

程设第二次作业

20377383 樊思涵

前言：看了同学们的展示作业，意识到自己的作业有很多不足，这次作业决定学习展示作业中函数注释的写法，改变代码风格。

任务 1.

准备活动：

调用的库：

```
1 import numpy as np
2 import jieba
3 import re
4 from matplotlib import pyplot as plt
5
```

将需要用到的文件地址作为全局变量

```
5 #文件名用作全局变量
6 filename_0=r'C:\Users\LF\Desktop\weibo.txt\weibo.txt'
7 filename_0_test=r'C:\Users\LF\Desktop\weibo.txt\test_weibo.txt'
8 filename_anger=r'C:\Users\LF\Desktop\emotion_lexicon\anger.txt'
9 filename_disgust=r'C:\Users\LF\Desktop\emotion_lexicon\disgust.txt'
10 filename_fear=r'C:\Users\LF\Desktop\emotion_lexicon\fear.txt'
11 filename_joy=r'C:\Users\LF\Desktop\emotion_lexicon\joy.txt'
12 filename_sadness=r'C:\Users\LF\Desktop\emotion_lexicon\sadness.txt'
13 filename_emotions=[filename_anger,filename_disgust,filename_fear,filename_joy,filename_sadness]
14
```

补充：

```
15 EMO=['anger','disgust','fear','joy','sadness']
16 TIMEMODE=['week','hour','month']
```

其中，test_weibo.txt 为 473 条小样本量测试用微博数据
按行读入文件

```
15 def fread_document(filename) -> list:
16     """
17     按行读入txt文件并返回列表
18     ->返回一维列表
19     :filename:原始文档目录
20     """
21     print("-----正在导入数据-----")
22     f=open(filename,encoding='UTF-8')
23     line = f.readline().strip() #读取第一行,不用读入列表
24     txt=[]
25     while line: # 直到读取完文件
26         line = f.readline().strip() # 读取一行文件，包括换行符
27         txt.append(line)
28     f.close() # 关闭文件
29     if txt[-1]=='':
30         txt.pop()
31     print("-----数据导入完成-----")
32     return txt
```

处理重复微博

```

34  def fdelete_repetition_txt(txt) -> list:
35      """
36      删除列表中的重复元素
37      ->返回与原始列表相同结构的列表
38      :txt:需删除重复元素的列表
39      """
40      if len(txt) != len(set(txt)):
41          print("有重复微博")
42          txt = [x for x in set(txt)]
43      return txt

```

展示结果

```

PS E:\code\py_code> python -u "e:\code\py_code\week3\week3.py"
-----正在导入数据-----
-----数据导入完成-----
有重复微博
处理前数据有2422485项.
处理重复数据后有272883项.
-----正在添加情绪词-----

```

没想到这么多重复数据！

对数据进行分割降噪处理

Step1.对\t 分割，得到 column = 4 的列表

```

46  def fcut_txt(txt,n) -> list:
47      """
48      将按行读入的数据按'\t'切割
49      ->返回column=4的二维列表
50      :txt:按行读入的数据列表
51      :n:txt的长度
52      """
53      data=[]
54      for i in range(n):
55          data.append(txt[i].split("\t"))
56      return data

```

Step2.删除每条微博后都有的“我在这里：xxxxx”

Step3.使用正则表达式删除一些无意义的文本

过滤效果如下:

分析：在删除无意义文本时，不能删除所有的表情，因为在情绪分词中记录了表情，不同表情显然能够表达不同的情绪。

任务 2.

在分词表中添加情绪词

```

88 def faddword(filename_emotions):
89     """
90     jieba添加自定义字典
91     """
92     print("-----正在添加情绪词-----")
93     for i in filename_emotions:
94         jieba.load_userdict(i)
95

```

创建闭包函数

取出现次数最多的 emotion 为对应微博的情绪

```

96 def fdic_emotion(filename_emotions,data,sentence,n):
97     """
98     闭包函数返回情绪，时间，地址列表函数
99     ->函数名
100     :filename_emotions:情绪地址列表
101     :data:column=4的全信息列表
102     :sentence:处理后的文本字符
103     :n:列表长度
104     """
105     emodict = []
106     for i in filename_emotions:
107         file = open(i,'r',encoding='utf-8')
108         emodict.append([line.strip() for line in file.readlines()])
109         file.close()
110     def splitword(): #分词获取情绪以及对应的时间地点
111         nonlocal emodict
112         emotion_list,time_list,address_list = [],[],[]
113         for i in range(n):
114             emotion_dict = {'anger':0,'disgust':0,'fear':0,'joy':0,'sadness':0}
115             splitword = jieba.lcut(sentence[i])
116             #print(splitword)

```



```

117 ~         for word in splitword:
118 ~             if word in emodict[0]:
119 ~                 emotion_dict['anger']+=1
120 ~             elif word in emodict[1]:
121 ~                 emotion_dict['disgust']+=1
122 ~             elif word in emodict[2]:
123 ~                 emotion_dict['fear']+=1
124 ~             elif word in emodict[3]:
125 ~                 emotion_dict['joy'] +=1
126 ~             elif word in emodict[4]:
127 ~                 emotion_dict['sadness']+=1
128 ~         if max(emotion_dict.values())==0:
129 ~             emotion = 'none'
130 ~         else:
131 ~             emotion = max(emotion_dict,key=emotion_dict.get)
132 ~             emotion_list.append(emotion)
133 ~             time_list.append(data[i][3])
134 ~             address_list.append(data[i][0])
135 ~     print(emotion_list,end = '')
136 ~     return emotion_list,time_list,address_list
137 ~ return splitword
138

```

结果展示（用测试数据文本）

Emotion_list

[illegible]

time_list

[illegible]

address_list

[3-67844, 16.109024] [3-67847, 16.152640] [3-67850, 16.19622] [3-67853, 16.23930] [3-67856, 16.28248] [3-67859, 16.32566] [3-67862, 16.36884] [3-67865, 16.41192] [3-67868, 16.45510] [3-67871, 16.49828] [3-67874, 16.54146] [3-67877, 16.58464] [3-67880, 16.62782] [3-67883, 16.67099] [3-67886, 16.71417] [3-67889, 16.75735] [3-67892, 16.79953] [3-67895, 16.84271] [3-67898, 16.88589] [3-67901, 16.92907] [3-67904, 16.97225] [3-67907, 17.01543] [3-67910, 17.05861] [3-67913, 17.10179] [3-67916, 17.14497] [3-67919, 17.18815] [3-67922, 17.23133] [3-67925, 17.27451] [3-67928, 17.31769] [3-67931, 17.36087] [3-67934, 17.40405] [3-67937, 17.44723] [3-67940, 17.49041] [3-67943, 17.53359] [3-67946, 17.57677] [3-67949, 17.61995] [3-67952, 17.66313] [3-67955, 17.70631] [3-67958, 17.74949] [3-67961, 17.79267] [3-67964, 17.83585] [3-67967, 17.87903] [3-67970, 17.92221] [3-67973, 17.96539] [3-67976, 18.00857] [3-67979, 18.05175] [3-67982, 18.09493] [3-67985, 18.13811] [3-67988, 18.18129] [3-67991, 18.22447] [3-67994, 18.26765] [3-67997, 18.31083] [3-68000, 18.35401] [3-68003, 18.39719] [3-68006, 18.44037] [3-68009, 18.48355] [3-68012, 18.52673] [3-68015, 18.56991] [3-68018, 18.61309] [3-68021, 18.65627] [3-68024, 18.69945] [3-68027, 18.74263] [3-68030, 18.78581] [3-68033, 18.82899] [3-68036, 18.87217] [3-68039, 18.91535] [3-68042, 18.95853] [3-68045, 19.00171] [3-68048, 19.04489] [3-68051, 19.08807] [3-68054, 19.13125] [3-68057, 19.17443] [3-68060, 19.21761] [3-68063, 19.26079] [3-68066, 19.30397] [3-68069, 19.34715] [3-68072, 19.39033] [3-68075, 19.43351] [3-68078, 19.47669] [3-68081, 19.51987] [3-68084, 19.56305] [3-68087, 19.60623] [3-68090, 19.64941] [3-68093, 19.69259] [3-68096, 19.73577] [3-68099, 19.77895] [3-68102, 19.82213] [3-68105, 19.86531] [3-68108, 19.90849] [3-68111, 19.95167] [3-68114, 19.99485] [3-68117, 20.03803] [3-68120, 20.08121] [3-68123, 20.12439] [3-68126, 20.16757] [3-68129, 20.21075] [3-68132, 20.25393] [3-68135, 20.29711] [3-68138, 20.34029] [3-68141, 20.38347] [3-68144, 20.42665] [3-68147, 20.46983] [3-68150, 20.51301] [3-68153, 20.55619] [3-68156, 20.59937] [3-68159, 20.64255] [3-68162, 20.68573] [3-68165, 20.72891] [3-68168, 20.77209] [3-68171, 20.81527] [3-68174, 20.85845] [3-68177, 20.90163] [3-68180, 20.94481] [3-68183, 20.98799] [3-68186, 21.03117] [3-68189, 21.07435] [3-68192, 21.11753] [3-68195, 21.16071] [3-68198, 21.20389] [3-68201, 21.24707] [3-68204, 21.29025] [3-68207, 21.33343] [3-68210, 21.37661] [3-68213, 21.41979] [3-68216, 21.46297] [3-68219, 21.50615] [3-68222, 21.54933] [3-68225, 21.59251] [3-68228, 21.63569] [3-68231, 21.67887] [3-68234, 21.72205] [3-68237, 21.76523] [3-68240, 21.80841] [3-68243, 21.85159] [3-68246, 21.89477] [3-68249, 21.93795] [3-68252, 21.98113] [3-68255, 22.02431] [3-68258, 22.06749] [3-68261, 22.11067] [3-68264, 22.15385] [3-68267, 22.19703] [3-68270, 22.24021] [3-68273, 22.28339] [3-68276, 22.32657] [3-68279, 22.36975] [3-68282, 22.41293] [3-68285, 22.45611] [3-68288, 22.49929] [3-68291, 22.54247] [3-68294, 22.58565] [3-68297, 22.62883] [3-68300, 22.67201] [3-68303, 22.71519] [3-68306, 22.75837] [3-68309, 22.80155] [3-68312, 22.84473] [3-68315, 22.88791] [3-68318, 22.93109] [3-68321, 22.97427] [3-68324, 23.01745] [3-68327, 23.06063] [3-68330, 23.10381] [3-68333, 23.14699] [3-68336, 23.19017] [3-68339, 23.23335] [3-68342, 23.27653] [3-68345, 23.31971] [3-68348, 23.36289] [3-68351, 23.40607] [3-68354, 23.44925] [3-68357, 23.49243] [3-68360, 23.53561] [3-68363, 23.57879] [3-68366, 23.62197] [3-68369, 23.66515] [3-68372, 23.70833] [3-68375, 23.75151] [3-68378, 23.79469] [3-68381, 23.83787] [3-68384, 23.88105] [3-68387, 23.92423] [3-68390, 23.96741] [3-68393, 24.01059] [3-68396, 24.05377] [3-68399, 24.09695] [3-68402, 24.14013] [3-68405, 24.18331] [3-68408, 24.22649] [3-68411, 24.26967] [3-68414, 24.31285] [3-68417, 24.35603] [3-68420, 24.39921] [3-68423, 24.44239] [3-68426, 24.48557] [3-68429, 24.52875] [3-68432, 24.57193] [3-68435, 24.61511] [3-68438, 24.65829] [3-68441, 24.70147] [3-68444, 24.74465] [3-68447, 24.78783] [3-68450, 24.83101] [3-68453, 24.87419] [3-68456, 24.91737] [3-68459, 24.96055] [3-68462, 25.00373] [3-68465, 25.04691] [3-68468, 25.09009] [3-68471, 25.13327] [3-68474, 25.17645] [3-68477, 25.21963] [3-68480, 25.26281] [3-68483, 25.30599] [3-68486, 25.34917] [3-68489

任务 3.

创建小时、星期、月份字典，统计每条的属性

```

154 def fplotime(emotion,time_mode,emotion_list,time_list):
155     """
156     生成指定情绪指定模式的情绪统计图
157     :emotion:目标的情绪的字符串
158     :time_mode:目标的时间图表模式
159     :emotion_list:微博情绪列表
160     :time_list:微博时间列表
161     """
162     week = ['Mon','Tue','Wed','Thu','Fri','Sat','Sun']
163     week_dict = {}
164     week_dict = week_dict.fromkeys(week,0)
165
166     month = ['Jan','Feb','Mar','Apr','May','Jun','Jul','Aug','Sep','Oct','Nov','Dec']
167     month_dict = {}
168     month_dict = month_dict.fromkeys(month,0)
169
170     hour = ['{:0>2d}'.format(i) for i in range(24)]
171     hour_dict = {}
172     hour_dict = hour_dict.fromkeys(hour,0)

```

对三种不同模式进行计数（仅仅贴出 week 代码）

```

174     if time_mode == 'week':
175         for tm in time_list:
176             if emotion_list[time_list.index(tm)] == emotion:
177                 week_dict[tm[0]] += 1
178         week_value = []
179         for value in week_dict.values():
180             week_value.append(value)
181         plt.plot(week,week_value,'o-',color='r',label='week_{}'.format(emotion))
182         plt.xlabel("week")#横坐标名字
183         plt.ylabel("times")#纵坐标名字
184         plt.legend(loc = "best")#图例
185         for a,b in zip(week,week_value):
186             plt.text(a,b+1,ha = 'center',va = 'bottom',fontsize=10)
187         #print(week_dict)

```

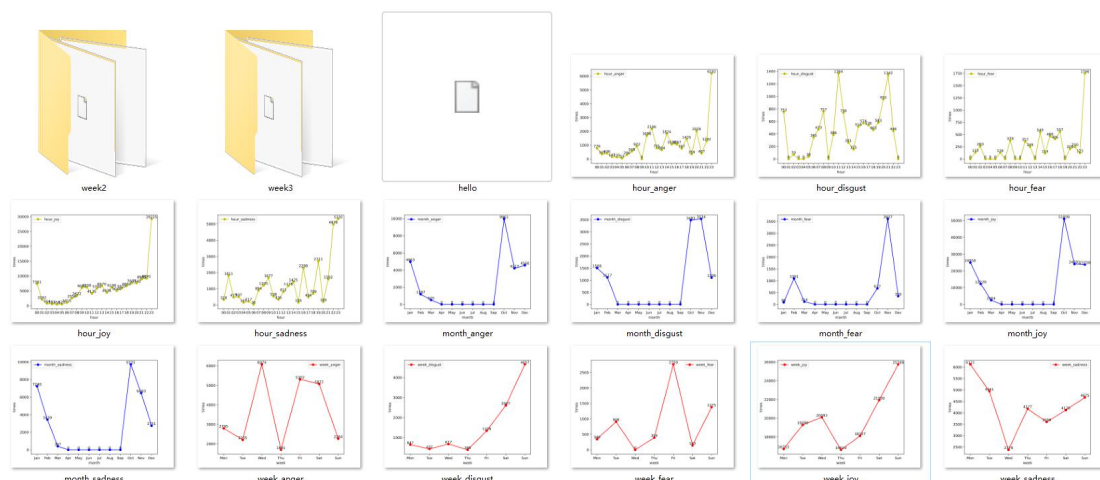
输出并保存图像

```

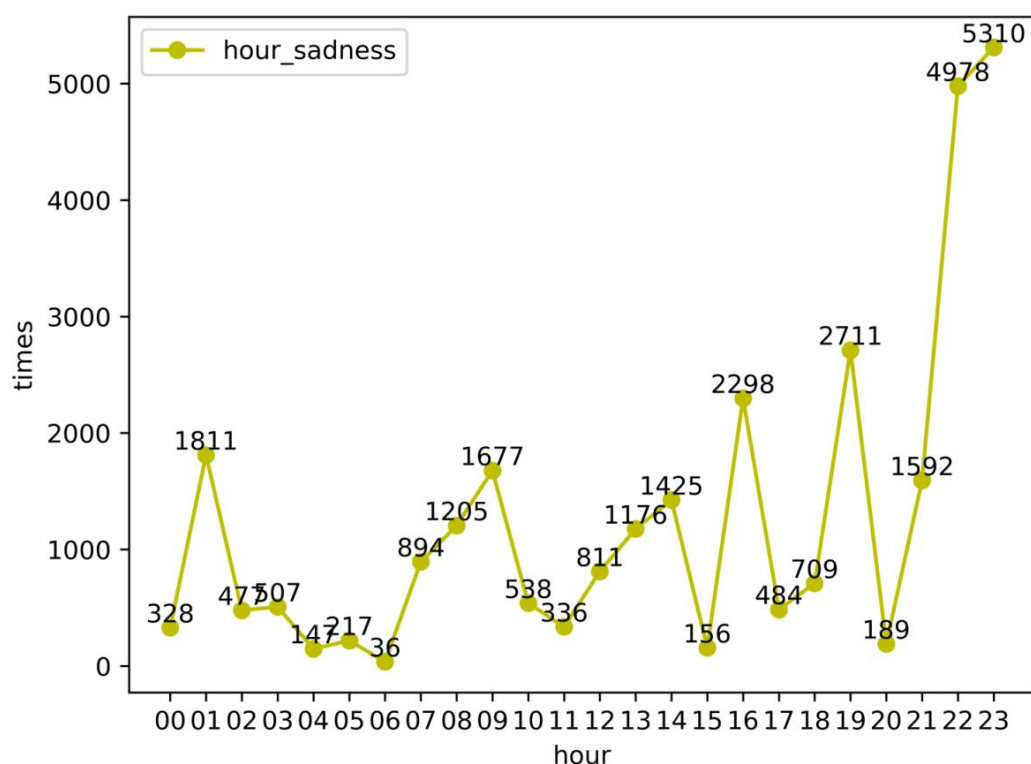
218     plt.savefig('{}_{}.png'.format(time_mode,emotion),dpi=800)
219     #plt.show()

```

一共输出 15 张图

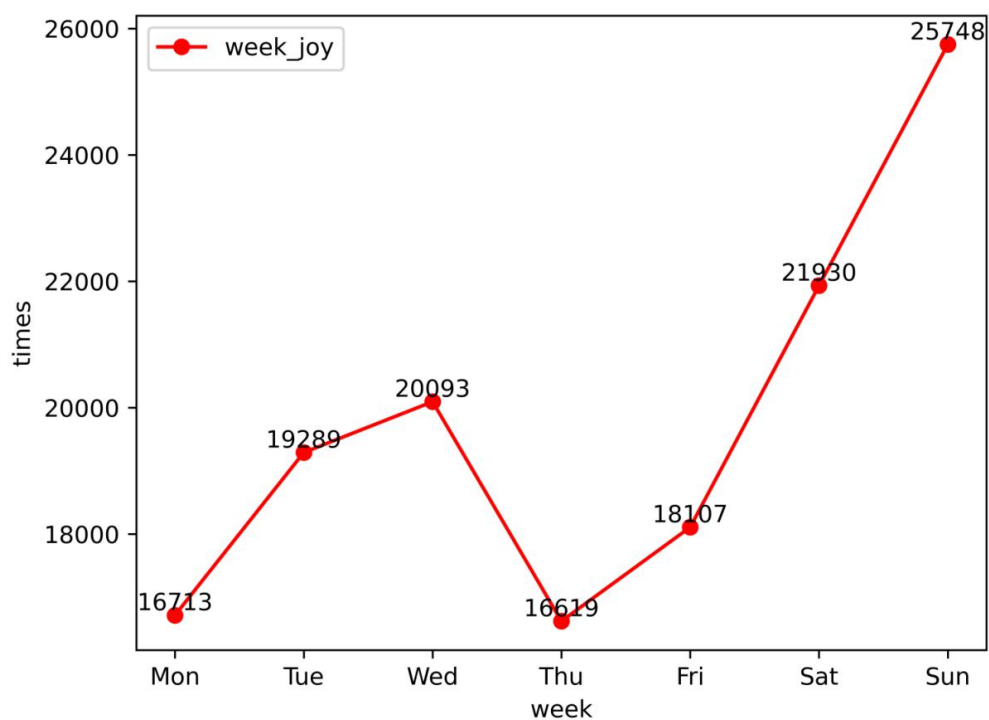


举例分析 1:



观察 24 小时内的悲伤情绪图，发现 22.23 点的悲伤微博数明显多于其他时间的微博数，考虑到经常有人在晚上“深夜emo”，该图像很好地证明了这一点。凌晨由于发微博人数问题导致数据较低也属于正常现象。

举例分析 2:



观察周一到周五的心情,周六和周日明显高于工作日,而周一是工作日的第一天,相应的数据也近似最低,

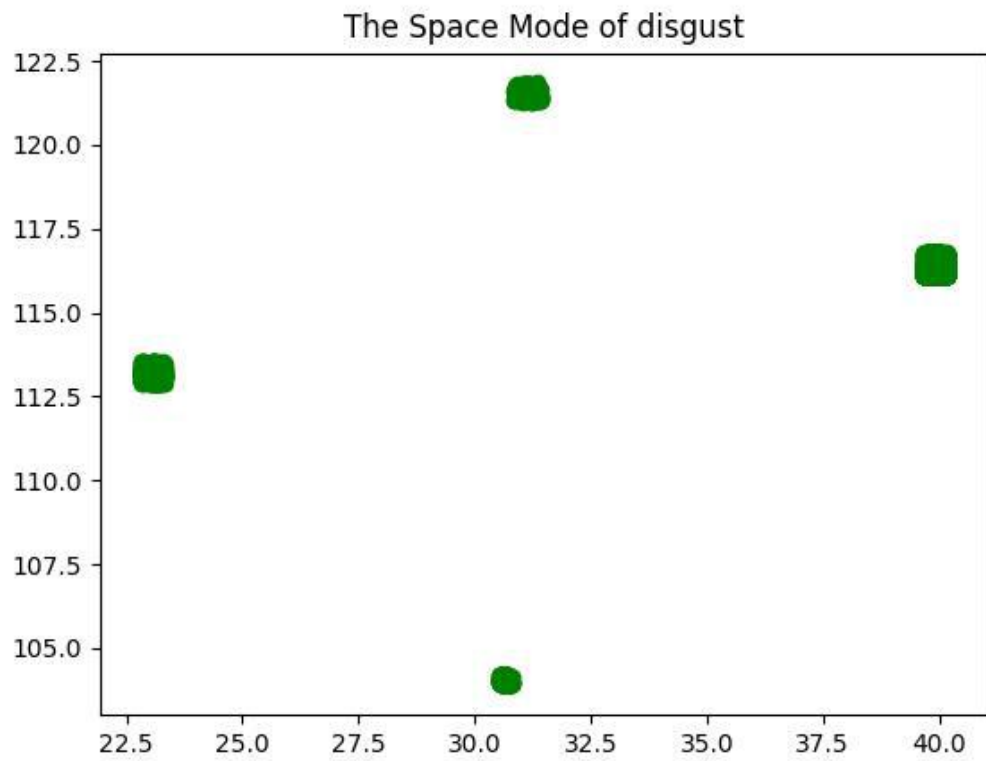
任务 6.

提取每条微博的地点坐标,并画出散点图

```
221 v def emotion_admode(address_list,emotion_list,n):
222 v     """
223     画出位置分布图
224     :address_list:微博地址列表
225     :emotion_list:微博情绪列表
226     :n:列表长度
227     """
228     x_list=[]
229     y_list=[]
230     add_emotion = []
231     color_list = ['red','green','purple','yellow','blue']
232 v     for i in range(n):
233         zuobiao = address_list[i].split(',')
234         x_list.append(float(zuobiao[0][1:]))
235         y_list.append(float(zuobiao[1][:-1]))
236         add_emotion.append(emotion_list[i])
237 v     for i in range(5):
238         x=[]
239         y=[]
240 v         for j in range(n):
241 v             if EMO[i] == add_emotion[j]:
242                 x.append(x_list[j])
243                 y.append(y_list[j])
244         plt.scatter(x,y, c = color_list[i])
245         plt.title('The Space Mode of ' + EMO[i])
246         plt.show()
```

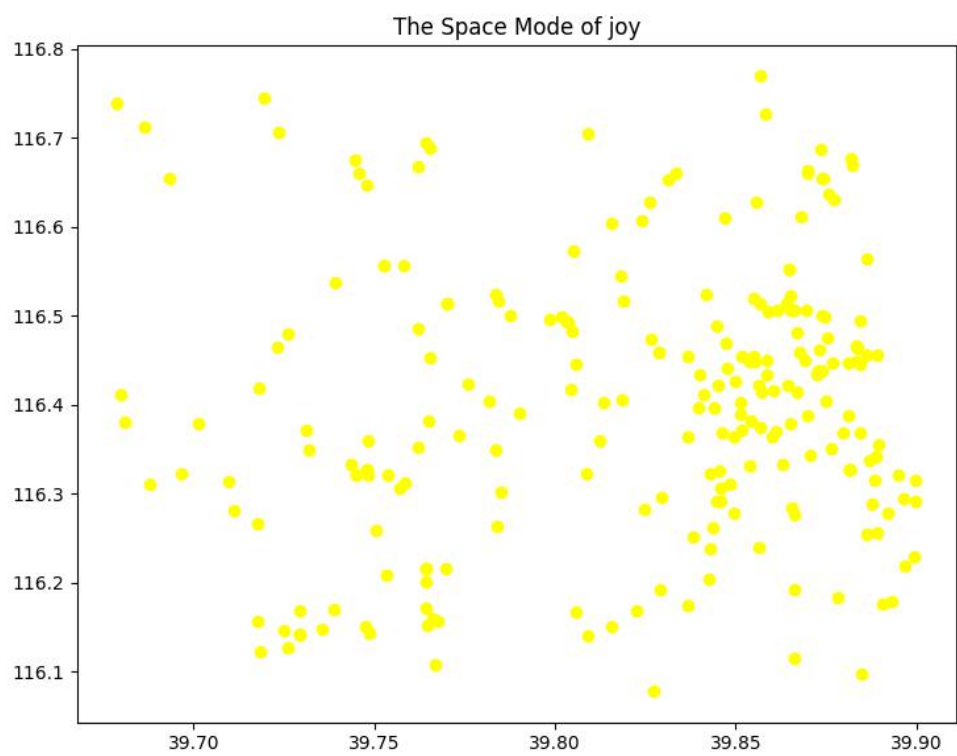
读入全部的数据

Figure 1



发现明显分成了四类，可以用肉眼聚类分析得知分成四个部分，需要将不同地区的数据分别分析才能得到有价值的信息。

通过筛选特定部分的微博获得同类的数据，再次进行可视化分析：



观察发现情绪为 joy 的微博有一定的聚集效应，如图中的右侧明显更为集中，这一现象可能与情绪的传染性有关，值得进一步研究。

Main 函数如下：

```
248 def main():
249     """
250     main函数
251     """
252     txt=fread_document(filename_0_test) #读取txt文件
253     m=len(txt)
254     txt=fdelete_repetition_txt(txt) #删除重复项
255     n=len(txt) #记录项数
256     print("处理前数据有%d项.\n处理重复数据后有%d项." %(m,n))
257     data=fcut_txt(txt,n) #简单切割内容、地址、时间
258     n=len(data)#更新项数
259     print(n)
260     sentence = fdelete_url_data(data,n) #简单分词处理
261     sentence = fclean_sentence(sentence,n) #正则表达式降噪
262     #faddword(filename_emotions) #添加情绪词
263     f=fdic_emotion(filename_emotions,data,sentence,n)
264     emotion_list,time_list,address_list = f()
265     time_list_new=ftime_list(time_list,n)
266     ...
267     for emotion in EMO:
268         for timemode in TIMEMODE:
269             fplotime(emotion,timemode,emotion_list,time_list_new)
270     ...
271     emotion_admode(address_list,emotion_list,n)
272
273 if __name__ == '__main__':
274     main()
```

Ref:

逐行读取 txt 文件

<https://cloud.tencent.com/developer/article/2080154>

学习正则表达式

https://blog.csdn.net/weixin_42550871/article/details/108943529

<https://blog.csdn.net/bang152101/article/details/89284249>

使用正则表达式切除 url 和域名

<https://www.lmlphp.com/user/151450/article/item/8191109/>