UPC
Universidad Peruana
de Ciencias Aplicadas

Ciencias de la Computación 2022-01

Administración de la Información

TRABAJO FINAL

Creación de conocimiento a partir de los datos en python

Integrantes:

Salinas Roca, Antonio - U201924931

Guillén Rojas, Daniel Carlos - U201920113

Wu Pan, Tito Peng - U201921200

Profesora:

Reyes Silva, Patricia Daniela

Lima, 2022

Índice

1. OBJETIVOS DEL PROYECTO

Crear conocimiento a partir de las características encontradas en videos de tendencia en YouTube, la plataforma de vídeos más grande del mundo. Esto se logrará respondiendo las preguntas mostradas:

Por Categoría de Videos

i. ¿Qué categorías de videos son las de mayor tendencia?
ii. ¿Qué categorías de videos son los que más gustan? ¿Y las que menos gustan?
iii. ¿Qué categorías de videos tienen la mejor proporción (ratio) de "Me gusta" / "No me gusta"?
iv. ¿Qué categorías de videos tienen la mejor proporción (ratio) de "Vistas" / "Comentarios"?

Por el tiempo transcurrido

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

Por Canales de YouTube

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

Por la geografía del país

vii. ¿En qué Estados se presenta el mayor número de "¿Vistas", "Me gusta" y "No me gusta"?

Adicionalmente, al cliente le gustaría conocer si:
¿Es factible predecir el número de "Vistas" o "Me gusta" o "No me gusta"?
¿Los videos en tendencia son los que mayor cantidad de comentarios positivos reciben?

2. CASO DE ANÁLISIS

El dataset que se utilizará fue elaborado por Mitchell J. extraído de la plataforma YouTube. Este conjunto de datos es un registro diario de los videos en tendencia que se encontraron en YouTube México. La fecha en que fueron extraídos los datos está acotada entre el 2017 y 2018, y está conformada por 20 variables.

El presente análisis mostraría las principales características que presentan los videos en tendencia y ahí aprovecharse para fines comerciales y/o educativos.

3. CONJUNTO DE DATOS

El dataset contiene las siguientes variables:

Nombre	Descripción
video_id	Identificador del vídeo
trending_date	Fecha en tendencia del vídeo
title	Título del vídeo
channel_title	Canal que subió el vídeo
category_id	Categoría del vídeo
publish_time	Fecha de publicación del vídeo
tags	Etiquetas asociadas al vídeo
views	Total, de vistas del vídeo
likes	Cantidad de me gusta del vídeo
dislikes	Cantidad de no me gusta del vídeo
comment_count	Cantidad de comentarios en el vídeo
thumbnail_link	Enlace de la miniatura del vídeo
comments_disabled	Indica si tiene comentarios desactivados
ratings_disabled	Indica si tiene los ratings desactivados
video_error_or_removed	Indica si el vídeo ha sido removido
description	Indica la descripción del vídeo
state	Estado en el cual se publicó el vídeo
lat	Indica la latitud geográfica del estado.
lon	Indica la longitud geográfica del estado.
geometry	Indica las coordenadas del estado.

4. ANÁLISIS EXPLORATORIO DE LOS DATOS

a. Cargar los datos

El dataset se encuentra en formato CSV, añadido a eso contábamos con un archivo tipo JSON que contenía los nombres de las categorías por id. Para empezar, se hizo un limpiado de datos en Rstudio para luego utilizarlos en nuestro proyecto.

```
MXvideos_original = pd.read_csv('cleaned_MXvideos.csv')
MXvideos_procesada = pd.read_csv('cleaned_MXvideos.csv')

MXvideos_procesada.info()
```

b. Inspeccionar los datos

Para la inspección de datos, primero mostramos los datos que se encuentran en el dataset.

```
MXvideos_procesada.head(20)
```

	video_id	trending_date	title	channel_title	category_id	publish_time	
0	SbOwzAI9ZfQ	17.14.11	Capítulo 12 | MasterChef 2017	MasterChef 2017	24	13/11/2017 1:06	Maste
1	klOV6Xh-DnI	17.14.11	ALEXA EX-INTEGRANTE DEL GRUPO TIMBIRICHE RENUN...	Micky Contreras Martinez	22	13/11/2017 0:11	
2	6L2ZF7Qzsbk	17.14.11	LOUIS CKAGÓ - EL PULSO DE LA REPÚBLICA	El Pulso De La República	25	13/11/2017 12:00	Chur
3	hcY52MFWMDM	17.14.11	Sismo de 6.7 sacude Costa Rica 12 Noviembre 2017	Casanare	25	12/11/2017 22:47	
4	_OXDcGPVAa4	17.14.11	DOG HACKS | MUSAS LESSLIE LOS POLINESIOS	Musas	26	13/11/2017 14:17	MUSA
5	Q9kK6NWZR1U	17.14.11	Asi se sintio Terremoto en iraq al bordo de ir...	MÚSICA & ENTRETENIMIENTO	10	12/11/2017 15:17	
6	c9VTD3n_IDs	17.14.11	La Resolana con el Capi | Programa 12 noviembr...	La Resolana	22	13/11/2017 2:00	la
7	XzULSsZYMRc	17.14.11	M6.7 Costa Rica Análisis de Terremotos en (((A...	concienciaradio	25	13/11/2017 1:18	terren

c. Pre-procesar los datos

Consideramos crear una nueva columna para la descripción de la categoría a partir de los archivos JSON que nos fue entregado.

```
MX_category_id = pd.read_json('MX_category_id.json')
category = {}
for item in MX_category_id["items"]:
  category[item["id"]] = item["snippet"]["title"]

#MXvideos_procesada["category"] = MXvideos_procesada["category_id"].map(category)
MXvideos_procesada["category"] = [category.get(str(id).replace(".0", "")) for id in MXvideos_proces
```

```
MXvideos_procesada.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 40451 entries, 0 to 40450
Data columns (total 21 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   video_id                40451 non-null  object
 1   trending_date           40451 non-null  object
 2   title                   40451 non-null  object
 3   channel_title           40451 non-null  object
 4   category_id             40451 non-null  int64
 5   publish_time            40451 non-null  object
 6   tags                    40451 non-null  object
 7   views                   40451 non-null  int64
 8   likes                   40451 non-null  int64
 9   dislikes                40451 non-null  int64
 10  comment_count           40451 non-null  int64
 11  thumbnail_link          40451 non-null  object
 12  comments_disabled       40451 non-null  object
 13  ratings_disabled        40451 non-null  object
 14  video_error_or_removed  40451 non-null  object
 15  description             36227 non-null  object
 16  state                   40451 non-null  object
 17  lat                     40451 non-null  float64
 18  lon                     40451 non-null  float64
 19  geometry                40451 non-null  object
 20  category                40199 non-null  object
dtypes: float64(2), int64(5), object(14)
memory usage: 6.5+ MB
```

Trabajamos las variables trending_date y publish_time como fechas y no como de tipo objeto. Ambas deben tener el mismo formato (yyyy-mm-dd o dd-mmyyyy). Adicionalmente, creamos una columna para la hora de publicación.

```
# Cambiando la columna trending_date a tipo datetime
MXvideos_procesada['trending_date'] = pd.to_datetime(MXvideos_procesada['trending_date'], format=

# Cambiando la columna publish_time a tipo datetime y creando la columna publish_hour
lsttime = []
lsthour = []
for i in range(len(MXvideos_procesada)):
  time, hour = MXvideos_procesada['publish_time'][i].split(" ")
  lsttime.append(time)
  lsthour.append(hour)

MXvideos_procesada['publish_hour'] = pd.to_datetime(lsthour, format="%H:%M")
MXvideos_procesada['publish_hour'] = MXvideos_procesada["publish_hour"].dt.strftime('%H:%M')
MXvideos_procesada['publish_date'] = pd.to_datetime(lsttime)
```

```
MXvideos_procesada.head(5)
```

	video_id	trending_date	title	channel_title	category_id	publish_time	
0	SbOwzAI9ZfQ	2017-11-14	Capítulo 12 | MasterChef 2017	MasterChef 2017	24	13/11/2017 1:06	
1	kIOV6Xh-DnI	2017-11-14	ALEXA EX-INTEGRANTE DEL GRUPO TIMBIRICHE RENUN...	Micky Contreras Martinez	22	13/11/2017 0:11	
2	6L2ZF7Qzsbk	2017-11-14	LOUIS CKAGÓ - EL PULSO DE LA REPÚBLICA	El Pulso De La República	25	13/11/2017 12:00	Chumel
3	hcY52MFWMDM	2017-11-14	Sismo de 6.7 sacude Costa Rica 12 Noviembre 2017	Casanare	25	12/11/2017 22:47	te
4	_OXDcGPVAa4	2017-11-14	DOG HACKS | MUSAS LESSLIE LOS POLINESIOS	Musas	26	13/11/2017 14:17	MUSAS|"I

5 rows × 23 columns

Para pre-procesar los datos primero verificamos los datos que poseen valores *NaN* o valores *None*.

```
def verificar_NaN(MXvideos_procesada):
  cont = 0
  for columna in MXvideos_procesada.columns:
    a = MXvideos_procesada[columna].isna()
    if sum(a) != 0:
      print(f"En la columna {columna} se han encontrado {sum(a)} valores NaN")
    else:
      cont += 1

  if cont == len(MXvideos_procesada.columns):
    print("No se han encontrado columnas con valores NaN")

def Nuevo_Valor(columna,df):
  b = randint(0,len(df)-1)
  while (df[columna].get(b) == np.nan or df[columna].get(b) == '[none]'):
    b = randint(0,len(df)-1)
  return df[columna].get(b)

def Modificar_valores_NaN(columna,df):
  df[columna].fillna(value= Nuevo_Valor(columna,df),inplace = True)
  return df[columna]

def Verificar_None(MXvideos_procesada):
  cont0 = 0
  for columna in MXvideos_procesada.columns:
    cont = 0
    for i in range(len(MXvideos_procesada)):
      if MXvideos_procesada[columna].get(i) == '[none]':
        cont += 1
    if cont != 0:
      print(f"En la columna {columna} se han encontrado {cont} valores None")
    else:
      cont0 += 1

  if cont0 == len(MXvideos_procesada.columns):
    print("No se han encontrado columnas con valores None")

def Remplazar_None(columna,df):
  for i in range(len(df[columna])):
      if df[columna].get(i) == '[none]':
        df.at[i, columna] = Nuevo_Valor(columna,df)
  return df[columna]
```

Mostramos las columnas las cuales contienen valores NaN.

```
#en esta parte eliminamos todas las filas que contengan valores NaN en la columna Video_id
MXvideos_procesada.dropna(axis=1)

#en esta parte verificamos que la columna Video_id ya no tenga valores NaN y cuales aun tiene
verificar_NaN(MXvideos_procesada)
print()

#Sabiendo que columnnas contienen los valores NaN procedemos a remplazarlos con la funcion Mo
#que remplaza cada valor con otro aleatorio de la columna siempre y cuando no sea otro NaN
MXvideos_procesada['category'] = Modificar_valores_NaN('category',MXvideos_procesada)
MXvideos_procesada['description'] = Modificar_valores_NaN('description',MXvideos_procesada)
verificar_NaN(MXvideos_procesada)
print()

#Ya que los [none] no se consideran valores NaN utilizaremos replace para remplazarlo con otr
Verificar_None(MXvideos_procesada)
MXvideos_procesada['tags'] = Remplazar_None('tags', MXvideos_procesada)
Verificar_None(MXvideos_procesada)
```

```
En la columna description se han encontrado 4224 valores NaN
En la columna category se han encontrado 252 valores NaN

No se han encontrado columnas con valores NaN

En la columna tags se han encontrado 7685 valores None
No se han encontrado columnas con valores None
```

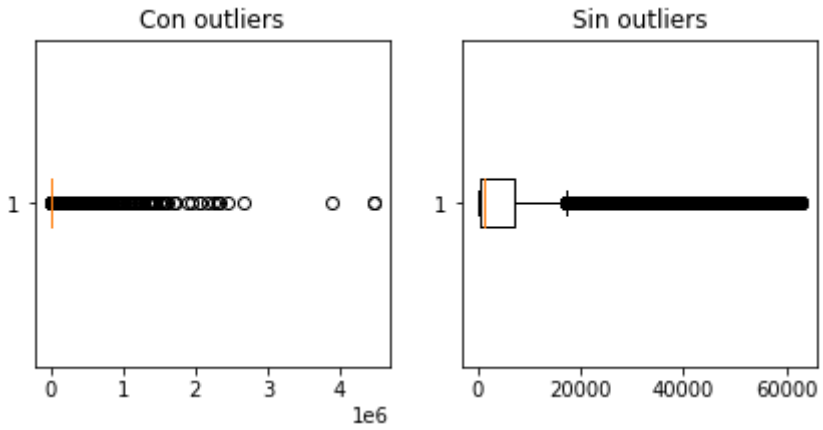
Al no encontrar que existen valores *NaN* o *nulos* en la sección *videos_id*, procedemos a crear una función para corregir valores errados.

```
def fix_outliers(df,columna):
  #calculamos los quantiles por arribadel 5% y por debajo del 95%
  media = (df[columna]).mean()
  mediana = (df[columna]).median()
  deciles = np.arange(1,20) *5
  deciles_dist = [np.percentile(df[columna], dec) for dec in deciles]
  for i in range(len(df[columna])):
    if (df[columna].get(i) <deciles_dist[0]):
      df.at[i,columna] = round(media)
    if (df[columna].get(i) >deciles_dist[len(deciles_dist) - 1]):
       df.at[i,columna] = round(mediana)
  return df[columna]
```

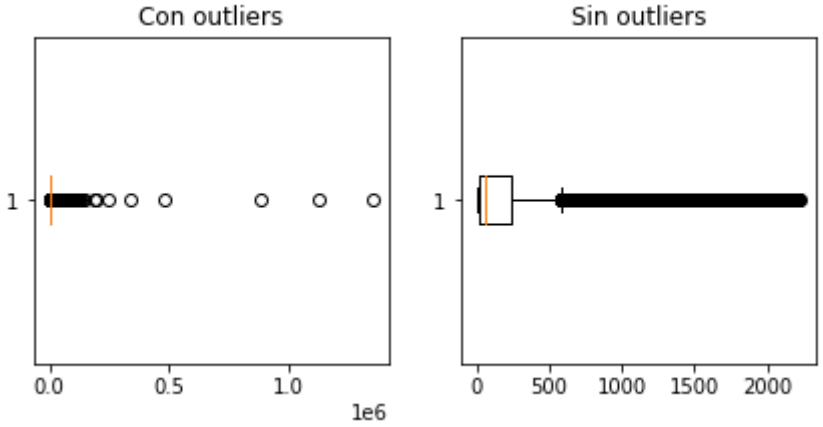
Elegimos las secciones likes, dislikes y views para comparar el antes y después de corregir errores contenidos en estas variables.

```
MXvideos_procesada['likes'] = fix_outliers(MXvideos_procesada,'likes')
MXvideos_procesada['dislikes'] = fix_outliers(MXvideos_procesada,'dislikes')
MXvideos_procesada['views'] = fix_outliers(MXvideos_procesada,'views')
```

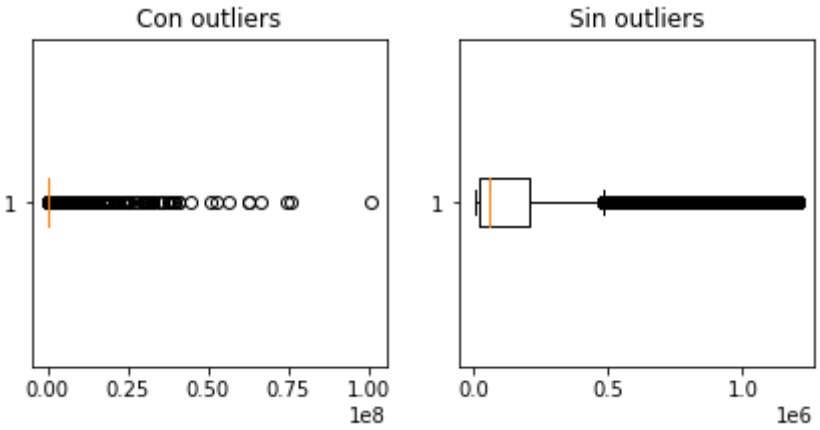
- Likes



- Dislikes



- Views



Por último, procedemos a guardar los datos corregidos

```
MXvideos_procesada.to_csv('MXvideos_cc50_procesada.csv')
```

d. Visualizar los datos

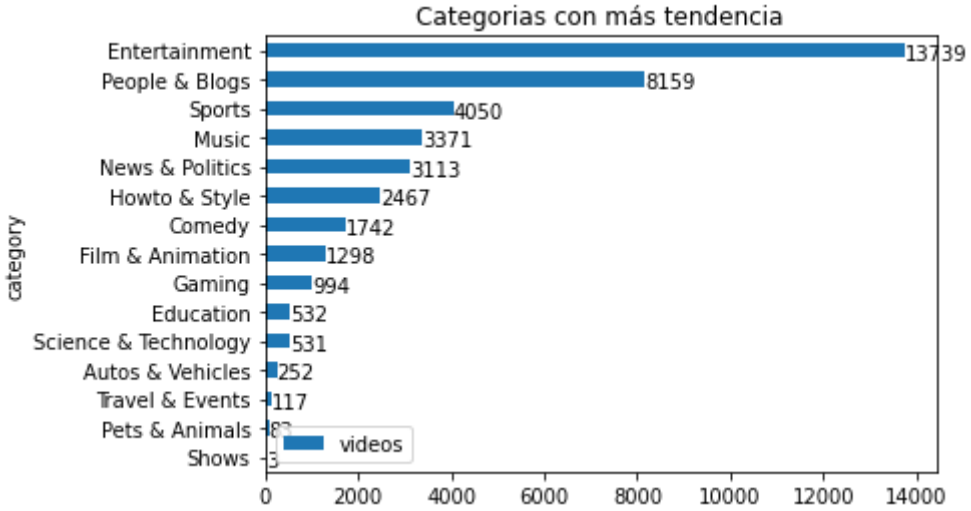
En esta sección se mostrará la solución a las preguntas que se formularon en los objetivos. Para la realización de esta sección utilizaremos librerías como matplotlib, pandas, numpy o cufflinks.

➔ *¿Qué categorías de videos son las de mayor tendencia?*

Para responder a esta pregunta tendremos que agrupar las categorías de los videos y ordenarlos de mayor a menor con respectos a los videos que contiene cada una en tendencia.

```
# ¿Qué categorías de videos son las de mayor tendencia?
dcat = df.groupby(by='category').size().reset_index(name='videos')
dcat = dcat.sort_values(by="videos")
ax = dcat.plot.barh(y='videos', x='category')
plt.title("Categorias con más tendencia")
for i, v in enumerate(dcat['videos']):
    plt.text(v, i-0.3, str(v))
```

Luego mostramos el gráfico.



Por último, mostramos los datos en una tabla.

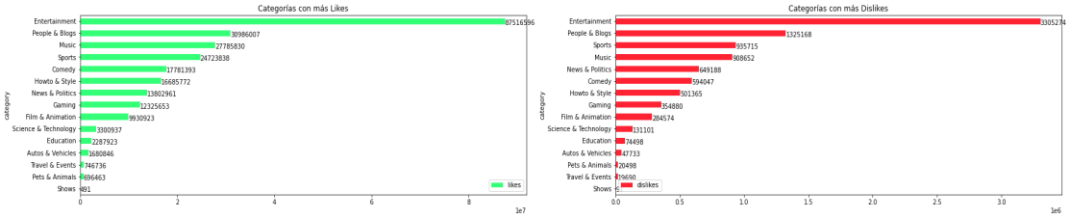
```
Entertainment          13739
People & Blogs          8159
Sports                  4050
Music                   3371
News & Politics         3113
Howto & Style           2467
Comedy                  1742
Film & Animation        1298
Gaming                   994
Education                532
Science & Technology     531
Autos & Vehicles         252
Travel & Events          117
Pets & Animals            83
Shows                      3
Name: category, dtype: int64
```

➔ *¿Qué categorías de videos son los que más gustan? ¿Y las que menos gustan?*

Para solucionar esta interrogante dividiremos en dos secciones, en likes y dislikes, y luego a cada uno será agrupada por categorías. Se ordenará de mayor a menor.

```
# ¿Qué categorías de videos son los que más gustan? ¿Y las que menos gustan?

dfc = df.copy()

fig = plt.figure(figsize=(18, 5))
ax1 = fig.add_subplot(121)

df_like = dfc.groupby(by='category').sum()['likes'].reset_index(name='likes')
df_dislike = dfc.groupby(by='category').sum()['dislikes'].reset_index(name='dislikes')

df_like = df_like.sort_values(by='likes')
df_dislike = df_dislike.sort_values(by='dislikes')

df_like.plot.barh(x='category', y='likes', title="Categorías con más Likes", ax=ax1,color='#33ff77')
for i, v in enumerate(df_like['likes']):
    plt.text(v, i-0.3, str(int(v)))
ax2 = fig.add_subplot(122)
df_dislike.plot.barh(x='category', y='dislikes', title="Categorías con más Dislikes", ax=ax2,color='#ff2233')
for i, v in enumerate(df_dislike['dislikes']):
    plt.text(v, i-0.3, str(int(v)))
```

Mostramos los gráficos obtenidos.



Por último, mostraremos también una tabla para entender mejor los datos.

category	
Entertainment	87516596
People & Blogs	30986007
Music	27785830
Sports	24723838
Comedy	17781393
Howto & Style	16685772
News & Politics	13802961
Gaming	12325653
Film & Animation	9930923
Science & Technology	3300937
Education	2287923
Autos & Vehicles	1680846
Travel & Events	746736
Pets & Animals	696463
Shows	491

Name: likes, dtype: int64

category	
Entertainment	3305274
People & Blogs	1325168
Sports	935715
Music	908652
News & Politics	649188
Comedy	594047
Howto & Style	501365
Gaming	354880
Film & Animation	284574
Science & Technology	131101
Education	74498
Autos & Vehicles	47733
Pets & Animals	20498
Travel & Events	19690
Shows	91

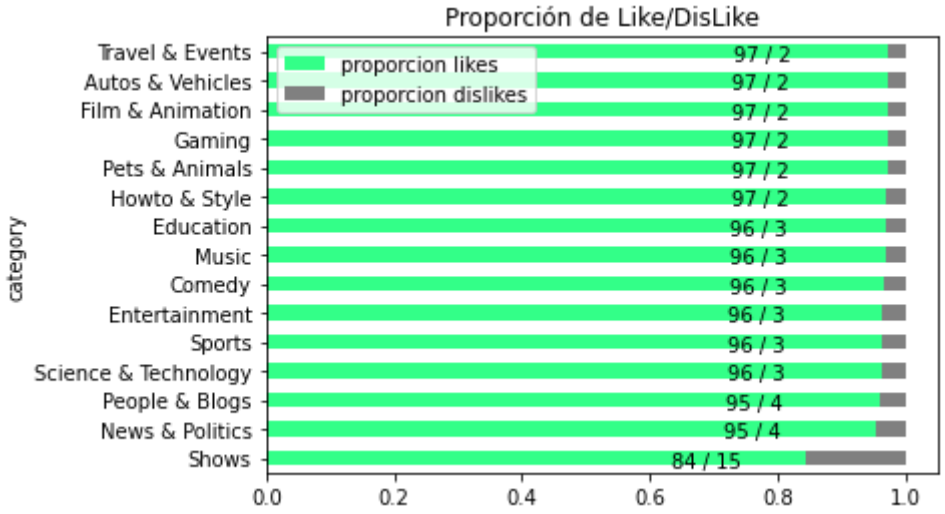
Name: dislikes, dtype: int64

➔ *¿Qué categorías de videos tienen la mejor proporción (ratio) de "Me gusta" / "No me gusta"?*

Dividiremos el dataset en likes y dislikes para luego compararlos y saber las proporciones con respecto al total de valoraciones.

```
# ¿Qué categorías de videos tienen la mejor proporción (ratio) de "Me gusta" / "No me gusta"?
dfc = df.copy()

d1 = dfc.groupby(by='category').sum()['likes'].reset_index(name='likes')
d2 = dfc.groupby(by='category').sum()['dislikes'].reset_index(name='dislikes')

d1['dislikes'] = d2['dislikes']

d1['proporcion likes'] = d1['likes']/(d1['likes'] + d1['dislikes'])
d1['proporcion dislikes'] = d1['dislikes']/(d1['likes'] + d1['dislikes'])


d1 = d1.sort_values(by='proporcion likes')

d1.plot.barh(x='category',
             y=['proporcion likes', 'proporcion dislikes'],
             color=['#33ff88', 'gray'],
             stacked=True,
             title="Proporción de Like/DisLike")
for i, v in enumerate(d1['proporcion likes']):
    plt.text(v*0.75, i-0.3, str(int(v*100))+" / "+str(int((1-v)*100)))
```

Mostramos el gráfico correspondiente.



Asimismo, mostramos una tabla con los datos agrupados.

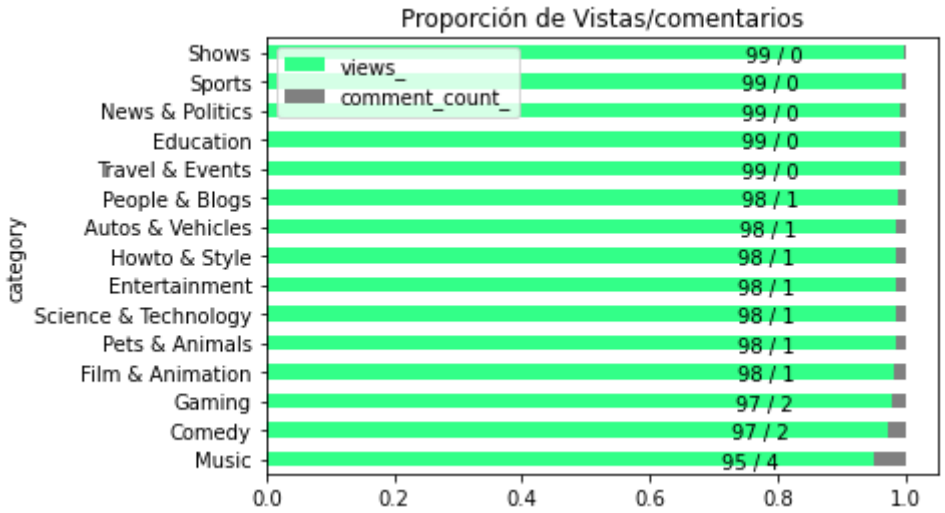
	category	likes	dislikes	proporcion likes	proporcion dislikes
14	Travel & Events	746736	19690	0.974309	0.025691
0	Autos & Vehicles	1680846	47733	0.972386	0.027614
4	Film & Animation	9930923	284574	0.972143	0.027857
5	Gaming	12325653	354880	0.972014	0.027986
10	Pets & Animals	696463	20498	0.971410	0.028590
6	Howto & Style	16685772	501365	0.970829	0.029171
2	Education	2287923	74498	0.968465	0.031535
7	Music	27785830	908652	0.968334	0.031666
1	Comedy	17781393	594047	0.967672	0.032328
3	Entertainment	87516596	3305274	0.963607	0.036393
13	Sports	24723838	935715	0.963533	0.036467
11	Science & Technology	3300937	131101	0.961801	0.038199
9	People & Blogs	30986007	1325168	0.958987	0.041013
8	News & Politics	13802961	649188	0.955080	0.044920
12	Shows	491	91	0.843643	0.156357

➔ *¿Qué categorías de videos tienen la mejor proporción (ratio) de "Vistas" / "Comentarios"?*

Agrupamos los datos en las secciones en vistas/comentarios con respecto a la categoría a la que pertenecen.

```
# ¿Qué categorías de videos tienen la mejor proporción (ratio) de "Vistas" / "Comentarios"?

dfc = df.copy()


dfc['views_'] = dfc['views'] / (dfc['comment_count']+dfc['views'])
dfc['comment_count_'] = dfc['comment_count'] / (dfc['comment_count']+dfc['views'])

dnx_ = dfc.groupby(by='category').mean()['views_'].reset_index(name='views_')
_ = dfc.groupby(by='category').mean()['comment_count_'].reset_index(name='comment_count_')
dnx_['comment_count_'] = _["comment_count_"]


dnx_ = dnx_.sort_values(by='views_')

dnx_.plot.barh(x='category', y=['views_', 'comment_count_'],
               color=['#33ff88', 'gray'],
               stacked=True,
               title="Proporción de Vistas/comentarios")
for i, v in enumerate(dnx_['views_']):
    plt.text(v*0.75, i-0.3, str(int(v*100))+" / "+str(int((1-v)*100)))
```

Mostramos un gráfico referente.



Asimismo, mostramos una tabla.

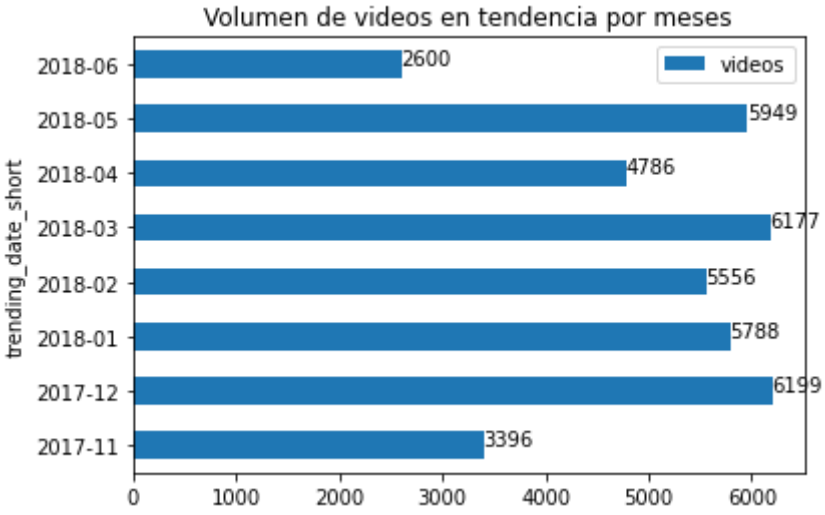
	category	views_	comment_count_
12	Shows	0.998521	0.001479
13	Sports	0.993159	0.006841
8	News & Politics	0.992554	0.007446
2	Education	0.991106	0.008894
14	Travel & Events	0.990275	0.009725
9	People & Blogs	0.989904	0.010096
0	Autos & Vehicles	0.985856	0.014144
6	Howto & Style	0.984948	0.015052
3	Entertainment	0.984642	0.015358
11	Science & Technology	0.984182	0.015818
10	Pets & Animals	0.983855	0.016145
4	Film & Animation	0.982725	0.017275
5	Gaming	0.977662	0.022338
1	Comedy	0.971434	0.028566
7	Music	0.951481	0.048519

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

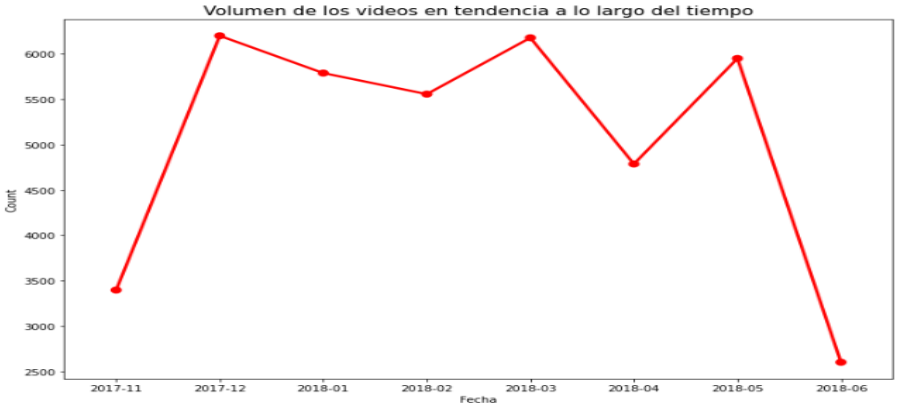
Para esta pregunta necesitaremos verificar cómo varían los videos en tendencias por año. Para ello, dividiremos los datos mediante las fechas en que fueron tendencia.

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

dfc = df.copy()

def testNumbers(x):
    return float(x[0]) > 2000

dfc = dfc[dfc['trending_date'].str.split("-").map(type) == list]
dfc = dfc[dfc['trending_date'].str.split("-").map(testNumbers)]
dfc['trending_date'] = pd.to_datetime(dfc['trending_date'], format="%Y-%m-%d")

dfc['trending_date_short'] = dfc['trending_date'].map(lambda x:"%d-%02d"%(x.year,x.month))


dfx = dfc.groupby(by='trending_date_short').size().reset_index(name='videos')
dfx = dfx.sort_values(by="trending_date_short")
dfx.plot.barh(x='trending_date_short', y='videos', title="Volumen de videos en tendencia por meses"
for i, v in enumerate(dfx['videos']):
    plt.text(v, i, str(int(v)))
```

Mostramos un gráfico de barras para saber las cantidades.



Asimismo, también podemos mostrarlo en un gráfico de líneas.



También mostramos una tabla con los datos.

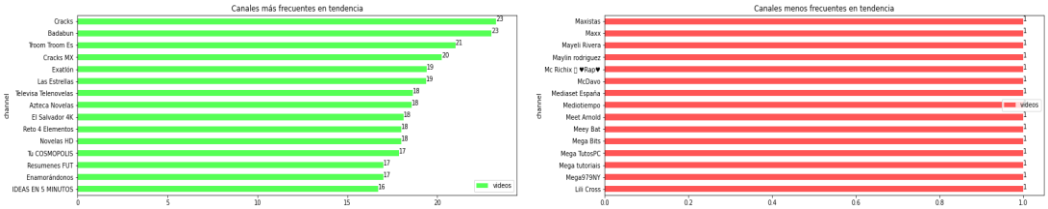
```
trending_date
2017-11    3396
2017-12    6199
2018-01    5788
2018-02    5556
2018-03    6177
2018-04    4786
2018-05    5949
2018-06    2600
Freq: M, Name: views, dtype: int64
```

➔ *¿Qué canales de YouTube son tendencia más frecuentemente? ¿Y cuáles con menos frecuencia?*

Para esto tomaremos los canales que posean videos que mayormente aparecen en tendencias y los canales con menor cantidad de videos con frecuencia en tendencia.

```
#¿Qué Canales de YouTube son tendencia más frecuentemente? ¿Y cuáles con menos frecuencia?

dfc = df.copy()

fig = plt.figure(figsize=(15, 5))
ax1 = fig.add_subplot(121)


def testNumbers(x):
    return float(x[0]) > 2000

dfc = dfc[dfc['trending_date'].str.split("-").map(type) == list]
dfc = dfc[dfc['trending_date'].str.split("-").map(testNumbers)]
dfc['trending_date'] = pd.to_datetime(dfc['trending_date'], format="%Y-%m-%d")

dfc['trending_date_short'] = dfc['trending_date'].map(lambda x:"%d-%02d"%(x.year,x.month))

dfc['channel_date'] = dfc['trending_date_short'] + "¶" + dfc['channel_title']

dfx = dfc.groupby(by='channel_date').size().reset_index(name='videos')

date, channel = zip(*dfx['channel_date'].str.split('¶'))
dfx['date'] = date
dfx['channel'] = channel

plt.rcParams['font.sans-serif']=['SimHei']

dfx = dfx.groupby(by='channel').mean()['videos'].reset_index(name='videos')
dfx = dfx.sort_values(by="videos")
dfx.tail(15).plot.barh(x='channel', y='videos', ax=ax1, color="#55ff55", title="Canales más frecuentes en tendencia")
for i, v in enumerate(dfx['videos'].tail(15)):
    plt.text(v, i, str(int(v)))

ax2 = fig.add_subplot(122)
dfx.head(15).plot.barh(x='channel', y='videos', ax=ax2, color="#ff5555", title="Canales menos frecuentes en tendencia")
for i, v in enumerate(dfx['videos'].head(15)):
    plt.text(v, i, str(int(v)))
```

Mostraremos su gráfica correspondiente.



Asimismo, mostramos los datos en dos tablas.

Cracks	186	FPFutebolOficial	1
Badabun	184	MohamdAssi Tv	1
Troom Troom Es	168	Phillipe LF MX	1
Cracks MX	162	The Anime Man	1
Las Estrellas	155	1theK (원더케이)	1
Televisa Telenovelas	149	Konohamaru The Eight	1
El Salvador 4K	145	Samsung US	1
Tu COSMOPOLIS	143	LuoKho Animation	1
Enamorándonos	136	Talking Tom and Friends Español	1
Ventaneando	133	EL procorito vlogs	1
Draw My Life en Español	131	MegaGlowen	1
Azteca Novelas	130	FueTT	1
Campechaneando	129	ECKO	1
TikTak Draw	128	Plano Invertido	1
Imagen Entretenimiento	126	Memes FUT	1
Name: channel_title, dtype: int64		Name: channel_title, dtype: int64	

➔ *¿En qué Estados se presenta el mayor número de "¿Vistas", "Me gusta" y "No me gusta"?*

Al no poder realizarse un mapa geográfico con la librería geopandas, solo mostraremos una tabla con los datos respectivos.

```
# ¿En qué Estados se presenta el mayor número de "Vistas", "Me gusta" y "No me gusta"?
dfc=df.copy()
data_state = dfc.groupby(dfc["state"]).sum()[["views", "likes", "dislikes"]]
data_state.reset_index(level=0, inplace=True)
```

```
states = dfc[["state","likes","views","dislikes"]]
statesMap = states.groupby(['state'], as_index=False)['likes','dislikes','views'].sum()
statesMap
```

	state	likes	dislikes	views
0	Aguascalientes	8024967	304649	205263001
1	Baja California	8146269	303996	200912249
2	Baja California Sur	7192867	274718	193485140
3	Campeche	8659190	328371	204245768
4	Chiapas	8021512	314216	214840441
5	Chihuahua	7647193	310111	205158940
6	Coahuila	7515544	285221	209300956
7	Colima	7567378	282294	205854423
8	Distrito Federal	7941841	291993	195938599
9	Durango	8017392	286450	201439056
10	Guanajuato	9009194	302195	211331116
11	Guerrero	7440796	277516	196554997
12	Hidalgo	7977851	263948	189868755
13	Jalisco	7877860	280736	185673872
14	Mexico	7770036	289601	199817346
15	Michoacan	8023836	306744	211203704
16	Morelos	7730334	300739	209396731
17	Nayarit	8088234	276689	203420458
18	Nuevo Leon	7860336	269870	186705558
19	Oaxaca	7409690	257274	189895371
20	Puebla	7323457	277636	188265138
21	Queretaro	8000944	272253	213596937
22	Quintana Roo	8085682	278445	185105918
23	San Luis Potosi	7540944	288906	202210042
24	Sinaloa	7187346	274284	189747097
25	Sonora	8562424	289953	207328997
26	Tabasco	7236377	275708	192171323
27	Tamaulipas	7535587	265170	207892337
28	Tlaxcala	7530046	273027	192486129
29	Veracruz	7760698	288909	195424427
30	Yucatan	7794895	275929	202889487
31	Zacatecas	7771649	284923	197406033

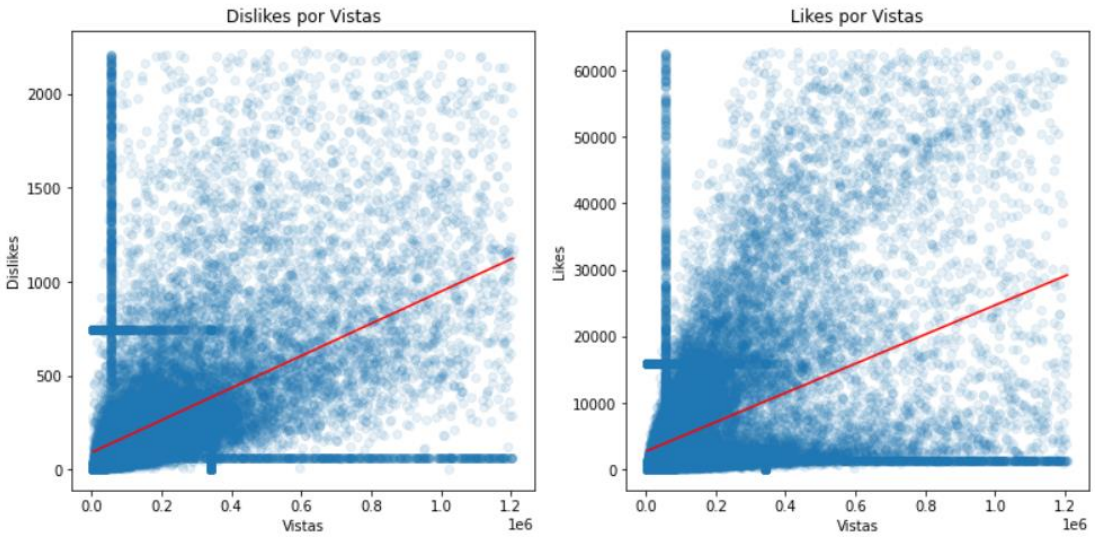
5. MODELIZAR Y EVALUAR LOS DATOS

Para esta sección buscamos identificar qué variables dentro del dataset son susceptibles a ser modeladas. A partir de ahí, podríamos obtener conocimiento acerca de este modelo.

En primer lugar, identificamos las variables que brindan información importante dentro de todo el conjunto de datos. Estas variables son las siguientes: “views”, “likes” y “dislikes”. Solo se tomarían en esas variables debido a que las demás no son del tipo numérico, por lo que no crearían una relación coherente con las mismas. Se omite “comment_count” ya que no existe una relación directa con los “views”.

Luego, procedemos a realizar una comparación de la relación de los valores de “likes” y “dislikes” con los “views”.

Obtenemos la siguiente gráfica:



A partir de esta gráfica podemos comentar que no sería posible realizar un modelo que pueda predecir la cantidad likes y dislikes por las visitas de un video en tendencia. Como ejemplo para llegar a esta afirmación, tenemos la comparación entre la cantidad de visitas de un video con la cantidad de likes: Podemos notar que un video con más visitas puede obtener una gran cantidad de likes, así como puede no tener muchos. Notamos el mismo comportamiento comparando las visitas con la cantidad de dislikes. Por lo que no es posible predecir la tendencia.

6. CONCLUSIONES DEL PROYECTO

Una vez concluido con la visualización de datos que responden las preguntas planteadas, se ha podido responder a cada una de ellas. A continuación, mostramos cada una de las preguntas trabajadas en el análisis.

¿Qué categorías de videos son las de mayor tendencia?

A partir del gráfico realizado, se obtuvo que las categorías con mayor cantidad de videos en tendencia fueron: Entertainment, People & Blogs y Sports. Estas son las 3 categorías con mayor cantidad de videos en tendencia, presentando 13739, 8159 y 4050 videos respectivamente.

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

En primer lugar, para la categoría de videos que más gustan, entre las 3 primeras encontramos a Entertainment, People & Blogs y Music, cada una con una cantidad total de 87516596, 30986007 y 27785830 likes respectivamente. Por otro lado, para la categoría de videos que menos gustan, en el top 3 tenemos a Entertainment, People & Blogs ySports, con una cantidad total de 3305274, 1325168 y 935715 dislikes respectivamente.

Pese a que la categoría Entertainment es la que mayor cantidad de videos en tendencia presenta, podemos observar que es la categoría que presenta una gran cantidad de likes y dislikes. Con ello podemos deducir que el disgusto de la audiencia hacia un video en específico también trae la atención de otros usuarios, por lo que un video con gran cantidad de dislikes también suele ser tendencia en internet.

¿Qué categorías de videos tienen la mejor proporción (ratio) de "Me gusta" / "No me gusta"?

Entre las categorías que presentan una mayor proporción de "Me gusta" y "No me gusta" tenemos las siguientes: Travel & Events, Autos & Vehicles y Film & Animation. Sus ratios son equivalentes a 97.4309 %, 97.2386 % y 97.2143 % respectivamente. Esto demuestra que videos de viajes y eventos, autos y películas captan en mayor proporción el agrado de los usuarios.

¿Qué categorías de videos tienen la mejor proporción (ratio) de "Vistas" / "Comentarios"?

Entre las categorías que presentan una mayor ratio de "views" y "comment_count" tenemos las siguientes: Gaming, Comedy y Music. Sus respectivas ratios son 0.977662, 0.971434 y 0.951481. Estos valores significan poca presencia de comentarios en videos acerca de películas, shows y deportes. Sin embargo, no suele significar algo negativo, pues un gran volumen de visitas muestra el interés por el contenido presentado, mas es irrelevante la necesidad de dejar un comentario dentro del video.

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

En el gráfico podemos observar dos puntos bajos en el volumen de videos en tendencia durante todo el periodo. Esto se debe a que las observaciones dentro del dataset comienzan desde el día 14 de noviembre del 2017 y terminan el junio del 2018, por lo que en estos meses no podemos obtener un volumen total, sino parcial. Omitiendo ello, podemos observar que mensualmente la cantidad de videos en tendencia se mantiene balanceado; sin embargo, en el mes de abril del 2018 sí podemos observar una caída en esa cuenta.

¿Qué Canales de YouTube son tendencia más frecuentemente? ¿Y cuáles con menos frecuencia?

Entre los canales que fueron tendencia con más frecuencia dentro del periodo comprendido en el dataset, tenemos a Cracks, que fue tendencia 186 veces; Badabun, 184 veces y Troom Troom Es, el cual fue tendencia 168 veces

Por otro lado, hay una gran cantidad de canales que solo fueron tendencia 1 vez durante este periodo. Dentro de esta lista tenemos canales como FPFutebolOficial, AKV. TVMohamdAssi Tv, Phillipe LF MX, entre otros.

¿En qué Estados se presenta el mayor número de "¿Vistas", "Me gusta" y "No me gusta"?

Con la realización de un gráfico de barras, podemos determinar rápidamente qué estados presentan un mayor volumen de videos vistos, con likes y con dislikes.

Con respecto a los estados que presentan un mayor número de "views", tenemos a Aguascalientes, Baja California y Baja California Sur, con 205263001, 200912249 y 193485140 de views totales respectivamente. Los estados que presentan un mayor número de likes en videos, en total, tenemos a Guanajuato, Campeche y Sonora, con 9009194, 8659190 y 8562424 cantidad de likes totales respectivamente. Por último, entre los estados que suman una mayor cantidad de dislikes en los videos, tenemos a Campeche, Aguascalientes y Baja California, con 328371, 304649 y 303996 cantidad de dislikes totales respectivamente.

Podemos afirmar que el estado de Hamburg cuenta con una gran cantidad de usuarios activos en la plataforma de Youtube, pues se encuentra en el top 3 dentro de estos casos.

7. ARCHIVAR Y PUBLICAR

Para la revisión del repositorio donde se encuentra almacenado toda la codificación revisar el siguiente enlace:

https://github.com/antoniosalinas2000/EB-2022-1-CC50.git