

基于随机森林和网格搜索参数调优的 泰坦尼克号生存者预测分析

摘要

本文以泰坦尼克号事件 2000 余名乘客的多因子数据集作为研究对象，旨在通过该数据集实现对数据集中乘客的生存情况的分析和预测。

对于数据集，首先进行了数据集因子的初步情况分析，得出所有变量的大致分布和缺失情况。随后进行数据的预处理操作。对非数值类型变量进行 one-hot 编码转换处理，进行缺失值的插补，其中使用了均值填补，随机森林算法填补等方法，得出最终的数据整体属性情况。在数据处理完成后，使用 pandas 库中的 Corr 自带库函数求得各变量关于 survived 属性的相关系数，通过选取相关系数较大的 8 个变量组成模型数据集。随后进行模型训练，经过多个模型的对比以及模型融合的使用，最终选择了效果最好的随机森林算法。在确定算法后，使用网格搜索进行参数的调优，最后得出最终的模型，使用该模型进行预测，最得出 test 集中的乘客的生存情况。

关键词：多因子；one-hot 编码；数据预处理；pandas 库；相关系数；随机森林；网格搜索；参数调优

abstract

This paper takes the multi factor data set of more than 2000 passengers in the Titanic incident as the research object, and aims to analyze and predict the survival situation of passengers in the data set through the data set.

For the data set, the preliminary analysis of data set factors is carried out, and the general distribution and missing of all variables are obtained. Then, the non numerical type variables are converted into one hot code to get the final data overall attribute. Then the data preprocessing operation is carried out, and the missing values are interpolated, in which the mean filling, random forest algorithm filling and other methods are used. After the data processing is completed, the correlation coefficient of each variable about the survived attribute is obtained by using the function of corr in pandas library. Eight variables with larger correlation coefficient are selected to form the model data set. Then, the model training is carried out. After the comparison of several models and the use of model fusion, the best random forest algorithm is selected. After the algorithm is determined, the grid search is used to optimize the parameters, and the final model is obtained. The survival situation of the passengers in the test set is obtained by using the model for prediction.

Keywords: Multiple factors;One hot coding;Data preprocessing;pandas database;correlation coefficient; Randomforest; Gridsearch; Parameter tuning

目录

1. 数据集因子分析	4
1.1 下载数据	4
1.2 导入数据并查看大小	4
1.3 查看具体数据信息	4
2. 数据预处理	5
2.1 对 Age 字段的预处理	5
2.2 对 Fare 字段的预处理	5
2.3 对 Embarked 字段的预处理	7
2.4 对 Cabin 字段的预处理	7
3. 特征工程	8
3.1 提取特征	8
3.2 特征数据选择	9
4. 模型训练及调优	10
4.1 数据集拆分	10
4.2 构建模型	10
4.3 网格搜索调优	10
4.4 预测实施	10
5. 附录	11
5.1 程序源代码	11
5.2 成绩证明	11
5.3 设计运行说明	11
5.3.1 输入说明	11
5.3.2 输出说明	11
5.3.3 注意事项	11

1. 数据集因子分析

1.1 下载数据

数据地址: www.kaggle.com/c/titanic/datasets

1.2 导入数据并查看大小

```
train=pd.read_csv('train.csv')
test=pd.read_csv('test.csv')
print('训练数据集: ',train.shape)
print('测试数据集: ',test.shape)
```

```
训练数据集: (891, 12)
测试数据集: (418, 11)
```

1.3 查看具体数据信息

```
full=train.append(test,ignore_index=True)
full.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1309 entries, 0 to 1308
Data columns (total 12 columns):
#   Column          Non-Null Count  Dtype
---  -
0   PassengerId     1309 non-null   int64
1   Survived        891 non-null    float64
2   Pclass         1309 non-null   int64
3   Name            1309 non-null   object
4   Sex             1309 non-null   object
5   Age             1046 non-null   float64
6   SibSp           1309 non-null   int64
7   Parch           1309 non-null   int64
8   Ticket          1309 non-null   object
9   Fare            1308 non-null   float64
10  Cabin           295 non-null    object
11  Embarked        1307 non-null   object
dtypes: float64(3), int64(4), object(5)
memory usage: 122.8+ KB
```

通过观察发现,数据集包含 1308 行数据信息,12 个序列。其中 Age 字段缺失 263 个,Cabin 字段缺失 1012 个,Embarked 字段缺失 2 个,Fare 字段缺失 1 个。

2. 数据预处理

2.1 对 Age 字段的预处理

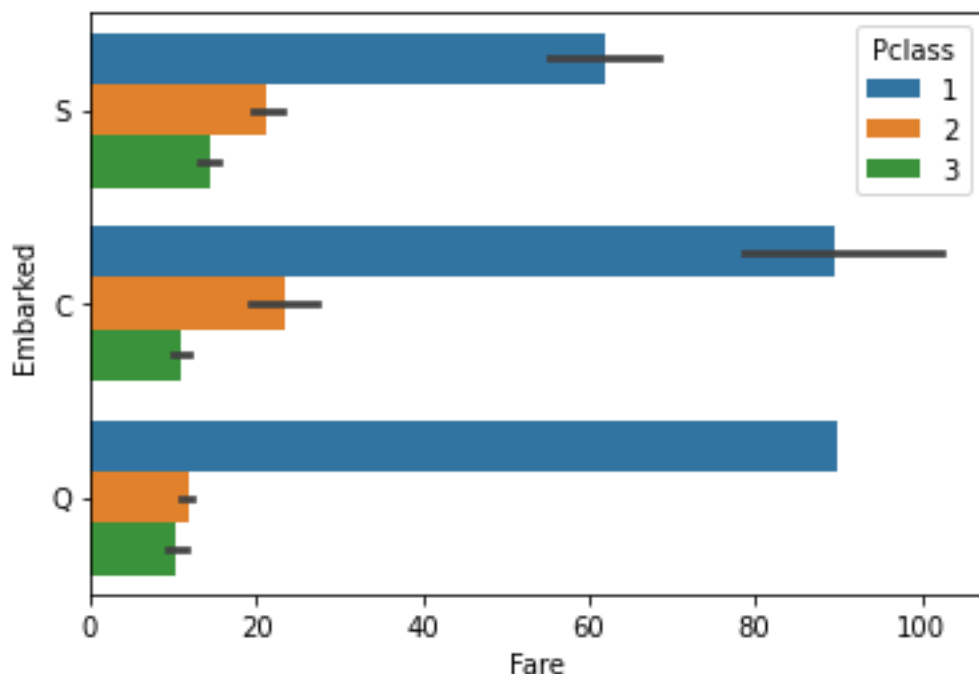
首先通过第一步的分析可以发现,Age 的缺失情况是较为严重的。根据常识,Age 这部分信息即使大量缺失,也需要进行算法填补,否则对预测结果的影响将不可估量。通过观察,Name 字段中的称呼与乘客的年龄具有十分大的关系,于是考虑使用 Name 字段进行机器学习填补 Age 字段的缺失值。

其次通过第一步的变量分析发现,Sex 和 Pclass 与 Age 也有一定的关系。于是综合考虑使用 Sex 字段,Pclass 字段和 Name 字段中的称谓字段(Mr,Mrs 等)进行随机森林算法预测出未知的 Age 字段。

2.2 对 Fare 字段的预处理

通过对 Cabin 字段的观察后会发现,Cabin 中出现多人一通同出行的情况,此时票为连续情况,Fare 字段表示的是总价。该现象会直接影响 Fare 字段的离散程度,因此应先对 Fare 字段进行处理,即对该情况的票价 Fare 除以船舱(Cabin)数量。

在初步处理完成后，考虑与 Fare 有较大关系的字段可能有 Pclass, Cabin, Embarked。故先做出 Pclass, Embarked 和 Fare 的关系图（如下图）。



通过观察图像发现，Fare 与 Pclass 关系较大，而与港口 Embarked 关系较小。

然后分析 Fare 和 Cabin 的关系。

Pclass	1	2	3			
Cabin						
A	41.244314	NaN	NaN			
B	86.671903	NaN	NaN			
C	88.730809	NaN	NaN			
D	57.335108	13.595833	NaN			
E	61.486765	11.587500	11.000000			
F	NaN	23.423077	5.182294			
G	NaN	NaN	14.205000			
T	35.500000	NaN	NaN			
U	70.268157	21.394537	13.351522			
PassengerId	Survived	Pclass	Embarked	title	Multi_Cabin_counts	
1043	1044	NaN	3	S	Mr	0

发现 Fare 缺失只有一个，于是使用均值直接填补。至此 Fare 字段预处理完毕。

2.3 对 Embarked 字段的预处理

在第一步对变量的观察时发现，Embarked 字段的缺失值只有两个，对整体准确率影响很小，于是直接使用众数填补即可。

```
[1 rows x 13 columns]
S      914
C      270
Q      123
Name: Embarked, dtype: int64
```

可以观察到，出现最多的 S 港占比为 69.8%，于是直接使用 S 进行填补。

2.4 对 Cabin 字段的预处理

在第一步的观察中发现，Cabin 字段的缺失极为严重。此处同样需要对 Cabin 字段进行机器学习方法填补。与 Age 字段的填补类似，考虑与 Cabin 相关的参数可能有 Embarked, Fare, Pclass，随后进行随机森林算法的预测。与 Age 字段不同的是，Cabin 字段为非数值类型，需要进行编码操作。填补完成后，将生成的字段加入总数据集中，完成 Cabin 字段的处理。

至此全部数据预处理完毕，观察一下结果。

```
Data columns (total 13 columns):
#      Column      Non-Null Count  Dtype
---  -
0      PassengerId  1309 non-null    int64
1      Survived      891 non-null     float64
2      Pclass         1309 non-null    int64
3      Sex           1309 non-null    object
4      Age           1309 non-null    float64
5      SibSp         1309 non-null    int64
6      Parch         1309 non-null    int64
7      Ticket        1309 non-null    object
8      Fare          1309 non-null    float64
9      Cabin         1309 non-null    object
10     Embarked       1309 non-null    object
11     title         1309 non-null    object
12     Multi_Cabin_counts 1309 non-null    int64
dtypes: float64(3), int64(5), object(5)
memory usage: 133.1+ KB
```

3. 特征工程

3.1 提取特征

通过数据预处理后本题特征共有三类。其中数值数据: "Age"、"Fare"、"PassengerId"; 分类数据: "Cabin"、"Embarked"、"Parch"、"Pclass"、"SibSp"、"Sex"; 字符串类: "Ticket"、"title"。

对于数值类数据可以无需处理。对于分类数据和字符串类数据, 可以用 one_hot 编码将其转换为二元特征的数值数据。注意到 ticket 字段在此意义不大, 故不进行使用。本次的独热编码采用了 pandas 库自带的 get_dummies 函数。

3.2 特征数据选择

在上一步完成后，现在的工作为选择出和 Survived 字段有关的特征数据。本次使用了 pandas 模块里面的 corr 方法对数据集中所有元素之间的相关性进行回归分析，再将与 Survived 字段相关性较大的特征提取出来进行模型构建。

```
Survived      1.000000
title_Mr      0.549199
Sex           0.543351
title_Mrs     0.341994
title_Miss    0.332795
Pclass_3      0.322308
Pclass_1      0.285904
Fare          0.267721
Family_small  0.238059
Family_single 0.203367
Embarked_C    0.168240
Cabin_B       0.167063
Cabin_E       0.166764
Embarked_S    0.149683
Cabin_C       0.135742
Cabin_D       0.135064
Cabin_G       0.123758
Pclass_2      0.093349
Parch         0.081629
title_Master  0.079996
Multi_Cabin_counts 0.072724
Age           0.072263
title_Royalty 0.050561
title_Ms      0.042470
SibSp         0.035322
title_Officer 0.031316
Family_large  0.029945
Cabin_T       0.026456
Family_size   0.016639
Cabin_A       0.011296
PassengerId   0.005007
Embarked_Q    0.003650
Cabin_F       0.002612
Name: Survived, dtype: float64
```

通过观察, 选取了前 8 个相关性较大的特征进行模型构建, 即 `title_Df`, `full['Sex']`, `Pclass_Df`, `full['Fare']`, `Family_size_Df`, `Embarked_Df`, `Cabin_Df`, `full['Age']`

4. 模型训练及调优

4.1 数据集拆分

按照惯例, 需要对数据集进行拆分, 形成训练数据集和测试数据集。利用 `sklearn.model_selection` 模块中的 `train_test_split()` 方法遵照二比八的原则对元素进行拆分。

4.2 构建模型

本文采取了随机森林的学习方法, 建立了预测模型。

```
model=RandomForestClassifier(random_state = 10, warm_start = True, n_estimators = 26, max_depth = 6, max_features = 'sqrt')
```

4.3 网格搜索调优

本文考虑到随机森林的参数可能不是最佳, 所以引入了网格搜索对模型进行参数遍历调优, 最终选出最佳参数返回给模型。

```
搜索时间: 562.62
{'max_depth': 8, 'max_features': 'sqrt', 'n_estimators': 20, 'random_state': 35, 'warm_start': True}
```

4.4 预测实施

在一切工作完成之后, 进行模型的 Pre 处理, 调用 `predict` 函数进行预测, 将所得结果上传至 Kaggle 网站,

最终成绩排名在 8%左右。最终结果集见文件 try2.csv, 成绩证明和设计运行说明见附录。

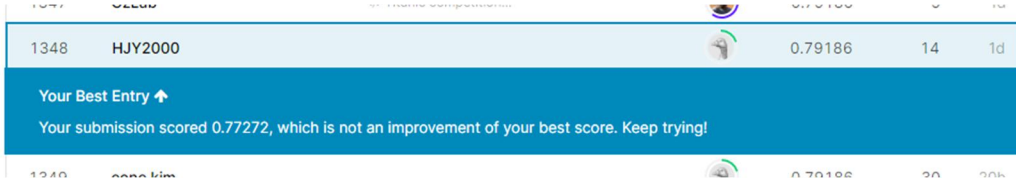
5. 附录

5.1 程序源代码



Titanic.py

5.2 成绩证明



1348	HJY2000	0.79186	14	1d
Your Best Entry ↑ Your submission scored 0.77272, which is not an improvement of your best score. Keep trying!				

1375-16814 + Load 15440 More

成绩排名：1318/16814=8.02%

5.3 设计运行说明

5.3.1 输入说明

输入文件为 train.csv, test.csv。分别为训练数据和待测试数据，都存放于 titanic 文件夹中

5.3.2 输出说明

输出结果为一个 CSV 文件，结果为最终的预测结果，文件名为 try2.csv, 存放于 titanic 文件夹中

5.3.3 注意事项

Titanic.py 文件在引用路径时使用了默认当前路径，故需要保持程序文件和 输入文件在同一目录。