



Universidad Peruana de Ciencias Aplicadas

Trabajo Final

Administración de la Información (CC52)

Tema:

Creación de conocimiento a partir de los datos en Python

Docente:

Patricia Daniela Reyes Silva

Integrantes:

Melendez Huamanchumo, Nander Emanuel - U201922331

Arroyo Bonifaz, Luis Roberto - U201716094

Junio, 2022

1. Objetivos del proyecto

El objetivo es realizar un proyecto de data science que nos permita determinar las tendencias, gustos, categorías y mejor valorados tipos de videos y canales de Youtube Francia. A través de un modelado de datos se conservará o añadirá información relevante que permita poder resolver alguna imperfección de los mismos para que nuestro análisis final pueda ser el más óptimo. Se utilizara el lenguaje de programación Python, y las librerías pandas, numpy, matplotlib y seaborn.

2. Caso de análisis

Los datos a manejar fueron extraídos de la plataforma kaggle, es un registro diario de los videos de Youtube de mayor tendencia, en nuestro caso estaremos utilizando los registros de Youtube Francia, los datos son de aproximadamente 3 años de antigüedad.

Un análisis de este tipo de datos podría ser beneficioso para diferentes tipos de empresas o inclusive personas naturales. Hoy en día las empresas utilizan mucho estas plataformas para poder anunciar sus productos por internet, tener esta información le podría dar una idea de en qué categorías de videos podría ser más beneficioso publicar sus anuncios. También para personas naturales que quieran adentrarse en el mundo del streaming, se podría dar una idea de que tipo de videos son los más apoyados, y podría definir su tipo de contenido futuro.

3. Conjunto de datos

Se tienen 2 conjuntos de datos, uno es un archivo de tipo csv que contendrá toda la información respecto a los vídeos tendencia de Youtube Francia, el otro archivo es un tipo json que tiene la información respecto a las categorías de videos disponibles. Respecto al archivo de información de los videos se tiene lo siguiente:

- **video_id**: Identificador único de un video.
- **trending_date**: Fecha en la que este video se ubicó en tendencia en su país.
- **title**: Título del video.
- **channel_title**: Nombre del canal desde el que fue subido el video.
- **category_id**: Id del tipo de categoría en el que se encuentra el video.
- **publish_time**: Fecha de publicación del video.
- **tags**: Etiquetas puestas en los videos para su búsqueda.
- **views**: Cantidad de visitas del video en la fecha de tendencia.
- **likes**: Cantidad de likes del video en la fecha de tendencia.
- **dislikes**: Cantidad de dislikes del video en la fecha de tendencia.
- **comment_count**: Cantidad de comentarios del video en la fecha de tendencia.
- **thumbnail_link**: Link de la imagen de miniatura del video.
- **comments_disabled**: Estado de los comentarios, si están o no activados.
- **ratings_disabled**: Estado de visualización de los likes y dislikes, si están o no activados.
- **video_error_or_removed**: Indica si el video ha sido a o no removido.
- **description**: Descripción del video.
- **state**: Estado desde el que fue subido el video.
- **lat**: Latitud desde la que fue subido el video.
- **lon**: Longitud desde la que fue subido el video.

- **geometry:** Coordenada desde la que fue subido el video.

Respecto al archivo de información de las categorías:

- **kind:** Tipo de información.
- **etag:** Link que incluye este tipo de categoría.
- **id:** identificador del tipo de categoría.
- **snippet:** Información respecto de la categoría que contiene lo siguiente:
- **channelId:** Identificador del canal que incluye los videos de dicha categoría.
- **title:** Título del tipo de categoría.
- **assignable:** Indica si se puede asignar tal categoría de video.

4. Análisis exploratorio

a. Carga

Para la carga de datos necesitamos 3 archivos, información de los videos, información de las categorías de los videos de Francia, y además la información de las categorías de los videos de US debido a que algunos id de las categorías no estaban presentes en el archivo de Francia. Primero se importaran las librerías que nos ayudaran con el desarrollo del proyecto.

```
1 import pandas as pd
2 import matplotlib.pyplot as plt
3 import numpy as np
4 import seaborn as sns
5 import json
6 %matplotlib inline
```

Carga de los datos de las categorías de US.

```
1 jDataUS = json.load(open('US_category_id.json'))
2 for elem in jDataUS['items']:
3     print(elem)
```

Carga de los datos de las categorías de Francia.

```
jData = json.load(open('FR_category_id.json'))
for elem in jData['items']:
    print(elem)
```

Carga de la información de los vídeos tendencia en Francia.

```
data = pd.read_csv("FRvideos_cc50.csv")
data.head()
```

b. Inspección

Se realiza una inspección de los datos, la cantidad de valores, si existen valores nulos, los tipos de datos.

```
1 data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40724 entries, 0 to 40723
Data columns (total 20 columns):
#   Column                      Non-Null Count  Dtype
---  -
0   video_id                   40724 non-null  object
1   trending_date              40724 non-null  object
2   title                     40724 non-null  object
3   channel_title             40724 non-null  object
4   category_id               40724 non-null  int64
5   publish_time              40724 non-null  object
6   tags                      40724 non-null  object
7   views                     40724 non-null  int64
8   likes                     40724 non-null  int64
9   dislikes                  40724 non-null  int64
10  comment_count              40724 non-null  int64
11  thumbnail_link             40724 non-null  object
12  comments_disabled          40724 non-null  bool
13  ratings_disabled           40724 non-null  bool
14  video_error_or_removed     40724 non-null  bool
15  description                 37812 non-null  object
16  state                     40724 non-null  object
17  lat                       40724 non-null  float64
18  lon                       40724 non-null  float64
19  geometry                   40724 non-null  object
```

```
1 data.describe()
```

	category_id	views	likes	dislikes	comment_count	lat	lon
count	40724.000000	4.072400e+04	4.072400e+04	4.072400e+04	4.072400e+04	40724.000000	40724.000000
mean	20.123809	4.199219e+05	1.738886e+04	8.149624e+02	1.832453e+03	36.986535	-2.314779
std	6.984422	1.772130e+06	8.720509e+04	1.139219e+04	1.404321e+04	19.889834	25.443453
min	1.000000	2.230000e+02	0.000000e+00	0.000000e+00	0.000000e+00	-21.033511	-61.532999
25%	17.000000	1.697450e+04	3.380000e+02	1.800000e+01	5.600000e+01	42.699989	-0.530030
50%	23.000000	7.372100e+04	1.892500e+03	8.300000e+01	2.350000e+02	45.899975	2.666648
75%	24.000000	2.708088e+05	7.969500e+03	3.350000e+02	8.410000e+02	48.516663	6.030009
max	44.000000	1.009116e+08	4.750254e+06	1.353661e+06	1.040912e+06	50.283325	55.712816

c. Pre-procesado

El primer detalle que se tuvo que modificar fue agregar una columna que tuviera la descripción del tipo de categoría de vídeo, para ello previamente se tuvo que identificar si todos los id de las categorías estaban presentes en el archivo de Francia, posterior a ello una vez tuvimos almacenados los ids y sus títulos de categoría los insertamos a nuestro dataframe.

Primero verificamos que los IDs de los videos del dataframe coincidan con los del JSON

```
1 def checkNotAvailableIDs(j, d):
2     jAvailableIDs = []
3     for i in j['items']:
4         jAvailableIDs.append(int(i['id']))
5
6     notAvailableIDs = []
7     for dfIDs in d['category_id']:
8         if dfIDs not in jAvailableIDs and dfIDs not in notAvailableIDs:
9             notAvailableIDs.append(dfIDs)
10
11     return notAvailableIDs
12
13 dataNonExistentIDs = checkNotAvailableIDs(jData, data)
14 dataNonExistentIDs
```

Usando la función de la celda anterior podemos ver que existen videos que tienen un id de categoría que no existen. En este caso es el ID 29

```
[ ] 1 data[data['category_id'].isin(dataNonExistentIDs)]
```

	video_id	trending_date	title	channel_title	category_id	publish_time	tags	views
552	ssLc5Vnos-A	17.16.11	Chapelet du 15 novembre 2017	KTOTV	29	2017-11-15T16:15:49.000Z	Chapelet" "Lourdes" "Eglise" "Catholique" "Foi...	5606
833	_7Oe5S2-CDw	17.18.11	Refugiés de Matthieu Tribes	Collectif Nation Refuge	29	2017-11-16T19:18:31.000Z	CNR" "respects" "quad" "bbdo" "matthieu tribes...	42396

Por lo que procederemos a extraer dichas categorías de la data disponible para Estados Unidos y las agregamos al json de Francia

```
[10] 1 for elem in jDataUS['items']:
2     if int(elem['id']) in dataNonExistentIDs and elem not in jData['items']:
3         jData['items'].append(elem)
4
5 for elem in jData['items']:
6     if int(elem['id']) in dataNonExistentIDs:
7         print(elem['id'], '->', elem['snippet']['title'])
```

```
29 -> Nonprofits & Activism
```

Ahora agregaremos una columna con los titulos de los IDs

```
[ ] 1 categoriesTitles = {}
2 for item in jData['items']:
3     categoriesTitles[int(item['id'])] = item['snippet']['title']
4
5 data['category_title'] = [categoriesTitles[id] for id in data['category_id']]
6 data
```

Adicionalmente, si revisamos la data guardada en el JSON, notaremos que hay algunos items con un parámetro **assignable** falso, por lo que también tendremos que descartarlos

```
[ ] 1 def checkNotAssignables(j):
2     result = []
3     for elem in j['items']:
4         if not elem['snippet']['assignable']:
5             result.append(int(elem['id']))
6
7     return result
8
9 notAssignables = checkNotAssignables(jData)
10 notAssignables
```

```
[18, 21, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44]
```

```
[ ] 1 data = data[~data['category_id'].isin(notAssignables)]
2 data
```

Ahora tenemos que actualizar el formato de `trending_date` y `publish_time`, pasará de tipo objeto a tipo `datetime`.

Ahora procedemos a modificar la columna `trending_date`, ya que esta, al representar una fecha, debe de estar representada en su respectivo formato.

```
[ ] 1 data['trending_date'] = pd.to_datetime(data['trending_date'], format='%y.%d.%m')
2 data
```

	video_id	trending_date	title	channel_title	category_id	publish_time	tags
0	Ro6eob0LrCY	2017-11-14	Malika LePen : Femme de Gauche - Trailer	Le Raptor Dissident	24	2017-11- 13T17:32:55.000Z	Raptor""Dissident""Expliquez""moi""cette"...

Hacemos lo mismo para la columna `publish_time`

```
1 data['publish_time'] = pd.to_datetime(data['publish_time'], format='%Y-%m-%d').dt.date
2 data
```

	video_id	trending_date	title	channel_title	category_id	publish_time
0	Ro6eob0LrCY	2017-11-14	Malika LePen : Femme de Gauche - Trailer	Le Raptor Dissident	24	2017-11-13

Ahora verificaremos si existen datos faltantes.

```
1 data.isna().sum()
```

```
video_id      0
trending_date 0
title         0
channel_title 0
category_id   0
publish_time  0
tags          0
views         0
likes         0
dislikes      0
comment_count 0
thumbnail_link 0
comments_disabled 0
ratings_disabled 0
video_error_or_removed 0
description    2912
state         0
lat          0
lon          0
geometry      0
category_title 0
dtype: int64
```

Tal como se puede apreciar, todas las columnas, salvo `description`, están completas. Los vacíos en esta columna no suelen ser inusuales, dado que existe la posibilidad de que un video no tenga descripción. Sin embargo, es preferible llenar dichos vacíos con un `""`.

```
[ ] 1 data['description'].fillna("", inplace=True)
2 data
```

Verificaremos si existen datos atípicos.

```
[ ] 1 data['video_id'].describe()
```

```
count      40724
unique     30581
top        Rter-Np-Td0
freq        6
Name: video_id, dtype: object
```

Viendo la descripción de datos de **video_id**, podemos apreciar que hay un total de **10,143** videos repetidos en nuestro dataframe, y esto se debe gracias a la variable **trending_date**, por lo que nos quedaremos únicamente con el último registro de cada uno de los videos repetidos, que son aquellos con la información más actualizada de los mismos

Usaremos una función llamada **drop_duplicates**, se usará como criterio de comparación la columna **video_id** y se ingresará como parámetro **keep='last'** para conservar el último de cada uno.

```
[ ] 1 data.drop_duplicates(subset=['video_id'], keep='last', inplace=True)
2 data
```

Ahora comprobaremos que el número de vistas no sea diferente a la suma de likes y dislikes

```
[ ] 1 len(data[data['likes'] + data['dislikes'] > data['views']])
```

```
0
```

En el resultado de la celda a continuación se puede apreciar que hay varios títulos de video que se repiten. Sin embargo, al ser algo muy común en esta plataforma, se puede pasar por alto

```
1 data['title'].describe()
```

```
count      30581
unique     30375
top         J+1
freq        16
Name: title, dtype: object
```

En el caso de los canales también se pasa por alto, ya que un canal puede subir más de 1 video

```
[ ] 1 data['channel_title'].describe()
```

```
count      30581
unique     6670
top        VikatanTV
freq        161
Name: channel_title, dtype: object
```

Incluso, este caso de repetición también pueden darse con los thumbnails

```
[ ] 1 data['thumbnail_link'].describe()
```

```
count      30581
unique     30566
top        https://i.ytimg.com/vi/tynT8gBA6bc/default.jpg
freq        2
Name: thumbnail_link, dtype: object
```

Los siguientes resultados sí pueden llegar a estar puestos en interpretación. ¿Cuántas visitas necesita un video para poder llegar a ser tendencia?

```
1 data['views'].describe()
```

```
count      3.058100e+04
mean       3.458344e+05
std        1.539143e+06
min        2.840000e+02
25%        1.197500e+04
50%        5.351300e+04
75%        2.160520e+05
max        1.009116e+08
Name: views, dtype: float64
```

Se puede apreciar que existen videos que con apenas 300 visitas ingresan a tendencia, lo que se realizara sera unicamente quedarnos con aquellos videos que sean mayor al promedio de visitas de un video en tendencias.

Responder a esta pregunta es difícil ya que, para cada creador, el mínimo de vistas requerido puede variar drásticamente, así que optaremos por usar la media para simular un mínimo ficticio

```
[ ] 1 minViewsRequired = int(data['views'].describe()['mean'])
    2 minViewsRequired
```

345834

Ahora visualizaremos los videos con vistas por encima de dicho umbral

```
[ ] 1 data = data[data['views'] > minViewsRequired]
    2 data
```

	video_id	trending_date	title	channel_title	category_id	publish_time	tags	views	likes	dislikes	...	htt
28	exMNBwvCFRY	2017-11-14	LES PIRES DOUBLAGES FRANÇAIS	SQUEEZIE	23	2017-11-10	squeezie doublage["squeezie batman"]squeezie...	3641421	218587	5809	...	htt
31	Uq-eW_7xiAA	2017-11-14	Et Si Tout L'air Disparaissait Pendant 30 Seco...	Poisson Fécond	27	2017-11-11	chris conte["poisson fécond"]khundar["air"]...	567108	36790	574	...	h

También verificamos que la fecha en la que el video se volvió tendencia no sea menor a la fecha de publicación

```
[29] 1 len(data[data['trending_date'] < data['publish_time']])
```

0

En la siguiente celda podemos comprobar que existen algunos videos que han sido removidos. ¿Sería prudente eliminarlos?

```
[30] 1 len(data[data['video_error_or_removed'] == True])
```

4

La relacion entre los comentarios y su estado de desable son coherentes

```
[31] 1 dataCmnt = data[['comment_count', 'comments_disabled']]
    2
    3 print("Comments count > 0 but they're Enabled:", len(dataCmnt[(dataCmnt['comment_count'] > 0) & (dataCmnt['comments_disabled'] == False)]))
    4 print("Comments count = 0 but they're Enabled:", len(dataCmnt[(dataCmnt['comment_count'] == 0) & (dataCmnt['comments_disabled'] == False)]))
    5 print("Comments count > 0 but they're Disabled:", len(dataCmnt[(dataCmnt['comment_count'] > 0) & (dataCmnt['comments_disabled'] == True)]))
    6 print("Comments count = 0 but they're Disabled:", len(dataCmnt[(dataCmnt['comment_count'] == 0) & (dataCmnt['comments_disabled'] == True)]))
```

Comments count > 0 but they're Enabled: 5429
Comments count = 0 but they're Enabled: 5332
Comments count > 0 but they're Disabled: 0
Comments count = 0 but they're Disabled: 97

Aqui se puede apreciar que la cantidad de likes y dislikes en ocasiones a pesar de los ratings estar desactivados se mayores a 0, hay que tener en cuenta que un youtuber puede desactivar la calificacion o incluso los comentarios posterior a su fecha de subida, es muy comun

```
[32] 1 dataLD = data[['likes', 'dislikes', 'ratings_disabled']]
    2
    3 print("Likes > 0 and Ratings Enabled ->", len(dataLD[(dataLD['likes'] > 0) & (dataLD['ratings_disabled'] == False)]))
    4 print("Likes = 0 and Ratings Enabled ->", len(dataLD[(dataLD['likes'] == 0) & (dataLD['ratings_disabled'] == False)]))
    5 print("Likes > 0 and Ratings Disabled ->", len(dataLD[(dataLD['likes'] > 0) & (dataLD['ratings_disabled'] == True)]))
    6 print("Likes = 0 and Ratings Disabled ->", len(dataLD[(dataLD['likes'] == 0) & (dataLD['ratings_disabled'] == True)]))
    7 print()
    8 print("Dislikes > 0 and Ratings Enabled ->", len(dataLD[(dataLD['dislikes'] > 0) & (dataLD['ratings_disabled'] == False)]))
    9 print("Dislikes = 0 and Ratings Enabled ->", len(dataLD[(dataLD['dislikes'] == 0) & (dataLD['ratings_disabled'] == False)]))
   10 print("Dislikes > 0 and Ratings Disabled ->", len(dataLD[(dataLD['dislikes'] > 0) & (dataLD['ratings_disabled'] == True)]))
   11 print("Dislikes = 0 and Ratings Disabled ->", len(dataLD[(dataLD['dislikes'] == 0) & (dataLD['ratings_disabled'] == True)]))
```

Likes > 0 and Ratings Enabled -> 5375
Likes = 0 and Ratings Enabled -> 0
Likes > 0 and Ratings Disabled -> 0
Likes = 0 and Ratings Disabled -> 54

Dislikes > 0 and Ratings Enabled -> 5375
Dislikes = 0 and Ratings Enabled -> 0
Dislikes > 0 and Ratings Disabled -> 0
Dislikes = 0 and Ratings Disabled -> 54

5. Modelado y evaluación

Por Categoría de Videos

1. ¿Qué categorías de videos son las de mayor tendencia?

Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son la categoría y la cantidad de visitas. Agrupamos los videos respecto a su categoría y sumaremos la cantidad de visitas.

```
1 dataTrend = data.copy()[['category_title', 'views']]
2 byTrends = dataTrend.groupby('category_title').sum()
3 byTrends.sort_values('views', inplace=True, ascending=False)
4
5 byTrends
```

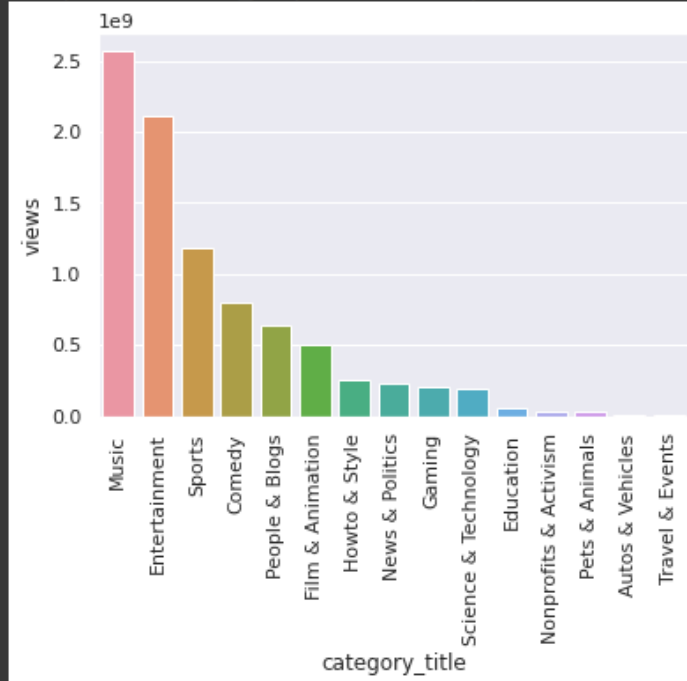
	views
category_title	
Music	2565875731
Entertainment	2108475351
Sports	1186105099
Comedy	805312605
People & Blogs	642343896
Film & Animation	496638184
Howto & Style	255509171
News & Politics	226701226
Gaming	201459321
Science & Technology	188424922
Education	51148632
Nonprofits & Activism	32091117
Pets & Animals	27061439
Autos & Vehicles	12804306
Travel & Events	2774005

```

1 byTrends.reset_index(inplace=True)
2 plt.xticks(rotation=90)
3 sns.barplot(x='category_title', y='views', data=byTrends)

```

<matplotlib.axes._subplots.AxesSubplot at 0x7fc6b6f788d0>



Se puede apreciar que las categorías de mayor tendencia son Música, entretenimiento y deportes.

2. ¿Qué categorías de videos son los que más gustan? ¿Y las que menos gustan?

Para los videos que más gustan:

Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son la categoría, la cantidad de likes. Agrupamos los videos respecto a su categoría y sumaremos la cantidad de likes.

```

2 dataL = data.copy()[['category_title', 'likes']]
3 byCategoryLikes = dataL.groupby('category_title').sum()
4 byCategoryLikes.sort_values('likes', inplace=True, ascending=False)
5
6 byCategoryLikes

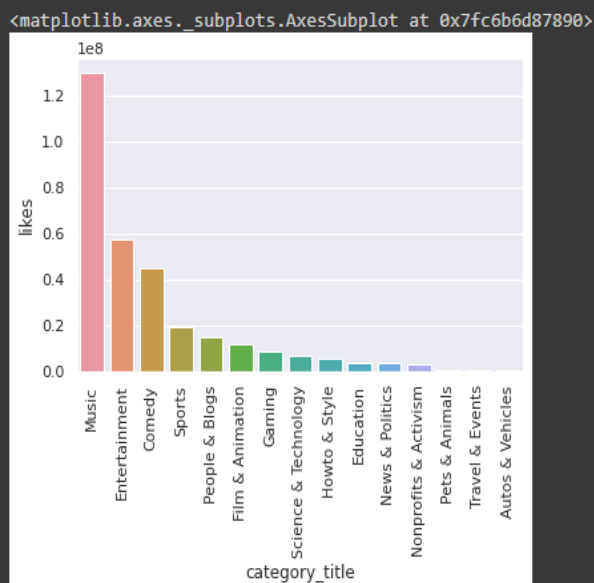
```

	likes
category_title	
Music	129399478
Entertainment	57382953
Comedy	44573622
Sports	19093586
People & Blogs	14998778
Film & Animation	11589477
Gaming	8263408
Science & Technology	6372109
Howto & Style	5093330
Education	3376352
News & Politics	3329905
Nonprofits & Activism	3144073
Pets & Animals	613375
Travel & Events	177898
Autos & Vehicles	165146

```

1 byCategoryLikes.reset_index(inplace=True)
2 plt.xticks(rotation=90)
3 sns.barplot(x='category_title', y='likes', data=byCategoryLikes)

```



Se puede apreciar que las categorías que tienen mayor cantidad de likes son música, entretenimiento y comedia.

Para los videos que menos gustan:

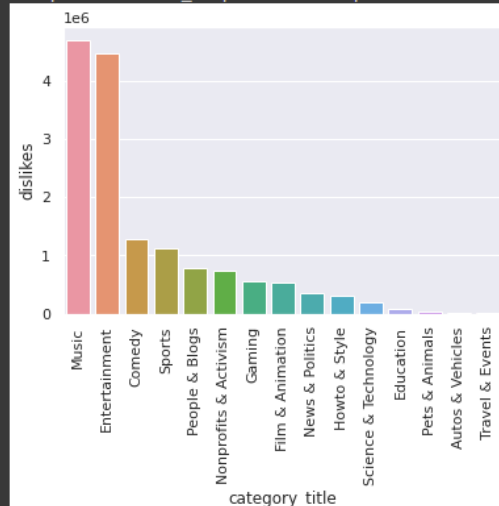
Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son la categoría, la cantidad de dislikes. Agrupamos los videos respecto a su categoría y sumaremos la cantidad de dislikes.

```
3 dataD = data.copy()[['category_title', 'dislikes']]
4 byCategoryDislikes = dataD.groupby('category_title').sum()
5 byCategoryDislikes.sort_values('dislikes', inplace=True, ascending=False)
6
7 byCategoryDislikes
```

	dislikes
category_title	
Music	4696360
Entertainment	4474380
Comedy	1275998
Sports	1117445
People & Blogs	784607
Nonprofits & Activism	741865
Gaming	547625
Film & Animation	544794
News & Politics	359116
Howto & Style	299615
Science & Technology	187808
Education	77091
Pets & Animals	24883
Autos & Vehicles	11771
Travel & Events	2251

```
1 byCategoryDislikes.reset_index(inplace=True)
2 plt.xticks(rotation=90)
3 sns.barplot(x='category_title', y='dislikes', data=byCategoryDislikes)
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fc6b6cca190>



Se puede apreciar que las categorías de los videos que menos gustan son los de música, entretenimiento y comedia.

3. ¿Qué categorías de videos tienen la mejor proporción (ratio) de “Me gusta” / “No me gusta”?

Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son la categoría, la cantidad de likes y la cantidad de dislikes, adicional a ello tendremos que agregar una columna que almacena el ratio de me gusta/no me gusta.

Primero tendremos que agrupar los videos por su categoría y sumar la cantidad de likes y dislikes, después podremos agregar la columna dividiendo la cantidad de likes y dislikes.

```
1 dataRatioLD = data.copy()[['category_title', 'likes', 'dislikes']]
2 byRatioLD = dataRatioLD.groupby('category_title').sum()
3 byRatioLD['ratio'] = byRatioLD['likes'] / byRatioLD['dislikes']
4 byRatioLD.sort_values('ratio', inplace=True, ascending=False)
5
6 byRatioLD
```

	likes	dislikes	ratio
category_title			
Travel & Events	177898	2251	79.030653
Education	3376352	77091	43.796967
Comedy	44573622	1275998	34.932360
Science & Technology	6372109	187808	33.928848
Music	129399478	4696360	27.553143
Pets & Animals	613375	24883	24.650364
Film & Animation	11589477	544794	21.273136
People & Blogs	14998778	784607	19.116294
Sports	19093586	1117445	17.086824
Howto & Style	5093330	299615	16.999583
Gaming	8263408	547625	15.089538
Autos & Vehicles	165146	11771	14.029904
Entertainment	57382953	4474380	12.824783
News & Politics	3329905	359116	9.272505
Nonprofits & Activism	3144073	741865	4.238066

Se puede apreciar que las categorías con mejor relación de me gusta/no me gusta son las de viajes y eventos, educación y comedia.

4. ¿Qué categorías de videos tienen la mejor proporción (ratio) de “Vistas” “Comentarios”?

Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son la categoría, la cantidad de visitas y la cantidad de comentarios,

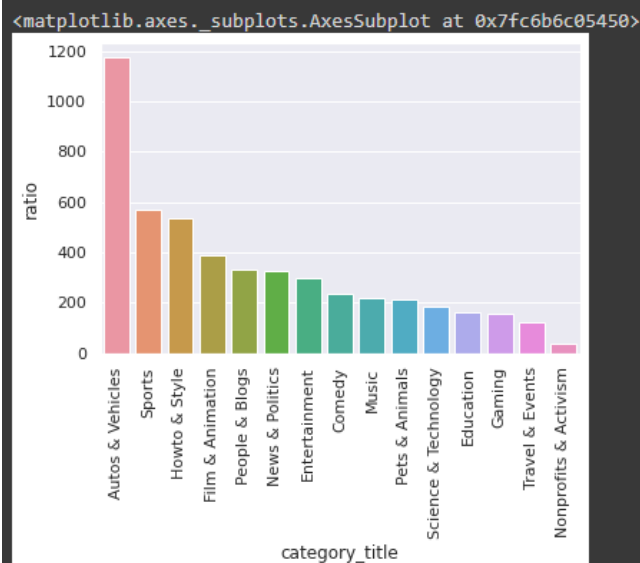
adicional a ello tendremos que agregar una columna que almacena el ratio de visitas/comentarios.

Primero tendremos que agrupar los videos por su categoría y sumar la cantidad de visitas y comentarios, después podremos agregar la columna dividiendo la cantidad de visitas y comentarios.

```
1 dataRatioVC = data.copy()[['category_title', 'views', 'comment_count']]
2 byRatioVC = dataRatioVC.groupby('category_title').sum()
3 byRatioVC['ratio'] = byRatioVC['views'] / byRatioVC['comment_count']
4 byRatioVC.sort_values('ratio', inplace=True, ascending=False)
5
6 byRatioVC
```

	views	comment_count	ratio
category_title			
Autos & Vehicles	12804306	10917	1172.877714
Sports	1186105099	2072692	572.253426
Howto & Style	255509171	477446	535.158261
Film & Animation	496638184	1285415	386.364080
People & Blogs	642343896	1935310	331.907496
News & Politics	226701226	701576	323.131387
Entertainment	2108475351	7066621	298.371082
Comedy	805312605	3452220	233.273837
Music	2565875731	11902613	215.572474
Pets & Animals	27061439	128673	210.311713
Science & Technology	188424922	1027271	183.422799
Education	51148632	311888	163.996794
Gaming	201459321	1295718	155.480838
Travel & Events	2774005	23117	119.998486
Nonprofits & Activism	32091117	910325	35.252374

```
1 byRatioVC.reset_index(inplace=True)
2 plt.xticks(rotation=90)
3 sns.barplot(x='category_title', y='ratio', data=byRatioVC)
```



Se puede apreciar que las categorías con mejor relación de visitas/comentarios son los de autos y vehículos, deportes y howto & style.

Por el tiempo transcurrido


5. ¿Cómo ha cambiado el volumen de los videos en tendencia a lo largo del tiempo?

Por Canales de YouTube

6. ¿Qué canales de YouTube son tendencia más frecuentemente?
¿Y cuáles con menos frecuencia?

Para poder resolver esta pregunta necesitaremos copiar únicamente del dataframe los datos que nos serán necesarios, y estos son el título del canal y la cantidad de visitas.

```
1 dataChannel = data.copy()[['channel_title', 'views']]
2 byChannelTrending = dataChannel.groupby('channel_title').sum()
3 byChannelTrending.sort_values('views', inplace=True, ascending=False)
4
5 byChannelTrending
```

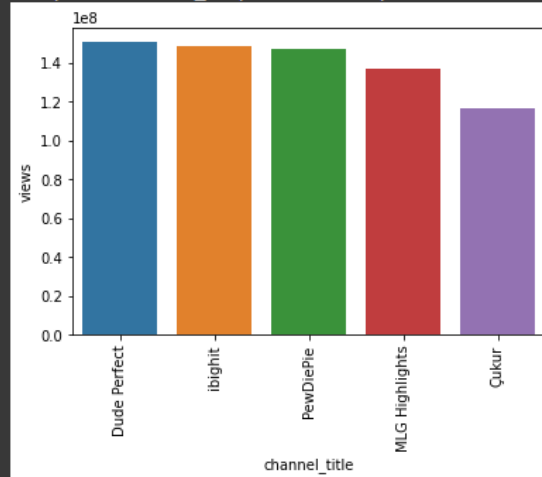
views 	
channel_title	
Dude Perfect	150548736
ibighit	148430214
PewDiePie	147319018
MLG Highlights	136466283
Çukur	116630925
...	...
Şahan Gökbakar	346794
Trailers FR	346682
AzealiaBanksVEVO	345691
Koolchi TV	345601
Siyah İnci	345512

1665 rows × 1 columns

Mayor frecuencia

```
1 byChannelTrending.reset_index(inplace=True)
2 plt.xticks(rotation=90)
3 sns.barplot(x='channel_title', y='views', data=byChannelTrending[:5])
```

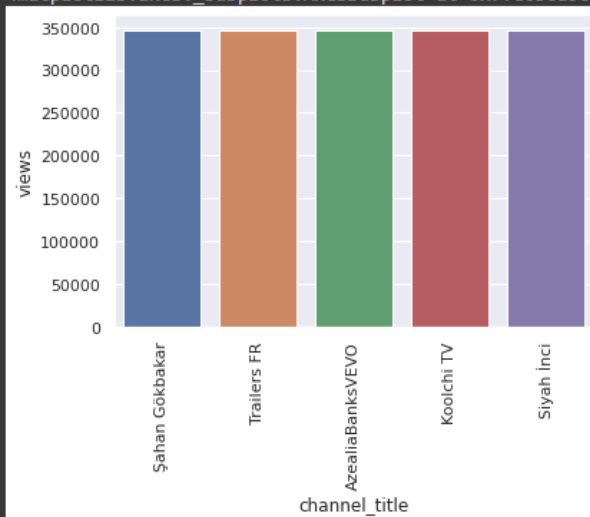
<matplotlib.axes._subplots.AxesSubplot at 0x7fee00d5d590>



Menor frecuencia

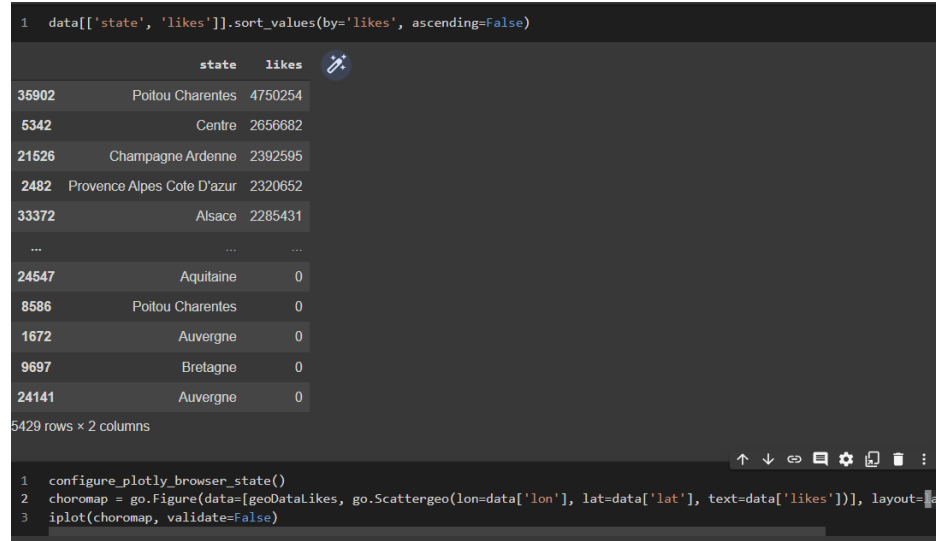
```
1 plt.xticks(rotation=90)
2 sns.barplot(x='channel_title', y='views', data=byChannelTrending[-5:])
```

<matplotlib.axes._subplots.AxesSubplot at 0x7fc6b6a96150>

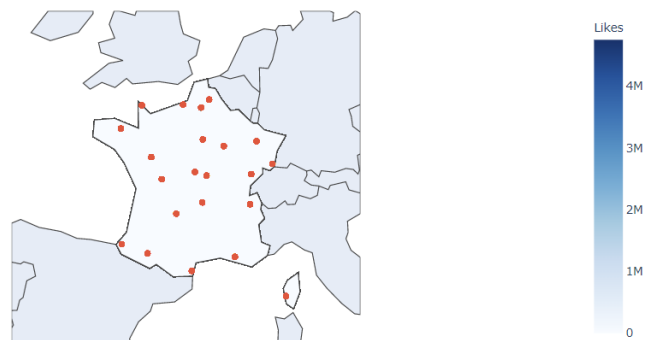


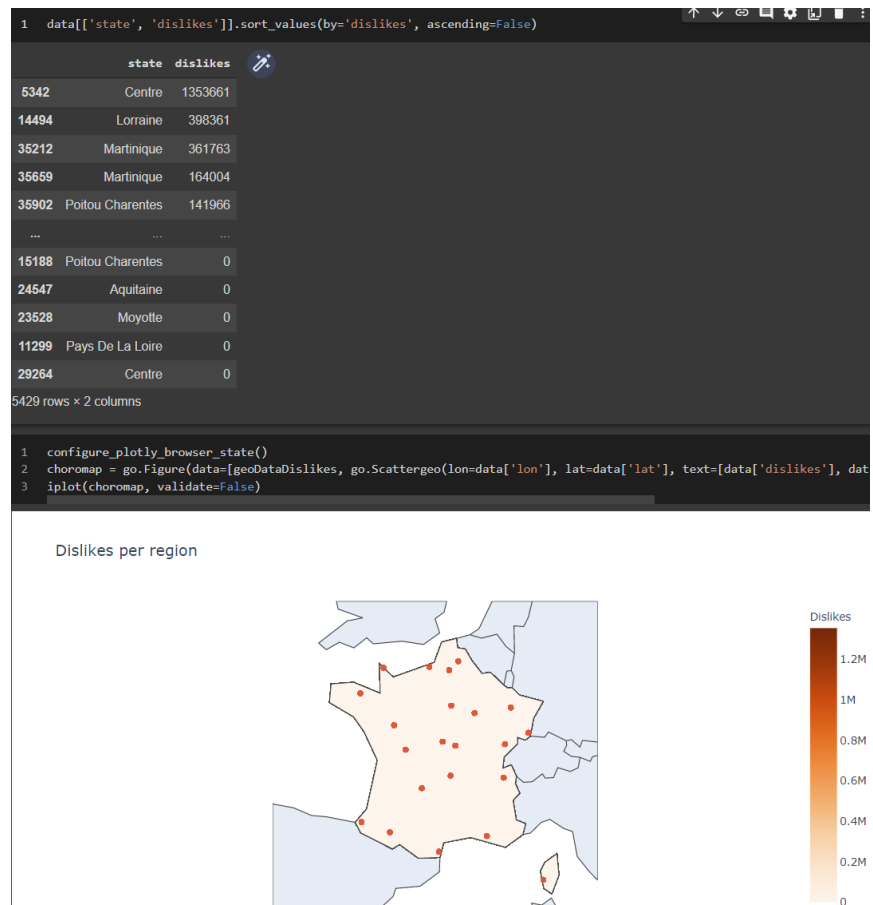
Por la geografía del país

7. ¿En qué Estados se presenta el mayor número de “Vistas”, “Me gusta” y “No me gusta”?



Likes per region





El estado con mayor número de likes es Poitou Charentes, mientras que el estado con mayor número de dislikes es Centre

Adicionalmente, al cliente le gustaría conocer si:

- ¿Es factible predecir el número de “Vistas” o “Me gusta” o “No me gusta”?

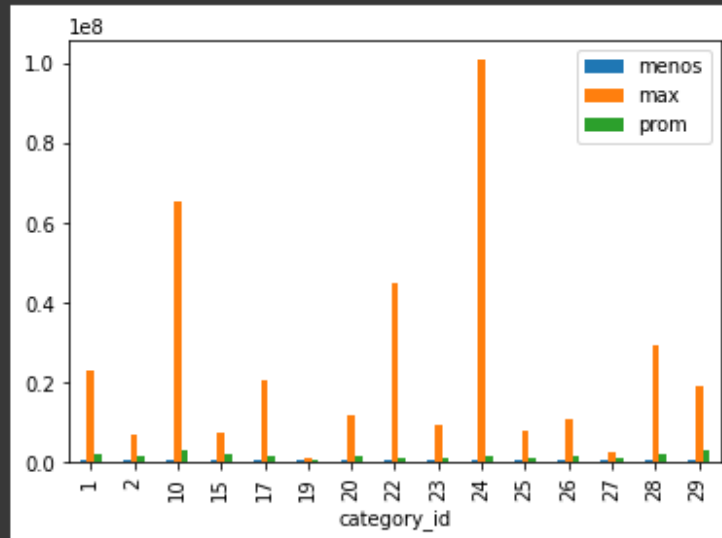
Primero obtendremos un dataframe que tendrá los datos por categoría de video la menor, mayor y cantidad de promedio de visitas.

- ¿Es factible predecir el número de "Vistas" o "Me gusta" o "No me gusta"?

```
1 dataViewsPred=data.copy()[['category_id','views']]
2 dataViewsPred.sort_values('category_id',ascending=True,inplace=True)
3
4 dataViewsPred=dataViewsPred.groupby('category_id')
5 dvp=pd.DataFrame()
6 dataMin=dataViewsPred['views'].min()
7
8 arrMinViews=dataMin
9
10 dvp['menos']=arrMinViews
11
12 dataMax=dataViewsPred['views'].max()
13 arrMaxViews=dataMax
14 dvp['max']=arrMaxViews
15
16 dataProm=dataViewsPred['views'].mean()
17 arrPromViews=dataProm
18 dvp['prom']=arrPromViews
19 dvp.reset_index(inplace=True)
20 print(dvp)
21
```

	category_id	menos	max	prom
0	1	346682	22762717	2.035402e+06
1	2	359952	6793122	1.422701e+06
2	10	345691	65396157	3.072905e+06
3	15	350853	7220717	1.804096e+06
4	17	346316	20761480	1.432494e+06
5	19	387325	1150661	5.548010e+05
6	20	346390	11880523	1.586294e+06
7	22	345512	44818108	1.249696e+06
8	23	347503	9407979	1.152092e+06
9	24	345601	100911567	1.336169e+06
10	25	347479	7906164	9.485407e+05
11	26	348045	10883016	1.511889e+06

```
35] 1 df=pd.DataFrame(dvp,columns=['category_id','menos','max','prom'])
     2 df.plot(x='category_id', y=['menos','max','prom'],kind='bar')
     3 plt.show()
```



- **¿Los videos en tendencia son los que mayor cantidad de comentarios positivos reciben?**
No existe una variable en nuestros datos que nos puedan indicar si un comentario recibido es positivo o no.

6. Conclusiones

El proyecto ha podido resolver necesidades que solicitaba la empresa consultora y se concluye lo siguiente:

- El análisis de estos datos puede ser beneficioso para la delimitación de categorías de vídeos objetivos a las cuales se les puede tomar como foco de anuncios a fin de generar un mayor alcance.
- Se establece que intentar predecir la cantidad de visitas, likes o dislikes no es recomendado, debido a que influyen muchas variables que no están incluidas en nuestros datos, el hecho de pertenecer a una categoría y subir un video no sera el motivo principal del éxito que pueda llegar a tener un video.

7. Repositorio

Enlace al repositorio del proyecto: <https://github.com/u201922331/EB-2022-1-CC50>