第一、二问

**import** numpy **as** np

**import** pandas **as** pd

**import** matplotlib.pyplot **as** plt

**from** sklearn.neural\_network **import** MLPRegressor

**from** sklearn.neural\_network **import** MLPClassifier

**import** math

**from** sklearn.linear\_model **import** LinearRegression

**from** sklearn.metrics.pairwise **import** pairwise\_distances\_argmin

**from** scipy.spatial.distance **import** cdist

**from** sklearn.cluster **import** DBSCAN

**from** sklearn.cluster **import** KMeans

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

plt.rcParams['axes.unicode\_minus'] **=** False

data1 **=** pd.read\_excel('B/附件一：已结束项目任务数据.xls')

data2 **=** pd.read\_excel('处理后.xlsx')

map2 **=** data2.copy()

map2["x"]**=**(data2["经度"]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data1["任务gps 纬度"].mean()))

map2["y"]**=**(data2["纬度"]**-**data1["任务gps 纬度"].mean())**\***111

complete**=**data1[data1.任务执行情况**==**1];

no**=**data1[data1.任务执行情况**==**0];

fig **=** plt.figure()

ax1 **=** plt.axes(**projection=**'3d')

ax1.scatter3D(complete["任务gps经度"],complete["任务gps 纬度"],complete["任务标价"], **cmap=**'Blues')

ax1.scatter3D(no["任务gps经度"],no["任务gps 纬度"],no["任务标价"], **cmap=**'Blues')

plt.savefig("分布三维散点图.png")

fig1 **=** plt.figure()

ax1 **=** plt.axes()

ax1.scatter(complete["任务gps经度"],complete["任务gps 纬度"],complete["任务标价"],'yellow')

ax1.scatter(no["任务gps经度"],no["任务gps 纬度"])

plt.figure() *#初始化一张图*

plt.hist(data1["任务标价"])  *#直方图关键操作*

plt.grid(**alpha=**0.5,**linestyle=**'-.') *#网格线，更好看*

plt.xlabel('频数')

plt.ylabel('任务标价')

plt.title(r'任务标价分布')

plt.show()

plt.figure() *#初始化一张图*

plt.hist(complete["任务标价"])  *#直方图关键操作*

plt.grid(**alpha=**0.5,**linestyle=**'-.') *#网格线，更好看*

plt.xlabel('频数')

plt.ylabel('任务标价')

plt.title(r'任务标价分布')

plt.show()

plt.figure() *#初始化一张图*

plt.hist(no["任务标价"])  *#直方图关键操作*

plt.grid(**alpha=**0.5,**linestyle=**'-.') *#网格线，更好看*

plt.xlabel('频数')

plt.ylabel('任务标价')

plt.title(r'任务标价分布')

plt.show()

**print**("最小值：",min(data1["任务标价"])

     ,"最大值：",data1["任务标价"].max()

     ,"均值：",data1["任务标价"].mean())

**print**("最小值：",min(complete["任务标价"])

     ,"最大值：",complete["任务标价"].max()

     ,"均值：",complete["任务标价"].mean())

**print**("最小值：",min(no["任务标价"])

     ,"最大值：",no["任务标价"].max()

     ,"均值：",no["任务标价"].mean())

data1.describe()

mapp **=** data1.copy()

mapp["x"]**=**(data1["任务gps经度"]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data1["任务gps 纬度"].mean()))

mapp["y"]**=**(data1["任务gps 纬度"]**-**data1["任务gps 纬度"].mean())**\***111

mapno**=**no.copy()

mapcomplete**=**complete.copy()

mapno["x"]**=**(mapno["任务gps经度"]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data1["任务gps 纬度"].mean()))

mapno["y"]**=**(mapno["任务gps 纬度"]**-**data1["任务gps 纬度"].mean())**\***111

mapcomplete["x"]**=**(mapcomplete["任务gps经度"]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data1["任务gps 纬度"].mean()))

mapcomplete["y"]**=**(mapcomplete["任务gps 纬度"]**-**data1["任务gps 纬度"].mean())**\***111

fig1 **=** plt.figure(**figsize=**(10,10))

ax1 **=** plt.axes()

ax1.scatter(mapcomplete["x"],mapcomplete["y"],mapcomplete["任务标价"],'yellow')

ax1.scatter(mapno["x"],mapno["y"])

plt.xlabel("x(公里)"),plt.ylabel("y(公里)")

plt.savefig("化为公里后.png")

dist**=**cdist(mapp[["x","y"]],map2[["x","y"]],**metric=**'euclidean')

dist2**=**cdist(mapp[["x","y"]],mapp[["x","y"]],**metric=**'euclidean')

jw**=**[[113.23,23.16],[113.11,23.05],[113.75,23.04],[114.07,22.62]]

mapp["label"]**=**0

mapp["dis"]**=**0

mapp["around"]**=**0

mapp["renwu"]**=**0

**for** i **in** range(4):

    jw[i][0]**=**(jw[i][0]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data1["任务gps 纬度"].mean()))

    jw[i][1]**=**(jw[i][1]**-**data1["任务gps 纬度"].mean())**\***111

**for** index,row **in** mapp.iterrows():

    label**=**0;

    mmm**=**length**=**(mapp.iloc[index,5]**-**jw[0][0])**\*\***2**+**(mapp.iloc[index,6]**-**jw[0][1])**\*\***2

    f**=**sum(map2.iloc[dist[index]**<**1,3])

    mapp.iloc[index,9]**=**f

    ff**=**sum(dist2[index]**<**1)

    mapp.iloc[index,10]**=**ff

**for** i **in** range(4):

        length**=**(mapp.iloc[index,5]**-**jw[i][0])**\*\***2**+**(mapp.iloc[index,6]**-**jw[i][1])**\*\***2

**if**(length**<**mmm):

            mmm**=**length

            mapp.iloc[index,7]**=**i

            mapp.iloc[index,8]**=**math.sqrt(length)

mapp2**=**mapp[mapp.任务执行情况**==**1]

model **=** LinearRegression()

model.fit(map1.iloc[:,8:10], map1["任务标价"])

mapp.to\_csv("导出结果1.csv",**encoding=**'utf\_8\_sig')

mapp2.to\_csv("导出结果2.csv")

plt.figure(**figsize=**(10,10)) *#初始化一张图*

**for** k **in** range(4):

    y**=**mapp[mapp.label**==**k]

    plt.scatter(y["x"], y["y"]) *# 将同一类的点表示出来*

    plt.scatter(jw[k][0],jw[k][1],**marker=**'\*',**linewidths=**10) *# 将同一类的点表示出来*

plt.savefig("聚类结果图.png")

plt.show()

mapp.iloc[:,8:10]

mapp["ingai"]**=**0

mapp["ingai"]**=**70.511**+**0.082**\***mapp["dis"]**-**0.020**\***mapp["around"]**-**0.567**\***mapp["renwu"]

mapp["change"]**=**mapp["ingai"]**-**mapp["任务标价"]

mapp[mapp.任务执行情况**==**0].count()

fmap**=**mapp[(mapp['任务执行情况']**==**0)**&**(mapp['change']**<=**0)]

plt.figure(**figsize=**(10,10)) *#初始化一张图*

**for** k **in** range(4):

    y**=**fmap[fmap.label**==**k]

    plt.scatter(y["x"], y["y"]) *# 将同一类的点表示出来*

    plt.scatter(jw[k][0],jw[k][1],**marker=**'\*',**linewidths=**10) *# 将同一类的点表示出来*

plt.savefig("聚类结果图.png")

plt.show()

fmap[fmap['renwu']**>**0].count

amap**=**mapp[mapp.任务执行情况**==**1]

x0**=**amap[["dis","around","renwu"]]

y0**=**amap["任务标价"]

x1**=**mapp[mapp.任务执行情况**==**0][["dis","around","renwu"]]

md**=**MLPRegressor(**solver=**'lbfgs',**activation=**'identity',**hidden\_layer\_sizes=**30).fit(x0,y0)

y1**=**md.predict(x1)

mapp[mapp.任务执行情况**==**0][(y1**-**mapp[mapp.任务执行情况**==**0]["任务标价"])**>**mapp[mapp.任务执行情况**==**0]["change"]]

mapp[mapp["around"]**==**0][mapp["任务执行情况"]**==**1].count()

mapp1**=**mapp[mapp["任务执行情况"]**==**0]

mapp1["network"]**=**0

mapp1["network"]**=**md.predict(x1)

amap**=**mapp[mapp.任务执行情况**==**1]

xx0**=**mapp[["dis","around","renwu","任务标价"]]

yy0**=**mapp["任务执行情况"]

xx1**=**mapp[["dis","around","renwu","ingai"]]

xx1**=**xx1.rename(**columns=**{'ingai':'任务标价'})

md1**=**MLPClassifier(**solver=**'lbfgs',**activation=**'logistic',**hidden\_layer\_sizes=**20,**max\_iter=**10000).fit(xx0,yy0)

yy1**=**md1.predict(xx1)

sum(yy1)**/**835

sum(yy0)**/**835

mapp["maybe"]**=**yy1

mapp.to\_csv("第二题.csv",**encoding=**'utf\_8\_sig')

第三问

db**=**DBSCAN(1,**n\_jobs=**5,**min\_samples=**3)

db.fit(mapp.values[:,5:7])

labels**=**db.labels\_

mappp**=**mapp[labels**==-**1][["x","y","dis","around","renwu"]]

mappp["任务标价"]**=**70.511**+**0.082**\***mappp["dis"]**-**0.020**\***mappp["around"]**-**0.567**\***mappp["renwu"]

max(labels)

m**=** [[] **for** row **in** range(84)]

dabaodian**=**[[] **for** row **in** range(84)]

arr **=** np.zeros((84,5))

mappp2**=**pd.DataFrame(arr,**columns=**["x","y","dis","around","renwu"])

**for** index,row **in** mapp.iterrows():

**if**(labels[index]**!=-**1):

        mappp2.loc[labels[index],"x"]**+=**mapp.loc[index,"x"]

        mappp2.loc[labels[index],"y"]**+=**mapp.loc[index,"y"]

        m[labels[index]].append([mapp.loc[index,"x"],mapp.loc[index,"y"]])

        dabaodian[labels[index]].append(index)

dabao**=**[[] **for** row **in** range(84)]

t**=**0

**for** i **in** m:

    x,y**=**0,0

**for** j **in** i:

        x**+=**j[0]

        y**+=**j[1]

    x**/=**len(i)

    y**/=**len(i)

    mmm**=**(x**-**jw[0][0])**\*\***2**+**(y**-**jw[0][1])**\*\***2

**for** j **in** range(4):

        length**=**(x**-**jw[j][0])**\*\***2**+**(y**-**jw[j][1])**\*\***2

**if**(length**<**mmm):

            mmm**=**length

    f**=**0

    fff**=**[[x,y]]

    ff**=**pd.DataFrame(fff)

    ddd**=**cdist(map2[["x","y"]],ff,**metric=**'euclidean')

    f**+=**sum(map2.iloc[ddd**<**2,3])

    dabao[t].append(math.sqrt(mmm))

    dabao[t].append(f)

    dabao[t].append(len(i))

    dabao[t].append(70.511**+**0.082**\***dabao[t][0]**-**0.020**\***dabao[t][1]**-**0.567**\***dabao[t][2])

    t**+=**1

md1**=**MLPClassifier(**solver=**'lbfgs',**activation=**'logistic',**hidden\_layer\_sizes=**10,**max\_iter=**10000).fit(xx0,yy0)

yy2**=**md1.predict(dabao)

yy3**=**md1.predict(mappp[["dis","around","renwu","任务标价"]])

over**=**0

**for** i **in** range(len(yy2)):

    over**+=**yy2[i]**\***dabao[i][2]

over**+=**sum(yy3)

over**/**835

mappp["是否成功"]**=**yy3

mappp.to\_csv("第三问（未被打包）.csv",**encoding=**'utf\_8\_sig')

mappp2**=**pd.DataFrame(dabao,**columns=**["dis","around","renwu","任务标价"])

mappp2["是否成功"]**=**yy2

mappp2.to\_csv("第三问（打包）.csv",**encoding=**'utf\_8\_sig')

dabaodian

第四问

data3 **=** pd.read\_excel('B/附件三：新项目任务数据.xls')

last **=** data3.copy()

last["x"]**=**(data3["任务GPS经度"]**-**data1["任务gps经度"].mean())**\***111**\***math.cos(math.radians(data3["任务GPS纬度"].mean()))

last["y"]**=**(data3["任务GPS纬度"]**-**data1["任务gps 纬度"].mean())**\***111

plt.figure(**figsize=**(10,10)) *#初始化一张图*

last["label"]**=**0

last["dis"]**=**0

last["around"]**=**0

last["renwu"]**=**0

distt**=**cdist(last[["x","y"]],map2[["x","y"]],**metric=**'euclidean')

**for** index,row **in** last.iterrows():

    label**=**0;

    mmm**=**length**=**(last.iloc[index,3]**-**jw[0][0])**\*\***2**+**(last.iloc[index,4]**-**jw[0][1])**\*\***2

    f**=**sum(map2.iloc[distt[index]**<**1,3])

    last.loc[index,"around"]**=**f

    ff**=**sum(distt[index]**<**1)

    last.loc[index,"renwu"]**=**ff

**for** i **in** range(4):

        length**=**(last.loc[index,"x"]**-**jw[i][0])**\*\***2**+**(last.loc[index,"y"]**-**jw[i][1])**\*\***2

**if**(length**<**mmm):

            mmm**=**length

            last.loc[index,"label"]**=**i

            last.loc[index,"dis"]**=**math.sqrt(length)

plt.figure(**figsize=**(10,10)) *#初始化一张图*

**for** k **in** range(4):

    y**=**last[last.label**==**k]

    plt.scatter(y["x"], y["y"]) *# 将同一类的点表示出来*

    plt.scatter(jw[k][0],jw[k][1],**marker=**'\*',**linewidths=**10) *# 将同一类的点表示出来*

plt.savefig("第四题聚类效果图")

db1**=**DBSCAN(0.35,**n\_jobs=**5,**min\_samples=**3)

db1.fit(last.values[:,3:5])

labels**=**db1.labels\_

lele**=**max(labels)**+**1

m**=** [[] **for** row **in** range(lele)]

dabaodian**=**[[] **for** row **in** range(lele)]

arr **=** np.zeros((lele,5))

lastt2**=**pd.DataFrame(arr,**columns=**["x","y","dis","around","renwu"])

**for** index,row **in** last.iterrows():

**if**(labels[index]**!=-**1):

        lastt2.loc[labels[index],"x"]**+=**last.loc[index,"x"]

        lastt2.loc[labels[index],"y"]**+=**last.loc[index,"y"]

        m[labels[index]].append([last.loc[index,"x"],last.loc[index,"y"]])

        dabaodian[labels[index]].append(index)

dabao**=**[[] **for** row **in** range(lele)]

t**=**0

**for** i **in** m:

    x,y**=**0,0

**for** j **in** i:

        x**+=**j[0]

        y**+=**j[1]

    x**/=**len(i)

    y**/=**len(i)

    mmm**=**(x**-**jw[0][0])**\*\***2**+**(y**-**jw[0][1])**\*\***2

**for** j **in** range(4):

        length**=**(x**-**jw[j][0])**\*\***2**+**(y**-**jw[j][1])**\*\***2

**if**(length**<**mmm):

            mmm**=**length

    f**=**0

    fff**=**[[x,y]]

    ff**=**pd.DataFrame(fff)

    ddd**=**cdist(map2[["x","y"]],ff,**metric=**'euclidean')

    f**+=**sum(map2.iloc[ddd**<**2,3])

    dabao[t].append(math.sqrt(mmm))

    dabao[t].append(f)

    dabao[t].append(len(i))

    dabao[t].append(70.511**+**0.082**\***dabao[t][0]**-**0.020**\***dabao[t][1]**-**0.567**\***dabao[t][2])

    t**+=**1

sum(labels**==-**1)

last1**=**last.loc[labels**==-**1]

last1["任务标价"]**=**70.511**+**0.082**\***last1["dis"]**-**0.020**\***last1["around"]**-**0.567**\***last1["renwu"]

dabao

md1**=**MLPClassifier(**solver=**'lbfgs',**activation=**'logistic',**hidden\_layer\_sizes=**20,**max\_iter=**10000).fit(xx0,yy0)

yy2**=**md1.predict(dabao)

yy3**=**md1.predict(last1[["dis","around","renwu","任务标价"]])

over**=**0

**for** i **in** range(len(yy2)):

    over**+=**yy2[i]**\***dabao[i][2]

over**+=**sum(yy3)

over**/**len(last)

last1["是否成功"]**=**yy3

mappp2**=**pd.DataFrame(dabao,**columns=**["dis","around","renwu","任务标价"])

mappp2["是否成功"]**=**yy2

last1.to\_csv("第四问（未被打包）.csv",**encoding=**'utf\_8\_sig')

mappp2.to\_csv("第四问（打包）.csv",**encoding=**'utf\_8\_sig')

result **=** open('打包组类(第四问).xls', 'w', **encoding=**'utf\_8')

**for** m **in** range(len(dabaodian)):

**for** n **in** range(len(dabaodian[m])):

        result.write(str(dabaodian[m][n]))

        result.write('\t')

    result.write('\n')

result.close()