**基于xgboost的鸢尾花识别**

实验手册

**OracleWDP·华育兴业**

# 实验目录

## 基于xgboost的鸢尾花识别

# 实验内容

## 基于xgboost的鸢尾花识别

知识点

1. xgboost的参数设置
2. xgboost的重要参数选择
3. xgboost多分类模型目标函数的设置
4. 格搜索的使用

实验目的

1. 学习使用xgboost建立xgboost分类模型
2. 学习xgboost参数的解读
3. 学会使用格搜索选择模型参数

实验环境

1. Oracle Linux 7.4
2. Python 3

实验步骤

1. 打开Jupyter，并新建python工程
   * + 1. 桌面空白处右键，点击Konsole打开一个终端
       2. 切换至/root/experiment/jupyter目录

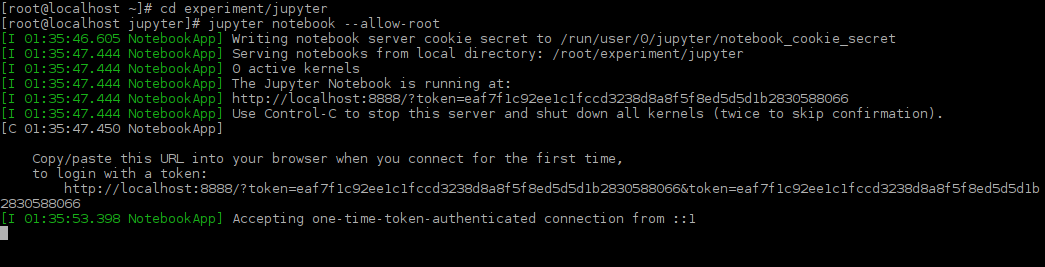
[Command 001]:

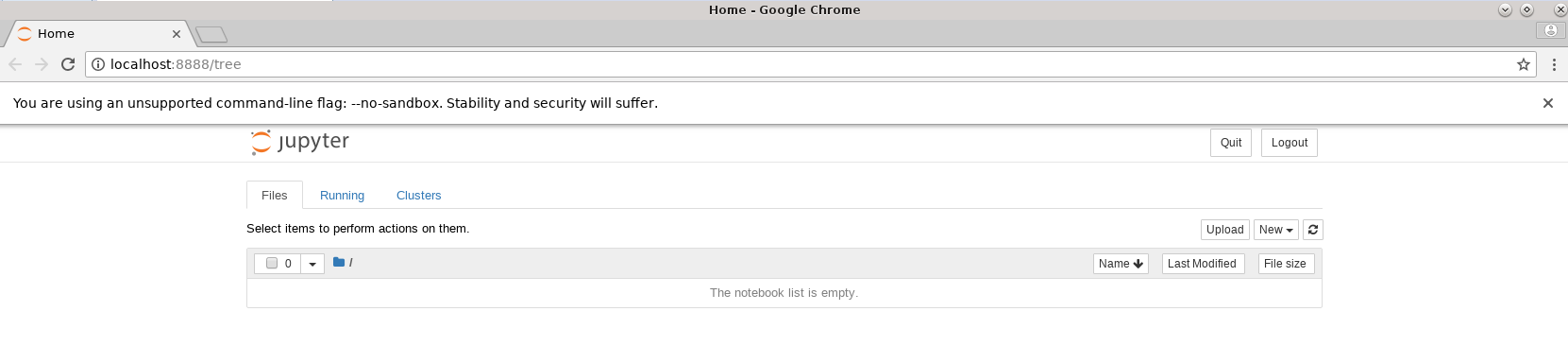
*cd /root/experiment/jupyter*

* + - 1. 启动Jupyter，root用户下运行需加’--allow-root’

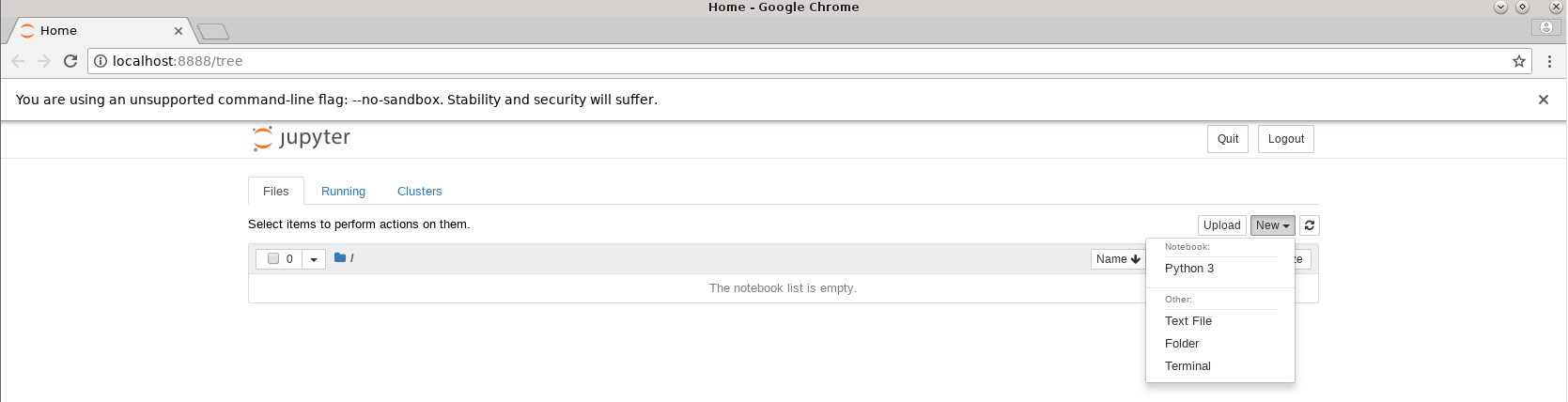
[Command 002]:

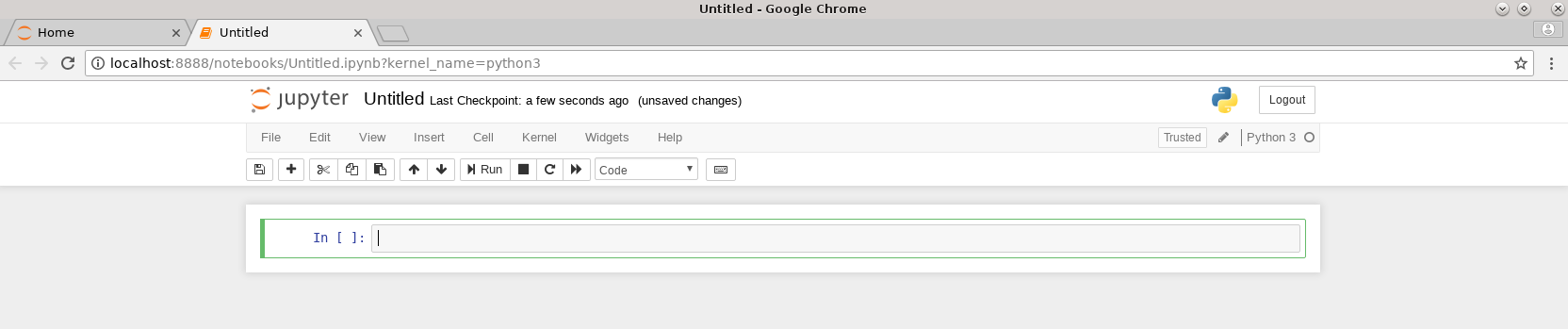
*jupyter notebook --allow-root*



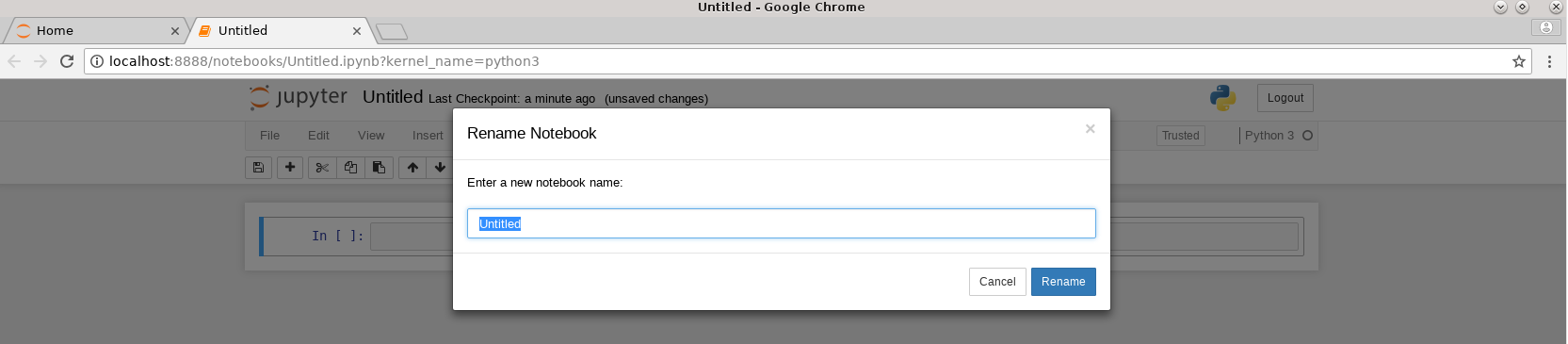


* + - 1. 依次点击右上角的 New，Python 3新建python工程





* + - 1. 点击Untitled，在弹出框中修改标题名，点击Rename确认



1. 读取数据
   * + 1. 输入代码后，使用shift+enter执行，下同。
       2. 数据说明：

鸢尾花数据集是一个非常著名的数据集，数据包括三种鸢尾花花萼、花瓣的长宽（厘米计量），每种鸢尾花包括50个样本，共150个样本。

字段解释：

sepal\_l: 花萼长度，连续变量，厘米计量

sepal\_w: 花萼宽度，连续变量，厘米计量

petal\_l：花瓣长度，连续变量，厘米计量

petal\_w：花瓣宽度，连续变量，厘米计量

classes：鸢尾花种类，离散变量，（0-2）

* + - 1. [Code 001]:

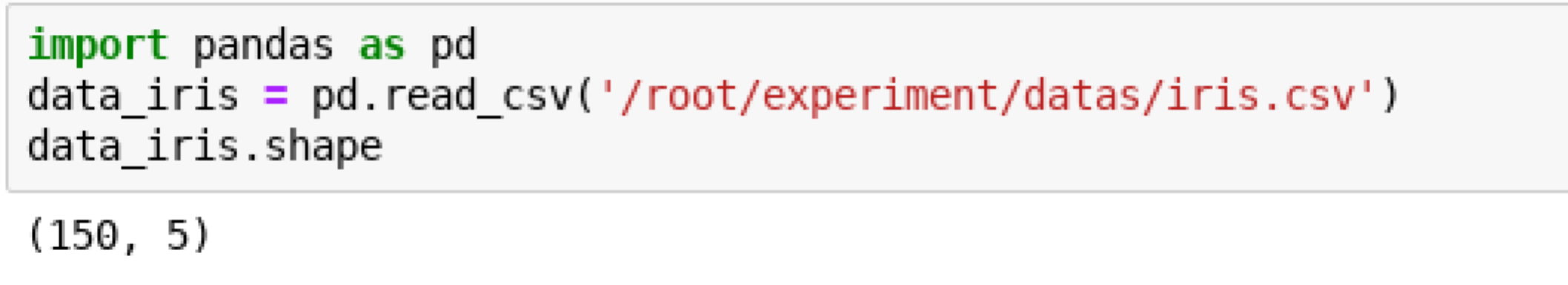
*# 使用pandas读取csv文件*

*import pandas as pd*

*data\_iris = pd.read\_csv('/root/experiment/datas/* *iris.csv'')*

*# 查看ata\_iris的维度*

*data\_iris.shape*

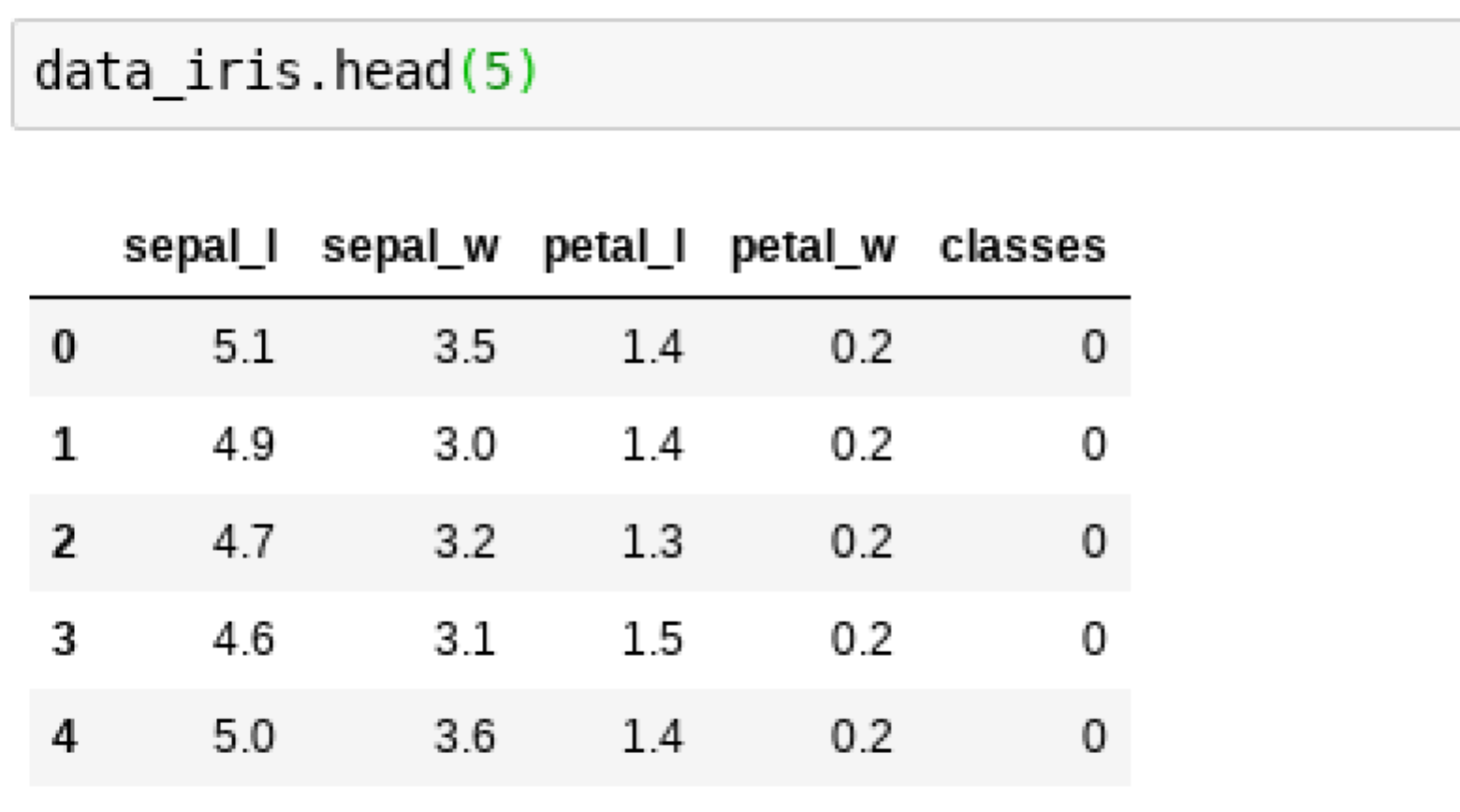
**

1. 描述性分析与可视化分析

[Code 002]:

*# 查看data\_iris的前五项*

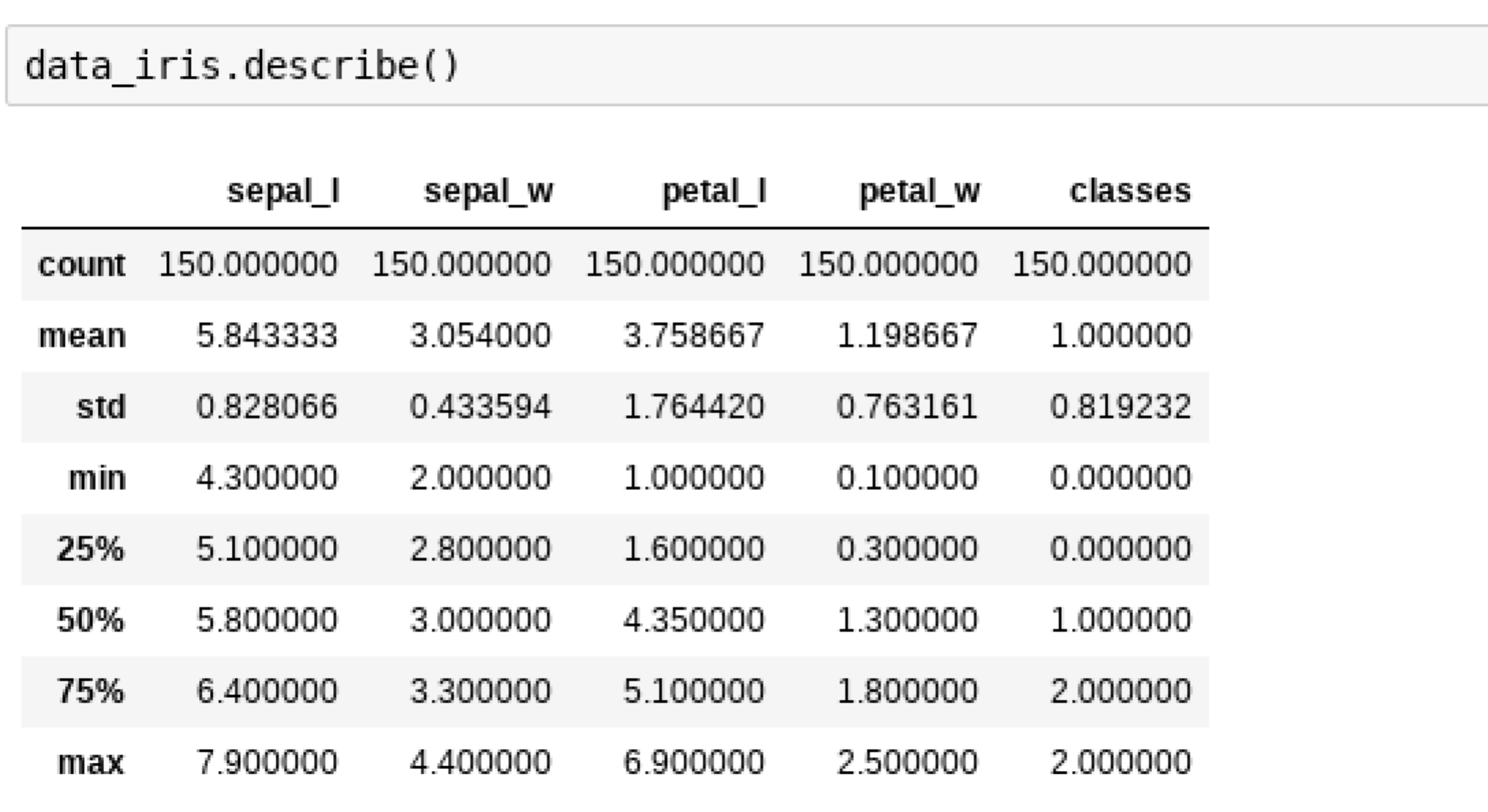
*data\_iris.head(5)*

**

[Code 003]:

*# 查看data\_iris中连续变量的统计描述*

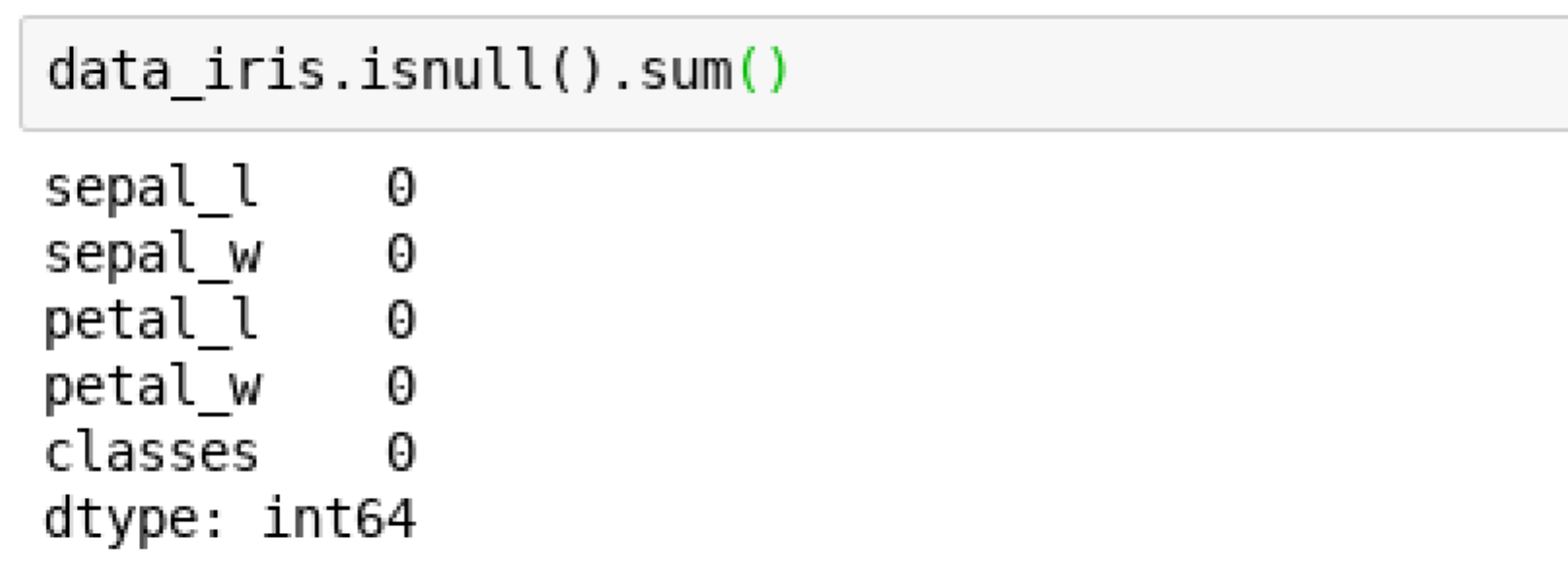
*data\_iris.describe()*

**

[Code 004]:

*# 查看data\_iris中各字段的缺失值*

*data\_iris.isnull().sum()*

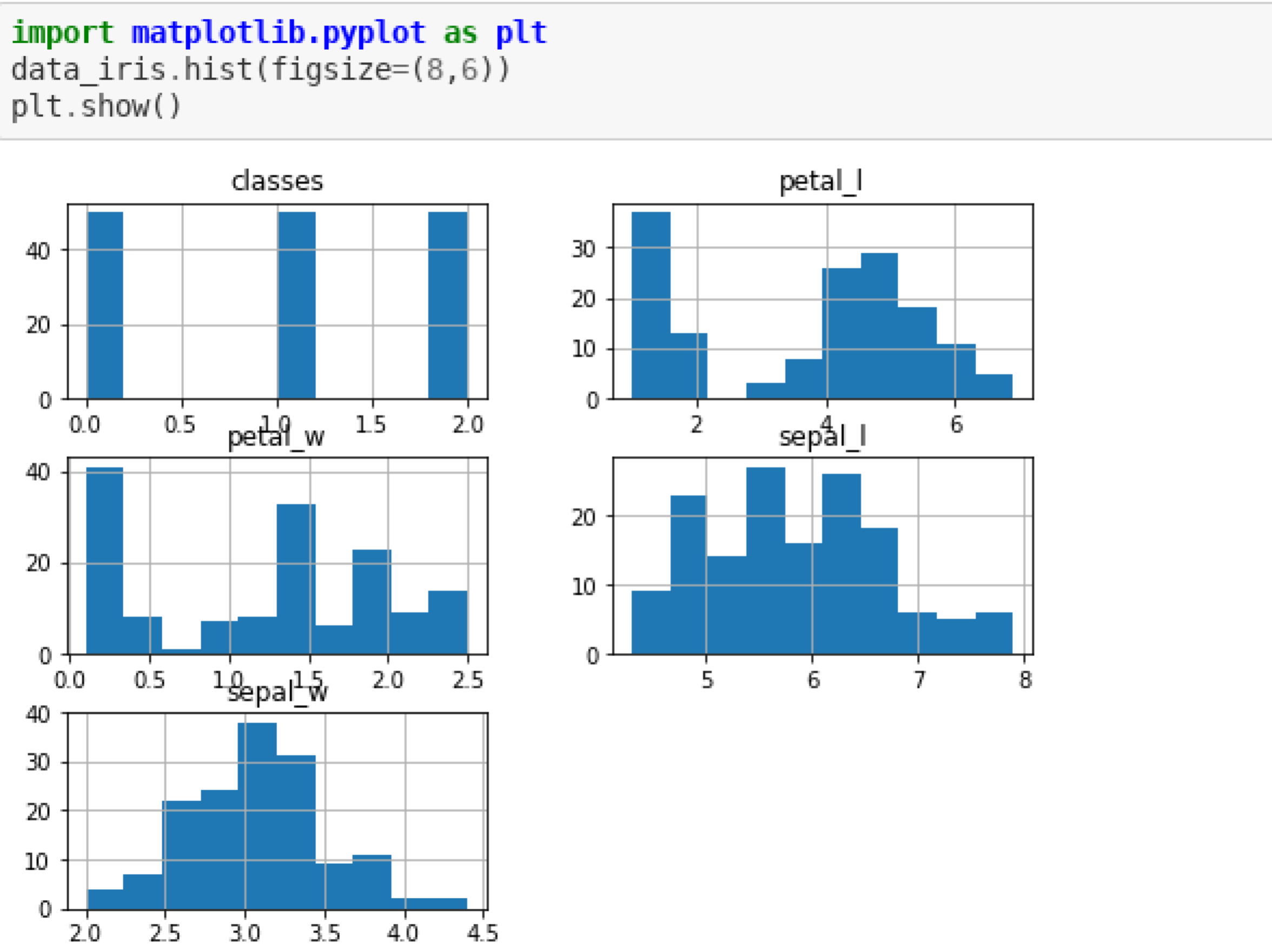


[Code 005]:

*# 使用matplotlib库进行可视化分析*

*import matplotlib.pyplot as plt*

*data\_iris.hist(figsize=(8,6))*

*plt.show()*

1. 数据处理

[Code 006]:

*# 为原始数据增加截距项*

*X = data\_iris.iloc[:,:-1]*

*y = data\_iris.iloc[:,-1]*

*from sklearn.model\_selection import train\_test\_split*

*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=5)*

1. 建立模型

[Code 007]:

*#设置xgboost参数，并用格搜索方法建立进行模型参数确定*

*＃存在警告提示，是工具自身问题，先忽略*

*import xgboost as xgb*

*params = {*

*'learning\_rate':0.2,*

*'n\_estimators':2,*

*'max\_depth':2,*

*'min\_child\_weight':1,*

*'gamma':0,*

*'subsample':0.8,*

*'colsample\_bytree':0.8,*

*'silent':True,*

*'objective':'multi:softmax',*

*}*

*model = xgb.XGBClassifier(\*\*params)*

*from sklearn.model\_selection import GridSearchCV*

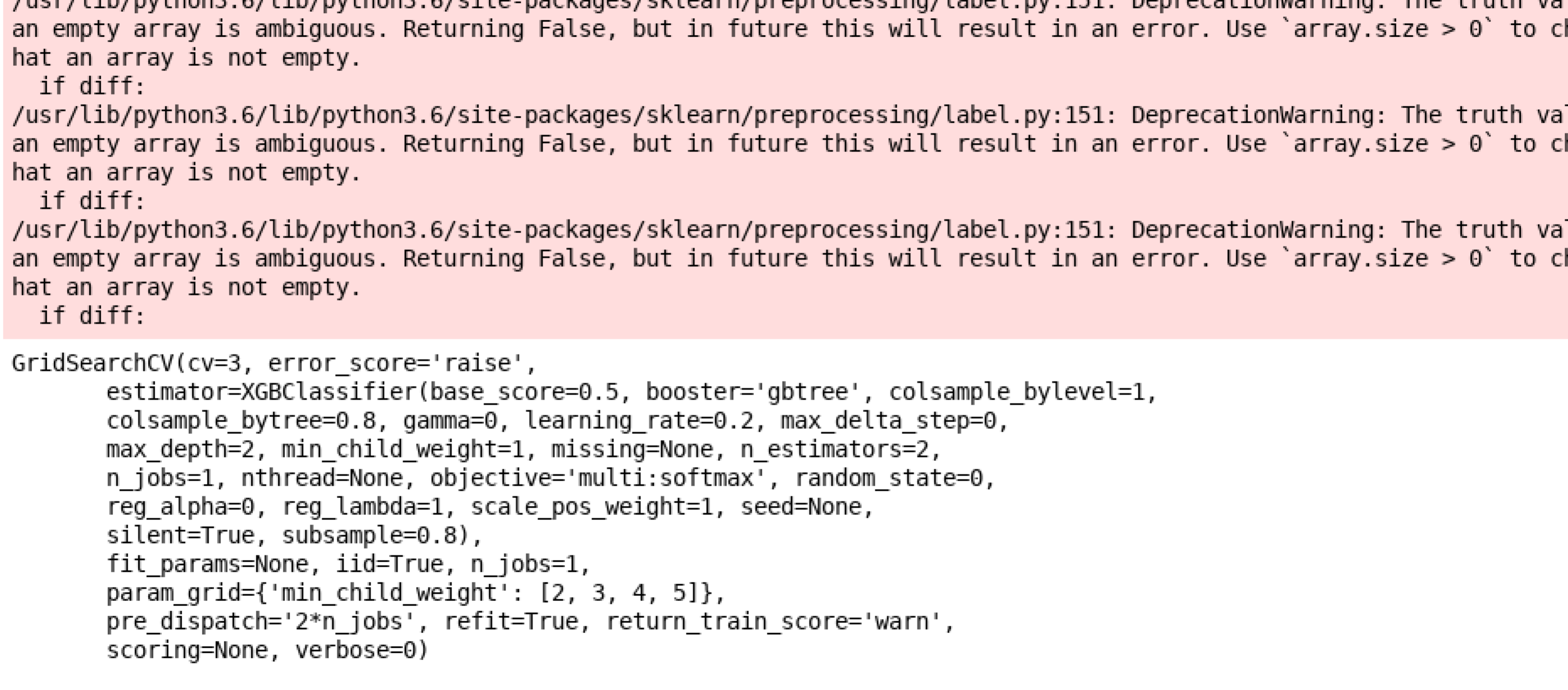
*grid\_params = {*

*'min\_child\_weight':[x for x in range(2,6)]*

*}*

*grid = GridSearchCV(model,grid\_params,cv=3)*

*grid.fit(X\_train,y\_train)*

**

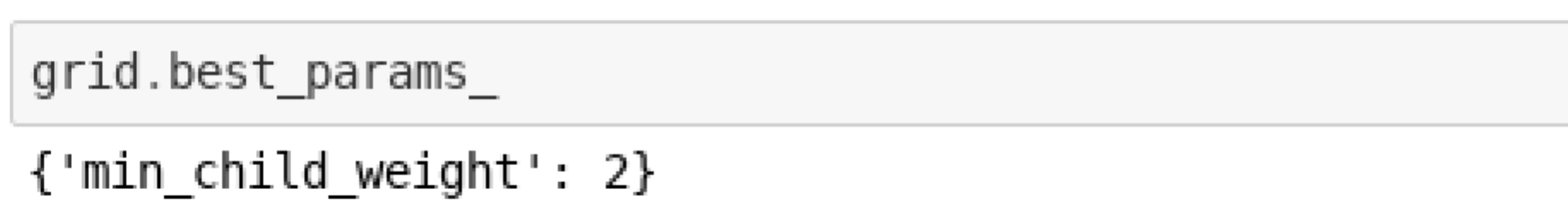
*#注意会有一些warning信息，这是xgboost包的一些问题，当前版本还未修复*

1. 格搜索结果

[Code 008]:

*# 使用predict对x进行预测*

*grid.best\_params\_*

**

[Code 009]:

*#格搜索结果*

*print("Accuracy:{0:.1f}%".format(100\*grid.best\_score\_))* **

[Code 010]:

*# 重新设置参数并训练模型*

*params = {*

*'learning\_rate':0.2,*

*'n\_estimators':3,*

*'max\_depth':2,*

*'min\_child\_weight':2,*

*'gamma':0,*

*'subsample':0.8,*

*'colsample\_bytree':0.8,*

*'silent':True,*

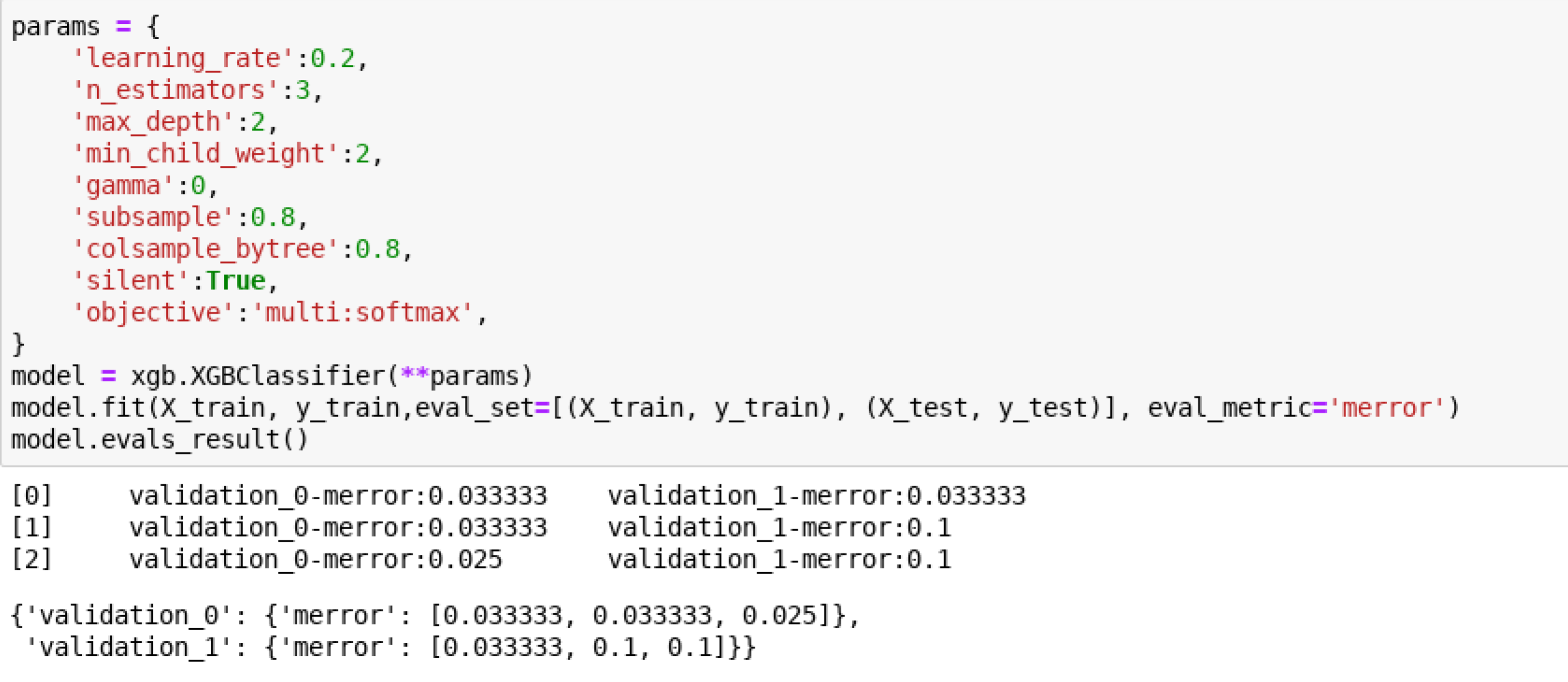
*'objective':'multi:softmax',*

*}*

*model = xgb.XGBClassifier(\*\*params)*

*model.fit(X\_train, y\_train,eval\_set=[(X\_train, y\_train), (X\_test, y\_test)], eval\_metric='merror')*

*model.evals\_result()*

**

1. 实验结论
   * + 1. 存在waring是xgboost包问题，暂时忽略。
       2. 格搜索结果显示最下叶子权重为2时，模型效果最好。
       3. 还可以同时设置其他参数进行格搜索确定最优参数。
       4. 最终模型测试错误率为0.1。