[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)

Yeh James
Nov 7, 2017 · 8 min read

這篇文章要教大家如何利用最基礎、簡單的機器學習知識加上Random Forest(隨機森林)演算法就可以獲得Kaggle上鐵達尼號生存預測比賽在全世界前16%排名的成績(此範例程式在撰寫當下可獲得 1499的排名，總參加隊伍為 8882隊)，這篇教學文章為了讓新手能夠快速上手，省去了許多較為複雜的統計知識，若是有一定基礎的同學可再結合Kaggle討論區文章以及下方的參考閱讀獲得前10%的成績甚至是前5%。以正式比賽來說前10%就可以獲得一個銅牌成就，並可放在個人的履歷以及Linkedin上，對於想跨科系轉職資料科學家的人來說是一個大大加分的成就。

Competition Medals

Competition medals are awarded for top competition results. The number of medals awarded per competition varies depending on the size of the competition. Note that InClass, playground, and getting started competitions do not award medals.

	0-99 Teams	100-249 Teams	250-999 Teams	1000+ Teams
Bronze	Top 40%	Top 40%	Top 100	Top 10%
Silver	Top 20%	Top 20%	Top 50	Top 5%
Gold	Top 10%	Top 10	Top 10 + 0.2%*	Top 10 + 0.2%*

* (Top 10 + 0.2%) means that an extra gold medal will be awarded for every 500 additional teams in the competition. For example, a competition with 500 teams will award gold medals to the top 11 teams and a competition with 5000 teams will award gold medals to the top 20 teams.

Kaggle 獎牌資格

Your most recent submission

Name	Submitted	Wait time	Execution time	Score
submit.csv	35 minutes ago	0 seconds	0 seconds	0.79904

Complete

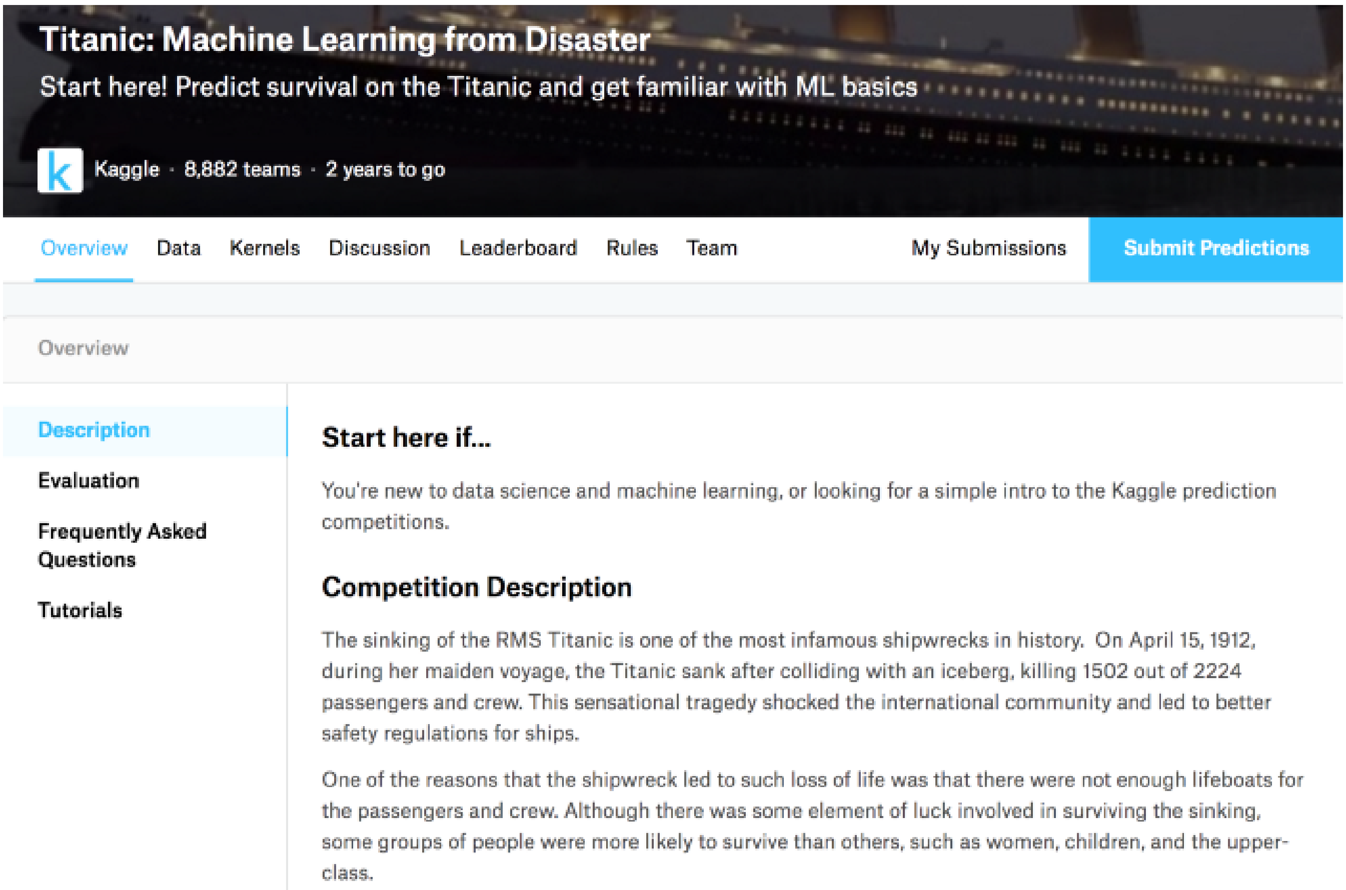
Jump to your position on the leaderboard ▾

範例程式分數

1496	▲3943	Prashanth S	0.79904	2	2h
1497	▲951	michele bernini	0.79904	20	35m
1498	new	Bayes_17214534	0.79904	16	1h
1499	▼195	Renard Korzeniowski	0.79425	1	2mo
1500	▼195	Jongwon Won	0.79425	14	20d
1501	▼195	Xavi Medina Torregrosa	0.79425	2	2mo
1502	▼195	davidjrm	0.79425	2	2mo
1503	▼195	Chengzhi Tan	0.79425	3	2mo
1504	▼195	2TBD - RM76240 RM76655 R...	0.79425	6	1mo
1505	▼195	2TBD RM76778	0.79425	5	1mo
1506	▼195	LinPeng	0.79425	1	2mo

0.79904成績為 1499/8882 大約為Top16%

首先介紹一下鐵達尼號生存預測這個比賽，你會拿到許多關於乘客的資訊像是乘客的性別、姓名、出發港口、住的艙等、房間號碼、年齡、兄弟姊妹＋老婆丈夫數量(Sibsp)、父母小孩的數量(parch)、票的費用、票的號碼這些去預估這個乘客是否會在鐵達尼號沈船的意外中生存下來。



In this challenge, we ask you to complete the analysis of what sorts of people were likely to survive. In particular, we ask you to apply the tools of machine learning to predict which passengers survived the tragedy.

鐵達尼號生存預測競賽介紹

Overview Data Kernels Discussion Leaderboard Rules Team My Submissions **Submit Predictions**

Data Dictionary

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

Variable Notes

pclass: A proxy for socio-economic status (SES)
1st = Upper
2nd = Middle
3rd = Lower

age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5

sibsp: The dataset defines family relations in this way...
Sibling = brother, sister, stepbrother, stepsister
Spouse = husband, wife (mistresses and fiancés were ignored)

parch: The dataset defines family relations in this way...
Parent = mother, father
Child = daughter, son, stepdaughter, stepson
Some children travelled only with a nanny, therefore parch=0 for them.

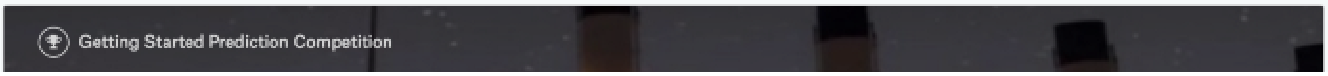
鐵達尼號生存預測競賽資料說明

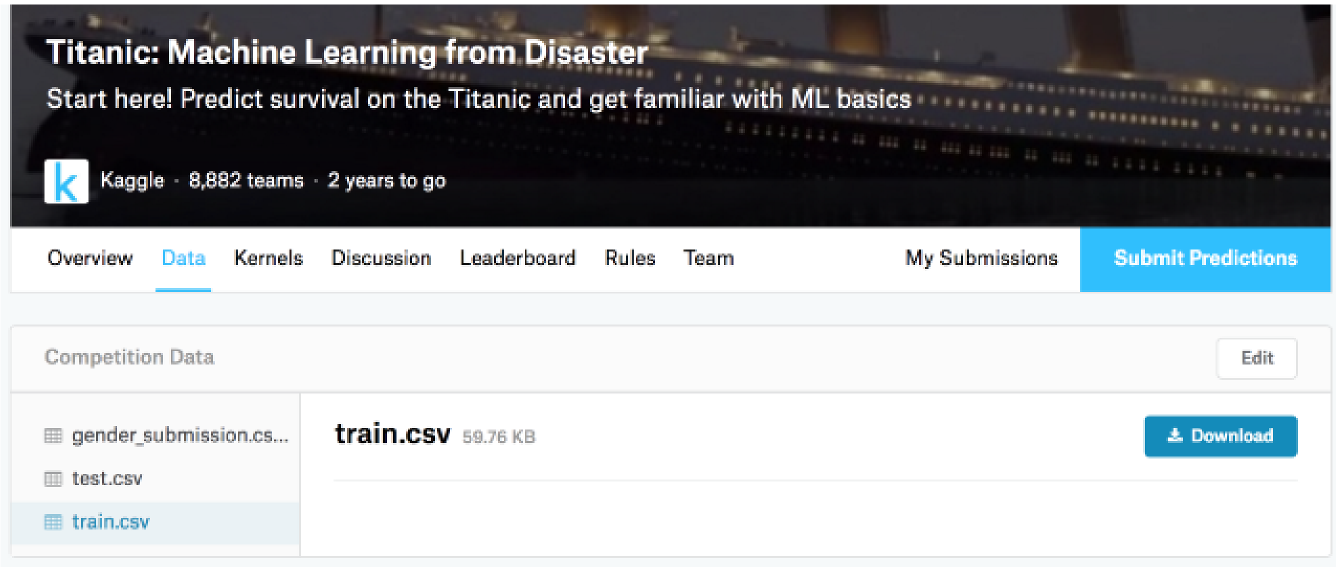
```
train.head(5)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

鐵達尼號生存預測競賽資料

首先先到Data頁面下載資料(test.csv, train.csv, gender_submission.csv)





使用Jupyter Notebook載入所需套件以及資料

```
from sklearn import preprocessing
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import RandomForestRegressor

import warnings
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
```

```
train = pd.read_csv("Titanic/train.csv")
test = pd.read_csv("Titanic/test.csv")
submit = pd.read_csv('Titanic/gender_submission.csv')
```

使用info()函式觀察train以及test資料是否有空值，發現train的Age,Cabin,Embark有空值以及Test的Age, Fare, cabin有空值的情況，之後我們要想辦法來補這些空值。通常比賽能夠準確預測的關鍵都是在如何補空值。

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
```

```
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

```
test.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
PassengerId    418 non-null int64
Pclass         418 non-null int64
Name           418 non-null object
Sex            418 non-null object
Age            332 non-null float64
SibSp          418 non-null int64
Parch          418 non-null int64
Ticket         418 non-null object
Fare           417 non-null float64
Cabin          91 non-null object
Embarked       418 non-null object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

使用Describe來觀察train以及test的資料分布

```
train.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

```
test.describe()
```

	PassengerId	Pclass	Age	SibSp	Parch	Fare
count	418.000000	418.000000	332.000000	418.000000	418.000000	417.000000
mean	1100.500000	2.265550	30.272590	0.447368	0.392344	35.627188
std	120.810458	0.841838	14.181209	0.896760	0.981429	55.907576
min	892.000000	1.000000	0.170000	0.000000	0.000000	0.000000

25%	996.250000	1.000000	21.000000	0.000000	0.000000	7.895800
50%	1100.500000	3.000000	27.000000	0.000000	0.000000	14.454200
75%	1204.750000	3.000000	39.000000	1.000000	0.000000	31.500000
max	1309.000000	3.000000	76.000000	8.000000	9.000000	512.329200

接下來由於要對整體資料做一些觀察，因此先將資料做合併

Combine Train and Test Data

```
data = train.append(test)
data
```

	Age	Cabin	Embarked	Fare	Name	Parch	PassengerId	Pclass	Sex	SibSp	Survived	Ticket
0	22.0	NaN	S	7.2500	Braund, Mr. Owen Harris	0	1	3	male	1	0.0	A/5 21171
1	38.0	C85	C	71.2833	Cumings, Mrs. John Bradley (Florence Briggs Th...	0	2	1	female	1	1.0	PC 17599
2	26.0	NaN	S	7.9250	Heikkinen, Miss. Laina	0	3	3	female	0	1.0	STON/O2. 3101282
3	35.0	C123	S	53.1000	Futrelle, Mrs. Jacques Heath (Lily May Peel)	0	4	1	female	1	1.0	113803
4	35.0	NaN	S	8.0500	Allen, Mr. William Henry	0	5	3	male	0	0.0	373450
5	NaN	NaN	Q	8.4583	Moran, Mr. James	0	6	3	male	0	0.0	330877
6	54.0	E46	S	51.8625	McCarthy, Mr. Timothy J	0	7	1	male	0	0.0	17463
7	2.0	NaN	S	21.0750	Palsson, Master. Gosta Leonard	1	8	3	male	3	0.0	349909
8	27.0	NaN	S	11.1333	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	2	9	3	female	0	1.0	347742
9	14.0	NaN	C	30.0708	Nasser, Mrs. Nicholas (Adele Achem)	0	10	2	female	1	1.0	237736
10	4.0	G6	S	16.7000	Sandstrom, Miss. Marguerite Rut	1	11	3	female	1	1.0	PP 9549
11	58.0	C103	S	26.5500	Bonnell, Miss. Elizabeth	0	12	1	female	0	1.0	113783

由於使用append合併之後會造成index重複問題，因此要將index重新設定

```
data.reset_index(inplace=True, drop=True)
```

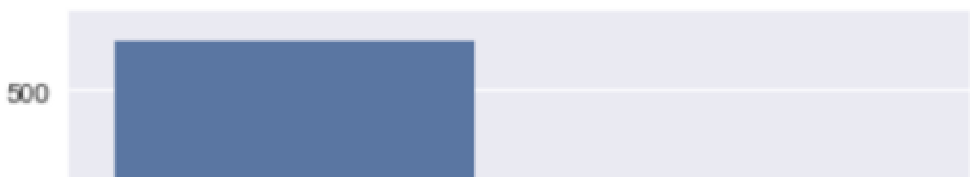
資料分析

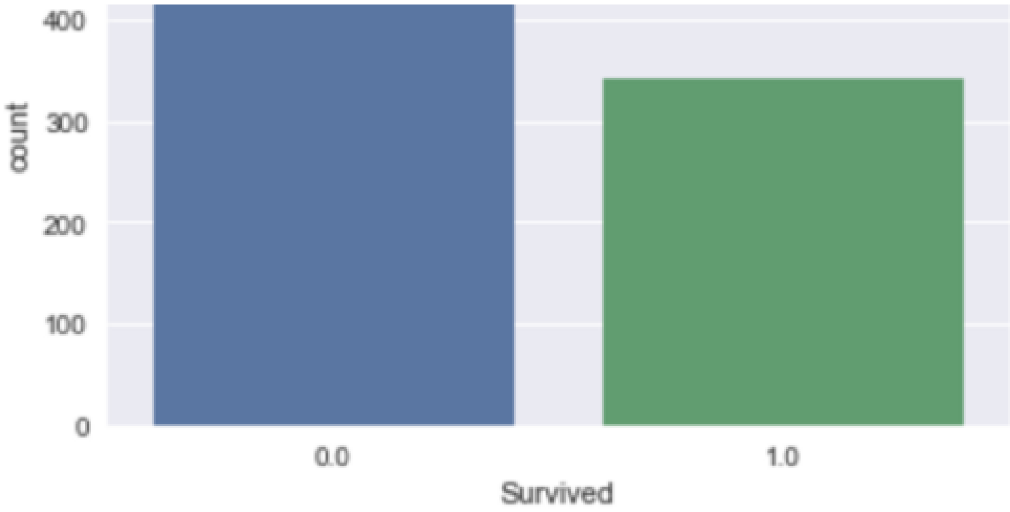
接下來要對資料開始做一些觀察以及分析。首先分析生存以及死亡的比例是否有相當大的落差，發現大概死亡的比例是6成、生存的比例大概是4成

Data Analysis

```
sns.countplot(data['Survived'])
```

<matplotlib.axes._subplots.AxesSubplot at 0x11bb5bdd8>

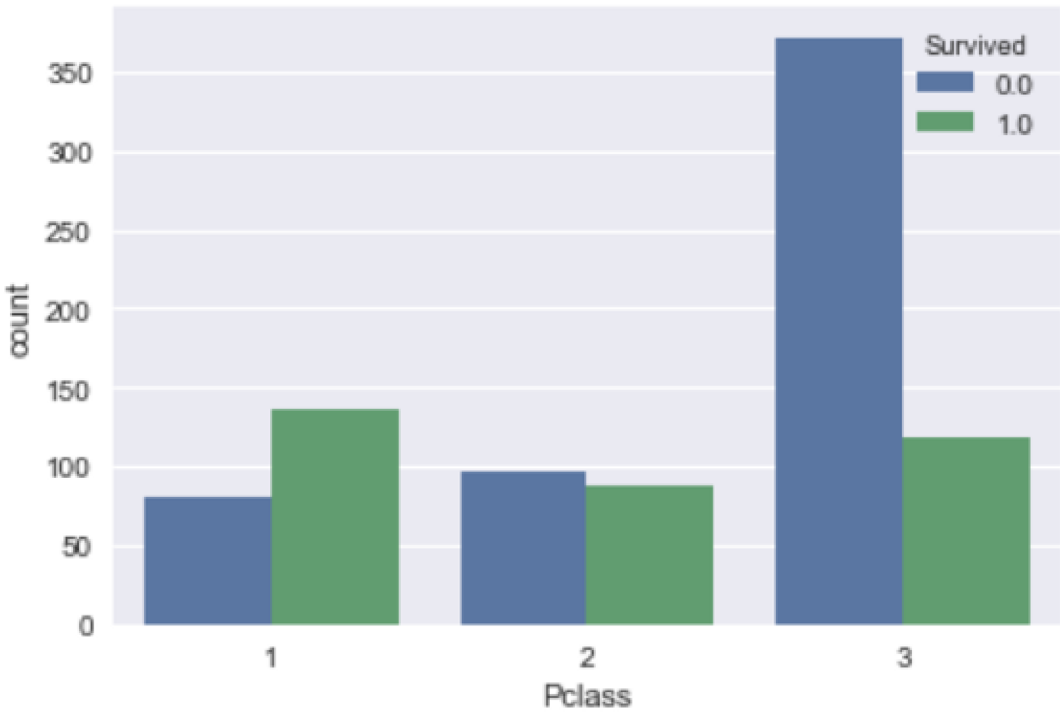




觀察艙等跟生存率的關係，可以發現在1艙等的生存率最高、再來是2艙等、最後是3艙等的

```
sns.countplot(data['Pclass'], hue=data['Survived'])
```

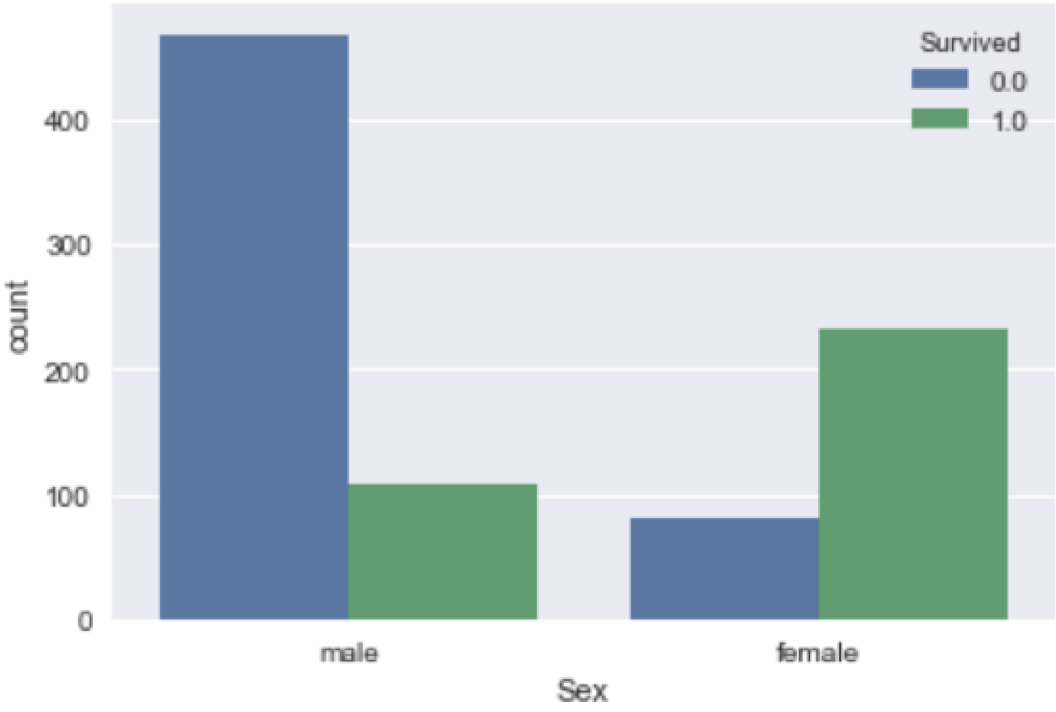
```
<matplotlib.axes._subplots.AxesSubplot at 0x11bc3cac8>
```



再來是觀察性別跟生存率的關係，發現女生生存率是男生的好幾倍。或許是像在電影裡頭一樣，在逃難的時候先讓女生以及小孩先搭船

```
sns.countplot(data['Sex'], hue=data['Survived'])
```

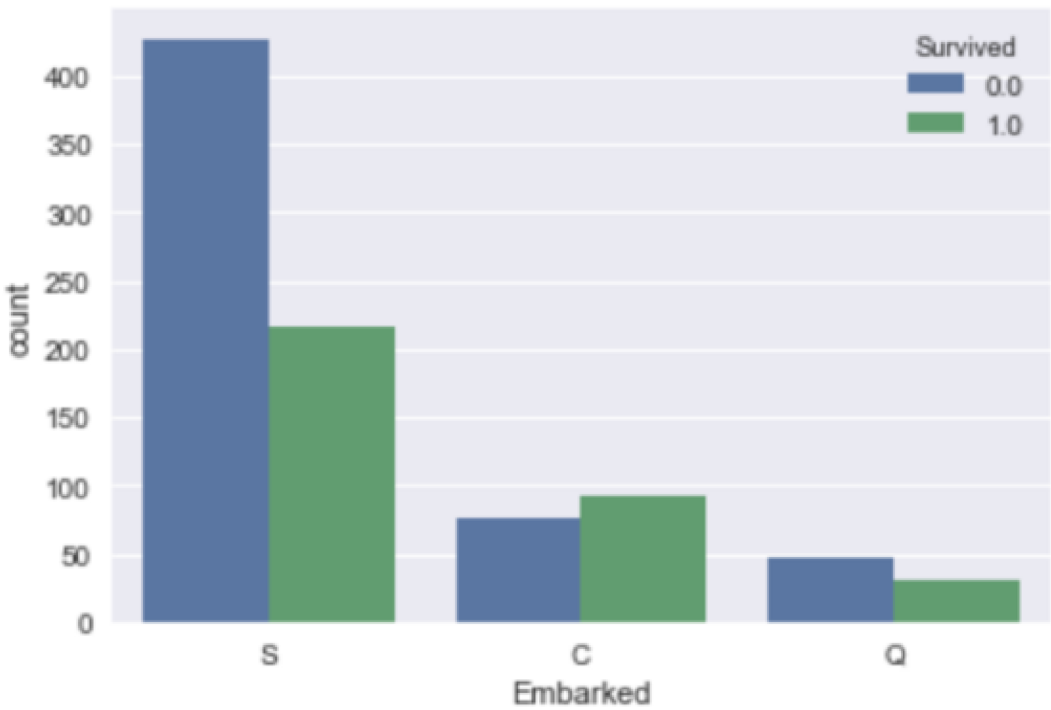
```
<matplotlib.axes._subplots.AxesSubplot at 0x11f112cf8>
```



出發港口跟生存率的差異，可以發現S港出發的都比較容易死亡，其原因可能是S城市出發的人買的票價都比較便宜

```
sns.countplot(data['Embarked'], hue=data['Survived'])
```

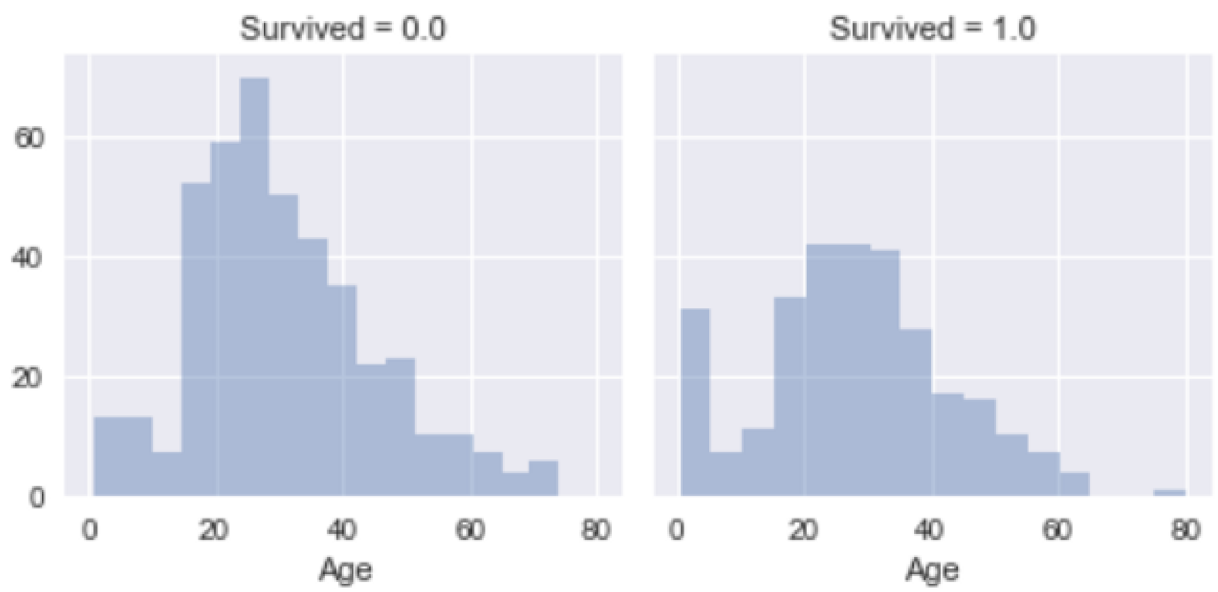
```
<matplotlib.axes._subplots.AxesSubplot at 0x11f1d7908>
```



年齡跟生存率的關係，可以發現的確年齡小的存活比例高出許多

```
g = sns.FacetGrid(data, col='Survived')
g.map(sns.distplot, 'Age', kde=False)
```

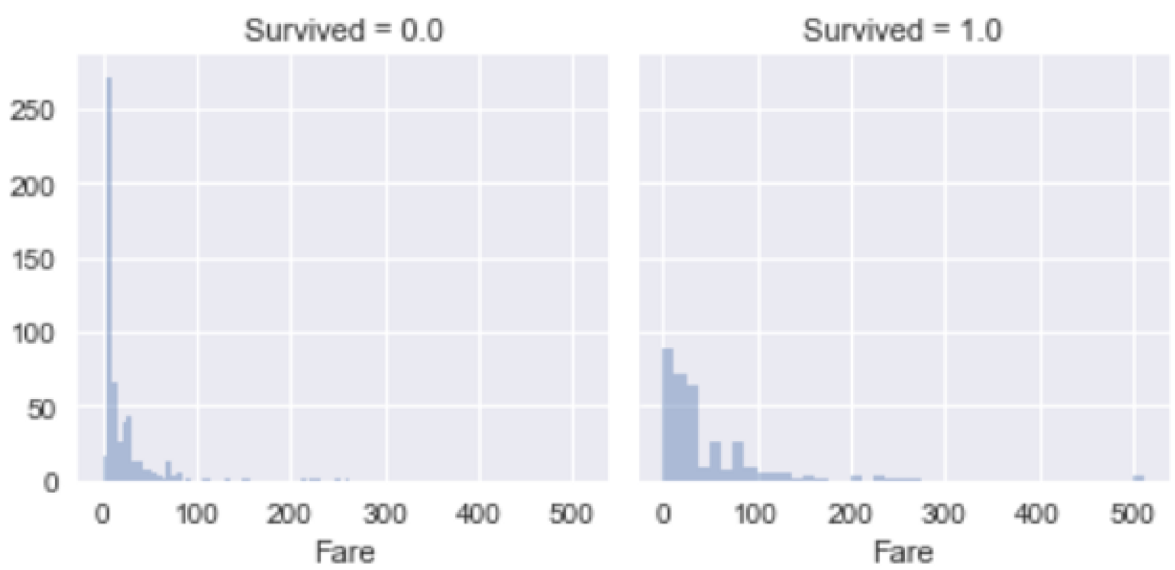
<seaborn.axisgrid.FacetGrid at 0x11f3a8fd0>



票價跟生存率的關係，可以發現票價低的乘客死亡率高出許多

```
g = sns.FacetGrid(data, col='Survived')
g.map(sns.distplot, 'Fare', kde=False)
```

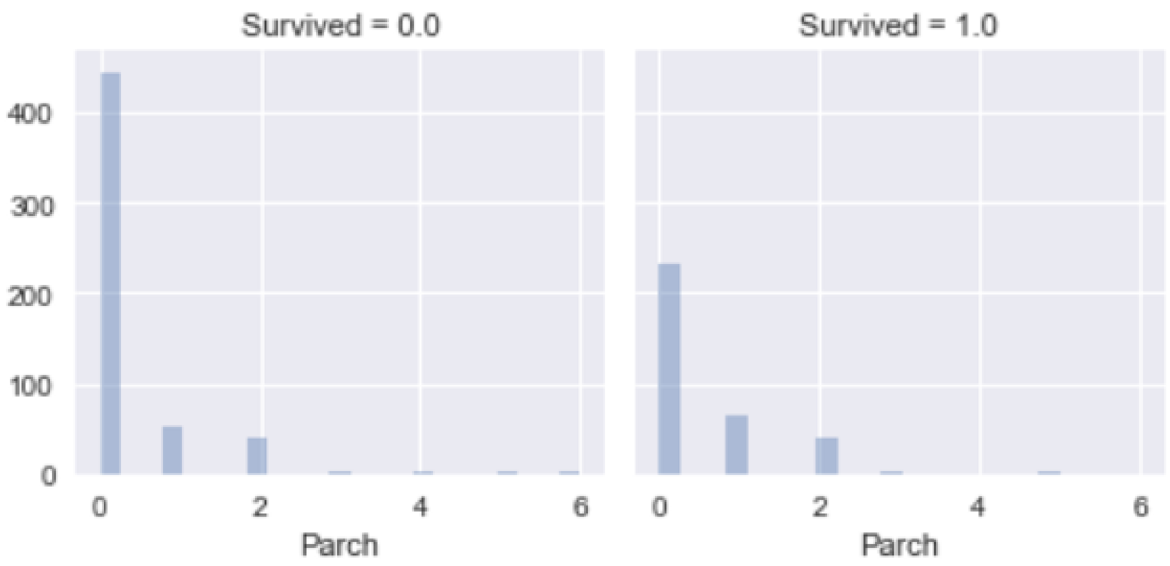
<seaborn.axisgrid.FacetGrid at 0x11f3ac0f0>



父母 + 小孩的數量跟生存率的關係，發現沒有跟父母小孩一起來的生存率比起有跟父母小孩來的低

```
g = sns.FacetGrid(data, col='Survived')
g.map(sns.distplot, 'Parch', kde=False)
```

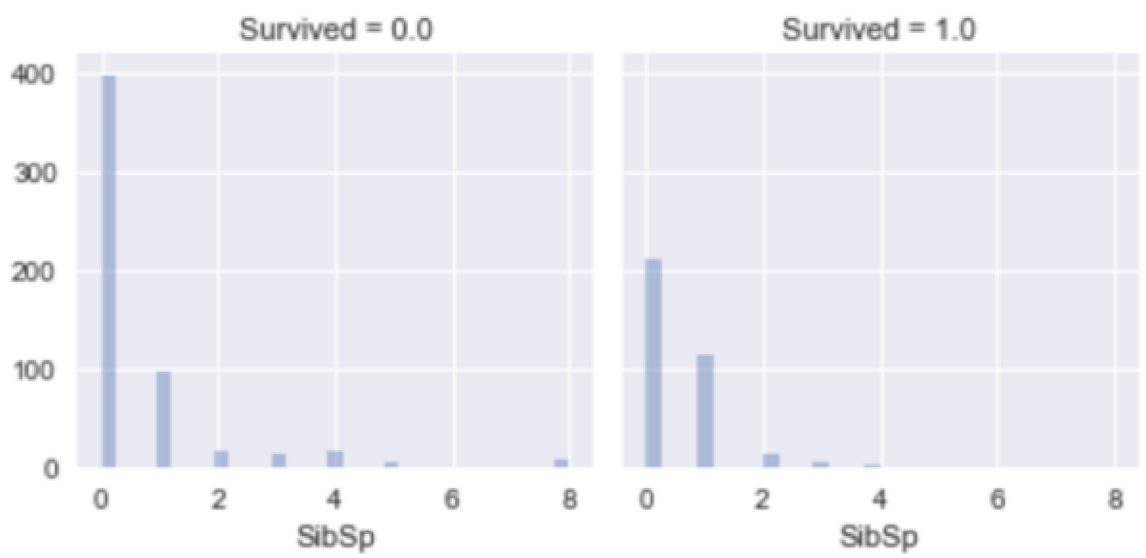
<seaborn.axisgrid.FacetGrid at 0x11f9e7e10>



兄弟姊妹＋丈夫妻子的數量跟生存率的關係，發現沒有帶兄弟姊妹＋丈夫妻子一起來的生存率比起有跟兄弟姊妹＋丈夫妻子來的低

```
g = sns.FacetGrid(data, col='Survived')
g.map(sns.distplot, 'SibSp', kde=False)
```

<seaborn.axisgrid.FacetGrid at 0x11f3bf1d0>

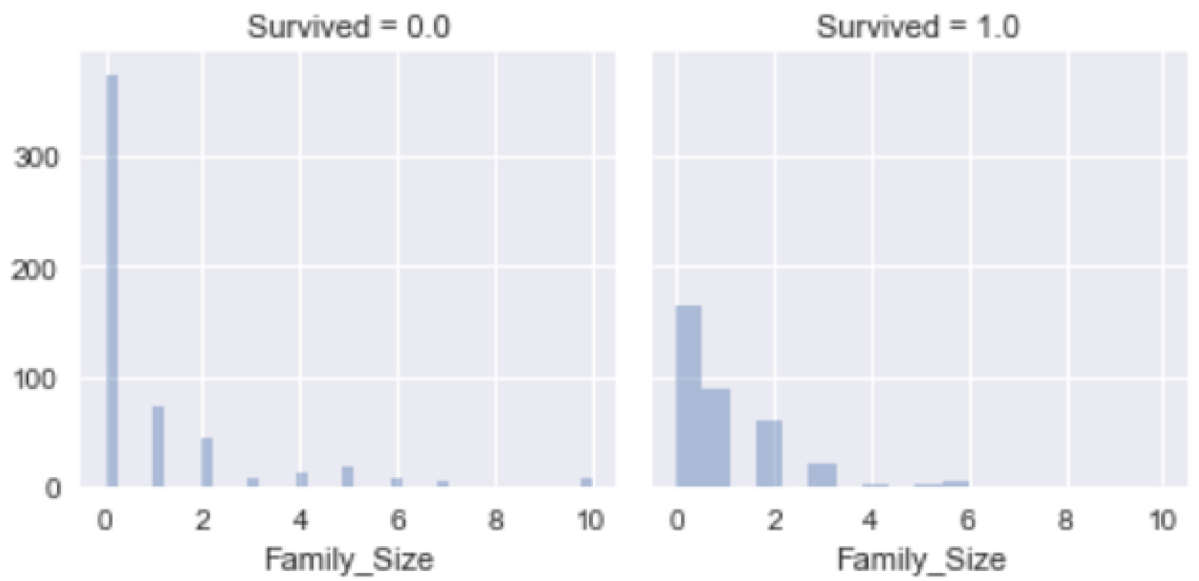


經過一些反覆的測試把“父母＋小孩”加上“兄弟姊妹＋丈夫妻子”的數量變成一個新的欄位叫做家庭大小，在預測上會更為準確

```
data['Family_Size'] = data['Parch'] + data['SibSp']
```

```
g = sns.FacetGrid(data, col='Survived')
g.map(sns.distplot, 'Family_Size', kde=False)
```

```
<seaborn.axisgrid.FacetGrid at 0x11fd7b320>
```



特徵工程

接下來要來處理之前提到一些特徵，像是姓名這個欄位的資料就不能直接拿來用，但如果直接丟掉是一種資訊的浪費，因此我們稍微觀察一下名字這個欄位，可以發現名字的這個欄位有稱謂的資訊(Mr., Miss.) 我們可以利用這些資訊在未來更加提升預測的準確度

Name
Braund, Mr. Owen Harris
Cumings, Mrs. John Bradley (Florence Briggs Th...
Heikkinen, Miss. Laina
Futrelle, Mrs. Jacques Heath (Lily May Peel)
Allen, Mr. William Henry

Moran, Mr. James
McCarthy, Mr. Timothy J
Palsson, Master. Gosta Leonard
Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)
Nasser, Mrs. Nicholas (Adele Achem)
Sandstrom, Miss. Marguerite Rut
Bonnell, Miss. Elizabeth
Saundercock, Mr. William Henry
Andersson, Mr. Anders Johan
Vestrom, Miss. Hulda Amanda Adolfina

將姓名的稱謂萃取出來，可以發現這些人的稱謂總共有'Mr', 'Mrs', 'Miss', 'Master', 'Don', 'Rev', 'Dr', 'Mme', 'Ms', 'Major', 'Lady', 'Sir', 'Mlle', 'Col', 'Capt', 'the Countess', 'Jonkheer', 'Dona'

```
data['Title1'] = data['Name'].str.split(", ", expand=True)[1]
```

```
data['Name'].str.split(", ", expand=True).head(3)
```

	0	1
0	Braund	Mr. Owen Harris
1	Cumings	Mrs. John Bradley (Florence Briggs Thayer)
2	Heikkinen	Miss. Laina

```
data['Title1'].head(3)
```

```
0                                 Mr. Owen Harris
1     Mrs. John Bradley (Florence Briggs Thayer)
2                                     Miss. Laina
Name: Title1, dtype: object
```

```
data['Title1'] = data['Title1'].str.split(".", expand=True)[0]
```

```
data['Title1'].head(3)
```

```
0       Mr
1      Mrs
```

```
2    Miss
Name: Title1, dtype: object
```

```
data['Title1'].unique()
```

```
array(['Mr', 'Mrs', 'Miss', 'Master', 'Don', 'Rev', 'Dr', 'Mme', 'Ms',
       'Major', 'Lady', 'Sir', 'Mlle', 'Col', 'Capt', 'the Countess',
       'Jonkheer', 'Dona'], dtype=object)
```

將稱謂對性別、生存率、以及年齡做分析，發現一個有趣的地方，像是Master平均年齡只有五歲非常小，都是男生，並且生存機率有大約6成

```
pd.crosstab(data['Title1'],data['Sex']).T.style.background_gradient(cmap='summer_r')
```

Title1	Capt	Col	Don	Dona	Dr	Jonkheer	Lady	Major	Master	Miss	Mlle	Mme	Mr	Mrs	Ms	Rev	Sir	the Countess
Sex																		
female	0	0	0	1	1	0	1	0	0	260	2	1	0	197	2	0	0	1
male	1	4	1	0	7	1	0	2	61	0	0	0	757	0	0	8	1	0

```
pd.crosstab(data['Title1'],data['Survived']).T.style.background_gradient(cmap='summer_r')
```

Title1	Capt	Col	Don	Dr	Jonkheer	Lady	Major	Master	Miss	Mlle	Mme	Mr	Mrs	Ms	Rev	Sir	the Countess
Survived																	
0.0	1	1	1	4	1	0	1	17	55	0	0	436	26	0	6	0	0
1.0	0	1	0	3	0	1	1	23	127	2	1	81	99	1	0	1	1

```
data.groupby(['Title1'])['Age'].mean()
```

```
Title1
Capt            70.000000
Col             54.000000
Don             40.000000
Dona            39.000000
Dr              43.571429
Jonkheer        38.000000
Lady            48.000000
Major           48.500000
Master           5.482642
Miss            21.774238
Mlle            24.000000
Mme             24.000000
Mr              32.252151
Mrs             36.994118
```

```
Ms              28.000000
Rev             41.250000
Sir             49.000000
the Countess    33.000000
```

若仔細觀察這些稱謂(’Mr’, ‘Mrs’, ‘Miss’, ‘Master’, ‘Don’, ‘Rev’, ‘Dr’, ‘Mme’, ‘Ms’, ‘Major’, ‘Lady’, ‘Sir’, ‘Mlle’, ‘Col’, ‘Capt’, ‘the Countess’, ‘Jonkheer’, ‘Dona’)會發現有些是稱謂的乘客非常少，如果我們只為了這些少數的乘客多了一個稱謂這樣對於機器學習的模型來說是一件不好的事情。因此我們把其中的稱謂做合併。

```
data['Title2'] = data['Title1'].replace(['Mlle','Mme','Ms','Dr','Major','Lady','the Counte
         ['Miss','Mrs','Miss','Mr','Mr','Mrs','Mrs','Mr','Mr','Mr','Mr','Mr','Mr','Mrs'])
```

```
data['Title2'].unique()
```

```
array(['Mr', 'Mrs', 'Miss', 'Master'], dtype=object)
```

```
pd.crosstab(data['Title2'],data['Sex']).T.style.background_gradient(cmap='summer_r')
```

Title2	Master	Miss	Mr	Mrs
Sex				
female	0	264	1	201
male	61	0	782	0

```
osstab(data['Title2'],data['Survived']).T.style.background_gradient(cmap='summer_r')
```

Title2	Master	Miss	Mr	Mrs
Survived				
0.0	17	55	451	26
1.0	23	130	87	102

再來把票號的資訊取出前面英文的部分，因為相同的英文代碼可能代表的是房間的位置，後面的號碼沒有意義所以省略，如果只有號碼的票號就用X來表示

```
data['Ticket_info'] = data['Ticket'].apply(lambda x : x.replace(".","").replace("/","").strip
```

```
data['Ticket_info'].unique()
```

```
array(['A5', 'PC', 'STONO2', 'X', 'PP', 'CA', 'SCParis', 'SCA4', 'A4',
       'SP', 'SOC', 'WC', 'SOTONOQ', 'WEP', 'STONO', 'C', 'SCPARIS', 'SOP',
       'Fa', 'LINE', 'FCC', 'SWPP', 'SCOW', 'PPP', 'SC', 'SCAH', 'AS',
       'SOPP', 'FC', 'SOTONO2', 'CASOTON', 'SCA3', 'STONOQ', 'AQ4', 'A',
       'LP', 'AQ3'], dtype=object)
```

由於登船港口(Embarked)只有遺漏少數，我們就直接補上出現次數最多的”S”，費用(Fare)也只有遺漏一筆，因此就直接補上平均值

```
data['Embarked'] = data['Embarked'].fillna('S')
```

```
data['Fare'] = data['Fare'].fillna(data['Fare'].mean())
```

觀察Cabin的資料後，只取出最前面的英文字母，剩下的用NoCabin來表示

```
data['Cabin'].head(10)
```

```
0     NaN
1     C85
2     NaN
3    C123
4     NaN
5     NaN
6     E46
7     NaN
8     NaN
9     NaN
Name: Cabin, dtype: object
```

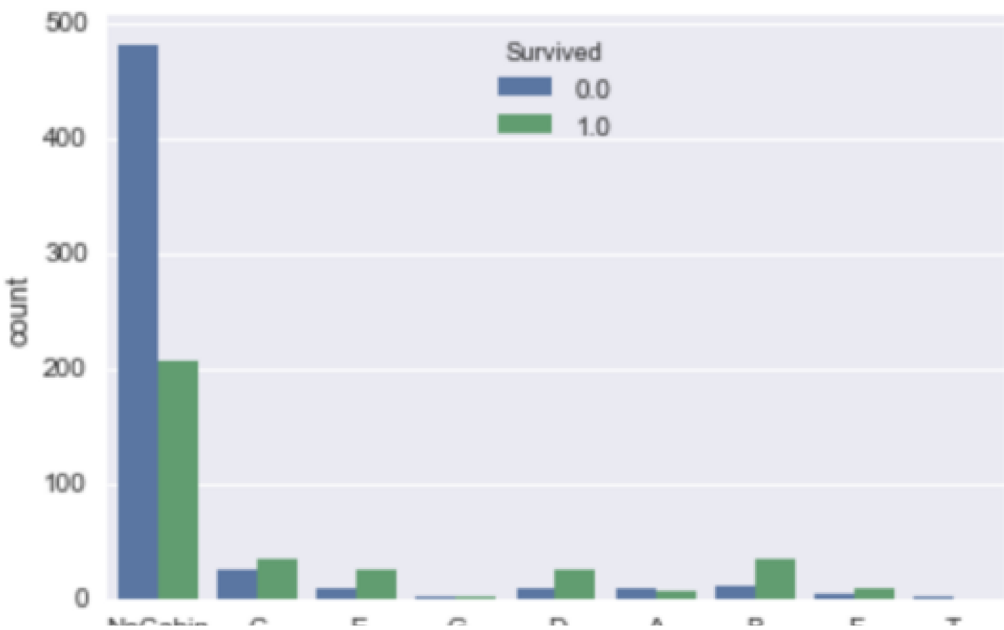
```
data["Cabin"] = data['Cabin'].apply(lambda x : str(x)[0] if not pd.isnull(x) else 'NoCabin')
```

```
data["Cabin"].unique()
```

```
array(['NoCabin', 'C', 'E', 'G', 'D', 'A', 'B', 'F', 'T'], dtype=object)
```

```
sns.countplot(data['Cabin'], hue=data['Survived'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x11dcc3668>
```



將類別資料轉為整數

```
data['Sex'] = data['Sex'].astype('category').cat.codes
data['Embarked'] = data['Embarked'].astype('category').cat.codes
data['Pclass'] = data['Pclass'].astype('category').cat.codes
data['Title1'] = data['Title1'].astype('category').cat.codes
data['Title2'] = data['Title2'].astype('category').cat.codes
data['Cabin'] = data['Cabin'].astype('category').cat.codes
data['Ticket_info'] = data['Ticket_info'].astype('category').cat.codes
```

使用隨機森林來推測年齡

```
dataAgeNull = data[data["Age"].isnull()]
dataAgeNotNull = data[data["Age"].notnull()]
remove_outlier = dataAgeNotNull[(np.abs(dataAgeNotNull["Fare"]-dataAgeNotNull["Fare"].mean())>(4*dataAgeNotNull["Fare"].
                    (np.abs(dataAgeNotNull["Family_Size"]-dataAgeNotNull["Family_Size"].mean())>(4*dataAgeNotNull["Far
                   ]
rfModel_age = RandomForestRegressor(n_estimators=2000,random_state=42)
ageColumns = ['Embarked', 'Fare', 'Pclass', 'Sex', 'Family_Size', 'Title1', 'Title2','Cabin','Ticket_info']
rfModel_age.fit(remove_outlier[ageColumns], remove_outlier["Age"])

ageNullValues = rfModel_age.predict(X= dataAgeNull[ageColumns])
dataAgeNull.loc[:,"Age"] = ageNullValues
data = dataAgeNull.append(dataAgeNotNull)
data.reset_index(inplace=True, drop=True)
```

載入隨機森林演算法(Random Forest)來預測存活率

```
from sklearn.ensemble import RandomForestClassifier

rf = RandomForestClassifier(criterion='gini',
                             n_estimators=1000,
                             min_samples_split=12,
                             min_samples_leaf=1,
                             oob_score=True,
                             random_state=1,
                             n_jobs=-1)

rf.fit(dataTrain.iloc[:, 1:], dataTrain.iloc[:, 0])
print("%.4f" % rf.oob_score_)
```

```
0.8294
```

將欲提交至kaggle的檔案寫出，上傳就大功告成了！

Submit

```
rf_res =  rf.predict(dataTest)
submit['Survived'] = rf_res
submit['Survived'] = submit['Survived'].astype(int)
submit.to_csv('submit.csv', index= False)
```

```
submit
```

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1
5	897	0
6	898	0
7	899	0
8	900	1
9	901	0
10	902	0

程式碼

In [1]:
```
from sklearn import preprocessing
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import RandomForestRegressor

import warnings
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

%matplotlib inline
pd.options.mode.chained_assignment = None
```

In [2]:
```
train = pd.read_csv("Titanic/train.csv")
test = pd.read_csv("Titanic/test.csv")
submit = pd.read_csv('Titanic/gender_submission.csv')
```

In [3]:
```
train.head(5)
```

Out[3]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Tic
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5

[資料分析&機器學習] 第4.1講 鐵達尼號災難生存預測.ipynb hosted with ♡ by GitHub view raw

比賽連結

https://www.kaggle.com/c/titanic/

感謝你閱讀完這篇文章，如果你覺得這些文章對你有幫助請在底下幫我拍個手（長按最多可以拍50下手）。

[Python資料分析＆機器學習]這系列文章是我在Hahow上面所開設課程的講義，如果你是新手想著看影片一步一步學習，可以參考這門課：https://hahow.in/cr/pydataml

如果你對什麼主題的文章有興趣的話，歡迎透過這個連結告訴我：
https://yehjames.typeform.com/to/XIIVQC
有任何問題也歡迎在底下留言或是來信告訴我： yehjames23@gmail.com

參考閱讀

1. Exploring Survival on the Titanic
2. Introduction to Ensembling/Stacking in Python
3. A Journey through Titanic
4. Titanic Data Science Solutions
5. Pytanic

Python Machine Learning Kaggle Titanic