**利用随机森林解决给定的分类问题**

1. 简要说明
   1. 所用模型：
      1. 随机森林：随机森林是一个包含多个决策树的[分类器](https://baike.baidu.com/item/%E5%88%86%E7%B1%BB%E5%99%A8)，并且其输出的类别是由个别树输出的类别的众数而定。
      2. 极限随机森林：极限随机森林是随机森林的一个变种，极限随机森林中的计算分割点方法中的随机性进一步增强。
   2. 训练集和数据集
      1. 训练集train\_dataset2，train\_dataset3；测试集：test\_dataset2 ，test\_dataset3。
      2. 训练集2中有460个正类样本（类别标签为1），3404个负类样本（类别标签为0）。
      3. 训练集3中有275个正类样本（类别标签为1），1273个负类样本（类别标签为0）。
   3. 使用的包

**import** csv

**from** sklearn.preprocessing **import** StandardScaler

**import** numpy **as** np

**import** matplotlib.pyplot **as** plt

**from** sklearn.decomposition **import** PCA

**from** sklearn.ensemble **import** ExtraTreesClassifier

**from** sklearn.metrics **import** roc\_curve,auc

1. 算法步骤（整体）

步骤1：读入数据：读取csv文件，将训练集和测试集的特征向量分别存于数组x和xx中，标签分别存于数组y和yy中。

步骤2：数据预处理：将数据集中的非012替换为0，对训练集和测试集的特征值进行标准化；使用主成分分析（PCA）方法降维，提取数据的主要特征分量。

步骤3：训练分类器：分别使用了两种模型，随机森林（RandomForestClassifier）和极限随机森林（ExtraTreesClassifier）。

步骤4：分类器的评价指标：使用的评价指标有AUC和MCC。

1. 模块说明
   1. 扩大数据集中012的占比（由于每个特征属性的取值过于少，所以将特征值为非012的属性值转换为0）
      1. 输入参数说明

x：数据集。

* + 1. 输出参数说明

e：转换后的数据集。

* + 1. 算法说明

通过遍历数据集矩阵中的数据，判断其非012时置为0。

* + 1. 相关代码



* 1. 将str类型的数据转换成int型
     1. 输入参数说明

x：数据集。

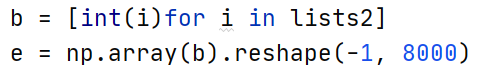
* + 1. 输出参数说明

e：转换后的数据集。

* + 1. 算法说明

通过遍历数据集矩阵中的数据，将每一项字符串型转化为int型，存储形式同csv文件相同，每一行为一个样本，每个样本有8000列。

* + 1. 相关代码



* 1. 特征降维——PCA
     1. sklearn中PCA库函数的参数说明

n\_components：PCA算法中所要保留的主成分个数n，也即保留下来的特征个数n。

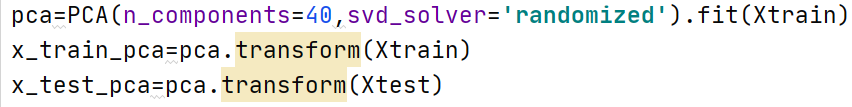
svd\_solver：指定奇异值分解SVD的方法。

* + 1. 输出参数说明

self：经过降维后得到的数据集，返回结果赋给pca。

* + 1. 相关代码

通过调用pca.transform对训练集和测试集降维。



* 1. 随机森林和极限随机森林
     1. sklearn中RandomForestClassifier库函数的参数说明

n\_estimators：弱学习器的最大迭代次数，即最大的弱学习器的个数。

max\_depth：决策树的最大深度。

min\_samples\_split：叶子节点的最少个数。

random\_state：控制随机状态。

1. 输出参数说明

将结果存放于clf\_proba中，通过调用clf\_proba.predict(xtest)得到分类结果。

1. 相关代码

随机森林中，参数取值为n\_estimators=50，max\_depth=None, min\_samples\_split=5, random\_state=0。

极限随机森林中，参数取值为n\_estimators=100，max\_depth=None, min\_samples\_split=10, random\_state=0。





* 1. 计算MCC
     1. 输入参数说明

test\_num:测试集的样本数。

pred\_y:存放预测标签的数组。

labels:存放真实标签的数组。

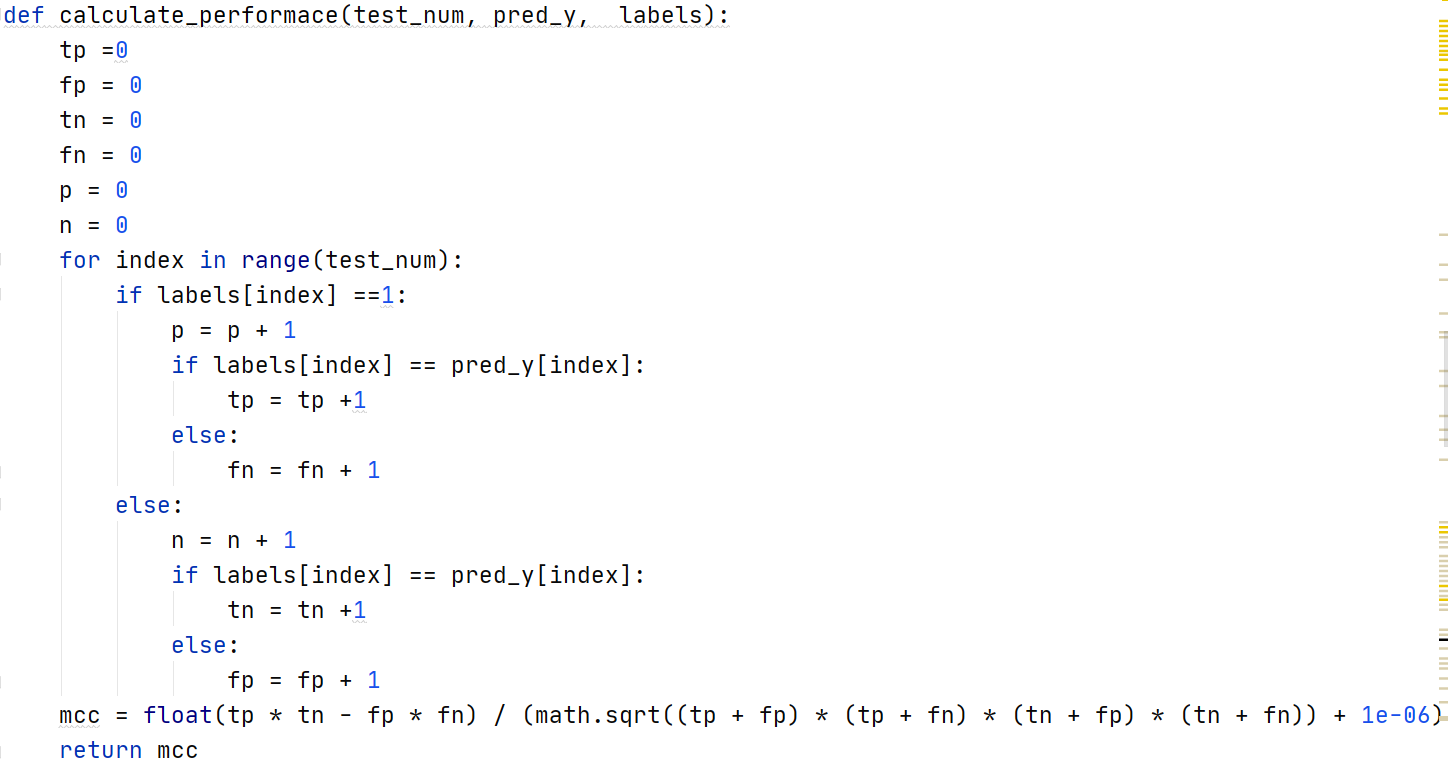
* + 1. 输出参数说明

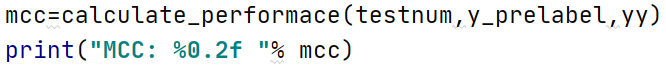
mcc:计算的MCC值。

* + 1. 算法说明

从index=0开始遍历测试集，当此时labels[index]==1，正样本数加1，并且此时真实标签与预测标签一样时，tp加1，不然fn加1，当此时labels[index]==0，负样本数加1，并且此时真实标签与预测标签一样时，tn加1，不然fp加1,最后根据公式计算MCC；MCC最终只取到小数点后两位。

（4）相关代码





* 1. ROC曲线及AUC
     1. sklearn中roc\_curve库函数的参数说明

**y\_true**：真实的样本标签。

**y\_score：**对每个样本的预测结果。（作业中实现了两种，一是随机森林生成的类标签y\_prelabel，另一个是生成的概率值y\_score）

pos\_label：正样本的标签。

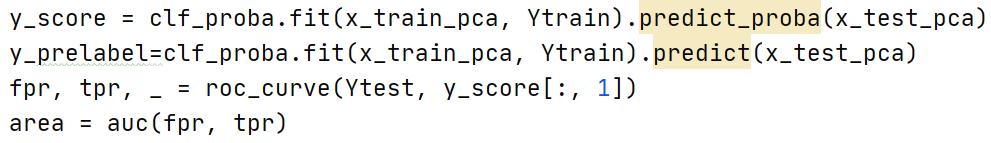
* + 1. 输出参数说明

fpr：False positive rate。

tpr：True positive rate。

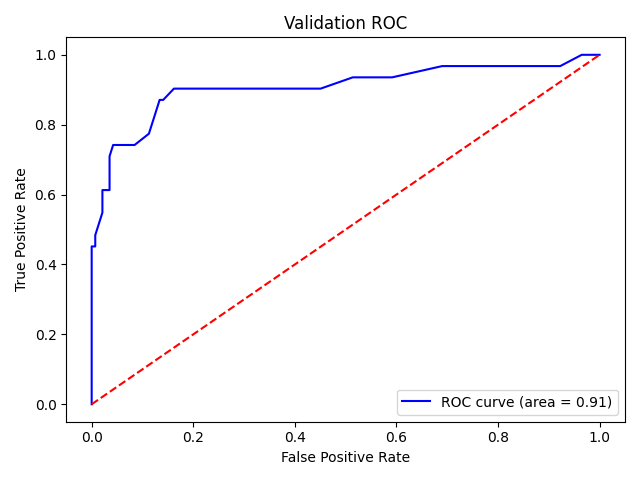
* + 1. 相关代码

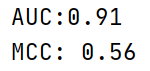
AUC最终只取到小数点后两位。



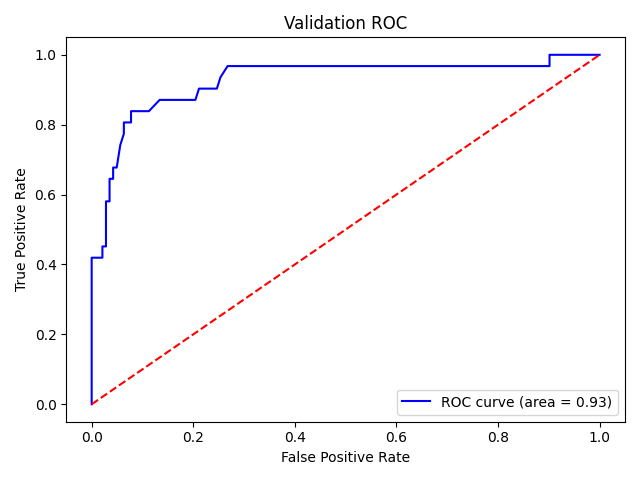


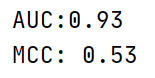
1. 结果演示（结果均取到小数点后两位）
   1. train\_dataset2，test\_dataset2
      1. 随机森林



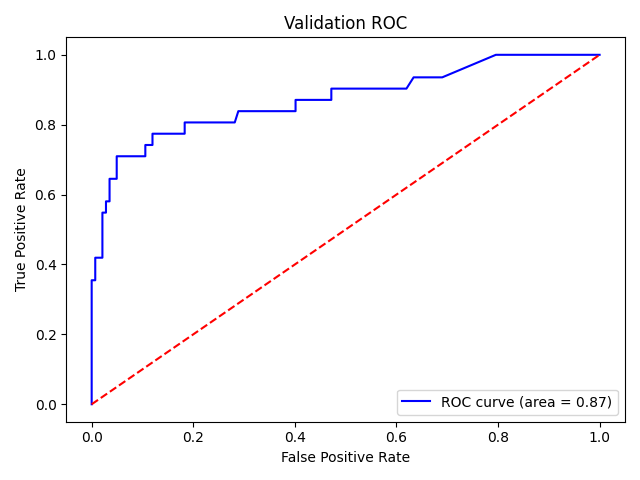


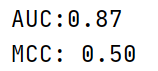
* + 1. 极限随机森林



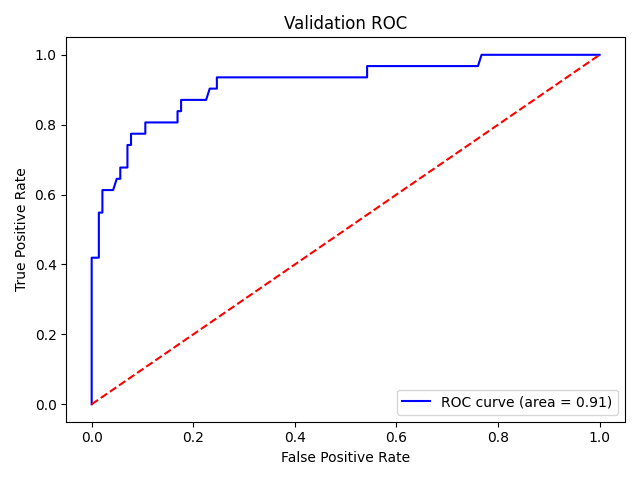


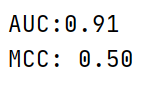
* 1. train\_dataset3，test\_dataset3
     1. 随机森林





* + 1. 极限随机森林





1. 总结
   1. 随机森林的优点：
      1. 可以用来解决分类和回归问题：随机森林可以同时处理分类和数值特征。
      2. 抗过拟合能力：通过平均决策树，降低过拟合的风险性。
      3. 只有在半数以上的基分类器出现差错时才会做出错误的预测：随机森林非常稳定，即使数据集中出现了一个新的数据点，整个算法也不会受到过多影响，它只会影响到一颗决策树，很难对所有决策树产生影响。
      4. 能够处理很高维度的数据，并且不用做特征选择。
      5. 对于不平衡的分类资料集来说，它可以平衡误差。

缺点：如果一些分类/回归问题的训练数据中存在噪音，随机森林中的数据集会出现过拟合的现象。比决策树算法更复杂，计算成本更高。由于其本身的复杂性，它们比其他类似的算法需要更多的时间来训练。

2． 极限随机森林与随机森林的区别：

（1）对于每个决策树的训练集，RF采用的是随机采样bootstrap来选择采样集作为每个决策树的训练集，而extra trees一般不采用随机采样，即每个决策树采用原始训练集，极限随机森林中的计算分割点方法中的随机性进一步增强。

（2）在选定了划分特征后，RF的决策树会基于基尼系数，均方差之类的原则，选择一个最优的特征值划分点，这和传统的决策树相同。但是extra trees比较的激进，他会随机的选择一个特征值来划分决策树。

3． 从实验数据来看，不管是随机森林模型还是极限随机森林模型，对于评价指标AUC和MCC来说，测试集2的表现结果大多是好于测试集3的，分析的可能原因：

1. 训练集2和训练集3的样本大小不同。
2. 以及正负样本数相差较大，数据不平衡。
3. 模型当前的参数对测试集2的分类比较友好。

4． 在随机森林模型中，用训练集3训练模型，最终用测试集3来测试模型时，AUC效果不是很好。

5． 从随机森林和极限随机森林的优缺点以及区别，再结合实验数据来看，不管怎么调整模型参数，在大多数情况下，极限随机森林的效果都是好于随机森林的，原因是：在极限随机数中，计算分割点方法中的随机性进一步增强，阈值是针对每个候选特征随即生成的，并且选择最佳阈值作为分割规则，这样能进一步减少一点模型的方差，总体上效果更好。

1. 参考文献

[1]赵卫东,董亮.机器学习[M].北京:人民邮电出版社,2018:87-93.

[2]李航.统计学习方法[M].北京:清华大学出版社,2019:297-309.