Psychoinformatics - Week 3 (Exercises)

by Jake Luke Harrison (B09207038@ntu.edu.tw)

1 Analyze what videos go viral? (8 points)

Please use YouTube APIs to carry out a data-driven or hypothesis-driven microstudy about the characteristics of viral videos.

You need to present, here in this notebook, AT LEAST two **statistical** figures or tables as supporting evidence for your arguments. Each of these figures/tables deserves 4 points.

In [ ]:
```
# Please carry out your analysis here
```

Setup

In [ ]:
```
pip install --upgrade google-api-python-client
```

In [ ]:
```
import os
from googleapiclient.discovery import build
from apiclient.discovery import build
from google.auth.transport import Response
import google_auth_oauthlib.flow
import googleapiclient.discovery
import googleapiclient.errors

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import random

from scipy import stats # for t-test
from textblob import TextBlob # for sentiment analysis
import datetime # for printing the time a cell is run
```

In [ ]:
```
api_key = 'AIzaSyBsMWP-VjnY-tTPOj19YQ-FsqMF3MWA1kY'

youtube = build(
    'youtube', # API service name
    'v3', # API version
    developerKey = api_key # my API key
)
```

Popular Videos in the United Kingdom

To explore and familiarise myself with the YouTube API, I retrieve a sample of 50 of the most popular videos in the United Kingdom.

I learn that this sample is of the dictionary type. I inspect the first item to understand the structure of this dictionary.

YouTube is constantly updated, so I print the current time and date to contextualise findings.

In [ ]:
```python
most_popular_GB = youtube.videos().list(
    part = 'snippet, contentDetails, statistics',
    chart = 'mostPopular', # most popular videos
    regionCode = 'GB', # United Kingdom
    maxResults = 50).execute()

type(most_popular_GB) # dictionary

items = most_popular_GB.get('items', [])  # 'items' is a key in the dictionary

items[0] # inspect first item in items

print('\nTime at completion:', datetime.datetime.now()) # show current date and time
```

```
Time at completion: 2023-09-26 02:37:08.627392
```

In the next two cells, I inspect the entire sample of 50 popular videos in the United Kingdom. I limit this inspection to the title, views, likes, and comments of each video.

I also perform simple sentiment analyses on the video descriptions and display the results. Video descriptions can be lengthy and a simple sentiment analysis can quickly convey, with a degree of accuracy, the tone—sentiment—of a description.

In [ ]:
```python
# Inspect sample of most popular videos

for item in items:

  # Create sub-dictionaries
  snippet = item.get('snippet', {})                    # video metadata
  statistics = item.get('statistics', {})              # video statistics

  # Access specific information in sub-dictionaries
  title = snippet.get('title')                         # title
  view_count = statistics.get('viewCount')             # views
  like_count = statistics.get('likeCount')             # likes
  comment_count = statistics.get('commentCount')       # comments
  description = snippet.get('description')             # description


  # Perform simple sentiment analysis on video description
  analysis = TextBlob(description)
  sentiment = 'Neutral'
  if analysis.sentiment.polarity > 0:
      sentiment = 'Positive'
  elif analysis.sentiment.polarity < 0:
      sentiment = 'Negative'

  print('Video Title: ', title, '\n',
        'View Count: ', view_count, '\n',
        'Like Count: ', like_count, '\n',
        'Comment Count: ', comment_count, '\n',
        'Description Sentiment: ', sentiment, '\n',
        # 'Description: ', description, '\n' # uncomment to view descriptions
        )
```

Video Title: SIDEMEN $100,000 MYSTERY BOX CHALLENGE (YOUTUBER EDITION)
 View Count: 5050428
 Like Count: 235966
 Comment Count: 5444
 Description Sentiment: Positive

Video Title: Elite 1 Squad Battles Rewards PAID OUT HUGE! - FC24 Road to Glory
 View Count: 158157
 Like Count: 3369
 Comment Count: 136
 Description Sentiment: Positive

Video Title: I Packed My FIRST WALKOUT On The RTG!
 View Count: 455172
 Like Count: 18437
 Comment Count: 617
 Description Sentiment: Positive

Video Title: HIGHLIGHTS- Wales v Australia- 2023 Rugby World Cup
 View Count: 209191
 Like Count: 1919
 Comment Count: 682
 Description Sentiment: Neutral

Video Title: Travelling EUROPE Completely Solo In My Truck - EP.2
 View Count: 349349
 Like Count: 22029
 Comment Count: 1777
 Description Sentiment: Positive

Video Title: Emotional Eddie Jones reacts to huge Rugby World Cup loss to Wales
 View Count: 553383
 Like Count: 2884
 Comment Count: 2424
 Description Sentiment: Positive

Video Title: 🇨🇳 **BIG BANG ZHANG DOES THE DOUBLE!** 💥 | Zhilei Zhang vs Joe Joyce Fight Highlights | #ZhangJoyce2
 View Count: 1578696
 Like Count: 14435
 Comment Count: 6118
 Description Sentiment: Positive

Video Title: IVE 아이브 'Either Way' MV
 View Count: 7603055
 Like Count: 330213
 Comment Count: 18802
 Description Sentiment: Neutral

Video Title: How much 'Titanium' does iPhone 15 Pro *actually* have? - NO SECRETS HERE!
 View Count: 4231696
 Like Count: 149120
 Comment Count: 6798
 Description Sentiment: Positive

Video Title: Kane nets first HAT-TRICK for Bayern 🔥 | Bayern Munich 7-0 Bochum | Bundesliga Highlights
 View Count: 1274292
 Like Count: 12010
 Comment Count: 1822
 Description Sentiment: Positive

Video Title: Arsenal 2-2 Tottenham | It Felt Like A Loss! (Lee Judges)
 View Count: 205789
 Like Count: 3263
 Comment Count: 1054
 Description Sentiment: Positive

Video Title: Our Engagement, My 30th Birthday & Greece Holiday
 View Count: 498637
 Like Count: 28295
 Comment Count: 339
 Description Sentiment: Positive

Video Title: Using 1 Item to Completely Break This Entire Game - Sunkenland
 View Count: 1698630
 Like Count: 84321
 Comment Count: 3509
 Description Sentiment: Negative

Video Title: I Actually Bought 100 Tiktok Shop Products
 View Count: 1501080
 Like Count: 47847
 Comment Count: 2637
 Description Sentiment: Positive

Video Title: Newcastle's BIGGEST EVER league away win! 🤯🔥 | Sheffield United 0-8 Newcastle | Highlights
 View Count: 1316037
 Like Count: 16969
 Comment Count: 2093
 Description Sentiment: Negative

Video Title: Saka and Son star in THRILLING North London Derby! 🍿 | Arsenal 2-2 Tottenham | Highlights
 View Count: 4215368
 Like Count: 54453
 Comment Count: 4859
 Description Sentiment: Positive

Video Title: Núñez scores VOLLEY as Liverpool move up to second! 🔴 | Liverpool 3-1 West Ham | Highlights
 View Count: 1437800
 Like Count: 14715
 Comment Count: 1165
 Description Sentiment: Positive

Video Title: This life-sized pop pop boat actually works
 View Count: 728004
 Like Count: 25341
 Comment Count: 1427
 Description Sentiment: Positive

Video Title: OFFICIAL TRAILER | Doctor Who 60th Anniversary Specials | Doctor Who
 View Count: 1389619
 Like Count: 67541
 Comment Count: 6636
 Description Sentiment: Positive

Video Title: PARIS SAINT-GERMAIN - OLYMPIQUE DE MARSEILLE (4 - 0) - Highlights - (PSG - OM) / 2023/2024
 View Count: 707965
 Like Count: 12525
 Comment Count: 339
 Description Sentiment: Neutral

Video Title: Kep1er 케플러 | 'Galileo' M/V
 View Count: 3018844
 Like Count: 131698
 Comment Count: 9347
 Description Sentiment: Neutral

Video Title: GENERAL KNOWLEDGE QUIZ W/ CHUNKZ, HARRY PINERO & KONAN
 View Count: 551026
 Like Count: 30384
 Comment Count: 460
 Description Sentiment: Positive

Video Title: 120hrs on Orient Express Luxury Sleeper Train | Paris - Istanbul
View Count: 333138
Like Count: 12725
Comment Count: 825
Description Sentiment: Positive

Video Title: ISHOWSPEED: Sundae Conversation with Caleb Pressley
View Count: 1084295
Like Count: 56560
Comment Count: 2684
Description Sentiment: Neutral

Video Title: HIGHLIGHTS - South Africa v Ireland - 2023 Rugby World Cup
View Count: 1162543
Like Count: 8303
Comment Count: 2053
Description Sentiment: Neutral

Video Title: SPEND THE DAY WITH ME | AUTUMN B&M HAUL | ZOE HAGUE
View Count: 72861
Like Count: 2059
Comment Count: 85
Description Sentiment: Positive

Video Title: I GOT BEHZINGA PREGNANT
View Count: 138122
Like Count: 8746
Comment Count: 113
Description Sentiment: Positive

Video Title: FULL FIGHT | Oogway vs. Armz Korleone (X Series 009)
View Count: 471834
Like Count: 17141
Comment Count: 2535
Description Sentiment: Positive

Video Title: Our Big Decision: We've SOLD Our House
View Count: 135363
Like Count: 4431
Comment Count: 262
Description Sentiment: Positive

Video Title: THE START OF OUR RTG! (#1)
View Count: 749236
Like Count: 28363
Comment Count: 1243
Description Sentiment: Positive

Video Title: Angela & Kai Cha Cha to Get The Party Started by Shirely Bassey ✨ BBC Strictly 2023
View Count: 378994
Like Count: 1232
Comment Count: 175
Description Sentiment: Positive

Video Title: 6 Months Pregnant Living in the Woods
View Count: 482320
Like Count: 33104
Comment Count: 2382
Description Sentiment: Positive

Video Title: Nigel Harman and Katya Jones Paso Doble to Smells Like Teen Spirit by Nirvana ✨ BBC Strictly 2023
View Count: 198755
Like Count: 2079
Comment Count: 275
Description Sentiment: Positive

Video Title: Race Highlights | 2023 Japanese Grand Prix
 View Count: 5933869
 Like Count: 119467
 Comment Count: 5201
 Description Sentiment: Positive

Video Title: 48 Hours In AMSTERDAM (I Fell In LOVE) - SOLO TRAVEL DIARIES
 View Count: 44475
 Like Count: 2919
 Comment Count: 161
 Description Sentiment: Neutral

Video Title: Gothic Beach 🖤 Palette & Collection Reveal! | Jeffree Star Cosmetics
 View Count: 407113
 Like Count: 37279
 Comment Count: 4330
 Description Sentiment: Positive

Video Title: I Tried The World's Fastest Vehicles
 View Count: 3852613
 Like Count: 159486
 Comment Count: 7861
 Description Sentiment: Positive

Video Title: Highlights | Leeds United 3-0 Watford | Piroe scores again!
 View Count: 198573
 Like Count: 2614
 Comment Count: 322
 Description Sentiment: Positive

Video Title: GUESS THE SINGER FT BURNA BOY
 View Count: 3432839
 Like Count: 189524
 Comment Count: 7920
 Description Sentiment: Positive

Video Title: We Attempted to Create the BEST Pokemon Fusion Team!
 View Count: 226695
 Like Count: 11920
 Comment Count: 576
 Description Sentiment: Negative

Video Title: Resumen de FC Barcelona vs RC Celta (3-2)
 View Count: 3495878
 Like Count: 78221
 Comment Count: 2756
 Description Sentiment: Positive

Video Title: Are Tools from TEMU Worth Considering?
 View Count: 159258
 Like Count: 3617
 Comment Count: 348
 Description Sentiment: Positive

Video Title: I Broke 1,487,000 Blocks Under my Enemy
 View Count: 321334
 Like Count: 18743
 Comment Count: 1510
 Description Sentiment: Positive

Video Title: why does tik tok literally have the BEST ramen hacks???
 View Count: 580110
 Like Count: 44483
 Comment Count: 1627
 Description Sentiment: Positive

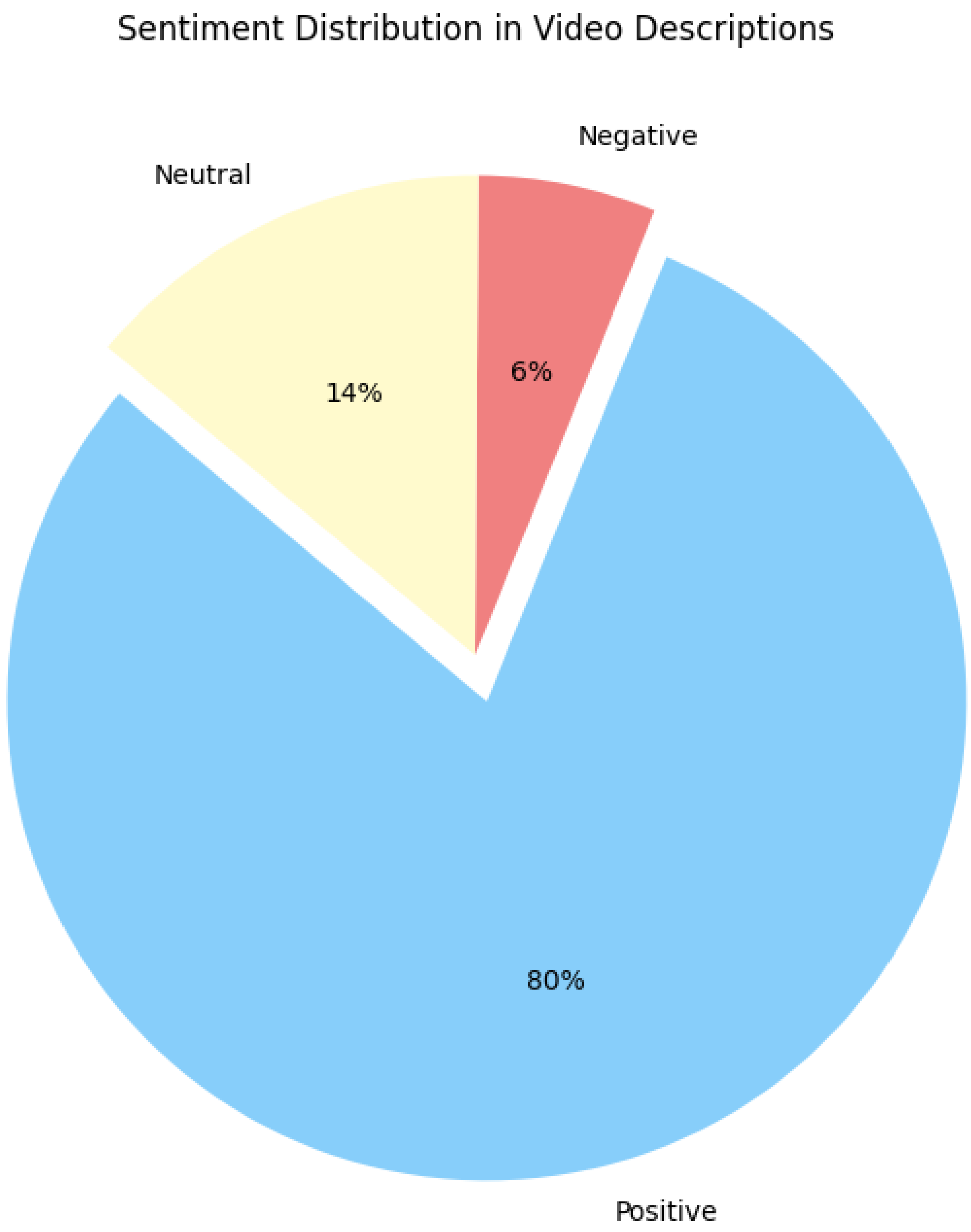
Video Title: "I'm not done yet!" Joe Joyce brutally honest following Zhang defeat | Eyes bout with Dubois
 View Count: 131021

```
 Like Count:  1138
 Comment Count:  228
 Description Sentiment:  Positive

Video Title:  FULL CARD HIGHLIGHTS | Virgo vs. Chalmers (X Series 009)
 View Count:  225571
 Like Count:  5230
 Comment Count:  411
 Description Sentiment:  Positive

Video Title:  HIGHLIGHTS | Atlético Madrid 3-1 Real Madrid | Morata and Griezmann sco
re in derby win for Atleti
 View Count:  356563
 Like Count:  3214
 Comment Count:  574
 Description Sentiment:  Positive

Video Title:  Be gentle with Apples new Titanium iPhone 15 Pro Max ... Yikes!
 View Count:  7019967
 Like Count:  228668
 Comment Count:  11219
 Description Sentiment:  Positive

Video Title:  Conor Benn's First Ringwalk In 525 Days
 View Count:  74312
 Like Count:  245
 Comment Count:  164
 Description Sentiment:  Positive

Video Title:  Uncle Roger vs Joshua Weissman (Flavor vs Texture)
 View Count:  868487
 Like Count:  46254
 Comment Count:  2321
 Description Sentiment:  Positive
```

In [ ]:

```
# Visualisation of sentiment analysis

# Initialise counters
positive_count = 0
negative_count = 0
neutral_count = 0

# Sentiment analysis
for item in items:
  snippet = item.get('snippet', {})                    # video metadata
  description = snippet.get('description')             # description

  analysis = TextBlob(description)
  sentiment = 'Neutral'
  if analysis.sentiment.polarity > 0:
    sentiment = 'Positive'
    positive_count += 1
  elif analysis.sentiment.polarity < 0:
    sentiment = 'Negative'
    negative_count += 1
  else:
    neutral_count += 1

# Visualisation of sentiment distribution in video descriptions
sentiment_labels = ['Positive', 'Negative', 'Neutral']
sentiment_counts = [positive_count, negative_count, neutral_count]

plt.figure(figsize=(8, 8))
plt.pie(sentiment_counts, labels=sentiment_labels,
        explode = (0.1, 0, 0,), textprops = {'fontsize': 10},
        autopct='%1.0f%%',
        # List of colour names: https://shorturl.at/jlqw5
        colors=['lightskyblue', 'lightcoral', 'lemonchiffon'],
```

```
        startangle=140)
plt.title('Sentiment Distribution in Video Descriptions')
plt.show()
```



It seems the descriptions of popular videos in the United Kingdom tend to be characterised by a positive sentiment.

In the next two cells, I would like to inspect the most common words in the titles of these videos.

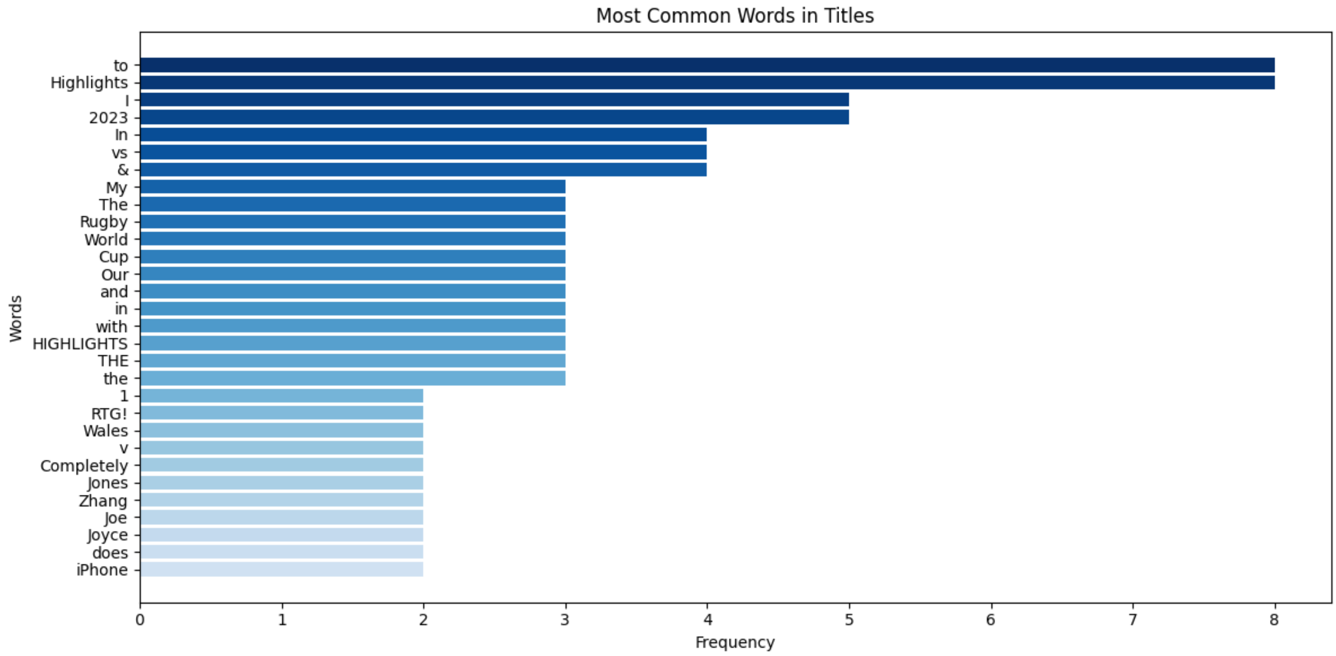
In [ ]:
```
# Most common words in titles of most popular videos

from collections import Counter
from textblob import TextBlob

# Initialize a counter to keep track of word frequencies
title_word_counter = Counter()

# Retrieve titles
for item in items:
  snippet = item.get('snippet', {})
  title = snippet.get('title')

  # Tokenise titles
  if title:
    title_words = title.split()
    title_words = \
```

```
        [word for word in title_words if '-' not in word and '|' not in word]
      title_word_counter.update(title_words)

# Find the most common words and their frequencies
common_title_words = title_word_counter.most_common(30)

# Print the most common words
print('Most Common Words in Titles:')
for word, count in common_title_words:
  print(f'{word}: {count}')
```

```
Most Common Words in Titles:
to: 8
Highlights: 8
I: 5
2023: 5
In: 4
vs: 4
&: 4
My: 3
The: 3
Rugby: 3
World: 3
Cup: 3
Our: 3
and: 3
in: 3
with: 3
HIGHLIGHTS: 3
THE: 3
the: 3
1: 2
RTG!: 2
Wales: 2
v: 2
Completely: 2
Jones: 2
Zhang: 2
Joe: 2
Joyce: 2
does: 2
iPhone: 2
```

In [ ]:

```
# Visualisation of most common words

# Extract the words and their frequencies from common_title_words
words, frequencies = zip(*common_title_words)

# Create light to dark colour gradient
colors = plt.cm.Blues(np.linspace(1, 0.2, len(words)))

# Create bar chart
plt.figure(figsize=(12, 6))
bars = plt.barh(range(len(words)), frequencies, tick_label=words, color=colors)
plt.xlabel('Frequency')
plt.ylabel('Words')
plt.title('Most Common Words in Titles')
plt.gca().invert_yaxis()

# Display the chart
plt.tight_layout()
plt.show()
```



When I first executed this code—a couple of days ago—this chart was dominated by terms related to football. Now, it seems to indicate the popularity of the 2023 rugby world cup, with the items "Highlights", "2023", "vs", "The", "Tugby", "World", "Cup", and "HIGHLIGHTS" all likely related to this phenomenon. Further inspection may identify the specific videos that contribute to the prominence of these terms and confirm or refute their connection to the 2023 rugby world cup.

The 2023 rugby world cup is an occassional and temporary phenomenon. It is not ongoing and it does not happen often—on a weekly or monthly basis. Furthermore, the shift in this chart from football to rugby indicates that YouTube popularity is a case of shifting sands—trends constantly change.

This list of common words was created from a sample of 50 popular YouTube videos in the United Kingdom. Therefore, it seems that in the United Kingdom, viral videos may be characterised by current trends. Whilst not surprising, this finding is not necessarily obvious. For example, it could be that viewers in the United Kingdom perceive YouTube primarily as a musical platform, stream music videos, and consume content related to sports via other platforms—or the television. YouTube could also be branded as a warehouse of memes that is detached from current events. A longitudinal study of virality on the platform may yield sturdier statements about its characteristics. For now, however, I would like to expand my exploration beyond the borders of the United Kingdom and compare different countries.

Comparing Countries

To further explore and familiarise myself with the YouTube API, I would like to explore whether the average number of views, comments, and likes differ between countries. For example, the most popular videos in one region may be characterised by a higher number of views, comments, or likes than the most popular videos in another region.

I first create three functions that return the **mean view count**, **mean comment count**, and **mean like count** for a given country. Here, `regionCode` functions as a proxy for country. 50 videos are sampled from a list of most popular videos— `chart = 'mostPopular'`.

In [ ]:
```
# Function for the average number of views in a given region (country)
def views_mean(country):
  popular_videos = youtube.videos().list(
      part = 'statistics',
```

```
        chart = 'mostPopular',
        regionCode = country,
        maxResults = 50).execute()
    items = popular_videos.get('items', [])
    total_views = 0
    total_videos = 0
    for item in items:
      statistics = item.get('statistics')
      view_count = statistics.get('viewCount')
      if view_count is not None:
        total_views += int(view_count)
        total_videos += 1
    mean = total_views / total_videos
    return int(mean)

# Function for the average number of comments in a given region (country)
def comments_mean(country):
    popular_videos = youtube.videos().list(
        part = 'statistics',
        chart = 'mostPopular',
        regionCode = country,
        maxResults = 50).execute()
    items = popular_videos.get('items', [])
    total_comments = 0
    total_videos = 0
    for item in items:
      statistics = item.get('statistics')
      comment_count = statistics.get('commentCount')
      if comment_count is not None:
        total_comments += int(comment_count)
        total_videos += 1
    mean = total_comments / total_videos
    return int(mean)

# Function for the average number of likes in a given region (country)
def likes_mean(country):
    popular_videos = youtube.videos().list(
        part = 'statistics',
        chart = 'mostPopular',
        regionCode = country,
        maxResults = 50).execute()
    items = popular_videos.get('items', [])
    total_likes = 0
    total_videos = 0
    for item in items:
      statistics = item.get('statistics')
      like_count = statistics.get('likeCount')
      if like_count is not None:
        total_likes += int(like_count)
        total_videos += 1
    mean = total_likes / total_videos
    return int(mean)
```

In the next two cells, I define a list of countries and then inspect and visualise the mean view count, comment count, and like count for each country in this list. I loosely control for population by selecting the countries with populations closest in number to the United Kingdom's population.

In [ ]:

```
# Population by country: https://shorturl.at/ansDI
# List of country codes: https://www.iban.com/country-codes


# Selection of countries
countries = ['TR',   # Turkey          (population: 86 million)
             'DE',   # Germany         (population: 83 million)
             'TH',   # Thailand        (population: 72 million)
             'GB',   # United Kingdom  (population: 68 million)
             'FR',   # France          (population: 65 million)
```

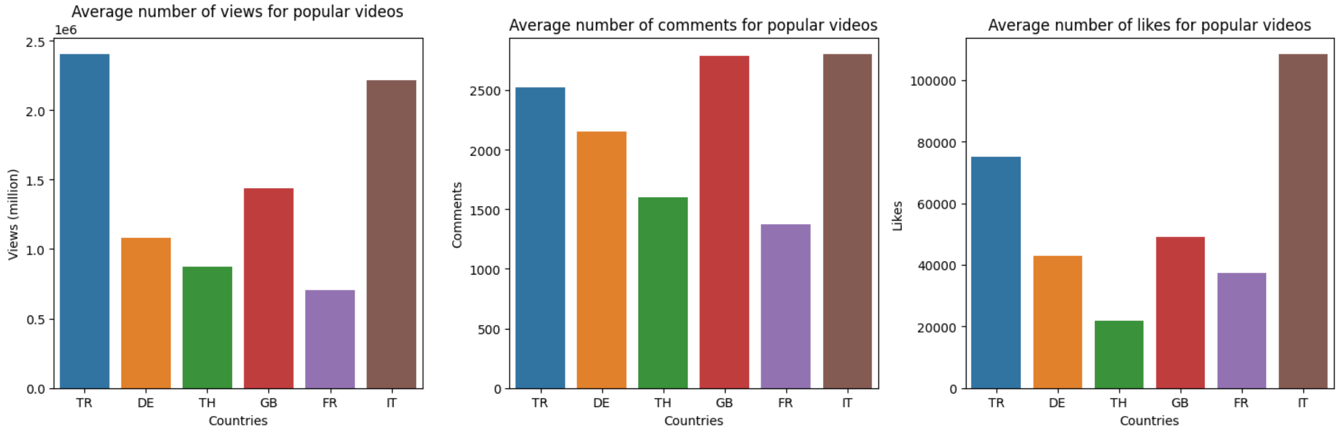
```
                'IT',   # Italy            (population: 59 million)
                ]

# Mean number of views, comments, and likes for each country
for country in countries:
  print(f'{country} mean views: ', views_mean(country), '\n',
        f'{country} mean comments: ', comments_mean(country), '\n',
        f'{country} mean likes: ', likes_mean(country), '\n')

print('\nTime at completion:', datetime.datetime.now())
```

```
TR mean views:  2402861
 TR mean comments:  2522
 TR mean likes:  75084

DE mean views:  1077721
 DE mean comments:  2154
 DE mean likes:  42777

TH mean views:  876310
 TH mean comments:  1599
 TH mean likes:  21726

GB mean views:  1434251
 GB mean comments:  2784
 GB mean likes:  48897

FR mean views:  704889
 FR mean comments:  1374
 FR mean likes:  37198

IT mean views:  2217386
 IT mean comments:  2801
 IT mean likes:  108403


Time at completion: 2023-09-26 03:10:38.727236
```

In [ ]:

```
# Variables to visualise
mean_views_by_country = [views_mean(country) for country in countries] # views
mean_comments_by_country = [comments_mean(country) for country in countries] # commen
mean_likes_by_country = [likes_mean(country) for country in countries] # likes

# Orientation
fig, axes = plt.subplots(1, 3, figsize=(15, 5)) # 1 horizontal line of 3 charts

# Bar plot for views
sns.barplot(x=countries, y=mean_views_by_country, ax=axes[0])
axes[0].set_title("Average number of views for popular videos")
axes[0].set_xlabel('Countries')
axes[0].set_ylabel('Views (million)')

# Bar plot for comments
sns.barplot(x=countries, y=mean_comments_by_country, ax=axes[1])
axes[1].set_title("Average number of comments for popular videos")
axes[1].set_xlabel('Countries')
axes[1].set_ylabel('Comments')

# Bar plot for likes
sns.barplot(x=countries, y=mean_likes_by_country, ax=axes[2])
axes[2].set_title("Average number of likes for popular videos")
axes[2].set_xlabel('Countries')
axes[2].set_ylabel('Likes')

# Adjust layout to prevent overlapping
plt.tight_layout()

# Show the plots
plt.show()
```

```
print('\nTime at completion:', datetime.datetime.now())
```




```
Time at completion: 2023-09-26 03:13:52.301771
```

It seems that videos viewed in Turkey and Italy are characterised by higher view counts and like counts than videos viewed in the United Kingdom. Of the countries in this list, I would have predicted the United Kingdom to have the highest mean view count, as the native language of viewers in the United Kingdom is generally English, and English is the most spoken—and perhaps therefore the most viewed—language worldwide. Thus, this chart does not conform to my intuitions.

Other aspects of these charts do conform to my intuitions. For example, Italy and the United Kingdom display the highest mean comment counts for popular videos. In comparison to Germany and Turkey, the cultures of the United Kingdom and Italy are often described as vocal and direct—people often speak their minds. Such intuitions may be supported or refuted by anthropological, sociological, and psychological research.

These charts are not sufficient to claim differences in means between countries. Statistically, this claim could be supported—or not—by the results of a t-test. I will demonstrate several t-tests in the next section.

Viral Words

I would like to investigate whether videos associated with one word tend to be more "viral" than videos associated with another word.

For the purpose of this microstudy, **view count** will function as a proxy for virality. Therefore, I would like to ask whether the **average view count** for videos associated with one word is higher than the **average view count** for videos associated with another word. To do this, I will need to compare the **means** of a sample of videos associated with the former word and a sample of videos associated with the latter word.

Statistically, this can be accomplished with a **two-sample t-test**.

In [ ]:

```
# @title Tren's YouTube Search Function
def youtube_search(q, max_results=100,order="relevance", token=None, location=None, l

    youtube = build('youtube','v3',developerKey=api_key)

    search_response = youtube.search().list(
    q=q,
    type="video",
    pageToken=token,
    order = order,
    part="id,snippet", # Part signifies the different types of data you want
```

```
    maxResults=max_results,
    location=location,
    locationRadius=location_radius).execute()

    title = []
    channelId = []
    channelTitle = []
    categoryId = []
    videoId = []
    viewCount = []
   #likeCount = []
    commentCount = []
    favoriteCount = []
    category = []
    tags = []
    videos = []

    for search_result in search_response.get("items", []):
        if search_result["id"]["kind"] == "youtube#video":

            title.append(search_result['snippet']['title'])

            videoId.append(search_result['id']['videoId'])

            response = youtube.videos().list(
                part='statistics, snippet',
                id=search_result['id']['videoId']).execute()

            channelId.append(response['items'][0]['snippet']['channelId'])
            channelTitle.append(response['items'][0]['snippet']['channelTitle'])
            categoryId.append(response['items'][0]['snippet']['categoryId'])
            favoriteCount.append(response['items'][0]['statistics']['favoriteCount'])
            viewCount.append(response['items'][0]['statistics']['viewCount'])
            #likeCount.append(response['items'][0]['statistics']['likeCount'])

        if 'commentCount' in response['items'][0]['statistics'].keys():
            commentCount.append(response['items'][0]['statistics']['commentCount'])
        else:
            commentCount.append([])
        if 'tags' in response['items'][0]['snippet'].keys():
            tags.append(response['items'][0]['snippet']['tags'])
        else:
            tags.append([])

    youtube_dict = {'tags':tags,'channelId': channelId,'channelTitle': channelTitle,'
                    #'likeCount':likeCount,
                    'commentCount':commentCount,'favoriteCount':favoriteCount}

    return youtube_dict
```

Tren's function helps me produce a sample of 50 videos associated with a given term.

Bairn means child in some Scots and English dialects. There are relatively few speakers of these Scots and English dialects. Furthermore, speakers of these Scots and English dialects are often accustomed to using standard English when searching for things online. Therefore, we would expect a sample of videos associated with the word *bairn* to have a lower **average view count** than a sample of videos associated with the word child.

I will assess the validity of my method with this test.

In [ ]:
```
# Bairn query
bairn_query = youtube_search("bairn") # sample size: 50
bairn_df = pd.DataFrame(data=bairn_query)

# Child query
child_query = youtube_search("child") # sample size: 50
```

```python
child_df = pd.DataFrame(data=child_query)

# Mean view count for bairn sample
bairn_df['viewCount'] = bairn_df['viewCount'].astype(float) # object to float
print('Bairn mean view count: ', bairn_df['viewCount'].mean())

# Mean view count for child sample
child_df['viewCount'] = child_df['viewCount'].astype(float) # object to float
print('Child mean view count: ', child_df['viewCount'].mean())

# Null hypothesis:
# The bairn and child samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The bairn and child samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test
t_stat, p_val = stats.ttest_ind(bairn_df['viewCount'], child_df['viewCount'])
print("\nt-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
Bairn mean view count:  714428.46
Child mean view count:  165491994.66

t-statistic = -3.6465823723506787
p-value = 0.00042779018485070695

Time at completion: 2023-09-26 03:44:09.442455
```

The **t-test** yields a **p-value** that is less than `α = 0.05`.

This conforms to my intuition that the **mean view count** for videos associated with the word *bairn* will be significantly lower than the **mean view count** for videos associated with the word child.

It seems my method is valid.

I would now like to investigate whether videos associated with certain countries tend to have higher view counts than videos associated with other countries.

In the next two cells, I will define a list of countries and then inspect and visualise the mean view count for each country in this list.

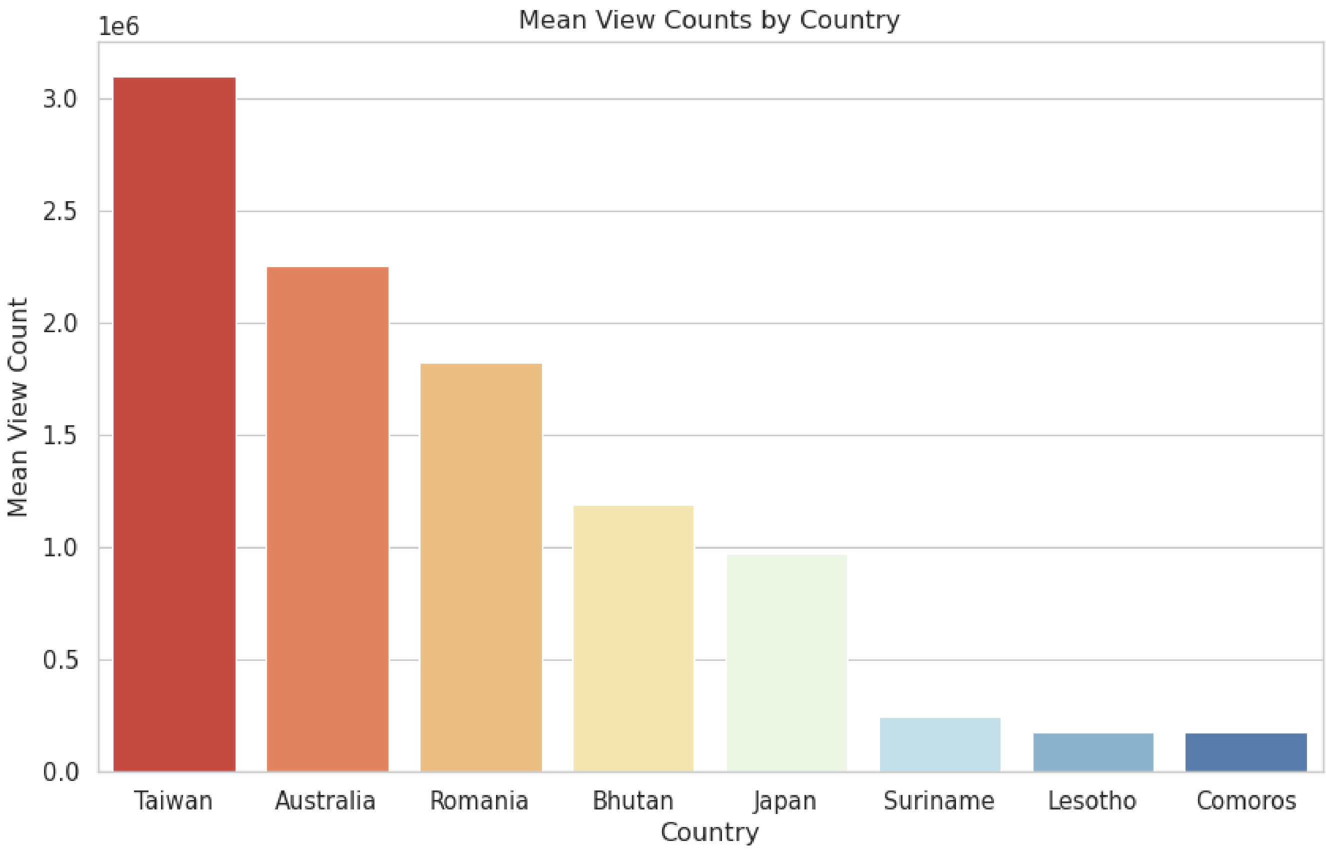
In [ ]:

```python
# Select countries
country_list = ['Japan', 'Australia', 'Taiwan', 'Romania',
                'Bhutan', 'Suriname', 'Lesotho', 'Comoros']

# Create list to store mean view counts
mean_view_counts = []

# Calculate mean view counts
for country in country_list:
  query = youtube_search(country) # sample of 50 videos
  df = pd.DataFrame(data=query) # create pandas DataFrame
  df['viewCount'] = df['viewCount'].astype(float) # object type to float type
  mean_view_count = df.viewCount.mean() # mean view count
  mean_view_counts.append(mean_view_count) # add to list
  print(f'{country}: {mean_view_count}')

print('\nTime at completion:', datetime.datetime.now())
```

```
Japan: 973410.16
Australia: 2253431.18
Taiwan: 3098332.32
Romania: 1822559.04
Bhutan: 1190774.38
Suriname: 244664.06
Lesotho: 175798.54
Comoros: 173247.12

Time at completion: 2023-09-26 04:03:34.701413
```

In [ ]:

```python
# Create DataFrame for visualisation
vis_data = pd.DataFrame({'Country': country_list,
                         'Mean View Count': mean_view_counts})

# Create bar chart
sns.set(style="whitegrid", context = 'paper', font_scale = 1.2)
plt.figure(figsize=(10, 6))
sns.barplot(
    x = "Country", y = "Mean View Count", data = vis_data,
    palette = sns.color_palette("RdYlBu", n_colors=8), # hot to cold colours
    order = vis_data.sort_values("Mean View Count", ascending=False).Country
    )
plt.xlabel("Country")
plt.ylabel("Mean View Count")
plt.title("Mean View Counts by Country")
plt.show()
```



Videos associated with Taiwan appear to have more views on average. In contrast, videos associated with Suriname appear to have less views on average. This seems to make sense, as Taiwan is an immensely more famous country than Suriname.

Importantly, I am using the English language—search terms are in English—to create these samples. In other words, I search for "Japan" rather than 日本 and "Taiwan" rather than 台灣. Thus, it would be more accurate to state that videos **in English** associated with Taiwan appear to have more views on average, and videos **in English** associated with Suriname appear to have less views on average.

In addition, these results are highly sensitive to time and date. An earlier execution of the cells above presented Japan as the country with the highest mean view count. It is therefore important to display the time of completion: `print('\nTime at completion:', datetime.datetime.now())`.

The results of a t-test indicate whether or not there is a statistically significant difference between the mean view counts of two countries. In the following cells, I will conduct and interpret two t-tests to compare the the mean view counts between different countries.

In [ ]:
```
# T-test for Taiwan and Suriname view counts

# Taiwan query
TW_query = youtube_search('Taiwan') # sample 50 videos
TW_df = pd.DataFrame(data=TW_query) # create DataFrame
TW_df['viewCount'] = TW_df['viewCount'].astype(float) # object to float

# Suriname query
SR_query = youtube_search('Suriname') # sample 50 videos
SR_df = pd.DataFrame(data=SR_query)
SR_df['viewCount'] = SR_df['viewCount'].astype(float)

# Null hypothesis:
# The Taiwan and Suriname samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Taiwan and Suriname samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test
t_stat, p_val = stats.ttest_ind(TW_df['viewCount'], SR_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 1.5245204709692932
p-value = 0.13059947084948886

Time at completion: 2023-09-26 04:21:57.264061
```

The t-test yields a p-value that is greater than `α = 0.05`. This indicates that I do not have sufficient evidence to reject the null hypothesis: the mean view counts for the Taiwan and Suriname samples are equal.

Despite the appearance of the bar chart above, I cannot argue on the basis of this statistical test that videos associated with Taiwan tend to have a higher or lower average view count than videos associated with Suriname. Further research may be conducted by increasing the sample size, which may partially account for the high p-value in this case.

In [ ]:
```
# T-test for Japan and Lesotho view counts

# Lesotho query
LS_query = youtube_search('Lesotho') # sample 50 videos
LS_df = pd.DataFrame(data=LS_query)
LS_df['viewCount'] = LS_df['viewCount'].astype(float)

# Null hypothesis:
# The Japan and Lesotho samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Japan and Lesotho samples do not have equal mean view counts.
```

```
# x̄_b_v ≠ x̄_c_v

# T-test
t_stat, p_val = stats.ttest_ind(JP_df['viewCount'], LS_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 1.3384561891910849
p-value = 0.1838447536857996

Time at completion: 2023-09-26 04:17:16.065221
```

Once again, the t-test yields a p-value that is greater than `α = 0.05`. This indicates that I do not have sufficient evidence to reject the null hypothesis: the mean view counts for the Japan and Lesotho samples are equal.

I cannot argue on the basis of this statistical test that videos associated with Japan tend to have a higher or lower average view count than videos associated with Lesotho.

I would now like to investigate whether videos associated with certain languages tend to have higher view counts than videos associated with other languages.

In the next two cells, I define a list of languages and then inspect and visualise their medians. This time, I calculate medians—rather than means—to minimise the influence of outliers.

In [ ]:

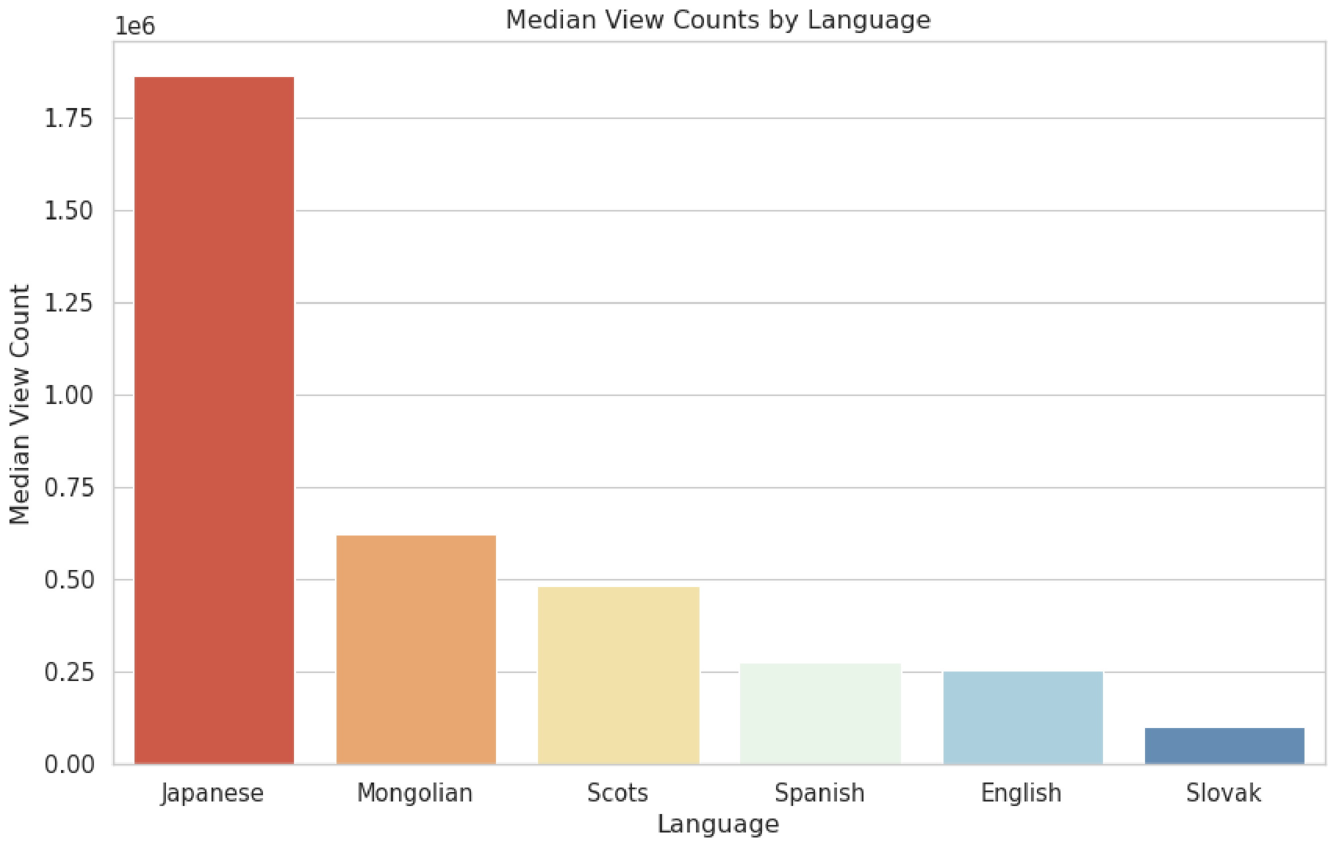
```
# Select languages
languages = ['English', 'Spanish', 'Japanese',
             'Mongolian', 'Slovak', 'Scots']

# Create list to store median view counts — help control against outliers
median_view_counts = []

# Calculate median view counts
for language in languages:
  query = youtube_search(language) # sample of 50 videos
  df = pd.DataFrame(data=query) # create pandas DataFrame
  df['viewCount'] = df['viewCount'].astype(float) # object type to float type
  median_view_count = df.viewCount.median() # median view count
  median_view_counts.append(median_view_count) # add to list
  print(f'{language}: {median_view_count}')

print('\nTime at completion:', datetime.datetime.now())
```

```
English: 254268.0
Spanish: 274091.0
Japanese: 1863941.0
Mongolian: 620703.0
Slovak: 102048.5
Scots: 484560.5

Time at completion: 2023-09-26 04:21:47.666154
```

In [ ]:

```
# Visualise medians

# Create DataFrame for visualisation
vis_data_ = pd.DataFrame({'Language': languages,
                          'Median View Count': median_view_counts})

# Create bar chart
sns.set(style="whitegrid", context = 'paper', font_scale = 1.2)
plt.figure(figsize=(10, 6))
sns.barplot(
    x = "Language", y = "Median View Count", data = vis_data_,
    palette = sns.color_palette("RdYlBu", n_colors=6), # hot to cold colours
```

```
    order = vis_data_.sort_values("Median View Count", ascending=False).Language
    )
plt.xlabel("Language")
plt.ylabel("Median View Count")
plt.title("Median View Counts by Language")
plt.show()
```



The average—median—view count of videos associated with the term "Japanese" appears to be higher than the average—median—view counts of videos associated with the terms "Mongolian", "Scots", "Spanish", "English", and "Slovak".

One may expect "English" to display the highest median. English is the most common second language and one may therefore expect videos pertaining to the language to have more views on average. However, I have used English terms to create these samples. For example, I used "English" and not "英文" to acquire a sample of videos that are associated with English. This is problematic because learners of English may use their native languages to search for videos about English. Taiwanese users may be more likely to search "英文" than "English"; Spanish users may be more likely to search "Inglés" than "English". Thus, this bar chart may be more precisely said to reflect the languages that yield the highest view counts among users of the English language. This may explain the relatively small median for English, as speakers of English may be less likely to search for videos pertaining to English and more likely to search for videos pertaining to Japanese, for example. Furthermore, this perspective may also explain the perhaps unexpectedly large median for Scots. Scots is a comparatively small language that may be expected to yield a small median, but it is—debatably—mutually intelligible with English and both geographically and culturally close to England. These factors may instill an interest in Scots among speakers of English.

A more problematic confounding variable is the linguistic function of these terms. "Slovak" can be a **noun** that refers to the Western Slavic language of Slovakia, but it can also be an **adjective** that refers to the people and culture of Slovakia. Similarly, "Japanese" can be a **noun** that refers to the language of Japan, but it can also be an **adjective** that refers to the people and culture of Japan. Thus, videos associated with the term "Japanese" do not necessarily pertain to the language and may instead be about Japanese people, food, technology, and so on. Thus, it may be more precisely said that the following tests compare the average view counts for **English** videos pertaining to the **languages, cultures, or people** of different countries

In the following cells, I will execute and interpret several t-tests to compare these average view counts—calculated as means.

In [ ]:
```
# English query
English_query = youtube_search("English") # sample 50 videos
English_df = pd.DataFrame(data=English_query)
English_df['viewCount'] = English_df['viewCount'].astype(float)

# Scots query
Scots_query = youtube_search("Scots") # sample 50 videos
Scots_df = pd.DataFrame(data=Scots_query)
Scots_df['viewCount'] = Scots_df['viewCount'].astype(float)

# Null hypothesis:
# The Scots and English samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Scots and English samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test for Scots and English
t_stat, p_val = stats.ttest_ind(Scots_df['viewCount'], English_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 2.0941765234117335
p-value = 0.03882459802056436

Time at completion: 2023-09-26 05:18:55.777009
```

This t-test yields a p-value that is less than `α = 0.05`. This indicates that I have sufficient evidence to reject the null hypothesis: the mean view counts for the "Scots" and "English" samples are equal.

I can argue on the basis of this statistical test that videos associated with "Scots" tend to have a higher average view count than videos associated with "English".

In [ ]:
```
# Slovak query
Slovak_query = youtube_search("Slovak")
Slovak_df = pd.DataFrame(data=Slovak_query)
Slovak_df['viewCount'] = Slovak_df['viewCount'].astype(float)

# Null hypothesis:
# The Slovak and English samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Slovak and English samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test for Slovak and English
t_stat, p_val = stats.ttest_ind(Slovak_df['viewCount'], English_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = -1.9277772230359609
p-value = 0.05677792556098464

Time at completion: 2023-09-26 05:50:39.748413
```

This t-test yields a p-value that is greater than `α = 0.05`. This indicates that I do not have sufficient evidence to reject the null hypothesis: the mean view counts for the "Slovak" and "English" samples are equal.

I cannot argue on the basis of this statistical test that videos associated with "Slovak" tend to have a higher or lower average view count than videos associated with "English".

In [ ]:

```
# Spanish query
Spanish_query = youtube_search("Spanish")
Spanish_df = pd.DataFrame(data=Spanish_query)

Spanish_df['viewCount'] = Spanish_df['viewCount'].astype(float)

# Null hypothesis:
# The Spanish and English samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Spanish and English samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test for Spanish and English
t_stat, p_val = stats.ttest_ind(Spanish_df['viewCount'], English_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 2.437148605902212
p-value = 0.016606892481431577

Time at completion: 2023-09-26 05:55:46.299543
```

This t-test yields a p-value that is less than `α = 0.05`. This indicates that I have sufficient evidence to reject the null hypothesis: the mean view counts for the "Spanish" and "English" samples are equal.

I can argue on the basis of this statistical test that videos associated with "Spanish" tend to have a higher average view count than videos associated with "English".

Please note the time at completion.

In [ ]:

```
# Mongolian query
Mongolian_query = youtube_search("Mongolian")
Mongolian_df = pd.DataFrame(data=Mongolian_query)

Mongolian_df['viewCount'] = Mongolian_df['viewCount'].astype(float)

# Null hypothesis:
# The Mongolian and English samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Mongolian and English samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test for Mongolian and English
t_stat, p_val = stats.ttest_ind(Mongolian_df['viewCount'], English_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))
```

```
print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 1.7164709088122665
p-value = 0.08923521018405252

Time at completion: 2023-09-26 05:58:34.322396
```

This t-test yields a p-value that is greater than `α = 0.05`. This indicates that I do not have sufficient evidence to reject the null hypothesis: the mean view counts for the "Mongolian" and "English" samples are equal.

I cannot argue on the basis of this statistical test that videos associated with "Mongolian" tend to have a higher or lower average view count than videos associated with "English".

In [ ]:

```
# Null hypothesis:
# The Mongolian and Slovak samples have equal mean view counts.
# x̄_b_v = x̄_c_v

# Alternative hypothesis:
# The Mongolian and Slovak samples do not have equal mean view counts.
# x̄_b_v ≠ x̄_c_v

# T-test for Mongolian and Slovak
t_stat, p_val = stats.ttest_ind(Mongolian_df['viewCount'], Slovak_df['viewCount'])
print("t-statistic = " + str(t_stat))
print("p-value = " + str(p_val))

print('\nTime at completion:', datetime.datetime.now())
```

```
t-statistic = 1.918839941822487
p-value = 0.05791432714957369

Time at completion: 2023-09-26 06:01:20.081524
```

This t-test yields a p-value that is greater than `α = 0.05`. This indicates that I do not have sufficient evidence to reject the null hypothesis: the mean view counts for the "Mongolian" and "Slovak" samples are equal.

I cannot argue on the basis of this statistical test that videos associated with "Mongolian" tend to have a higher or lower average view count than videos associated with "Slovak".

Summary

Firstly, I explored and familiarised myself with the YouTube API by working with a sample of popular videos in the United Kingdom. I learned that the descriptions of popular videos in the United Kingdom tend to be characterised by a positive sentiment. I also learned that viral videos in the United Kingdom may be characterised by current trends—the 2023 rugby world cup at the time of writing.

Secondly, I further explored and familiarised myself with the YouTube API by comparing countries. At the time of writing, popular—viral—videos viewed in Italy and Turkey were seemingly characterised by higher view counts and like counts on average than popular videos viewed in the United Kingdom. Popular videos in Italy and the United Kingdom were seemingly characteristed by the highest comment counts on average.

Thirdly, I applied Tren's function and t-tests to investigate differences in average view counts between video samples associated with different terms. For example, a t-test supported the hypothesis that videos associated with "Scots" tend to have a higher average view count than videos associated with "English". I would therefore argue that if all other variables are controlled for, there is a higher probability of a video associated with "Scots" becoming viral than a video associated with "English".

Please submit the PDF version of your notebook to NTU COOL before 10/6 (Friday).