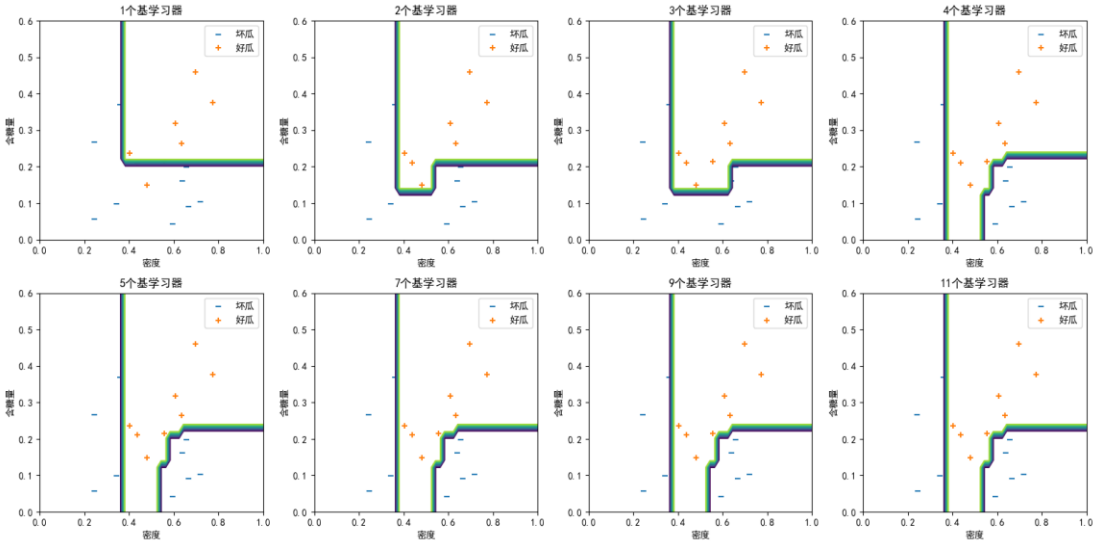
机器学习作业四

计算机科学与技术卓越班 蔡嘉轩 20185670

1.

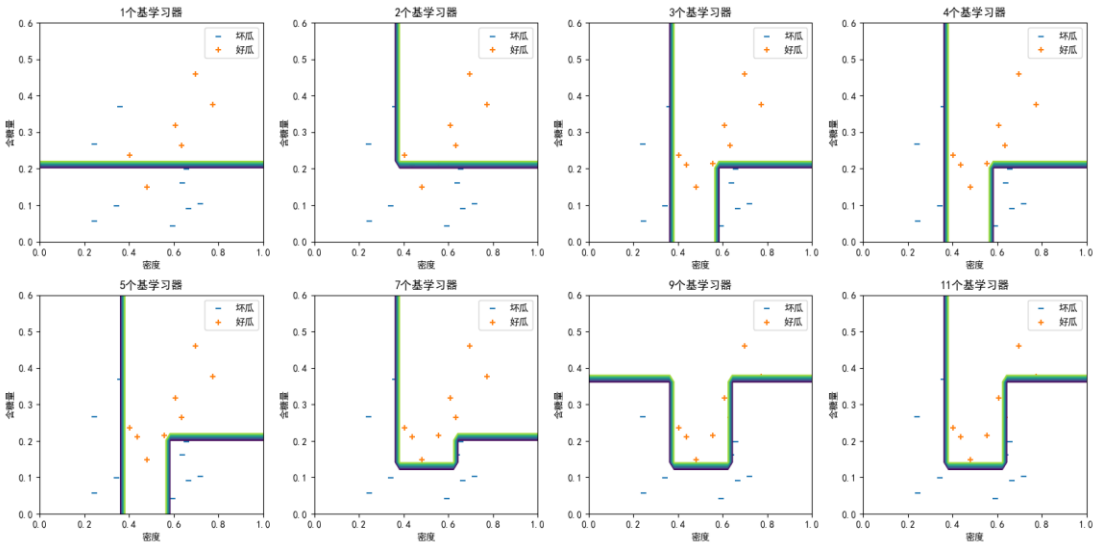
编写下列代码，调用 sklearn 库中的 adaboostclassifier，以不剪枝决策树作为基学习器，基学习器个数参数分别设置为 1 2 3 4 5 7 9 11，训练之后绘图观测决策边界与分类准确度，如下下图所示。

```
import numpy as np
import pandas as pd
from sklearn.ensemble import AdaBoostClassifier
from sklearn.tree import DecisionTreeClassifier
import matplotlib.pyplot as plt

# 从csv文件读取西瓜数据集3.0α的数据集
data = np.array(pd.read_csv('watermelon_3a.csv'))

# 分离自变量和标签
x, y = data[:, [1, 2]], data[:, 3]

# 调用sklearn库中的adaboostclassifier，以不剪枝决策树作为基学习器，基学习器个数参数分别设置为1 2 3 4 5 7 9 11
adas = []
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=1))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=2))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=3))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=4))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=5))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=7))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=9))
adas.append(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2), n_estimators=11))

# train
for ada in adas:
    ada.fit(x, y)

# 结果可视化
# 定义标签数组
label_set = []

# 设置中文字体 解决乱码
plt.rcParams['font.sans-serif']=['SimHei']

# 设置边界
minx, maxx = x[:, 0].min() - 1, x[:, 0].max() + 1
miny, maxy = x[:, 1].min() - 1, x[:, 1].max() + 1

# 生成网格
bg_x, bg_y = np.meshgrid(np.arange(minx, maxx, 0.02), np.arange(miny, maxy, 0.02))
for ada in adas:
    melon_labels = ada.predict(np.c_[bg_x.ravel(), bg_y.ravel()])
    melon_labels = melon_labels.reshape(bg_x.shape)
    label_set.append(melon_labels)

# 子图分割
fig, mysubplots = plt.subplots(nrows=2, ncols=4, figsize=(16, 8))
subs = mysubplots.flatten()
for k, sub in enumerate(subs):
    # 边界
    sub.contour(bg_x, bg_y, label_set[k])
    num = [1, 2, 3, 4, 5, 7, 9, 11]
    for i, n, m in zip([0, 1], ['坏瓜', '好瓜'], ['_', '+']):
        idx = np.where(y == i)
        sub.scatter(x[idx, 0], x[idx, 1], marker=m, label=n)
    sub.set_xlim(0, 1)
    sub.set_ylim(0, 0.6)
    sub.legend(loc='upper right')
    sub.set_title('%s个基学习器' % num[k])
    sub.set_ylabel('含糖量')
    sub.set_xlabel('密度')

plt.show()
```



与书上的图片 8.4 比较，可以发现当 Adaboost 集成的的基学习器数量增大的过程中，决策边界逐渐变得更加复杂，并且错误率不断变小。

由于使用深度为 2 的不剪枝的决策树作为基学习器训练效果很好，我们重复实现一下西瓜书上的做法。效仿西瓜书上，将基学习器不剪枝决策树的最大深度调整为 1，使用树桩作为基学习器再次运行程序，决策边界与图片 8.4 一致。更加明显地看出如上结论。



2.

相同：

(1)都是首先生成多个分类器，然后给每个分类器都赋予一个权值，最后将所有分类器按照权值累加；

(2)都是通过将表现较好或一般的数个模型组合在一起来共同生成一个性能好的模型；

(3)都可以看作重复选择一个表现一般的模型并且每次基于先前模型的表现进行调整，即在每次使用基模型不断在空间中搜索最优模型的过程。

不同：

(1)AdaBoost 是使用每个分类器的分类结果改变样本的权值从而生成新的分类器和生成权值，但每个样本本身不会改变；而 GradientBoosting 将每个分类器对样本的预测值与真实值的差值（梯度方向）传入下一个分类器来生成新的分类器和生成权值，而每个样本的权值不变。

(2)AdaBoost 通过提升错分数据点的权重来定位模型的不足并生成新学习器；而 Gradient Boosting 是通过算梯度来定位模型的不足并生成新学习器，相比 AdaBoost, Gradient Boosting 可以使用于更多种类的目标函数。

(3)Gradient Boosting 算法适用范围更广，函数取值可以连续取值、二值离散取值和多值离散取值，损失函数可以是平方差，绝对偏差，Huber，logistic 型。而 AdaBoost 算法只能用于二分类的情况，函数取值只能是二值离散取值(0,1)。