# 案例一 玻璃类型识别

## 【1】案例背景

本题由 **顾翔 921127940122**负责

玻璃碎片是犯罪现场的常见物之一，若能根据碎片准确识别出玻璃类别，对于案件侦查将是极大的帮助，本题使用一份玻璃类型数据集。该数据一共有214个样本，每个样本包含9个特征，其中type特征指示玻璃类别，请先从数据集随机抽取70%样本作为训练集，然后利用本章学习的任意分类算法，使用训练集训练分类模型，最后选择合适的评价指标，在测试集上评估分类效果

## 【2】方法陈述

首先使用pandas对数据进行处理，接着，按照题目要求，将数据集随机抽取数据，分成70%训练集和30%测试集。

对数据集进行训练拟合，对所采用的决策树、SVM、随机森林进行调参，对在测试集上的表现以正确率为评定指标，比较分析结果

运行环境，python3安装有pandas、scikit-learn、numpy库，运行软件：Anaconda、jupyter notebook

## 【3】实验代码

### 步骤一、数据处理

1.数据处理，去掉无关内容id

import pandas as pd

df=pd.read\_csv('glass.data')

df.columns=['id','RI','NA2O','MGO','AL203','SIO2','K2O','CAO','BAO','FE2O3','Type']

del df['id']

df.head()



2.将数据集随机抽取数据，分成70%训练集和30%测试集

3.经过z-score标准化后的结果经测试反而变差了，因此不使用标准化

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

*#from sklearn import preprocessing*

train\_x,test\_x=train\_test\_split(df,train\_size=0.7,random\_state=5)

train\_y=train\_x['Type']

del train\_x['Type']

test\_y=test\_x['Type']

del test\_x['Type']

*#train\_x=preprocessing.scale(train\_x.values)*

*#test\_x=preprocessing.scale(test\_x.values)*

### 步骤二、数据处理

1. **决策树**

* 这里选用了来自scikit-learn的DecisionTreeClassifier库，不纯度度量方法采用信息熵效果较优于基尼指数
* 库参考文档：<https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html>

from sklearn import tree

clf=tree.DecisionTreeClassifier(criterion='entropy')

clf=clf.fit(train\_x,train\_y)

score=clf.score(test\_x, test\_y)

**print**('决策树的正确率：',score)

**决策树的正确率： 0.734375**

1. **SVM**

* 这里选用了来自scikit-learn的svm库
* C为惩罚因子，这里进行了粗略的调参，使得模型既不过于欠拟合也不过拟合
* 采用了高斯核，在此数据集上的表现比linear更优
* 库参考文档<https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

import sklearn.svm as svm

model=svm.SVC(C=1000000,kernel='rbf')

model.fit(train\_x,train\_y)

score=model.score(test\_x, test\_y)

**print**('SVM的正确率：',score)

**SVM的正确率： 0.75**

1. **随机森林**

* 所有基评估器都是决策树的Bagging算法，效果理论上优于单个决策树
* 库参考文档<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html>

from sklearn.ensemble import RandomForestClassifier

rfc=RandomForestClassifier(random\_state=45)

rfc.fit(train\_x,train\_y)

score=rfc.score(test\_x, test\_y)

**print**('随机森林的正确率：',score)

**随机森林的正确率： 0.90625**

## 结果分析

1. 模型评估标准采用 正确率=(TP+TN)/(TN+FN+FP+TP)
2. 由多个决策树组成的集成模型--随机森林的效果优于单个决策树，达到了较优的水平，在轻微调参的情况下，随机森林效果优于采用了高斯核的SVM，SVM略优于单个决策树