注：因设备原因，数据集过大难以计算，故截取部分数据集。

```
def main():
    D = pd.read_table(r'weibo.txt', encoding = 'utf-8')
    D = sampling(D, 50)  # 系统抽样1/50
    # 1.数据清洗
    D['text'] = D['text'].apply(cut)  # 删去结尾无效字符
    D.drop_duplicates(subset = 'text', keep='first', inplace=True)   # 删除重复项
    tokens = cut_words(D['text'])  # 分词

    # 2.计算情绪向量
    emos = tokens.apply(calculate(emo_names))

    # 3.时间分析
    time_analysis(D, emos, period = 'day')

    # 4.空间分析
    space_analysis(D, emos)
```

1. 实现一个函数，对微博数据进行清洗，去除噪声（如 url 等），过滤停用词。注意分词的时候应该将情绪词典加入 Jieba 或 pyltp 的自定义词典，以提高这些情绪词的识别能力。

```
def sampling(D, n):  # 系统抽样1/n
    df = D.iloc[np.array(D.index) % n == 0]
    df.reset_index(inplace=True, drop=True)  # 重置索引
    return df

def cut(str):  # 用正则表达式删去无意义字符
    c = re.findall(".+?(?=http|我在:|我在这里:)", str, re.S)
    if c == []:
        return str
    else:
        return c[0]

def cut_words(words):  # 分词
    with open(r'stopwords_list.txt', encoding='utf-8') as f:  # 停用词表stop
        stop = f.read().split('\n')
    for dict_file_name in emo_names:  # 将情感词典加入jieba分词
        jieba.load_userdict('emotion_lexicon/' + dict_file_name + '.txt')
    return words.apply(lambda x : [i for i in jieba.lcut(x) if i not in stop])
```

```
    D = pd.read_table(r'weibo.txt', encoding = 'utf-8')
    D = sampling(D, 50)  # 系统抽样1/50
    # 1.数据清洗
    D['text'] = D['text'].apply(cut)  # 删去结尾无效字符
    D.drop_duplicates(subset = 'text', keep='first', inplace=True)   # 删除重复项
    tokens = cut_words(D['text'])  # 分词
```

2. 实现两个函数，实现一条微博的情绪分析，返其情绪向量或情绪值。目前有两种方法，一是认为一条微博的情绪是混合的，即一共有 n 个情绪词，如果 joy 有 n1 个，则 joy 的比例是 n1/n；二是认为一条微博的情绪是唯一的，即 n 个情绪词里，anger 的情绪词最多，则该微博的情绪应该为 angry。注意，这里要求用闭包实现，尤其是要利用闭包实现一次加载情绪词典且局部变量持久化的特点。同时，也要注意考虑一些特别的情况，如无情绪词出现，不同情绪的情绪词出现数目一样等，并予以处理（如定义为无情绪，便于在后面的分析中去除）。

```python
def calculate(emo_names):
    dics = {}  # 构建字典存储情绪词典以避免闭包与exec()引起的错误
    for dict_file_name in emo_names:  # 加载情绪词典为计算情绪向量做准备
        with open('emotion_lexicon/' + dict_file_name + '.txt', mode = 'r', encoding = 'utf-8') as f:
            exec(f"dics['{dict_file_name}']=f.read().split('\\n')")  # 利用exec函数一次定义5个变量
    def token_to_vector(sentence):  # 计算情感分词为情绪向量
        nonlocal dics
        dic = {'anger': 0, 'disgust': 0, 'fear': 0, 'joy': 0, 'sadness': 0}
        for i in sentence:  # 对于句子中的每个情感词
            for emo in emo_names:
                dic[f'{emo}'] += eval(f"i in dics['{emo}']")  # 若情感词在字典中，则对应情绪值+1
        emo_vector = np.array(list(dic.values()))
        sum_emo = sum(emo_vector)  # 无情绪词时各得分均为0
        if sum_emo != 0:
            emo_vector = emo_vector / sum_emo  # 句子的五大情绪得分
        return emo_vector
    return token_to_vector
```

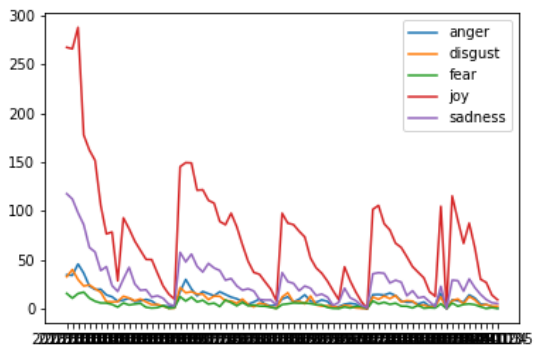
针对['anger', 'disgust', 'fear', 'joy', 'sadness']五类情绪，笔者使用了格式化字符串 f"……"配合 exec()和 eval()函数来执行代码的方式，更简洁地完成了任务，但这样破坏了程序的安全性，易导致程序不稳定，故不推荐常用。

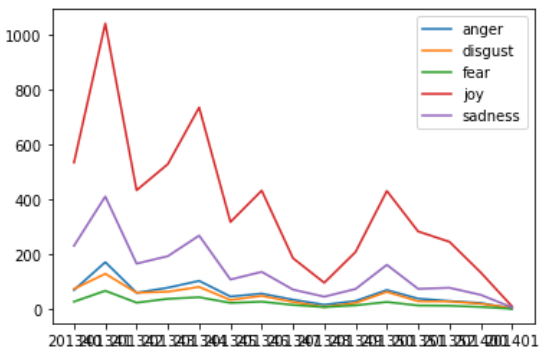
参见深度辨析 Python 的 eval() 与 exec() - 知乎 (zhihu.com)；

关于 python：为什么使用"eval"是一个坏习惯？ | 码农家园 (codenong.com)

3. 微博中包含时间，可以讨论不同时间情绪比例的变化趋势，实现一个函数，可以通过参数来控制并返回对应情绪的时间模式，如 joy 的小时模式，sadness 的周模式等。

```python
def time_analysis(D, emos, period = 'day'):
    D.rename(columns = {'weibo_created_at': 'time'}, inplace = True)
    # 将原始数据的时间转换为struct_time
    D['time'] = D['time'].apply(lambda x: time.strptime(x, "%a %b %d %H:%M:%S %z %Y"))
    '''
    关于time库，参考https://docs.python.org/zh-cn/3/library/time.html#time.gmtime
    '''
    emo_time = pd.DataFrame(list(emos), columns = emo_names)
    # 计算同一时间段的情绪值
    if period == 'hour':
        emo_time['time'] = D['time'].apply(lambda x: time.strftime("%Y%m%d%H", x))
    elif period == 'day':
        emo_time['time'] = D['time'].apply(lambda x: time.strftime("%Y%m%d", x))
    elif period == 'week':
        emo_time['time'] = D['time'].apply(lambda x: time.strftime("%Y%U", x))
    elif period == 'month':
        emo_time['time'] = D['time'].apply(lambda x: time.strftime("%Y%m", x))
    # 分组求和
    emo_period = emo_time.groupby('time')
    emo_period = emo_period.sum()
    # 作折线图
    plt.plot(emo_period)
    plt.legend(emo_names)
```

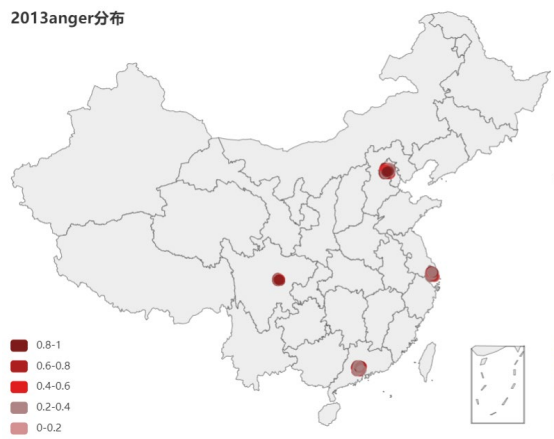


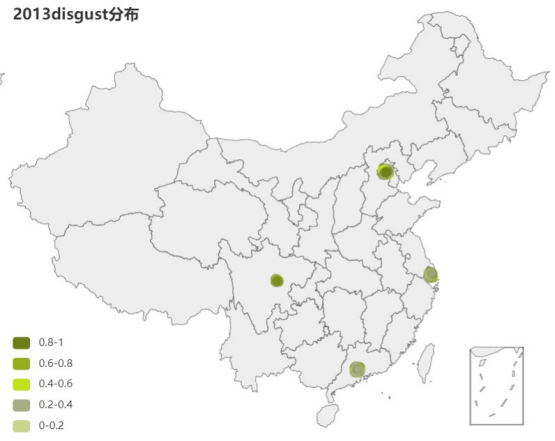


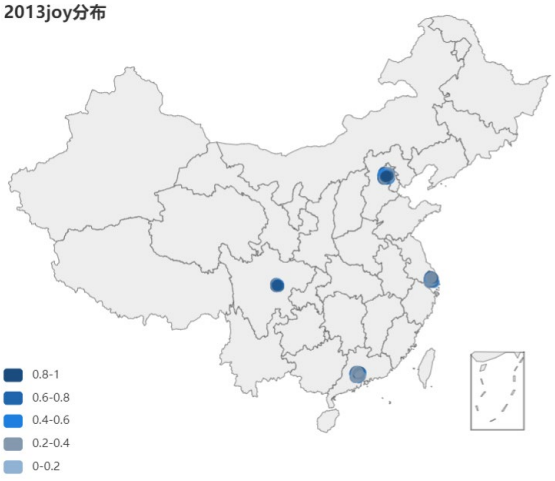
可以看到，总体上五大情绪中 joy 情绪最高，sadness 其次，且 joy 波动最大。在时间趋势上，各情绪呈现周期式涨跌，随时间推移整体呈向下趋势。每日的情绪波动呈规律性涨跌，可能是周六日上网人数较多，工作日上网人数较少所致。可能是 2013 年第 41 周发生了某个重大事件引起人们关注，随时间推移人们对此的关注逐渐下降。

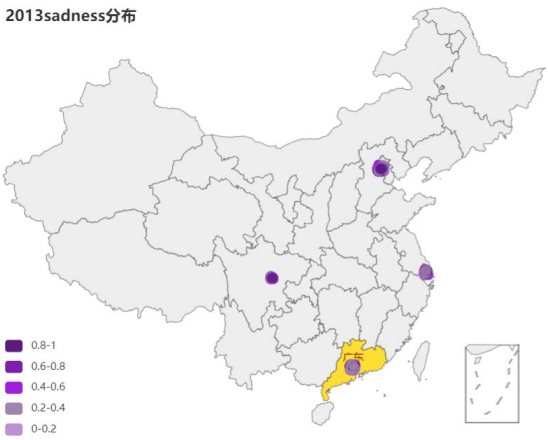
4. 微博中包含空间，可以讨论情绪的空间分布，实现一个函数，可以通过参数来控制并返回对应情绪的空间分布，即围绕某个中心点，随着半径增加该情绪所占比例的变化，中心点可默认值可以是城市的中心位置。

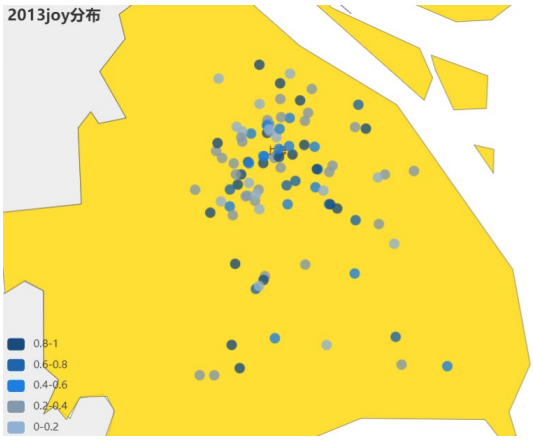
```
def space_analysis(D, emos):
    for k in range(5):
        g = Geo().add_schema(maptype = "china")  # 加载图表模型中的中国地图
        color_h = k*70  # 颜色H值
        emo = emo_names[k]
        for i in D['location'].index:
            if (emos[i][0] != 0):  # 排除值为0的点
                location = eval(D['location'][i])  # 经纬度坐标
                g.add_coordinate(str(i), location[1], location[0])  # 添加点
                g.add("", [(str(i), emos[i][0])])  # 显示点
        pieces = [  # 设置HSL渐变颜色
                {'min': 0, 'max': 0.2, 'label': '0-0.2', 'colorHue': color_h, 'colorLightness':0.7},
                {'min': 0.2, 'max': 0.4, 'label': '0.2-0.4', 'colorHue': color_h, 'colorLightness':0.6},
                {'min': 0.4, 'max': 0.6, 'label': '0.4-0.6', 'colorHue': color_h, 'colorLightness':0.5},
                {'min': 0.6, 'max': 0.8, 'label': '0.6-0.8', 'colorHue': color_h, 'colorLightness':0.4},
                {'min': 0.8, 'label': '0.8-1', 'colorHue': color_h, 'colorLightness':0.3}
            ]
        g.set_series_opts(label_opts = opts.LabelOpts(is_show=False))
        g.set_global_opts(title_opts = opts.TitleOpts(title=f"2013{emo}分布"),
                          visualmap_opts = opts.VisualMapOpts(is_piecewise=True, pieces=pieces))
        exec(f"g.render('geo_{emo}.html')")  # 输出地图
```











可以看到，微博数据主要来自北京、上海、广东、四川 4 个地区，总体上五大情绪北京和四川情绪值较高，上海和广东情绪值较低。北京的情绪点最多最密集，广东的情绪点最少且较稀疏。这个结果的原因可能是爬取的微博数据主要来自于北京用户，或微博的话题主要是北京用户关注。

对于管理者来说，可以重点关注北京人群的情绪变化特征，进行情绪调控、疏导和预防。

5. （附加）讨论字典方法进行情绪理解的优缺点，有无可能进一步扩充字典来提高情绪识别的准确率？如何扩充，有无自动或半自动的扩充思路？

（1）优点：计算相较于机器学习、神经网络等方法简单，实现比较容易。

缺点：情感分析完全依赖于字典和规则，较难处理表达方式不规范、与上下文相关联、同一词语情绪受语境影响的文本。而且字典需要经常更新，以收录网络新词。

（2）可以通过扩充新词和网络热词来提高情绪识别的准确率。

（3）扩充新词的方法有基于知识库、语料库、知识库和语料库相结合的方法[1]。

参考[1]王科,夏睿.情感词典自动构建方法综述[J].自动化学报,2016,42(04):495-511.DOI:10.16383/j.aas.2016.c150585.

6. （附加）可否对情绪的时间和空间分布进行可视化？（如通过 matplotlib 绘制曲线，或者用 pyecharts（注意版本的兼容性）在地图上标注不同的情绪）

见 4、5 问

7. （附加）思考情绪时空模式的管理意义，如营销等。

通过社交媒体了解某一事件（如疫情爆发、俄乌战争等）对公众情绪的影响，分析情绪变化的时空特征，能了解并预测民众的心理需求，对培育健康社会心态、提高应急响应、支持决策具有重要意义。