import pandas as pd

import numpy as np

from snownlp.normal import zh

import matplotlib.pyplot as plt

from pymongo import MongoClient

from mongoengine import connect

from pandas.io.json import json\_normalize

%matplotlib inline

import csv

data = pd.read\_csv("D:\Program Code\Python\Jupyter\课程设计\MyData.csv",delimiter=',', encoding='utf8')

data['time'] = data['time'].apply(str)

data = data.drop(columns='\_id')

data = data.drop\_duplicates(subset='userId')

data['time'] = pd.to\_datetime((data['time']/1000), unit='s')

data = data[data['time']>=pd.to\_datetime('2019-02-05 00:00:00')]

data.set\_index(data["time"], inplace=True)

data.head()

from pyecharts import Bar

from pyecharts import Line

from pyecharts import Overlap

score\_total = data['score'].value\_counts().sort\_index()

bar = Bar("《流浪地球》各评分数量", width=700)

line = Line("", width=700)

bar.add("", score\_total.index, score\_total.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='40%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

line.add("", score\_total.index, score\_total.values+1000, is\_smooth=True)

overlap = Overlap(width=700)

overlap.add(bar)

overlap.add(line)

Overlap

import jieba

from collections import Counter

from pyecharts import WordCloud

jieba.add\_word('屈楚萧')

jieba.add\_word('刘启')

jieba.add\_word('吴京')

jieba.add\_word('刘培强')

jieba.add\_word('李光洁')

jieba.add\_word('王磊')

jieba.add\_word('吴孟达')

jieba.add\_word('达叔')

jieba.add\_word('韩子昂')

jieba.add\_word('赵今麦')

jieba.add\_word('韩朵朵')

swords = [x.strip() for x in open ('stopwords.txt',encoding='utf-8')

def plot\_word\_cloud(data, swords):

text = ''.join(data['content'])

words = list(jieba.cut(text))

ex\_sw\_words = []

for word in words:

if len(word)>1 and (word not in swords):

ex\_sw\_words.append(word)

c = Counter()

c = Counter(ex\_sw\_words)

wc\_data = pd.DataFrame({'word':list(c.keys()), 'counts':list(c.values())}).sort\_values(by='counts', ascending=False).head(100)

wordcloud = WordCloud(width=1300, height=620)

wordcloud.add("", wc\_data['word'], wc\_data['counts'], word\_size\_range=[20, 100])

return wordcloud

gender\_total = data['gender'].value\_counts()

bar = Bar("《流浪地球》观众性别", width=700)

bar.add("", ['未知', '男', '女'], gender\_total.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='60%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

gender\_low = data.loc[data['score']<5, 'gender'].value\_counts()

bar = Bar("《流浪地球》低分评论观众性别", width=700)

bar.add("", ['未知', '男', '女'], gender\_low.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='60%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

gender\_low/gender\_low.sum()\*100

level\_total = data['userLevel'].value\_counts().sort\_index()

bar = Bar("《流浪地球》观众等级", width=700)

bar.add("", level\_total.index, level\_total.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='40%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

level\_low = data.loc[data['score']<5, 'userLevel'].value\_counts().sort\_index()

bar = Bar("《流浪地球》低分评论的观众等级", width=700)

bar.add("", level\_low.index, level\_low.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='40%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

mapping = {'liucixin':'刘慈欣|大刘', 'guofan':'郭帆', 'quchuxiao':'屈楚萧|刘启|户口', 'wujing':'吴京|刘培强',

'liguangjie':'李光洁|王磊', 'wumengda':'吴孟达|达叔|韩子昂', 'zhaojinmai':'赵今麦|韩朵朵'}

for key, value in mapping.items():

data[key] = data['content'].str.contains(value, na=False)

# 总体提及次数

#补充na=False

staff\_count = pd.Series({key: data.loc[data[key], 'score'].count() for key in mapping.keys()}).sort\_values()

bar = Bar("《流浪地球》演职员总体提及次数", width=700)

bar.add("", ['李光洁','郭帆','赵今麦','吴孟达','屈楚萧','刘慈欣','吴京'], staff\_count.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='60%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

bar

average\_score = pd

bar = Bar("《流浪地球》演职员平均分", width=700)

bar.add("", ['赵今麦','吴孟达','吴京','屈楚萧','李光洁','刘慈欣','郭帆'], np.round(average\_score.values,2), is\_stack=True, is\_label\_show=True,

bar\_category\_gap='60%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

staff\_count\_low = pd.Series({key: data.loc[data[key]&(data['score']<5), 'score'].count() for key in mapping.keys()}).sort\_values()

staff\_count\_low

staff\_count\_pct = np.round(staff\_count\_low/staff\_count\*100, 2).sort\_values()

staff\_count\_pct

bar = Bar("《流浪地球》演职员低分评论提及百分比", width=700)

bar.add("", ['郭帆','刘慈欣','李光洁','屈楚萧','赵今麦','吴京','吴孟达'], staff\_count\_pct.values, is\_stack=True, is\_label\_show=True,

bar\_category\_gap='60%', label\_color = ['#130f40'],

legend\_text\_size=18,xaxis\_label\_textsize=18,yaxis\_label\_textsize=18)

Bar

data[data['wumengda']&(data['score']<5)].nlargest(5, 'upCount')

for i in data[data['wumengda']&(data['score']<5)].nlargest(5, 'upCount')['content']:

print(i+'\n')

data[data['wujing']&(data['score']<5)].nlargest(5, 'upCount')

for i in data[data['wujing']&(data['score']<5)].nlargest(5, 'upCount')['content']:

print(i+'\n')

data[data['zhaojinmai']&(data['score']<5)].nlargest(5, 'upCount')

for i in data[data['zhaojinmai']&(data['score']<5)].nlargest(5, 'upCount')['content']:

print(i+'\n')