

# 28-matplotlib(热力图)

数据采集—气温数据

填充热力图的绘制

基于seaborn绘制热力图

如何绘制填充表格热力图，先给大家看一下效果图。



从效果图里我们可以发现，所谓的填充表格热力图就是**将原本为数字表（数组）的单元格以颜色来填充，颜色的深浅表示数值的大小**。我想，对于这样的图来说，总比直接看密密麻麻的数值表要轻松的多吧，毕竟颜色感官比数字感官要直接，要具有更强的冲击。除了填充表格热力图，还有更为常见的地图热力图等。那填充表格热力图是如何应用Python来实现的呢？就让我们手把手的进行讲解吧~

## 数据采集—气温数据

在绘图之前，需要说明一下绘图的数据源，案例中的数据是**通过爬虫获取的，用的是上海9月份每天的最高气温**，即生成两列数据（日期和最高气温）。在有了原始数据的基础上，还需要对**数据进行清洗和**

**整理**，关于这部分是做任何数据分析或可视化都必经的坎。详细可以通过下面的代码来了解：

- **步骤一：数据采集**

```
# ===== Python3 + Jupyter ===== #
# 导入所需的第三方包
import datetime
import calendar
import requests
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# 采集数据
# 上海2017年9月份历史气温数据
url = 'http://lishi.tianqi.com/shanghai/201709.html'


# 发送爬虫请求
response = requests.get(url).text
# 解析源代码
soup = BeautifulSoup(response, 'html.parser')
# 根据HTML标记语言，查询目标标记下的数据
datas = soup.findAll('div', {'class': 'tqtongji2'})[0].findAll('ul')[1:]

# 抓取日期数据
date = [i.findAll('li')[0].text for i in datas]
# 抓取最高温数据
high = [i.findAll('li')[1].text for i in datas]

# 创建数据框
df = pd.DataFrame({'date':date, 'high':high})
# 变量类型
df.dtypes
```

```
# 变量类型
df.dtypes
```

```
date    object
high    object
dtype: object
```

 每天进步一点点2015

- **步骤二：数据整理**

```

# 将date变量转换为日期类型
df.date = pd.to_datetime(df.date)
# 将high变量转换成数值型
df.high = df.high.astype('int')

# 数据处理
# 由日期型数据衍生出weekday
df['weekday'] = df.date.apply(pd.datetime.weekday)

# 由日期型数据计算week_of_month，即当前日期在本月中是第几周
# 由于没有现成的函数，这里自定义一个函数来计算week_of_month
def week_of_month(tgtdate):
    # 由日期型参数tgtdate计算该月的天数
    days_this_month = calendar.mdays[tgtdate.month]    # 通过循环当月的所有天数，
    # 找出第二周的第一个日期
    for i in range(1, days_this_month + 1):
        d = datetime.datetime(tgtdate.year, tgtdate.month, i)
        if d.day - d.weekday() > 0:
            startdate = d
            break
    # 返回日期所属月份的第一周
    return (tgtdate - startdate).days // 7 + 1

df['week_of_month'] = df.date.apply(week_of_month)
df.head()

```

	date	high	weekday	week_of_month
0	2017-09-01	29	4	0
1	2017-09-02	30	5	0
2	2017-09-03	29	6	0
3	2017-09-04	32	0	1
4	2017-09-05	34	1	1

每天进步一点点2015

到此为止，我们就**完成了数据的采集和清洗过程**，接下来我们就可以借助该数据完成填充热力（日历）图的绘制。

## 填充热力图的绘制

## 基于matplotlib绘制热力图

其实，我**需要绘制的是一个数据表**，只不过把表中的每一个单元格用颜色填充起来。而**表的结构是：列代表周一到周日，行代表9月份第一周到第五周**。很显然，我们刚刚完成的数据并不符合这样的结构，故需要通过pandas模块中的pivot\_table函数制作一个透视表，然后才可以绘图。关于热力图，我们可以使用matplotlib模块中的pcolor函数，具体我们可以看下方的绘图语句：

```
# =====绘图前的数据整理=====
# 构建数据表（日历）
target = pd.pivot_table(data = df.iloc[:, 1:], values = 'high',
                        index = 'week_of_month', columns = 'weekday')

target

# 缺失值填充（不填充的话pcolor函数无法绘制）
target.fillna(0, inplace=True)
# 删除表格的索引名称
target.index.name = None
# 对索引排序（为了让“第一周”到“第五周”的刻度从y轴的高到底显示）
target.sort_index(ascending=False, inplace=True)

# =====开始绘图=====
# 设置中文和负号正常显示
plt.rcParams['font.sans-serif'] = 'Microsoft YaHei'
plt.rcParams['axes.unicode_minus'] = False

plt.pcolor(target, # 指定绘图数据
          cmap=plt.cm.Blues, # 指定填充色
          edgecolors = 'white' # 指点单元格之间的边框色
        )

# 添加x轴和y轴刻度标签（加0.5是为了让刻度标签居中显示）
plt.xticks(np.arange(7)+0.5, ['周一', '周二', '周三', '周四', '周五', '周六', '周日'])
plt.yticks(np.arange(5)+0.5, ['第五周', '第四周', '第三周', '第二周', '第一周'])

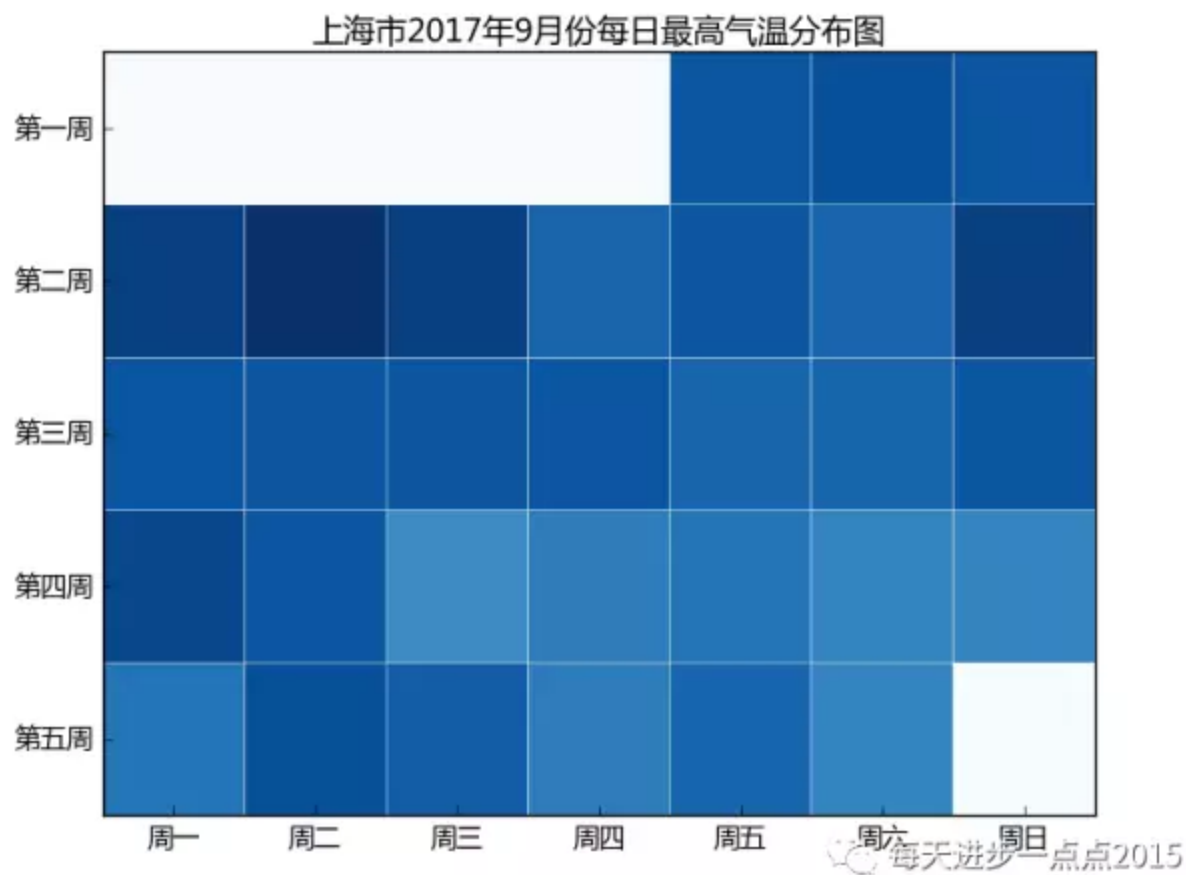
# 消除图框顶部和右部的刻度线
plt.tick_params(top='off', right = 'off')
# 添加标题
plt.title('上海市2017年9月份每日最高气温分布图')
# 显示图形
plt.show()
```

绘图数据的表结构

target

weekday	0	1	2	3	4	5	6
week_of_month							
0	NaN	NaN	NaN	NaN	29.0	30.0	29.0
1	32.0	34.0	32.0	27.0	29.0	27.0	32.0
2	29.0	29.0	29.0	29.0	27.0	27.0	29.0
3	31.0	29.0	22.0	24.0	25.0	23.0	23.0
4	25.0	30.0	28.0	24.0	27.0	23.0	NaN

热力图展现



OK，一张填充表格热力图就奇迹般的显示了，而且看上去还蛮舒服的。从图框看，9月份的第一天是周五，之后的每一天都有对应的颜色显示。但我在绘图过程中发现几个问题：

- 1) 绘图用的数据，不能包含缺失值，否则填充图是绘制不出来的，所有需要对缺失值做填充处理；

2) 最终的**图例无法实现**，即颜色的深浅，代表了具体的数值范围是什么？

3) 不方便将具体的温度值显示在每个单元格内；

为解决上面的三个问题，我们借助于seaborn模块中的heatmap函数重新绘制一下热力图，而且这些问题在heatmap函数看来根本不算问题。

## 基于seaborn绘制热力图

```
# 通过透视图函数形成绘图数据
target = pd.pivot_table(data = df.iloc[:, 1:], values = 'high',
                        index = 'week_of_month', columns = 'weekday')

# 绘图
ax = sns.heatmap(target, # 指定绘图数据
                 cmap=plt.cm.Blues, # 指定填充色
                 linewidths=.1, # 设置每个单元方块的间隔
                 annot=True # 显示数值
                )

# 添加x轴刻度标签(加0.5是为了让刻度标签居中显示)
plt.xticks(np.arange(7)+0.5, ['周一', '周二', '周三', '周四', '周五', '周六', '周日'])
# 可以将刻度标签置于顶部显示
# ax.xaxis.tick_top()

# 添加y轴刻度标签
plt.yticks(np.arange(5)+0.5, ['第一周', '第二周', '第三周', '第四周', '第五周'])
# 旋转y刻度0度，即水平显示
plt.yticks(rotation = 0)

# 设置标题和坐标轴标签
ax.set_title('上海市2017年9月份每日最高气温分布图')
ax.set_xlabel('')
ax.set_ylabel('')

# 显示图形
plt.show()
```



完美热力图的绘制太简单了！不需要做任何的特殊处理，只需要将绘图数据扔给heatmap函数即可。想想是不是有点小激动啊~激动过后，还得跟着步骤操作一表哦~