Python 数据可视化作业 2

班级：2018211313 班　　学号：2018211366　　姓名：蒋潇逸

版本：1.0

更新：*December 13, 2020*

本文档是 Python 数据可视化作业 2 报告。

目录

1　作业 1　2

2　作业 2　7

3　作业 3　9

4　作业 4　10

5　作业 5　12

6　作业 6　14

7　作业 7　16

1 作业 1

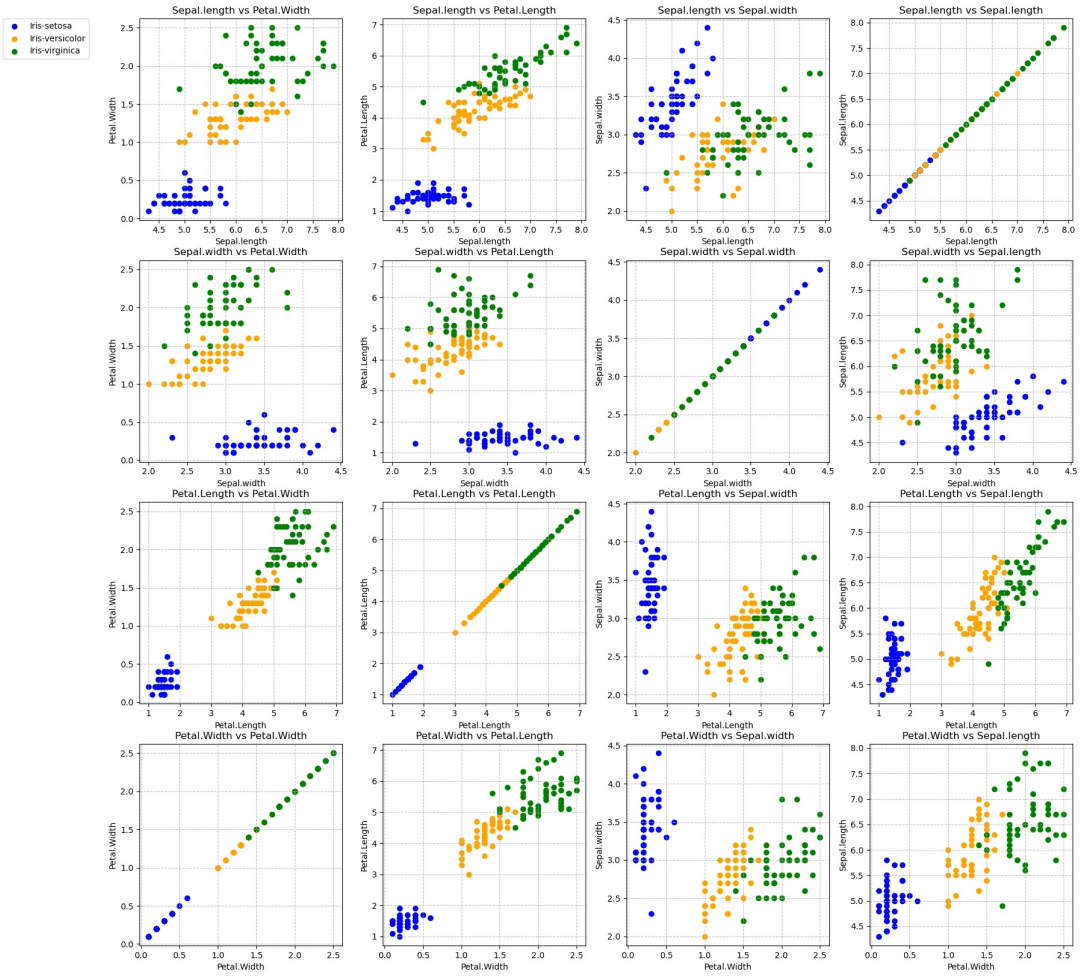
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# 生成16个子图
fig = plt.figure(figsize=(20, 20))
ax1 = fig.add_subplot(441)
ax2 = fig.add_subplot(442)
ax3 = fig.add_subplot(443)
ax4 = fig.add_subplot(444)
ax5 = fig.add_subplot(445)
ax6 = fig.add_subplot(446)
ax7 = fig.add_subplot(447)
ax8 = fig.add_subplot(448)
ax9 = fig.add_subplot(449)
ax10 = fig.add_subplot(4, 4, 10)
ax11 = fig.add_subplot(4, 4, 11)
ax12 = fig.add_subplot(4, 4, 12)
ax13 = fig.add_subplot(4, 4, 13)
ax14 = fig.add_subplot(4, 4, 14)
ax15 = fig.add_subplot(4, 4, 15)
ax16 = fig.add_subplot(4, 4, 16)
# 读取数据
iris = pd.read_csv('iris.csv')
colors = ['b', 'orange', 'g']  # 定义三种散点的颜色
Species = iris.Species.unique()  # 对类别去重
for i in range(len(Species)):
    ax1.scatter(iris.loc[iris.Species == Species[i], 'Sepal.length'],
                iris.loc[iris.Species == Species[i], 'Petal.Width'],
                   s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax1.set_title('Sepal.length vs Petal.Width')
ax1.set_xlabel('Sepal.length')
ax1.set_ylabel('Petal.Width')
ax1.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
ax1.legend(bbox_to_anchor=(-0.25, 1), loc=1, borderaxespad=0)

```

```
for i in range(len(Species)):  # x和y轴交换一下位置
    ax2.scatter(iris.loc[iris.Species == Species[i], 'Sepal.length'],
                iris.loc[iris.Species == Species[i], 'Petal.Length'],
                    s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax2.set_title('Sepal.length vs Petal.Length')
ax2.set_xlabel('Sepal.length')
ax2.set_ylabel('Petal.Length')
ax2.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax3.scatter(iris.loc[iris.Species == Species[i], 'Sepal.length'],
                iris.loc[iris.Species == Species[i], 'Sepal.width'],
                   s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax3.set_title('Sepal.length vs Sepal.width')
ax3.set_xlabel('Sepal.length')
ax3.set_ylabel('Sepal.width')
ax3.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax4.scatter(iris.loc[iris.Species == Species[i], 'Sepal.length'],
                iris.loc[iris.Species == Species[i], 'Sepal.length'],
                    s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax4.set_title('Sepal.length vs Sepal.length')
ax4.set_xlabel('Sepal.length')
ax4.set_ylabel('Sepal.length')
ax4.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):
    ax5.scatter(iris.loc[iris.Species == Species[i], 'Sepal.width'],
                iris.loc[iris.Species == Species[i], 'Petal.Width'],
                   s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax5.set_title('Sepal.width vs Petal.Width')
ax5.set_xlabel('Sepal.width')
ax5.set_ylabel('Petal.Width')
ax5.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
```

```
72
73  for i in range(len(Species)):  # x和y轴交换一下位置
74      ax6.scatter(iris.loc[iris.Species == Species[i], 'Sepal.width'],
75                  iris.loc[iris.Species == Species[i], 'Petal.Length'],
                        s=35, c=colors[i], label=Species[i])
76  # 添加轴标签和标题
77  ax6.set_title('Sepal.width vs Petal.Length')
78  ax6.set_xlabel('Sepal.width')
79  ax6.set_ylabel('Petal.Length')
80  ax6.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
81
82  for i in range(len(Species)):  # x和y轴交换一下位置
83      ax7.scatter(iris.loc[iris.Species == Species[i], 'Sepal.width'],
84                  iris.loc[iris.Species == Species[i], 'Sepal.width'],
                       s=35, c=colors[i], label=Species[i])
85  # 添加轴标签和标题
86  ax7.set_title('Sepal.width vs Sepal.width')
87  ax7.set_xlabel('Sepal.width')
88  ax7.set_ylabel('Sepal.width')
89  ax7.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
90
91  for i in range(len(Species)):  # x和y轴交换一下位置
92      ax8.scatter(iris.loc[iris.Species == Species[i], 'Sepal.width'],
93                  iris.loc[iris.Species == Species[i], 'Sepal.length'],
                        s=35, c=colors[i], label=Species[i])
94  # 添加轴标签和标题
95  ax8.set_title('Sepal.width vs Sepal.length')
96  ax8.set_xlabel('Sepal.width')
97  ax8.set_ylabel('Sepal.length')
98  ax8.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
99
100 for i in range(len(Species)):
101     ax9.scatter(iris.loc[iris.Species == Species[i], 'Petal.Length'],
102                 iris.loc[iris.Species == Species[i], 'Petal.Width'],
                       s=35, c=colors[i], label=Species[i])
103 # 添加轴标签和标题
104 ax9.set_title('Petal.Length vs Petal.Width')
105 ax9.set_xlabel('Petal.Length')
106 ax9.set_ylabel('Petal.Width')
```

```
ax9.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax10.scatter(iris.loc[iris.Species == Species[i], 'Petal.Length'],
                 iris.loc[iris.Species == Species[i], 'Petal.Length'],
                 s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax10.set_title('Petal.Length vs Petal.Length')
ax10.set_xlabel('Petal.Length')
ax10.set_ylabel('Petal.Length')
ax10.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax11.scatter(iris.loc[iris.Species == Species[i], 'Petal.Length'],
                 iris.loc[iris.Species == Species[i], 'Sepal.width'],
                 s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax11.set_title('Petal.Length vs Sepal.width')
ax11.set_xlabel('Petal.Length')
ax11.set_ylabel('Sepal.width')
ax11.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax12.scatter(iris.loc[iris.Species == Species[i], 'Petal.Length'],
                 iris.loc[iris.Species == Species[i], 'Sepal.length'],
                 s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax12.set_title('Petal.Length vs Sepal.length')
ax12.set_xlabel('Petal.Length')
ax12.set_ylabel('Sepal.length')
ax12.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):
    ax13.scatter(iris.loc[iris.Species == Species[i], 'Petal.Width'],
                 iris.loc[iris.Species == Species[i], 'Petal.Width'],
                 s=35, c=colors[i], label=Species[i])
```

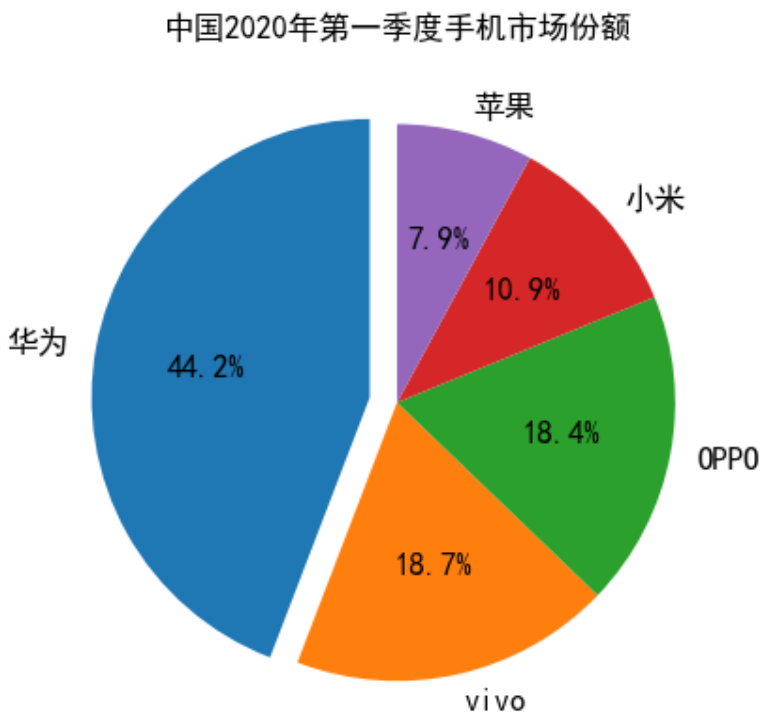
```
# 添加轴标签和标题
ax13.set_title('Petal.Width vs Petal.Width')
ax13.set_xlabel('Petal.Width')
ax13.set_ylabel('Petal.Width')
ax13.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax14.scatter(iris.loc[iris.Species == Species[i], 'Petal.Width'],
                 iris.loc[iris.Species == Species[i], 'Petal.Length'],
                     s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax14.set_title('Petal.Width vs Petal.Length')
ax14.set_xlabel('Petal.Width')
ax14.set_ylabel('Petal.Length')
ax14.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax15.scatter(iris.loc[iris.Species == Species[i], 'Petal.Width'],
                 iris.loc[iris.Species == Species[i], 'Sepal.width'],
                    s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax15.set_title('Petal.Width vs Sepal.width')
ax15.set_xlabel('Petal.Width')
ax15.set_ylabel('Sepal.width')
ax15.grid(True, linestyle='--', alpha=0.8)  # 设置网格线

for i in range(len(Species)):  # x和y轴交换一下位置
    ax16.scatter(iris.loc[iris.Species == Species[i], 'Petal.Width'],
                 iris.loc[iris.Species == Species[i], 'Sepal.length'],
                     s=35, c=colors[i], label=Species[i])
# 添加轴标签和标题
ax16.set_title('Petal.Width vs Sepal.length')
ax16.set_xlabel('Petal.Width')
ax16.set_ylabel('Sepal.length')
ax16.grid(True, linestyle='--', alpha=0.8)  # 设置网格线
plt.savefig("C:\\Users\\Lenovo\\Desktop\\prm1.jpg")
plt.show()
```



2 作业 2

```
import matplotlib.pyplot as plt
import matplotlib
from matplotlib import font_manager as fm

matplotlib.rcParams['font.sans-serif'] = ['SimHei']

label_list = ["华为", "vivo", "OPPO", "小米", "苹果"]  # 各部分标签

```

```
size = [28.4, 12.0, 11.8, 7.0, 5.1]  # 中国2020年第一季度手机出货量
explode = [0.1, 0, 0, 0, 0]  # 各部分的突出显示比例

# 使得第一个扇区在12点方向开始
patches, texts, autotexts = \
    plt.pie(size, explode=explode, labels=label_list, labeldistance
       =1.1, autopct="%1.1f%%", shadow=False, startangle=90,
            pctdistance=0.6)

proptease = fm.FontProperties()
proptease.set_size('large')
# font size include: ‘xx-small’ ,x-small’ ,'small’ ,'medium’ , ‘
   large’ , ‘x-large’ , ‘xx-large’  or number, e.g. '12'
plt.setp(texts, fontproperties=proptease)
plt.setp(autotexts, fontproperties=proptease)
plt.title('中国2020年第一季度手机市场份额', fontproperties=proptease)
plt.show()
```

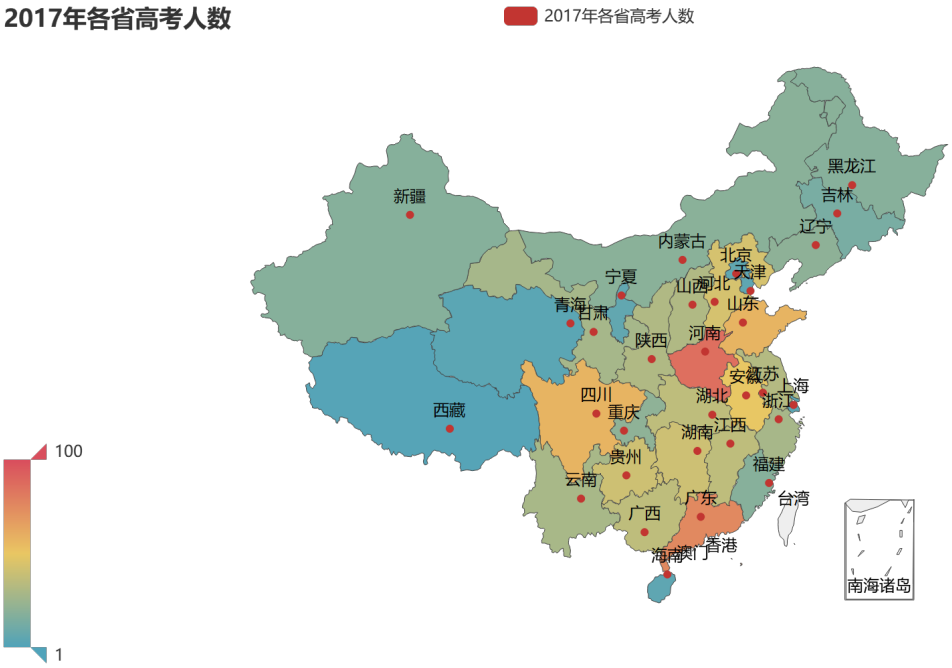


在饼图中，数据项数在 4 到 7 项是比较合理的，既能展示出数据的细节、又不会太过臃肿和冗余。

3 作业 3

```
from pyecharts import options as opts
from pyecharts.charts import Map
import pandas as pd
from pyecharts.render import make_snapshot
from snapshot_phantomjs import snapshot


class Data:
    provinces = []
    num = []


def map_visualmap() -> Map:
    fileNameStr = '中国大学数量.csv'
    df = pd.read_csv(fileNameStr, encoding='utf-8')  # 不加dtype=str
    Data.num = df[lambda df: df.columns[1]].tolist()
    del Data.num[0]
    for i in range(len(Data.num)):
        Data.num[i] = float(Data.num[i].replace("万", ""))
    print(Data.num)
    Data.provinces = df[lambda df: df.columns[0]].tolist()
    del Data.provinces[0]
    print(Data.provinces)
    c = (
        Map()
            .add("2017年各省高考人数", [list(z) for z in zip(Data.
               provinces, Data.num)], "china")
            .set_global_opts(
            title_opts=opts.TitleOpts(title="2017年各省高考人数"),
            visualmap_opts=opts.VisualMapOpts(min_=1.0, max_=100.0))
            .set_series_opts(label_opts=opts.LabelOpts(is_show=True))
    )
    return c
```

```
map_visualmap().render("map" + str('0') + ".html")
make_snapshot(snapshot, map_visualmap().render(), "map1.png")
print("done")
```

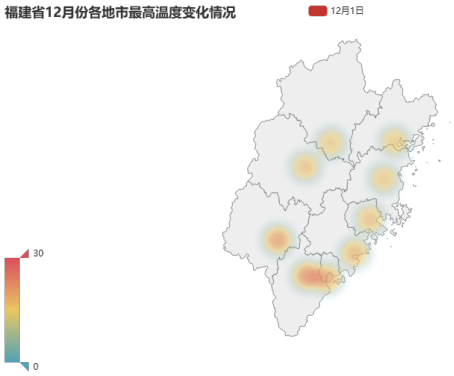


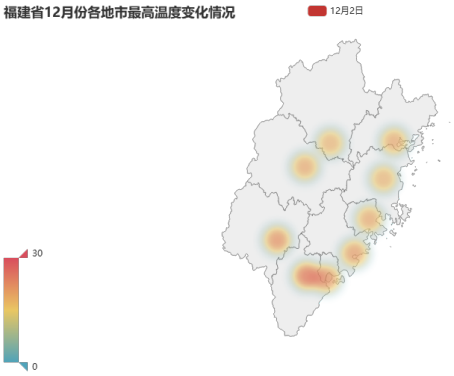
4 作业 4

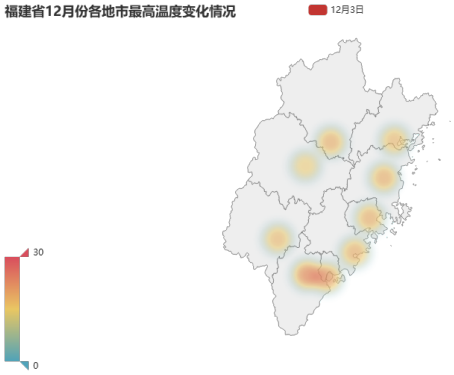
```
from pyecharts import options as opts
from pyecharts.charts import Geo
from pyecharts.globals import ChartType
import random
from pyecharts.render import make_snapshot
from snapshot_phantomjs import snapshot


class Data:
    guangdong_city = ["福州市", "厦门市", "泉州市", "莆田市", "三明市
        ", "漳州市", "南平市", "龙岩市", "宁德市"]

```

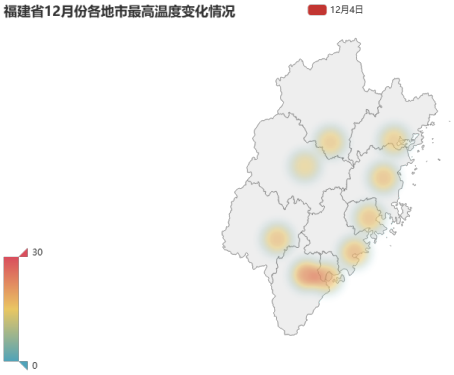
```
def geo_fujian(title, data) -> Geo:
    c = (
        Geo()
            .add_schema(maptype="福建")
            .add(
            title,
            [list(z) for z in zip(Data.guangdong_city, data)],
            type_=ChartType.HEATMAP,
        )
            .set_global_opts(
            visualmap_opts=opts.VisualMapOpts(max_=30),  # is_
               piecewise=True),
            title_opts=opts.TitleOpts(title="福建省12月份各地市最高温
               度变化情况"),
        )
    )
    return c


TEMP = [[17, 20, 19, 18, 18, 22, 17, 21, 17],
        [19, 22, 21, 20, 20, 24, 19, 22, 20],
        [19, 21, 19, 19, 16, 22, 19, 18, 18],
        [18, 20, 19, 18, 15, 22, 17, 18, 17],
        [19, 20, 19, 20, 18, 22, 17, 19, 18],
        [21, 23, 22, 21, 21, 24, 21, 22, 20],
        [21, 24, 23, 21, 21, 26, 22, 23, 20],
        [17, 19, 19, 17, 18, 21, 18, 21, 18],
        [18, 21, 21, 19, 21, 23, 21, 23, 18],
        [19, 22, 22, 20, 20, 24, 19, 22, 18]]
for i in range(10):
    str_date = "12月" + str(i + 1) + "日"
    make_snapshot(snapshot, geo_fujian(str_date, TEMP[i]).render(),
                  str(i + 1) + ".png", pixel_ratio=1)
```

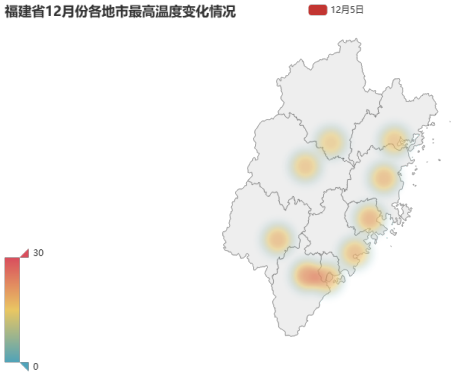
下列给出了 12 月 1 日到 12 月 10 日福建省的热力图，gif 变化图可见附件。

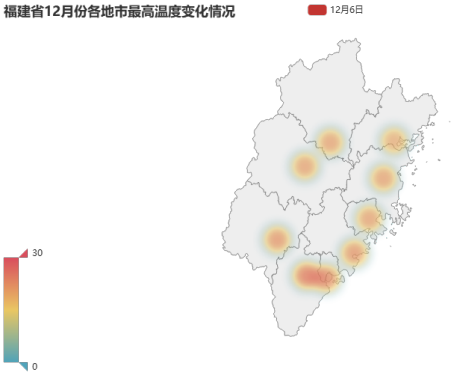

福建省12月份各地市最高温度变化情况
12月1日
30
0

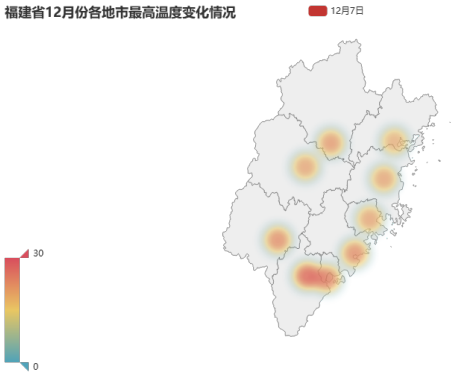

福建省12月份各地市最高温度变化情况
12月2日
30
0

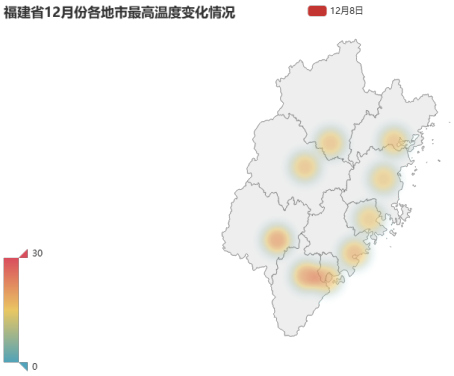

福建省12月份各地市最高温度变化情况
12月3日
30
0

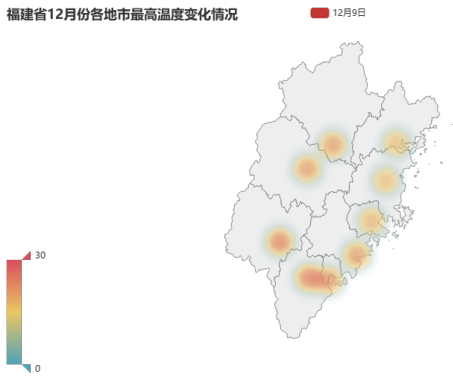

福建省12月份各地市最高温度变化情况
12月4日
30
0

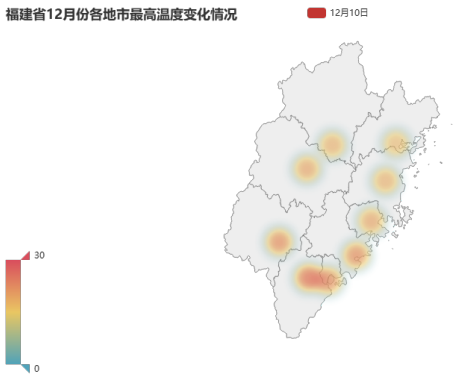

福建省12月份各地市最高温度变化情况
12月5日
30
0


福建省12月份各地市最高温度变化情况
12月6日
30
0


福建省12月份各地市最高温度变化情况
12月7日
30
0

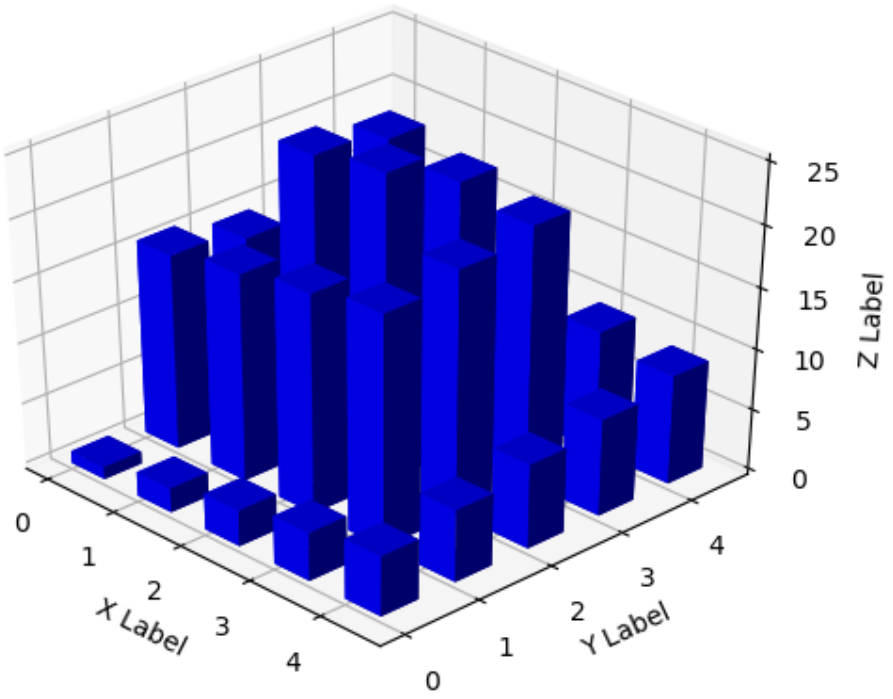

福建省12月份各地市最高温度变化情况
12月8日
30
0





5 作业 5

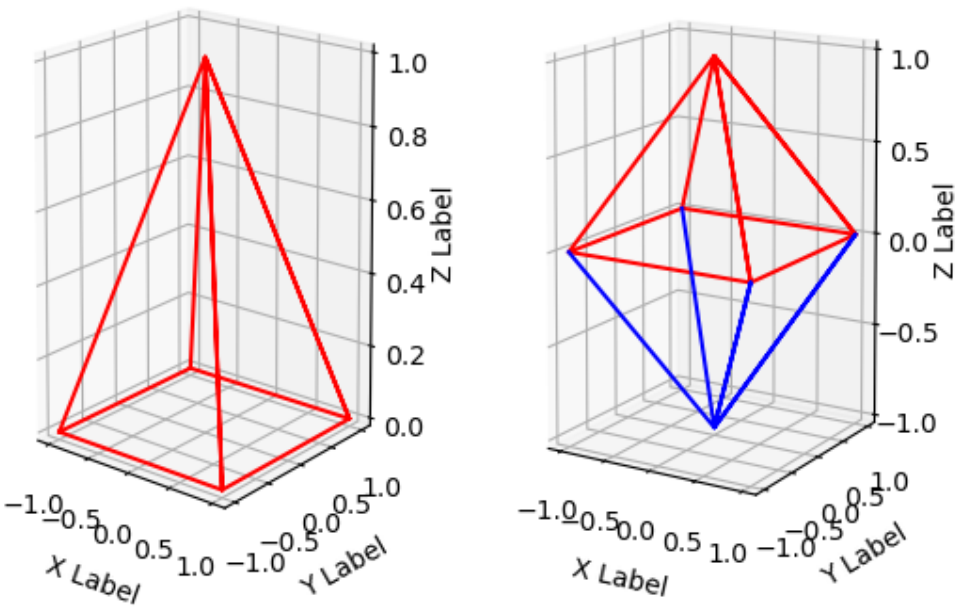
```python
import numpy as np
import matplotlib.pyplot as plt
import math
from mpl_toolkits.mplot3d import Axes3D

# 1.生成fig对象和ax对象
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')

# 2.生成数据
x = np.array([0, 1, 2, 3, 4, 4, 4, 4, 4, 3, 2, 1, 0, 0, 0, 0, 1, 2,
    3, 3, 3, 2, 1, 1, 2])  # 生成x轴的数据
y = np.array([0, 0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4, 4, 3, 2, 1, 1, 1,
    1, 2, 3, 3, 3, 2, 2])  # 生成y轴的数据
z = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
    17, 18, 19, 20, 21, 22, 23, 24, 25])  # 生成z轴的数据

width = depth = 0.5  # width和depth表示直方柱的宽度和深度在单元格中比例
# 3.调用bar3d, 画3D直方图
ax.bar3d(x, y, 0, width, depth, z, shade=True, color='b')
plt.savefig("C:\\Users\\Lenovo\\Desktop\\5.jpg")
# 4.显示图形
plt.show()
```



6 作业 6

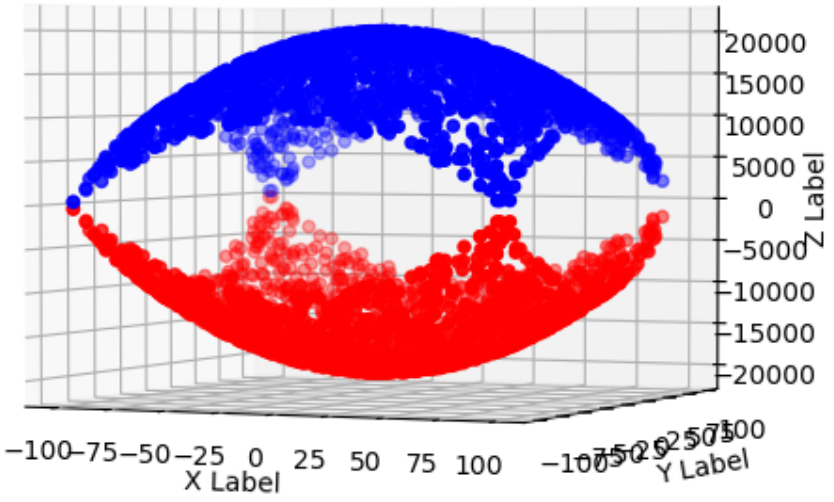
```
import numpy as np
import matplotlib.pyplot as plt
import math
from mpl_toolkits.mplot3d import Axes3D

# 1.生成fig对象和ax对象
fig = plt.figure()
ax1 = fig.add_subplot(121, projection='3d')
ax2 = fig.add_subplot(122, projection='3d')
ax1.set_xlabel('X Label')
ax1.set_ylabel('Y Label')
ax1.set_zlabel('Z Label')
ax2.set_xlabel('X Label')
```

```
ax2.set_ylabel('Y Label')
ax2.set_zlabel('Z Label')

# 2.生成数据
x1 = np.array([-1, 1, 1, -1, -1, 0, 1, 0, 1, 0, -1])  # 生成x轴的数据
y1 = np.array([-1, -1, 1, 1, -1, 0, -1, 0, 1, 0, 1])  # 生成y轴的数据
z1 = np.array([0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0])  # 生成z轴的数据
x2 = np.array([-1, 0, 1, 0, 1, 0, -1])  # 生成x轴的数据
y2 = np.array([-1, 0, -1, 0, 1, 0, 1])  # 生成y轴的数据
z2 = np.array([0, -1, 0, -1, 0, -1, 0])  # 生成z轴的数据

# 3.调用plot, 画3D的线图
ax1.plot(x1, y1, z1, "r")
ax2.plot(x1, y1, z1, "r")
ax2.plot(x2, y2, z2, "b")
plt.savefig("C:\\Users\\Lenovo\\Desktop\\6.jpg")
# 4.显示图形
plt.show()
```



7 作业 7

```python
import numpy as np
import matplotlib.pyplot as plt
import math
from mpl_toolkits.mplot3d import Axes3D

# 1.生成fig对象和ax对象
fig = plt.figure()
ax = fig.add_subplot(projection='3d')
ax.set_xlabel('X Label')
ax.set_ylabel('Y Label')
ax.set_zlabel('Z Label')

# 2.生成数据
x2 = np.random.randint(-100, 100, 2000)  # x轴，生成5000个在-100到100之间的整数
y2 = np.random.randint(-100, 100, 2000)  # y轴，生成5000个在-100到100之间的整数
z2 = 20000 - x2 ** 2 - y2 ** 2  # 生成z轴的数据
z3 = -20000 + x2 ** 2 + y2 ** 2  # 生成z轴的数据
# z2 = x2**3+ y2**3                    #生成z轴的数据
ax.scatter(x2, y2, z2, zdir='z', s=20, c='b', depthshade=True)
ax.scatter(x2, y2, z3, zdir='z', s=20, c='r', depthshade=True)
plt.savefig("C:\\Users\\Lenovo\\Desktop\\7.jpg")
# 4.显示图形
plt.show()
```


20000
15000
10000
5000
0
−5000
−10000
−15000
−20000
Z Label
−100 −75 −50 −25 0 25 50 75 100
X Label
Y Label