Statistical Analysis of Student Status

11510449 翁健林 **(70%)** 11510705 杨超 **(30%)**

Initialization

Language: Python 3.7 (pandas, matplotlib, PIL, tkinter)

Pandas: Data processing

Matplotlib: Generate chart

PIL: Import image

Tkinter: Visual interface

IDE: PyCharm

Function

1. Provincial and urban distribution
2. Statistics of various graduation directions
3. Statistics of study abroad
4. domestic postgraduate school
5. Statistics of work (including work citied, work types and salary)

Data

The following are the columns used:

姓名	省	市	区县	毕业GPA	毕业去向	留学地	留学大学	专业1	大学	专业2	工作省份	工作城市	深造学位	工作单位	月薪
李涵一	安徽	宣城	宣州	3.3	毕业工作						广东	深圳			15
陈世程	广东	深圳	保安	2.5	毕业工作						广东	深圳		其他企业	30
杨洽	广东	潮州	饶平	3.5	毕业工作						广东	深圳		自己创业	16
钟浩霖	广西	梧州	万秀	3.5	毕业工作						广东	深圳		自己创业	8
常青崧	河北	邯郸	丛台	3	毕业工作						广东	深圳		国企	8
张猛	河南	郑州	金水	3	毕业工作						广东	深圳			10
罗航宇	湖南	娄底	新化	3.68	毕业工作						广东	深圳		其他企业	14
朱鸿仪	江苏	南京	鼓楼	3.3	毕业工作						江苏	南京		国企	15
范晨辉	江西	抚州	临川	2.2	毕业工作						广东	广州		其他企业	10
廖思卿	江西	萍乡	安源	3.8	毕业工作						广东	深圳		其他企业	15

Which including Provinces, cities, districts, GPA, gra_direction (what to do after graduation), countries, universities, work_provinces, work_city and salary.

Algorithm Design

Read Data → Data Preprocessing → Data Processing → Generate Chart → Visual Interface → Save Data

At first, read the data as csv file, preprocessing data, drop the data that is not finished, for example:

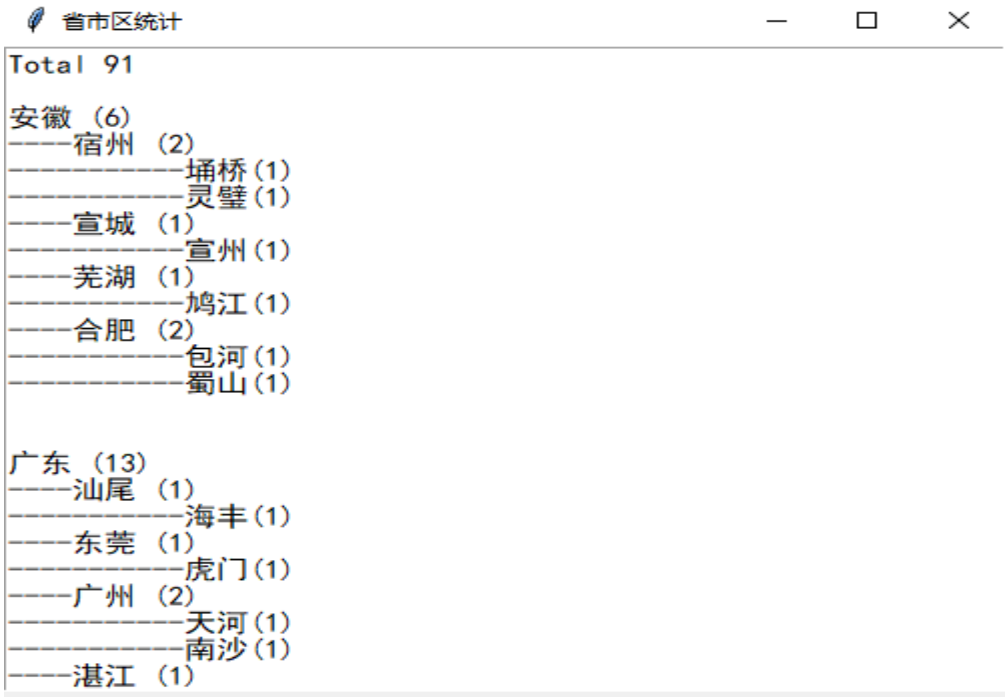
姓名	省	市	区县	毕业GPA	毕业去向	留学地	留学大学	专业1	大学	专业2	工作省份	工作城市	深造学位	工作单位	月薪
刁金龙	广东														
丁立德	广东														
高王珏	广东														
谷岳洪	广东														
何雨京	广东														
赫小榕	广东														
李运来	广东														

This processing uses panda, and select data we need. Next is using matplotlib to generate chart, and save as PNG file. At last use tkinter to visual interface.

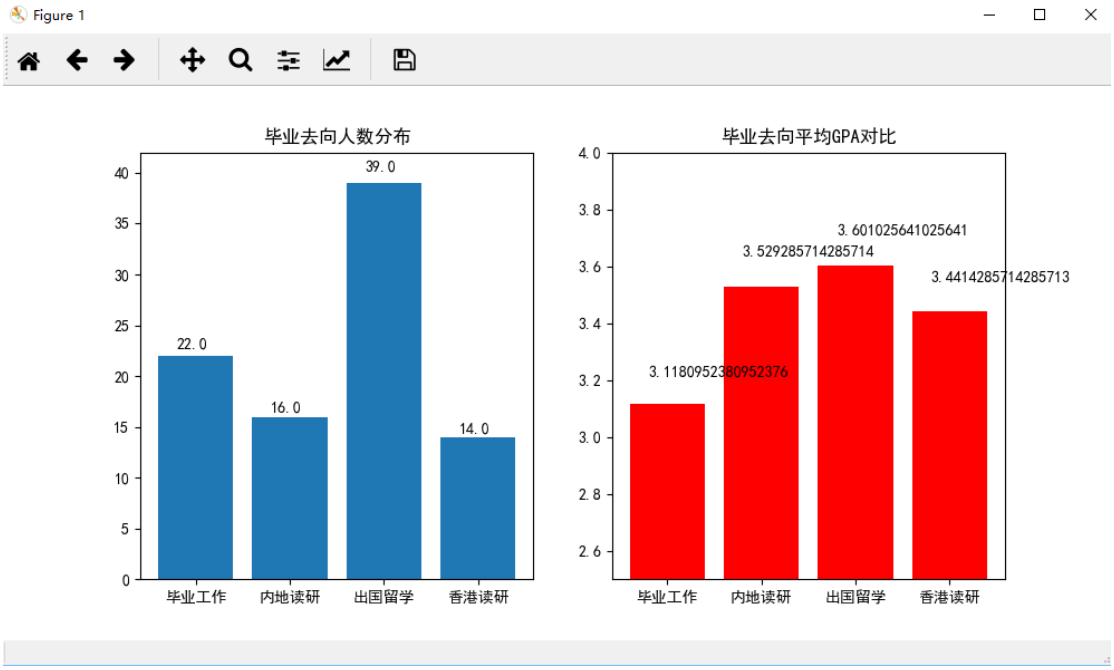
Output

Initial interface:

Click the first button “打开省市文件”, the following window appears:

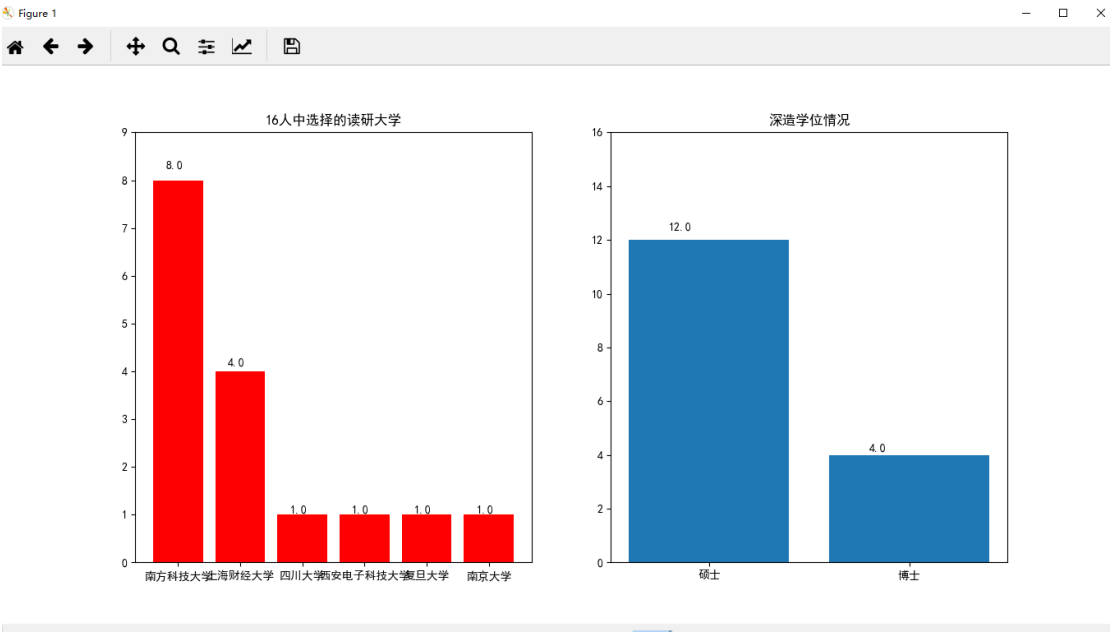


Click the second button “毕业去向统计”, we will get the statistics of graduation:



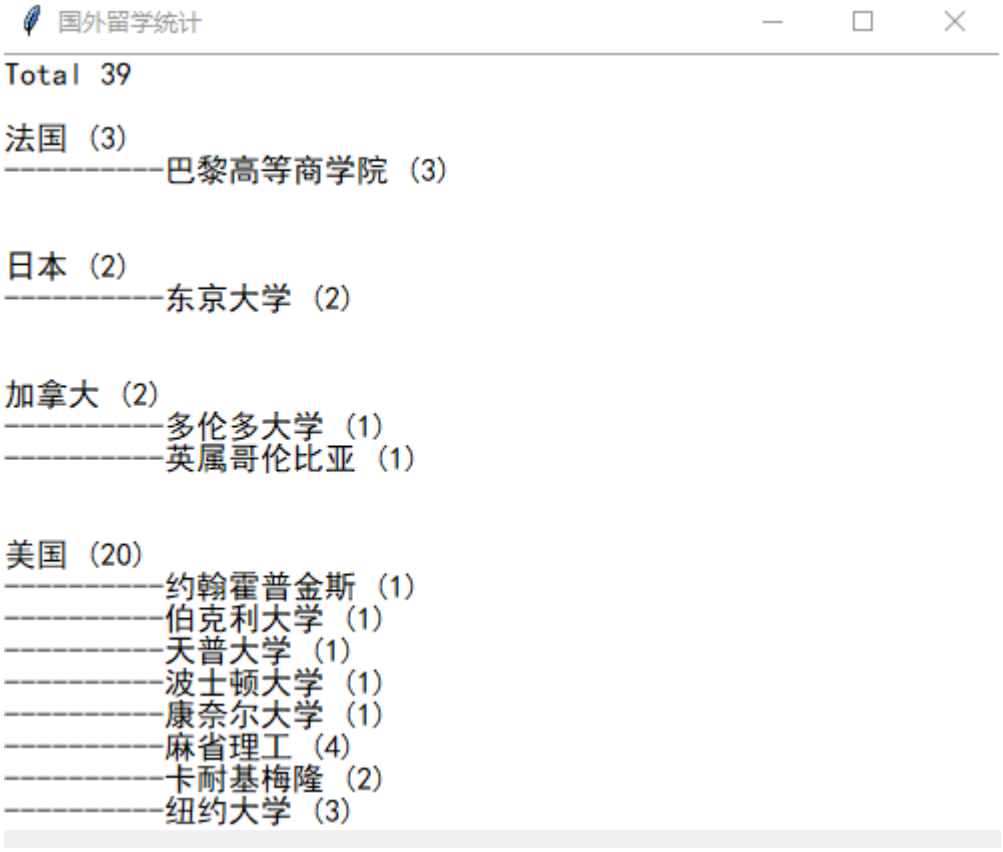
Seen from the first chart, the quantity of study abroad is the largest, including 39 students, and the following is going to work after graduation, around 22 students. From the second chart, we know that “Study abroad” has the highest average GPA (3.60), in the other hand, “Go Work” has the lowest GPA.

Next click the “国内深造统计” button, the following window shows:



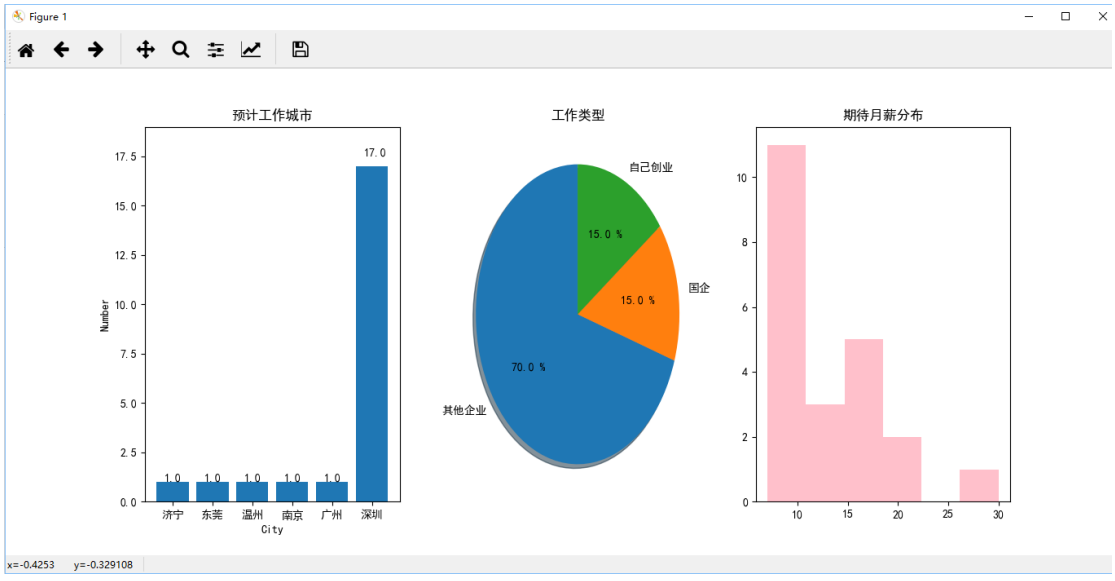
We can get the information that most students would like to stay in SUSTech, 3 quarters students are going to be Master’s, quarter is going to be Doctor.

Close last window, and click “国外留学统计” button, show the new window:



It is the same as the first window, because I use the same algorithm. Rolling the mouse, more information will be showed. Most students are supposed to go to American for new study. Of course, it’s a good choice.

Click the last button “毕业工作统计”, the charts of going to work will be showed below:



It’s significant that almost all students would like to work in Shenzhen, most students chose other enterprises, and the salary requested is below 10k.

Code Analyze

Import the packages that are needed

```
#-*- coding: utf-8 -*-

import pandas as pd
import matplotlib.pyplot as plt
import tkinter as tk
import PIL
from tkinter import *
from PIL import ImageTk, Image
```

Read file and drop Nan data

```
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['font.serif'] = ['SimHei']
```

Solve the problem that chart can't show Chinese

```
#解决不显示中文的问题
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['font.serif'] = ['SimHei']
```

Create a new function to Distinguishing provinces and cities, then save as TXT file.

```
#获取地区分布情况
def getProvince():
    path = 'E:\desktop\savefile2\province.txt'
    doc = open(path,'w+',encoding="utf-8")
    provincelist = df['省']

    #用字典计数
    dict = {}
    for province in provincelist:
        dict[province] = dict.get(province, 0) + 1

    citylist = df['市']
    districtlist = df['区县']
    print('Total '+str(len(provincelist))+'\n', file=doc)
    for key, value in zip(dict.keys(), dict.values()):
        cities = list(set(citylist[df['省']==key]))
        # print (cities)
        print(key + ' (' + str(value) + ')',file=doc)
        print(key + ' (' + str(value) + ')')
        for city in cities:
            print('----'+city+' (' + str(citylist.tolist().count(city))+ ')', file=doc)
            print('----'+city+' (' + str(citylist.tolist().count(city))+ ')')
            districts = list(set(districtlist[df['市'] == city]))
            for district in districts:
                print('----------' + str(district) + '(' + str(districtlist.tolist().count(district)) + ')',file=doc)
                print('----------' + str(district) + '(' + str(districtlist.tolist().count(district)) + ')')
        print('\n',file=doc)
        print('\n')
    print('Save to doc success')
    doc.close()
```

Generate a new window and show the TXT file, set the font "SimHei".

```
root = Tk()
root.title('省市区统计')
root.geometry('500x400')
text = Text(root, font='SimHei')

# filename = 'E:\desktop\savefile2\province.txt'
with open(path, encoding='utf-8') as f:
    for line in f:
        text.insert(END, line)
        # print (line, end='')

text.pack()
root.mainloop()
```

Graduation Statistics Function and Save as PNG file.

```
#统计毕业去向
def graduation():
    gras = df['毕业去向'].tolist()
    print ('Total ', len(gras))
    gra = list(set(gras))
    count = []
    for i in gra:
        count.append(gras.count(i))
    labels = gra
    fracs = count
    explode = [0,0,0,0]
    plt.axes(aspect=1)
    #画出饼图
    plt.pie(x=fracs, labels=labels, explode=explode, autopct = '%3.1f %%',
            shadow=True, labeldistance=1.1, startangle=90, pctdistance=0.6)
    plt.legend()
    #将图片保存至本地文件
    plt.savefig('E:\desktop\savefile2\graduatePie.png')
    print('Save image success!')
    plt.show()
```

Mean GPA Statistics and Save chart as PNG file.

```
#平均GPA统计
def meanGPA():
    # df_mean = df.groupby('毕业去向')['毕业GPA'].mean()
    # gras = df['毕业去向'].tolist()
    gra = list(set(df['毕业去向'].tolist()))
    mean = []
    for i in gra:
        meangpa = df['毕业GPA'][df['毕业去向']==i].mean()
        mean.append(meangpa)
    print(gra,'\n',mean)
    a = plt.bar(gra, mean, label='Average GPA')
    # plt.legend()
    autolabel(a)
    plt.ylim((0,4.0))
    plt.xlabel('毕业去向')
    plt.ylabel('平均GPA')
    plt.title('Average GPA Statistics')
    plt.savefig('E:\desktop\savefile2\meanGPA.png')
    print ('Save image success')
    plt.show()
```

Combine “Graduation” and “Mean GPA” charts as a new chart

```
#毕业去向人数分布以及毕业去向平均GPA
def twoPic():
    gras = df['毕业去向'].tolist()
    gra = list(set(gras))
    count = []
    mean = []
    for i in gra:
        count.append(gras.count(i))

    for i in gra:
        meangpa = df['毕业GPA'][df['毕业去向'] == i].mean()
        mean.append(meangpa)
    # print (mean)


    plt.figure(1,figsize=(10,5))
    ax1 = plt.subplot(1,2,1)
    ax2 = plt.subplot(1,2,2)
    plt.sca(ax1)
    a = plt.bar(gra, count, label='Number')
    autolabel(a)
    plt.ylim((0,42))
    plt.title('毕业去向人数分布')
    plt.sca(ax2)
    b = plt.bar(gra, mean, label='Average GPA',fc = 'r')
    autolabel(b)
    plt.ylim((2.5,4.0))
    plt.title('毕业去向平均GPA对比')
    plt.savefig('E:\desktop\savefile2\GPA_Count.png')
    print('Save image success!')
    plt.show()
```

Work Statistics Function

```
#工作统计
def goWork():
    work = df[['姓名', '性别', '毕业GPA', '工作省份', '工作城市',
    '工作单位', '月薪']][df['毕业去向']=='毕业工作']
    cities = list(set(work['工作城市']))
    print (cities)
    count = []
    for city in cities:
        count.append(work['工作城市'].tolist().count(city))
    print (count)

    danwei = df['工作单位'][df['毕业去向'] == '毕业工作'].dropna()
    worktype = pd.value_counts(danwei)
    wt = worktype.index.tolist()
    wn = worktype.tolist()
```

Three charts in one figure, and save it.

```
plt.figure(1, figsize=(14,6))
ax1 = plt.subplot(1,3,1)
plt.sca(ax1)
a = plt.bar(cities, count)
autolabel(a)
plt.ylim((0,19))
plt.xlabel('City')
plt.ylabel('Number')
plt.title('预计工作城市')

ax2 = plt.subplot(1,3,2)
plt.sca(ax2)
plt.pie(x = wn, labels=wt, startangle=90, autopct = '%3.1f %%',
        labeldistance=1.1, pctdistance=0.6)
plt.title('工作类型')

ax3 = plt.subplot(1,3,3)
plt.sca(ax3)
salary = df['月薪'][df['毕业去向'] == '毕业工作']
plt.hist(salary, bins=6, fc = 'pink')
plt.title('期待月薪分布')

plt.savefig('E:\desktop\savefile2\goWork.png')
print('Save image success!')
plt.show()
```

Show values in chart, because some charts don't show values.

```
def autolabel(rects):
    for rect in rects:
        height = rect.get_height()
        plt.text(rect.get_x() + rect.get_width() / 2. - 0.2, 1.03 * height, '%s' % float(height))
```

Main window function.

```
def thismain():
    root = tk.Tk()
    root.title('这是一个巨丑的小程序')

    canvas = tk.Canvas(root, width=620, height=420, bd=0, highlightthickness=0)
    imgpath = 'E:/Desktop/LOGO.png'
    img = PIL.Image.open(imgpath)
    photo = ImageTk.PhotoImage(img)

    canvas.create_image(300, 200, image=photo)
    canvas.pack()

    Button(root, text='打开省市文件',command = getProvince).pack(side=LEFT, expand=YES, fill=X )
    Button(root, text='毕业去向统计', command = twoPic).pack(side=LEFT, expand=YES, fill=X)
    Button(root, text='国内深造统计', command = inChina).pack(side=LEFT, expand=YES, fill=X)
    Button(root, text='国外留学统计', command = inForeign).pack(side=LEFT, expand=YES, fill=X)
    Button(root, text='毕业工作统计', command = goWork).pack(side=LEFT, expand=YES, fill=X)

    root.mainloop()

if __name__ == '__main__':
    thismain()
```

Run the main function, get the main window:

Full-code is submitted to annex and github

[The full-code is submitted to github]

Save files to local: