|  |  |
| --- | --- |
| python程序 | 数据预处理 |
| #对表单1进行缺失值处理  import pandas as pd  dataset1 = pd.read\_excel("附件.xlsx",sheet\_name = "合并")  data = pd.DataFrame(dataset1)  data = data.fillna(0)  ych = list()  for i in data.index:      b = 0      for j in range(6,19):          b += data.iloc[i,j]      if b >= 105 or b <= 85:          ych.append(i)  data = data.drop(ych) | |

|  |  |
| --- | --- |
| 问题1 python程序 | 绘图、分类模型 |
| #绘制箱型图  plt.figure(figsize=(40,20))  a = 0  for i in range(6,18):      xlab = data\_lx1.columns[i]      hxcf\_x = pd.DataFrame({"风化高钾":data\_lx1.iloc[:,i],"无风化高钾":data\_lx2.iloc[:,i],"风化铅钡":data\_lx3.iloc[:,i],"无风化铅钡":data\_lx4.iloc[:,i]})      plt.subplot(3,4,a+1)      a += 1      #plt.subplots(dpi=1080,facecolor='w')# 设置画布大小，分辨率，和底色      hxcf\_x.boxplot()      plt.xlabel(xlab,fontsize = 16) # 我们设置横纵坐标的标题。      plt.tick\_params(labelsize = 16)  #绘制分布图  plt.figure(figsize=(30,10))  for n,i in enumerate(["二氧化硅(SiO2)","氧化钠(Na2O)","氧化钾(K2O)","氧化钙(CaO)","氧化镁(MgO)","氧化铝(Al2O3)","氧化铁(Fe2O3)","氧化铜(CuO)","氧化铅(PbO)","氧化钡(BaO)","五氧化二磷(P2O5)","氧化锶(SrO)","氧化锡(SnO2)","二氧化硫(SO2)"]):      plt.subplot(2,7,n+1)  #     plt.title(i)      sns.distplot(data\_lx1[i])      plt.xlabel(i, fontsize = 14)      plt.ylabel('')      plt.tick\_params(labelsize = 14)  #建立逻辑回归模型  from sklearn.linear\_model import LogisticRegression  lr = LogisticRegression ()  lr.fit(X\_train,y\_train)  lr.predict(X\_test)  #输出模型系数  print('训练模型自变量参数为：',lr.coef\_)  print('训练模型截距为：',lr.intercept\_)  #模型评价  print('模型的平均正确率为：',lr.score(X\_test,y\_test))  #预测精度  from sklearn.metrics import accuracy\_score  y\_predict=lr.predict(X\_test)  accuracy\_score(y\_test,y\_predict)  #逐步回归  from sklearn.linear\_model import LinearRegression as LR  from sklearn.metrics import r2\_score  model\_sio2=LR().fit(data\_2\_x1,data\_2\_si02)  perfomance\_reg(model\_sio2,data\_2\_x1,data\_2\_si02,'训练集')  data1\_sio2 = model\_sio2.predict(data\_1\_x1)  data1\_sio2 = pd.DataFrame(data1\_sio2,columns=['二氧化硅(SiO2)'])  data\_1\_x2 = pd.concat([data\_1\_x1,data1\_sio2],axis=1) | |

|  |  |
| --- | --- |
| 问题2 python程序 | 绘图、分类模型、聚类模型 |
| #建立随机森林分类模型  model\_rf = RandomForestClassifier(random\_state=9)  model\_rf.fit(X\_train,y\_train)  #模型评价  expected = y\_test  predicted = model\_rf.predict(X\_test)  print(metrics.classification\_report(expected,predicted))  print(metrics.confusion\_matrix(expected,predicted))  #绘制聚类手肘图  d=[]  for i in range(1,11):    #k取值1~11，做kmeans聚类，看不同k值对应的簇内误差平方和      km=KMeans(n\_clusters=i,init='k-means++',n\_init=10,max\_iter=300,random\_state=0)      km.fit(data\_gj\_jl)      d.append(km.inertia\_)  #inertia簇内误差平方和  #绘图参数设置  fig = plt.figure(dpi=300,figsize=(5,3)) #设置分辨率，画布大小  ax = fig.add\_subplot(111)  #设置背景色  ax.patch.set\_facecolor('white')#设置画布外颜色  fig.patch.set\_facecolor('white')#设置画布内颜色  #设置画框的颜色  ax.spines['bottom'].set\_color('black')  ax.spines['top'].set\_color('black')  ax.spines['left'].set\_color('black')  ax.spines['right'].set\_color('black')  #作图  plt.plot(range(1,11),d,marker='o',color = 'black')  plt.xlabel('number of clusters',color='black')  plt.ylabel('distortions',color='black')  #设置横纵坐标轴的颜色  plt.tick\_params(axis='x',colors='black')  plt.tick\_params(axis='y',colors='black')  #聚类  Kmean1= KMeans(n\_clusters=3,random\_state=0)  Kmean1.fit(data\_qb\_jl)  #获取聚类结果  qb\_yc = Kmean1.labels\_ | |

|  |  |
| --- | --- |
| 问题4 python程序 | 绘图、相关性分析模型 |
| corr\_gj = data\_gj.corr(method="pearson")  corr\_qb = data\_qb.corr(method="pearson")  plt.subplots(figsize=(9,9),dpi=1080,facecolor='w')# 设置画布大小，分辨率，和底色  p1 = sns.heatmap(corr\_gj ,annot=True, vmax=1, square=True, cmap="Blues", fmt='.2g')  p2 = sns.heatmap(corr\_qb ,annot=True, vmax=1, square=True, cmap="Blues", fmt='.2g') | |