## 模型训练阶段代码说明（完整代码在CNN\_better.ipython文件中）



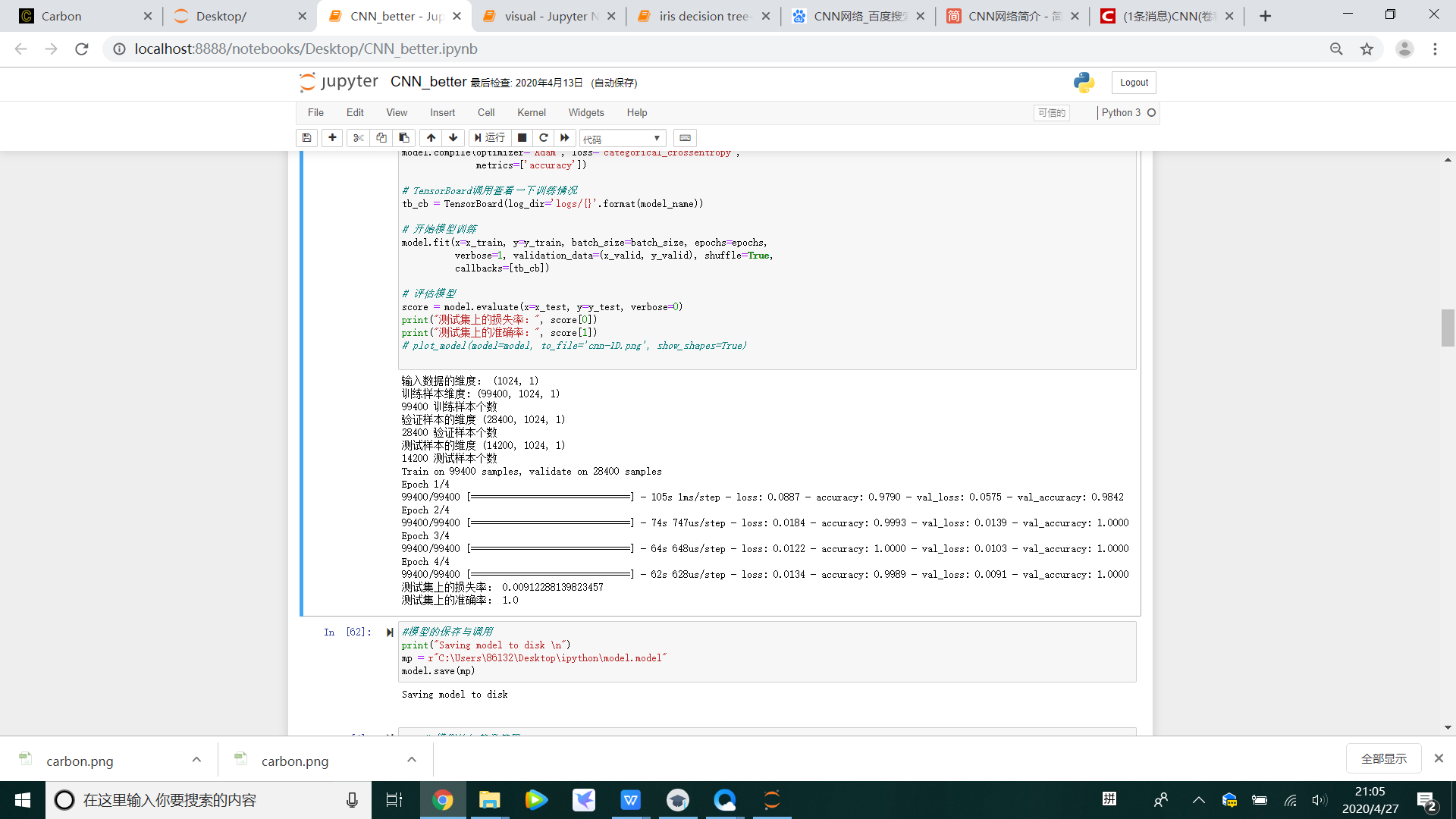
我们最终采用的训练方式是基于CNN（卷积神经网络）的机器学习算法。

利用keras自己搭建一个一维的卷积网络，网络中有20个卷积核，卷积步长为8。使用Relu函数作为激活函数，池化方式选择最大值池化。

CNN本质上就是使用机器学习来不断优化网络，以达到输出较好预测值的目的。网络中每一个神经元都有一个参数，相当于方程式中变量前的参数。利用训练数据，通过不断地前向、后向传输迭代，将整个神经网络中的参数不断优化，从而使得网络能够较好地通过输入的参数，自动前向传播，输出预测值。

因为CNN自带卷积特征提取，故数据预处理阶段只进行了加窗处理。然后取到处理后的1024帧的数据，按照3：6：1的比例划分测试集、训练集和验证集，将训练集的数据作为输入传入网络中进行训练，同时每一轮训练完毕之后，用验证集合查看训练的结果。全部训练结束之后，用测试集来最终确定模型的优良程度。

下图可见，我们的模型在测试集上准确度很高，达到了100%的准确度。

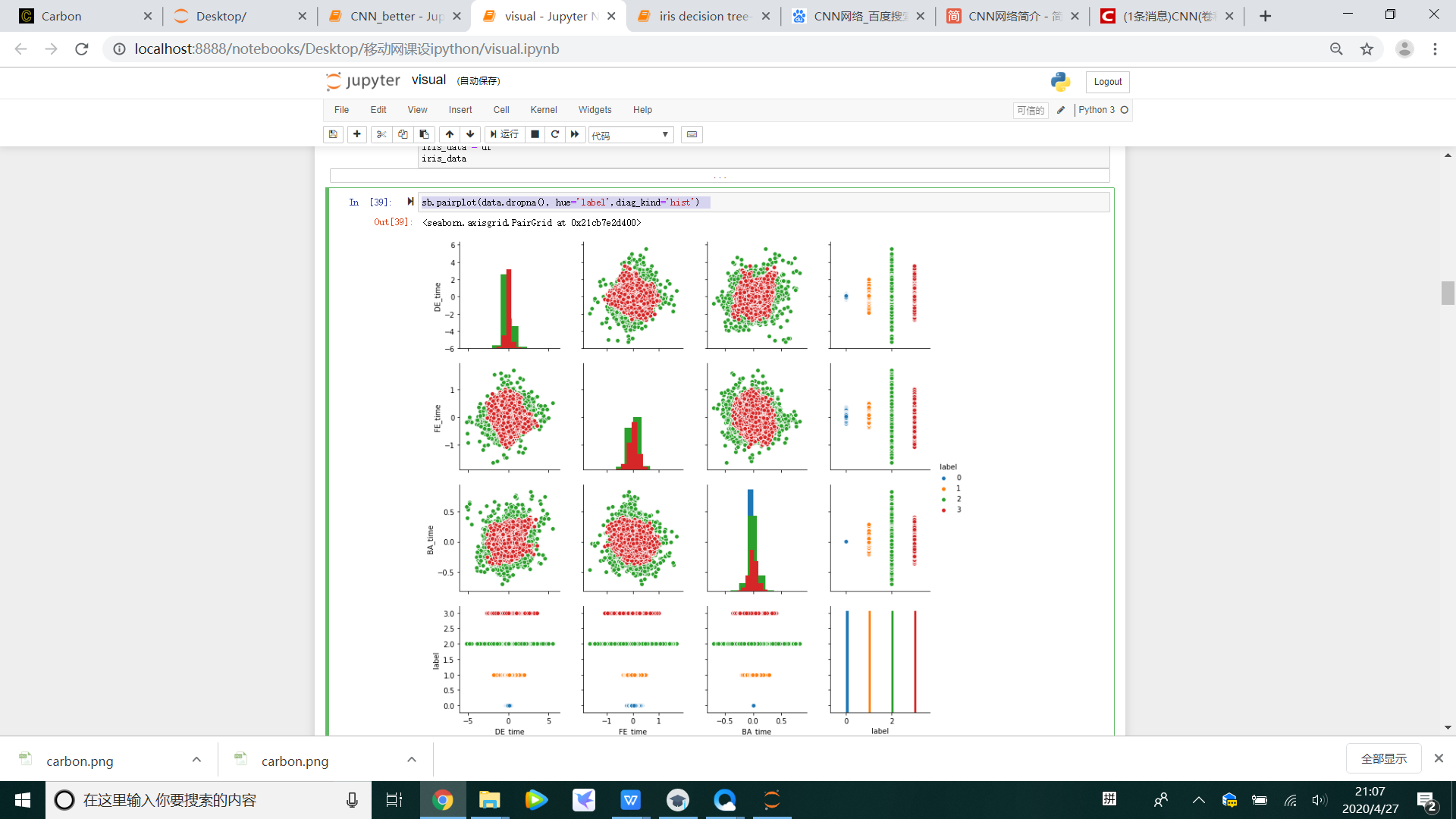


## 可视化结果展示（完整代码在visual.ipython里）

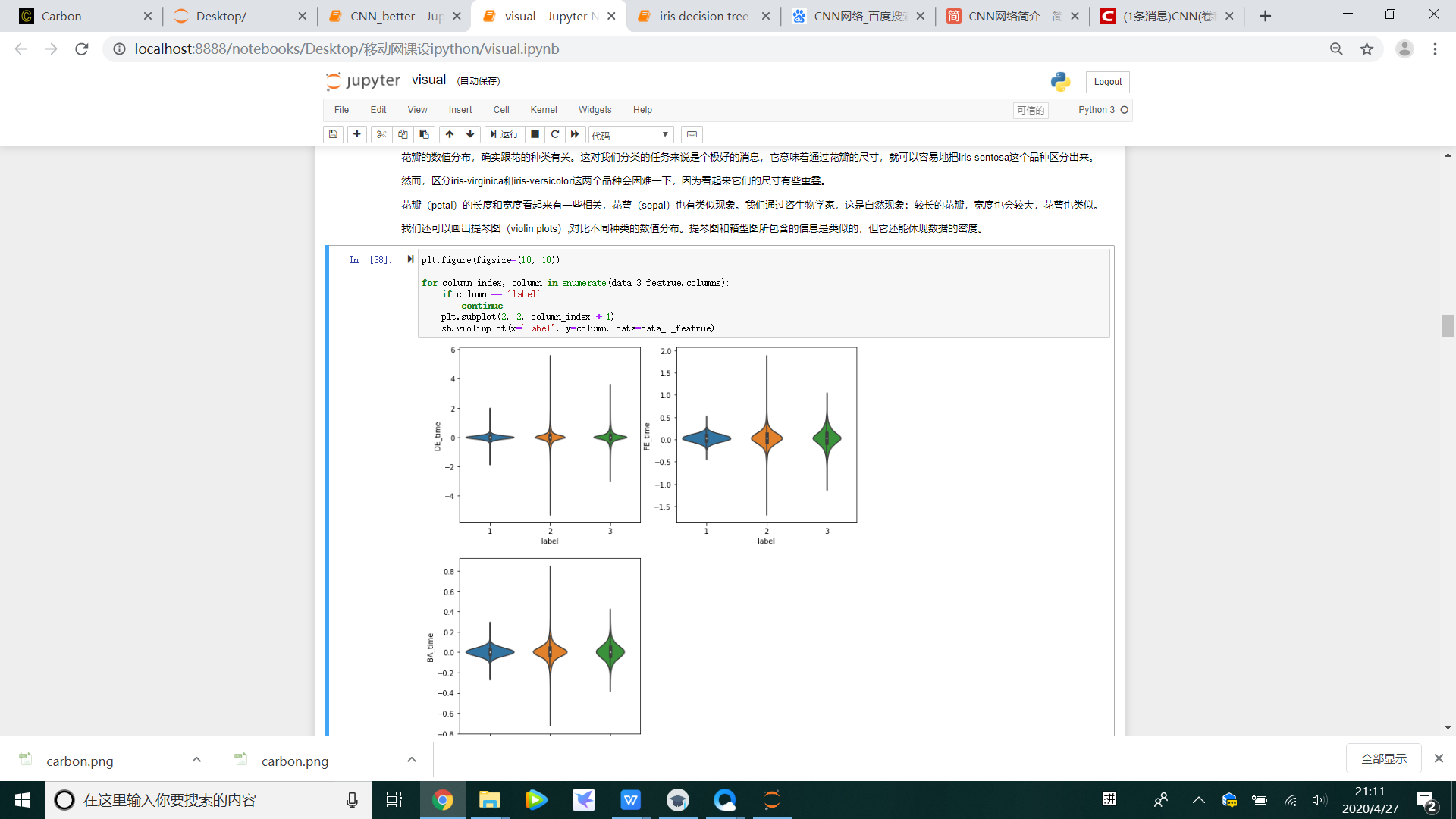
可视化的代码是套用的鸢尾花种类预测的机器学习教学文件，主要调用sklearn库自带的各种可视化接口。

首先用点状图和柱状图观察数据的特性（此时的数据已经经过了过采样，所以每个种类数据个数相同）

可以看到，不同种类之间，数据之间还是存在较大差异的。



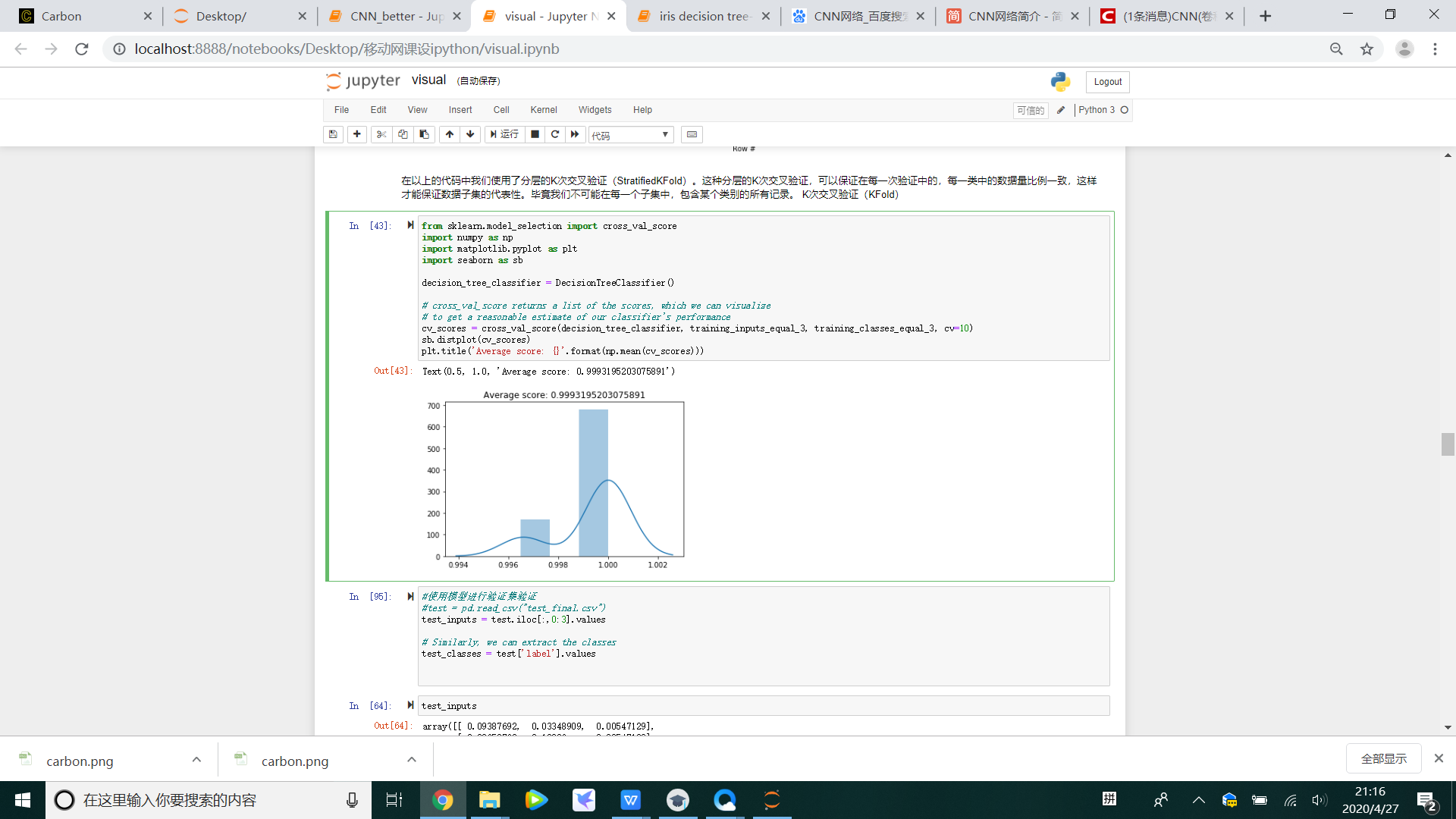
其次画出提琴图，我们可以看到虽然各数据之间，其分布有明显的差异，但是其变化范围几乎重合，这给我们的预测带来了难度。通过单一的一条数据显然是无法获得较好的预测结果的。



接下来，我尝试了使用随机森林算法来进行预测，不过失败了（第一次模型构建尝试，最后采用的CNN是在RF算法失败之后进行的新的尝试）

过程如下：

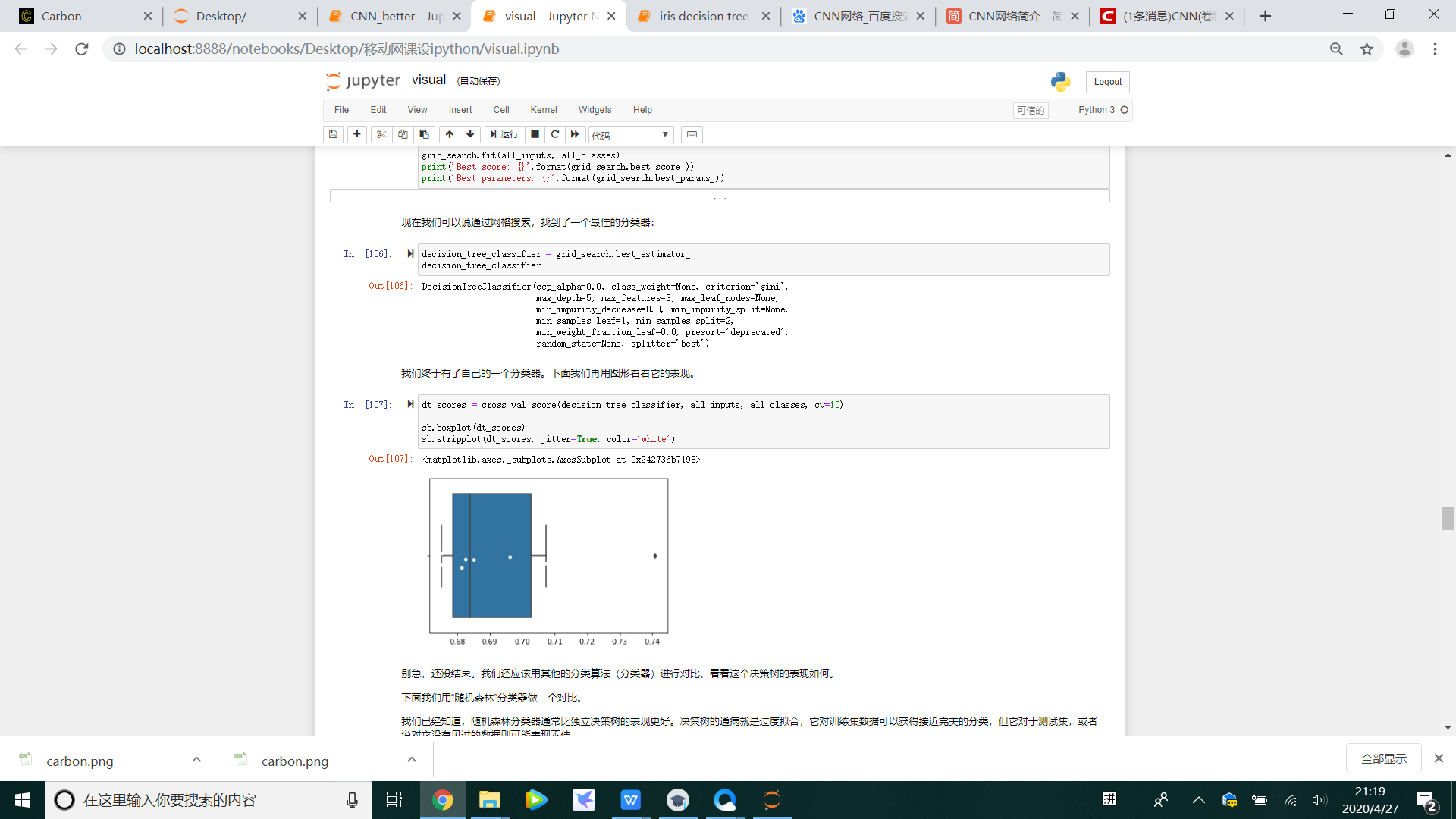
一开始，我在过采样之后的数据集上进行训练集和测试集的划分，发现效果非常好，一时间以为自己已经成功了（其实我这个时候还没有进行任何的特征提取操作，是不可能预测的非常准确的）



从上图可以看到，我们的结果好像已经很好了，多次运行，结果都能稳定在99%以上。

可是我忽略了一个问题：我使用了过采样。过采样其实是直接将很多缺失类型的数据进行了理想填充，可是这样运行出来的模型结果无疑没有任何说服力，因为几乎等同于在测试集中混入了训练集的数据。

我优化了一下训练集和测试集的选择，发现该模型的准确度急剧下降。



上图是预测结果准确度的箱型图，可以看到，结果非常不理想。

故而，我暂时放弃了使用现成的决策树接口，去手动搭建神经网络了。现在回想，其实决策树算法的失败原因就是我未进行合适的特征提取，只是简单的进行过采样，导致模型过拟合，在非训练集的数据集上，效果不佳。但是这个过程也让我对CWRU数据集的数据分布有了很好的理解，还是收获颇丰的。