Data Mining Research and Practice HW0

Department: IAM Student ID: 309652008 Name: 廖家緯

September 26, 2021

1 讀取資料

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

Figure 1. data frame

- PassengerId: 乘客編號
- Survived: 1 (存活)、0 (死亡)
- Pclass: 票務艙
- Name: 姓名
- Sex: 性別
- Age: 年齡
- SibSp: 在船上的兄弟姊妹、配偶人數
- Parch: 在船上父母和子女人數
- Ticket: 船票號碼
- Fare: 船票價格
- Cabin: 船艙號碼
- Embarked: 登入港口

```
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Figure 2. train data information

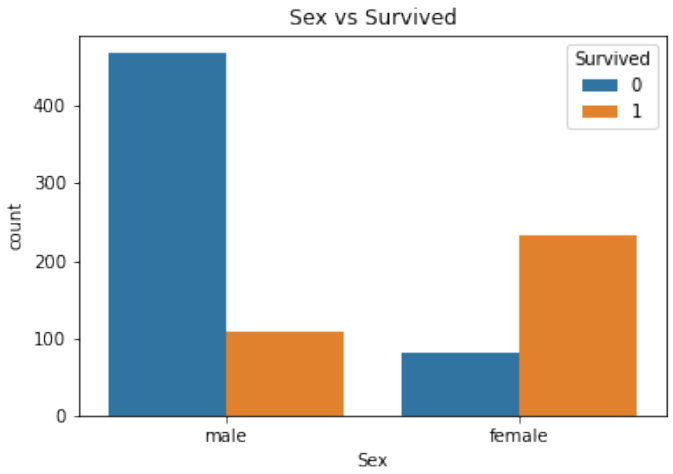
```
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  418 non-null    int64
 1   Pclass       418 non-null    int64
 2   Name         418 non-null    object
 3   Sex          418 non-null    object
 4   Age          332 non-null    float64
 5   SibSp        418 non-null    int64
 6   Parch        418 non-null    int64
 7   Ticket       418 non-null    object
 8   Fare         417 non-null    float64
 9   Cabin        91 non-null     object
 10  Embarked     418 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

Figure 3. test data information

2 資料分析與視覺化

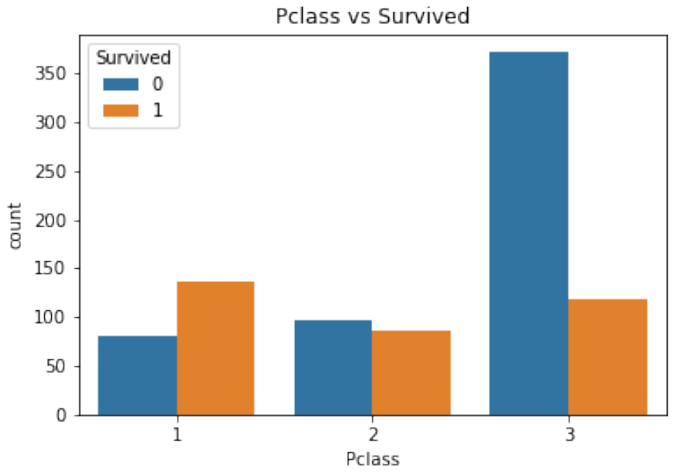
以下將不同類別的生存及死亡人數做視覺化。

2.1 性別



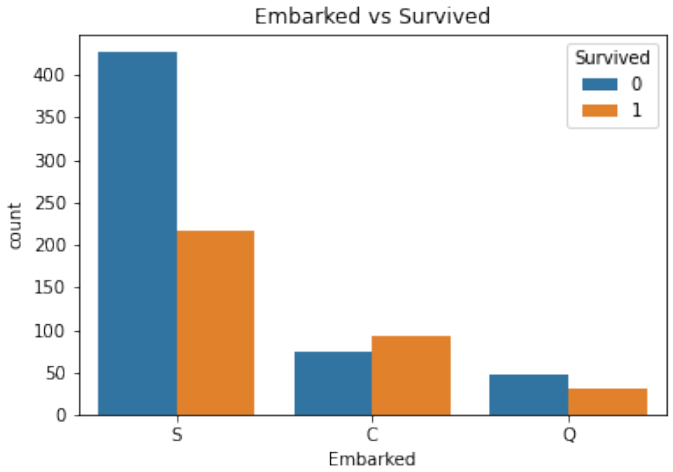
從上圖我們可以發現男生的存活人數較低，而女生存活人數較高 (可能在危難之際，男生會選擇保護女生 XD)。

2.2 票務艙



從上圖我們可以發現傳票艙 3 的乘客死亡人數高很多，因此為一個重要資訊。

2.3 登入港口



從上圖我們可以發現登入港口為 S 的死亡人數偏高。

3 資料前處理 (特徵工程)

3.1 填補缺失值

從 Figure 2. 和 Figure 3. 可知，年齡 (Age)、票價 (Fare)、船艙號碼 (Cabin) 有缺失資料，而年齡與票價屬於數值型資料，可將缺失資料填補中位數；船艙號碼屬於類別型資料，且缺失資料超過一半，因此考慮直接捨棄。

3.2 特徵轉換

許多模型須將類別型資料轉換成數值型資料才能使用，因此這裡將性別 (Sex) 與登入港口 (Embarked) 轉換成整數。

3.3 加入新特徵

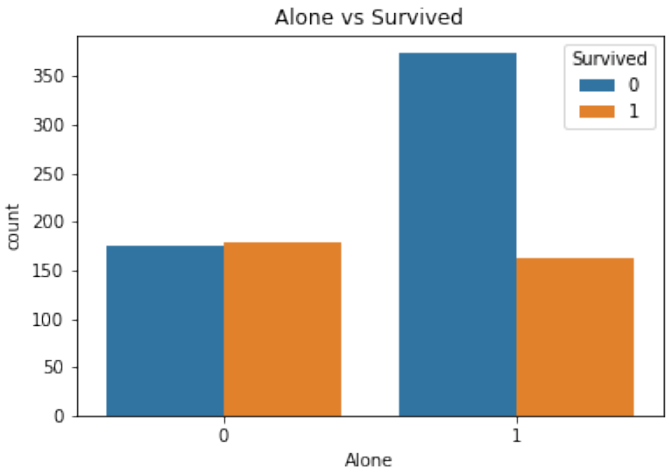
我們可以根據原始資料提供的線索，定義以下新的特徵:

- 家庭大小為在船上的父母、子女、兄弟姊妹、配偶人數再加上自，因此可以表示為

$$\text{Family Size} = \text{SibSp} + \text{Parch} + 1$$

- 我們可以再根據家庭大小判斷該乘客是否為孤單一人，因此定義

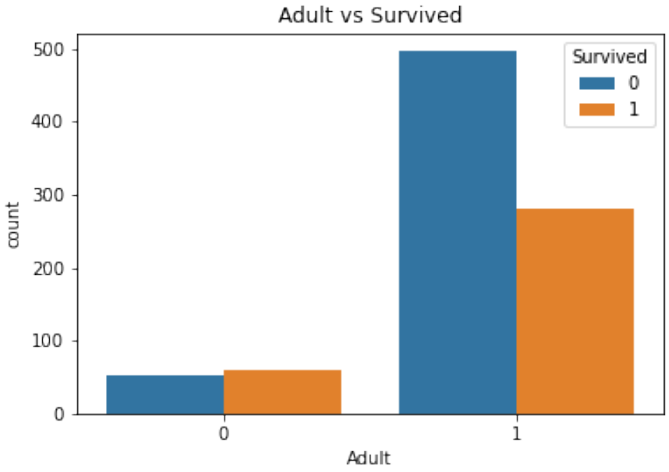
$$\text{Alone} = \begin{cases} 1 & \text{if Family Size} > 1 \\ 0 & \text{otherwise} \end{cases}$$



從上圖我們可以發現孤單一人的死亡人數較存活人數高。

- 我們可以根據年齡判斷乘客是否已成年，因此定義

$$\text{Adult} = \begin{cases} 1 & \text{if Age} \geq 18 \\ 0 & \text{otherwise} \end{cases}$$



從上圖我們可以發現成年人的死亡人數較存活人數高。

3.4 程式碼

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier


class DataPreprocessing():
    def __init__(self, df):
        self.df = df.copy()

```

```
    def fill_missing_fare(self):
        self.df['Fare'] = self.df['Fare'].fillna(self.df['Fare'].median())

        return None

    def fill_missing_age(self):
        self.df['Age'] = self.df['Age'].fillna(self.df['Age'].median())

        return None

    def add_sex_code(self):
        self.df['Sex_Code'] = self.df['Sex'].map({'male': 0, 'female':
            1}).astype('int')

        return None

    def add_embarked_code(self):
        self.df['Embarked_Code'] = self.df['Embarked'].map({'S': 1, 'C': 2, 'Q':
            3}).fillna(0).astype('int')

        return None

    def add_family_size(self):
        self.df['Family_Size'] = self.df['SibSp'] + self.df['Parch'] + 1

        return None

    def add_alone(self):
        self.df['Alone'] = self.df['Family_Size'].map(lambda x: 0 if x>1 else 1)

        return None

    def add_adult(self):
        self.df['Adult'] = self.df['Age'].map(lambda x: 1 if x>=18 else 0)

        return None

    def feature_transform(self):
        self.fill_missing_fare()
        self.fill_missing_age()

        self.add_sex_code()
        self.add_embarked_code()
        self.add_family_size()
        self.add_alone()
        self.add_adult()

        return None

    def get_data(self, train=True):
        self.feature_transform()
        X = self.df[['Pclass', 'Fare', 'Sex_Code', 'Embarked_Code', 'Alone',
            'Adult']]
```

```
59
60          if train: return X, self.df['Survived']
61          else: return X
62
63
64  def save_predict(ID, prediction):
65      df = pd.DataFrame({'PassengerId': ID, 'Survived': prediction})
66      df.to_csv('submission.csv', index=False)
67
68      return None
```

4 模型預測

首先將資料分成 train data 及 validation data，並使用 sklearn 裡面的 RandomForestClassifier 進行訓練。我們可以參考 validation data 的準確度來調整特徵選取與特徵工程，最終選取 validation accuracy 為 0.84293 的模型做為我最終的選擇，最後將此模型對 test data 做預測，上傳至 Kaggle。

5 上傳 Kaggle 分數