**离群点分析与异常检测**

3220190975-宋迎新

实验目的：从给定数据集（Anomaly Detection Meta-Analysis Benchmarks提供的benchmark）中找出异常点。

代码中需要调用的库

|  |
| --- |
| import pandas as pd  from pyod.models.iforest import IForest  from pyod.models.knn import KNN  from pyod.models.lof import LOF  import os  import numpy as np  from pyod.models.cblof import CBLOF  from sklearn.metrics import average\_precision\_score  from pyod.utils.data import get\_outliers\_inliers  from mpl\_toolkits.mplot3d import Axes3D  import matplotlib.pyplot as plt  from sklearn.metrics import precision\_recall\_curve |

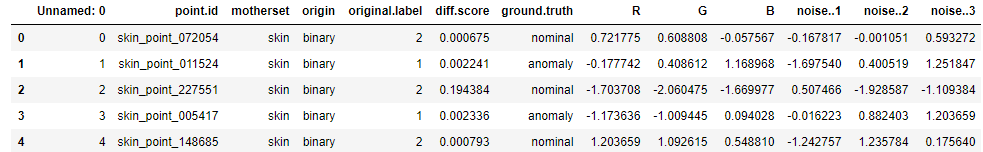
# 1.数据集1：skin\_benchmarks分析

一共包含1740个数据集，数据集有9个特征（属性）groundtruth 为标注标签。预测输入为R、P、G三个特征。

## 1.1数据集检测

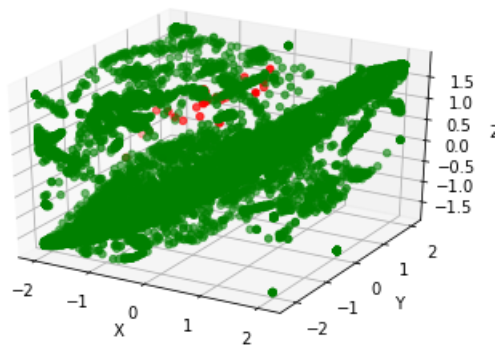
考虑到数据集较大，首先从整体数据集中取一部分进行试验，实验所用工具为PyOD，主要用到其中的KNN异常检测算法，进行数据离群点分析和异常检测。

|  |
| --- |
| 下面的代码功能为读取数据合并子数据集  path = './skin/benchmarks/'  files = os.listdir(path)[350:355]  train\_csv = list(files)  data\_list = []  for fileitem in train\_csv:  tmp = pd.read\_csv(path + fileitem)  data\_list.append(tmp)  dataset = pd.concat(data\_list,ignore\_index = False)  dataset\_new=dataset.drop\_duplicates()  dataset\_new.to\_csv('./skin/prodata/gloable\_test.csv')  可视化功能  def draw\_plt(p11,p12,p13,p21,p22,p23):  ax = plt.subplot(111, projection='3d')  ax.scatter(p11, p12, p13, c='r') # 绘制数据点  ax.scatter(p21, p22, p23, c='g') # 绘制数据点  ax.set\_zlabel('Z') # 坐标轴  ax.set\_ylabel('Y')  ax.set\_xlabel('X')  plt.show()  df = pd.read\_csv("data/data/skin/prodata/gloable\_test.csv")  print("数据数量：%d" %len(df))  df.head() |



|  |
| --- |
| 生成训练集  for i in range(len(df)):  if df['ground.truth'][i]=="nominal":  label+=[0]  if df['ground.truth'][i]=='anomaly':  label+=[1]  count+=1 |

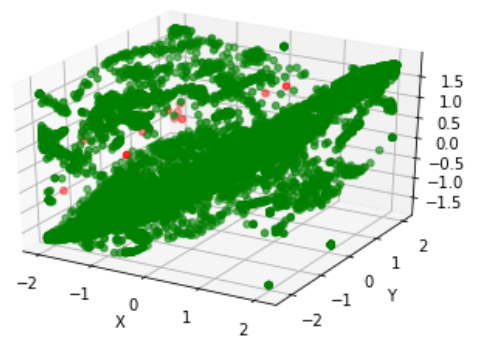
利用数据可视化，画出三维散点图（包含正常点和离群点）



KNN算法是基于邻近性的异常检测算法。这里我们选择PyOD的三个KNN探测器中的最大KNN算法计算离群值。

|  |
| --- |
| clf\_name='KNN'  clf=KNN(contamination=outliers\_fraction)  clf.fit(X)  y\_pred=clf.labels\_  n\_out=np.count\_nonzero(y\_pred)  n\_normal=len(y\_pred)-n\_out  sum\_all=0  #sum\_out=0  TP=0  FP=0  FN=0  for i in range(len(df)):  if(Y[i]==y\_pred[i]):  sum\_all+=1  if Y[i]==0:  TP+=1  if(Y[i]==1):  if(y\_pred[i]==0):  FP+=1  if(Y[i]==0):  if(y\_pred[i]==1):  FN+=1  precision=TP/(TP+FP)  recall=TP/(TP+FN)  F1=2\*precision\*recall/(precision+recall)  dfx = df  dfx['outlier'] = y\_pred.tolist()  OX1 = dfx['R'][dfx['outlier'] == 1].values.reshape(-1,1)  OX2 = dfx['G'][dfx['outlier'] == 1].values.reshape(-1,1)  OX3 = dfx['B'][dfx['outlier'] == 1].values.reshape(-1,1)  IX1 = dfx['R'][dfx['outlier'] == 0].values.reshape(-1,1)  IX2 = dfx['G'][dfx['outlier'] == 0].values.reshape(-1,1)  IX3 = dfx['B'][dfx['outlier'] == 0].values.reshape(-1,1)  draw\_plt(OX1,OX2,OX3,IX1,IX2,IX3) |

结果为异常值的数量28，正常值29972



识别准确率：0.9984

Precision：0.9992

Recall：0.9993

F1：0.9992

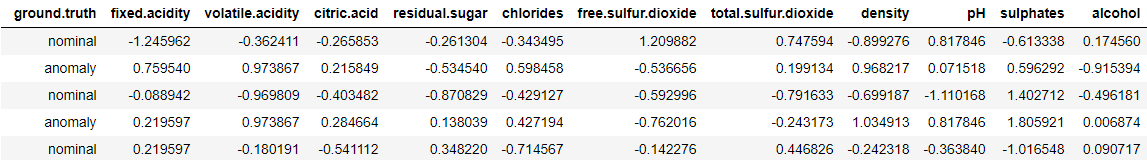
# 2.数据集2：wine\_benchmarks分析

一共包含1680个数据集，数据集有17个特征（属性）groundtruth 为标注标签。选择其中的12个属性为预测输入。

## 2.1数据集检测

考虑到数据集较大，首先从整体数据集中取一部分进行试验，实验所用工具为PyOD，主要用到其中的KNN异常检测算法，进行数据离群点分析和异常检测。

|  |
| --- |
| 下面的代码功能为读取数据合并子数据集  path = './wine/benchmarks/'  files = os.listdir(path)[:3]  train\_csv = list(files)  data\_list = []  i=1  for fileitem in train\_csv:  tmp = pd.read\_csv(path + fileitem)  tmp\_new=tmp[['ground.truth','fixed.acidity','volatile.acidity','citric.acid','residual.sugar','chlorides','free.sulfur.dioxide','total.sulfur.dioxide','density','pH','sulphates','alcohol']]  tmp\_new.drop\_duplicates()  if i==1:  tmp\_new.to\_csv("data/data/wine/prodata/gloable\_test.csv",mode='a',index=False)  i=0  else:  tmp\_new.to\_csv("data/data/wine/prodata/gloable\_test.csv",mode='a',index=False, header=False)  data\_list.append(tmp)  dataset = pd.concat(data\_list,ignore\_index = False)  dataset\_new=dataset.drop\_duplicates()  dataset\_new.to\_csv('./skin/prodata/gloable\_test.csv')  展示数据  df = pd.read\_csv("data/data/wine/prodata/gloable\_test.csv")  print("数据数量：%d" %len(df))  df.head() |



|  |
| --- |
| 生成训练集  for i in range(len(df)):  if df['ground.truth'][i]=="nominal":  label+=[0]  if df['ground.truth'][i]=='anomaly':  label+=[1]  count+=1 |

KNN算法是基于邻近性的异常检测算法。这里我们选择PyOD的三个KNN探测器中的最大KNN算法计算离群值。

|  |
| --- |
| clf\_name='KNN'  clf=KNN(contamination=outliers\_fraction)  clf.fit(X)  y\_pred=clf.labels\_  n\_out=np.count\_nonzero(y\_pred)  n\_normal=len(y\_pred)-n\_out  sum\_all=0  #sum\_out=0  TP=0  FP=0  FN=0  for i in range(len(df)):  if(Y[i]==y\_pred[i]):  sum\_all+=1  if Y[i]==0:  TP+=1  if(Y[i]==1):  if(y\_pred[i]==0):  FP+=1  if(Y[i]==0):  if(y\_pred[i]==1):  FN+=1  precision=TP/(TP+FP)  recall=TP/(TP+FN)  F1=2\*precision\*recall/(precision+recall) |

结果为异常值的数量3975，正常值7134

识别准确率：0.5649

Precision：0.6612

Recall：0.6612

F1：0.6612