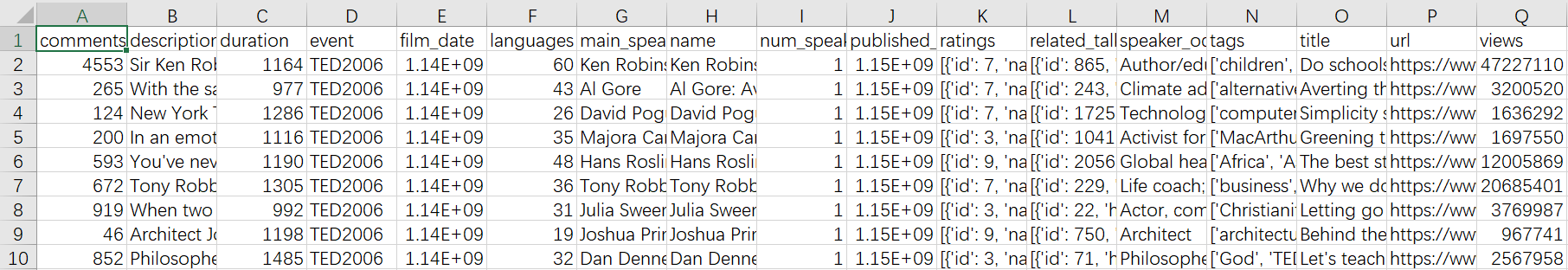
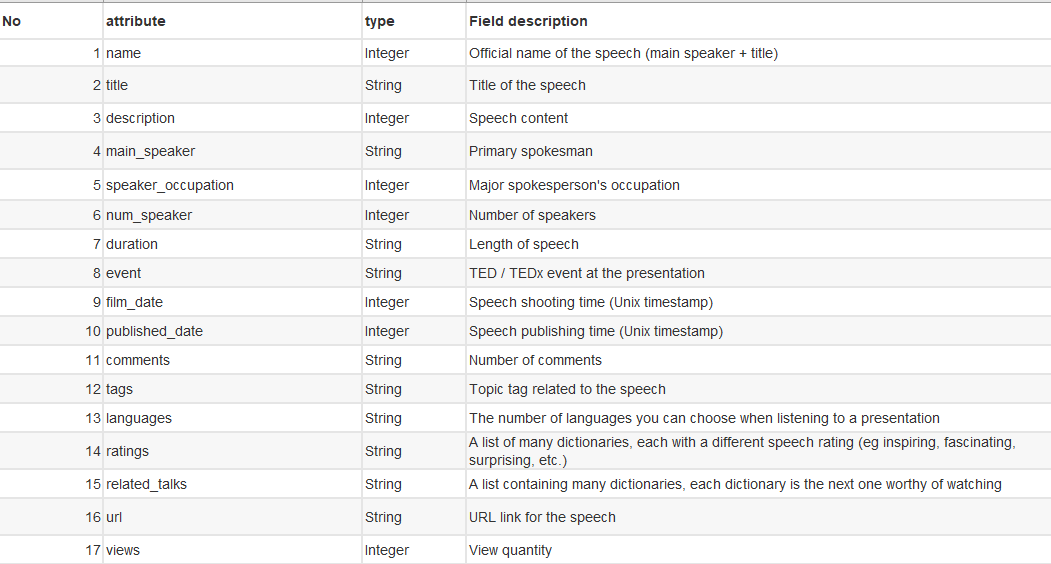
**Analysis of Ted Talks**

**——Analysis by Wang Zhihang**

First download and open the CSV file from kaggle and observe its characteristics. The data set is a 2550 \* 17 data set.



## Features



Preparing

Load library

import pandas as pd

import matplotlib.pyplot as plt

import datetime

import seaborn as sns

import ast

import numpy as np

Read and clean the data

pd.set\_option('display.max\_columns', None)

data=pd.read\_csv("C:/Users/w/a1/ted\_main.csv")

df = data.dropna() #clean

#print(df.isnull().any())

rename the columns

df.columns = ['comments','description','duration','event','film\_date','languages','main\_speaker','name','num\_speaker','published\_date','ratings','related\_talks','speaker\_occupation','tags','title','url','views']

df2 = df.copy()

Remove the useless columns and reorder the columns

df.drop(['description', 'name', 'speaker\_occupation','url','num\_speaker', 'ratings','related\_talks','speaker\_occupation','tags' ],axis= 1,inplace=True)

df = df[['title','main\_speaker', 'views','comments','event', 'duration','film\_date', 'published\_date', 'languages']]

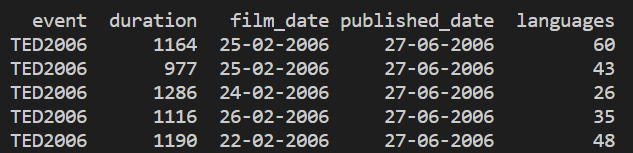
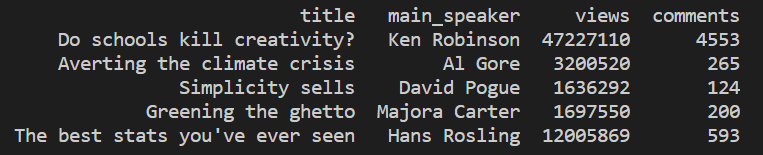
Analysis：

1.Overall analysis

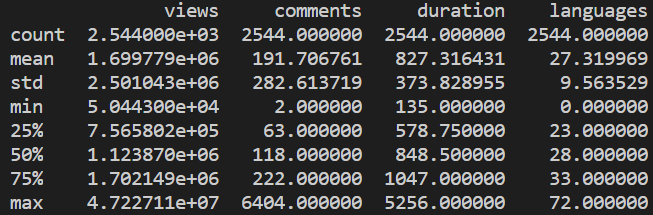
print(df.head(5))

print(df.describe())

Several useful columns such as title maispeaker views comments event are retained



We can see the overall situation of digital data such as views, comments, duration, languages, etc.



Analyze the correlation diagram between these data

correlations = df.corr()

# plot correlation matrix

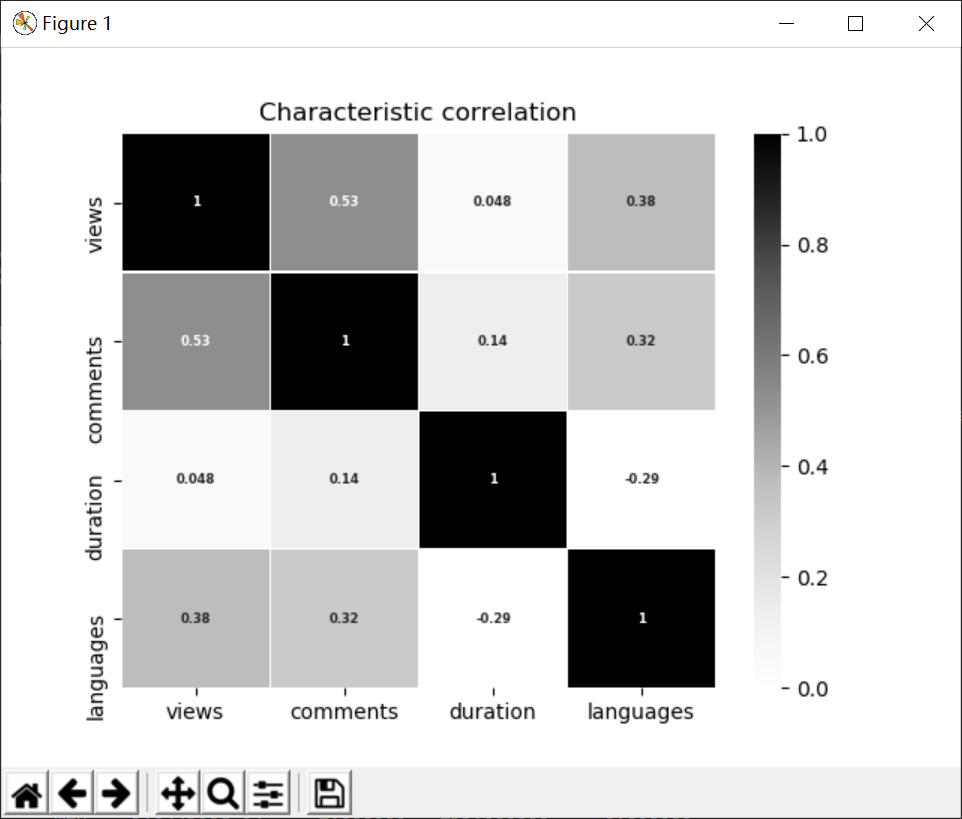
fig = plt.figure()

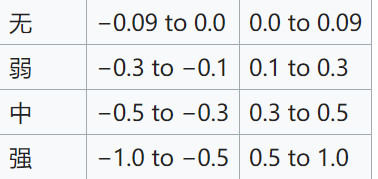
ax = fig.add\_subplot(figsize=(20,20)) #size 20\*20

ax = sns.heatmap(correlations,cmap=plt.cm.Greys, linewidths=0.05,vmax=1, vmin=0 ,annot=True,annot\_kws={'size':6,'weight':'bold'})

ax.set\_title('Characteristic correlation')#set the title

plt.show()





According to the Pearson product-moment **correlation** coefficient，  
Greater than 0.5 has a strong association, so comments and views have a strong association, the number of comments and the number of languages ​​are moderately related, and the number of languages ​​and pageviews are also moderately correlated.

2.views data analysis

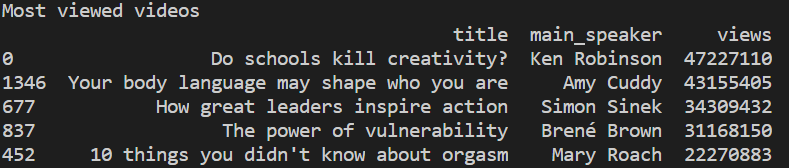
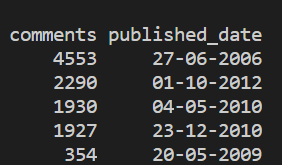
a = df.sort\_values("views",inplace=False,ascending=False)

a = a[['title', 'main\_speaker', 'views', 'comments','published\_date']]

print("Most viewed videos")

print(a.head())

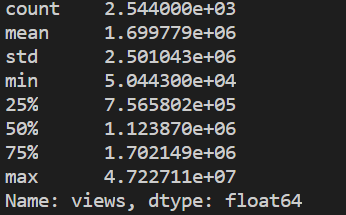
These are the most viewed videos among so many videos.

The most viewed is the Do schools kill creativity? The pageview reached an astonishing 47.2billion, equivalent to a total population of Spain

Analyze pageview digital features

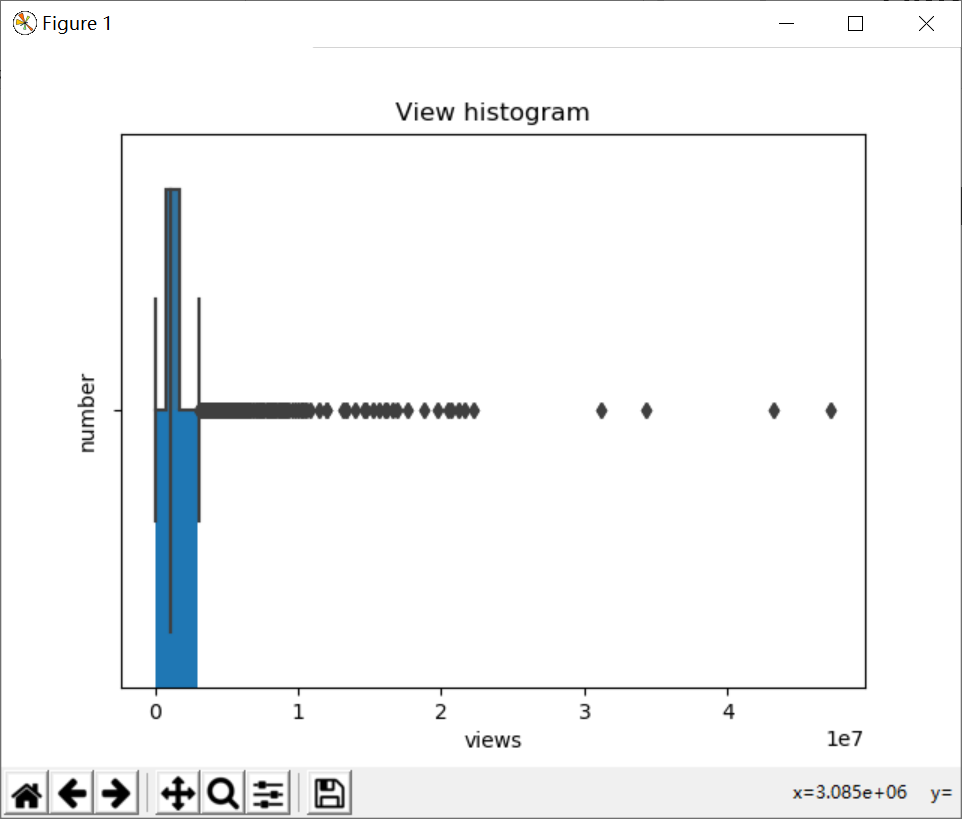
print(df['views'].describe())



The average number of views on the TED talks was 1.6 million. The median was 1.12 million. This shows that the average popularity of the TED talks is very high.

The boxplot of it

sns.boxplot(df['views'])



A large number of video playback is concentrated around 3008500

plt.hist(df.views, range=(0,3000000), bins=100, rwidth=1)

plt.xlabel(u"views")# plots an axis lable

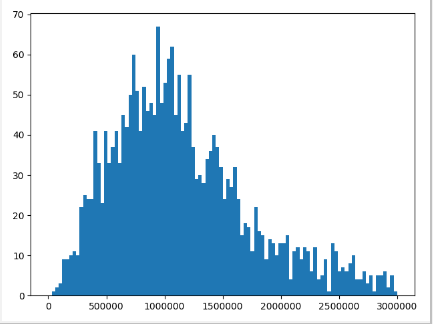
plt.ylabel(u"number")

plt.title(u"View histogram")

hist：

plt.hist(df.views, range=(0,3000000), bins=100, rwidth=1)

plt.show()



The data distribution is a bit like a normal distribution

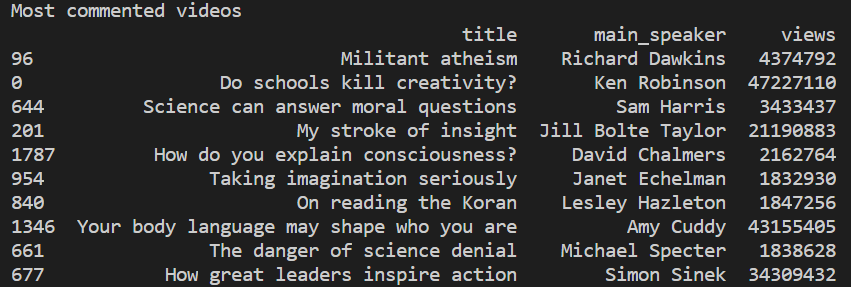
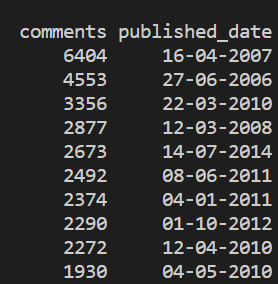
3.the analysis of comments

a = df.sort\_values("comments",inplace=False,ascending=False)

a = a[['title', 'main\_speaker', 'views', 'comments','published\_date']]

print("Most commented videos")

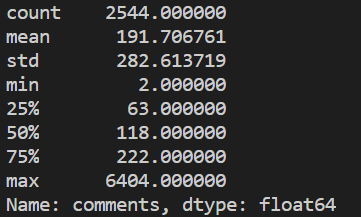
print(a.head(10))

The most commented is Militant atheism, not the most viewed Do schools kill creativity?

Analyze the numerical characteristics of the commentary:

print(df['comments'].describe())

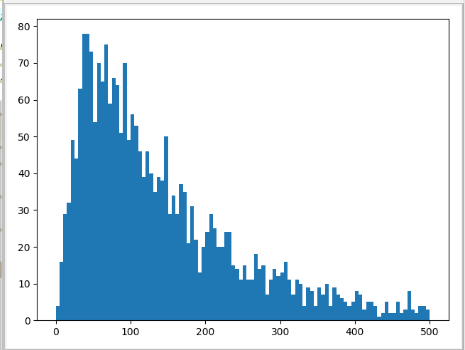


Visualize with boxplot

sns.boxplot(df['comments'])

hist：

plt.hist(df.views, range=(0,500), bins=100, rwidth=1)

  
Rising around 0-70, then falling steadily

4.Let's analyze the relationship between comments and page views.

a = sns.jointplot(x = 'views', y = 'comments', data = df)

plt.title("relationship between views and comments")

b = df[['views', 'comments']].corr()

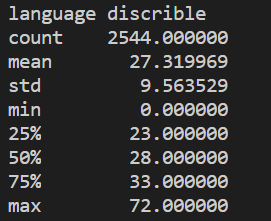
print(b)

5. Analysis of the number of languages

a = df['languages'].describe()

print("language discrible")

print(a)

  
A video can have up to 72 different languages, 75% in 33, and the median is 27.

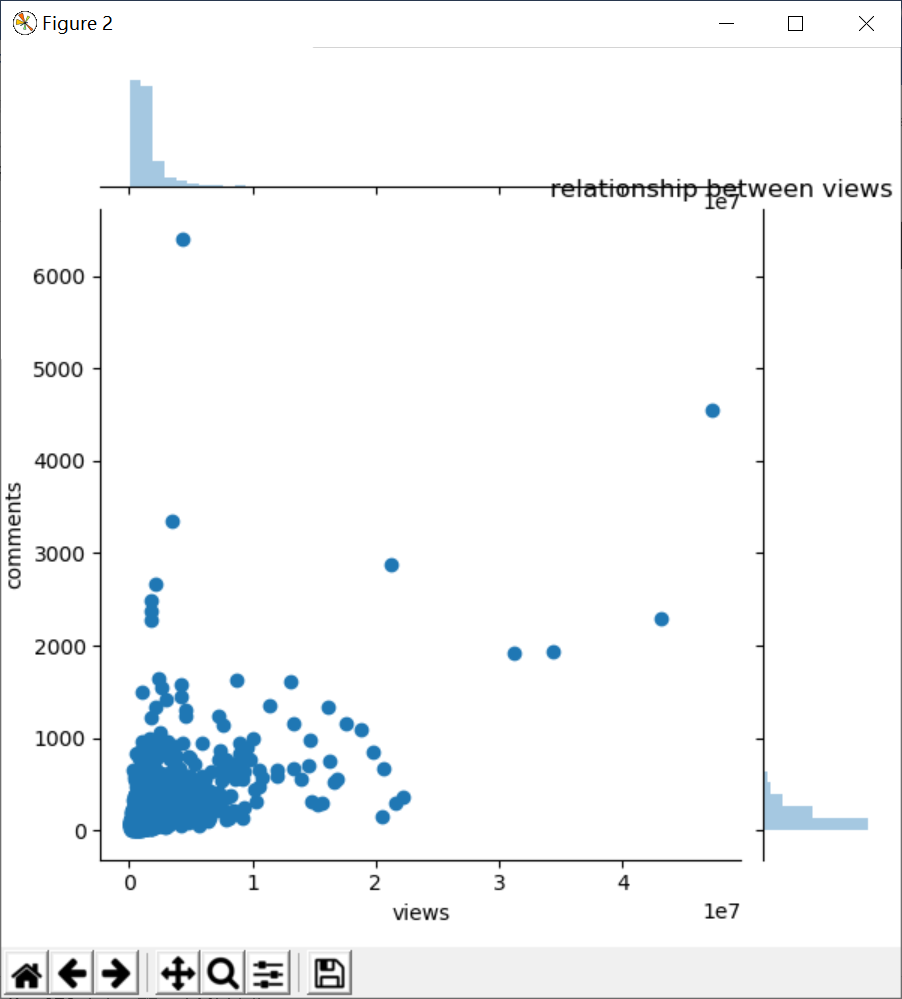
Analyze the relationship between page views and the number of languages

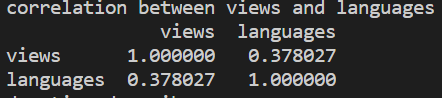
a = sns.jointplot(x = 'views', y = 'languages', data = df)

b = df[['views', 'languages']].corr()

print("correlation between views and languages")

print(b)





Weak connection

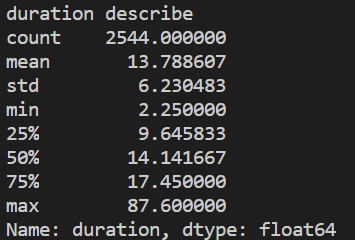
Analysis of duration data below

df['duration'] = df['duration']/60

a = df['duration'].describe()

print("duration describe")

print(a)



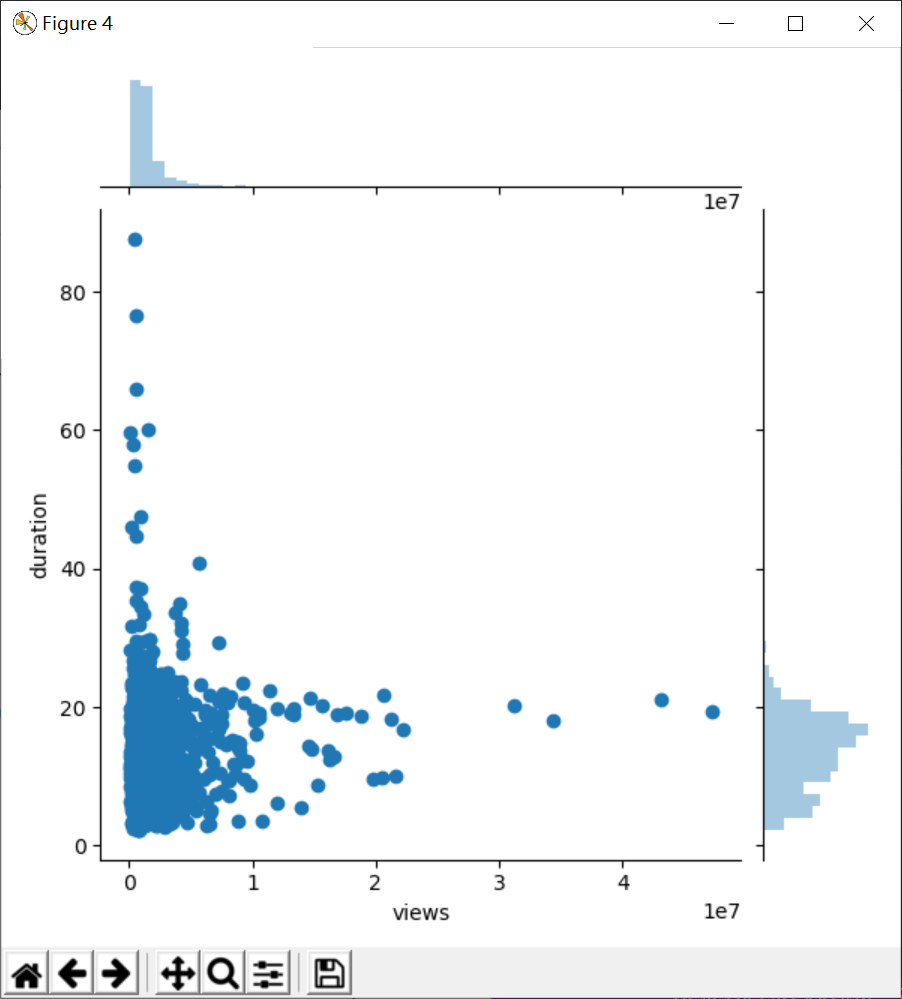
The duration is in minutes. The longest one is ted video 87.6 minutes, the shortest is only 2.25 minutes, and 75% is at 17.45, so most of them are mainly short and medium video.

a = sns.jointplot(x = 'views', y = 'duration', data = df)

b = df[['views', 'duration']].corr()

print("correlation between views and duration")

print(b)





Let's analyze which speakers are the most talked

speaker\_df = df.groupby('main\_speaker').count().reset\_index()[['main\_speaker', 'comments']]

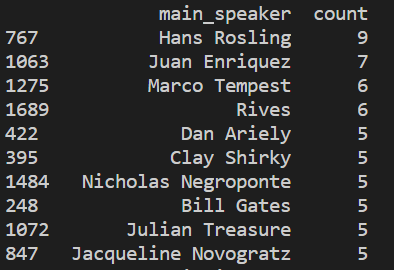
speaker\_df.columns = ['main\_speaker', 'count']

speaker\_df = speaker\_df.sort\_values('count', ascending=False)

showdata = speaker\_df.head(10)

print("speakers who publised most videos")

print(speaker\_df.head(10))

   
It can be seen that the most is Hans Rosling, the number of times is 9

Each ted video has its own source, let's analyze the occasion of its release.

events\_df = df[['title', 'event']].groupby('event').count().reset\_index()

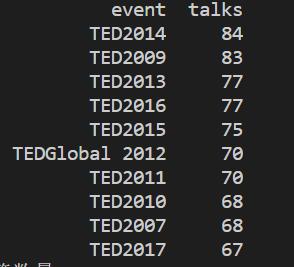
events\_df.columns = ['event', 'talks']

events\_df = events\_df.sort\_values('talks', ascending=False)

s = events\_df.head(10)

print("ted events describtion")

print(s)



The most is Ted2014, which released 84 speeches.

Finally, the analysis of the label Because each video corresponds to multiple tags, it needs to be saved with a list, then split and counted.

# transform string to list

df2['tags'] = df2['tags'].apply(lambda x: ast.literal\_eval(x))

# divide the tags

s = df2.apply(lambda x: pd.Series(x['tags']),axis=1).stack().reset\_index(level=1, drop=True)

s.name = 'theme'

# add the tags into the dataframe

theme\_df = df2.drop('tags', axis = 1).join(s)

print("the number of tag：{}".format(len(theme\_df['theme'].value\_counts())))

#most popular tags

Count the total number of tags and find the 10 most popular tags

pop\_themes = pd.DataFrame(theme\_df['theme'].value\_counts()).reset\_index()

pop\_themes.columns = ['theme', 'talks']

print("most popular themes")

print(pop\_themes.head(10))

Visualize with pie map

#pie diagram

labels = pop\_themes.head(10)['theme']

sizes = pop\_themes.head(10)['talks']

explode = (0.1, 0, 0, 0,0,0,0,0,0,0) # only "explode" the 2nd slice (i.e. 'Hogs')

fig1, ax1 = plt.subplots()

ax1.pie(sizes, explode=explode, labels=labels, autopct='%1.1f%%',

shadow=True, startangle=90)

ax1.axis('equal') # Equal aspect ratio ensures that pie is drawn as a circle.

plt.title("most popular themes")

plt.show()

