# 实验环境

linux(manjaro)+python3.8+jupyter notebook

# 数据集来源

我的数据来自与kaggle(数据挖掘比赛平台)上的泰坦尼克号数据，[数据集链接](https://www.kaggle.com/c/titanic/data)

# 实践过程

我训练了两种模型——决策树模型和高斯贝叶斯模型。

## 数据观察

1. 观察整体数据情况并对特征做一定的处理

* 使用train.head()函数查看训练集前5行。通过观察发现，发现PassengerId和Name是没有用的特征，所以删除; Ticket是票编号，种类特别多，后面不好处理，于是我们把它变成票编号共用数(同一个票编号可以被多个人共用); Cabin是船舱编号(字母+数字，例如C85)，后面的数字觉得没有意义，就把数字部分除去了。

最后我们确定了train的特征，如表格所示

|  |
| --- |
| 因变量 |
| Survived存活与否 |

|  |
| --- |
| 自变量 |
| Pclass客舱等级 |
| Sex性别 |
| Age年龄 |
| Parch直系亲友数 |
| SibSp旁系亲友数 |
| Ticket票编号共用数 |
| Fare票价 |
| Cabin船舱编号 |
| Embarked上船的港口编号 |

1. 观察每个自变量和因变量的关系

自变量x可以分成两类，非数值变量或数值变量，非数值变量使用条行图barplot，数值变量使用箱图boxplot。

|  |  |  |
| --- | --- | --- |
| 自变量类型 | 自变量集合 | 使用的函数 |
| 非数值变量 | Pclass、Sex、Parch、SibSp、Ticket、Cabin、Embarked | barplot |
| 数值变量 | Age、Fare | boxplot |

观察得到每个自变量都和因变量有较为密切的关系，所以保留

## 数据预处理

1. 缺失值处理：缺失值处理采用填充众数的方法
2. 编码处理：有些标签变量使用字符串作为属性值，例如Sex特征man和felman,会影响后面对模型的训练，所以使用LabelEncoder函数进行处理，例如man编码为1，felman编码为0
3. 模型建立

* 选用了两种模型——决策树模型sklearn.tree.DecisionTreeClassifier和贝叶斯模型GaussianNB

## 模型选择

1. 决策树：这里我使用了网格调参处理器GridSearchCV来找到每种模型最优的超参数。通过训练我得到了决策树的最优超参数为{'criterion': 'entropy', 'max*depth': 3, 'min*samples\_split': 4};
2. 贝叶斯：因为贝叶斯没有超参数，所以就不需要进行模型选择

## 模型训练

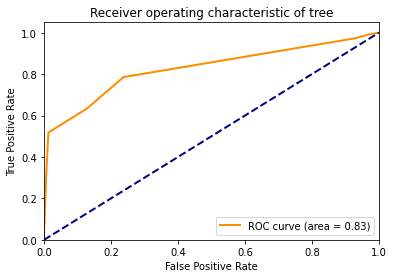
我先对特征的处理、数据预处理、找到最优超参数的分类模型分别使用sklearn中的自定义处理器封装，在用sklearn中的管道Pipeline将处理器有序链接起来形成流水线，因为有两个模型，所以有两条流水线，分别为pipeline\_tree和pipeline\_bayes。最后使用fit函数对训练集进行训练，用predict\_prob输出测试集的结果概率

## 模型评估

我采用ROC曲线的方法来对两种模型进行评估，AUC值越高说明模型越好

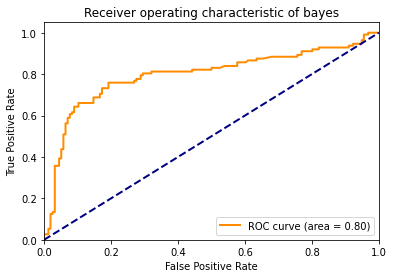
* 决策树

决策树的ROC曲线和AUC值，可以看到AUC=0.83



* 贝叶斯

贝叶斯的ROC曲线和AUC值，可以看到AUC=0.80



因为决策树的AUC值大于贝叶斯，所以决策树模型更好

# 实践过程思考和不足

## 思考

### 自定义转换器

当时我觉得那些对训练集的预处理，要在测试集上复原是一件很麻烦的事情，通过网络的查询，我发现有人通过对训练集和测试集一起处理的方法来简化操作(比如一起做归一化)，但是我仔细想想觉得这种方式只适合于比赛的情况，在工业上并不使用(因为工业上的应用的数据集是慢于训练集出现的)，所以我就学习了自定义转换器的知识，来让处理一步到位，简化操作。

### 管道

一般我们会把最终训练好的模型的保存起来，但是这会有一个问题，对训练集的预处理并没有保留下来，将来在使用这个模型前还需要进行与测试集相同的预处理，不仅麻烦并且需要记录具体对训练集做了那些预处理。所以我学习了管道的知识，将多个预处理转换器+评估器合并成一个流水线，最终保存的模型可以直接使用。

## 不足

听了同学们的汇报，有很多收获，自己有很多忽略的点。(但是最近比较忙，就暂时没有对实践做进一步的改进)

* 预处理还可以加上离群值的处理
* 需要补充训练集和测试集拟合曲线，来判断是否有过拟合。(我当时没有做是因为我的模型评估都是基于测试集，选取的是使测试集得分最高的超参数，觉得测试集得分最高的点一定不是过拟合点)
* 使用更多厉害的模型，比如随机森林。(我当时没有做的原因一是厉害的模型的使用方法和简单的模型是一样的，都是用sklearn的评估器封装起来，但是我不是很懂复杂模型的内部原理; 二是觉得项目的目的是让我们掌握完整的数据挖掘过程，不需要太关注用了多厉害的模型)

# 补充

* 整个项目放在github上，链接为<https://github.com/Shuojia-Huang/DataMining-Titanic.git>
* 学习过程记录在博客上，链接为<https://www.cnblogs.com/Serenaxy/p/14131489.html>