#coding=utf-8

import os

import pandas as pd

import numpy as np

from scipy import stats

from pandas import DataFrame

pd.set\_option('max\_rows', None)

#取财务报表每个单元格的首位数字

def Get\_First\_num(input):

output = 0

try:

if input != 'nan' and len(input) > 0:

input = float(input.replace(',', '')) #去除數字27,712,000,000的逗號

input = abs(input) #將負數轉成正樹

if input > 20 or input<-20: #剔除過小值,如eps

output = int(str(input)[0])

else:

pass

except Exception as e:

return

return output

#計算各表的數字的數目

def Count\_num(num\_count, input):

output = Get\_First\_num(input)

if output == 1: #如果字母為1

num\_count[0] += 1 #加一次

elif output == 2: #如果字母為2

num\_count[1] += 1 #加一次

elif output == 3: #如果字母為3

num\_count[2] += 1 #加一次

elif output == 4: #如果字母為4

num\_count[3] += 1 #加一次

elif output == 5: #如果字母為5

num\_count[4] += 1 #加一次

elif output == 6: #如果字母為6

num\_count[5] += 1 #加一次

elif output == 7: #如果字母為7

num\_count[6] += 1 #加一次

elif output == 8: #如果字母為8

num\_count[7] += 1 #加一次

elif output == 9: #如果字母為9

num\_count[8] += 1 #加一次

def compute\_num(root,file):

flow\_num=[]

#try:

path\_cash = os.path.join(root, file)

cash\_flow = pd.read\_excel(path\_cash, delimiter='\t')

cash\_flow = cash\_flow[3:]

cash\_flow = cash\_flow[:-10]

for i in range(0, len(cash\_flow.columns)):

cash\_flow.rename(columns={cash\_flow.columns[i]:str(i)},inplace=True)

for i in range(0, len(cash\_flow.columns)):

data = cash\_flow.pop(str(i))

num = [0, 0, 0, 0, 0, 0, 0, 0, 0]

for t in data:

Count\_num(num, str(t))

flow\_num.append(num)

# except Exception as e:

# #print(str(e))

return flow\_num

def Get\_actual(season,cash\_flow\_num,profitsheet\_num,Assetsheet\_num):

actual\_num=[] #三張表首字母次數加總的結果

for i in range(0, season):

list1 = []

for t in range(0,9):

list1.append(profitsheet\_num[i][t] + cash\_flow\_num[i][t]+ Assetsheet\_num[i][t])

actual\_num.append(list1)

return actual\_num

def Get\_expected(sum\_sheet):#sum\_sheet為三張表,其會計科目的首字母總數

expected\_benford=[]

for i in sum\_sheet:

benford = [0.301, 0.1761, 0.1249, 0.0969, 0.0792, 0.0669,0.058, 0.0512, 0.0458]#本福特定律中各數的機率

list1=[round(x\*i,2) for x in benford]

expected\_benford.append(list1)

return expected\_benford

def Chi\_test(season,actual\_num,expected\_benford,P):

finally\_answer=[]

for i in range(0, season):

if stats.chisquare(actual\_num[i],

f\_exp=expected\_benford[i])[1] > P:

finally\_answer.append(0)

else:

finally\_answer.append(1)

chi\_answer=[]

for i in range(0, season):

chi\_answer=stats.chisquare(actual\_num[i],expected\_benford[i])

return finally\_answer,chi\_answer

def pear\_test(season,actual\_num,expected\_benford):

pearson\_answer=[]

for i in range(0, season):

pearson\_answer=stats.pearsonr(actual\_num[i]/np.sum(actual\_num[i]),expected\_benford[i])

return pearson\_answer

def distance\_test(season,actual\_num,expected\_benford):

m=[]

d=[]

for i in range(0, season):

d=np.sum(np.square(list(set(actual\_num[i])-set(expected\_benford[i]))))

d=d\*\*0.5

for i in range(0, season):

m=map(abs,list(set(actual\_num[i])-set(expected\_benford[i])))

m=max(m)

return d,m

#=======自訂選取區================================== #

season=12#取選季節長度,若最新數據為2017第一季,則會往前抓取21季的數據

P=0.1 #90%信心水準

#=======列表存放區================================== #

profitsheet\_num=[] #存放利潤表的首字母分佈 eg:[ 9, 5, 4, 3, 2, 1, 5, 6, 7]

cash\_flow\_num=[] #存放現金流量表的首字母分佈 [ 3, 5, 6, 7, 8, 9, 2, 1, 4]

Assetsheet\_num=[] #存放資產負債表的首字母分佈 [ 2, 3, 5, 6, 8, 9, 3, 2, 3]

actual\_num=[] #存放三表加總的首字母分 [ 14, 13, 15, 16, 18, 19, 10, 9, 14]

sum\_sheet=[] #存放三者加總的首字母分佈 [128,128,128,128,128,128,128,128,128]

name=[] #上市公司的代碼

cannot\_reject=[] #不拒絕次數加總

can\_reject=[] #拒絕次數加總

company=[] #上市公司的代碼

#=======資料存放區================================== #

path ='C:/Users/42556/Desktop/Bedford-law-test-master/data'

#=======取出所有代碼================================ #

for root, dirs, files in os.walk(path):

for file in files:

name.append(file[0:6])

name\_dict = {}.fromkeys(name).keys()

name\_list =[]

for i in name\_dict:

name\_list.append(i)

print(name\_list)

#name\_list.remove('.DS\_St') #只有mac系統有 .DS\_St

for root, dir, files in os.walk(path):

for name in name\_list:

for file in files:

if file[0:6] == str(name):

if file[6:12]=='profit':

profitsheet\_num=compute\_num(root,file)

elif file[6:10]=='cash':

cash\_flow\_num=compute\_num(root,file)

elif file[6:11]=='asset':

Assetsheet\_num=compute\_num(root,file)

else:

pass

#print(name)

#print(len(cash\_flow\_num),len(profitsheet\_num),len(Assetsheet\_num))

#print(DataFrame(Assetsheet\_num,columns=['1','2','3','4','5','6','7','8','9']))

#print(cash\_flow\_num)

#print(profitsheet\_num)

#print(Assetsheet\_num)

actual\_num=Get\_actual(season,cash\_flow\_num,profitsheet\_num,Assetsheet\_num)

for i in range(0,season):

sum\_sheet.append(sum(actual\_num[i]))

expected\_benford=Get\_expected(sum\_sheet)

#print(expected\_benford)

#print(actual\_num)

finally\_answer,chi\_answer=Chi\_test(season,actual\_num,expected\_benford,P)

pearson\_answer=pear\_test(season,actual\_num,expected\_benford)

d,m=distance\_test(season,actual\_num,expected\_benford)

#print(pearson\_answer)

#print(finally\_answer)

print(chi\_answer)

print(d)

print(m)

sum\_sheet.clear()

actual\_num.clear()

expected\_benford.clear()

company.append(name)

cannot\_reject.append(finally\_answer.count(0))

can\_reject.append(finally\_answer.count(1))

dict = {"B\_cannot\_reject": cannot\_reject,

"can\_reject":can\_reject,

"A\_company": company,

}

df\_final = pd.DataFrame(dict)

print(df\_final.sort\_values("B\_cannot\_reject",ascending=False))

print(df\_final["B\_cannot\_reject"].mean())

print(df\_final["can\_reject"].mean())

writer = pd.ExcelWriter('output.xlsx')

df\_final.to\_excel(writer,'Sheet1')

writer.save()