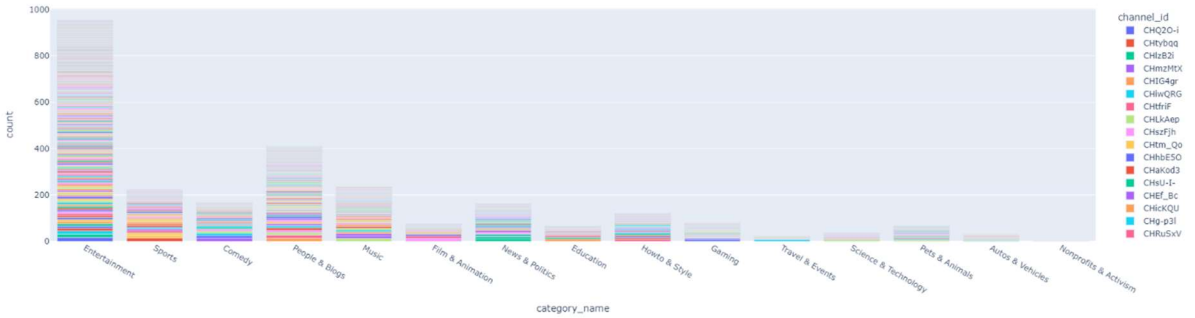
Q1. Visualization of data

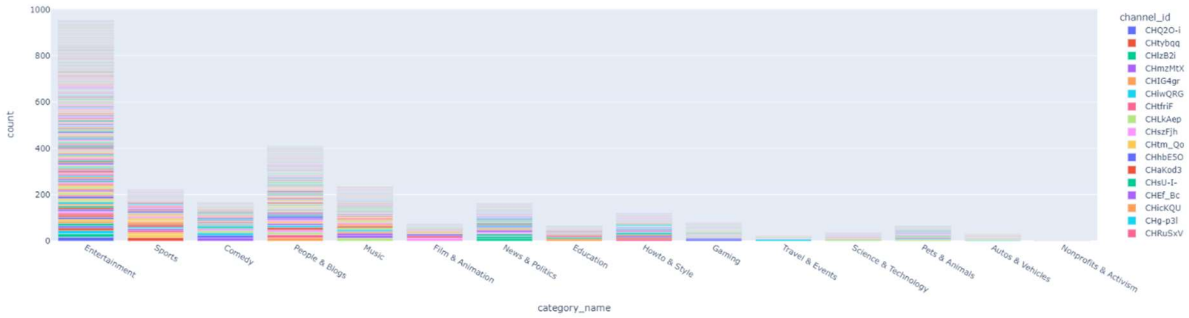
#1. Whole period [category->channel->number of videos]

- I thought that total amount of video number by category is more important than the individual channel's video number.
- For that reason, I decided to use stacked bar plot, and bars are colored by channels



#2. Monthly [category->channel-> number of videos]

- For this data, its upload date is range from march 25th to July 29th.
- I wanted to make interactive plot but I couldn't.
- Instead, if you change month_num integer (3 to 7), you can get the chart form data relate to month_num integer.



(* month_num =5)

#3. Monthly top 10 channels

- This chart uses same function to filter monthly data with #2.
- You should enter month_num
- Since there is no specific sub-sorting condition, they are displayed in order of name when they have the same rank.

	index	count
0	CHYRrUD	9
2	CH29-Il	6
3	CHk4XjB	6
1	CHUQVGX	6
4	CHlRNDV	5
5	CHszFjh	5
6	CHArK9M	5
12	CHLkAep	4
16	CHcQTRi	4
15	CHEbRSm	4

(*month_num = 7)

#4. Weekly top 5 channels

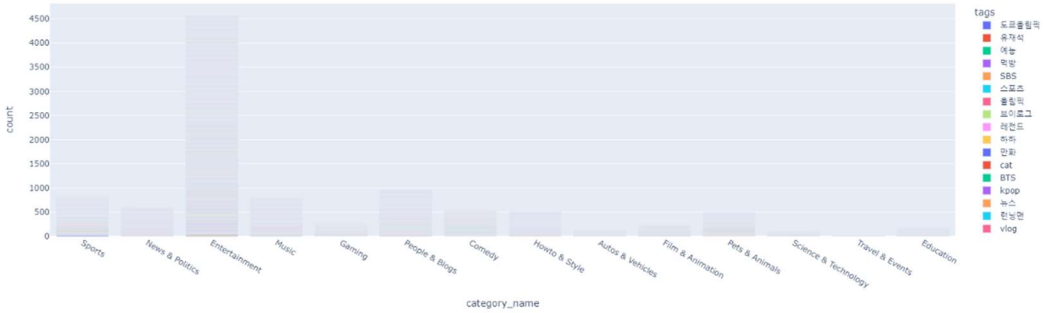
- It requires week_num.
- week_num is range from 0(march 21st to 28th) to 18(July 24th to 31st)

Investigate from 2021-05-16 00:00:00 to 2021-05-23 00:00:00

	index	count
0	CHDV9zg	3
2	CH3IZKs	3
3	CH4DnB5	3
1	CH9GtSL	3
10	CHExw7_	2

(* week_num = 8)

#5. Monthly [category-> tags -> number of keywords]



Q2. Develop new indicators that can classify videos to meet popular video criteria and use them to explain how they correlate with engagement.

A. My indicator

- Increasing rate of view, likes, comments(per day or per hour) : If the videos are popular, many people watch and share the contents of video
- Views of non subscribers : Subscribers can easily click the video but non subscribers are hard to see the movie and it is close to the meaning of popular.
- Number of links shared : In order for a video to become famous, it has a ripple effect for many people to share it with the people around it.

#Code

- Code for moule's importing and function

```
import os
import sys
import pandas as pd
import numpy as np
import torch
import random
```

```
# seed
seed = 7777
random.seed(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed_all(seed)
```

```
from google.colab import drive
drive.mount('/content/drive')
df = pd.read_csv("./drive/MyDrive/pretest_data.csv",error_bad_lines=Fal
se)
```

```
import datetime
df['published_date'] = df['published_date'].astype('datetime64[ns]')
df['on_trending_date'] = df['on_trending_date'].astype('datetime64[ns]'
)
df['off_trending_date '] = df['off_trending_date'].astype('datetime64[n
s]')
df = df.sort_values(by='published_date',ascending=True)
```

```
def get_month_df(df,month_num=3):
  date_start_str=['2021-03-01','2021-04-01','2021-05-01','2021-06-
01','2021-07-01']
  date_end_str=['2021-03-31','2021-04-30','2021-05-31','2021-06-
30','2021-07-31']

  date_start = datetime.datetime.strptime(date_start_str[month_num-
3], '%Y-%m-%d')
  date_end = datetime.datetime.strptime(date_end_str[month_num-
3], '%Y-%m-%d')

  return df[(df['published_date'] <= date_end) & (df['published_date']
>= date_start)]
```

```
def get_week_df(df,week_num=3):
  date_start_str='2021-03-21'
  date_end_str='2021-03-28'

  date_start = datetime.datetime.strptime(date_start_str, '%Y-%m-%d')+d
atetime.timedelta(days=7*week_num)
  date_end = datetime.datetime.strptime(date_end_str, '%Y-%m-%d')+datet
ime.timedelta(days=7*week_num)

  return date_start,date_end, df[(df['published_date'] <= date_end) & (
df['published_date'] >= date_start)]
```

- Q1-1

```
import plotly.express as px

df_total_num = df.sort_values(by=['category_name','channel_id','publish
ed_date'] ,ascending=True)
df_total_num = df_total_num.groupby(["category_name","channel_id"])['ch
annel_id'].count().reset_index(name="count").sort_values(by='count',asc
ending=False)
fig = px.bar(df_total_num, x="category_name" ,y='count', color = "chann
el_id")
fig.show()
```

- Q1-2

```
month_num = 5 #3 월에서 7 월까지 있음

df_month_num = get_month_df(df,month_num)
df_month_num = df_month_num.sort_values(by=['category_name','channel_id
','published_date'] ,ascending=True)
df_month_num = df_month_num.groupby(["category_name","channel_id"])['ch
annel_id'].count().reset_index(name="count").sort_values(by='count',asc
ending=False)

# df_month_num.head(20)
fig = px.bar(df_month_num, x="category_name" ,y='count', color = "chann
el_id")
fig.show()
```

- Q1-3

```
month_num = 7 #3 to 7 (3 월에서 7 월까지 있음)

df_month_num = get_month_df(df,month_num)
df_month_num_count = df_month_num['channel_id'].value_counts().reset_in
dex(name="count").sort_values(by='count',ascending=False)
df_month_num_top10 = df_month_num_count.iloc[:10]
df_month_num_top10
```

```
week_num = 8 #0 to 18 (3 월 21 에서 7 월 29 일까지 있음)

week_start, week_end, df_week_num = get_week_df(df,week_num)
df_week_num_count = df_week_num['channel_id'].value_counts().reset_inde
x(name="count").sort_values(by='count',ascending=False)
df_week_num_top5 = df_week_num_count.iloc[:5]
print(f"Investigate from {week_start} to {week_end}")
df_week_num_top5
```

- Q1-4

```
month_num = 7 #3 to 7 (3 월에서 7 월까지 있음)

df_month_num = get_month_df(df,month_num)
df_month_num_filtered = df_month_num[['category_name','tags']].dropna()

df_tag = []
for i, tags in enumerate(df_month_num_filtered['tags'].values):
  category = df_month_num_filtered['category_name'].iloc[i]
  tag_list = list(tags.split('|'))
  for tag in tag_list:
    df_tag.append([category,tag])

df_tag = pd.DataFrame(df_tag,columns=['category_name','tags'])
df_tag = df_tag.groupby(["category_name","tags"])['tags'].count().reset
_index(name="count").sort_values(by='count',ascending=False)

# df_tag(20)
fig = px.bar(df_tag, x="category_name" ,y='count', color = "tags")
fig.show()
```