R Notebook

This is the mark-up file for the Datenanalyse 2 homework assignment.

```
library("rio")
```

```
## Warning: package 'rio' was built under R version 3.5.3
```

```
x <- import("https://docs.google.com/spreadsheets/d/1SWEakSjZUvvV3w8peOf5FHrGI9NTEDls3c9zETVZ5kQ/export

str(x)
```

```
## 'data.frame':    720 obs. of  31 variables:
##  $ Artist_Albums_Number             : int  0 0 1 1 1 1 1 1 1 1 ...
##  $ Artist_Albums_Tracks_Number      : int  0 0 8 8 8 8 8 8 8 8 ...
##  $ Artist_Appearances_Number        : int  9 9 2 2 2 2 2 2 2 2 ...
##  $ Artist_Appearances_Tracks_Number : int  502 502 30 30 30 30 30 30 30 30 ...
##  $ Artist_Compilations_Number       : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Artist_Compilations_Tracks_Number: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Artist_Follower                  : int  713401 713401 601346 601346 601346 601346 601346 601346 60
##  $ Artist_ID                        : chr  "2NjfBq1NflQcKSeiDooVjY" "2NjfBq1NflQcKSeiDooVjY" "1qQLhyr
##  $ Artist_Popularity                : int  91 91 83 83 83 83 83 83 83 83 ...
##  $ Artist_Singles_Number            : int  3 3 15 15 15 15 15 15 15 15 ...
##  $ Artist_Singles_Tracks_Number     : int  10 10 15 15 15 15 15 15 15 15 ...
##  $ Genre                            : chr  "pop" "pop" "Hip Hop" "Hip Hop" ...
##  $ Release_Date                     : chr  "2019-05-10" "2019-07-15" "2019-10-25" "2019-08-23" ...
##  $ Streams                          : int  106824437 2327995 79193552 54619683 48552840 46784729 4343
##  $ Track_Artist                     : chr  "Tones and I" "Tones and I" "Apache 207" "Apache 207" ...
##  $ Track_Duration_ms                : int  209754 200755 157093 158853 176066 163146 139693 191760 17
##  $ Track_ID                         : chr  "1rgnBhdG2JDFTbYkYRZAku" "2grAr8pWMuLWn8ZYEE9wDV" "6hw1Sy
##  $ Track_Popularity                 : int  76 72 78 77 73 75 73 75 69 69 ...
##  $ Track_Title                      : chr  "Dance Monkey" "Never Seen the Rain" "Roller" "Roller" ..
##  $ Title_Artist_Google_searches_11m : int  20904 572 8880 8880 1975 1156 3260 10880 220 568 ...
##  $ Title_Artist_Youtube_searches_11m: int  308911 7320 7660 7660 1530 990 2240 7915 154 441 ...
##  $ Title_Google_searches_11m        : int  1288732 2799 4805454 4805454 47025 33165 47709 45925 8977
##  $ Title_Youtube_searches_11m       : int  18353181 33600 3446454 3446454 32325 28872 38436 31975 593
##  $ Total_tracks                     : int  512 512 53 53 53 53 53 53 53 53 ...
##  $ Artist_Google_searches_11m       : int  299212 299212 1468281 1468281 1468281 1468281 1468281 1468
##  $ Artist_Youtube_searches_11m      : int  2451500 2451500 1076400 1076400 1076400 1076400 1076400 10
##  $ commentCount                     : num  172604 2272 22183 22183 13376 ...
##  $ dislikeCount                     : int  317322 3194 27802 27802 11440 12957 10493 605 5333 4287 .
##  $ likeCount                        : int  7424686 109395 748270 748270 385252 378780 299481 24361 2
##  $ video_ID                         : chr  "q0hyYWKXF0Q" "UdRJY-jlEhQ" "Fo3DAhiNKQo" "Fo3DAhiNKQo" .
##  $ viewsCount                       :integer64 738528171 10258864 66995452 66995452 22170062 2864717
```

```
x$commentCount <- as.integer(x$commentCount)
x$viewsCount <- as.numeric(x$viewsCount)
```

```
library("dplyr")
```

```
## Warning: package 'dplyr' was built under R version 3.5.3
```

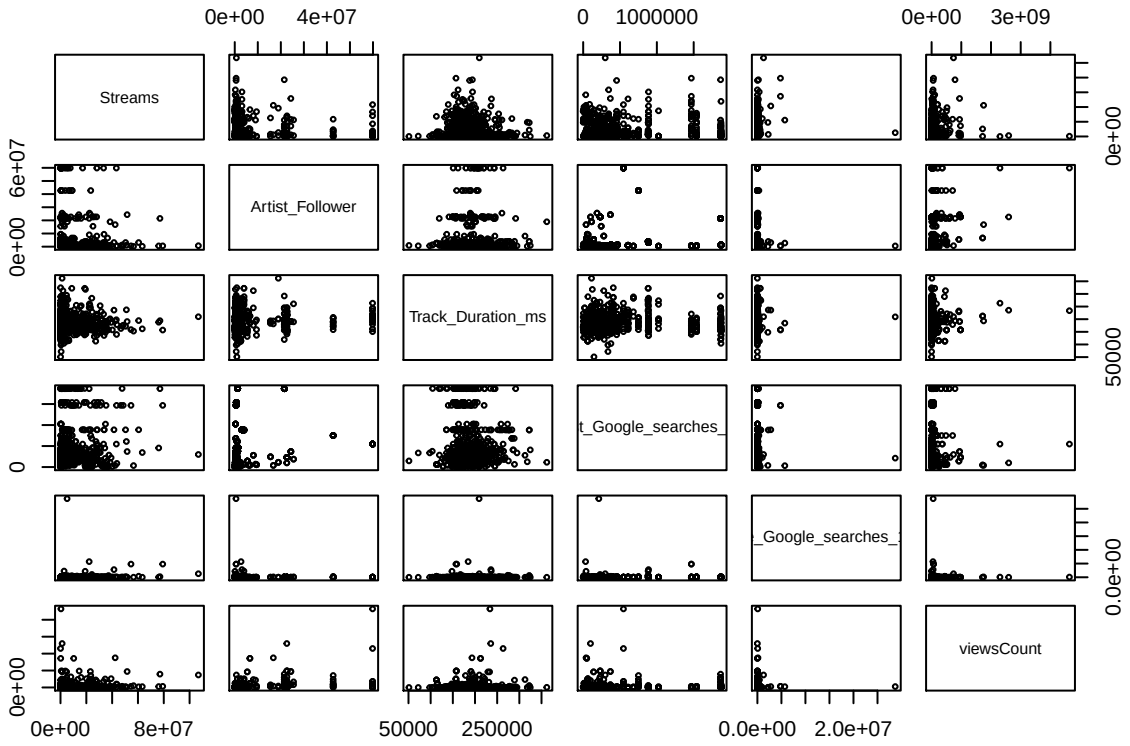
```
##
## Attaching package: 'dplyr'
```

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

```
##
##     filter, lag
```

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

```
drop.cols <- c('Artist_ID', 'Genre', 'Release_Date', 'Track_Artist', 'Track_ID', 'Track_Title', 'video_
numeric_x <- select(x, -one_of(drop.cols))

keep.cols <- c('Streams', 'Artist_Follower', 'Track_Duration_ms', 'Artist_Google_searches_11m', 'Title_
               'viewsCount')

# keep.cols <- c('Streams', 'viewsCount', 'Title_Youtube_searches_11m')

selected_pairs <- select(x, keep.cols)

pairs(selected_pairs, cex=0.5)
```



Descriptive statistics

```
summary(numeric_x)
```

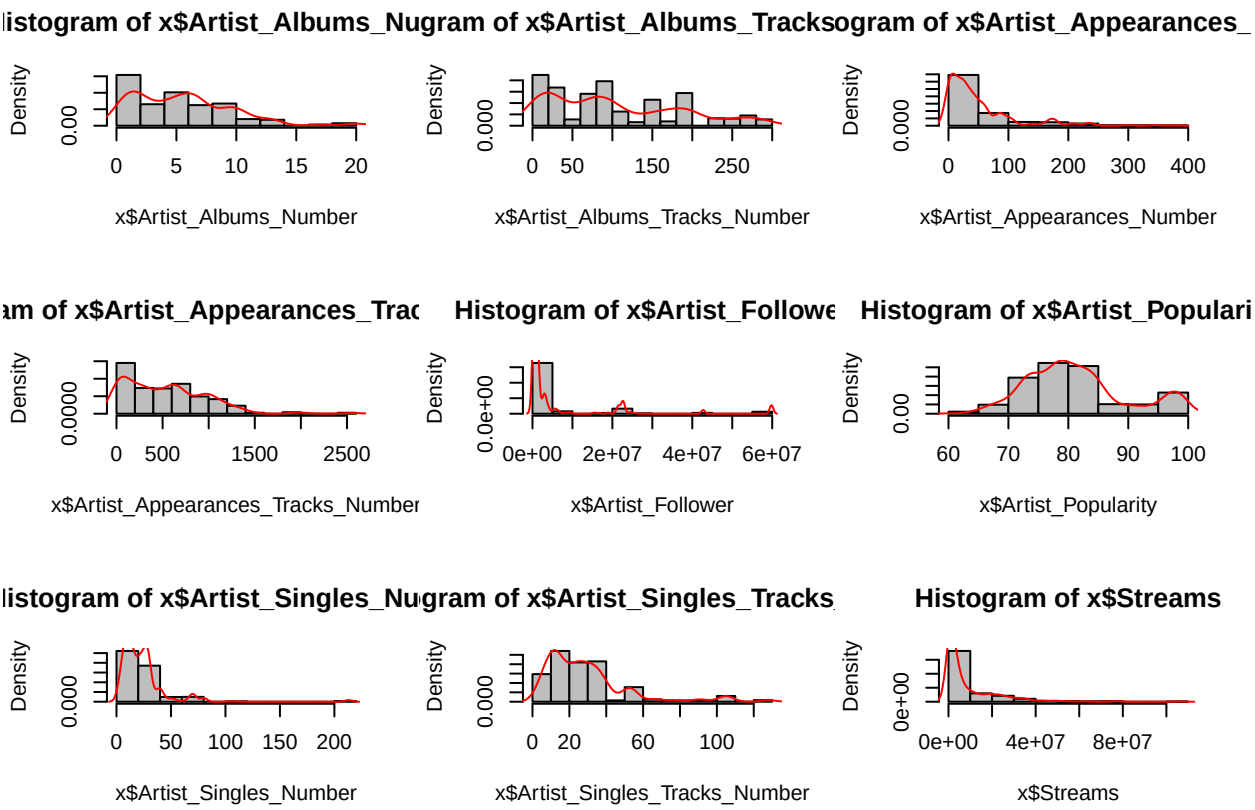
```
##  Artist_Albums_Number Artist_Albums_Tracks_Number
##  Min.   : 0.000       Min.   :  0.0
##  1st Qu.: 2.000       1st Qu.: 26.0
##  Median : 5.000       Median : 86.0
##  Mean   : 5.508       Mean   :103.2
```

```
##  3rd Qu.: 8.000       3rd Qu.:159.0
##  Max.   :20.000       Max.   :299.0
##
##  Artist_Appearances_Number Artist_Appearances_Tracks_Number
##  Min.   :  0.00            Min.   :   0.0
##  1st Qu.: 12.00            1st Qu.: 140.0
##  Median : 28.00            Median : 479.0
##  Mean   : 48.43            Mean   : 526.9
##  3rd Qu.: 59.00            3rd Qu.: 786.0
##  Max.   :375.00            Max.   :2583.0
##
##  Artist_Compilations_Number Artist_Compilations_Tracks_Number
##  Min.   :0.0000             Min.   : 0.000
##  1st Qu.:0.0000             1st Qu.: 0.000
##  Median :0.0000             Median : 0.000
##  Mean   :0.1056             Mean   : 2.579
##  3rd Qu.:0.0000             3rd Qu.: 0.000
##  Max.   :2.0000             Max.   :57.000
##
##  Artist_Follower    Artist_Popularity Artist_Singles_Number
##  Min.   :    9449   Min.   :60.00     Min.   :  3.00
##  1st Qu.:  575873   1st Qu.:74.00     1st Qu.: 11.00
##  Median :  889326   Median :80.00     Median : 19.00
##  Mean   : 5710132   Mean   :81.22     Mean   : 23.18
##  3rd Qu.: 3129993   3rd Qu.:84.25     3rd Qu.: 29.00
##  Max.   :59828212   Max.   :99.00     Max.   :213.00
##
##  Artist_Singles_Tracks_Number    Streams          Track_Duration_ms
##  Min.   :  4.00               Min.   :    43688   Min.   : 51104
##  1st Qu.: 12.00               1st Qu.:   799953   1st Qu.:162634
##  Median : 26.00               Median :  3033628   Median :182656
##  Mean   : 29.01               Mean   :  8595051   Mean   :187680
##  3rd Qu.: 35.00               3rd Qu.: 11802780   3rd Qu.:204396
##  Max.   :128.00               Max.   :106824437   Max.   :361946
##
##  Track_Popularity Title_Artist_Google_searches_11m
##  Min.   : 0.00    Min.   :     0
##  1st Qu.:50.00    1st Qu.:     1
##  Median :58.00    Median :  1215
##  Mean   :58.65    Mean   : 10283
##  3rd Qu.:69.00    3rd Qu.:  7100
##  Max.   :99.00    Max.   :398000
##
##  Title_Artist_Youtube_searches_11m Title_Google_searches_11m
##  Min.   :      0                   Min.   :       0
##  1st Qu.:     10                   1st Qu.:       0
##  Median :   1292                   Median :    4666
##  Mean   :  58811                   Mean   :  106718
##  3rd Qu.:   9904                   3rd Qu.:   32263
##  Max.   :6870200                   Max.   :28689090
##
##  Title_Youtube_searches_11m  Total_tracks    Artist_Google_searches_11m
##  Min.   :        0          Min.   :   5.0   Min.   :      1
##  1st Qu.:        0          1st Qu.: 239.0   1st Qu.: 183522
```

```
##  Median :     6488          Median : 678.0   Median : 336545
##  Mean   :   662186          Mean   : 661.6   Mean   : 513368
##  3rd Qu.:   179120          3rd Qu.: 955.8   3rd Qu.: 608772
##  Max.   :134580909          Max.   :2699.0   Max.   :1871000
##
##  Artist_Youtube_searches_11m  commentCount       dislikeCount
##  Min.   :      10            Min.   :     0.0   Min.   :      2
##  1st Qu.:  270454            1st Qu.:   698.5   1st Qu.:    586
##  Median :  504090            Median :  5281.0   Median :   4594
##  Mean   : 2179428            Mean   : 32745.3   Mean   :  33026
##  3rd Qu.: 1705727            3rd Qu.: 19694.0   3rd Qu.:  18845
##  Max.   :28298181            Max.   :934238.0   Max.   :1203541
##                              NA's   :5
##    likeCount          viewsCount
##  Min.   :      32   Min.   :3.290e+03
##  1st Qu.:   24622   1st Qu.:1.698e+06
##  Median :  138002   Median :6.880e+06
##  Mean   :  808778   Mean   :7.298e+07
##  3rd Qu.:  427237   3rd Qu.:3.175e+07
##  Max.   :22120897   Max.   :4.641e+09
##
```

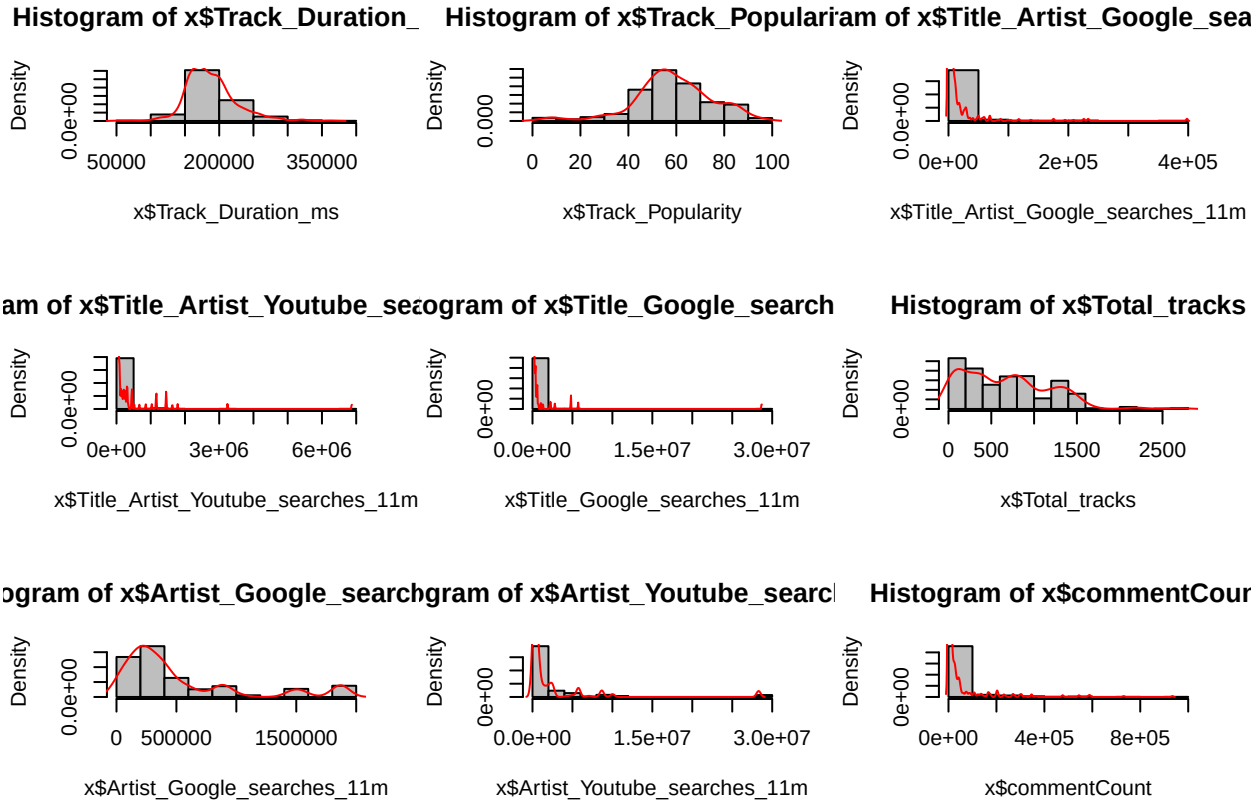
Histograms and kernel density plots of base variables

```
par(mfrow=c(3,3))

hist(x$Artist_Albums_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Albums_Number), col = "red")

hist(x$Artist_Albums_Tracks_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Albums_Tracks_Number), col = "red")

hist(x$Artist_Appearances_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Appearances_Number), col = "red")

hist(x$Artist_Appearances_Tracks_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Appearances_Tracks_Number), col = "red")

hist(x$Artist_Follower, probability = TRUE, col = "gray")
lines(density(x$Artist_Follower), col = "red")

hist(x$Artist_Popularity, probability = TRUE, col = "gray")
lines(density(x$Artist_Popularity), col = "red")

hist(x$Artist_Singles_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Singles_Number), col = "red")

hist(x$Artist_Singles_Tracks_Number, probability = TRUE, col = "gray")
lines(density(x$Artist_Singles_Tracks_Number), col = "red")

hist(x$Streams, probability = TRUE, col = "gray")
lines(density(x$Streams), col = "red")
```

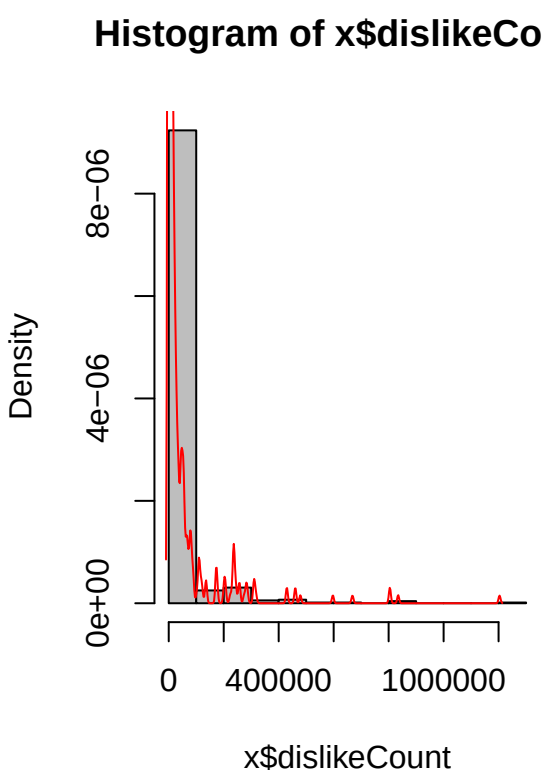


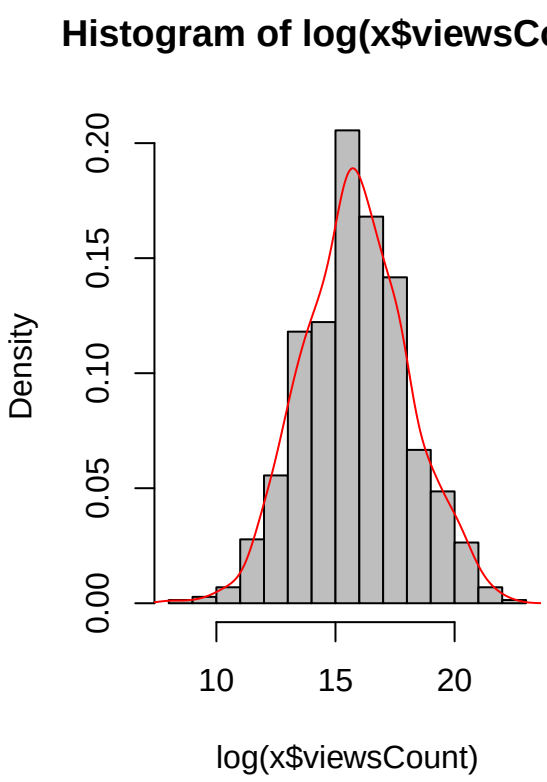

```r
par(mfrow=c(3,3))

hist(x$Track_Duration_ms, probability = TRUE, col = "gray")
lines(density(x$Track_Duration_ms), col = "red")

hist(x$Track_Popularity, probability = TRUE, col = "gray")
lines(density(x$Track_Popularity), col = "red")

hist(x$Title_Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(x$Title_Artist_Google_searches_11m), col = "red")

hist(x$Title_Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(x$Title_Artist_Youtube_searches_11m), col = "red")

hist(x$Title_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(x$Title_Google_searches_11m), col = "red")

hist(x$Total_tracks, probability = TRUE, col = "gray")
lines(density(x$Total_tracks), col = "red")

hist(x$Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(x$Artist_Google_searches_11m), col = "red")

hist(x$Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(x$Artist_Youtube_searches_11m), col = "red")
```

```r
hist(x$commentCount, probability = TRUE, col = "gray")
lines(density(x$commentCount, na.rm = TRUE), col = "red")
```




```r
par(mfrow=c(1,2))

hist(x$dislikeCount, probability = TRUE, col = "gray")
lines(density(x$dislikeCount), col = "red")

hist(log(x$viewsCount), probability = TRUE, col = "gray")
lines(density(log(x$viewsCount), na.rm = TRUE), col = "red")
```






```
#hist(x$viewsCount, probability = TRUE, col = "gray")
#lines(density(x$viewsCount, na.rm = TRUE), col = "red")
```

Distribution testing

1) Normality

```
strictly_positive_variables <- c('Artist_Follower', 'Artist_Popularity', 'Artist_Singles_Number',
                                 'Artist_Singles_Tracks_Number' , 'Streams', 'Track_Duration_ms', 'Total
                                 'viewsCount', 'Artist_Google_searches_11m', 'Artist_Youtube_searches_1


summary(select(x, strictly_positive_variables))
```

```
##  Artist_Follower    Artist_Popularity Artist_Singles_Number
##  Min.   :    9449   Min.   :60.00     Min.   :  3.00
##  1st Qu.:  575873   1st Qu.:74.00     1st Qu.: 11.00
##  Median :  889326   Median :80.00     Median : 19.00
##  Mean   : 5710132   Mean   :81.22     Mean   : 23.18
##  3rd Qu.: 3129993   3rd Qu.:84.25     3rd Qu.: 29.00
##  Max.   :59828212   Max.   :99.00     Max.   :213.00
##  Artist_Singles_Tracks_Number    Streams          Track_Duration_ms
##  Min.   :  4.00               Min.   :    43688   Min.   : 51104
##  1st Qu.: 12.00               1st Qu.:   799953   1st Qu.:162634
##  Median : 26.00               Median :  3033628   Median :182656
##  Mean   : 29.01               Mean   :  8595051   Mean   :187680
##  3rd Qu.: 35.00               3rd Qu.: 11802780   3rd Qu.:204396
##  Max.   :128.00               Max.   :106824437   Max.   :361946
```

```
##   Total_tracks      viewsCount        Artist_Google_searches_11m
##  Min.   :   5.0   Min.   :3.290e+03   Min.   :      1
##  1st Qu.: 239.0   1st Qu.:1.698e+06   1st Qu.: 183522
##  Median : 678.0   Median :6.880e+06   Median : 336545
##  Mean   : 661.6   Mean   :7.298e+07   Mean   : 513368
##  3rd Qu.: 955.8   3rd Qu.:3.175e+07   3rd Qu.: 608772
##  Max.   :2699.0   Max.   :4.641e+09   Max.   :1871000
##  Artist_Youtube_searches_11m
##  Min.   :      10
##  1st Qu.:  270454
##  Median :  504090
##  Mean   : 2179428
##  3rd Qu.: 1705727
##  Max.   :28298181
```

```
for (i in 1:length(strictly_positive_variables)){

  column_name <- strictly_positive_variables[i]

  sub_df <- numeric_x[column_name]
  sub_df <- as.numeric(as.character(unlist(sub_df[[1]])))

  test_statistic <- ks.test(sub_df, "pnorm", mean=mean(sub_df), sd=sd(sub_df))$statistic
  critical_value <- 1.3581 / sqrt (length(sub_df))

  if (test_statistic > critical_value) {
message(paste(" ", column_name , " is not approximately normally distributed.", test_statistic, critical
} else {
message(paste(" ", column_name , " is approximately normally distributed!", test_statistic, critical_val
}}
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Artist_Follower  is not approximately normally distributed. 0.383035516591054 0.0506133986707077
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Artist_Popularity  is not approximately normally distributed. 0.119291819521821 0.0506133986707077
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Artist_Singles_Number  is not approximately normally distributed. 0.255514154746977 0.050613398670
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Artist_Singles_Tracks_Number  is not approximately normally distributed. 0.161639370078695 0.050613
```

```
##   Streams  is not approximately normally distributed. 0.248187587219419 0.0506133986707077
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Track_Duration_ms  is not approximately normally distributed. 0.0724951409388148 0.0506133986707077
```

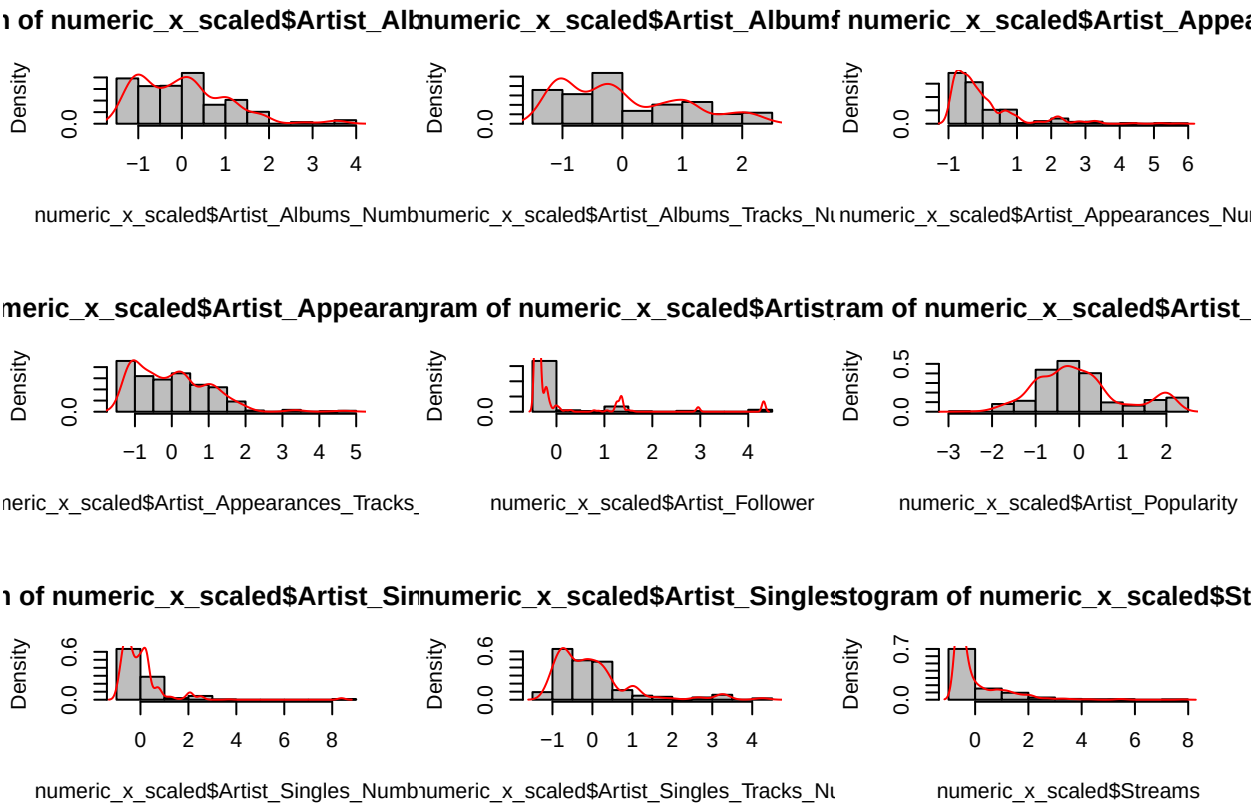
```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##   Total_tracks  is not approximately normally distributed. 0.11590620951236 0.0506133986707077
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
##   viewsCount  is not approximately normally distributed. 0.395670837683742 0.0506133986707077
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
##   Artist_Google_searches_11m  is not approximately normally distributed. 0.241095798188388 0.0506133
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
##   Artist_Youtube_searches_11m  is not approximately normally distributed. 0.332369983188504 0.050613
```

None of the strictly positive variables in their base specification passes the KS test.

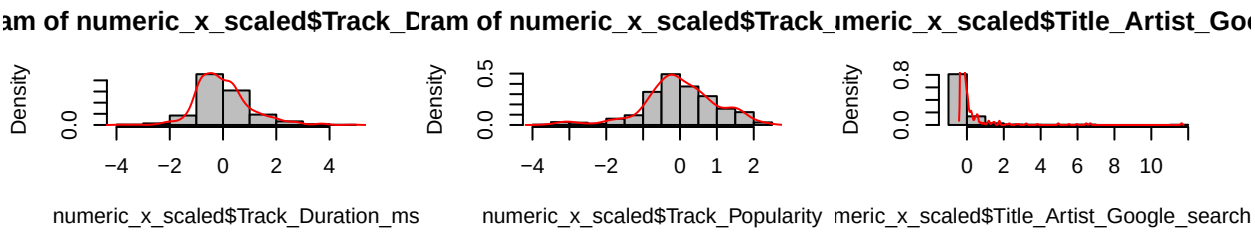
2) Standard normality

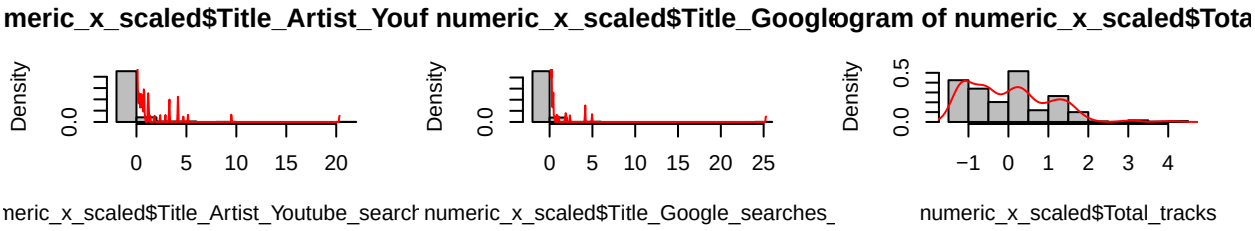
```r
numeric_x_scaled <- scale(numeric_x, center = TRUE, scale = TRUE)
numeric_x_scaled <- as.data.frame(numeric_x_scaled)

par(mfrow=c(3,3))

hist(numeric_x_scaled$Artist_Albums_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Albums_Number), col = "red")

hist(numeric_x_scaled$Artist_Albums_Tracks_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Albums_Tracks_Number), col = "red")

hist(numeric_x_scaled$Artist_Appearances_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Appearances_Number), col = "red")

hist(numeric_x_scaled$Artist_Appearances_Tracks_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Appearances_Tracks_Number), col = "red")

hist(numeric_x_scaled$Artist_Follower, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Follower), col = "red")

hist(numeric_x_scaled$Artist_Popularity, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Popularity), col = "red")

hist(numeric_x_scaled$Artist_Singles_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Singles_Number), col = "red")

hist(numeric_x_scaled$Artist_Singles_Tracks_Number, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Singles_Tracks_Number), col = "red")

hist(numeric_x_scaled$Streams, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Streams), col = "red")
```

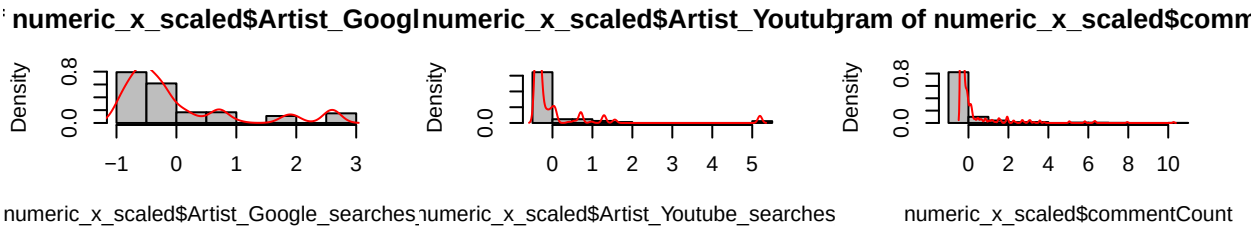



```r
par(mfrow=c(3,3))

hist(numeric_x_scaled$Track_Duration_ms, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Track_Duration_ms), col = "red")

hist(numeric_x_scaled$Track_Popularity, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Track_Popularity), col = "red")

hist(numeric_x_scaled$Title_Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Title_Artist_Google_searches_11m), col = "red")

hist(numeric_x_scaled$Title_Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Title_Artist_Youtube_searches_11m), col = "red")

hist(numeric_x_scaled$Title_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Title_Google_searches_11m), col = "red")

hist(numeric_x_scaled$Total_tracks, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Total_tracks), col = "red")

hist(numeric_x_scaled$Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Google_searches_11m), col = "red")

hist(numeric_x_scaled$Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$Artist_Youtube_searches_11m), col = "red")
```

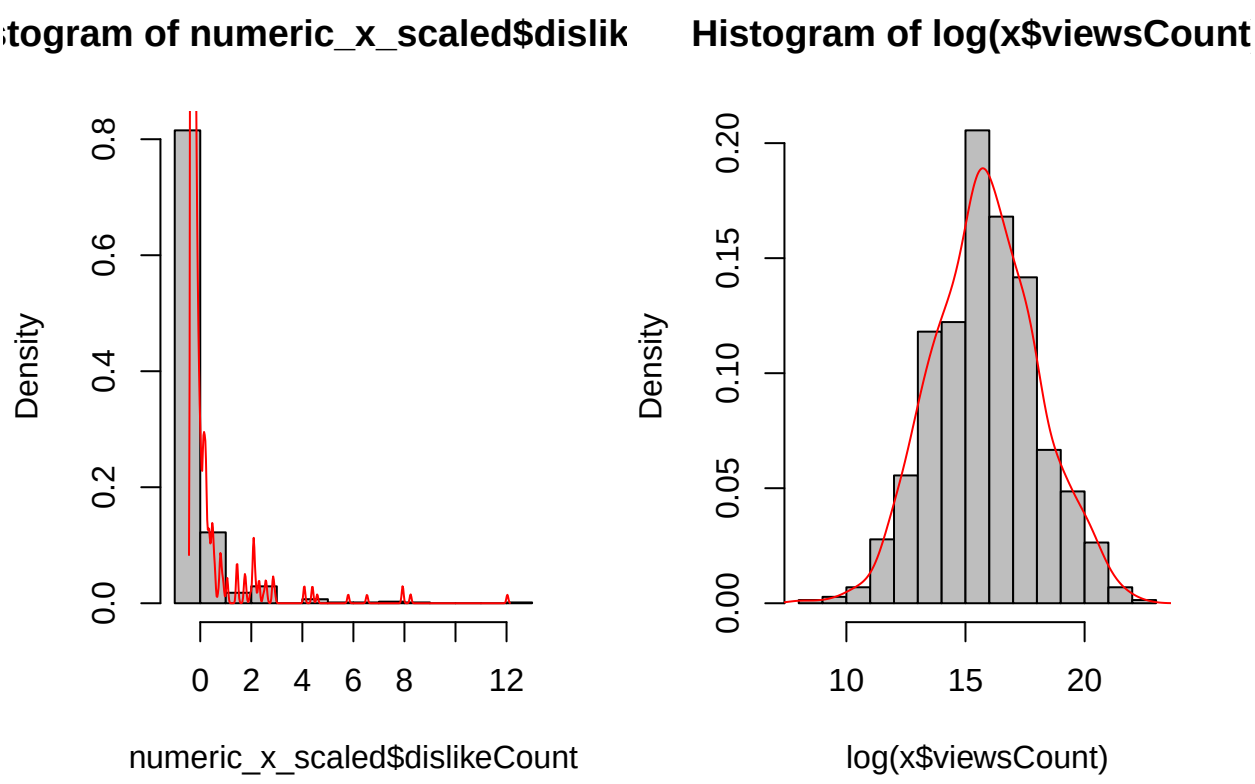
```r
hist(numeric_x_scaled$commentCount, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$commentCount, na.rm = TRUE), col = "red")
```








```r
par(mfrow=c(1,2))

hist(numeric_x_scaled$dislikeCount, probability = TRUE, col = "gray")
lines(density(numeric_x_scaled$dislikeCount), col = "red")

hist(log(x$viewsCount), probability = TRUE, col = "gray")
lines(density(log(x$viewsCount)), col = "red")
```




```
for (i in 1:length(strictly_positive_variables)){

  column_name <- strictly_positive_variables[i]

  sub_df <- numeric_x_scaled[column_name]
  sub_df <- as.numeric(as.character(unlist(sub_df[[1]])))

  test_statistic <- ks.test(sub_df, "pnorm", mean=mean(sub_df), sd=sd(sub_df))$statistic
  critical_value <- 1.3581 / sqrt (length(sub_df))

  if (test_statistic > critical_value) {
message(paste(" Z-transformed ", column_name , " is not approximately normally distributed.", test_stati
} else {
message(paste(" Z-transformed ", column_name , " is approximately normally distributed!", test_statistic
}}
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##  Z-transformed  Artist_Follower  is not approximately normally distributed. 0.383035516591054 0.0506
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##  Z-transformed  Artist_Popularity  is not approximately normally distributed. 0.119291819521822 0.050
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

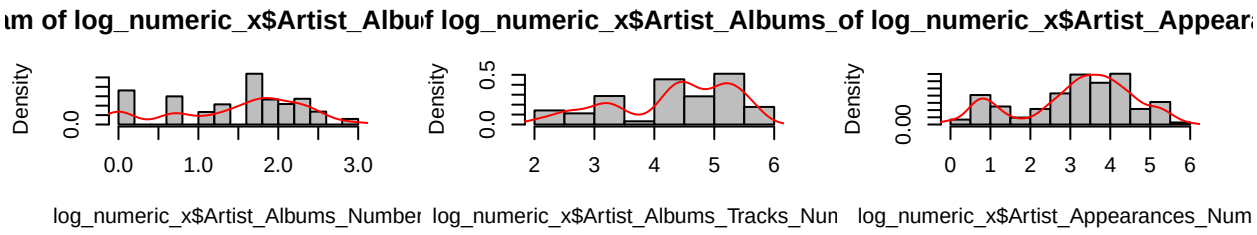
```
##  Z-transformed  Artist_Singles_Number  is not approximately normally distributed. 0.255514154746977 (

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  Artist_Singles_Tracks_Number  is not approximately normally distributed. 0.161639370(

##  Z-transformed  Streams  is not approximately normally distributed. 0.248187587219419 0.050613398670

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  Track_Duration_ms  is not approximately normally distributed. 0.0724951409388148 0.05

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  Total_tracks  is not approximately normally distributed. 0.11590620951236 0.0506133981

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  viewsCount  is not approximately normally distributed. 0.395670837683742 0.0506133980

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  Artist_Google_searches_11m  is not approximately normally distributed. 0.241095798188

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Z-transformed  Artist_Youtube_searches_11m  is not approximately normally distributed. 0.332369983181
```

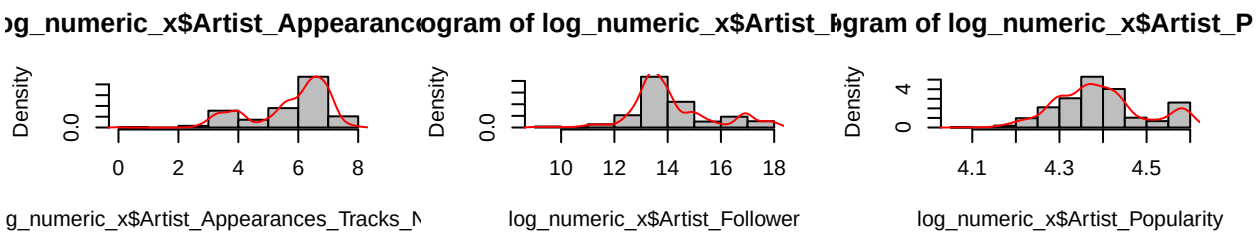
Again, none of the z-transformed variables is approximately normally distributed, however only Track_Duration_ms is close to the critical value at $\alpha = 0.05$.

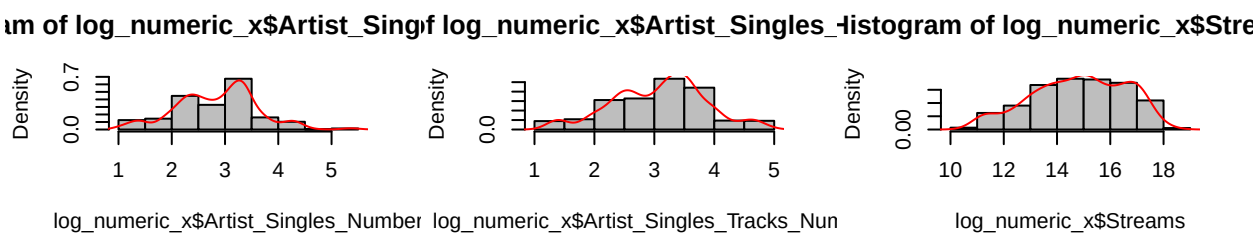
3) Log-normality

```r
log_numeric_x <- log(numeric_x)

par(mfrow=c(3,3))

hist(log_numeric_x$Artist_Albums_Number, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Albums_Number), col = "red")

hist(log_numeric_x$Artist_Albums_Tracks_Number, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Albums_Tracks_Number), col = "red")

hist(log_numeric_x$Artist_Appearances_Number, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Appearances_Number), col = "red")

hist(log_numeric_x$Artist_Appearances_Tracks_Number, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Appearances_Tracks_Number), col = "red")

hist(log_numeric_x$Artist_Follower, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Follower), col = "red")

hist(log_numeric_x$Artist_Popularity, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Popularity), col = "red")

hist(log_numeric_x$Artist_Singles_Number, probability = TRUE, col = "gray")
```

```
lines(density(log_numeric_x$Artist_Singles_Number), col = "red")

hist(log_numeric_x$Artist_Singles_Tracks_Number, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Singles_Tracks_Number), col = "red")

hist(log_numeric_x$Streams, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Streams), col = "red")
```

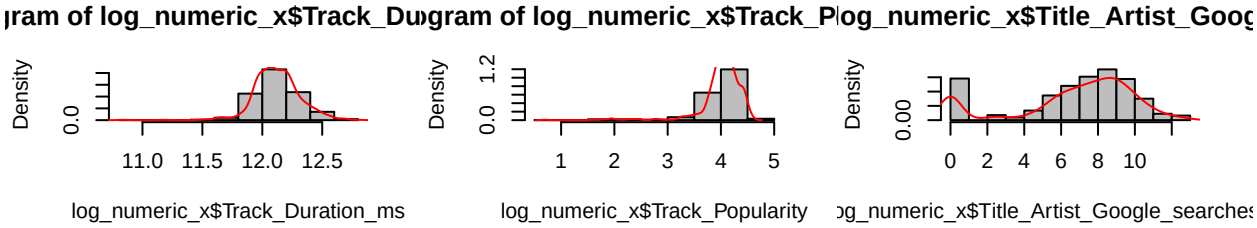


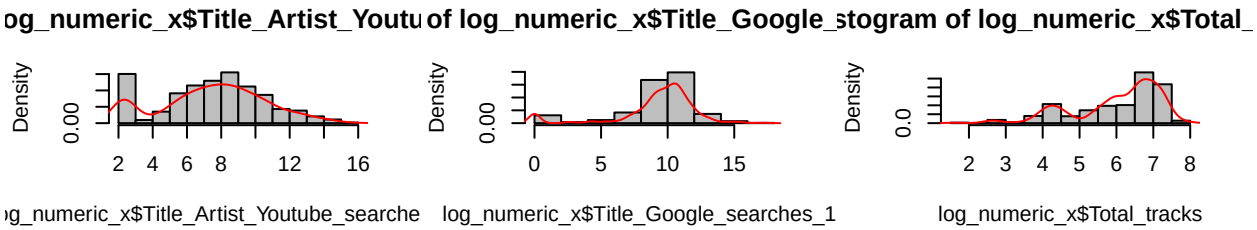


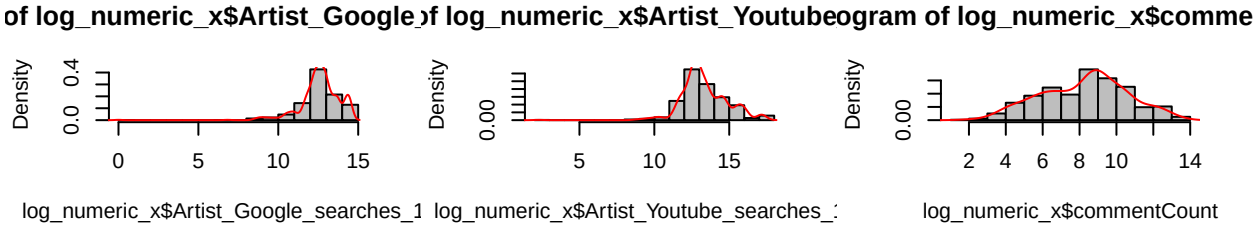



```
par(mfrow=c(3,3))

hist(log_numeric_x$Track_Duration_ms, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Track_Duration_ms), col = "red")

hist(log_numeric_x$Track_Popularity, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Track_Popularity), col = "red")

hist(log_numeric_x$Title_Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Title_Artist_Google_searches_11m), col = "red")

hist(log_numeric_x$Title_Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Title_Artist_Youtube_searches_11m), col = "red")

hist(log_numeric_x$Title_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Title_Google_searches_11m), col = "red")

hist(log_numeric_x$Total_tracks, probability = TRUE, col = "gray")
```

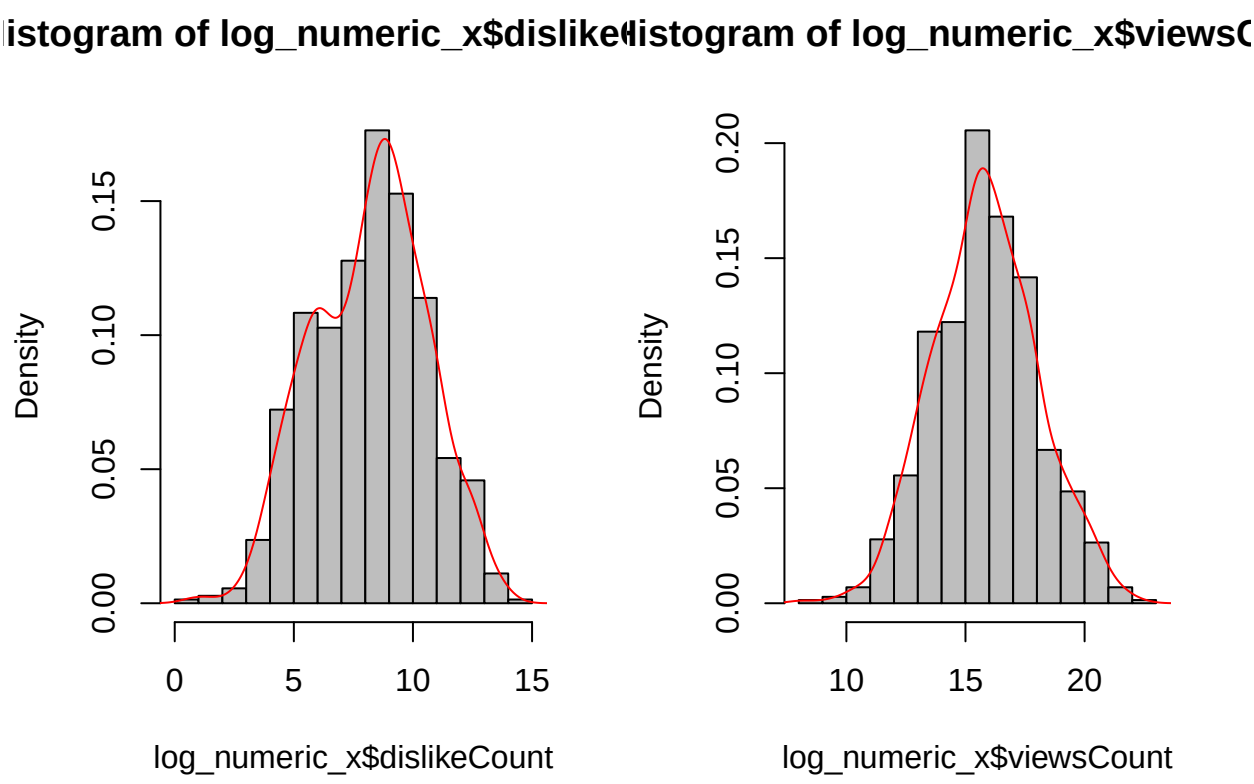
```r
lines(density(log_numeric_x$Total_tracks), col = "red")

hist(log_numeric_x$Artist_Google_searches_11m, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Google_searches_11m), col = "red")

hist(log_numeric_x$Artist_Youtube_searches_11m, probability = TRUE, col = "gray")
lines(density(log_numeric_x$Artist_Youtube_searches_11m), col = "red")

hist(log_numeric_x$commentCount, probability = TRUE, col = "gray")
lines(density(log_numeric_x$commentCount, na.rm = TRUE), col = "red")
```








```r
par(mfrow=c(1,2))

hist(log_numeric_x$dislikeCount, probability = TRUE, col = "gray")
lines(density(log_numeric_x$dislikeCount), col = "red")

hist(log_numeric_x$viewsCount, probability = TRUE, col = "gray")
lines(density(log_numeric_x$viewsCount), col = "red")
```




```
for (i in 1:length(strictly_positive_variables)){

  column_name <- strictly_positive_variables[i]

  sub_df <- numeric_x[column_name]
  sub_df <- sub_df[sub_df > 1]
  sub_df <- log(sub_df)
  # sub_df <- as.numeric(as.character(unlist(sub_df[[1]])))

  test_statistic <- ks.test(sub_df, "pnorm", mean=mean(sub_df), sd=sd(sub_df))$statistic
  critical_value <- 1.3581 / sqrt (length(sub_df))

  if (test_statistic > critical_value) {
message(paste(" Log-transformed ", column_name , " is not approximately normally distributed.", test_sta
} else {
message(paste(" Log-ransformed ", column_name , " is approximately normally distributed!", test_statist
}}
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##  Log-transformed  Artist_Follower  is not approximately normally distributed. 0.200068126290355 0.05
```

```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test
```

```
##  Log-transformed  Artist_Popularity  is not approximately normally distributed. 0.0998682392279409 0
```

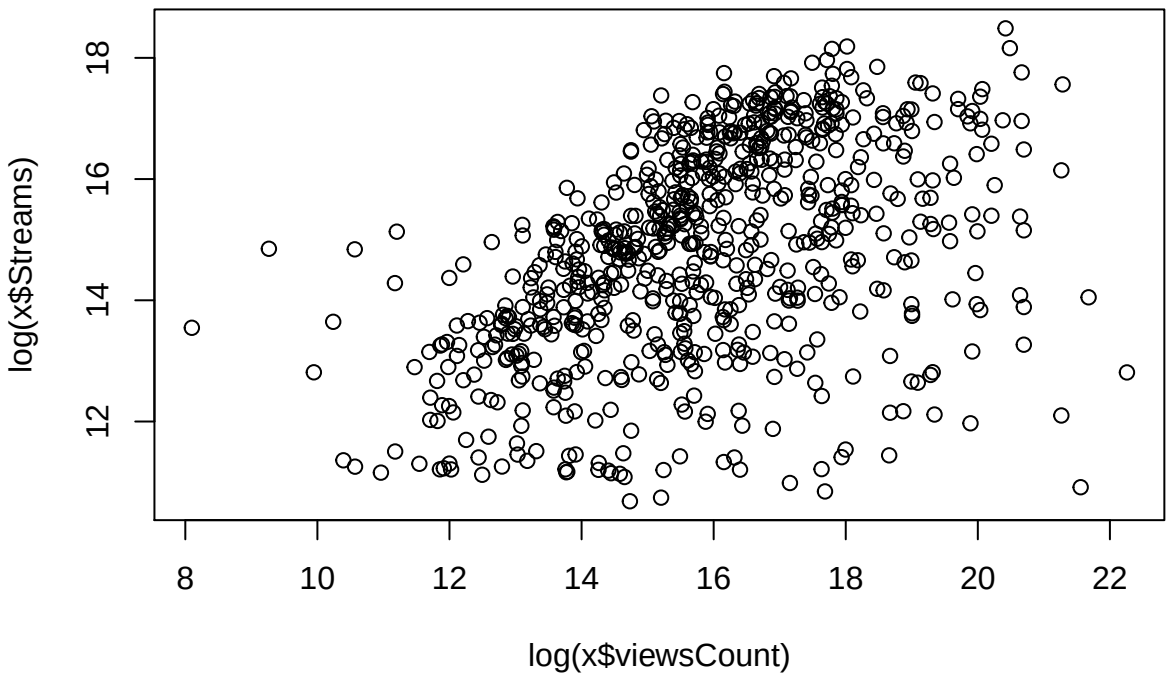
```
## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Artist_Singles_Number  is not approximately normally distributed. 0.10423226256727

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Artist_Singles_Tracks_Number  is not approximately normally distributed. 0.0916509

##  Log-transformed  Streams  is not approximately normally distributed. 0.0534913776972901 0.050613398

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Track_Duration_ms  is not approximately normally distributed. 0.059977602334861 0.

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Total_tracks  is not approximately normally distributed. 0.178763330511317 0.05061

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-ransformed  viewsCount  is approximately normally distributed! 0.0202979838407569 0.05061339867

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Artist_Google_searches_11m  is not approximately normally distributed. 0.10568825

## Warning in ks.test(sub_df, "pnorm", mean = mean(sub_df), sd = sd(sub_df)):
## ties should not be present for the Kolmogorov-Smirnov test

##  Log-transformed  Artist_Youtube_searches_11m  is not approximately normally distributed. 0.12063640
```

After log-transforming the variables, streams and viewsCount are approximately normally distributed, so we can proceed with testing whether they are also jointly (log-) normally distributed.

H_0: two variables are jointly normal distributed H_1: two variables are not jointly normal distributed

```
plot(log(x$viewsCount), log(x$Streams))

bivariate_df <- select(log_numeric_x, c('Streams', 'viewsCount'))

# install.packages("MVN")

library("MVN")
```

```
## Warning: package 'MVN' was built under R version 3.5.3

## sROC 0.1-2 loaded
```




```
mvn(bivariate_df, mvnTest = "mardia")$multivariateNormality # Not jointly normal
```

```
##              Test        Statistic              p value Result
## 1 Mardia Skewness 125.482980981752 3.59864584165554e-26     NO
## 2 Mardia Kurtosis 1.91258360408705    0.055801380314662    YES
## 3             MVN             <NA>                 <NA>     NO
```

```
mvn(bivariate_df, mvnTest = "hz")$multivariateNormality # Not jointly normal
```

```
##            Test       HZ p value MVN
## 1 Henze-Zirkler 9.570169       0  NO
```

```
mvn(bivariate_df, mvnTest = "royston")$multivariateNormality #
```

```
##      Test        H     p value MVN
## 1 Royston 37.26924 8.25193e-09  NO
```

```
mvn(bivariate_df, mvnTest = "energy")$multivariateNormality
```

```
##          Test Statistic p value MVN
## 1 E-statistic  9.542325       0  NO
```

Result: all tests reject the Null hypothesis that the two variables log-Streams and log-viewsCount are jointly normally distributed. Hence, Steiger's Z test cannot be meaningfully conducted. All results displayed in the correlogram are thefore to be treated with caution.

4) Box-Cox transformations

```
library("psych")
```

```
## Warning: package 'psych' was built under R version 3.5.2
```

```
library("car")
```

```
## Warning: package 'car' was built under R version 3.5.2
## Loading required package: carData
## Warning: package 'carData' was built under R version 3.5.2
##
## Attaching package: 'car'
## The following object is masked from 'package:psych':
##
##     logit
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
ksD <- function (p, x) {
  y <- bcPower(x, p)
  ks.test(y, "pnorm", mean=mean(y), sd=sd(y))$statistic
}

oldw <- getOption("warn")
options(warn = -1)

min_values <- c()

for (column_index in 1:length(strictly_positive_variables)){

  column_name <- strictly_positive_variables[column_index]

  x_sub <- as.numeric(x[[paste(column_name)]])

  result <- optimize(ksD, c(-5,5), x=x_sub)

  min_values[column_index] <- result$minimum

  message(paste(column_index, ', minimum value is: ', result$minimum))

}
```

```
## 1 , minimum value is:  -0.205660850905614
## 2 , minimum value is:  -1.72547245696245
## 3 , minimum value is:  -0.0460865059585334
## 4 , minimum value is:  0.174644177497162
## 5 , minimum value is:  0.037975342271715
## 6 , minimum value is:  0.212785305428911
## 7 , minimum value is:  0.791280509608183
## 8 , minimum value is:  -0.00130968618131601
## 9 , minimum value is:  0.139522250128656
```

```
## 10 , minimum value is:  -0.0808474940393077
```

```
options(warn = oldw)
```

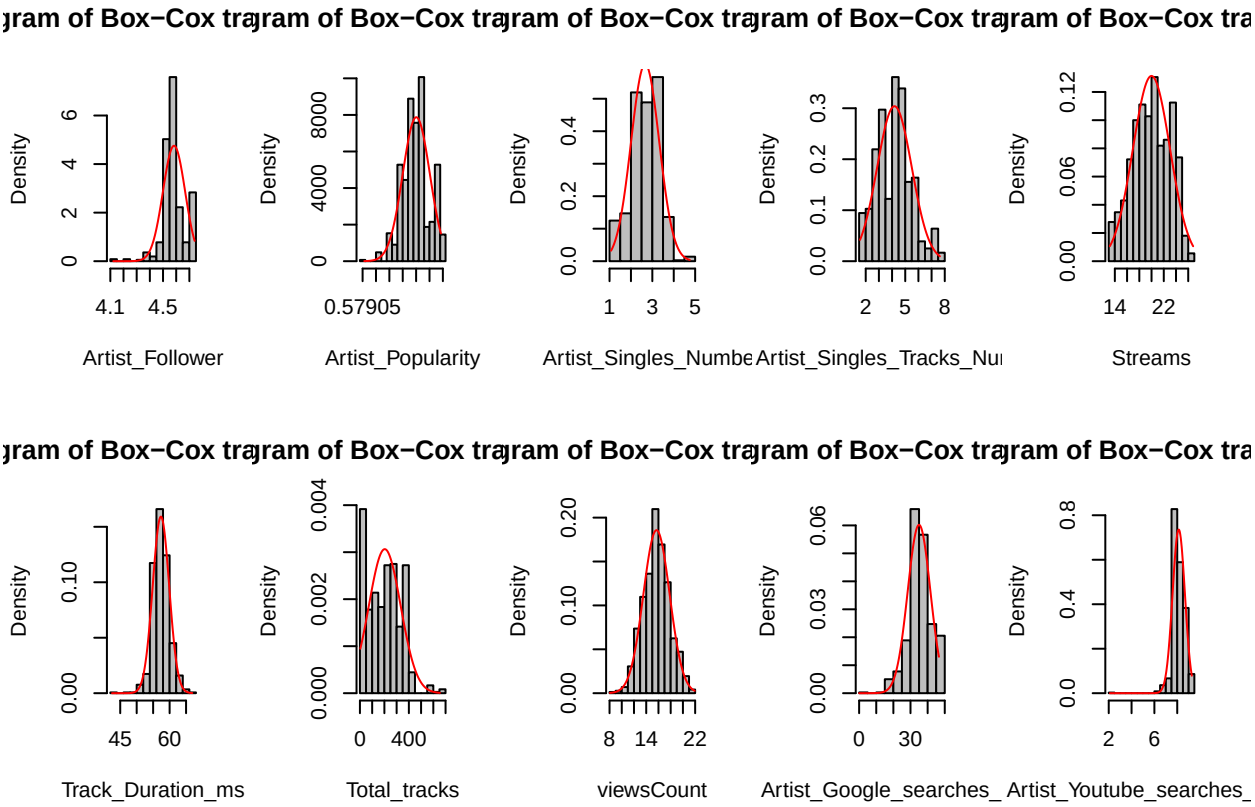
Box-Cox transformations

```
par(mfrow=c(2,5))

column_index <- 1
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Follower_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 2
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Popularity_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 3
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Singles_Number_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 4
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Singles_Tracks_Number_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 5
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Streams_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 6
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Track_Duration_ms_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 7
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Total_tracks_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 8
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
viewsCount_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 9
column_name <- strictly_positive_variables[column_index]
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Google_searches_11m_trans <- bcPower(x_sub, min_values[column_index])

column_index <- 10
column_name <- strictly_positive_variables[column_index]
```

```r
x_sub <- as.numeric(x[[paste(column_name)]])
Artist_Youtube_searches_11m_trans <- bcPower(x_sub, min_values[column_index])

hist_trans_list <- list(Artist_Follower_trans, Artist_Popularity_trans, Artist_Singles_Number_trans,
                        Artist_Singles_Tracks_Number_trans, Streams_trans, Track_Duration_ms_trans, Total_
                        viewsCount_trans, Artist_Google_searches_11m_trans, Artist_Youtube_searches_11m_tra

for (trans_index in 1:length(hist_trans_list)){

  column_name <- strictly_positive_variables[trans_index]

  selected_trans <- hist_trans_list[trans_index]
  selected_trans <- as.numeric(as.character(unlist(selected_trans[[1]])))

  hist(selected_trans, col = "gray", probability = TRUE, main = "Histogram of Box-Cox transformed", xlab
  points(seq(min(selected_trans), max(selected_trans), length.out = 500),
       dnorm(seq(min(selected_trans), max(selected_trans), length.out = 500),
             mean(selected_trans),sd(selected_trans)), type = "l", col = "red")

  test_statistic <- ks.test(selected_trans, "pnorm", mean=mean(selected_trans), sd=sd(selected_trans))$s
  critical_value <- 1.3581 / sqrt (length(selected_trans))

  if (test_statistic > critical_value) {
message(paste("Transformed ", column_name , " is not approximately normally distributed.", test_statisti
} else {
message(paste("Transformed ", column_name , " is approximately normally distributed!", test_statistic, c
}}
```

```
## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Follower  is not approximately normally distributed. 0.138923550877875 0.0506133

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Popularity  is not approximately normally distributed. 0.0710941963773345 0.0506

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Singles_Number  is not approximately normally distributed. 0.101464083335985 0.05

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Singles_Tracks_Number  is not approximately normally distributed. 0.0667677753054

## Transformed  Streams  is not approximately normally distributed. 0.052248164370101 0.050613398670707

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Track_Duration_ms  is not approximately normally distributed. 0.0593726003494802 0.0506

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Total_tracks  is not approximately normally distributed. 0.0869707482286224 0.050613398
```

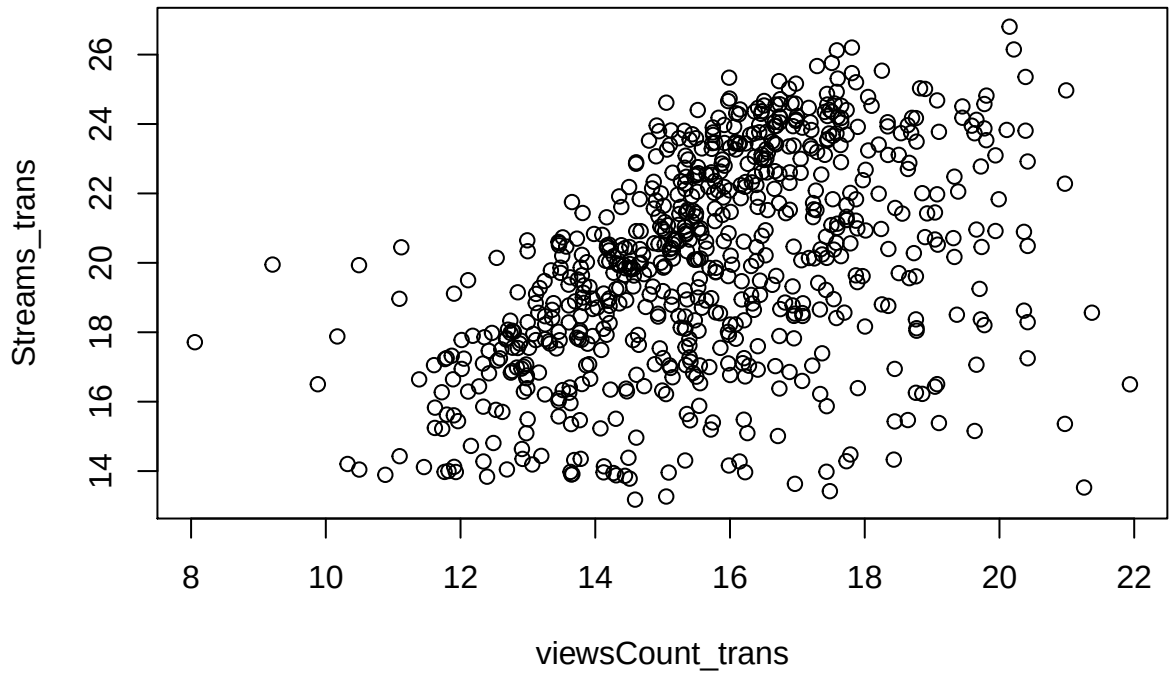
```
## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  viewsCount  is approximately normally distributed! 0.0197285296674473 0.050613398670707

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Google_searches_11m  is not approximately normally distributed. 0.079543598425828

## Warning in ks.test(selected_trans, "pnorm", mean = mean(selected_trans), :
## ties should not be present for the Kolmogorov-Smirnov test

## Transformed  Artist_Youtube_searches_11m  is not approximately normally distributed. 0.10059928290604
```




```
#test_statistic <- ks.test(Artist_Follower_trans, "pnorm", mean=mean(Artist_Follower_trans), sd=sd(Arti
#critical_value <- 1.3581 / sqrt (length(x_sub))

#if (test_statistic > critical_value) {
#message(paste("Transformed ", column_name , " is not approximately normally distributed.", test_statis
#} else {
#message(paste("Transformed ", column_name , " is approximately normally distributed!", test_statistic,
#}
```

Again, after Box-Cox transforming the variables with λ equal to the optimized, minimum value to pass the KS test only viewsCount appears to be approximately normally distributed. Similarly, Streams is close to passing the KS test and therefore I'll also test for joint normality using the optimally Box-Cox-transformed variables.

```r
plot(viewsCount_trans, Streams_trans)
```




```r
bivariate_df <- data.frame(viewsCount_trans, Streams_trans)

mvn(bivariate_df, mvnTest = "mardia")$multivariateNormality # Not jointly normal
```

```
##              Test        Statistic              p value Result
## 1 Mardia Skewness 117.602845502251 1.73611339400134e-24     NO
## 2 Mardia Kurtosis 1.40217211324614    0.160863856949058    YES
## 3             MVN             <NA>                 <NA>     NO
```

```r
mvn(bivariate_df, mvnTest = "hz")$multivariateNormality # Not jointly normal
```

```
##            Test       HZ p value MVN
## 1 Henze-Zirkler 8.825176       0  NO
```

```r
mvn(bivariate_df, mvnTest = "royston")$multivariateNormality #
```

```
##      Test        H    p value MVN
## 1 Royston 32.84475 7.5373e-08  NO
```

```r
mvn(bivariate_df, mvnTest = "energy")$multivariateNormality
```

```
##          Test Statistic p value MVN
## 1 E-statistic   8.66207       0  NO
```

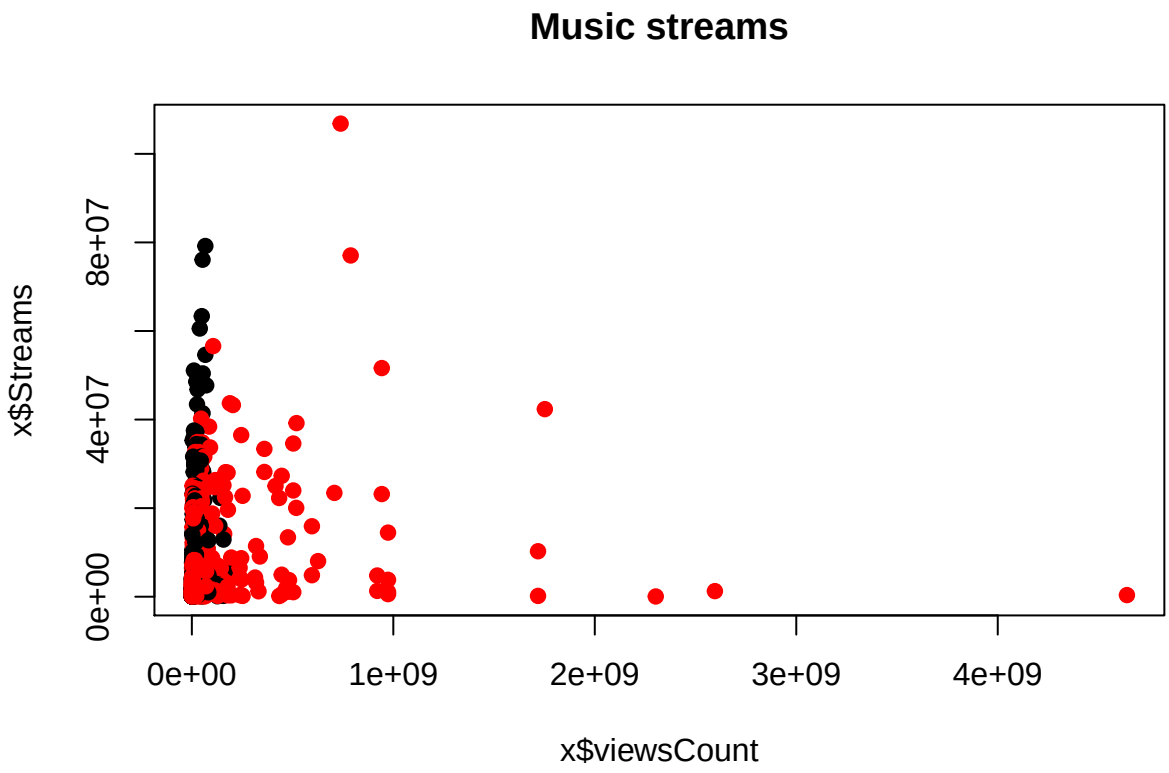
Result: all tests reject the Null hypothesis that the two variables optimal Box-Cox-Streams and optimal Box-Cox-viewsCount are jointly normally distributed.

Table by genre

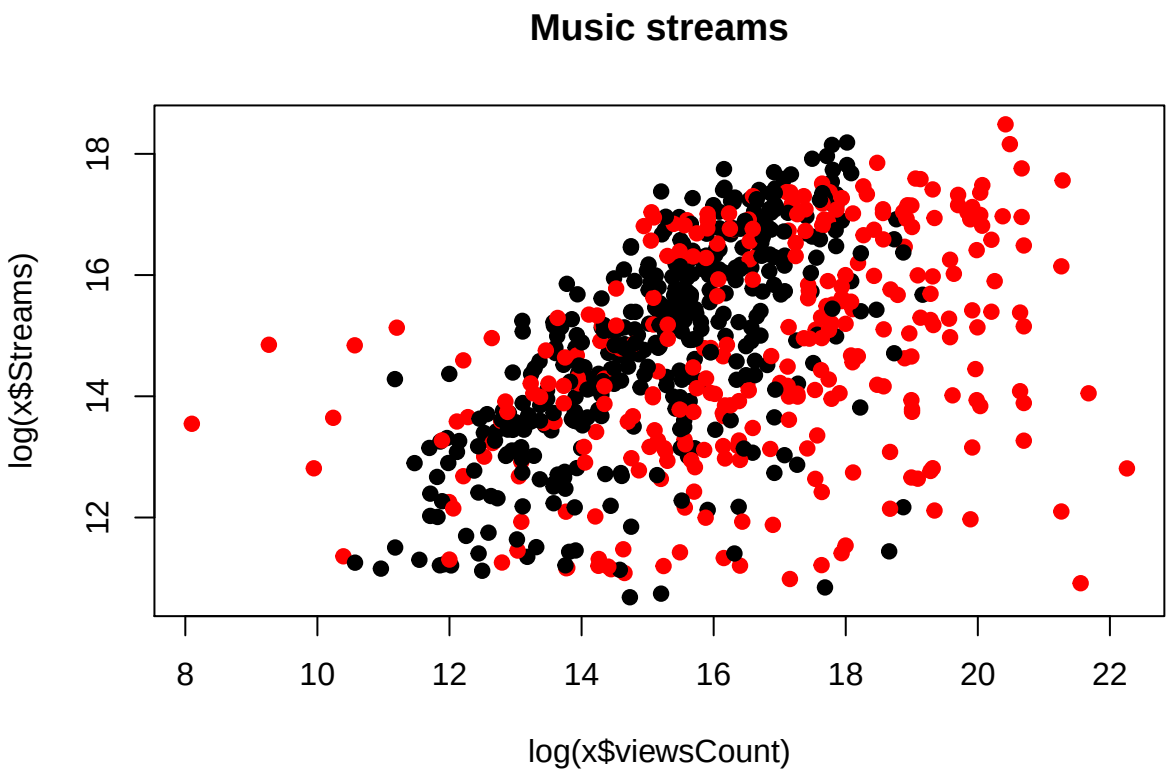
```
table(x$Genre)
```

```
##
##   dance     edm Hip Hop   house   latin   metal     pop     r&b     rap
##       4       8     411      13       2      11     140       3     126
##    rock
##       2
```

```
##1

col <- ifelse(x$Genre == "Hip Hop", "black", "red")

plot(x$viewsCount, x$Streams, main="Music streams", pch=19, col=col)
```



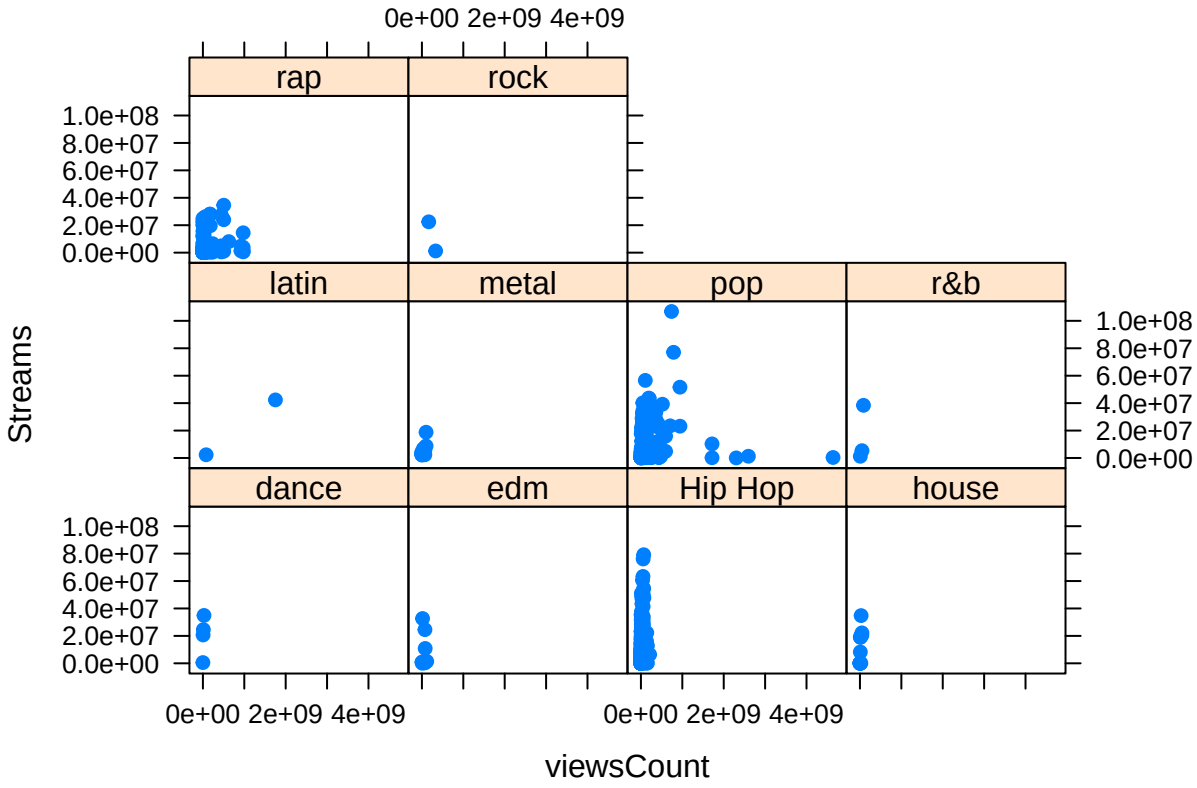

```
plot(log(x$viewsCount), log(x$Streams), main="Music streams", pch=19, col=col)
```



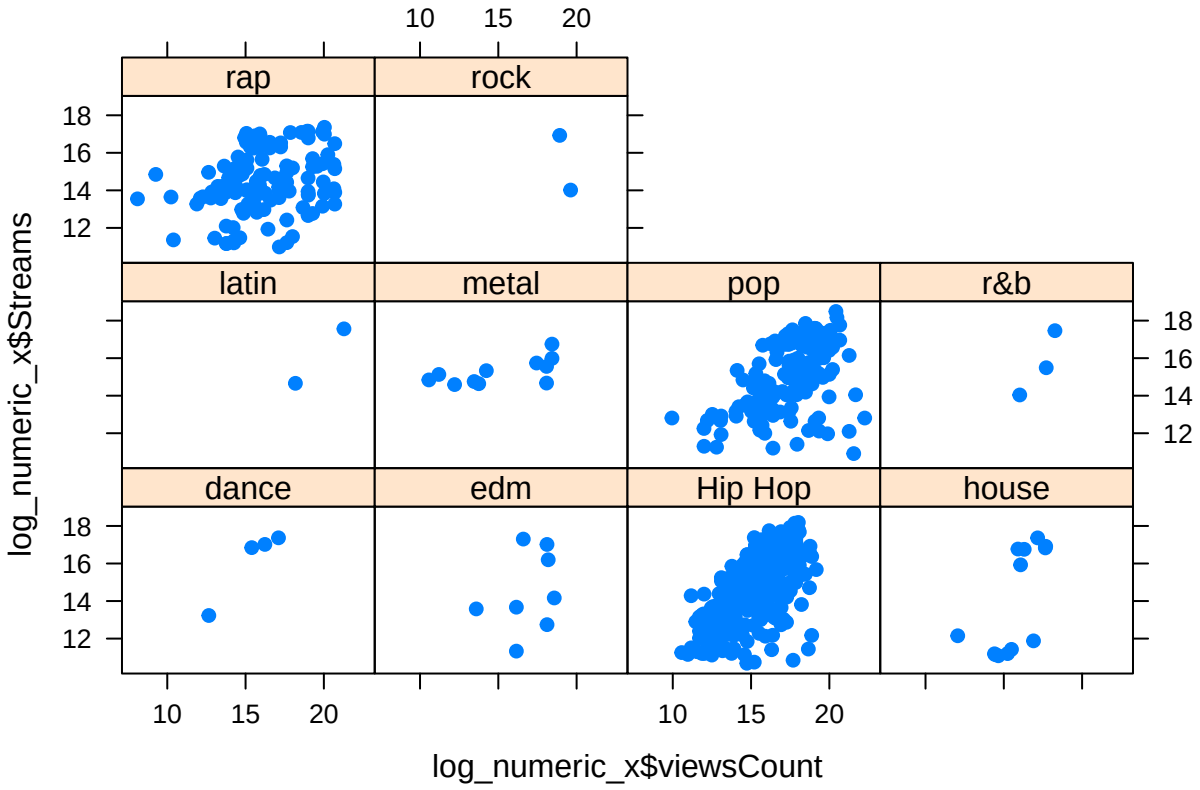

```
library("lattice")
```

```
## Warning: package 'lattice' was built under R version 3.5.1
```

```
xyplot(Streams~viewsCount|Genre, data=x, pch=19)
```




```
xyplot(log_numeric_x$Streams~log_numeric_x$viewsCount|x$Genre, pch=19)
```




```
library("ggplot2")
```

```
## Warning: package 'ggplot2' was built under R version 3.5.1
```

```
##
## Attaching package: 'ggplot2'
```

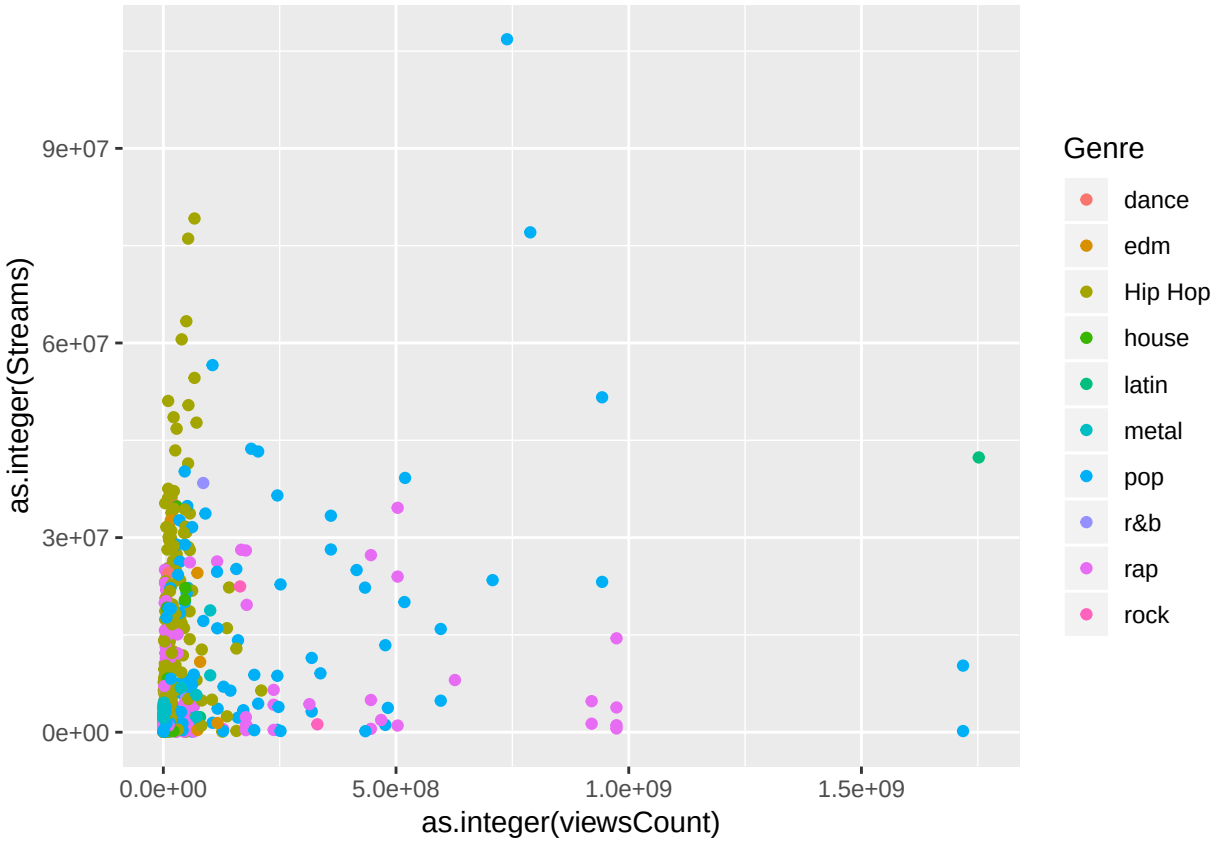
```
## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
d <-ggplot(x, aes(x=as.integer(viewsCount), y=as.integer(Streams), colour=Genre))
d + geom_point(shape=19)
```

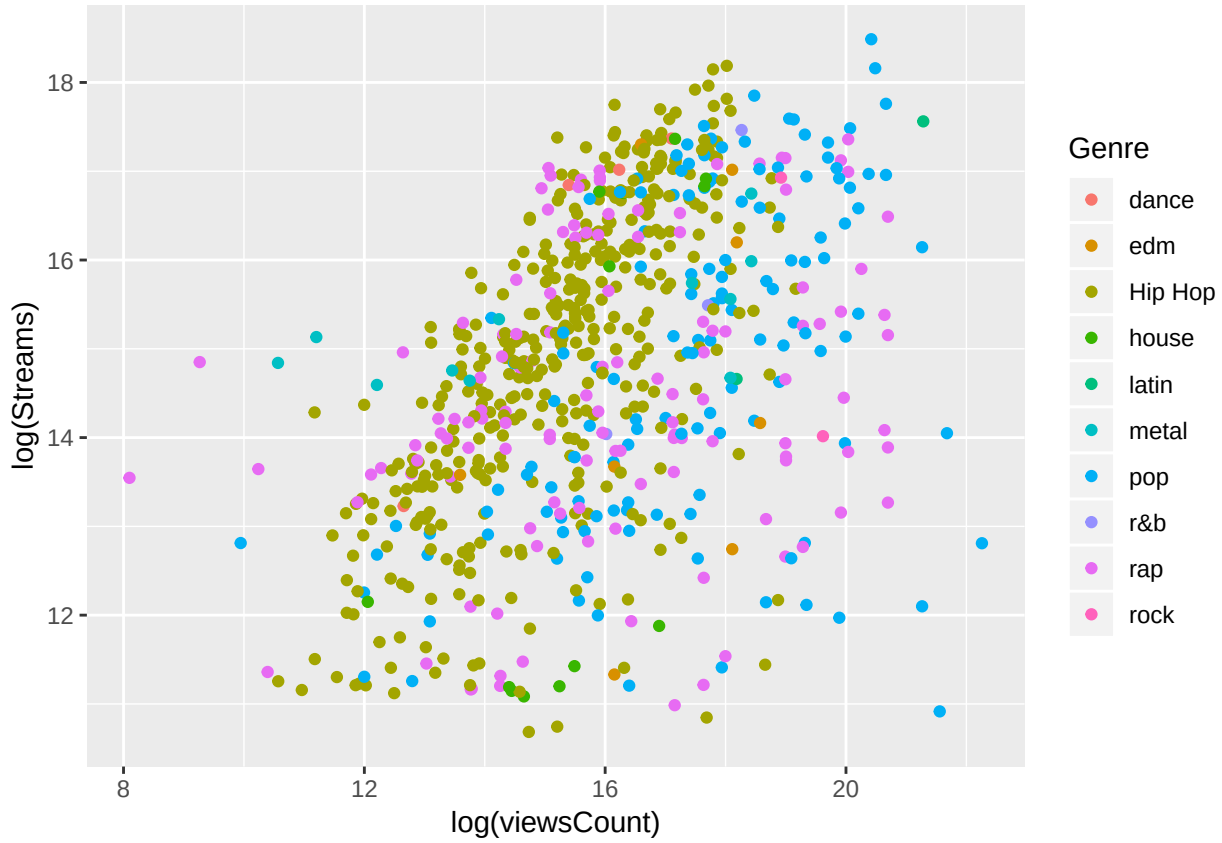
```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion to integer range
```

```
## Warning in FUN(X[[i]], ...): NAs introduced by coercion to integer range
```

```
## Warning: Removed 3 rows containing missing values (geom_point).
```

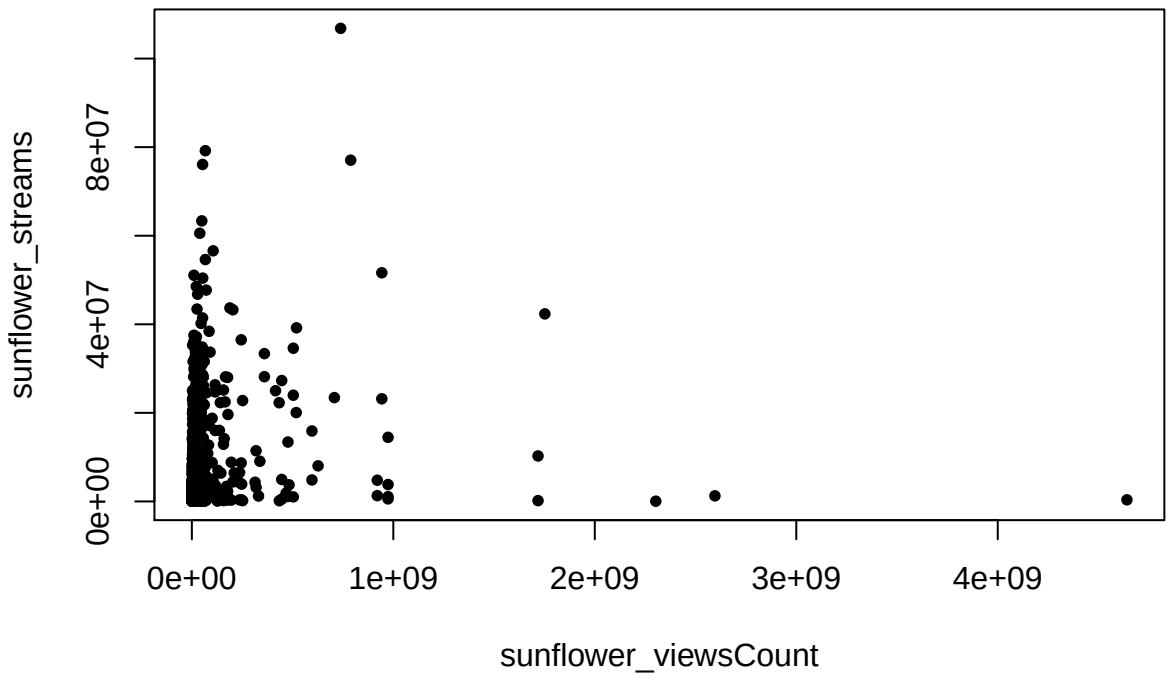



```
d <-ggplot(x, aes(x=log(viewsCount), y=log(Streams), colour=Genre))
d + geom_point(shape=19)
```



Using sunflower plot to overcome problem of overplotting.

```
sunflower_viewsCount <- 2*round(x$viewsCount/2)
sunflower_streams  <- 2*round(x$Streams/2)
sunflowerplot(sunflower_streams~sunflower_viewsCount)
```




```
library("Rmpfr")
```

```
## Warning: package 'Rmpfr' was built under R version 3.5.3
```

```
## Loading required package: gmp
```

```
## Warning: package 'gmp' was built under R version 3.5.3
```

```
##
## Attaching package: 'gmp'
```

```
## The following object is masked from 'package:rio':
##
##     factorize
```

```
## The following objects are masked from 'package:base':
##
##     %*%, apply, crossprod, matrix, tcrossprod
```

```
## C code of R package 'Rmpfr': GMP using 64 bits per limb
```

```
##
## Attaching package: 'Rmpfr'
```

```
## The following object is masked from 'package:gmp':
##
##     outer
```

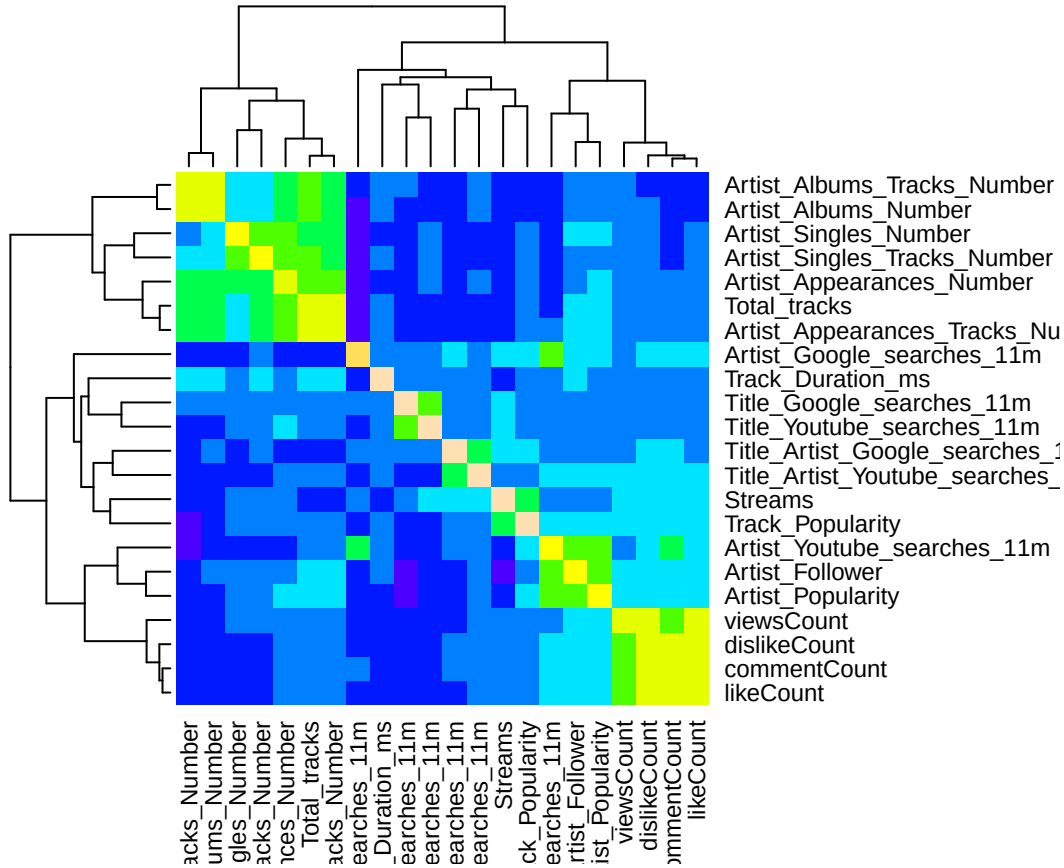
```
## The following objects are masked from 'package:stats':
##
##     dbinom, dgamma, dnorm, dpois, pnorm
```

```
## The following objects are masked from 'package:base':
##
##     cbind, pmax, pmin, rbind
```

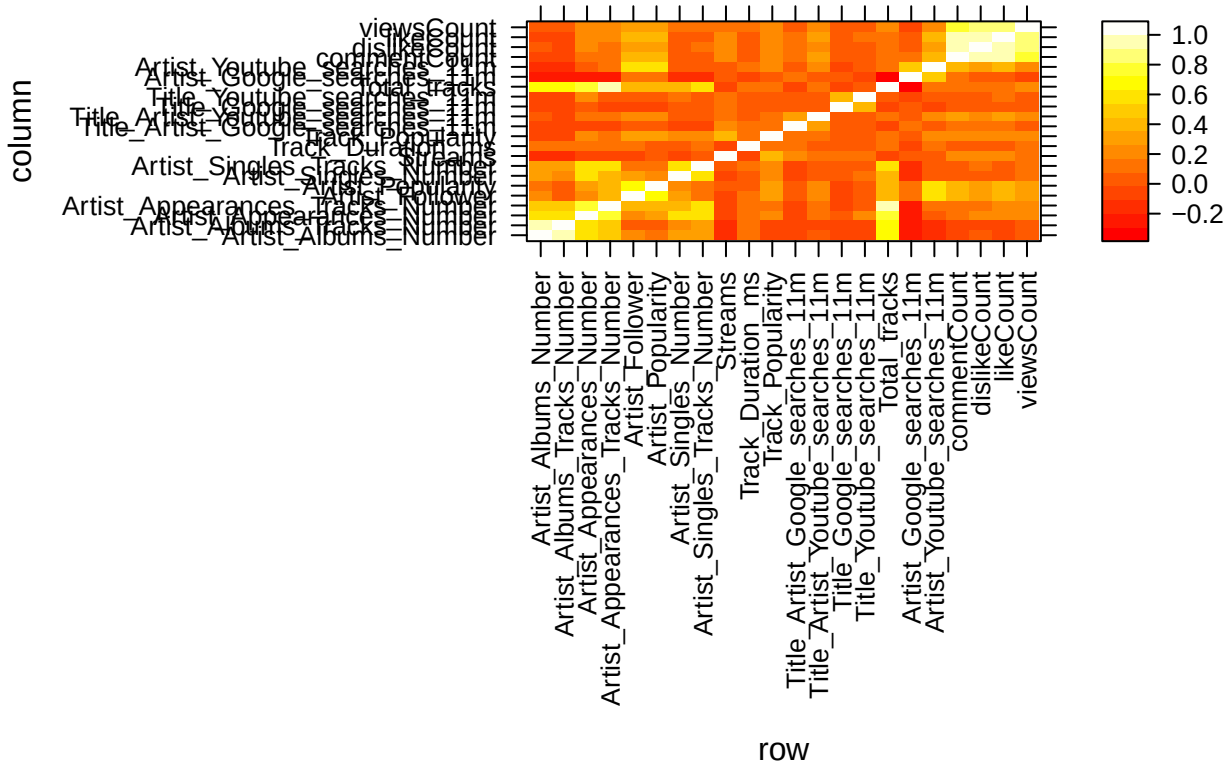
```
# (one <- mpfr(1, 120))

cor <- cor(numeric_x)
drop.cor_cols <- c('Artist_Compilations_Number', 'Artist_Compilations_Tracks_Number')
numeric_cor_x <- select(numeric_x, -one_of(drop.cor_cols))

numeric_cor_x$viewsCount <- as.numeric(numeric_cor_x$viewsCount)

str(numeric_cor_x)
```

```
## 'data.frame':    720 obs. of  22 variables:
##  $ Artist_Albums_Number             : int  0 0 1 1 1 1 1 1 1 1 ...
##  $ Artist_Albums_Tracks_Number      : int  0 0 8 8 8 8 8 8 8 8 ...
##  $ Artist_Appearances_Number        : int  9 9 2 2 2 2 2 2 2 2 ...
##  $ Artist_Appearances_Tracks_Number : int  502 502 30 30 30 30 30 30 30 30 ...
##  $ Artist_Follower                  : int  713401 713401 601346 601346 601346 601346 601346 601346 60
##  $ Artist_Popularity                : int  91 91 83 83 83 83 83 83 83 83 ...
##  $ Artist_Singles_Number            : int  3 3 15 15 15 15 15 15 15 15 ...
##  $ Artist_Singles_Tracks_Number     : int  10 10 15 15 15 15 15 15 15 15 ...
##  $ Streams                          : int  106824437 2327995 79193552 54619683 48552840 46784729 434
##  $ Track_Duration_ms                : int  209754 200755 157093 158853 176066 163146 139693 191760 1
##  $ Track_Popularity                 : int  76 72 78 77 73 75 73 75 69 69 ...
##  $ Title_Artist_Google_searches_11m : int  20904 572 8880 8880 1975 1156 3260 10880 220 568 ...
##  $ Title_Artist_Youtube_searches_11m: int  308911 7320 7660 7660 1530 990 2240 7915 154 441 ...
##  $ Title_Google_searches_11m        : int  1288732 2799 4805454 4805454 47025 33165 47709 45925 8977
##  $ Title_Youtube_searches_11m       : int  18353181 33600 3446454 3446454 32325 28872 38436 31975 593
##  $ Total_tracks                     : int  512 512 53 53 53 53 53 53 53 53 ...
##  $ Artist_Google_searches_11m       : int  299212 299212 1468281 1468281 1468281 1468281 1468281 1468
##  $ Artist_Youtube_searches_11m      : int  2451500 2451500 1076400 1076400 1076400 1076400 1076400 10
##  $ commentCount                     : int  172604 2272 22183 22183 13376 10741 8662 303 5795 4485 ..
##  $ dislikeCount                     : int  317322 3194 27802 27802 11440 12957 10493 605 5333 4287 .
##  $ likeCount                        : int  7424686 109395 748270 748270 385252 378780 299481 24361 2
##  $ viewsCount                       : num  7.39e+08 1.03e+07 6.70e+07 6.70e+07 2.22e+07 ...
```

```
clean_cor <- cor(numeric_cor_x[complete.cases(numeric_cor_x), ])
heatmap(clean_cor, revC=T, col=topo.colors(10))
```




```
library("lattice")
levelplot(clean_cor, scales=list(x=list(rot=90)), aspect = "fill", col.regions=heat.colors(100))
```



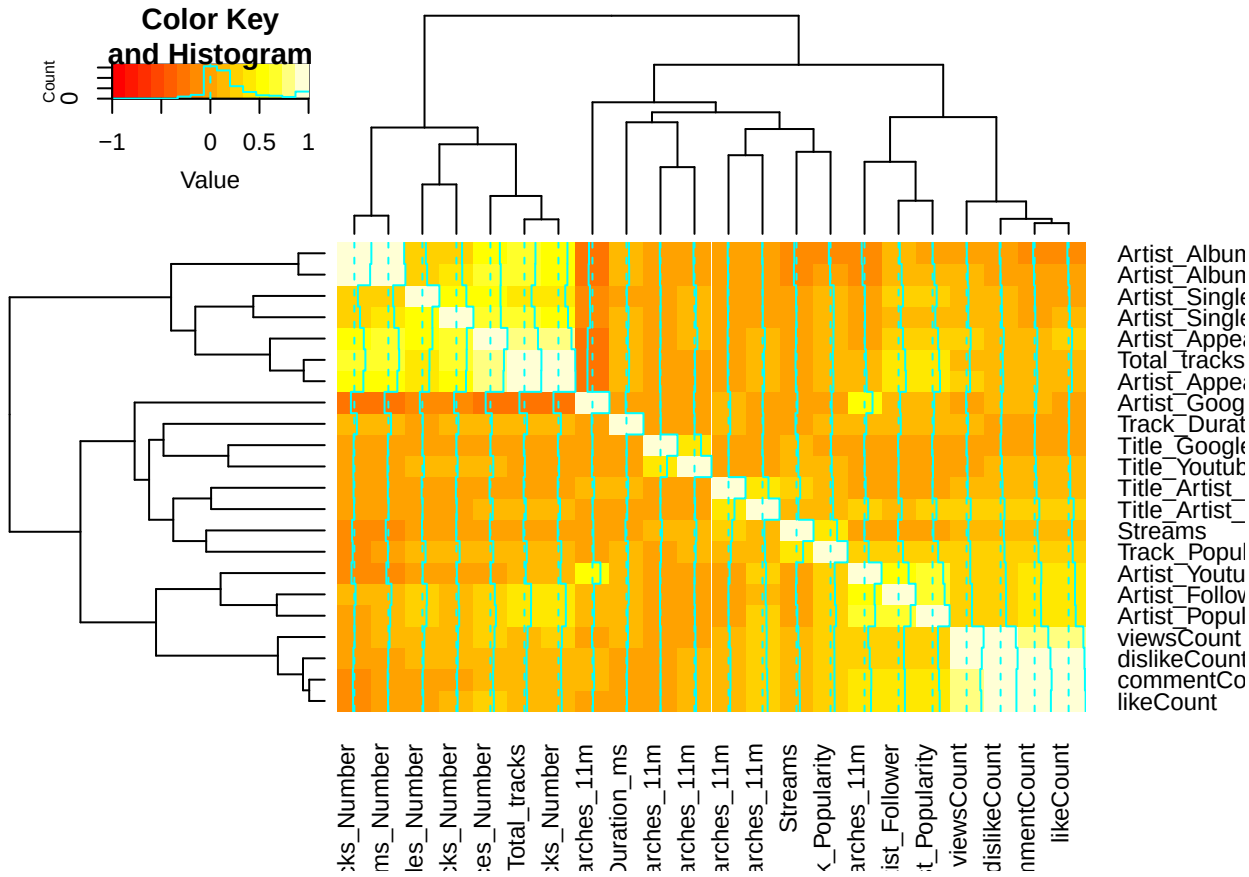

```
library("gplots")
```

```
## Warning: package 'gplots' was built under R version 3.5.2
```

```
##
## Attaching package: 'gplots'
```

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

```
gplots::heatmap.2(clean_cor, revC=T, na.rm=T)
```



Tests for significance of Bravais-Pearson, Spearman and Kendall correlation coefficients

```
x$stream_quantile_ind <- 0

stream_quantiles <- quantile(x$Streams, probs = c(0.25, 0.5, 0.75))

streams_q_25 <- stream_quantiles[1]
streams_median <- stream_quantiles[2]
streams_q_75 <- stream_quantiles[3]

x$stream_quantile_ind <- ifelse(x$Streams < streams_q_25, 1, x$stream_quantile_ind + 0)
x$stream_quantile_ind <- ifelse(((x$Streams >= streams_q_25) & (x$Streams < streams_median)), 2, x$strea
x$stream_quantile_ind <- ifelse(((x$Streams >= streams_median) & (x$Streams < streams_q_75)), 3, x$strea
x$stream_quantile_ind <- ifelse((x$Streams >= streams_q_75), 4, x$stream_quantile_ind + 0)

# bottom_25 <- subset(x , Streams < q_25)
# top_50_75 <- subset(x, Streams >= q_25 & Streams < median)
# top_25_50 <- subset(x , Streams >= median & Streams < q_75)
# top_25 <- subset(x, Streams >= q_75)

x$stream_quantile_ind <- as.factor(x$stream_quantile_ind)
x$Genre <- as.factor(x$Genre)

tab<-table(x$Genre, x$stream_quantile_ind)
tab

##
```

```
##             1   2   3   4
##   dance     1   0   0   3
##   edm       3   2   1   2
##   Hip Hop  96 100 116  99
##   house     7   0   1   5
##   latin     0   1   0   1
##   metal     0   5   5   1
##   pop      42  22  34  42
##   r&b       0   1   1   1
##   rap      31  48  22  25
##   rock      0   1   0   1
```

```
# critical Chi^2 value (df= 27): 40.11

chisq.test(tab)
```

```
## Warning in chisq.test(tab): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 52.499, df = 27, p-value = 0.002312
```

```
chisq.test(tab, simulate.p.value = TRUE)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  tab
## X-squared = 52.499, df = NA, p-value = 0.0004998
```

```
library("vcd")
```

```
## Warning: package 'vcd' was built under R version 3.5.3
```

```
## Loading required package: grid
```

```
assocstats(tab)
```

```
##                     X^2 df   P(> X^2)
## Likelihood Ratio 60.166 27 0.00025092
## Pearson          52.499 27 0.00231197
##
## Phi-Coefficient   : NA
## Contingency Coeff.: 0.261
## Cramer's V        : 0.156
```

The p-value is smaller than the confidence level $\alpha = 0.05$, hence we reject the Null hypothesis of no independence and conclude that there exists a dependence between the songs' genre and their placement within the four quantile ranges of the distribution of their amount of streams. Cramer's V (~0.16) suggests that there is a weak dependence between the ranking of a track and its genre.

```
x$viewsCount_quantile_ind <- 0

viewsCount_quantiles <- quantile(x$viewsCount, probs = c(0.25, 0.5, 0.75))

viewsCount_q_25 <- viewsCount_quantiles[1]
```

```r
viewsCount_median <- viewsCount_quantiles[2]
viewsCount_q_75 <- viewsCount_quantiles[3]

x$viewsCount_quantile_ind <- ifelse(x$viewsCount < viewsCount_q_25, 1, x$viewsCount_quantile_ind + 0)
x$viewsCount_quantile_ind <- ifelse(((x$viewsCount >= viewsCount_q_25) & (x$viewsCount < viewsCount_medi
x$viewsCount_quantile_ind <- ifelse(((x$viewsCount >= viewsCount_median) & (x$viewsCount < viewsCount_q_
x$viewsCount_quantile_ind <- ifelse((x$viewsCount >= viewsCount_q_75), 4, x$viewsCount_quantile_ind + 0)

# bottom_25 <- subset(x , Streams < q_25)
# top_50_75 <- subset(x, Streams >= q_25 & Streams < median)
# top_25_50 <- subset(x , Streams >= median & Streams < q_75)
# top_25 <- subset(x, Streams >= q_75)

x$viewsCount_quantile_ind <- as.factor(x$viewsCount_quantile_ind)

tab<-table(x$Genre, x$viewsCount_quantile_ind)
tab
```

```
##
##             1   2   3   4
##   dance     1   1   2   0
##   edm       1   0   3   4
##   Hip Hop 127 125 116  43
##   house     1   5   5   2
##   latin     0   0   0   2
##   metal     6   0   0   5
##   pop      13  18  27  82
##   r&b       0   0   1   2
##   rap      31  30  27  38
##   rock      0   0   0   2
```

```r
# critical Chi^2 value (df= 27): 40.11

chisq.test(tab)
```

```
## Warning in chisq.test(tab): Chi-squared approximation may be incorrect
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 173.9, df = 27, p-value < 2.2e-16
```

```r
chisq.test(tab, simulate.p.value = TRUE)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  tab
## X-squared = 173.9, df = NA, p-value = 0.0004998
```

```r
library("vcd")
assocstats(tab)
```

```
##                     X^2 df P(> X^2)
## Likelihood Ratio 178.58 27        0
```

```
## Pearson          173.90 27        0
##
## Phi-Coefficient   : NA
## Contingency Coeff.: 0.441
## Cramer's V        : 0.284
```

The p-value is smaller than the confidence level $\alpha = 0.05$, hence we reject the Null hypothesis of no independence and conclude that there exists a dependence between the songs' genre and their placement within the four quantile ranges of the distribution of their views on Youtube. Cramer's V (~0.3) suggests that there is a semi-weak dependence between the ranking of a music video and its genre.

Let's see whether there exists an ordinal relationship between the placement of streams within the distributional range and the placement of the corresponding music video's views:

```
ab2 <- na.omit(cbind(x$stream_quantile_ind, x$viewsCount_quantile_ind))
nrow(ab2)*(nrow(ab2)-1)/2
```

```
## [1] 258840
```

```
#
ind <- order(ab2[,1], ab2[,2])
ab2 <- ab2[ind,]
#b
C <- D <- Tx <- Ty <- Txy <- 0
for (i in 1:(nrow(ab2)-1)) {
  if (i%%100==0) cat(i, "\n")
  for(j in (i+1):nrow(ab2)) {
    if (ab2[i,1]==ab2[j,1]) {
      if (ab2[i,2]==ab2[j,2]) {
        Txy <- Txy+1
      } else {
        Tx <- Tx+(ab2[i,2]<ab2[j,2])
      }
    } else {
      if (ab2[i,2]==ab2[j,2]) Ty <- Ty+1
      if (ab2[i,2]<ab2[j,2]) C <- C+1
      if (ab2[i,2]>ab2[j,2]) D <- D+1
    }
  }
}
```

```
## 100
## 200
## 300
## 400
## 500
## 600
## 700
```

```
c(C=C, D=D, Tx=Tx, Ty=Ty, Txy=Txy)
```

```
##      C      D     Tx     Ty    Txy
## 115660  35319  43420  43421  21020
```

```
k_t <- (C - D)/(nrow(ab2)*(nrow(ab2)-1)/2)
k_t # (without ties)
```

```
## [1] 0.3103887
```

```r
library("ryouready")
```

```
## Warning: package 'ryouready' was built under R version 3.5.3
```

```r
ord.tau(table(ab2[,1], ab2[,2]))
```

```
## Kendall's (and Stuart's) Tau statistics
##  Tau-b: 0.413
##  Tau-c: 0.413
```

```r
cor(as.numeric(x$stream_quantile_ind), as.numeric(x$viewsCount_quantile_ind), method = "kendall")
```

```
## [1] 0.4132778
```

```r
cor.test(as.numeric(x$stream_quantile_ind), as.numeric(x$viewsCount_quantile_ind), method="kendall")
```

```
##
##  Kendall's rank correlation tau
##
## data:  as.numeric(x$stream_quantile_ind) and as.numeric(x$viewsCount_quantile_ind)
## z = 13.297, p-value < 2.2e-16
## alternative hypothesis: true tau is not equal to 0
## sample estimates:
##       tau
## 0.4132778
```

```r
tab<-table(x$stream_quantile_ind, x$viewsCount_quantile_ind)
chisq.test(tab)
```

```
##
##  Pearson's Chi-squared test
##
## data:  tab
## X-squared = 230.12, df = 9, p-value < 2.2e-16
```

```r
chisq.test(tab, simulate.p.value = TRUE)
```

```
##
##  Pearson's Chi-squared test with simulated p-value (based on 2000
##  replicates)
##
## data:  tab
## X-squared = 230.12, df = NA, p-value = 0.0004998
```

```r
library("vcd")
assocstats(tab)
```

```
##                     X^2 df P(> X^2)
## Likelihood Ratio 261.31  9        0
## Pearson          230.12  9        0
##
## Phi-Coefficient   : NA
## Contingency Coeff.: 0.492
## Cramer's V        : 0.326
```

The rank correlation coefficient by Kendall's Tau (here for a quadratic table) is around 0.41 and the test yields that this coefficient is significant, hence we can conclude that there exists a positive relationship between the placement of streams of a song on Spotify and the placement of views of the corresponding music video on

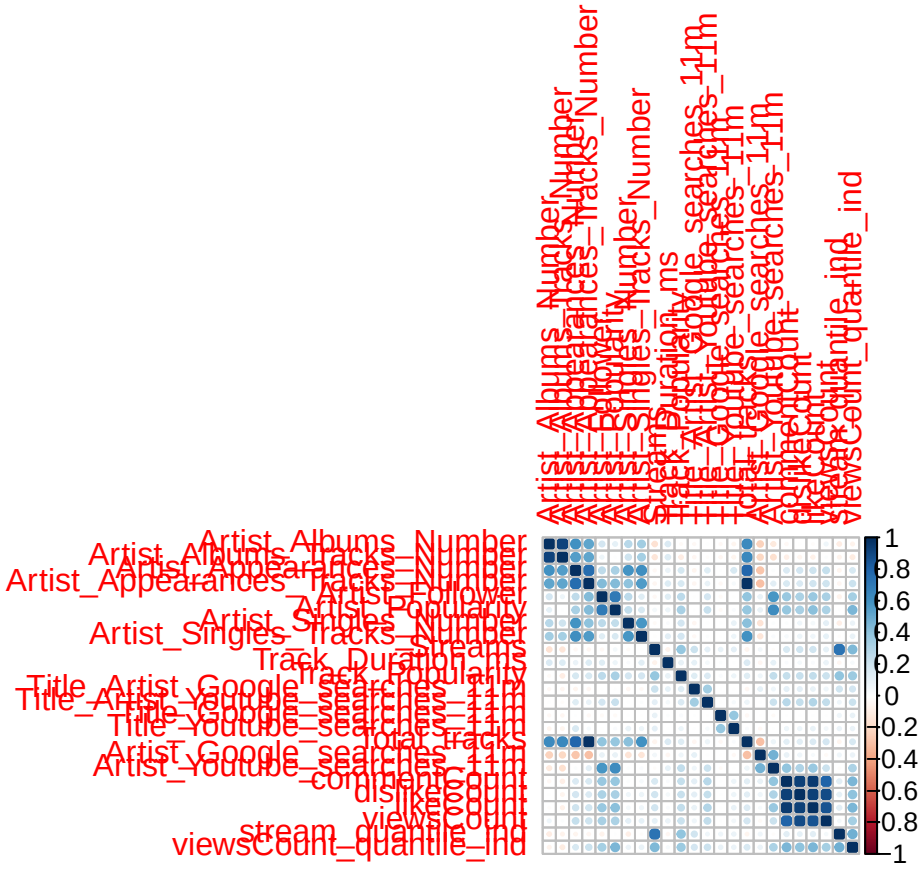
Youtube, meaning a higher rank of the song's music video in Youtube views is associated with a higher rank inside of Spotify's streams.

Even if not appropriate since the two variables have an ordinal scale and using χ^2 test for independence would neglect additional information, Cramer's V (~0.33) states there exists are semi-weak relationship but without inferring anything about the direction of the relationship, only the strength of this relationship. Therefore, Kendall's Tau gives us the "correct" estimate, indicating the positive relationship between the two ordinal variables.

```
numeric_cor_x$stream_quantile_ind <- as.numeric(x$stream_quantile_ind)
numeric_cor_x$viewsCount_quantile_ind <- as.numeric(x$viewsCount_quantile_ind)

clean_cor <- cor(numeric_cor_x[complete.cases(numeric_cor_x), ])

library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
corrplot(clean_cor, method="circle")
```




```
cor.mtest <- function(mat, ...) {
  mat <- as.matrix(mat)
  n <- ncol(mat)
  p.mat<- matrix(NA, n, n)
  diag(p.mat) <- 0
  for (i in 1:(n - 1)) {
    for (j in (i + 1):n) {
      tmp <- cor.test(mat[, i], mat[, j], ..., method = "kendall")
```

```
      p.mat[i, j] <- p.mat[j, i] <- tmp$p.value
    }
  }
  colnames(p.mat) <- rownames(p.mat) <- colnames(mat)
  p.mat
}

# matrix of the p-value of the correlation
p.mat <- cor.mtest(clean_cor)

col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD", "#4477AA"))
significance_level <- 0.05

corrplot(clean_cor, method="color", col=col(200),
         type="upper", order="hclust",
         addCoef.col = "black", # Add coefficient of correlation
         tl.col="black", tl.srt=90, #Text label color and rotation
         # Combine with significance
         p.mat = p.mat, sig.level = significance_level, insig = "blank",
         # hide correlation coefficient on the principal diagonal
         diag=FALSE)
```

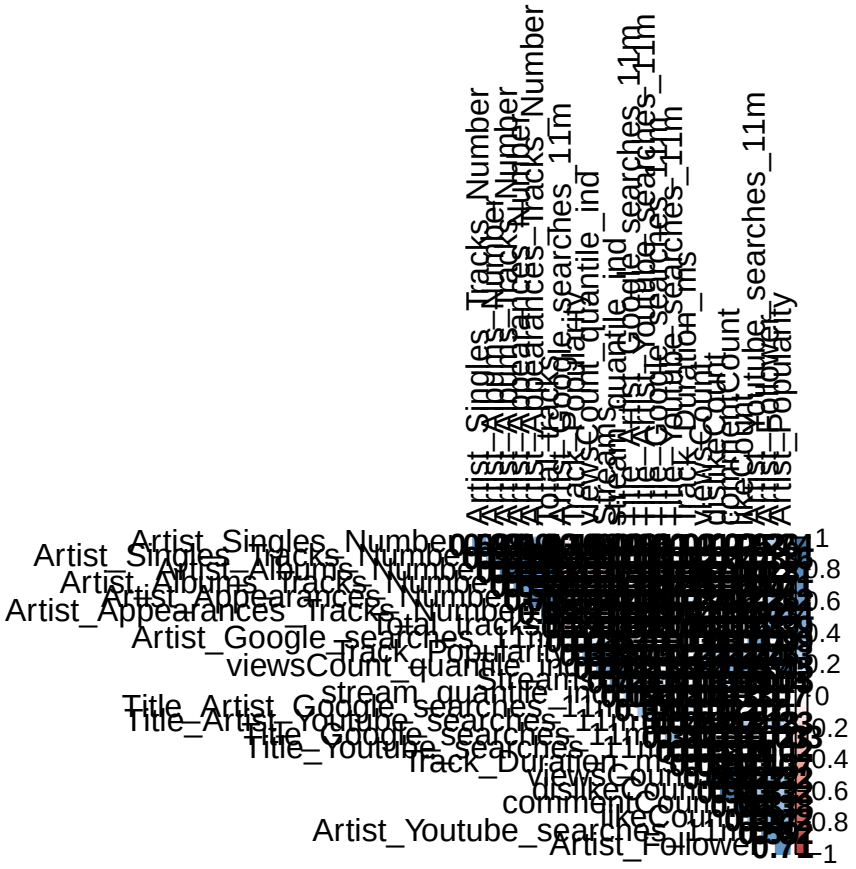


Illustration of assumption that two variables are jointly normally distributed to perform Steiger's Z test:

```
model <- lm(Streams ~ ., data = numeric_x)
```

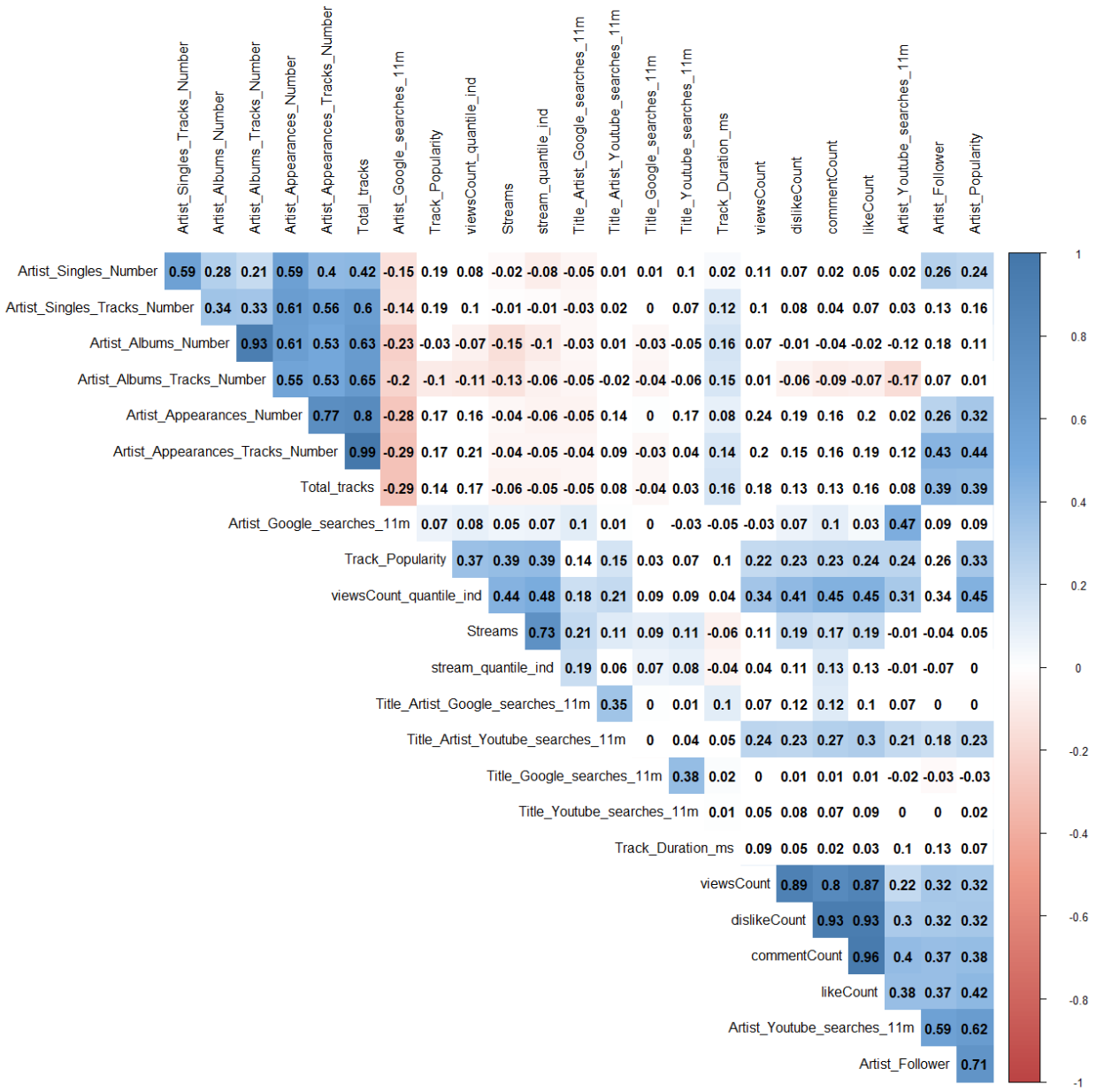


Figure 1: Correlogram with significant Spearman correlation coefficients at $\alpha = 0.05$

```r
print(model)
```

```
##
## Call:
## lm(formula = Streams ~ ., data = numeric_x)
##
## Coefficients:
##                         (Intercept)                Artist_Albums_Number
##                           -1.194e+07                           -9.414e+05
##       Artist_Albums_Tracks_Number           Artist_Appearances_Number
##                            3.647e+04                           -1.943e+04
##  Artist_Appearances_Tracks_Number          Artist_Compilations_Number
##                            9.882e+02                            2.831e+06
## Artist_Compilations_Tracks_Number                       Artist_Follower
##                           -2.434e+04                           -3.502e-02
##                  Artist_Popularity               Artist_Singles_Number
##                            9.483e+04                            1.941e+03
##      Artist_Singles_Tracks_Number                    Track_Duration_ms
##                           -3.492e+04                           -1.987e+01
##                   Track_Popularity   Title_Artist_Google_searches_11m
##                            3.082e+05                            5.520e+01
## Title_Artist_Youtube_searches_11m          Title_Google_searches_11m
##                            4.016e-02                            6.632e-01
##        Title_Youtube_searches_11m                         Total_tracks
##                            1.006e-01                                NA
##        Artist_Google_searches_11m        Artist_Youtube_searches_11m
##                            1.573e+00                           -5.737e-01
##                       commentCount                         dislikeCount
##                           -9.329e+01                            6.654e+01
##                         likeCount                           viewsCount
##                            4.410e+00                           -2.290e-02
```

```r
summary(model)
```

```
##
## Call:
## lm(formula = Streams ~ ., data = numeric_x)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -39295247  -5755473  -2838198   2932973  67155124
##
## Coefficients: (1 not defined because of singularities)
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -1.194e+07  6.750e+06  -1.769  0.07732
## Artist_Albums_Number             -9.414e+05  3.069e+05  -3.067  0.00224
## Artist_Albums_Tracks_Number       3.647e+04  1.518e+04   2.402  0.01658
## Artist_Appearances_Number        -1.943e+04  1.671e+04  -1.162  0.24545
## Artist_Appearances_Tracks_Number  9.882e+02  1.979e+03   0.499  0.61775
## Artist_Compilations_Number        2.831e+06  4.097e+06   0.691  0.48979
## Artist_Compilations_Tracks_Number -2.434e+04  1.558e+05  -0.156  0.87594
## Artist_Follower                  -3.502e-02  5.378e-02  -0.651  0.51516
## Artist_Popularity                 9.483e+04  8.126e+04   1.167  0.24361
## Artist_Singles_Number             1.941e+03  2.818e+04   0.069  0.94510
```

```
## Artist_Singles_Tracks_Number      -3.492e+04  2.813e+04  -1.241  0.21487
## Track_Duration_ms                 -1.987e+01  1.207e+01  -1.647  0.10001
## Track_Popularity                   3.082e+05  2.701e+04  11.407  < 2e-16
## Title_Artist_Google_searches_11m   5.520e+01  1.328e+01   4.155 3.65e-05
## Title_Artist_Youtube_searches_11m  4.016e-02  1.444e+00   0.028  0.97782
## Title_Google_searches_11m          6.632e-01  3.826e-01   1.733  0.08351
## Title_Youtube_searches_11m         1.006e-01  8.153e-02   1.233  0.21785
## Total_tracks                              NA         NA      NA       NA
## Artist_Google_searches_11m         1.573e+00  1.012e+00   1.554  0.12064
## Artist_Youtube_searches_11m       -5.737e-01  1.352e-01  -4.244 2.50e-05
## commentCount                      -9.329e+01  1.968e+01  -4.741 2.58e-06
## dislikeCount                       6.654e+01  1.589e+01   4.189 3.16e-05
## likeCount                          4.410e+00  8.309e-01   5.308 1.49e-07
## viewsCount                        -2.290e-02  3.854e-03  -5.942 4.45e-09
##
## (Intercept)                       .
## Artist_Albums_Number              **
## Artist_Albums_Tracks_Number       *
## Artist_Appearances_Number
## Artist_Appearances_Tracks_Number
## Artist_Compilations_Number
## Artist_Compilations_Tracks_Number
## Artist_Follower
## Artist_Popularity
## Artist_Singles_Number
## Artist_Singles_Tracks_Number
## Track_Duration_ms
## Track_Popularity                  ***
## Title_Artist_Google_searches_11m  ***
## Title_Artist_Youtube_searches_11m
## Title_Google_searches_11m         .
## Title_Youtube_searches_11m
## Total_tracks
## Artist_Google_searches_11m
## Artist_Youtube_searches_11m       ***
## commentCount                      ***
## dislikeCount                      ***
## likeCount                         ***
## viewsCount                        ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10650000 on 692 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.3081, Adjusted R-squared:  0.2861
## F-statistic: 14.01 on 22 and 692 DF,  p-value: < 2.2e-16
```

```
for (coef_index in 1:length(model$coefficients)){

  message(paste(names(model$coefficients)[coef_index] ,': ' ,model$coefficients[coef_index]))


}
```

```
## (Intercept) :  -11940817.8938362
```

```
## Artist_Albums_Number :  -941350.434783719
## Artist_Albums_Tracks_Number :  36467.9501334639
## Artist_Appearances_Number :  -19429.2203446856
## Artist_Appearances_Tracks_Number :  988.165369617914
## Artist_Compilations_Number :  2830847.69977351
## Artist_Compilations_Tracks_Number :  -24339.2235306729
## Artist_Follower :  -0.0350173650352788
## Artist_Popularity :  94829.2443999745
## Artist_Singles_Number :  1940.83778316338
## Artist_Singles_Tracks_Number :  -34923.5654574994
## Track_Duration_ms :  -19.871692246822
## Track_Popularity :  308150.983877512
## Title_Artist_Google_searches_11m :  55.2026210273784
## Title_Artist_Youtube_searches_11m :  0.0401573112799306
## Title_Google_searches_11m :  0.663179014773174
## Title_Youtube_searches_11m :  0.100564357772875
## Total_tracks :  NA
## Artist_Google_searches_11m :  1.57282778397425
## Artist_Youtube_searches_11m :  -0.573699529909453
## commentCount :  -93.2857864404099
## dislikeCount :  66.5424785464309
## likeCount :  4.41014549531601
## viewsCount :  -0.0228998146690282
```