Data Analysis

Glean the Trend in India and Pakistan from Google Search Keywords Using Topic Modeling

Hyerim Hwang

To whom might be interested in these countries to promote their products or services

Table of Contents

Introduction

Google is the most commonly used search engine in the world across all platforms, “with more than 5.4 billion searches each day and 78,881 searches in 1 second" (Google Search Statistics). Google Trend, also, offers an well-structured information within the limit of 1 - 30 Categories. There are top 10 popular keywords per each category, and lists on each year by each country from 2001 until now. The initial trends per each country are expected to see and capture each historical interests or big events over time.

The report tried to capture the global trends, however, Global dataset in Google trends does not support China and three Higher internet usage countries in Asia don’t use Google as a primary search engine (Internet Stats & Facts for 2019). Due to these reasons, the report decided to rather look up the regions that have a lower broadband penetration rate. The lower broadband penetration rate will be a great indicator to find out a less compatible market for the search engine, so it will lead to get more prominent results by using the Google search dataset to see those region’s overall trends.

Methodology

Choosing data - India[1], Pakistan[2]

To capture the regions' broadband penetration rates, the internet world statistics dataset was available based on the research of 4,536,248,808 internet users on Jun 30, 2019. The dataset below (Figure. 1) contained three different variables. At first, there's some country's broadband penetration rates that were lower than the average penetration rates in each region. Also, the country naming list of Google Trends were added to compare the regions' data that were lower penetration rates with it. Then, the ideal target countries came out with the name of India, Pakistan and the Philippines containing with each rate of internet penetration rate per each country.

[1] https://trends.google.com/trends/yis/2018/IN/
[2] https://trends.google.com/trends/yis/2018/PK/

```
under_mean_by_internet_penetration = [{'region' : 'Oceania', 'under_mean_country': {'American Samoa': '43.1%', 'Chrismas Island': '45.4
                                      {'region' : 'Africa', 'under_mean_country': {'Angola': '22.3%', 'Benin': '32.2%', 'Burkina Faso':
                                      {'region' : 'Middle East', 'under_mean_country': {'Yemen':'26.6%', 'Iraq':'49.4%', 'Palestine (St
                                      {'region' : 'Asia', 'under_mean_country': {'Afganistan': '19.7%', 'Bhutan': '48.1%', 'Cambodia':

country_google = ['Argentina','Australia','Austria','Bangladesh','Belarus','Belgium','Brazil','Bulgaria','Canada',
                  'Chile','Colombia','Costa Rica','Croatia','Czechia','Denmark','Egypt','Estonia','Finland','France',
                  'Germany','Greece','Guatemala','Hong Kong','Hungary','India','Indonesia','Ireland','Israel','Italy',
                  'Japan','Kazakhstan','Kenya','Latvia','Lithuania','Malaysia','Mexico','Netherlands','New Zealand',
                  'Nigeria','Norway','Pakistan','Panama','Peru','Philippines','Poland','Portugal','Puerto Rico','Romania',
                  'Russia','Saudi Arabia','Serbia','Singapore','Slovakia','Slovenia','South Africa','South Korea','Spain',
                  'Sweden','Switzerland','Taiwan','Thailand','Turkey','Ukraine','United Arab Emirates','United Kingdom','United States'
]

for region in under_mean_by_internet_penetration:
    lists = []
    for country in country_google:
        dicts ={}
        if country in region['under_mean_country'].keys():
            dicts['region'] = region['region']
            dicts['under_mean_country'] = country
            dicts['Penetration_Population'] = region['under_mean_country'][country]
            dicts['mean_of_the region'] = region['mean']
            lists.append(dicts)
print(lists)

lists = [{'region': 'Asia', 'under_mean_country': 'India', 'Penetration_Population': '40.9%', 'mean_of_the region': '54.2%'},
         {'region': 'Asia', 'under_mean_country': 'Pakistan', 'Penetration_Population': '35.0%', 'mean_of_the region': '54.2%'},
```

Figure. 1: Combining two dataset to get the target countries with lower broadband penetration rates.

Since the datasets are India, Pakistan and the Philippines, the report tried to narrow down to see the correlation and difference between India and Pakistan throughout their each keyword. So the following step was building hypotheses that are

- Expect to see any changes in trends in each region over time
- Expect to see correlation and difference with two countries over time

I began to collect the data in Google Trend with each year's search keywords and ranking (Figure. 2). It also required to gather the keyword's href attribute as well to see all of the search volumes by subregion that was located on each keyword's content page (Figure. 3).

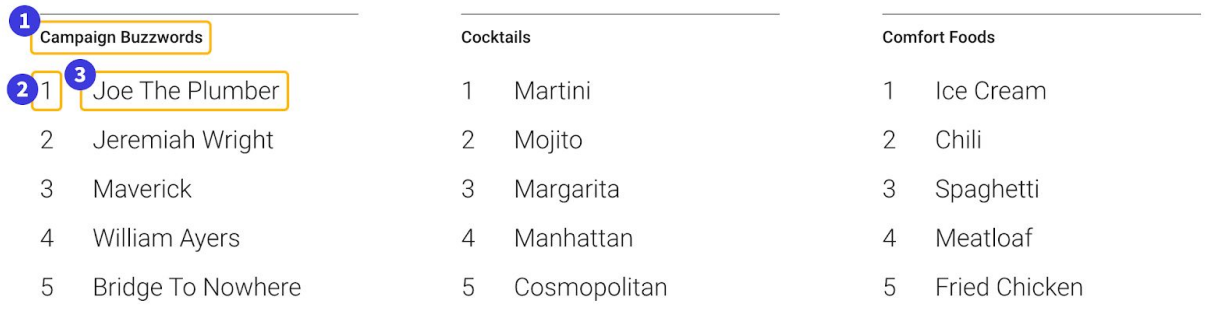


Figure. 2: The homepage contains a list that has 1: category | 2: ranking | 3: keyword.

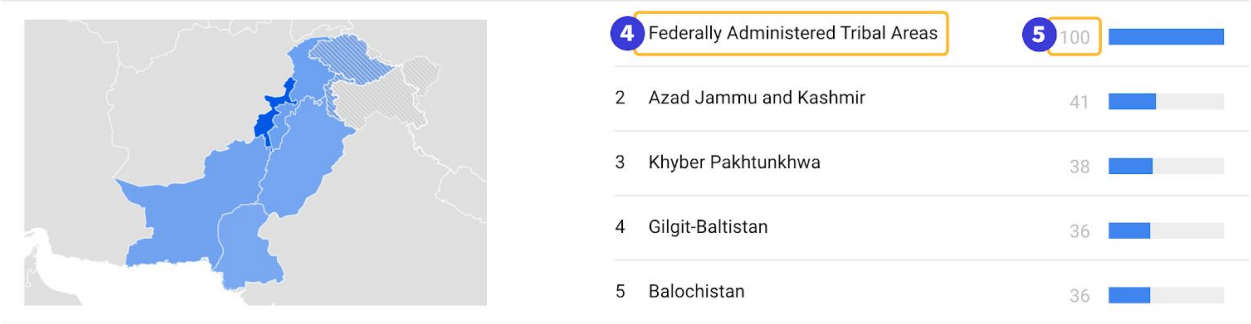


Figure. 3: The content page contains 4: a name of subregion | 5: search volumes.

Collecting data - Selenium

The datasets with multiple pages were needed to use web scraping method, so the Selenium library in Python was selected which allows to click on “show 5 more” buttons by an automatic bot based on my code. To analyze the data based on the robust basement, I built the MongoDB cluster and database to store each year’s keyword and other information on each collection. Inside of the collection, I stored the data of each keyword including the data of ranking in each year, search keyword grouping by category, search volumes by region.

Cleaning data - Pandas and Regular Expression

The dataset are huge so applying pandas library was a primary tool to clean and recreate data set for fitting to analyze the data, and also utilized the regular expression library for potential overlapping keyword in each country - Split the keywords into words and lowercase the words and remove punctuation (Appendix 1).

Analyzing data - Word Cloud and Countvectorizer from sklearn

To get a visual representation of the most common words, the report used the word cloud library that was a key to understanding the data and ensuring (Figure. 4).

Figure. 4: From the word cloud, we can discovered search Queries became generally more diversity and increased.

Then, each of keywords were needed to convert to the groups into a simple vector representation to get a matrix of token counts (Topic Modeling in Python: LDA). It returned a bar plot that contains top 10 most frequent words based on the outcome of this operation (Figure. 5).

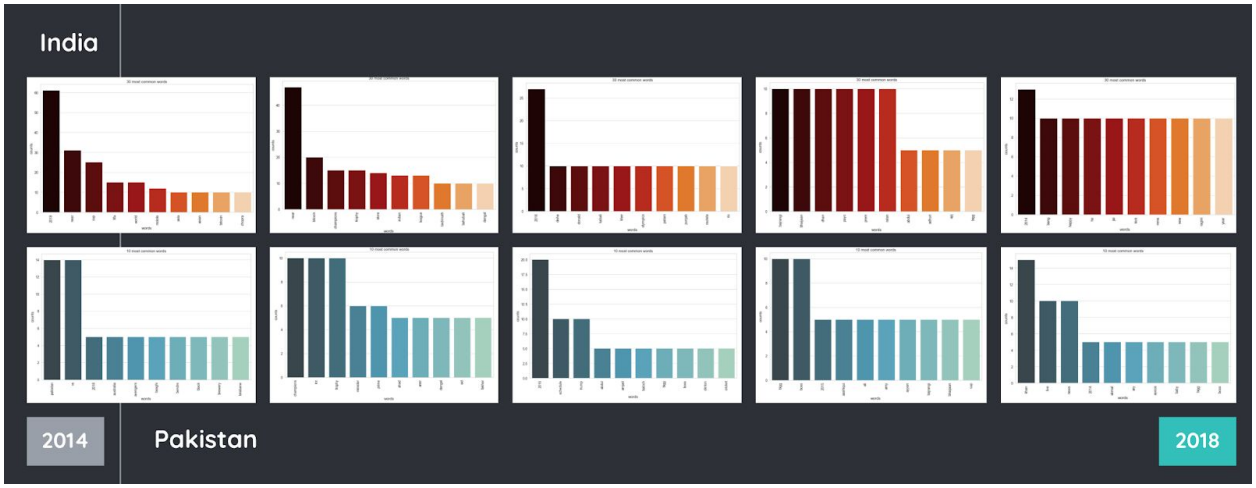


Figure. 5: As time goes by, top keyword occurs less than in the past, so the frequency of each keaword has been reduced over time.

It was actually one of the preparations of the textual data in a format that will serve as an input for training LDA model. However, LDA model only supports English, the report made classifications from top 30 most frequent words applying Countvectorizer (Appendix 1 and Appendix 2), and see the trends from the classifications.

Limitation

Since the original background knowledge in India and Pakistan was not enough and some words in their languages(Hindi and Urdu) discovered a lot, the report researched through Google a lot. Also, one of the popular Topic model studies was LDA analysis, but ‘english’ is currently the only supported string value in LDA analysis, the report categorized some of top 30 keywords in manually.

Findings

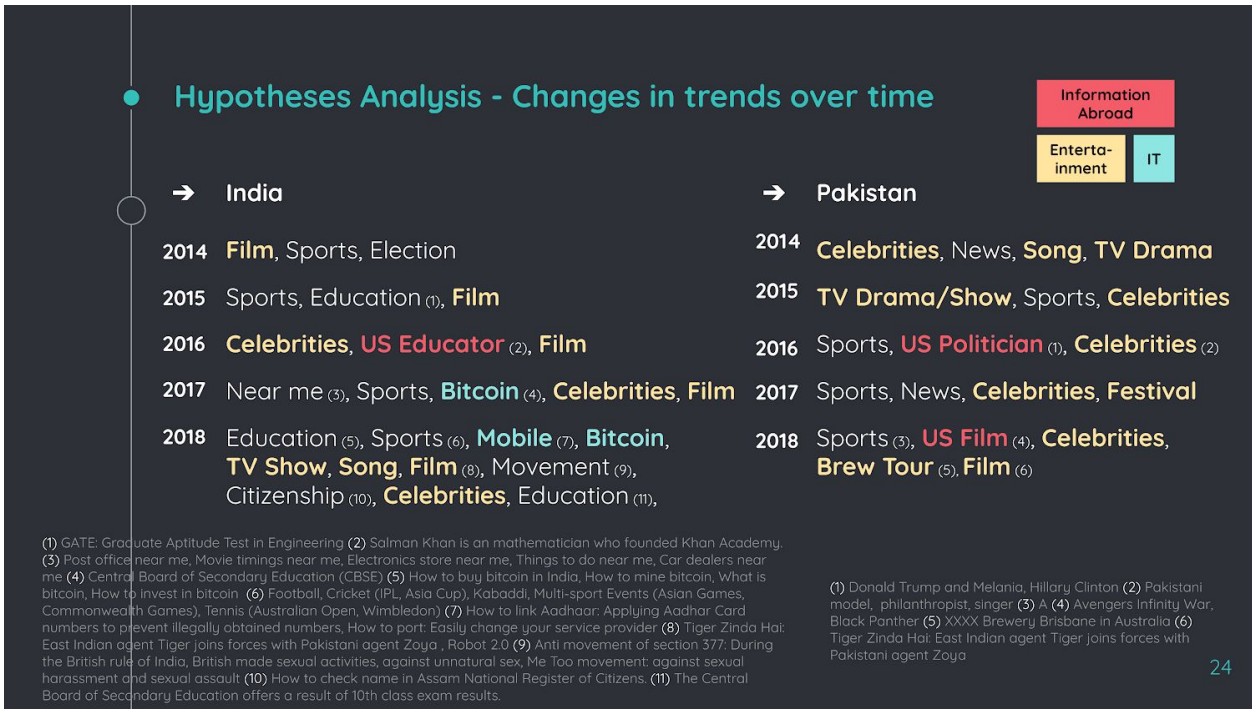


Figure. 6: Create each category over time and the classifications with color codings

In overall, the keywords that go through India was Film, and Pakistan got celebrities, and both can be classified by entertainment.

In the early stage of searching in Google, India got the film and sports on their most of keywords, and Pakistan had the celebrities the most. These keywords are quite similar topics that might be under the entertainment classification. Since Pakistan searched a lot of Indian actors, actress and model, it also can tell that they have a common background and interest.

Sports keyword is a key to describe these two countries' hobbies and interests that have been played the same major sport event for a long time.

India and Pakistan both enjoy with different types of sports and hold sports events annually which are Football(FIFA World Cup), Cricket (IPL, Asia Cup), Kabaddi (Pro Kabaddi League) and Multi-sport Events (Asian Games, Commonwealth Games). They are also interested in

watching the tennis games abroad such as Australian Open, or Wimbledon. These might cause of sharing the same culture in the past for a long time.

India has been gradually increased with a variety of topics in their interests, however, Pakistan still stick with their existing topics.

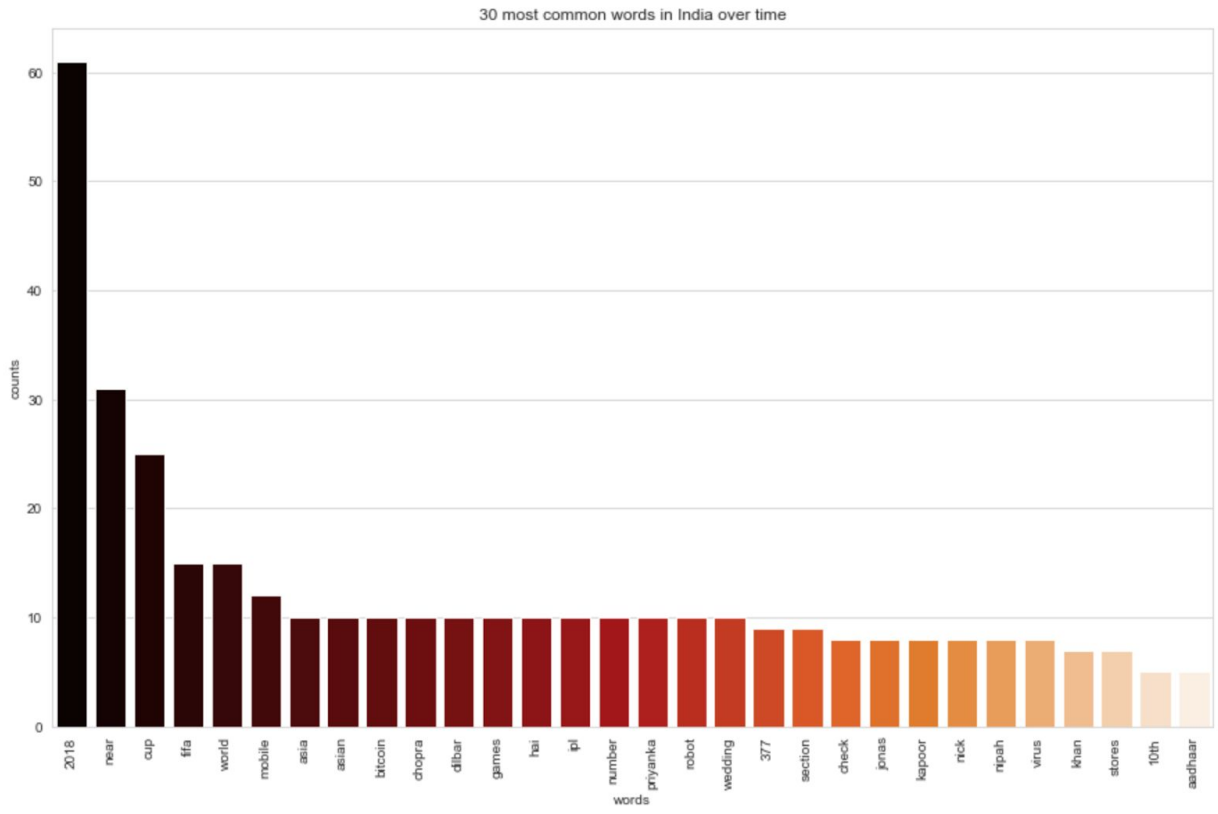
India became to pay closer attention to education and human rights. The Graduate Aptitude Test in Engineering and The Central Board of Secondary Education offers a result of 10th class exam results.are the evidence of how Indea cares a lot, and the movement of Anti movement of section 377 and Me Too movement, or checking the Assam National Register of Citizens can be part of human rights. Pakistan, however, kept focusing on entertainment topics.

Conclusion

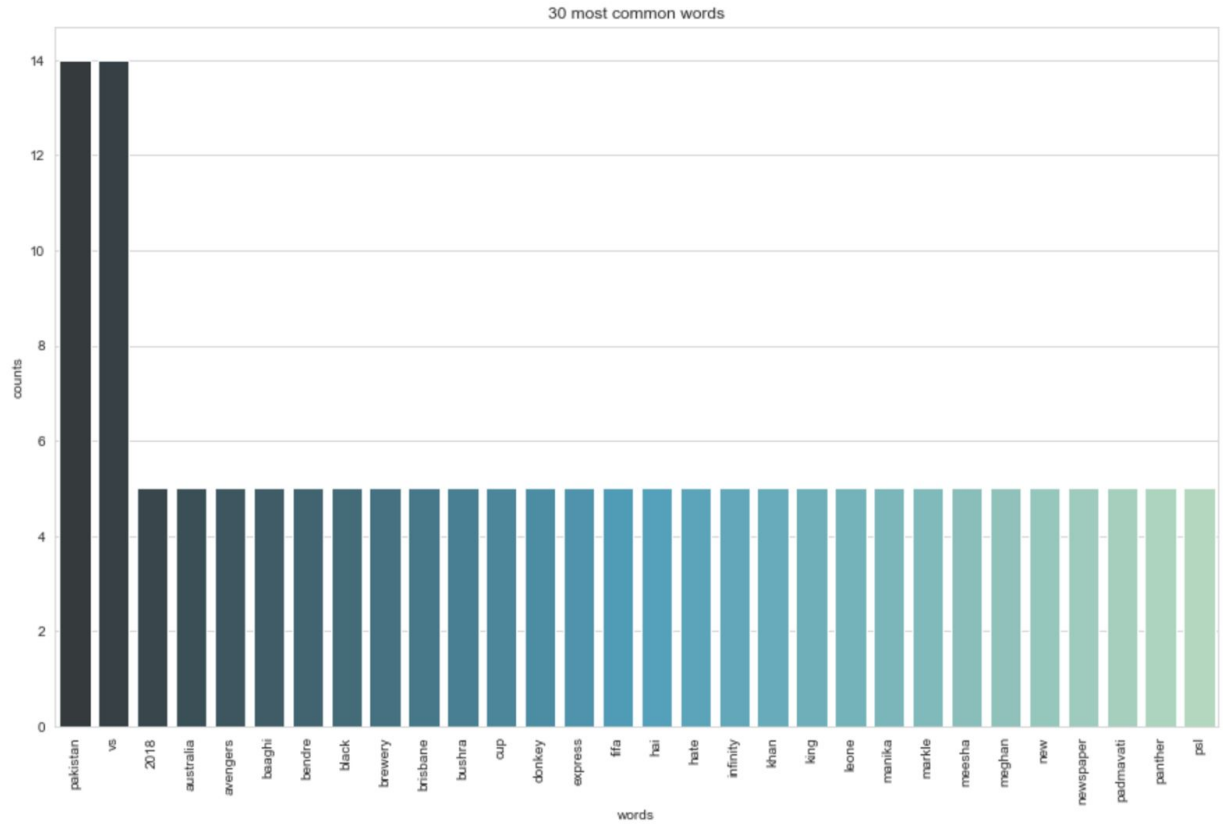
After the dissolution of the British Raj in 1947 (Wikipedia), the relations between India and Pakistan have been through largely number of historical and political events and still have a very complex and controversial connection. However, India and Pakistan have a common understanding and culturally bound - both love to play or watch sports games, and like various movies, actors, and actresses in each other's film. There was a Jammu and Kashmir Reorganisation Act in 2019 that Pakistan eventually cut off others -India and China, but It also told that there can be always a chance to get along with one another. Due to the limitation of not existing dataset in 2019 from my dataset, it will need a further study to gain more evidence to prove my findings.

Appendix

Appendix. 1 Top 30 most frequent words in India over time



Appendix. 2 Top 30 most frequent words in Pakistan over time



Appendix. 3 Code in Python using Jupyter Notebook

```
# Importing modules
import pymongo
from pymongo import MongoClient
import pandas as pd
from pandas.io.json import json_normalize

# Read data from MongoDB
cluster =
MongoClient('mongodb+srv://rimho:0000@cluster0-yehww.mongodb.net/test?retryWrites=true&w=majority')
db = cluster['3_Google_search_trends_db']
collections = [db[c] for c in ['IN']]
documents =  [collection.find() for collection in collections]

products = []
for document in documents:
   for p in document:
      products.append(p)
table = json_normalize(products)

# # Print head
table.head()

#### Data Cleaning
Since the goal of this analysis is to perform topic modeling, we will solely focus on the text data from each paper,
and drop other metadata columns

# Remove the columns
columns_name = list(table)
basic_df = table.drop(table.columns[[0, 6]], axis=1)
cols = basic_df.columns.tolist()
df = basic_df[cols]#.sort_values(by='year', ascending=False)
keyword_PK = df.loc[df['country'] == 'PK']

# Print out the first rows of papers
```

df

Remove punctuation/lower casing

Next, let's perform a simple preprocessing on the content of paper_text column to make them more amenable for analysis, and reliable results. To do that, we'll use a regular expression to remove any punctuation, and then lowercase the text

```
# Load the regular expression library
import re

# Remove punctuation
keyword_PK['keyword_processed'] = keyword_PK['keyword'].map(lambda x: re.sub('[,\.!?]', '', x))
# Convert the titles to lowercase
keyword_PK['keyword_processed'] = keyword_PK['keyword'].map(lambda x: x.lower())

# Print out the first rows of papers
keyword_PK['keyword_processed'].head()

keyword_PK_2014 = keyword_PK.loc[df['year'] == '2014']
keyword_PK_2015 = keyword_PK.loc[df['year'] == '2015']
keyword_PK_2016 = keyword_PK.loc[df['year'] == '2016']
keyword_PK_2017 = keyword_PK.loc[df['year'] == '2017']
keyword_PK_2018 = keyword_PK.loc[df['year'] == '2018']
keyword_PK_2014
```

Exploratory Analysis

To verify whether the preprocessing happened correctly, we'll make a word cloud using the wordcloud package to get a visual representation of the most common words. It is key to understanding the data and ensuring we are on the right track, and if any more preprocessing is necessary before training the model.

```
# Import the wordcloud library
from wordcloud import WordCloud
import matplotlib
```

```
import matplotlib.pyplot as plt

# Join the different processed titles together.
long_string = ','.join(list(keyword_PK_2014['keyword_processed'].values))

# Create a WordCloud object
wordcloud = WordCloud(background_color="white", max_words=1000, contour_width=3,
colormap=matplotlib.cm.twilight_shifted)

# Generate a word cloud
wordcloud.generate(long_string)

# Visualize the word cloud
wordcloud.to_image()

long_string = ','.join(list(keyword_PK_2015['keyword_processed'].values))
wordcloud = WordCloud(background_color="white", max_words=1000, contour_width=3,
colormap=matplotlib.cm.twilight_shifted)
wordcloud.generate(long_string)
wordcloud.to_image()

long_string = ','.join(list(keyword_PK_2016['keyword_processed'].values))
wordcloud = WordCloud(background_color="white", max_words=1000, contour_width=3,
colormap=matplotlib.cm.twilight_shifted)
wordcloud.generate(long_string)
wordcloud.to_image()

long_string = ','.join(list(keyword_PK_2017['keyword_processed'].values))
wordcloud = WordCloud(background_color="white", max_words=1000, contour_width=3,
colormap=matplotlib.cm.twilight_shifted)
wordcloud.generate(long_string)
wordcloud.to_image()

long_string = ','.join(list(keyword_PK_2018['keyword_processed'].values))
```

```
wordcloud = WordCloud(background_color="white", max_words=1000, contour_width=3,
colormap=matplotlib.cm.twilight_shifted)
wordcloud.generate(long_string)
wordcloud.to_image()
```

Prepare text for LDA analysis

Next, let's work to transform the textual data in a format that will serve as an input for training LDA model. We start by converting the documents into a simple vector representation (Bag of Words BOW). Next, we will convert a list of titles into lists of vectors, all with length equal to the vocabulary.

We'll then plot the ten most frequent words based on the outcome of this operation (the list of document vectors). As a check, these words should also occur in the word cloud.

```
# Load the library with the CountVectorizer method
from sklearn.feature_extraction.text import CountVectorizer
import numpy as np

import matplotlib.pyplot as plt
import seaborn as sns
sns.set_style('whitegrid')
%matplotlib inline

# Helper function
def plot_10_most_common_words(count_data, count_vectorizer):
  import matplotlib.pyplot as plt
  words = count_vectorizer.get_feature_names()
  total_counts = np.zeros(len(words))
  for t in count_data:
    total_counts+=t.toarray()[0]

  count_dict = (zip(words, total_counts))
  count_dict = sorted(count_dict, key=lambda x:x[1], reverse=True)[0:30]
  words = [w[0] for w in count_dict]
  counts = [w[1] for w in count_dict]
```

```
  x_pos = np.arange(len(words))

  plt.figure(2, figsize=(15, 15/1.6180))
  plt.subplot(title='30 most common words')
  sns.set_context("notebook", font_scale=1.25, rc={"lines.linewidth": 2.5})
  sns.barplot(x_pos, counts, palette="GnBu_d")
  plt.xticks(x_pos, words, rotation=90)
  plt.xlabel('words')
  plt.ylabel('counts')
  plt.show()
  print(words)

# Initialise the count vectorizer with the English stop words
count_vectorizer = CountVectorizer(stop_words='english')

keyword_PK_2018.loc[keyword_PK_2018['keyword_processed'].str.contains('brewery', regex=False)]

PK_top_30 = ['khan', '2016', 'bigg', 'boss', 'cup', 'world', 'vs', 'icc', 'pakistan', 'live', 'ptv', 'sports', 'champions',
'express', 'mir', 'momina', 'mustehsan', 'news', 'psl', 'reham', 'samaa', 'schedule', 'trophy', 'trump', 'tv', 'tiger',
'baloch', 'fast', 'furious', 'qandeel']

def find_category(df, arr):
  new_arr = []
  for kw in arr:
    collection = {}
    new_df = df.loc[df['keyword_processed'].str.contains(kw, regex=False)]
    new_df = new_df.loc[new_df['search_volume'] == '100']
    collection['keyword'] = kw
    collection['year'] = list(new_df['year'])
    collection['category'] = list(new_df['category'])
    collection['all_keyword'] = list(new_df['keyword'])
    collection['region'] = list(new_df['region'])
    new_arr.append(collection)
  return new_arr
```

```
results = find_category(keyword_PK, PK_top_30)
results_df = pd.DataFrame(results)
results_df['year'] = results_df['year'].apply(lambda x: ','.join(map(str, x)))
results_df['category'] = results_df['category'].apply(lambda x: ','.join(map(str, x)))
results_df['all_keyword'] = results_df['all_keyword'].apply(lambda x: ','.join(map(str, x)))
results_df['region'] = results_df['region'].apply(lambda x: ','.join(map(str, x)))
results_df
```

References

Google searches in 1 second, Retrieved from:
https://www.internetlivestats.com/google-search-statistics/#trend

Internet Usage Statistics, Retrieved from:
https://www.internetworldstats.com/stats.htm

HostingFacts Team. Dec 17, 2018, Internet Stats & Facts for 2019. Retrieved from:
https://hostingfacts.com/internet-facts-stats/

Topic Modeling in Python: Latent Dirichlet Allocation (LDA)
https://towardsdatascience.com/end-to-end-topic-modeling-in-python-latent-dirichlet-allocation-lda-35ce4ed6b3e0

wordcloud.ImageColorGenerator. Retrieved from:
https://amueller.github.io/word_cloud/generated/wordcloud.ImageColorGenerator.html

sklearn.feature_extraction.text.CountVectorizer. Retrieved
from:https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.CountVectorizer.html