromLast = TRUE), ]
ind <- dup[order(dup$like),]$index

duplicated <- video[ind, -6][, -10]
#duplicated
```

- Duplicated rows (Data cannot be provided due to copyright reasons)

There are duplicated rows and many same videos but come from different type. The reason is unknown, but it's clear that a short video with no more than 60 seconds should not be contained in several different types. Hence, we have to see such duplicated observations as wrong observations, which can't provide proper information to our analysis. So we decide to delete them.

Authors of these videos

```
sort(table(duplicated$author), decreasing = TRUE)
```

```
##
##    7 2162  480   38 1325   22 2175    1  487  603  155  482  661   94   98  273
##   44   24   22   20   14   12   12   10   10   10    8    8    6    4    4    4
##  691  717  160  121  179  289  358  440  449  478  496  622  635  677  692  736
##    4    4    3    2    2    2    2    2    2    2    2    2    2    2    2    2
##  749  769  770  773  790  796 1192 1469 1841 2167
##    2    2    2    2    2    2    2    2    2    2
```

Author7 have 44 these kind of videos!

```
## save the index
dup <- ind

## delete them
video_origin <- video
video <- video[-ind,]
```

1.1.3 split day and time variables

```
library(lubridate)
```

```
##
##     'lubridate'

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

```
time <- paste(video$   , video$   )
time <- strftime(time)
```

```
time <- paste(video$发布日期, video$发布时间)
time <- strftime(time)
```

```
video$quarter <- quarter(time)
video$month <- month(time)
video$week <- week(time)
video$weekday <- wday(time)
video$day <- day(time)
video$hour <- hour(time)
video$minute <- minute(time)
video$second <- second(time)

video[1:5, ]
```

```
##   index author    like comment  share            BGM duration
## 2     2      2 3263395   59098 214594         58.33  2019/1/4
## 3     3      2 2962712   52549 109605         57.60 2019/2/11
## 4     4      2 2953051   55461 215838         50.80 2019/1/26
## 5     5      2 2821680   52263 125418         51.57 2019/1/29
## 6     6      3 2712442   48474 345495         31.73 2019/2/25
##       type title quarter month week weekday day hour minute second
## 2 17:17:59      11       1     1    1       6   4   17     17     59
## 3 17:33:23      12       1     2    6       2  11   17     33     23
## 4 17:44:18      12       1     1    4       7  26   17     44     18
## 5 17:06:04      13       1     1    5       3  29   17      6      4
## 6 16:25:15      21       1     2    8       2  25   16     25     15
```

```
video <- video[, -c(8, 9)]
video[1:5, ]
```

```
##   index author    like comment  share            BGM duration type title
## 2     2      2 3263395   59098 214594         58.33       11
## 3     3      2 2962712   52549 109605         57.60       12
## 4     4      2 2953051   55461 215838         50.80       12
## 5     5      2 2821680   52263 125418         51.57       13
## 6     6      3 2712442   48474 345495         31.73       21
##   quarter month week weekday day hour minute second
## 2       1     1    1       6   4   17     17     59
## 3       1     2    6       2  11   17     33     23
## 4       1     1    4       7  26   17     44     18
## 5       1     1    5       3  29   17      6      4
## 6       1     2    8       2  25   16     25     15
```

1.1.4 hour —> time period

```
ind1 <- video$hour >= 0 & video$hour < 9
ind2 <- video$hour >= 9 & video$hour < 12
ind3 <- video$hour >= 12 & video$hour < 16
ind4 <- video$hour >= 16 & video$hour < 18
ind5 <- video$hour >= 18 & video$hour < 20
ind6 <- video$hour >= 20

video$period <- 0
video$period[ind1] <- "20:00-9:00"
video$period[ind6] <- "20:00-9:00"
video$period[ind2] <- "09:00-12:00"
video$period[ind3] <- "12:00-16:00"
video$period[ind4] <- "16:00-18:00"
video$period[ind5] <- "18:00-20:00"
```

The criterion of splitting time periods within a day is chosen by ourselves, and we want to make the number of observations in each period roughly equal, while keep the period easy to interpret, i.e., "20:00-9:00" can be seen as "at night".

1.1.5 split BGM

```
tp <- video$BGM == "    "
video$BGM[tp] <- "Original"
video$BGM[!tp] <- "Non-original"
table(video$BGM)
```

```
tp <- video$BGM == "作者创作的原声"
video$BGM[tp] <- "Original"
video$BGM[!tp] <- "Non-original"
table(video$BGM)
```

```
##
## Non-original     Original
##          859         5272
```

We divide BGM in two categories, only consider it is original or not.

1.1.6 author

```
table(table(video$author))
```

```
##
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16
## 1660  427  181   94   62   36   21   17   13   10   12    9    5    2    5    5
##   17   18   19   20   22   23   24   25   26   27   28   32   34   39   40   46
##    4    1    3    1    2    2    3    3    2    1    1    2    2    1    1    1
##   49   58   61   74   80   98
##    1    1    1    1    1    1
```

```
## save author
video$author_index <- 0
video$author_index <- video$author
```

```
### 1 --> once; 2-3 --> several; 4-11 --> often; >11 --> frequent
author_level <- function(level, data) {
  author_unique <- table(data$author)

  if (level == "once") {
    tp <- author_unique[author_unique == 1]
  } else if (level == "several") {
    tp <- author_unique[author_unique == 2 | author_unique == 3]
  } else if (level == "often") {
    tp <- author_unique[author_unique >= 4 & author_unique <= 11]
  } else if (level == "frequent") {
    tp <- author_unique[author_unique >= 12]
  }

  ind <- as.integer(names(tp))
  indd <- data$author %in% ind
  data$author[indd] <- level
  return(data)
}

video <- author_level("once", video)
video <- author_level("several", video)
video <- author_level("often", video)
video <- author_level("frequent", video)

table(video$author)
```

```
##
## frequent    often     once  several
##     1540     1534     1660     1397
```

The criterion of splitting time periods within a day is chosen by ourselves. "Once" means he has only one contribution, "several" means 2 or 3 videos, "often" for [4, 11] and "frequent" for more than 11.

Now we test whether there are authors who contribute videos in more than one type.

```
matrix <- table(video$author_index, video$type)
matrix[1:10, ]
```

```
##
##
##   1     0    0    0    3    0    0    0
##   2     0    0    0   74    0    0    0
##   3     0    0    0    1    0    0    0
##   4     0    0    0    5    0    0    0
##   5     0    0    0    1    0    0    0
##   6     0    0    0    1    0    0    0
##   7    15    0    0    0    0    0    0
##   8     0    0    0    1    0    0    0
```

```
##   9     0    0    0    1    0    0    0
##   10    0    0    0    6    0    0    0
```

```
sumoftype <- apply(matrix, 1, sum)
maxoftype <- apply(matrix, 1, max)
difference <- sumoftype - maxoftype
table(difference)
```

```
## difference
##    0
## 2595
```

There is no such author.

1.1.7 Rename and factoring

```
### rename
table(video$type)
```

```
##
##
##  978  951  359  951  928  981  983
```

```
##
## 宠物 穿搭 剧情 美食 美妆 汽车 游戏
##  978  951  359  951  928  981  983
```

```
ind1 <- video$type == " "
ind2 <- video$type == " "
ind3 <- video$type == " "
ind4 <- video$type == " "
ind5 <- video$type == " "
ind6 <- video$type == " "
ind7 <- video$type == " "
```

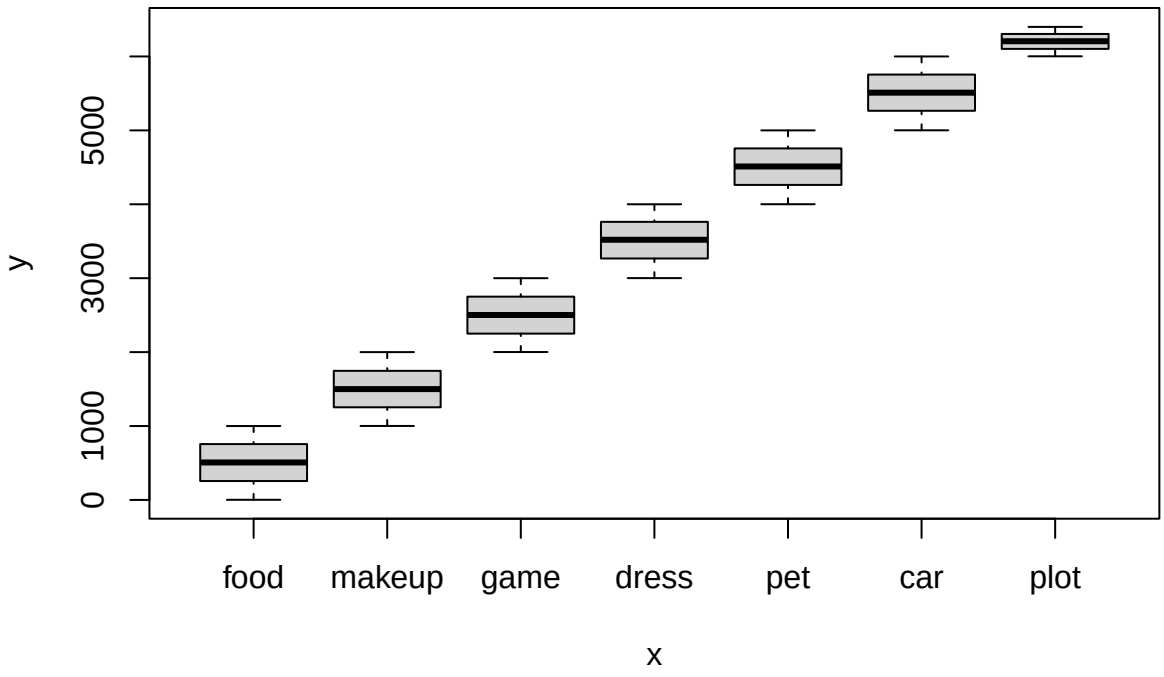
```
ind1 <- video$type == "美食"
ind2 <- video$type == "美妆"
ind3 <- video$type == "游戏"
ind4 <- video$type == "穿搭"
ind5 <- video$type == "宠物"
ind6 <- video$type == "汽车"
ind7 <- video$type == "剧情"
```

```
video$type[ind1] <- "food"
video$type[ind2] <- "makeup"
video$type[ind3] <- "game"
video$type[ind4] <- "dress"
video$type[ind5] <- "pet"
video$type[ind6] <- "car"
video$type[ind7] <- "plot"
table(video$type)
```

```
##
##    car  dress   food   game makeup    pet   plot
##    981    951    951    983    928    978    359
```

```
### factoring
video$author <-
  factor(video$author,
         levels = c("once", "several", "often", "frequent"))
video$BGM <- factor(video$BGM)
video$quarter <- factor(video$quarter)
video$month <- factor(video$month)
video$week <- factor(video$week)
video$weekday <- factor(video$weekday)
video$day <- factor(video$day)
video$hour <- factor(video$hour)
video$period <- factor(video$period)
```

```
## this order is compatible to the index
video$type <-
  factor(video$type,
         levels = c("food", "makeup", "game", "dress", "pet", "car", "plot"))
plot(video$type, video$index)
```



```
video$type <-
  factor(video$type,
         levels = c("car", "pet", "dress", "makeup", "food", "game", "plot"))
```

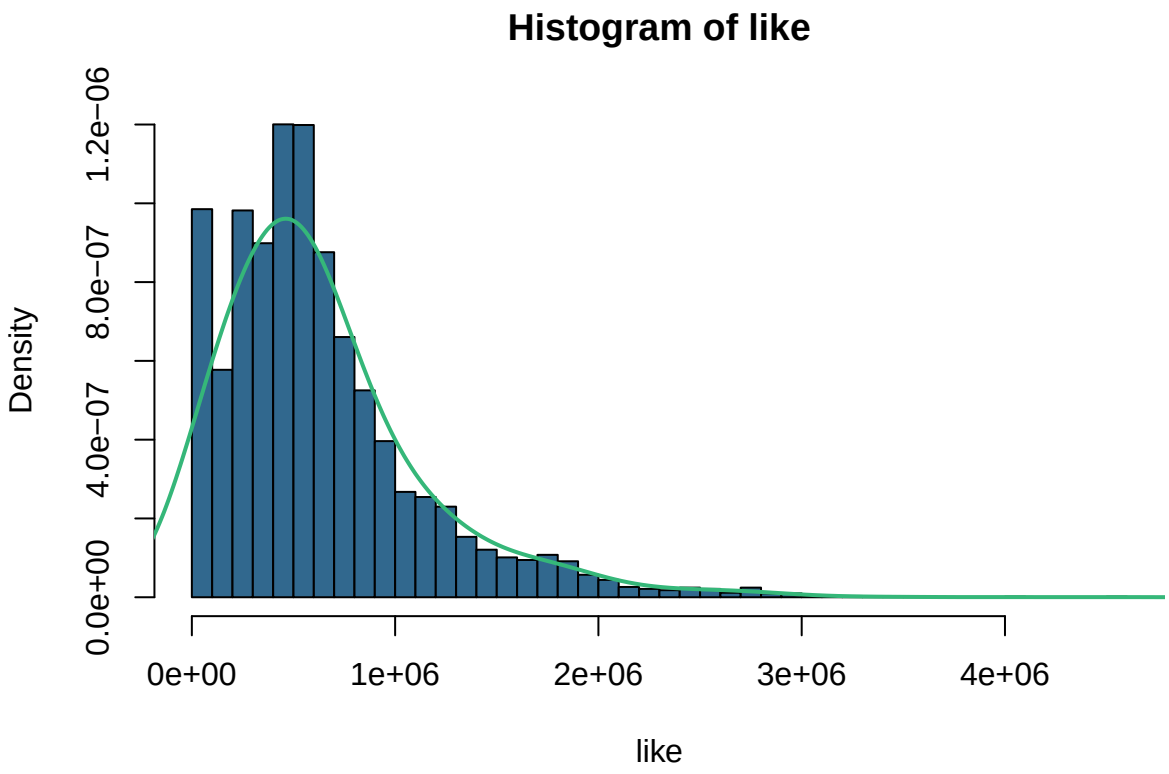
The index intervals of all types of videos don't overlap.

1.1.8 Data transformation

```
### original data
library(viridis)
```

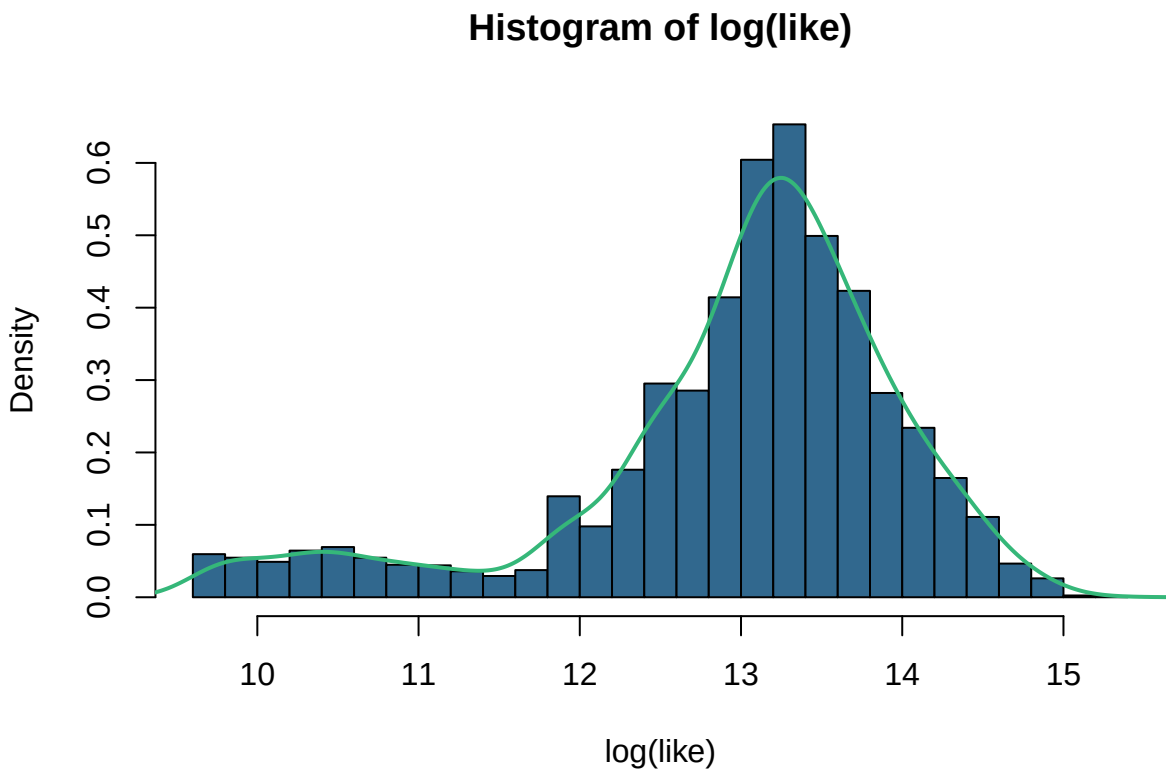
```
##      viridisLite
```

```
viridis_palatte = viridis(7)
a <- video$like

n <- length(a)
s <- sd(a)
iqr <- IQR(a)
hstariqr <- 2.6 * iqr * n ^ (-1 / 3)
nobreaks <- (max(a) - min(a)) / hstariqr
hist(
  a,
  breaks = round(nobreaks),
  probability = TRUE,
  col = viridis_palatte[3],
  xlab = "like",
  main = "Histogram of like"
)
lines(density(a, kernel = "gaussian", bw = 200000),
      col = viridis_palatte[5],
      lwd = 2)
```



Clearly, this is skew and not normal.

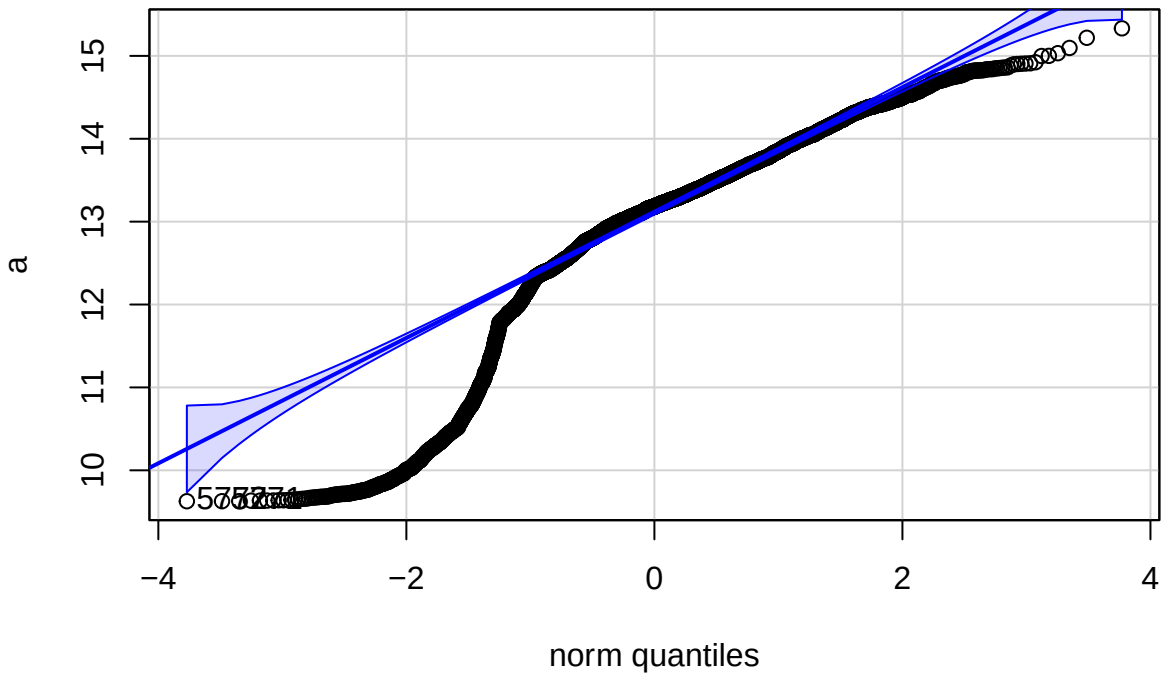
```
### use log
a <- video$like
a <- log(a)
n <- length(a)
s <- sd(a)
iqr <- IQR(a)
hstariqr <- 2.6 * iqr * n ^ (-1 / 3)
nobreaks <- (max(a) - min(a)) / hstariqr
hist(
  a,
  breaks = round(nobreaks),
  probability = TRUE,
  col = viridis_palatte[3],
  xlab = "log(like)",
  main = "Histogram of log(like)"
)
lines(density(a, kernel = "gaussian", bw = 0.2),
      col = viridis_palatte[5],
      lwd = 2)
```




```
### Q-Q plot
library(car)
```

```
##      carData
```

```
qqPlot(a)
```




```
## [1] 5772 5771
```

This is skew and not normal yet, but if we can delete data with log(like) less than 11.5, then the data may obey normal assumption. Let's first see the data with log(like_number) less than 11.

```
abnormal <- video[log(video$like) < 11.5, ]
### first 10 rows
abnormal[1:10, ]
```

```
##      index   author  like comment share          BGM duration type title
## 5400  5400 frequent 96458    3757  7907     Original    12.67  car    12
## 5401  5401     once 96004    4876  1811     Original    14.98  car     4
## 5402  5402    often 95314    3010  2485 Non-original    14.87  car    44
## 5403  5403     once 95095    2184  4876 Non-original     7.96  car    43
## 5404  5404 frequent 94508   29496   571     Original    12.43  car    25
## 5405  5405     once 94139    3475  2557     Original     6.67  car    19
## 5406  5406     once 93973     690   304     Original    15.00  car     9
## 5407  5407  several 93023    1270  2049 Non-original    21.43  car    28
## 5408  5408 frequent 92741    1378  6715     Original    49.92  car    24
## 5409  5409     once 92697    2331  3600     Original    15.08  car    17
##      quarter month week weekday day hour minute second      period author_index
## 5400       1     3   10       3   5   15     50     55 12:00-16:00         2225
## 5401       1     2    7       4  13   12     58     44 12:00-16:00         2313
## 5402       1     3   12       6  22   14     58     32 12:00-16:00         2308
## 5403       2     4   14       3   2   11     34     29 09:00-12:00         2314
```

```
## 5404       1     3   13       7  30   19     47     29 18:00-20:00         2185
## 5405       1     3   10       6   8   18     42     33 18:00-20:00         2315
## 5406       1     3   12       7  23    9     52      5 09:00-12:00         2316
## 5407       2     4   16       2  22   14     18      8 12:00-16:00         2317
## 5408       2     4   16       2  22    9     28     21 09:00-12:00         2163
## 5409       2     4   16       1  21   11     37     41 09:00-12:00         2318
```

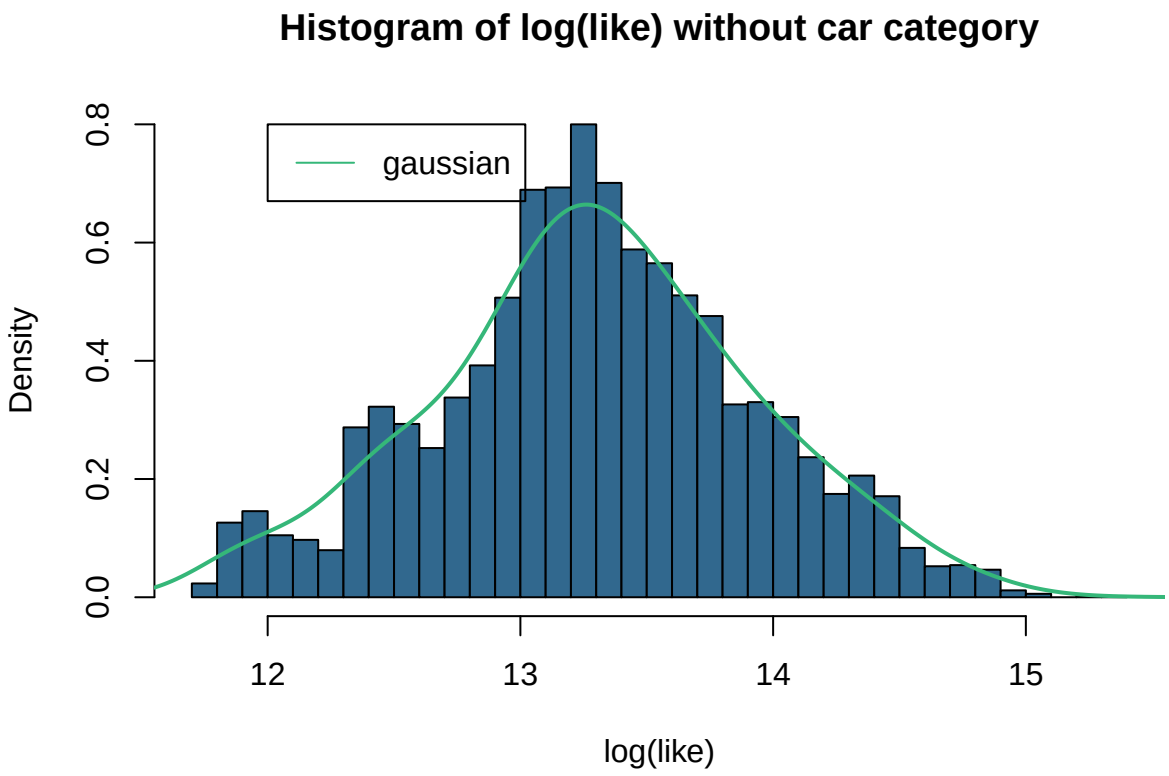
Note that the categories of these 10 videos are all cars. We guess this holds for all such data. Let's verify.

```
### only one type: car
table(abnormal$type)
```

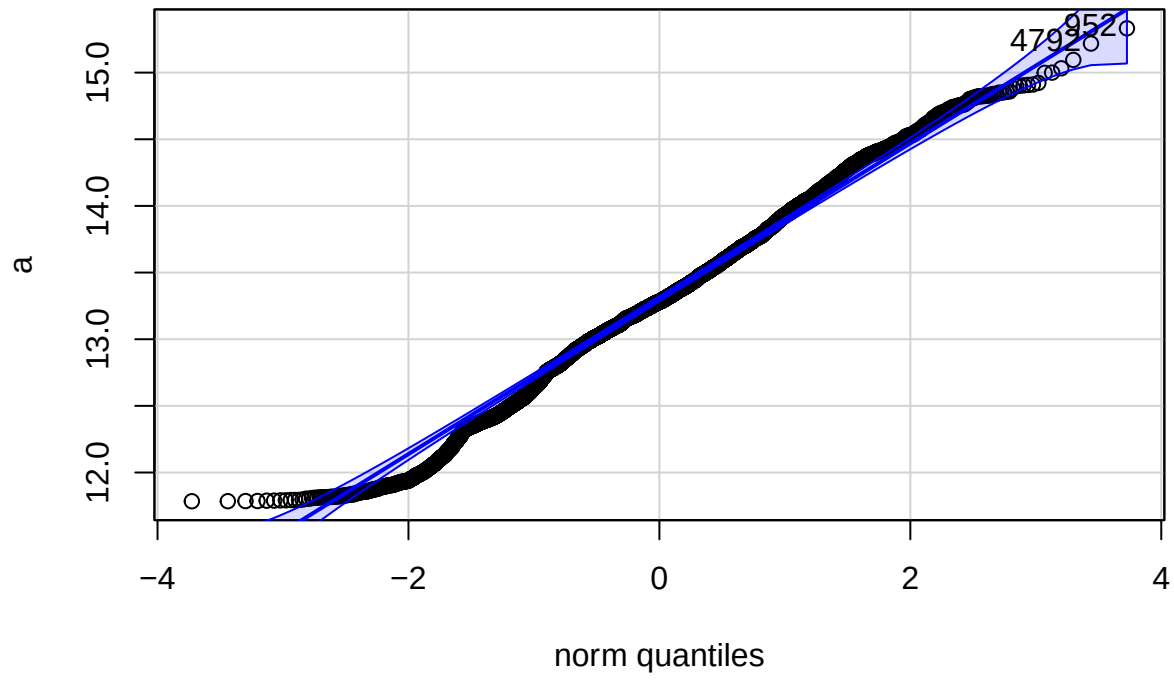
```
##
##    car    pet  dress makeup   food   game   plot
##    601      0      0      0      0      0      0
```

Therefore, we put forward a reasonable assumption: video data with car category are generated from another population, which is different from the population generating other category video data. Then, we should divide original data into two groups for further data transformation.

```
ind <- video$type == "car"
video_car <- video[ind, ]
video_car$type <- NULL
video_nocar <- video[!ind, ]

### use log & without category==car
a <- video_nocar$like
a <- log(a)
n <- length(a)
s <- sd(a)
iqr <- IQR(a)
hstariqr <- 2.6 * iqr * n ^ (-1 / 3)
nobreaks <- (max(a) - min(a)) / hstariqr
hist(
  a,
  breaks = round(nobreaks),
  probability = TRUE,
  col = viridis_palatte[3],
  xlab = "log(like)",
  main = "Histogram of log(like) without car category"
)
lines(density(a, kernel = "gaussian", bw = 0.2),
      col = viridis_palatte[5],
      lwd = 2)
legend(
  12,
  0.8,
  legend = c("gaussian"),
  col = c(viridis_palatte[5]),
  lty = 1,
  cex = 1
)
```




```
### Q-Q plot
qqPlot(a)
```

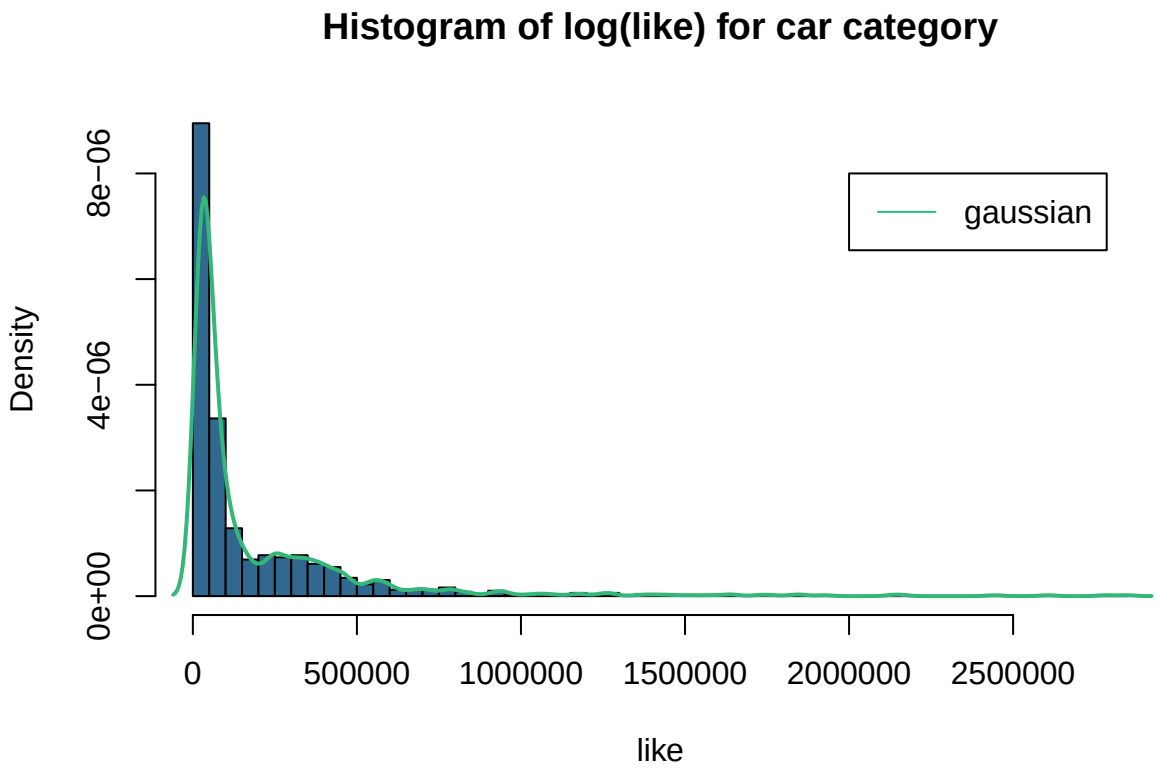



```
## [1]  952 4792
```

Clearly, data without car category is symmetric and seem to obey a normal distribution. This strengthens our assumption that data with car category are generated from another population comparing to other data.

```
### histogram for car type
a <- video_car$like
n <- length(a)
s <- sd(a)
iqr <- IQR(a)
hstariqr <- 2.6 * iqr * n ^ (-1 / 3)
nobreaks <- (max(a) - min(a)) / hstariqr
hist(
  a,
  breaks = round(nobreaks),
  probability = TRUE,
  col = viridis_palatte[3],
  xlab = "like",
  main = "Histogram of log(like) for car category"
)
lines(density(a, kernel = "gaussian", bw = 25000),
      col = viridis_palatte[5],
      lwd = 2)
legend(
  2000000,
```

```
  8e-06,
  legend = c("gaussian"),
  col = c(viridis_palatte[5]),
  lty = 1,
  cex = 1
)
```



The histogram suggests that the data with car type may obey an exponential distribution.

We use nonparametric gaussian density estimation to fit data with car category and other data respectively, and it looks well.

1.2 Data Visualization

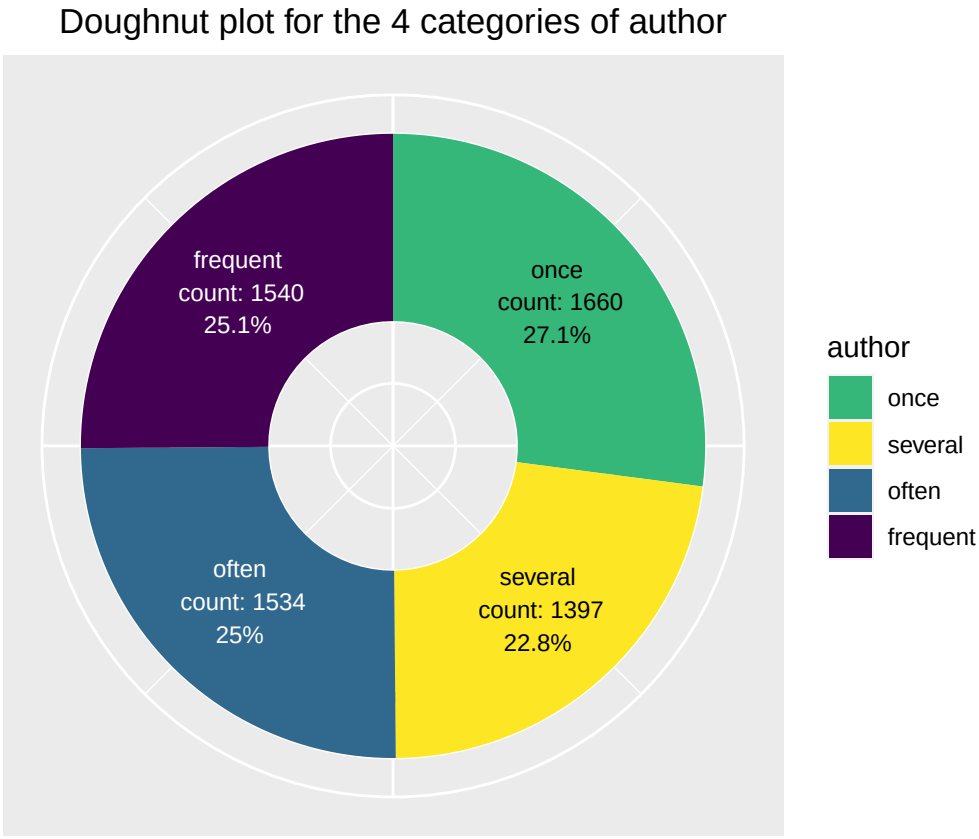
```
library(ggplot2)
library(ggthemes)
library(hrbrthemes)
library(viridis)
```

1.2.1 A doughnut plot for four categories of author

```
### pie chart for video$author
library(dplyr)
```

```
##
##     'dplyr'

## The following object is masked from 'package:car':
##
##     recode

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

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

```
tp <-
  data.frame(
    author = c("once", "several", "often", "frequent"),
    Freq = c(1660, 1397, 1534, 1540)
  )

tp$fraction <- tp$Freq / sum(tp$Freq)
tp$maxy <- cumsum(tp$fraction)
tp$miny <- c(0, head(tp$maxy, n = -1))
tp$labelPosition <- (1 * tp$maxy + 1 * tp$miny) / 2
tp$fraction <- round(tp$fraction, digits = 3) * 100
tp$label <-
  paste0(tp$author, "\n count: ", tp$Freq, "\n", tp$fraction, "%")

ggplot(tp, aes(
  ymax = maxy,
  ymin = miny,
  xmax = 4,
  xmin = 1,
  fill = author
)) +
  geom_rect() +
  geom_text(
    x = 2.5,
    aes(y = labelPosition, label = label),
    size = 3,
    colour = c("black", "black", "white", "white")
  ) +
  theme(axis.text.y = element_blank()) +
  coord_polar(theta = "y") +
  xlim(c(-1, 4)) +
  theme(axis.title = element_blank()) +
  theme(axis.text = element_blank()) +
  theme(axis.ticks = element_blank()) +
  scale_fill_viridis(discrete = TRUE,
                     breaks = c("once", "several", "often", "frequent")) +
  ggtitle("Doughnut plot for the 4 categories of author") +
  theme(plot.title = element_text(hjust = 0.5))
```

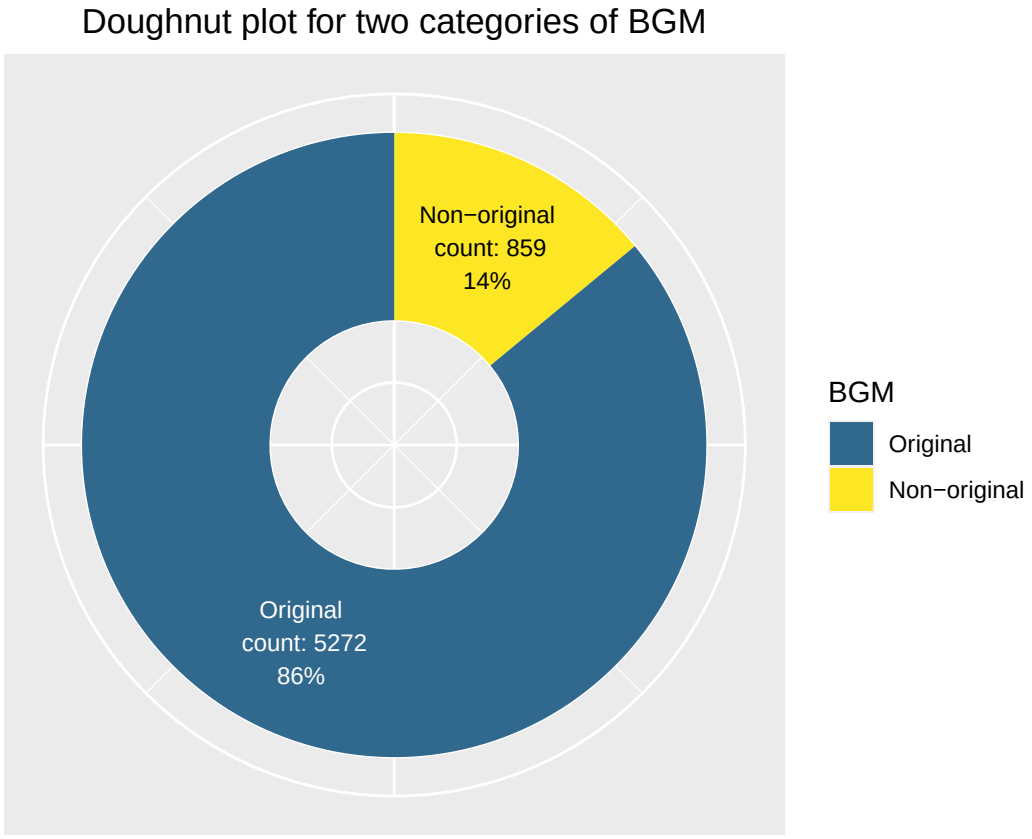


The criterion(explained before) of classifying 4 categories of author is decided by ourselves, and we purposely divide them roughly equally. The pie chart verify that.

1.2.2 A doughnut plot of two categories of BGM

```
tp <- as.data.frame(table(video$BGM))
colnames(tp) <- c("BGM", "Freq")

tp$fraction <- tp$Freq / sum(tp$Freq)
tp$maxy <- cumsum(tp$fraction)
tp$miny <- c(0, head(tp$maxy, n = -1))
tp$labelPosition <- (1 * tp$maxy + 1 * tp$miny) / 2
tp$fraction <- round(tp$fraction, digits = 3) * 100
tp$label <-
  paste0(tp$BGM, "\n count: ", tp$Freq, "\n", tp$fraction, "%")

ggplot(tp, aes(
  ymax = maxy,
  ymin = miny,
  xmax = 4,
  xmin = 1,
  fill = BGM
)) +
  geom_rect() +
  geom_text(
```

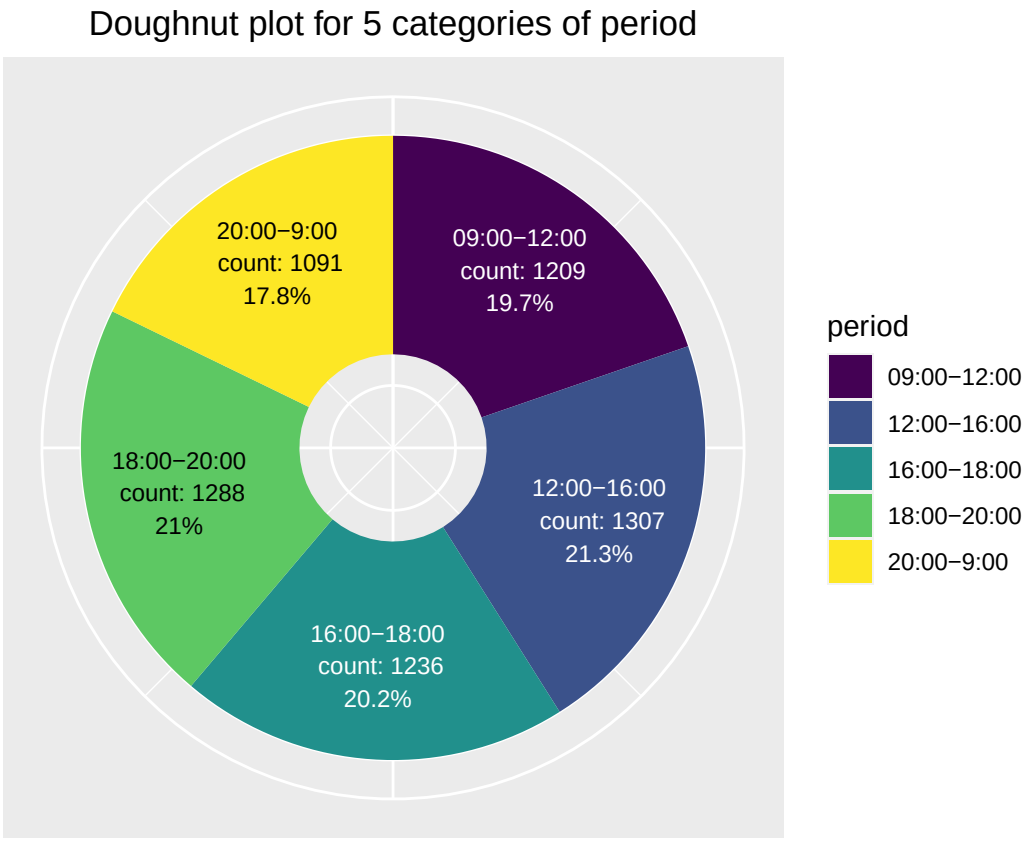
```
  x = 2.5,
  aes(y = labelPosition, label = label),
  size = 3,
  colour = c("black", "white")
) +
theme(axis.text.y = element_blank()) +
coord_polar(theta = "y") +
xlim(c(-1, 4)) +
theme(axis.title = element_blank()) +
theme(axis.text = element_blank()) +
theme(axis.ticks = element_blank()) +
scale_fill_manual(
  values = c(viridis_palatte[3], viridis_palatte[7]),
  breaks = c("Original", "Non-original")
) +
ggtitle("Doughnut plot for two categories of BGM") +
theme(plot.title = element_text(hjust = 0.5))
```



Most of the videos use original audio (not necessarily music, but maybe the author's voice), while only 14% videos use made-up snatches of music.

1.2.3 A doughnut plot of 5 categories of period

```r
tp <- as.data.frame(table(video$period))
colnames(tp) <- c("period", "Freq")

tp$fraction <- tp$Freq / sum(tp$Freq)
tp$maxy <- cumsum(tp$fraction)
tp$miny <- c(0, head(tp$maxy, n = -1))
tp$labelPosition <- (1 * tp$maxy + 1 * tp$miny) / 2
tp$fraction <- round(tp$fraction, digits = 3) * 100
tp$label <-
  paste0(tp$period, "\n count: ", tp$Freq, "\n", tp$fraction, "%")

ggplot(tp, aes(
  ymax = maxy,
  ymin = miny,
  xmax = 8,
  xmin = 1,
  fill = period
)) +
  geom_rect() +
  geom_text(
    x = 5,
    aes(y = labelPosition, label = label),
    size = 3,
    colour = c("white", "white", "white", "black", "black")
  ) +
  theme(axis.text.y = element_blank()) +
  coord_polar(theta = "y") +
  xlim(c(-2, 8)) +
  theme(axis.title = element_blank()) +
  theme(axis.text = element_blank()) +
  theme(axis.ticks = element_blank()) +
  scale_fill_viridis(begin = 0,
                     end = 1 ,
                     discrete = TRUE) +
  ggtitle("Doughnut plot for 5 categories of period") +
  theme(plot.title = element_text(hjust = 0.5))
```

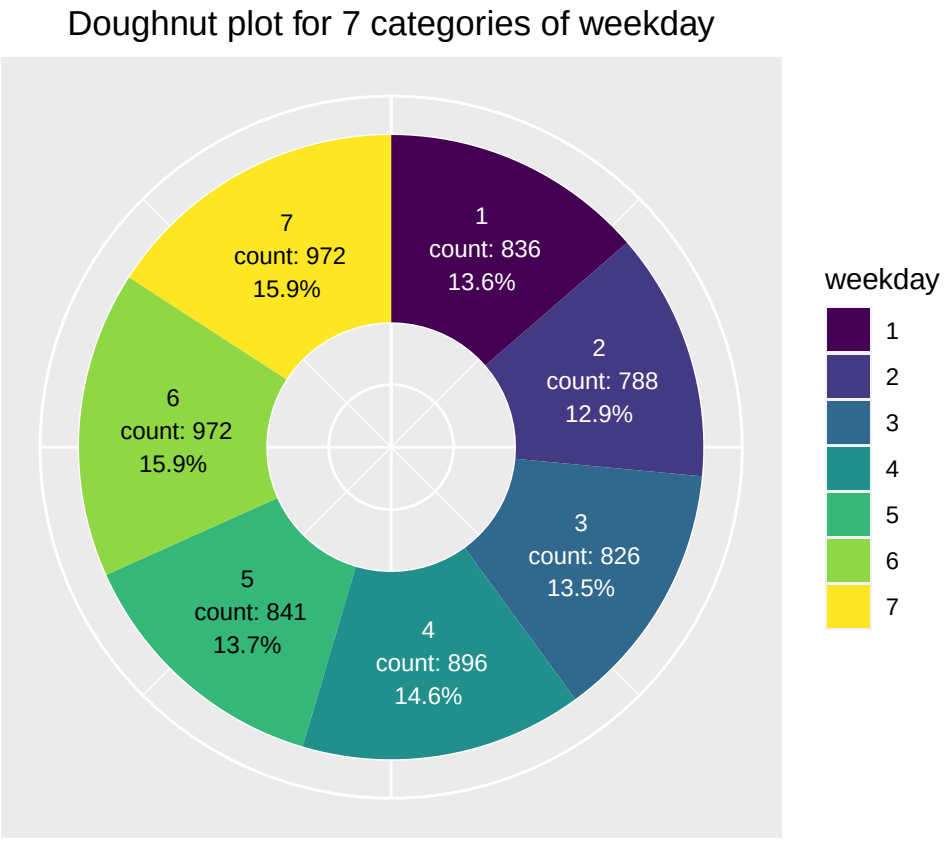


Similar to the division of author, we purposely divide period approximately equally.

1.2.4 A doughnut plot of 7 categories of weekday

```
tp <- as.data.frame(table(video$weekday))
colnames(tp) <- c("weekday", "Freq")

tp$fraction <- tp$Freq / sum(tp$Freq)
tp$maxy <- cumsum(tp$fraction)
tp$miny <- c(0, head(tp$maxy, n = -1))
tp$labelPosition <- (1 * tp$maxy + 1 * tp$miny) / 2
tp$fraction <- round(tp$fraction, digits = 3) * 100
tp$label <-
  paste0(tp$weekday, "\n count: ", tp$Freq, "\n", tp$fraction, "%")

ggplot(tp, aes(
  ymax = maxy,
  ymin = miny,
  xmax = 4,
  xmin = 1,
  fill = weekday
)) +
  geom_rect() +
  geom_text(
    x = 2.5,
```

```
  aes(y = labelPosition, label = label),
  size = 3,
  colour = c("white", "white", "white", "white", "black", "black", "black")
) +
theme(axis.text.y = element_blank()) +
coord_polar(theta = "y") +
xlim(c(-1, 4)) +
theme(axis.title = element_blank()) +
theme(axis.text = element_blank()) +
theme(axis.ticks = element_blank()) +
scale_fill_viridis(begin = 0,
                   end = 1 ,
                   discrete = TRUE) +
ggtitle("Doughnut plot for 7 categories of weekday") +
theme(plot.title = element_text(hjust = 0.5))
```

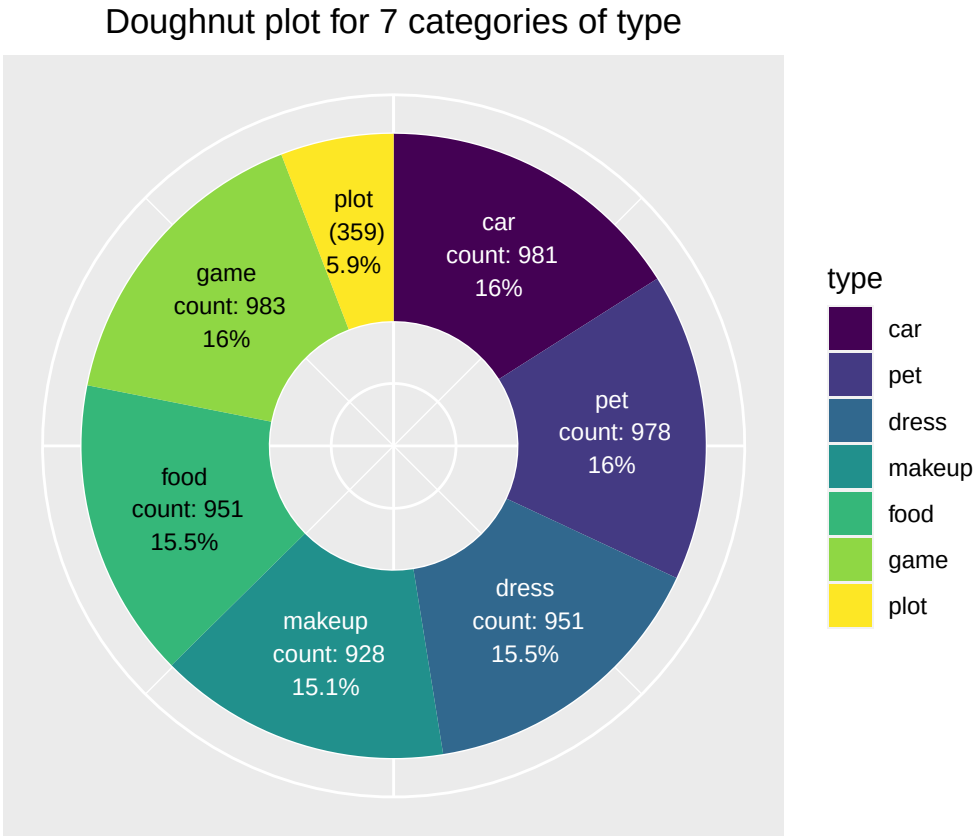


There are slightly more works on the weekend, while the counts of contributions on each day of a week don't differ much.

1.2.5 A doughnut plot of 7 categories of type

```
tp <- as.data.frame(table(video$type))
colnames(tp) <- c("type", "Freq")
```

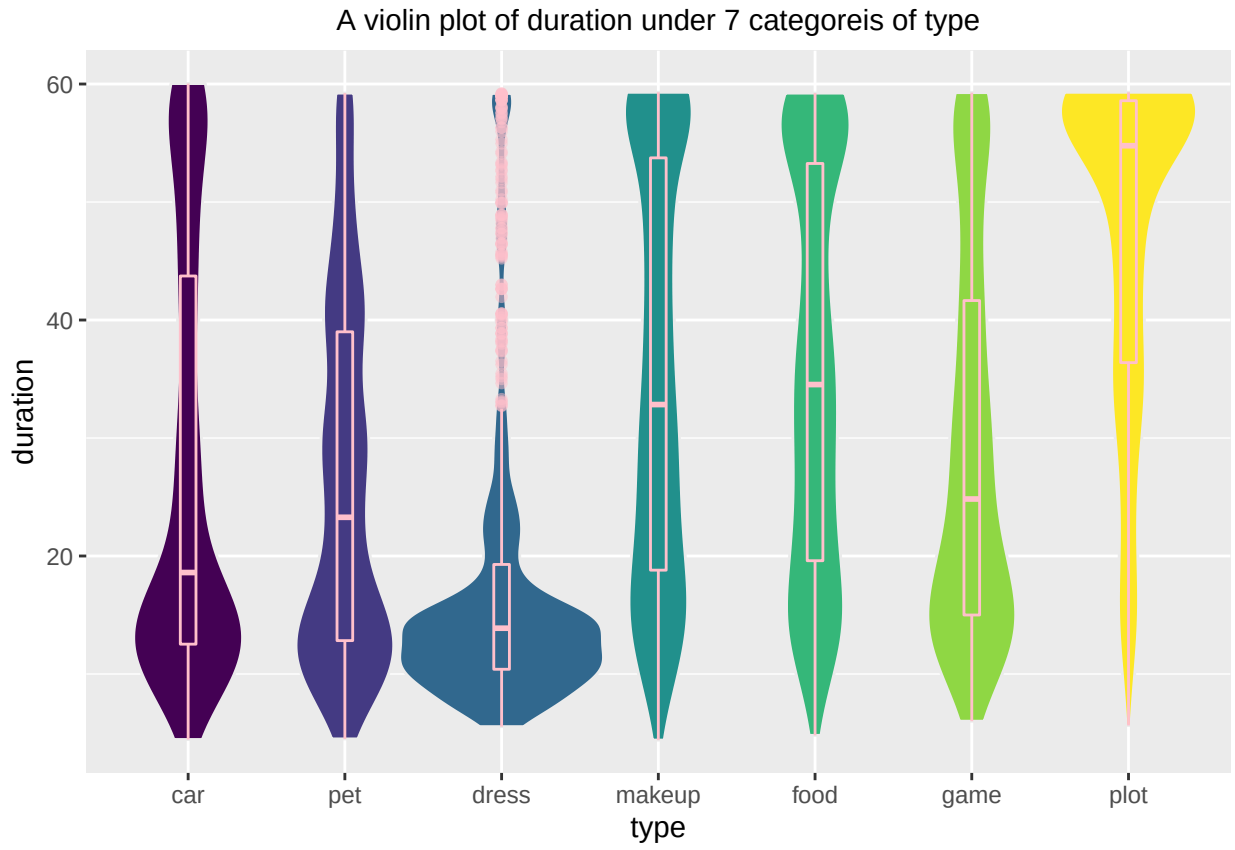
```r
tp$fraction <- tp$Freq / sum(tp$Freq)
tp$maxy <- cumsum(tp$fraction)
tp$miny <- c(0, head(tp$maxy, n = -1))
tp$labelPosition <- (1 * tp$maxy + 1 * tp$miny) / 2
tp$fraction <- round(tp$fraction, digits = 3) * 100
tp$label <-
  paste0(tp$type, "\n count: ", tp$Freq, "\n", tp$fraction, "%")
tp$label[7] = paste0(tp$type[7], "\n (", tp$Freq[7], ")\n", tp$fraction[7], "%")

ggplot(tp, aes(
  ymax = maxy,
  ymin = miny,
  xmax = 4,
  xmin = 1,
  fill = type
)) +
  geom_rect() +
  geom_text(
    x = 2.5,
    aes(y = labelPosition, label = label),
    size = 3,
    colour = c("white", "white", "white", "white", "black", "black", "black")
  ) +
  theme(axis.text.y = element_blank()) +
  coord_polar(theta = "y") +
  xlim(c(-1, 4)) +
  theme(axis.title = element_blank()) +
  theme(axis.text = element_blank()) +
  theme(axis.ticks = element_blank()) +
  scale_fill_viridis(begin = 0,
                     end = 1 ,
                     discrete = TRUE) +
  ggtitle("Doughnut plot for 7 categories of type") +
  theme(plot.title = element_text(hjust = 0.5))
```



Before we delete the duplicated data, there are exactly 400 records in type plot and 1000 records in each of the other 6 types. After deletion, such number and proportion for each type generally keep.

1.2.6 A violin plot of duration under 7 categories of type

```
video %>%
  ggplot(aes(x = type, y = duration, fill = type)) +
  geom_violin(width = 1.3, color = "#EBEBEB") +
  geom_boxplot(width = 0.1,
               color = "pink",
               alpha = 0.5) +
  scale_fill_viridis(discrete = TRUE) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11)) +
  ggtitle("A violin plot of duration under 7 categoreis of type") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
xlab("type")
```

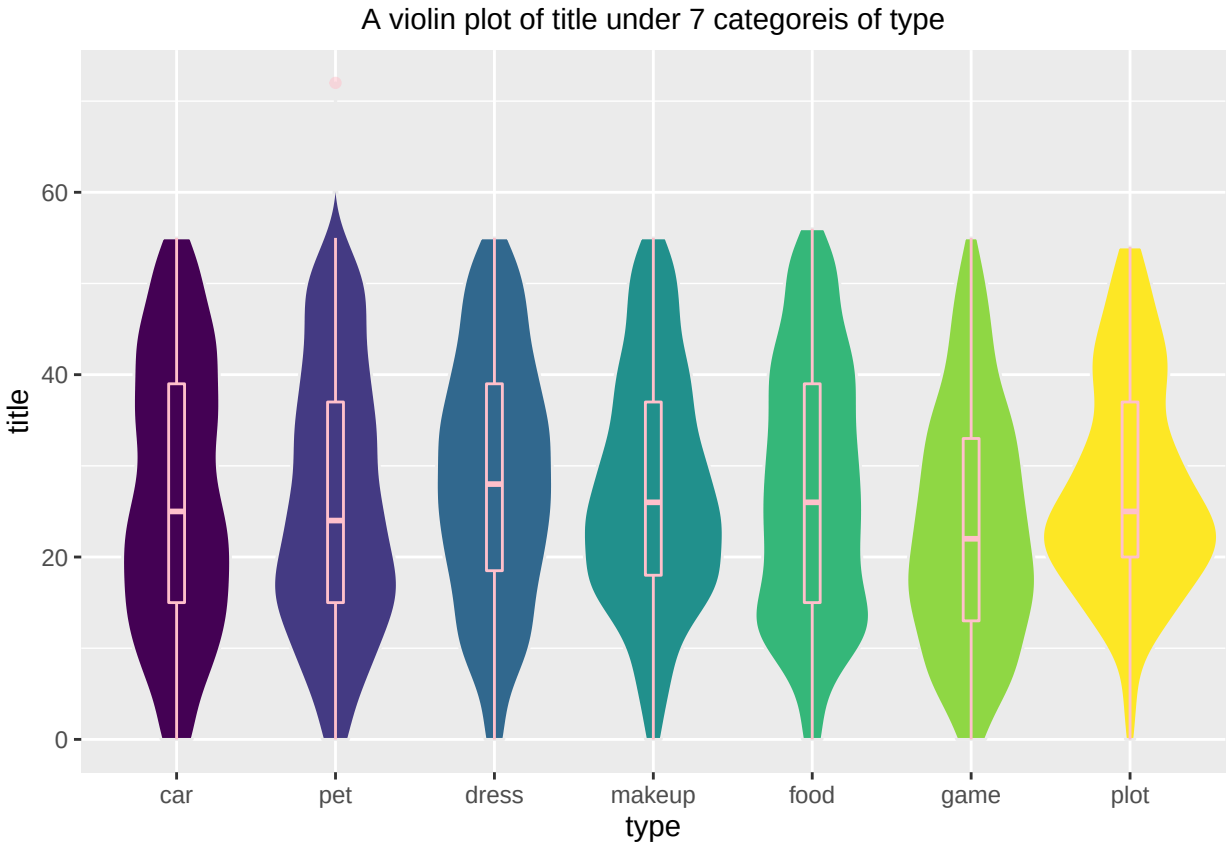
```
## $x
## [1] "type"
##
## attr(,"class")
## [1] "labels"
```

Type dress and plot have special distributions of duration. Most of the videos in type dress are short, while most in Plot are longer, near the upper restriction of video length. This is perhaps because the story in plot video need more time to perform, while a dress video only shares one tiny skill of making up. We can also notice that in type car, makeup, food and game, the density at the middle duration is low, while it's higher at short duration and duration near 60. This implies that some of the videos are designed as a very short works, but many of others are intended to carry much content(maybe can produce a video of several minutes) while can't exceed the time restriction of 60 seconds.

1.2.7 A violin plot of title under 7 categories of type

```
video %>%
  ggplot(aes(x = type, y = title, fill = type)) +
  geom_violin(width = 1.1, color = "#EBEBEB") +
  geom_boxplot(width = 0.1,
               color = "pink",
```

```
              alpha = 0.5) +
  scale_fill_viridis(discrete = TRUE) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11)) +
  ggtitle("A violin plot of title under 7 categoreis of type") +
  theme(plot.title = element_text(hjust = 0.5))
```



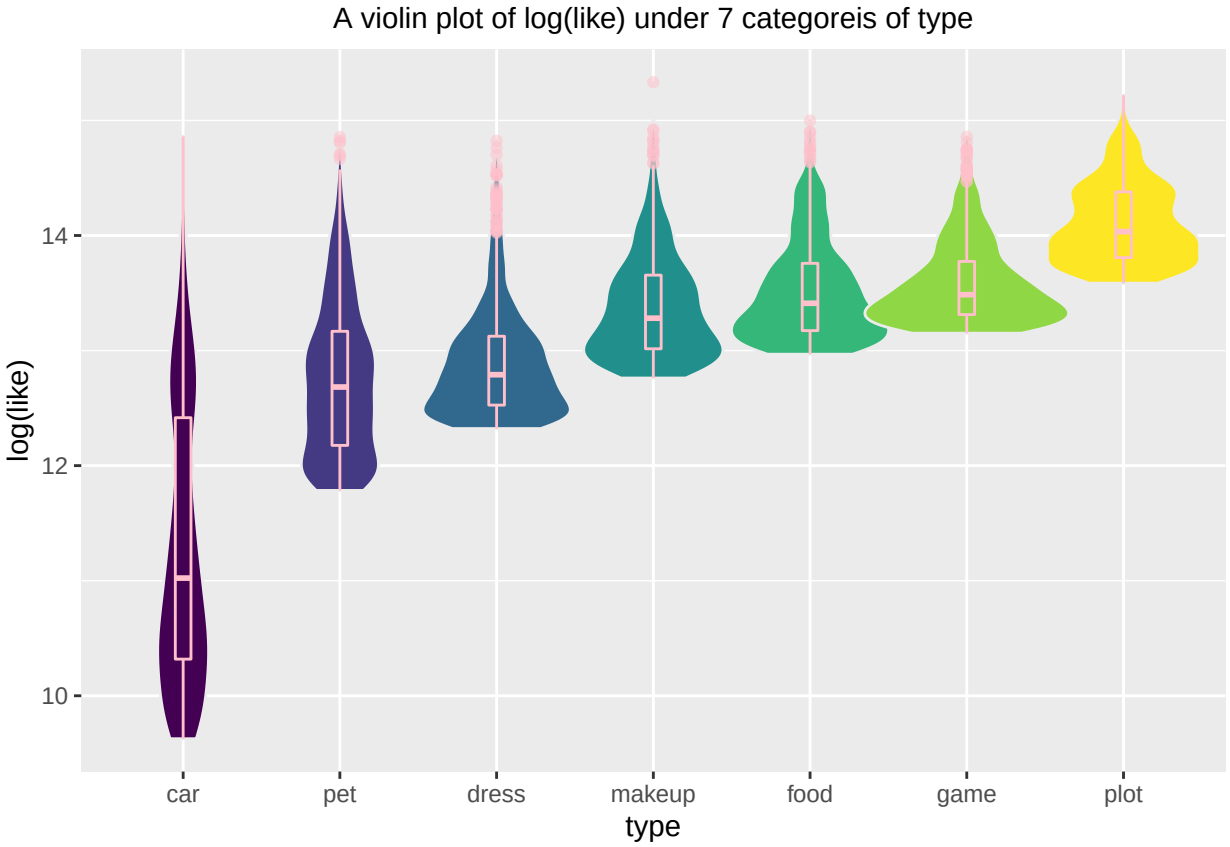

```
xlab("type")
```

```
## $x
## [1] "type"
##
## attr(,"class")
## [1] "labels"
```

There's no much difference between the distributions of title under different types, and the distribution curves are relatively flat. Nevertheless, type plot has a bit difference, since the distribution there has a peak at around 20(s). This may tell us that plot videos are likely to introduce its hook in the story by one or two sentences, resulting in a title around 20 characters.

1.2.8 A violin plot of log(like) under 7 categories of type

```
video %>%
  ggplot(aes(x = type, y = log(like), fill = type)) +
  geom_violin(width = 1.3, color = "#EBEBEB") +
  geom_boxplot(width = 0.1,
               color = "pink",
               alpha = 0.5) +
  scale_fill_viridis(discrete = TRUE) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11)) +
  ggtitle("A violin plot of log(like) under 7 categoreis of type") +
  theme(plot.title = element_text(hjust = 0.5))
```



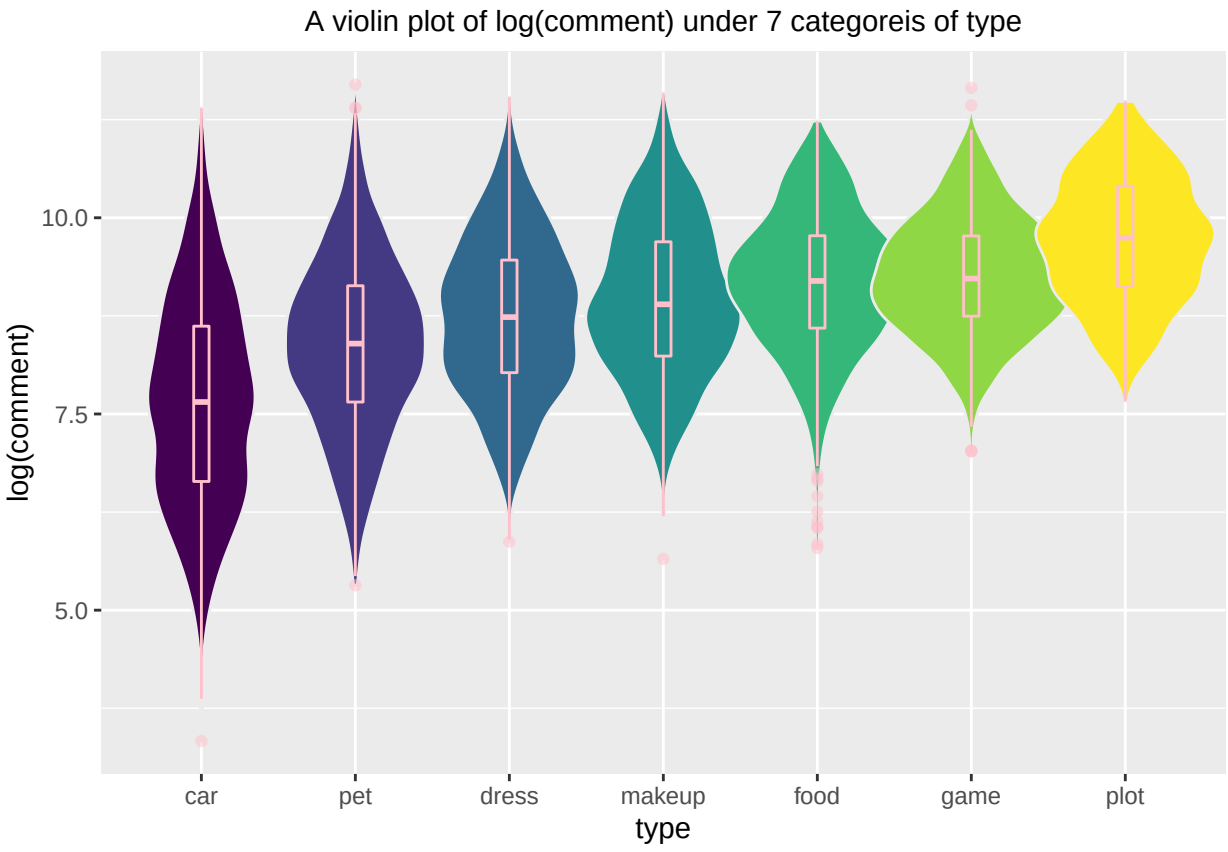

```
xlab("type")
```

```
## $x
## [1] "type"
##
## attr(,"class")
## [1] "labels"
```

Type has a significant influence on the value of like. From low to high, the order of types is car, pet, dress, makeup, food, game and plot. The mean like in type plot is around 20 times that of car. In each type, the log(like) shows obvious right skewness.

1.2.9 A violin plot of log(comment) under 7 categories of type

```
video %>%
  ggplot(aes(
    x = type,
    y = log(comment),
    fill = type
  )) +
  geom_violin(width = 1.3, color = "#EBEBEB") +
  geom_boxplot(width = 0.1,
               color = "pink",
               alpha = 0.5) +
  scale_fill_viridis(discrete = TRUE) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11)) +
  ggtitle("A violin plot of log(comment) under 7 categoreis of type") +
  theme(plot.title = element_text(hjust = 0.5))
```



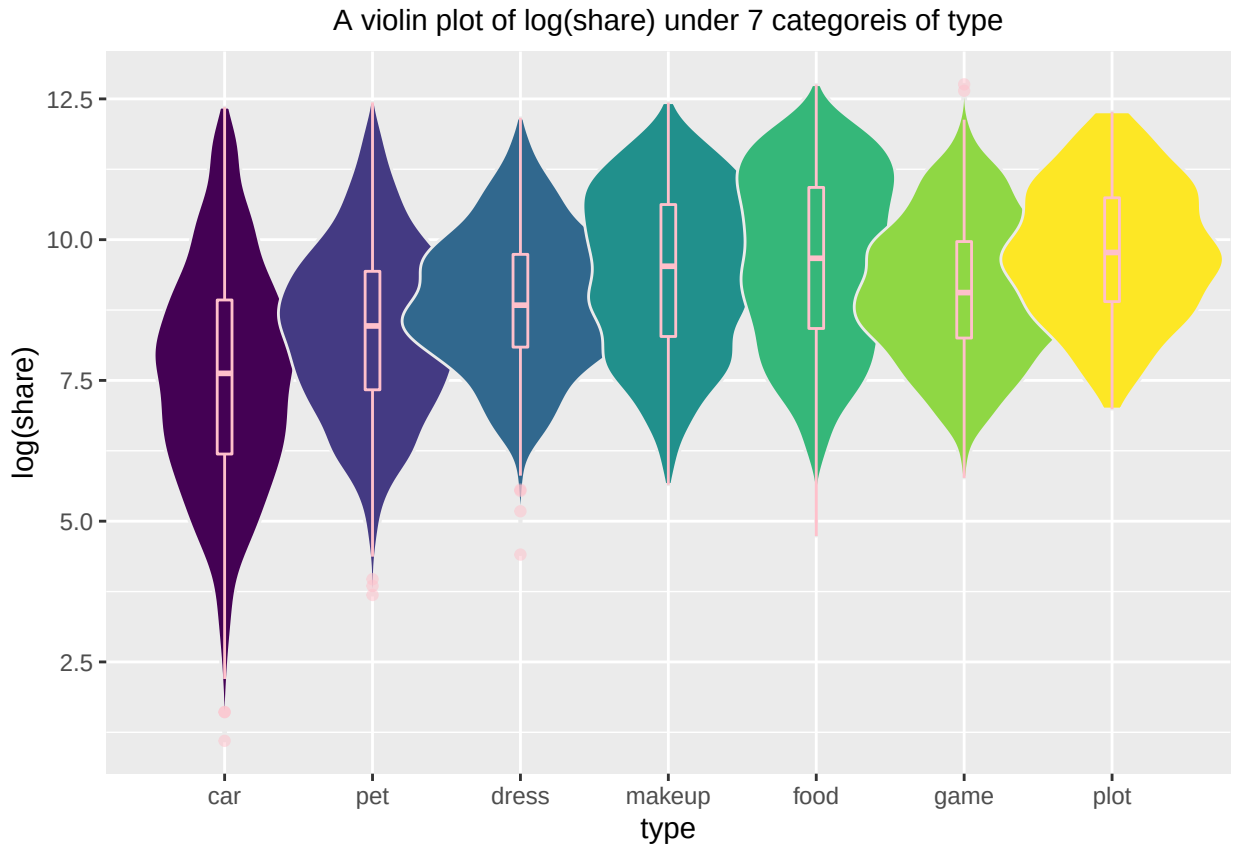

```
xlab("type")
```

```
## $x
## [1] "type"
##
## attr(,"class")
## [1] "labels"
```

Type also has a significant influence on the value of comment.The order of types is the same as that for like. The mean like in type plot is around 7 times that of car. In each type, the distribution of log(like) shows symmetry.

1.2.10 A violin plot of log(share) under 7 categories of type

```
video %>%
  ggplot(aes(x = type, y = log(share), fill = type)) +
  geom_violin(width = 1.6, color = "#EBEBEB") +
  geom_boxplot(width = 0.1,
               color = "pink",
               alpha = 0.5) +
  scale_fill_viridis(discrete = TRUE) +
  theme(legend.position = "none",
        plot.title = element_text(size = 11)) +
  ggtitle("A violin plot of log(share) under 7 categoreis of type") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
xlab("type")
```

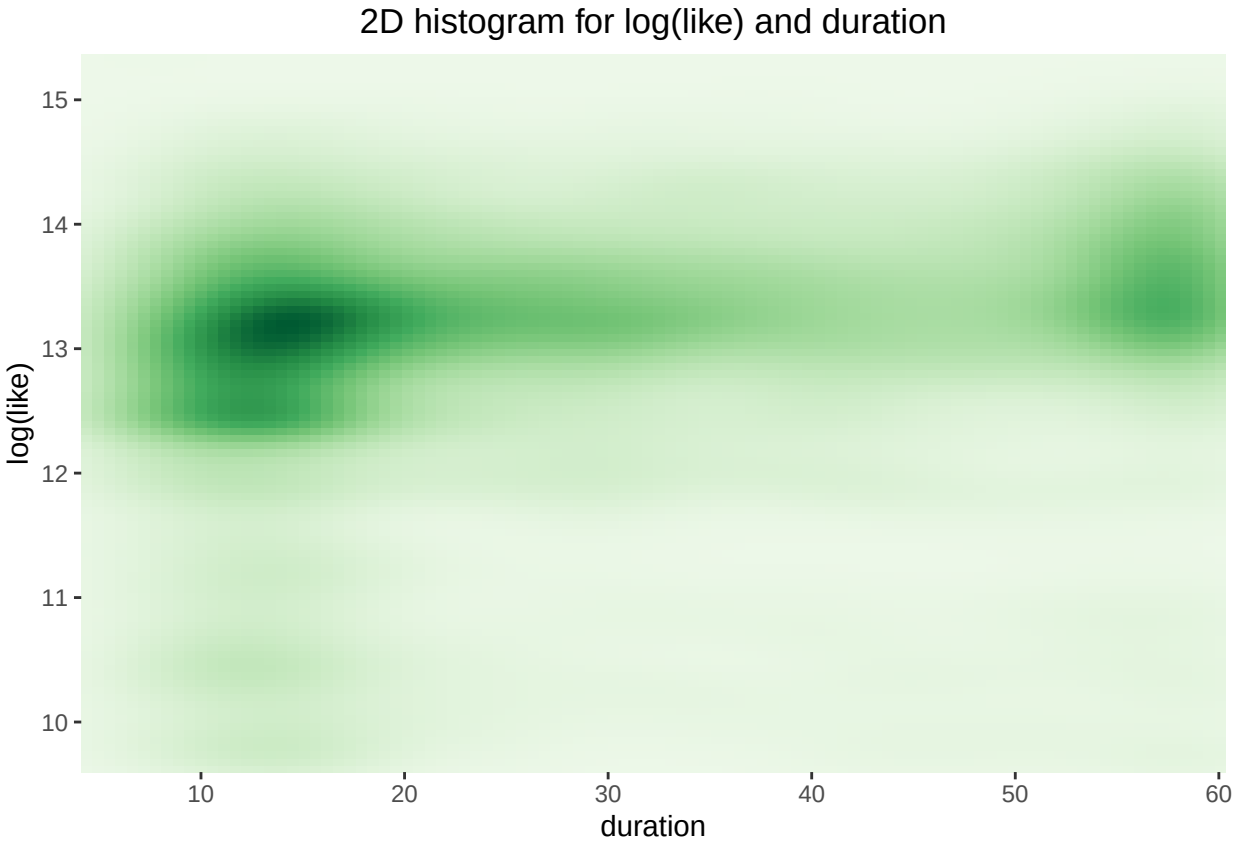
```
## $x
## [1] "type"
##
```

```
## attr(,"class")
## [1] "labels"
```

The feature of share under different types is similar to that of comment, only that the difference between types is not as distinct. Now we compare the three violin plots for like, comment and share. An obvious difference is that comment and share have much fewer "very high" values than like. Since like, comment and share are naturally closely connected, we suspect that some of the very high like values are attained by cheating using technical methods.

1.2.11 2D histogram for log(like) and duration

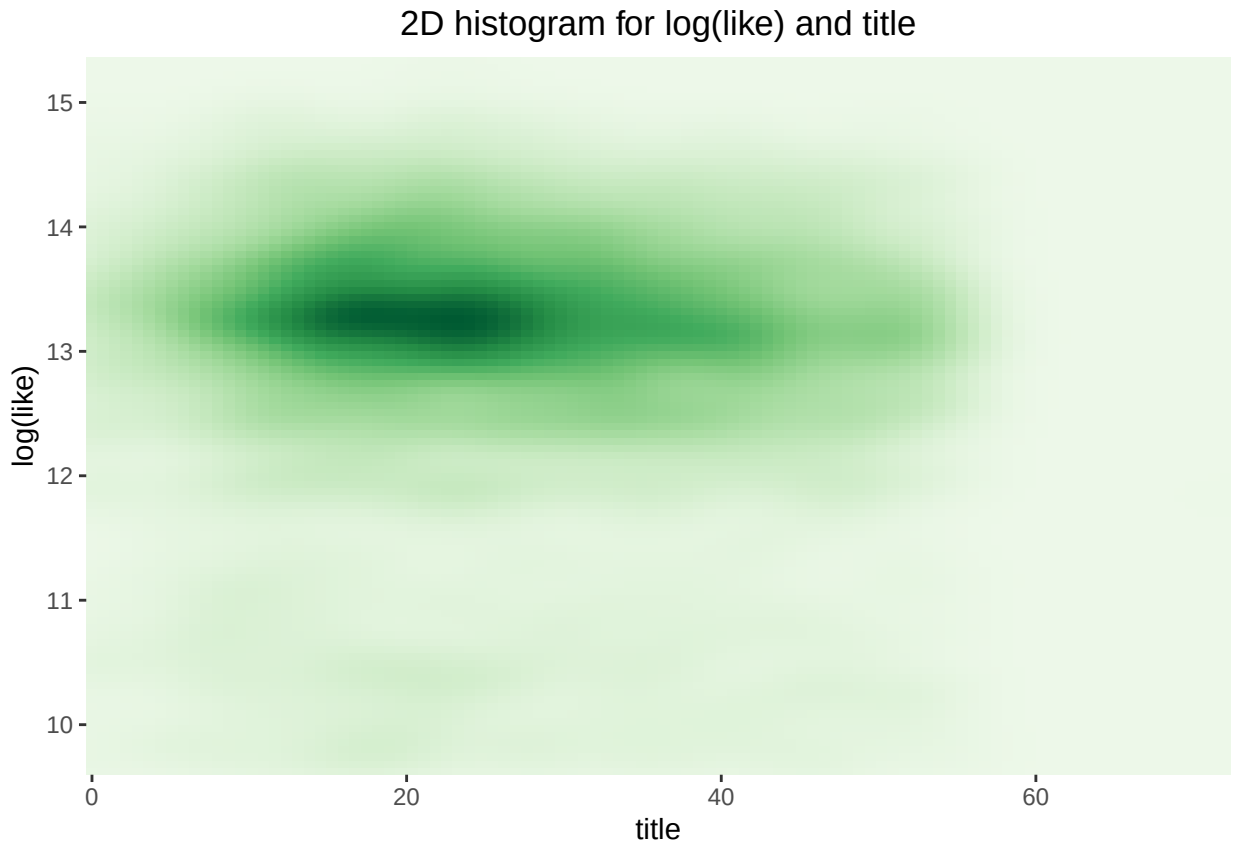
```
ggplot(video, aes(x = duration, y = log(like))) +
  stat_density_2d(aes(fill = ..density..), geom = "raster", contour = FALSE) +
  scale_fill_distiller(palette = "viridis", direction = 1) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  theme(legend.position = 'none') +
  ggtitle("2D histogram for log(like) and duration") +
  theme(plot.title = element_text(hjust = 0.5))
```



In such 2D histograms, the darker the color of a region, the greater the joint density there. We notice that the dark stripe is parallel to the axis, which indicates that given any value of duration, the conditional distribution of log(like) is generally the same. So, duration is not an important variable interpreting the variation of like.

1.2.12 2D histogram for log(like) and title

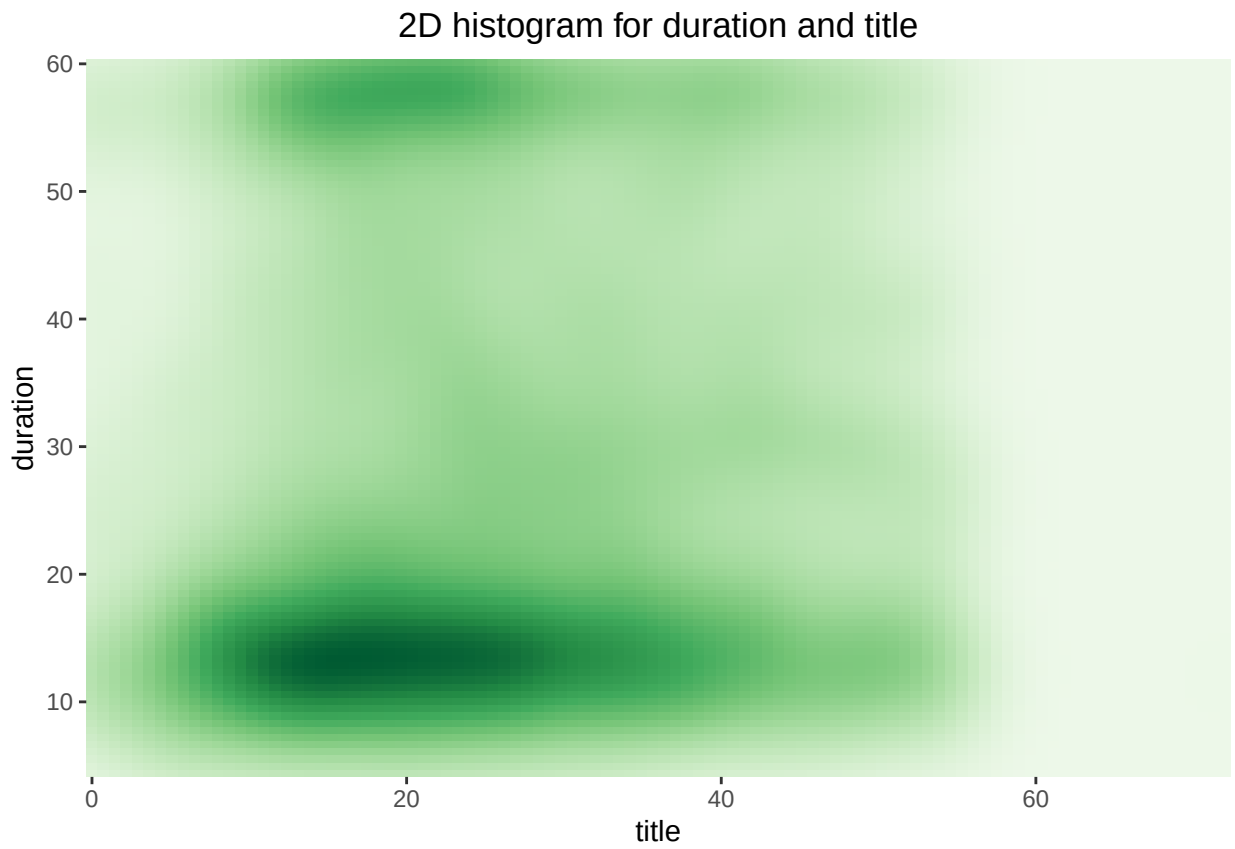
```
ggplot(video, aes(x = title, y = log(like))) +
  stat_density_2d(aes(fill = ..density..), geom = "raster", contour = FALSE) +
  scale_fill_distiller(palette = "viridis", direction = 1) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  theme(legend.position = 'none') +
  ggtitle("2D histogram for log(like) and title") +
  theme(plot.title = element_text(hjust = 0.5))
```



Similar to last plot, the dark stripe is parallel to the axis. So, title is not an important variable interpreting the variation of like either.

1.2.13 2D histogram for duration and title

```
ggplot(video, aes(x = title, y = duration)) +
  stat_density_2d(aes(fill = ..density..), geom = "raster", contour = FALSE) +
  scale_fill_distiller(palette = "viridis", direction = 1) +
  scale_x_continuous(expand = c(0, 0)) +
  scale_y_continuous(expand = c(0, 0)) +
  theme(legend.position = 'none') +
  ggtitle("2D histogram for duration and title") +
  theme(plot.title = element_text(hjust = 0.5))
```



This shows that the correlation between title and duration is not strong.

1.2.14 Heatmap for type and author

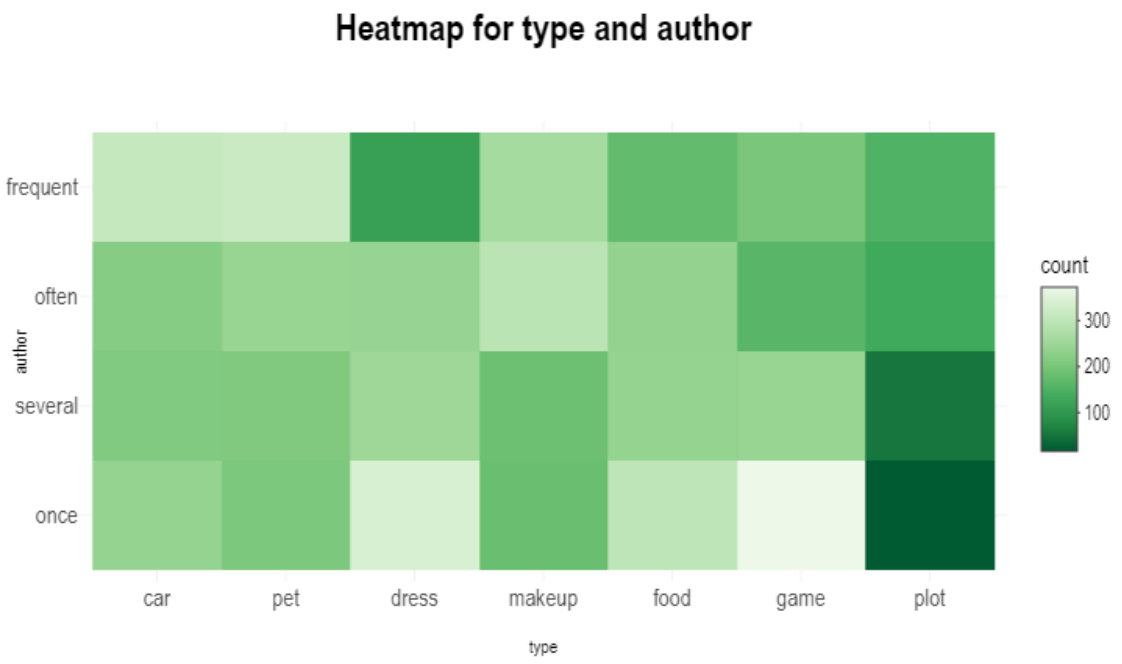
```
library(hrbrthemes)
library(plotly)
```

```
##
##     'plotly'

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

## The following object is masked from 'package:stats':
##
##     filter

## The following object is masked from 'package:graphics':
##
##     layout
```

```
tpFrame <- video %>%
  group_by(type, author) %>%
  summarize(count = n())
```

```
## `summarise()` has grouped output by 'type'. You can override using the `.groups` argument.
```

```
tpFrame <- tpFrame %>%
  mutate(text = paste0("type: ", type, "\n", "author: ", author, "\n", "count: ", count))

p <- ggplot(tpFrame, aes(type, author, fill = count, text = text)) +
  geom_tile() +
  scale_fill_distiller(palette = "viridis") +
  theme_ipsum() +
  ggtitle("Heatmap for type and author") +
  theme(plot.title = element_text(hjust = 0.5))

ggplotly(p, tooltip = "text")
```



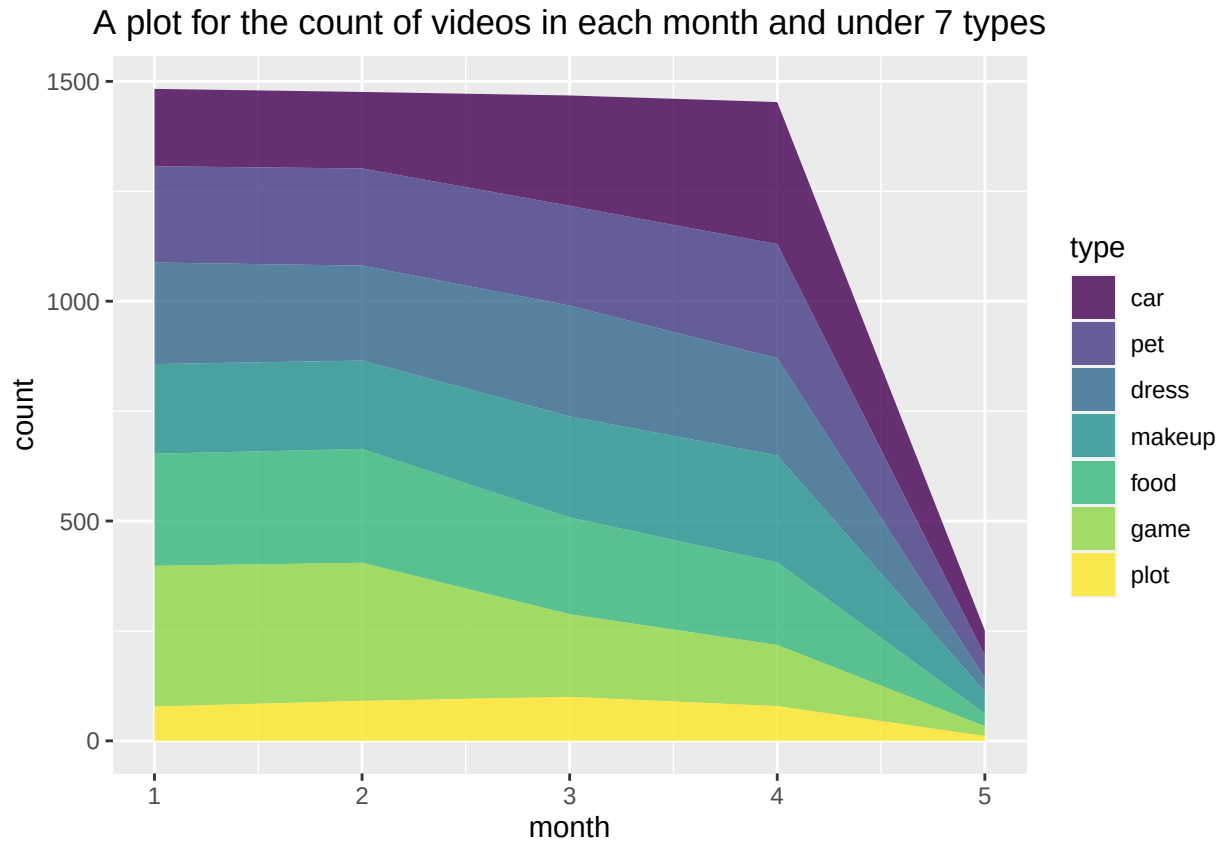
There are some differences in every type's author's classification. For example, in type game there are many contributors who makes only one video, but in type plot, more authors make larger numbers of videos. Such regular may indicate that plot videos are strong in speciality, while many authors in type game and dress just make videos for fun. This can also explain why plot videos get more likes generally–just like "practice makes perfect".

1.2.15 Time series plot for the count of videos in each month

```
tpFrame <- video %>%
  group_by(month, type) %>%
  summarize(count = n())
```

```
## `summarise()` has grouped output by 'month'. You can override using the `.groups` argument.
```

```
tpFrame <-
  data.frame(month = tpFrame$month,
             count = tpFrame$count,
             type = tpFrame$type)

tpFrame$month <- as.numeric(tpFrame$month)

ggplot(tpFrame, aes(x = month, y = count, fill = type)) +
  geom_area(alpha = 0.8 , size = .5) +
  scale_fill_viridis(discrete = T) +
  ggtitle("A plot for the count of videos in each month and under 7 types") +
  theme(plot.title = element_text(hjust = 0.5))
```



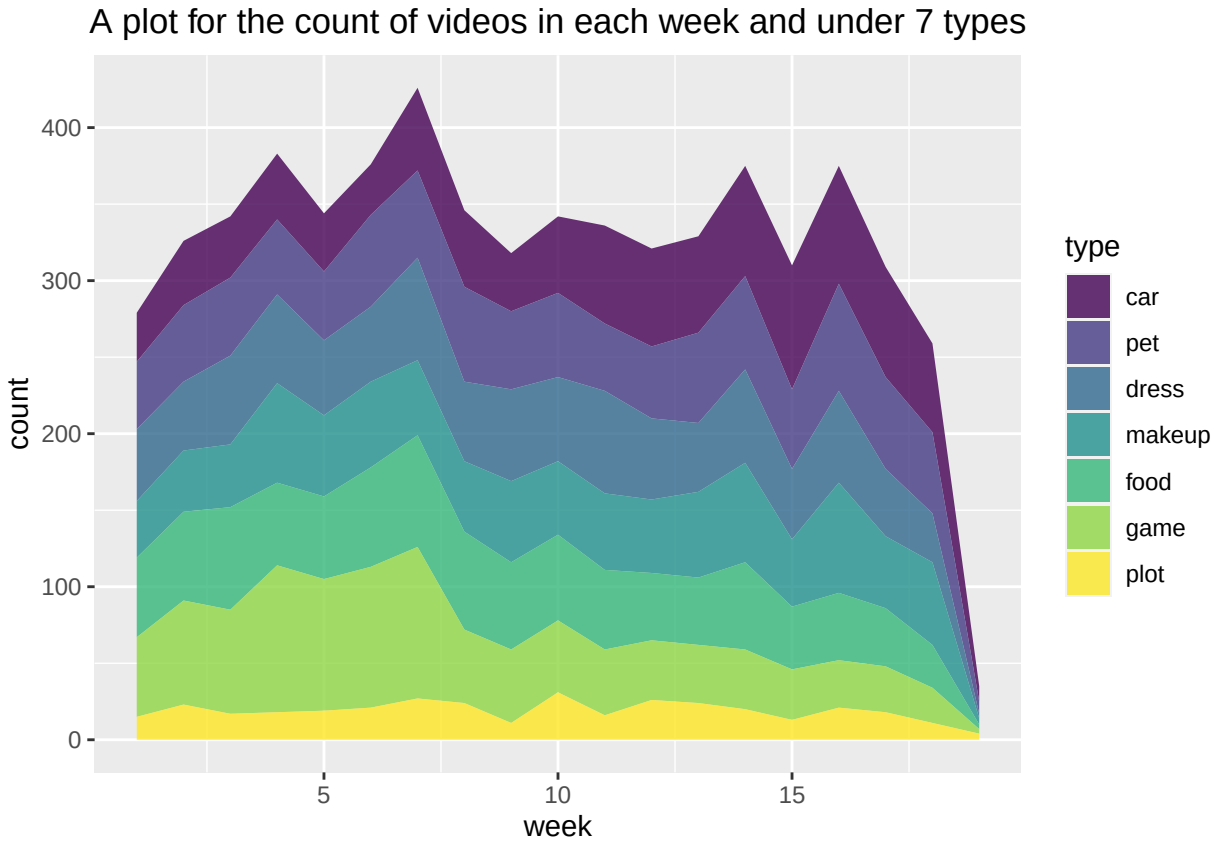
The total count of videos in the former 4 months is roughly stable, while the proportion of type car gets bigger. The records end at May 9th so the count in May is naturally small.

1.2.16 Time series plot for the count of videos in each week

```
tpFrame <- video %>%
  group_by(week, type) %>%
  summarize(count = n())
```

```
## `summarise()` has grouped output by 'week'. You can override using the `.groups` argument.
```

```
tpFrame <-
  data.frame(week = tpFrame$week,
             count = tpFrame$count,
             type = tpFrame$type)

tpFrame$week <- as.numeric(tpFrame$week)

ggplot(tpFrame, aes(x = week, y = count, fill = type)) +
  geom_area(alpha = 0.8 , size = .5) +
  scale_fill_viridis(discrete = T) +
  ggtitle("A plot for the count of videos in each week and under 7 types") +
  theme(plot.title = element_text(hjust = 0.5))
```



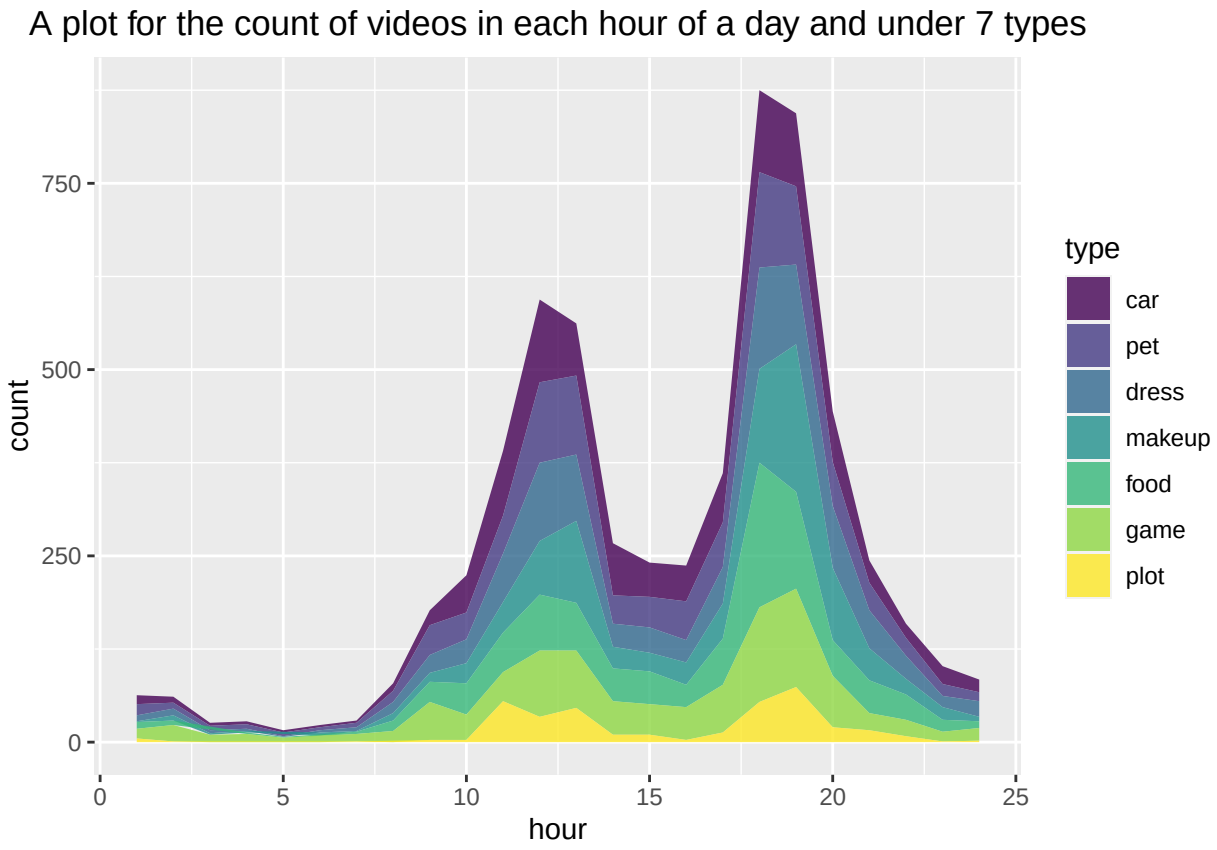
The weekly count is not so stable, but most of the significant variation in the first several weeks comes from the fluctuation of type game. In week 15, makeup, dress, pet and car type suffer from a decrease in production at the same time. Generally speaking, this series plot shows some interesting findings but helps little with finding the law behind the variation.

1.2.17 Time series plot for the count of videos in each hour of a day

```
tpFrame <- video %>%
  group_by(hour, type) %>%
  summarize(count = n())
```

```
## `summarise()` has grouped output by 'hour'. You can override using the `.groups` argument.
```

```
tpFrame <-
  data.frame(hour = tpFrame$hour,
             count = tpFrame$count,
             type = tpFrame$type)

tpFrame$hour <- as.numeric(tpFrame$hour)

ggplot(tpFrame, aes(x = hour, y = count, fill = type)) +
  geom_area(alpha = 0.8 , size = .5) +
  scale_fill_viridis(discrete = T) +
  ggtitle("A plot for the count of videos in each hour of a day and under 7 types") +
  theme(plot.title = element_text(hjust = 0.5))
```



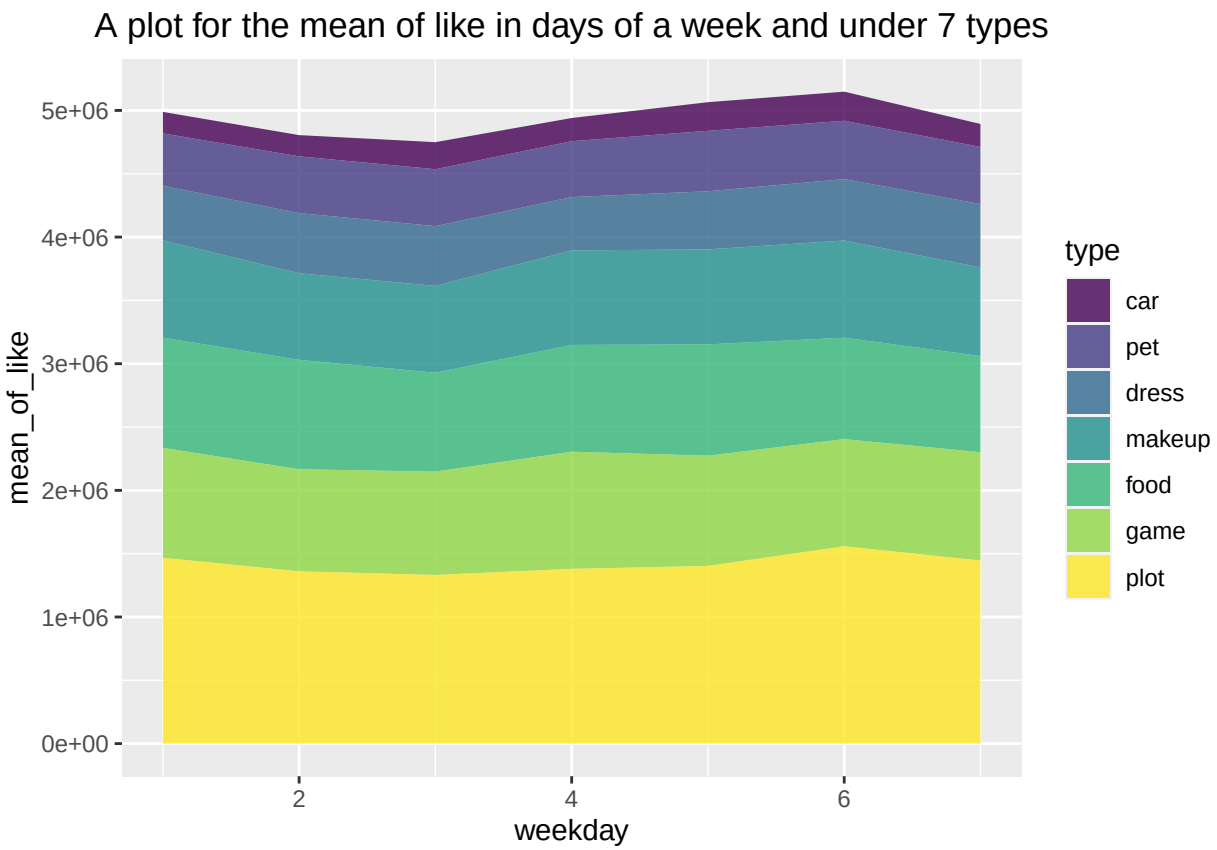
No matter what the type is, there are much more videos released in around 12 o'clock and 18 o'clock, which are lunch and supper times respectively. During the sleeping times, the number of videos are very small correspondingly.

1.2.18 A plot for the mean of like in days of a week and under 7 types

```
tpFrame <- video %>%
  group_by(weekday, type) %>%
  summarize(mean_of_like = mean(like))
```

```
## `summarise()` has grouped output by 'weekday'. You can override using the `.groups` argument.
```

```
tpFrame$weekday <- as.numeric(tpFrame$weekday)

ggplot(tpFrame, aes(x = weekday, y = mean_of_like, fill = type)) +
  geom_area(alpha = 0.8 , size = .5) +
  scale_fill_viridis(discrete = T) +
  ggtitle("A plot for the mean of like in days of a week and under 7 types") +
  theme(plot.title = element_text(hjust = 0.5))
```



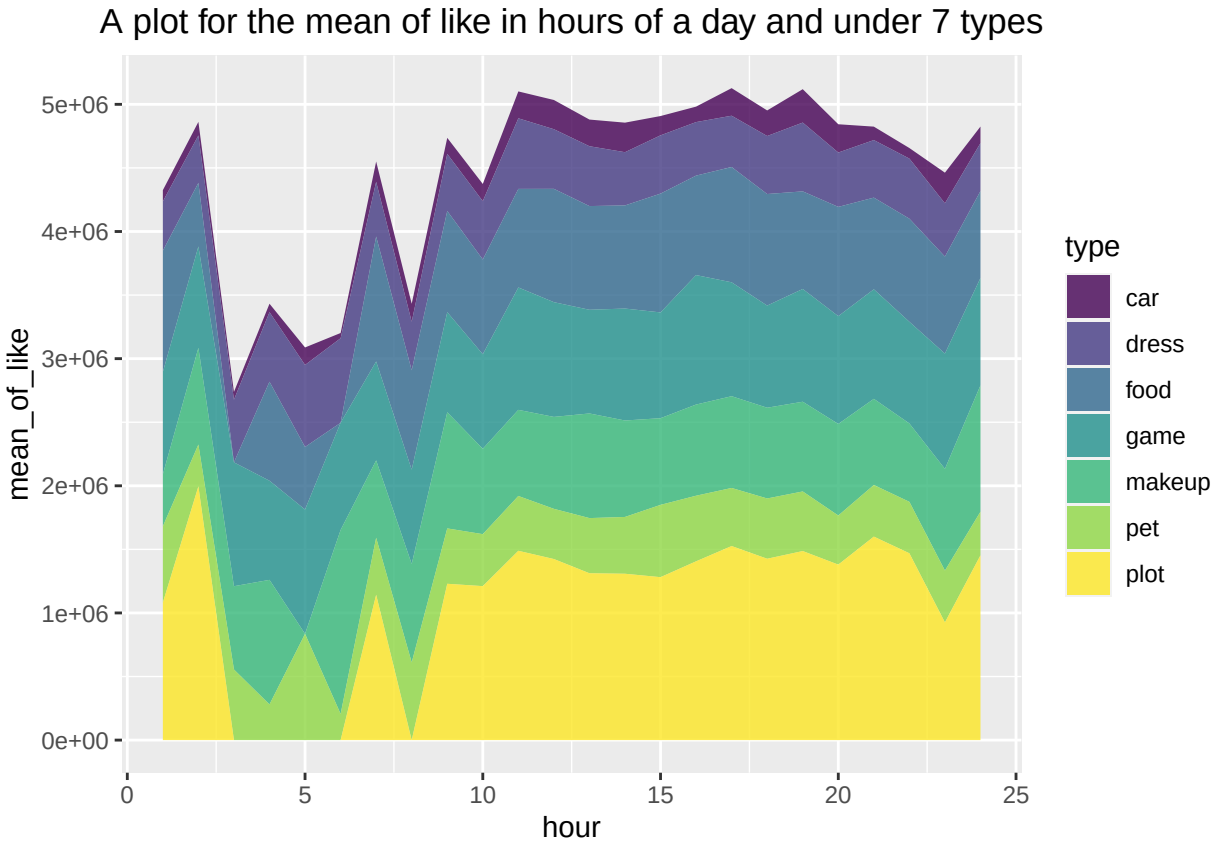
From Monday to Sunday, the average like number of each type keeps stable.

1.2.19 A plot for the mean of like in hours of a day and under 7 types

```
tpFrame <- video %>%
  group_by(hour, type) %>%
  summarize(mean_of_like = mean(like))
```

```
## `summarise()` has grouped output by 'hour'. You can override using the `.groups` argument.
```

```
tpFrame$hour <- as.numeric(tpFrame$hour)
to_merge <-
  data.frame(
    hour = c(3, 4, 5, 6, 8, 3, 6, 5),
    mean_of_like = c(0, 0, 0, 0, 0, 0, 0, 0),
    type = c("plot", "plot", "plot", "plot", "plot", "food", "food", "makeup")
  )
tpFrame <- rbind(tpFrame, to_merge)

ggplot(tpFrame, aes(x = hour, y = mean_of_like, fill = type)) +
  geom_area(alpha = 0.8 , size = 0.5) +
  scale_fill_viridis(discrete = T) +
  ggtitle("A plot for the mean of like in hours of a day and under 7 types") +
  theme(plot.title = element_text(hjust = 0.5))
```



At first sight we may think the videos released at midnight have much less ability to attract likes. However, we should notice that some types have exactly 0 records in some hours at night, and their average like numbers will be plotted as zero. Look at it carefully, we find that the attraction to like of type pet, makeup and game don't decrease–that of makeup even increases for some time!

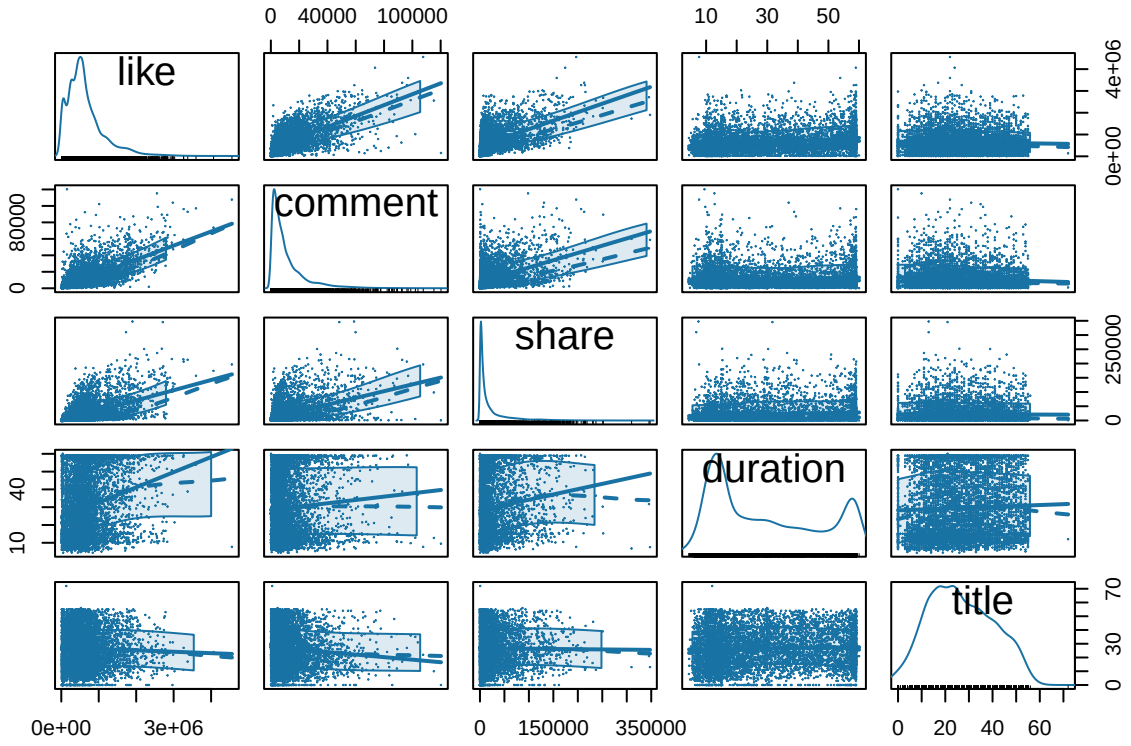
2 Regression Analysis on the number of likes

2.1 Scatterplot & Transformation

```
videos <-
  as.data.frame(video[, c("like", "comment", "share", "duration", "title")])

cor(videos)
```

```
##                like     comment        share   duration        title
## like      1.0000000  0.65334504  0.528246411 0.24206912 -0.037561205
## comment   0.6533450  1.00000000  0.457905718 0.06929331 -0.088128001
## share     0.5282464  0.45790572  1.000000000 0.11463756 -0.008051137
## duration  0.2420691  0.06929331  0.114637562 1.00000000  0.039049993
## title    -0.0375612 -0.08812800 -0.008051137 0.03904999  1.000000000
```

```
library(car)
spm(
  ~ like + comment + share + duration + title,
  data = videos,
  col = "#1972A4",
  cex = 0.05,
  pch = 19
)
```



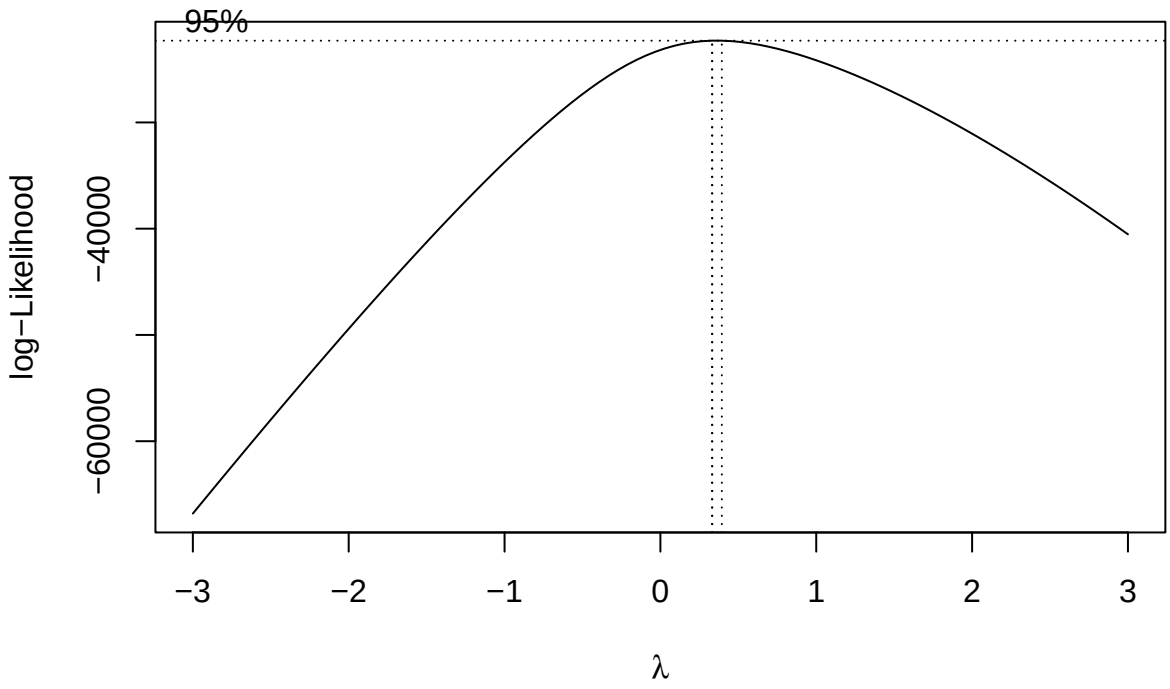
```
library(MASS)
```

```
##
##     'MASS'
```

```
## The following object is masked from 'package:plotly':
##
##     select
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
boxcox(like ~ . - index - author_index,
       lambda = seq(-3, 3, length = 20),
       data = video)
```



The suggested λ is close to 0. So we take the log.

We first use all independent variables.

```
## index is meaningless
fit1 <- lm(log(like) ~ . - like - index - author_index, data = video)
summary(fit1)
```

```
##
## Call:
## lm(formula = log(like) ~ . - like - index - author_index, data = video)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.83462 -0.30435 -0.02799  0.28089  2.75314
##
## Coefficients: (6 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1.110e+01  9.801e-02 113.296  < 2e-16 ***
## authorseveral      4.232e-02  2.079e-02   2.035   0.0419 *
## authoroften        9.377e-02  2.100e-02   4.464 8.17e-06 ***
## authorfrequent     1.791e-01  2.284e-02   7.842 5.20e-15 ***
## comment            1.968e-05  6.552e-07  30.042  < 2e-16 ***
## share              4.821e-06  2.492e-07  19.343  < 2e-16 ***
## BGMOriginal        4.897e-02  2.186e-02   2.240   0.0251 *
## duration           2.845e-03  5.071e-04   5.610 2.11e-08 ***
## typepet            1.355e+00  2.580e-02  52.492  < 2e-16 ***
## typedress          1.494e+00  2.666e-02  56.058  < 2e-16 ***
## typemakeup         1.770e+00  2.718e-02  65.127  < 2e-16 ***
## typefood           1.860e+00  2.754e-02  67.553  < 2e-16 ***
## typegame           2.043e+00  2.691e-02  75.902  < 2e-16 ***
## typeplot           2.211e+00  3.801e-02  58.170  < 2e-16 ***
## title              1.448e-03  5.397e-04   2.684   0.0073 **
## quarter2          -7.115e-02  3.228e-01  -0.220   0.8256
## month2             9.601e-02  9.241e-02   1.039   0.2989
## month3             6.326e-02  1.709e-01   0.370   0.7112
## month4             3.482e-02  1.343e-01   0.259   0.7954
## month5                    NA         NA      NA       NA
## week2             -3.275e-02  6.412e-02  -0.511   0.6095
## week3             -1.021e-01  7.770e-02  -1.315   0.1887
## week4             -4.155e-02  8.373e-02  -0.496   0.6197
## week5             -9.826e-02  8.756e-02  -1.122   0.2618
## week6             -7.714e-02  1.126e-01  -0.685   0.4935
## week7             -2.056e-01  1.316e-01  -1.562   0.1183
## week8             -2.837e-01  1.464e-01  -1.939   0.0526 .
## week9             -1.762e-01  1.639e-01  -1.075   0.2824
## week10            -1.730e-01  1.878e-01  -0.921   0.3571
## week11            -2.298e-01  2.091e-01  -1.099   0.2719
## week12            -2.277e-01  2.270e-01  -1.003   0.3159
## week13            -1.616e-01  2.300e-01  -0.702   0.4825
## week14            -3.517e-02  2.475e-01  -0.142   0.8870
## week15            -1.205e-01  2.682e-01  -0.449   0.6531
## week16            -1.287e-01  2.870e-01  -0.448   0.6539
## week17            -5.613e-02  3.020e-01  -0.186   0.8525
## week18            -7.422e-02  3.227e-01  -0.230   0.8181
## week19            -3.410e-01  3.529e-01  -0.966   0.3340
## weekday2          -3.614e-02  3.655e-02  -0.989   0.3228
## weekday3           9.169e-03  3.699e-02   0.248   0.8042
## weekday4           6.263e-02  3.099e-02   2.021   0.0433 *
## weekday5           6.005e-02  3.209e-02   1.871   0.0613 .
## weekday6           3.279e-02  3.528e-02   0.930   0.3526
## weekday7           6.348e-02  3.380e-02   1.878   0.0604 .
```

```
## day2            -1.408e-01  5.811e-02  -2.423   0.0154 *
## day3            -1.148e-01  5.891e-02  -1.948   0.0515 .
## day4            -4.296e-03  5.706e-02  -0.075   0.9400
## day5             3.579e-02  6.201e-02   0.577   0.5639
## day6            -4.839e-02  6.357e-02  -0.761   0.4466
## day7            -5.471e-02  6.627e-02  -0.826   0.4090
## day8            -5.943e-02  6.834e-02  -0.870   0.3846
## day9            -9.899e-02  7.185e-02  -1.378   0.1683
## day10           -5.742e-02  7.282e-02  -0.789   0.4304
## day11           -3.195e-02  7.274e-02  -0.439   0.6605
## day12            3.124e-02  7.692e-02   0.406   0.6847
## day13           -5.188e-02  7.627e-02  -0.680   0.4964
## day14           -5.193e-03  7.937e-02  -0.065   0.9478
## day15            5.506e-02  7.805e-02   0.705   0.4805
## day16           -4.586e-02  8.093e-02  -0.567   0.5710
## day17            1.789e-02  7.957e-02   0.225   0.8221
## day18           -1.714e-02  8.055e-02  -0.213   0.8315
## day19            3.608e-03  8.059e-02   0.045   0.9643
## day20            4.176e-02  8.081e-02   0.517   0.6053
## day21            3.743e-02  8.019e-02   0.467   0.6406
## day22            1.067e-01  8.206e-02   1.301   0.1935
## day23            9.447e-02  8.105e-02   1.166   0.2439
## day24           -2.280e-02  8.238e-02  -0.277   0.7820
## day25           -8.105e-03  8.365e-02  -0.097   0.9228
## day26           -1.539e-01  8.005e-02  -1.923   0.0545 .
## day27           -1.093e-01  7.949e-02  -1.375   0.1692
## day28           -2.708e-02  8.071e-02  -0.336   0.7372
## day29            1.008e-02  7.747e-02   0.130   0.8965
## day30           -1.477e-01  8.339e-02  -1.771   0.0766 .
## day31                   NA         NA      NA       NA
## hour1           -1.356e-02  1.019e-01  -0.133   0.8941
## hour2           -2.759e-02  1.324e-01  -0.208   0.8350
## hour3           -7.003e-02  1.287e-01  -0.544   0.5865
## hour4            1.813e-01  1.589e-01   1.141   0.2539
## hour5           -1.166e-01  1.383e-01  -0.843   0.3995
## hour6           -9.614e-02  1.274e-01  -0.755   0.4505
## hour7           -8.766e-03  9.582e-02  -0.091   0.9271
## hour8           -6.792e-02  8.321e-02  -0.816   0.4143
## hour9           -4.802e-02  8.082e-02  -0.594   0.5524
## hour10          -3.743e-02  7.709e-02  -0.486   0.6273
## hour11          -2.504e-02  7.518e-02  -0.333   0.7392
## hour12          -2.983e-02  7.551e-02  -0.395   0.6928
## hour13          -1.666e-02  7.948e-02  -0.210   0.8340
## hour14          -7.866e-02  8.029e-02  -0.980   0.3273
## hour15          -6.560e-02  8.039e-02  -0.816   0.4145
## hour16           3.013e-02  7.751e-02   0.389   0.6975
## hour17          -1.743e-02  7.412e-02  -0.235   0.8141
## hour18           1.357e-02  7.426e-02   0.183   0.8550
## hour19          -7.848e-02  7.663e-02  -1.024   0.3058
## hour20          -8.403e-02  8.020e-02  -1.048   0.2948
## hour21          -8.209e-02  8.447e-02  -0.972   0.3312
## hour22          -4.412e-02  9.097e-02  -0.485   0.6277
## hour23          -1.679e-01  9.470e-02  -1.773   0.0763 .
## minute          -4.799e-04  4.108e-04  -1.168   0.2428
```

```
## second            -6.753e-04  4.193e-04  -1.611   0.1073
## period12:00-16:00         NA         NA      NA       NA
## period16:00-18:00         NA         NA      NA       NA
## period18:00-20:00         NA         NA      NA       NA
## period20:00-9:00          NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5641 on 6034 degrees of freedom
## Multiple R-squared:  0.714,  Adjusted R-squared:  0.7095
## F-statistic: 156.9 on 96 and 6034 DF,  p-value: < 2.2e-16
```

2.2 fit2

Then we use the stepwise method to delete some useless independent variables so that the model is not too complicated.

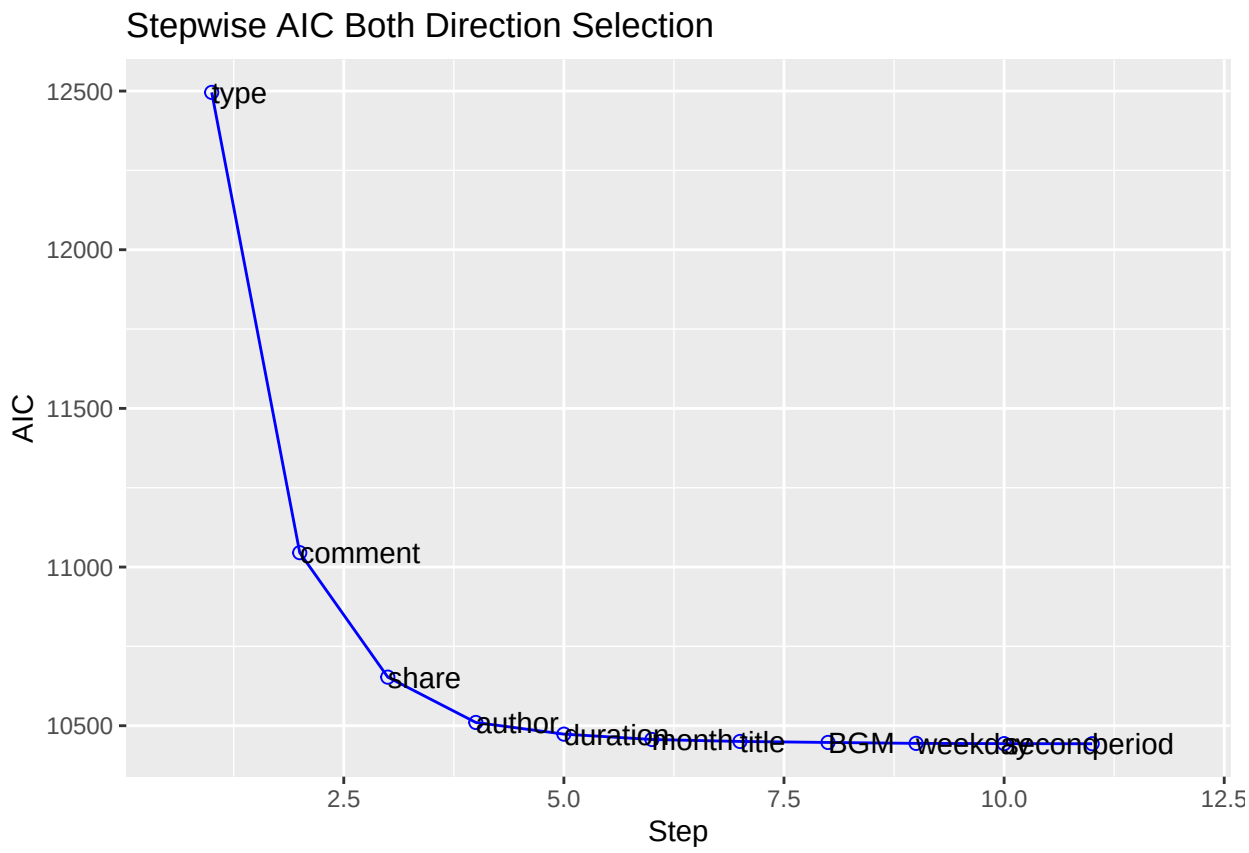
```
library(olsrr)
```

```
##
##    'olsrr'

## The following object is masked from 'package:MASS':
##
##     cement

## The following object is masked from 'package:datasets':
##
##     rivers
```

```
step_plot <- ols_step_both_aic(fit1)
step_plot
```

```
##
##
##                                 Stepwise Summary
## -------------------------------------------------------------------------------------
## Variable     Method        AIC         RSS        Sum Sq      R-Sq      Adj. R-Sq
## -------------------------------------------------------------------------------------
## type        addition    12495.750    2748.340    3965.874    0.59067      0.59027
## comment     addition    11045.325    2168.636    4545.578    0.67701      0.67664
## share       addition    10653.106    2033.583    4680.631    0.69712      0.69673
## author      addition    10510.275    1984.812    4729.402    0.70439      0.70386
## duration    addition    10473.482    1972.293    4741.921    0.70625      0.70567
## month       addition    10456.602    1964.305    4749.909    0.70744      0.70668
## title       addition    10450.432    1961.689    4752.524    0.70783      0.70702
## BGM         addition    10447.304    1960.049    4754.164    0.70807      0.70721
## weekday     addition    10444.387    1955.286    4758.928    0.70878      0.70764
## second      addition    10443.534    1954.376    4759.837    0.70892      0.70773
## period      addition    10443.221    1951.728    4762.485    0.70931      0.70793
## -------------------------------------------------------------------------------------
```

```
plot(step_plot)
```



Redundant variables: quarter, day, hour and minute.

Based on the stepwise output, the new model is given by

```
fit2 <-
  lm(
    log(like) ~ author + comment + share + BGM + duration +
      type + title + month + weekday + second + period,
    data = video
  )
```

Use anova() to verify

```
anova(fit2, fit1)
```

```
## Analysis of Variance Table
##
## Model 1: log(like) ~ author + comment + share + BGM + duration + type +
##     title + month + weekday + second + period
## Model 2: log(like) ~ (index + author + comment + share + BGM + duration +
##     type + title + quarter + month + week + weekday + day + hour +
##     minute + second + period + author_index) - like - index -
##     author_index
```

```
##   Res.Df    RSS Df Sum of Sq      F   Pr(>F)
## 1   6101 1951.7
## 2   6034 1920.1 67    31.604 1.4823 0.006664 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

ANOVA suggests that there is a significant effect to add on more predictors. This contradicts the stepwise method, however, we prefer fit2 since fit1 has too many coefficients.

2.3 fit3

- Covariates Transformation

We first see the scatter plot for log(like) and comment, log(like) and share.

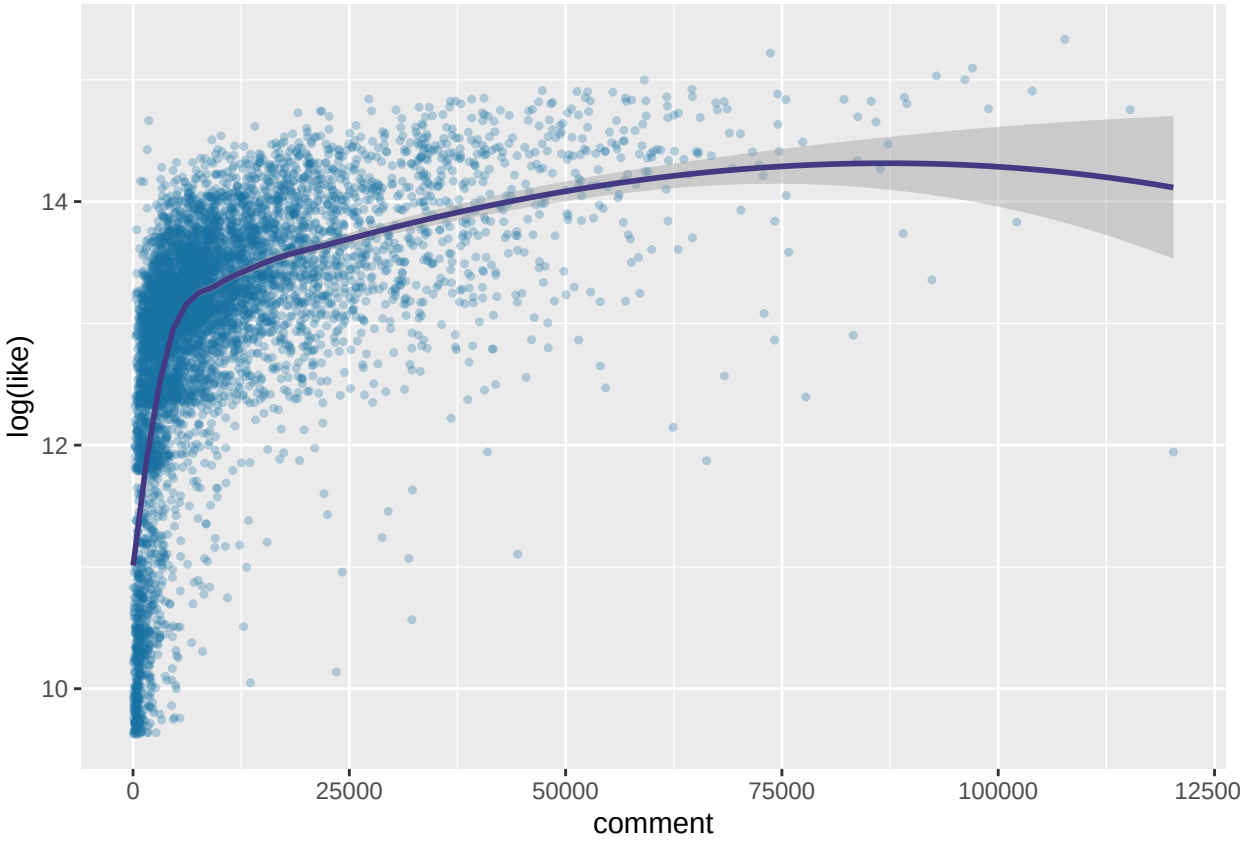
```
library(patchwork)
```

```
##
##     'patchwork'
```

```
## The following object is masked from 'package:MASS':
##
##     area
```

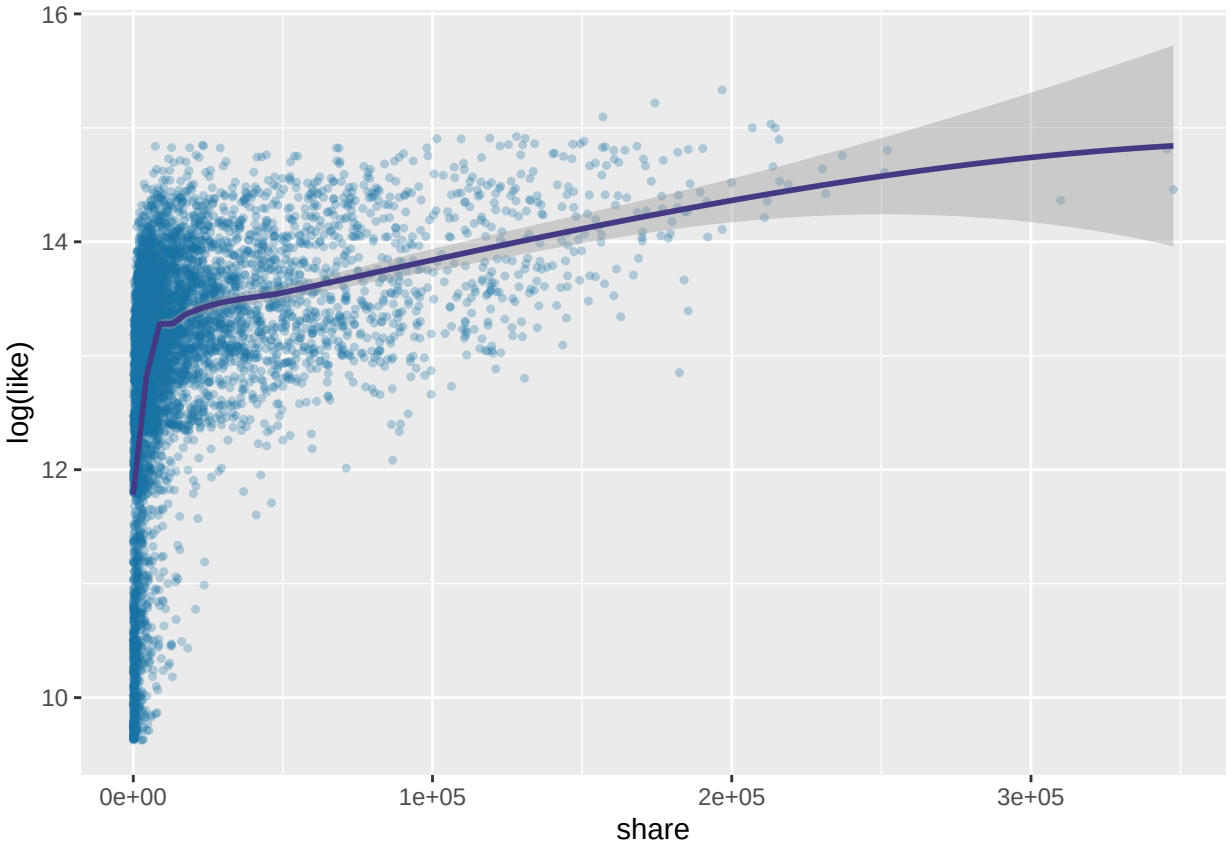
```
p1 <-
  ggplot(video, aes(
    x = (video$comment),
    y = log(video$like))) +
  labs(x = "comment",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9, color = "#1972A4") +
  geom_smooth(method = "loess", color = viridis_palatte[2])

p2 <-
  ggplot(video, aes(
    x = (video$share),
    y = log(video$like)
  )) +
  labs(x = "share",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9, color = "#1972A4") +
  geom_smooth(method = "loess", color = viridis_palatte[2])

p1
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
p2
```

```
## `geom_smooth()` using formula 'y ~ x'
```



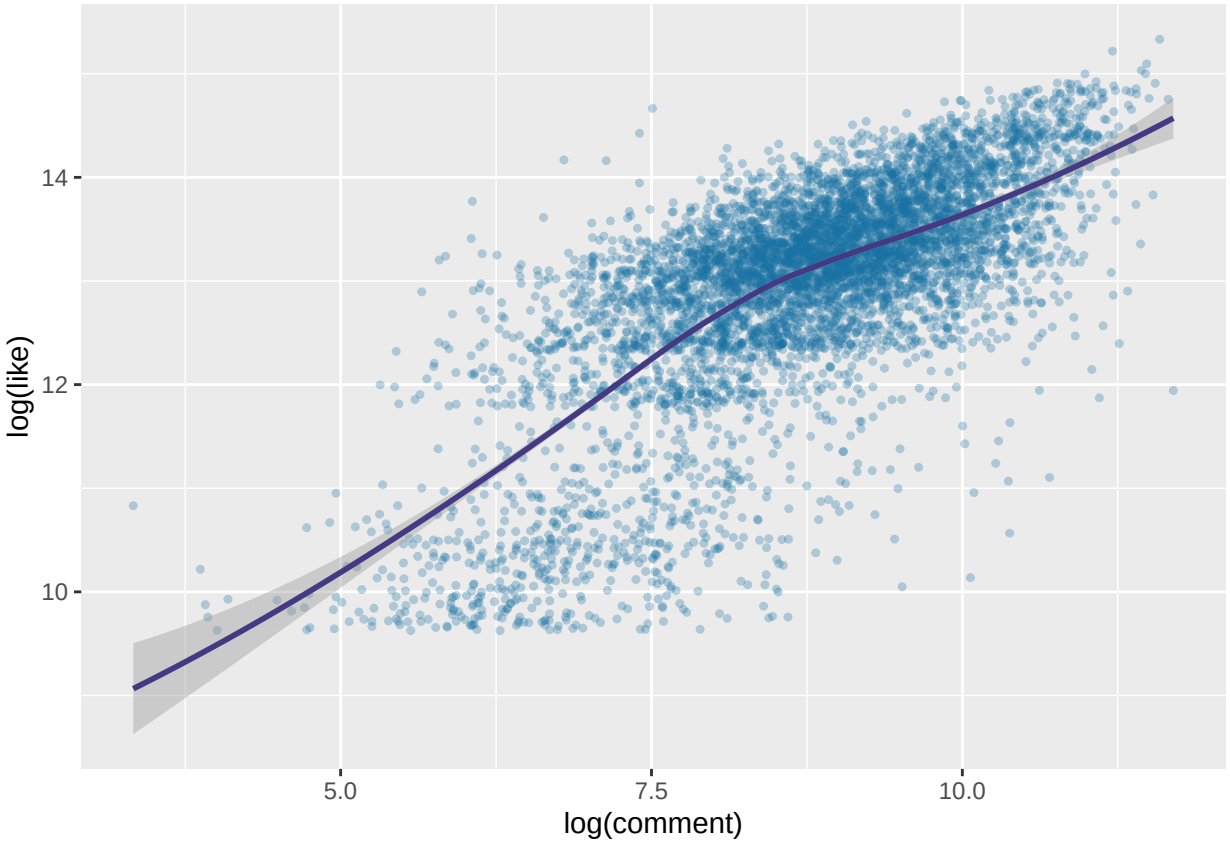
The linear relation is not significant. So we consider transformation for covariates: 'comment' and 'share'. Since we have taken the logarithm for 'like' and the number of likes, comments, and shares belong to the same level to some extent, it is reasonable to take the logarithm for 'comment' and 'share' too.

After taking the logarithm, the scatter plots are given by

```
p3 <-
  ggplot(video, aes(
    x = log(video$comment),
    y = log(video$like)
  )) +
  labs(x = "log(comment)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9, color = "#1972A4") +
  geom_smooth(method = "loess", color = viridis_palatte[2])

p4 <-
  ggplot(video, aes(
    x = log(video$share),
    y = log(video$like)
  )) +
  labs(x = "log(share)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9, color = "#1972A4") +
  geom_smooth(method = "loess", color = viridis_palatte[2])
```

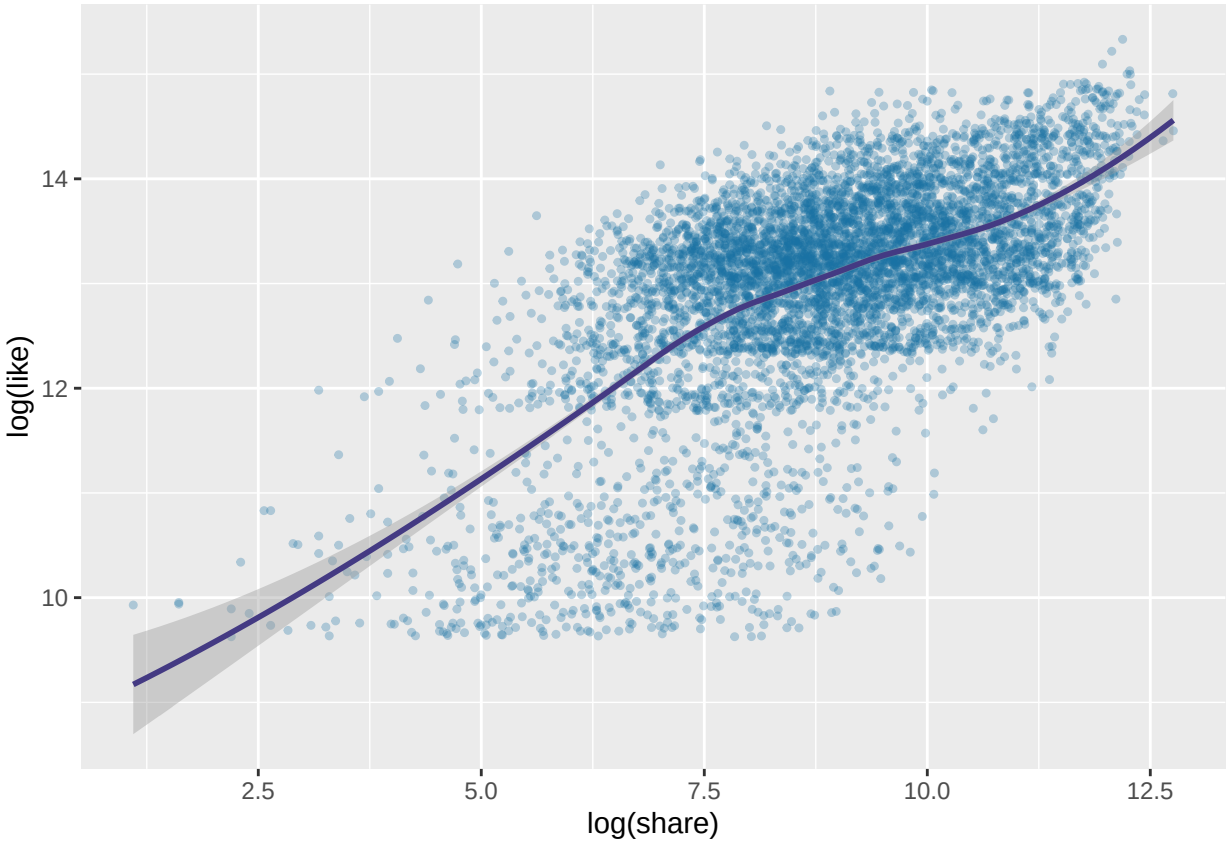
```
p3
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
p4
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Obviously, there is a stronger linear relation after taking the logarithm. Now, we have

```
fit3 <-
  lm(
    log(like) ~ author + log(comment) + log(share) + BGM + duration +
      type + title + month + weekday + second + period,
    data = video
  )
```

Comparison: (fit1, fit2, fit3)

```
library(stargazer)
```

```
##
## Please cite as:
```

```
##  Hlavac, Marek (2018). stargazer: Well-Formatted Regression and Summary Statistics Tables.
```

```
##  R package version 5.2.2. https://CRAN.R-project.org/package=stargazer
```

```
tp <-
  c(
    "author",
    "BGM",
    "duration",
```

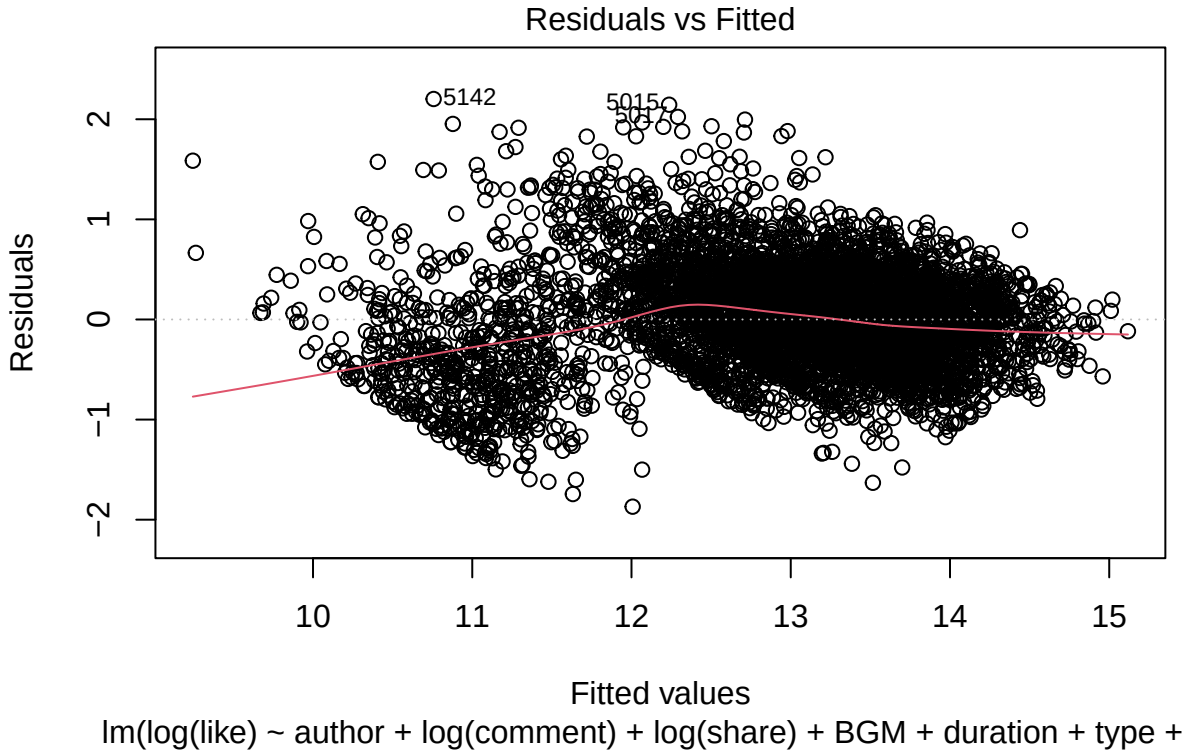
```
    "type",
    "title",
    "quarter",
    "month",
    "week",
    "weekday",
    "day",
    "hour",
    "minute",
    "second",
    "period"
  )

stargazer(fit1,
          fit2,
          fit3,
          omit = tp,
          omit.labels = tp,
          type = "text")
```

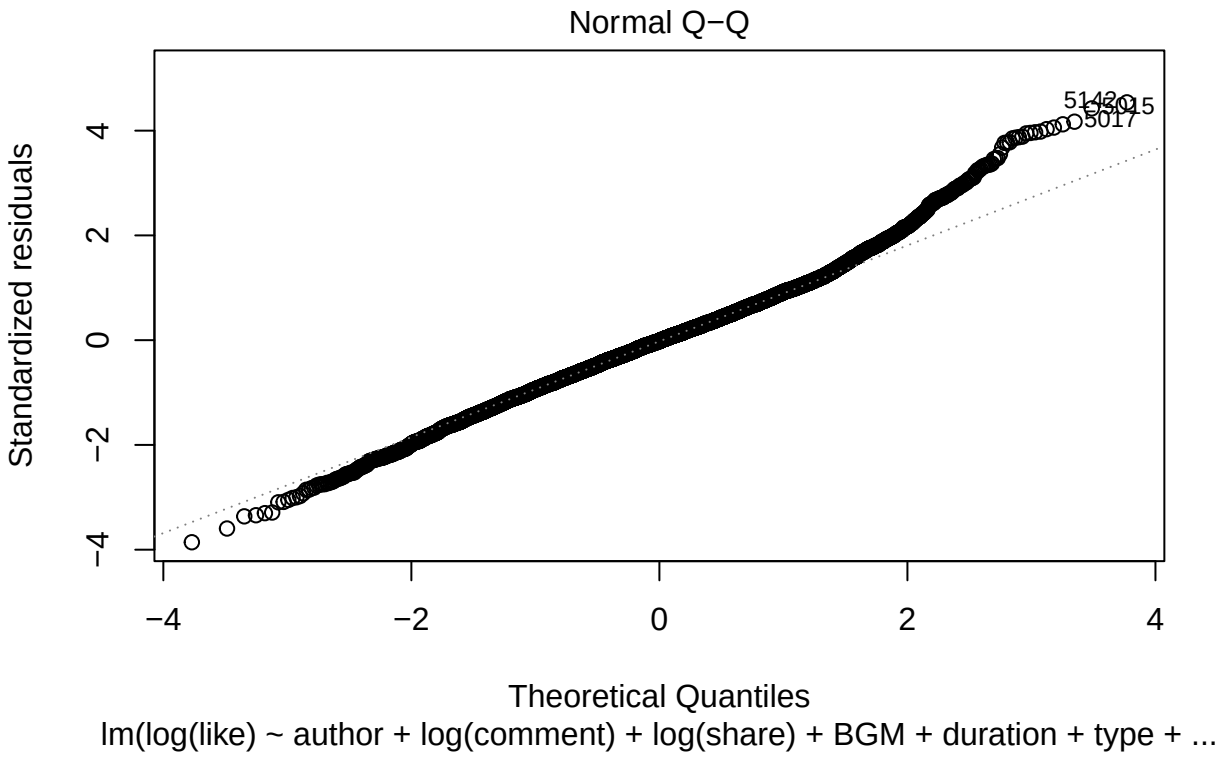
```
##
## ================================================================================================
##                                              Dependent variable:
##                      ------------------------------------------------------------------------
##                                                  log(like)
##                                (1)                   (2)                    (3)
## ------------------------------------------------------------------------------------------------
## comment                     0.00002***            0.00002***
##                             (0.00000)             (0.00000)
##
## share                       0.00000***            0.00000***
##                             (0.00000)             (0.00000)
##
## log(comment)                                                             0.314***
##                                                                          (0.007)
##
## log(share)                                                               0.131***
##                                                                          (0.005)
##
## Constant                    11.105***             10.979***              7.682***
##                              (0.098)               (0.043)               (0.062)
##
## ------------------------------------------------------------------------------------------------
## author                         Yes                   Yes                    Yes
## BGM                            Yes                   Yes                    Yes
## duration                       Yes                   Yes                    Yes
## type                           Yes                   Yes                    Yes
## title                          Yes                   Yes                    Yes
## quarter                        Yes                    No                     No
## month                          Yes                   Yes                    Yes
## week                           Yes                   Yes                    Yes
## weekday                        Yes                   Yes                    Yes
## day                            Yes                   Yes                    Yes
```

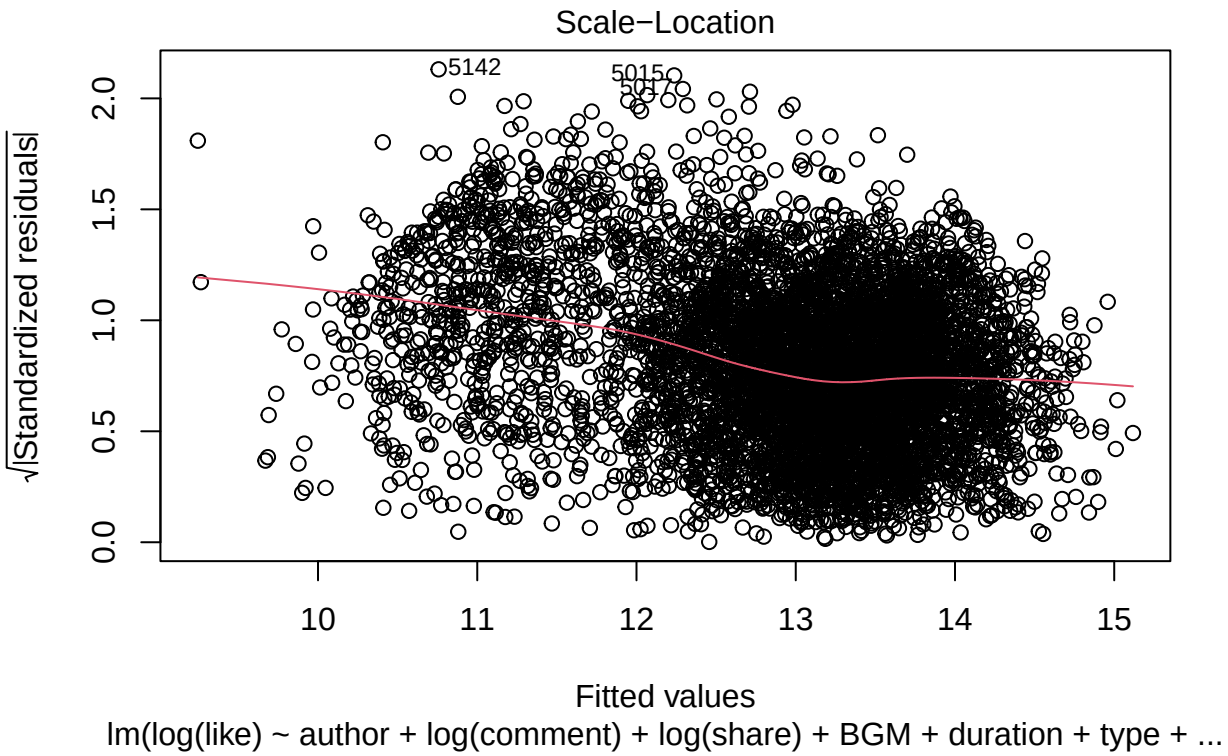
```
## hour                          Yes                          No                          No
## minute                        Yes                          No                          No
## second                        Yes                         Yes                         Yes
## period                        Yes                         Yes                         Yes
## ------------------------------------------------------------------------------------------------------
## Observations                6,131                       6,131                       6,131
## R2                          0.714                       0.709                       0.785
## Adjusted R2                 0.709                       0.708                       0.784
## Residual Std. Error     0.564 (df = 6034)           0.566 (df = 6101)           0.486 (df = 6101)
## F Statistic         156.932*** (df = 96; 6034) 513.354*** (df = 29; 6101) 769.419*** (df = 29; 6101)
## ======================================================================================================
## Note:                                                                      *p<0.1; **p<0.05; ***p<0.01
```

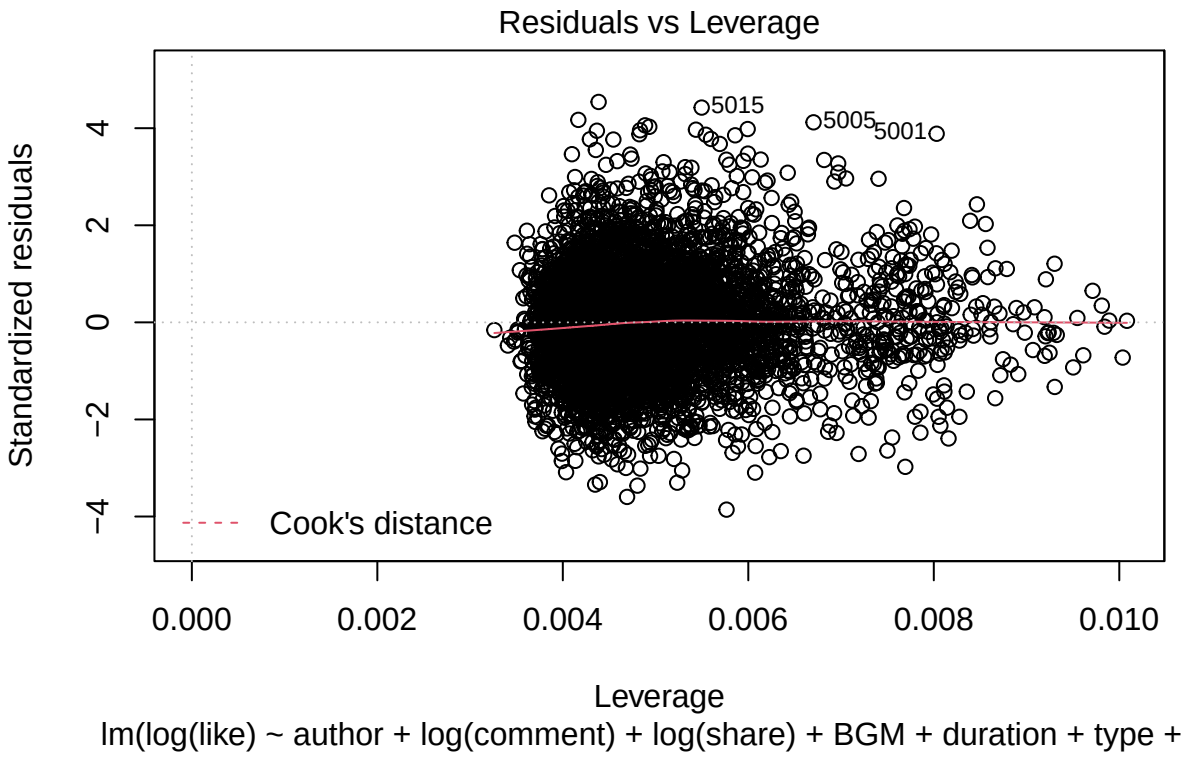
2.4 fit4

```
plot(fit3)
```




Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(like) ~ author + log(comment) + log(share) + BGM + duration + type + ...
5142
5015
5017
−4
−2
0
2
4


Scale−Location
5142
5015
5017
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
10
11
12
13
14
15
Fitted values
lm(log(like) ~ author + log(comment) + log(share) + BGM + duration + type + ...



Significantly, there are two clusters in the residual plot. Based on our previous analysis, we guess these data are videos with type car. If these data with fitted values less than 11.5 are videos with type car, then there may exist an interaction between type and other variables. Let's verify the guess first.

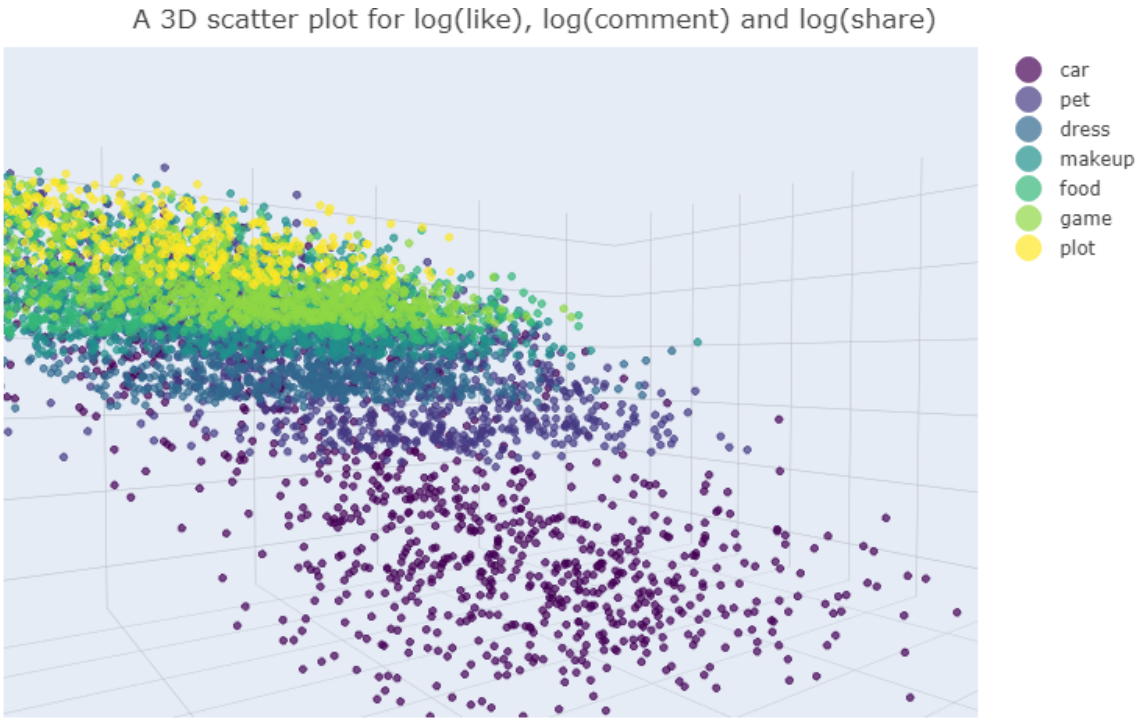
```
index <- which(fitted(fit3) < 11.5)
table(video[index, ]$type)
```

```
##
##    car    pet  dress makeup   food   game   plot
##    618      0      0      0      0      0      0
```

All videos with fitted values < 11.5 are type car. This reminds us to consider the interaction of types with other variables. Let's first look at the 3D scatter plot for 'log(like)', 'log(comment)' and 'log(share)'

```
library(plotly)
fig <-
  plot_ly(
    x = log(video$comment),
    y = log(video$share),
    z = log(video$like),
    color = video$type,
    colors = viridis(7)
  ) %>% add_markers(size = 12)

fig <- fig %>%
  layout(title = "A 3D scatter plot for log(like), log(comment) and log(share)",
```

```
         scene = list(bgcolor = "#e5ecf6"))
fig
```

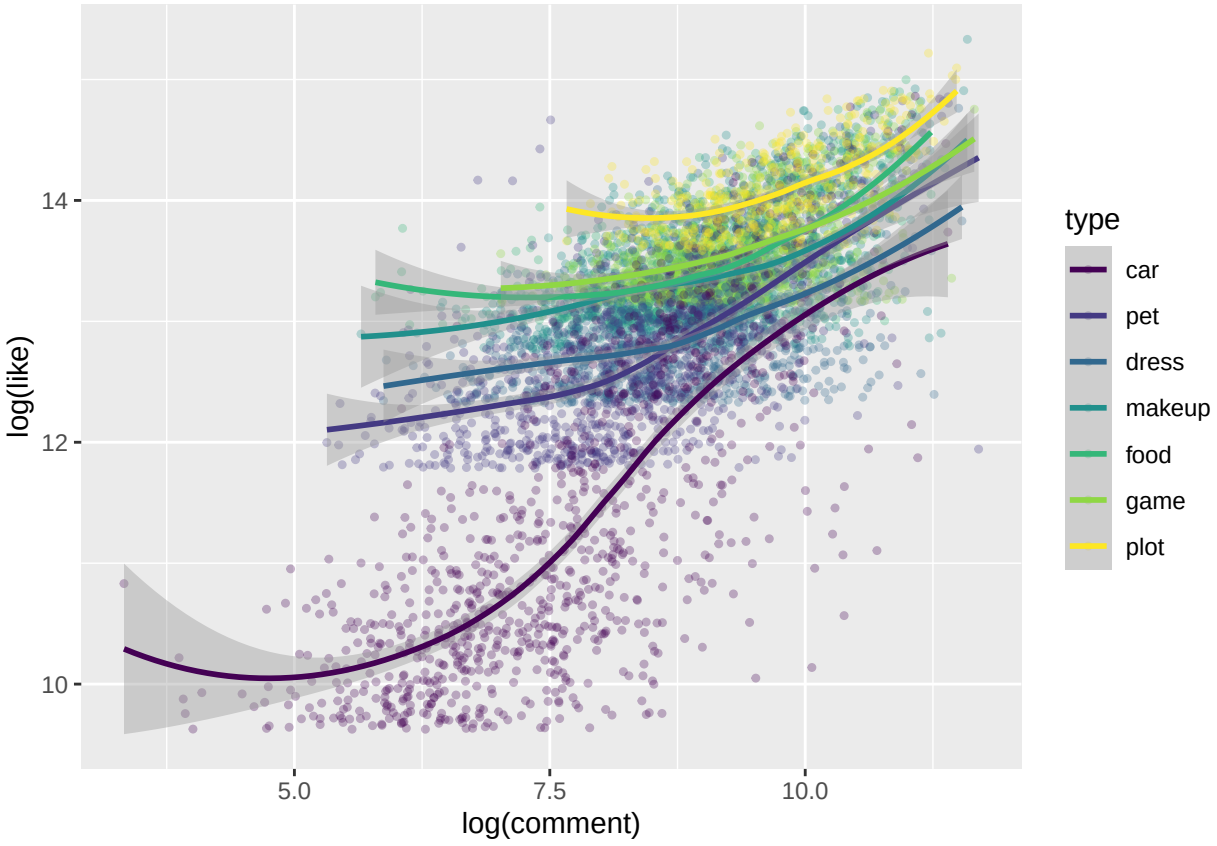


The type has a decisive influence on the number of likes. The linear relationship between 'log(like)', 'log(comment)' and 'log(share)' of different types of short videos is also different. We then use the smoothing method to see the relation further.

```
library(patchwork)
p1 <-
  ggplot(video, aes(
    x = log(video$comment),
    y = log(video$like),
    color = video$type
  )) +
  scale_color_viridis(discrete = TRUE, option = "D")+
  scale_fill_viridis(discrete = TRUE) +
  labs(x = "log(comment)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9) +
  geom_smooth(method = "loess")

p2 <-
  ggplot(video, aes(
    x = log(video$comment),
    y = log(video$like),
    color = video$type
  )) +
```

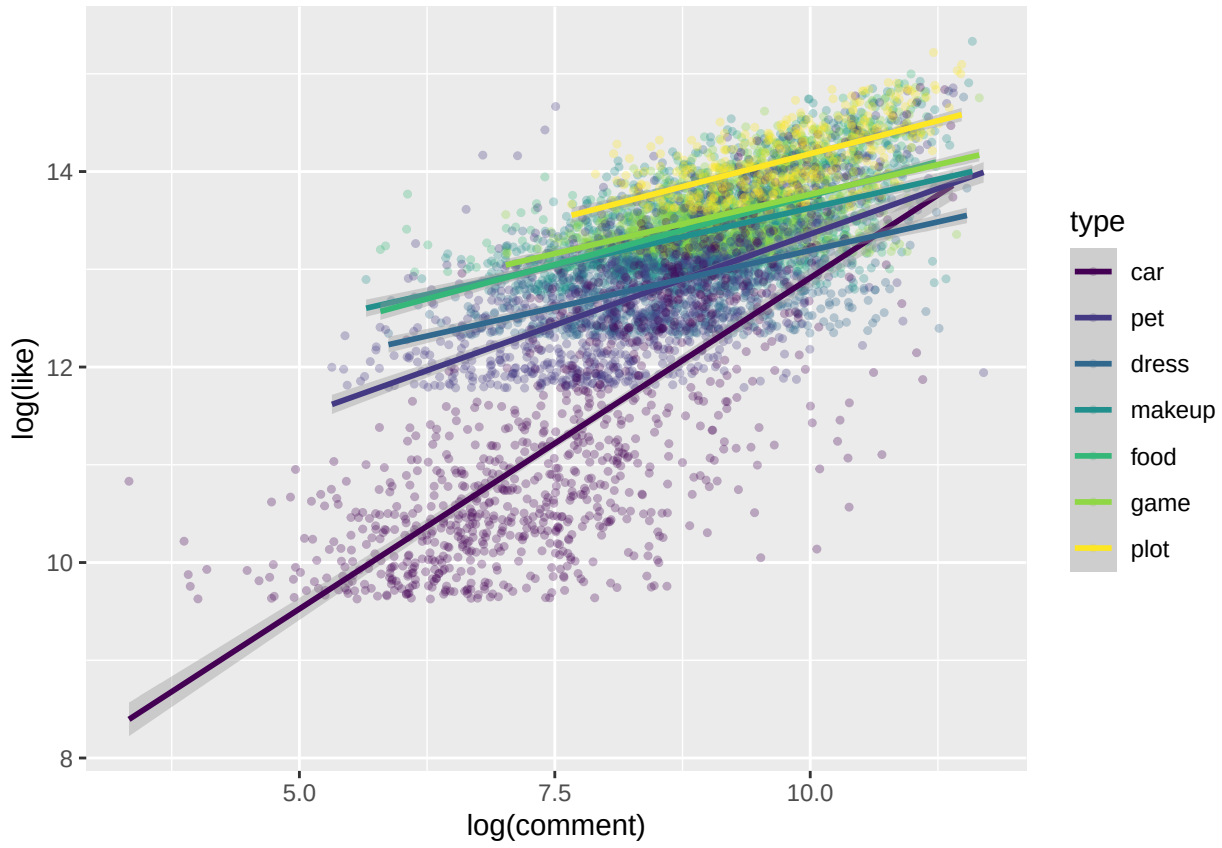
```
  scale_color_viridis(discrete = TRUE, option = "D")+
  scale_fill_viridis(discrete = TRUE) +
  labs(x = "log(comment)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9) +
  geom_smooth(method = "lm")

p3 <-
  ggplot(video, aes(
    x = log(video$share),
    y = log(video$like),
    color = video$type
  )) +
  scale_color_viridis(discrete = TRUE, option = "D") +
  scale_fill_viridis(discrete = TRUE) +
  labs(x = "log(share)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9) +
  geom_smooth(method = "loess")

p4 <-
  ggplot(video, aes(
    x = log(video$share),
    y = log(video$like),
    color = video$type
  )) +
  scale_color_viridis(discrete = TRUE, option = "D") +
  scale_fill_viridis(discrete = TRUE) +
  labs(x = "log(share)",
       y = "log(like)",
       color = "type") +
  geom_point(alpha = .3, size = .9) +
  geom_smooth(method = "lm")

p1
```

```
## `geom_smooth()` using formula 'y ~ x'
```



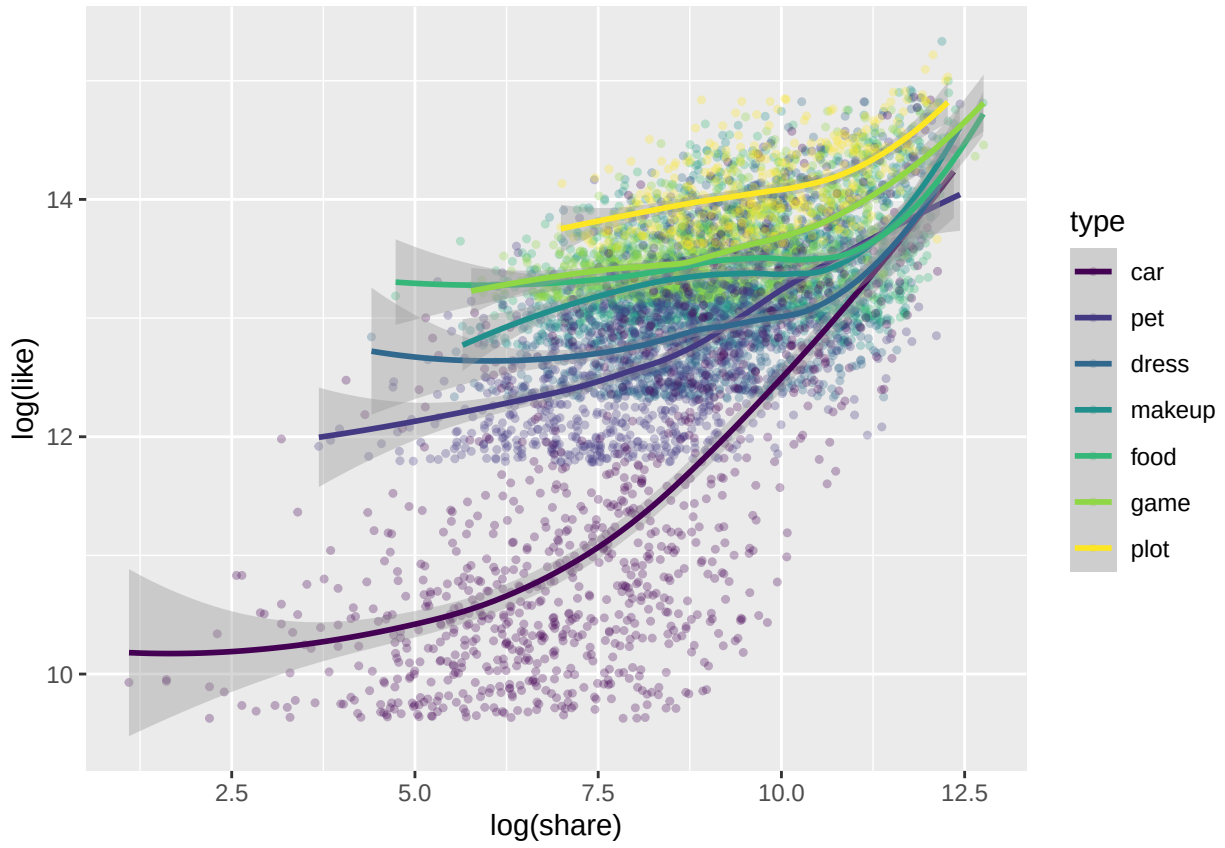

```
p2
```

```
## `geom_smooth()` using formula 'y ~ x'
```



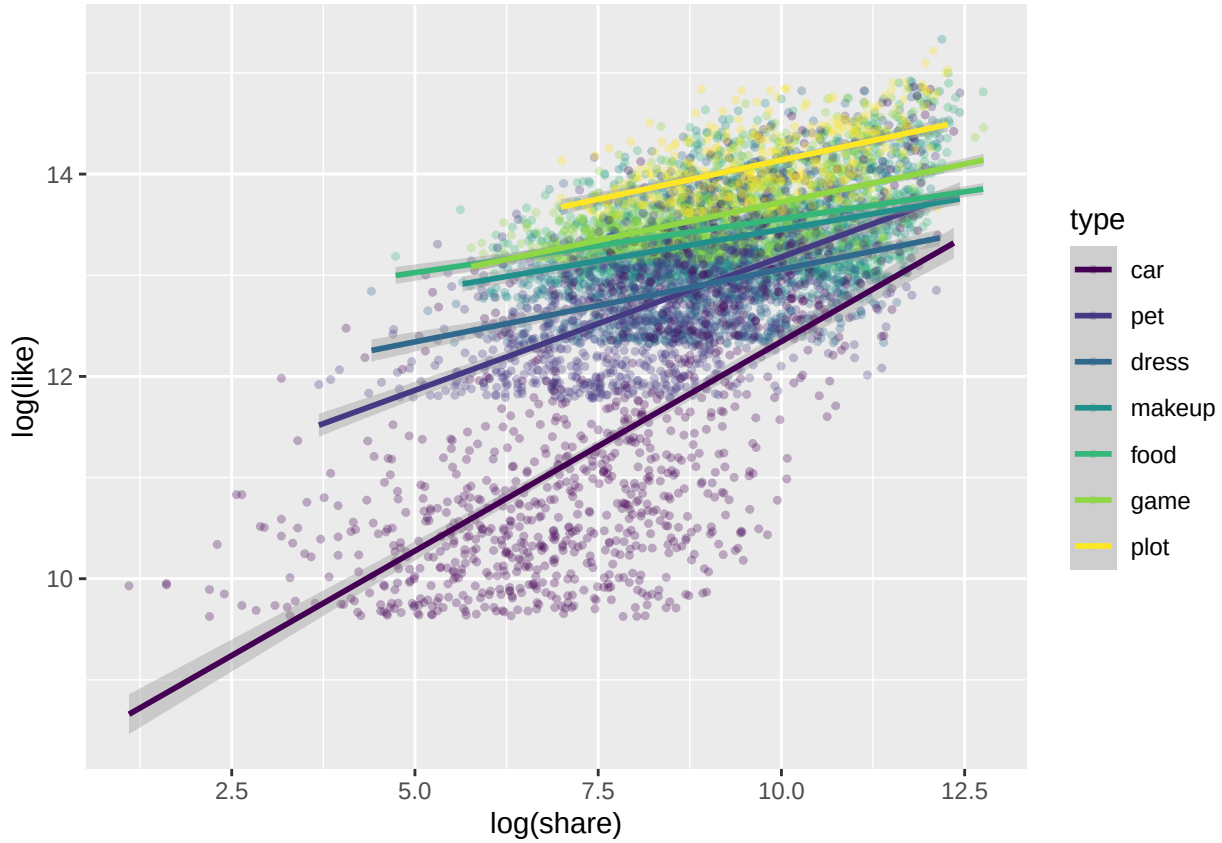

```
p3
```

```
## `geom_smooth()` using formula 'y ~ x'
```




```
p4
```

```
## `geom_smooth()` using formula 'y ~ x'
```



We found that the slopes of short videos in the car type are significantly different from those of other short videos, and the slopes of other types are basically the same.

Therefore, it is reasonable to consider the interaction between type and other variables. Then, we revise our linear model by introducing the interaction.

Now, we obtain fit4 by introducing a new variable: 'iscar' with values 1 and 0, where 1 represents the car type and 0 means non-car type and consider the interaction.

```
ind <- video$type == "car"
video$iscar <- 0
video[ind,]$iscar <- 1

fit4 <-
  lm(
    log(like) ~ author + log(comment) + log(share) + BGM + duration + type +
      title + month + weekday + second + period + iscar:(
        author + log(comment) + log(share) + BGM + duration + title + month +
          weekday + second + period
      ),
    data = video
  )

## compare R-square
data.frame(summary(fit3)$r.square, summary(fit4)$r.square)
```

```
##   summary.fit3..r.square summary.fit4..r.square
## 1               0.785283              0.8205919
```

Compared with fit3, we find that R-squared is improved significantly.

2.5 fit5

Moreover, are there interactions between other numerical variables? Let's see.

```
temp <- video[, c(4, 5, 7, 9)]
temp$comment <- log(temp$comment)
temp$share <- log(temp$share)
names(temp) <-
  c("log(comment)", "log(share)", "duration", "title")

cor(temp)
```

```
##              log(comment)  log(share)   duration       title
## log(comment)   1.00000000  0.55667947 0.08462539 -0.09969676
## log(share)     0.55667947  1.00000000 0.13118510 -0.02863499
## duration       0.08462539  0.13118510 1.00000000  0.03904999
## title         -0.09969676 -0.02863499 0.03904999  1.00000000
```

There is a strong correlation between comments and shares. This may be due to the fact that comments and shares both like a video to a deeper level than likes. Sharing will make the video be seen by more people, so the probability of the video being commented Increase

We add 'log(comment):log(share)' and 'log(share):duration' into the model and get fit5.

```
fit5 <-
  lm(
    log(like) ~ author + log(comment) + log(share) + BGM + duration + type +
      title + month + weekday + second + period + iscar:(
        author + log(comment) + log(share) + BGM + duration + title + month +
          weekday + second + period
      ) + log(comment):log(share) + log(share):duration,
    data = video
  )

## compare R-square
data.frame(summary(fit3)$r.square,
           summary(fit4)$r.square,
           summary(fit5)$r.square)
```

```
##   summary.fit3..r.square summary.fit4..r.square summary.fit5..r.square
## 1               0.785283              0.8205919              0.8304033
```

R-squared is improved.

2.6 fit6

Now we use the stepwise method to obtain fit6 after variable selection and introducing interactions.

```r
library(MASS)
fit6 <- stepAIC(fit5, direction = "both")
```

```
## Start:  AIC=-10211.31
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + second + period + iscar:(author +
##     log(comment) + log(share) + BGM + duration + title + month +
##     weekday + second + period) + log(comment):log(share) + log(share):duration
##
##                            Df Sum of Sq    RSS      AIC
## - weekday:iscar             6     0.822 1139.5 -10218.9
## - second:iscar              1     0.000 1138.7 -10213.3
## - period:iscar              4     1.383 1140.1 -10211.9
## <none>                                  1138.7 -10211.3
## - log(share):duration       1     0.830 1139.5 -10208.8
## - title:iscar               1     0.920 1139.6 -10208.4
## - duration:iscar            1     0.949 1139.7 -10208.2
## - BGM:iscar                 1     1.494 1140.2 -10205.3
## - author:iscar              3     2.618 1141.3 -10203.2
## - month:iscar               4    17.386 1156.1 -10126.4
## - log(share):iscar          1    54.040 1192.8  -9929.0
## - log(comment):log(share)   1    65.854 1204.6  -9868.6
## - log(comment):iscar        1   111.410 1250.1  -9641.0
## - type                      5   289.011 1427.7  -8834.6
##
## Step:  AIC=-10218.89
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + second + period + author:iscar +
##     log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
##     title:iscar + month:iscar + second:iscar + period:iscar +
##     log(comment):log(share) + log(share):duration
##
##                            Df Sum of Sq    RSS      AIC
## - second:iscar              1     0.002 1139.5 -10220.9
## - period:iscar              4     1.401 1140.9 -10219.4
## <none>                                  1139.5 -10218.9
## - weekday                   6     2.248 1141.8 -10218.8
## - log(share):duration       1     0.858 1140.4 -10216.3
## - title:iscar               1     0.954 1140.5 -10215.8
## - duration:iscar            1     1.013 1140.5 -10215.4
## - BGM:iscar                 1     1.452 1141.0 -10213.1
## + weekday:iscar             6     0.822 1138.7 -10211.3
## - author:iscar              3     2.678 1142.2 -10210.5
## - month:iscar               4    17.523 1157.0 -10133.3
## - log(share):iscar          1    54.669 1194.2  -9933.6
## - log(comment):log(share)   1    66.201 1205.7  -9874.7
## - log(comment):iscar        1   111.590 1251.1  -9648.1
## - type                      5   289.449 1429.0  -8841.2
##
## Step:  AIC=-10220.88
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + second + period + author:iscar +
##     log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
```

```
##     title:iscar + month:iscar + period:iscar + log(comment):log(share) +
##     log(share):duration
##
##                            Df Sum of Sq    RSS      AIC
## - second                    1     0.076 1139.6 -10222.5
## - period:iscar              4     1.401 1140.9 -10221.3
## <none>                                  1139.5 -10220.9
## - weekday                   6     2.254 1141.8 -10220.8
## + second:iscar              1     0.002 1139.5 -10218.9
## - log(share):duration       1     0.858 1140.4 -10218.3
## - title:iscar               1     0.958 1140.5 -10217.7
## - duration:iscar            1     1.012 1140.5 -10217.4
## - BGM:iscar                 1     1.454 1141.0 -10215.1
## + weekday:iscar             6     0.823 1138.7 -10213.3
## - author:iscar              3     2.679 1142.2 -10212.5
## - month:iscar               4    17.539 1157.1 -10135.2
## - log(share):iscar          1    54.875 1194.4  -9934.5
## - log(comment):log(share)   1    66.200 1205.7  -9876.7
## - log(comment):iscar        1   112.213 1251.8  -9647.0
## - type                      5   289.455 1429.0  -8843.1
##
## Step:  AIC=-10222.47
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + period + author:iscar +
##     log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
##     title:iscar + month:iscar + period:iscar + log(comment):log(share) +
##     log(share):duration
##
##                            Df Sum of Sq    RSS      AIC
## - period:iscar              4     1.400 1141.0 -10222.9
## <none>                                  1139.6 -10222.5
## - weekday                   6     2.239 1141.8 -10222.4
## + second                    1     0.076 1139.5 -10220.9
## - log(share):duration       1     0.860 1140.5 -10219.8
## - title:iscar               1     0.953 1140.6 -10219.3
## - duration:iscar            1     1.015 1140.6 -10219.0
## - BGM:iscar                 1     1.452 1141.1 -10216.7
## + weekday:iscar             6     0.817 1138.8 -10214.9
## - author:iscar              3     2.670 1142.3 -10214.1
## - month:iscar               4    17.585 1157.2 -10136.6
## - log(share):iscar          1    54.800 1194.4  -9936.5
## - log(comment):log(share)   1    66.168 1205.8  -9878.4
## - log(comment):iscar        1   112.402 1252.0  -9647.8
## - type                      5   289.380 1429.0  -8845.1
##
## Step:  AIC=-10222.94
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + period + author:iscar +
##     log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
##     title:iscar + month:iscar + log(comment):log(share) + log(share):duration
##
##                            Df Sum of Sq    RSS      AIC
## - weekday                   6     2.199 1143.2 -10223.1
## <none>                                  1141.0 -10222.9
```

```
## + period:iscar             4     1.400 1139.6 -10222.5
## - period                   4     1.759 1142.8 -10221.5
## + second                   1     0.075 1140.9 -10221.3
## - log(share):duration      1     0.857 1141.9 -10220.3
## - title:iscar              1     0.915 1141.9 -10220.0
## - duration:iscar           1     1.223 1142.2 -10218.4
## - BGM:iscar                1     1.463 1142.5 -10217.1
## + weekday:iscar            6     0.835 1140.2 -10215.4
## - author:iscar             3     2.839 1143.8 -10213.7
## - month:iscar              4    17.470 1158.5 -10137.8
## - log(share):iscar         1    56.564 1197.6  -9928.3
## - log(comment):log(share)  1    66.520 1207.5  -9877.5
## - log(comment):iscar       1   112.565 1253.6  -9648.1
## - type                     5   289.047 1430.0  -8848.6
##
## Step:  AIC=-10223.14
## log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + period + author:iscar + log(comment):iscar +
##     log(share):iscar + BGM:iscar + duration:iscar + title:iscar +
##     month:iscar + log(comment):log(share) + log(share):duration
##
##                           Df Sum of Sq    RSS      AIC
## <none>                                 1143.2 -10223.1
## + weekday                  6     2.199 1141.0 -10222.9
## + period:iscar             4     1.360 1141.8 -10222.4
## - period                   4     1.783 1145.0 -10221.6
## + second                   1     0.060 1143.2 -10221.5
## - log(share):duration      1     0.788 1144.0 -10220.9
## - title:iscar              1     0.869 1144.1 -10220.5
## - duration:iscar           1     1.257 1144.5 -10218.4
## - BGM:iscar                1     1.405 1144.6 -10217.6
## - author:iscar             3     2.826 1146.0 -10214.0
## - month:iscar              4    17.665 1160.9 -10137.1
## - log(share):iscar         1    57.010 1200.2  -9926.8
## - log(comment):log(share)  1    66.699 1209.9  -9877.5
## - log(comment):iscar       1   113.182 1256.4  -9646.3
## - type                     5   291.367 1434.6  -8841.2
```

Use anova() to verify

```
anova(fit5, fit6)
```

```
## Analysis of Variance Table
##
## Model 1: log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + weekday + second + period + iscar:(author +
##     log(comment) + log(share) + BGM + duration + title + month +
##     weekday + second + period) + log(comment):log(share) + log(share):duration
## Model 2: log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + period + author:iscar + log(comment):iscar +
##     log(share):iscar + BGM:iscar + duration:iscar + title:iscar +
##     month:iscar + log(comment):log(share) + log(share):duration
##   Res.Df    RSS  Df Sum of Sq      F Pr(>F)
```

```
## 1   6076 1138.7
## 2   6094 1143.2 -18   -4.4983 1.3335 0.1555
```

The p-values is 0.1555, which suggests that there is no significant effect to add on more predictors. So we choose fit6.

Comparison: (fit5,fit6)

```
library(stargazer)

tp <-
  c(
    "author",
    "BGM",
    "comment",
    "share",
    "duration",
    "type",
    "title",
    "quarter",
    "month",
    "weekday",
    "second",
    "period"
  )

stargazer(fit5,
          fit6,
          omit = tp,
          omit.lables = tp,
          type = "text")
```

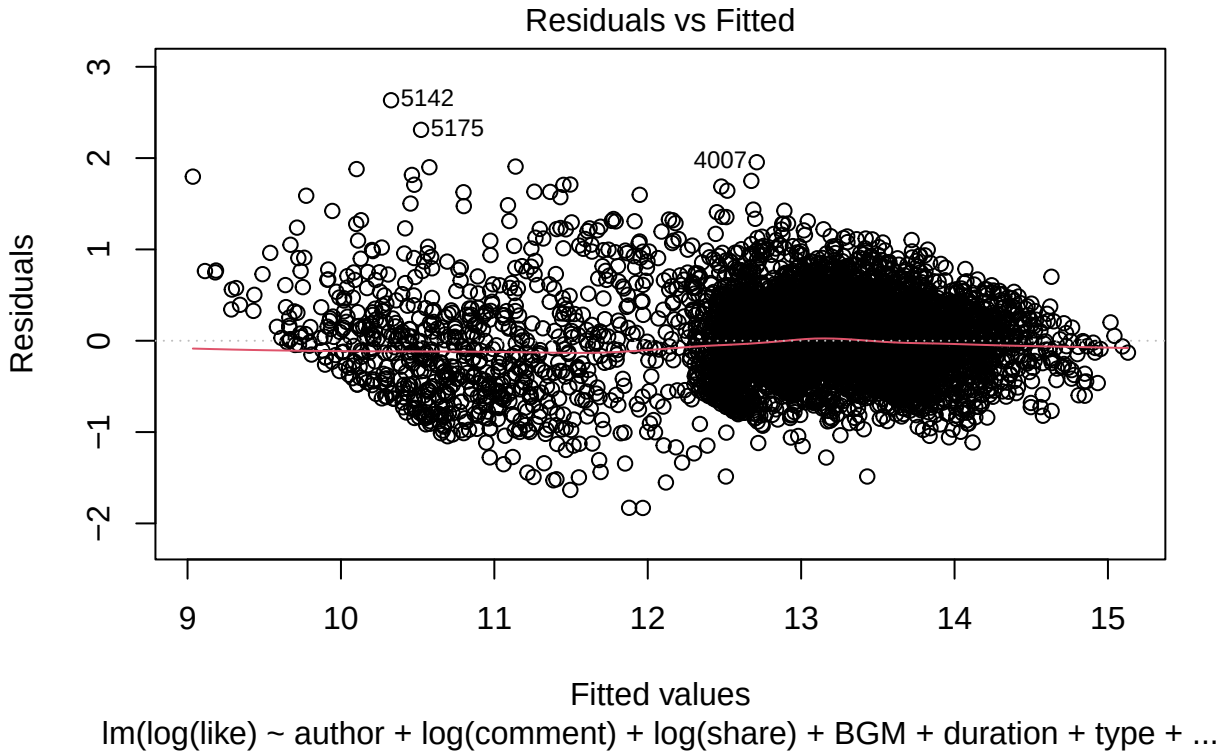
```
##
## ===============================================================================
##                                        Dependent variable:
##                      ---------------------------------------------------------
##                                             log(like)
##                                  (1)                          (2)
## -------------------------------------------------------------------------------
## Constant                       8.512***                     8.563***
##                                (0.208)                      (0.204)
##
## -------------------------------------------------------------------------------
## Observations                    6,131                        6,131
## R2                              0.830                        0.830
## Adjusted R2                     0.829                        0.829
## Residual Std. Error      0.433 (df = 6076)            0.433 (df = 6094)
## F Statistic          550.929*** (df = 54; 6076) 824.914*** (df = 36; 6094)
## ===============================================================================
## Note:                                            *p<0.1; **p<0.05; ***p<0.01
##
## ================================================================================
## author BGM comment share duration type title quarter month weekday second period
## --------------------------------------------------------------------------------
```

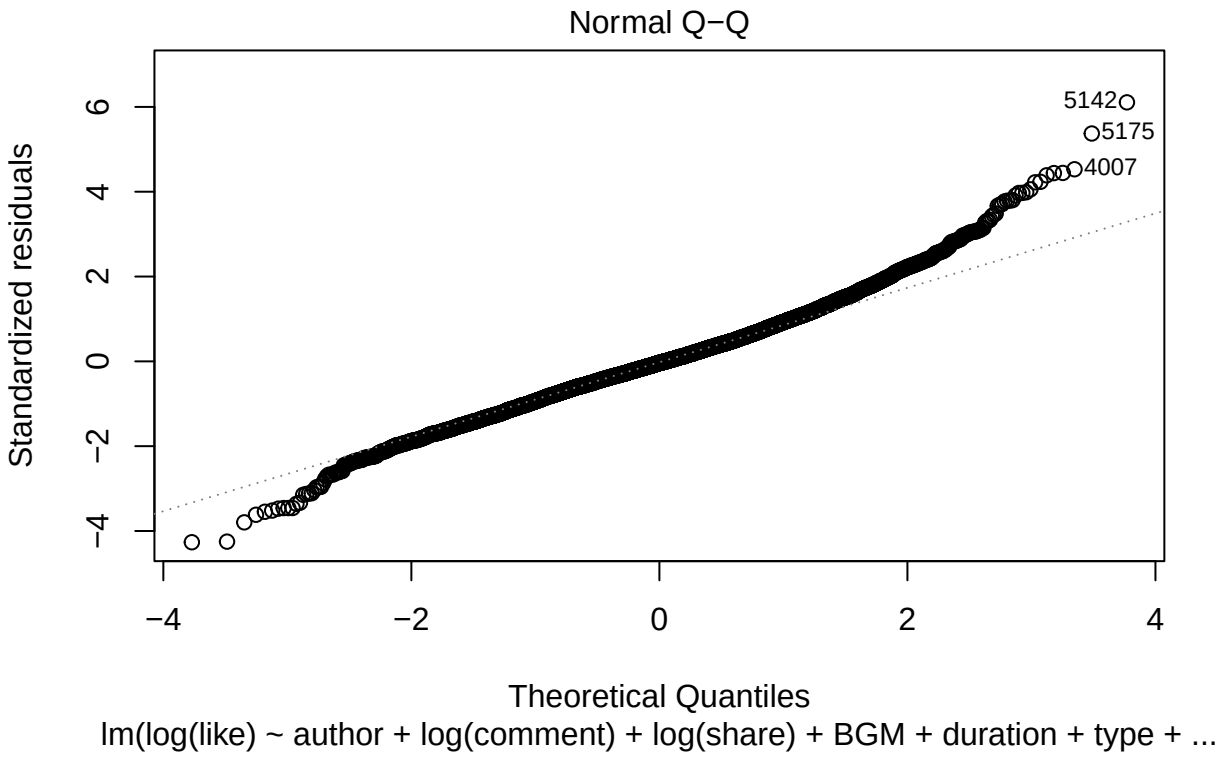
The number of coefficients decreased significantly.

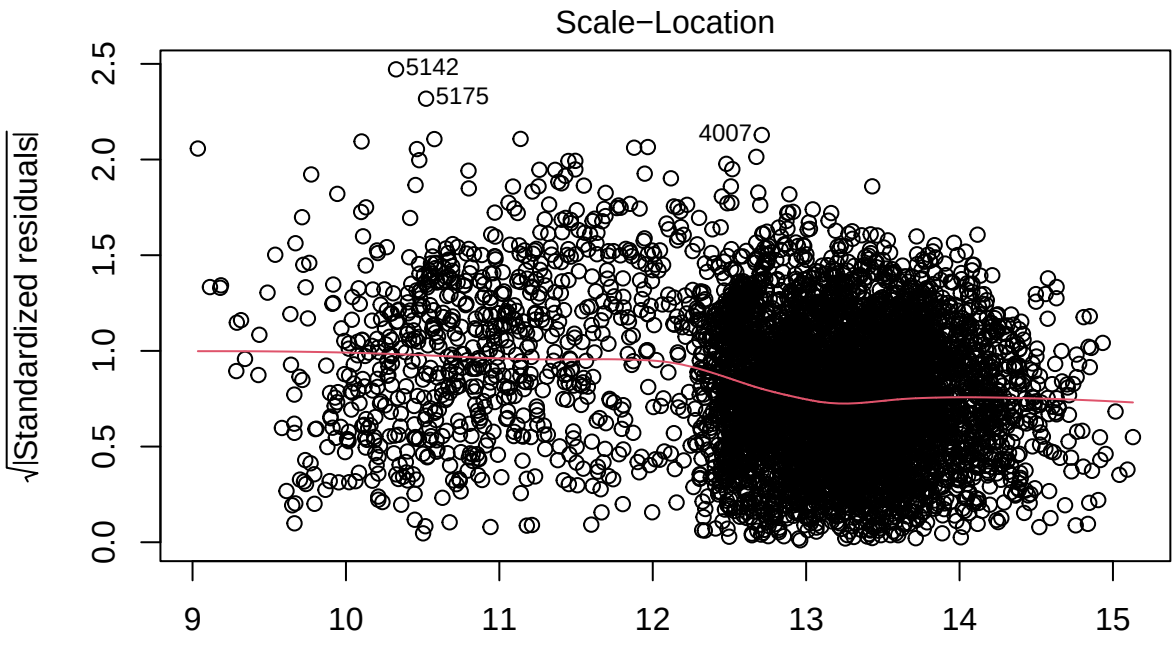
2.7 fit7 (model diagnostics for fit6)

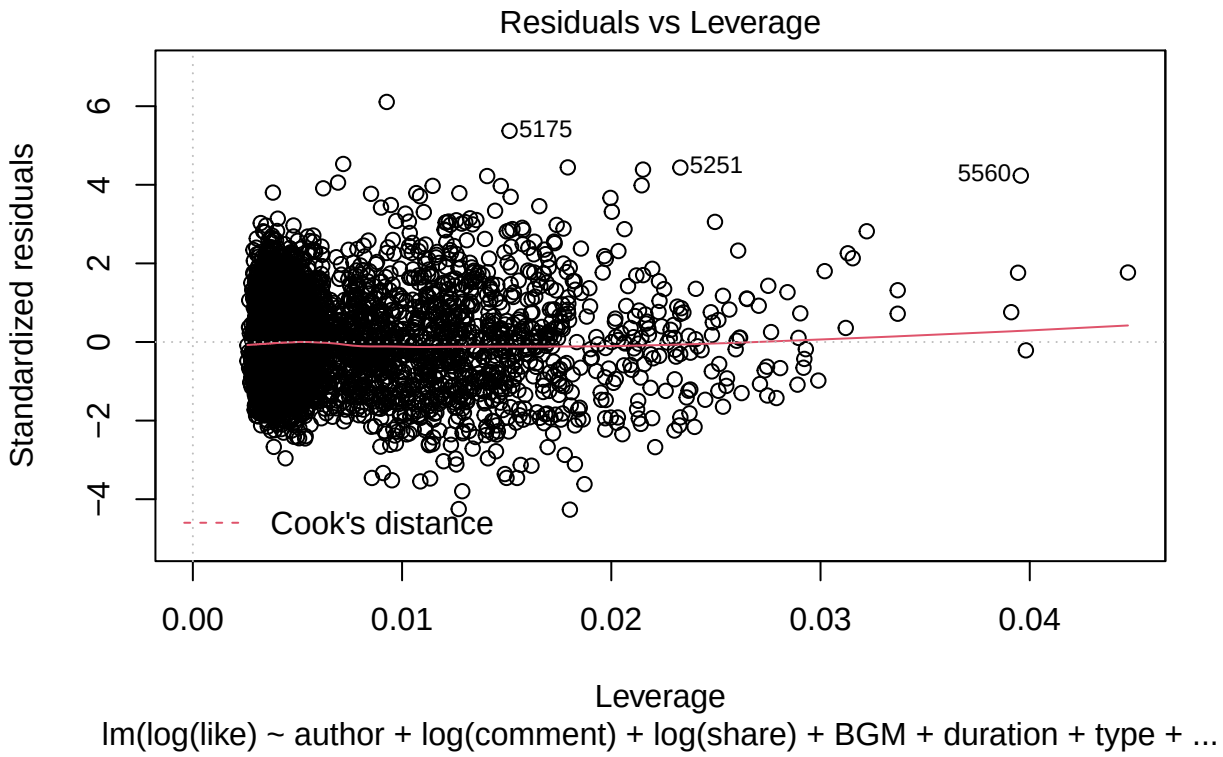
2.7.1 Basic assumptions

```
fit <- fit6
plot(fit)
```



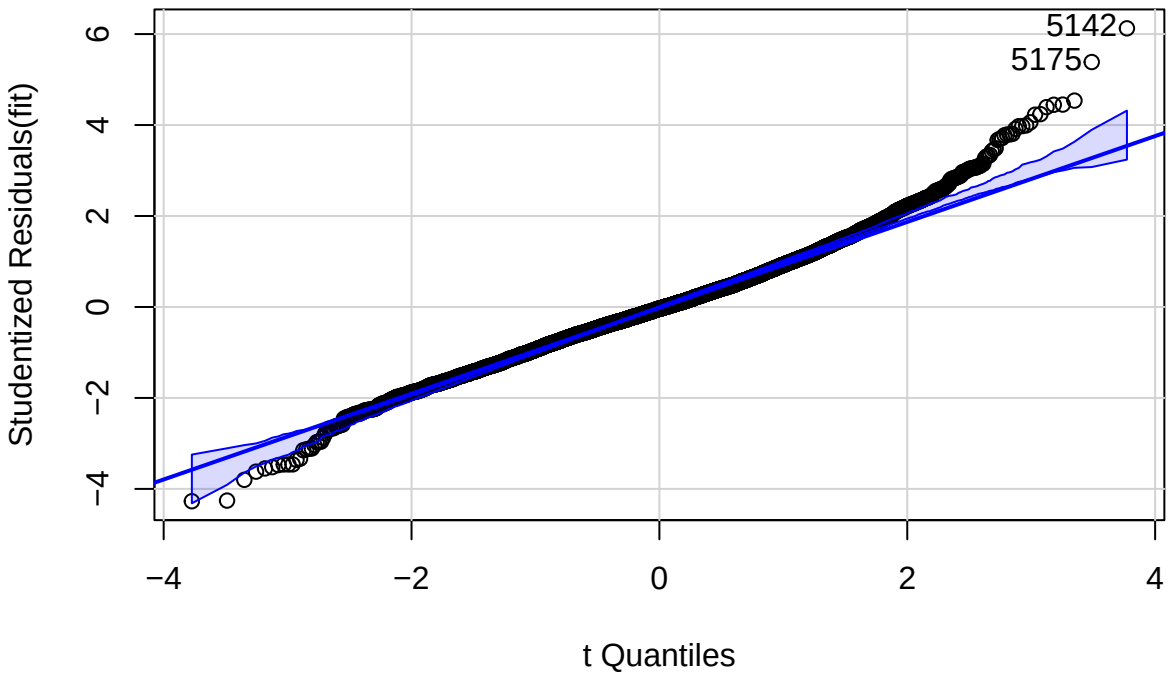

Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(log(like) ~ author + log(comment) + log(share) + BGM + duration + type + ...
5142
5175
4007
−4
−2
0
2
4
6


Scale−Location
√|Standardized residuals|
0.0
0.5
1.0
1.5
2.0
2.5
5142
5175
4007
9
10
11
12
13
14
15
Fitted values
lm(log(like) ~ author + log(comment) + log(share) + BGM + duration + type + ...



- Test Normality

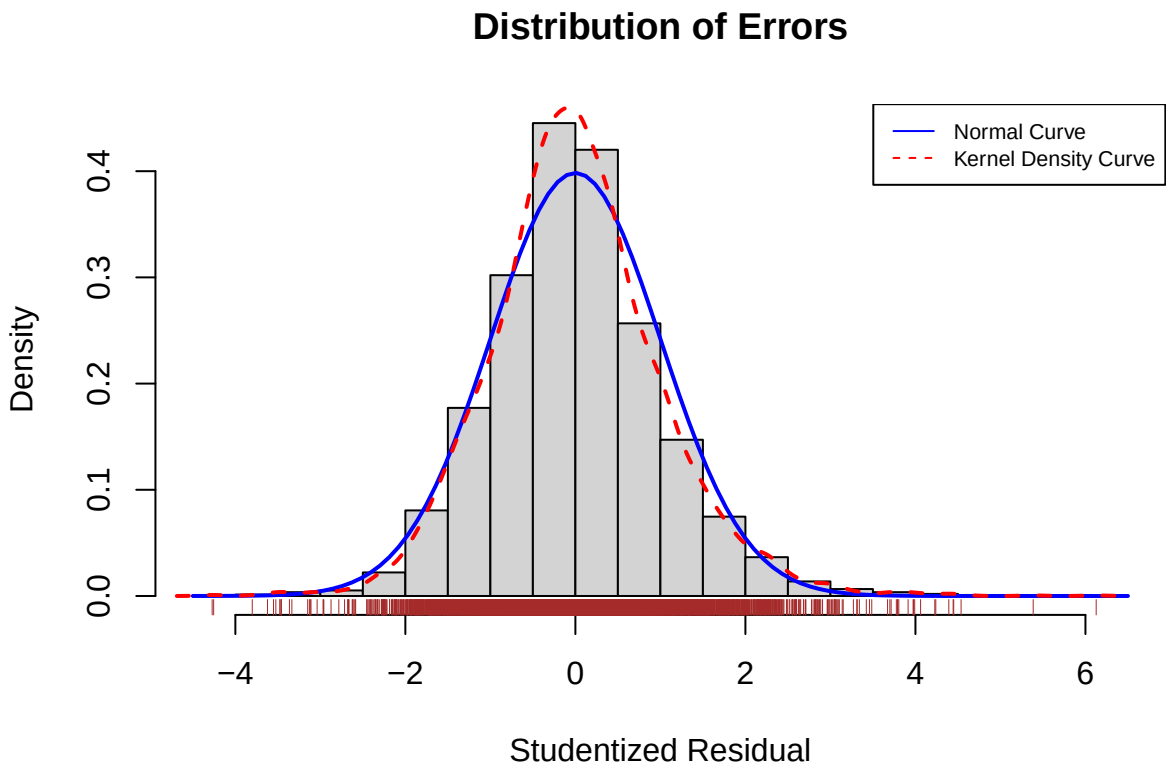
```
qqPlot(fit)
```




```
## 5142 5175
## 4914 4947
```

```
residplot <- function(fit, nbreaks = 20) {
  z <- rstudent(fit)
  hist(
    z,
    breaks = nbreaks,
    freq = FALSE,
    xlab = "Studentized Residual",
    main = "Distribution of Errors"
  )
  rug(jitter(z), col = "brown")
  curve(
    dnorm(x, mean = mean(z), sd = sd(z)),
    add = TRUE,
    col = "blue",
    lwd = 2
  )
  lines(
    density(z)$x,
    density(z)$y,
    col = "red",
    lwd = 2,
    lty = 2
```

```
  )
  legend(
    "topright",
    legend = c("Normal Curve", "Kernel Density Curve"),
    lty = 1:2,
    col = c("blue", "red"),
    cex = .7
  )
}
residplot(fit)
```




```
tp <- fit$residuals
ks.test(tp, "pnorm", mean = mean(tp), sd = sd(tp))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  tp
## D = 0.038621, p-value = 2.279e-08
## alternative hypothesis: two-sided
```

From the plot, the normality assumption seems to be acceptable. However, the test suggests that it is violated. This will be handled by the bootstrap method later.

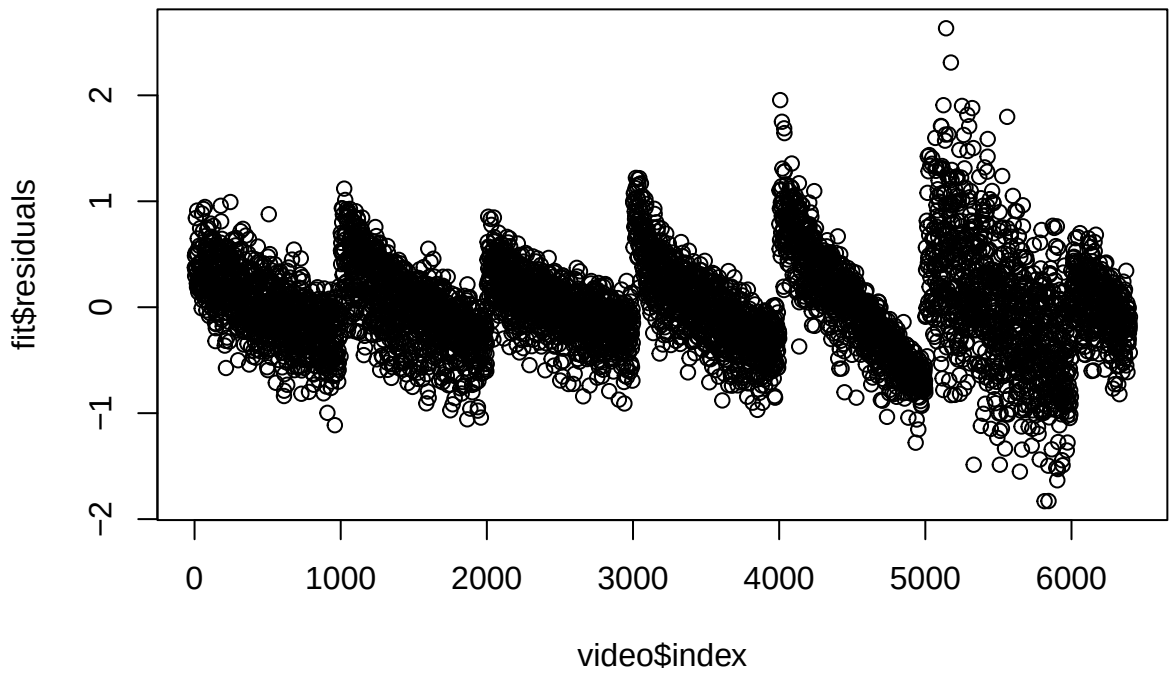
- Test Independence of Errors

```
durbinWatsonTest(fit)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.4882218      1.023426       0
##  Alternative hypothesis: rho != 0
```

The p-value is 0, it seems that the errors are not independent. But the reason may be that the data set is sorted by type. We can see the residual and index plot.

```
plot(video$index, fit$residuals)
```



We think this is the reason leading to the p-value being 0 in the independence test. So let's randomly reorder the index and do the test again.

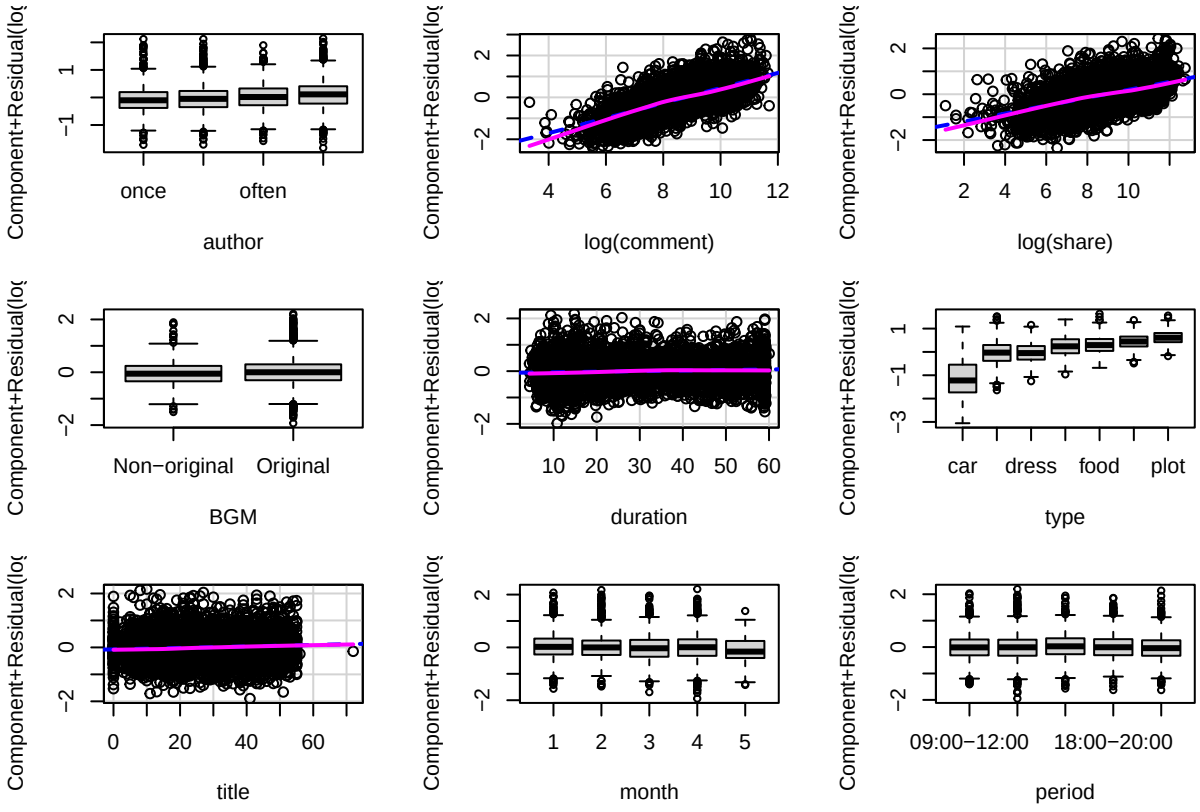
```
set.seed(1234)
rand <- video[sample(nrow(video), nrow(video)), ]

fit_rand <- lm(fit, data = rand)

durbinWatsonTest(fit_rand)
```

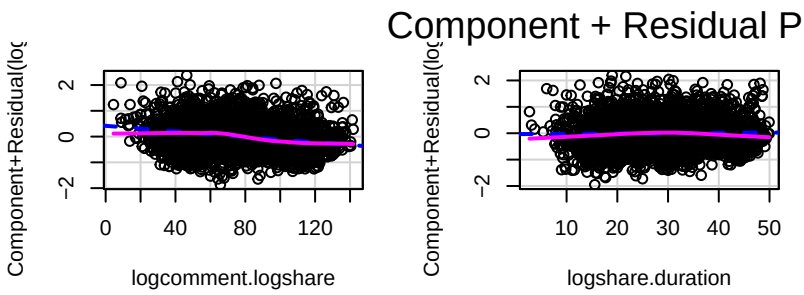
```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.0229123      1.953387   0.086
##  Alternative hypothesis: rho != 0
```

The p-value suggests that the errors are independent.

- Test Linearity

```
temp <- video
temp$logcomment.logshare <- log(video$comment) * log(video$share)
temp$logshare.duration <- log(video$share) * log(video$duration)

tpfit <- lm(
  log(like) ~ author + log(comment) + log(share) + BGM + duration + type +
    title + month + period + logcomment.logshare + logshare.duration,
  data = temp
)

crPlots(tpfit)
```





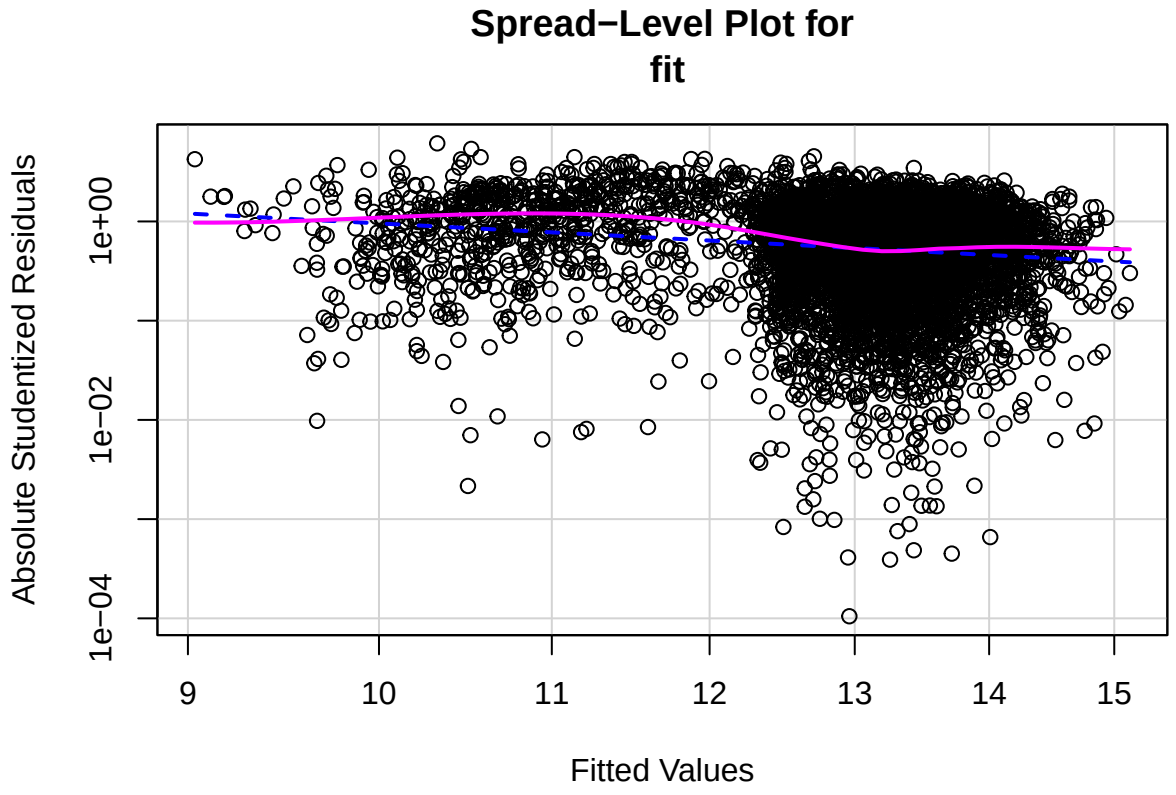
The component plus residual plots confirm that we have met the linearity assumption.

- Test Homoscedasticity

```
ncvTest(fit)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 646.1316, Df = 1, p = < 2.22e-16
```

```
spreadLevelPlot(fit)
```

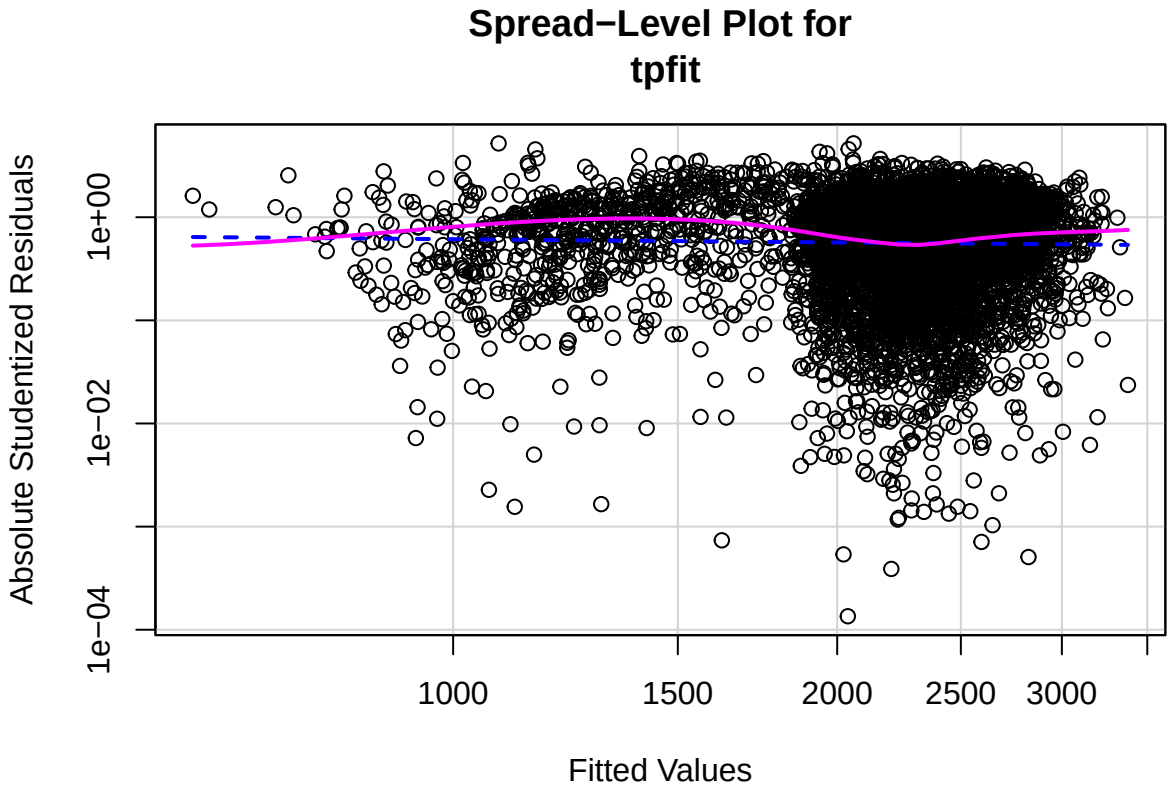



```
##
## Suggested power transformation:  3.178884
```

```
## use suggested power transformation
tpfit <- lm(
  log(like)^3 ~ author + log(comment) + log(share) +
    BGM + duration + type + title + month + period + author:iscar +
    log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
    title:iscar + month:iscar + log(comment):log(share) + log(share):duration,
  data = video
)

ncvTest(tpfit)
```

```
## Non-constant Variance Score Test
## Variance formula: ~ fitted.values
## Chisquare = 39.22436, Df = 1, p = 3.778e-10
```

```
spreadLevelPlot(tpfit)
```




```
##
## Suggested power transformation:  1.103794
```

No matter how the data is transformed, the problem of heteroscedasticity exists. We think the heteroscedasticity is caused by the categorical variable 'type' and the linear model cannot handle the problem.

- Test Multicollinearity

```
library(car)

## NA in 'author:iscar'   delete this term
fit <- lm(
  log(like) ~ author + log(comment) + log(share) +
    BGM + duration + type + title + month + period +
    log(comment):iscar + log(share):iscar + BGM:iscar + duration:iscar +
    title:iscar + month:iscar + log(comment):log(share) + log(share):duration,
  data = video
)
vif(fit)
```

```
##                             GVIF Df GVIF^(1/(2*Df))
## author                  1.414397  3        1.059486
## log(comment)           34.638158  1        5.885419
## log(share)             63.970439  1        7.998152
```

```
## BGM                       1.312721  1        1.145740
## duration                 40.283032  1        6.346892
## type                    127.047246  6        1.497375
## title                     1.295379  1        1.138147
## month                     2.296257  4        1.109501
## period                    1.108753  4        1.012988
## log(comment):iscar       56.139516  1        7.492631
## log(share):iscar         30.466217  1        5.519621
## BGM:iscar                 7.759016  1        2.785501
## duration:iscar            4.845321  1        2.201209
## title:iscar               4.926617  1        2.219598
## month:iscar              10.864195  4        1.347410
## log(comment):log(share) 129.776983  1       11.391970
## log(share):duration      44.334897  1        6.658446
```

Reference:

Fox & Georges Monette (1992) Generalized Collinearity Diagnostics, Journal of the American Statistical Association, 87:417, 178-183, DOI: 10.1080/01621459.1992.10475190

O'brien, R. M. (2007). A caution regarding rules of thumb for variance inflation factors. Quality & quantity, 41(5), 673-690.

Based on these references, we think there exists multicollinearity between 'log(comment):log(share)' and other variables, so we delete this term and calculate the GVIF again.

```
fit <- lm(
  log(like) ~ author + log(comment) + log(share) + BGM + duration + type +
    title + month + period + log(comment):iscar + log(share):iscar +
    BGM:iscar + duration:iscar + title:iscar + month:iscar + log(share):duration,
  data = video
)

vif(fit)
```

```
##                          GVIF Df GVIF^(1/(2*Df))
## author                1.411668  3        1.059145
## log(comment)          2.337807  1        1.528989
## log(share)            5.765688  1        2.401185
## BGM                   1.312711  1        1.145736
## duration             40.039367  1        6.327667
## type                 93.435884  6        1.459518
## title                 1.295246  1        1.138089
## month                 2.291395  4        1.109207
## period                1.108498  4        1.012959
## log(comment):iscar   51.564868  1        7.180868
## log(share):iscar     26.570996  1        5.154706
## BGM:iscar             7.756006  1        2.784961
## duration:iscar        4.832885  1        2.198382
## title:iscar           4.926557  1        2.219585
## month:iscar          10.834822  4        1.346954
## log(share):duration 43.954612  1        6.629827
```

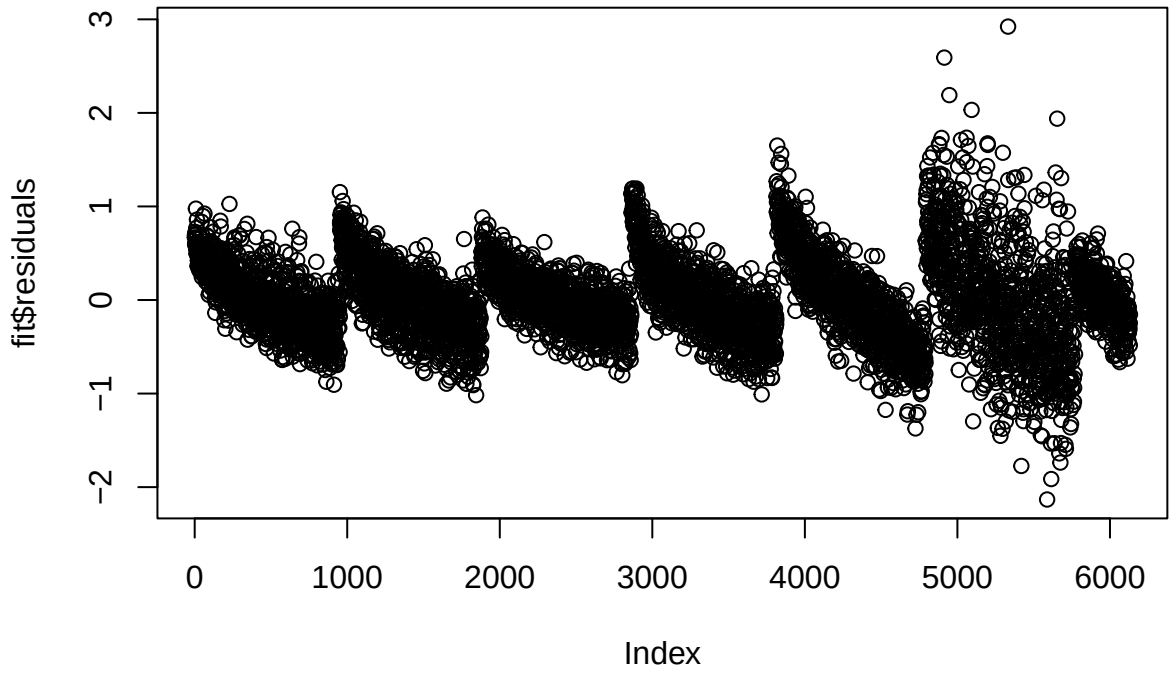
Now, it seems that there is no multicollinearity.

In the previous introduction of interaction, 'log(comment) and log(share)' have a strong correlation, and R^2 increased by 1% after the model was added to this item. But due to the collinearity problem, we must delete this variable.

2.7.2 Unusual Observations

- Outliers

```
plot(fit$residuals)
```




```
outlierTest(fit)
```

```
##       rstudent unadjusted p-value Bonferroni p
## 5560  6.636533         3.4884e-11   2.1388e-07
## 5142  5.842781         5.3988e-09   3.3100e-05
## 5175  4.949308         7.6479e-07   4.6889e-03
## 5816 -4.819020         1.4776e-06   9.0591e-03
## 5321  4.606808         4.1721e-06   2.5579e-02
```

```
## delete outliers
out_ind <-
  c(
    which(video$index == 5560),
    which(video$index == 5142),
```

```
    which(video$index == 5175),
    which(video$index == 5816),
    which(video$index == 5321)
  )
fit <- lm(fit, data = video[-out_ind, ])
outlierTest(fit)
```

```
##      rstudent unadjusted p-value Bonferroni p
## 5882 4.631242         3.7103e-06     0.022729
```

```
## delete outliers
out_ind <-
  c(out_ind, which(video$index == 5882))
fit <- lm(fit, data = video[-out_ind, ])
outlierTest(fit)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##       rstudent unadjusted p-value Bonferroni p
## 5843 -4.371407          1.255e-05      0.07687
```

```
## update data set
video_filted <- video[-out_ind,]
```

There are 6 outliers and we delete them.

Analyze these 6 outliers

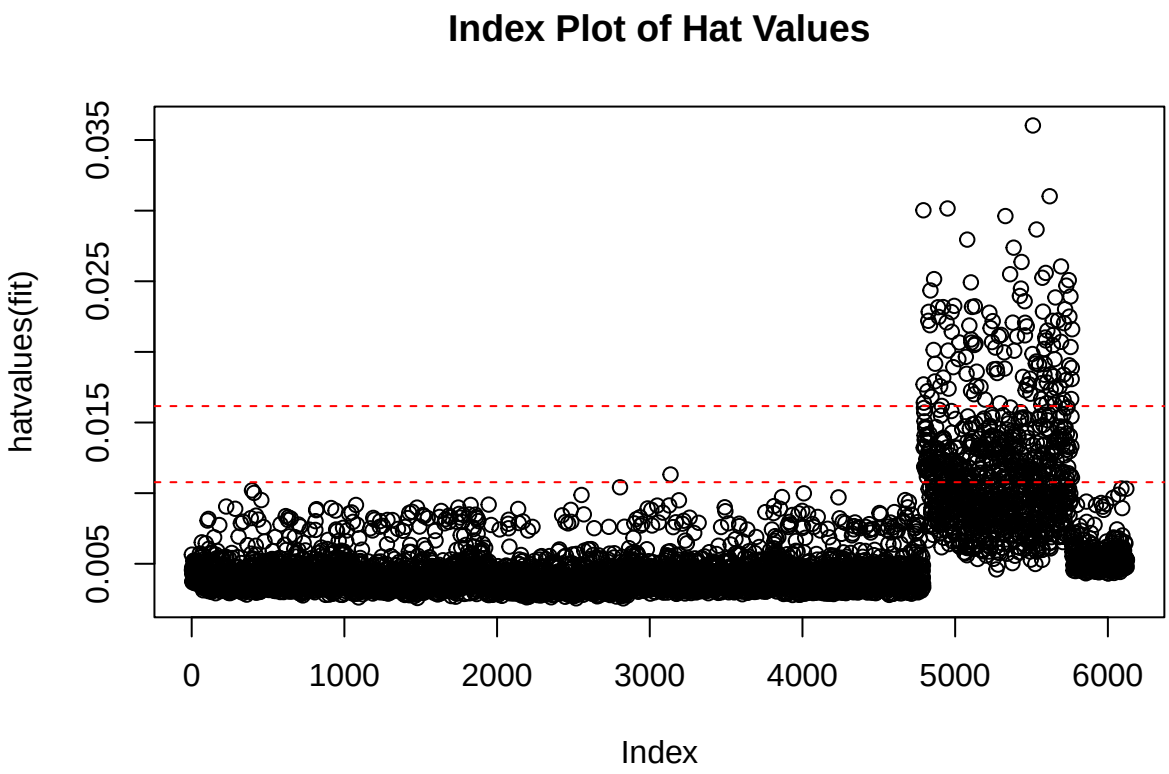
```
video[out_ind, -c(10:18,20)]
```

```
##      index   author   like comment share          BGM duration type title
## 5560  5560  several  50579      28    14     Original    10.00  car    39
## 5142  5142     once 424929     729  1152     Original    14.79  car    10
## 5175  5175     once 373832    2574   189 Non-original    22.30  car    24
## 5816  5816 frequent  25269   23508   686     Original    10.83  car    41
## 5321  5321     once 159732     678    24     Original    10.00  car    54
## 5882  5882  several  20523      60     3     Original    11.56  car    30
##      author_index
## 5560         2390
## 5142         2214
## 5175         2223
## 5816         2185
## 5321         2272
## 5882         2431
```

Control comment: 5560, 5142, 5175, 5321, 5882. Water army: 5816. All of these 6 outliers are type car.

- High Leverage Points

```r
hat.plot <- function(fit) {
  p <- length(coefficients(fit))
  n <- length(fitted(fit))
  plot(hatvalues(fit), main = "Index Plot of Hat Values")
  abline(h = c(2, 3) * p / n,
         col = "red",
         lty = 2)
  identify(1:n, hatvalues(fit), names(hatvalues(fit)))
}

hat.plot(fit)
```




```
## integer(0)
```

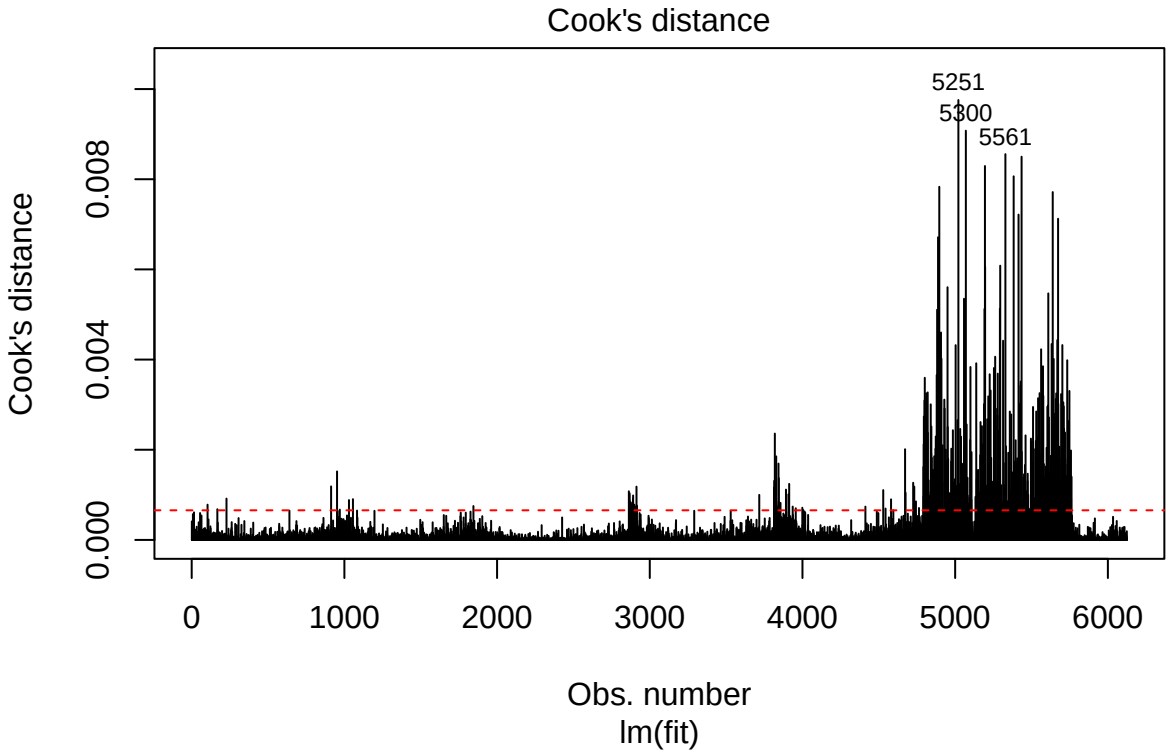
Previous result: index 5001-6000 — type car

The mean levels of hat values that exceed the red line corresponding to different types of videos are significantly different. Besides, we can clearly find that the hat values corresponding to the car type of video are significantly different from other types.

Obviously, most of the videos with car type are high leverage points. This can also be seen from the scatter plot.

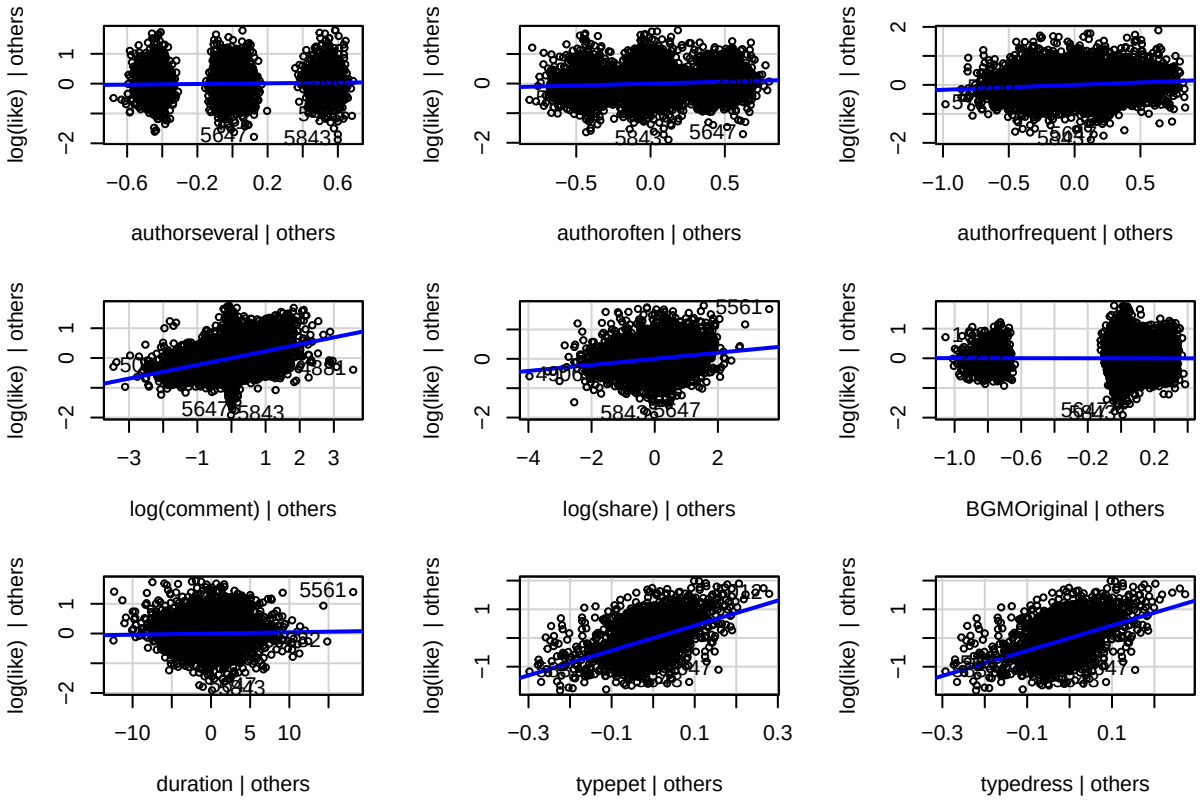
- Influential Observations

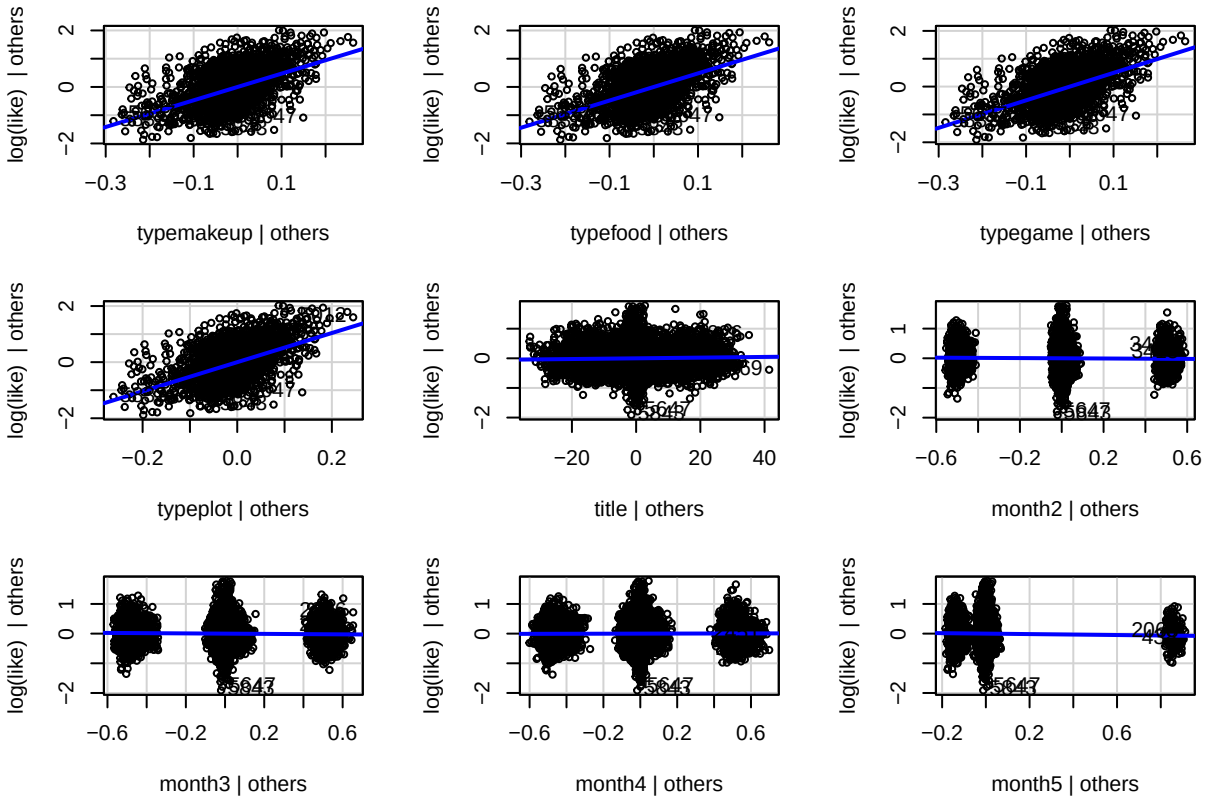
```
cutoff <- 4 / (nrow(video_filted) - length(fit$coefficients) - 2)
plot(fit, which = 4, cook.levels = cutoff)
abline(h = cutoff, lty = 2, col = "red")
```

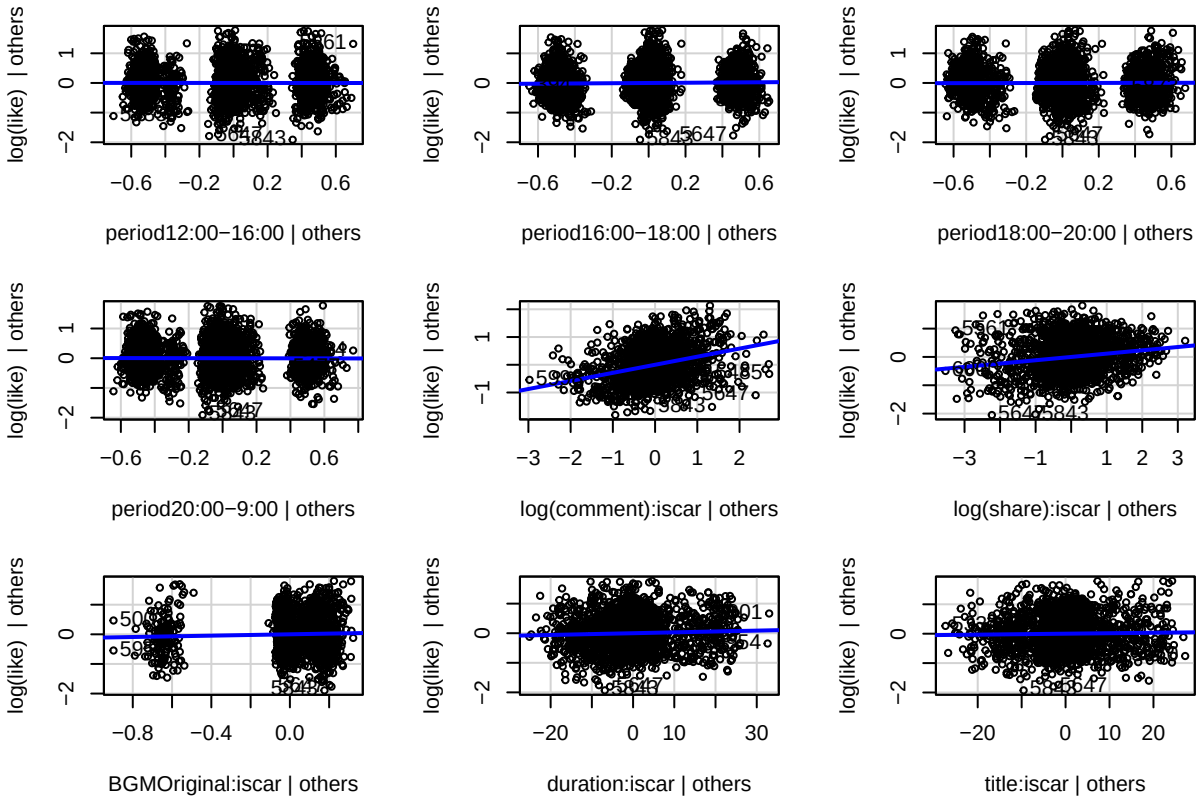


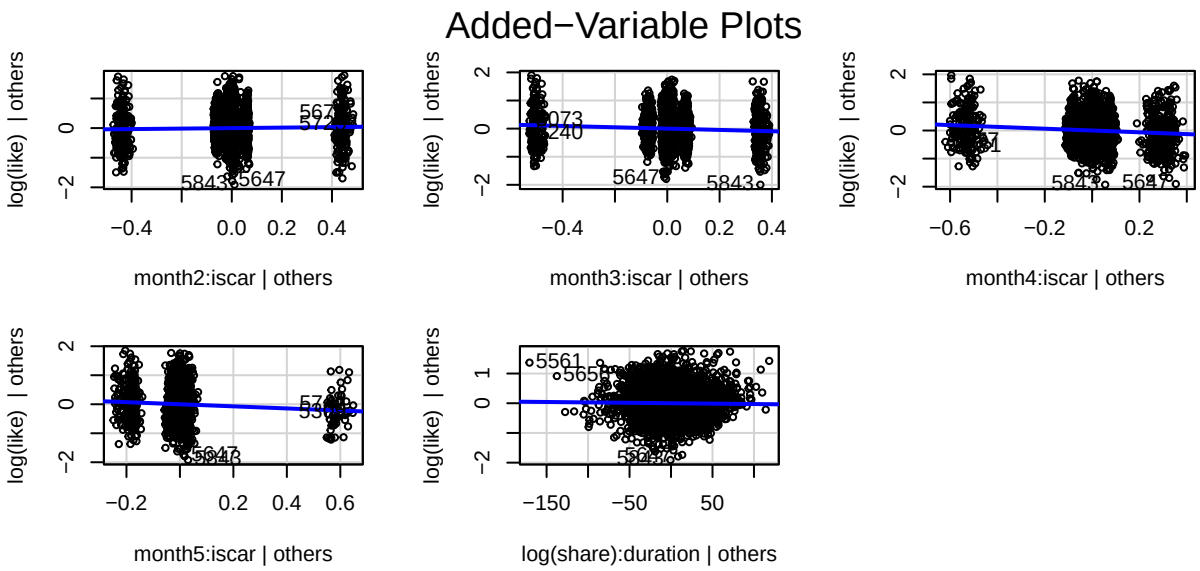
Similarly, most of the videos with car type are influential observations. This can also be seen from the scatter plot.

```
library(car)
avPlots(fit, ask = FALSE, id.method = "identify")
```


log(like) | others
authorseveral | others
authoroften | others
authorfrequent | others
log(comment) | others
log(share) | others
BGMOriginal | others
duration | others
typepet | others
typedress | others

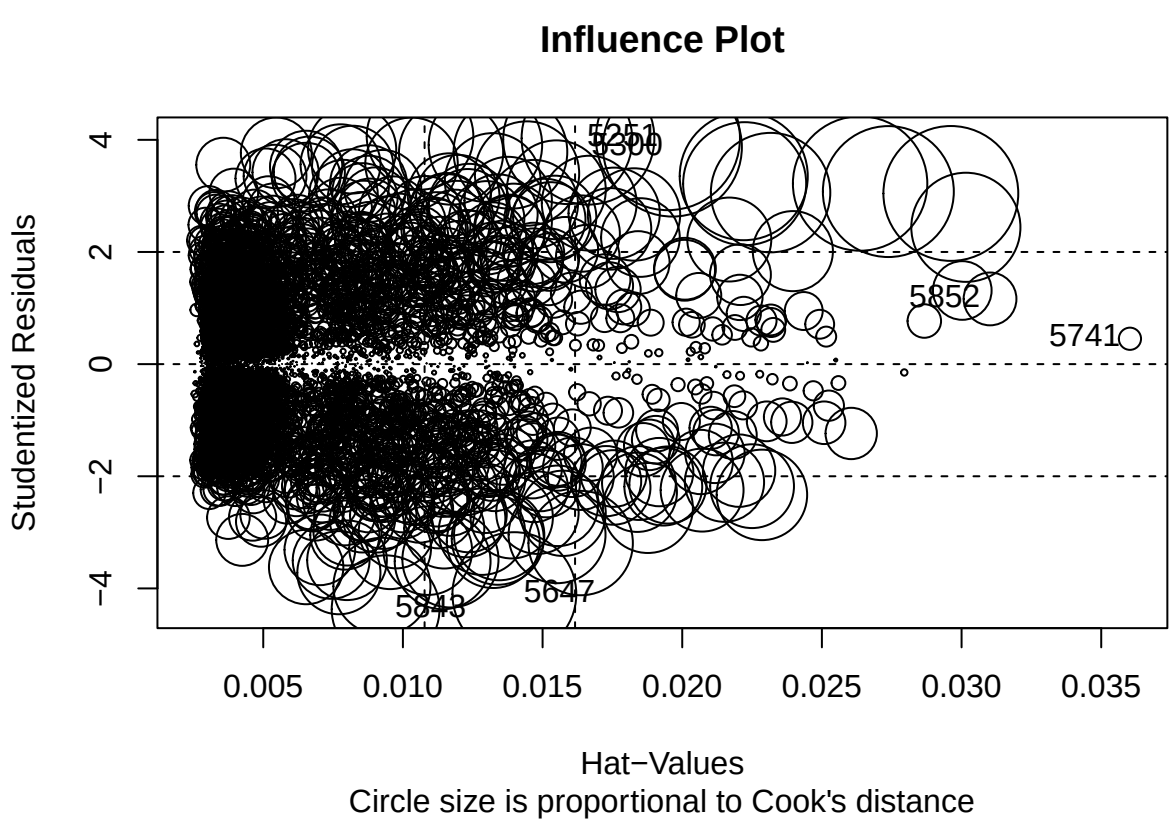

log(like) | others
typemakeup | others
typefood | others
typegame | others
typeplot | others
title | others
month2 | others
month3 | others
month4 | others
month5 | others


log(like) | others
period12:00−16:00 | others
period16:00−18:00 | others
period18:00−20:00 | others
period20:00−9:00 | others
log(comment):iscar | others
log(share):iscar | others
BGMOriginal:iscar | others
duration:iscar | others
title:iscar | others



Combination

```
influencePlot(fit, id.method = "identify", main = "Influence Plot",
              sub="Circle size is proportional to Cook's distance")
```




```
##         StudRes         Hat        CookD
## 5251  4.0297723 0.019489526 0.0097568832
## 5300  3.8702514 0.019648460 0.0090764432
## 5647 -4.1017174 0.013990540 0.0072151324
## 5741  0.4523333 0.036028580 0.0002317624
## 5843 -4.3714067 0.009380477 0.0054671056
## 5852  1.1626952 0.031019422 0.0013113281
```

```
fit7 <- fit
summary(fit7)
```

```
##
## Call:
## lm(formula = fit, data = video[-out_ind, ])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.91236 -0.27828 -0.02089  0.26333  1.77624
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)          5.4479253  0.1080809  50.406  < 2e-16 ***
## authorseveral        0.0561276  0.0161751   3.470 0.000524 ***
## authoroften          0.1315636  0.0162727   8.085 7.44e-16 ***
## authorfrequent       0.1696917  0.0178447   9.509  < 2e-16 ***
```

```
## log(comment)          0.2304517  0.0072415  31.824  < 2e-16 ***
## log(share)            0.1034664  0.0081409  12.709  < 2e-16 ***
## BGMOriginal          -0.0053372  0.0185570  -0.288 0.773656
## duration              0.0040684  0.0020313   2.003 0.045237 *
## typepet               4.3480376  0.1161446  37.436  < 2e-16 ***
## typedress             4.3991388  0.1172019  37.535  < 2e-16 ***
## typemakeup            4.7245895  0.1191876  39.640  < 2e-16 ***
## typefood              4.8151574  0.1196077  40.258  < 2e-16 ***
## typegame              4.9370893  0.1190137  41.483  < 2e-16 ***
## typeplot              5.1965843  0.1236874  42.014  < 2e-16 ***
## title                 0.0011833  0.0004665   2.537 0.011215 *
## month2               -0.0339130  0.0172781  -1.963 0.049717 *
## month3               -0.0398872  0.0176789  -2.256 0.024093 *
## month4                0.0113857  0.0181161   0.628 0.529710
## month5               -0.0785620  0.0340947  -2.304 0.021243 *
## period12:00-16:00    -0.0026257  0.0177858  -0.148 0.882642
## period16:00-18:00     0.0398108  0.0179899   2.213 0.026938 *
## period18:00-20:00     0.0096695  0.0178812   0.541 0.588689
## period20:00-9:00     -0.0088191  0.0187388  -0.471 0.637918
## log(comment):iscar    0.2913273  0.0141689  20.561  < 2e-16 ***
## log(share):iscar      0.1168715  0.0101082  11.562  < 2e-16 ***
## BGMOriginal:iscar     0.1142736  0.0458258   2.494 0.012670 *
## duration:iscar        0.0029304  0.0010107   2.899 0.003752 **
## title:iscar           0.0015970  0.0010995   1.453 0.146413
## month2:iscar          0.0831535  0.0502799   1.654 0.098217 .
## month3:iscar         -0.2352353  0.0469505  -5.010 5.59e-07 ***
## month4:iscar         -0.3209012  0.0454976  -7.053 1.94e-12 ***
## month5:iscar         -0.3605778  0.0760205  -4.743 2.15e-06 ***
## log(share):duration -0.0002790  0.0002165  -1.289 0.197603
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4402 on 6092 degrees of freedom
## Multiple R-squared:  0.8236, Adjusted R-squared:  0.8227
## F-statistic: 888.8 on 32 and 6092 DF,  p-value: < 2.2e-16
```

2.8 final model

Now fit7 is very close to our final model. But fit7 has 32 coefficients and the model is too complicated. Therefore, in the last step, we try to delete some variables to make the model as simple as possible.

```
## delete 'log(share):duration'
finalfit1 <- lm(
  log(like) ~ author + log(comment) + log(share) + BGM + duration + type +
    title + month + period + log(comment):iscar + log(share):iscar +
    BGM:iscar + duration:iscar + month:iscar,
  data = video_filted
)
anova(finalfit1, fit)
```

```
## Analysis of Variance Table
##
## Model 1: log(like) ~ author + log(comment) + log(share) + BGM + duration +
```

```
##     type + title + month + period + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + month:iscar
## Model 2: log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + period + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + title:iscar + month:iscar +
##     log(share):duration
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1   6094 1181.1
## 2   6092 1180.4  2    0.6969 1.7983 0.1657
```

```
## delete 'BGM', 'period'
finalfit2 <- lm(
  log(like) ~ author + log(comment) + log(share) + duration + type +
    title + month + log(comment):iscar + log(share):iscar +
    BGM:iscar + duration:iscar + month:iscar,
  data = video_filted
)
anova(finalfit2, finalfit1)
```

```
## Analysis of Variance Table
##
## Model 1: log(like) ~ author + log(comment) + log(share) + duration + type +
##     title + month + log(comment):iscar + log(share):iscar + BGM:iscar +
##     duration:iscar + month:iscar
## Model 2: log(like) ~ author + log(comment) + log(share) + BGM + duration +
##     type + title + month + period + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + month:iscar
##   Res.Df    RSS Df Sum of Sq      F Pr(>F)
## 1   6099 1182.8
## 2   6094 1181.1  5    1.7158 1.7705 0.1153
```

It is reasonable to delete these features since the p-values are greater than 0.05.

```
finalfit <- finalfit2
summary(finalfit)
```

```
##
## Call:
## lm(formula = log(like) ~ author + log(comment) + log(share) +
##     duration + type + title + month + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + month:iscar, data = video_filted)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.93373 -0.27857 -0.02079  0.25991  1.78161
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         5.5425997  0.0930066  59.594  < 2e-16 ***
## authorseveral       0.0561062  0.0161535   3.473 0.000518 ***
## authoroften         0.1351116  0.0161595   8.361  < 2e-16 ***
## authorfrequent      0.1722682  0.0176477   9.762  < 2e-16 ***
## log(comment)        0.2288701  0.0071810  31.872  < 2e-16 ***
```

```
## log(share)          0.0956691  0.0049258  19.422  < 2e-16 ***
## duration            0.0015401  0.0004211   3.657 0.000257 ***
## typepet             4.3318744  0.1110835  38.997  < 2e-16 ***
## typedress           4.3835230  0.1121871  39.073  < 2e-16 ***
## typemakeup          4.7087887  0.1139618  41.319  < 2e-16 ***
## typefood            4.8017388  0.1145632  41.913  < 2e-16 ***
## typegame            4.9232855  0.1138714  43.235  < 2e-16 ***
## typeplot            5.1784563  0.1186612  43.641  < 2e-16 ***
## title               0.0014432  0.0004203   3.434 0.000600 ***
## month2             -0.0332211  0.0172580  -1.925 0.054280 .
## month3             -0.0405972  0.0176480  -2.300 0.021460 *
## month4              0.0109772  0.0180926   0.607 0.544056
## month5             -0.0791018  0.0340257  -2.325 0.020117 *
## log(comment):iscar  0.2921019  0.0141602  20.628  < 2e-16 ***
## log(share):iscar    0.1174509  0.0101026  11.626  < 2e-16 ***
## iscar:BGMOriginal   0.1057465  0.0417918   2.530 0.011421 *
## duration:iscar      0.0035465  0.0009544   3.716 0.000204 ***
## month2:iscar        0.0833615  0.0502515   1.659 0.097190 .
## month3:iscar       -0.2343296  0.0469170  -4.995 6.06e-07 ***
## month4:iscar       -0.3175441  0.0453502  -7.002 2.79e-12 ***
## month5:iscar       -0.3558016  0.0758462  -4.691 2.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4404 on 6099 degrees of freedom
## Multiple R-squared:  0.8232, Adjusted R-squared:  0.8225
## F-statistic:  1136 on 25 and 6099 DF,  p-value: < 2.2e-16
```

This is our final linear model exactly.

2.9 The overall modeling process

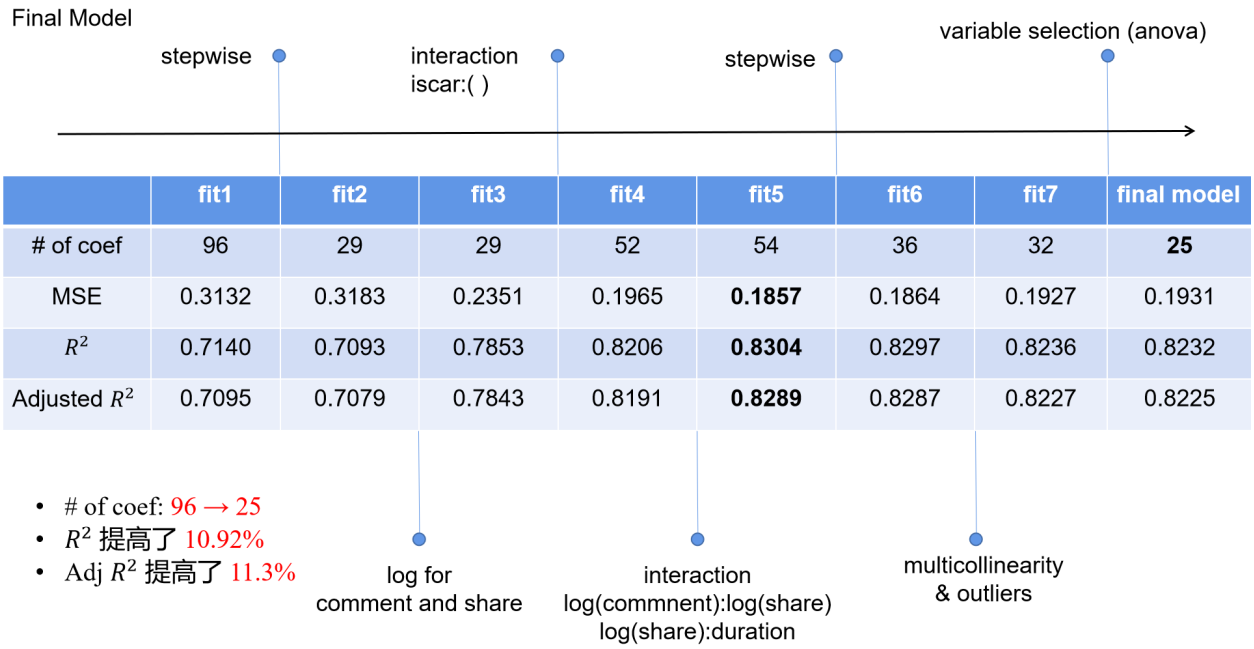
Comparison: (fit1, fit7, finalfit)

```
library(stargazer)

tp <-
  c(
    "author",
    "comment",
    "share",
    "BGM",
    "duration",
    "type",
    "title",
    "quarter",
    "month",
    "week",
    "weekday",
    "day",
    "hour",
    "minute",
    "second",
```

```
    "period"
  )

stargazer(
  fit1,
  fit7,
  finalfit,
  omit = tp,
  omit.lables = tp,
  type = "text"
)
```

```
##
## ================================================================================================
##                                              Dependent variable:
##                     ---------------------------------------------------------------------------
##                                                  log(like)
##                               (1)                    (2)                      (3)
## ------------------------------------------------------------------------------------------------
## Constant                   11.105***              5.448***                 5.543***
##                             (0.098)                (0.108)                  (0.093)
##
## ------------------------------------------------------------------------------------------------
## Observations                6,131                  6,125                    6,125
## R2                          0.714                  0.824                    0.823
## Adjusted R2                 0.709                  0.823                    0.823
## Residual Std. Error     0.564 (df = 6034)      0.440 (df = 6092)        0.440 (df = 6099)
## F Statistic         156.932*** (df = 96; 6034) 888.764*** (df = 32; 6092) 1,136.105*** (df = 25; 609
## ================================================================================================
## Note:                                                                *p<0.1; **p<0.05; ***p<0.
##
## ================================================================================================
## author comment share BGM duration type title quarter month week weekday day hour minute second perio
## ------------------------------------------------------------------------------------------------
```

From fit1 to finalfit, the number of coefficients dropped from 96 to 25, the R^2 increased by 10.92%, and the adjusted R^2 increased by 11.3%.



	fit1	fit2	fit3	fit4	fit5	fit6	fit7	final model
# of coef	96	29	29	52	54	36	32	**25**
MSE	0.3132	0.3183	0.2351	0.1965	**0.1857**	0.1864	0.1927	0.1931
R^2	0.7140	0.7093	0.7853	0.8206	**0.8304**	0.8297	0.8236	0.8232
Adjusted R^2	0.7095	0.7079	0.7843	0.8191	**0.8289**	0.8287	0.8227	0.8225

- # of coef: 96 → 25
- R^2 提高了 10.92%
- Adj R^2 提高了 11.3%

2.10 Cross validation & relative importance

2.10.1 cross validation

```
library(caret)
```

```
##      lattice
```

```
set.seed(1234)

train_control <- trainControl(method = "CV", number = 10)

model <-
  train(
    log(like) ~ author + log(comment) + log(share) + duration + type +
      title + month + log(comment):iscar + log(share):iscar +
      BGM:iscar + duration:iscar + month:iscar,
    data = video_filted,
    method = "lm",
    trControl = train_control
  )

print(model)
```

```
## Linear Regression
##
## 6125 samples
##    9 predictor
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5513, 5512, 5512, 5512, 5513, 5512, ...
## Resampling results:
##
##   RMSE      Rsquared   MAE
##   0.442026  0.8212553  0.3405349
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

Our linear model is robust since the performance persists. It has a strong generalization ability for short video data.

2.10.2 relative importance

```
relweights <- function(fit, ...) {
  R <- cor(fit$model)
  nvar <- ncol(R)
  rxx <- R[2:nvar, 2:nvar]
  rxy <- R[2:nvar, 1]
  svd <- eigen(rxx)
  evec <- svd$vectors
  ev <- svd$values
  delta <- diag(sqrt(ev))
  lambda <- evec %*% delta %*% t(evec)
  lambdasq <- lambda ^ 2
  beta <- solve(lambda) %*% rxy
  rsquare <- colSums(beta ^ 2)
  rawwgt <- lambdasq %*% beta ^ 2
  import <- (rawwgt / rsquare) * 100
  import <- as.data.frame(import)
  row.names(import) <- names(fit$model[2:nvar])
  names(import) <- "weights"
  import <- import[order(import), 1, drop = FALSE]
  dotchart(
    import$weights,
    labels = row.names(import),
    xlab = " % of R-Square",
    pch = 19,
    main = "Relative Importance of Predictor Variables",
    ...
  )
  return(import)
}
```

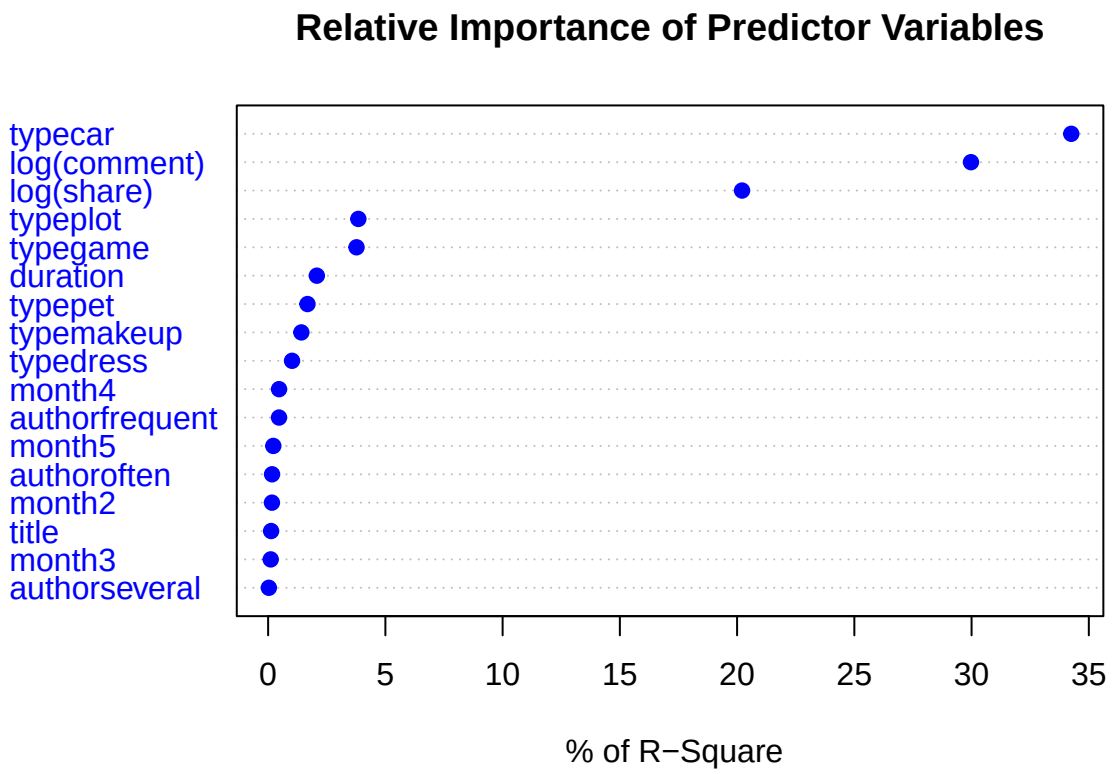
Introduce dummy variables corresponding to each categorical variable so that we can calculate its relative importance.

```
temp <- video_filted

##
```

```
temp$authorseveral <- 0
temp$authoroften <- 0
temp$authorfrequent <- 0

temp$typemakeup <- 0
temp$typegame <- 0
temp$typedress <- 0
temp$typepet <- 0
temp$typecar <- 0
temp$typeplot <- 0

temp$month2 <- 0
temp$month3 <- 0
temp$month4 <- 0
temp$month5 <- 0

##
temp$authorseveral[temp$author == "several"] <- 1
temp$authoroften[temp$author == "often"] <- 1
temp$authorfrequent[temp$author == "frequent"] <- 1

temp$typemakeup[temp$type == "makeup"] <- 1
temp$typegame[temp$type == "game"] <- 1
temp$typedress[temp$type == "dress"] <- 1
temp$typepet[temp$type == "pet"] <- 1
temp$typecar[temp$type == "car"] <- 1
temp$typeplot[temp$type == "plot"] <- 1

temp$month2[temp$month == 2] <- 1
temp$month3[temp$month == 3] <- 1
temp$month4[temp$month == 4] <- 1
temp$month5[temp$month == 5] <- 1
```

Relative importance

```
tpfit <-
  lm(
    log(like) ~ authorseveral + authoroften + authorfrequent + log(comment) +
      log(share) + duration + typemakeup + typegame + typedress + typepet +
      typecar + typeplot + title + month2 + month3 + month4 + month5,
    data = temp
  )

relweights(tpfit, col = "blue")
```




```
##                    weights
## authorseveral   0.03105729
## month3          0.11051268
## title           0.12743954
## month2          0.16307306
## authoroften     0.16778152
## month5          0.22035172
## authorfrequent  0.46529306
## month4          0.46642185
## typedress       1.02053921
## typemakeup      1.41696851
## typepet         1.67898373
## duration        2.07663335
## typegame        3.77013906
## typeplot        3.84745103
## log(share)     20.21180714
## log(comment)   29.97439092
## typecar        34.25115632
```

Factors influecning the number of likes: type > comment > share > duration > month, title, author

3 Categorical Data Analysis

Categorical variables: author, type, BGM, month, weekday and hour.

3.1 Independence: chi-squared independence test

H_0: the two categorical variables are independent.

- author

```
## author and type
table <- table(video$author, video$type)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 445.22, df = 18, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##            car pet dress makeup food game plot
##   once     238 206   341    182  301  374   18
##   several  213 211   251    187  239  243   53
##   often    219 243   242    296  236  163  135
##   frequent 311 318   117    263  175  203  153
```

```
round(chisq.test(table)$expected)
```

```
##
##            car pet dress makeup food game plot
##   once     266 265   257    251  257  266   97
##   several  224 223   217    211  217  224   82
##   often    245 245   238    232  238  246   90
##   frequent 246 246   239    233  239  247   90
```

'author' and 'type' are not independent.

```
## author and BGM
table <- table(video$author, video$BGM)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 243.17, df = 3, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##            Non-original Original
##   once               350     1310
##   several            252     1145
##   often              211     1323
##   frequent            46     1494
```

```
round(chisq.test(table)$expected)
```

```
##
##            Non-original Original
##   once               233     1427
##   several            196     1201
##   often              215     1319
##   frequent           216     1324
```

'author' and 'BGM' are not independent.

```
## author and month
table <- table(video$author, video$month)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 24.742, df = 12, p-value = 0.01609
```

```
chisq.test(table)$observed
```

```
##
##              1   2   3   4   5
##   once     445 406 359 387  63
##   several  350 340 318 337  52
##   often    336 366 395 379  58
##   frequent 352 364 396 350  78
```

```
round(chisq.test(table)$expected)
```

```
##
##              1   2   3   4  5
##   once     402 400 397 393 68
##   several  338 336 334 331 57
##   often    371 369 367 364 63
##   frequent 373 371 369 365 63
```

'author' and 'month' are not independent.

```
## author and weekday
table <- table(video$author, video$weekday)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 19.571, df = 18, p-value = 0.3575
```

```
chisq.test(table)$observed
```

```
##
##              1   2   3   4   5   6   7
##   once     227 207 248 225 234 250 269
##   several  194 181 180 184 205 217 236
##   often    202 199 206 245 187 250 245
##   frequent 213 201 192 242 215 255 222
```

```
round(chisq.test(table)$expected)
```

```
##
##              1   2   3   4   5   6   7
##   once     226 213 224 243 228 263 263
##   several  190 180 188 204 192 221 221
##   often    209 197 207 224 210 243 243
##   frequent 210 198 207 225 211 244 244
```

'author' and 'weekday' are independent.

```
## author and hour
table <- table(video$author, video$hour)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 301.36, df = 69, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##              0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
##   once      22  35   7  11   7  11  14  36  57  69  89 148 119  86  70  70 124
##   several   15  12   6   6   4   9   9  18  50  49  72 131 126  55  60  58  80
##   often     10   4   8   7   4   2   5  15  34  53  86 138 155  63  61  54  89
##   frequent  16  10   5   4   1   1   1  10  36  53 144 177 162  63  50  55  68
##
##              17  18  19  20  21  22  23
##   once      187 167 107  74  64  41  45
##   several   187 195  95  58  46  32  24
##   often     267 239 129  62  25  14  10
##   frequent  234 243 113  50  24  15   5
```

```
round(chisq.test(table)$expected)
```

```
##
##              0  1 2 3 4 5 6  7  8  9  10  11  12 13 14 15 16  17  18  19 20 21
##   once      17 17 7 8 4 6 8 21 48 61 106 161 152 72 65 64 98 237 229 120 66 43
##   several   14 14 6 6 4 5 7 18 40 51  89 135 128 61 55 54 82 199 192 101 56 36
##   often     16 15 7 7 4 6 7 20 44 56  98 149 141 67 60 59 90 219 211 111 61 40
##   frequent  16 15 7 7 4 6 7 20 44 56  98 149 141 67 61 60 91 220 212 112 61 40
##
##              22 23
##   once       28 23
##   several    23 19
##   often      26 21
##   frequent   26 21
```

'author' and 'hour' are not independent.

- type

```
## type and BGM
table <- table(video$type, video$BGM)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 181.15, df = 6, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##          Non-original Original
##   car             145      836
##   pet             153      825
##   dress           235      716
##   makeup           62      866
##   food            133      818
##   game            125      858
##   plot              6      353
```

```
round(chisq.test(table)$expected)
```

```
##
##          Non-original Original
##   car             137      844
##   pet             137      841
##   dress           133      818
##   makeup          130      798
##   food            133      818
##   game            138      845
##   plot             50      309
```

'type' and 'BGM' are not independent.

```
## type and month
table <- table(video$type, video$month)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 229.95, df = 24, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##            1   2   3   4   5
##   car    176 174 251 323  57
##   pet    219 221 227 260  51
##   dress  231 216 252 221  31
##   makeup 204 201 230 243  50
##   food   255 259 220 188  29
##   game   320 314 188 139  22
##   plot    78  91 100  79  11
```

```
round(chisq.test(table)$expected)
```

```
##
##            1   2   3   4  5
##   car    237 236 235 232 40
##   pet    237 235 234 232 40
##   dress  230 229 228 225 39
##   makeup 224 223 222 220 38
##   food   230 229 228 225 39
##   game   238 237 235 233 40
##   plot    87  86  86  85 15
```

'type' and 'month' are not independent.

```
## type and weekday
table <- table(video$type, video$weekday)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 44.326, df = 36, p-value = 0.1607
```

```
chisq.test(table)$observed
```

```
##
##            1   2   3   4   5   6   7
##   car    142 150 142 139 131 144 133
##   pet    129 131 155 159 120 150 134
##   dress  132 111 123 139 131 158 157
##   makeup 136 116 104 134 134 146 158
##   food   110 125 121 141 146 156 152
##   game   140 118 126 130 135 154 180
##   plot    47  37  55  54  44  64  58
```

```
round(chisq.test(table)$expected)
```

```
##
##            1   2   3   4   5   6   7
##   car    134 126 132 143 135 156 156
##   pet    133 126 132 143 134 155 155
##   dress  130 122 128 139 130 151 151
##   makeup 127 119 125 136 127 147 147
##   food   130 122 128 139 130 151 151
##   game   134 126 132 144 135 156 156
##   plot    49  46  48  52  49  57  57
```

'type' and 'weekday' are independent.

```
## type and hour
table <- table(video$type, video$hour)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 514.83, df = 138, p-value < 2.2e-16
```

```
chisq.test(table)$observed
```

```
##
##            0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16
##   car     12   8   4   4   3   3   3  10  20  50  87 111  70  70  46  48  66
##   pet     15   8   6   6   5   4   6  15  40  36  51 108 106  38  41  52  60
##   dress    8   9   4   3   1   3   5  15  24  32  65 105  89  31  34  30  49
##   makeup   1   7   2   2   0   4   1  10  12  27  41  72 110  29  25  30  47
##   food     9   6   0   2   1   0   3  14  27  42  53  75  64  44  44  30  62
##   game    13  22  10  11   6   9  10  15  51  34  39  89  77  45  41  44  64
##   plot     5   1   0   0   0   0   1   0   3   3  55  34  46  10  10   3  13
##
##            17  18  19  20  21  22  23
##   car    110  98  68  30  19  24  17
##   pet    128 105  59  37  24  16  12
##   dress  136 107  83  51  31  15  21
##   makeup 126 198  97  43  21  17   6
##   food   194 130  48  44  34  16   9
##   game   127 132  69  23  22  13  17
##   plot    54  74  20  16   8   1   2
```

```
round(chisq.test(table)$expected)
```

```
##
##           0  1 2 3 4 5 6  7  8  9 10 11 12 13 14 15 16  17  18 19 20 21 22 23
##   car    10 10 4 4 3 4 5 13 28 36 63 95 90 43 39 38 58 140 135 71 39 25 16 13
##   pet    10 10 4 4 3 4 5 13 28 36 62 95 90 43 38 38 58 140 135 71 39 25 16 13
##   dress  10  9 4 4 2 4 4 12 27 35 61 92 87 41 37 37 56 136 131 69 38 25 16 13
##   makeup 10  9 4 4 2 3 4 12 27 34 59 90 85 40 36 36 55 132 128 67 37 24 15 13
##   food   10  9 4 4 2 4 4 12 27 35 61 92 87 41 37 37 56 136 131 69 38 25 16 13
##   game   10 10 4 4 3 4 5 13 28 36 63 95 90 43 39 38 58 140 135 71 39 25 16 13
##   plot    4  4 2 2 1 1 2  5 10 13 23 35 33 16 14 14 21  51  49 26 14  9  6  5
```

'type' and 'hour' are not independent.

- BGM

```
## BGM and month
table <- table(video$BGM, video$month)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 25.288, df = 4, p-value = 4.404e-05
```

```
chisq.test(table)$observed
```

```
##
##                   1    2    3    4    5
##   Non-original  263  198  180  193   25
##   Original     1220 1278 1288 1260  226
```

```
round(chisq.test(table)$expected)
```

```
##
##                   1    2    3    4   5
##   Non-original  208  207  206  204  35
##   Original     1275 1269 1262 1249 216
```

'BGM' and 'month' are not independent.

```
## BGM and weekday
table <- table(video$BGM, video$weekday)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 3.1542, df = 6, p-value = 0.7893
```

```
chisq.test(table)$observed
```

```
##
##                   1   2   3   4   5   6   7
##   Non-original  124 109 113 128 109 148 128
##   Original      712 679 713 768 732 824 844
```

```
round(chisq.test(table)$expected)
```

```
##
##                   1   2   3   4   5   6   7
##   Non-original  117 110 116 126 118 136 136
##   Original      719 678 710 770 723 836 836
```

'BGM' and 'weekday' are independent.

```
## BGM and hour
table <- table(video$BGM, video$hour)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 66.934, df = 23, p-value = 3.584e-06
```

```
chisq.test(table)$observed
```

```
##
##                    0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
##   Non-original     7  13   6   4   5   4   4  12  25  18  50  74  78  39  47  54
##   Original        56  48  20  24  11  19  25  67 152 206 341 520 484 228 194 183
##
##                   16  17  18  19  20  21  22  23
##   Non-original    60 100  88  61  48  28  16  18
##   Original       301 775 756 383 196 131  86  66
```

```
round(chisq.test(table)$expected)
```

```
##
##                    0  1  2  3  4  5  6  7   8   9  10  11  12  13  14  15  16  17
##   Non-original     9  9  4  4  2  3  4 11  25  31  55  83  79  37  34  33  51 123
##   Original        54 52 22 24 14 20 25 68 152 193 336 511 483 230 207 204 310 752
##
##                   18  19  20  21 22 23
##   Non-original   118  62  34  22 14 12
##   Original       726 382 210 137 88 72
```

'BGM' and 'hour' are not independent.

- month

```r
## month and weekday
table <- table(video$month, video$weekday)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 81.319, df = 24, p-value = 3.746e-08
```

```r
chisq.test(table)$observed
```

```
##
##       1   2   3   4   5   6   7
##   1 193 186 225 226 212 224 217
##   2 203 212 195 191 197 235 243
##   3 217 169 170 209 199 264 240
##   4 192 201 223 201 189 213 234
##   5  31  20  13  69  44  36  38
```

```r
round(chisq.test(table)$expected)
```

```
##
##       1   2   3   4   5   6   7
##   1 202 191 200 217 203 235 235
##   2 201 190 199 216 202 234 234
##   3 200 189 198 215 201 233 233
##   4 198 187 196 212 199 230 230
##   5  34  32  34  37  34  40  40
```

'month' and 'weekday' are not independent.

```r
## month and hour
table <- table(video$month, video$hour)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 115.59, df = 92, p-value = 0.04868
```

```r
chisq.test(table)$observed
```

```
##
##       0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1  21  21   5   7   3   5   4  17  48  53  91 138 106  57  54  63  87 208 229
##   2  16  15   6   7   7   5   6  20  43  54  96 140 129  65  56  63  88 237 195
##   3  12  13   6   3   3   5   9  17  38  51  88 132 142  76  65  50  76 214 192
##   4  12  11   6  10   2   7   8  20  40  48  99 154 152  61  60  46  96 183 205
```

```
##   5   2   1   3   1   1   1   2   5   8  18  17  30  33   8   6  15  14  33  23
##
##      19  20  21  22  23
##   1 101  82  41  26  16
##   2 100  51  32  25  20
##   3 118  56  48  26  28
##   4 114  46  32  21  20
##   5  11   9   6   4   0
```

```
round(chisq.test(table)$expected)
```

```
##
##      0  1 2 3 4 5 6  7  8  9 10  11  12 13 14 15 16  17  18  19 20 21 22 23
##   1 15 15 6 7 4 6 7 19 43 54 95 144 136 65 58 57 87 212 204 107 59 38 25 20
##   2 15 15 6 7 4 6 7 19 43 54 94 143 135 64 58 57 87 211 203 107 59 38 25 20
##   3 15 15 6 7 4 6 7 19 42 54 94 142 135 64 58 57 86 210 202 106 58 38 24 20
##   4 15 14 6 7 4 5 7 19 42 53 93 141 133 63 57 56 86 207 200 105 58 38 24 20
##   5  3  2 1 1 1 1 1  3  7  9 16  24  23 11 10 10 15  36  35  18 10  7  4  3
```

'month' and 'hour' are not independent.

- weekday

```
## weekday and hour
table <- table(video$weekday, video$hour)
chisq.test(table)
```

```
##
##  Pearson's Chi-squared test
##
## data:  table
## X-squared = 182.48, df = 138, p-value = 0.006691
```

```
chisq.test(table)$observed
```

```
##
##      0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##   1  9   8   5   5   2   1   2   9  36  31  65  83 109  37  38  43  48  94  94
##   2  3  10   1   4   3   3   6  10  28  15  53  76  73  28  34  29  54 106 124
##   3  9   7   2   4   1   6   4  14  18  28  54  72  74  37  28  27  46 119 115
##   4  9   8   6   4   3   4   4  13  22  38  43  80  63  41  36  38  49 153 129
##   5  9  11   4   2   2   1   3  14  22  29  52  82  58  39  29  39  57 126 124
##   6  8  10   6   6   2   3   8  12  27  41  50  91  87  49  38  26  50 148 127
##   7 16   7   2   3   3   5   2   7  24  42  74 110  98  36  38  35  57 129 131
##
##      19  20  21  22  23
##   1  41  30  29   7  10
##   2  57  29  15  20   7
##   3  67  37  27  14  16
##   4  69  33  24  16  11
##   5  66  30  11  18  13
##   6  92  37  28  12  14
##   7  52  48  25  15  13
```

```
round(chisq.test(table)$expected)
```

```
##
##      0  1 2 3 4 5 6  7  8  9 10 11 12 13 14 15 16  17  18 19 20 21 22 23
##   1  9  8 4 4 2 3 4 11 24 31 53 81 77 36 33 32 49 119 115 61 33 22 14 11
##   2  8  8 3 4 2 3 4 10 23 29 50 76 72 34 31 30 46 112 108 57 31 20 13 11
##   3  8  8 4 4 2 3 4 11 24 30 53 80 76 36 32 32 49 118 114 60 33 21 14 11
##   4  9  9 4 4 2 3 4 12 26 33 57 87 82 39 35 35 53 128 123 65 36 23 15 12
##   5  9  8 4 4 2 3 4 11 24 31 54 81 77 37 33 33 50 120 116 61 33 22 14 12
##   6 10 10 4 4 3 4 5 13 28 36 62 94 89 42 38 38 57 139 134 70 39 25 16 13
##   7 10 10 4 4 3 4 5 13 28 36 62 94 89 42 38 38 57 139 134 70 39 25 16 13
```

'weekday' and 'hour' are not independent.

- Summary

Independent: author and weekday, type and weekday, BGM and weekday

Possible explanations for dependence(for example):

'author' & 'BGM': authors who frequently post videos may put more effort into them, so they may use original BGM more frequently.

'type' & 'hour': there are more short videos of game type in the midnight, possibly because the audience is mainly young people, and young people often stay up late.

'BGM' & 'month': internet hot songs may be different every month, and popular tastes will also change.

3.2 Group difference by ANOVA

Motivation: There are too many categorical variables in our data set, so we want to use ANOVA to test statistical differences among the means of 'log(like)' of two or more groups for each grouping factor.

3.2.1 Preparation

Before applying the ANOVA, we should divide our data into two subgroups since one assumption for ANOVA is the homogeneity of variance. But we find that the variances of different types are significantly different.

```
attach(video)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     index
```

```
table(type)
```

```
## type
##    car    pet  dress makeup   food   game   plot
##    981    978    951    928    951    983    359
```

```
data.frame(setNames(aggregate(
  log(like), by = list(type), FUN = mean
), c("BGM", "mean")), setNames(aggregate(
  log(like), by = list(type), FUN = sd
), c("BGM", "sd"))[-1])
```

```
##      BGM     mean        sd
## 1    car 11.33050 1.2473565
## 2    pet 12.75801 0.6795224
## 3  dress 12.90131 0.4885776
## 4 makeup 13.38167 0.4589971
## 5   food 13.51814 0.4345032
## 6   game 13.58708 0.3598626
## 7   plot 14.10621 0.3499170
```

```
detach(video)
```

```
bartlett.test(log(like) ~ type, data = video)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by type
## Bartlett's K-squared = 2561.3, df = 6, p-value < 2.2e-16
```

Homogeneity of variance is violated severally. The variance of the short video of the car type is significantly larger than other short videos, so we divide the data into car type short videos and non-car type short videos.

```
car.ind <- which(video$type == "car")

video.car <- video[car.ind, ]
video.noncar <- video[-car.ind, ]
video.noncar$type <-
  factor(video.noncar$type,
         levels = c("food", "makeup", "game", "dress", "pet", "plot"))
```

We conduct the test for several interested grouping factor on the whole data, car-type data and non-car type data respectively.

```
## find p-value
col <- function(data) {
  pval.author <-
    bartlett.test(log(like) ~ author, data = data)$p.value
  pval.BGM <- bartlett.test(log(like) ~ BGM, data = data)$p.value
  pval.month <-
    bartlett.test(log(like) ~ month, data = data)$p.value
  pval.weekday <-
    bartlett.test(log(like) ~ weekday, data = data)$p.value
  pval.period <-
    bartlett.test(log(like) ~ period, data = data)$p.value
```

```
  col <-
    round(c(pval.author, pval.BGM, pval.month, pval.weekday, pval.period),
          8)
  return(col)
}

col1 <- col(video)
col2 <- col(video.car)
col3 <- col(video.noncar)

## p-value table
data.frame(
  grouping.factor = c("author", "BGM", "month", "weekday", "period"),
  video = col1,
  video.car = col2,
  video.noncar = col3
)
```

```
##   grouping.factor      video  video.car video.noncar
## 1          author 0.00261660 0.00321821   0.00000000
## 2             BGM 0.00603653 0.00060279   0.68333738
## 3           month 0.00000000 0.75343779   0.36339772
## 4         weekday 0.00198178 0.66625642   0.07431736
## 5          period 0.00000001 0.19244529   0.06665090
```

Obviously, if we use 'video', then the homogeneity of variance assumption would be violated for each grouping factor. But for both 'video.car' and 'video.noncar', the assumptions are satisfied better.

So we use data 'video.noncar' and 'video.car' repectively.

3.2.2 One-way ANOVA

```
attach(video.noncar)
```

```
## The following object is masked _by_ .GlobalEnv:
##
##     index
```

- One-way ANOVA with categorical grouping factor author

```
### Group info
table(author)
```

```
## author
##     once  several    often frequent
##     1422     1184     1315     1229
```

```
data.frame(setNames(aggregate(
  log(like), by = list(author), FUN = mean
), c("author", "mean")), setNames(aggregate(
  log(like), by = list(author), FUN = sd
), c("author", "sd"))[-1])
```

```
##     author     mean        sd
## 1     once 13.19027 0.5706682
## 2  several 13.24019 0.5890459
## 3    often 13.34458 0.6285714
## 4 frequent 13.39264 0.6995171
```

```
### Test for group difference
aov_author <- aov(log(like) ~ author)
summary(aov_author)
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## author         3   33.9  11.315   29.21 <2e-16 ***
## Residuals   5146 1993.3   0.387
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The p-value of the F test indicates that various types of 'author' indeed will influence the number of likes.

```
### Outlier test
library(car)
outlierTest(aov_author)
```

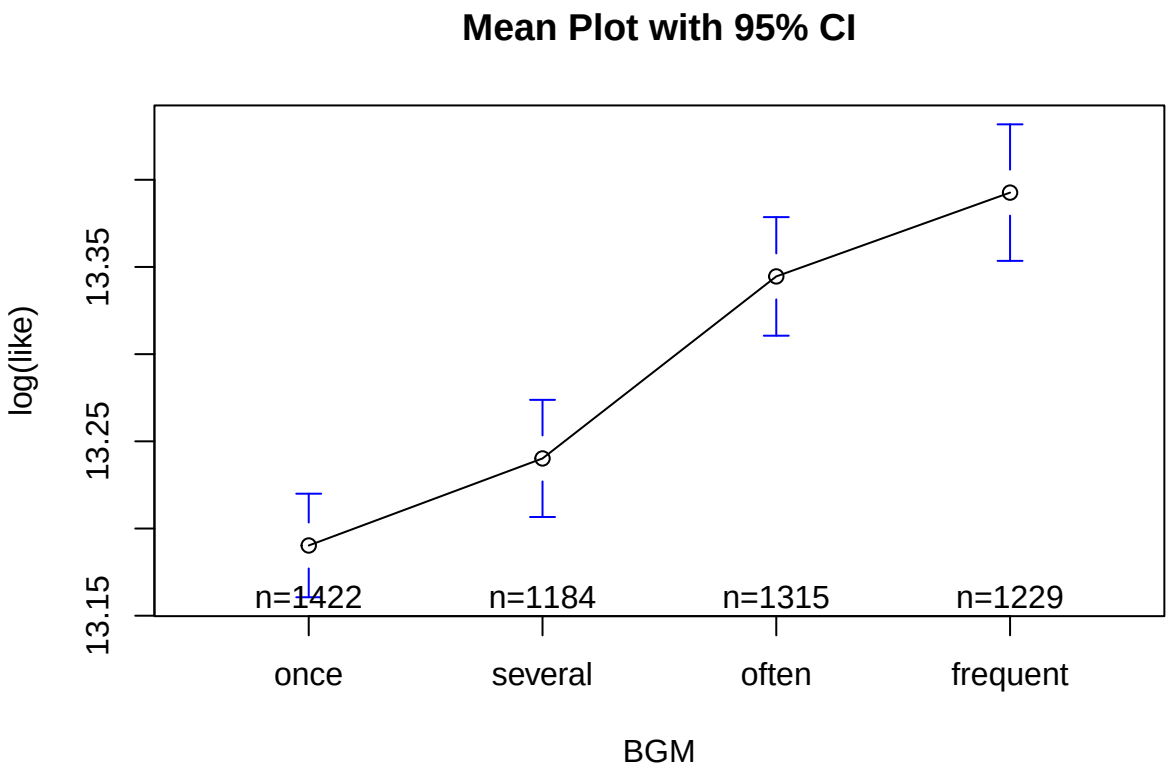
```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 952 3.366017         0.00076821           NA
```

There is no indication of outliers in the data.

```
### Plot group means, confidence intervals
library(gplots)
```

```
##
##    'gplots'

## The following object is masked _by_ '.GlobalEnv':
##
##     residplot

## The following object is masked from 'package:stats':
##
##     lowess
```

```
plotmeans(log(like) ~ author,
          xlab = "BGM",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```

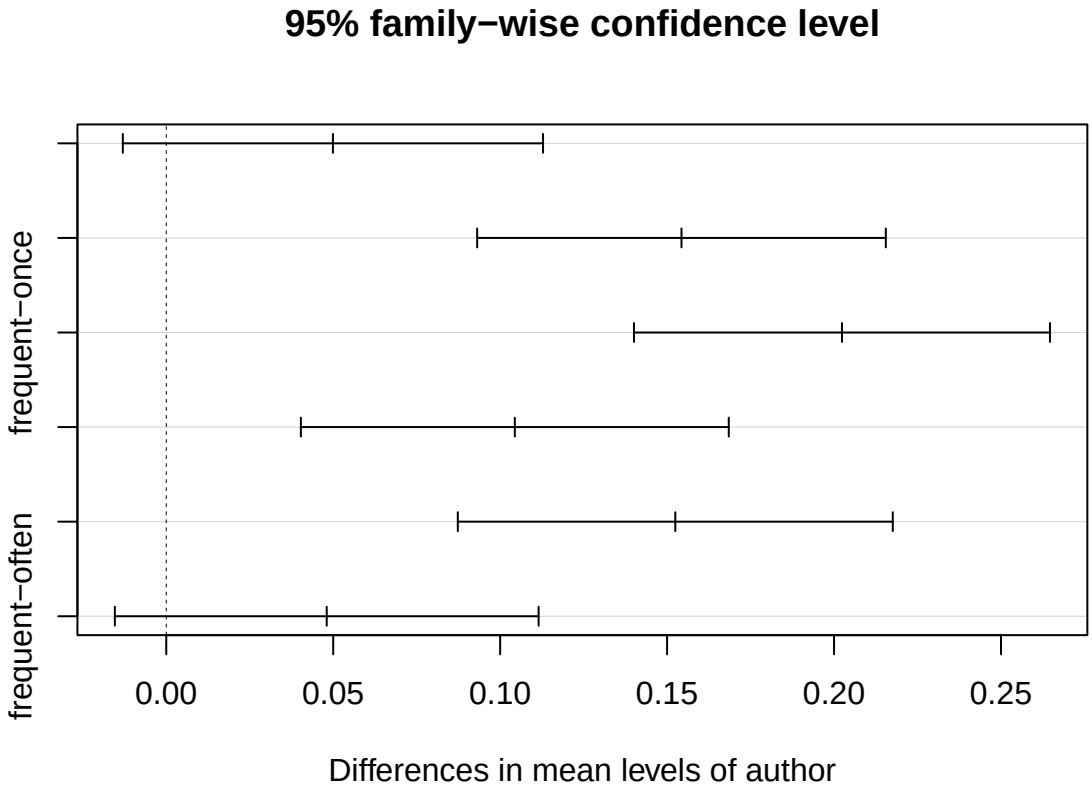


Videos posted by diligent authors have a high number of likes, and short video bloggers have to work hard.

```
### Multiple comparison
TukeyHSD(aov_author)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = log(like) ~ author)
##
## $author
##                         diff         lwr       upr     p adj
## several-once     0.04991585 -0.01300919 0.1128409 0.1739944
## often-once       0.15431042  0.09311955 0.2155013 0.0000000
## frequent-once    0.20236853  0.14007527 0.2646618 0.0000000
## often-several    0.10439457  0.04031696 0.1684722 0.0001687
## frequent-several 0.15245268  0.08732153 0.2175838 0.0000000
## frequent-often   0.04805811 -0.01539919 0.1115154 0.2089331
```

```
plot(TukeyHSD(aov_author))
```




```
### Homogeneity of variance
bartlett.test(log(like) ~ author)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by author
## Bartlett's K-squared = 63.651, df = 3, p-value = 9.749e-14
```

Homogeneity of variance is not satisfied.

- One-way ANOVA with categorical grouping factor BGM

```
### Group info
table(BGM)
```

```
## BGM
## Non-original     Original
##          714         4436
```

```
data.frame(setNames(aggregate(
  log(like), by = list(BGM), FUN = mean
), c("BGM", "mean")), setNames(aggregate(
  log(like), by = list(BGM), FUN = sd
), c("BGM", "sd"))[-1])
```

```
##            BGM     mean        sd
## 1 Non-original 13.17602 0.6321553
## 2     Original 13.30770 0.6248595
```

```
### Test for group difference
aov_bgm <- aov(log(like) ~ BGM)
summary(aov_bgm)
```

```
##               Df Sum Sq Mean Sq F value   Pr(>F)
## BGM            1   10.7  10.664   27.22 1.88e-07 ***
## Residuals   5148 2016.6   0.392
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

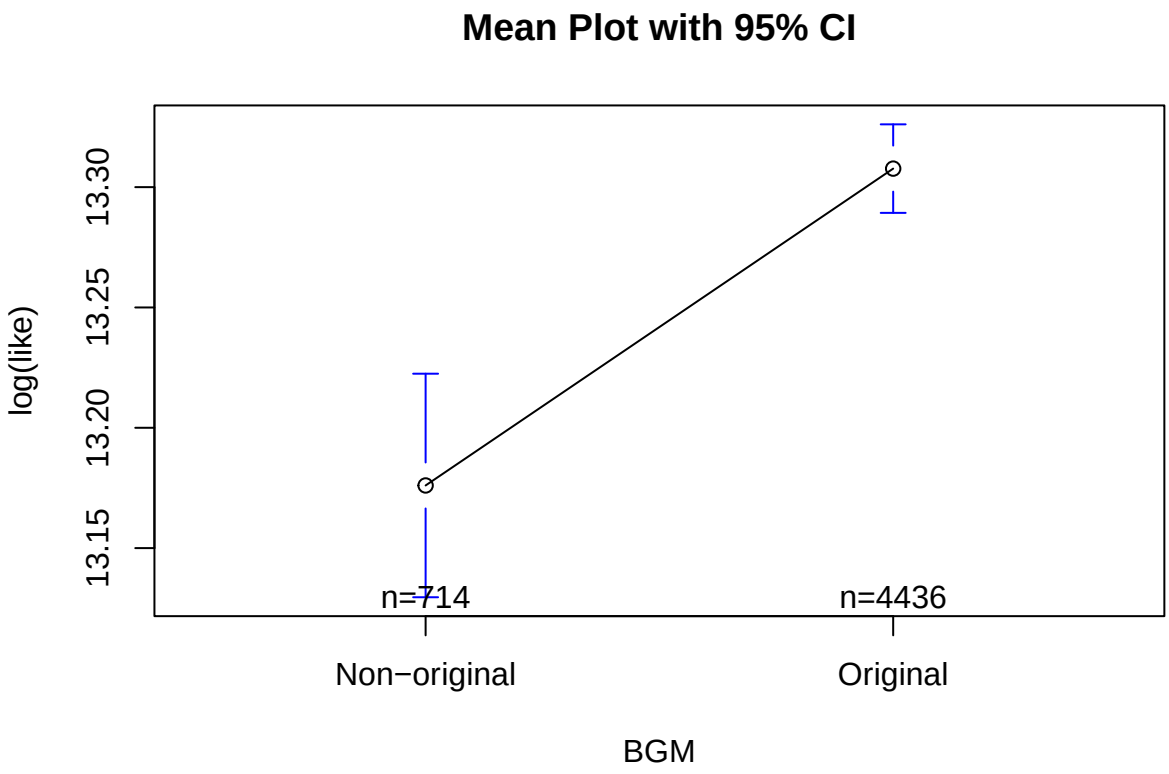
The p-value of the F test indicates that various types of BGM indeed will influence the number of likes.

```
### Outlier test
library(car)
outlierTest(aov_bgm)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 952 3.451005         0.00056299           NA
```

There is no indication of outliers in the data.

```
### Plot group means, confidence intervals
library(gplots)
plotmeans(log(like) ~ BGM,
          xlab = "BGM",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```

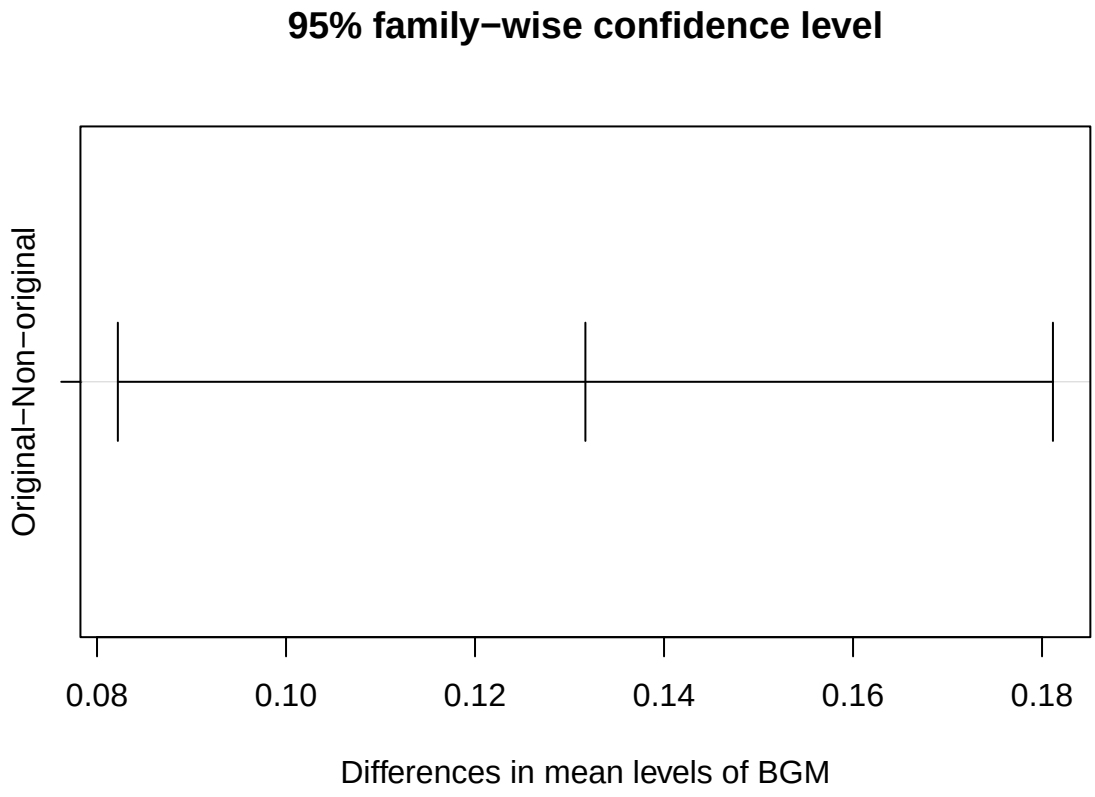


From the figure of 95% confident interval, we can clearly find the difference between various types of BGM. Short videos with original BGM have higher likes and a small standard deviation. However, short videos with non-original BGM have few likes and large variance.

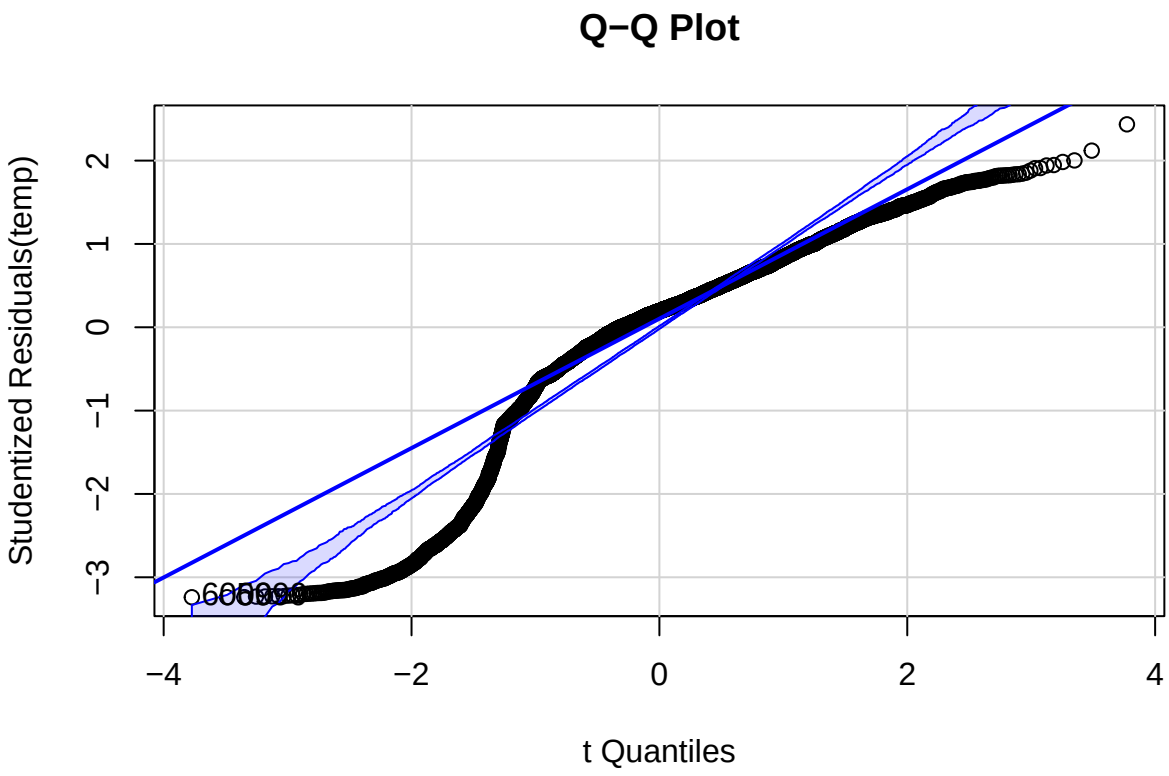
```
### Multiple comparison
TukeyHSD(aov_bgm)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = log(like) ~ BGM)
##
## $BGM
##                              diff        lwr       upr p adj
## Original-Non-original 0.1316785 0.08220228 0.1811547 2e-07
```

```
plot(TukeyHSD(aov_bgm))
```

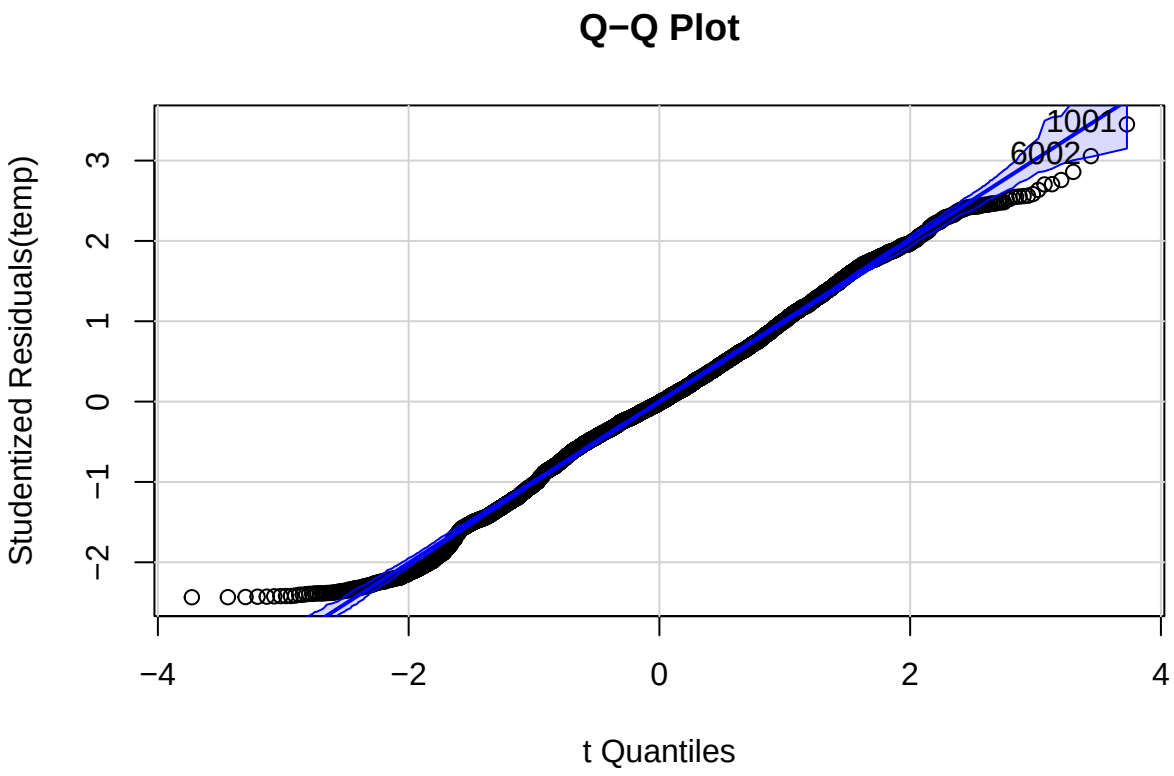



```
### Normality
## use 'video'
temp <- lm(log(like) ~ BGM, data = video)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```




```
## 5999 6000
## 5771 5772
```

```
## use 'video.noncar'
temp <- lm(log(like) ~ BGM, data = video.noncar)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```




```
## 1001 6002
##  952 4792
```

```
re <- temp$residuals
ks.test(re, "pnorm", mean = mean(re), sd = sd(re))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  re
## D = 0.026508, p-value = 0.001438
## alternative hypothesis: two-sided
```

The two Q-Q plots strengthen our confidence that we can not easily use ANOVA for the whole data. The ks.test shows that the normality assumption should be rejected although the Q-Q plot looks good.

```
### Homogeneity of variance
bartlett.test(log(like) ~ BGM)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by BGM
## Bartlett's K-squared = 0.16639, df = 1, p-value = 0.6833
```

Homogeneity of variance is satisfied.

- One-way ANOVA with categorical grouping factor BGM

```
### Group info
table(BGM)
```

```
## BGM
## Non-original     Original
##          714         4436
```

```
data.frame(setNames(aggregate(
  log(like), by = list(BGM), FUN = mean
), c("BGM", "mean")), setNames(aggregate(
  log(like), by = list(BGM), FUN = sd
), c("BGM", "sd"))[-1])
```

```
##            BGM     mean        sd
## 1 Non-original 13.17602 0.6321553
## 2     Original 13.30770 0.6248595
```

```
### Test for group difference
aov_bgm <- aov(log(like) ~ BGM)
summary(aov_bgm)
```

```
##               Df Sum Sq Mean Sq F value   Pr(>F)
## BGM            1   10.7  10.664   27.22 1.88e-07 ***
## Residuals   5148 2016.6   0.392
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

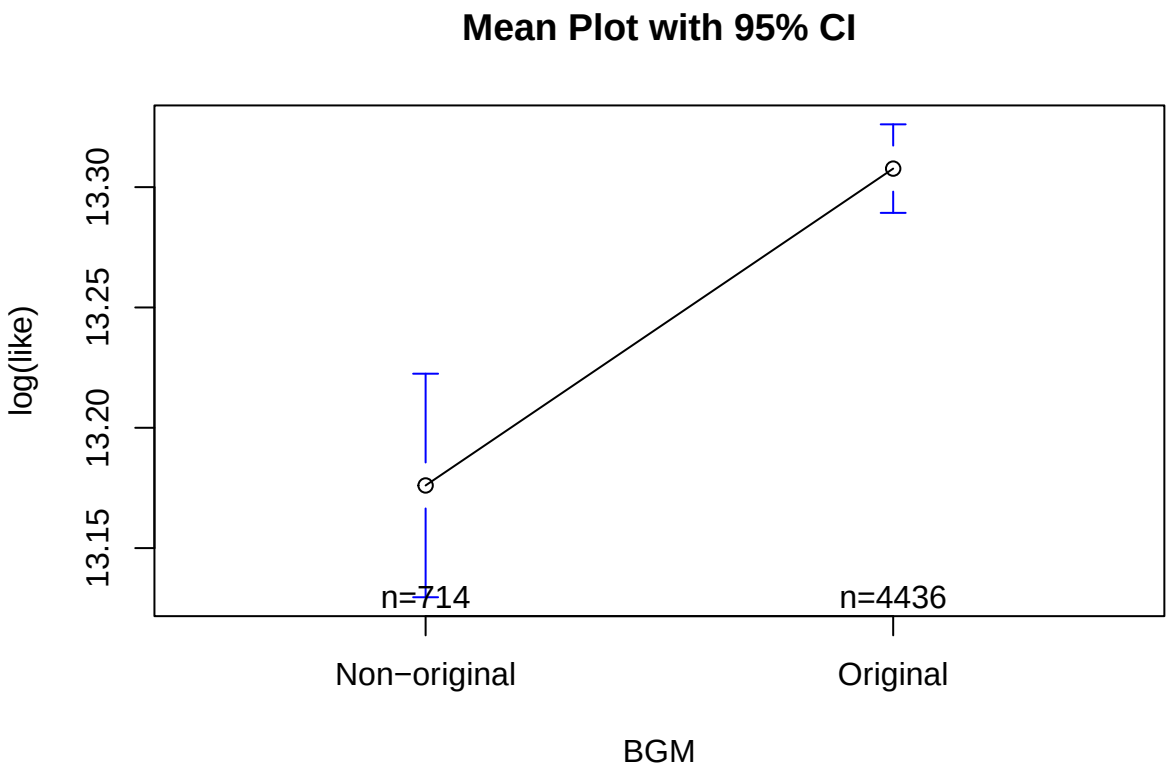
The p-value of the F test indicates that various types of BGM indeed will influence the number of likes.

```
### Outlier test
library(car)
outlierTest(aov_bgm)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 952 3.451005         0.00056299           NA
```

There is no indication of outliers in the data.

```
### Plot group means, confidence intervals
library(gplots)
plotmeans(log(like) ~ BGM,
          xlab = "BGM",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```

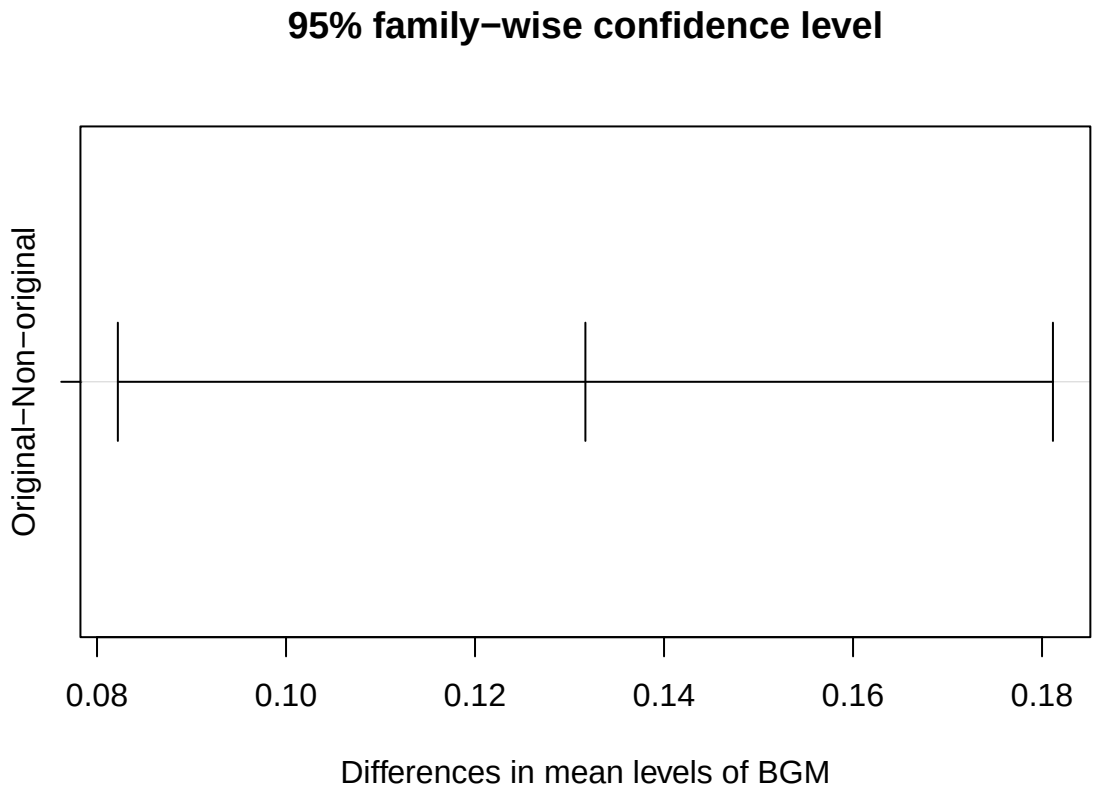


From the figure of 95% confident interval, we can clearly find the difference between various types of BGM. Short videos with original BGM have higher likes and a small standard deviation. However, short videos with non-original BGM have few likes and large variance.

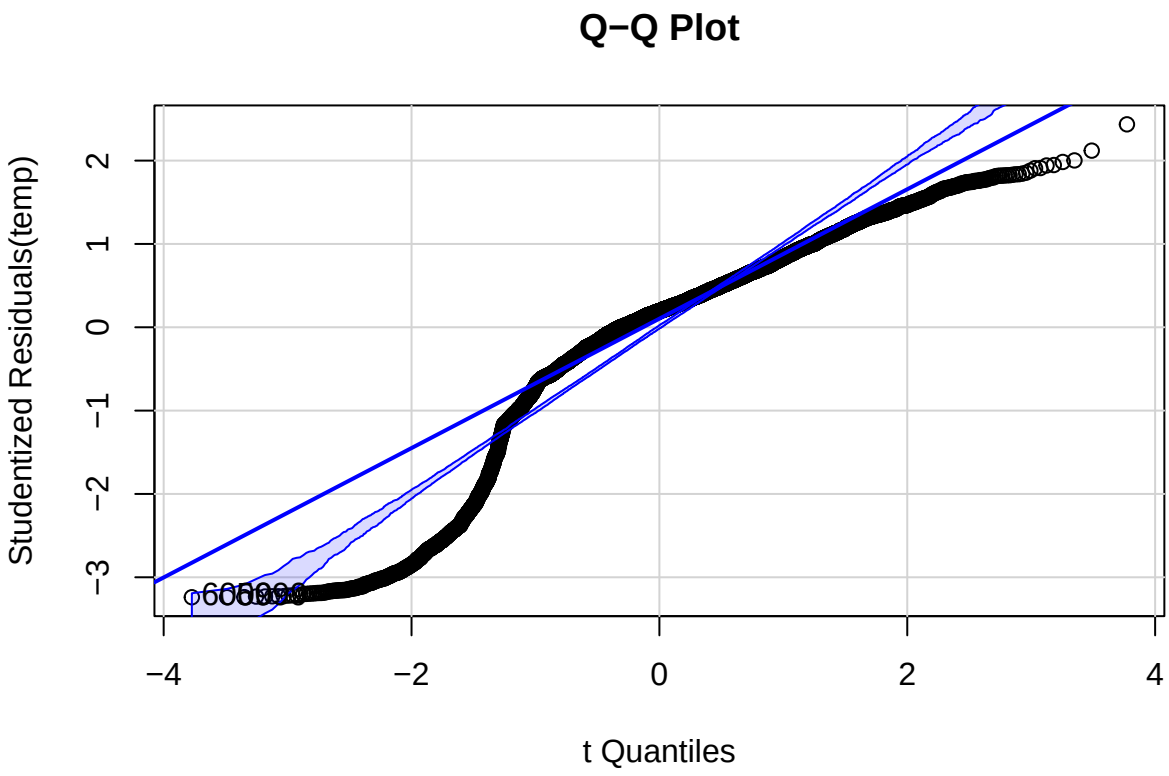
```
### Multiple comparison
TukeyHSD(aov_bgm)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = log(like) ~ BGM)
##
## $BGM
##                              diff        lwr       upr p adj
## Original-Non-original 0.1316785 0.08220228 0.1811547 2e-07
```

```
plot(TukeyHSD(aov_bgm))
```

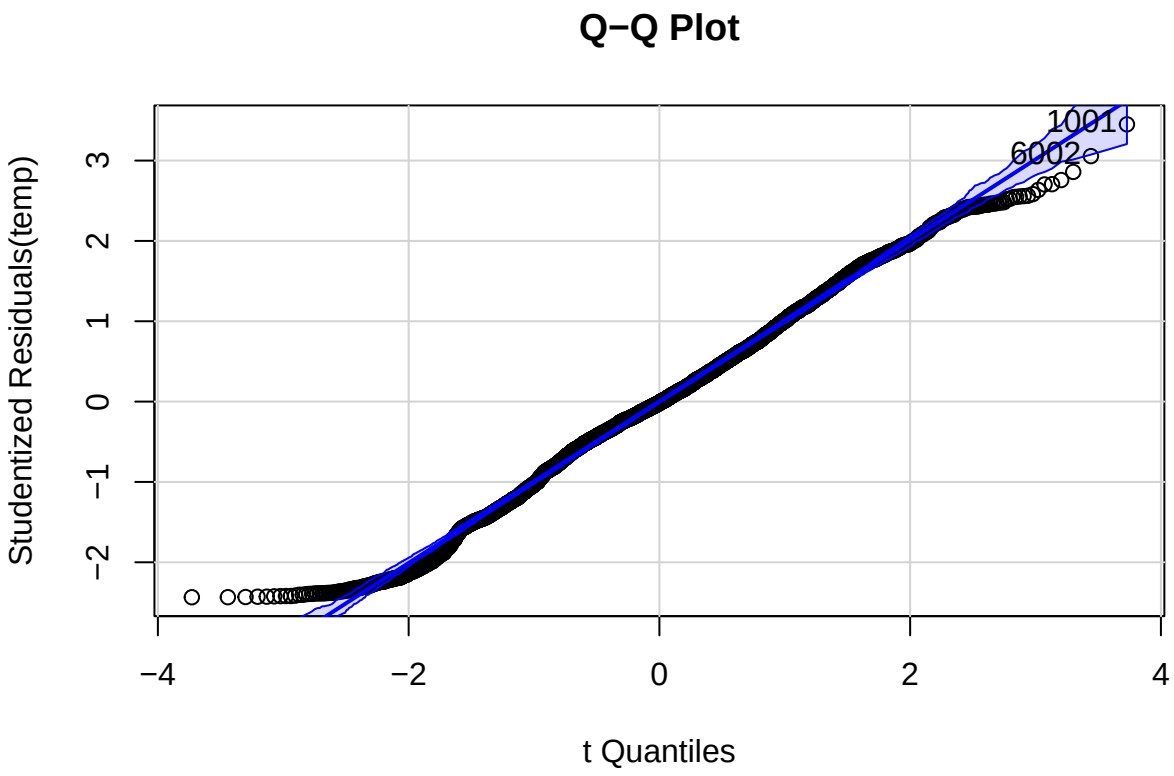



```
### Normality
## use 'video'
temp <- lm(log(like) ~ BGM, data = video)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```




```
## 5999 6000
## 5771 5772
```

```
## use 'video.noncar'
temp <- lm(log(like) ~ BGM, data = video.noncar)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```




```
## 1001 6002
##  952 4792
```

```
re <- temp$residuals
ks.test(re, "pnorm", mean = mean(re), sd = sd(re))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  re
## D = 0.026508, p-value = 0.001438
## alternative hypothesis: two-sided
```

The two Q-Q plots strengthen our confidence that we can not easily use ANOVA for the whole data. The ks.test shows that the normality assumption should be rejected although the Q-Q plot looks good.

```
### Homogeneity of variance
bartlett.test(log(like) ~ BGM)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by BGM
## Bartlett's K-squared = 0.16639, df = 1, p-value = 0.6833
```

Homogeneity of variance is satisfied.

- One-way ANOVA with categorical grouping factor weekday

```
### Group info
table(weekday)
```

```
## weekday
##   1   2   3   4   5   6   7
## 694 638 684 757 710 828 839
```

```
data.frame(setNames(aggregate(
  log(like), by = list(weekday), FUN = mean
), c("Wweekday", "mean")),
setNames(aggregate(
  log(like), by = list(weekday), FUN = sd
), c("weekday", "sd"))[-1])
```

```
##   Wweekday     mean        sd
## 1        1 13.28266 0.6471261
## 2        2 13.26161 0.6256805
## 3        3 13.25640 0.6071589
## 4        4 13.27317 0.6525595
## 5        5 13.33371 0.6019502
## 6        6 13.30425 0.6478952
## 7        7 13.30578 0.6048957
```

```
### Test for group difference
aov_weekday <- aov(log(like) ~ weekday)
summary(aov_weekday)
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## weekday        6    3.3  0.5450   1.385  0.216
## Residuals   5143 2024.0  0.3935
```

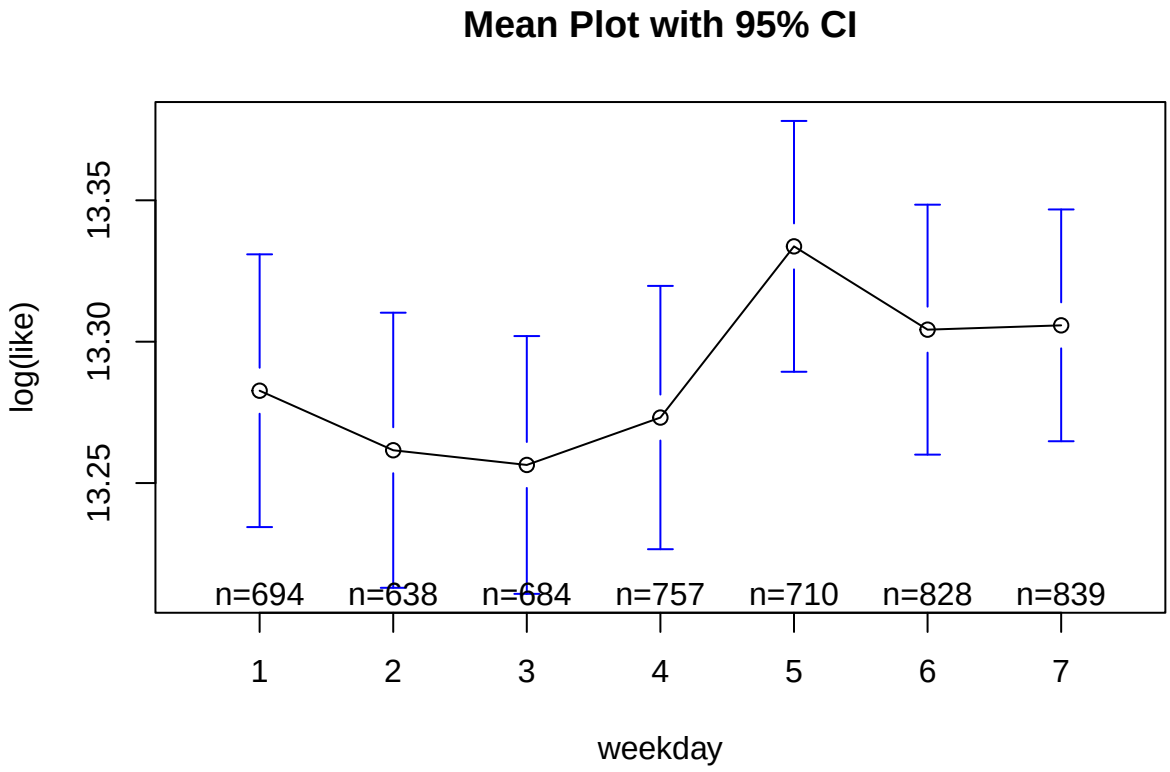
The p-value of the F test indicates the independent variable weekday will not influence the number of likes.

```
### Outlier test
library(car)
outlierTest(aov_weekday)
```

```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 952 3.237495          0.0012135           NA
```

There is no indication of outliers in the data.

```
### Plot group means, confidence intervals
library(gplots)
plotmeans(log(like) ~ weekday,
          xlab = "weekday",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```



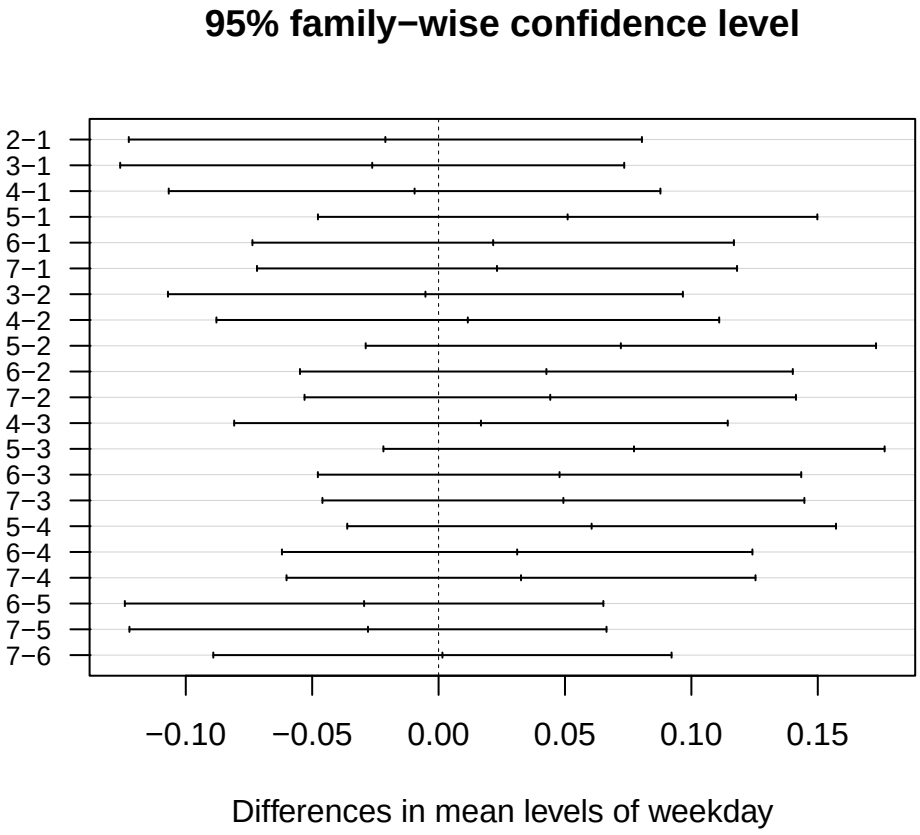
There is no significant difference between each weekday, maybe a little more on Friday.

```
### Multiple comparison
TukeyHSD(aov_weekday)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = log(like) ~ weekday)
##
## $weekday
##             diff         lwr        upr     p adj
## 2-1 -0.021056277 -0.12254142 0.08042886 0.9964632
## 3-1 -0.026257839 -0.12594914 0.07343346 0.9871639
## 4-1 -0.009494189 -0.10673453 0.08774615 0.9999538
## 5-1  0.051048124 -0.04771960 0.14981585 0.7301383
## 6-1  0.021592915 -0.07363249 0.11681832 0.9942417
## 7-1  0.023115824 -0.07182452 0.11805616 0.9915434
## 3-2 -0.005201562 -0.10704141 0.09663829 0.9999990
```

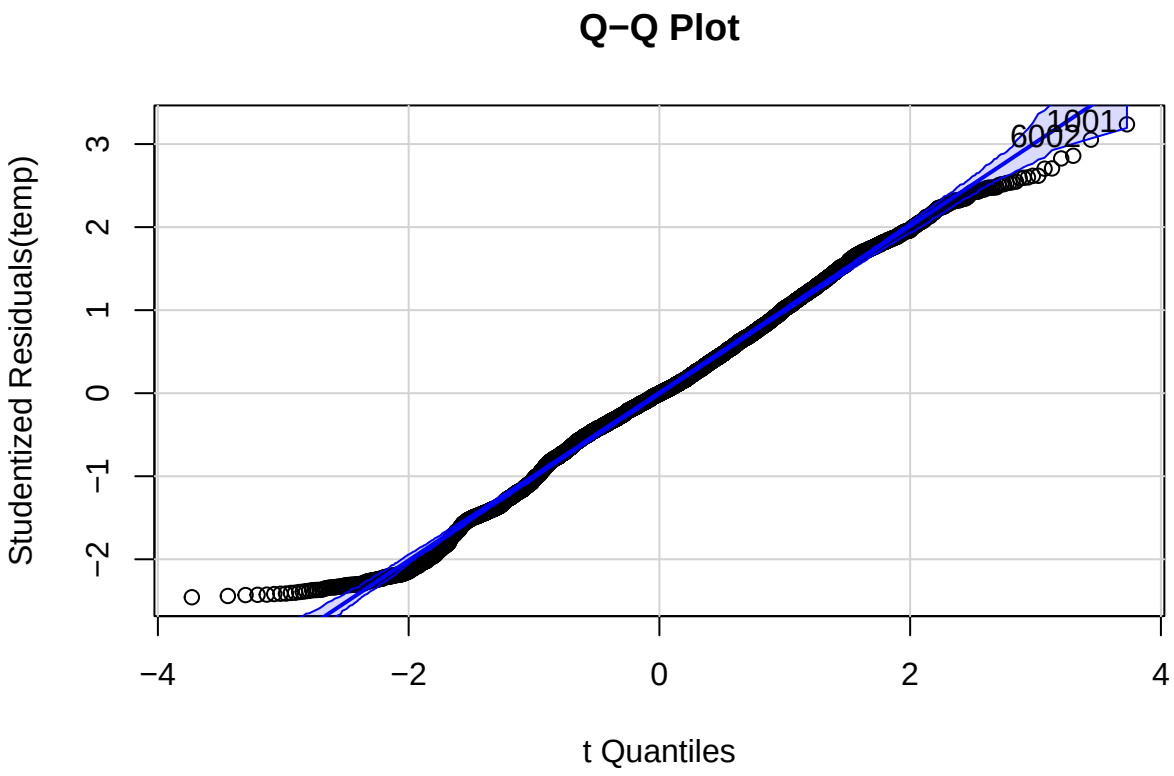
```
## 4-2  0.011562088 -0.08787977 0.11100395 0.9998711
## 5-2  0.072104401 -0.02883154 0.17304035 0.3485117
## 6-2  0.042649193 -0.05482326 0.14012165 0.8565425
## 7-2  0.044172102 -0.05302187 0.14136608 0.8328382
## 4-3  0.016763650 -0.08084682 0.11437412 0.9987748
## 5-3  0.077305963 -0.02182620 0.17643813 0.2439736
## 6-3  0.047850755 -0.04775259 0.14345410 0.7591102
## 7-3  0.049373664 -0.04594575 0.14469307 0.7280863
## 5-4  0.060542313 -0.03612471 0.15720934 0.5159031
## 6-4  0.031087105 -0.06195766 0.12413187 0.9572589
## 7-4  0.032610014 -0.06014298 0.12536301 0.9454837
## 6-5 -0.029455208 -0.12409510 0.06518469 0.9698177
## 7-5 -0.027932299 -0.12228536 0.06642076 0.9765277
## 7-6  0.001522909 -0.08911546 0.09216128 1.0000000
```

```r
tuk <- TukeyHSD(aov_weekday)
psig <- as.numeric(apply(tuk$weekday[, 2:3], 1, prod) >= 0) + 1
op <- par(mar = c(4.2, 9, 3.8, 2))
plot(tuk, col = psig, yaxt = "n")
for (j in 1:length(psig)) {
  axis(
    2,
    at = j,
    labels = rownames(tuk$weekday)[length(psig) - j + 1],
    las = 1,
    cex.axis = .8,
    col.axis = psig[length(psig) - j + 1]
  )
}
```




```
par(op)
```

```
### Normality
temp <- lm(log(like) ~ weekday, data = video.noncar)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```




```
## 1001 6002
##  952 4792
```

```
re <- temp$residuals
ks.test(re, "pnorm", mean = mean(re), sd = sd(re))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  re
## D = 0.026306, p-value = 0.001605
## alternative hypothesis: two-sided
```

The ks.test shows that the normality assumption should be rejected although the Q-Q plot looks good.

```
### Homogeneity of variance
bartlett.test(log(like) ~ weekday)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by weekday
## Bartlett's K-squared = 11.492, df = 6, p-value = 0.07432
```

Homogeneity of variance is satisfied.

- One-way ANOVA with categorical grouping factor period

```
### Group info
table(period)
```

```
## period
## 09:00-12:00 12:00-16:00 16:00-18:00 18:00-20:00  20:00-9:00
##         961        1073        1060        1122         934
```

```
data.frame(setNames(aggregate(
  log(like), by = list(period), FUN = mean
), c("period", "mean")),
setNames(aggregate(
  log(like), by = list(period), FUN = sd
), c("period", "sd"))[-1])
```

```
##        period     mean        sd
## 1 09:00-12:00 13.27430 0.6613731
## 2 12:00-16:00 13.28930 0.6317801
## 3 16:00-18:00 13.31370 0.6189417
## 4 18:00-20:00 13.32543 0.6154773
## 5  20:00-9:00 13.23444 0.6071840
```

```
### Test for group difference
aov_period <- aov(log(like) ~ period)
summary(aov_period)
```

```
##               Df Sum Sq Mean Sq F value Pr(>F)
## period         4    5.1   1.281   3.258 0.0112 *
## Residuals   5145 2022.1   0.393
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

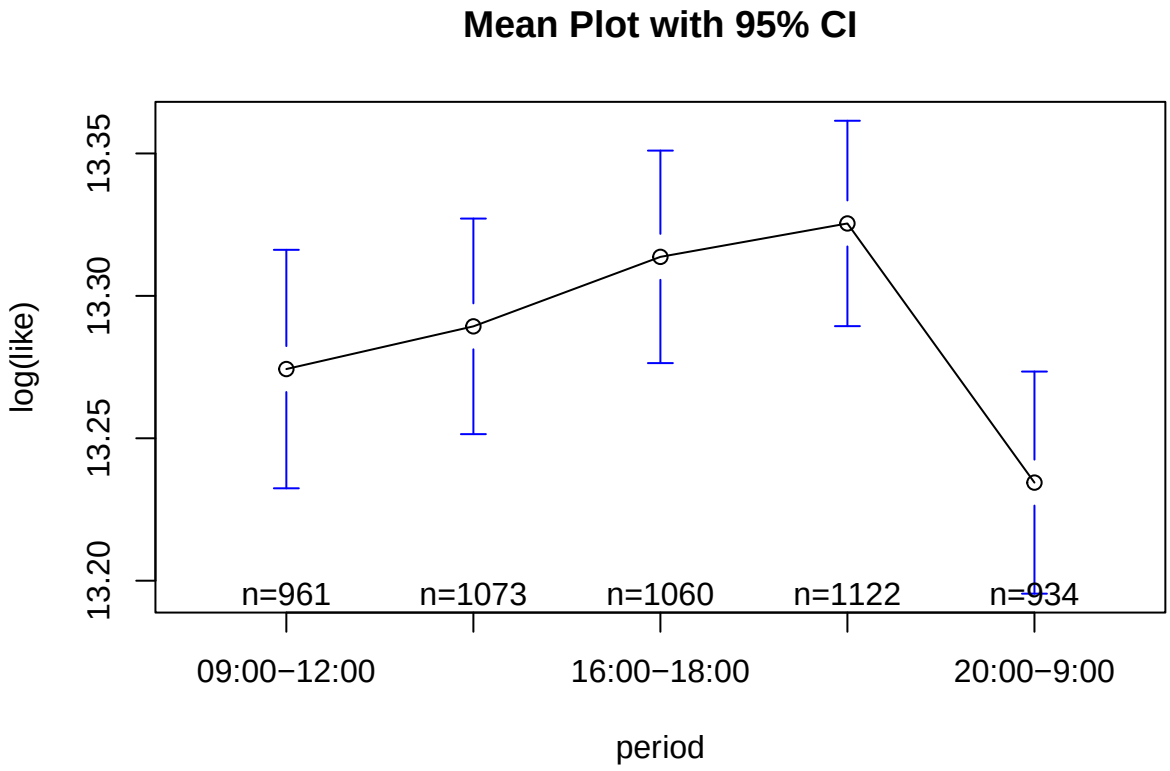
The p-value of the F test indicates period indeed will influence the number of likes.

```
### Outlier test
library(car)
outlierTest(aov_period)
```

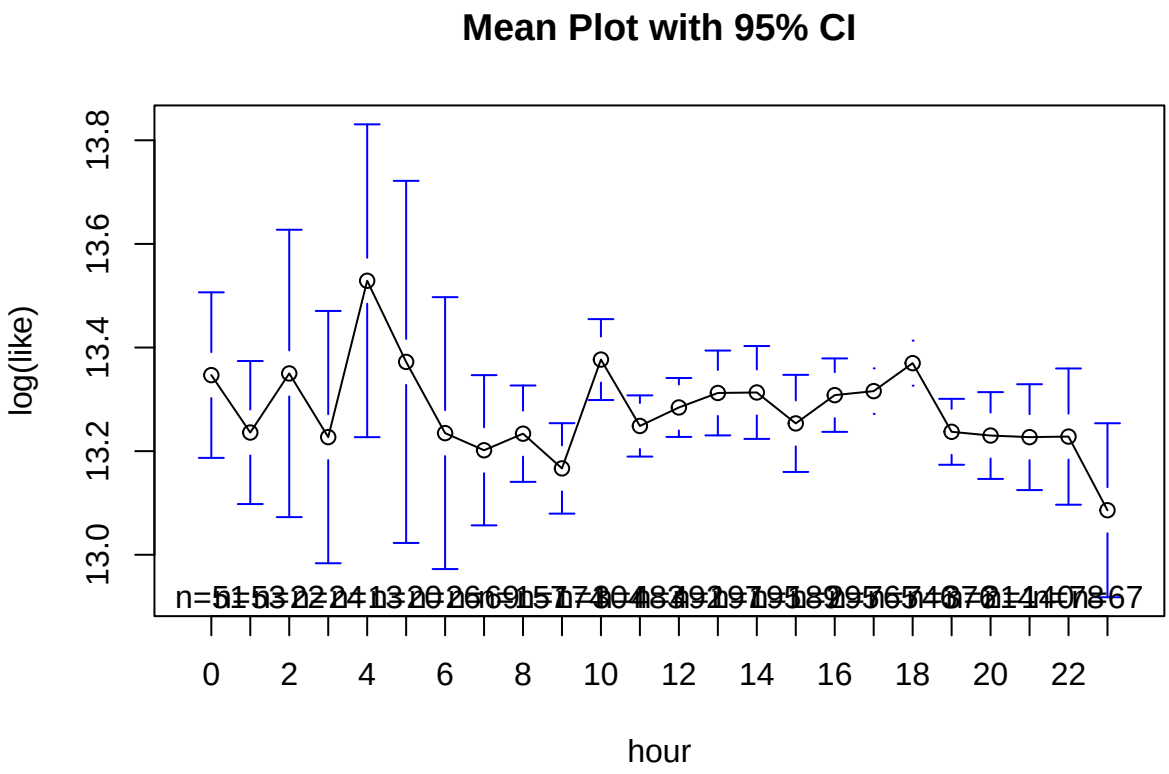
```
## No Studentized residuals with Bonferroni p < 0.05
## Largest |rstudent|:
##     rstudent unadjusted p-value Bonferroni p
## 952   3.2631          0.0011092           NA
```

There is no indication of outliers in the data.

```
### Plot group means, confidence intervals
library(gplots)
plotmeans(log(like) ~ period,
          xlab = "period",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```




```
### Plot group means, confidence intervals
library(gplots)
plotmeans(log(like) ~ hour,
          xlab = "hour",
          ylab = "log(like)",
          main = "Mean Plot with 95% CI")
```



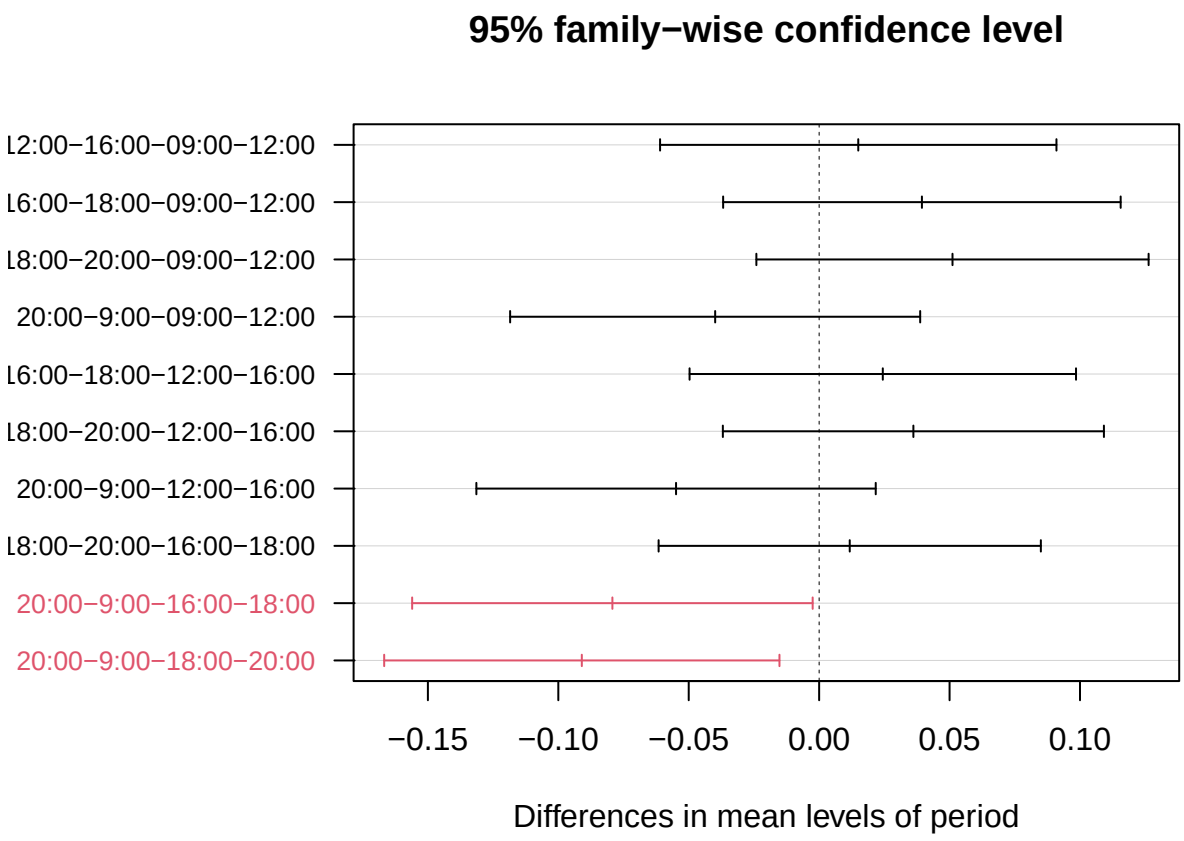
The variance of the number of likes of short videos posted in the early morning(2:00-7:00) is very large, but the overall average is very low. So we can conclude that some short videos with excellent quality are released in the early morning.

Golden time for short video release: 18:00-20:00.

```
### Multiple comparison
TukeyHSD(aov_period)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = log(like) ~ period)
##
## $period
##                                diff         lwr          upr     p adj
## 12:00-16:00-09:00-12:00  0.01499499 -0.06098279  0.090972762 0.9833258
## 16:00-18:00-09:00-12:00  0.03939601 -0.03680157  0.115593594 0.6206771
## 18:00-20:00-09:00-12:00  0.05112313 -0.02406671  0.126312974 0.3419134
## 20:00-9:00-09:00-12:00  -0.03986294 -0.11846648  0.038740607 0.6381065
## 16:00-18:00-12:00-16:00  0.02440103 -0.04968142  0.098483467 0.8974354
## 18:00-20:00-12:00-16:00  0.03612815 -0.03691738  0.109173673 0.6600139
## 20:00-9:00-12:00-16:00  -0.05485792 -0.13141283  0.021696984 0.2883000
## 18:00-20:00-16:00-18:00  0.01172712 -0.06154701  0.085001249 0.9924533
## 20:00-9:00-16:00-18:00  -0.07925895 -0.15603201 -0.002485889 0.0390568
## 20:00-9:00-18:00-20:00  -0.09098607 -0.16675904 -0.015213095 0.0093436
```

```
tuk <- TukeyHSD(aov_period)
psig <- as.numeric(apply(tuk$period[, 2:3], 1, prod) >= 0) + 1
op <- par(mar = c(4.2, 9, 3.8, 2))
plot(tuk, col = psig, yaxt = "n")
for (j in 1:length(psig)) {
  axis(
    2,
    at = j,
    labels = rownames(tuk$period)[length(psig) - j + 1],
    las = 1,
    cex.axis = .8,
    col.axis = psig[length(psig) - j + 1]
  )
}
```




```
par(op)
```

Significantly different in mean levels: 20:00-9:00 and 16:00-18:00, 20:00-9:00 and 18:00-20:00

```
### Normality
temp <- lm(log(like) ~ period, data = video.noncar)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```


```
## 1001 6002
##  952 4792
```

```
re <- temp$residuals
ks.test(re, "pnorm", mean = mean(re), sd = sd(re))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  re
## D = 0.026579, p-value = 0.001384
## alternative hypothesis: two-sided
```

The ks.test shows that the normality assumption should be rejected although the Q-Q plot looks good.

```
### Homogeneity of variance
bartlett.test(log(like) ~ period)
```

```
##
##  Bartlett test of homogeneity of variances
##
## data:  log(like) by period
## Bartlett's K-squared = 8.787, df = 4, p-value = 0.06665
```

Homogeneity of variance is satisfied.

3.2.3 Two-way ANOVA

- Two-way ANOVA with categorical grouping factor weekday and period

```
### Group info
table(weekday, period)
```

```
##         period
## weekday 09:00-12:00 12:00-16:00 16:00-18:00 18:00-20:00 20:00-9:00
##       1         146         189         113         113        133
##       2         110         130         129         153        116
##       3         115         128         144         160        137
##       4         125         138         179         174        141
##       5         124         139         160         167        120
##       6         149         166         175         193        145
##       7         192         183         160         162        142
```

```
data.frame(setNames(
  aggregate(log(like), by = list(weekday, period), FUN = mean),
  c("weekday", "period", "mean")
),
setNames(aggregate(
  log(like), by = list(weekday, period), FUN = sd
), c("weekday", "period", "sd"))[-c(1, 2)])
```

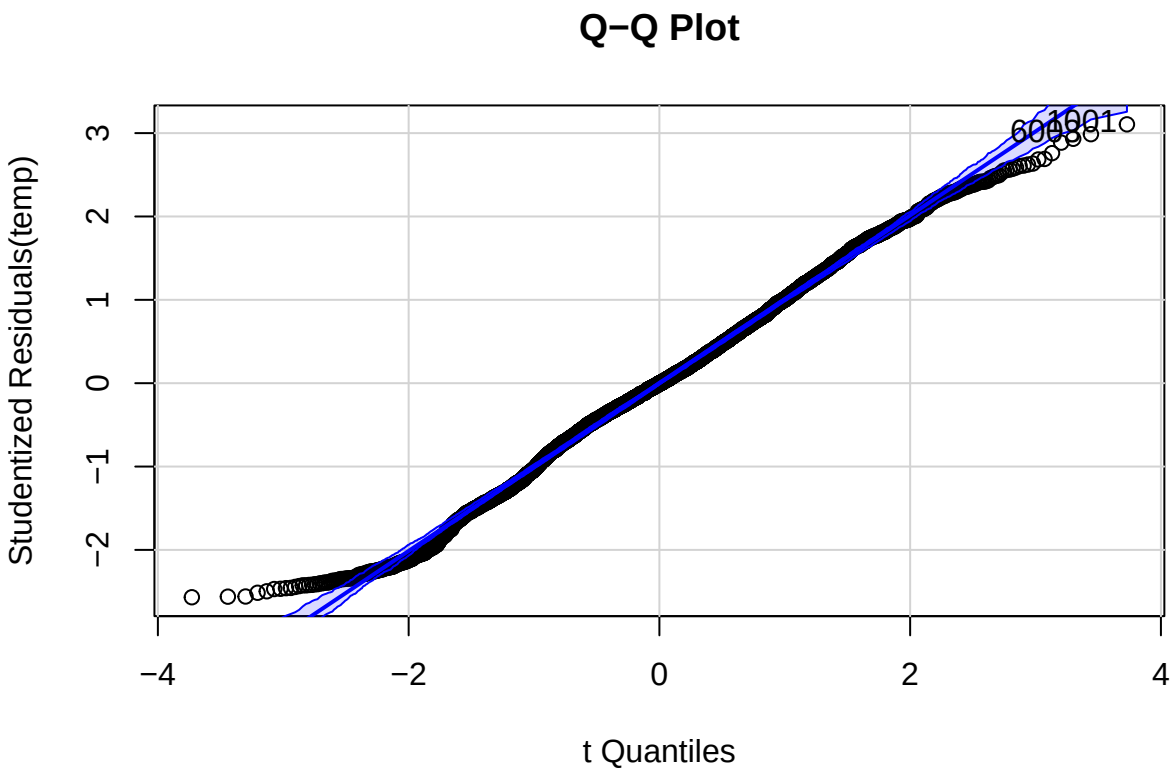
```
##    weekday      period     mean        sd
## 1        1 09:00-12:00 13.32390 0.6622316
## 2        2 09:00-12:00 13.23403 0.7226723
## 3        3 09:00-12:00 13.27263 0.6900872
## 4        4 09:00-12:00 13.23663 0.6766477
## 5        5 09:00-12:00 13.28528 0.6519060
## 6        6 09:00-12:00 13.23205 0.6833446
## 7        7 09:00-12:00 13.31089 0.5865117
## 8        1 12:00-16:00 13.28902 0.6549773
## 9        2 12:00-16:00 13.34766 0.6410711
## 10       3 12:00-16:00 13.20894 0.5919334
## 11       4 12:00-16:00 13.13549 0.5962807
## 12       5 12:00-16:00 13.31278 0.6179567
## 13       6 12:00-16:00 13.39438 0.6696262
## 14       7 12:00-16:00 13.30716 0.6104540
## 15       1 16:00-18:00 13.28663 0.7093024
## 16       2 16:00-18:00 13.27351 0.6024167
## 17       3 16:00-18:00 13.30031 0.5903714
## 18       4 16:00-18:00 13.38521 0.6488555
## 19       5 16:00-18:00 13.36971 0.5795965
## 20       6 16:00-18:00 13.27542 0.6253114
## 21       7 16:00-18:00 13.28314 0.5855681
## 22       1 18:00-20:00 13.22776 0.6025794
## 23       2 18:00-20:00 13.25311 0.5894339
## 24       3 18:00-20:00 13.32006 0.6064573
## 25       4 18:00-20:00 13.34002 0.6563351
## 26       5 18:00-20:00 13.38787 0.5864271
## 27       6 18:00-20:00 13.32416 0.6234101
```

```
## 28       7 18:00-20:00 13.38861 0.6267588
## 29       1  20:00-9:00 13.27164 0.6051419
## 30       2  20:00-9:00 13.18926 0.5784881
## 31       3  20:00-9:00 13.16663 0.5578713
## 32       4  20:00-9:00 13.21557 0.6584988
## 33       5  20:00-9:00 13.28462 0.5804436
## 34       6  20:00-9:00 13.28359 0.6400301
## 35       7  20:00-9:00 13.22808 0.6146903
```

```
### Test for group difference
aov.weekday.period <- aov(log(like) ~ weekday * period)
summary(aov.weekday.period)
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
## weekday            6    3.3  0.5450   1.389  0.215
## period             4    5.0  1.2620   3.216  0.012 *
## weekday:period    24   11.7  0.4865   1.240  0.194
## Residuals       5115 2007.2  0.3924
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
### Normality
temp <- lm(log(like) ~ weekday * period, data = video.noncar)
qqPlot(temp,
       simulate = TRUE,
       main = "Q-Q Plot",
       labels = FALSE)
```



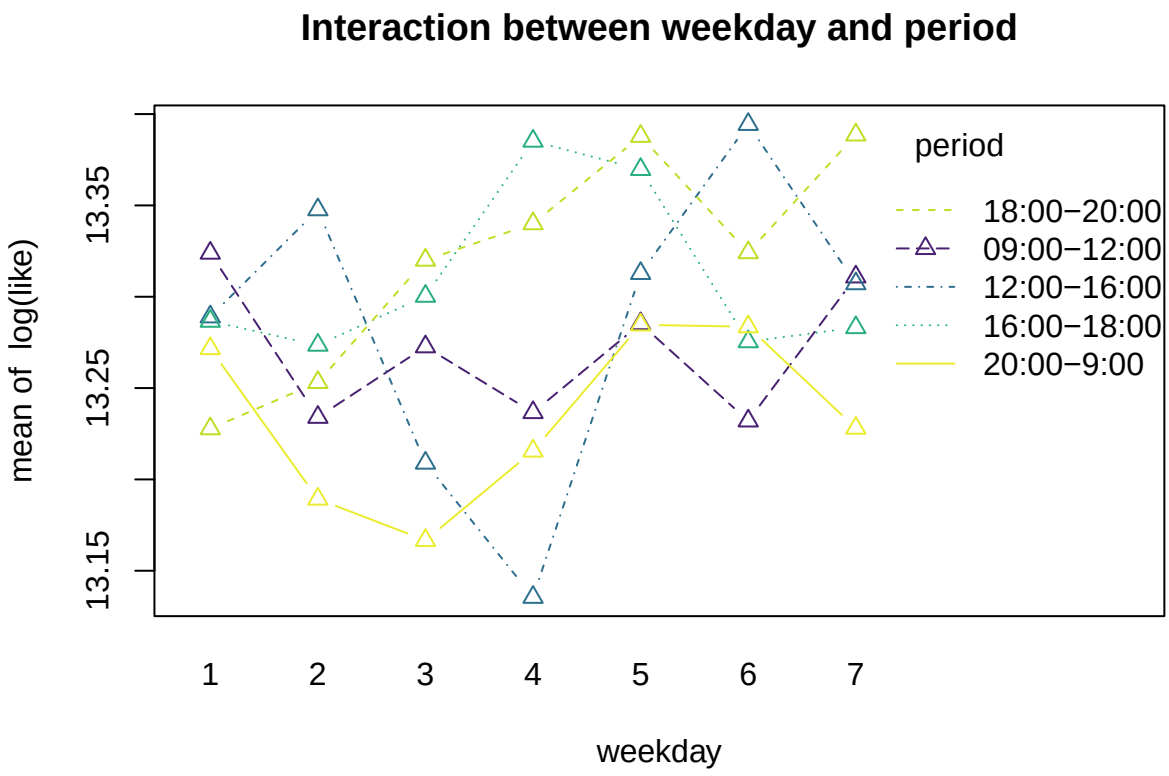

```
## 1001 6003
##  952 4793
```

```
re <- temp$residuals
ks.test(re, "pnorm", mean = mean(re), sd = sd(re))
```

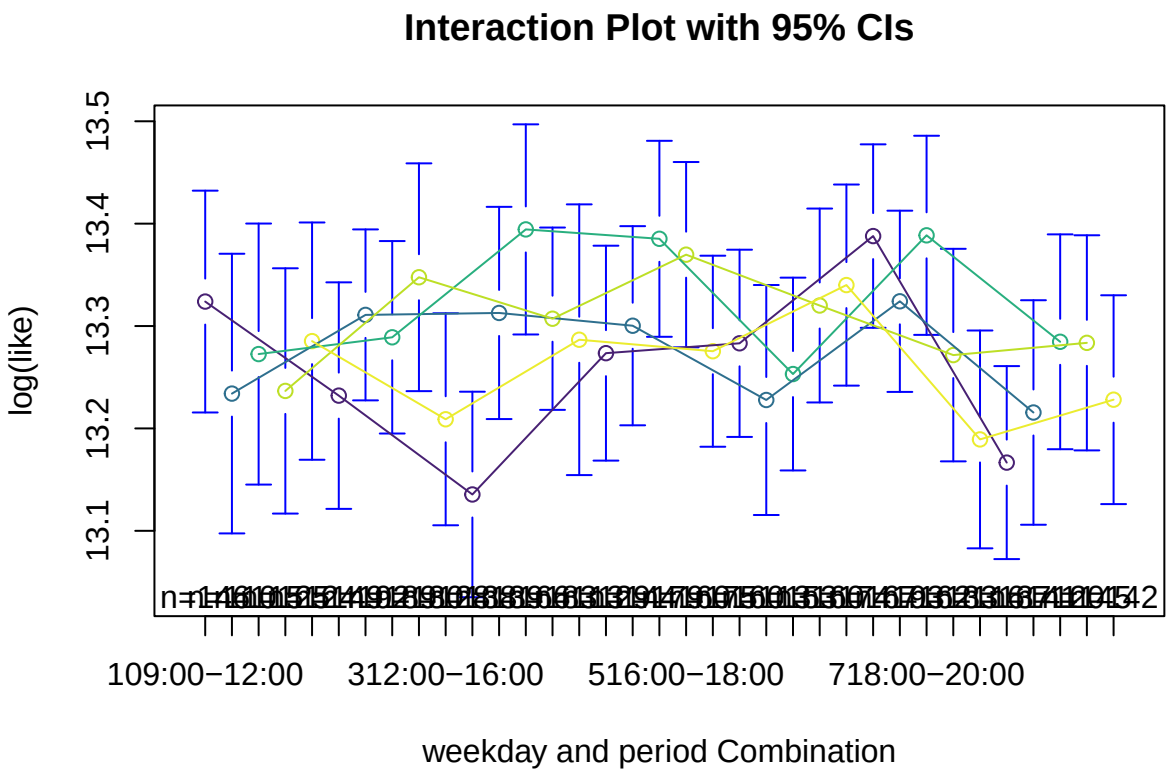
```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  re
## D = 0.023924, p-value = 0.005506
## alternative hypothesis: two-sided
```

The ks.test shows that the normality assumption should be rejected although the Q-Q plot looks good.

```
interaction.plot(
  weekday,
  period,
  log(like),
  type = "b",
  col = c("#482173", "#2e6f8e", "#29af7f", "#bddf26", "#ebec2f"),
  pch = 24,
  main = "Interaction between weekday and period"
)
```




```
library(gplots)
plotmeans(
  log(like) ~ interaction(weekday, period, sep = ""),
  connect = list(
    c(1, 6, 11, 16, 21, 26, 31),
    c(2, 7, 12, 17, 22, 27, 32),
    c(3, 8, 13, 18, 23, 28, 33),
    c(4, 9, 14, 19, 24, 29, 34),
    c(5, 10, 15, 20, 25, 30, 35)
  ),
  col = c("#482173", "#2e6f8e", "#29af7f", "#bddf26", "#ebec2f"),
  main = "Interaction Plot with 95% CIs",
  xlab = "weekday and period Combination"
)
```




```
detach(video.noncar)
```

3.2.4 One-way ANCOVA

```
temp <- video.noncar
temp$log.comment <- log(video.noncar$comment)
temp$log.share <- log(video.noncar$share)
```

- Covariavte is log(comment) (log(share) is similar)

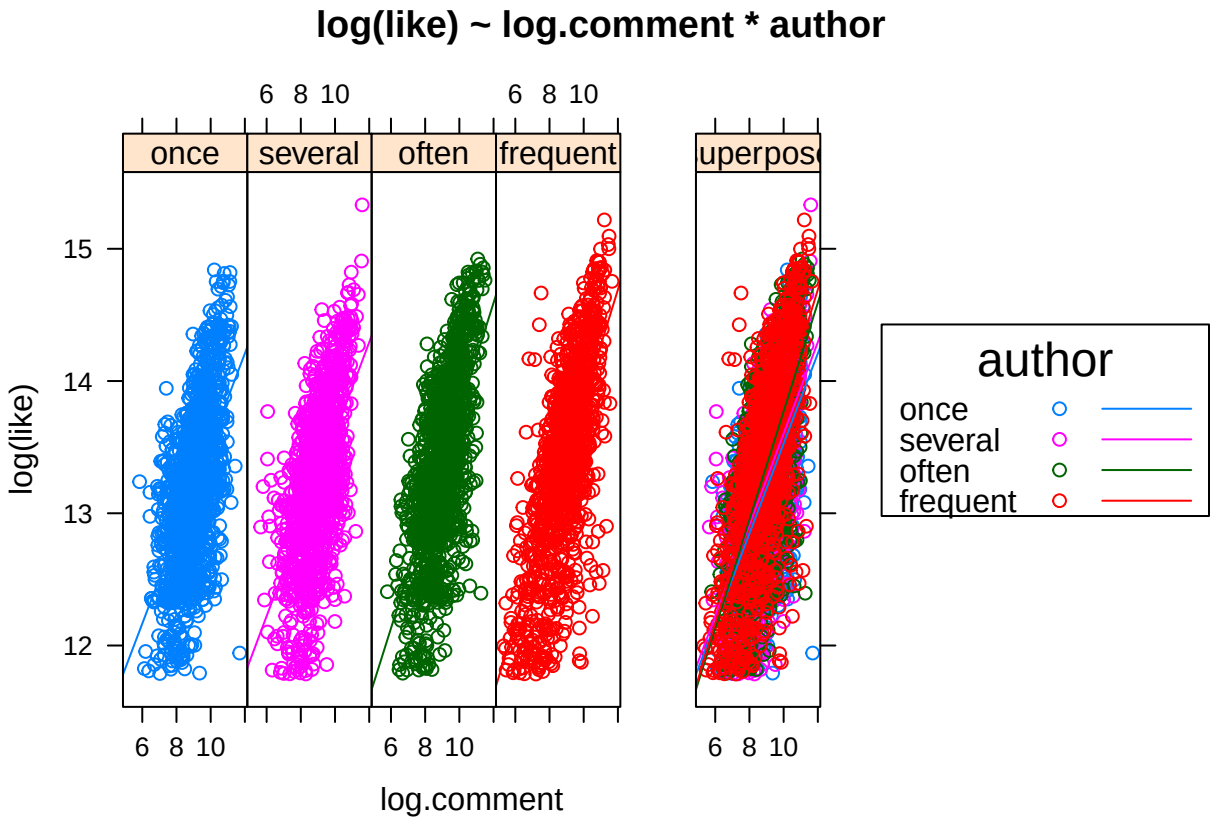
```
## author
library(HH)
```

```
##      grid

##      latticeExtra

##
##    'latticeExtra'

## The following object is masked from 'package:ggplot2':
##
##     layer
```

```
##      multcomp

##      mvtnorm

##      survival

##
##    'survival'

## The following object is masked from 'package:caret':
##
##     cluster

##      TH.data

##
##    'TH.data'

## The following object is masked from 'package:MASS':
##
##     geyser

##      gridExtra

##
##    'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine

##
##    'HH'

## The following object is masked _by_ '.GlobalEnv':
##
##     residplot

## The following object is masked from 'package:gplots':
##
##     residplot

## The following objects are masked from 'package:car':
##
##     logit, vif

## The following object is masked from 'package:lubridate':
##
##     interval
```

```
cova <- ancova(log(like) ~ log.comment * author, data = temp)
cova
```

```
## Analysis of Variance Table
##
## Response: log(like)
##                       Df  Sum Sq Mean Sq  F value    Pr(>F)
## log.comment            1  779.84  779.84 3353.124 < 2.2e-16 ***
## author                 3   44.25   14.75   63.428 < 2.2e-16 ***
## log.comment:author     3    7.26    2.42   10.407  7.98e-07 ***
## Residuals           5142 1195.88    0.23
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
library(effects)
```

```
## Use the command
##     lattice::trellis.par.set(effectsTheme())
##   to customize lattice options for effects plots.
## See ?effectsTheme for details.
```

```
effect("author", cova)
```

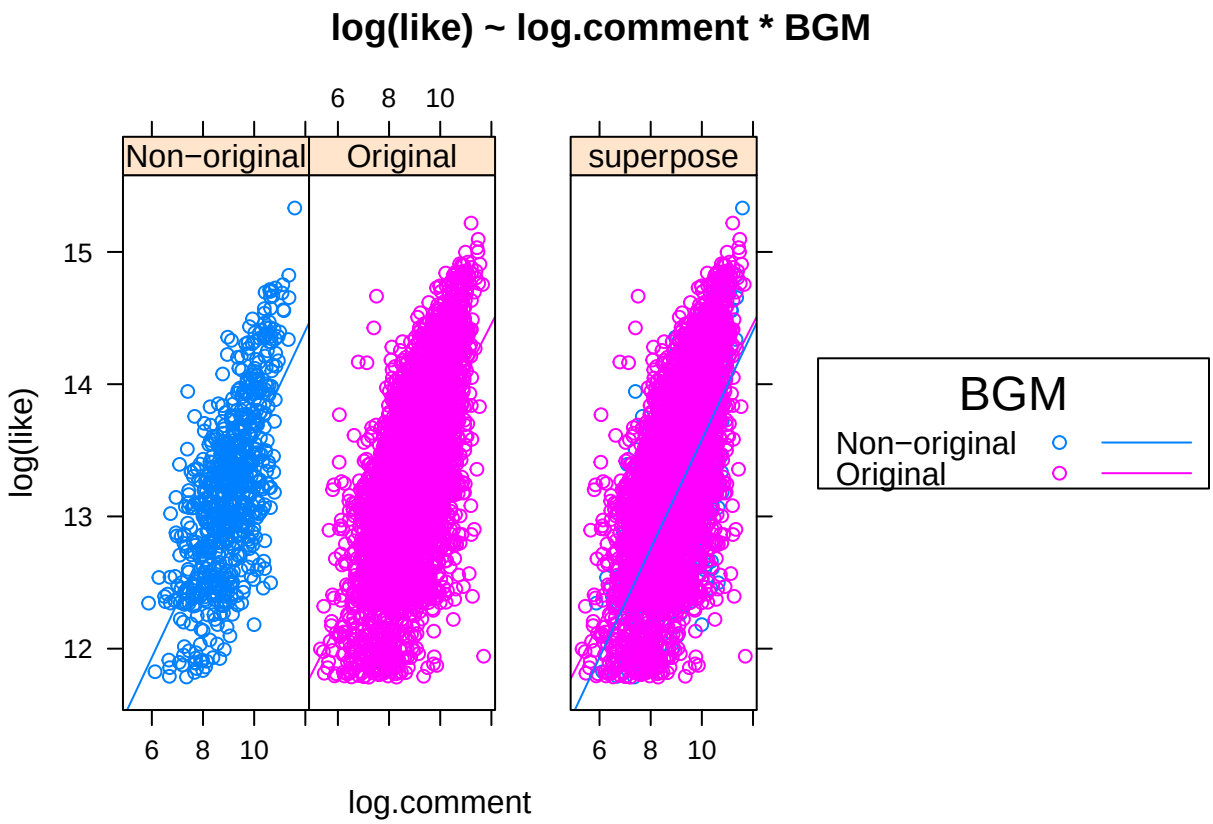
```
## NOTE: author is not a high-order term in the model
```

```
##
##  author effect
## author
##     once  several    often frequent
## 13.17350 13.24389 13.34418 13.41270
```

The effect of 'author'

```
## BGM
library(HH)
cova <- ancova(log(like) ~ log.comment * BGM, data = temp)
cova
```

```
## Analysis of Variance Table
##
## Response: log(like)
##                    Df  Sum Sq Mean Sq   F value    Pr(>F)
## log.comment         1  779.84  779.84 3258.6166 < 2.2e-16 ***
## BGM                 1   15.11   15.11   63.1294  2.36e-15 ***
## log.comment:BGM     1    0.77    0.77    3.2074   0.07337 .
## Residuals        5146 1231.52    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



```
library(effects)
effect("BGM", cova)
```

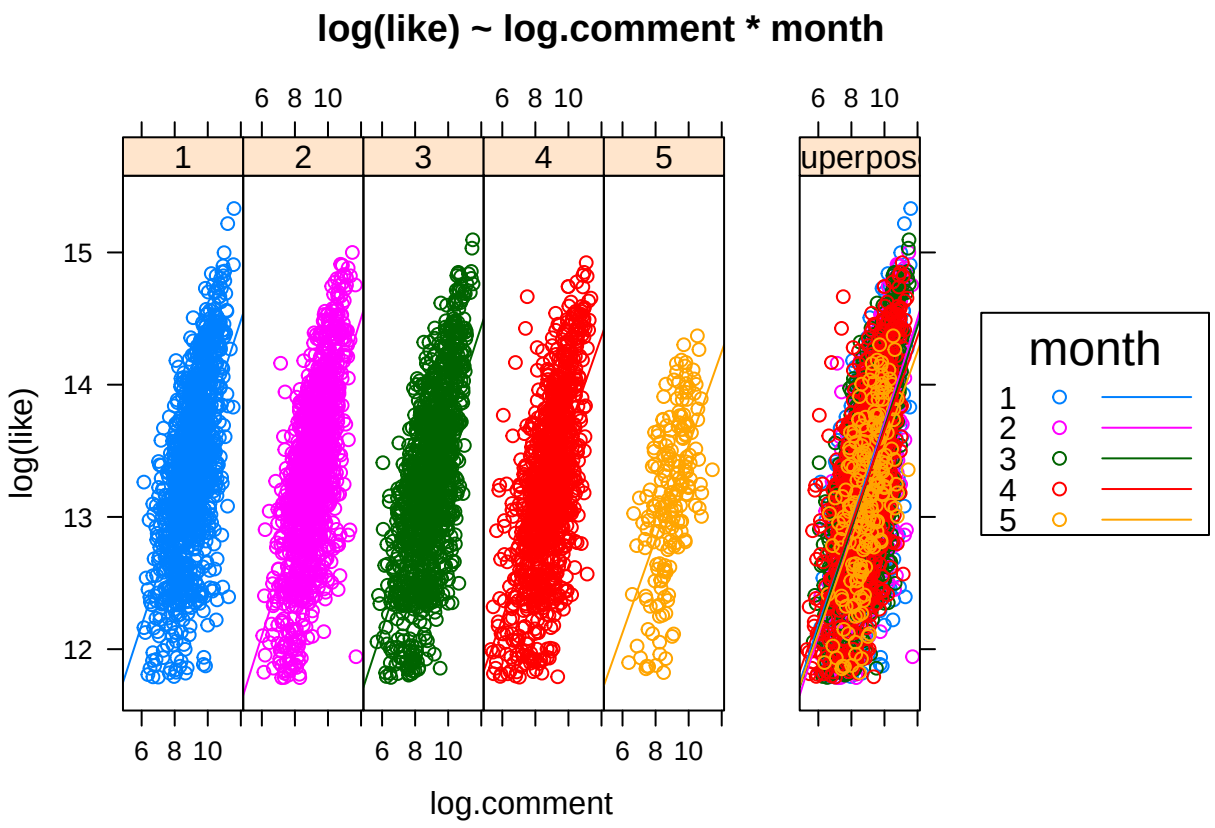
```
## NOTE: BGM is not a high-order term in the model
```

```
##
##  BGM effect
## BGM
## Non-original     Original
##     13.15268     13.31114
```

The effect of 'BGM'

```
## month
library(HH)
cova <- ancova(log(like) ~ log.comment * month, data = temp)
cova
```

```
## Analysis of Variance Table
##
## Response: log(like)
##                      Df  Sum Sq Mean Sq   F value    Pr(>F)
## log.comment           1  779.84  779.84 3227.2003 < 2.2e-16 ***
## month                 4    3.88    0.97    4.0159  0.002964 **
## log.comment:month     4    1.45    0.36    1.5052  0.197763
## Residuals          5140 1242.06    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
library(effects)
effect("month", cova)
```

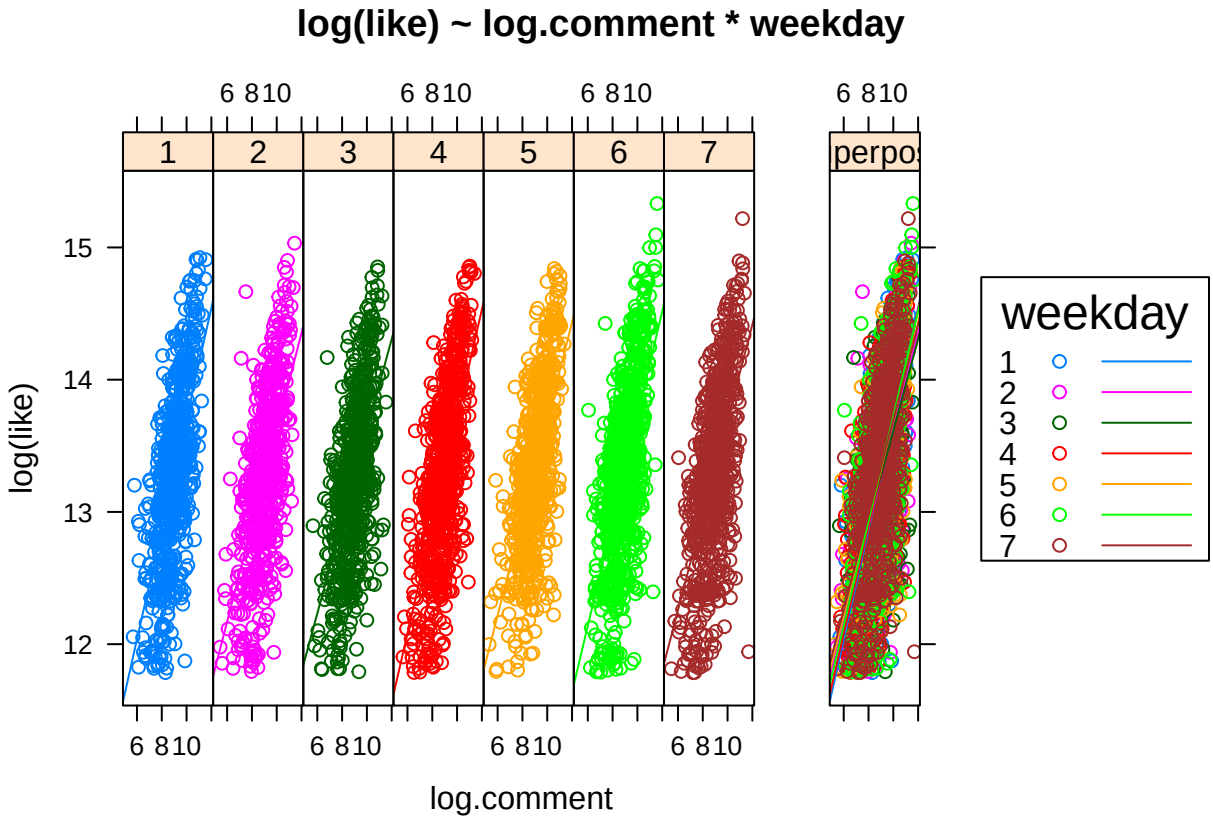
```
## NOTE: month is not a high-order term in the model

##
##  month effect
## month
##        1        2        3        4        5
## 13.31855 13.28686 13.27768 13.28464 13.17344
```

The effect of 'month'

```
## weekday
library(HH)
cova <- ancova(log(like) ~ log.comment * weekday, data = temp)
cova
```

```
## Analysis of Variance Table
##
## Response: log(like)
##                       Df  Sum Sq Mean Sq   F value  Pr(>F)
## log.comment            1  779.84  779.84 3229.2897 < 2e-16 ***
## weekday                6    3.79    0.63    2.6183 0.01551 *
```

```
## log.comment:weekday    6    3.31    0.55    2.2862 0.03312 *
## Residuals           5136 1240.29    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
library(effects)
effect("weekday", cova)
```

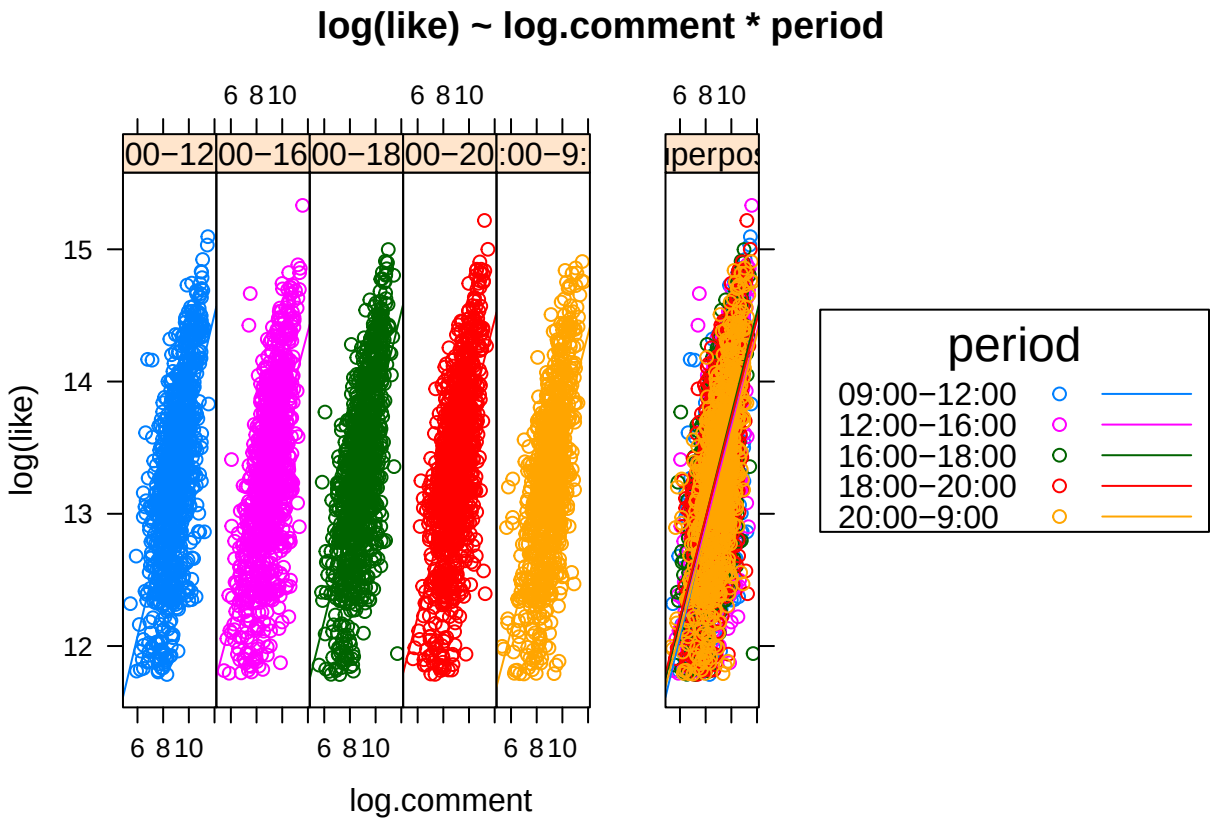
```
## NOTE: weekday is not a high-order term in the model

##
##  weekday effect
## weekday
##        1        2        3        4        5        6        7
## 13.27292 13.24767 13.25610 13.28845 13.31120 13.31253 13.32245
```

The effect of 'weekday'

```
## period
library(HH)
cova <- ancova(log(like) ~ log.comment * period, data = temp)
cova
```

```
## Analysis of Variance Table
```

```
##
## Response: log(like)
##                       Df  Sum Sq Mean Sq   F value    Pr(>F)
## log.comment            1  779.84  779.84 3242.1856 < 2.2e-16 ***
## period                 4   10.01    2.50   10.4075 2.148e-08 ***
## log.comment:period     4    1.06    0.27    1.1061    0.3517
## Residuals           5140 1236.32    0.24
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```




```
library(effects)
effect("period", cova)
```

```
## NOTE: period is not a high-order term in the model

##
##  period effect
## period
## 09:00-12:00 12:00-16:00 16:00-18:00 18:00-20:00  20:00-9:00
##    13.27436    13.26753    13.34410    13.32971    13.22167
```

The effect of 'period'

4 Use nonparametric method to verify linear regression & ANOVA

Motivation: By ks.test(), the normality assumptions for both linear regression and ANOVA are violated.

4.1 Permutation test for linear regression

```
residual <- finalfit$residuals
ks.test(residual,
        "pnorm",
        mean = mean(residual),
        sd = sd(residual))
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  residual
## D = 0.028644, p-value = 8.631e-05
## alternative hypothesis: two-sided
```

Although the fitting effect of the linear model is very good, with an R^2 as high as 82.31%, the model does not pass the normality test.

Therefore, we use non-parametric methods to test the significance of regression coefficients.

```
library(lmPerm)
set.seed(1234)

model_p <-
  lmp(
    log(like) ~ author + log(comment) + log(share) + duration + type +
      title + month + log(comment):iscar + log(share):iscar +
      BGM:iscar + duration:iscar + month:iscar,
    data = video_filted,
    perm = "Prob"
  )
```

```
## [1] "Settings:  unique SS : numeric variables centered"
```

```
summary(model_p)
```

```
##
## Call:
## lmp(formula = log(like) ~ author + log(comment) + log(share) +
##     duration + type + title + month + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + month:iscar, data = video_filted,
##     perm = "Prob")
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.93080 -0.27876 -0.02068  0.25959  1.78591
```

```
##
## Coefficients:
##                      Estimate Iter Pr(Prob)
## author1            -0.091563 5000   <2e-16 ***
## author2            -0.035126 5000   <2e-16 ***
## author3             0.044260 1881   0.0505 .
## log(comment)        0.275092 5000   <2e-16 ***
## log(share)          0.114355 5000   <2e-16 ***
## duration            0.002155 5000   <2e-16 ***
## type1              -0.968064 5000   <2e-16 ***
## type2              -0.228710 5000   <2e-16 ***
## type3              -0.177701 5000   <2e-16 ***
## type4               0.149596  181   0.3591
## type5               0.241603 5000   <2e-16 ***
## type6               0.363717 5000   <2e-16 ***
## title               0.001429 5000   <2e-16 ***
## month1              0.053981 5000   <2e-16 ***
## month2              0.034666 5000   0.0040 **
## month3             -0.023118 5000   0.0028 **
## month4              0.015126 3189   0.0304 *
## log(comment):iscar  0.292480 5000   <2e-16 ***
## log(share):iscar    0.117649 5000   <2e-16 ***
## duration:iscar      0.003547 5000   <2e-16 ***
## month1:iscar        0.165074 5000   <2e-16 ***
## month2:iscar        0.248342 5000   <2e-16 ***
## month3:iscar       -0.069663 2719   0.0357 *
## month4:iscar       -0.152742 5000   <2e-16 ***
## iscar:BGM1         -0.055011 4536   0.0216 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4404 on 6099 degrees of freedom
## Multiple R-Squared: 0.8232,  Adjusted R-squared: 0.8225
## F-statistic:  1136 on 25 and 6099 DF,  p-value: < 2.2e-16
```

```
summary(finalfit)
```

```
##
## Call:
## lm(formula = log(like) ~ author + log(comment) + log(share) +
##     duration + type + title + month + log(comment):iscar + log(share):iscar +
##     BGM:iscar + duration:iscar + month:iscar, data = video_filted)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.93373 -0.27857 -0.02079  0.25991  1.78161
##
## Coefficients:
##                      Estimate Std. Error t value Pr(>|t|)
## (Intercept)         5.5425997  0.0930066  59.594  < 2e-16 ***
## authorseveral       0.0561062  0.0161535   3.473 0.000518 ***
## authoroften         0.1351116  0.0161595   8.361  < 2e-16 ***
## authorfrequent      0.1722682  0.0176477   9.762  < 2e-16 ***
## log(comment)        0.2288701  0.0071810  31.872  < 2e-16 ***
```

```
## log(share)          0.0956691  0.0049258  19.422  < 2e-16 ***
## duration            0.0015401  0.0004211   3.657 0.000257 ***
## typepet             4.3318744  0.1110835  38.997  < 2e-16 ***
## typedress           4.3835230  0.1121871  39.073  < 2e-16 ***
## typemakeup          4.7087887  0.1139618  41.319  < 2e-16 ***
## typefood            4.8017388  0.1145632  41.913  < 2e-16 ***
## typegame            4.9232855  0.1138714  43.235  < 2e-16 ***
## typeplot            5.1784563  0.1186612  43.641  < 2e-16 ***
## title               0.0014432  0.0004203   3.434 0.000600 ***
## month2             -0.0332211  0.0172580  -1.925 0.054280 .
## month3             -0.0405972  0.0176480  -2.300 0.021460 *
## month4              0.0109772  0.0180926   0.607 0.544056
## month5             -0.0791018  0.0340257  -2.325 0.020117 *
## log(comment):iscar  0.2921019  0.0141602  20.628  < 2e-16 ***
## log(share):iscar    0.1174509  0.0101026  11.626  < 2e-16 ***
## iscar:BGMOriginal   0.1057465  0.0417918   2.530 0.011421 *
## duration:iscar      0.0035465  0.0009544   3.716 0.000204 ***
## month2:iscar        0.0833615  0.0502515   1.659 0.097190 .
## month3:iscar       -0.2343296  0.0469170  -4.995 6.06e-07 ***
## month4:iscar       -0.3175441  0.0453502  -7.002 2.79e-12 ***
## month5:iscar       -0.3558016  0.0758462  -4.691 2.78e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4404 on 6099 degrees of freedom
## Multiple R-squared:  0.8232, Adjusted R-squared:  0.8225
## F-statistic:  1136 on 25 and 6099 DF,  p-value: < 2.2e-16
```

Except for the significant difference between the food type and April's coefficient, the significance level of the other coefficients are basically the same.

4.2 Permutation test for ANOVA

- One-way ANOVA permutation test

```
library(lmPerm)
set.seed(1234)
## author
aov_author_p <- aovp(log(like) ~ author, data = video.noncar, perm = "Prob")
```

```
## [1] "Settings:  unique SS "
```

```
summary(aov_author_p)
```

```
## Component 1 :
##               Df R Sum Sq R Mean Sq Iter  Pr(Prob)
## author         3    33.94   11.3148 5000 < 2.2e-16 ***
## Residuals   5146  1993.29    0.3873
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## BGM
aov_bgm_p <- aovp(log(like) ~ BGM, data = video.noncar, perm = "Prob")
```

```
## [1] "Settings:  unique SS "
```

```
summary(aov_bgm_p)
```

```
## Component 1 :
##               Df R Sum Sq R Mean Sq Iter  Pr(Prob)
## BGM            1    10.66   10.6638 5000 < 2.2e-16 ***
## Residuals   5148  2016.57    0.3917
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## weekday
aov_weekday_p <-
  aovp(log(like) ~ weekday, data = video.noncar, perm = "Prob")
```

```
## [1] "Settings:  unique SS "
```

```
summary(aov_weekday_p)
```

```
## Component 1 :
##               Df R Sum Sq R Mean Sq Iter Pr(Prob)
## weekday        6     3.27   0.54505 5000   0.2996
## Residuals   5143  2023.97   0.39354
```

```
## period
aov_period_p <-
  aovp(log(like) ~ period, data = video.noncar, perm = "Prob")
```

```
## [1] "Settings:  unique SS "
```

```
summary(aov_period_p)
```

```
## Component 1 :
##               Df R Sum Sq R Mean Sq Iter  Pr(Prob)
## period         4     5.12   1.28059 5000 < 2.2e-16 ***
## Residuals   5145  2022.11   0.39302
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The results are compatible with conclusions in ANOVA section.

- Two-way ANOVA permutation test

```
library(lmPerm)
set.seed(1234)

## weekday*period
aov_weekday_period_p <-
  aovp(log(like) ~ weekday * period, data = video.noncar, perm = "Prob")
```

```
## [1] "Settings:  unique SS "
```

```
summary(aov_weekday_period_p)
```

```
## Component 1 :
##                  Df R Sum Sq R Mean Sq Iter Pr(Prob)
## weekday           6     3.27   0.54578 5000   0.3998
## period            4     4.61   1.15369 5000   <2e-16 ***
## weekday:period   24    11.68   0.48652 5000   0.5832
## Residuals      5115  2007.24   0.39242
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The same conclusion compared with two-way ANOVA before.

4.3 Bootstrap method

Since the normality assumption is violated, we use bootstrap to calculate some statistics and generate confidence intervals of them.

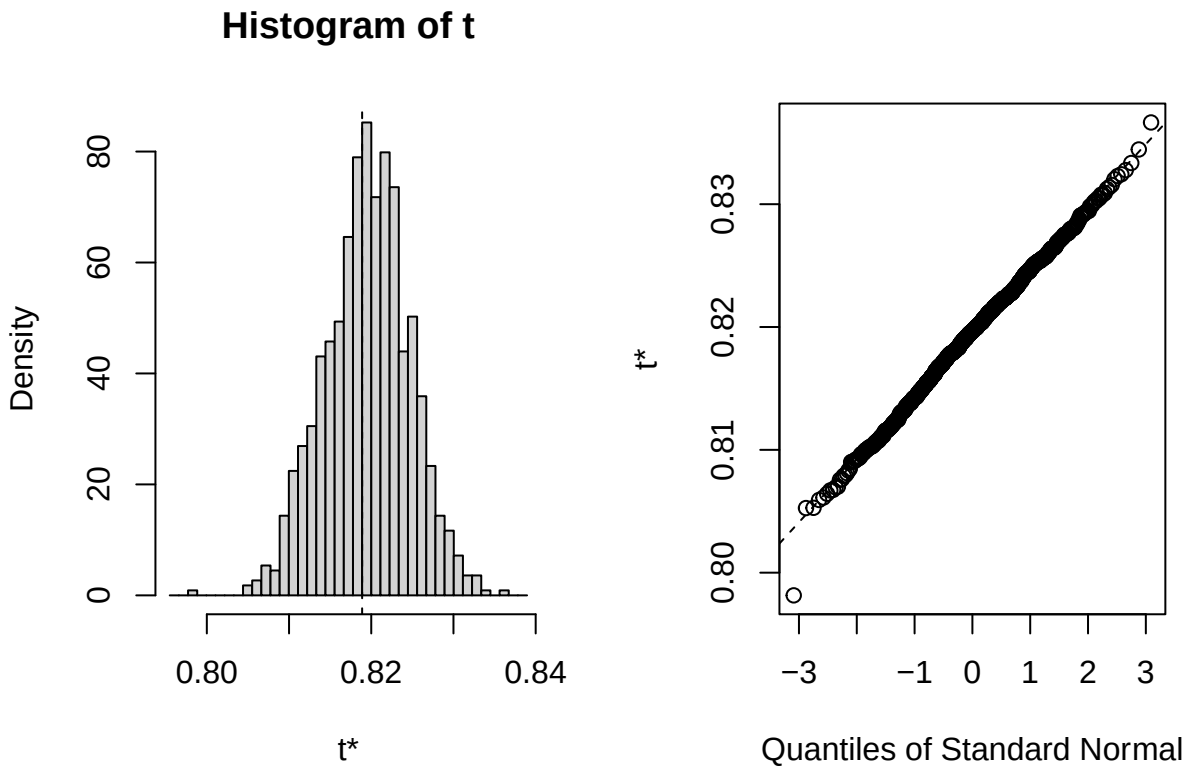
```
library(boot)
```

- Bootstrap for R-squared

```
rsq <- function(formula, data, indices) {
  d <- data[indices, ]
  fit <- lm(formula, data = d)
  return(summary(fit)$r.square)
}

set.seed(1234)
results <-
  boot(
    data = video,
    statistic = rsq,
    R = 1000,
    formula = log(like) ~ author + log(comment) + log(share) + duration +
      type + title + month + log(comment):iscar + log(share):iscar +
      BGM:iscar + duration:iscar + month:iscar
  )
print(results)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = video, statistic = rsq, R = 1000, formula = log(like) ~
##     author + log(comment) + log(share) + duration + type + title +
##         month + log(comment):iscar + log(share):iscar + BGM:iscar +
##         duration:iscar + month:iscar)
##
##
## Bootstrap Statistics :
##      original       bias    std. error
## t1* 0.8188927 0.0006021934 0.005129816
```

```
plot(results)
```




```
boot.ci(results, type = c("perc"))
```

```
## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 1000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = results, type = c("perc"))
##
```

```
## Intervals :
## Level     Percentile
## 95%   ( 0.8093,  0.8293 )
## Calculations and Intervals on Original Scale
```

```
## linear regression R-squared
summary(fit)$r.square
```

```
## [1] 0.8235863
```

The R^2 obtained by our linear regression model is 0.82. Therefore, although the normality assumption is violated for linear regression, the R^2 obtained by linear regression is credible from the bootstrap result. By the bootstrap method, a 95% confidence interval of R^2 is given above.

- Bootstrap for regression coefficients

```
bs <- function(formula, data, indices) {
  d <- data[indices,]
  fit <- lm(formula, data = d)
  return(coef(fit))
}

library(boot)
set.seed(1234)
results <-
  boot(
    data = video,
    statistic = bs,
    R = 1000,
    formula = log(like) ~ author + log(comment) + log(share) + duration +
      type + title + month + log(comment):iscar + log(share):iscar +
    BGM:iscar + duration:iscar + month:iscar
  )

print(results)
```

```
##
## ORDINARY NONPARAMETRIC BOOTSTRAP
##
##
## Call:
## boot(data = video, statistic = bs, R = 1000, formula = log(like) ~
##     author + log(comment) + log(share) + duration + type + title +
##         month + log(comment):iscar + log(share):iscar + BGM:iscar +
##         duration:iscar + month:iscar)
##
##
## Bootstrap Statistics :
##          original        bias     std. error
## t1*   5.692049162 -3.316030e-03 0.1526192051
## t2*   0.055110179 -2.967679e-04 0.0156843828
## t3*   0.130400128 -4.947452e-04 0.0159518150
```

```
## t4*   0.167924673 -2.174634e-04 0.0177992895
## t5*   0.228926767  2.691741e-04 0.0069021204
## t6*   0.095684796 -1.967455e-04 0.0043657547
## t7*   0.001576306 -4.845218e-06 0.0003688772
## t8*   4.182236031  3.740057e-03 0.1663421393
## t9*   4.233215051  3.346773e-03 0.1669689328
## t10*  4.558850362  3.653672e-03 0.1678071218
## t11*  4.651048100  3.791256e-03 0.1689880483
## t12*  4.772730744  3.620341e-03 0.1679911193
## t13*  5.028851035  4.196014e-03 0.1699349538
## t14*  0.001496700  7.796168e-06 0.0004372416
## t15* -0.033099120 -1.088155e-03 0.0146531444
## t16* -0.040411029 -6.362052e-04 0.0156379459
## t17*  0.010830303 -9.950547e-04 0.0164380934
## t18* -0.079149685 -2.076353e-03 0.0300565422
## t19*  0.280007829 -1.722951e-04 0.0216785794
## t20*  0.111569286  2.106820e-04 0.0149282934
## t21*  0.111443911  3.629749e-04 0.0684351779
## t22*  0.003363174  4.584507e-05 0.0013823542
## t23*  0.071812036  4.327382e-03 0.0762219179
## t24* -0.229087371  2.179171e-03 0.0743591507
## t25* -0.320664460 -1.897155e-05 0.0688469116
## t26* -0.369247886 -1.577552e-03 0.0984369394
```

Comparison: (linear regression & bootstrap)

```
## estimated value
coef <- data.frame(summary(finalfit)$coefficients[, 1], results$t[1000,])
colnames(coef) <- c("linear regression", "bootstrap")
round(coef,10)
```

```
##                    linear regression    bootstrap
## (Intercept)              5.542599695  5.699335698
## authorseveral            0.056106195  0.071073288
## authoroften              0.135111588  0.107554304
## authorfrequent           0.172268242  0.143493814
## log(comment)             0.228870098  0.225584028
## log(share)               0.095669082  0.093110542
## duration                 0.001540102  0.002298377
## typepet                  4.331874422  4.212197800
## typedress                4.383523009  4.270610363
## typemakeup               4.708788749  4.602260703
## typefood                 4.801738812  4.697243420
## typegame                 4.923285491  4.816307259
## typeplot                 5.178456313  5.074222883
## title                    0.001443155  0.001180056
## month2                  -0.033221102 -0.030777121
## month3                  -0.040597206 -0.058623092
## month4                   0.010977232 -0.000799236
## month5                  -0.079101800 -0.009959729
## log(comment):iscar       0.292101885  0.279096090
## log(share):iscar         0.117450856  0.122201897
## iscar:BGMOriginal        0.105746479  0.145349137
```

```
## duration:iscar           0.003546472  0.003121935
## month2:iscar             0.083361504 -0.054247292
## month3:iscar            -0.234329582 -0.261277822
## month4:iscar            -0.317544068 -0.427411191
## month5:iscar            -0.355801602 -0.588706650
```

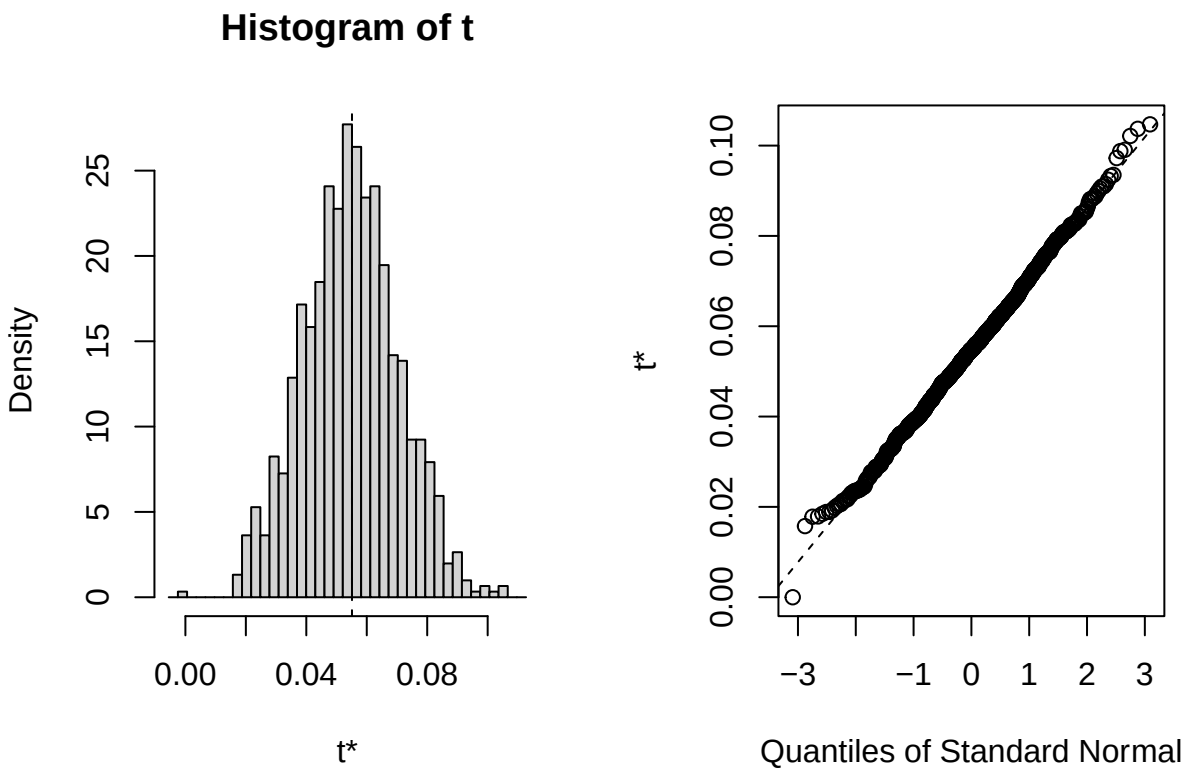
Surprisingly similar!

```
## confidence interval
ci <- data.frame(confint(finalfit), confint(results))
colnames(ci) <-
  c("linear 2.5%", "linear 97.5%", "boot 2.5%", "boot 97.5%")
ci
```

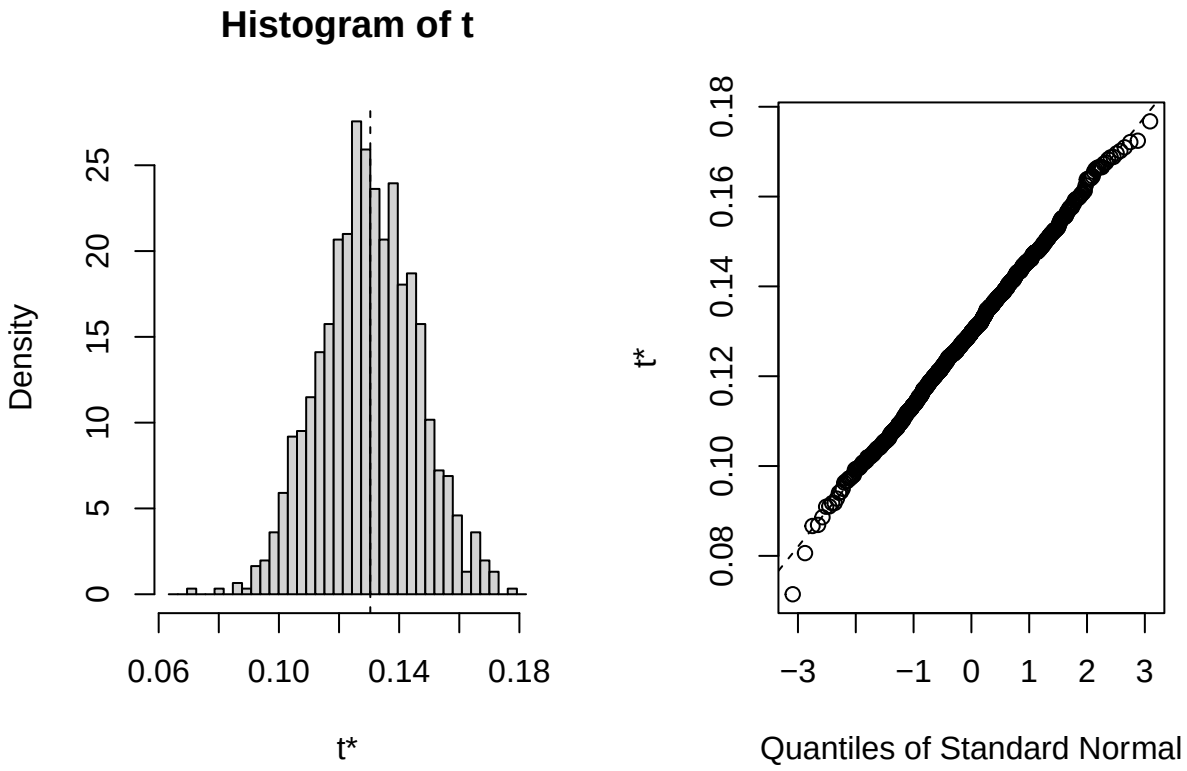
```
##                       linear 2.5%  linear 97.5%     boot 2.5%   boot 97.5%
## (Intercept)         5.3602739272  5.7249254629  5.3880442976  5.986697803
## authorseveral       0.0244396246  0.0877727648  0.0236931918  0.085263076
## authoroften         0.1034333031  0.1667898722  0.0994482320  0.161291931
## authorfrequent      0.1376725867  0.2068638962  0.1312556815  0.201524609
## log(comment)        0.2147928095  0.2429473856  0.2159425875  0.242946489
## log(share)          0.0860128442  0.1053253191  0.0869854315  0.104451441
## duration            0.0007145148  0.0023656901  0.0008474287  0.002326987
## typepet             4.1141114848  4.5496373597  3.8432433915  4.510979767
## typedress           4.1635966091  4.6034494081  3.8954511182  4.567229334
## typemakeup          4.4853833733  4.9321941242  4.2158621785  4.887425397
## typefood            4.5771545742  5.0263230491  4.3107246317  4.990009313
## typegame            4.7000572961  5.1465136852  4.4329006135  5.106675772
## typeplot            4.9458384327  5.4110741926  4.6881315389  5.371562020
## title               0.0006192028  0.0022671065  0.0006227720  0.002354211
## month2             -0.0670528354  0.0006106314 -0.0624149135 -0.005703125
## month3             -0.0751935123 -0.0060008988 -0.0717921604 -0.010766954
## month4             -0.0244906761  0.0464451397 -0.0219764945  0.040785889
## month5             -0.1458042258 -0.0123993743 -0.1389350792 -0.022998587
## log(comment):iscar  0.2643429092  0.3198608598  0.2380191569  0.322162180
## log(share):iscar    0.0976461814  0.1372555295  0.0830855174  0.142004380
## iscar:BGMOriginal   0.0238197145  0.1876732440 -0.0259495624  0.247969940
## duration:iscar      0.0016754396  0.0054175047  0.0006894803  0.006086666
## month2:iscar       -0.0151492466  0.1818722548 -0.0719295496  0.229786465
## month3:iscar       -0.3263035021 -0.1423556622 -0.3788555190 -0.089257222
## month4:iscar       -0.4064464409 -0.2286416945 -0.4551869287 -0.176010501
## month5:iscar       -0.5044868931 -0.2071163102 -0.5656077990 -0.179754649
```

Almost the same!

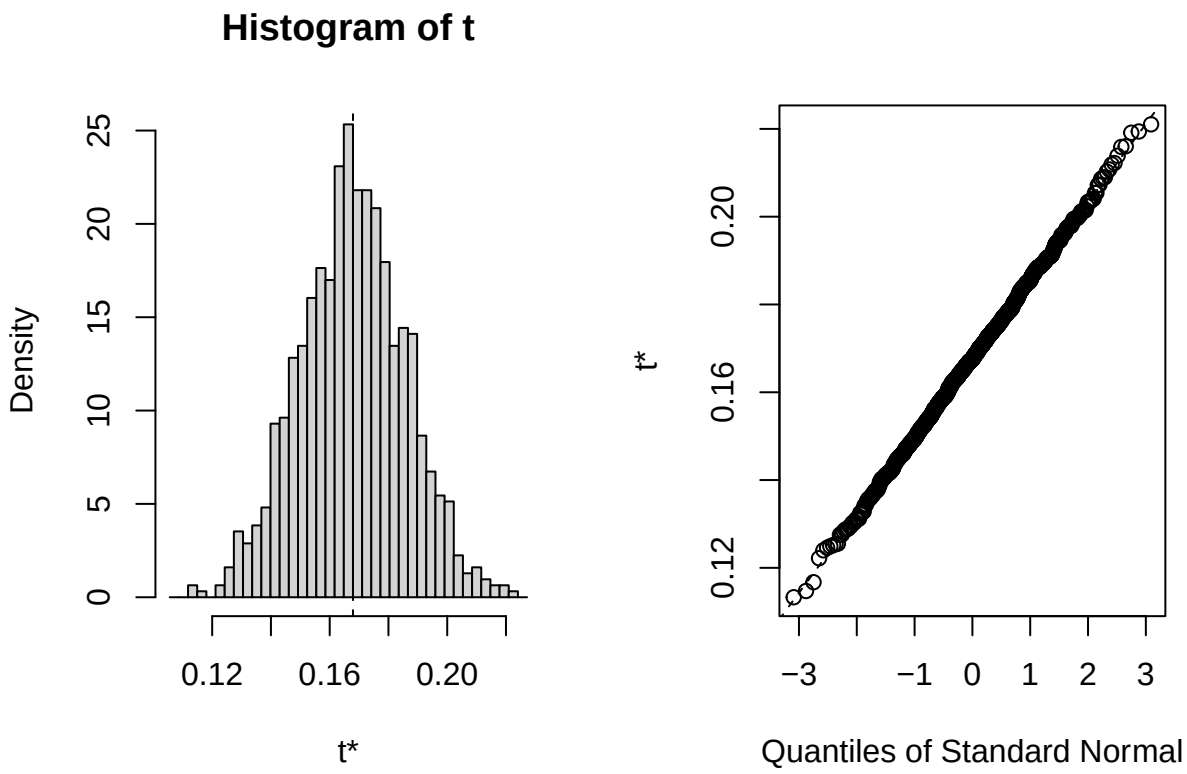
```
## authorseveral
plot(results, index = 2)
```



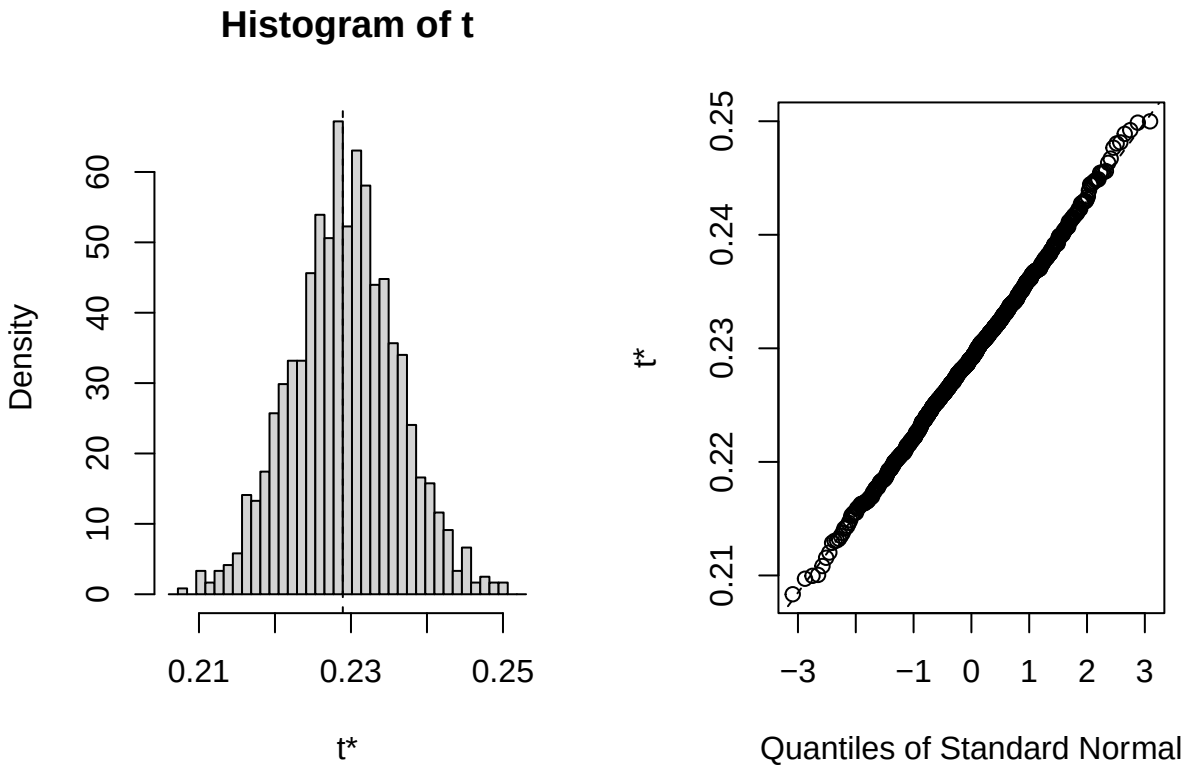

```
## authoroften
plot(results, index = 3)
```



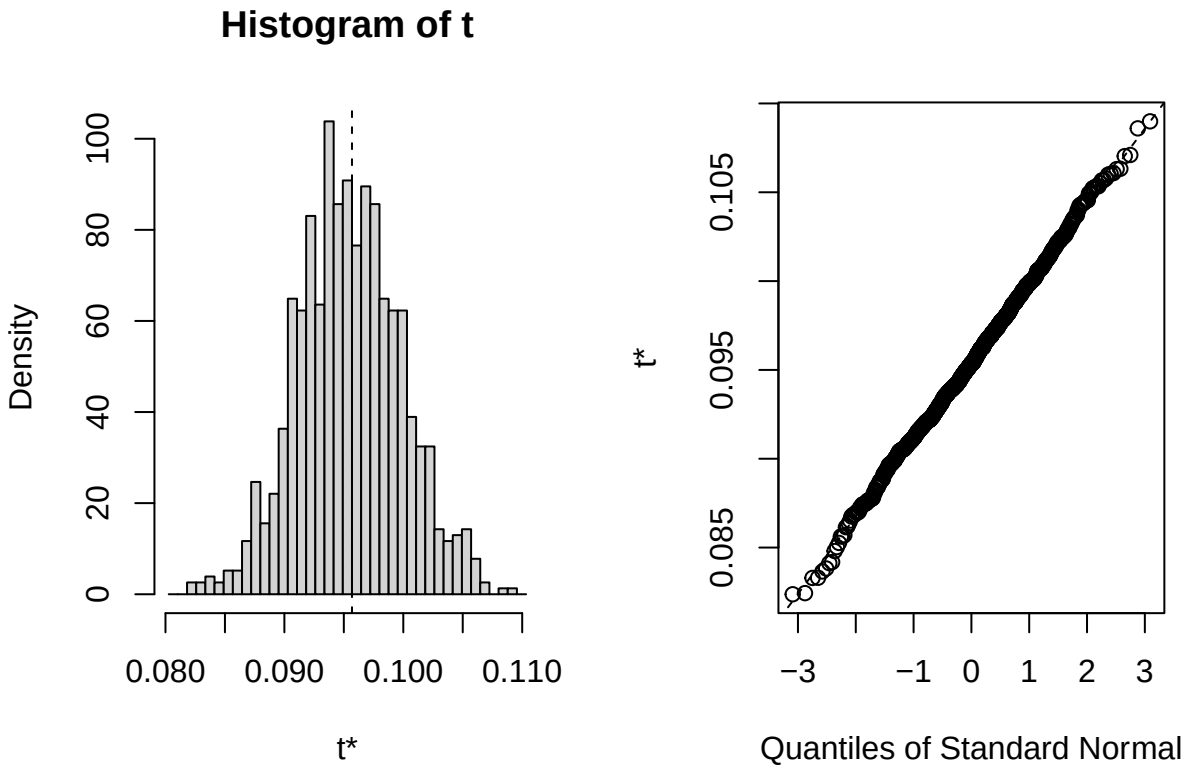

```
## authorfrequent
plot(results, index = 4)
```



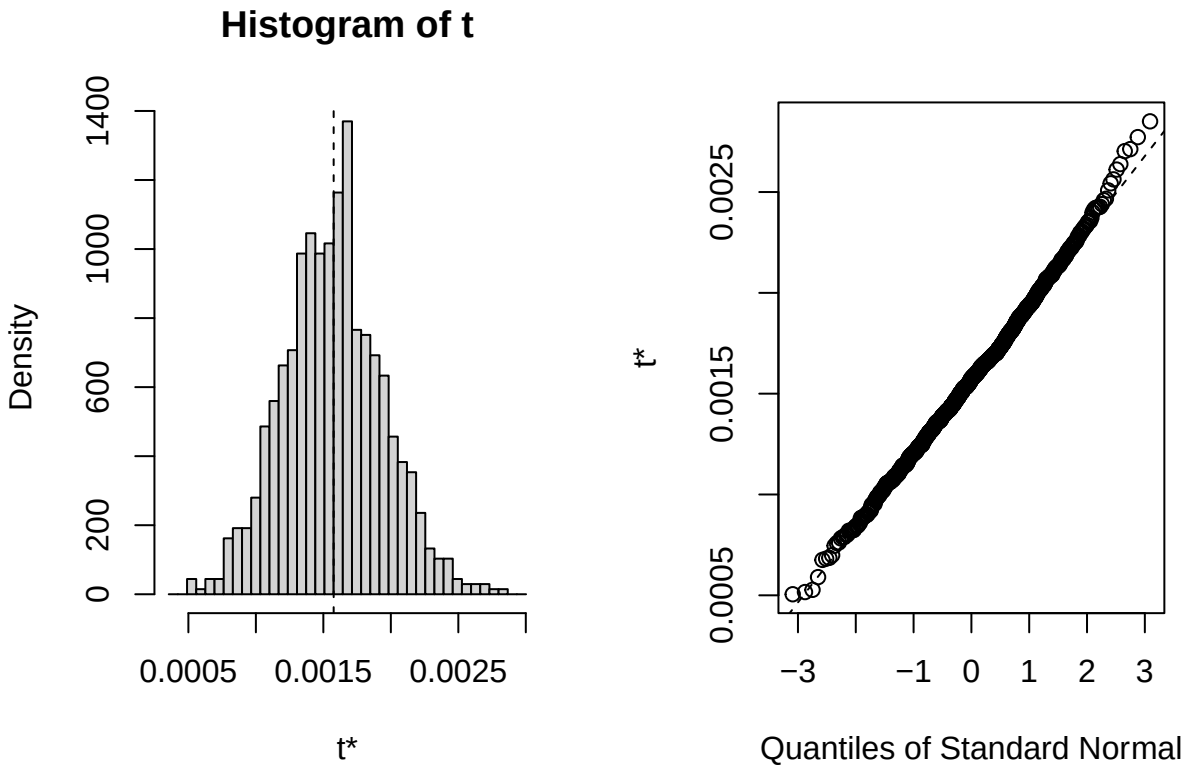

```
## log(comment)
plot(results, index = 5)
```



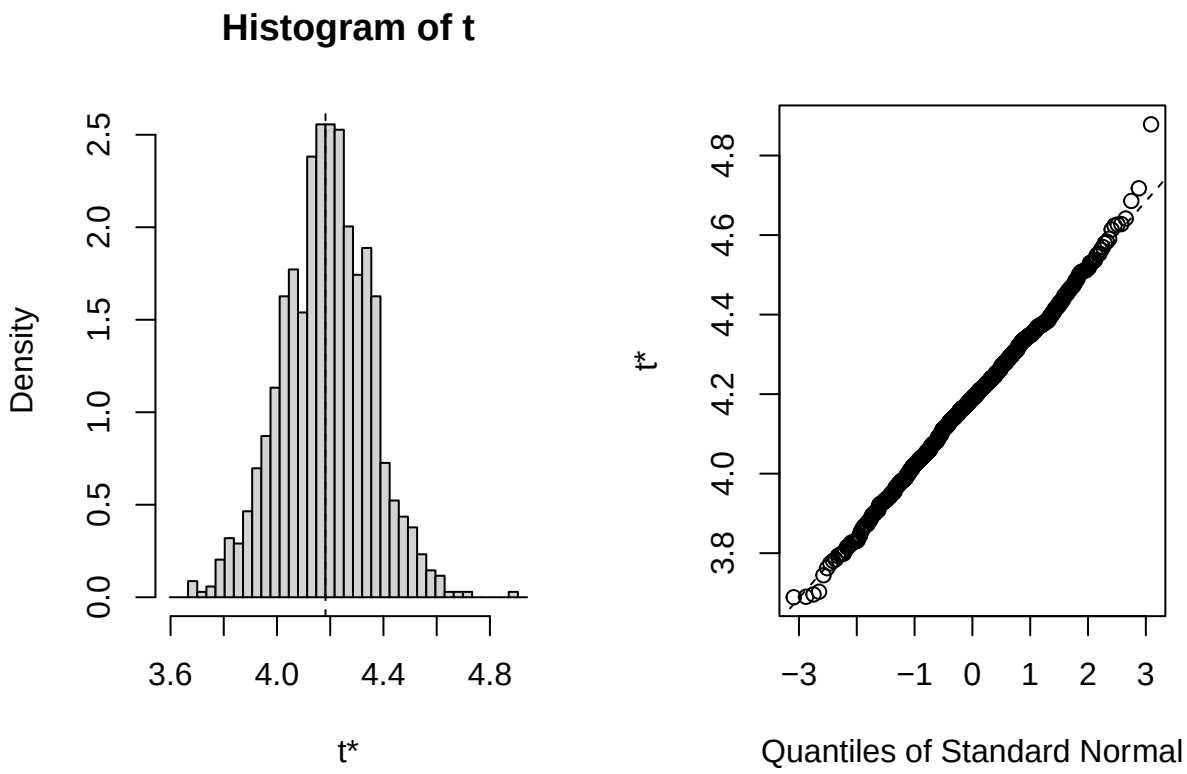

```
## log(share)
plot(results, index = 6)
```



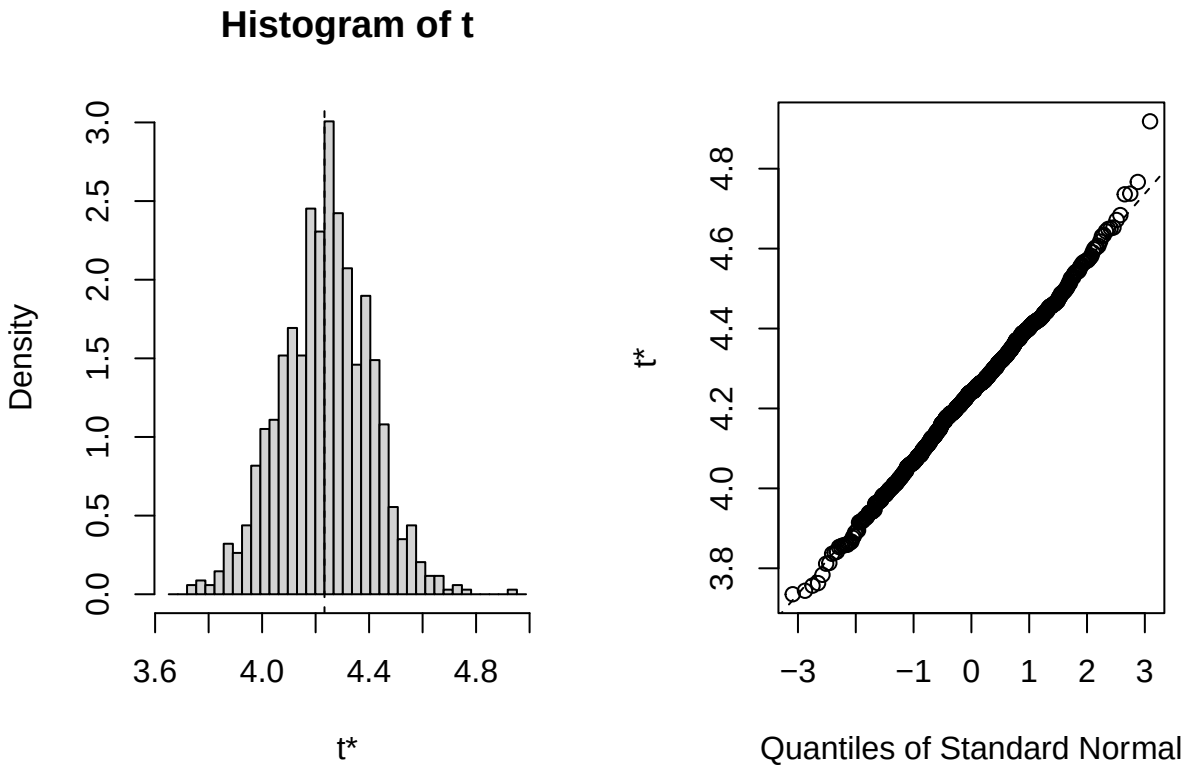

```
## duration
plot(results, index = 7)
```



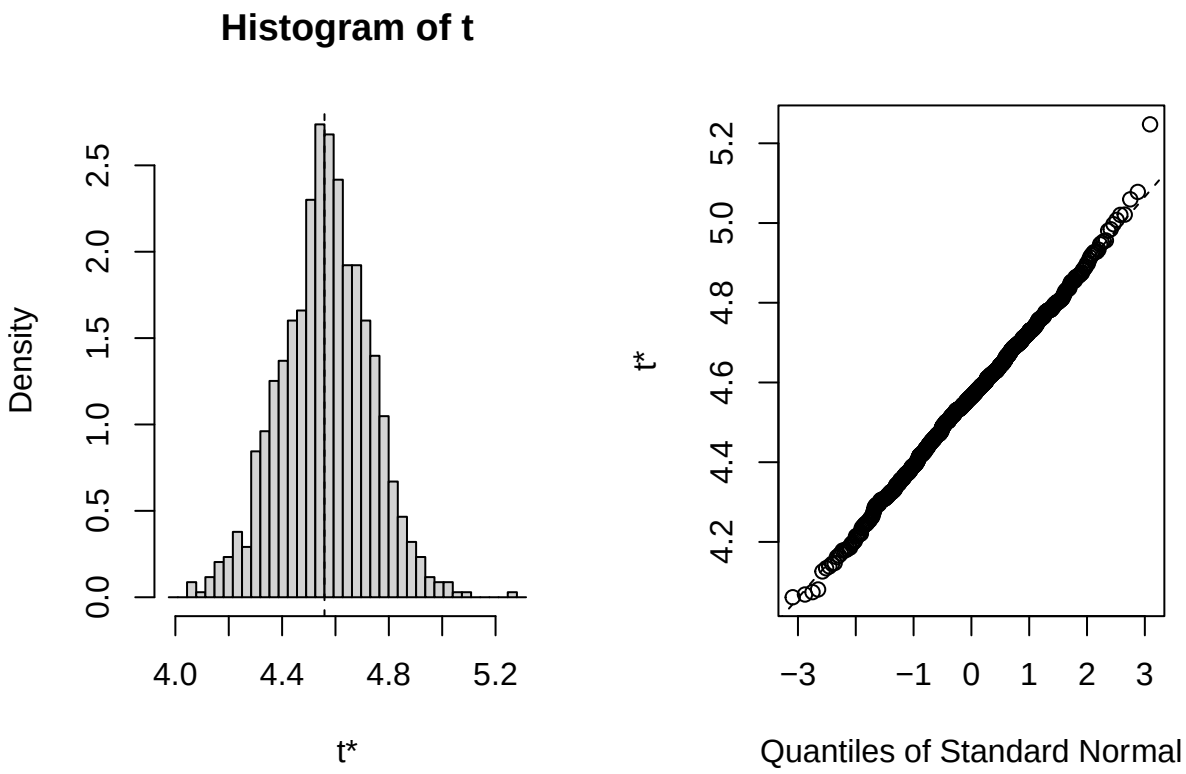

```
## typemakeup
plot(results, index = 8)
```



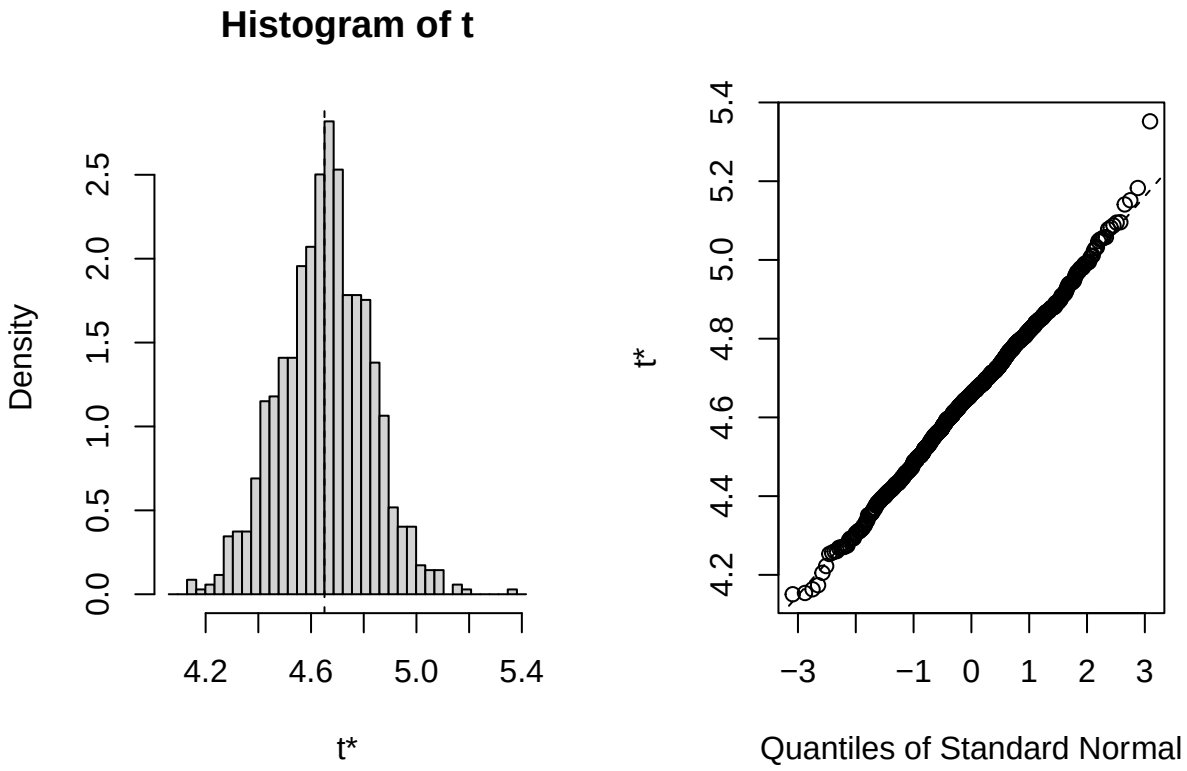

```
## typegame
plot(results, index = 9)
```



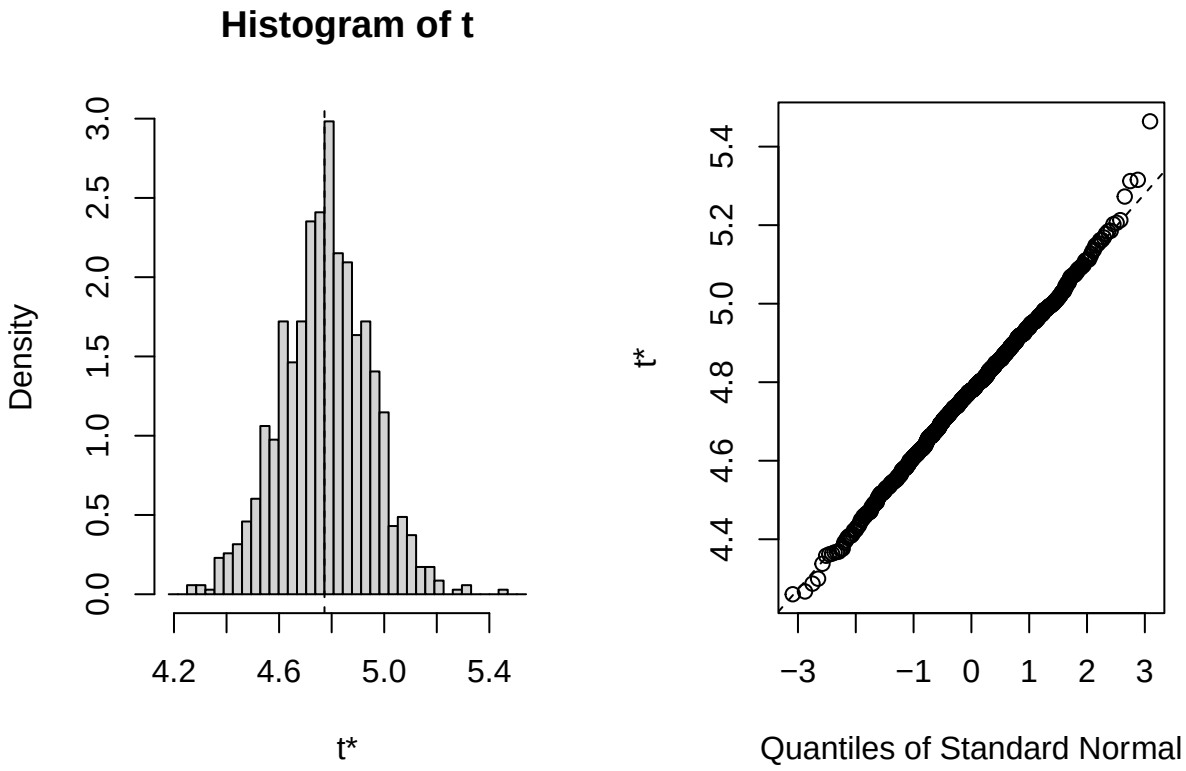

```
## typedress
plot(results, index = 10)
```



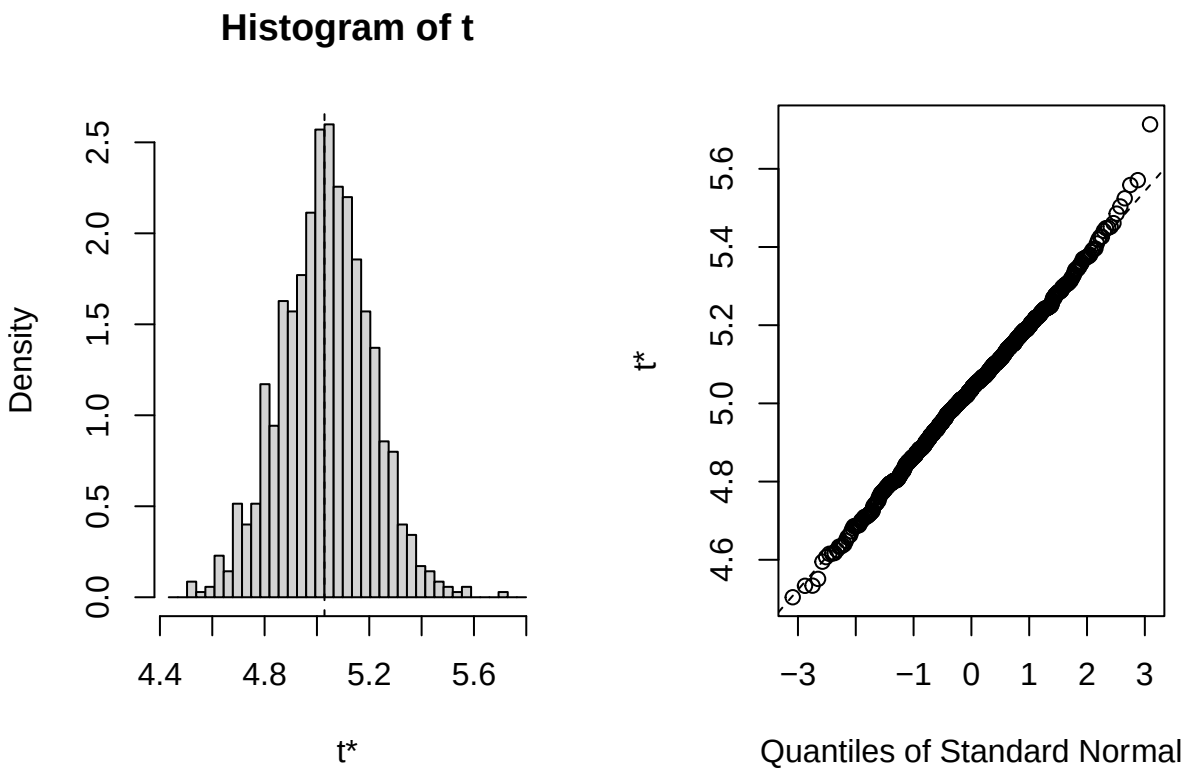

```
## typepet
plot(results, index = 11)
```



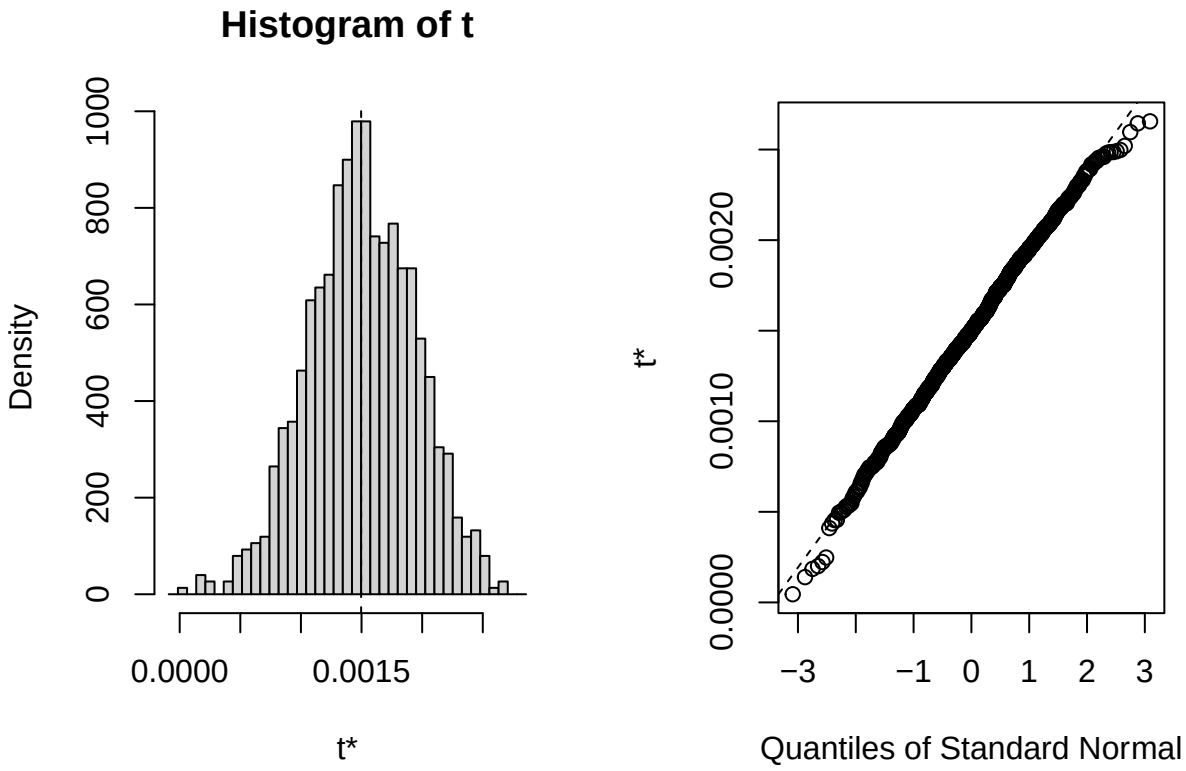

```
## typecar
plot(results, index = 12)
```



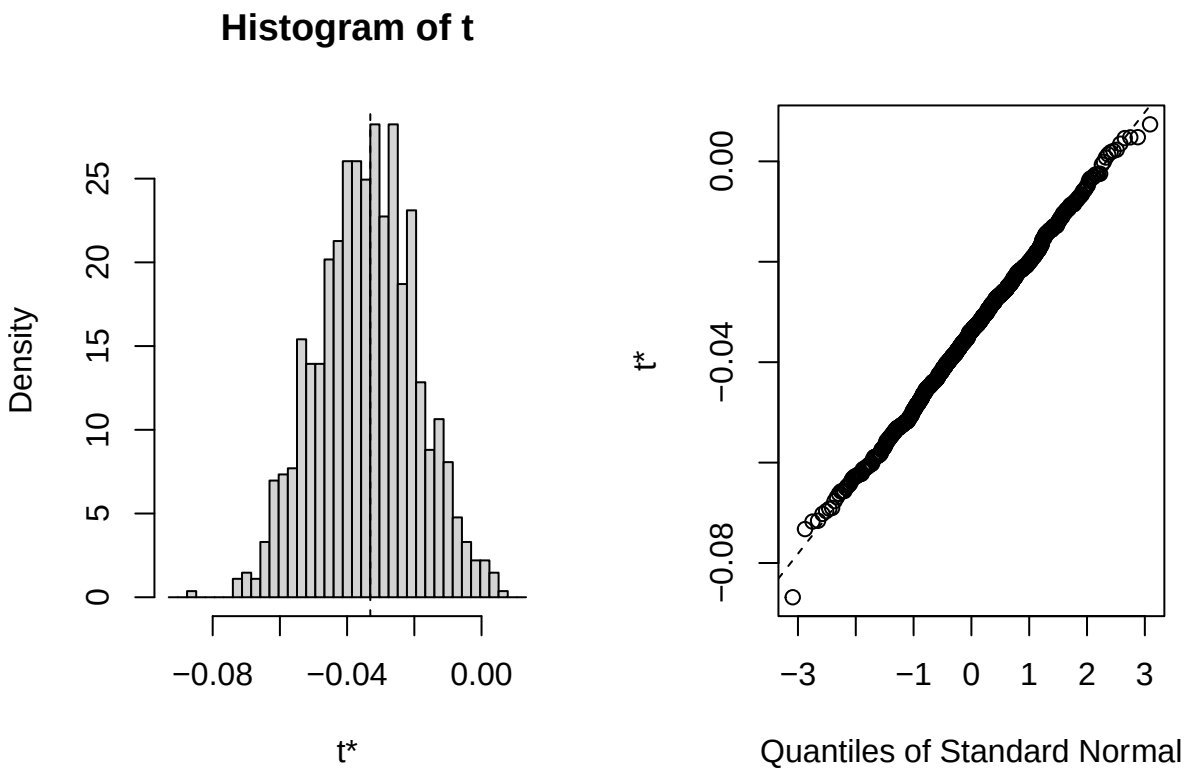

```
## typeplot
plot(results, index = 13)
```



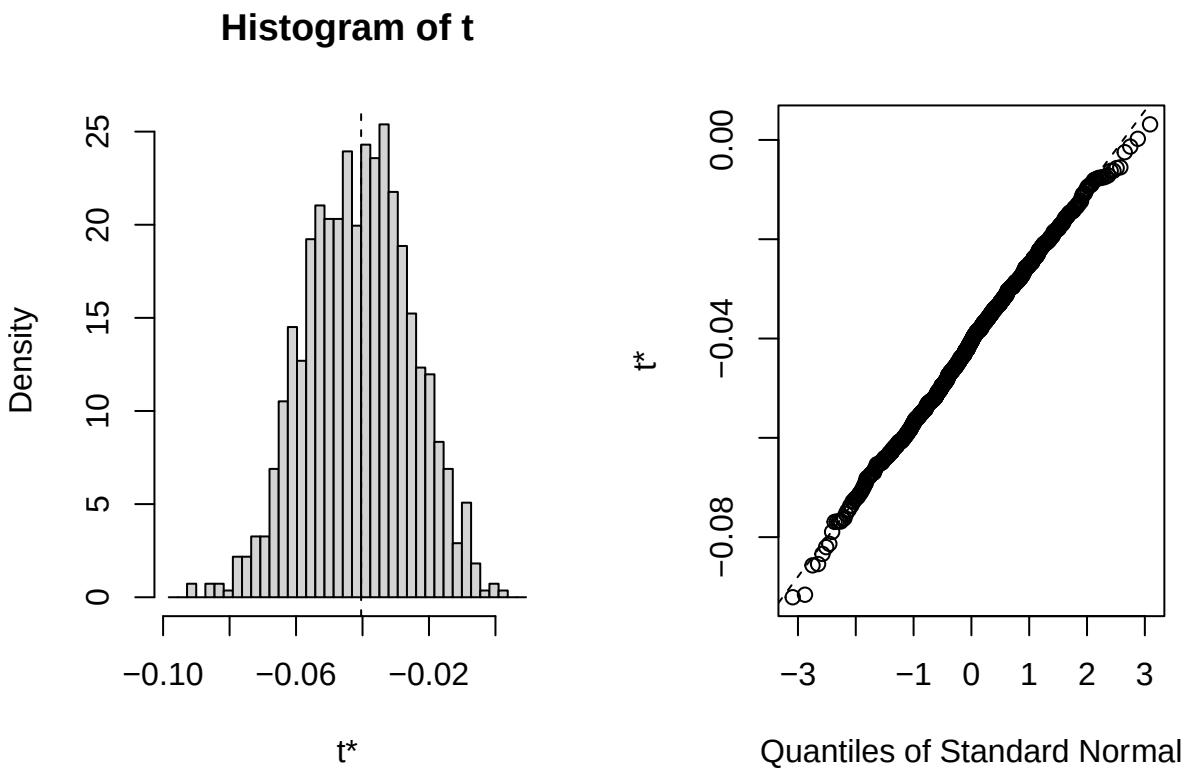

```
## title
plot(results, index = 14)
```



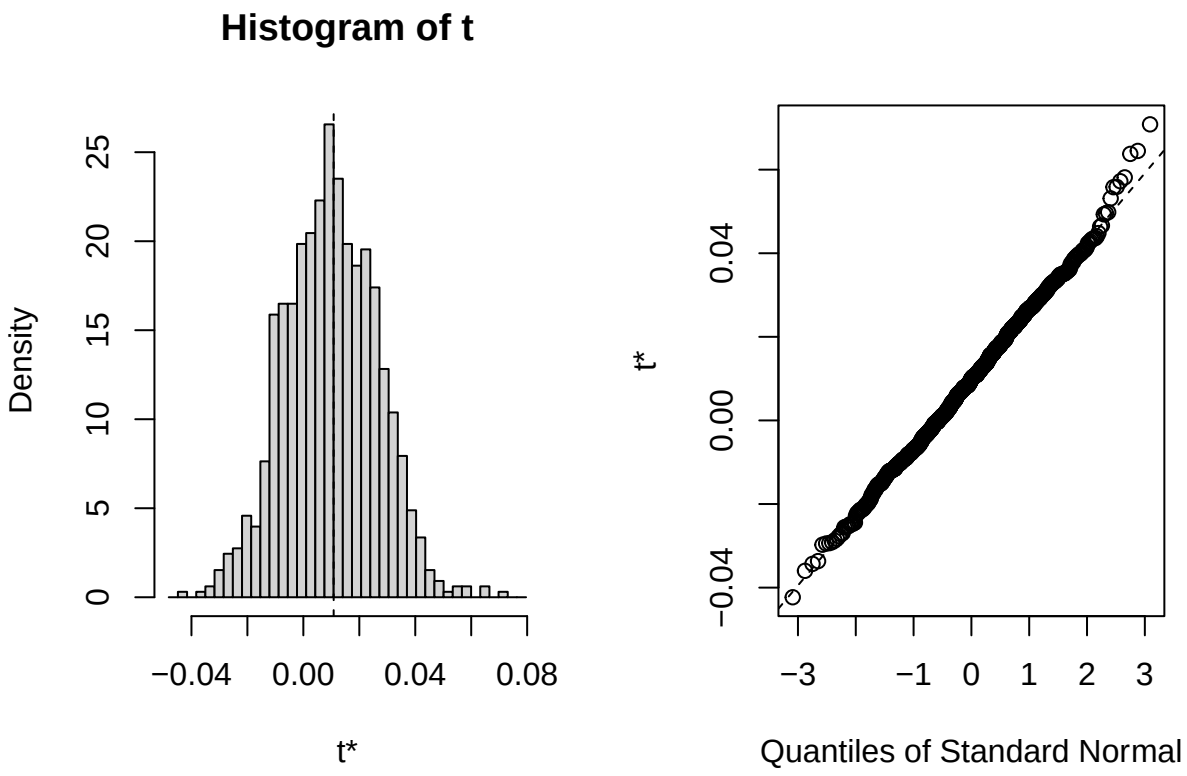

```
## month2
plot(results, index = 15)
```



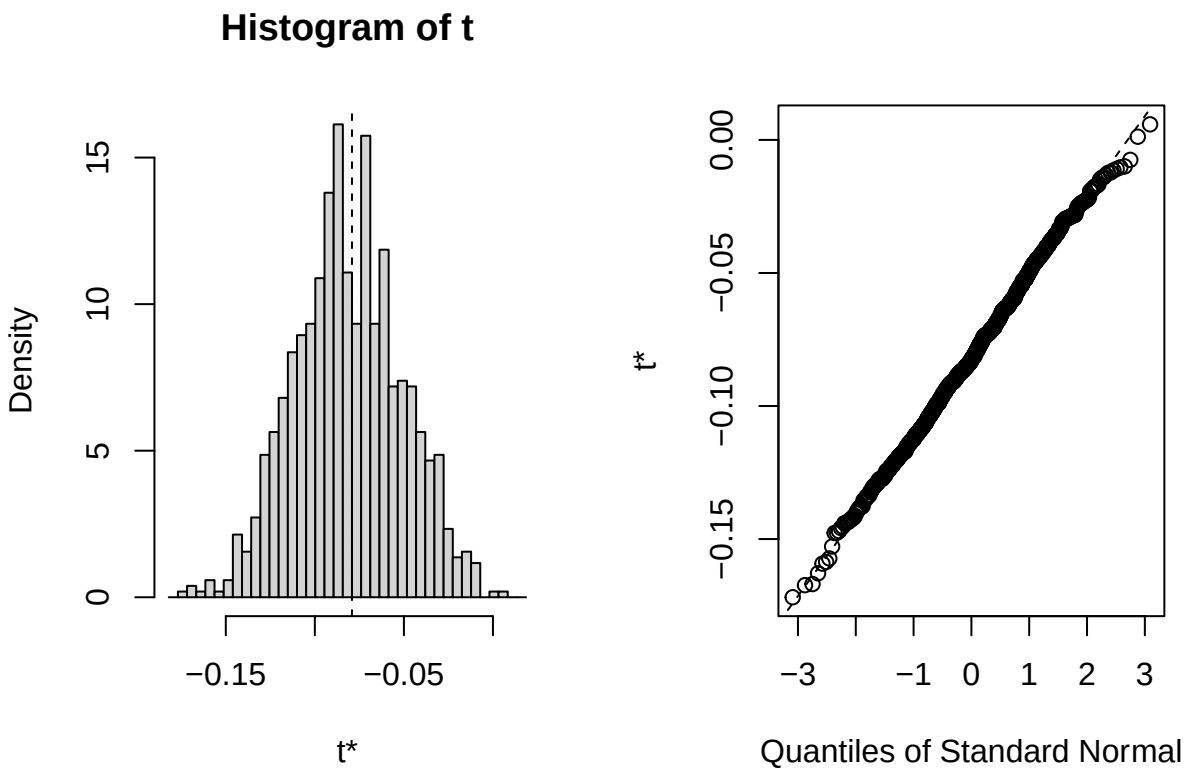

```
## month3
plot(results, index = 16)
```



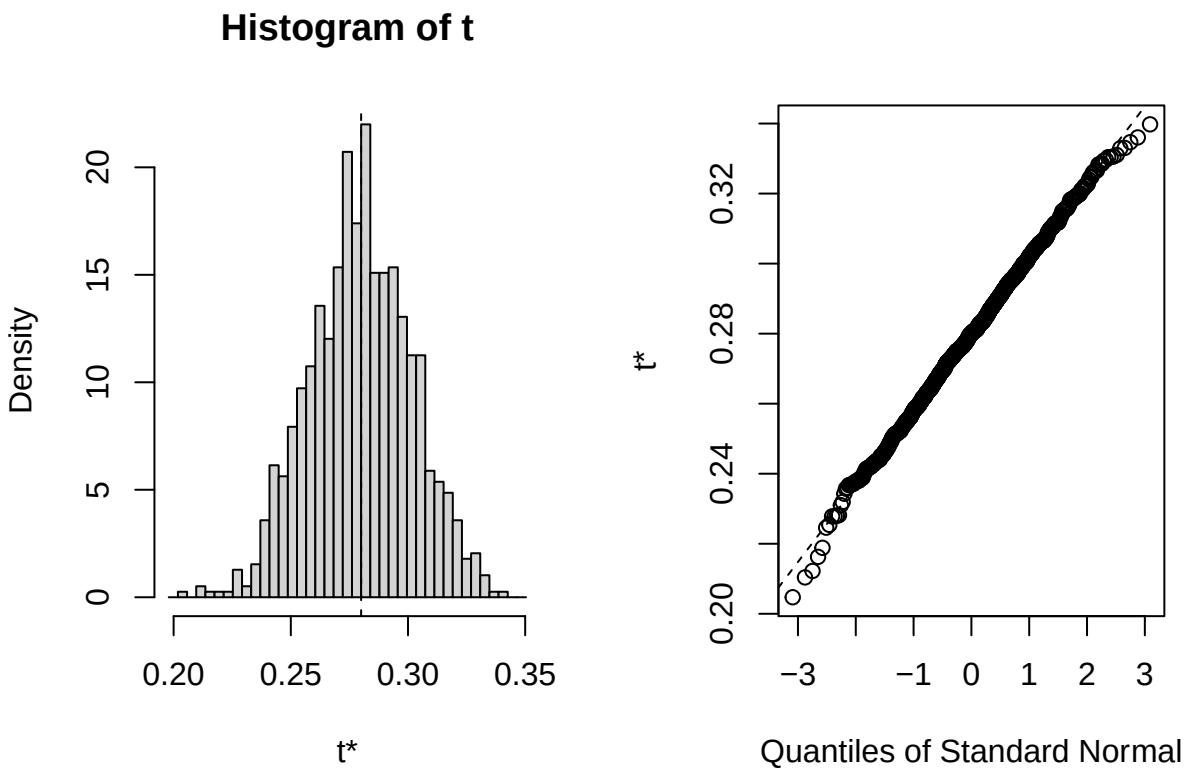

```
## month4
plot(results, index = 17)
```



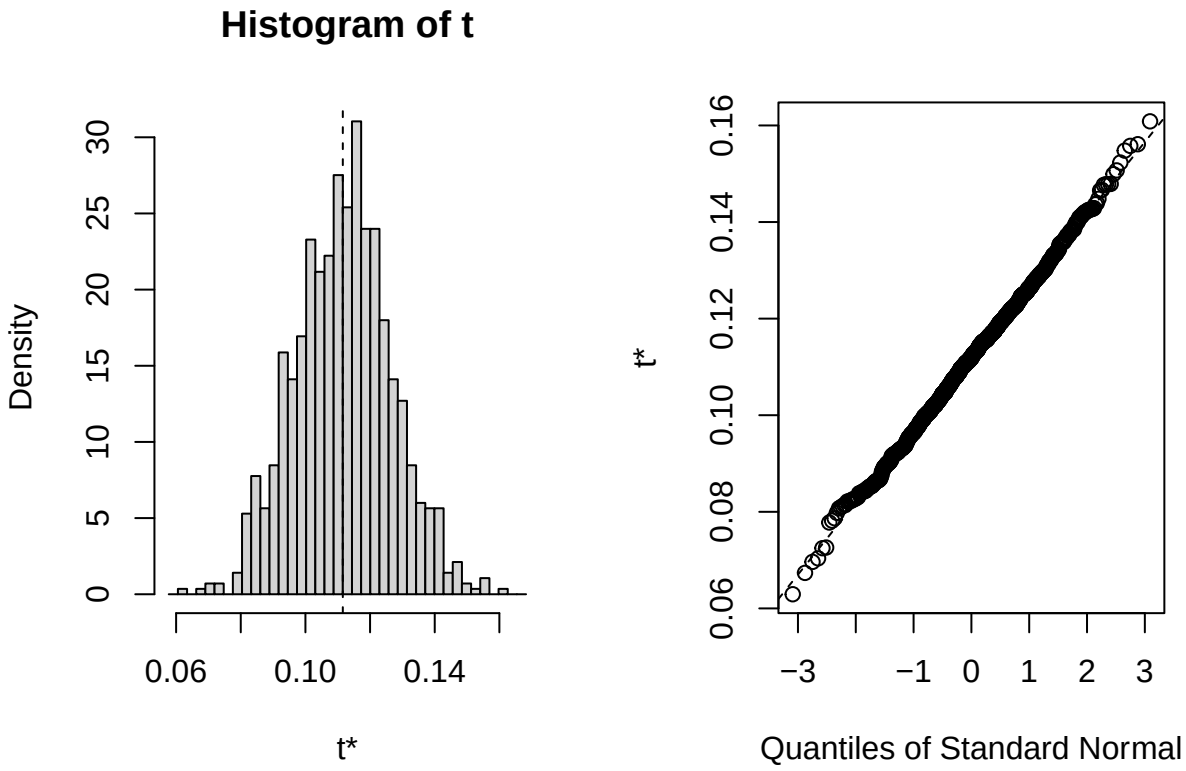

```
## month5
plot(results, index = 18)
```



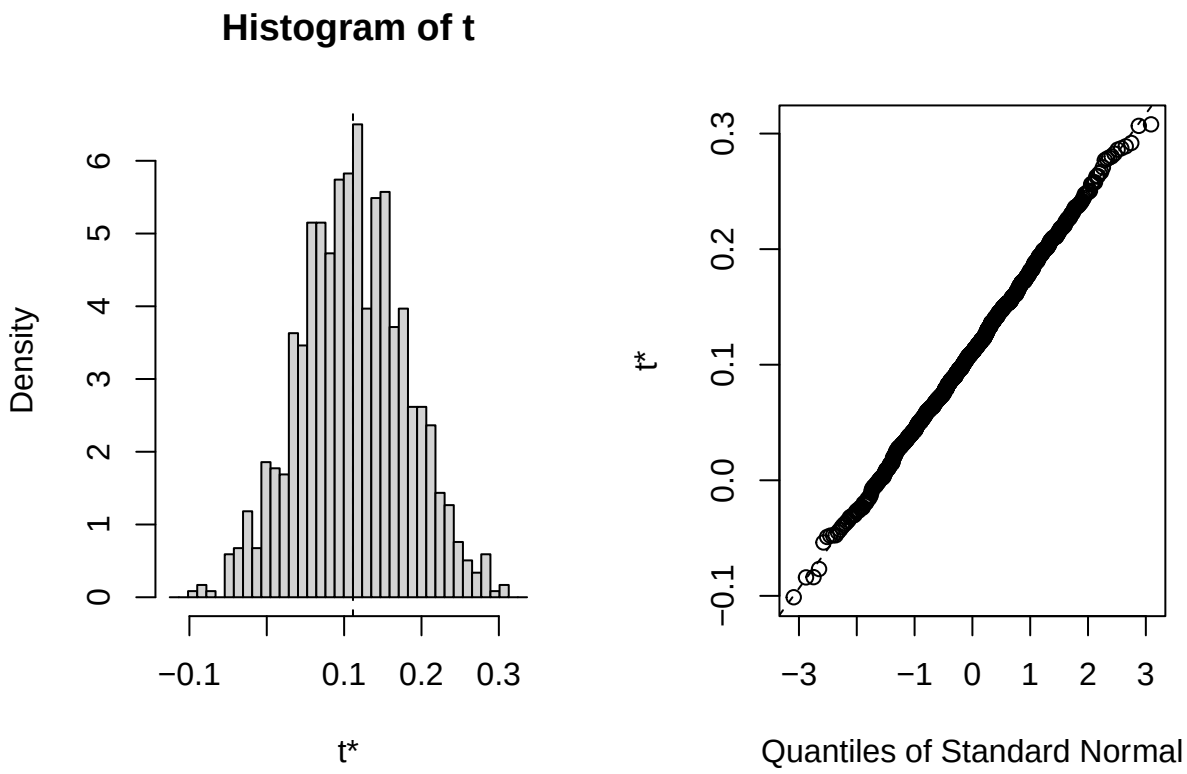

```
## log(comment):iscar
plot(results, index = 19)
```



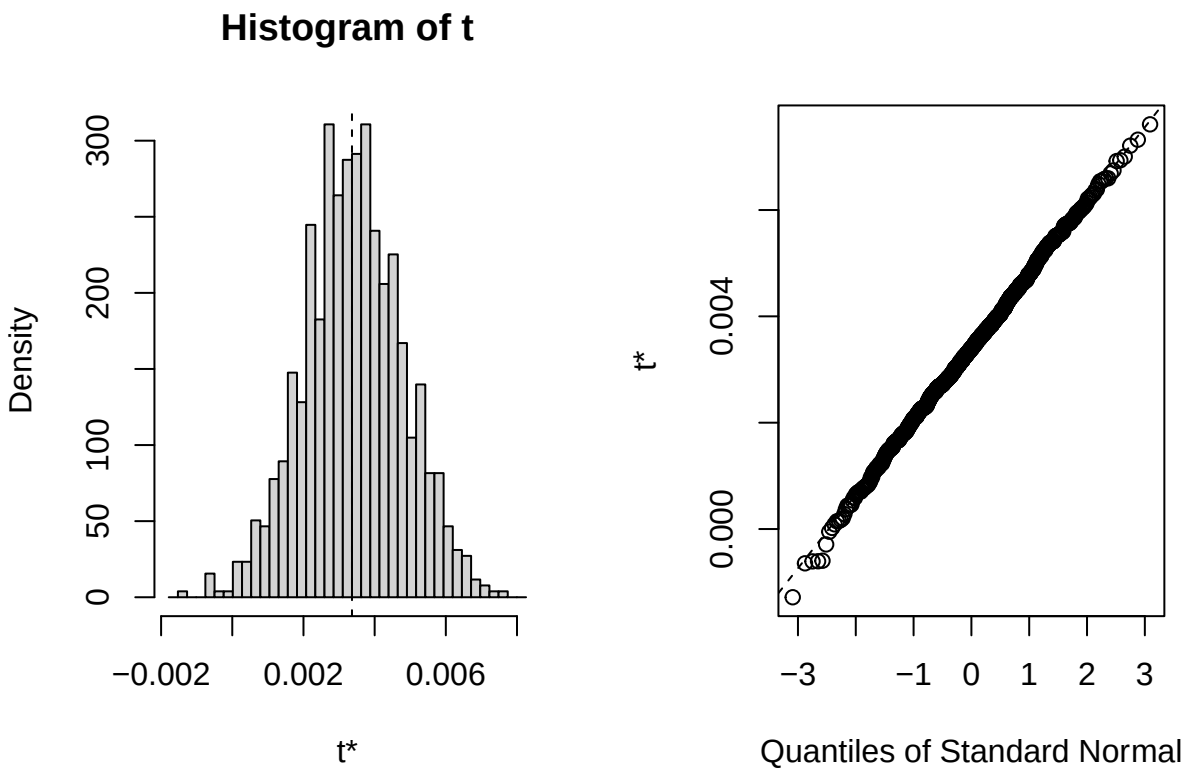

```
## log(share):iscar
plot(results, index = 20)
```



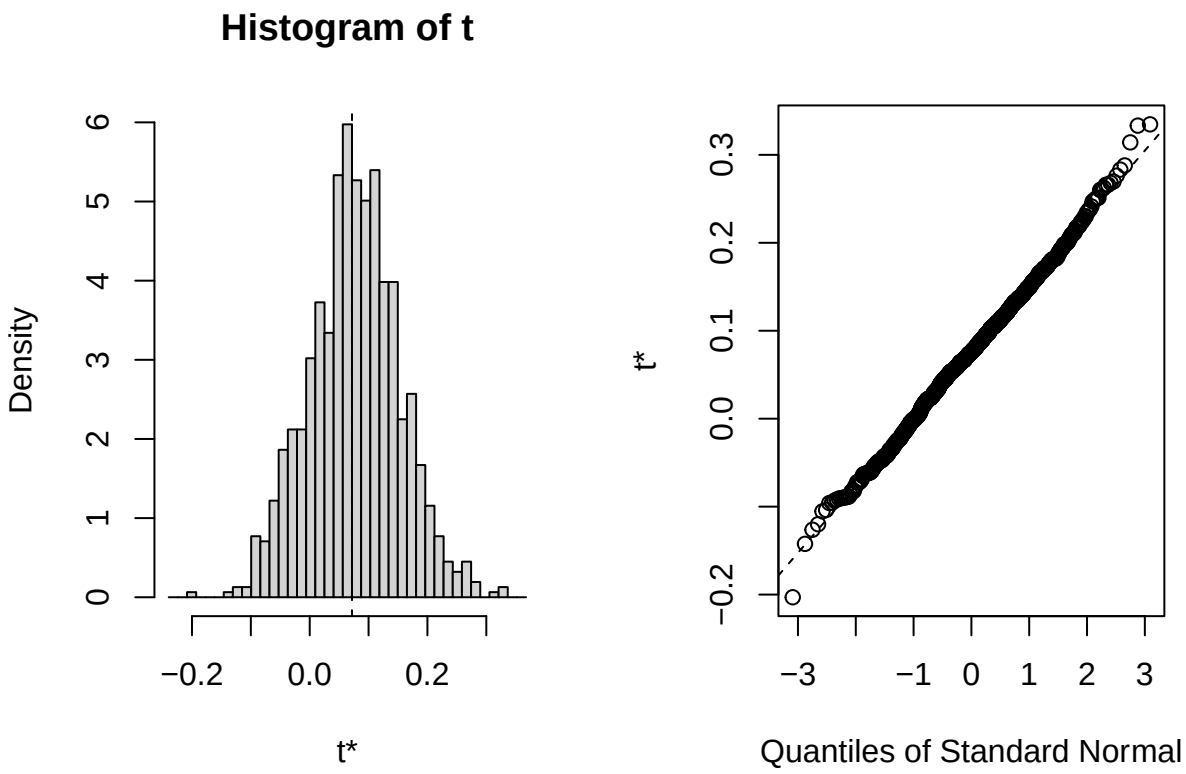

```
## iscar:BGMOriginal
plot(results, index = 21)
```



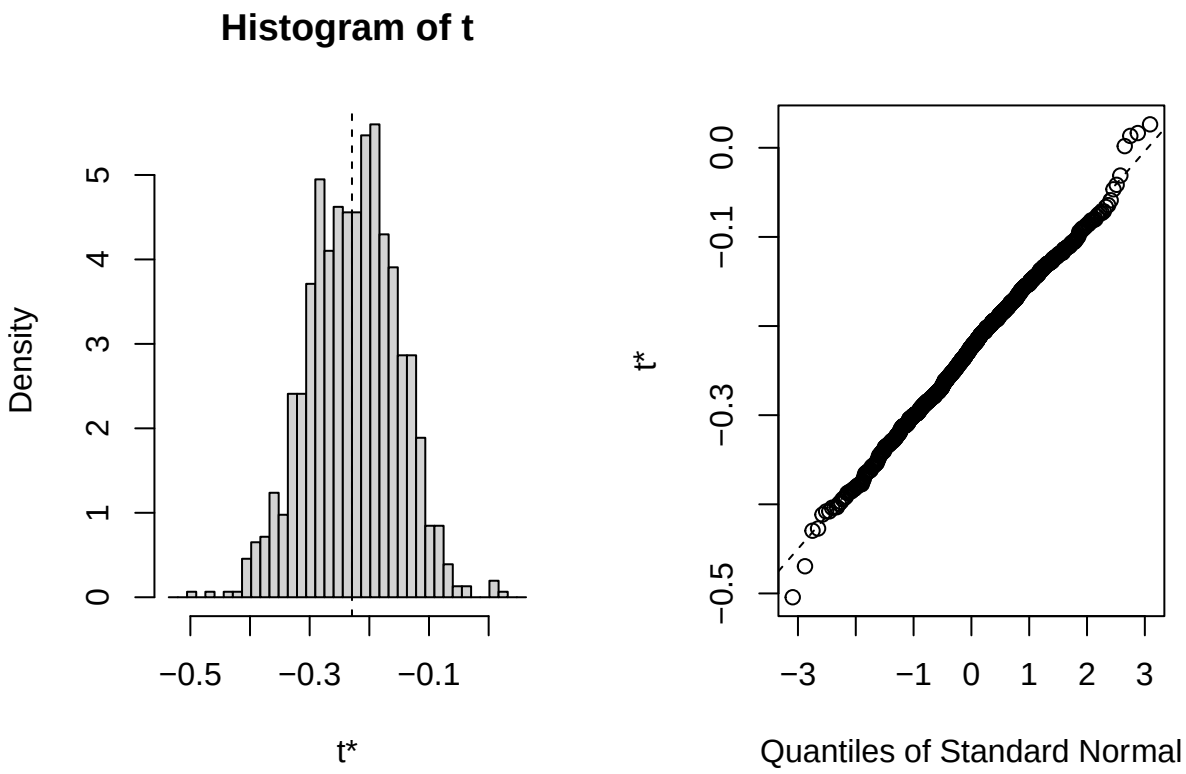

```
## duration:iscar
plot(results, index = 22)
```



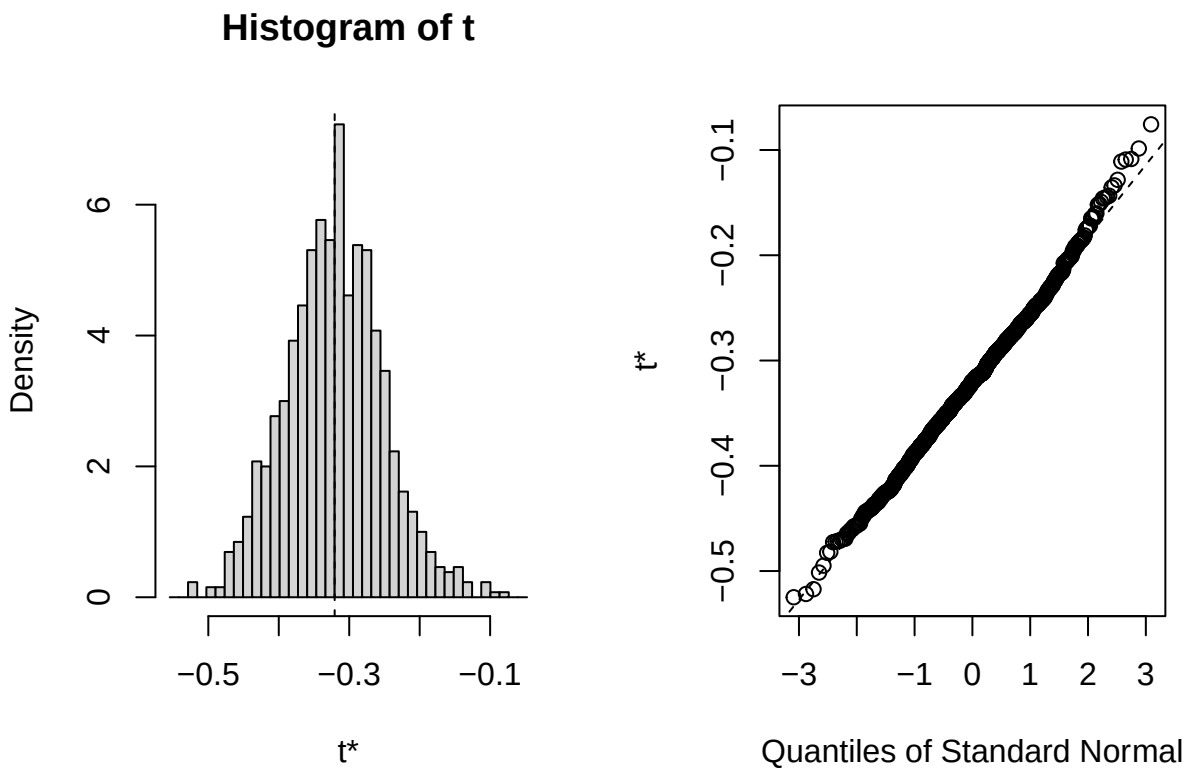

```
## month2:iscar
plot(results, index = 23)
```



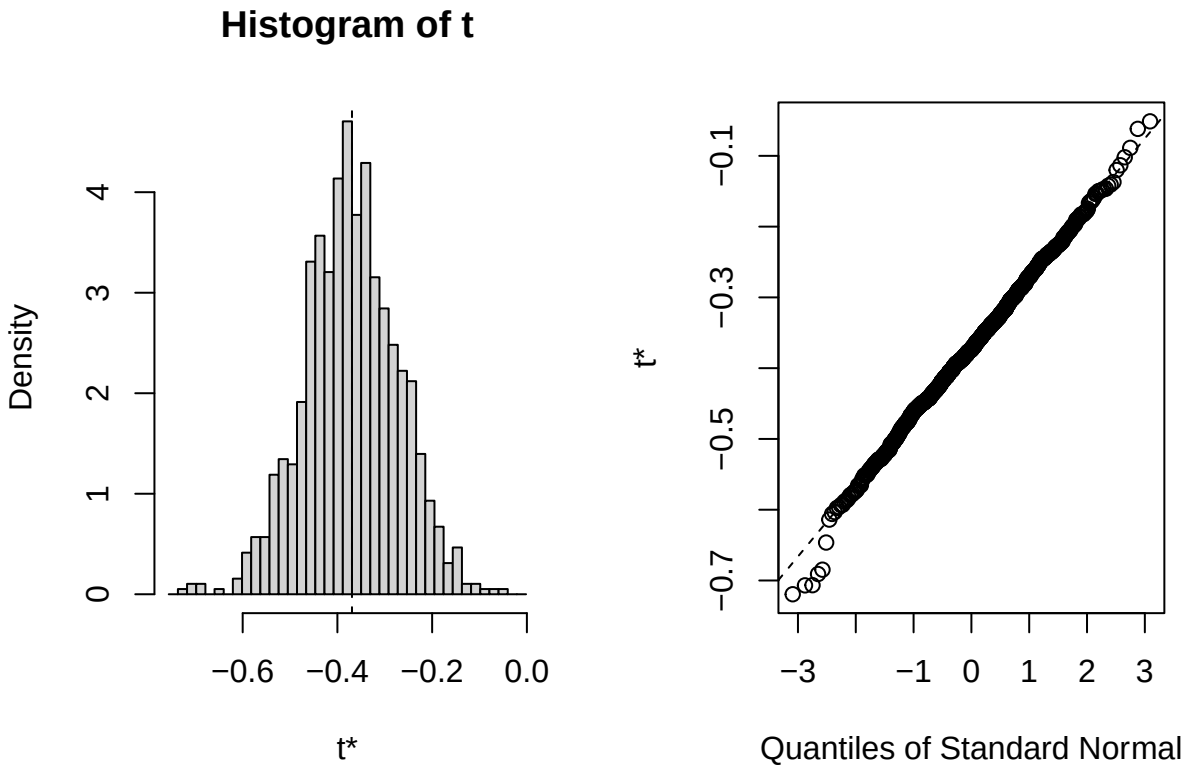

```
## month3:iscar
plot(results, index = 24)
```




```
## month4:iscar
plot(results, index = 25)
```




```
## month5:iscar
plot(results, index = 26)
```



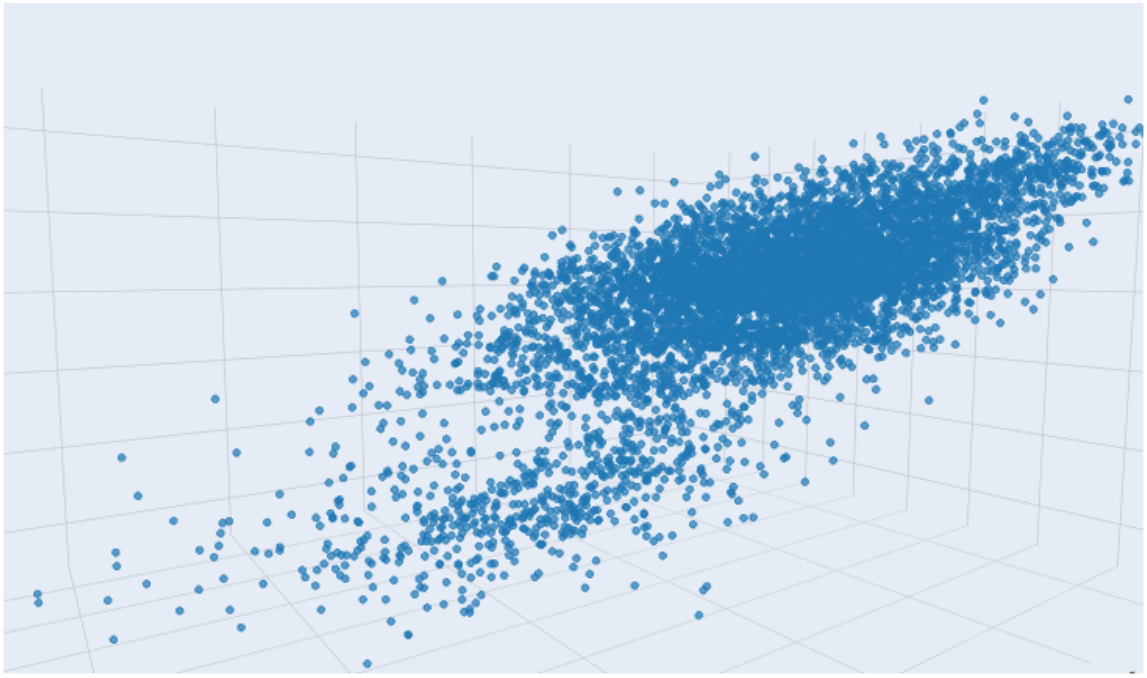
5 Clustering analysis of short video

- Goal: Use the number of likes, comments and shares to aggregate short video data into different clusters.

5.1 Scatter plot

```
library(viridisLite)
## Randomly reorder the index
set.seed(1234)
train <- sample(nrow(video), 1 * nrow(video))
numdata <- log(video[train, c("like", "comment", "share")])



library(plotly)
fig <-
  plot_ly(
    numdata,
    x = numdata$comment,
    y = numdata$share,
    z = numdata$like,
```

```
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



The scatter plot seems to show two clusters, and both clusters are ellipsoidal. Both clusters are closer to the center, the more the points are. And the two clusters are almost centrally symmetrical, conforming to the characteristics of multivariate Gaussian distribution.

Therefore, we first consider the Gaussian mixture model.

5.2 Gaussian Mixture Model

```
library("mclust")
```

```
## Package 'mclust' version 5.4.7
## Type 'citation("mclust")' for citing this R package in publications.

##
##     'mclust'

## The following object is masked from 'package:mvtnorm':
##
##     dmvnorm
```

```
set.seed(1234)
video_mclust <- Mclust(numdata, G = 1:5)
print(video_mclust)
```

```
## 'Mclust' model object: (VEE,5)
##
## Available components:
##  [1] "call"           "data"           "modelName"      "n"
##  [5] "d"              "G"              "BIC"            "loglik"
##  [9] "df"             "bic"            "icl"            "hypvol"
## [13] "parameters"     "z"              "classification" "uncertainty"
```

```
table(video_mclust$classification)
```

```
##
##    1    2    3    4    5
##  607 2637  395 2265  227
```

```
cluster <- video_mclust$classification
```

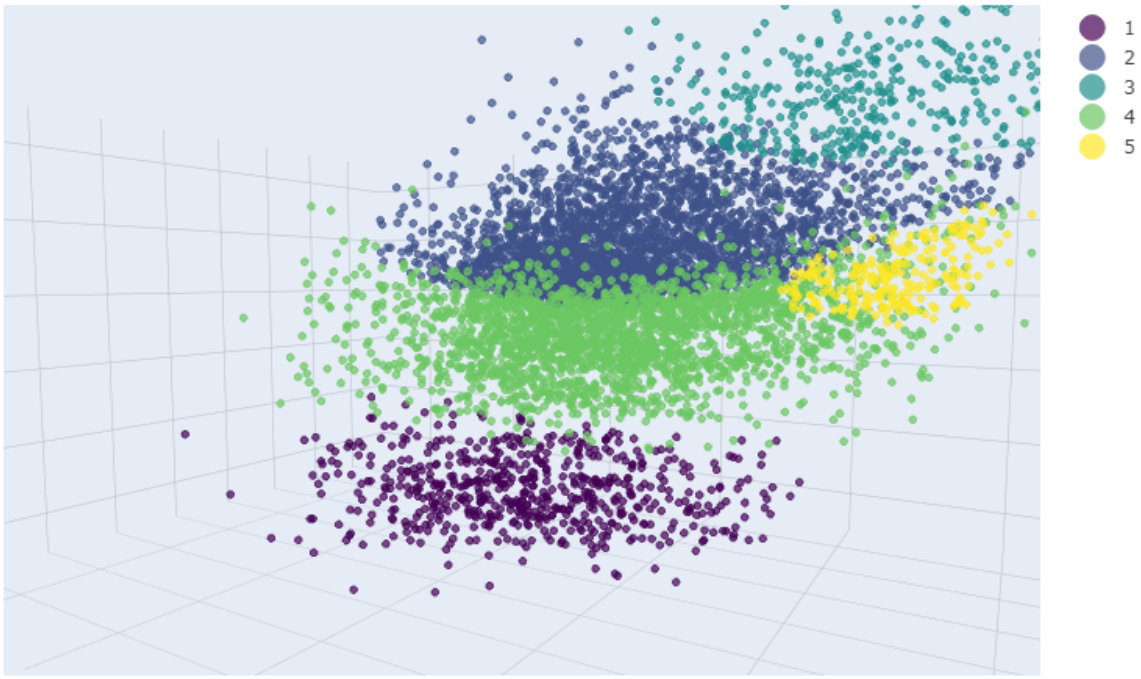
```
tp <- video[train,]
tp$cluster <- 0
ind1 <- cluster == 1
ind2 <- cluster == 2
ind3 <- cluster == 3
ind4 <- cluster == 4
ind5 <- cluster == 5
tp[ind1, ]$cluster <- 1
tp[ind2, ]$cluster <- 2
tp[ind3, ]$cluster <- 3
tp[ind4, ]$cluster <- 4
tp[ind5, ]$cluster <- 5
table(tp$cluster,tp$type)
```

```
##
##     car pet dress makeup food game plot
##   1 606   1     0      0    0    0    0
##   2  61 241   216    451  557  848  263
##   3  18  30    22     54   99   77   95
##   4 275 705   690    363  182   50    0
##   5  21   1    23     60  113    8    1
```

- Visualization

```
library(plotly)
temp <- numdata
temp$cluster <- cluster
temp$cluster <- factor(temp$cluster)
fig <-
  plot_ly(
```

```
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    color = temp$cluster,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))

fig
```



The Gaussian mixture model is clustered into five clusters, and the head and tail can be distinguished well. If you gather two clusters, then the two clusters are well distinguished.

The clustering effect of Gaussian mixture model is already very good, but we need to compare it with several other methods.

5.3 Hierarchical Clustering

```
library(dendextend)
```

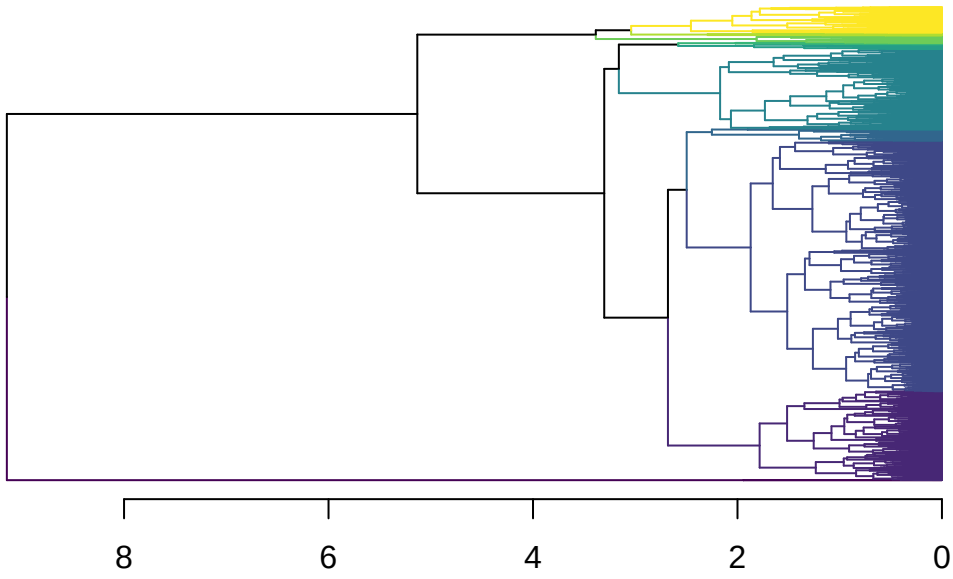
```
##
## ---------------------
## Welcome to dendextend version 1.15.2
```

```
## Type citation('dendextend') for how to cite the package.
##
## Type browseVignettes(package = 'dendextend') for the package vignette.
## The github page is: https://github.com/talgalili/dendextend/
##
## Suggestions and bug-reports can be submitted at: https://github.com/talgalili/dendextend/issues
## You may ask questions at stackoverflow, use the r and dendextend tags:
##   https://stackoverflow.com/questions/tagged/dendextend
##
##  To suppress this message use:  suppressPackageStartupMessages(library(dendextend))
## ----------------------
```

```
##
##    'dendextend'
```

```
## The following object is masked from 'package:stats':
##
##     cutree
```

```
library(colormap)

video_dist <- dist(numdata)
video_average <- hclust(video_dist, method = "average")
dend <- as.dendrogram(video_average)

leafcolor <-
  colormap(
    colormap = colormaps$viridis,
    nshades = 10,
    format = "hex",
    alpha = 1,
    reverse = FALSE
  )

dend %>%
  set("labels_col", value = leafcolor, k = 10) %>%
  set("branches_k_color", value = leafcolor, k = 10) %>%
  plot(
    horiz = TRUE,
    axes = TRUE,
    leaflab = "none",
    main = "Cluster Dendrogram"
  )
```



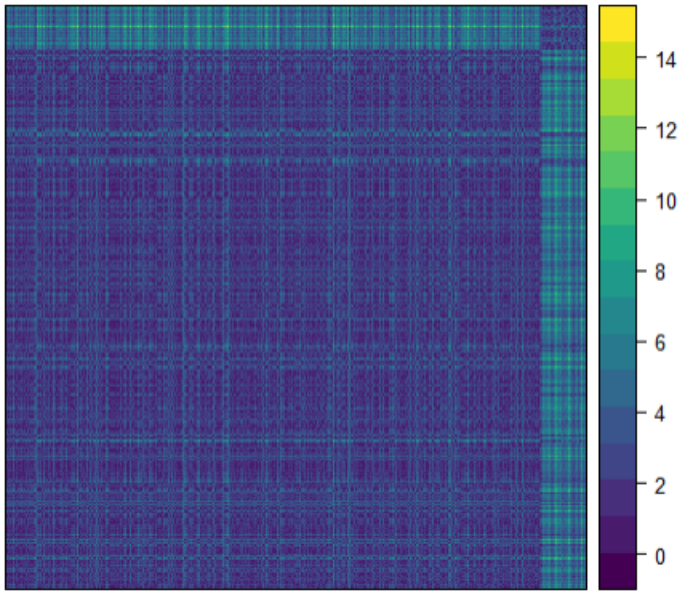

```
## cut at height = 4
cluster <- cutree(video_average, h = 4)

tp <- video[train,]
tp$cluster <- 0
ind1 <- cluster == 1
ind2 <- cluster == 2
ind3 <- cluster == 3
tp[ind1,]$cluster <-1
tp[ind2,]$cluster <-2
tp[ind3,]$cluster <-3
## 3 clusters
table(tp$cluster,tp$type)
```

```
##
##      car pet dress makeup food game plot
##   1 547 954   948    927  951  983  359
##   2 432  24     3      1    0    0    0
##   3   2   0     0      0    0    0    0
```

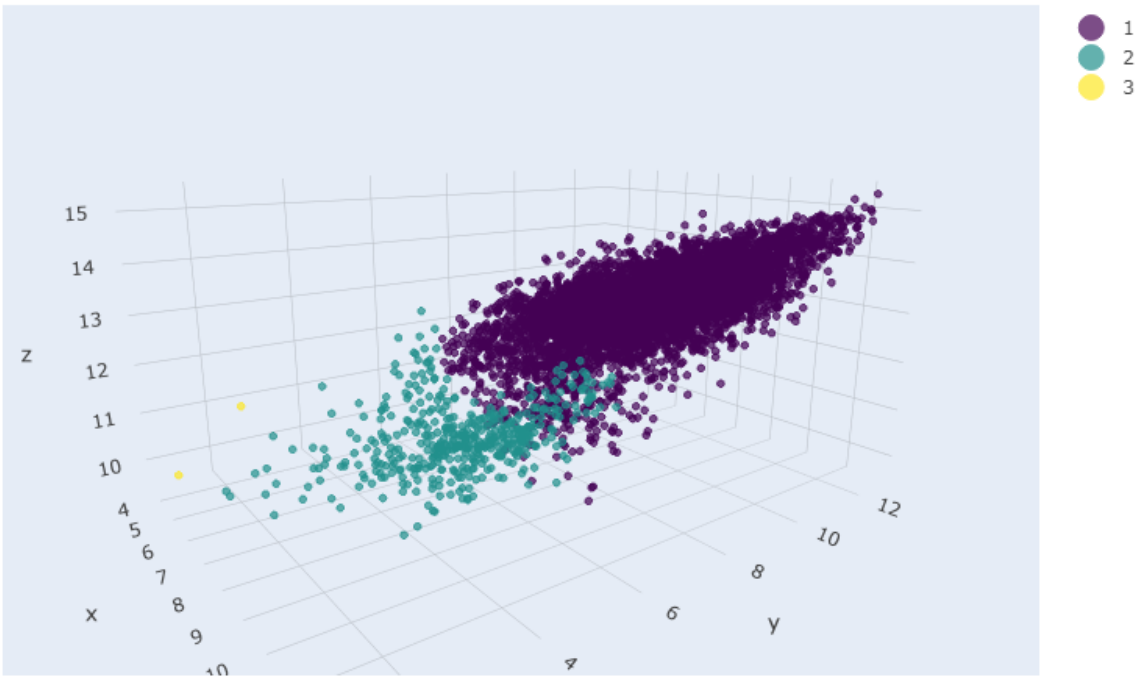
- Visualiztion

```
library(dplyr)
video_hier <- data.frame(numdata, cluster)
video_hier <- arrange(video_hier, cluster)
```

```
video_dist <- dist(video_hier[c(1, 2, 3)])
library(lattice)
library(viridisLite)
#### the figure is too large
#levelplot(
#  as.matrix(video_dist),
#  xlab = "video",
#  ylab = "video",
#  col.regions = viridis(100),
#  scales = list(draw = FALSE)
#)
```




```
library(plotly)
temp <- numdata
temp$cluster <- cluster
temp$cluster <- factor(temp$cluster)
fig <-
  plot_ly(
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    color = temp$cluster,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```

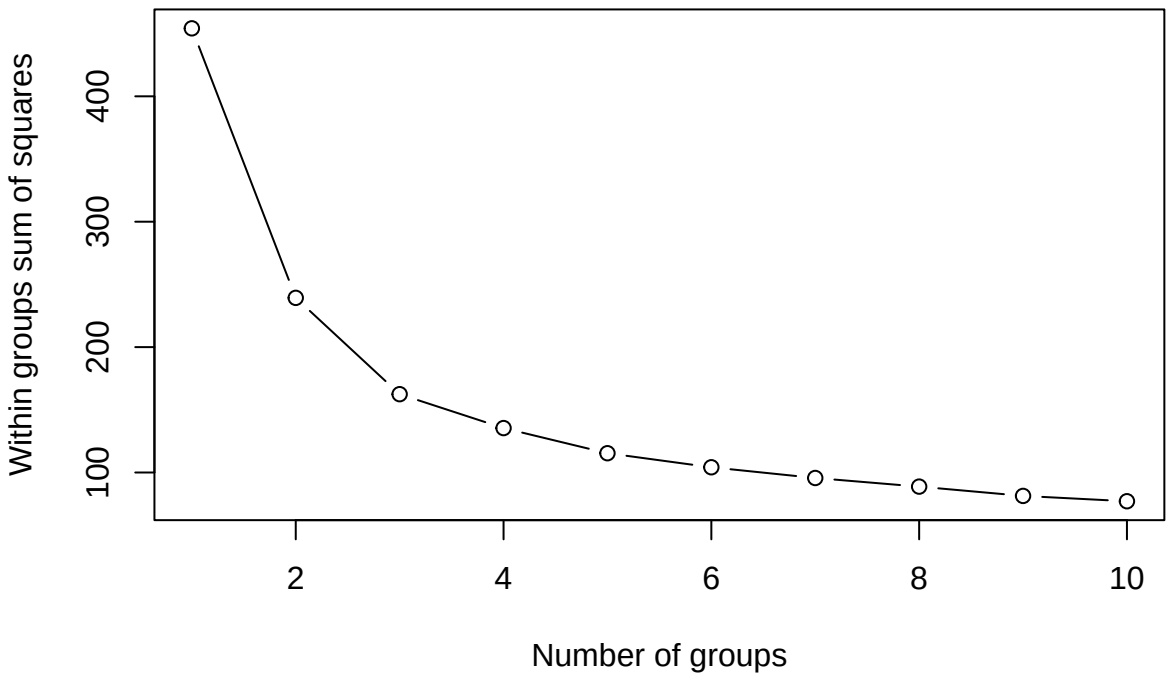


The hierarchical clustering effect is obviously worse than that of the Gaussian mixture model, and the two major clusters are not completely separated.

The two yellow points happen to be the outliers tested by the linear model.

5.4 KMeans

```
rge <- apply(numdata, 2, max) - apply(numdata, 2, min)
numdata.dat <-
  sweep(numdata, 2, rge, FUN = "/")
n <- nrow(numdata.dat)
wss <- rep(0, 10)
wss[1] <- (n - 1) * sum(apply(numdata.dat, 2, var))
for (i in 2:10)
  wss[i] <- sum(kmeans(numdata.dat, centers = i)$withinss)

plot(1:10,
     wss,
     type = "b",
     xlab = "Number of groups",
     ylab = "Within groups sum of squares")
```



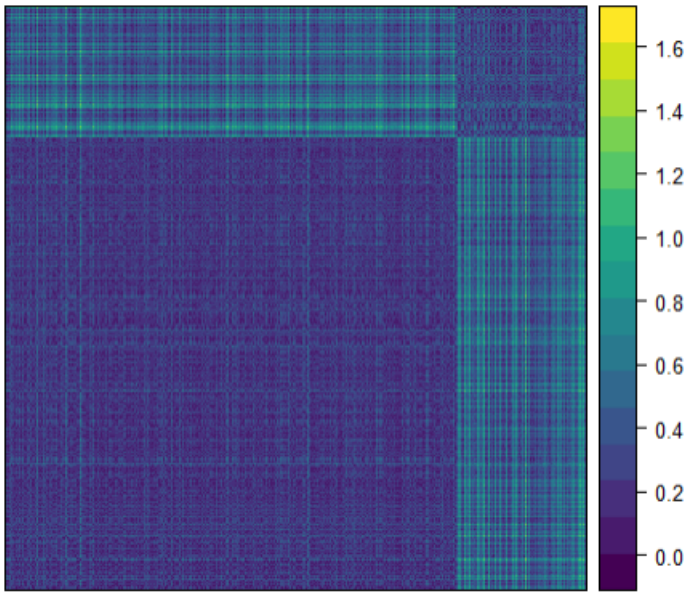
The plot sugguests that we should consider k = 2 or 3.

- k = 2

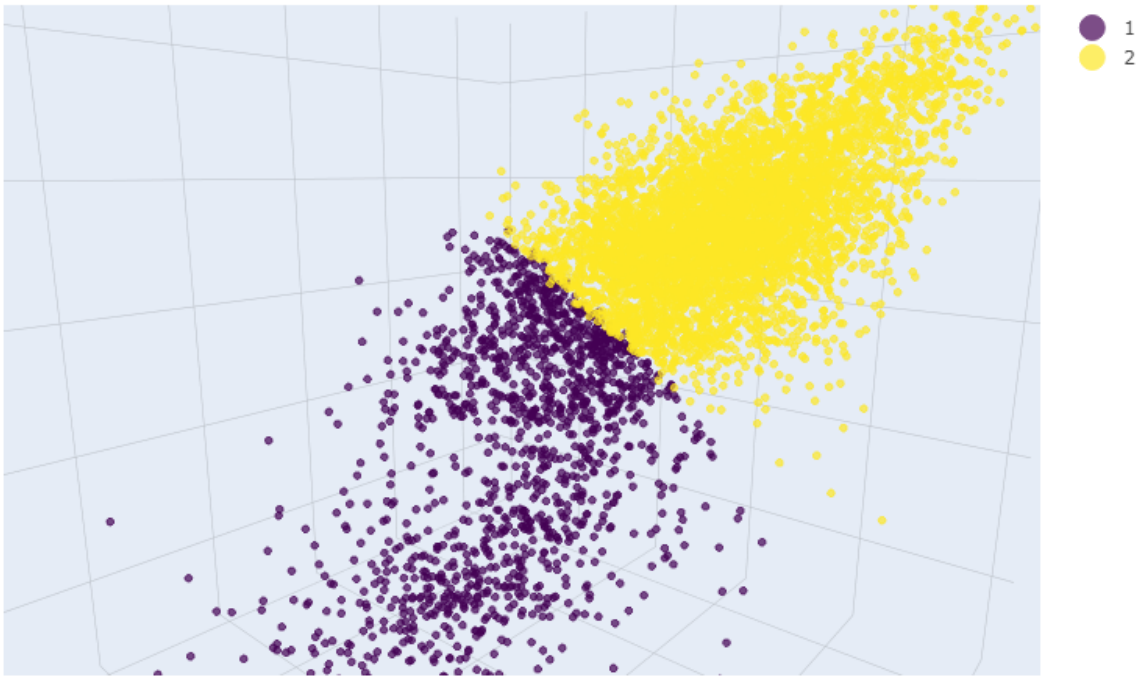
```
numdata_kmeans <- kmeans(numdata.dat, centers = 2)
cluster <- numdata_kmeans$cluster
```

- Visualization

```
library(dplyr)
numdata_kmeans <- data.frame(numdata.dat, cluster)
numdata_reorder <-
  arrange(numdata_kmeans, cluster)
video_dist <- dist(numdata_reorder[c(1, 2, 3)])

library(lattice)
library(viridisLite)
#### the figure is too large
#levelplot(
#  as.matrix(video_dist),
#  xlab = "video",
#  ylab = "video",
#  col.regions = viridis(100),
#  scales = list(draw = FALSE)
#)
```




```
library(plotly)
temp <- numdata_kmeans
temp$cluster <- factor(temp$cluster)
fig <-
  plot_ly(
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    color = temp$cluster,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))

fig
```




```
tp <- video[train,]
tp$cluster <- 0
ind1 <- cluster == 1
ind2 <- cluster == 2
tp[ind1,]$cluster <-1
tp[ind2,]$cluster <-2
table(tp$cluster,tp$type)
```

```
##
##      car pet dress makeup food game plot
##    1 720 415   170     46   18    4    0
##    2 261 563   781    882  933  979  359
```

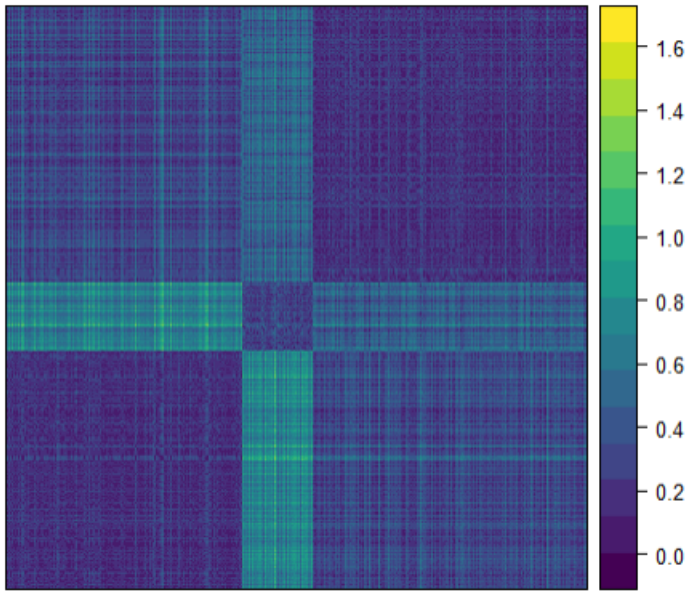
- k = 3

```
numdata_kmeans <- kmeans(numdata.dat, centers = 3)
cluster <- numdata_kmeans$cluster
```

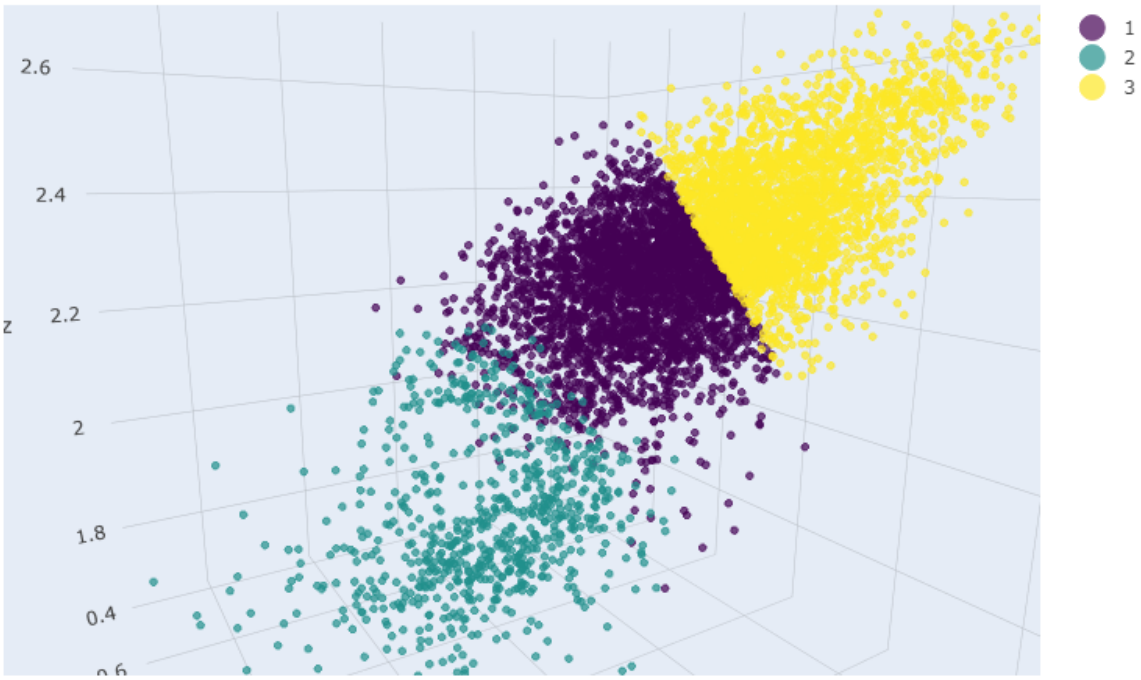
- Visualization

```
library(dplyr)
numdata_kmeans <- data.frame(numdata.dat, cluster)
numdata_reorder <-
  arrange(numdata_kmeans, cluster)
video_dist <- dist(numdata_reorder[c(1, 2, 3)])

library(lattice)
```

```
library(viridisLite)
#### the figure is too large
#levelplot(
#  as.matrix(video_dist),
#  xlab = "video",
#  ylab = "video",
#  col.regions = viridis(100),
#  scales = list(draw = FALSE)
#)
```




```
library(plotly)
temp <- numdata_kmeans
temp$cluster <- factor(temp$cluster)
fig <-
  plot_ly(
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    color = temp$cluster,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))

fig
```




```
tp <- video[train,]
tp$cluster <- 0
ind1 <- cluster == 1
ind2 <- cluster == 2
ind3 <- cluster == 3
tp[ind1,]$cluster <-1
tp[ind2,]$cluster <-2
tp[ind3,]$cluster <-3
table(tp$cluster,tp$type)
```

```
##
##     car pet dress makeup food game plot
##   1 253 642   706    458  360  412   52
##   2 618 114     4      0    0    0    0
##   3 110 222   241    470  591  571  307
```

KMeans clustering separates the above major cluster, which is different from the previous Gaussian mixture model. The oblique direction is the direction of the first principal component, and a cross section perpendicular to the principal component direction separates them.

5.5 Generative model

Motivation: The Gaussian mixture model can be used not only for clustering, but also as a generative model, so we use gaussian mixture model to generate new data. And we use the generated data to test the classifier we will build later.

- Density function learned by Mclust()

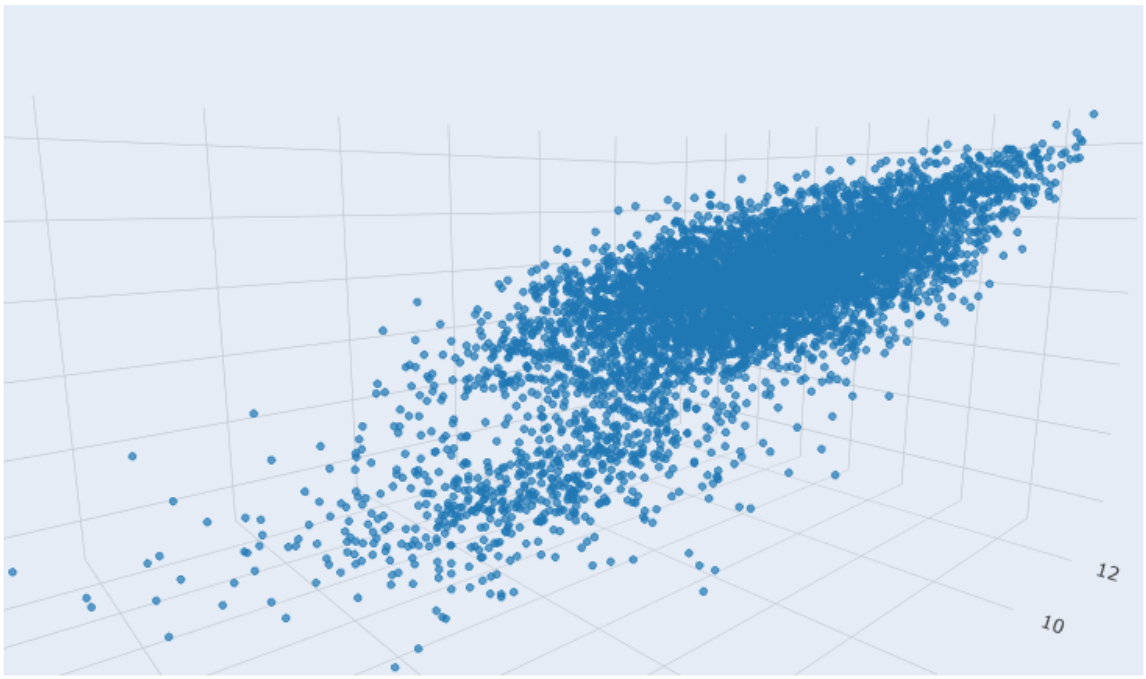
```
library(MASS)
mu1 <- video_mclust$parameters$mean[, 1]
mu2 <- video_mclust$parameters$mean[, 2]
mu3 <- video_mclust$parameters$mean[, 3]
mu4 <- video_mclust$parameters$mean[, 4]
mu5 <- video_mclust$parameters$mean[, 5]
sigma1 <- video_mclust$parameters$variance$sigma[, , 1]
sigma2 <- video_mclust$parameters$variance$sigma[, , 2]
sigma3 <- video_mclust$parameters$variance$sigma[, , 3]
sigma4 <- video_mclust$parameters$variance$sigma[, , 4]
sigma5 <- video_mclust$parameters$variance$sigma[, , 5]

p <- video_mclust$parameters$pro
```

- function for generating new data

```
gaussian.generate <- function(n) {
  newdata <- data.frame(matrix(rep(0, 3 * n), nrow = n))

  set.seed(1234)
  for (i in 1:n) {
    z <- rmultinom(1, 1, p)
    newdata[i, ] <- t(z) %*%
      rbind(
        mvrnorm(1, mu1, sigma1),
        mvrnorm(1, mu2, sigma2),
        mvrnorm(1, mu3, sigma3),
        mvrnorm(1, mu4, sigma4),
        mvrnorm(1, mu5, sigma5)
      )
  }
  return(newdata)
}
```

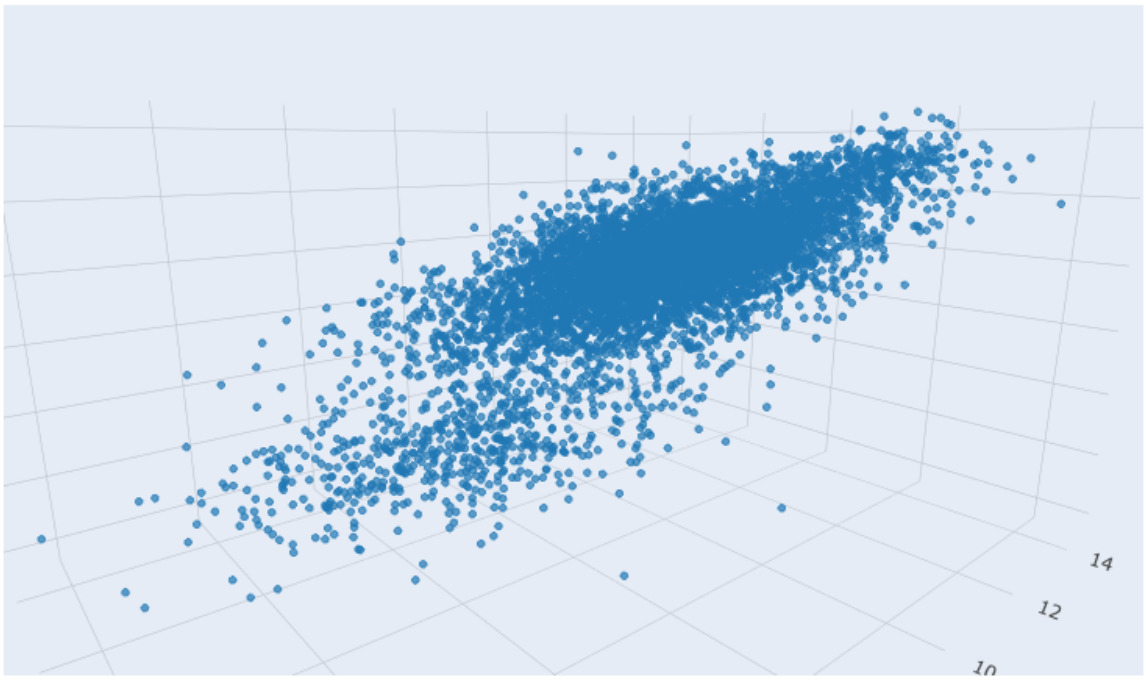
- Visualization(origin data, n = 6131)

```
library(plotly)
library(viridisLite)
temp <- log(video[c("like","comment","share")])
fig <-
  plot_ly(
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



- Visualization(generated data, n = 6131)

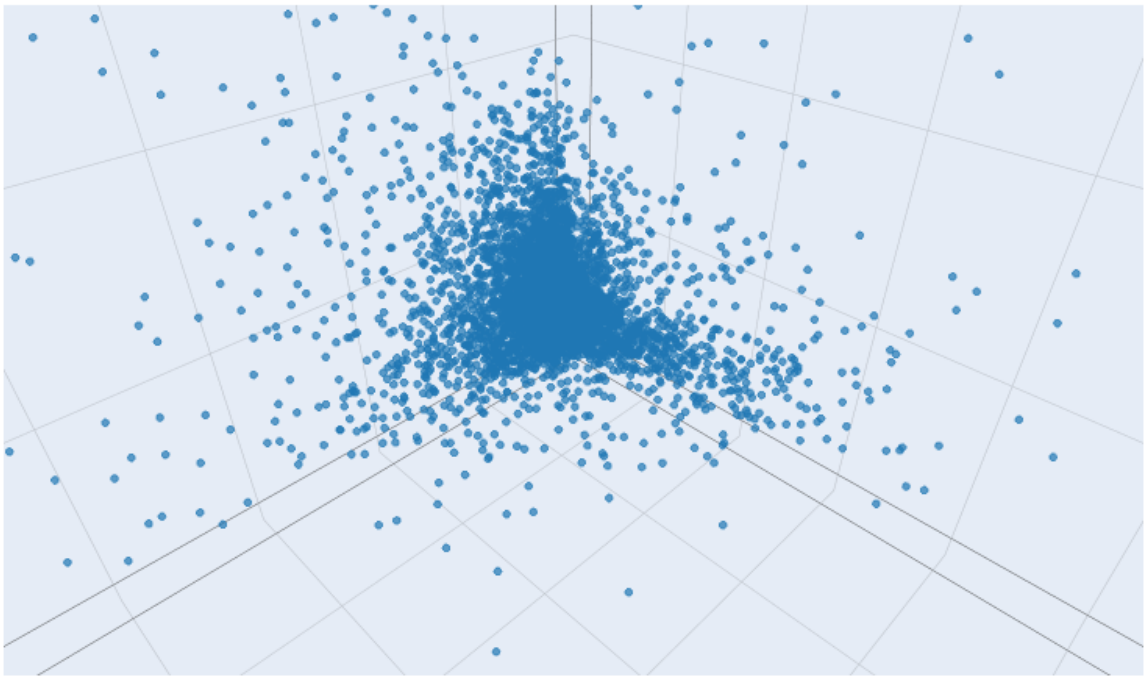
```
library(plotly)
library(viridisLite)
temp <- gaussian.generate(6131)
fig <-
  plot_ly(
    temp,
    x = temp$X2,
    y = temp$X3,
    z = temp$X1,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(
    title = "",
    scene = list(
      bgcolor = "#e5ecf6"
    )
  )
fig
```



Remove log and return to the original metric space.

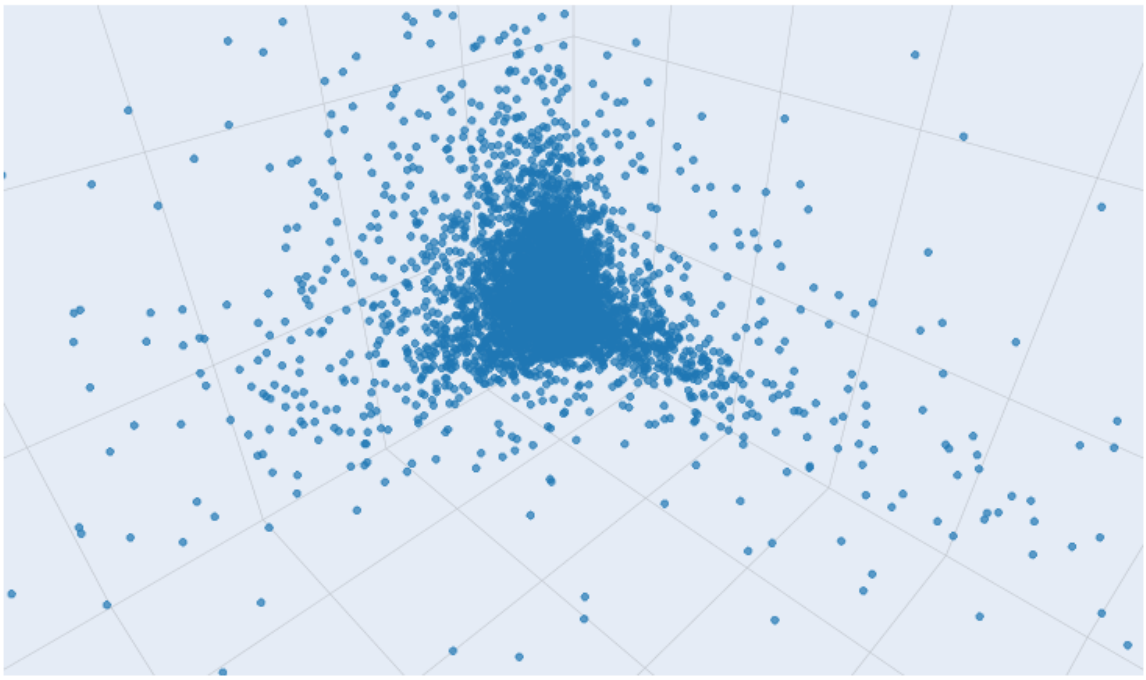
- Visualization(origin data, n = 6131)

```
library(plotly)
library(viridisLite)
temp <- video[c("like","comment","share")]
fig <-
  plot_ly(
    temp,
    x = temp$comment,
    y = temp$share,
    z = temp$like,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



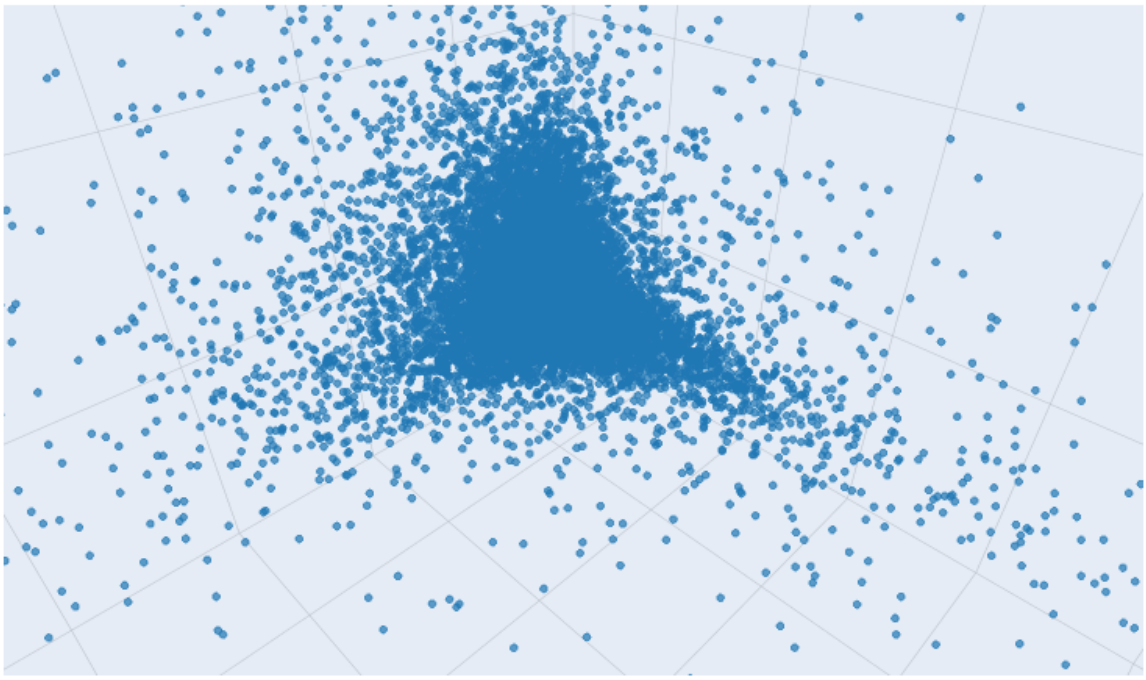
- Visualization(generated data, n = 6131)

```
library(plotly)
library(viridisLite)
temp <- exp(gaussian.generate(6131))
fig <-
  plot_ly(
    temp,
    x = temp$X2,
    y = temp$X3,
    z = temp$X1,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(
           bgcolor = "#e5ecf6",
           xaxis = list(range = c(1, 120245)),
           yaxis = list(range = c(1, 350000)),
           zaxis = list(range = c(1, 4556768))
         ))

fig
```



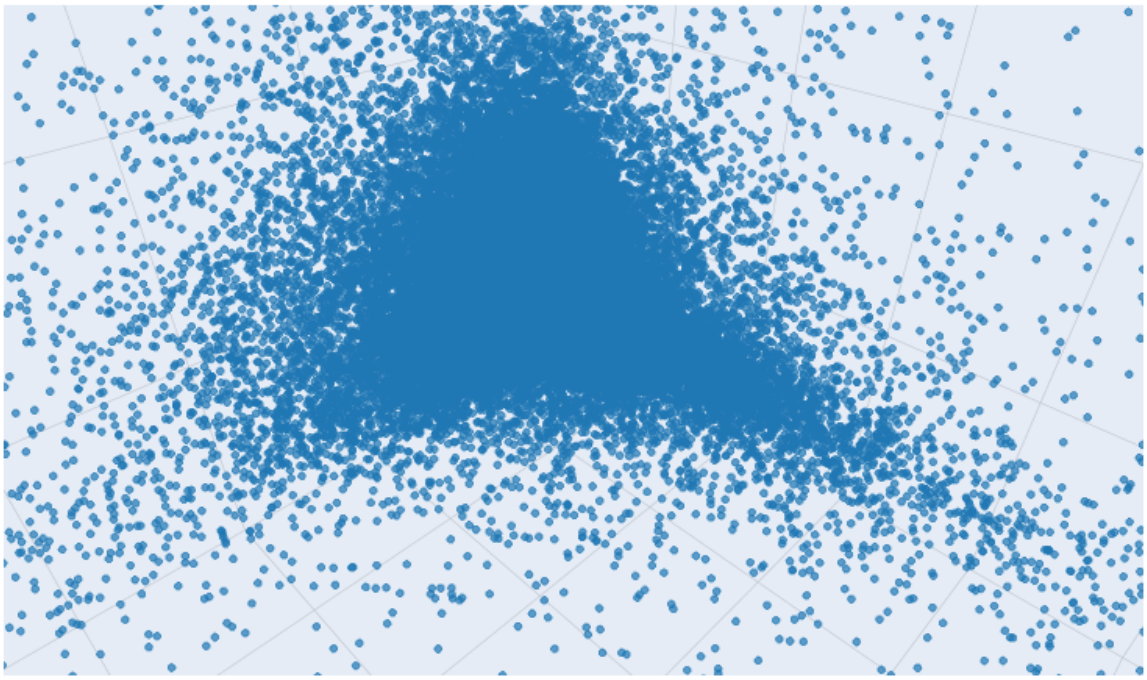
- Visualization(generated data, n = 20000)

```
library(plotly)
library(viridisLite)
temp <- exp(gaussian.generate(20000))
fig <-
  plot_ly(
    temp,
    x = temp$X2,
    y = temp$X3,
    z = temp$X1,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(
           bgcolor = "#e5ecf6",
           xaxis = list(range = c(1, 120245)),
           yaxis = list(range = c(1, 350000)),
           zaxis = list(range = c(1, 4556768))
         ))

fig
```



- Visualization(generated data, n = 100000)

```
library(plotly)
library(viridisLite)
temp <- exp(gaussian.generate(100000))
fig <-
  plot_ly(
    temp,
    x = temp$X2,
    y = temp$X3,
    z = temp$X1,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(
           bgcolor = "#e5ecf6",
           xaxis = list(range = c(1, 120245)),
           yaxis = list(range = c(1, 350000)),
           zaxis = list(range = c(1, 4556768))
         ))

fig
```



No matter which metric space is in, the distribution of our generated data is almost the same as that of the original data. The establishment of the generative model was very successful.

6 Classification of short video types based on PCA

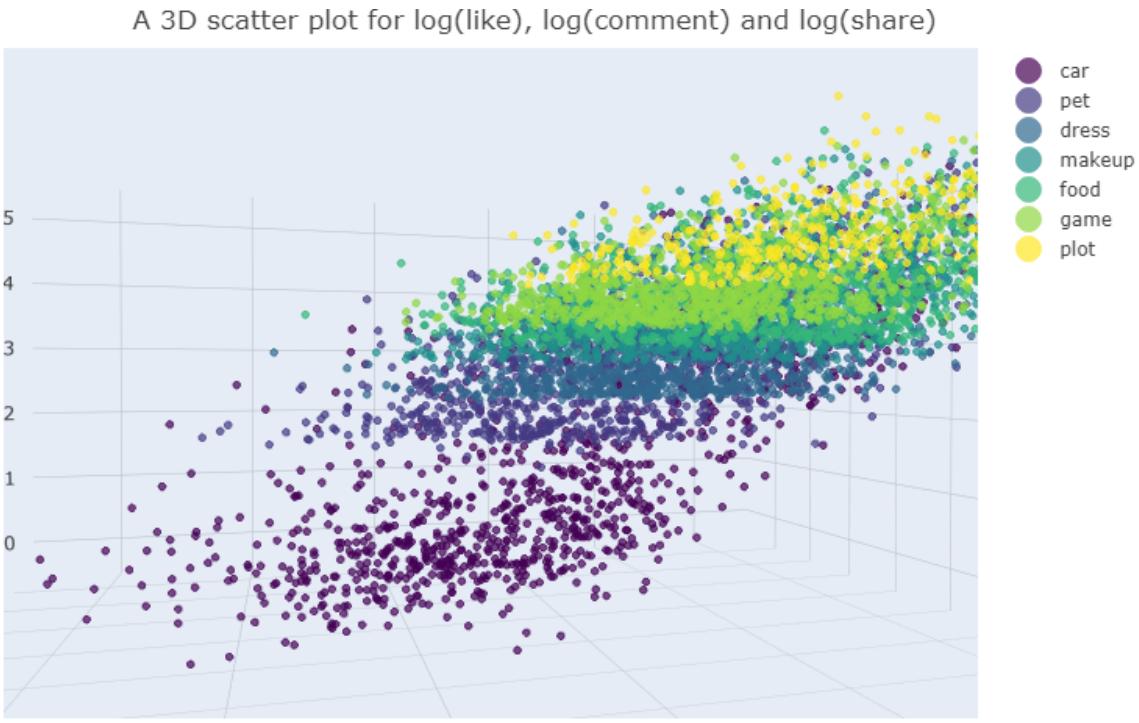
Motivation: Open the short video interface of the mobile phone (take Tiktok as an example), all we can directly know are the number of likes, comments, shares, BGM, and title words. Then I want to use these data to determine which type of short video belongs to, and then decide whether I need to finish watching it, because if it's not the type I like, then I don't need to waste time.

Goal: Use the number of likes, comments, shares, BGM, and the number of title words to classify the type of each short video.

6.1 Scatter plot

```
library(plotly)
fig <-
  plot_ly(
    x = log(video$comment),
    y = log(video$share),
    z = log(video$like),
    color = video$type,
    colors = viridis(7)
  ) %>% add_markers(size = 12)

fig <- fig %>%
```

```
  layout(title = "A 3D scatter plot for log(like), log(comment) and log(share)",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



The three-dimensional scatter plot seems to have obvious stratification effect by type.

This layering feature is very consistent with the hyperplane segmentation characteristics of support vector machines. Therefore, I first consider the support vector machine, and use the number of likes, comments and shares to predict the short video type.

6.2 Support Vector Machine

```
## split the sample set
set.seed(1234)
video.class <- log(video[c("like", "comment", "share")])
video.class$type <- video$type
train <- sample(nrow(video.class), 0.7 * nrow(video.class))
video.train <- video.class[train,]
video.validate <- video.class[-train,]
table(video.train$type_code)
```

```
## < table of extent 0 >
```

```
table(video.validate$type_code)
```

```
## < table of extent 0 >
```

```
library(e1071)
set.seed(1234)
fit.svm <-
  svm(type ~ like + comment + share, data = video.train) ## have taken logarithm
fit.svm
```

```
##
## Call:
## svm(formula = type ~ like + comment + share, data = video.train)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  3651
```

```
svm.pred <- predict(fit.svm, na.omit(video.validate))
svm.perf <- table(na.omit(video.validate)$type,
                  svm.pred,
                  dnn = c("Actual", "Predicted"))
## confusion matrix
svm.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    173  31    42      9   20    6    2
##   pet      3 110    72     23   12   49   11
##   dress    0  26   163     42   17   39    6
##   makeup   0   6    47     84   38   93   14
##   food     0   1    12     39   91  117   27
##   game     0   0     1     18   25  211   35
##   plot     0   0     0      1    9   66   49
```

- The function of calculating F1-score for multi-class classification

```
myF1score <- function(matrix, dataset) {
  TP <- diag(matrix)
  FP <- apply(matrix, 2, sum) - TP
  FN <- apply(matrix, 1, sum) - TP
  precision <- TP / (TP + FP)
  recall <- TP / (TP + FN)
  F1 <- 2 * precision * recall / (precision + recall)

  ## weight
  w <- table(dataset$type) / sum(table(dataset$type))
  F1_w <- w %*% F1

  return(as.double(F1_w))
}
```

- Tune parameters

```
### tuning an RBF support vector machine
#set.seed(1234)
#
### varies the parameters
#tuned <- tune.svm(
#  type ~ like + comment + share,
#  data = video.train,
#  gamma = 2 ^ (-7:7),
#  cost = 2 ^ (-7:7)
#)
#tuned ## print the best model
```

- Use best parameters

```
fit.svm <-
  svm(
    type ~ like + comment + share,
    data = video.train,
    gamma = 2,
    cost = 2
  )
svm.pred <- predict(fit.svm, na.omit(video.validate))

## evaluate the cross-validation performance
svm.perf <- table(na.omit(video.validate)$type,
                  svm.pred,
                  dnn = c("Actual", "Predicted"))
## confusion matrix
svm.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    182  30    33     11   21    4    2
##   pet      5 116    66     24   10   48   11
##   dress    3  32   145     47   23   38    5
##   makeup   0  12    34    101   45   75   15
##   food     0   2    11     64   83  100   27
##   game     0   0     1     24   31  203   31
##   plot     0   0     0      9   14   57   45
```

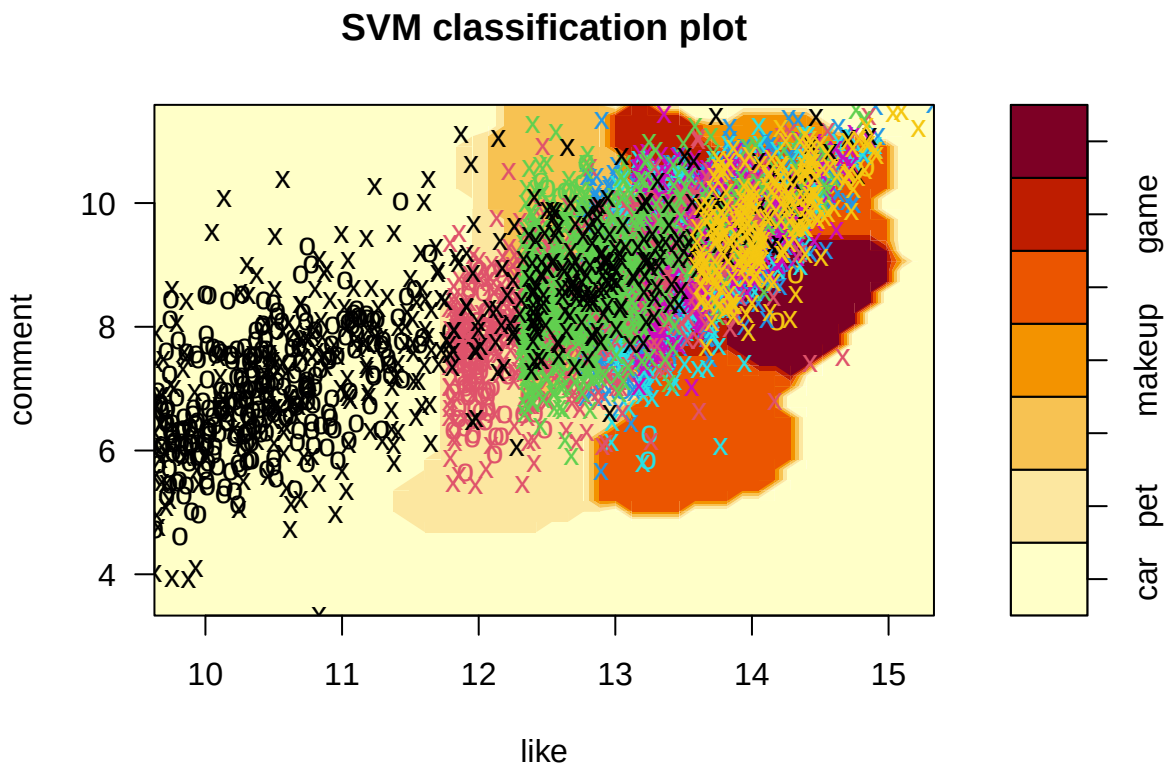
```
## accuracy
sum(diag(svm.perf)) / sum(svm.perf)
```

```
## [1] 0.4755435
```

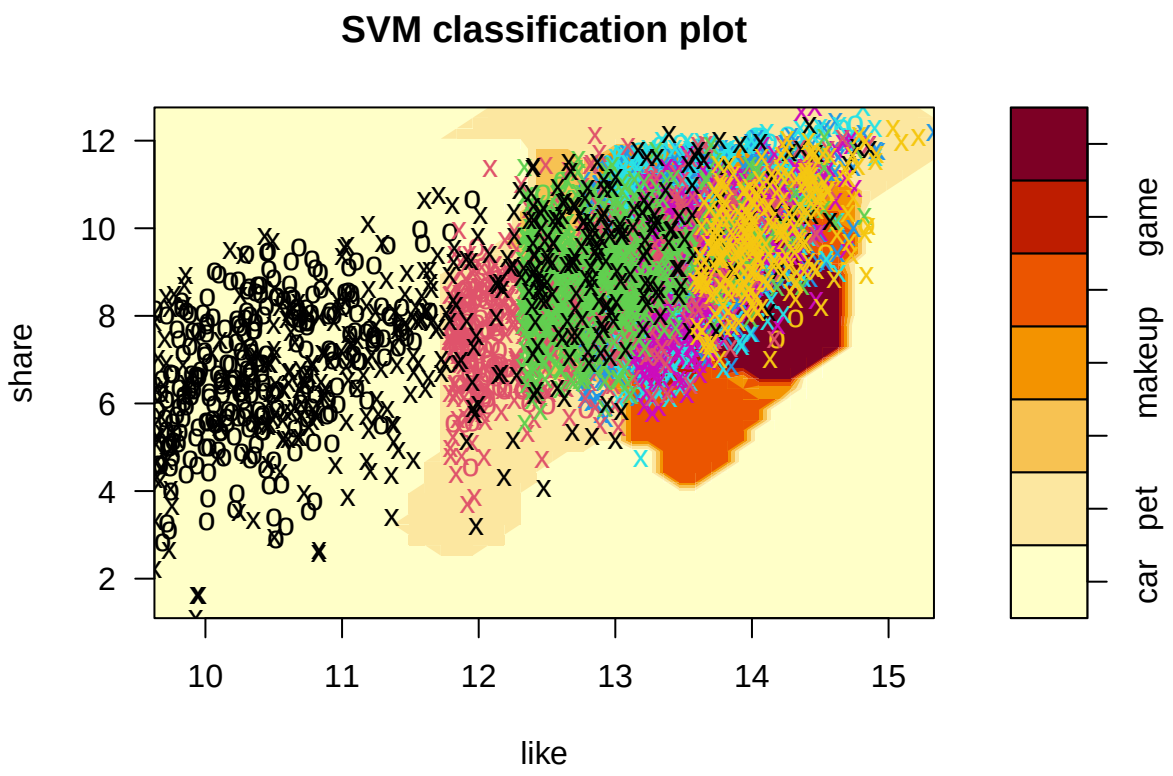
```
## F1score
myF1score(svm.perf, video.validate)
```

```
## [1] 0.4797693
```

```
## visualization
plot(fit.svm, video.train, comment ~ like, slice = list(share = 8))
```




```
plot(fit.svm, video.train, share ~ like, slice = list(comment = 8))
```



- PCA for better classification performance

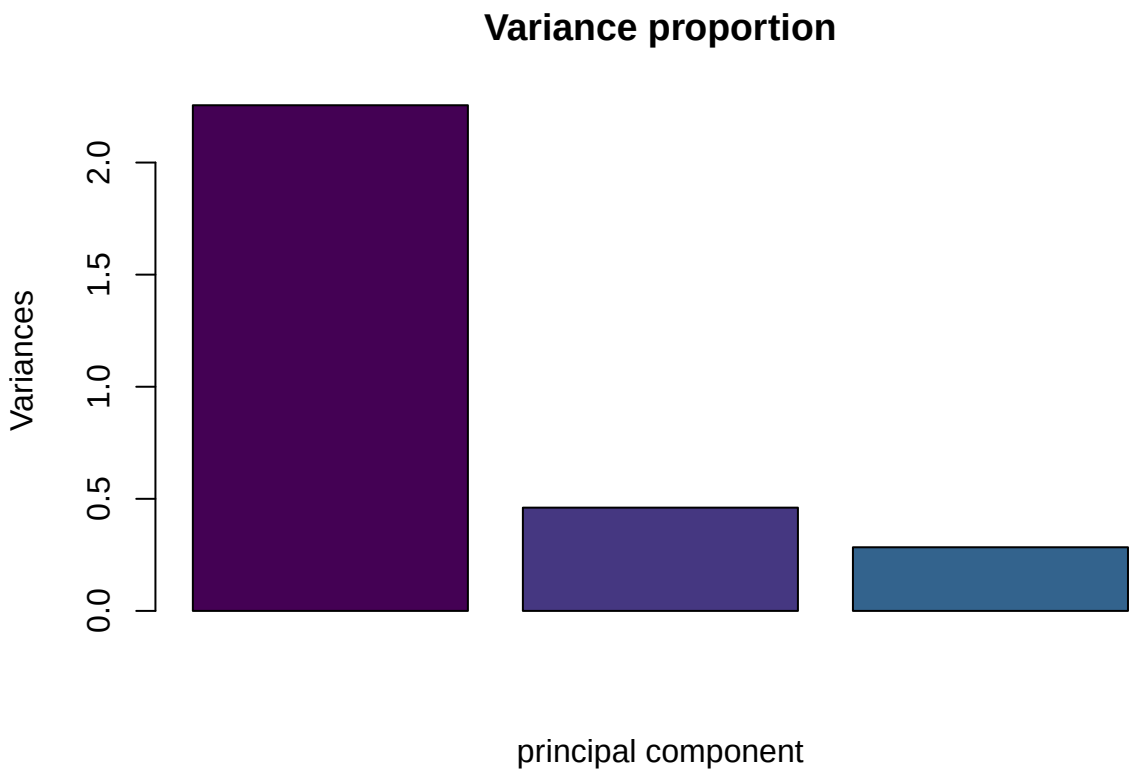
```
numdata <- log(video[, c(3, 4, 5)])
num_pca <- prcomp(numdata, scale = TRUE)
print(num_pca)
```

```
## Standard deviations (1, .., p=3):
## [1] 1.5018790 0.6786647 0.5327042
##
## Rotation (n x k) = (3 x 3):
##                PC1        PC2        PC3
## like    -0.5982274  0.2441831  0.7632160
## comment -0.5820373  0.5222362 -0.6232992
## share   -0.5507782 -0.8170948 -0.1702923
```

```
summary(num_pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3
## Standard deviation     1.5019 0.6787 0.53270
## Proportion of Variance 0.7519 0.1535 0.09459
## Cumulative Proportion  0.7519 0.9054 1.00000
```

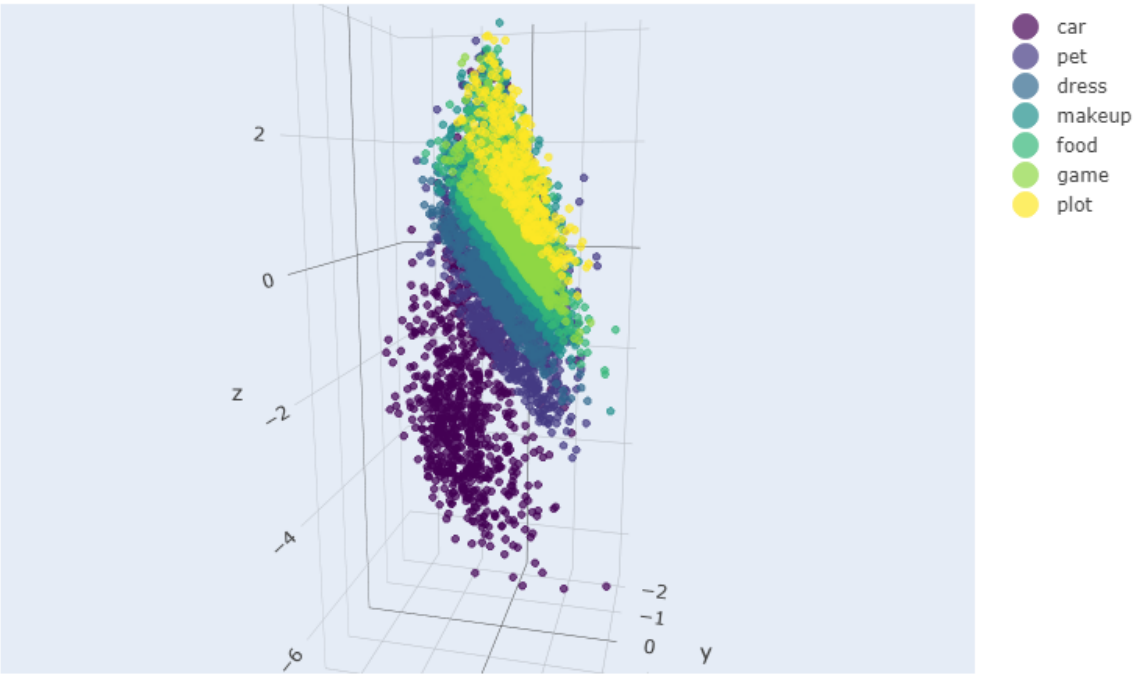
```
plot(
  num_pca,
  col = c("#440154FF", "#453781FF", "#33638DFF"),
  main = "Variance proportion",
  xlab = "principal component"
)
```



- Visualization

```
principal1 <- predict(num_pca)[, 1]
principal2 <- predict(num_pca)[, 2]
principal3 <- predict(num_pca)[, 3]

library(plotly)
fig <-
  plot_ly(
    numdata,
    x = -principal3,
    y = -principal2,
    z = -principal1,
    color = video$type,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
```

```
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



Using the three principal components as the coordinate axis to make a scatter plot, there is an obvious layering effect, and the layering effect of different types is more obvious than before.

So we use PCA to get three principal components and then use SVM to classify.

- SVM based on PCA

```
temp <- data.frame(principal1, principal2, principal3)
temp$type <- video$type

## split the sample set
set.seed(1234)
train <- sample(nrow(temp), 0.7 * nrow(temp))
video.train <- temp[train, ]
video.validate <- temp[-train, ]
table(video.train$type_code)
```

```
## < table of extent 0 >
```

```
table(video.validate$type_code)
```

```
## < table of extent 0 >
```

```
library(e1071)
set.seed(1234)
type <- video.train$type
principal1 <- video.train$principal1
principal2 <- video.train$principal2
principal3 <- video.train$principal3
fit.svm <-
  svm(type ~ principal1 + principal2 + principal3)
fit.svm
```

```
##
## Call:
## svm(formula = type ~ principal1 + principal2 + principal3)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  3660
```

```
svm.pred <- predict(fit.svm, na.omit(video.validate))
svm.perf <- table(na.omit(video.validate)$type,
                  svm.pred,
                  dnn = c("Actual", "Predicted"))
## confusion matrix
svm.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    178  33    35      8   20    7    2
##   pet      7 125    54     28    8   47   11
##   dress    3  50   141     38   18   38    5
##   makeup   0   9    47     86   37   88   15
##   food     0   3    12     50   83  115   24
##   game     0   0     2     23   25  209   31
##   plot     0   0     0      4    9   63   49
```

- Tune paramaters

```
### tuning an RBF support vector machine
#set.seed(1234)
#
### varies the parameters
#tuned <- tune.svm(
#  type ~ principal1 + principal2 + principal3,
#  gamma = 10 ^ (-8:8),
#  cost = 10 ^ (-8:8)
#)
#tuned ## print the best model
```

- Use best parameters

```
fit.svm <-
  svm(
    type ~ principal1 + principal2 + principal3,
    gamma = 0.125,
    cost = 128
  )

svm.pred <- predict(fit.svm, na.omit(video.validate))

## evaluate the cross-validation performance
svm.perf <- table(na.omit(video.validate)$type,
                  svm.pred,
                  dnn = c("Actual", "Predicted"))
## confusion matrix
svm.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    182  25    38     11   20    5    2
##   pet      4 109    69     30   13   46    9
##   dress    3  19   162     46   24   34    5
##   makeup   0   4    39    101   40   84   14
##   food     0   1     4     62   91  107   22
##   game     0   0     1     21   24  213   31
##   plot     0   0     0      8    8   61   48
```

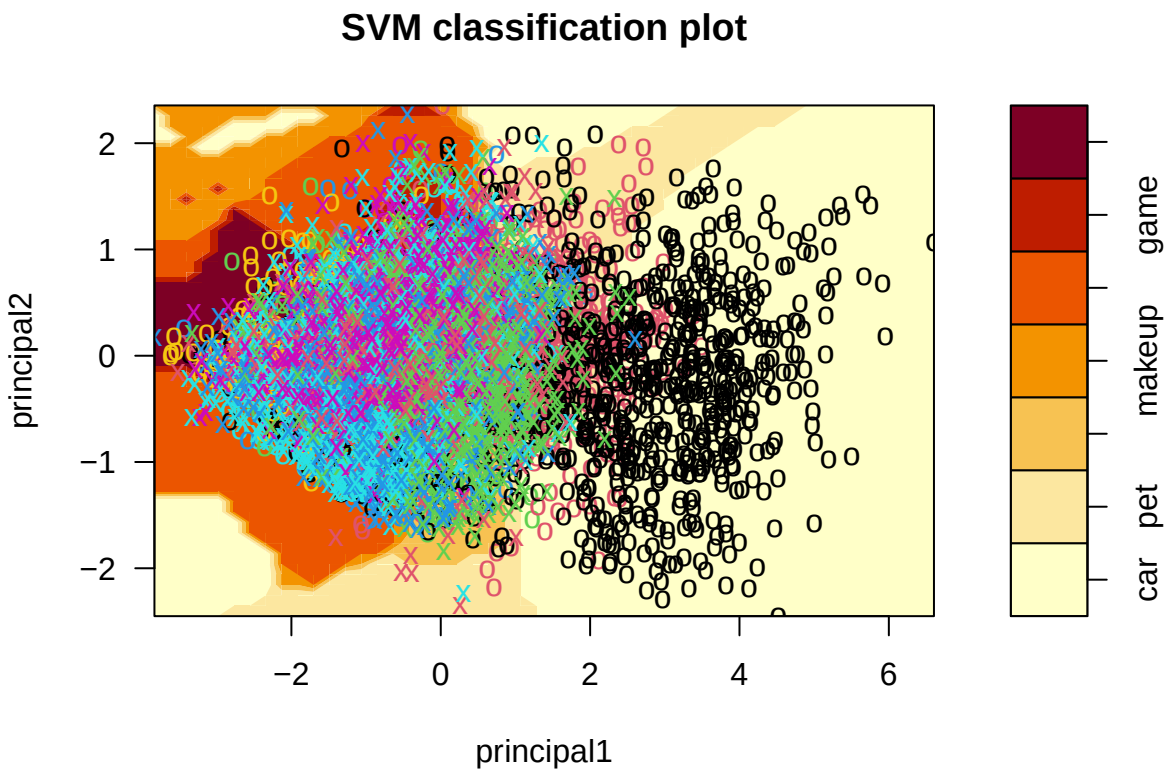
```
## accuracy
sum(diag(svm.perf)) / sum(svm.perf)
```

```
## [1] 0.4923913
```

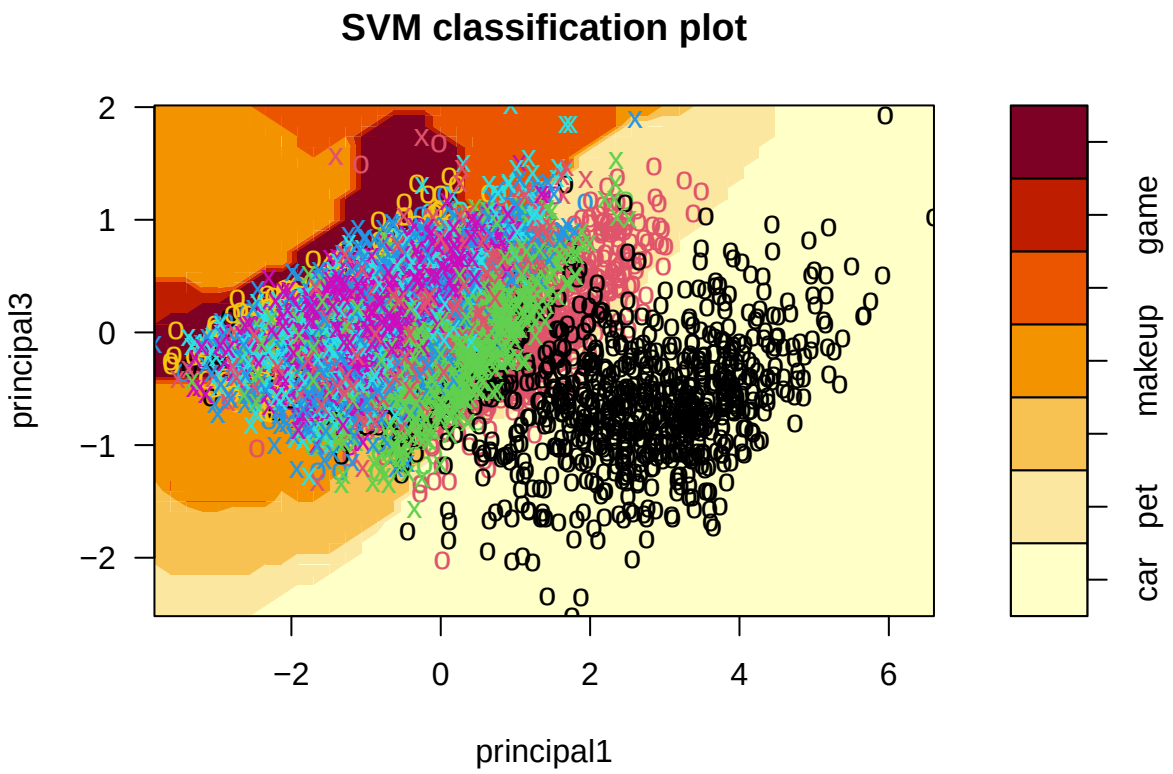
```
## F1score
myF1score(svm.perf, video.validate)
```

```
## [1] 0.4960718
```

```
## visualization
plot(fit.svm, temp, principal2 ~ principal1, slice = list(principal3 = 0))
```




```
plot(fit.svm, temp, principal3 ~ principal1, slice = list(principal2 = 0))
```



Better performance than SVM without PCA.

6.3 Decision Tree

```
## split the sample set
set.seed(1234)
train <- sample(nrow(video), 0.7 * nrow(video))
video.train <- video[train, ]
video.validate <- video[-train, ]
table(video.train$type)
```

```
##
##    car    pet  dress makeup   food   game   plot
##    698    698    658    646    664    693    234
```

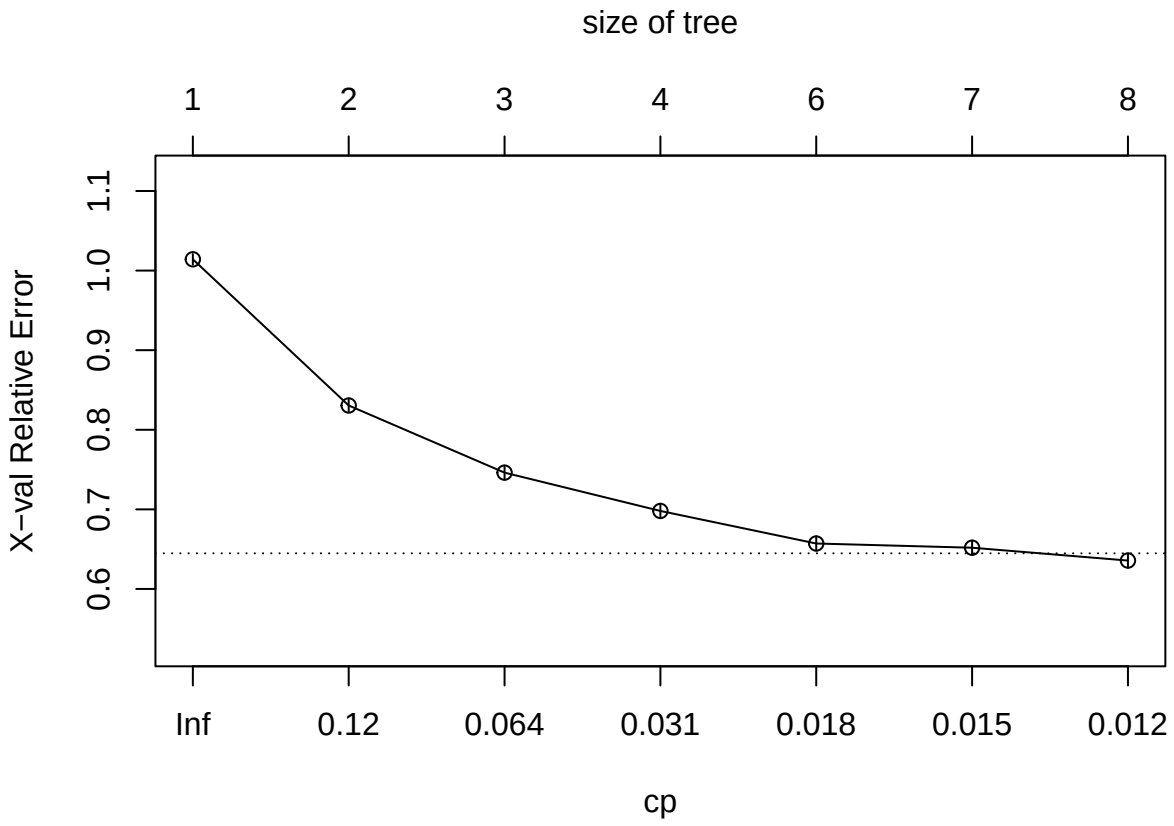
```
table(video.validate$type)
```

```
##
##    car    pet  dress makeup   food   game   plot
##    283    280    293    282    287    290    125
```

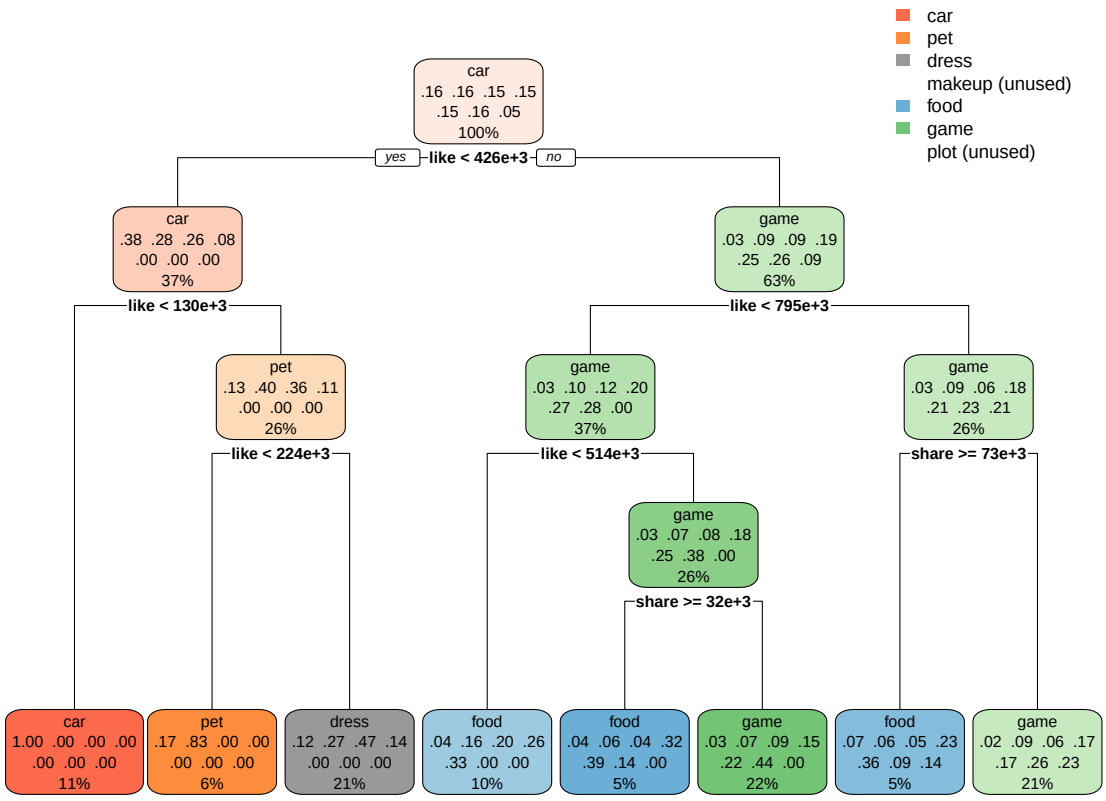
```
## grow the tree
library(rpart)
```

```
##
##     'rpart'

## The following object is masked from 'package:dendextend':
##
##     prune
```

```
set.seed(1234)

dtree <- rpart(
  type ~ like + comment + share + BGM + title,
  data = video.train,
  method = "class",
  parms = list(split = "information")
)
## plot
plotcp(dtree)
```




```
dtree.pruned <- prune(dtree, cp = 0.012)
library(rpart.plot)
rpart.plot(dtree.pruned)
```




```
## prediction
dtree.pred <- predict(dtree.pruned, video.validate, type = "class")
dtree.perf <- table(video.validate$type, dtree.pred,
                    dnn = c("Actual", "Predicted"))
## confusion matrix
dtree.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    182  23    40      0   30    8    0
##   pet      0  98    86      0   33   63    0
##   dress    0   0   188      0   48   57    0
##   makeup   0   0    59      0  104  119    0
##   food     0   0     0      0  135  152    0
##   game     0   0     0      0   27  263    0
##   plot     0   0     0      0   17  108    0
```

```
## accuracy
sum(diag(dtree.perf)) / sum(dtree.perf)
```

```
## [1] 0.4706522
```

Worse performance than PCA+SVM.

6.4 Random Forest

```
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
##     'randomForest'

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:dplyr':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
set.seed(1234)

### grow the forest
fit.forest <-
  randomForest(
    type ~ like + comment + share + BGM + title,
    data = video.train,
    na.action = na.roughfix,
    importance = TRUE
  )

#### classifies new cases
forest.pred <- predict(fit.forest, video.validate)
forest.perf <- table(video.validate$type,
                     forest.pred,
                     dnn = c("Actual", "Predicted"))
## confusion matrix
forest.perf
```

```
##         Predicted
## Actual   car pet dress makeup food game plot
##   car    197  24    24     12   22    2    2
##   pet      6 134    45     20   21   44   10
##   dress   12  42   143     38   31   23    4
##   makeup   4  31    25     90   60   58   14
##   food     0  10    20     40  112   85   20
##   game     2   3    12     28   52  167   26
##   plot     1   1     1     15   12   33   62
```

```r
## accuracy
sum(diag(forest.perf)) / sum(forest.perf)
```

```
## [1] 0.4918478
```

Worse performance than PCA+SVM.

6.5 Use generated data to test the classifier(PCA+SVM)

- Generate new data(n = 20000)

```r
newdata <- gaussian.generate(20000)
```

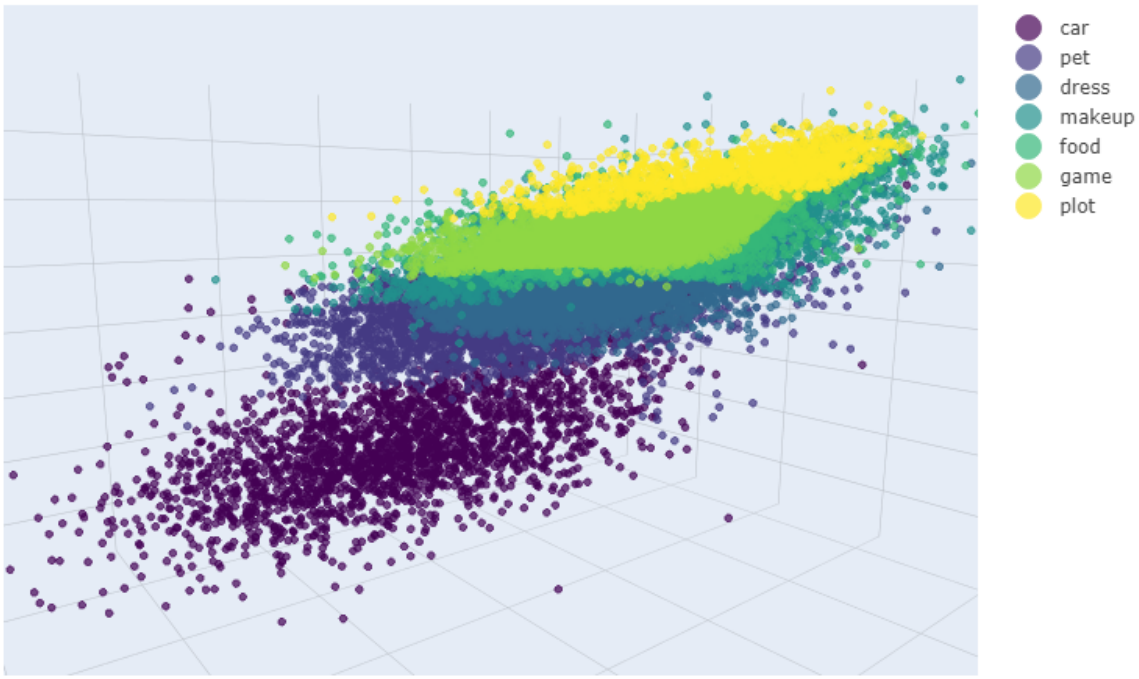
- Classify

```r
## PCA
num_pca <- prcomp(newdata[c(1, 2, 3)], scale = TRUE)
principal1 <- predict(num_pca)[, 1]
principal2 <- predict(num_pca)[, 2]
principal3 <- predict(num_pca)[, 3]
```

```r
## SVM
svm.pred <-
  predict(fit.svm, data.frame(principal1, principal2, principal3))

newdata$type <- svm.pred
```

- Visualization

```r
library(plotly)
library(viridisLite)
temp <- newdata
fig <-
  plot_ly(
    temp,
    x = temp$X2,
    y = temp$X3,
    z = temp$X1,
    color = temp$type,
    colors = viridis(7)
  ) %>%  add_markers(size = 12)

fig <- fig %>%
  layout(title = "",
         scene = list(bgcolor = "#e5ecf6"))
fig
```



The effect is good, our classifier has good generalization performance.

7 Conclusion

非汽车类别(iscar=0)	点赞数	汽车类别(iscar=1)	点赞数
作者类别(once)	1 倍	作者类别(once)	1 倍
作者类别(several)	1.05771 倍	作者类别(several)	1.05771 倍
作者类别(often)	1.144665 倍	作者类别(often)	1.144665 倍
作者类别(frequent)	1.187996 倍	作者类别(frequent)	1.187996 倍
评论数	(评论数)^(0.22887) 倍	评论数	(评论数)^(0.52097) 倍
转发数	(转发数)^(0.09567) 倍	转发数	(转发数)^(0.21391) 倍
视频时长	(exp(视频时长))^(0.0015) 倍	视频时长	(exp(视频时长))^(0.0051) 倍
标题字数	(exp(标题字数))^(0.0014) 倍	标题字数	(exp(标题字数))^(0.0014) 倍
视频类别(宠物)	1 倍	BGM(非原创)	1 倍
视频类别(穿搭)	1.053006 倍	BGM(原创)	1.11154 倍
视频类别(美妆)	1.457779 倍		
视频类别(美食)	1.599777 倍		
视频类别(游戏)	1.806536 倍		
视频类别(剧情)	2.331663 倍		

- 聚类模型：高斯混合模型
- 生成模型：高斯混合模型
- 分类器：PCA+SVM

给所有想成为短视频博主的朋友的建议，希望尽上一份绵薄之力：

入行务必选定领域，剧情易火也需勤劳；
游戏达人或须熬夜，汽车专家移步为好；
时间标题越长越好，细看实则影响很小；
转评数目尽力提高，内容质量必不可少。