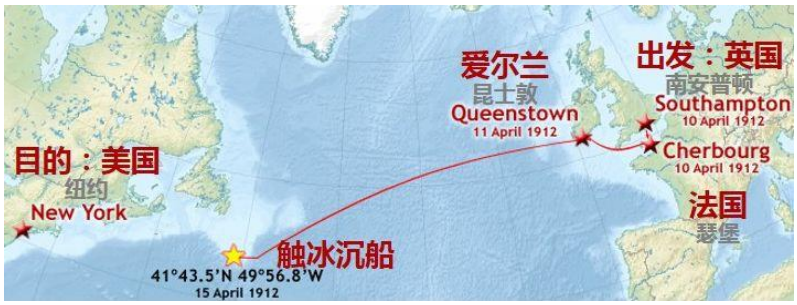
泰坦尼克号生存预测

贺耕群　2022103798

一、分析背景与目的

1.1 分析背景

1912 年 4 月 10 日，号称 “世界工业史上的奇迹”的豪华客轮泰坦尼克号开始了自己的处女航，从英国的南安普顿出发驶往美国纽约，途经法国瑟堡-奥克特维尔以及爱尔兰昆士敦。1912 年 4 月 14 日夜晚，泰坦尼克号撞上冰山。随后，船裂成两半后沉入大西洋。更加不幸的是，船上没有足够的救生艇，这直接导致了 2224 名乘客和船员中有 1502 人遇难。



1.2 分析目的

虽然乘客和船员最终的生存情况可能与运气相关，但也不乏一些人群，他们的生存比率似乎要高于其他人群。因此，本文分析的目的是：基于当时船员和乘客的特征信息，预测乘客的生存情况。

二、数据来源及描述

本次分析的数据来源于机器学习竞赛网站 Kaggle（Titanic - Machine Learning from Disaster | Kaggle）。其中，训练集中包含 891 个样本（11 个特征变量和 1 个标签变量），测试集中包含 418 个样本（11 个特征变量），分别占总样本的 68.1%和 31.9%。各变量的含义详见表 2-1：

表 2-1 变量及描述

变量	描述
PassengerId	乘客编号
Survived	生存情况（1=存活，0=死亡）

Pclass	客舱等级（1=一等舱，2=二等舱，3=三等舱）
Name	姓名
Sex	性别（female，male）
Age	年龄
SibSp	乘客在船上的兄弟姐妹/配偶数 （同代直系亲属数）
Parch	乘客在船上的父母/子女数 （不同代直系亲属数）
Ticket	船票编号
Fare	船票价格
Cabin	客舱号
Embarked	登船港口（S=Southampton 英国南安普顿，C=Cherbourg 法国瑟堡市，Q=Queenstown 爱尔兰昆士敦）

三、数据预处理

3.1 缺失值处理

训练集和测试集中数据缺失情况统计如表 3-1：

表 3-1 数据缺失情况

变量 样本缺失情况	训练集 （total=891）	测试集 （total=418）
PassengerId	0	0
Survived	0	
Pclass	0	0
Name	0	0
Sex	0	0
Age	177	86
SibSp	0	0

Parch	0	0
Ticket	0	0
Fare	0	1
Cabin	687	327
Embarked	2	0

从上述变量的缺失情况统计可以看出，缺失情况比较严重的是变量 Cabin，其次是变量 Age，最后变量 Embarked 和 Fare 也有少量的缺失。

对于变量 Cabin，其数据在总样本中的缺失比率高达 77.46%，考虑到该变量数据缺失情况严重，本分析对该变量的缺失值进行了保留；对于数值型变量 Age（见图 3-1），由于数据呈现偏态分布，本文采用中位数来填补其缺失值；对于数值型变量 Fare（见图 3-2），由于数据分布比较集中，本文采用众数来填补其缺失值；对于分类型变量 Embarked（见图 3-3），由于其取值大部分集中在“S”，本文采用“S”来填补其缺失值。

图 3-1 总样本中 Age 的频数分布直方图

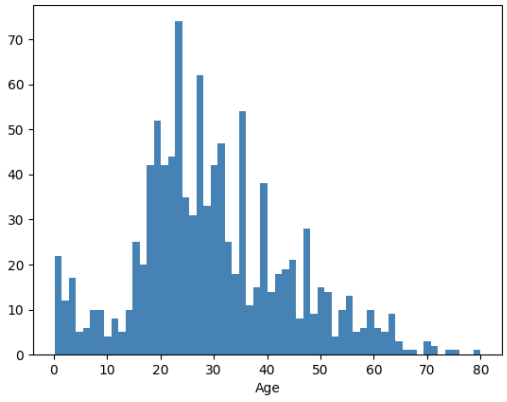


图 3-2 总样本中 Fare 的频数分布直方图

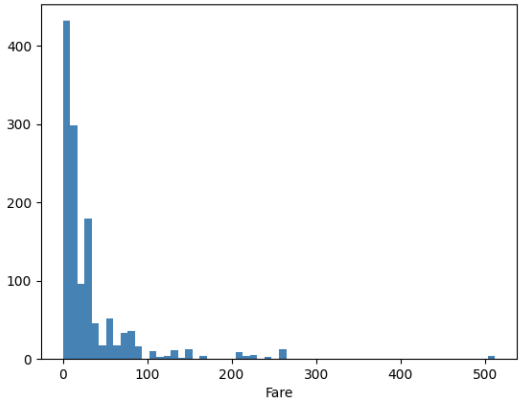
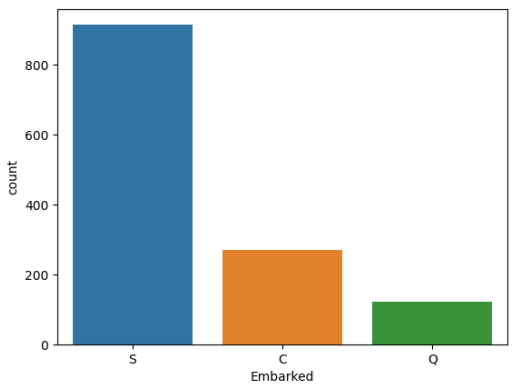


图 3-3 总样本中 Embarked 的频数分布图



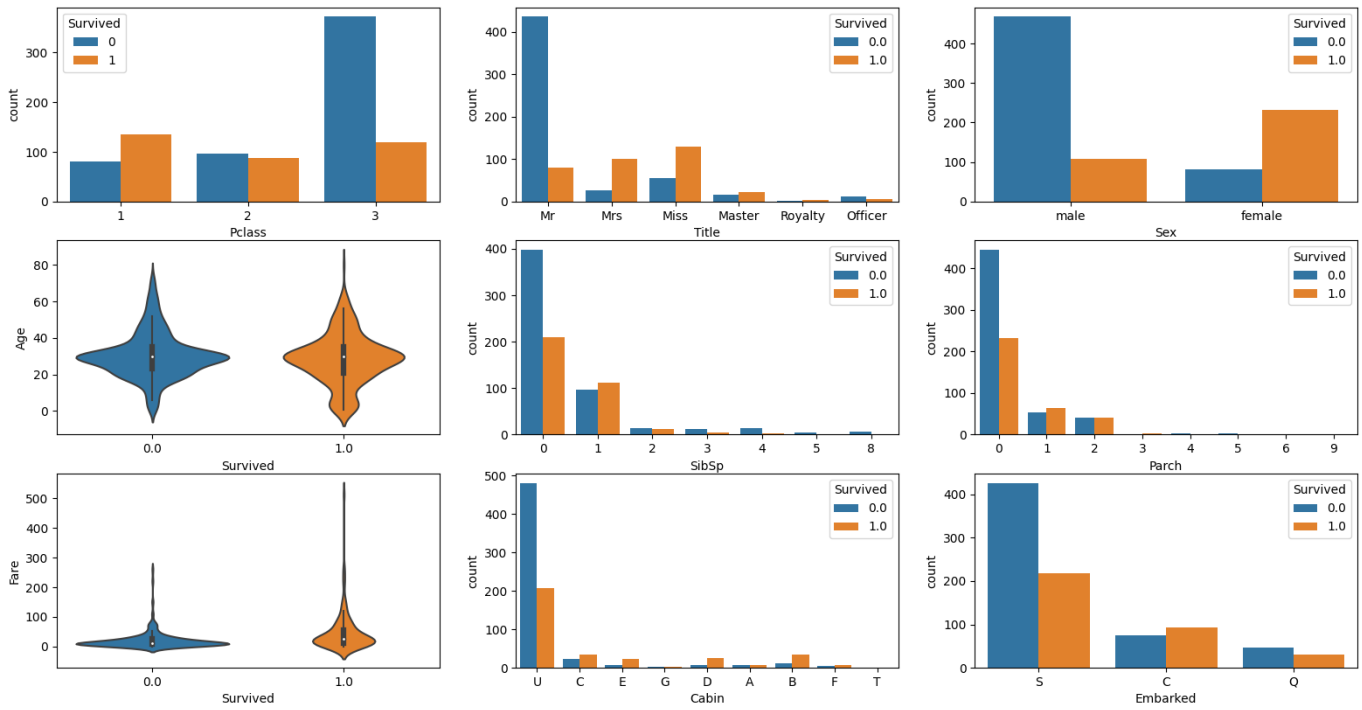
以上处理完成后，再次检查数据集中缺失值是否处理干净。若还存在缺失值，则会抛出异常，不能进行后续分析。以下是本文分析过程中所定义的异常类：

```
class CheckNull(Exception):
    def __init__(self,dat):
        self.dat = dat

    def __str__(self):
        print("存在缺失数据")

def check(dat):
    s = full.isnull().any()
    if True in list(s):
        raise CheckNull(dat)
    else:
        print(dat.head(5))
```

3.2 特征处理与分析

观察每个特征与 Survived 的关系。本文考虑到 PassengerId 和 Ticket 并不能提供有价值的信息，下图（图 3-4）仅展示了其余 9 个特征变量与 Survived 的关系。其中，Title 是从变量 Name 中提取出的一个变量。基于当时的文化背景，Name 中的头衔在一定程度上反映了乘客的身份信息，且这部分信息对 Survived 变量的分析及预测非常有价值。所以，将 Name 中的头衔提取出来，并分为 Officer/Royalty/Mr/Mrs/Miss/Master 六类（其含义详见表 3-2）。

图 3-4 各特征变量与 Suvived 的关系



（注：图 3-4 中 Cabin 取值为客舱号首字母，且当 Cabin=U 时，表示缺失字段）

表 3-2 Title 取值及含义

类别	含义
Officer	政府官员
Royalty	王室（皇室）
Mr	已婚男士
Mrs	已婚妇女
Miss	年轻未婚女子
Master	有技能的人/教师

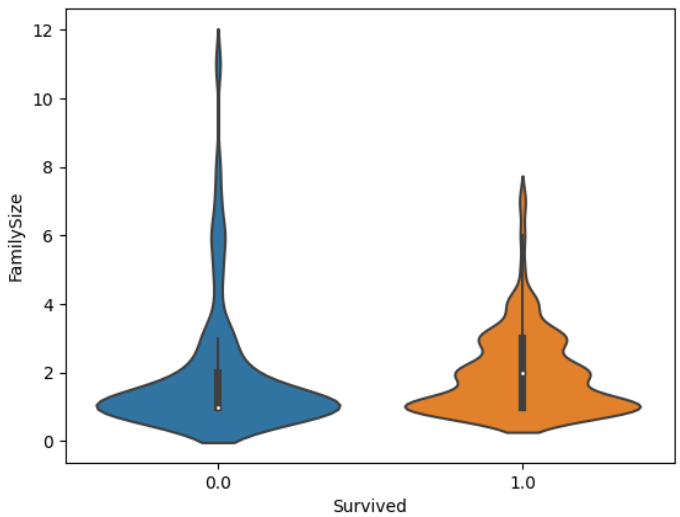
从图 3-4 可以大致看出，Pclass 等级越高，存活率越高；具有 Mrs 和 Miss 头衔的乘客存活率更高；女性比男性的存活率更高；不同年龄段乘客的存活率相差不大；SibSp 和 Parch 与 Survived 的关系非常类似；Fare 较小时，死亡率高于存活率，Fare 较大时，存活率高于死亡率；Cabin 数据缺失时的死亡率最高；Embarked 取值为 S 的样本量最多，但取值为 C 时生存率最高。

考虑到 SibSp 和 Parch 与 Survived 的关系非常类似，本文创造了一个新的变量 FamilySize 代替这两个变量：

$$FamilySize = Parch + SibSp + 1$$

从 FamilySize 与 Survived 的关系可以大致看出（见图 3-5），当 FamilySize=1 时，死亡率大于生存率，当 2≤FamilySize≤4 时，生存率大于死亡率，当 FamilySize≥5 时，死亡率大于生存率。故根据 FamilySize 取值，将其划分为三类：Family_Small（小家庭），Family_Medium（中等家庭），Family_Large（大家庭）。另外，考虑到 Cabin 数据缺失字段较多，且 Cabin=U 时的死亡率最高，故本文进一步将 Cabin 划分为两类：0=缺失字段/1=非缺失字段。

图 3-5 FamilySize 与 Suvived 的关系



3.3 OneHotEncoder 编码

One-Hot 编码将已经转换为数值型的类别特征,映射为一个稀疏向量对象,对于某一个类别映射的向量中只有一位有效,即只有一位数字是 1,其他数字位都是 0。如现有两个特征属性:

Pclass:[“1”= 一等舱，“2”= 二等舱，“3”= 三等舱]

Sex:[“female”，“male”]

由于机器学习算法不接收字符型的特征值,需要将字符型分类值的特征数字化,对于某一个样本,如[“1”,“female”],最直接的方法可以采用序列化的方式:[1,0],但是这样的特征处理并不能直接放入机器学习算法中。对于类似 Pclass 状况是 3 维,有 Sex 是 2 维的问题，本文采用 One-Hot 编码的方式对上述样本[“1”,“female”]进行编码,“1”对应[1,0,0],“female”对应[1,0],则完整的特征数字化的结果为:[1,0,0,1,0]。故本文将 Pclass, Title, Sex, FamilySize, Cabin, Embarked 六个离散的特征变量进行 One-Hot 编码。同时,为避免 One-Hot 编码带来的多重共线性问题,本文将每一组 One-Hot 编码后的变量删除了一列。最终，共得到 15 个特征变量。

表 3-3 特征变量及描述

变量	描述
Survived	生存情况（1=存活/0=死亡）
Sex	性别（1=Male/0=Female）
Age	年龄
Fare	船票价格
Cabin	客舱号（1=数据缺失/0=未缺失）
Pclass_2	客舱等级（1=“Pclass 为 2”/0=“Pclass 不为 2”）
Pclass_3	客舱等级（1=“Pclass＝3”/0=“Pclass≠3”）
Title_Miss	头衔（1=“Title=Miss”/0=“Title≠Miss”）
Title_Mr	头衔（1=“Title=Mr”/0=“Title≠Mr”）
Title_Mrs	头衔（1=“Title=Mrs”/0=“Title≠Mrs”）
Title_Officer	头衔（1=“Title=Officer”/0=“Title≠Officer”）
Title_Royalty	头衔（1=“Title=Royalty”/0=“Title≠Royalty”）

Family_Small	家庭大小（1=“FamilySize=1”/0=“FamilySize≠1”）
Family_Large	家庭大小（1=“FamilySize≥5”/0=“FamilySize<5”）
Embarked_Q	登船港口（1=“Embarked=Q”/0=“Embarked≠Q”）
Embarked_S	登船港口（1=“Embarked=S”/0=“Embarked≠S”）

3.4 特征选取

根据图 3-6 中展示的变量之间的相关关系，以及表 3-4 中展示的各变量与 Survived 的相关系数，可以看出与 Title_Mrs 和 Title_Miss 与 Survived 呈现较强的正相关性，Title_Mr 和 Sex 与 Survived 呈现较强的负相关性。因此，本文猜测泰坦尼克号遇险时可能执行的“*Lady first*”的逃生政策。

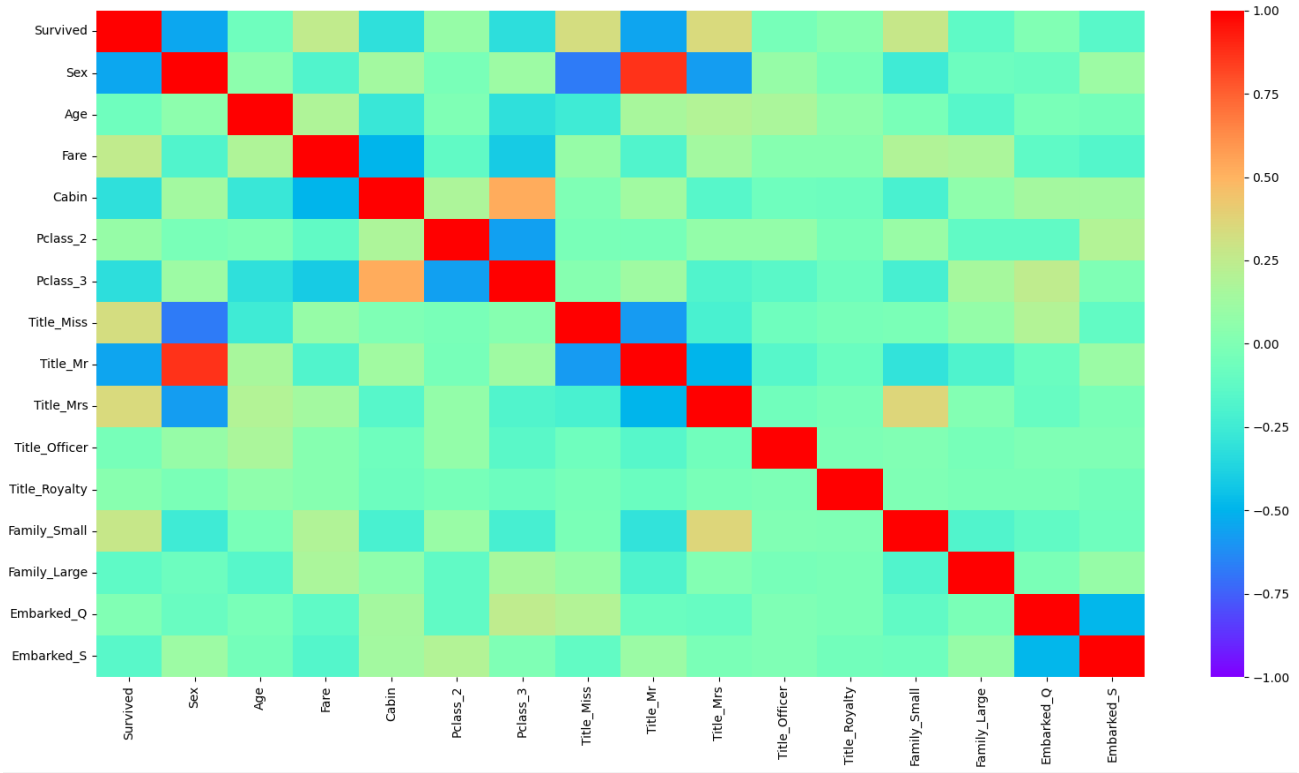
从表 3-4 中可以看出，Age 与 Survived 相关性较弱，本文将其该特征变量舍弃。同时，考虑到 Sex/ Fare/Cabin/Pclass/Title/FamilySize/Embarked 这些特征变量均至少存在一种状态与 Survived 有较强的相关关系，且本次分析的最终目的是进行预测，本文将表 3-4 中的所有变量均纳入后续的模型。

表 3-4 各变量与 Survived 的相关系数

变量	与 Survived 的相关系数
Survived	1.000000
Title_Mrs	0.344935
Title_Miss	0.332795
Family_Small	0.279855
Fare	0.257307
Pclass_2	0.093349
Title_Royalty	0.033391
Embarked_Q	0.003650
Title_Officer	-0.031316
Age	-0.064910
Family_Large	-0.125147
Embarked_S	-0.149683
Cabin	-0.316912

Pclass_3	-0.322308
Sex	-0.543351
Title_Mr	-0.549199

图 3-6 变量间相关系数



四、模型选择与评估

4.1 模型选择

由于标签 Survived 取值为 0 和 1（0=死亡，1=存活），本问题是一个二分类问题。因此，本文选择逻辑回归和随机森林进行拟合和预测。

逻辑回归是一种监督式学习的分类回归算法，通过 Logistic 函数(Sigmoid 函数)，将数据特征映射到[0,1]区间的一个概率值（样本属于正例的可能性），再与分类阈值比较得出样本所属的分类。逻辑回归适用于需要计算分类概率的问题，且可根据求出的参数解释自变量对因变量的影响；但是当特征变量较多时，逻辑回归的拟合效果往往不好，且逻辑回归依赖没有数据缺失情况的数据集。

$$y = \frac{1}{1+e^{-z}}$$

$$z = w^T x + b$$

（上式中，x 表示模型所选取的特征变量；w 表示模型参数；b 表示偏置项）

随机森林是将多棵决策树集成的组合分类模型。它通过自助法(Bootstrap)重采样技术，从原始训练样本集 N 中有放回地重复随机抽取 m 个样本生成新的训练样本集合，然后根据自助样本集生成 m 个分类树组成随机森林，新数据的分类结果按分类树投票多少形成的分数而定。随机森林模型简单直观，往往容易实现，并且单个决策树可以并行训练；但是随机森林模型的可解释性不如逻辑回归模型。

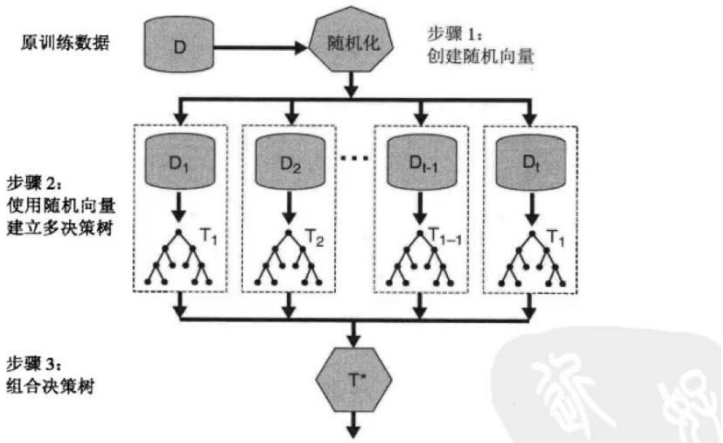
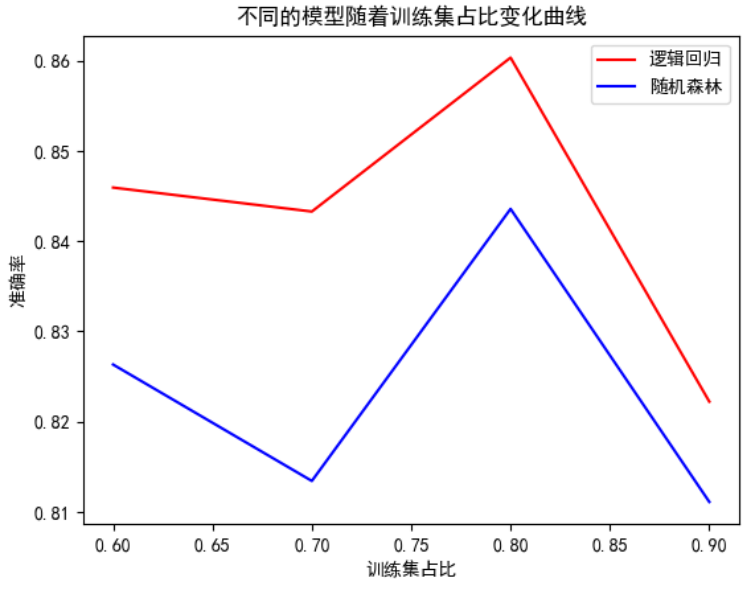


图 5-40　随机森林

（图片来源：《数据挖掘导论（完整版）》Pang-Ning Tan 等）

为比较两个模型的拟合效果，本文从训练集中随机抽取不同比例的样本来训练模型，并分别比较两个模型的准确率（见图 4-1）。从图 4-1 可以看出，在本文的数据集中，逻辑回归模型的效果优于随机森林模型。因此，本文选择逻辑回归模型进行后续的分析。

图 4-1 不同模型准确率随训练集占比变化曲线



4.2 模型调参

调参主要是为了防止过拟合问题。本文在分析过程中运用 GridSearchCV 网格搜索寻找逻辑回归模型的最佳参数，最终结果如下：

C = 100

class_weight = None

max_iter = 100

Solver = 'lbfgs'

其中，C 为正则化系数 λ 的倒数；class_weight 表示分类模型中各类型的权重；max_iter 表示最大迭代次数；solver 表示逻辑回归模型损失函数的优化方法。

4.3 模型评估

基于 4.2 中选择的最优参数，本节对模型的预测效果进行评估。对于逻辑回归模型预测结果的评估，通常会计算出混淆矩阵(Confusion Matrix，见表 4-1)，并根据混淆矩阵进一步计算出型拟合的准确率(Accuracy)、召回率(Recall)、精确率(Precision)以及 F1-Score：

表 4-1 混淆矩阵

		真实值	
		Positive	Negative
预测值	Positive	TP=485	FP=64
	Negative	FN=89	TN=253

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = 0.8282$$

$$\mathrm{Re}call = \frac{TP}{TP + FN} = 0.8215$$

$$\mathrm{Pr}ecision = \frac{TP}{TP + FP} = 0.8116$$

$$F1\text{-}Score = \frac{2 * \mathrm{Pr}ecision * \mathrm{Re}call}{\mathrm{Pr}ecision + \mathrm{Re}call} = 0.8158$$

其中，准确率(Accuracy)表示模型预测正确的样本占总样本的比例；召回率(Recall)表示真实值为 Positive 的样本中，模型预测正确的样本比例；精确率(Precision)表示预测值为 Positive 的样本中，模型预测正确的样本比例；F1-Score 则是一个综合 Precision 和 Recall 的评估指标，F1-Score 取值范围在

[0,1]内，其取值越接近 1 代表模型的拟合效果越好。

本文在分析过程中将上述评估方式打包在类 Performance 中，便于调用。类 Performance 定义如下：

```
class Performance:
    def __init__(self, source, pred):
        self.source = source
        self.pred = pred

    def cm(self):
        self.confusion = confusion_matrix(self.source, self.pred)
        disp = ConfusionMatrixDisplay(confusion_matrix=self.confusion)
        disp.plot()
        plt.show()

    def classify(self):
        self.accuracy = accuracy_score(self.source, self.pred)
        self.precision = precision_score(self.source, self.pred, average='macro')
        self.recall = recall_score(self.source, self.pred, average='macro')
        self.f1 = f1_score(self.source, self.pred, average='macro')
        classify = pd.DataFrame(data=[self.accuracy, self.precision, self.recall, self.f1],
                                index=['Accuracy','Precision','Recall','F1-score'],
                                columns = ['Logistic Regression'])
        print(classify)
```

最后，将测试集的预测值提交到 Kaggle 平台，得到的准确率为 0.75358。

五、项目链接

Hhh2324/-final: 编程基础 final project (github.com)