本次实验选择编程类工具Pyecharts。

在Pycharm中新建python项目，并引入pyecharts，openpyxl等必要包。

创建main.py，写入以下内容。

import copy

import openpyxl

from pyecharts.charts import Graph, Tree, WordCloud, Page

from pyecharts import options as opts

from pyecharts import globals

globals.\_WarningControl.ShowWarning = False

bigBoss = ['1007', '1059', '1068']

def getData():

filename = "email\_dev\_inside.xlsx"

ws = openpyxl.load\_workbook(filename)['Sheet1']

theData, resNodes, theLinks, theCategories, theNodes = [], [], [], [], set()

for i in range(2, ws.max\_row + 1):

row = str(i)

theData.append([ws['G' + row].value[:4], ws['H' + row].value[:4], ws['I' + row].value])

theLinks.append({'source': ws['G' + row].value[:4], 'target': ws['H' + row].value[:4]})

theNodes.add(ws['G' + row].value[:4])

theNodes.add(ws['H' + row].value[:4])

for node in theNodes:

resNodes.append({'name': node, 'symbolSize': 1, 'draggable': 'False', 'value': 1, 'category': '',

'label': {'normal': {'show': 'False'}}})

return theData, resNodes, theLinks, theCategories

def drawEmailGraph(graph\_nodes: list, graph\_links: list, graph\_categories: list):

Graph(opts.InitOpts(width="2000px", height="2000px")).add(

"",

graph\_nodes,

graph\_links,

graph\_categories,

repulsion=50,

linestyle\_opts=opts.LineStyleOpts(curve=0.2),

label\_opts=opts.LabelOpts(is\_show=False),

).set\_global\_opts(

legend\_opts=opts.LegendOpts(is\_show=False),

title\_opts=opts.TitleOpts(title="研发部邮件来往图"),

).render("研发部邮件来往图.html")

def getLittleBoss(big\_boss: list):

little\_boss = {}

for bBoss in big\_boss:

little\_boss[bBoss] = set()

for link in links:

if link['source'] in big\_boss and link['target'] not in bigBoss:

little\_boss[link['source']].add(link['target'])

elif link['target'] in big\_boss and link['source'] not in bigBoss:

little\_boss[link['target']].add(link['source'])

for k, v in little\_boss.items():

little\_boss[k] = list(v)

little\_boss[k].sort()

return little\_boss

def getRank():

the\_rank = {}

for bBoss in bigBoss:

the\_rank[bBoss] = {}

littleBoss = getLittleBoss(bigBoss)

for k, v in littleBoss.items():

employee = getLittleBoss(v)

for lBoss in v:

the\_rank[k][lBoss] = employee[lBoss]

return the\_rank

def getGroupEmail():

email\_content = copy.deepcopy(rank)

for bBoss, groups in email\_content.items():

for lBoss, employeeList in groups.items():

email\_content[bBoss][lBoss] = getListEmail(employeeList)

return email\_content

def getListEmail(theEmployeeList: list):

emailContent = {}

wordList = []

for d in data:

if d[0] in theEmployeeList or d[1] in theEmployeeList:

if emailContent.\_\_contains\_\_(d[2]):

emailContent[d[2]] += 1

else:

emailContent[d[2]] = 0

for k, v in emailContent.items():

wordList.append(tuple((k, str(v))))

return wordList

def getTreeData():

tree\_data, children = {'name': '成员结构图', 'children': []}, []

for bBoss, groups in rank.items():

tree\_data['children'].append({'name': bBoss, 'children': []})

for lBoss, employeeList in groups.items():

for employee in employeeList:

children.append({'name': employee, 'value': 1})

tree\_data['children'][tree\_data['children'].\_\_len\_\_() - 1]['children'].append(

{'name': lBoss, 'children': children})

children = []

return tree\_data

def drawTree(tree\_data):

Tree(opts.InitOpts(width="3500px", height="2000px")).add(

"",

[tree\_data],

collapse\_interval=2,

orient="TB",

initial\_tree\_depth=-1,

label\_opts=opts.LabelOpts(

position="top",

horizontal\_align="right",

vertical\_align="middle",

rotate=-90,

),

).set\_global\_opts(title\_opts=opts.TitleOpts(title="成员结构图")).render("成员结构图.html")

def drawWordCloud(cloud\_data: dict):

page = Page()

for bBoss, groups in cloud\_data.items():

for lBoss, employeeList in groups.items():

wordCloud = WordCloud()

wordCloud.add(series\_name=bBoss + '-' + lBoss + "邮箱内容分析", data\_pair=employeeList,

word\_size\_range=[6, 66]).set\_global\_opts(

title\_opts=opts.TitleOpts(

title=bBoss + '-' + lBoss + "邮箱内容分析", title\_textstyle\_opts=opts.TextStyleOpts(font\_size=23)

),

tooltip\_opts=opts.TooltipOpts(is\_show=True),

)

page.add(wordCloud)

page.render("邮箱内容分析.html")

if \_\_name\_\_ == '\_\_main\_\_':

data, nodes, links, categories = getData() # 从xlsx拿到数据

rank = getRank() # 拿到整个阶级顺序

drawEmailGraph(nodes, links, categories) # 画出邮件来往网络图

drawTree(getTreeData()) # 画出阶级图

drawWordCloud(getGroupEmail()) # 画出每个群体的邮件内容词云

# 由EmailGraph可见，研发部分为3个群体。群体负责人的邮箱用户名依次是1007，1059，1068。

# 群体之间都不会互相通信，同时每个群体又分为多个小群体，小群体之间同样也不会互相通信。

# 拿到3个大群体的负责人之后，可以拿到这三个人的通信记录，便可以得到每个小群体的主要负责人。

# 同样地，可以拿到每个小群体的成员，拿的时候要注意避免将3个大群体的负责人认作为小群体的成员。

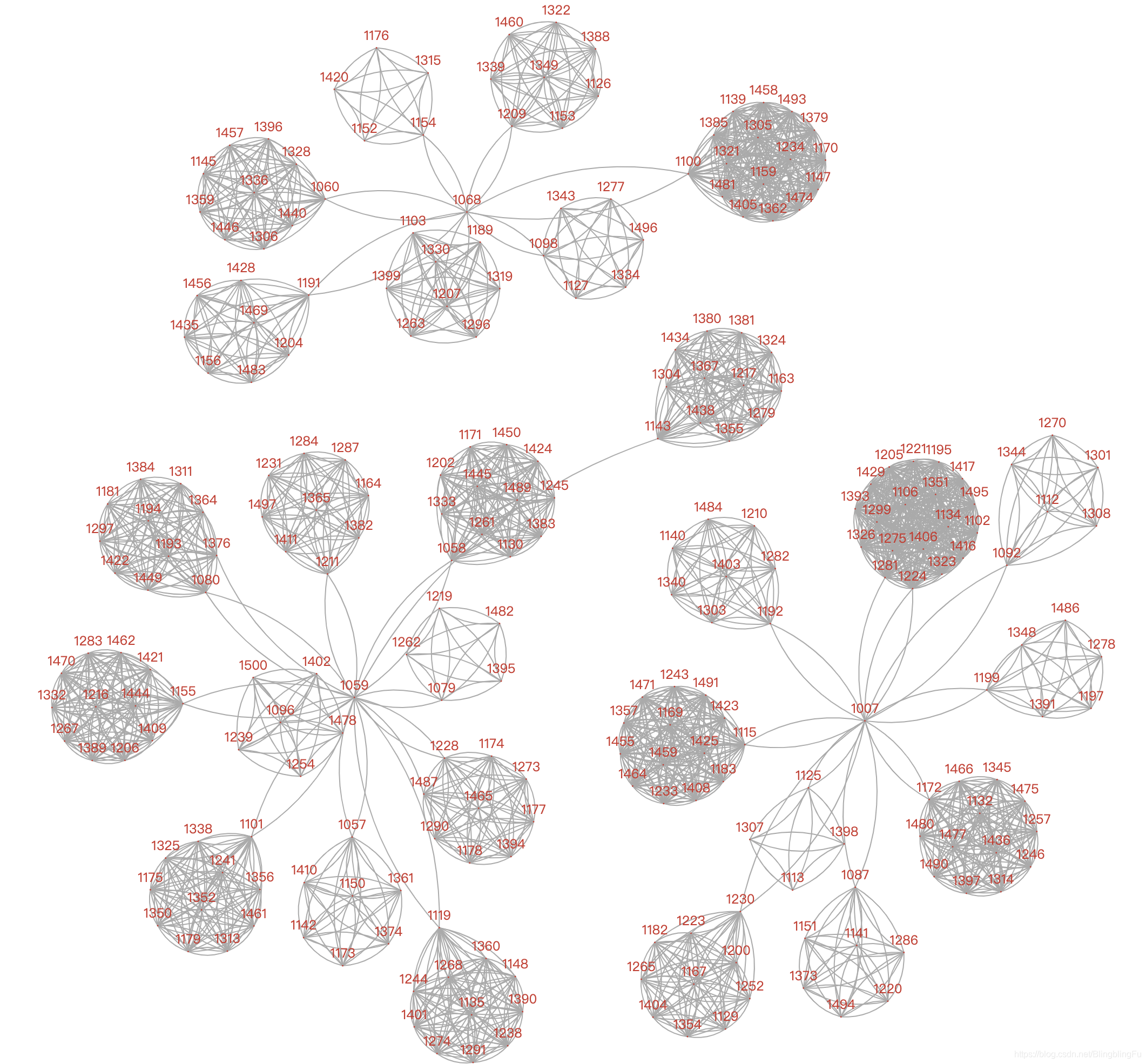
# 然后对每个大群体和小群体的邮件内容生成词云，即可获得每个群体的工作内容。

# 由Tree可以清晰地看出来部门的层次

# 由WordCloud可以看出来每个群体的主要工作内容

邮件往来网络图

邮件往来网络图如下所示，可以通过网络图可以得出主要负责人。

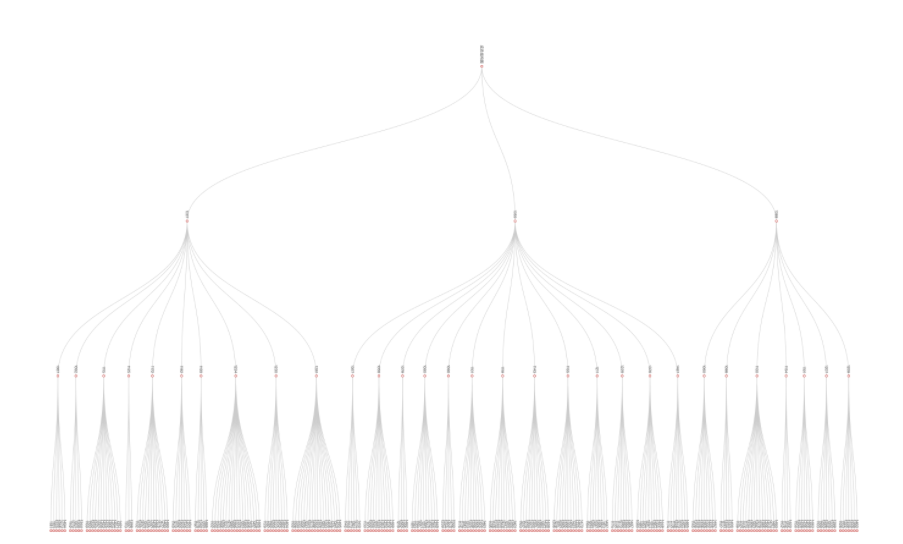
对上图进行分析：

此研发部门主要分为3个群体，负责人的邮箱用户名依次是1007，1059和1068

每个群体又分为不同的小群体，并且每个小群体中有一个负责人。

每个大群体之间不会相互通信，每个小群体之间也不会相互通信。

职位树图



对上图进行分析：

可以清晰地看出整个部门的结构层次，图中数字为邮箱用户名。