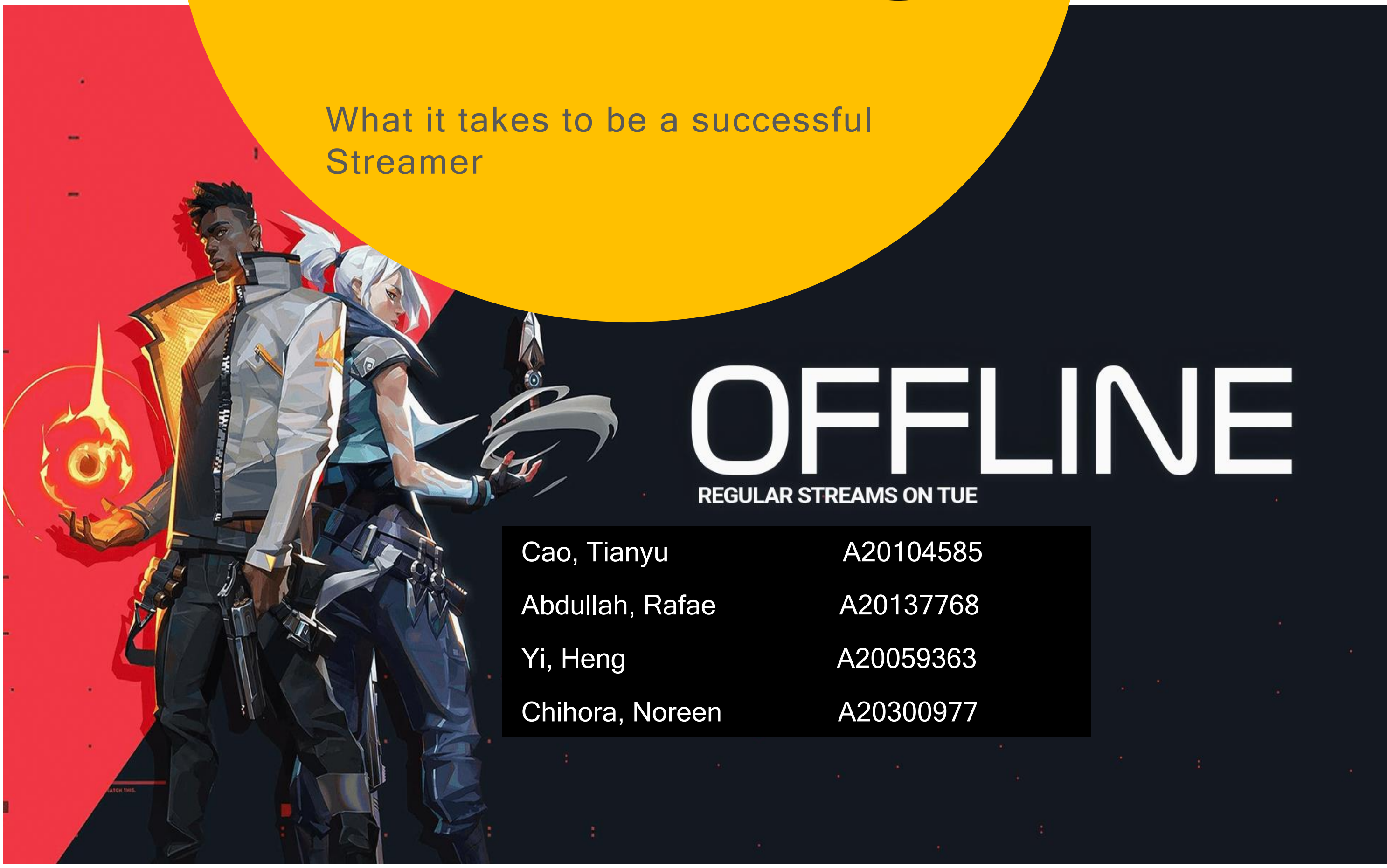
Video Game Streaming

What it takes to be a successful Streamer



Cao, Tianyu	A20104585
Abdullah, Rafae	A20137768
Yi, Heng	A20059363
Chihora, Noreen	A20300977

Date: 12/02/2021

Table of Contents:

Executive Summary 1

Statement of Scope 2-4

Project objectives.. 2

Variables.. 2

Project Schedule 4-5

Data Preparation and Data Dictionary 5-9

Data Access.. 5

Data Cleaning.. 6

Data Transformation.. 7

Data Reduction.. 7

Data Consolidation.. 7

Data Dictionary 9

Top Video Name Analysis 9

Numerical Data Analysis 10

Text Mining and Sentiment Analysis .... 11 - 23

Topic Analysis 14

Sentiment Analysis 16

Classification 20

Named-Entity Recognition 22

Refences 24

Appendix.. 25 - 142

EXECUTIVE SUMMARY

Over the last few years, the gaming live streaming industry has been experiencing some growth. While the estimated compound annual growth rate (CAGR) of the total video streaming market is 21% from 2021 to 2028 with the current valuation of USD 50.11 billion (2020 data) [1], the growth of live video streaming from April 2019 to April 2020 was 99% among four major streaming platforms [2]. This has increased the earning potential for video game streamers since the major portion (54%) of live streaming is occupied by games/esports content [3]. Our goal for this project is to figure out what makes a successful streamer. We will be collecting data from twitchtracker and twitch to figure out the attributes that contribute the most to one’s success as a streamer.

We looked at attributes like number of subscribers, total followers, total views, total hours streamed, total hours watched, average viewers and the top game streamed by the top 20 channels. We performed sentiment analysis by looking at data from chat boxes of the successful top 20 channels to see if there is a pattern in how people feel about these channels. To get better results of numerical data analysis, we also decided to look at total followers, followers gained, total hours streamed, total hours watched, average viewers by the top 46 channels in the recent 30 days (This data scrapped at the end of November).

Our statistical analysis on top 46 channels shows among all relevant independent variables only number of hours people watch a channel’s view have significant impact on number of followers, and ultimately amount of income (Note: it is true for top channels, but might not be true for channels with low ranking). The problem is a streamer cannot directly influence it. If the number of hours a streamer streams.

Fortunately, in our text analysis we have discovered a few common characteristics among the toppers. From that we can infer that making people surprise with performance, making viewers laugh with playing and talking with them are the key to success! Also, it is observed that behind the scene they play as a team (i.e. there are multiple players behind single channel). In addition to this, we have found viewers get excited most with racing games and fighting games. Thus, to make people engaging, increasing number of watch hour, and ultimately increasing number of followers and subscribers (which converts to money), gamers can focus on this games and apply these insights.

STATEMENT OF SCOPE

Our project aims to help increase income for video game live streaming channels and platforms. Our intended target variable is the number of subscribers, and our predictor variables are total followers, total views, total hours streamed, total hours watched, average viewers and the top game streamed. At first, we decided to work on top 10 channels. However, to get better statistical and sentiment analysis, later we decided focus on the top 20 channels' in terms of number of subscribers on various statistical measures chatbox text analysis. Further, we decided to perform statistical analysis on top 46 channels (in terms ofaverage viewers by the top 46 channels in the recent 30 days). Since this project's outcome will help to increase the number of followers and viewing hours for video game live streaming channels and platforms, it will help them to earn more.

Objectives

- To increase income for video game live streaming channels and platforms
- To identify what factors cause a video game live streaming channel to get subscribers

Variables

These were our variables for the first statistical analysis on top 20 channels:

1. Number of subscribers: This is our dependent/target variable. Since to keep subscription, one has to pay fees regularly, the number of subscribers is directly related to income. Thus, we choose as a target variable.
2. Total hours streamed: independent variable. We wanted to see whether streaming for longer has significant impact on our target variable.

3. Total hours watched: independent variable. We wanted to see whether watching for longer by viewers has significant impact on our target variable.
4. Average number of viewers: independent variable. We wanted to see whether number of viewers per video has significant impact on our target variable.
5. Total number of views: independent variable. This is related to the last variable. But it takes total instead of average.
6. Number of followers: independent variable. One can follow a channel without paying money. We wanted to see whether number of followers has significant impact on our target variable.

Also, we have listed name of top game. We wanted to see if top video of each of top 20 channels have a common name. This was not directly part of the statistical analysis, however.

Note: The numeric data is for January 01, 2021 to October 21, 2021 period. The reason is if a streamer streams 1 hour regularly for 10 years up to December 2020, and another streamer streams 3 hours regularly for 3 hours up to 2021, the first streamer will have the higher number of hours streamed. To avoid this and to be more relevant, we selected the period in this way. This note is true for all the numeric independent variables except number of followers.

However, using linear regression to analyze numerical data of top 20 channels leads to nearly no significant predictor. This might be caused by the fact that the overall top 20 channels have been popular and famous for a couple of years and 20 is not a large sample size compared to the entire population. All statistics related to these 20 channels tends to be stable and similar after cumulation for years. So it will be hard to detect how average viewers, hour streamed or other predictors affect number of subscribers or followers. This motivated us to find another dataset of the numerical predictors. This time our variables were as follows:

1. Time streamed: independent variable. We wanted to see whether streaming for longer has significant impact on our target variable.

2. Time watched: independent variable. We wanted to see whether watching for longer by viewers has significant impact on our target variable.

3. Average number of viewers: independent variable. We wanted to see whether number of viewers per video has significant impact on our target variable.

4. Number of followers gained: Target variable, since with common sense we can followers convert to subscribers, which was our target variable at first to know factors increasing income.

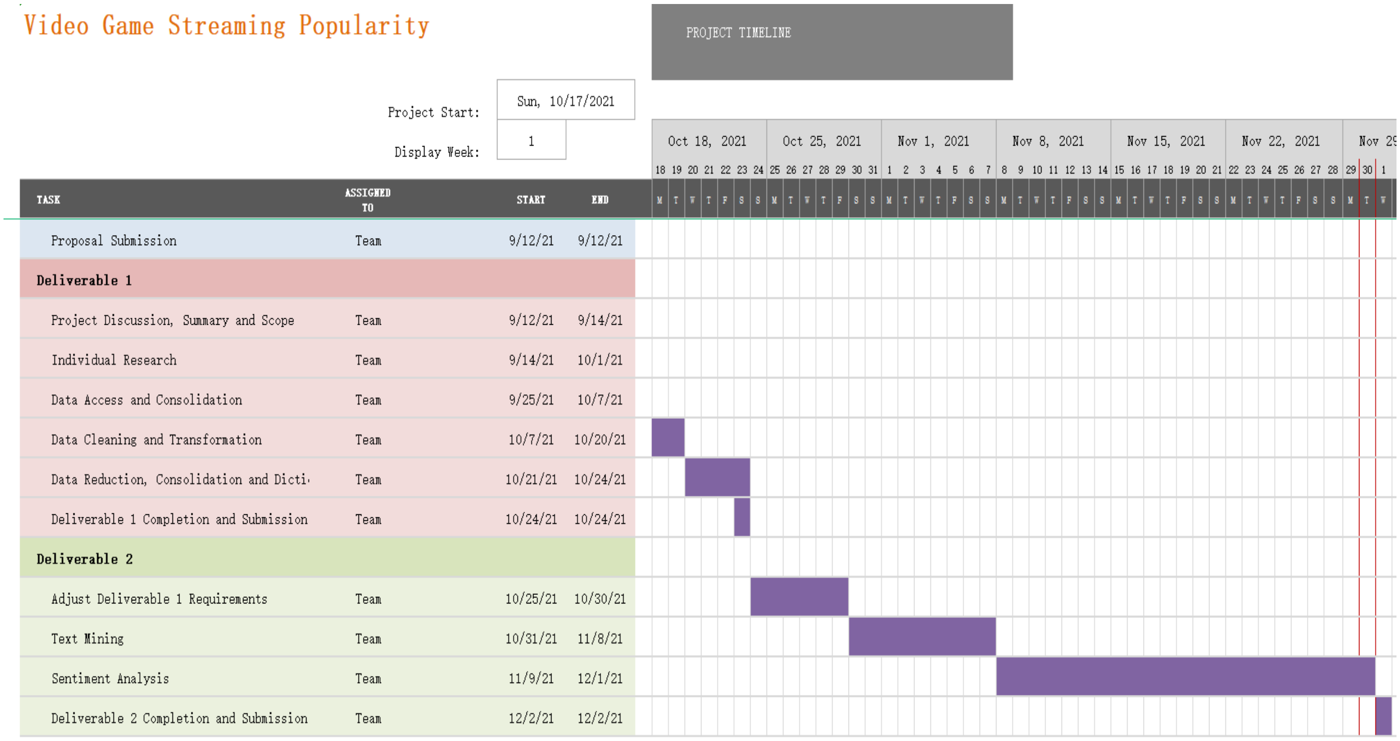
5. Total number of followers (from beginning): This is another target variable, since with common sense we can followers convert to subscribers, which was our target variable at first to know factors increasing income.

Data are of “recent” top 46 popular channels (the rank is based on the number of viewers in the past 30 days.). Our thinking was the usage of recent and relevant sample will make it clearer to see the relationship between response variable and predictors.

We will also look at Chatbox text analysis to see how it potentially affects streamers’ popularities.

PROJECT SCHEDULE

We have been meeting on a regular basis, keeping our meeting times flexible to accommodate our busy school and work schedules. This has been working for us and we are on track to successfully finish our project by the end of the semester. Each member contributed to the project. We all worked on providing code for different parts of the code. Chase (Cao) worked on scrapping data from the chatbox and provided a skeleton for the variable scrapping. Based on that, all of us scraped texts from chatbox. Rafae performed text analysis while Chase performed statistical analysis. Rafae took the responsibility to write reports with the help of Chase and Heng, while Noreen and Heng took the responsibility to prepare PowerPoint Slides.



Video Game Streaming Popularity

Project Start: Sun, 10/17/2021

Display Week: 1

TASK	ASSIGNED TO	START	END
Proposal Submission	Team	9/12/21	9/12/21
Deliverable 1			
Project Discussion, Summary and Scope	Team	9/12/21	9/14/21
Individual Research	Team	9/14/21	10/1/21
Data Access and Consolidation	Team	9/25/21	10/7/21
Data Cleaning and Transformation	Team	10/7/21	10/20/21
Data Reduction, Consolidation and Dicti	Team	10/21/21	10/24/21
Deliverable 1 Completion and Submission	Team	10/24/21	10/24/21
Deliverable 2			
Adjust Deliverable 1 Requirements	Team	10/25/21	10/30/21
Text Mining	Team	10/31/21	11/8/21
Sentiment Analysis	Team	11/9/21	12/1/21
Deliverable 2 Completion and Submission	Team	12/2/21	12/2/21

Data Preparation and Data Dictionary:

Data Access

To generate a list of the top 20 channels, and statistical variables associated with them, we used https://twitchtracker.com/ as the source since this website is solely dedicated to consolidate data on Twitch streaming and it updates the data on a regular basis.

At first we scraped the list of top 20 channels from https://twitchtracker.com . The relevant statistics on those channels were scraped from the same website. These are the statistics of October 21, 2021.

Then we go to https://twitchtracker.com/channels/viewership/english/personality to scrape data for the "recent" top 46 popular channels (The rank is based on the number of viewers in the past 30 days.) in the last week of November. The numerical variables scraped are: time streamed, time watched, average number of viewers, number of followers gained and total number of followers where the first three are predictors and the last two are response variables. It needs to be clarified that the number of subscribers is the "ultimate" response variable which directly tells us how much money a streamer has made. And number of followers is highly positively related to number of subscribers. Almost all the people will follow a channel first and then decide whether to subscribe. So for the analysis here we will consider followers gained and total followers as response variables as well.

Then we used https://www.twitch.tv/ to scrape chatbox data for sentiment analysis of the top channels's chatbox. Python Selenium package was used to scrape data with the help of XPath selector. Scraping chatbox text was not easy since it is difficult to find live videos and scraping chat from older videos was hard due to the amount of chats. Moreover, videos are often lengthy, from 5 hours to 33 hours are common lengths of the videos. However, using loops and XPath selector, we scraped chat text. We scraped texts in October and in November.

Howerver, 2 channels out of top 20 (Criticalrole and Adinross) requires subscription to view, so we could access 18 channels' chatbox.

Data Cleaning

There was no missing data in the data that we scrapped. All the attributes/variables had data that did not require any cleaning.

However, a lot of cleaning was done for text sentiment analysis. Firstly, we removed all stop words, numerical values, and punctuations and we changed the case of the letters to lower-case. Secondly, we eliminated frequently used words which are not relevant in analysis (such as https).

Instead of removing all the channels name from analysis, we checked top words and only remove those when channel's name came as top word. By this way, we can easily check which channels' names were mentioned mostly from our removed word list. It might help in future analysis. Here is example R code which lists these words:

```
list_remove = c("didnt","bahroo", "admiralbahroo", "summit", "miz", "hasan", "ðÿ", "https")

tidy_dataset3 = filter(tidy_dataset3, !(word %in% list_remove))
```

Data Transformation

When collecting and sorting the number of subscribers, number of viewers and viewing time of the top 20, because it is collected in the form of row vectors, it is necessary to convert all row vectors into vertical quantities. After that, all the data will be integrated in the form of a list by Python. See codes in the Appendix. For the data scraped from the “recent” top 46 popular channels, it has a lot of “M” and “K” which represent million and thousand respectively. Also the number in the dataset like “73,776” is considered as “character” but not “number”. To perform linear regression, we need to transform “K” and “M” into 1000 and 1000000 respectively and make sure all data is recognized as numerical data by R. See codes in the Appendix.

Data Reduction

Among the millions of channels, we took top 20 for text analysis. This reduction of data was to help us complete the project within time.

Gaming video lengths are often very high since many players play behind the screen. Instead of scraping all texts from a 33 hours video, for example, we scraped texts from different parts of the video. The reason was we found that often the chatboxes are full of “hiii” and etc., and instead of scraping those unnecessary texts we tried to scrape texts when people were actually engaging and the game’s situation was exciting. Also, saving time was another reason behind this.

Also, we tried to avoid chatbox in which language of chatting was not English since we had to clean this during sentiment analysis ultimately. We scrapped just a few chats from those as sample. In this way, we had to eliminate 7 channels out of 18 in our final text analytics.

Moreover, stemming of the words was performed for sentiment analysis.

Data Consolidation

After scraping numerical and categorical variable data from top 20 channels, these data were consolidated into 20 lists and then converted into 20 NumPy arrays. Here is an example code (details in appendix):

```
#Unify the list

list1=list_channel_name1+list_game_name1+list_total_followers1+list_avg_viewers1+list_total_views1+lis
t_hours_streamed1+list_hours_watched1


#Turn all lists into vectors

array1 = np.array(list1)
```

And finally these NumPy arrays were converted into a dataframe.

```
data =
np.vstack((array1,array2,array3,array4,array5,array6,array7,array8,array9,array10,array11,arra
y12,array13,array14,array15,array16,array17,array18,array19,array20))
```

On the other hand we had text from 18 channels. In many cases, texts from different parts / hours of a video streaming were scraped and unified like this:

```
SYKKUNO4text = [SYKKUNO, SYKKUNO2, SYKKUNO3, SYKKUNO4]

flatListSYKKUNO4 = []

for elem in SYKKUNO4text:

    flatListSYKKUNO4.extend(elem)

flatListSYKKUNO4

SYKKUNO4chatcomb = pd.DataFrame(flatListSYKKUNO4)

SYKKUNO4chatcomb.to_csv('SYKKUNO4chatcomb.txt', index = False, sep = '\t')
```

Then, as described in data reduction part, we consolidated 11 channel’s chat into one for text analysis in python. Here is code:

```
chatbox=pd.concat([XQCOW1,XQCOW2,XQCOW3,XQCOW4,XQCOW5,RANBOO1,RANBOO2,RANBOO3,RANBOO4,RANBOO5,
ADMIRALBAHROO,castro_1021,mizkif,montana,NICKMERCS,philzachatcomb,summit1gchatcomb,SYKKUNO4cha
tcomb,trainWRECKStvcomb],ignore_index=True)print(chatbox)chatbox_df=
pd.DataFrame(chatbox)chatbox_df.to_csv('D:\Documents\\final_chatbox_df.txt',index=False,sep='\
t')
```

```
  chatbox_df=
  pd.DataFrame(chatbox)
chatbox_df.to_csv('D:\Documents\\final_chatbox_df.txt',index=False,sep='\t')
```

Data Dictionary

Attribute Name	**Description**	**Data Type**	**Source**	**Example**
channel_name	Name of Channel	character	www.twitchtracker.com	RANBOOLIVE
game_name	Name of Most Popular Game/Video	character	www.twitchtracker.com	Dungeons & Dragons
total_followers	Total Number of Followers	integer	www.twitchtracker.com	893000
avg_viewers	Average Views for the Channel	integer	www.twitchtracker.com	20401
total_views	Total Number of Views for the Channel	integer	www.twitchtracker.com	26000000
hours_streamed	Number of Hours Streamed	integer	www.twitchtracker.com	1438
hours_watched	Number of Hours Channel Was Viewed	integer	www.twitchtracker.com	29300000

Top Video Name Analysis

We have found that Minecraft and Fortnite are the two games which was top ranked video for more than 1 channels. However, 6 out of 20 top video names are “Just Chatting”. It reflects how much fans love to interact with their favorite channels.

Numerical Data Analysis

For this section, the data analysis is based on the recent top 46 popular channels.

First of all, the plot of the 5 numerical variables is shown below.

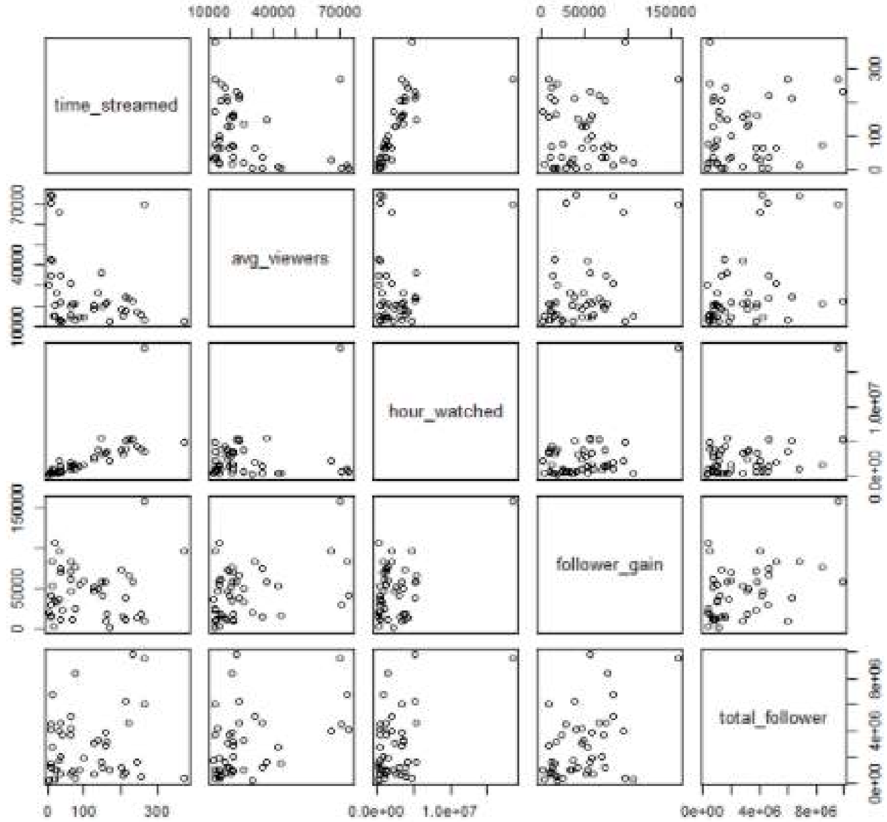
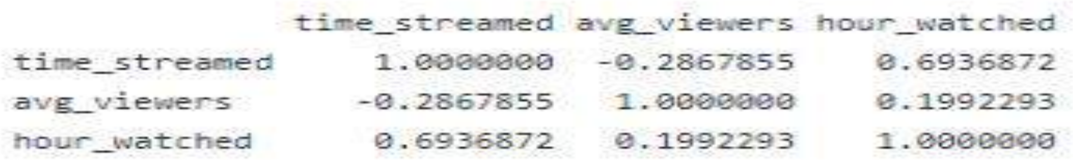


Figure: Plot of the 5 numerical variables

It's clear to see that time streamed and hour watched are positively related which makes sense intuitively. Also the number of hours watched seems to affect number of followers gained and total number of followers positively.

Because it’s linear regression with multiple predictors. We have to check the multicollinearity first. The correlation matrix of the three predictors are shown below.

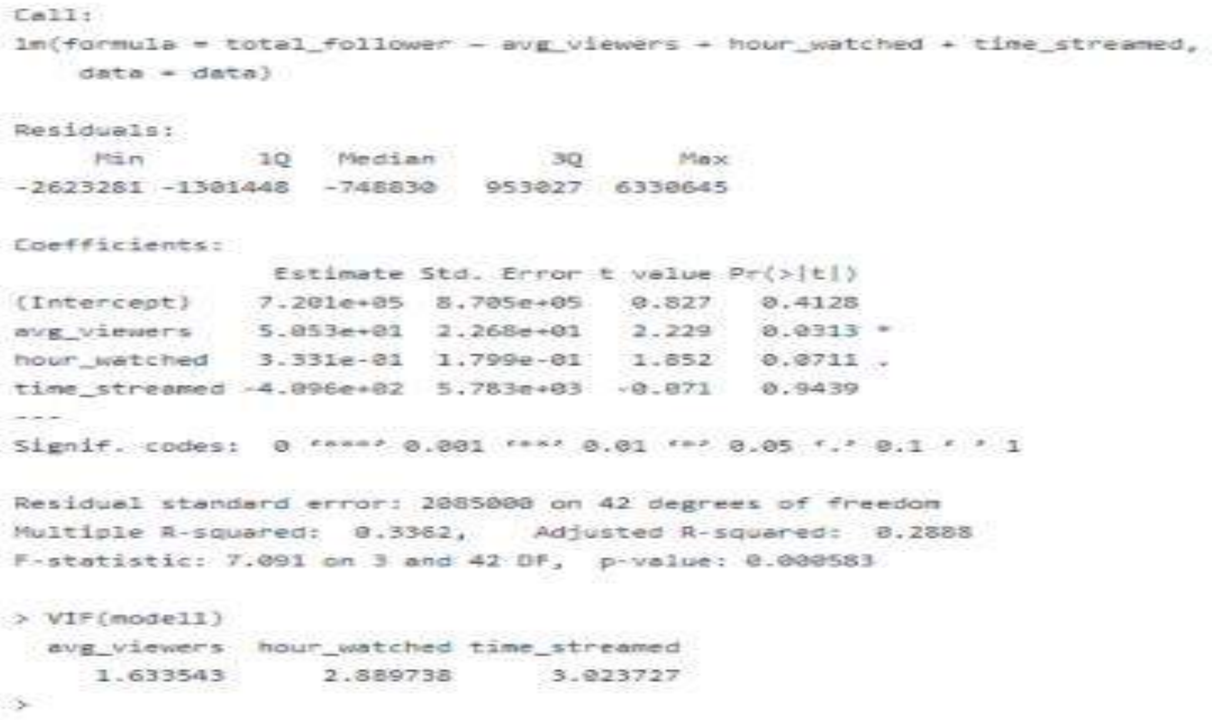


```
              time_streamed avg_viewers hour_watched
time_streamed     1.0000000  -0.2867855    0.6936872
avg_viewers      -0.2867855   1.0000000    0.1992293
hour_watched      0.6936872   0.1992293    1.0000000
```

Figure: Correlation among predictors

As we can see from the plot, time streamed and hour watched are positively correlated. We will check VIF after linear regression analysis is performed.

For model 1, the response variable is total number of followers. The summary of linear regression and VIF are shown below.

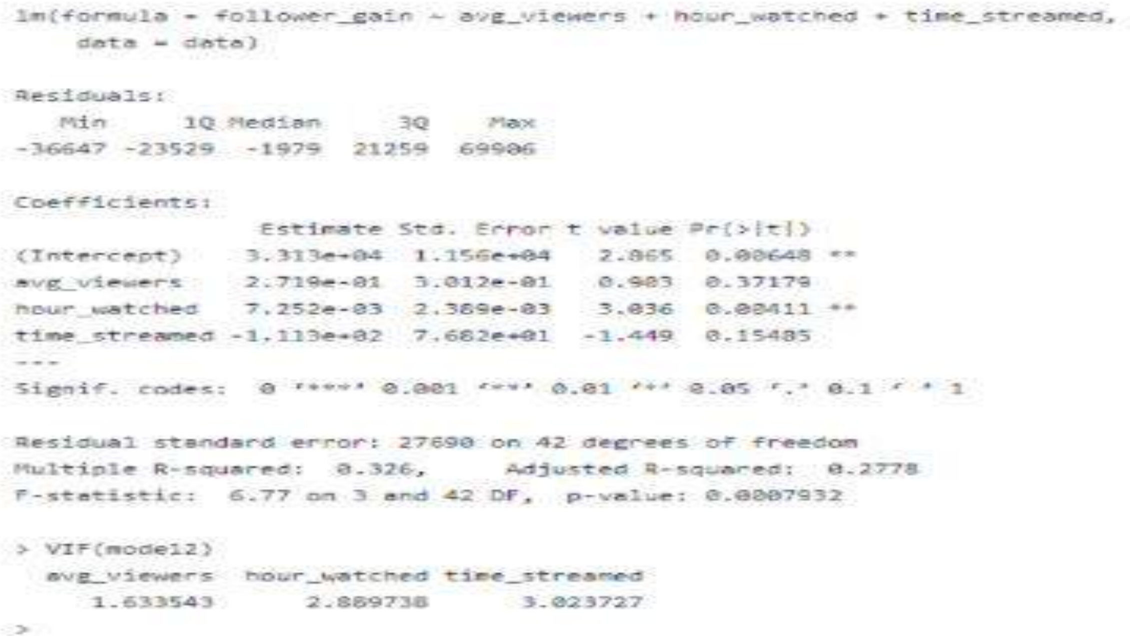


```
Call:
lm(formula = total_follower ~ avg_viewers + hour_watched + time_streamed, 
    data = data)

Residuals:
     Min       1Q   Median       3Q      Max 
-2623281 -1301448  -748830   953027  6330645 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)  
(Intercept)    7.201e+05  8.705e+05   0.827   0.4128  
avg_viewers    5.053e+01  2.268e+01   2.229   0.0313 *
hour_watched   3.331e-01  1.799e-01   1.852   0.0711 .
time_streamed -4.096e+02  5.783e+03  -0.071   0.9439  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2085000 on 42 degrees of freedom
Multiple R-squared:  0.3362,	Adjusted R-squared:  0.2888 
F-statistic: 7.091 on 3 and 42 DF,  p-value: 0.000583

> VIF(model1)
  avg_viewers  hour_watched time_streamed 
     1.633543      2.889738      3.023727 
> 
```

Figure: Linear Regression model 1

The VIF shows that there is a moderate multicollinearity which is not necessary to be adjusted for the model. And we could see that average number of viewers and number of hours watched contribute the most variation of the model and time streamed contributes the least.

For model 2, the response variable is number of followers gained in the past 30 days. The summary of the model and corresponding VIF are



```
lm(formula = follower_gain ~ avg_viewers + hour_watched + time_streamed,
    data = data)

Residuals:
   Min     1Q Median     3Q    Max 
-36647 -23529  -1979  21259  69986 

Coefficients:
                Estimate Std. Error t value Pr(>|t|)   
(Intercept)    3.313e+04  1.156e+04   2.865  0.00648 **
avg_viewers    2.719e-01  3.012e-01   0.903  0.37179   
hour_watched   7.252e-03  2.389e-03   3.036  0.00411 **
time_streamed -1.113e+02  7.682e+01  -1.449  0.15485   
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 27690 on 42 degrees of freedom
Multiple R-squared:  0.326,	Adjusted R-squared:  0.2778 
F-statistic:  6.77 on 3 and 42 DF,  p-value: 0.0007932

> VIF(model2)
  avg_viewers  hour_watched time_streamed 
     1.633543      2.889738      3.023727 
> 
```

Figure: Linear Regression model 2

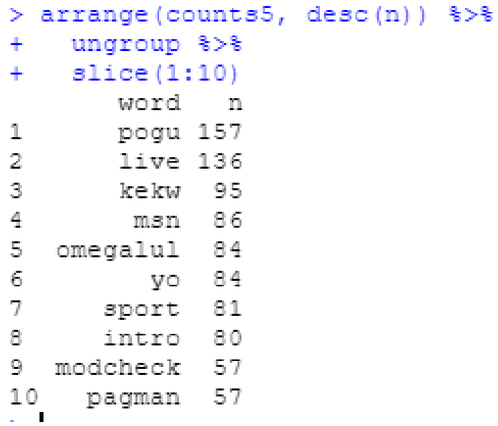
Now we could see that the number of hours watched is the most important factor.

In summary, to understand the models, let’s interpret the predictors first. The number of viewers reflects how many people enter your channel. The number of hours watched represent how long people are willing to stay in your channel. The number of hours streamed is straightfoward: how long a streamer works. From the 2 models, the number of hours watched is the most important factor to make a streamer gain more followers, which means that attracting people into your channel is just the first step and what really matters is making people stay in your channel. But how a streamer attracts people and makes them stay? We will see something useful in the sentiment analysis of the data in chatbox.

Text Mining and Sentiment Analysis

Before going for any deep analysis on chats, we wanted to see top words after completing all the data cleaning and consolidation. Here is the outcome:

Top 10 frequent words in chatbox



```
> arrange(counts5, desc(n)) %>%
+   ungroup %>%
+   slice(1:10)
       word   n
1      pogu 157
2      live 136
3      kekw  95
4       msn  86
5  omegalul  84
6        yo  84
7     sport  81
8     intro  80
9  modcheck  57
10   pagman  57
```

Honestly, it is not possible to know meaning for most of the word for general people. Thanks to google, we have figured out the meaning. They are as follows:

I. Pogu: It means surprising. “This emote is used when something cool or exciting is happening on stream” – according to streamerfacts.com

II. Live: Understandable

III. Kekw: It suggests laughter

IV. msn: We guess this came as part of spamming comment. We did not remove it so that in future analysts can check why this word came actually

V. omegalil: “When you're laughing TOO hard, you use OMEGALUL” - according to streamerfacts.com

VI. yo: could not figure it out. Maybe it is suggesting joy.

VII. Sport: Understandable

VIII. Intro: For what reason, especially at the beginning of a stream, people use it.

IX. Modcheck: It is an emote for looking around for something.

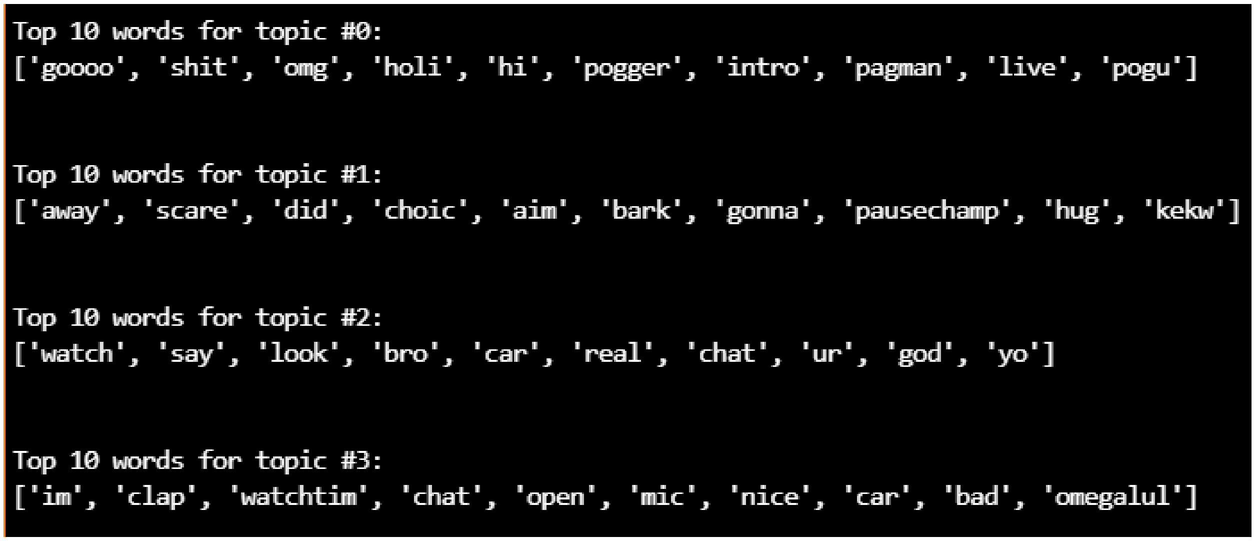
X. Pagman: It suggests surprise.

In summary, we can say, top channel’s common characteristics are play in a way which can surprise others and make others laugh! Streamers talks while streaming, maybe top streamers say something that make others laugh.

Topic Analysis:

In python, we figured out 4 topics by using both NMF and LDA method separately.

The outcome 4 topics by NMF is as follows:



```
Top 10 words for topic #0:
['goooo', 'shit', 'omg', 'holi', 'hi', 'pogger', 'intro', 'pagman', 'live', 'pogu']


Top 10 words for topic #1:
['away', 'scare', 'did', 'choic', 'aim', 'bark', 'gonna', 'pausechamp', 'hug', 'kekw']


Top 10 words for topic #2:
['watch', 'say', 'look', 'bro', 'car', 'real', 'chat', 'ur', 'god', 'yo']


Top 10 words for topic #3:
['im', 'clap', 'watchtim', 'chat', 'open', 'mic', 'nice', 'car', 'bad', 'omegalul']
```

Based on explanation of different unknown words we provided in the previous section, we can say:

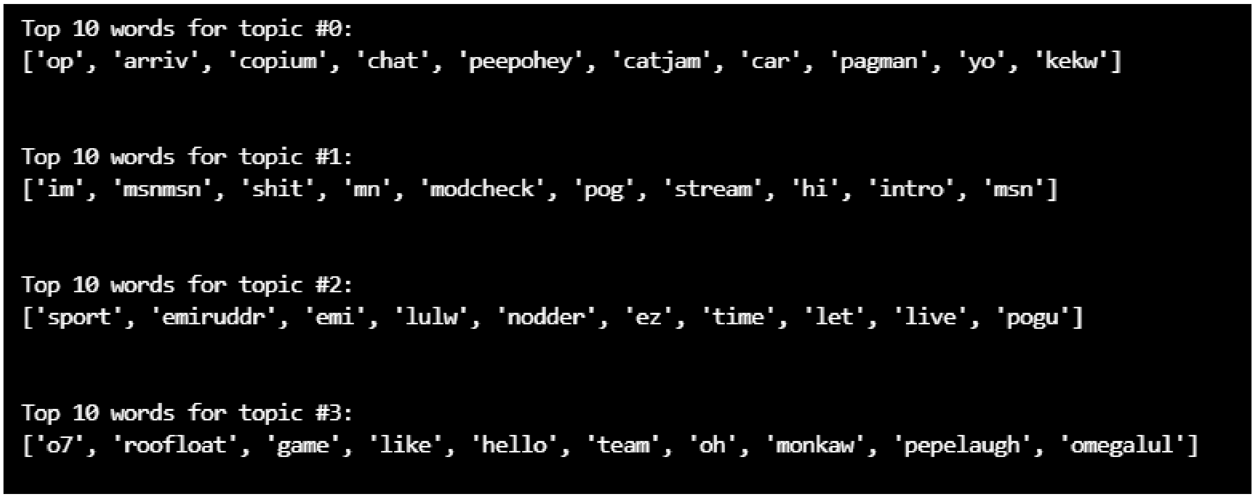
Topic#0 is related to surprise.

Topic#1 is related to laughter as well as scare

Topic#2 is related to car related games

Topic#3 is related to appreciation of playing wonderful, or driving a car nicely

The 4 topics provided by LDA are as follows:



```
Top 10 words for topic #0:
['op', 'arriv', 'copium', 'chat', 'peepohey', 'catjam', 'car', 'pagman', 'yo', 'kekw']


Top 10 words for topic #1:
['im', 'msnmsn', 'shit', 'mn', 'modcheck', 'pog', 'stream', 'hi', 'intro', 'msn']


Top 10 words for topic #2:
['sport', 'emiruddr', 'emi', 'lulw', 'nodder', 'ez', 'time', 'let', 'live', 'pogu']


Top 10 words for topic #3:
['o7', 'roofloat', 'game', 'like', 'hello', 'team', 'oh', 'monkaw', 'pepelaugh', 'omegalul']
```

Topic#0 is difficult to categorize. Car and laughter might be the focus here.

Topic#1 is related to hi, hello. As we described earlier, a huge portion of chat is filled up with hi.

Topic#2 is related to surprise. However, it is hard to categorize.

Topic#3 is related to laughter.

According to the analysis, NMF provided more clearer and distinguishable topics. It should be noted here that, we did not remove “hi” from analysis to show that “intro” type chat can be a category by itself, it has that much huge portion in chatbox.

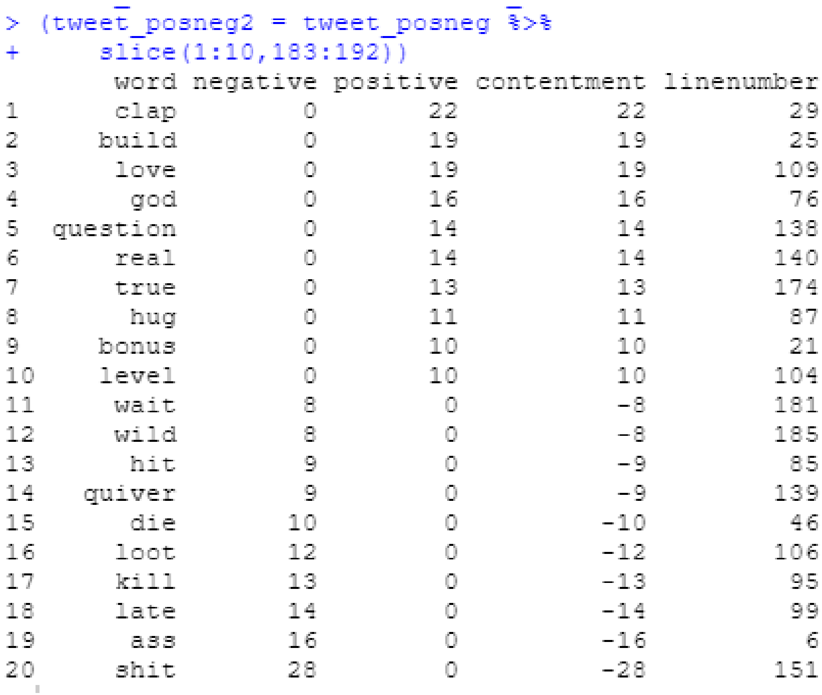
We wanted to go deeper with NMF analysis. By digging deeper, we have found that 66% of chat are related to NMF topic#0 that means surprise. The similar result we found by analyzing top 10 words. Here is an example code, details are in Appendix.

```
topic_values2 = nmf.transform(doc_term_matrix2)

chat_data['topic2'] = topic_values2.argmax(axis=1)


print('Proportion of topic 0' + str(len(chat_data[chat_data['topic2'] == 0]) / len(chat_data['topic2'])))
```

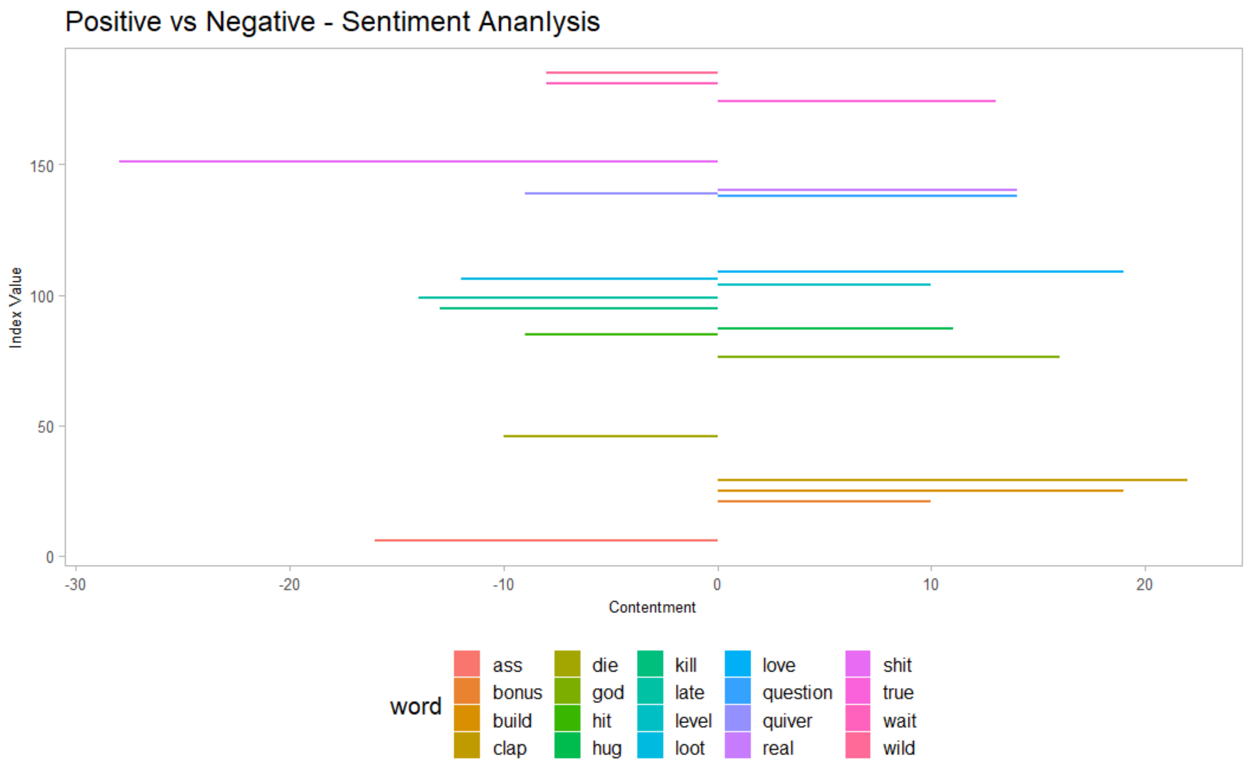
Sentiment Analysis:

We used NRC dictionary in R to perform this.

This is our result for positive vs negative emotions. We took a total of 20 words.

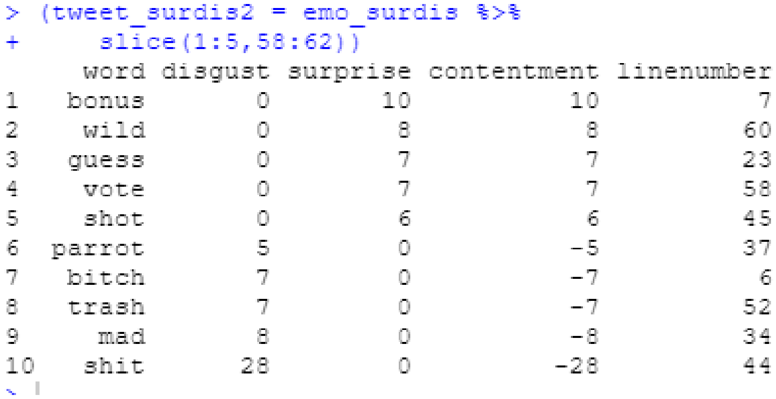


```
> (tweet_posneg2 = tweet_posneg %>%
+      slice(1:10,183:192))
       word negative positive contentment linenumber
1      clap        0       22          22         29
2     build        0       19          19         25
3      love        0       19          19        109
4       god        0       16          16         76
5  question        0       14          14        138
6      real        0       14          14        140
7      true        0       13          13        174
8       hug        0       11          11         87
9     bonus        0       10          10         21
10    level        0       10          10        104
11     wait        8        0          -8        181
12     wild        8        0          -8        185
13      hit        9        0          -9         85
14   quiver        9        0          -9        139
15      die       10        0         -10         46
16     loot       12        0         -12        106
17     kill       13        0         -13         95
18     late       14        0         -14         99
19      ass       16        0         -16          6
20     shit       28        0         -28        151
```

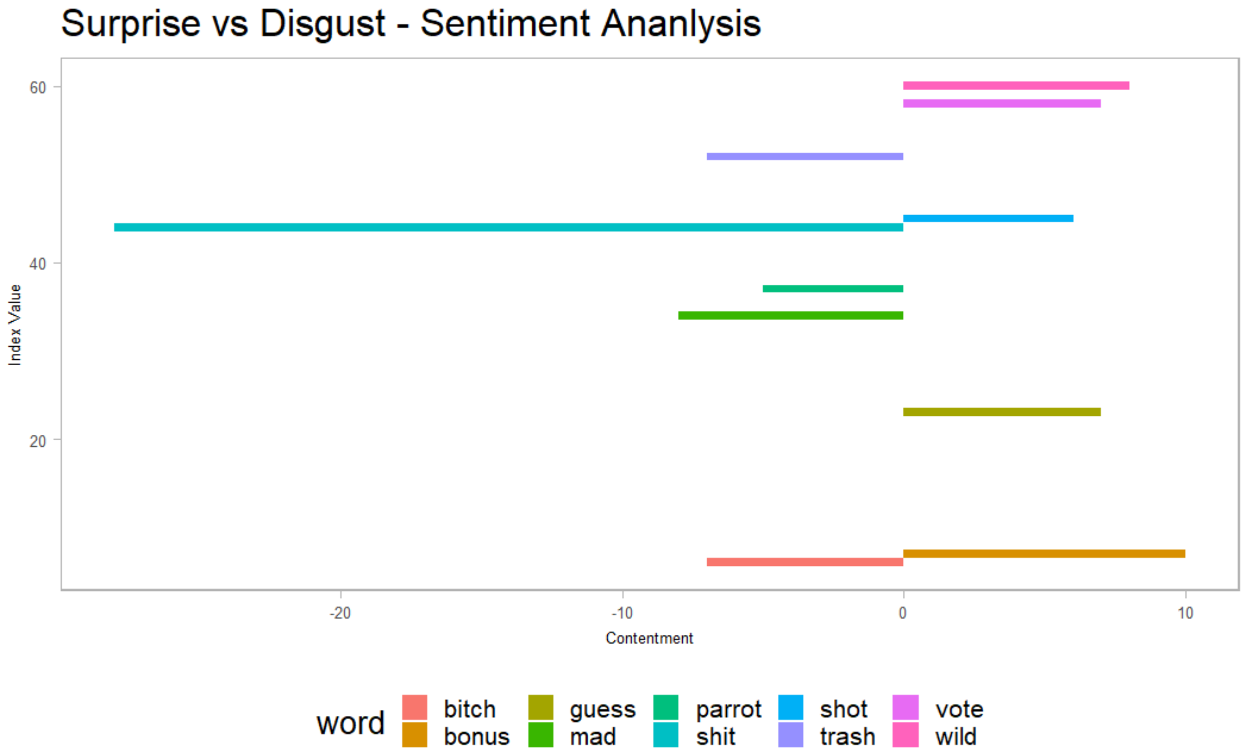


As we can see, it is hard to analyze the sentiment here since “loot” and “kill” were used not from negative emotion. These are emotions against gamer’s enemy. However, we can see clap, build, and love, and shit outperformed others noticeably.

Since surprise emo outperformed other in top words list, we wanted to see surprise vs disgust analysis. In this case, we took a 5 words from each emotion since after observing the number of words and their scores for emotions, we realized 5 words are enough.

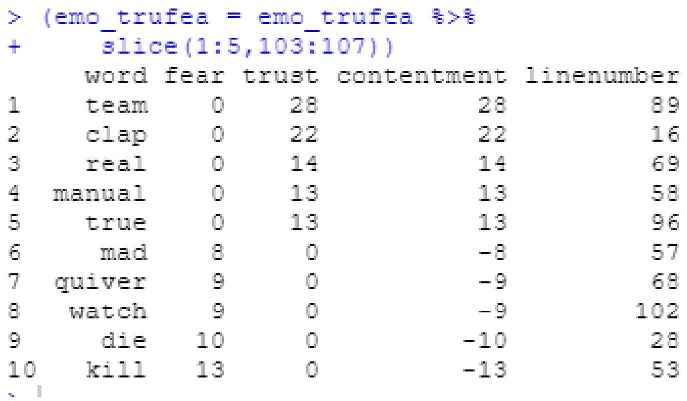


```
> (tweet_surdis2 = emo_surdis %>%
+      slice(1:5,58:62))
     word disgust surprise contentment linenumber
1   bonus       0       10          10          7
2    wild       0        8           8         60
3   guess       0        7           7         23
4    vote       0        7           7         58
5    shot       0        6           6         45
6  parrot       5        0          -5         37
7   bitch       7        0          -7          6
8   trash       7        0          -7         52
9     mad       8        0          -8         34
10   shit      28        0         -28         44
>
```

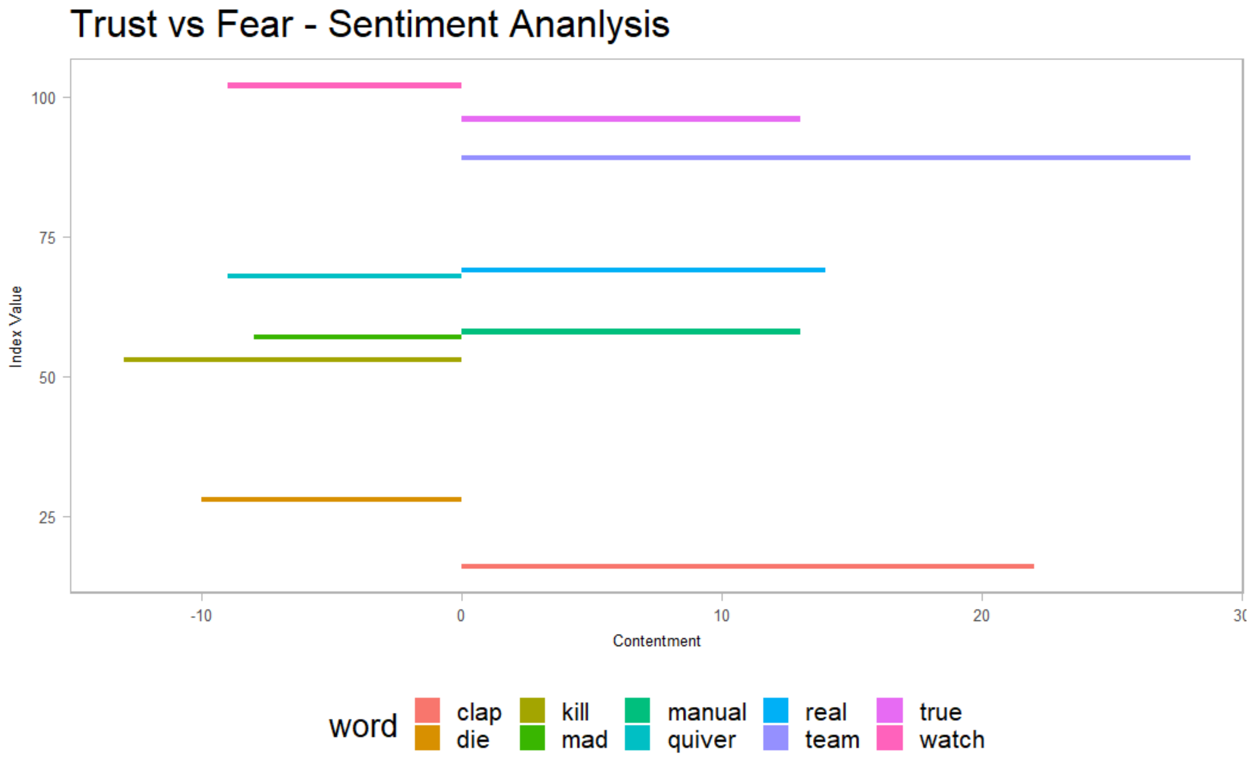


Since the people use many sign languages, as expected, the analysis could not give much insights. However, shit comes again here. It shows people got involved with the gamers.

Finally, we wanted to see how much confidence and trust these gamers have gained over time. Here is the outcome of trust vs fear emotion analysis. In this case, again, we took a 5 words from each emotion since after observing the number of words and their scores for emotions, we realized 5 words are enough.



```
> (emo_trufea = emo_trufea %>%
+     slice(1:5,103:107))
     word fear trust contentment linenumber
1    team    0    28          28         89
2    clap    0    22          22         16
3    real    0    14          14         69
4  manual    0    13          13         58
5    true    0    13          13         96
6     mad    8     0          -8         57
7  quiver    9     0          -9         68
8   watch    9     0          -9        102
9     die   10     0         -10         28
10   kill   13     0         -13         53
```

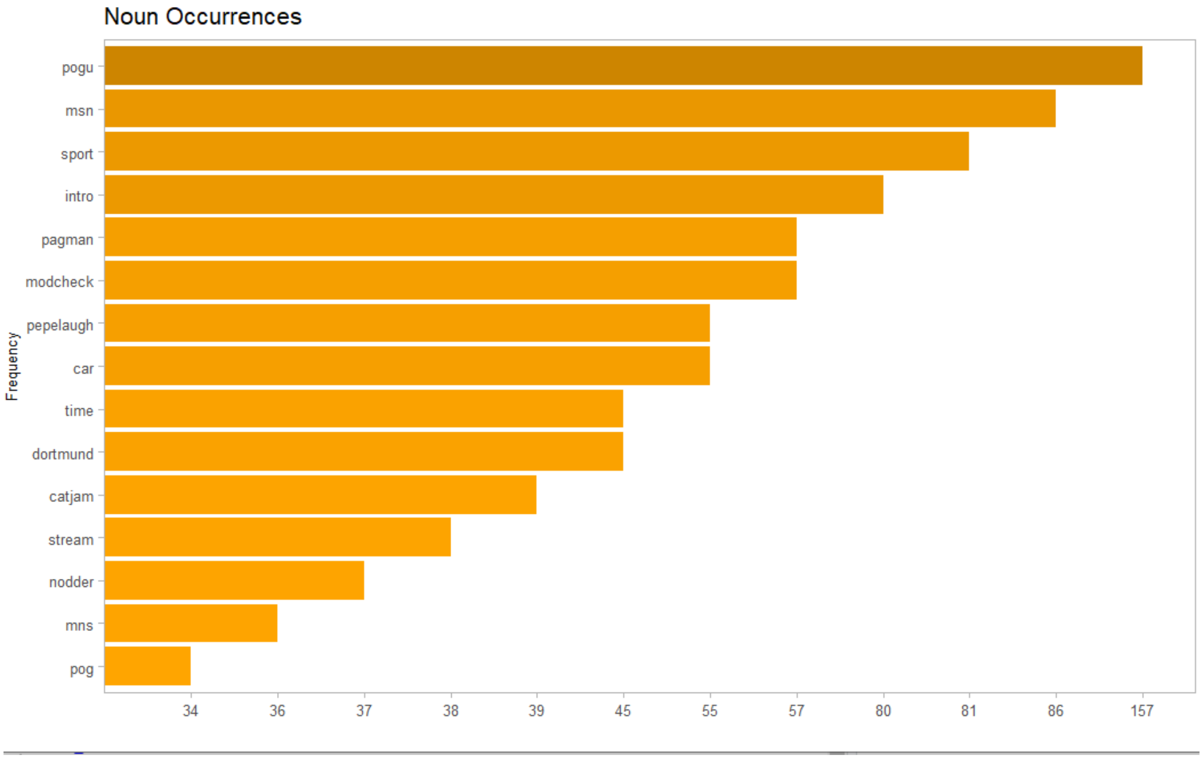


This plot shows trust clearly wins. Since team is the outperforming word, we can say people love to see team effort. During scrapping long hours' videos (say 33hours long), we have noticed people chatted about this that multiple players are playing behind one channel. This might be a very useful insight for streamers who want to be successful.

Classification:

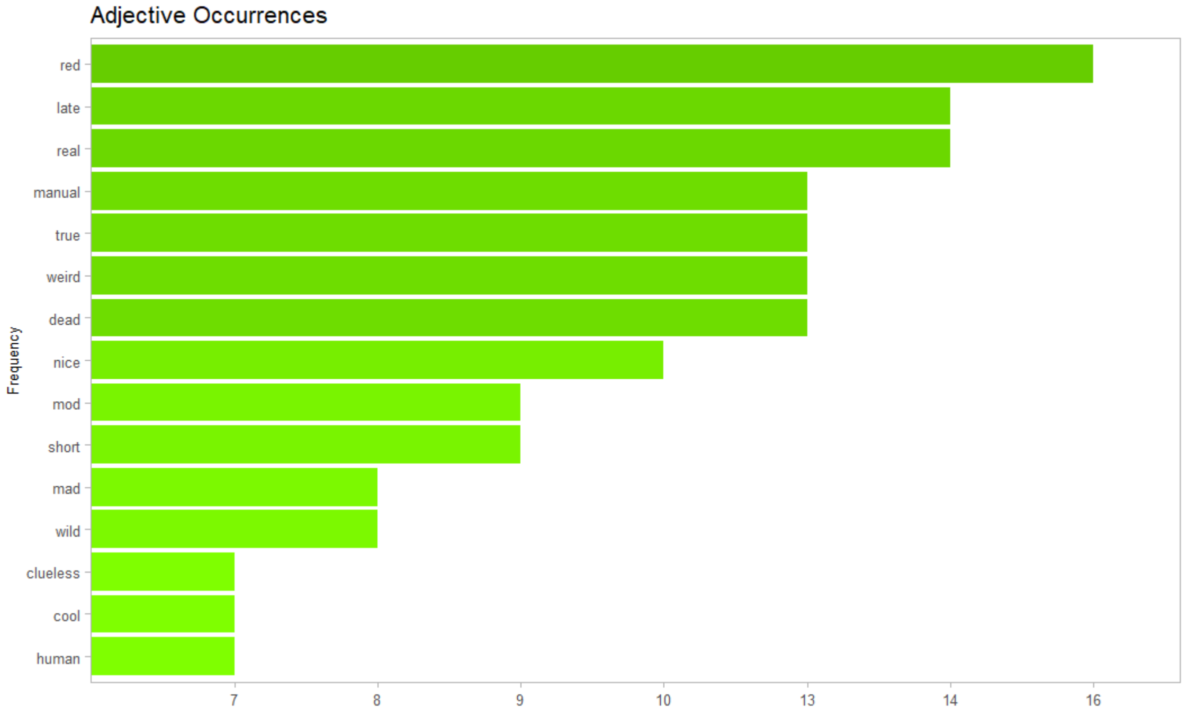
In R, we used ud_model to perform this.

Here are the most frequently used nouns:

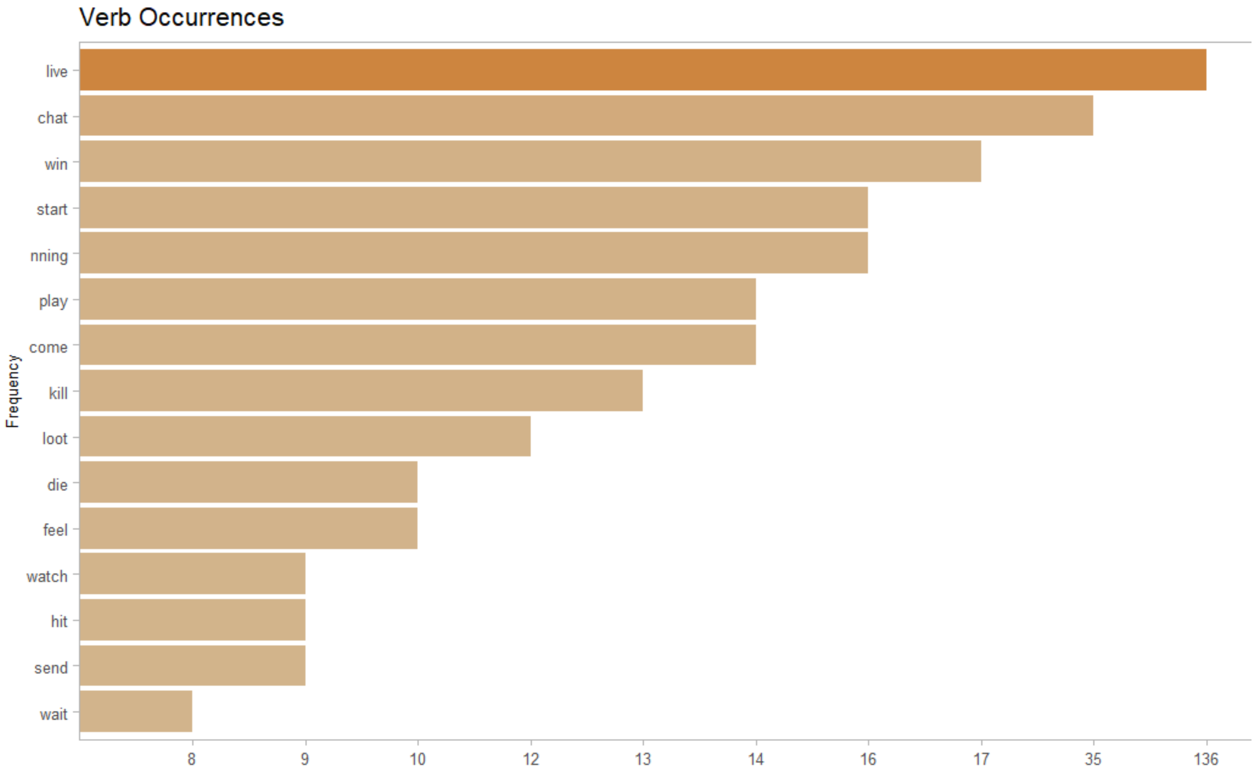


We can see, in many cases they came from top 10 words which are basically emoticon.

Here are the most frequently used adjectives:



And, here are the most frequently used verbs:



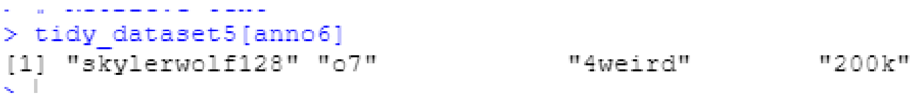
This is an important graph since it tells us action words. Running and kills are 2 words which are among the top verbs. So, we can guess racing games and fighting games excites people mostly. Also they want to see streamers win.

Named Entity Recognition:

Since people do not talk about any common name of people or place or organization, this analysis is not relevant here. In R, we used NLP, OpenNLP, and stringr package to perform this.

For "person" there is no output. As we guessed, NER analysis couldn't produce any result.

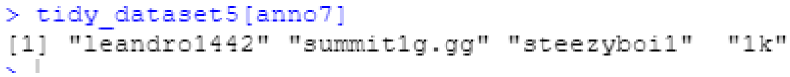
For location, here is the output:



```
> tidy_dataset5[anno6]
[1] "skylerwolf128" "o7"              "4weird"          "200k"
>
```

As expected, all of them are irrelevant.

For organization, here is the output:



```
> tidy_dataset5[anno7]
[1] "leandro1442" "summit1g.gg" "steezyboi1"   "1k"
>
```

Again, these are usernames in some cases, and not helpful in analysis.

References:

1. https://www.grandviewresearch.com/industry-analysis/video-streaming-market

2. https://www.streamingmedia.com/Articles/News/Online-Video-News/On-Demand-Viewing-Growing-Much-Faster-Than-Live-Says-Conviva-133418.aspx

3. https://findstack.com/live-streaming-statistics/

GitHub Links:

1. Link for all codes: https://github.com/msis5193-pds1-2021fall-lab/Game-live-streaming-project/blob/main/codes_all_in_one.txt

2. Link for all text from chats: https://github.com/msis5193-pds1-2021fall-lab/Game-live-streaming-project/blob/main/final_chatbox_df.txt

Appendix:

```
# Game live streaming project

## Chase(Cao) Heng Noreen Rafae

## Include both python and R codes


## python prerequisite

from selenium import webdriver

from selenium.webdriver.ie.options import ElementScrollBehavior

from selenium.webdriver.support import expected_conditions as EC

import selenium.webdriver.support.ui as ui

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.firefox.service import Service

from selenium.webdriver.common.by import By

import pandas as pd

from selenium.webdriver.support.ui import Select

import numpy as np

import os

import matplotlib.pyplot as plt

from sklearn.model_selection import KFold


service=Service(r'C:\Users\...\Documents\geckodriver.exe')

driver = webdriver.Firefox(service=service)



# Scrape + transform numerical data of each of top 20 channels
```

```
## Python
@author: Noreen and Yi Heng
#1.Channel CRITICALROLE
url = 'https://twitchtracker.com/criticalrole/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name1= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name1.text)
list_channel_name1 = channel_name1.text.split(".")

# Scrape name of games by loop
game_name1= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name1.text)
list_game_name1=game_name1.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/criticalrole/statistics'
driver.get(url)
driver.maximize_window()

total_followers1= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers1.text)
list_total_followers1 = total_followers1.text.split(".")
```

```
#scrape average viewers
avg_viewers1=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers1.text)
list_avg_viewers1 = avg_viewers1.text.split(".")

#scrap total views
total_views1=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views1.text)
list_total_views1 = total_views1.text.split()

#scrap hours streamed
hours_streamed1=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed1.text)
list_hours_streamed1 = hours_streamed1.text.split()

#scrap hours watched
hours_watched1=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched1.text)
list_hours_watched1 = hours_watched1.text.split()

#Modify the original data
list_total_followers1[0]=893000
```

```
list_avg_viewers1[0]=20401

list_total_views1[0]=26900000

list_hours_streamed1[0]=1438

list_hours_watched1[0]=29300000


#Unify the list

list1=list_channel_name1+list_game_name1+list_total_followers1+list_avg_viewers1+list_total_views1+list_hours_streamed1+list_hours_watched1



#2.Channel XQCOW

url = 'https://twitchtracker.com/xqcow/games'

driver.get(url)

driver.maximize_window()


#name of channel

channel_name2= driver.find_element(By.XPATH,'//*[@id="app-title"]')

print(channel_name2.text)

list_channel_name2 = channel_name2.text.split(".")


# Scrape name of games by loop

game_name2= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name2.text)

list_game_name2=game_name2.text.split(".")


#scrap total viewers
```

```
url = 'https://twitchtracker.com/xqcow/statistics'
driver.get(url)
driver.maximize_window()

total_followers2=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers2.text)
list_total_followers2 = total_followers2.text.split()

#scrape average viewers
avg_viewers2=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers2.text)
list_avg_viewers2 = avg_viewers2.text.split(".")

#scrap total views
total_views2=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views2.text)
list_total_views2 = total_views2.text.split()

#scrap hours streamed
hours_streamed2=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed2.text)
list_hours_streamed2 = hours_streamed2.text.split()
```

```
#scrap hours watched
hours_watched2= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched2.text)
list_hours_watched2 = hours_watched2.text.split()

#Modify the original data
list_total_followers2[0]=9360000
list_avg_viewers2[0]=30161
list_total_views2[0]=430000000
list_hours_streamed2[0]=15645
list_hours_watched2[0]=472000000

#Unify the list
list2=list_channel_name2+list_game_name2+list_total_followers2+list_avg_viewers2+list_total_views2+ list_hours_streamed2+list_hours_watched2



#3.Channel RANBOOLIVE
url = 'https://twitchtracker.com/ranboolive/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name3= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name3.text)
```

```
list_channel_name3 = channel_name3.text.split(".")

# Scrape name of games by loop
game_name3= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name3.text)
list_game_name3=game_name3.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/ranboolive/statistics'
driver.get(url)
driver.maximize_window()

total_followers3=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers3.text)
list_total_followers3 = total_followers3.text.split()

#scrape average viewers
avg_viewers3=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers3.text)
list_avg_viewers3 = avg_viewers3.text.split(".")

#scrap total views
total_views3=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
```

```
print(total_views3.text)
list_total_views3 = total_views3.text.split()

#scrap hours streamed
hours_streamed3=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed3.text)
list_hours_streamed3 = hours_streamed3.text.split()

#scrap hours watched
hours_watched3=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched3.text)
list_hours_watched3 = hours_watched3.text.split()

#Modify the original data
list_total_followers3[0]=3880000
list_avg_viewers3[0]=62975
list_total_views3[0]=58500000
list_hours_streamed3[0]=1039
list_hours_watched3[0]=65500000

#Unify the list
list3=list_channel_name3+list_game_name3+list_total_followers3+list_avg_viewers3+list_total_views3+
list_hours_streamed3+list_hours_watched3
```

```
#4.Channel GAULES
url = 'https://twitchtracker.com/gaules/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name4= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name4.text)
list_channel_name4 = channel_name4.text.split(".")

# Scrape name of games by loop
game_name4= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name4.text)
list_game_name4=game_name4.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/gaules/statistics'
driver.get(url)
driver.maximize_window()

total_followers4=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers4.text)
list_total_followers4 = total_followers4.text.split()
```

```
#scrape average viewers
avg_viewers4= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers4.text)
list_avg_viewers4 = avg_viewers4.text.split(".")

#scrap total views
total_views4= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views4.text)
list_total_views4 = total_views4.text.split()

#scrap hours streamed
hours_streamed4= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed4.text)
list_hours_streamed4 = hours_streamed4.text.split()

#scrap hours watched
hours_watched4= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched4.text)
list_hours_watched4 = hours_watched4.text.split()

#Modify the original data
list_total_followers4[0]=2980000
list_avg_viewers4[0]=10865
```

```
list_total_views4[0]=346000000
list_hours_streamed4[0]=30069
list_hours_watched4[0]=327000000

#Unify the list
list4=list_channel_name4+list_game_name4+list_total_followers4+list_avg_viewers4+list_total_views4+list_hours_streamed4+list_hours_watched4



#5.Channel IBAI
url = 'https://twitchtracker.com/ibai/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name5= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name5.text)
list_channel_name5 = channel_name5.text.split(".")

# Scrape name of games by loop
game_name5= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name5.text)
list_game_name5=game_name5.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/ibai/statistics'
```

```
driver.get(url)
driver.maximize_window()

total_followers5= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers5.text)
list_total_followers5 = total_followers5.text.split()

#scrape average viewers
avg_viewers5= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers5.text)
list_avg_viewers5 = avg_viewers5.text.split(".")

#scrap total views
total_views5= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views5.text)
list_total_views5 = total_views5.text.split()

#scrap hours streamed
hours_streamed5= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed5.text)
list_hours_streamed5 = hours_streamed5.text.split()

#scrap hours watched
```

```
hours_watched5=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched5.text)
list_hours_watched5 = hours_watched5.text.split()

#Modify the original data
list_total_followers5[0]=8200000
list_avg_viewers5[0]=47090
list_total_views5[0]=269000000
list_hours_streamed5[0]=3323
list_hours_watched5[0]=156000000

#Unify the list
list5=list_channel_name5+list_game_name5+list_total_followers5+list_avg_viewers5+list_total_views5+
list_hours_streamed5+list_hours_watched5



#6.Channel NICKMERCS
url = 'https://twitchtracker.com/nickmercs/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name6= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name6.text)
list_channel_name6 = channel_name6.text.split(".")
```

```
# Scrape name of games by loop
game_name6= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name6.text)
list_game_name6=game_name6.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/nickmercs/statistics'
driver.get(url)
driver.maximize_window()

total_followers6= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers6.text)
list_total_followers6 = total_followers6.text.split()

#scrape average viewers
avg_viewers6= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers6.text)
list_avg_viewers6 = avg_viewers6.text.split(".")

#scrap total views
total_views6= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views6.text)
```

```
list_total_views6 = total_views6.text.split()

#scrap hours streamed
hours_streamed6=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed6.text)
list_hours_streamed6 = hours_streamed6.text.split()

#scrap hours watched
hours_watched6=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched6.text)
list_hours_watched6 = hours_watched6.text.split()

#Modify the original data
list_total_followers6[0]=6190000
list_avg_viewers6[0]=22352
list_total_views6[0]=201000000
list_hours_streamed6[0]=10429
list_hours_watched6[0]=233000000

#Unify the list
list6=list_channel_name6+list_game_name6+list_total_followers6+list_avg_viewers6+list_total_views6+
list_hours_streamed6+list_hours_watched6
```

```
#7.Channel 加藤純一です
url = 'https://twitchtracker.com/kato_junichi0817/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name7= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name7.text)
list_channel_name7 = channel_name7.text.split(".")

# Scrape name of games by loop
game_name7= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name7.text)
list_game_name7=game_name7.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/kato_junichi0817/statistics'
driver.get(url)
driver.maximize_window()

total_followers7=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers7.text)
list_total_followers7 = total_followers7.text.split()

#scrape average viewers
```

```
avg_viewers7=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers7.text)
list_avg_viewers7 = avg_viewers7.text.split(".")


#scrap total views
total_views7=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views7.text)
list_total_views7 = total_views7.text.split()


#scrap hours streamed
hours_streamed7=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed7.text)
list_hours_streamed7 = hours_streamed7.text.split()


#scrap hours watched
hours_watched7=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched7.text)
list_hours_watched7 = hours_watched7.text.split()


#Modify the original data
list_total_followers7[0]=457000
list_avg_viewers7[0]=22634
list_total_views7[0]=2640000
```

```
list_hours_streamed7[0]=1665
list_hours_watched7[0]=37700000

#Unify the list
list7=list_channel_name7+list_game_name7+list_total_followers7+list_avg_viewers7+list_total_views7+list_hours_streamed7+list_hours_watched7


#8.Channel HASANABI
url = 'https://twitchtracker.com/hasanabi/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name8= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name8.text)
list_channel_name8 = channel_name8.text.split(".")

# Scrape name of games by loop
game_name8= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name8.text)
list_game_name8=game_name8.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/hasanabi/statistics'
driver.get(url)
driver.maximize_window()

total_followers8=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers8.text)
list_total_followers8 = total_followers8.text.split()

#scrape average viewers
avg_viewers8=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers8.text)
list_avg_viewers8 = avg_viewers8.text.split(".")

#scrap total views
total_views8=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views8.text)
list_total_views8 = total_views8.text.split()

#scrap hours streamed
hours_streamed8=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed8.text)
list_hours_streamed8 = hours_streamed8.text.split()

#scrap hours watched
```

```
hours_watched8= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched8.text)
list_hours_watched8 = hours_watched8.text.split()

#Modify the original data
list_total_followers8[0]=1570000
list_avg_viewers8[0]=13742
list_total_views8[0]=88400000
list_hours_streamed8[0]=9355
list_hours_watched8[0]=129000000

#Unify the list
list8=list_channel_name8+list_game_name8+list_total_followers8+list_avg_viewers8+list_total_views8+list_hours_streamed8+list_hours_watched8



#9.Channel AURONPLAY
url = 'https://twitchtracker.com/auronplay/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name9= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name9.text)
list_channel_name9 = channel_name9.text.split(".")
```

```
# Scrape name of games by loop

game_name9= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name9.text)

list_game_name9=game_name9.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/auronplay/statistics'

driver.get(url)

driver.maximize_window()


total_followers9=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')

print(total_followers9.text)

list_total_followers9 = total_followers9.text.split()


#scrape average viewers

avg_viewers9=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')

print(avg_viewers9.text)

list_avg_viewers9 = avg_viewers9.text.split(".")


#scrap total views

total_views9=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')

print(total_views9.text)
```

```
list_total_views9 = total_views9.text.split()


#scrap hours streamed
hours_streamed9= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed9.text)
list_hours_streamed9 = hours_streamed9.text.split()


#scrap hours watched
hours_watched9= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched9.text)
list_hours_watched9 = hours_watched9.text.split()


#Modify the original data
list_total_followers9[0]=10500000
list_avg_viewers9[0]=89701
list_total_views9[0]=197000000
list_hours_streamed9[0]=1930
list_hours_watched9[0]=173000000


#Unify the list
list9=list_channel_name9+list_game_name9+list_total_followers9+list_avg_viewers9+list_total_views9+list_hours_streamed9+list_hours_watched9


#10.Channel MONTANABLACK88
```

```
url = 'https://twitchtracker.com/montanablack88/games'

driver.get(url)

driver.maximize_window()


#name of channel

channel_name10= driver.find_element(By.XPATH,'//*[@id="app-title"]')

print(channel_name10.text)

list_channel_name10 = channel_name10.text.split(".")


# Scrape name of games by loop

game_name10= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name10.text)

list_game_name10=game_name10.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/montanablack88/statistics'

driver.get(url)

driver.maximize_window()


total_followers10=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')

print(total_followers10.text)

list_total_followers10 = total_followers10.text.split()


#scrape average viewers
```

```
avg_viewers10=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers10.text)
list_avg_viewers10 = avg_viewers10.text.split(".")

#scrap total views
total_views10=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views10.text)
list_total_views10 = total_views10.text.split()

#scrap hours streamed
hours_streamed10=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed10.text)
list_hours_streamed10 = hours_streamed10.text.split()

#scrap hours watched
hours_watched10=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched10.text)
list_hours_watched10 = hours_watched10.text.split()

#Modify the original data
list_total_followers10[0]=4180000
list_avg_viewers10[0]=27705
list_total_views10[0]=133000000
```

```
list_hours_streamed10[0]=4617

list_hours_watched10[0]=1280000


#Unify the list

list10=list_channel_name10+list_game_name10+list_total_followers10+list_avg_viewers10+list_total_views10+list_hours_streamed10+list_hours_watched10


#11.Channel CASTRO_1021

url = 'https://twitchtracker.com/castro_1021/games'

driver.get(url)

driver.maximize_window()


#name of channel

channel_name11= driver.find_element(By.XPATH,'//*[@id="app-title"]')

print(channel_name11.text)

list_channel_name11 = channel_name11.text.split(".")


# Scrape name of games by loop

game_name11= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name11.text)

list_game_name11=game_name11.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/castro_1021/statistics'

driver.get(url)

driver.maximize_window()
```

```
total_followers11=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers11.text)
list_total_followers11 = total_followers11.text.split()

#scrape average viewers
avg_viewers11=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers11.text)
list_avg_viewers11 = avg_viewers11.text.split(".")

#scrap total views
total_views11=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views11.text)
list_total_views11 = total_views11.text.split()

#scrap hours streamed
hours_streamed11=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed11.text)
list_hours_streamed11 = hours_streamed11.text.split()

#scrap hours watched
hours_watched11=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
```

```
print(hours_watched11.text)
list_hours_watched11 = hours_watched11.text.split()

#Modify the original data
list_total_followers11[0]=3190000
list_avg_viewers11[0]=16093
list_total_views11[0]=123000000
list_hours_streamed11[0]=7493
list_hours_watched11[0]=121000000

#Unify the list
list11=list_channel_name11+list_game_name11+list_total_followers11+list_avg_viewers11+list_total_views11+list_hours_streamed11+list_hours_watched11

#12.Channel CASIMITO
url = 'https://twitchtracker.com/casimito/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name12= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name12.text)
list_channel_name12 = channel_name12.text.split(".")

# Scrape name of games by loop
game_name12= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
```

```
print(game_name12.text)

list_game_name12=game_name12.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/casimito/statistics'

driver.get(url)

driver.maximize_window()


total_followers12=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')

print(total_followers12.text)

list_total_followers12 = total_followers12.text.split()


#scrape average viewers

avg_viewers12=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')

print(avg_viewers12.text)

list_avg_viewers12 = avg_viewers12.text.split(".")


#scrap total views

total_views12=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')

print(total_views12.text)

list_total_views12 = total_views12.text.split()


#scrap hours streamed
```

```
hours_streamed12= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed12.text)
list_hours_streamed12 = hours_streamed12.text.split()

#scrap hours watched
hours_watched12= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched12.text)
list_hours_watched12 = hours_watched12.text.split()

#Modify the original data
list_total_followers12[0]=899000
list_avg_viewers12[0]=8478
list_total_views12[0]=18100000
list_hours_streamed12[0]=3093
list_hours_watched12[0]=26200000

#Unify the list
list12=list_channel_name12+list_game_name12+list_total_followers12+list_avg_viewers12+list_total_views12+list_hours_streamed12+list_hours_watched12


#13.Channel SYKKUNO
url = 'https://twitchtracker.com/sykkuno/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name13= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name13.text)
list_channel_name13 = channel_name13.text.split(".")

# Scrape name of games by loop
game_name13= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name13.text)
list_game_name13=game_name13.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/sykkuno/statistics'
driver.get(url)
driver.maximize_window()

total_followers13= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers13.text)
list_total_followers13 = total_followers13.text.split()

#scrape average viewers
avg_viewers13= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers13.text)
```

```
list_avg_viewers13 = avg_viewers13.text.split(".")


#scrap total views

total_views13=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')

print(total_views13.text)

list_total_views13 = total_views13.text.split()


#scrap hours streamed

hours_streamed13=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')

print(hours_streamed13.text)

list_hours_streamed13 = hours_streamed13.text.split()


#scrap hours watched

hours_watched13=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')

print(hours_watched13.text)

list_hours_watched13 = hours_watched13.text.split()


#Modify the original data

list_total_followers13[0]=3790000

list_avg_viewers13[0]=19324

list_total_views13[0]=84000000

list_hours_streamed13[0]=3982

list_hours_watched13[0]=76900000
```

```
#Unify the list
list13=list_channel_name13+list_game_name13+list_total_followers13+list_avg_viewers13+list_total_views13+list_hours_streamed13+list_hours_watched13


#14.Channel SUMMIT1G
url = 'https://twitchtracker.com/summit1g/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name14= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name14.text)
list_channel_name14 = channel_name14.text.split(".")

# Scrape name of games by loop
game_name14= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name14.text)
list_game_name14=game_name14.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/summit1g/statistics'
driver.get(url)
driver.maximize_window()
```

```
total_followers14=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers14.text)
list_total_followers14 = total_followers14.text.split()

#scrape average viewers
avg_viewers14=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers14.text)
list_avg_viewers14 = avg_viewers14.text.split(".")

#scrap total views
total_views14=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views14.text)
list_total_views14 = total_views14.text.split()

#scrap hours streamed
hours_streamed14=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed14.text)
list_hours_streamed14 = hours_streamed14.text.split()

#scrap hours watched
hours_watched14=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched14.text)
```

```
list_hours_watched14 = hours_watched14.text.split()

#Modify the original data
list_total_followers14[0]=6000000
list_avg_viewers14[0]=23684
list_total_views14[0]=478000000
list_hours_streamed14[0]=16113
list_hours_watched14[0]=382000000

#Unify the list
list14=list_channel_name14+list_game_name14+list_total_followers14+list_avg_viewers14+list_total_views14+list_hours_streamed14+list_hours_watched14



#15.Channel XROHAT
url = 'https://twitchtracker.com/xrohat/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name15= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name15.text)
list_channel_name15 = channel_name15.text.split(".")

# Scrape name of games by loop
game_name15= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
```

```
print(game_name15.text)
list_game_name15=game_name15.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/xrohat/statistics'
driver.get(url)
driver.maximize_window()

total_followers15= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers15.text)
list_total_followers15 = total_followers15.text.split()

#scrape average viewers
avg_viewers15= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers15.text)
list_avg_viewers15 = avg_viewers15.text.split(".")

#scrap total views
total_views15= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views15.text)
list_total_views15 = total_views15.text.split()

#scrap hours streamed
```

```
hours_streamed15= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed15.text)
list_hours_streamed15 = hours_streamed15.text.split()

#scrap hours watched
hours_watched15= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched15.text)
list_hours_watched15 = hours_watched15.text.split()

#Modify the original data
list_total_followers15[0]=300000
list_avg_viewers15[0]=3250
list_total_views15[0]=6920000
list_hours_streamed15[0]=2680
list_hours_watched15[0]=8710000

#Unify the list
list15=list_channel_name15+list_game_name15+list_total_followers15+list_avg_viewers15+list_total_views15+list_hours_streamed15+list_hours_watched15



#16.Channel TRAINWRECKSTV
url = 'https://twitchtracker.com/trainwreckstv/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name16= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name16.text)
list_channel_name16 = channel_name16.text.split(".")

# Scrape name of games by loop
game_name16= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name16.text)
list_game_name16=game_name16.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/trainwreckstv/statistics'
driver.get(url)
driver.maximize_window()

total_followers16=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers16.text)
list_total_followers16 = total_followers16.text.split()

#scrape average viewers
avg_viewers16=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers16.text)
```

```
list_avg_viewers16 = avg_viewers16.text.split(".")

#scrap total views
total_views16=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views16.text)
list_total_views16 = total_views16.text.split()

#scrap hours streamed
hours_streamed16=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed16.text)
list_hours_streamed16 = hours_streamed16.text.split()

#scrap hours watched
hours_watched16=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched16.text)
list_hours_watched16 = hours_watched16.text.split()

#Modify the original data
list_total_followers16[0]=1590000
list_avg_viewers16[0]=9207
list_total_views16[0]=72800000
list_hours_streamed16[0]=9284
list_hours_watched16[0]=85500000
```

```
#Unify the list

list16=list_channel_name16+list_game_name16+list_total_followers16+list_avg_viewers16+list_total_views16+list_hours_streamed16+list_hours_watched16



#17.Channel MIZKIF

url = 'https://twitchtracker.com/mizkif/games'

driver.get(url)

driver.maximize_window()


#name of channel

channel_name17= driver.find_element(By.XPATH,'//*[@id="app-title"]')

print(channel_name17.text)

list_channel_name17 = channel_name17.text.split(".")


# Scrape name of games by loop

game_name17= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name17.text)

list_game_name17=game_name17.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/mizkif/statistics'

driver.get(url)

driver.maximize_window()
```

```
total_followers17=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers17.text)
list_total_followers17 = total_followers17.text.split()

#scrape average viewers
avg_viewers17=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers17.text)
list_avg_viewers17 = avg_viewers17.text.split(".")

#scrap total views
total_views17=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views17.text)
list_total_views17 = total_views17.text.split()

#scrap hours streamed
hours_streamed17=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed17.text)
list_hours_streamed17 = hours_streamed17.text.split()

#scrap hours watched
hours_watched17=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched17.text)
```

```
list_hours_watched17 = hours_watched17.text.split()

#Modify the original data
list_total_followers17[0]=1600000
list_avg_viewers17[0]=11441
list_total_views17[0]=76100000
list_hours_streamed17[0]=6809
list_hours_watched17[0]=77900000

#Unify the list
list17=list_channel_name17+list_game_name17+list_total_followers17+list_avg_viewers17+list_total_views17+list_hours_streamed17+list_hours_watched17



#18.Channel ADINROSS
url = 'https://twitchtracker.com/adinross/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name18= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name18.text)
list_channel_name18 = channel_name18.text.split(".")

# Scrape name of games by loop
game_name18= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
```

```
print(game_name18.text)
list_game_name18=game_name18.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/adinross/statistics'
driver.get(url)
driver.maximize_window()

total_followers18=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers18.text)
list_total_followers18 = total_followers18.text.split()

#scrape average viewers
avg_viewers18=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers18.text)
list_avg_viewers18 = avg_viewers18.text.split(".")

#scrap total views
total_views18=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views18.text)
list_total_views18 = total_views18.text.split()

#scrap hours streamed
```

```
hours_streamed18= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed18.text)
list_hours_streamed18 = hours_streamed18.text.split()


#scrap hours watched
hours_watched18= driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched18.text)
list_hours_watched18 = hours_watched18.text.split()


#Modify the original data
list_total_followers18[0]=5070000
list_avg_viewers18[0]=15779
list_total_views18[0]=38400000
list_hours_streamed18[0]=3403
list_hours_watched18[0]=53700000


#Unify the list
list18=list_channel_name18+list_game_name18+list_total_followers18+list_avg_viewers18+list_total_views18+list_hours_streamed18+list_hours_watched18



#19.Channel      ADMIRALBAHROO
url = 'https://twitchtracker.com/admiralbahroo/games'
driver.get(url)
driver.maximize_window()

#name of channel
channel_name19= driver.find_element(By.XPATH,'//*[@id="app-title"]')
print(channel_name19.text)
list_channel_name19 = channel_name19.text.split(".")

# Scrape name of games by loop
game_name19= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')
print(game_name19.text)
list_game_name19=game_name19.text.split(".")

#scrap total viewers
url = 'https://twitchtracker.com/admiralbahroo/statistics'
driver.get(url)
driver.maximize_window()

total_followers19=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers19.text)
list_total_followers19 = total_followers19.text.split()

#scrape average viewers
avg_viewers19=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers19.text)
```

```
list_avg_viewers19 = avg_viewers19.text.split(".")

#scrap total views
total_views19=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views19.text)
list_total_views19 = total_views19.text.split()

#scrap hours streamed
hours_streamed19=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed19.text)
list_hours_streamed19 = hours_streamed19.text.split()

#scrap hours watched
hours_watched19=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched19.text)
list_hours_watched19 = hours_watched19.text.split()

#Modify the original data
list_total_followers19[0]=850000
list_avg_viewers19[0]=7331
list_total_views19[0]=78300000
list_hours_streamed19[0]=12778
list_hours_watched19[0]=93700000
```

```
#Unify the list

list19=list_channel_name19+list_game_name19+list_total_followers19+list_avg_viewers19+list_total_views19+list_hours_streamed19+list_hours_watched19



#20.Channel PHILZA

url = 'https://twitchtracker.com/philza/games'

driver.get(url)

driver.maximize_window()


#name of channel

channel_name20= driver.find_element(By.XPATH,'//*[@id="app-title"]')

print(channel_name20.text)

list_channel_name20 = channel_name20.text.split(".")


# Scrape name of games by loop

game_name20= driver.find_element(By.XPATH,'//*[@id="games"]/tbody/tr[1]/td[2]/a')

print(game_name20.text)

list_game_name20=game_name20.text.split(".")


#scrap total viewers

url = 'https://twitchtracker.com/philza/statistics'

driver.get(url)

driver.maximize_window()
```

```
total_followers20=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[1]/div/div[2]/span')
print(total_followers20.text)
list_total_followers20 = total_followers20.text.split()

#scrape average viewers
avg_viewers20=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[5]/div/div[2]/span')
print(avg_viewers20.text)
list_avg_viewers20 = avg_viewers20.text.split(".")

#scrap total views
total_views20=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[2]/div/div[2]/span')
print(total_views20.text)
list_total_views20 = total_views20.text.split()

#scrap hours streamed
hours_streamed20=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[3]/div/div[2]/span')
print(hours_streamed20.text)
list_hours_streamed20 = hours_streamed20.text.split()

#scrap hours watched
hours_watched20=
driver.find_element(By.XPATH,'/html/body/div[3]/div[4]/section[1]/div/div[1]/div[4]/div/div[2]/span')
print(hours_watched20.text)
```

```
list_hours_watched20 = hours_watched20.text.split()

#Modify the original data
list_total_followers20[0]=3640000
list_avg_viewers20[0]=9646
list_total_views20[0]=44600000
list_hours_streamed20[0]=3321
list_hours_watched20[0]=32000000

#Unify the list
list20=list_channel_name20+list_game_name20+list_total_followers20+list_avg_viewers20+list_total_views20+list_hours_streamed20+list_hours_watched20


#Turn all lists into vectors
array1 = np.array(list1)
array2 = np.array(list2)
array3 = np.array(list3)
array4 = np.array(list4)
array5 = np.array(list5)
array6 = np.array(list6)
array7 = np.array(list7)
array8 = np.array(list8)
array9 = np.array(list9)
array10 = np.array(list10)
array11 = np.array(list11)
```

```
array12 = np.array(list12)
array13 = np.array(list13)
array14 = np.array(list14)
array15 = np.array(list15)
array16 = np.array(list16)
array17 = np.array(list17)
array18 = np.array(list18)
array19 = np.array(list19)
array20 = np.array(list20)


#Combine all vectors into one data frame
data =
np.vstack((array1,array2,array3,array4,array5,array6,array7,array8,array9,array10,array11,array12,array1
3,array14,array15,array16,array17,array18,array19,array20))



#scrap subscibers of top 20
url = 'https://twitchtracker.com/subscribers'
driver.get(url)
driver.maximize_window()
subsciber_elem =
driver.find_elements_by_xpath('/html/body/div[3]/div[3]/section[2]/table/tbody/tr/td[5]/span')
subsciber_list = list()
for n in subsciber_elem:
    subsciber_list.append(n.text)
```

```
#Modify the original data

subsciber_list[0: 20] =
[85802,66216,53923,49894,45267,44558,43996,40350,38213,34463,31395,30777,30085,29589,28595,2
8324,24294,22498,22023,21947]

del subsciber_list[-1]



#Turn lists into vector

array21 = np.array(subsciber_list)


#Convert to column vector

array = array21.reshape(-1, 1)


#Merge data frame

data2 = np.hstack((data,array))


#join data into csv file

df = pd.DataFrame(data2,columns=['Channel name','Game name','Total followers','Average
viewers','Total views','Hours streamed','Hours watched','Subscibers'])


#Export CSV file

df.to_csv(r'C:\Users\Yi Heng\Desktop\video_game_streaming.csv',index=False)


# Scrape numerical data of each of recent top 46 popular channels. The statistics provided by the website
will be updated everyday. Finally we scrape it on 12/1/2021

## Python
```

```
@author: Chase(Cao)


url = 'https://twitchtracker.com/channels/viewership/english/personality'
driver.get(url)
driver.maximize_window()
driver.implicitly_wait(5)
# Scrape time_streamed by loop
time_streamed=[]
loop = range(1, 13)
for x in loop:

time_streamed.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[5]/span'%x).text)


loop = range(15, 27)
for x in loop:

time_streamed.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[5]/span'%x).text)


loop = range(29, 41)
for x in loop:

time_streamed.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[5]/span'%x).text)


loop = range(43, 53)
```

```
for x in loop:

time_streamed.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[5]/span'%x).text)

# Scrape avg_viewers by loop
avg_viewers=[]
loop = range(1, 13)
for x in loop:

avg_viewers.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[4]/span'%x).text)

loop = range(15, 27)
for x in loop:

avg_viewers.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[4]/span'%x).text)

loop = range(29, 41)
for x in loop:

avg_viewers.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[4]/span'%x).text)

loop = range(43, 53)
for x in loop:
```

```
avg_viewers.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[4]/span'%x).text)

# Scrape hour_watched by loop
hour_watched=[]
loop = range(1, 13)
for x in loop:

hour_watched.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[7]/span'%x).text)

loop = range(15, 27)
for x in loop:

hour_watched.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[7]/span'%x).text)

loop = range(29, 41)
for x in loop:

hour_watched.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[7]/span'%x).text)

loop = range(43, 53)
for x in loop:
```

```
hour_watched.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[7]/span'%x).text)


# Scrape follower_gain by loop
follower_gain=[]
loop = range(1, 13)
for x in loop:

follower_gain.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[9]/span'%x).text)


loop = range(15, 27)
for x in loop:

follower_gain.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[9]/span'%x).text)


loop = range(29, 41)
for x in loop:

follower_gain.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[9]/span'%x).text)


loop = range(43, 53)
for x in loop:
```

```
follower_gain.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[9]/span'%x).text)

# Scrape total_follower by loop
total_follower=[]
loop = range(1, 13)
for x in loop:

total_follower.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[10]/span'%x).text)

loop = range(15, 27)
for x in loop:

total_follower.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[10]/span'%x).text)

loop = range(29, 41)
for x in loop:

total_follower.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[10]/span'%x).text)

loop = range(43, 53)
for x in loop:
```

```
total_follower.append(driver.find_element(By.XPATH,'/html/body/div[3]/div[2]/div[2]/table/tbody/tr[%d]/td[10]/span'%x).text)

list=list(zip(time_streamed, avg_viewers, hour_watched, follower_gain, total_follower))
raw_numerical_data= pd.DataFrame(list,columns=['time_streamed', 'avg_viewers', 'hour_watched', 'follower_gain', 'total_follower'])
raw_numerical_data.to_csv('D:\Documents\\raw_numerical_data.txt',index=False,sep='\t')

# Transform and analyze the data of recent top 46 popular channels
## R
@author: Chase(Cao)

#install.packages('regclass')
library(regclass)
## Read data
data=read.table("D:\\Documents\\raw_numerical_data.txt",header=TRUE)
## Data transformation
### Transform k to 1000 and M to 1000000
data3k<-as.numeric(sub("K", "e3", data[,3], fixed = TRUE))
data3m<-as.numeric(sub("M", "e6", data[,3], fixed = TRUE))
data3k[is.na(data3k)] <- 0
data3m[is.na(data3m)] <- 0
data[,3]<-data3k+data3m

data4k<-as.numeric(sub("K", "e3", data[,4], fixed = TRUE))
```

```
data4m<-as.numeric(sub("M", "e6", data[,4], fixed = TRUE))
data4k[is.na(data4k)] <- 0
data4m[is.na(data4m)] <- 0
data[,4]<-data4k+data4m

data5k<-as.numeric(sub("K", "e3", data[,5], fixed = TRUE))
data5m<-as.numeric(sub("M", "e6", data[,5], fixed = TRUE))
data5k[is.na(data5k)] <- 0
data5m[is.na(data5m)] <- 0
data[,5]<-data5k+data5m

### Make the data in the second column numerical data.
data[,2]<-as.numeric(sub(",", "", data[,2], fixed = TRUE))

## transformed data
data
plot(data)

## check correlation of data matrix
cor(data[,1:3])

## Model
model1<-lm(total_follower~avg_viewers+hour_watched+time_streamed,data=data)
summary(model1)
VIF(model1)
```

```
model2<-lm(follower_gain~avg_viewers+hour_watched+time_streamed,data=data)
summary(model2)
VIF(model2)

# Scrape chatbox of top 20 channels
## Python

@author: All of us

# Chaneel 1-4
@author: Chase(Cao)

## For channel 1 CRITICALROLE, it required subscription to watch. So skip it.

## For channel 2 XQCOW
## Run the codes below 5 times with different number (from 1 to 5) to get 5 files of chatbox data at 5 different time point.
url1 = 'https://www.twitch.tv/videos/1217149538'
driver.get(url1)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
XQCOW5=[]
loop = range(10, 110) # scrape 100 chats.
for x in loop:
```

```
XQCOW5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
XQCOW5_df= pd.DataFrame(XQCOW5)
XQCOW5_df.to_csv('D:\Documents\XQCOW5_df.txt',index=False,sep='\t')

## For channel 3 RANBOOLIVE
url2 = 'https://www.twitch.tv/videos/1206589824?filter=archives&sort=time'
driver.get(url2)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
RANBOO5=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:

RANBOO5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
RANBOO5_df= pd.DataFrame(RANBOO5)
RANBOO5_df.to_csv('D:\Documents\RANBOO5_df.txt',index=False,sep='\t')



## For channel 4 GAULES (Not in English. Will be removed)
url3 = 'https://www.twitch.tv/videos/1184792471'
driver.get(url3)
# wait the webpage to load for a while
driver.implicitly_wait(100)
```

```
# Scrape text in chatbox by loop
GAULES=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:

GAULES.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(GAULES)


# Channel 5-8
@author: Heng


## For channel 5 IBAI
url12 = 'https://www.twitch.tv/videos/1215770800'  #Just Chatting
driver.get(url12)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
IBAI=[]
loop = range(1,60 ) # scrape  chats.
for x in loop:

IBAI.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(IBAI)


IBAIO4text = [IBAI]
```

```
flatListIBAIO4 = []
for elem in IBAIO4text:
    flatListIBAIO4.extend(elem)
flatListIBAIO4

IBAIO4chatcomb = pd.DataFrame(flatListIBAIO4)
IBAIO4chatcomb.to_csv('C:/Users/Yi Heng/Desktop/IBAI.txt', index = False, sep = '\t')



## For channel 6 NICKMERCS
url13 = 'https://www.twitch.tv/nickmercs/clip/BeautifulLivelyDumplingsDuDudu-ajQgJKZXYc3-CrTk' #Apex Legends
driver.get(url13)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
NICKMERCS=[]
loop = range(2,47 ) # scrape  chats.
for x in loop:

NICKMERCS.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(NICKMERCS)

NICKMERCStext = [NICKMERCS]
```

```
flatListNICKMERCS = []
for elem in NICKMERCStext:
    flatListNICKMERCS.extend(elem)
flatListNICKMERCS

NICKMERCSchatcomb = pd.DataFrame(flatListNICKMERCS)
NICKMERCSchatcomb.to_csv('C:/Users/Yi Heng/Desktop/NICKMERCS.txt', index = False, sep = '\t')



## For channel 7 加藤純一です
url14 = 'https://www.twitch.tv/videos/1213588970'  #ポケダンのインストールをする
driver.get(url14)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
加藤純一です=[]
loop = range(1,60 ) # scrape  chats.
for x in loop:
    加藤純一です.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(加藤純一です)
```

```
加藤純一ですtext = [加藤純一です]


flatList加藤純一です = []
for elem in 加藤純一ですtext:
    flatList加藤純一です.extend(elem)
flatList加藤純一です


加藤純一ですchatcomb = pd.DataFrame(flatList加藤純一です)
加藤純一ですchatcomb.to_csv('C:/Users/Yi Heng/Desktop/加藤純一です.txt', index = False, sep = '\t')


## For channel 8 HasanAbi
#hello 11/25 on 11/26
driver = webdriver.Firefox(executable_path=r'C:\Users\Yi Heng\Documents\geckodriver.exe')
url14 = 'https://www.twitch.tv/videos/1213588970'  #Just Chatting
driver.get(url14)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
HasanAbi=[]
loop = range(1,100 ) # scrape  chats.
for x in loop:

HasanAbi.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(HasanAbi)
```

```
HasanAbitext = [HasanAbi]


flatListHasanAbi = []
for elem in HasanAbitext:
    flatListHasanAbi.extend(elem)
flatListHasanAbi


HasanAbichatcomb = pd.DataFrame(flatListHasanAbi)
HasanAbichatcomb.to_csv('C:/Users/Yi Heng/Desktop/HasanAbi.txt', index = False, sep = '\t')


# Channel 9-12
@author: Noreen
## For channel 10 MONTANABLACK88


montana_url = driver.get('https://www.twitch.tv/videos/1215758353')
driver.implicitly_wait(30)
driver.maximize_window() #wait 30sec for page to fully load


#click start watching button
driver.find_element_by_xpath('/html/body/div[1]/div/div[2]/div[1]/main/div[2]/div[3]/div/div/div[2]/div/div[2]/div/div/div/div/div[5]/div/div[3]/button/div/div').click()
driver.implicitly_wait(30)
 # Scrape text in chatbox by loop
montanablack88=[]
loop = range(56,100) # scrape  chats.
```

```
for x in loop:

montanablack88.append(driver.find_element_by_xpath('/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(montanablack88)

montana2=[]
loop = range(128,160) # scrape chats from different hours.
for x in loop:

montana2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(montana2)

montana3=[]
loop = range(87,105) # scrape chats from different hours.
for x in loop:

montana3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span[1]'%x).text)
print(montana3)

montana4=[]
loop = range(13, 37) # scrape chats from different hours.
for x in loop:

montana4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
```

```
print(montana4)



mon_text = [montanablack88, montana2, montana3, montana4]


flatList_mon = []
for elem in mon_text:
    flatList_mon.extend(elem)
flatList_mon


a_chatcomb = pd.DataFrame(flatList_mon)
a_chatcomb.to_csv('montana.txt', index = False, sep = '\t')



## For Channel 9 AURONPLAY


auronplay_url = 'https://www.twitch.tv/videos/1215601745'
driver.get(auronplay_url)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
auronplay=[]
loop = range(3,30) # scrape  chats.
for x in loop:
```

```
auronplay.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(auronplay)


auronplay2=[]
loop = range(34, 77) # scrape chats from different hours.
for x in loop:


auronplay2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(auronplay2)


auronplay3=[]
loop = range(18, 57) # scrape chats from different hours.
for x in loop:


auronplay3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(auronplay3)


auronplay4=[]
loop = range(9, 40) # scrape chats from different hours.
for x in loop:


auronplay4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(auronplay4)
```

```
a_text = [auronplay, auronplay2, auronplay3, auronplay4]

flatLista = []
for elem in a_text:
    flatLista.extend(elem)
flatLista

a_chatcomb = pd.DataFrame(flatLista)

a_chatcomb.to_csv('auronplay.txt', index = False, sep = '\t')

## For channel 11 CASTRO_1021

castro_url = 'https://www.twitch.tv/videos/1215738441'
driver.get(castro_url)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
castro=[]
loop = range(21,100) # scrape  chats.
for x in loop:

castro.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(castro)
```

```
castro2=[]

loop = range(21, 77) # scrape chats from different hours.

for x in loop:


castro2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)

print(castro2)


castro3=[]

loop = range(16, 80) # scrape chats from different hours.

for x in loop:


castro3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)

print(castro3)


castro4=[]

loop = range(72,150) # scrape chats from different hours.

for x in loop:


castro4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)

print(castro4)


cas_text = [castro4, castro2, castro3, castro4]
```

```
flatList_castro = []
for elem in cas_text:
    flatList_castro.extend(elem)
flatList_castro

c_chatcomb = pd.DataFrame(flatList_castro)

c_chatcomb.to_csv('castro_1021.txt', index = False, sep = '\t')


## For channel 12 CASIMITO

casimito_url = 'https://www.twitch.tv/videos/1215228372'
driver.get(casimito_url)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
casimito=[]
loop = range(30,100) # scrape  chats.
for x in loop:

casimito.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(casimito)

casimito2=[]
```

```
loop = range(51,100) # scrape chats from different hours.
for x in loop:

casimito2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(casimito2)

casimito3=[]
loop = range(89,150) # scrape chats from different hours.
for x in loop:

casimito3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(casimito3)

casimito4=[]
loop = range(12,80) # scrape chats from different hours.
for x in loop:

casimito4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(casimito4)

casimito_text = [casimito, casimito2, casimito3, casimito4]

flatList_casimito = []
for elem in casimito_text:
    flatList_casimito.extend(elem)
flatList_casimito

casimito_chatcomb = pd.DataFrame(flatList_casimito)

casimito_chatcomb.to_csv('casimito.txt', index = False, sep = '\t')


#SCRAP CHATS FOR MIZKIF

miz_url = 'https://www.twitch.tv/videos/1214890407'
driver.get(miz_url)
driver.maximize_window()
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
miz=[]
loop = range(49,100) # scrape  chats.
for x in loop:

miz.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(miz)

miz2=[]
loop = range(16, 77) # scrape chats from different hours.
```

```
for x in loop:

miz2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(miz2)

miz3=[]
loop = range(23,100) # scrape chats from different hours.
for x in loop:

miz3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(miz3)

miz4=[]
loop = range(25,100) # scrape chats from different hours.
for x in loop:

miz4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(miz4)

miz5=[]
loop = range(17, 80) # scrape chats from different hours.
for x in loop:

miz5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
```

```
print(miz5)


miz6=[]
loop = range(34,100) # scrape chats from different hours.
for x in loop:

miz6.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]/span'%x).text)
print(miz6)


miz_text = [miz, miz2, miz3, miz4, miz5, miz6]


flatList_miz = []
for elem in miz_text:
    flatList_miz.extend(elem)
flatList_miz


miz_chatcomb = pd.DataFrame(flatList_miz)


miz_chatcomb.to_csv('mizkif.txt', index = False, sep = '\t')


# Channel 13-16
@author: Rafae


## For Channel 13 SYKKUNO
url12 = 'https://www.twitch.tv/videos/1211448557'  #pOKEMON
```

```
driver.get(url12)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

SYKKUNO=[]

loop = range(39, 77) # scrape  chats.

for x in loop:


SYKKUNO.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(SYKKUNO)


SYKKUNO2=[]

loop = range(21, 77) # scrape chats from different hours.

for x in loop:


SYKKUNO2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(SYKKUNO2)


SYKKUNO3=[]

loop = range(3, 57) # scrape chats from different hours.

for x in loop:


SYKKUNO3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(SYKKUNO3)
```

```
SYKKUNO4=[]

loop = range(3, 57) # scrape chats from different hours.

for x in loop:


SYKKUNO4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(SYKKUNO4)


SYKKUNO4text = [SYKKUNO, SYKKUNO2, SYKKUNO3, SYKKUNO4]


flatListSYKKUNO4 = []

for elem in SYKKUNO4text:

    flatListSYKKUNO4.extend(elem)

flatListSYKKUNO4


SYKKUNO4chatcomb = pd.DataFrame(flatListSYKKUNO4)

SYKKUNO4chatcomb.to_csv('SYKKUNO4chatcomb.txt', index = False, sep = '\t')






## For channel 14 SUMMIT1G

url13 = 'https://www.twitch.tv/summit1g'

driver.get(url13)
```

```
vdo13 = 'https://www.twitch.tv/videos/1211292509'
driver.get(vdo13)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
#SUMMIT1G=[]
#loop = range(13, 100) # scrape 100 chats.
#for x in loop:
#
SUMMIT1G.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
#print(SUMMIT1G)


SUMMIT1G2=[]
loop = range(13, 100) # scrape 100 chats.
for x in loop:

SUMMIT1G2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(SUMMIT1G2)


SUMMIT1G3=[]
loop = range(21, 100) # scrape 100 chats.
for x in loop:

SUMMIT1G3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
```

```
print(SUMMIT1G3)
type(SUMMIT1G3)
summit1gtext = [SUMMIT1G2, SUMMIT1G3]
summit1gtext

flatList = []
for elem in summit1gtext:
    flatList.extend(elem)
flatList

#summit1gtextcombined = SUMMIT1G2.extend(SUMMIT1G3)
#print(summit1gtextcombined)
s#ummit1gtextcombined.values.tolist()
summit1gchatcomb = pd.DataFrame(flatList)
summit1gchatcomb.to_csv('summit1gchatcomb.txt', index = False, sep = '\t')

#summit1gchat = pd.DataFrame(summit1gtext)
#summit1gchat.to_csv('summit1gchat.txt', index = False, sep = '\t')

#summit1gchat1 = pd.DataFrame(SUMMIT1G2)
#summit1gchat1.to_csv('summit1gchat1.txt', index = False, sep = '\t')

SUMMIT1G4=[]
loop = range(24, 150) # scrape 100 chats.
for x in loop:
```

```
SUMMIT1G4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(SUMMIT1G4)


SUMMIT1G5=[]
loop = range(7, 100) # scrape 100 chats.
for x in loop:

SUMMIT1G5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(SUMMIT1G5)


SUMMIT1G6=[]
loop = range(3, 100) # scrape 100 chats.
for x in loop:

SUMMIT1G6.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(SUMMIT1G6)


summit1gtext = [SUMMIT1G2, SUMMIT1G3, SUMMIT1G4, SUMMIT1G5, SUMMIT1G6]


flatList = []
for elem in summit1gtext:
   flatList.extend(elem)
flatList
```

```
summit1gchatcomb = pd.DataFrame(flatList)

summit1gchatcomb.to_csv('summit1gchatcomb.txt', index = False, sep = '\t')




## For channel 15 XROHAT

url14 = 'https://www.twitch.tv/xrohat'

driver.get(url14)


# wait the webpage to load for a while

driver.implicitly_wait(60)


xrohat=[]

loop = range(120, 94) # scrape 100 chats.

for x in loop:


xrohat.append(driver.find_element(By.XPATH,'/html/body/div[2]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/section/div/div[3]/div/div[2]/div[3]/div/div/div[%d]/div/div[2]/div/div/span[2]'%x).text)

print(xrohat)


xrohat = pd.DataFrame(xrohat)

xrohat.to_csv('xrohat.txt', index = False, sep = '\t')

#Since language is not english, we will avoid this channel's text
```

```
## For channel 16 TRAINWRECKSTV
url15 = 'https://www.twitch.tv/videos/1209676541' #18+ warning
driver.get(url14)

# wait the webpage to load for a while
driver.implicitly_wait(60)

TRAINWRECKSTV=[]
loop = range(1, 50) # conversation mainly?.
for x in loop:

TRAINWRECKSTV.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]
/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV)

TRAINWRECKSTV2=[]
loop = range(2, 33) # one game.
for x in loop:

TRAINWRECKSTV2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2
]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV2)

TRAINWRECKSTV3=[]
loop = range(6, 50) # another game within same video.
```

```
for x in loop:

TRAINWRECKSTV3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV3)


TRAINWRECKSTV5=[]        # scrape chats from different hours.
loop = range(69, 150)
for x in loop:

TRAINWRECKSTV5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV5)



TRAINWRECKSTV6=[]      # scrape chats from different hours.
loop = range(4, 150).
for x in loop:

TRAINWRECKSTV6.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV6)


/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[6]/div/div[2]/div/div[1]/div/span[2]/span[2]


TRAINWRECKSTV7=[]
```

```
loop = range(6, 150) # another game from same video.
for x in loop:

TRAINWRECKSTV7.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(TRAINWRECKSTV7)



trainWRECKStvcomb = [TRAINWRECKSTV, TRAINWRECKSTV2, TRAINWRECKSTV3, TRAINWRECKSTV5, TRAINWRECKSTV6, TRAINWRECKSTV7]


flatListRECK = []
for elem in trainWRECKStvcomb:
    flatListRECK.extend(elem)
flatListRECK


trainWRECKStvcomb = pd.DataFrame(flatListRECK)
trainWRECKStvcomb.to_csv('trainWRECKStvcomb.txt', index = False, sep = '\t')



driver.quit()


# For channel 19-20
@author: Rafae


## For channel 19 ADMIRALBAHROO
```

```
url19 = 'https://www.twitch.tv/videos/1215697012'
driver.get(url19)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
ADMIRALBAHROO1=[]
loop = range(30, 77) # scrape  chats.
for x in loop:

ADMIRALBAHROO1.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(ADMIRALBAHROO1)


ADMIRALBAHROO2=[]
loop = range(31, 77) # scrape chats from different time.
for x in loop:

ADMIRALBAHROO2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(ADMIRALBAHROO2)


ADMIRALBAHROO3=[]
loop = range(1, 57) # scrape chats from different time.
for x in loop:

ADMIRALBAHROO3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
```

```
print(ADMIRALBAHROO3)


ADMIRALBAHROO4=[]

loop = range(70, 170) # scrape chats from different hours.

for x in loop:


ADMIRALBAHROO4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(ADMIRALBAHROO4)


ADMIRALBAHROO5=[]

loop = range(3, 70) # scrape chats from different time.

for x in loop:


ADMIRALBAHROO5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(ADMIRALBAHROO5)


ADMIRALBAHROO6=[]

loop = range(2, 60) # scrape chats from different hours.

for x in loop:


ADMIRALBAHROO6.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(ADMIRALBAHROO6)
```

```
ADMIRALBAHROOtext = [ADMIRALBAHROO1, ADMIRALBAHROO2, ADMIRALBAHROO3, ADMIRALBAHROO4, ADMIRALBAHROO5, ADMIRALBAHROO6]

flatListADMIRALBAHROO = []
for elem in ADMIRALBAHROOtext:
    flatListADMIRALBAHROO.extend(elem)
flatListADMIRALBAHROO

ADMIRALBAHROOchatcomb = pd.DataFrame(flatListADMIRALBAHROO)
ADMIRALBAHROOchatcomb.to_csv('ADMIRALBAHROOchatcomb.txt', index = False, sep = '\t')
ADMIRALBAHROOchatcomb.to_csv('ADMIRALBAHROOchatcomb.csv', index = False, sep = '\t')




## For channel 20 philza
url20 = 'https://www.twitch.tv/videos/1214690145'  #18+ warning
driver.get(url20)

# wait the webpage to load for a while
driver.implicitly_wait(60)

philza1=[]
loop = range(43, 100) # scrape  chats - 16 minutes.
for x in loop:
```

```
philza1.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(philza1)


philza2=[]
loop = range(3, 70) # scrape from 25 minutes.
for x in loop:

philza2.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(philza2)


philza3=[]
loop = range(2, 50) # scrape chats from different hour.
for x in loop:

philza3.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(philza3)


philza4=[]
loop = range(7, 70) # scrape chats from different hour.
for x in loop:

philza4.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(philza4)
```

```
philza5=[]

loop = range(3, 50) # scrape chats.

for x in loop:


philza5.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(philza5)


philzatext = [philza1, philza2, philza3, philza4, philza5]


flatListphilza = []

for elem in philzatext:

    flatListphilza.extend(elem)

flatListphilza


philzachatcomb = pd.DataFrame(flatListphilza)

philzachatcomb.to_csv('philzachatcomb.txt', index = False, sep = '\t')

philzachatcomb.to_csv('philzachatcomb.csv', index = False, sep = '\t')


# Combine all chat into one DataFrame

## Python


@author: Chase(Cao)

XQCOW1 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\XQCOW1_df.txt', sep='\t')
```

```
XQCOW2 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\XQCOW2_df.txt', sep='\t')
XQCOW3 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\XQCOW3_df.txt', sep='\t')
XQCOW4 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\XQCOW4_df.txt', sep='\t')
XQCOW5 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\XQCOW5_df.txt', sep='\t')
RANBOO1 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\RANBOO1_df.txt', sep='\t')
RANBOO2 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\RANBOO2_df.txt', sep='\t')
RANBOO3 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\RANBOO3_df.txt', sep='\t')
RANBOO4 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\RANBOO4_df.txt', sep='\t')
RANBOO5 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\The_2nd_and_3rd_channels_raw_chat_data\\RANBOO5_df.txt', sep='\t')
ADMIRALBAHROO = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\ADMIRALBAHROOchatcomb.txt', sep='\t')
```

```
castro_1021 = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\castro_1021.txt', sep='\t')

mizkif = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\mizkif.txt', sep='\t')

montana = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\montana.txt', sep='\t')

NICKMERCS = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\NICKMERCS.txt', sep='\t')

philzachatcomb = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\philzachatcomb.txt', sep='\t')

summit1gchatcomb = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\summit1gchatcomb.txt', sep='\t')

SYKKUNO4chatcomb = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\SYKKUNO4chatcomb.txt', sep='\t')

trainWRECKStvcomb = pd.read_table('D:\Documents\\GitHub\\Game-live-streaming-project\\raw_chat_box_data_for_top_20\\trainWRECKStvcomb.txt', sep='\t')



chatbox=pd.concat([XQCOW1,XQCOW2,XQCOW3,XQCOW4,XQCOW5,RANBOO1,RANBOO2,RANBOO3,RANBOO4,RANBOO5,ADMIRALBAHROO,castro_1021,mizkif,montana,NICKMERCS,philzachatcomb,summit1gchatcomb,SYKKUNO4chatcomb,trainWRECKStvcomb],ignore_index=True)

print(chatbox)

chatbox_df= pd.DataFrame(chatbox)

chatbox_df.to_csv('D:\Documents\\final_chatbox_df.txt',index=False,sep='\t')


# Sentiment analysis

@author: Rafae
```

```
#IN R
library(tidyverse)
library(tidytext)
library(SnowballC)

wd = "C:\\Users\\MRS ELECTRONIC\\Documents\\GitHub\\Game-live-streaming-project"
setwd(wd)

text_data = read.delim("final_chatbox_df.txt")
summary(text_data)

chattext = select(text_data, X0)
tidy_dataset = unnest_tokens(chattext, word, X0)

#Removing stop words
data("stop_words")

tidy_dataset2 = anti_join(tidy_dataset, stop_words)

# Remove the numerical values from the column
patterndigits = '\\b[0-9]+\\b'
# Use regex
tidy_dataset2$word = str_remove_all(tidy_dataset2$word, patterndigits)


# Replace/remove all T-Mobile and Sprint new lines, tabs, and blank spaces with a value of nothing
# and then filter out or remove those values

tidy_dataset2$word = str_replace_all(tidy_dataset2$word, '[:space:]', '')

tidy_dataset3 = filter(tidy_dataset2,!(word == ''))


#Removing frequent but unnecessary. However, we were cautious to not remove any word that

#we don't understand. The reason is sometimes viewers might have used shoetened

#language that is meaningful. We didn't want to lose them.

list_remove = c("didnt","bahroo", "admiralbahroo", "summit", "miz", "hasan", "��", "https")

tidy_dataset3 = filter(tidy_dataset3, !(word %in% list_remove))


#Stemming

stemmed_data = wordStem(tidy_dataset3$word, language="en")


tidy_dataset4 = mutate_at(tidy_dataset3, "word", funs(wordStem((.), language="en")))




#Follow-Up Analysis

#Top10

counts5 = count(tidy_dataset4, word)


arrange(counts5, desc(n)) %>%

  ungroup %>%

  slice(1:10)


##IN PYTHON

import pandas as pd
import matplotlib.pyplot as plt
import os
import regex

import nltk
from nltk import word_tokenize, sent_tokenize
from nltk.corpus import stopwords
from nltk.stem import LancasterStemmer, WordNetLemmatizer, PorterStemmer

os.chdir(r'C:\Users\MRS ELECTRONIC\Documents\GitHub\Game-live-streaming-project')

chat_data = pd.read_fwf('final_chatbox_df.txt')

chat_data.rename(columns={'0': 'chattext'}, inplace=True)

#Remove stop word
stop = stopwords.words('english')

chat_data['chattext'] = chat_data['chattext'].apply(lambda x: " ".join(x for x in x.split() if x not in stop))

#Remove numerical values
patterndigits = '\\b[0-9]+\\b'
chat_data['chattext'] = chat_data['chattext'].str.replace(patterndigits,")

#Remove punctuation
patternpunc = '[^\w\s]'
```

```
chat_data['chattext'] = chat_data['chattext'].str.replace(patternpunc,")


#Convert to lowercase and remove 2 company names

chat_data['chattext'] = chat_data['chattext'].apply(lambda x: " ".join(x.lower() for x in x.split()))


#Removing unneccessary words coming frequently

list_remove = ["didnt","bahroo", "admiralbahroo", "summit", "miz", "hasan", "ðÿ", "https"]


chat_data['chattext'] = chat_data['chattext'].apply(lambda x: " ".join(x for x in x.split() if x not in list_remove))


# Stem the data using PorterStemmer()


porstem = PorterStemmer()


chat_data['chattext'] = chat_data['chattext'].apply(lambda x: " ".join([porstem.stem(word) for word in x.split()]))



# Creating a document-term matrix

#===============================

from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer()


tokens_data = pd.DataFrame(vectorizer.fit_transform(chat_data['chattext']).toarray(), columns=vectorizer.get_feature_names())
```

```
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.decomposition import LatentDirichletAllocation
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.decomposition import NMF

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score, plot_confusion_matrix


vectorizer = CountVectorizer(max_df=0.8, min_df=4, stop_words='english')

doc_term_matrix = vectorizer.fit_transform(chat_data['chattext'].values.astype('U'))

doc_term_matrix.shape
# Performing LDA
LDA = LatentDirichletAllocation(n_components=4, random_state=35)
LDA.fit(doc_term_matrix)

first_topic = LDA.components_[0]
top_topic_words = first_topic.argsort()[-10:]

for i,topic in enumerate(LDA.components_):
    print(f'Top 10 words for topic #{i}:')

    print([vectorizer.get_feature_names()[i] for i in topic.argsort()[-10:]])

    print('\n')



topic_values = LDA.transform(doc_term_matrix)

topic_values.shape

chat_data['topic'] = topic_values.argmax(axis=1)


##Proportion of topics in chat

print('Proportion of topic 0' + str(len(chat_data[chat_data['topic'] == 0]) / len(chat_data['topic'])))

print('Proportion of topic 1' + str(len(chat_data[chat_data['topic'] == 1]) / len(chat_data['topic'])))

print('Proportion of topic 2' + str(len(chat_data[chat_data['topic'] == 2]) / len(chat_data['topic'])))

print('Proportion of topic 3' + str(len(chat_data[chat_data['topic'] == 3]) / len(chat_data['topic'])))



# Perform Non-Negative Matrix Factorization (NMF)


tfidf_vect = TfidfVectorizer(max_df=0.8, min_df=4, stop_words='english')


doc_term_matrix2 = tfidf_vect.fit_transform(chat_data['chattext'].values.astype('U'))


nmf = NMF(n_components=4, random_state=42)


nmf.fit(doc_term_matrix2)
```

```
for i, topic in enumerate(nmf.components_):
    print(f'Top 10 words for topic #{i}:')
    print([vectorizer.get_feature_names()[i] for i in topic.argsort()[-10:]])
    print('\n')

topic_values2 = nmf.transform(doc_term_matrix2)
chat_data['topic2'] = topic_values2.argmax(axis=1)

##Proportion of topics in chat
print('Proportion of topic 0' + str(len(chat_data[chat_data['topic2'] == 0]) / len(chat_data['topic2'])))
print('Proportion of topic 1' + str(len(chat_data[chat_data['topic2'] == 1]) / len(chat_data['topic2'])))
print('Proportion of topic 2' + str(len(chat_data[chat_data['topic2'] == 2]) / len(chat_data['topic2'])))
print('Proportion of topic 3' + str(len(chat_data[chat_data['topic2'] == 3]) / len(chat_data['topic2'])))



##IN R

library(wordcloud)
library(udpipe)
library(lattice)

# in the NRC dictionary
#===================================
## Positive - Negative with nrc dictionary

nrc_posneg = get_sentiments('nrc') %>%
  filter(sentiment == 'positive' |
           sentiment == 'negative')

nrow(nrc_posneg)

newjoin2 = inner_join(tidy_dataset4, nrc_posneg)
counts8 = count(newjoin2, word, sentiment)
spread3 = spread(counts8, sentiment, n, fill = 0)
content_data2 = mutate(spread3, contentment = positive - negative, linenumber = row_number())
tweet_posneg = arrange(content_data2, desc(contentment))

(tweet_posneg2 = tweet_posneg %>%
    slice(1:10,183:192))

ggplot(tweet_posneg2, aes(x=linenumber, y=contentment, fill=word)) +
  coord_flip() +
  theme_light(base_size = 15) +
  labs(
    x='Index Value',
    y='Contentment',
    title='Positive vs Negative - Sentiment Ananlysis'
  ) +
  theme(
    legend.position = 'bottom',
    panel.grid = element_blank(),
    axis.title = element_text(size = 10),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10)
  ) +
  geom_col()



## We are taking Surprise - Disgust to see whether people were surprised

nrc_surdis = get_sentiments('nrc') %>%
  filter(sentiment == 'surprise' |
           sentiment == 'disgust')

nrow(nrc_surdis)

newjoin2 = inner_join(tidy_dataset4, nrc_surdis)
counts8 = count(newjoin2, word, sentiment)
spread4 = spread(counts8, sentiment, n, fill = 0)
content_data3 = mutate(spread4, contentment = surprise - disgust, linenumber = row_number())
emo_surdis = arrange(content_data3, desc(contentment))
emo_surdis
(tweet_surdis2 = emo_surdis %>%
    slice(1:5,58:62))
```

```
ggplot(tweet_surdis2, aes(x=linenumber, y=contentment, fill=word)) +
  coord_flip() +
  theme_light(base_size = 20) +
  labs(
    x='Index Value',
    y='Contentment',
    title='Surprise vs Disgust - Sentiment Ananlysis'
  ) +
  theme(
    legend.position = 'bottom',
    panel.grid = element_blank(),
    axis.title = element_text(size = 10),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10)
  ) +
  geom_col()



## We are taking Trust - Fear to see whether people relied on player's performance


nrc_trufea = get_sentiments('nrc') %>%
  filter(sentiment == 'trust' |
           sentiment == 'fear')


nrow(nrc_trufea)
```

```
newjoin2 = inner_join(tidy_dataset4, nrc_trufea)

counts8 = count(newjoin2, word, sentiment)

spread4 = spread(counts8, sentiment, n, fill = 0)

content_data3 = mutate(spread4, contentment = trust - fear, linenumber = row_number())

emo_trufea = arrange(content_data3, desc(contentment))

emo_trufea

(emo_trufea = emo_trufea %>%

    slice(1:5,103:107))


ggplot(emo_trufea, aes(x=linenumber, y=contentment, fill=word)) +

  coord_flip() +

  theme_light(base_size = 20) +

  labs(

    x='Index Value',

    y='Contentment',

    title='Trust vs Fear - Sentiment Ananlysis'

  ) +

  theme(

    legend.position = 'bottom',

    panel.grid = element_blank(),

    axis.title = element_text(size = 10),

    axis.text.x = element_text(size = 10),

    axis.text.y = element_text(size = 10)

  ) +

  geom_col()
```

```
### POST (Parts of Speech Tagging)

library(Rcpp)


ud_model = udpipe_download_model(language = "english")

tidy_post1 = tidy_dataset4 %>%

  select(word)


tidy_post1 = tidy_dataset4 %>%

  select(word)


ud_model = udpipe_load_model(ud_model$file_model)



tagging_data = as.data.frame(udpipe_annotate(ud_model, x = tidy_post1$word))


# Most Occuring NOUNS

#======================

noun_stats = subset(tagging_data, upos %in% c("NOUN"))


noun_stats2 = txt_freq(noun_stats$token)


noun_stats2$key = factor(noun_stats2$key, levels = rev(noun_stats2$key))
```

```
noun_stats2 %>%
  slice(1:15) %>%
  ggplot(aes(x=key, y=as.factor(freq), fill=freq)) +
  coord_flip() +
  theme_light(base_size = 15) +
  labs(
    x='Frequency',
    y='',
    title='Noun Occurrences'
  ) +
  theme(
    legend.position = 'none',
    panel.grid = element_blank(),
    axis.title = element_text(size = 10),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    title = element_text(size = 13)
  ) +
  scale_fill_gradient(low="orange", high="orange3") +
  geom_col()


#==========================
# Most Occuring ADJECTIVES
#==========================
adjstats = subset(tagging_data, upos %in% c("ADJ"))
```

```
adjstats2 = txt_freq(adjstats$token)

adjstats2$key = factor(adjstats2$key, levels = rev(adjstats2$key))

adjstats2 %>%
  slice(1:15) %>%
  ggplot(aes(x=key, y=as.factor(freq), fill=freq)) +
  coord_flip() +
  theme_light(base_size = 15) +
  labs(
    x='Frequency',
    y='',
    title='Adjective Occurrences'
  ) +
  theme(
    legend.position = 'none',
    panel.grid = element_blank(),
    axis.title = element_text(size = 10),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    title = element_text(size = 13)
  ) +
  scale_fill_gradient(low="chartreuse", high="chartreuse3") +
  geom_col()
```

```
#=====================
# Most Occuring VERBS
#=====================
verbstats = subset(tagging_data, upos %in% c("VERB"))

verbstats2 = txt_freq(verbstats$token)

verbstats2$key = factor(verbstats2$key, levels = rev(verbstats2$key))

verbstats2 %>%
  slice(1:15) %>%
  ggplot(aes(x=key, y=as.factor(freq), fill=freq)) +
  coord_flip() +
  theme_light(base_size = 15) +
  labs(
    x='Frequency',
    y='',
    title='Verb Occurrences'
  ) +
  theme(
    legend.position = 'none',
    panel.grid = element_blank(),
    axis.title = element_text(size = 10),
    axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10),
    title = element_text(size = 13)
  ) +
  scale_fill_gradient(low="tan", high="tan3") +
  geom_col()

####===================
##Named-Entity Recognition

library(stringr)
library(NLP)
library(openNLP)

sent_token_annotator = Maxent_Sent_Token_Annotator()
word_token_annotator = Maxent_Word_Token_Annotator()
pos_tag_annotator = Maxent_POS_Tag_Annotator()

tidy_dataset5 = unlist(tidy_dataset4)
tidy_dataset5 = paste(tidy_dataset5)
tidy_dataset5 = as.String(tidy_dataset5)

anno1 = annotate(tidy_dataset5, list(sent_token_annotator,
                                word_token_annotator))

anno2 = annotate(tidy_dataset5, pos_tag_annotator, anno1)

(annotate(tidy_dataset5, Maxent_POS_Tag_Annotator(probs = TRUE), anno2))
```

```
anno2wrd = subset(anno2, type == "word")

tags = sapply(anno2wrd$features, `[[`, "POS")


sprintf("%s/%s", tidy_dataset5[anno2wrd], tags)



anno3 = annotate(tidy_dataset5, list(sent_token_annotator,
                                     word_token_annotator))


# Named-entity Person
#=======================
entity_annotator = Maxent_Entity_Annotator(kind='person')
entity_annotator

anno5 = entity_annotator(tidy_dataset5, anno3)

tidy_dataset5[anno5]


# Named-entity locations
#=======================
loc_annotator = Maxent_Entity_Annotator(kind='location')
```

```
anno6 = loc_annotator(tidy_dataset5, anno3)


# Retrieve text

tidy_dataset5[anno6]




#=============================

# Named-entity organizations

#=============================

org_annotator = Maxent_Entity_Annotator(kind='organization')


anno7 = org_annotator(tidy_dataset5, anno3)

anno7


# Retrieve text

tidy_dataset5[anno7]
```

```
fro
m
sel
eni
um
imp
ort
web
dri
ver
        from selenium.webdriver.support import expected_conditions as EC
        import selenium.webdriver.support.ui as ui
        from selenium.webdriver.common.keys import Keys
        from selenium.webdriver.firefox.service import Service
        from selenium.webdriver.common.by import By
        import pandas as pd
        from selenium.webdriver.support.ui import Select
        service=Service('D:\Documents\geckodriver.exe')
        driver = webdriver.Firefox(service=service)
        # I wrote codes for 17 channels but I ran these codes the whole day and different bugs
        appeared at different time even though I never changed my codes. I tried to
        # revised the codes everywhere according to infinitely many bugs. The most closed one
        is that I got 15 channels's chatbox and then failed. At the end, I could only gaurantee
        # the first 3 channels's chatbox. Sad.

        #1. For channel CRITICALROLE, it doesn't stream everyday and record video of past
        streaming is only avaiable for subscribers.

        # For channel xQcOW, the data of chatbox is scraped from his last streaming
        url1 = 'https://www.twitch.tv/videos/1185296137'
        driver.get(url1)
        # wait the webpage to load for a while
```

```
driver.implicitly_wait(40)
# Scrape text in chatbox by loop
XQCOW=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
XQCOW.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(XQCOW)

#For channel RANBOOLIVE
url2 = 'https://www.twitch.tv/videos/1184954374'
driver.get(url2)
# wait the webpage to load for a while
driver.implicitly_wait(40)
# Scrape text in chatbox by loop
RANBOO=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
RANBOO.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(RANBOO)

#For channel GAULES
url3 = 'https://www.twitch.tv/videos/1184792471'
driver.get(url3)
# wait the webpage to load for a while
driver.implicitly_wait(100)
# Scrape text in chatbox by loop
```

```
GAULES=[]

loop = range(2, 101) # scrape 100 chats.

for x in loop:

GAULES.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[
2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div
/span[2]'%x).text)

print(GAULES)

list=list(zip(XQCOW, RANBOO, GAULES))

raw_chat= pd.DataFrame(list,columns=['XQCOW','RANBOO','GAULES'])

raw_chat.to_csv('D:\Documents\chatbox_data.txt',index=False,sep='\t')


#For IBAI

url4 = 'https://www.twitch.tv/videos/1185671627'

driver.get(url4)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

IBAI=[]

loop = range(2, 101) # scrape 100 chats.

for x in loop:

IBAI.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]
/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/s
pan[2]'%x).text)

print(IBAI)


#For NICKMERCS

url5 = 'https://www.twitch.tv/videos/1183946438'

driver.get(url5)

# wait the webpage to load for a while

driver.implicitly_wait(60)
```

```
# Scrape text in chatbox by loop

NICKMERCS=[]

loop = range(2, 101) # scrape 100 chats.

for x in loop:

NICKMERCS.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(NICKMERCS)


#For 加藤純一です

url6 = 'https://www.twitch.tv/videos/1185274905'

driver.get(url6)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

junichi=[]

loop = range(2, 101) # scrape 100 chats.

for x in loop:

junichi.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(junichi)


#For hasanabi

url7 = 'https://www.twitch.tv/hasanabi'

driver.get(url7)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

hasanabi=[]
```

```
loop = range(2, 101) # scrape 100 chats.
for x in loop:
hasanabi.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/section/div/div[3]/div/div[2]/div[3]/div/div/div[%d]/div/div[2]/div/div/span[2]'%x).text)
print(hasanabi)


#For AURONPLAY
url8 = 'https://www.twitch.tv/videos/1185822982'
driver.get(url8)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
AURONPLAY=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
AURONPLAY.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(AURONPLAY)


# For MONTANABLACK88
url9 = 'https://www.twitch.tv/videos/1185712458'
driver.get(url9)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
MONTANABLACK88=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
```

```
MONTANABLACK88.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(MONTANABLACK88)


# For CASTRO_1021
url10 = 'https://www.twitch.tv/videos/1183769633'
driver.get(url10)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
CASTRO_1021=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
CASTRO_1021.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(CASTRO_1021)


# For CASIMITO
url11 = 'https://www.twitch.tv/videos/1185287697'
driver.get(url11)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
CASIMITO=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
```

```
CASIMITO.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(CASIMITO)


# For SYKKUNO
url12 = 'https://www.twitch.tv/videos/1183283146'
driver.get(url12)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
SYKKUNO=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
SYKKUNO.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(SYKKUNO)


# For SUMMIT1G
url13 = 'https://www.twitch.tv/summit1g'
driver.get(url13)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
SUMMIT1G=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
```

```
SUMMIT1G.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/section/div/div[3]/div/div[2]/div[3]/div/div/div[%d]/div/div[2]/div/div/span[2]'%x).text)

print(SUMMIT1G)


# For XROHAT It's been a long time after his last streaming. The chat box replay is no longer avaiable even for any of his video.


# TRAINWRECKSTV

url14 = 'https://www.twitch.tv/videos/1184207653'

driver.get(url14)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

TRAINWRECKSTV=[]

loop = range(2, 101) # scrape 100 chats.

for x in loop:

TRAINWRECKSTV.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)

print(TRAINWRECKSTV)


# For MIZKIF

url15 = 'https://www.twitch.tv/mizkif'

driver.get(url15)

# wait the webpage to load for a while

driver.implicitly_wait(60)

# Scrape text in chatbox by loop

MIZKIF=[]

loop = range(2, 101) # scrape 100 chats.
```

```
for x in loop:
MIZKIF.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/section/div/div[3]/div/div[2]/div[3]/div/div/div[%d]/div/div[2]/div/div/span[2]'%x).text)
print(MIZKIF)



# For ADINROSS replay only avaiable for subscribers


# For ADMIRALBAHROO
url16 = 'https://www.twitch.tv/videos/1185756507'
driver.get(url16)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
admiralbahroo=[]
loop = range(2, 101) # scrape 100 chats.
for x in loop:
admiralbahroo.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div/span[2]'%x).text)
print(admiralbahroo)


# PHILZA
url17 = 'https://www.twitch.tv/videos/1184963530'
driver.get(url17)
# wait the webpage to load for a while
driver.implicitly_wait(60)
# Scrape text in chatbox by loop
PHILZA=[]
```

```
loop = range(2, 101) # scrape 100 chats.

for x in loop:

PHILZA.append(driver.find_element(By.XPATH,'/html/body/div[1]/div/div[2]/div[1]/div[
2]/div/div[1]/div/div/div/div/div/div[2]/div/div/ul/li[%d]/div/div[2]/div/div[1]/div
/span[2]'%x).text)

print(PHILZA)


#list=list(zip(XQCOW, RANBOO, GAULES, IBAI, NICKMERCS, junichi, hasanabi, AURONPLAY,
MONTANABLACK88, CASTRO_1021, CASIMITO, SYKKUNO, SUMMIT1G, TRAINWRECKSTV, MIZKIF,
admiralbahroo, PHILZA))

#raw_chat= pd.DataFrame(list)

#raw_chat.to_csv('D:\Documents\chatbox_data.txt',index=False,sep='\t')
```