一、实验环境

1、软件环境:Anaconda3 (内置 Python3.7)

建议使用该软件中的 vscode 工具或者Jupyter Notebook

2、所需工具包 sklearn 机器学习工具包，Anaconda3 已经集成。若需下载以及学习 sklearn 中 API 相关的内容，网址为:https://scikit-learn.org/

3、可视化决策树需要安装graphviz和pydotplus，直接在Anaconda中的Environments界面中安装即可，如图1所示安装“graphviz”。安装“pydotplus”的方法类似。

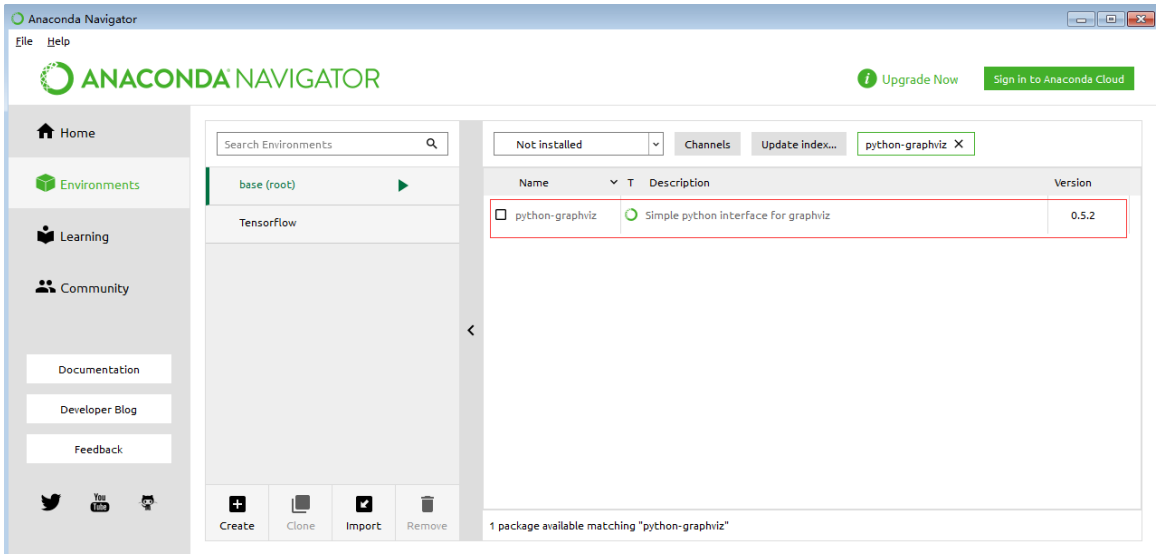

图1 Environments界面

4、所需数据集全部在sklearn.datasets库中，分别为：

fetch_california_housing

load_wine

请大家提前下载好，保证课程效果。

5、评价指标

评价指标所在的库为：sklearn. metrics

回归模型的评价指标：R2、MSE、MAE

分类模型的评价指标：accuracy_score（准确率）

二、决策树模型所需函数

所有决策树模型函数均在sklearn.tree的库中

函数名	说明
DecisionTreeRegressor	决策树回归函数
RandomForestRegressor	随机森林回归函数
DecisionTreeClassifier	决策树分类函数
RandomForestClassifier	随机森林分类函数

三、应用实例

使用 load_wine 数据集构建决策树分类模型

```
import numpy as np
import matplotlib.pyplot as plt
from matplotlib.colors import ListedColormap
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_wine
#加载数据集
wine=load_wine()
#取数据集中的前 2 个特征
x=wine.data[:,:2]
y=wine.target
#拆分数据集
x_train,x_test,y_train,y_test=train_test_split(x,y)
#初始化决策树分类器，设置决策树分类器的最大深度为 3
decisionTree=DecisionTreeClassifier(max_depth=3)
#拟合训练集
decisionTree.fit(x_train,y_train)
#输出训练集上的分类评分
print(decisionTree.score(x_train,y_train))
#定义图像分区的颜色
map_light=ListedColormap(['#FFAAAA','#AAFFAA','#AAAAFF'])
map_bold=ListedColormap(['#FF0000','#00FF00','#0000FF'])
#使用样本的特征值创建图像的横轴和纵轴
```

```
x_min,x_max=x_train[:,0].min()-1,x_train[:,0].max()+1
y_min,y_max=x_train[:,1].min()-1,x_train[:,1].max()+1
xx,yy=np.meshgrid(np.arange(x_min,x_max,.2),np.arange(y_min,y_max,.2))
#np.c_是按行连接两个矩阵，就是把两矩阵左右相加，要求行数相等
z=decisionTree.predict(np.c_[xx.ravel(),yy.ravel()])
#给样本分配不同的颜色
z=z.reshape(xx.shape)
plt.figure()
plt.pcolormesh(xx,yy,z,cmap=map_light)
#把样本用散点表示出来
plt.scatter(x[:,0],x[:,1],c=y,cmap=map_bold,edgecolor='black',s=20)
#设置 x 轴的坐标范围
plt.xlim(xx.min(),xx.max())
#设置 y 轴的坐标范围
plt.ylim(yy.min(),yy.max())
plt.show()
```

训练集评分效果，如图 2 所示。

```
train_score:0.8947368421052632
```

图 2 训练集评分结果

分类可视化效果，如图 3 所示。

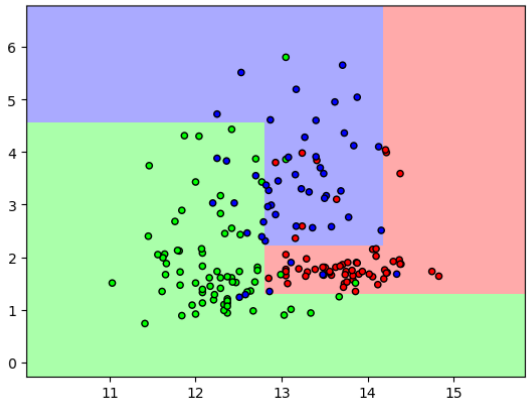

图3 分类可视化效果

使用 graphviz 可视化决策树构建过程

```
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.tree import export_graphviz
#导入可视化工具
import graphviz
#导入转换 dot 文件的工具
```

```
import pydotplus
wine=load_wine()
x=wine.data[:,:2]
y=wine.target
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.2,random_state=1)
#决策树分类函数，设置树的最大深度为 3
ctr=DecisionTreeClassifier(max_depth=3)
#拟合数据
ctr.fit(x_train,y_train)
#将决策树可视化，即用图来表示
dot_data=export_graphviz(ctr,out_file=None,feature_names=wine.feature_names[:2],
impurity=False,rounded=True,filled=True)
#将 dot 类型的数据转换为图
graph = pydotplus.graph_from_dot_data(dot_data)
#保存为图片格式
graph.write_png('e:\\wine.png')
```

wine.png 的效果，如图 4 所示。

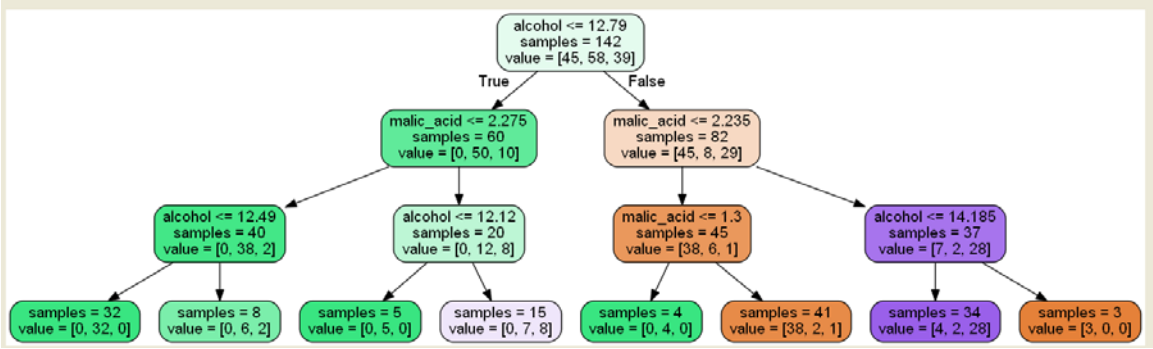

图4　决策树构建效果

四、实验内容

1、回归算法

所用数据集：fetch_california_housing

实验内容：使用决策树和随机森林预测房价

实验步骤：

（1）使用数据集中的任意 2 个特征生成一颗决策树，树的深度为 3，并将其可视化显示

（2）运用决策树和随机森林方法，使用 fetch_california_housing 的全部特征来预测房价，分别使用 R^2、MSE、MAE 来评估实验结果

（3）参数优化（使用 GridSearchCV）

2、分类算法

所用的数据集：load_wine

实验内容：分别使用决策树和随机森林预测酒的分类

实验步骤：

（1）使用 load_wine 中的全部特征来预测酒的分类

（2）参数调优（使用 GirdSearchCV）