import pandas as pd  
import re  
import datetime  
from matplotlib import pyplot as plt  
from wordcloud import WordCloud  
from pyecharts.charts import Map  
from pyecharts import options as opts  
import numpy as np  
  
  
plt.rcParams['font.sans-serif'] = ['SimHei'] # 用来正常显示中文标签  
plt.rcParams['axes.unicode\_minus'] = False # 用来正常显示负号  
  
def create\_wordcloud(cut\_text,savename):  
  
 # 生成词云  
 plt.figure(figsize=(16, 8))  
  
 plt.axis('off')  
 cloud = WordCloud(  
 font\_path="qqq.ttf",  
 background\_color='white',  
 max\_words=800,  
 max\_font\_size=100,  
 width=1000,  
 height=500  
 )  
 word\_cloud = cloud.generate(cut\_text)  
 savename = savename + '词云.png'  
 word\_cloud.to\_file(savename) # 保存的图片  
  
  
def max\_plot\_draw\_bar(x,y,name):  
  
 plt.figure(figsize=(120, 40)) # 定义坐标图尺寸  
 plt.title(name, fontsize=50)  
  
 # 45为旋转的角度，不然x轴会特别密  
 plt.xticks(rotation=45)  
 # 绘制柱状图  
 z = [i for i in range(len(x))]  
 plt.xticks(z, x)  
 plt.tick\_params(labelsize=50) # 坐标轴单位大小  
 for ii in y:  
 plt.scatter(z,ii)  
 savename = './pic/' + name  
 plt.savefig(savename, # province⽂件名：png、jpg、pdf  
 dpi=100, # 保存圖⽚像素密度  
 facecolor='white', # 視圖與邊界之間顏⾊設置  
 edgecolor='lightgreen', # 視圖邊界顏⾊設置  
 bbox\_inches='tight') # 保存圖⽚完整  
 print('over')  
 plt.clf()  
  
  
def plot\_draw\_bar(x,y,name):  
  
 plt.figure(figsize=(120, 40)) # 定义坐标图尺寸  
 plt.title(name, fontsize=50)  
  
 # 45为旋转的角度，不然x轴会特别密  
 plt.xticks(rotation=45)  
 # 绘制柱状图  
 z = [i for i in range(len(x))]  
 plt.xticks(z, x)  
 plt.tick\_params(labelsize=50) # 坐标轴单位大小  
 plt.bar(z,y, width=0.7)  
 savename = './pic/' + name  
 plt.savefig(savename, # province⽂件名：png、jpg、pdf  
 dpi=100, # 保存圖⽚像素密度  
 facecolor='white', # 視圖與邊界之間顏⾊設置  
 edgecolor='lightgreen', # 視圖邊界顏⾊設置  
 bbox\_inches='tight') # 保存圖⽚完整  
 print('over')  
 plt.clf()  
  
  
def plot\_draw\_plot(x,y, name):  
 z = [i for i in range(len(x))]  
  
  
 # 45为旋转的角度，不然x轴会特别密  
 plt.xticks(rotation=45)  
  
 # 绘制柱状图  
 plt.figure(figsize=(90, 30)) # 定义坐标图尺寸  
 plt.xticks(z, x)  
 plt.tick\_params(labelsize=50) # 坐标轴单位大小  
 plt.title(name, fontsize=50)  
 plt.xlabel('time\_2', fontsize=20) # x坐標名称  
 plt.ylabel('J', fontsize=20) # y坐標名称  
 plt.tick\_params(labelsize=20) #坐标轴单位大小  
  
  
 plt.plot(z, y) # 线型坐标点  
 plt.scatter(z, y,s=100,c='r') # 线型坐标点  
  
 save\_name = './pic/' + name # ⽂件名  
 plt.savefig(save\_name, # ⽂件名：png、jpg、pdf  
 dpi=100, # 保存圖⽚像素密度  
 facecolor='white', # 視圖與邊界之間顏⾊設置  
 edgecolor='lightgreen', # 視圖邊界顏⾊設置  
 bbox\_inches='tight') # 保存圖⽚完整  
 print('over')  
 plt.clf()  
  
  
def plot\_draw\_pie(x,y, name):  
 z = [i for i in range(len(x))]  
  
  
 # 45为旋转的角度，不然x轴会特别密  
 plt.xticks(rotation=45)  
  
 # 绘制柱状图  
 plt.figure(figsize=(50, 50)) # 定义坐标图尺寸  
 plt.title(name, fontsize=50)  
 plt.tick\_params(labelsize=20) #坐标轴单位大小  
  
  
 plt.pie(y,labels = x) # 线型坐标点  
 plt.rcParams.update({'font.size': 20})  
 plt.legend(loc=(1, 0.8))  
  
 save\_name = './pic/' + name # ⽂件名  
  
 plt.savefig(save\_name)  
 print('over')  
 plt.clf()  
  
  
  
def plot\_draw\_zhifang\_bar(yy,xx,name,tt):  
  
 plt.figure(figsize=(150, 40)) # 定义坐标图尺寸  
  
 #colors = ["#8dd3c7", "#bebada"]  
 labels = tt  
 z = [i for i in range(len(yy))]  
 print(xx)  
 print('^^^^^^^^^')  
 plt.xticks(z, yy)  
 plt.tick\_params(labelsize=40) # 坐标轴单位大小  
 plt.hist(xx, bins=z,histtype="bar", rwidth=0.8,  
 stacked=True, label=labels)  
 plt.xlabel("国家", fontsize=50)  
 plt.ylabel("数量", fontsize=50)  
 plt.title(name, fontsize=50)  
 plt.rcParams.update({'font.size': 20})  
 plt.legend(labels=tt, loc='best')  
  
 plt.savefig('./pic/' +name)  
 plt.clf()  
  
  
  
  
  
  
def map\_visualmap(sequence, year):  
 name\_map = {  
 'Singapore Rep.': '新加坡',  
 'Dominican Rep.': '多米尼加',  
 'Palestine': '巴勒斯坦',  
 'Bahamas': '巴哈马',  
 'Timor-Leste': '东帝汶',  
 'Afghanistan': '阿富汗',  
 'Guinea-Bissau': '几内亚比绍',  
 "Côte d'Ivoire": '科特迪瓦',  
 'Siachen Glacier': '锡亚琴冰川',  
 "Br. Indian Ocean Ter.": '英属印度洋领土',  
 'Angola': '安哥拉',  
 'Albania': '阿尔巴尼亚',  
 'United Arab Emirates': '阿联酋',  
 'Argentina': '阿根廷',  
 'Armenia': '亚美尼亚',  
 'French Southern and Antarctic Lands': '法属南半球和南极领地',  
 'Australia': '澳大利亚',  
 'Austria': '奥地利',  
 'Azerbaijan': '阿塞拜疆',  
 'Burundi': '布隆迪',  
 'Belgium': '比利时',  
 'Benin': '贝宁',  
 'Burkina Faso': '布基纳法索',  
 'Bangladesh': '孟加拉国',  
 'Bulgaria': '保加利亚',  
 'The Bahamas': '巴哈马',  
 'Bosnia and Herz.': '波斯尼亚和黑塞哥维那',  
 'Belarus': '白俄罗斯',  
 'Belize': '伯利兹',  
 'Bermuda': '百慕大',  
 'Bolivia': '玻利维亚',  
 'Brazil': '巴西',  
 'Brunei': '文莱',  
 'Bhutan': '不丹',  
 'Botswana': '博茨瓦纳',  
 'Central African Rep.': '中非',  
 'Canada': '加拿大',  
 'Switzerland': '瑞士',  
 'Chile': '智利',  
 'China': '中国',  
 'Ivory Coast': '象牙海岸',  
 'Cameroon': '喀麦隆',  
 'Dem. Rep. Congo': '刚果民主共和国',  
 'Congo': '刚果',  
 'Colombia': '哥伦比亚',  
 'Costa Rica': '哥斯达黎加',  
 'Cuba': '古巴',  
 'N. Cyprus': '北塞浦路斯',  
 'Cyprus': '塞浦路斯',  
 'Czech Rep.': '捷克',  
 'Germany': '德国',  
 'Djibouti': '吉布提',  
 'Denmark': '丹麦',  
 'Algeria': '阿尔及利亚',  
 'Ecuador': '厄瓜多尔',  
 'Egypt': '埃及',  
 'Eritrea': '厄立特里亚',  
 'Spain': '西班牙',  
 'Estonia': '爱沙尼亚',  
 'Ethiopia': '埃塞俄比亚',  
 'Finland': '芬兰',  
 'Fiji': '斐',  
 'Falkland Islands': '福克兰群岛',  
 'France': '法国',  
 'Gabon': '加蓬',  
 'United Kingdom': '英国',  
 'Georgia': '格鲁吉亚',  
 'Ghana': '加纳',  
 'Guinea': '几内亚',  
 'Gambia': '冈比亚',  
 'Guinea Bissau': '几内亚比绍',  
 'Eq. Guinea': '赤道几内亚',  
 'Greece': '希腊',  
 'Greenland': '格陵兰',  
 'Guatemala': '危地马拉',  
 'French Guiana': '法属圭亚那',  
 'Guyana': '圭亚那',  
 'Honduras': '洪都拉斯',  
 'Croatia': '克罗地亚',  
 'Haiti': '海地',  
 'Hungary': '匈牙利',  
 'Indonesia': '印度尼西亚',  
 'India': '印度',  
 'Ireland': '爱尔兰',  
 'Iran': '伊朗',  
 'Iraq': '伊拉克',  
 'Iceland': '冰岛',  
 'Israel': '以色列',  
 'Italy': '意大利',  
 'Jamaica': '牙买加',  
 'Jordan': '约旦',  
 'Japan': '日本',  
 'Kazakhstan': '哈萨克斯坦',  
 'Kenya': '肯尼亚',  
 'Kyrgyzstan': '吉尔吉斯斯坦',  
 'Cambodia': '柬埔寨',  
 'Korea': '韩国',  
 'Kosovo': '科索沃',  
 'Kuwait': '科威特',  
 'Lao PDR': '老挝',  
 'Lebanon': '黎巴嫩',  
 'Liberia': '利比里亚',  
 'Libya': '利比亚',  
 'Sri Lanka': '斯里兰卡',  
 'Lesotho': '莱索托',  
 'Lithuania': '立陶宛',  
 'Luxembourg': '卢森堡',  
 'Latvia': '拉脱维亚',  
 'Morocco': '摩洛哥',  
 'Moldova': '摩尔多瓦',  
 'Madagascar': '马达加斯加',  
 'Mexico': '墨西哥',  
 'Macedonia': '马其顿',  
 'Mali': '马里',  
 'Myanmar': '缅甸',  
 'Montenegro': '黑山',  
 'Mongolia': '蒙古',  
 'Mozambique': '莫桑比克',  
 'Mauritania': '毛里塔尼亚',  
 'Malawi': '马拉维',  
 'Malaysia': '马来西亚',  
 'Namibia': '纳米比亚',  
 'New Caledonia': '新喀里多尼亚',  
 'Niger': '尼日尔',  
 'Nigeria': '尼日利亚',  
 'Nicaragua': '尼加拉瓜',  
 'Netherlands': '荷兰',  
 'Norway': '挪威',  
 'Nepal': '尼泊尔',  
 'New Zealand': '新西兰',  
 'Oman': '阿曼',  
 'Pakistan': '巴基斯坦',  
 'Panama': '巴拿马',  
 'Peru': '秘鲁',  
 'Philippines': '菲律宾',  
 'Papua New Guinea': '巴布亚新几内亚',  
 'Poland': '波兰',  
 'Puerto Rico': '波多黎各',  
 'Dem. Rep. Korea': '朝鲜',  
 'Portugal': '葡萄牙',  
 'Paraguay': '巴拉圭',  
 'Qatar': '卡塔尔',  
 'Romania': '罗马尼亚',  
 'Russia': '俄罗斯',  
 'Rwanda': '卢旺达',  
 'W. Sahara': '西撒哈拉',  
 'Saudi Arabia': '沙特阿拉伯',  
 'Sudan': '苏丹',  
 'S. Sudan': '南苏丹',  
 'Senegal': '塞内加尔',  
 'Solomon Is.': '所罗门群岛',  
 'Sierra Leone': '塞拉利昂',  
 'El Salvador': '萨尔瓦多',  
 'Somaliland': '索马里兰',  
 'Somalia': '索马里',  
 'Serbia': '塞尔维亚',  
 'Suriname': '苏里南',  
 'Slovakia': '斯洛伐克',  
 'Slovenia': '斯洛文尼亚',  
 'Sweden': '瑞典',  
 'Swaziland': '斯威士兰',  
 'Syria': '叙利亚',  
 'Chad': '乍得',  
 'Togo': '多哥',  
 'Thailand': '泰国',  
 'Tajikistan': '塔吉克斯坦',  
 'Turkmenistan': '土库曼斯坦',  
 'East Timor': '东帝汶',  
 'Trinidad and Tobago': '特里尼达和多巴哥',  
 'Tunisia': '突尼斯',  
 'Turkey': '土耳其',  
 'Tanzania': '坦桑尼亚',  
 'Uganda': '乌干达',  
 'Ukraine': '乌克兰',  
 'Uruguay': '乌拉圭',  
 'United States': '美国',  
 'Uzbekistan': '乌兹别克斯坦',  
 'Venezuela': '委内瑞拉',  
 'Vietnam': '越南',  
 'Vanuatu': '瓦努阿图',  
 'West Bank': '西岸',  
 'Yemen': '也门',  
 'South Africa': '南非',  
 'Zambia': '赞比亚',  
 'Zimbabwe': '津巴布韦',  
 'Comoros': '科摩罗'  
 }  
 # c = (  
 # Map(opts.InitOpts(width='1500px',height='800px')) # opts.InitOpts() 设置初始参数:width=画布宽,height=画布高  
 # .add(series\_name=year, data\_pair=sequence, maptype="world" ) # 系列名称(显示在中间的名称 )、数据 、地图类型  
 # .set\_global\_opts(  
 # title\_opts=opts.TitleOpts(title="Map-世界地图"),  
 # visualmap\_opts=opts.VisualMapOpts(max\_=37, min\_=52),  
 # )  
 # )  
 # c.render(path='./test.html')  
  
 c = (  
 Map(opts.InitOpts(width='1500px',height='800px'))  
 .add(year, [list([i,j]) for i,j in sequence.items()], "world", name\_map=name\_map)  
 .set\_series\_opts(label\_opts=opts.LabelOpts(is\_show=False))  
 .set\_global\_opts(  
 title\_opts=opts.TitleOpts(title="Map-世界地图"),  
 visualmap\_opts=opts.VisualMapOpts(max\_=200),  
 )  
 .render("map\_world.html")  
 )  
  
  
  
if \_\_name\_\_=='\_\_main\_\_':  
 df = pd.read\_excel("1.xlsx", sheet\_name=0)  
 title = df['标题']  
  
 dir = df['导演: ']  
 scip = df['编剧:']  
 actor = df['主演: ']  
  
 type = df['类型']  
 country = df['制片国家/地区:']  
 year = df[' 上映日期']  
 long = df['片长: ']  
 type = df['类型']  
  
 type\_l = []  
 for i in type:  
 if isinstance(i, str):  
 x = i.strip().split(' / ')  
 type\_l = type\_l + x  
  
 t = list(set(type\_l))  
 i = [0 for i in range(len(t))]  
 zero\_type = dict(zip(t, i))  
 print(zero\_type)  
 #year = [re.sub('/.','',i) for i in year]  
 year\_l = []  
 for j,i in enumerate(year):  
 if isinstance(i, datetime.date):  
 year\_l.append(i.year)  
 else:  
 year\_l.append(i)  
 year\_l.sort()  
 dic\_year = {}  
 for i in year\_l:  
 if i in dic\_year:  
 dic\_year[i] += 1  
 else:  
 dic\_year[i] = 1  
  
 xx = dic\_year.keys()  
 y = dic\_year.values()  
 name = '年份分布'  
 plot\_draw\_plot(xx,y, name)#年份分布图  
 type\_l  
 dic\_type = {}  
 for i in type\_l:  
 if i in dic\_type:  
 dic\_type[i] += 1  
 else:  
 dic\_type[i] = 1  
 x = dic\_type.keys()  
 y = dic\_type.values()  
 name = '类型分布饼状图'  
 plot\_draw\_pie(x,y,name)  
  
 # dic\_type\_year = {}  
 # for j in xx:  
 # dic\_type\_year[j] = zero\_type.copy()  
 #  
 # for i,j in enumerate(year\_l):  
 # x = type[i].strip().split(' / ')  
 # for z in x:  
 # dic\_type\_year[j][z] +=1  
 # ty\_year\_lis = []  
 # for ii in xx:  
 # temp = []  
 # for i in t:  
 # temp.append(dic\_type\_year[ii][i])  
 # print(temp)  
 # ty\_year\_lis.append(temp)  
 # name = '类型点状分布'  
 # print(xx)  
 # xx = list(xx)  
 #  
 # plot\_draw\_zhifang\_bar\_2(xx, ty\_year\_lis, name,t)  
  
  
 long\_l = []  
 for i in long:  
 if isinstance(i, str):  
 x = int(i.replace('分钟', ''))  
 long\_l.append(x)  
 else:  
 long\_l.append(int(x))  
 long\_l.sort()  
 long = []  
 for i in long\_l:  
 long.append(i//10\*10)  
 print(long)  
 # df['导演:']  
 dic\_long = {}  
 for i in long:  
 if i in dic\_long:  
 dic\_long[i] += 1  
 else:  
 dic\_long[i] = 1  
  
 x = dic\_long.keys()  
 y = dic\_long.values()  
 name = '时间段分布'  
 plot\_draw\_bar(x,y, name)#时间时段分布图  
  
 dir = df['导演: ']  
 scip = df['编剧:']  
 actor = df['主演: ']  
  
 all = dir + scip + actor  
 dic\_long = {}  
 name\_l = []  
 for i in all:  
 if isinstance(i, str):  
 x = i.strip().split(' / ')  
 name\_l = name\_l + x  
 # s = ' '.join(name\_l)  
 # savename = '人物'  
 # create\_wordcloud(s, savename)  
 # print(s)  
  
 dic\_country = {}  
 country\_l = []  
 for i in country:  
 if isinstance(i, str):  
 x = i.strip().split(' / ')  
 country\_l = country\_l + x  
  
 for i in country\_l:  
 if i in dic\_country:  
 dic\_country[i] += 1  
 else:  
 dic\_country[i] = 1  
 x = dic\_country.keys()  
 y = dic\_country.values()  
 name = '国家'  
 # dic\_country['中国'] = dic\_country['中国大陆']  
 # del dic\_country['中国大陆']  
  
  
 #map\_visualmap(dic\_country, name)  
  
  
  
  
  
 dic\_dic = {}  
 for i in country\_l:  
 dic\_dic[i] = zero\_type.copy()  
 print(']]]]',zero\_type)  
 for j,i in enumerate(country):  
 i = i.strip().split(' / ')  
 for z in i:  
 t\_j = type[j].strip().split(' / ')  
 for ii in t\_j:  
 if ii in dic\_dic[z]:  
 dic\_dic[z][ii] +=1  
  
 print('\*\*\*\*\*\*',dic\_dic)  
 ty\_co\_lis = []  
 country\_l = list(dic\_country.keys())  
  
 print('+++++',country\_l,'++++++')  
 print('------', t, '++++++')  
 print(dic\_dic['中国香港']['历史'])  
 for ty in t:  
 temp = []  
  
 for jjj in range(len(country\_l)):  
 x = country\_l[jjj]  
 z = dic\_dic[x][ty]  
 temp.append(z)  
 print(temp)  
 ty\_co\_lis.append(temp)  
  
 name = '直方'  
 plot\_draw\_zhifang\_bar(country\_l,ty\_co\_lis, name,t)