KATHOLIEKE UNIVERSITEIT LEUVEN

FACULTY OF BIOSCIENCE ENGINEERING

REPORT: MULTIVARIATE ANALYSIS OF DATASET IN R

YOUTUBE: THE STATISTICS OF SUCCESS

ADITYA BADOLA

R0768685, MSC BIOINFORMATICS

Prof E Schrevens

IOP16a: Applied Multivariate Statistical Analysis

2020-2021, KU Leuven

Introduction

Problem description:

YouTube is a great platform to experience growth in terms of quantitative variables like total views or subscriber count. But what features of the uploaded videos from an individual channel really contribute to their success? As a regular YouTuber myself, I dove into the data of my personal YouTube channel and determined the most interesting set of meta-data/attributes of my videos.

Data Description:

There are 4 files being used by my R code at different stages/analysis objectives:

1. **Total_views.csv**: The total number of views attained between 13/8/2020 and 19/1/2021 [2x161 matrix]
2. **Device_view.csv**: The number of views contributed everyday by the following devices: Computer, Mobile phone, Tablet, TV, Game console [6x161 matrix]
3. **Traffic_view.csv**: The number of views contributed everyday by the following youtube features: Browse_features, Channel_pages, Direct_or_unknown, End_screens, External, Notifications, Other_YouTube_features, Playlist_page, Playlists, Suggested_videos, YouTube_search [12x161 matrix]
4. **Videos_added.csv**: The number of videos uploaded everyday [2x161 matrix]

There are 2 more additional files called device_sum.csv and traffic_sum.csv that describe cumulative number of hours and views contributed by all devices and traffic sources separately; but are not important for the analysis. The head for each file is shown in Annex below.

Methods

Objective:

It should be noted that the primary content of my channel is music, which most likely allows listeners to replay the video multiple times [1] and access on many different devices (like gaming consoles and tablets) which sets it apart from other video+audio content on YouTube. So a good combination of my videos in various playlists/ recommendations can contribute to the success of my channel in the long run. In this project, my objective is to analyze what kind of features and their combinations have helped my channel grow in the past 6 months. At the end, I will also create a tree based model to predict whether a video was uploaded on a day if given a set of meta data observed on that day.

Analysis:

First, I center my data and combine the device and traffic data into a mega matrix [17x161 dimension]. To visually inspect the distribution, I create plots for the device, traffic & view trend [2]. Then, I begin with exploratory analysis with a correlation plot. Since the attributes have different scales, I perform PCA on the correlation matrix. A quick screeplot analysis follows with a biplot for the first 2 components that describe 83% variance in the data. Here, I mark the scaled vector loading for the total views variable to verify that most variables have high correlation with the views vector and only one appears to be orthogonal to it. This is followed by a factor analysis to check for specific variance of each attribute, which I plot against the psi value/uniqueness. I color the most significant (least psi) as black and rest as red. The pattern of farthest variables here follows suit as that of biplot earlier. I also attempted to regress my variables (Annex, 5) using PCR and PLSR [4] but they seem to offer no new significant information about my data distribution. Hence, I move on to hierarchical cluster analysis over the dissimilarities in my data with respect to centroid and an optimum cut at k=3. This separation of observations can be quite insightful of the decision regarding

whether a video was uploaded to the channel corresponding to those variable settings. So, I fetched and combined a new ordinal variable that describes whether a video was uploaded that day for the given feature combinations. Note, this model assumes that the outcome of an uploaded video will affect all variables on that day instantly but in reality a video may take a few days to garner attention. However, considering that my frequency of uploads is also high, I expect my tree based model to not perform bad. I also performed a cost complexity analysis to prune the tree and tested the model on new data to measure the accuracy.

Results

Interpretation:

This generally optimistic trend of views (Fig 1) is a good sign for this project because I can try to ascertain the structure of significance of features/meta-data in my videos that contribute to their success. The more views the channel accumulates, the more possibility that the viewer came through a different permutation of devices and traffic sources. Looking at individual statistics, I affirm this notion (below, Fig 2).

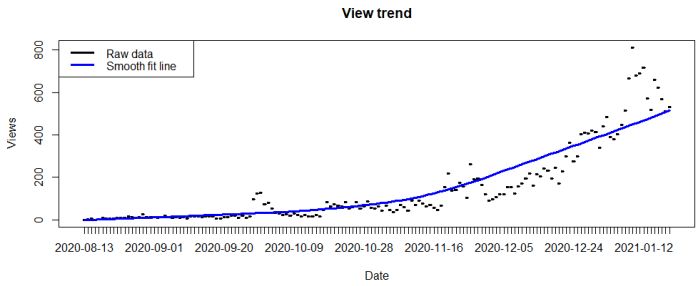


Figure 1 Views over past 6 months

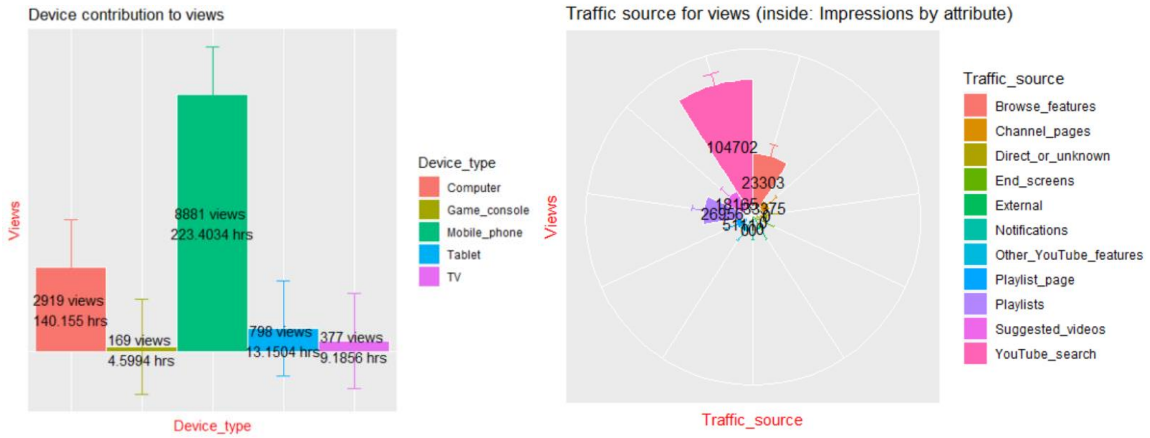


Figure 2 Statistics of considered meta data

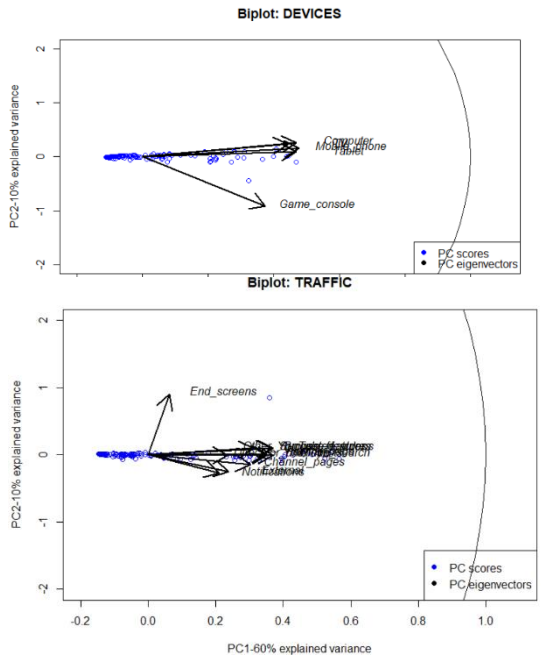


Figure 3 Biplot of Devices & Traffic sources

Device data show the dominance of mobile phone viewers with the highest viewership while game consoles are the lowest. Despite being a fraction of the biggest contributor, the intermediary variables are very significant due to the number of hours spent by viewers on my channel using them.

I performed an internal PCA and biplot analysis to check for orthogonal variables within device data and noticed Game console as a near-orthogonal effector (Fig 3). The variance is explained up to 93% by the first two components (Annex, 3).

This distribution looks oddly similar to traffic source data where the variable Youtube_search is dominant over the other variables. It displays the value of impressions (appearance of my videos on YouTube) it makes across all devices. Notably, some variables do not make any impressions at all, but still share a small proportion of viewers (this is the case where I privated some videos so their meta

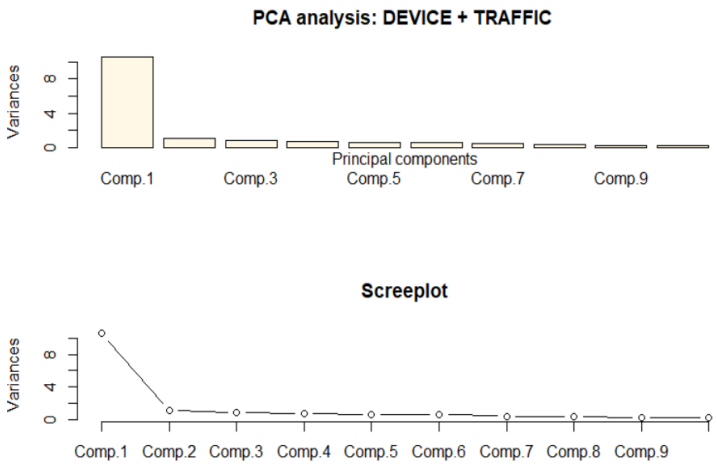


Figure 4 PCA for combined data

Looking at the correlation plot (Fig 6) between the combined datasets shows strong positive correlation between certain variables like Playlists & TV or Computer & YouTube search. There are many negative correlations that are marked significant by * as well such as End screen & Notifications, which are likely to have some effect on the Views.

data is actually the number of times I accessed them myself, hence no other impressions made by them on YouTube). These variables benefit from a sort of hierarchy that can be detected by different components in PCA or as specific variance in factor analysis.

An internal PCA for the traffic data (Annex, 3) revealed that 4 components are likely for proper assessment and biplot for the first two PCs (~70% variance) may be insufficient (Fig 3). However, I did see a near-orthogonal effector variable (End screen).

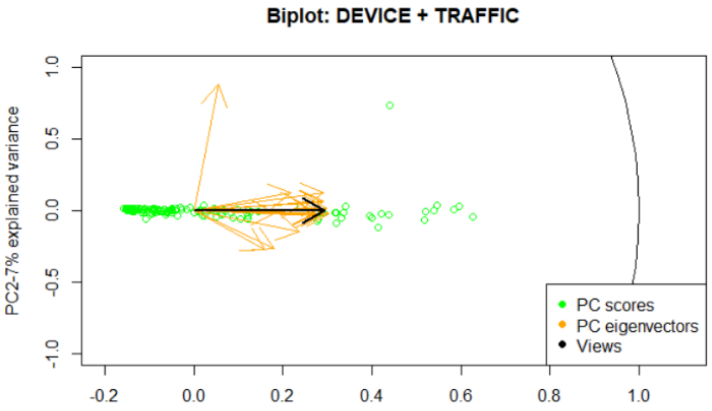


Figure 5 Biplot for combined data

So I performed an initial PCA to reduce the 17x161 matrix into 2x161 matrix with the help of 2 PCs that explain over 73% of the variance (Fig 4, Annex, 4). The biplot analysis of this data (Fig 5) showed End screen as a near-orthogonal variable while the rest of the variables tend to be more or less well correlated. The black eigenvector for Views is marked separately to check for strong/weak correlations against other effectors.

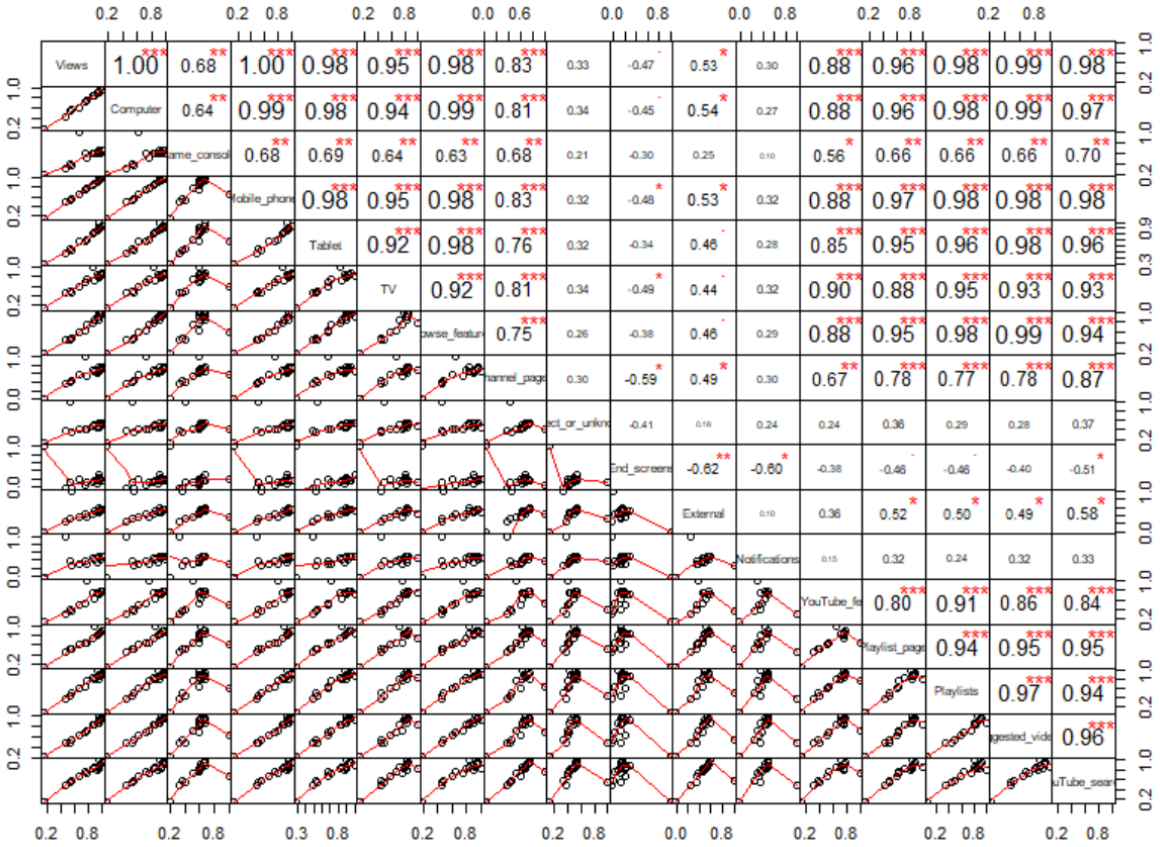


Figure 6 Correlation plot for all 17 variables

Since the biplot is quite messy, only the vectors are shown without labels but I can still find the most significant ones by checking the measure of communality (actually 1-communality) using psi, the specific variance in factor analysis (Fig 7).

By doing so, I observed 4 factors (Annex, 5) that could describe the dataset without Views variable (to detect hierarchy in variance of effectors, not the response variable Views). In theory, any variable that displays less than 5-10% specific variance has high communality (and hence more significant), so I marked the variables psi variance red if they were greater than 10% specific variance. Now, the reason for high variance may be attributed to low significance on the Views or even lack of more observational data but for the sake of analysis, I only look at the pattern of low/high variance among the attributes and compare it to what I saw in PCA. There are highly variant variables like External, Game console and Notifications that were also expressed lowly in the PCs earlier with the exception of End screen that was entirely explained by PC2 (orthogonal behavior) and also appears high in the psi-variance graph. Notably, the most important features seem to be: computer, mobile phone, browse features, playlists, YouTube search.

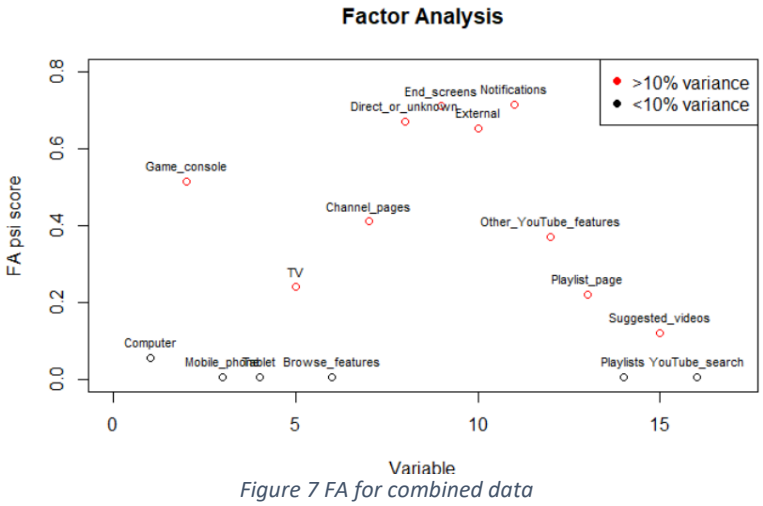


Figure 7 FA for combined data

A profile plot (Fig 8) for all my observations does not show a clear demarcation of different subgroups, however I saw some grouping structure in the biplot which I would like to analyze. In order to understand the clusters attained by the transformed components, I moved on to hierarchical cluster analysis (Fig 9). Here, I tried different k's until k=3 distributed the datapoints most appropriately (as can be seen in the dendrogram [3] and cluster plot: 130, 21, 9 in clusters). The centroid method performed the best for my data as compared to Ward D (Annex, 7)/single/complete linkage settings. The leftmost cluster contains all the observations with small correlation with Views while the rightmost cluster describes the most highly correlated ones.

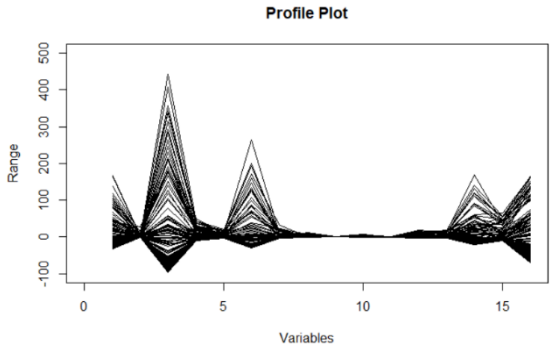


Figure 8 Profile plot for all 17 variables

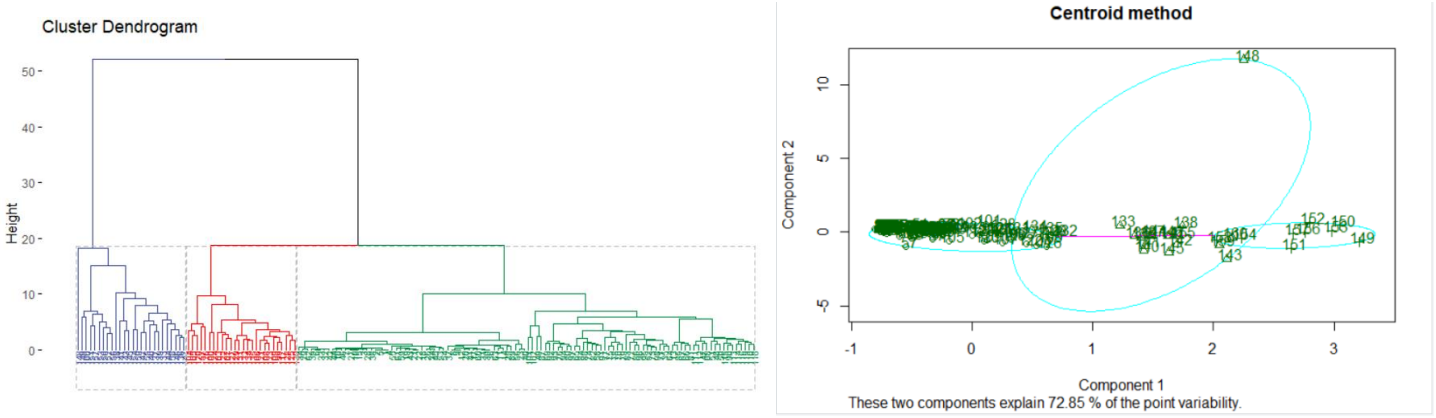


Figure 9 Cluster dendrogram (L) & cluster plot (R)

The big oval in the center describes some intermediary correlated variables and also one particular observation that appears to be driven by the End screen variable. This cluster and the rightmost cluster are likely to capture any further observations from my channel since they are all indexed >130 (the latest points in time, particularly last one month).

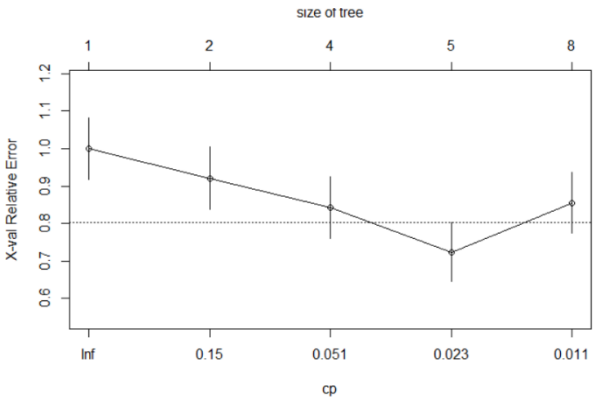


Figure 10 cp vs cross validation error for tree

Finally, looking at the many attributes of my tree, I thought there should be a way to reverse engineer the concept of having uploaded a video, given the different combinations of meta data. So I combined my matrix with a new feature vector that tells if a video was uploaded in the considered time period. This ordinal data was factored in my tree based model which I pruned at 8 splits after looking at the cost complexity curve (Fig 10, also Annex, 8).

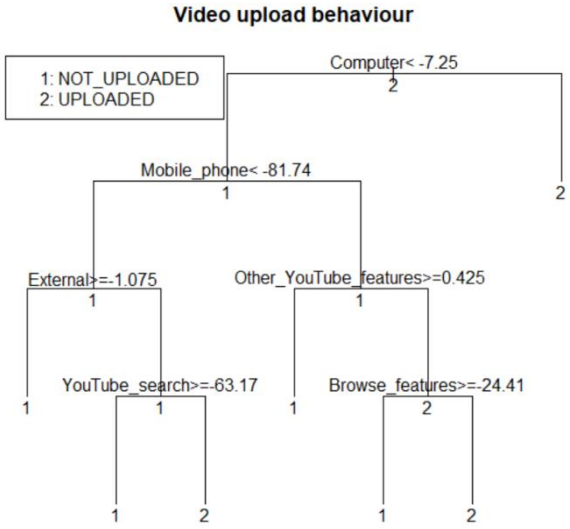


Figure 11 TBM for complete dataset

The resultant tree (Fig 11) could have grossly overfit the small number of observations I have (17 attributes, 161 observations means barely 9 observations per variable) so I included a cross validation step as well. Finally, I tested my model and achieved 58.3% accuracy. The RMSE is 0.64 which is quite large, and I expect it to decrease as more observational data is fed to the model. An interesting remark here would be that the model curated the most important decisions for: computer, mobile phone, external, other YouTube features, YouTube search and browse features, which largely coincides with the outcome of factor analysis where I had declared a few other variables to be equally important.

Discussion: This project uncovered a few interesting insights about my YouTube channel that may potentially hold true for others that are music-centric as well. Most of the meta data related to the videos shows some level of hierarchy in the strength of correlation with the views it contributes to my channel. I focused on views because they are proportional to the number of hours and impressions generated by the videos, which is quantitative for success in YouTube. Hence, with dimension reduction techniques, I was able to affirm the order of 'importance' of variables and assess their specific variance. I also determined 3 groups of videos that share some similarity with the correlation structure of the variable space. I was then able to generate a tree based model for predicting whether a video was uploaded if given combination of certain attributes, with some success but it could benefit from more observational data in the future.

References:

1. Website (no date a). Available at: https://www.quora.com/Why-do-music-videos-on-YouTube-get-so-much-more-views?share=1 (Accessed: 23 January 2021).
2. Website (no date b). Available at: https://www.statology.org/ggplot-pie-chart/ (Accessed: 23 January 2021).
3. Website (no date c). Available at: https://rpkgs.datanovia.com/factoextra/reference/fviz_dend.html (Accessed: 23 January 2021).
4. Website (no date d). Available at: https://p.cygnus.cc.kuleuven.be/bbcswebdav/pid-29342340-dt-content-rid-285446304_2/courses/B-KUL-I0P16a-2021/Manual_PLS%20in%20R.pdf (Accessed: 23 January 2021).

Annex:

1. Head of each file used in this project

```
> head(total_views)
       Date Views
1 2020-08-13     0
2 2020-08-14     3
3 2020-08-15     5
4 2020-08-16     1
5 2020-08-17     3
6 2020-08-18    10
> head(new_videos)
       Date  Video_added
1 2020-08-13     UPLOADED
2 2020-08-14 NOT_UPLOADED
3 2020-08-15 NOT_UPLOADED
4 2020-08-16     UPLOADED
5 2020-08-17     UPLOADED
6 2020-08-18 NOT_UPLOADED
> head(traffic_view)
       Date Browse_features Channel_pages Direct_or_unknown End_screens External Notifications Other_YouTube_features
1 13-08-2020               0             0                 0           0        0             0                      0
2 14-08-2020               0             0                 0           0        0             0                      0
3 15-08-2020               0             1                 0           0        0             0                      0
4 16-08-2020               0             0                 0           0        0             0                      0
5 17-08-2020               0             0                 0           0        0             0                      0
6 18-08-2020               0             0                 1           0        0             0                      0
  Playlist_page Playlists Suggested_videos YouTube_search
1             0         0                0              0
2             0         0                0              3
3             0         0                0              4
4             0         0                0              1
5             0         0                0              3
6             0         0                0              9
> head(dev_view)
       Date Computer Game_console Mobile_phone Tablet TV
1 13-08-2020        0            0            0      0  0
2 14-08-2020        0            0            3      0  0
3 15-08-2020        1            0            4      0  0
4 16-08-2020        0            0            1      0  0
5 17-08-2020        0            0            3      0  0
6 18-08-2020        3            0            6      1  0
```

2. PCA results for device and traffic data:

```
> summary(pca_devices)
Importance of components:
                          Comp.1     Comp.2     Comp.3     Comp.4      Comp.5
Standard deviation     2.0397404 0.70380319 0.46051715 0.30600802 0.195967745
Proportion of Variance 0.8321082 0.09906779 0.04241521 0.01872818 0.007680671
Cumulative Proportion  0.8321082 0.93117594 0.97359115 0.99231933 1.000000000
```

```
> summary(pca_traffic)
Importance of components:
                          Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
Standard deviation     2.5511783 1.03461319 0.91283755 0.83764453 0.76936738 0.68709314 0.58832858
Proportion of Variance 0.5916828 0.09731131 0.07575204 0.06378621 0.05381147 0.04291791 0.03146641
Cumulative Proportion  0.5916828 0.68899412 0.76474616 0.82853237 0.88234384 0.92526175 0.95672816
                           Comp.8     Comp.9     Comp.10     Comp.11
Standard deviation     0.43207693 0.3319293 0.318624066 0.278570270
Proportion of Variance 0.01697186 0.0100161 0.009229209 0.007054672
Cumulative Proportion  0.97370002 0.9837161 0.992945328 1.000000000
```

3. PCA results for combined data:

```
> summary(pca_all)
Importance of components:
                          Comp.1     Comp.2     Comp.3    Comp.4     Comp.5     Comp.6     Comp.7
Standard deviation     3.3968875 1.04987430 0.92542690 0.8472486 0.78293735 0.76724836 0.62613346
Proportion of Variance 0.6787556 0.06483741 0.05037735 0.0422253 0.03605829 0.03462765 0.02306136
Cumulative Proportion  0.6787556 0.74359299 0.79397034 0.8361956 0.87225394 0.90688159 0.92994294
                           Comp.8     Comp.9    Comp.10     Comp.11     Comp.12    Comp.13   Comp.14
Standard deviation     0.61875385 0.46592635 0.42699983 0.355595651 0.335954061 0.26957197 0.2526929
Proportion of Variance 0.02252096 0.01276984 0.01072523 0.007438133 0.006639125 0.00427465 0.0037561
Cumulative Proportion  0.95246391 0.96523375 0.97595898 0.983397110 0.990036236 0.99431089 0.9980670
                          Comp.15      Comp.16      Comp.17
Standard deviation     0.181082031 8.399571e-03 1.439973e-08
Proportion of Variance 0.001928865 4.150164e-06 1.219719e-17
Cumulative Proportion  0.999995850 1.000000e+00 1.000000e+00
> #Plot the loadings of the first 2 components since they explain 67% & 6% variance respectively
> pca_all$loadings[,1:2]
                              Comp.1       Comp.2
Views                     0.29237021  0.001460564
Computer                  0.28507474 -0.013994427
Game_console              0.20833888 -0.126743104
Mobile_phone              0.29026921  0.023877252
Tablet                    0.27889675 -0.126650090
TV                        0.26022445  0.020107959
Browse_features           0.27618175 -0.060042859
Channel_pages             0.22504854  0.151003637
Direct_or_unknown         0.16978541  0.019280971
End_screens               0.05224566 -0.886347080
External                  0.17172810  0.265614185
Notifications             0.15072132  0.274728078
Other_YouTube_features    0.23372332 -0.048552983
Playlist_page             0.26125655  0.013636839
Playlists                 0.26777448  0.034721613
Suggested_videos          0.27769942 -0.062003895
YouTube_search            0.27874993  0.026711860
```

4. Factor Analysis on complete dataset:

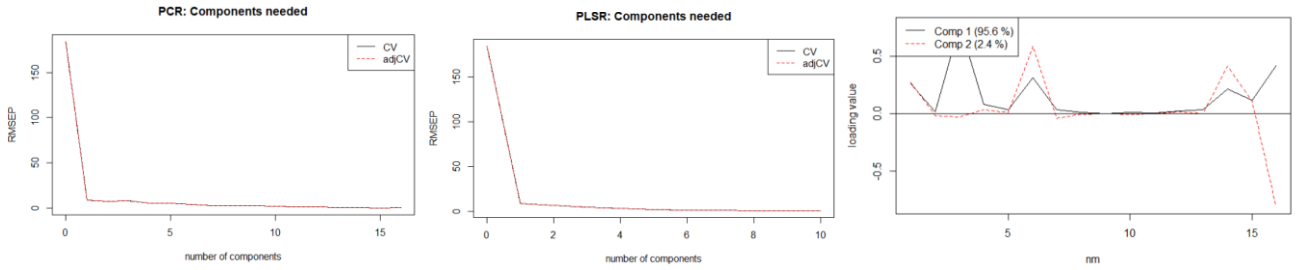
A) Command call:

```
> factanal(m,factors=4,scores="Bartlett",rotation="varimax")

Call:
factanal(x = m, factors = 4, scores = "Bartlett", rotation = "varimax")

Uniquenesses:
              Computer           Game_console           Mobile_phone                 Tablet                     TV
                 0.056                  0.515                  0.005                  0.005                  0.240
       Browse_features          Channel_pages      Direct_or_unknown            End_screens               External
                 0.005                  0.410                  0.672                  0.711                  0.654
         Notifications Other_YouTube_features          Playlist_page              Playlists      Suggested_videos
                 0.716                  0.371                  0.221                  0.005                  0.121
        YouTube_search
                 0.005

Loadings:
                      Factor1 Factor2 Factor3 Factor4
Computer               0.766   0.555   0.187   0.118
Game_console           0.418   0.514   0.206
Mobile_phone           0.754   0.632   0.143
Tablet                 0.687   0.560   0.444  -0.108
TV                     0.661   0.555
Browse_features        0.846   0.400   0.270   0.215
Channel_pages          0.433   0.626
Direct_or_unknown      0.267   0.452   0.221
End_screens                            0.533
External               0.312   0.492
Notifications          0.253   0.363   0.264   0.134
Other_YouTube_features 0.661   0.435
Playlist_page          0.651   0.564   0.178
Playlists              0.897   0.402          -0.168
Suggested_videos       0.733   0.523   0.244
YouTube_search         0.537   0.823   0.157

                Factor1 Factor2 Factor3 Factor4
SS loadings       5.869   4.349   0.907   0.164
Proportion Var    0.367   0.272   0.057   0.010
Cumulative Var    0.367   0.639   0.695   0.706

Test of the hypothesis that 4 factors are sufficient.
The chi square statistic is 757.27 on 62 degrees of freedom.
The p-value is 3.3e-120
```

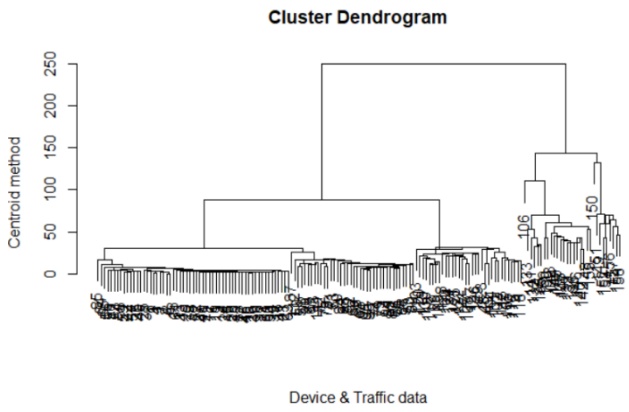
B) The specific variance:

```
> as.data.frame(fam$uniquenesses)
                       fam$uniquenesses
Computer                     0.05552633
Game_console                 0.51494676
Mobile_phone                 0.00500000
Tablet                       0.00500000
TV                           0.23981934
Browse_features              0.00500000
Channel_pages                0.41029578
Direct_or_unknown            0.67179298
End_screens                  0.71111286
External                     0.65399767
Notifications                0.71622989
Other_YouTube_features       0.37066880
Playlist_page                0.22128391
Playlists                    0.00500000
Suggested_videos             0.12080189
YouTube_search               0.00500000
```

5. PCR, PLSR & variance vs number of components curve



6. Classic dendrogram (centroid method, k=3)

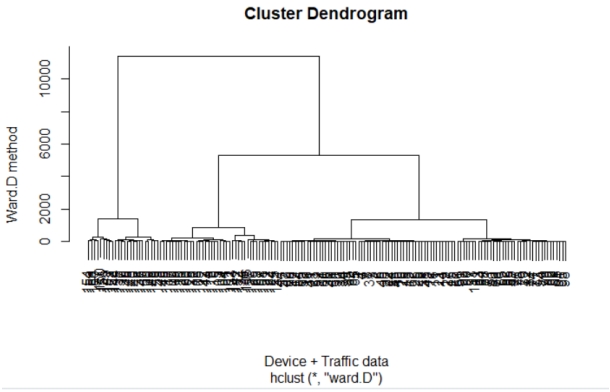


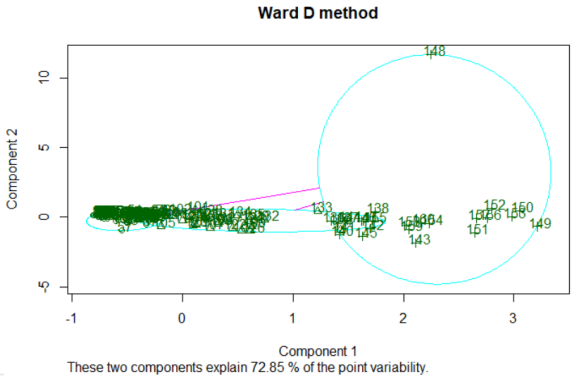
For 3 clusters, number of obserations distributed as:

1 2 3

130 21 9

7. Ward D dendrogram (k=3) & cluster plot



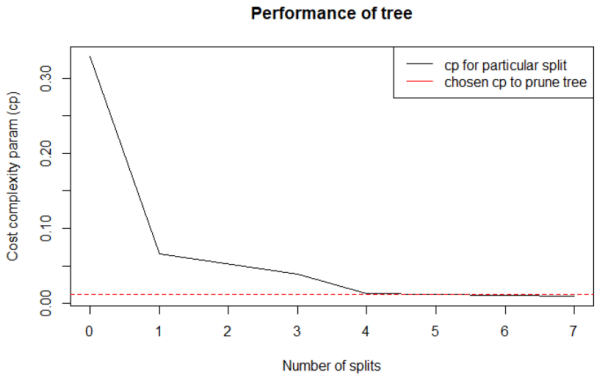


For 3 clusters, number of obserations distributed as:

1 2 3

96 40 24

8. Performance of tree with different cost complexities




```
> printcp(tree_mega)

Classification tree:
rpart(formula = added_labels ~ ., data = as.data.frame(new_mega_devtra),
    method = "class")

Variables actually used in tree construction:
[1] Browse_features   Computer          Direct_or_unknown Mobile_phone      Tablet
[6] YouTube_search

Root node error: 76/160 = 0.475

n= 160

        CP nsplit rel error  xerror     xstd
1 0.328947      0   1.00000 1.00000 0.083114
2 0.065789      1   0.67105 0.92105 0.082565
3 0.039474      3   0.53947 0.84211 0.081536
4 0.013158      4   0.50000 0.72368 0.079050
5 0.010000      7   0.46053 0.85526 0.081742
```

9. CODE:

```
## All packages needed for this code
#install.packages("PerformanceAnalytics")
#install.packages("plotrix")
#install.packages("andrews")
#install.packages(c("factoextra", "dendextend"))
#install.packages("caret")

library(andrews)
library(PerformanceAnalytics)
library(scales)
library(plotrix)
library(pls)
library(cluster)
library(factoextra)
library(rpart)
library(caret)


## Set working directory
setwd("C://Users//Aditya//Desktop//AppliedMultivariateStatisticalAnalysis//Project/updated_data")

#################################################################################################################
#######
#Show total view trend till date
total_views <- read.csv("total_views.csv", header=T)
head(total_views)

x = total_views$Date
y = total_views$Views
plot(x,y,col=heat.colors(5),main="View trend",xlab="Date",ylab="Views")
lines(lowess(x, y), col = "blue", lwd = 3)
legend("topleft", legend = c("Raw data", "Smooth fit line"),
```

```
       lwd = 3, col = c("black", "blue"))

#######################################--DEVICES ANALYSIS--
#################################################################
## Read data from devices data file

dev_view <- read.csv("device_view.csv", header=T)
head(dev_view[,c(2:6)])

## Visualize correlation

# Center the dataframe
d<-apply(dev_view[,c(2:6)], 2, scale, scale=FALSE, center=TRUE)

cor_devices = cor(d)
chart.Correlation(cor_devices, histogram = TRUE, method = "pearson")

## Generate Pie Chart of total view coverage

#Extract required data
dev_stat <- read.csv("device_sum.csv", header=T)
head(dev_stat[,c(1,3,4)])

#Extract the important columns
dev_stat <- dev_stat[,c(1,3,4)]

xaxisTitles<- colnames(dev_stat)

win.graph()
par(mfrow=c(2,1))
require(ggplot2)
se <- function(x) sqrt(var(x)/length(x))
plot <- ggplot(dev_stat, aes(Device_type, Views, fill = Device_type)) +
  geom_bar(width = 1, stat = "identity", color = "white") +
  geom_errorbar(aes(ymin = Views - se(Views),
                    ymax = Views + se(Views),
                    color = Device_type),
                width = .2) +
  scale_y_continuous(breaks = 0:nlevels(dev_stat$Device_type)) +
  theme_gray() + scale_x_discrete(labels = xaxisTitles)+
  ggtitle("Device contribution to views") +
  geom_text(aes(label = paste0(Views, " views \n", Watch_time_hours," hrs")), position =
position_stack(vjust=0.5))+
  theme(axis.ticks = element_blank(),
        axis.text = element_blank(),
        axis.title = element_text(color="red"),
        axis.line = element_blank())
plot

############################----------------PCA

##Analyze with PCA model
#To decide whether to use cor=T/F, check if min/max in dataframe are comparable [if no, cor=T]
summary(d)
pca_devices = princomp(d,cor=T,scores=T)
summary(pca_devices)
par(mfrow=c(2,1))
plot(pca_devices,col="#FFF8E7", main="PCA analysis: DEVICES DATA")
mtext("Principal components", side=1)
plot(pca_devices,type="l",main='Screeplot')

##Look at the loadings of first 2 PCs since they explain 93% of the data [0.832, 0.099 resp]
pca_devices$loadings[,1:2]

############################----------------BIPLOT

#Can directly do biplot(pca_devices) but its easier to reconstruct
par(mfrow=c(1,1))
scalefactor<-0.07
scores<-pca_devices$scores*scalefactor
win.graph()
plot(scores[,2]~scores[,1],,xlim=c(-0.2,1.1),ylim=c(-2,2),main='Biplot: DEVICES',xlab='PC1-83%
explained variance',ylab='PC2-10% explained variance',col="blue")

for (i in seq(1,nrow(pca_devices$loadings),by=1))
```

```
  arrows(0,0,pca_devices$loadings[i,1],pca_devices$loadings[i,2],lwd=2)
text(pca_devices$loadings[,1]+.16,pca_devices$loadings[,2]+0.02*2,
     as.character(dimnames(cor_devices)[[2]]),font=3)

draw.circle(0,0,1,border='black')
legend("bottomright",legend=c("PC scores","PC eigenvectors"),col=c("blue","black"),pch=16)

#######################################--TRAFFIC ANALYSIS--
################################################################
## Read data from traffic data file
traffic_view <- read.csv("traffic_view.csv", header=T)
head(traffic_view[,c(2:12)])

par(mfrow=c(1,1))
## Visualize correlation

#Center the dataframe
t<-apply(traffic_view[,c(2:12)], 2, scale, scale=FALSE, center=TRUE)

cor_traffic = cor(t)
chart.Correlation(cor_traffic, histogram = TRUE, method = "pearson")

#Another way to view correlation between plots
library(GGally)
ggpairs(traffic_view[,c(2:12)])

############################
## Display proportion of data contributed by each attribute
#Extract required data
traffic_stat <- read.csv("traffic_sum.csv", header=T)
head(traffic_stat)

#Extract the important columns
traffic_stat <- traffic_stat[,c(1,2,3,5,6)]
traffic_stat[is.na(traffic_stat)] = 0

xaxisTitles<- colnames(traffic_stat)

require(ggplot2)
se <- function(x) sqrt(var(x)/length(x))
plot <- ggplot(traffic_stat, aes(Traffic_source, Views, fill = Traffic_source)) +
  geom_bar(width = 1, stat = "identity", color = "white") +
  geom_errorbar(aes(ymin = Views - se(Views),
                    ymax = Views + se(Views),
                    color = Traffic_source),
                width = .2) +
  scale_y_continuous(breaks = 0:nlevels(traffic_stat$Traffic_source)) +
  theme_gray() + scale_x_discrete(labels = xaxisTitles)+
  ggtitle("Traffic source for views (inside: Impressions by attribute)") +
  geom_text(aes(label = paste0(Impressions)), position = position_stack(vjust=0.5))+
  theme(axis.ticks = element_blank(),
        axis.text = element_blank(),
        axis.title = element_text(color="red"),
        axis.line = element_blank())
plot+ coord_polar()

############################----------------PCA

##Analyze with PCA model
pca_traffic = princomp(t,cor=T)
summary(t)
plot(pca_traffic,col="#FFF8E7", main="PCA analysis: TRAFFIC SOURCE")
mtext("Principal components", side=1)
plot(pca_traffic,type="l",main='Screeplot')

summary(pca_traffic)
##Look at the loadings of first 5 PCs since they explain 87.9% of the data
[0.591,0.097,0.075,0.063,0.053 resp]
pca_traffic$loadings[,1:5]

############################---------------BIPLOT

#Try biplot even though it is useless since more than 2 PCs are significant here
par(mfrow=c(1,1))
scalefactor<-0.07
```

```
scores<-pca_traffic$scores*scalefactor
win.graph()
plot(scores[,2]~scores[,1],,xlim=c(-0.2,1.1),ylim=c(-2,2),main='Biplot: TRAFFIC',xlab='PC1-60%
explained variance',ylab='PC2-10% explained variance',col="blue")

for (i in seq(1,nrow(pca_traffic$loadings),by=1))
  arrows(0,0,pca_traffic$loadings[i,1],pca_traffic$loadings[i,2],lwd=2)
text(pca_traffic$loadings[,1]+.16,pca_traffic$loadings[,2]+0.02*2,
     as.character(dimnames(cor_traffic)[[2]]),font=3)

draw.circle(0,0,1,border='black')
legend("bottomright",legend=c("PC scores","PC eigenvectors"),col=c("blue","black"),pch=16)


#######################################--COMBINED META DATA ANALYSIS--
############################################################

mega_devtra = cbind(d,t)
#See how these variables affect the view trend
view_all <- cbind(y,mega_devtra)
colnames(view_all)[colnames(view_all) == "y"] <- "Views"
chart.Correlation(cor(view_all), histogram = F, method = "pearson")

m<-apply(mega_devtra, 2, scale, scale=FALSE, center=TRUE)

summary(mega_devtra)

##########################----------PCA

##Analyze with PCA model
pca_mega = princomp(mega_devtra,cor=T)

par(mfrow=c(2,1))
plot(pca_mega,col="#FFF8E7", main="PCA analysis: DEVICE + TRAFFIC ")
mtext("Principal components", side=1)
plot(pca_mega,type="l",main='Screeplot')

summary(pca_mega)
##Look at the loadings of first 2 PCs since they explain 66% & 7% variance resp [0.659,0.068]
pca_mega$loadings[,1:2]


##########################----------FACTOR ANALYSIS

## FACTOR ANALYSIS
fam <- factanal(m,factors=4,scores="Bartlett",rotation="varimax")
#See effect of individual variables on each factor
fam$loadings
#Check psi (specific variances)
sp_var <- as.data.frame(fam$uniquenesses)
plot(c(1:16),sp_var$`fam$uniquenesses`,main="Factor Analysis",xlab="Variable",ylab="FA psi
score",xlim=c(0,17),ylim=c(0,0.8),col= ifelse(sp_var$`fam$uniquenesses` >= 0.1, "red", "black"))
text(sp_var,labels=dimnames(sp_var)[[1]],cex=0.7,pos=3)
legend("topright",legend=c(">10% variance","<10% variance"),col=c("red","black"),pch=16)

##########################----------PCR

## TRY PCR
#Modify all 0's to 1 so that the model can run
view_all_rep = view_all
pcr_o1 <- pcr(y ~ view_all_rep[,c(2:17)], data = as.data.frame(view_all_rep), scale=F,validation =
"CV")
summary(pcr_o1)
plot(RMSEP(pcr_o1), legendpos = "topright", main="PCR: Components needed")
predplot(pcr_o1)

##########################----------PLSR

## TRY PLSR
pls_o1 <- plsr(y ~ view_all[,c(2:17)] , ncomp = 10, data = as.data.frame(view_all), validation =
"LOO")
#Check in LOOCV there is no significant change after 6th component
summary(pls_o1)
#2 or 1 components seem to be enough to describe the data using PLSR
plot(RMSEP(pls_o1), legendpos = "topright", main="PLSR: Components needed")
```

```
#Plot the loadings of the first 2 components
plot(pls_o1, "loadings", comps = 1:2, legendpos = "topleft", xlab = "nm")
abline(h = 0)
#1 component is enough

colnames(view_all_rep)[colnames(view_all_rep) == "y"] <- "Views"

#I am overlaying the PC component of Views on the mega biplot to confirm my
#hypothesis that majority of variables are contributing to it positively
pca_all<-princomp(view_all_rep,cor=T)


############################----------------BIPLOT

#Try biplot
par(mfrow=c(1,1))
scalefactor<-0.06
scores<-pca_mega$scores*scalefactor
#win.graph()
plot(scores[,2]~scores[,1],xlim=c(-0.2,1.1),ylim=c(-1,1),main='Biplot: DEVICE + TRAFFIC',xlab='PC1-
66% explained variance',ylab='PC2-7% explained variance',col="green")

for (i in seq(1,nrow(pca_mega$loadings),by=1))
  arrows(0,0,pca_mega$loadings[i,1],pca_mega$loadings[i,2],lwd=1,col="orange")

draw.circle(0,0,1,border='black')
arrows(0,0,pca_all$loadings[1,1],pca_all$loadings[1,2],lwd=2)
legend("bottomright",legend=c("PC scores","PC
eigenvectors","Views"),col=c("green","orange","black"),pch=16)

summary(pca_all)
#Plot the loadings of the first 2 components since they explain 67% & 6% variance respectively
pca_all$loadings[,1:2]

############################----------------HIERARCHICAL CLUSTER ANALYSIS

# Cluster
#Treat all records as profiles
plot(c(0,16),c(-100,500),type="n",ylab="Range",xlab="Variables",main="Profile Plot")
# Use a loop to generate a profile line for each observation.
for (k in (1:160))
{
  points(1:16,mega_devtra[k,],type="l")
}
#Try Andrews plot
andrews(mega_devtra,type=4,ymax=2,clr=2)
#Not as good as traditional points above

#Ward.D method
par(mfrow=c(1,1))
clust_mega_ward <- hclust(dist(mega_devtra), method="ward.D")
plot(clust_mega_ward,xlab="Device + Traffic data",ylab="Ward.D method")
extract_clust_mega_ward <- cutree(clust_mega_ward,k=3)
table(extract_clust_mega_ward)
clusplot(mega_devtra,extract_clust_mega_ward,stand=TRUE,labels=3,main="Ward D method")

# Centroid method
win.graph()
par(mfrow=c(1,1))
clust_mega <- hclust(dist(mega_devtra), method="centroid")
plot(clust_mega,xlab="Device & Traffic data",ylab="Centroid method",sub="")
extract_clust_mega <- cutree(clust_mega,k=3)
table(extract_clust_mega)
clusplot(mega_devtra,extract_clust_mega,stand=TRUE,labels=3,main="Centroid method")

#Display dendogram more nicely
win.graph()
fviz_dend( hcut(mega_devtra, k = 3, stand = TRUE), k_colors = "aaas",tittle = "Lower", ggtheme =
theme_classic(),cex = 0.5, k = 3, rect = T)

############################----------------TBM

#Run Tree building methodology
#Add another variable, categorical "videos added"-> UPLOADED/NOT_UPLOADED
new_videos <- read.csv("videos_added.csv", header=T)
```

```
#If number of videos uploaded is not 0, mark it as uploaded
new_videos$Video_added[new_videos['Video_added'] > 0] <- "UPLOADED"
new_videos$Video_added[new_videos['Video_added']== 0] <- "NOT_UPLOADED"
added_labels <- as.factor(new_videos$Video_added)

new_mega_devtra <- as.data.frame(cbind(added_labels, mega_devtra))

# Fit the model

#TRAIN & TEST SETS
indexes = createDataPartition(new_mega_devtra$added_labels, p = .85, list = F)
train = new_mega_devtra[indexes, ]
test = new_mega_devtra[-indexes, ]

test_x = test[, 2:17]
test_y = test[,1]

#Choose method as class since we are trying to predict a factor variable
tree_mega <- rpart(added_labels ~.,data=train,method="class")
rpart.control(xval=10)
summary(tree_mega)

win.graph()
plot(tree_mega,uniform=T)
text(tree_mega,splits=T,all=T)
title("Video upload behaviour")
legend("topleft",legend=c("1: NOT_UPLOADED","2: UPLOADED"))

# Results of cross validation
printcp(tree_mega)
plotcp(tree_mega)

#Plot cost complexity vs # splits
plot(tree_mega$cptable[,2],tree_mega$cptable[,1],xlab='Number of splits',ylab='Cost complexity param
(cp)',type='l',main="Performance of tree")
abline(0.013,0,col="red",lty=2)
legend("topright",legend=c("cp for particular split", "chosen cp to prune
tree"),col=c("black","red"),lty=1)

# Pruning the tree at a specific cost-complexity parameter cp
tree_prune<-prune(tree_mega,cp=0.013)

summary(tree_prune)

# Plot the pruned tree
win.graph()
plot(tree_prune)
text(tree_prune)
title("Pruned tree at cp=0.013")
legend("topright",legend=c("1: NOT_UPLOADED","2: UPLOADED"))
#No difference observed before/after pruning [too few variables]

#Check accuracy
pred_y = as.data.frame(predict(tree_prune, data.frame(test_x)))
pred_label <- ifelse(pred_y$'1' > pred_y$'2', 1,2)

print(data.frame(test_y, pred_label))

mse = mean((test_y - pred_label)^2)
mae = caret::MAE(test_y, pred_label)
rmse = caret::RMSE(test_y, pred_label)

cat("MSE: ", mse, "MAE: ", mae, " RMSE: ", rmse)
x = 1:length(test_y)

plot(x, test_y, col = "red", type = "l", lwd=2, main = "test data prediction")
lines(x, pred_label, col = "blue", lwd=2)
legend("topright",  legend = c("original label", "predicted label"),
       fill = c("red", "blue"), col = 2:3,  adj = c(0, 0.6))
grid()
acc <- length(test_y[test_y == pred_label])/length(test_y)
#########################################################################################################
#######
```