ted-talks

October 28, 2024

#TED_ED(LESSONS WORTH SHARING)

Founded in 1984 by Richard Saulman as a non profit organisation that aimed at bringing experts from the fields of Technology, Entertainment and Design together, TED Conferences have gone on to become the Mecca of ideas from virtually all walks of life. As of 2015, TED and its sister TEDx chapters have published more than 2000 talks for free consumption by the masses and its speaker list boasts of the likes of Al Gore, Jimmy Wales, Shahrukh Khan and Bill Gates. Ted, which operates under the slogan 'Ideas worth spreading' has managed to achieve an incredible feat of bringing world renowned experts from various walks of life and study and giving them a platform to distill years of their work and research into talks of 18 minutes in length. What's even more incredible is that their invaluable insights is available on the Internet for free. Since the time I begin watching TED Talks in high school, they have never ceased to amaze me. I have learned an incredible amount, about fields I was completely alien to, in the form of poignant stories, breathtaking visuals and subtle humor. So in this notebook, I wanted to attempt at finding insights about the world of TED, its speakers and its viewers and try to answer a few questions that I had always had in the back of my mind. The main dataset contains metadata about every TED Talk hosted on the TED.com website until September 21, 2017.

[30]:
```
# Features Available
# Name: The official name of the TED Talk, including both the title and the␣
↪speaker.
# Title: The title of the talk itself.
# Description: A brief summary or blurb describing what the talk is about.
# Main_speaker: The first named speaker of the talk.
# Speaker_occupation: The occupation or profession of the main speaker.
# Num_speaker: The total number of speakers involved in the talk.
# Duration: The duration of the talk measured in seconds.
# Event: The TED or TEDx event at which the talk was delivered.
# Film_date: The Unix timestamp indicating when the talk was filmed.
# Published_date: The Unix timestamp for when the talk was published on TED.com.
# Comments: The count of first-level comments made on the talk.
# Tags: Themes or keywords associated with the talk.
# Languages: The number of languages in which the talk is available for viewing.
# Ratings: A stringified dictionary representing various ratings given to the␣
↪talk
#           (e.g., inspiring, fascinating, jaw-dropping).
# Related_talks: A list of dictionaries containing recommended talks to watch␣
↪next.
```

```
# url: The URL where the talk can be accessed online.
# Views: The total number of views the talk has received.
```

[1]:
```
# IMPORT LIBRARIES
import pandas as pd
import numpy as np
from scipy  import stats
import seaborn as sns
import matplotlib.pyplot as plt
import json
from pandas import json_normalize
! pip install wordcloud
from wordcloud import WordCloud, STOPWORDS
month_order =␣
 ↪['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
day_order = ['Mon','Tue','Wed','Thu','Fri','Sat','Sun']
```

```
Requirement already satisfied: wordcloud in c:\users\91949\anaconda3\lib\site-
packages (1.9.3)
Requirement already satisfied: numpy>=1.6.1 in
c:\users\91949\anaconda3\lib\site-packages (from wordcloud) (1.26.4)
Requirement already satisfied: pillow in c:\users\91949\anaconda3\lib\site-
packages (from wordcloud) (10.2.0)
Requirement already satisfied: matplotlib in c:\users\91949\anaconda3\lib\site-
packages (from wordcloud) (3.8.0)
Requirement already satisfied: contourpy>=1.0.1 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.2.0)
Requirement already satisfied: cycler>=0.10 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (0.11.0)
Requirement already satisfied: fonttools>=4.22.0 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (4.25.0)
Requirement already satisfied: kiwisolver>=1.0.1 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (1.4.4)
Requirement already satisfied: packaging>=20.0 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (23.1)
Requirement already satisfied: pyparsing>=2.3.1 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (3.0.9)
Requirement already satisfied: python-dateutil>=2.7 in
c:\users\91949\anaconda3\lib\site-packages (from matplotlib->wordcloud) (2.8.2)
Requirement already satisfied: six>=1.5 in c:\users\91949\anaconda3\lib\site-
packages (from python-dateutil>=2.7->matplotlib->wordcloud) (1.16.0)
```

[2]:
```
df = pd.read_csv("D:/Top Mentor/Projects Set 1 - 1 to 6 Topics/3 Project -␣
 ↪Analytics/Project 1 - Analyze Ted Talks/ted_main.csv")
```

[3]:
```
df.columns
```

[3]:
```
Index(['comments', 'description', 'duration', 'event', 'film_date',
       'languages', 'main_speaker', 'name', 'num_speaker', 'published_date',
       'ratings', 'related_talks', 'speaker_occupation', 'tags', 'title',
       'url', 'views'],
      dtype='object')
```

[4]:
```
#I'm just going  to reorder the columns in the order I've listed the features␣
 ↪for my convenience
df =␣
 ↪df[['name','title','description','main_speaker','speaker_occupation','num_speaker','duratio
         'languages','ratings','related_talks','url','views']]
```

[5]:
```
df.columns
```

[5]:
```
Index(['name', 'title', 'description', 'main_speaker', 'speaker_occupation',
       'num_speaker', 'duration', 'event', 'film_date', 'published_date',
       'comments', 'tags', 'languages', 'ratings', 'related_talks', 'url',
       'views'],
      dtype='object')
```

[6]:
```
#Before  we go any further, let us convert the Unix timestamps into a human␣
 ↪readable format.
import datetime

def convert_to_date(x):
    try:
        # Attempt to treat x as a Unix timestamp
        return datetime.datetime.fromtimestamp(int(x)).strftime('%d-%m-%y')
    except ValueError:
        # If x is not a Unix timestamp, return it as is
        return x

df['film_date'] = df['film_date'].apply(convert_to_date)
```

Analysis 1: Most Viewed Talks of All Time: For starters, let us perform some easy analysis. I want to know what the 15 most viewed TED talks of all time are. The number of views gives us a good idea of the popularity of the TED Talk

Observations ->Ken Robinson’s talk on Do Schools Kill Creativity? is the most popular TED Talk of all time with 47.2 million views. -> Also coincidentally, it is also one of the first talks to ever be uploaded on the TED Site (the main dataset is sorted by published date). ->Robinson’s talk is closely followed by Amy Cuddy’s talk on Your Body Language May Shape Who You Are. ->There are only 2 talks that have surpassed the 40 million mark and 4 talks that have crossed the 30 million mark.

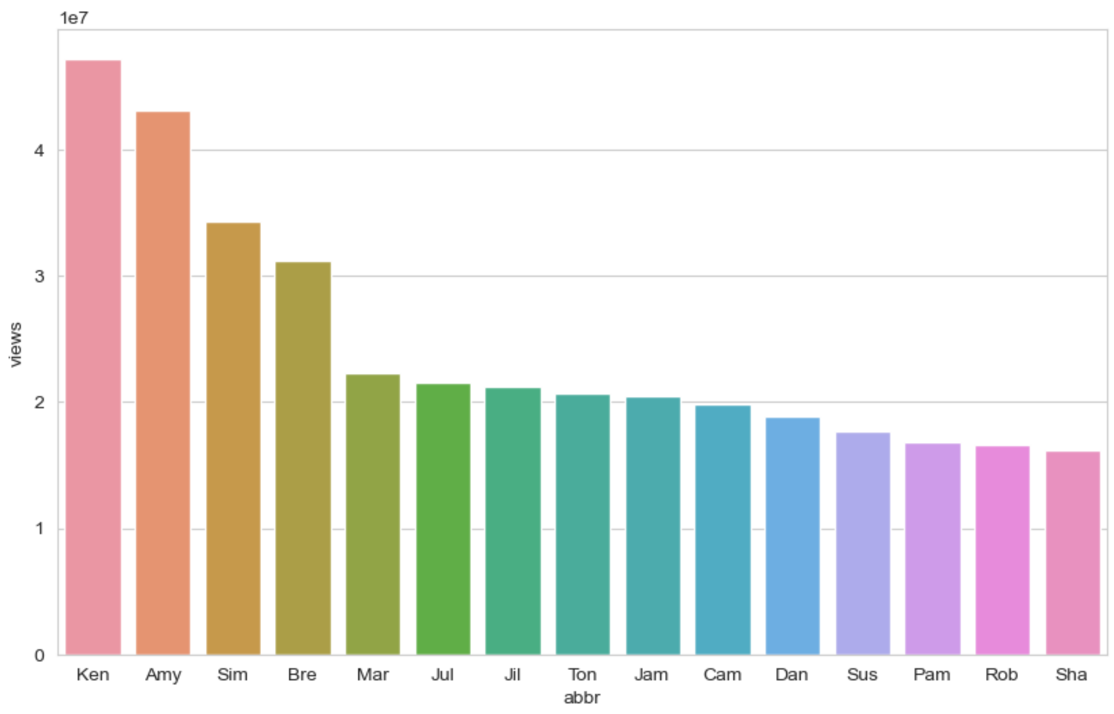
[7]:
```
pop_talks = df[['title','main_speaker','views','film_date']].
 ↪sort_values('views',ascending=False)[:15]
print(pop_talks)
```

```
                                                  title       main_speaker  \
0                           Do schools kill creativity?       Ken Robinson
1346                Your body language may shape who you are          Amy Cuddy
677                      How great leaders inspire action        Simon Sinek
837                           The power of vulnerability        Brené Brown
452                 10 things you didn't know about orgasm        Mary Roach
1776           How to speak so that people want to listen    Julian Treasure
201                               My stroke of insight  Jill Bolte Taylor
5                                   Why we do what we do        Tony Robbins
2114    This is what happens when you reply to spam email       James Veitch
1416    Looks aren't everything. Believe me, I'm a model.    Cameron Russell
500                           The puzzle of motivation           Dan Pink
1163                            The power of introverts         Susan Cain
1036                                  How to spot a liar       Pamela Meyer
2109    What makes a good life? Lessons from the longe…  Robert Waldinger
1129                        The happy secret to better work        Shawn Achor

         views film_date
0     47227110  25-02-06
1346  43155405  26-06-12
677   34309432  17-09-09
837   31168150  06-06-10
452   22270883  06-02-09
1776  21594632  10-06-13
201   21190883  27-02-08
5     20685401  02-02-06
2114  20475972  08-12-15
1416  19787465  27-10-12
500   18830983  24-07-09
1163  17629275  28-02-12
1036  16861578  13-07-11
2109  16601927  14-11-15
1129  16209727  11-05-11
```

Analysis 2: Let us make a bar chart to visualize these 15 talks in terms of the number of views they garnered.

[8]:
```
pop_talks['abbr'] = pop_talks['main_speaker'].apply(lambda x: x[:3])
sns.set_style('whitegrid')
plt.figure(figsize=(10,6))
sns.barplot(x='abbr',y='views',data = pop_talks)
```

[8]:
```
<Axes: xlabel='abbr', ylabel='views'>
```

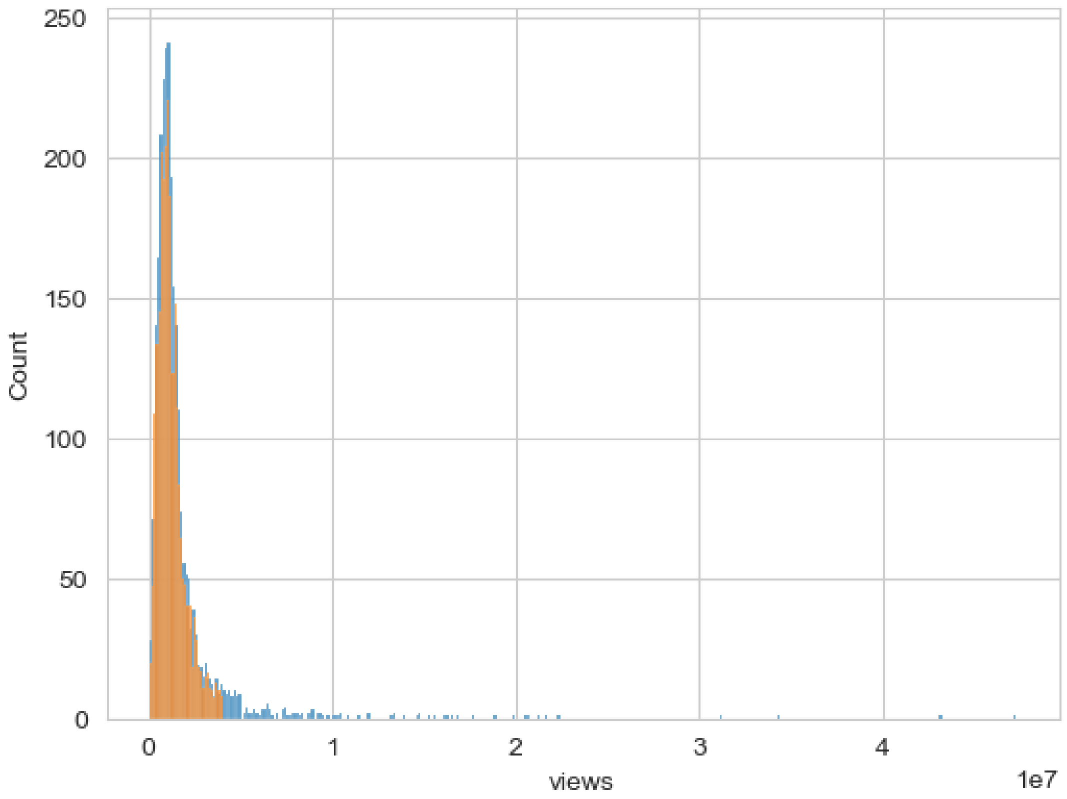


Let us investigate the summary statistics and the distribution of the views granted on various TED Talks.

```
[9]: sns.histplot(df['views'])
     sns.histplot(df[df['views']<0.4e7]['views'])
```

```
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

```
[9]: <Axes: xlabel='views', ylabel='Count'>
```



Calculate and verify the average number of views on TED Talks in 1.6 million. and the median number of views is 1.12 million. This suggests a very high average level of popularity of TED Talks. We also notice that the majority of talks have views less than 4 million. We will consider this as the cutoff point when constructing box plots in the later sections

[10]:
```
df['views'].describe()
```

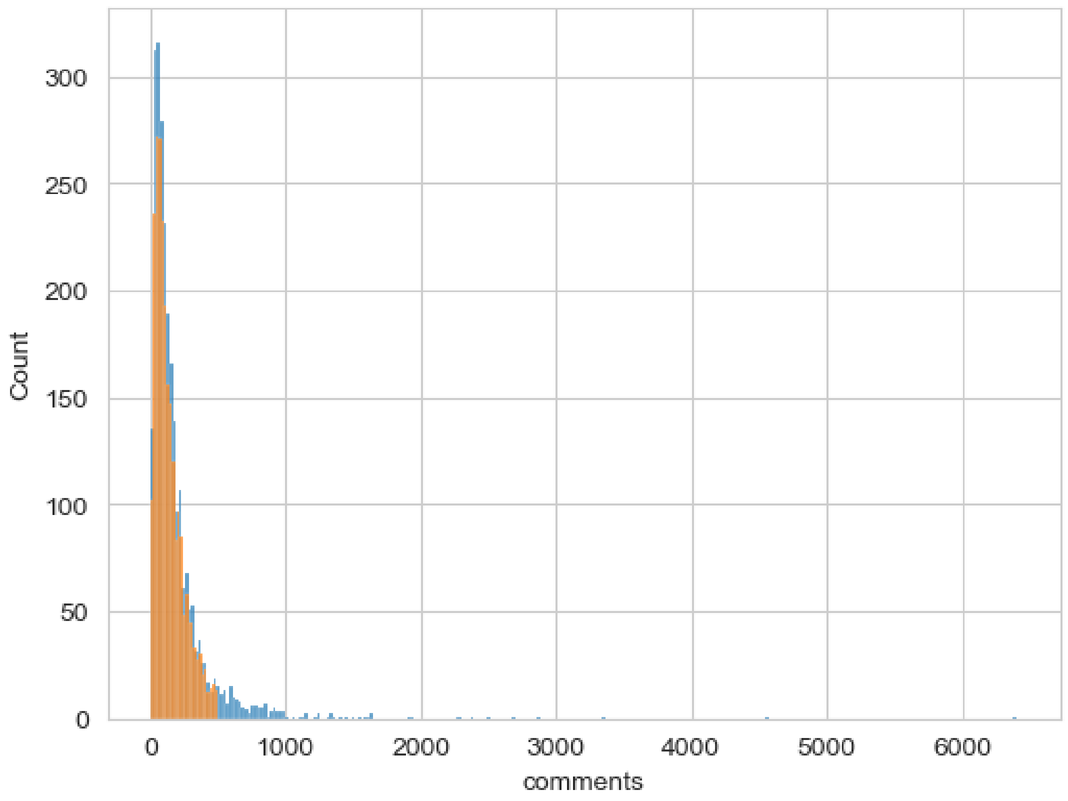
[10]:
```
count    2.550000e+03
mean     1.698297e+06
std      2.498479e+06
min      5.044300e+04
25%      7.557928e+05
50%      1.124524e+06
75%      1.700760e+06
max      4.722711e+07
Name: views, dtype: float64
```

[31]:
```
# Analysis 5: Performing textual analysis of comments

# Observations:
# 1. On average, there are 191.5 comments on every TED Talk.
#    Assuming the comments are constructive criticism,
#    this suggests that the TED Online Community is highly involved in␣
↪discussions
#    surrounding TED Talks.

# 2. There is a significant standard deviation associated with the number of␣
↪comments.
#    In fact, the standard deviation is larger than the mean,
#    indicating that the measures may be sensitive to outliers.
#    We will plot this data to examine the nature of the distribution.

# 3. The minimum number of comments on a talk is 2, and the maximum is 6404.
#    The range of comments is therefore 6402 (6404 - 2).
#    The minimum count may be attributed to talks that were posted very␣
↪recently,
#    resulting in fewer comments at the time of analysis.
```

[11]:
```
df['comments'].describe()
sns.histplot(df['comments'])
sns.histplot(df[df['comments']<500]['comments'])
```

```
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

[11]:
```
<Axes: xlabel='comments', ylabel='Count'>
```

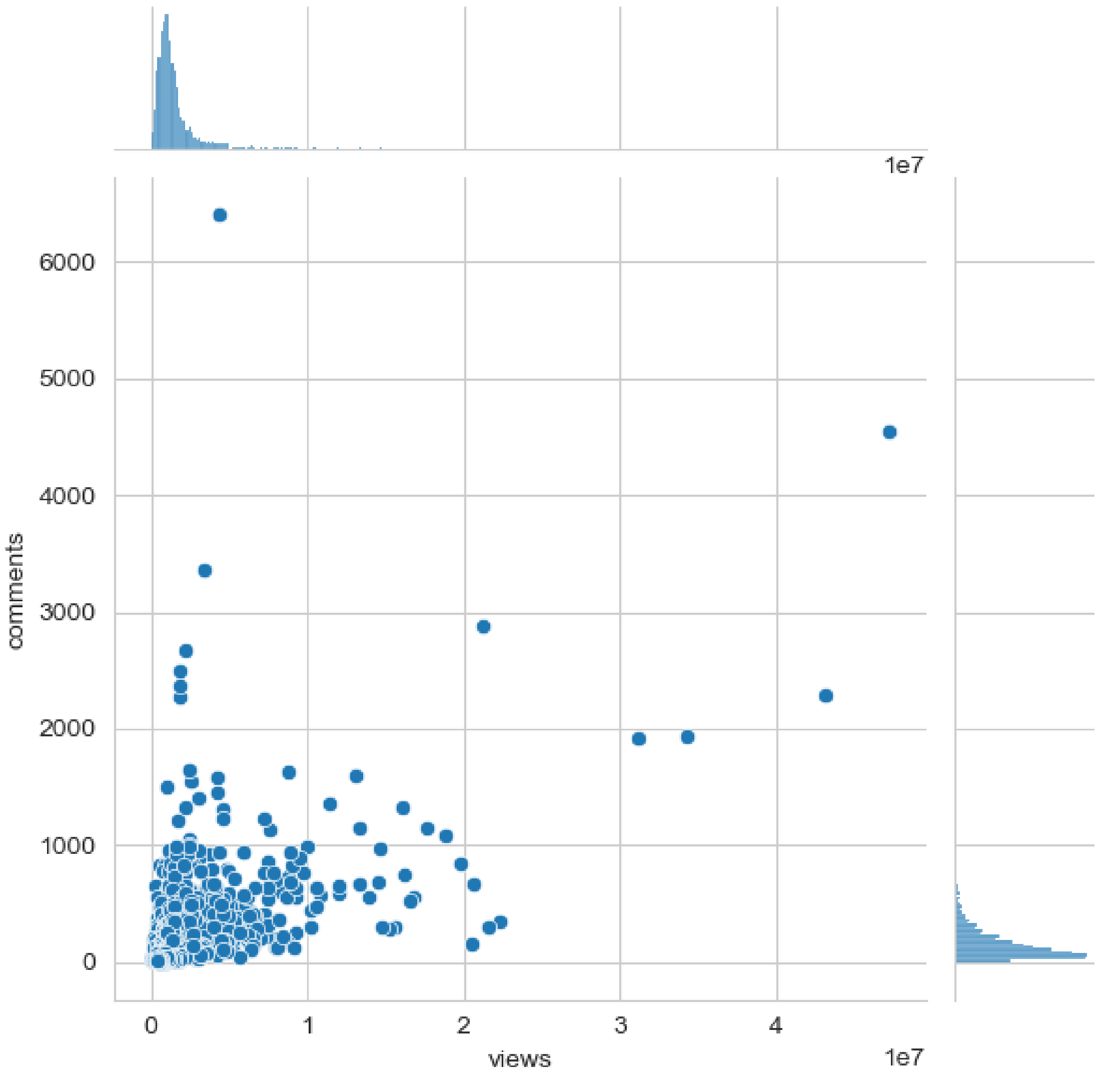


If the number of views is correlated with the number of comments. we should think that this is the case as more popular videos tend to have more comments.

[12]:
```
sns.jointplot(x='views', y ='comments', data = df)
df[['views','comments']].corr()
```

```
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

[12]:
```
             views  comments
views     1.000000  0.530939
comments  0.530939  1.000000
```



Let us now check the number of views and comments on the 10 most commented TED TALKS of all time

[13]:
```
df[['title', 'main_speaker','views','comments']].
↪sort_values('comments',ascending=False).head(10)
```

[13]:
```
                                   title        main_speaker     views  \
96                      Militant atheism     Richard Dawkins   4374792
0                Do schools kill creativity?      Ken Robinson  47227110
644          Science can answer moral questions          Sam Harris   3433437
201                  My stroke of insight  Jill Bolte Taylor  21190883
1787       How do you explain consciousness?     David Chalmers   2162764
954              Taking imagination seriously     Janet Echelman   1832930
840                   On reading the Koran     Lesley Hazleton   1847256
```

```
1346  Your body language may shape who you are          Amy Cuddy  43155405
661               The danger of science denial    Michael Specter   1838628
677           How great leaders inspire action        Simon Sinek  34309432

      comments
96        6404
0         4553
644       3356
201       2877
1787      2673
954       2492
840       2374
1346      2290
661       2272
677       1930
```

Discussion quotient which is simply the ratio of the number of comments to the number of views

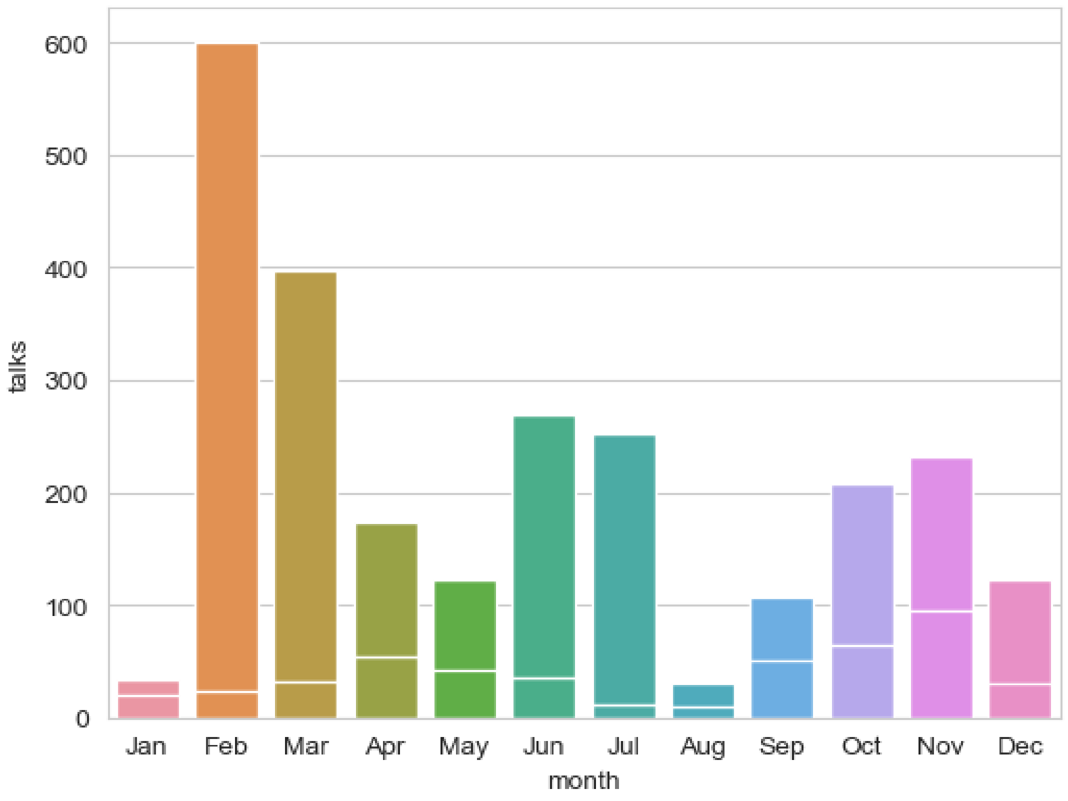
[14]:
```
df['dis_quo'] =df ['comments']/df['views']
df[['title','main_speaker','views','comments','dis_quo','film_date']].
↪sort_values('dis_quo',ascending=False).head(10)
```

[14]:
```
                                    title          main_speaker    views  \
744        The case for same-sex marriage        Diane J. Savino   292395
803                E-voting without fraud         David Bismark   543551
96                       Militant atheism       Richard Dawkins  4374792
694   Inside a school for suicide bombers  Sharmeen Obaid-Chinoy  1057238
954          Taking imagination seriously         Janet Echelman  1832930
840                    On reading the Koran       Lesley Hazleton  1847256
876          Curating humanity's heritage      Elizabeth Lindsey   439180
1787    How do you explain consciousness?        David Chalmers  2162764
661          The danger of science denial        Michael Specter  1838628
561                 Dance to change the world      Mallika Sarabhai   481834

      comments   dis_quo film_date
744        649  0.002220  02-12-09
803        834  0.001534  14-07-10
96        6404  0.001464  02-02-02
694       1502  0.001421  10-02-10
954       2492  0.001360  03-03-11
840       2374  0.001285  10-10-10
876        555  0.001264  08-12-10
1787      2673  0.001236  18-03-14
661       2272  0.001236  11-02-10
561        595  0.001235  04-11-09
```

Analysing TED TALKS by the month and the year

[15]:
```
df['month'] = df['film_date'].apply(lambda x: month_order[int(x.
 ↪split('-')[1])-1])

month_df = pd.DataFrame(df['month'].value_counts()).reset_index()
month_df.columns = ['month','talks']
sns.barplot(x='month', y = 'talks', data = month_df,order = month_order)
df_x = df[df['event'].str.contains('TEDx')]
x_month_df = pd.DataFrame(df_x['month'].value_counts().reset_index())
x_month_df.columns = ['month','talks']
sns.barplot(x = 'month',y ='talks', data = x_month_df, order = month_order)
```

[15]:
```
<Axes: xlabel='month', ylabel='talks'>
```

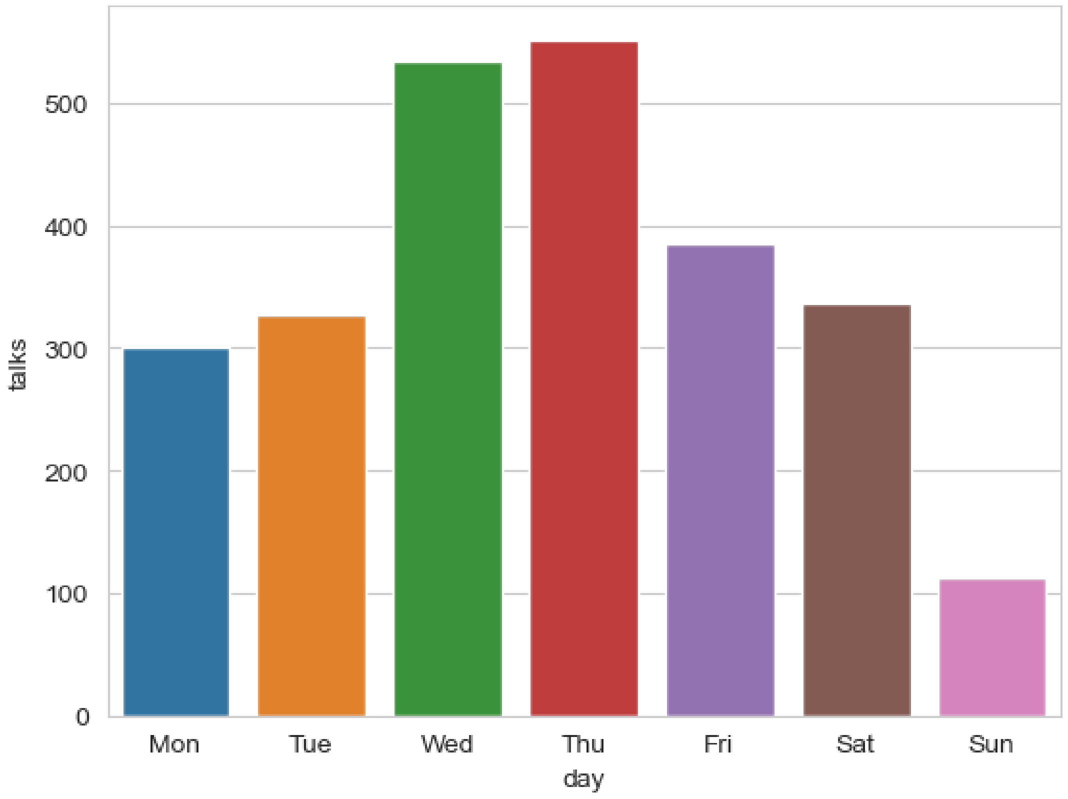


The most popular days for conducting TED and TEDx conferences

[16]:
```
def getday(x):
    day, month, year = (int(i) for i in x.split('-'))
    answer = datetime.date(year,month,day).weekday()
    return day_order[answer]

df['day'] = df['film_date'].apply(getday)
```

```
day_df = pd.DataFrame(df['day'].value_counts()).reset_index()
day_df.columns = ['day','talks']
sns.barplot(x ='day', y ='talks', data = day_df, order = day_order)
```

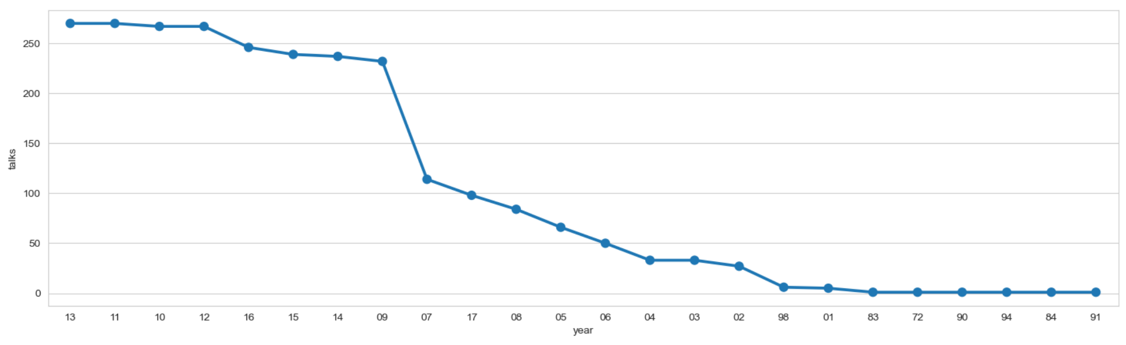
```
[16]: <Axes: xlabel='day', ylabel='talks'>
```



Let us know the visualize the number of TED Talks through the years

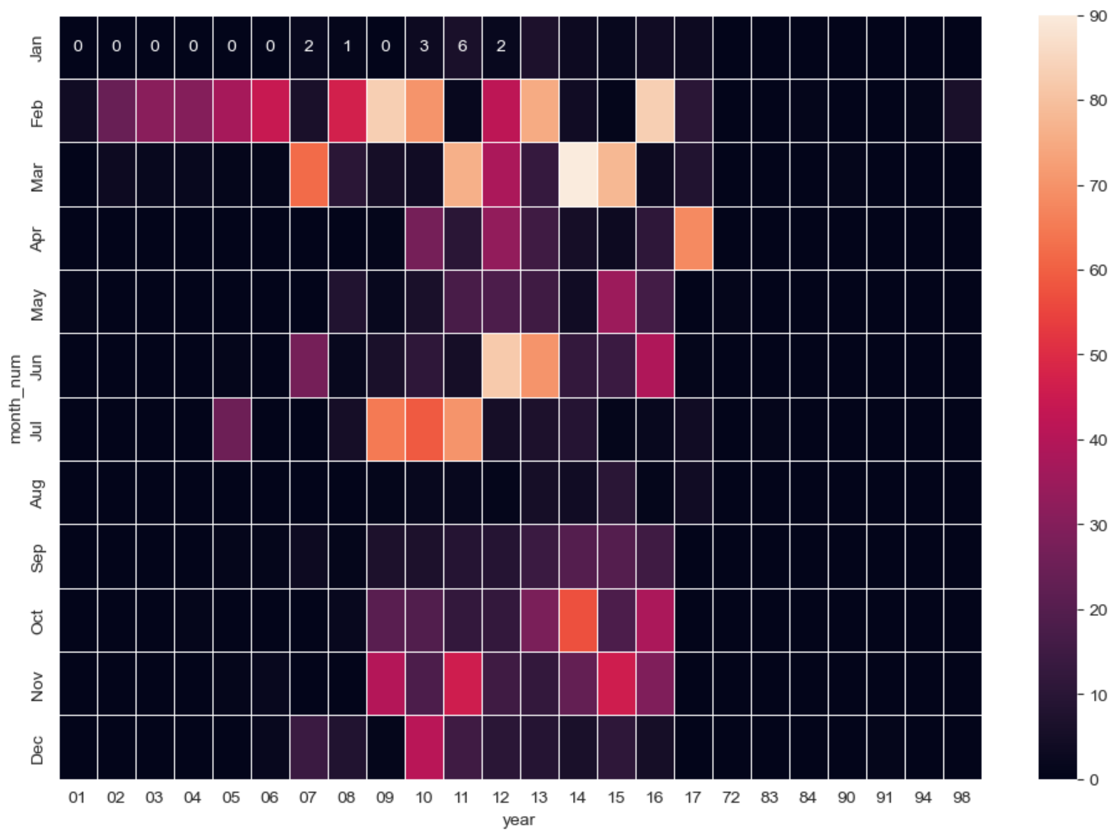
```
[17]: df['year'] = df['film_date'].apply(lambda x: x.split('-')[2])
year_df = pd.DataFrame(df['year'].value_counts().reset_index())
year_df.columns = ['year','talks']
plt.figure(figsize = (18,5))
sns.pointplot(x='year', y ='talks',data = year_df)
```

```
[17]: <Axes: xlabel='year', ylabel='talks'>
```



Let us construct a heat map that shows us the number of talks by month and year

[18]:
```
months = {
    'Jan': 1, 'Feb': 2, 'Mar': 3, 'Apr': 4, 'May': 5,
    'Jun': 6, 'Jul': 7, 'Aug': 8, 'Sep': 9, 'Oct': 10,
    'Nov': 11, 'Dec': 12
}
month_order = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep',␣
 ↪'Oct', 'Nov', 'Dec']

# Copying df to hmap_df and formatting 'film_date'
hmap_df = df.copy()
hmap_df['film_date'] = hmap_df['film_date'].apply(lambda x: month_order[int(x.
 ↪split('-')[1]) - 1] + " " + x.split('-')[2])

# Pivoting and formatting for heatmap
hmap_df = pd.pivot_table(hmap_df[['film_date', 'title']], index='film_date',␣
 ↪aggfunc='count').reset_index()
hmap_df['month_num'] = hmap_df['film_date'].apply(lambda x: months[x.
 ↪split()[0]])
hmap_df['year'] = hmap_df['film_date'].apply(lambda x: x.split()[1])
hmap_df = hmap_df.sort_values(['year', 'month_num'])

# Reshaping for heatmap
hmap_df = hmap_df[['month_num', 'year', 'title']]
hmap_df = hmap_df.pivot(index = 'month_num',columns=  'year',values= 'title')
hmap_df = hmap_df.fillna(0)

# Plotting the heatmap
f, ax = plt.subplots(figsize=(12, 8))
sns.heatmap(hmap_df, annot=True, linewidths=.5, ax=ax, fmt='g',␣
 ↪yticklabels=month_order)
plt.show()
```



TFD SPEAKERS

```
[19]: speaker_df = df.groupby('main_speaker').count().
       ↪reset_index()[['main_speaker','comments']]
      speaker_df.columns = ['main_speaker','appearances']
      speaker_df = speaker_df.sort_values('appearances',ascending =False)
      speaker_df.head(10)
```

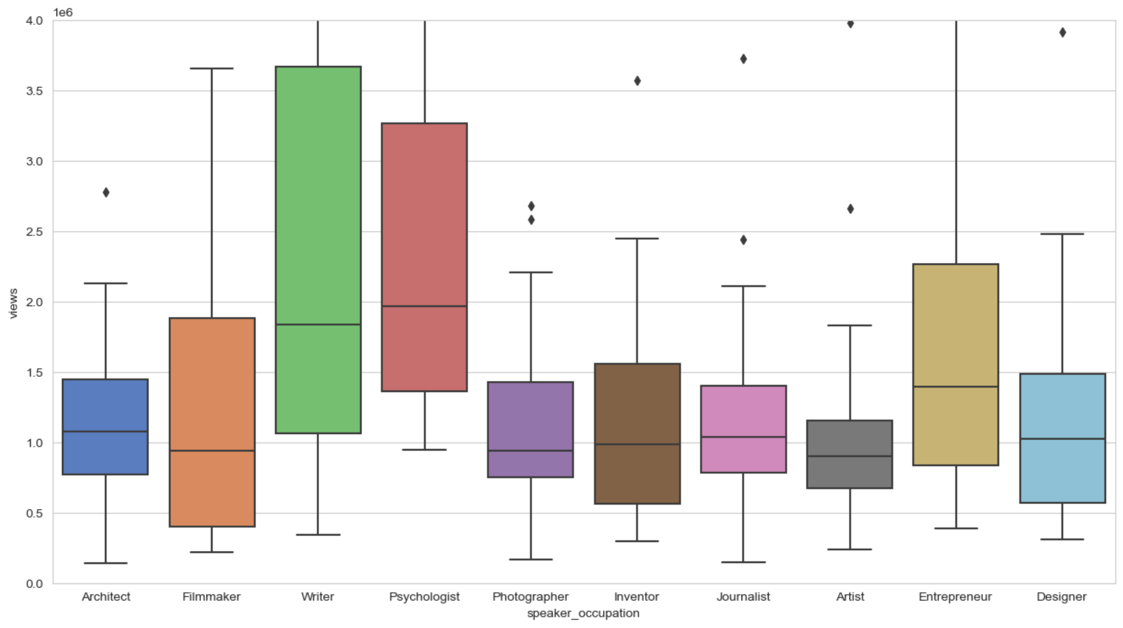
```
[19]:              main_speaker  appearances
      770            Hans Rosling            9
      1066          Juan Enriquez            7
      1693                  Rives            6
      1278          Marco Tempest            6
      397             Clay Shirky            5
      1487    Nicholas Negroponte            5
      1075         Julian Treasure            5
      424              Dan Ariely            5
      850    Jacqueline Novogratz            5
      248              Bill Gates            5
```

[20]:
```
occupation_df = df.groupby('speaker_occupation').count().
↪reset_index()[['speaker_occupation','comments']]
occupation_df.columns = ['occupation', 'appearances']
occupation_df = occupation_df.sort_values('appearances',ascending = False)
occupation_df.head(10)
```

[20]:
```
        occupation  appearances
1426        Writer           45
83          Artist           34
413       Designer           34
753     Journalist           33
515   Entrepreneur           31
71       Architect           30
733       Inventor           27
1131  Psychologist           26
1011  Photographer           25
567      Filmmaker           21
```

Doing Some professions tend to attract a larger number of viewers

[21]:
```
fig,ax = plt.subplots(nrows=1, ncols =1,figsize=(15,8))
sns.boxplot(x='speaker_occupation',y='views',data=df[df['speaker_occupation'].
↪isin(occupation_df.head(10)[
                                                                              ␣
↪   'occupation'])],palette ='muted',ax =ax)
ax.set_ylim([0,0.4e7])
plt.show()
```



Finally, Let us check the number of talks which have had more than one speaker

```
[22]: df['num_speaker'].value_counts()
      df[df['num_speaker'] == 5] [['title','description','main_speaker','event']]
```

```
[22]:                               title  \
      2507  A dance to honor Mother Earth

                                                  description  \
      2507  Movement artists Jon Boogz and Lil Buck debut …

                      main_speaker    event
      2507  Jon Boogz and Lil Buck  TED2017
```

TED Events Which TED Events tend to hold the most number of TED.com Upload worth events

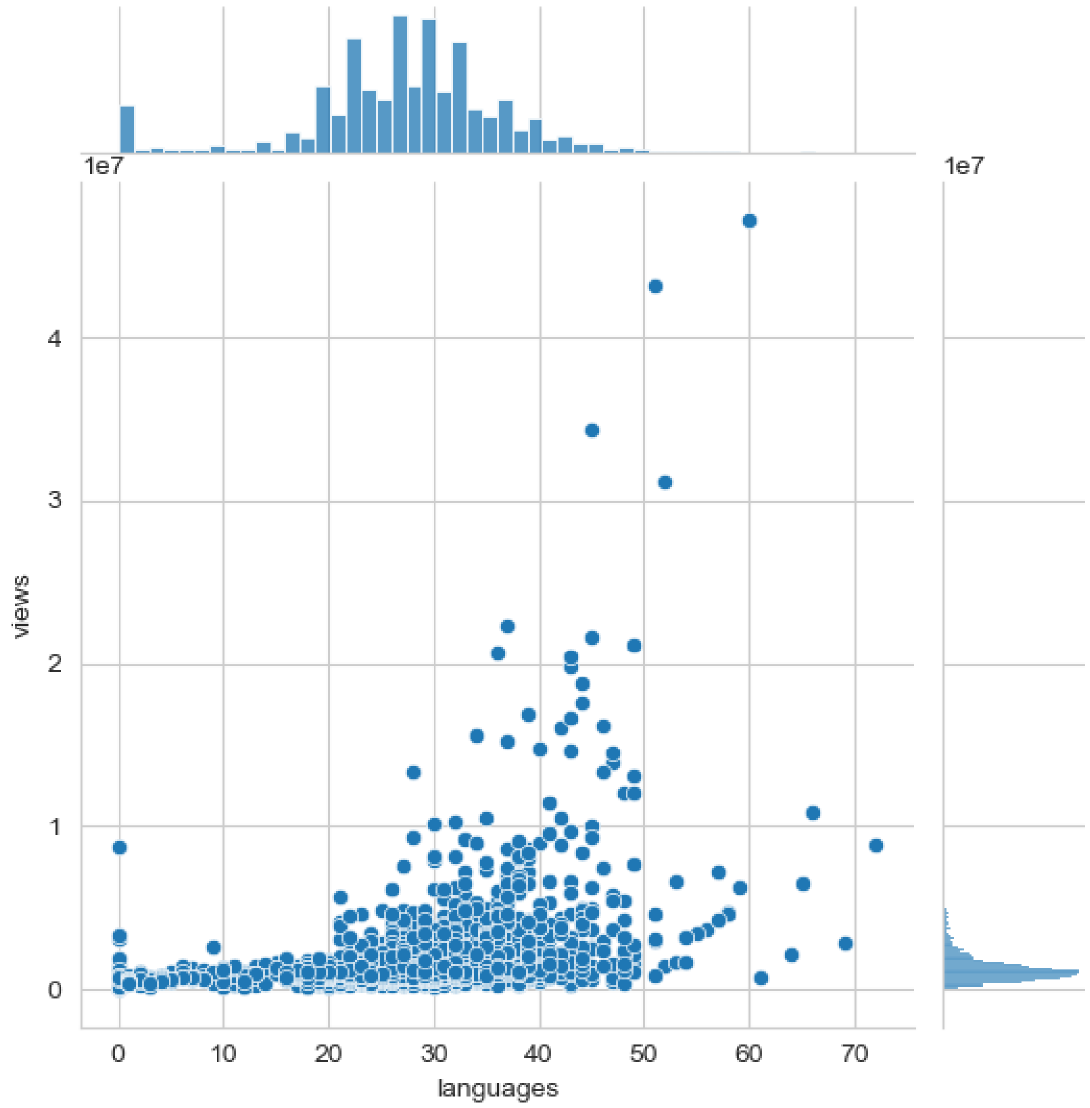
```
[23]: events_df = df[['title','event']].groupby('event').count().reset_index()
      events_df.columns =['event','talks']
      events_df = events_df.sort_values('talks',ascending= False)
      events_df.head(10)
```

```
[23]:             event  talks
      64          TED2014     84
      59          TED2009     83
      63          TED2013     77
      66          TED2016     77
      65          TED2015     75
      99   TEDGlobal 2012     70
      61          TED2011     70
      60          TED2010     68
      98   TEDGlobal 2011     68
      57          TED2007     68
```

TED Languages One remarkable aspect of TED Talks is the Sheer number of languages in which is accessible

```
[24]: df['languages'].describe()
      df[df['languages']==72]
      sns.jointplot(x='languages',y='views',data=df)
      plt.show()
```

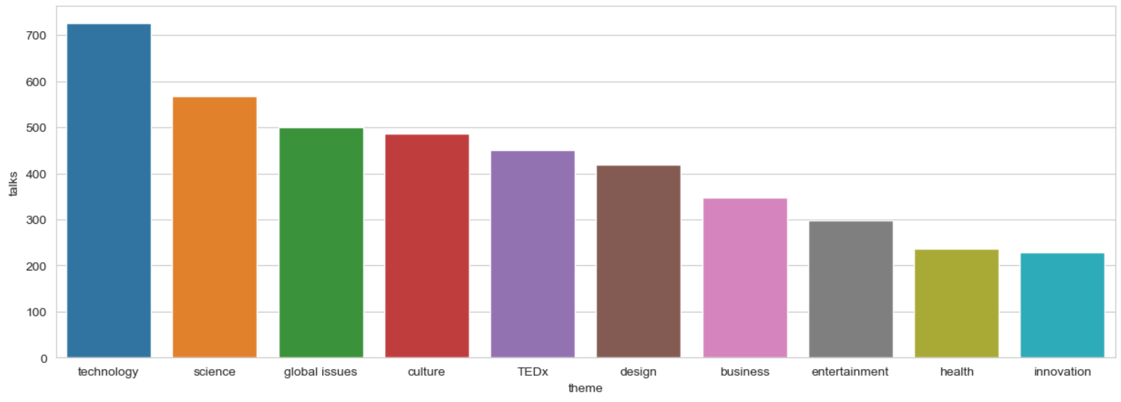
```
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
C:\Users\91949\anaconda3\Lib\site-packages\seaborn\_oldcore.py:1119:
FutureWarning: use_inf_as_na option is deprecated and will be removed in a
future version. Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```

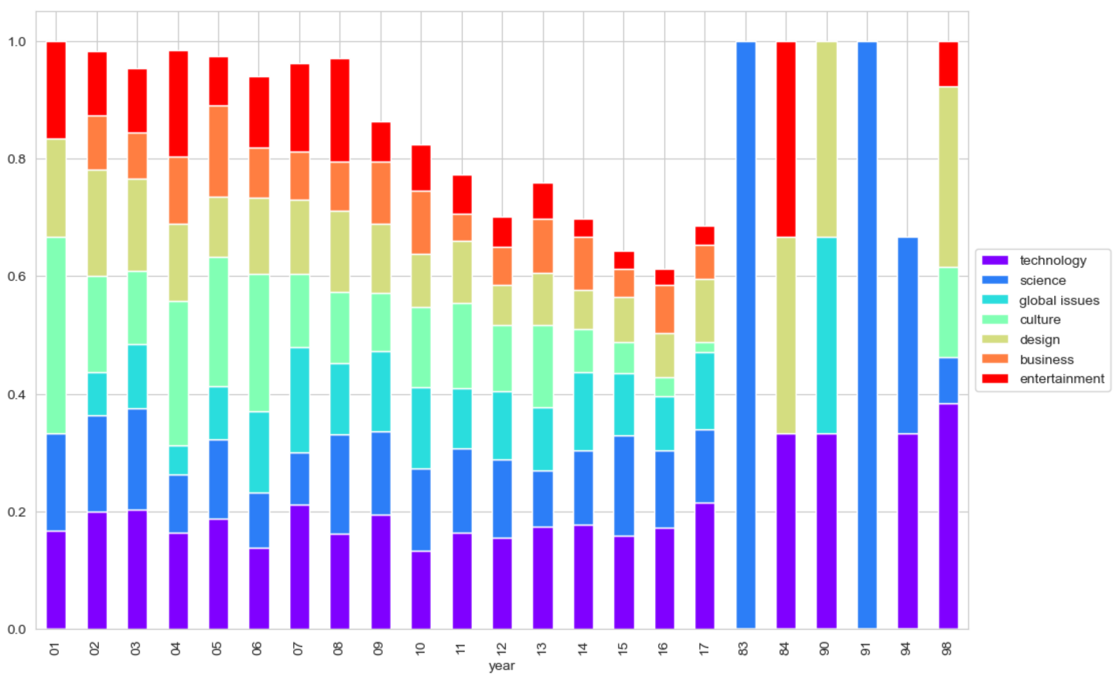


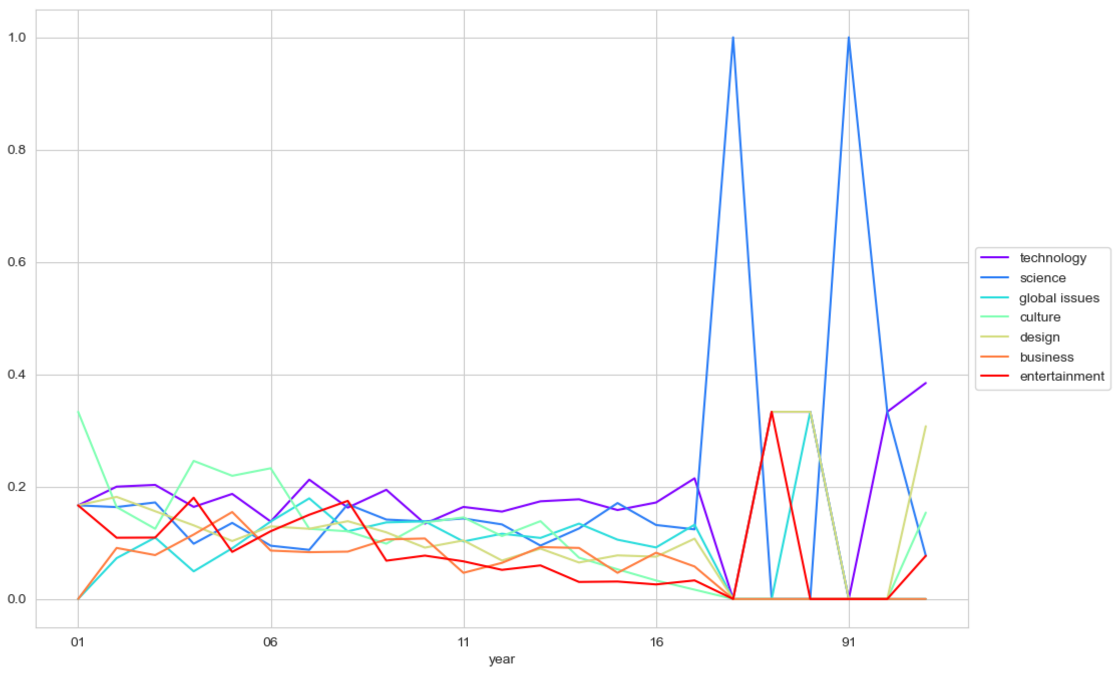
TED THEMES

[25]:
```
import ast
df['tags'] = df ['tags'].apply(lambda x: ast.literal_eval(x) if␣
 ↪isinstance(x,str) else x)
s = df.apply(lambda x: pd.Series(x['tags']),axis =1).stack().
 ↪reset_index(level=1,drop=True)
s.name ='theme'
theme_df = df.drop('tags',axis=1).join(s)
theme_df.head()

len(theme_df['theme'].value_counts())
pop_themes = pd.DataFrame(theme_df['theme'].value_counts()).reset_index()
```

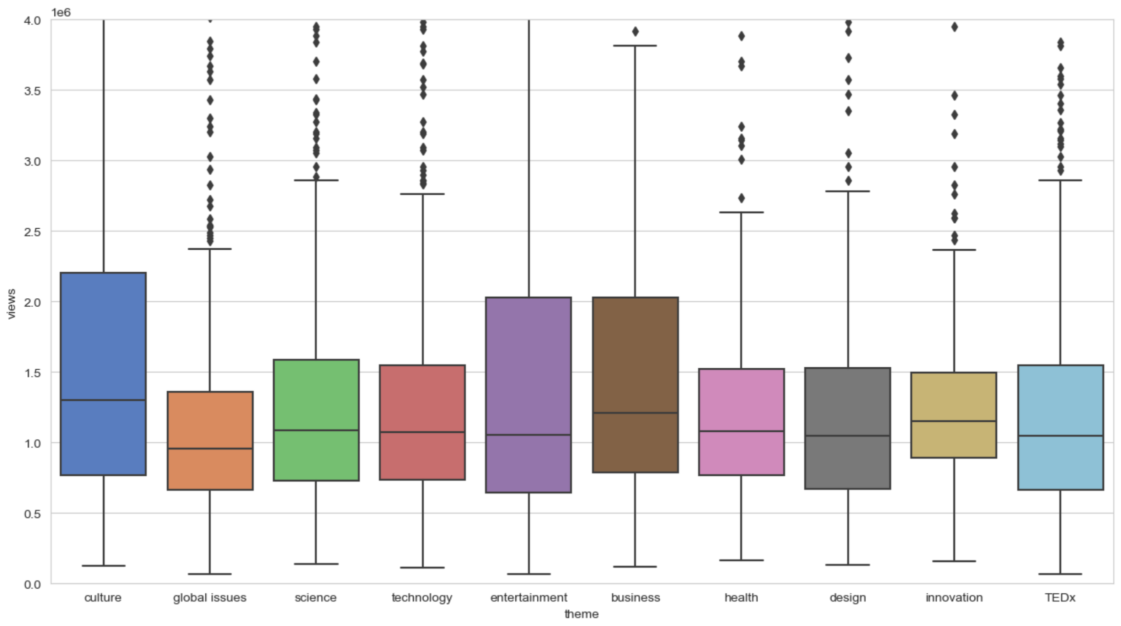
```
pop_themes.columns=['theme','talks']
pop_themes.head(10)
plt.figure(figsize =(15,5))
sns.barplot(x='theme', y ='talks', data = pop_themes.head(10))
plt.show()

themes = list(pop_themes.head(8)['theme'])
themes.remove('TEDx')
pop_theme_talks = theme_df[theme_df['theme'].isin(pop_themes.head(10)['theme'])]
ctab = pd.crosstab([pop_theme_talks['year']],
                   pop_theme_talks['theme']).apply(lambda x: x/x.sum(), axis=1)
ctab[themes].plot(kind='bar', stacked =True,
colormap='rainbow',figsize=(12,8)).legend(loc='center left',bbox_to_anchor=(1,0.
 ↪5))
plt.show()

ctab[themes].plot(kind='line',stacked=False,
                  colormap ='rainbow',figsize=(12,8)).legend(loc='center␣
 ↪left',bbox_to_anchor=(1,0.5))
plt.show()

pop_theme_talks = theme_df[theme_df['theme'].isin(pop_themes.head(10)['theme'])]
fig,ax = plt.subplots(nrows =1, ncols =1,figsize =(15,8))
sns.boxplot(x='theme', y ='views', data = pop_theme_talks,palette = 'muted',ax␣
 ↪= ax)
ax.set_ylim([0,0.4e7])
```







```
[25]: (0.0, 4000000.0)
```



Talk Duration and Word Counts Convert to minute