**Final R code**

#----------------------------------------------------------------------------------#

### 1- DATA PRE-PROCESSING

library(dplyr)

### Transforming our variables

# 1- Frequency , 2- Dislike Percentage, 3- Trending Days

# The objective is to form a table of unique values based on video\_id and append 3 columns with variables for # the above attributes

### Forming the table with unique values based on video\_id such

# Reading the dataset

df <- read.csv('CAvideos.csv')

# Sorting as per 'video\_id'

dfsort <- arrange(df,video\_id)

# Assigning categories to the videos

dfsort$category\_id <- ordered(dfsort$category\_id,

                              levels = c(1, 2, 10, 15, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 30, 31, 32, 33, 34,

                                         35, 36, 37, 39, 40, 41, 42, 43, 44, 38),

                              labels = c("Film & Animation", "Autos & Vehicles", "Music", "Pets & Animals", "Sports", "Short Movies",

                                         "Travel & Events", "Gaming", "Videoblogging", "People & Blogs", "Comedy", "Entertainment",

                                         "News & Politics", "Howto & Style",  "Education", "Science & Technology", "Movies",

                                         "Anime/Animation", "Action/Adventure", "Classics", "Comedy", "Documentary", "Drama",

                                         "Family","Horror", "Sci-Fi/Fantasy", "Thriller", "Shorts", "Shows", "Trailers", "Foreign"))

View(dfsort)

#changing the column name of category id to category

colnames(dfsort)[5] <- "category"

# Extracting unique values from sorted dataset

dfnew <- dfsort[order(dfsort$video\_id, -abs(dfsort$likes), -abs(dfsort$dislikes), -abs(dfsort$comment\_count)), ]

videoUniq <-  dfnew[ !duplicated(dfnew$video\_id), ]

## 1- Frequency of a video [No. of times it trended]

freq.df = count(df, df$video\_id)

videoUniq['frequency'] <- freq.df$n

### 2- Dislike Percentage for each video

videoUniq['dislike\_percent'] <-  (videoUniq$dislikes) / (videoUniq$likes + videoUniq$dislikes)

## 3- Days to Trending for each video

# Separating 'trending\_date' and 'publish\_time' into 2 different columns

y.df <- videoUniq[2]

x.df <- videoUniq[6]

View(y.df)

View(x.df)

# Extracting the date from column 'publish\_time'

#x.df %>%

x.df <- mutate(x.df, publish\_time = as.Date(substr(publish\_time, 1, 10)))

# Converting the date-format of 'trending\_date' from yyddmm to yymmdd

#y.df %>%

y.df <-  mutate(y.df, trending\_date= as.Date(trending\_date, format = "%y.%d.%m"))

View(y.df)

View(x.df)

# Combining the new dates into one single dataframe

z.df <- videoUniq[1]

z.df['date\_of\_publishing'] <- x.df$publish\_time

z.df['date\_of\_trending'] <- y.df$trending\_date

View(z.df)

# Finding the number of days between the 2 dates

z.df <- mutate(z.df, No\_of\_days= as.numeric(date\_of\_trending-date\_of\_publishing, units = "days") )

View(z.df)

# So, 'z.df' is the dataframe with the columns that can be added to our original dataset

videoUniq ['date\_of\_publishing'] <- z.df$date\_of\_publishing

videoUniq ['date\_of\_trending'] <- z.df$date\_of\_trending

videoUniq ['days\_to\_trend'] <- z.df$No\_of\_days

#drpping the original trending date column and publish time

videoUniq <- videoUniq[,-c(2)]

videoUniq <- videoUniq[,-c(5)]

View(videoUniq)

##-----------------------------------------------------------------------------------------------------------------------##

### 2- PRINCIPAL COMPONENT ANALYSIS - PCA

videoUniq1 <- videoUniq

pcs <- prcomp(na.omit(videoUniq1[,-c(1,2,3,4,5,10,11,12,13,14,17,18)]), scale=T)

summary(pcs)

pcs$rot

##-----------------------------------------------------------------------------------------------------------------------##

### 3- CLASSIFICATION TREE

#Classification tree

library(rpart)

library(rpart.plot)

library(rattle)

videoUniq.df <- videoUniq

videoUniq.df <- videoUniq.df[ , -c(1,2,3,5,10,14,17,18)]

default1.ct <- rpart(days\_to\_trend ~ category + frequency, data = videoUniq.df, method = "class", control = rpart.control(cp = 0.001))

prp(default1.ct, type = 5, extra = 10, under = TRUE, split.font = 50, varlen = -100)

printcp(default1.ct)

##-----------------------------------------------------------------------------------------------------------------------##

### 4- LINEAR REGRESSION

library(caret)

library(tidyverse)

library(forecast)

library(leaps)

library(dplyr)

# Selecting variables for regression

videoRegVar.df <- videoUniq[, c(6,7,8,9,15,16,19)]

# Data partition

set.seed(123)

training.index <- createDataPartition(videoRegVar.df$days\_to\_trend, p = 0.9, list = FALSE)

videoReg.train.df <- videoRegVar.df[training.index, ]

videoReg.valid.df <- videoRegVar.df[-training.index, ]

### Run regression

video.lm <- lm(days\_to\_trend ~ ., data = videoReg.train.df)

options(scipen = 999)

summary(video.lm)

##-----------------------------------------------------------------------------------------------------------------------##

### 5- RANDOM FOREST

library(randomForest)

videoUniq\_new<- videoUniq[complete.cases(videoUniq), ]

videoUniq1 <- videoUniq\_new[, c(6,7,9,8,11,12,13,15,16,19)]

train.index <- createDataPartition(videoUniq1$days\_to\_trend, p = 0.8, list = FALSE)

train.df <- videoUniq1[train.index,]

valid.df <- videoUniq1[-train.index,]

train.df$days\_to\_trend <- as.character(videoUniq$days\_to\_trend)

train.df$days\_to\_trend <- as.factor(videoUniq$days\_to\_trend)

rf.reg <- randomForest(days\_to\_trend ~., data = train.df)

plot(rf.reg, col = "red")

##-----------------------------------------------------------------------------------------------------------------------##

### 6- ASSOCIATION RULE MINING

assoc1.df <- head(dfsort, 5000)

assoc.df <- assoc1.df[, c(1,2,14,16,18,5,4)]

assoc.df <- assoc.df[complete.cases(assoc.df), ]

assoc2.df <- assoc.df

assoc.df[] <- lapply(assoc.df, as.character)

assoc.df$days\_to\_trend<- as.numeric(as.character(assoc.df$days\_to\_trend))

assoc.df$date\_of\_publishing <-assoc2.df$date\_of\_publishing

glimpse(assoc.df)

transactionData <- ddply(assoc.df,c("video\_id","date\_of\_publishing"),

                         function(df1)paste(df1$category,

                                            collapse = ","))

transactionData$video\_id <- NULL

transactionData$date\_of\_publishing <- NULL

colnames(transactionData) <- c("items")

library(csv)

library(arules)

library(RColorBrewer)

dev.off()

write.csv(transactionData,"D:/R-Documents/market\_basket\_transactions.csv", quote = FALSE, row.names = TRUE)

trans <- read.transactions('D:/R-Documents/market\_basket\_transactions.csv', format = 'basket', sep=',')

summary(trans2)

# Create an item frequency plot for the top 15 items

itemFrequencyPlot(trans,topN=15,type="absolute",col=brewer.pal(8,'Pastel2'),

                  main="Absolute Item Frequency Plot",

                  ylab= 'Absolute Frequency of Categories')

itemFrequencyPlot(trans,topN=15,type="relative",col=brewer.pal(8,'Pastel2'),

                  ylab= 'Relative Frequency of Categories',

                  main="Relative Item Frequency Plot")

rules <- apriori(trans, parameter = list(supp = 0.2, conf = 0.5, target = "rules"))

inspect(head(sort(rules, by = "lift")))

##-----------------------------------------------------------------------------------------------------------------------##

### 7- LINEAR DISCRIMINANT ANALYSIS - LDA

library(MASS)

library(dplyr)

library(ggplot2)

## Fit the model

video1<- videoRegVar.df[order(videoRegVar.df$days\_to\_trend),]

lda\_1 <- lda(days\_to\_trend~.,data=video1)

##Compute LDA

lda\_1

## Make Predictions

predict\_1 <- predict(lda\_1,data=video1)

## Model Accuracy

mean(predict\_1$class==video1$days\_to\_trend)

## LDA plot using ggplot 2

lda\_1 <- cbind(video1[1:24229, ], predict\_1$x)

ggplot(lda\_1, aes(LD1, LD2)) +

  geom\_point(aes(color = days\_to\_trend))

# videos having less than 8 days\_to\_trend

fil <- filter(video1, days\_to\_trend < 8)

ggplot(aes(x=frequency,y=days\_to\_trend), data = fil)+

  geom\_boxplot()

##-----------------------------------------------------------------------------------------------------------------------##

### 8- VISUALIZATIONS

##Most influential creators

creators.df <- table(videoUniq$channel\_title)

barchart(head(sort(creators.df, decreasing = TRUE),15))

##Video Category Distribution

videoUniq$category\_id <- ordered(videoUniq$category)

barchart(table(videoUniq$category))

##Time Series Analysis (number of Views over year)

view.ts <- ts(videoUniq$views, start = c(2017, 1), end = c(2018, 6), freq = 12)

plot(view.ts, xlab = "Year", ylab = "Views on Youtube Videos")

##Time Series Analysis (number of likes over year)

like.ts <- ts(videoUniq$likes, start = c(2017, 1), end = c(2018, 6), freq = 12)

plot(like.ts, xlab = "Year", ylab = "Likes on Youtube Videos")

##Time Series Analysis (number of dislikes over year)

dislike.ts <- ts(videoUniq$dislikes, start = c(2017, 1), end = c(2018, 3), freq = 12)

plot(dislike.ts, xlab = "Year", ylab = "Dislikes on Youtube Videos")

## Scatter plot to find out correlation between likes and views for videos

ggplot(videoUniq) +

  geom\_point(aes(x = views, y = likes), color = "navy", alpha = .7) +

  theme\_classic()

##Days to trending status

barchart(table(videoUniq$days\_to\_trend))

# Filled Density Plot (Distribution for dislike percentage)

ggplot(videoUniq, aes(x=dislike\_percent)) +

  geom\_histogram(aes(y=..density..),

                 binwidth=.5,

                 colour="black") +

  geom\_density(alpha=.2, fill="#FF6666")

## Not Instant Hits : Success by dislike percentage

ggplot(videoUniq) +

  geom\_point(aes(x = likes, y = dislikes), color = "navy", alpha = .7) +

  theme\_classic()

##-----------------------------------------------------------------------------------------------------------------------##

### 9- DATA MINING AND SENTIMENT ANALYSIS

library(tidyverse)

library(tidytext)

library(glue)

library(stringr)

library(SnowballC)

library(tm)

library(wordcloud)

library(wordcloud2)

library(gridExtra) #viewing multiple plots together

library(RColorBrewer)

video\_uniq\_mine <- head(videoUniq$tags,5000)

str\_replace\_all(video\_uniq\_mine, "[^[:alnum:]]", " ")

video\_uniq\_mine <- iconv(video\_uniq\_mine, 'UTF-8', 'ASCII')

corpus <- Corpus(VectorSource(video\_uniq\_mine))

inspect(corpus[1:5])

# Convert the text to lower case

corpus <- tm\_map(corpus, tolower)

# Remove punctuations

corpus <- tm\_map(corpus, removePunctuation)

# Remove numbers

corpus <- tm\_map(corpus, removeNumbers)

# Remove english stopwords

cleanset <- tm\_map(corpus, removeWords, stopwords('english'))

# Remove URL and foreign language urls'

removeURL <- function(x) gsub('http[[:alnum:]]\*', '', x)

cleanset <- tm\_map(cleanset, content\_transformer(removeURL))

# Eliminate extra white spaces

cleanset <- tm\_map(cleanset, stripWhitespace)

# Remove additional stopwords

# Remove additional stopwords

cleanset <- tm\_map(cleanset, removeWords, c('ã˜â','ã™â€','ã\u0090â','ã™â€žã˜â','ã™å','ã™ë','â\u0081ã',

                                            'noahã','â€žã','â€˜ã','ã™â€ž','â\u008dã','ùšø','ùˆø','ùƒø','ùšù',

                                            'ùˆù','ñ€ð','ù\u0081ø','ùšùˆø','ñ\u0081ð','ûœø','ù\u0081ùšø',

                                            'noahå','ùƒù','ukur','˜çš','ù\u0081ù','noahé','æžœçˆ','ùˆù\u0081ø',

                                            'espnespn','ñœð','ùšùˆù','è\u0081žå','ñƒð','â€\u009d','â€žÃ','Â\u0081Ã',

                                            'â€šÃ','â€˜Ã','â€\u009dÃ ','â€ž','noahÃ','Â\u008dÃ','ËœÃ','Â\u0090Ã',

                                            'â€žÂ','Â\u009dÃ','Â\u008fÃ','â€\u009dÂ','Â\u009dÂ','â€šÂ','â€œÃ',

                                            'Â\u008fÂ','Â\u0090Â ','Â\u0081Â ','â€™Ã','Â\u008dÂ','â€žâ€',

                                            'Â\u009dÅ','Â\u008dâ€','ËœÂ','Ëœâ€','ËœÅ','â€œÂ','Â\u008dÅ','â€˜Â',

                                            'Â\u0090Å','â€š','Â\u0081â€','â€˜ËœÃ','thÃ','Â\u0081Å','â€™Â','Â\u0081â€œÃ',

                                            'ËœÂ\u0081Ã','Â\u008dÂ\u0081Ã'))

# Text stemming (reduces words to their root form)

cleanset <- tm\_map(cleanset, stemDocument)

View(cleanset)

inspect(cleanset[1:10])

dtm <- TermDocumentMatrix(cleanset)

m <- as.matrix(dtm)

v <- sort(rowSums(m),decreasing=TRUE)

d <- data.frame(word = names(v),freq=v)

head(d, 200)

# Generate the WordCloud

par(bg='mistyrose')

png(file="WordCloud.png",width=1000,height=1000, bg="grey30")

wordcloud(d$word, d$freq, max.words = 200, col=terrain.colors(length(d$word), alpha=0.9),

          random.order=FALSE, rot.per=0.3 )

title(main = "Most words in tags", font.main = 1, col.main = "cornsilk3", cex.main = 1.5)

dev.off()

title(main = "Most words in tags", font.main = 1, col.main = "cornsilk3", cex.main = 1.5)

dev.off()

#finding frequency of words

findFreqTerms(dtm, lowfreq = 4)

#finding association between frequent terms

findAssocs(dtm, terms = "beautiful", corlimit = 0.3)

#barplot of different words with frequency

barplot(d[1:10,]$freq, las = 2, names.arg = d[1:10,]$word,

        col ="lightblue", main ="Most frequent words",

        ylab = "Word frequencies")

#Sentiment Analysis

library(syuzhet)

library(lubridate)

sentiment <- iconv(cleanset)

s. <- get\_nrc\_sentiment(sentiment)

barplot(colSums(s.), las=2, col= rainbow(10), ylab= 'Count', main = 'Youtube Video Tags')

##-----------------------------------------------------------------------------------------------------------------------##

### 10- LOGISTIC REGRESSION

videoRegVar1.df <- videoUniq[, c(4,6,7,8,9,11,12,13,15,16,19)]

logit.reg <- glm(days\_to\_trend ~ category + views+ likes+ dislikes+comment\_count+ frequency, data = videoRegVar1.df, family = "binomial")

options(scipen=999)

summary(logit.reg)

##-----------------------------------------------------------------------------------------------------------------------##