scikit-learn reports

資訊 114 F7401254 張暐俊

1. 改善決策樹分類模型

- **增加更多的輸入特徵**

```
# 取出訓練資料需要分析的資料欄位
df_x = df_train[['Sex', 'Age', 'Fare']]
# 取出訓練資料的答案
df_y = df_train['Survived']
```

上圖為助教在 hint 中範例程式碼使用的三個特徵，而我在閱讀完 train.cv 以及在搜索每個欄位上方英文字幕代表意義時，找到網路上有人分析好的相關資料，我決定將我認為會提高準確率的特徵加入。

```
[12] # 取出訓練資料需要分析的資料欄位
     df_x = df_train[['Pclass', 'Sex', 'Age', 'Fare', 'Embarked']]
     # 取出訓練資料的答案
     df_y = df_train['Survived']
```

一開始我只有選擇 Pclass，因為我先入為主的認為船艙的艙等對於存活率肯定是有影響的，但結果預測的結果打了我的臉，準確率並沒有太大的變化。但有趣的是，在我看完網路上他人分析的資料後，我再加入 Embarked，也是就是登船的地點，配合著等等會說明的前處理，我的準確率從 0.72 來到了 0.759。　此外，我也有嘗試加入 SibSp 跟 Parch 兩筆特徵，但反倒使我的準確率下降了，無論是兩者同時加入，抑或是個別加入 SibSp 或 Parch。

- **使用不同的前處理方法**

在加入不同特徵前，我先摸索的正是前處理方式，因為我在閱讀完 train.csv 檔案之後，我發現裡面的 data 有的有缺失，有的沒辦法直接做運算，有的需要處理完之後實作上才會更方便。我首先下手的是「age」的前處理。根據助教提供的四個 strategy 去做調整，結論是助教使用的 median 是最佳的選擇了，使用其他的話，準確率皆會降至 0.72 以下。那原因應該是因為乘客的年齡分佈並非那麼的集中，有許多的極端值，像是高齡老人或是嬰兒，也因此相較於 mean 來說，median 或 most_frequency 是來得較好，但在實測後，我依舊維持原樣，選擇了 median。

```python
# 數值型態資料前處理
# 創造 imputer 並設定填補策略
imputer = SimpleImputer(strategy='median')
age = df_x['Age'].to_numpy().reshape(-1, 1)
# 根據資料學習需要填補的值
imputer.fit(age)
# 填補缺失值
df_x['Age'] = imputer.transform(age)
```

接著我開始著手摸索類別型前處理，根據助教上課提到的方法有 label encoder 跟 one-hot encoder，在研究了一下後，我決定使用 one-hot encoder 來做替換，因為我認為在性別上應該是平等的，可是使用 label encoder 的話，可能會造成產生距離的問題，也讓我推估是造成準確率無法提高的原因。在更改之後，有使得我的準確率從 0.72 提高至 0.73，正式突破 0.72 大關。

接著我加入了 Pclass 跟 embarked 兩個欄位，因為 Pclass 已經是以數字顯示，所以我只有著手 embarked 的部分，這次我則選擇 label encoder，因為根據前述，我在閱讀資料時發現，登船地點的不同的確是影響生存率的原因之一，所以我希望他們本身之間含有距離關係。

```python
df_train['Embarked'] = df_train['Embarked'].fillna('S')
# 創造 Label Encoder
le = LabelEncoder()
# 給予每個類別一個數值
le.fit(df_x['Embarked'])
# 轉換所有類別成為數值
df_x['Embarked'] = le.transform(df_x['Embarked'])
```

- **調整超參數**

```
def __init__(*, criterion='gini', splitter='best', max_depth=None,
min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0,
max_features=None, random_state=None, max_leaf_nodes=None,
min_impurity_decrease=0.0, class_weight=None, ccp_alpha=0.0)
在分頁中開啟   查看原始碼
A decision tree classifier.
```

最後則是研究超參數，這邊我並沒有多做探索，我選擇增加的只有上圖的 criterion、max_depth 跟 max_leaf_node，相較於助教在課堂上使用的 gini，我選擇使用 entropy，相較於 gini 的隨機性來說，entropy 的選擇更傾向於平均值，但他同樣有缺點，就是他的樹不能太深，否則其缺點同樣會很明顯。

因此在調整並測驗後，我選 depth = 3 當作我的樹深，我的 test_accuracy 則是來到了新高的 0.81。

```python
# 創造決策樹模型
model = DecisionTreeClassifier(
    random_state=1012,
    criterion='entropy',
    max_depth = 3,
    max_leaf_nodes = 2 ** 3 )
# 訓練決策樹模型
model.fit(train_x, train_y)
```

```
train accuracy: 0.8300561797752809
test accuracy: 0.8156424581005587
```

- **額外發現**

隨著調整上方三項，我的 train_accuracy 也會隨之改變，尤其在我調整模型的超參數時最為明顯，也因此我有在想，儘管我使得我的 test_accuracy 來到了 0.81 隻高，會不會我在選擇 max_depth = 4 時，儘管 test_accuracy 略低，但是 train_accuracy 的數值反而較高，這樣子比較好呢？或許在兩者之間取得平衡，也是在調整這些參數或是方法時，需要去注意並思考的一環。

```python
# 創造決策樹模型
model = DecisionTreeClassifier(
    random_state=1012,
    criterion='entropy',
    max_depth = 4,
    max_leaf_nodes = 2 ** 4 )
# 訓練決策樹模型
model.fit(train_x, train_y)
```

```
train accuracy: 0.8441011235955056
test accuracy: 0.7932960893854749
```

2. 使用不同模型

- 測試結果

模型類別	模型名稱	Train Accuracy	Test Accuracy
sklearn.naive_bayes	GaussianNB	0.77528	0.80446
	BernoulliNB	0.78370	0.79888
sklearn.svm	LinearSVC	0.77528	0.77653
	NuSVC	0.79353	0.80446
	SVC	0.78370	0.79888
sklearn.ensemble	RandomForestClassifier	0.91573	0.79329
	ExtraTreesClassifier	0.83707	0.79329
	GradientBoostingClassifier	0.90308	0.76536
	AdaBoostClassifier	0.81179	0.81005
sklearn.neural_network	MLPClassifier	0.77668	0.81005
sklearn.linear_model	SGDClassifier	0.68398	0.74301
	LogisticRegression	0.77808	0.80446

3. 參考資料

- 机器学习：泰坦尼克数据分析

  https://zhuanlan.zhihu.com/p/447348987
- Decision Trees Explained — Entropy, Information Gain, Gini Index, CCP Pruning

  https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c
- Different Types of Distance Metrics used in Machine Learning

  https://medium.com/@kunal_gohrani/different-types-of-distance-metrics-used-in-machine-learning-e9928c5e26c7
- Titanic Machine Learning by k-nearest neighbors (KNN) algorithm

  https://medium.com/analytics-vidhya/titanic-machine-learning-

by-k-nearest-neighbors-knn-algorithm-530d8bdd8323

- 機器學習_學習筆記系列(69)：K 維樹(KD Tree)

  https://tomohiroliu22.medium.com/機器學習-學習筆記系列-69-k 維樹-kd-tree-b3b8591c9245

- 機器學習_學習筆記系列(70)：球樹(Ball Tree)

  https://tomohiroliu22.medium.com/機器學習-學習筆記系列-70-球樹-ball-tree-7bccb54cbf1e

- [資料分析&機器學習] 第 4.1 講 ：Kaggle 競賽-鐵達尼號生存預測-(前16%排名)

  https://medium.com/jameslearningnote/資料分析-機器學習-第 4-1講-kaggle 競賽-鐵達尼號生存預測-前 16-排名-a8842fea7077

- [資料分析&機器學習] 第 3.4 講：支援向量機(Support Vector Machine)介紹

  https://medium.com/jameslearningnote/資料分析-機器學習-第 3-4講-支援向量機-support-vector-machine-介紹-9c6c6925856b

- scikit-learn 中文社区

  https://scikit-learn.org.cn

- scikit-learn
  https://scikit-learn.org/stable/index.html

- 樸素貝葉斯（數值範例）

  https://medium-com.translate.goog/@balajicena1995/naive-bayes-numerical-example-

afcfa2433f95?_x_tr_sl=en&_x_tr_tl=zh-TW&_x_tr_hl=zh-TW&_x_tr_pto=sc&_x_tr_hist=true