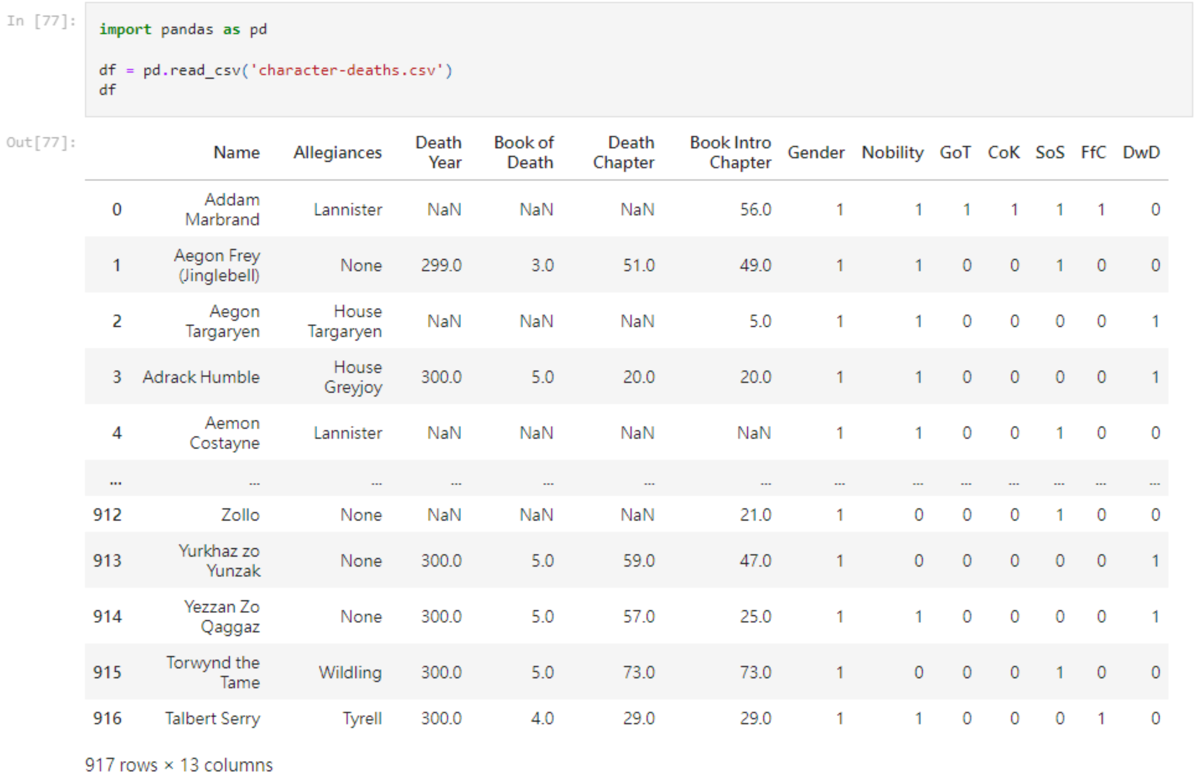
Game of Thrones - Decision Tree

312706034 資管碩二 張棨翔

- 以DataFrame的形式讀取初始資料集



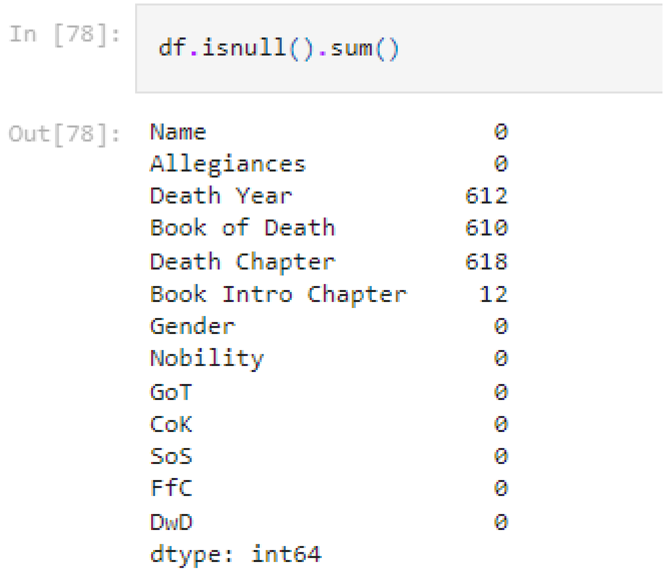
In [77]:
```python
import pandas as pd

df = pd.read_csv('character-deaths.csv')
df
```

Out[77]:

	Name	Allegiances	Death Year	Book of Death	Death Chapter	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD
0	Addam Marbrand	Lannister	NaN	NaN	NaN	56.0	1	1	1	1	1	1	0
1	Aegon Frey (Jinglebell)	None	299.0	3.0	51.0	49.0	1	1	0	0	1	0	0
2	Aegon Targaryen	House Targaryen	NaN	NaN	NaN	5.0	1	1	0	0	0	0	1
3	Adrack Humble	House Greyjoy	300.0	5.0	20.0	20.0	1	1	0	0	0	0	1
4	Aemon Costayne	Lannister	NaN	NaN	NaN	NaN	1	1	0	0	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
912	Zollo	None	NaN	NaN	NaN	21.0	1	0	0	0	1	0	0
913	Yurkhaz zo Yunzak	None	300.0	5.0	59.0	47.0	1	0	0	0	0	0	1
914	Yezzan Zo Qaggaz	None	300.0	5.0	57.0	25.0	1	1	0	0	0	0	1
915	Torwynd the Tame	Wildling	300.0	5.0	73.0	73.0	1	0	0	0	1	0	0
916	Talbert Serry	Tyrell	300.0	4.0	29.0	29.0	1	1	0	0	0	1	0

917 rows × 13 columns

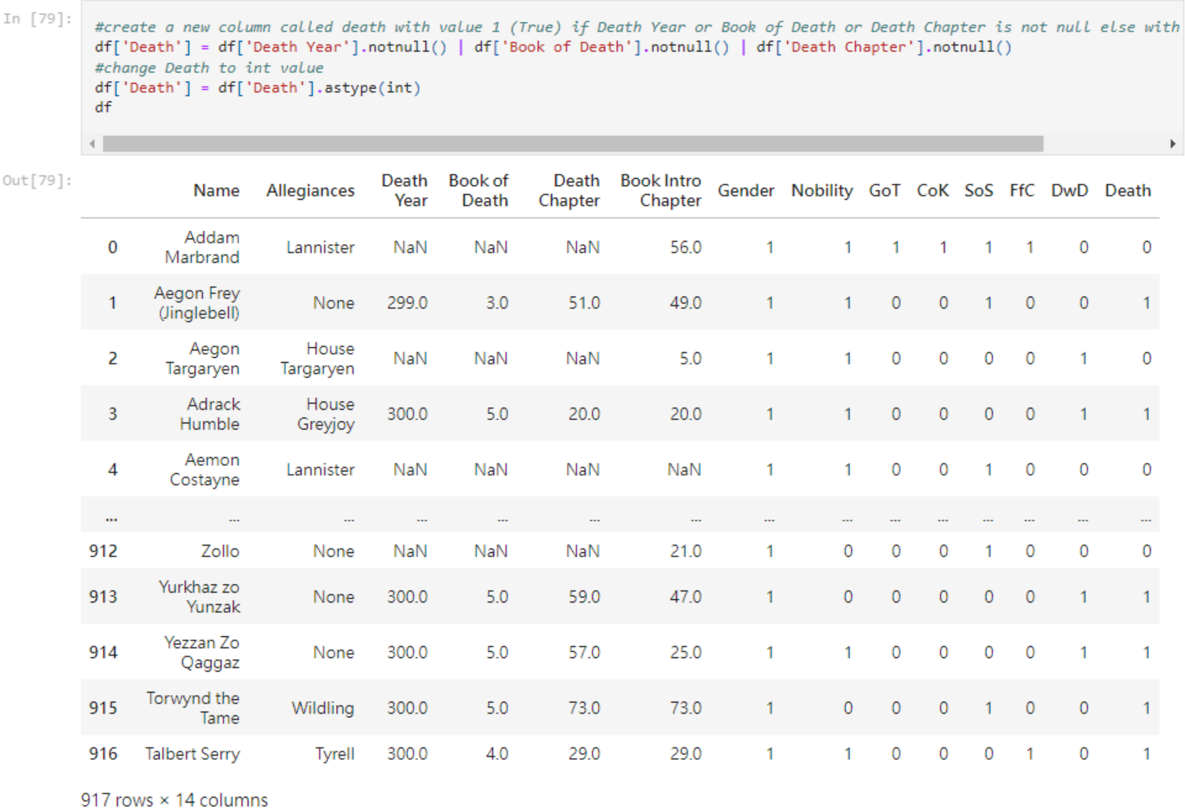
- 檢查各欄位的缺失值狀況



In [78]:
```python
df.isnull().sum()
```

Out[78]:
```
Name                    0
Allegiances             0
Death Year            612
Book of Death         610
Death Chapter         618
Book Intro Chapter     12
Gender                  0
Nobility                0
GoT                     0
CoK                     0
SoS                     0
FfC                     0
DwD                     0
dtype: int64
```

- 建立新欄位"Death"以3項欄位值是否為NaN來決定要填入1(死亡)或0(存活)



In [79]:

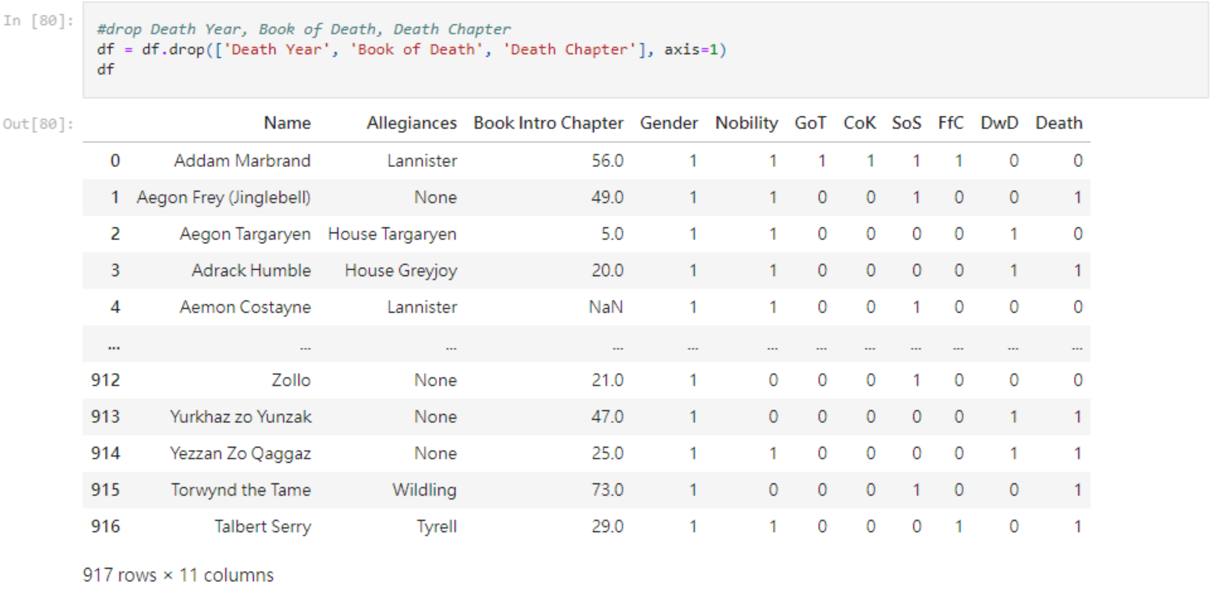
```
#create a new column called death with value 1 (True) if Death Year or Book of Death or Death Chapter is not null else with
df['Death'] = df['Death Year'].notnull() | df['Book of Death'].notnull() | df['Death Chapter'].notnull()
#change Death to int value
df['Death'] = df['Death'].astype(int)
df
```

Out[79]:

	Name	Allegiances	Death Year	Book of Death	Death Chapter	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Death
0	Addam Marbrand	Lannister	NaN	NaN	NaN	56.0	1	1	1	1	1	1	0	0
1	Aegon Frey (Jinglebell)	None	299.0	3.0	51.0	49.0	1	1	0	0	1	0	0	1
2	Aegon Targaryen	House Targaryen	NaN	NaN	NaN	5.0	1	1	0	0	0	0	1	0
3	Adrack Humble	House Greyjoy	300.0	5.0	20.0	20.0	1	1	0	0	0	0	1	1
4	Aemon Costayne	Lannister	NaN	NaN	NaN	NaN	1	1	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
912	Zollo	None	NaN	NaN	NaN	21.0	1	0	0	0	1	0	0	0
913	Yurkhaz zo Yunzak	None	300.0	5.0	59.0	47.0	1	0	0	0	0	0	1	1
914	Yezzan Zo Qaggaz	None	300.0	5.0	57.0	25.0	1	1	0	0	0	0	1	1
915	Torwynd the Tame	Wildling	300.0	5.0	73.0	73.0	1	0	0	0	1	0	0	1
916	Talbert Serry	Tyrell	300.0	4.0	29.0	29.0	1	1	0	0	0	1	0	1

917 rows × 14 columns

- 刪除作為新欄位"Death"判斷依據的欄位(這些欄位之後不會使用到)



In [80]:

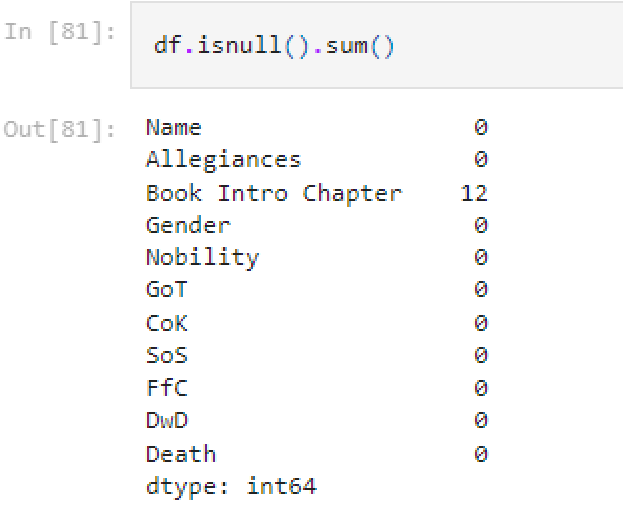
```
#drop Death Year, Book of Death, Death Chapter
df = df.drop(['Death Year', 'Book of Death', 'Death Chapter'], axis=1)
df
```

Out[80]:

	Name	Allegiances	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Death
0	Addam Marbrand	Lannister	56.0	1	1	1	1	1	1	0	0
1	Aegon Frey (Jinglebell)	None	49.0	1	1	0	0	1	0	0	1
2	Aegon Targaryen	House Targaryen	5.0	1	1	0	0	0	0	1	0
3	Adrack Humble	House Greyjoy	20.0	1	1	0	0	0	0	1	1
4	Aemon Costayne	Lannister	NaN	1	1	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
912	Zollo	None	21.0	1	0	0	0	1	0	0	0
913	Yurkhaz zo Yunzak	None	47.0	1	0	0	0	0	0	1	1
914	Yezzan Zo Qaggaz	None	25.0	1	1	0	0	0	0	1	1
915	Torwynd the Tame	Wildling	73.0	1	0	0	0	1	0	0	1
916	Talbert Serry	Tyrell	29.0	1	1	0	0	0	1	0	1

917 rows × 11 columns

- 檢查目前的欄位缺失值狀況，目前剩"Book Intro Chapter"有缺失值



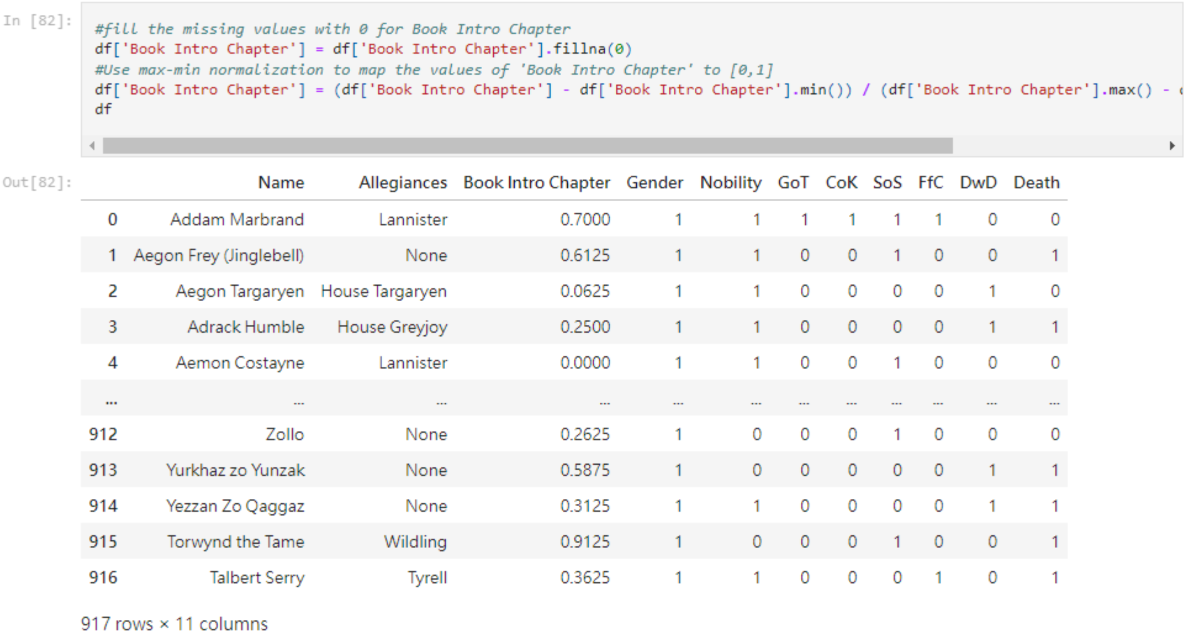
In [81]:

```
df.isnull().sum()
```

Out[81]:

```
Name                   0
Allegiances            0
Book Intro Chapter    12
Gender                 0
Nobility               0
GoT                    0
CoK                    0
SoS                    0
FfC                    0
DwD                    0
Death                  0
dtype: int64
```

- 用數字0填補缺失值，並且透過Min-Max的轉換，讓"Book Intro Chapter"的欄位值落在[0,1]



In [82]:

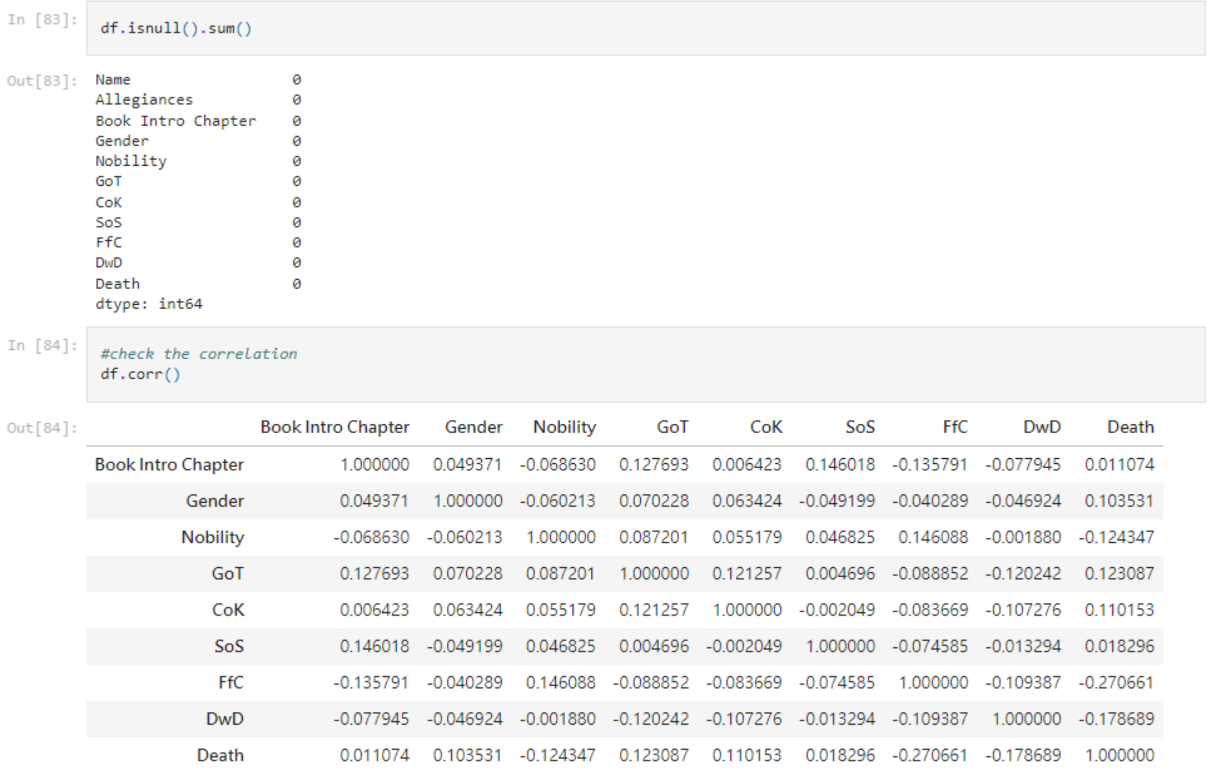
```
#fill the missing values with 0 for Book Intro Chapter
df['Book Intro Chapter'] = df['Book Intro Chapter'].fillna(0)
#Use max-min normalization to map the values of 'Book Intro Chapter' to [0,1]
df['Book Intro Chapter'] = (df['Book Intro Chapter'] - df['Book Intro Chapter'].min()) / (df['Book Intro Chapter'].max() - (
df
```

Out[82]:

	Name	Allegiances	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Death
0	Addam Marbrand	Lannister	0.7000	1	1	1	1	1	1	0	0
1	Aegon Frey (Jinglebell)	None	0.6125	1	1	0	0	1	0	0	1
2	Aegon Targaryen	House Targaryen	0.0625	1	1	0	0	0	0	1	0
3	Adrack Humble	House Greyjoy	0.2500	1	1	0	0	0	0	1	1
4	Aemon Costayne	Lannister	0.0000	1	1	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...
912	Zollo	None	0.2625	1	0	0	0	1	0	0	0
913	Yurkhaz zo Yunzak	None	0.5875	1	0	0	0	0	0	1	1
914	Yezzan Zo Qaggaz	None	0.3125	1	1	0	0	0	0	1	1
915	Torwynd the Tame	Wildling	0.9125	1	0	0	0	1	0	0	1
916	Talbert Serry	Tyrell	0.3625	1	1	0	0	0	1	0	1

917 rows × 11 columns

- 檢查目前的缺失值，可以觀察到所有缺失值都已經被處理完畢



In [83]:
```
df.isnull().sum()
```

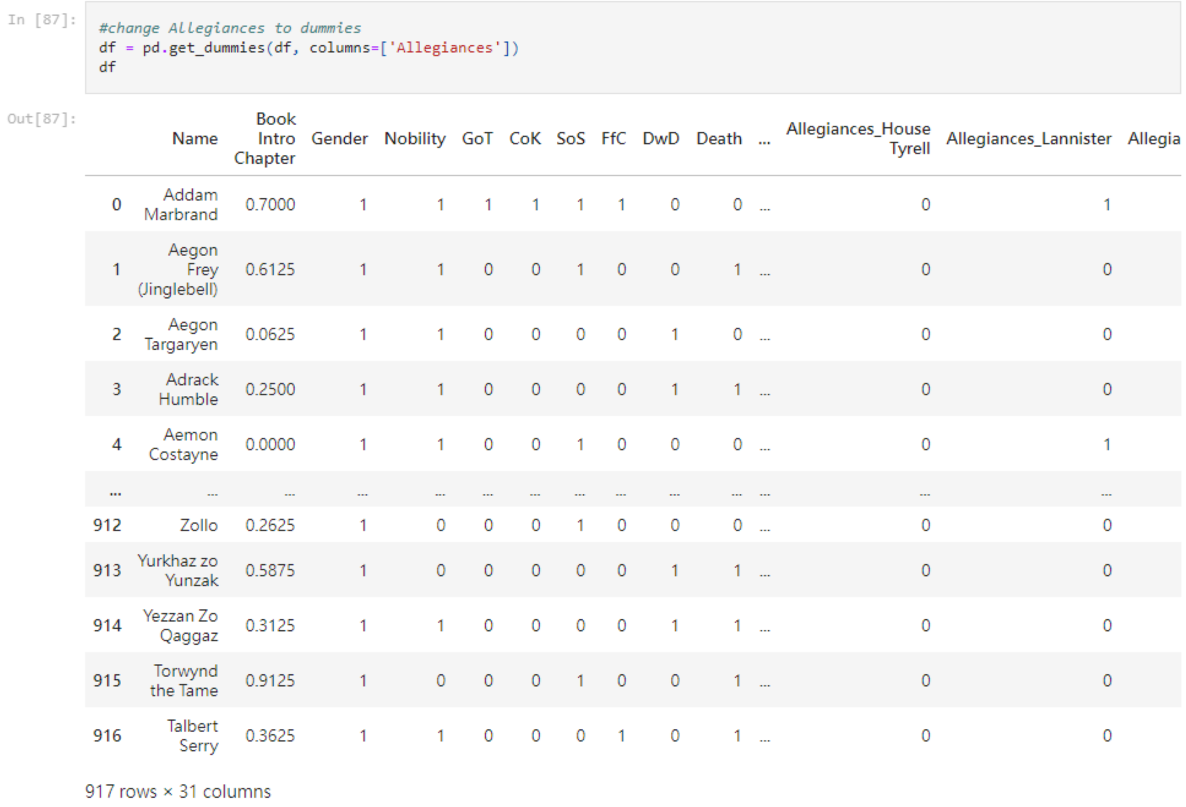
Out[83]:
```
Name                 0
Allegiances          0
Book Intro Chapter   0
Gender               0
Nobility             0
GoT                  0
CoK                  0
SoS                  0
FfC                  0
DwD                  0
Death                0
dtype: int64
```

In [84]:
```
#check the correlation
df.corr()
```

Out[84]:

	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Death
Book Intro Chapter	1.000000	0.049371	-0.068630	0.127693	0.006423	0.146018	-0.135791	-0.077945	0.011074
Gender	0.049371	1.000000	-0.060213	0.070228	0.063424	-0.049199	-0.040289	-0.046924	0.103531
Nobility	-0.068630	-0.060213	1.000000	0.087201	0.055179	0.046825	0.146088	-0.001880	-0.124347
GoT	0.127693	0.070228	0.087201	1.000000	0.121257	0.004696	-0.088852	-0.120242	0.123087
CoK	0.006423	0.063424	0.055179	0.121257	1.000000	-0.002049	-0.083669	-0.107276	0.110153
SoS	0.146018	-0.049199	0.046825	0.004696	-0.002049	1.000000	-0.074585	-0.013294	0.018296
FfC	-0.135791	-0.040289	0.146088	-0.088852	-0.083669	-0.074585	1.000000	-0.109387	-0.270661
DwD	-0.077945	-0.046924	-0.001880	-0.120242	-0.107276	-0.013294	-0.109387	1.000000	-0.178689
Death	0.011074	0.103531	-0.124347	0.123087	0.110153	0.018296	-0.270661	-0.178689	1.000000

- 針對類別資料做編碼的處理，依據原始屬性產生新的二元屬性欄位(0、1)



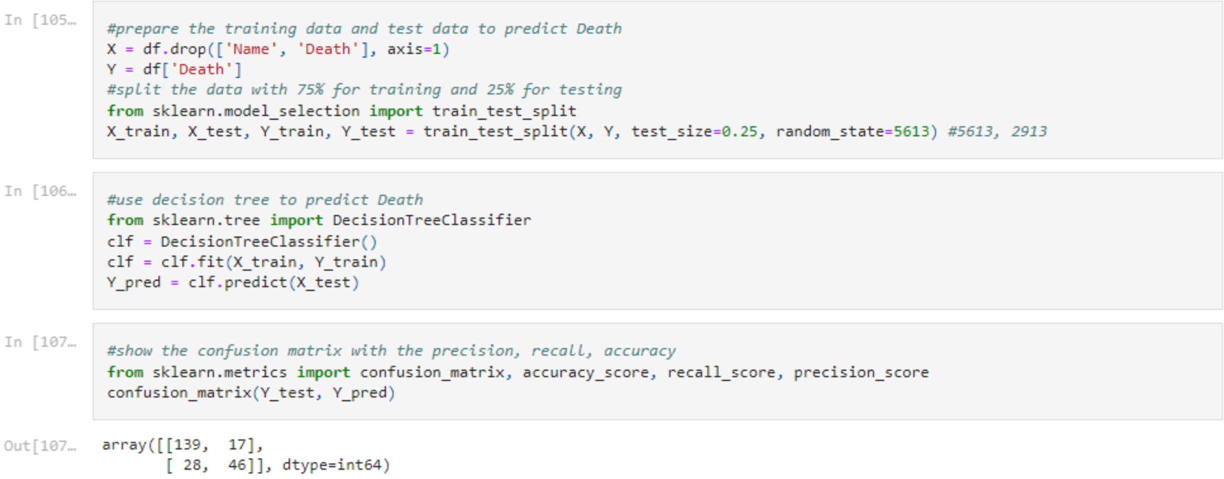
In [87]:
```
#change Allegiances to dummies
df = pd.get_dummies(df, columns=['Allegiances'])
df
```

Out[87]:

	Name	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Death	...	Allegiances_House Tyrell	Allegiances_Lannister	Allegia
0	Addam Marbrand	0.7000	1	1	1	1	1	1	0	0	...	0	1	
1	Aegon Frey (Jinglebell)	0.6125	1	1	0	0	1	0	0	1	...	0	0	
2	Aegon Targaryen	0.0625	1	1	0	0	0	0	1	0	...	0	0	
3	Adrack Humble	0.2500	1	1	0	0	0	0	1	1	...	0	0	
4	Aemon Costayne	0.0000	1	1	0	0	1	0	0	0	...	0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
912	Zollo	0.2625	1	0	0	0	1	0	0	0	...	0	0	
913	Yurkhaz zo Yunzak	0.5875	1	0	0	0	0	0	1	1	...	0	0	
914	Yezzan Zo Qaggaz	0.3125	1	1	0	0	0	0	1	1	...	0	0	
915	Torwynd the Tame	0.9125	1	0	0	0	1	0	0	1	...	0	0	
916	Talbert Serry	0.3625	1	1	0	0	0	1	0	1	...	0	0	

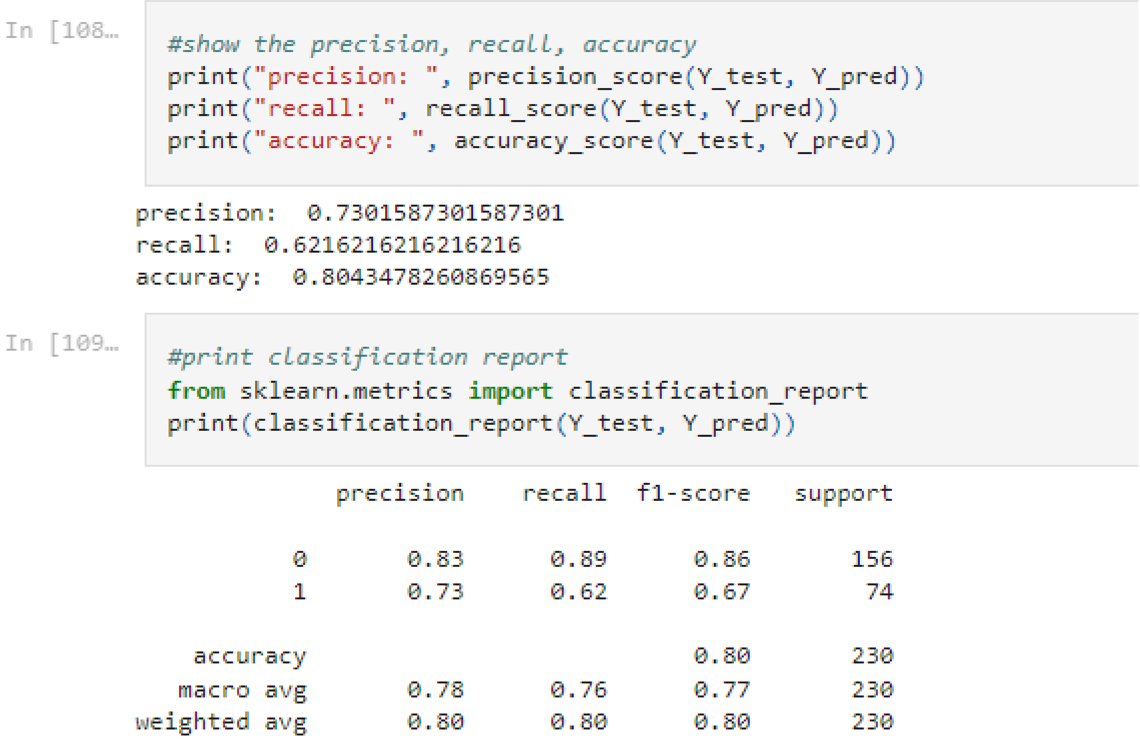
917 rows × 31 columns

- 準備用來訓練Decision Tree的資料，訓練資料75%測試資料25%，並套用實驗後找到的指定random_state = 5613(經過20000次的實驗取得，最後產生簡易的confusion matrix



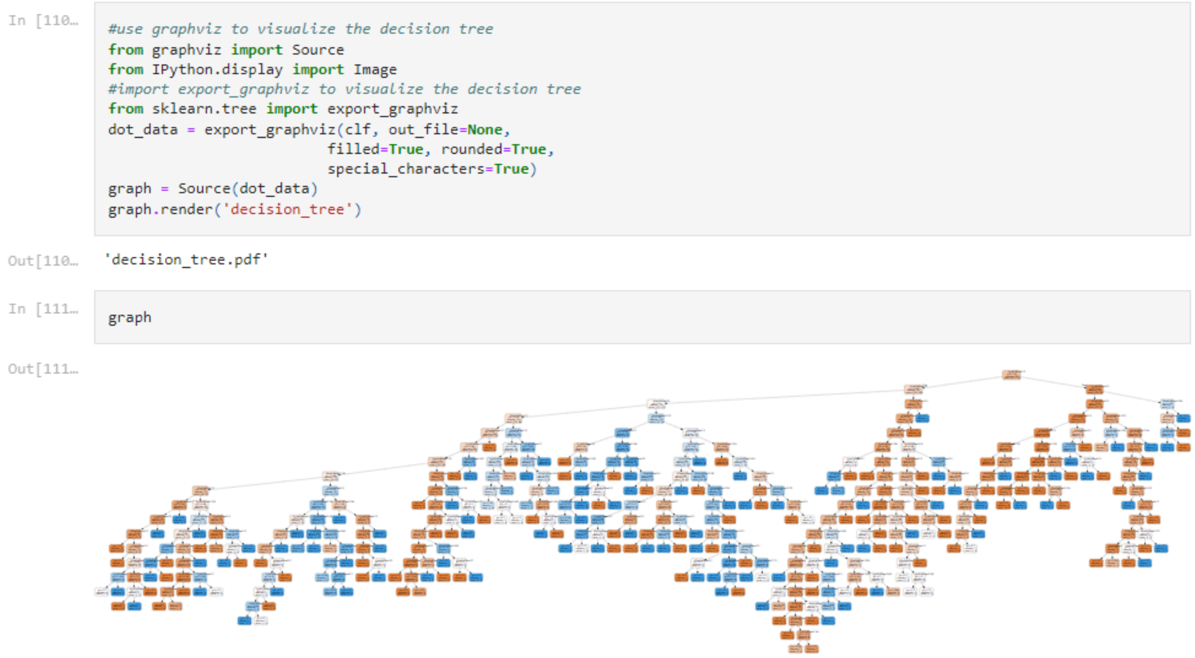
```
In [105…  #prepare the training data and test data to predict Death
          X = df.drop(['Name', 'Death'], axis=1)
          Y = df['Death']
          #split the data with 75% for training and 25% for testing
          from sklearn.model_selection import train_test_split
          X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=5613) #5613, 2913

In [106…  #use decision tree to predict Death
          from sklearn.tree import DecisionTreeClassifier
          clf = DecisionTreeClassifier()
          clf = clf.fit(X_train, Y_train)
          Y_pred = clf.predict(X_test)

In [107…  #show the confusion matrix with the precision, recall, accuracy
          from sklearn.metrics import confusion_matrix, accuracy_score, recall_score, precision_score
          confusion_matrix(Y_test, Y_pred)

Out[107…  array([[139,  17],
                 [ 28,  46]], dtype=int64)
```

- 印出訓練後的Decision Tree在測試資料集上的precision、recall和accuracy，並產生更詳細的Decision Tree分類表現報告



```
In [108…  #show the precision, recall, accuracy
          print("precision: ", precision_score(Y_test, Y_pred))
          print("recall: ", recall_score(Y_test, Y_pred))
          print("accuracy: ", accuracy_score(Y_test, Y_pred))

          precision:  0.7301587301587301
          recall:  0.6216216216216216
          accuracy:  0.8043478260869565

In [109…  #print classification report
          from sklearn.metrics import classification_report
          print(classification_report(Y_test, Y_pred))

                        precision    recall  f1-score   support

                     0       0.83      0.89      0.86       156
                     1       0.73      0.62      0.67        74

              accuracy                           0.80       230
             macro avg       0.78      0.76      0.77       230
          weighted avg       0.80      0.80      0.80       230
```

- 用graphviz套件繪製訓練出來的Decision Tree的結果



In [110…

```
#use graphviz to visualize the decision tree
from graphviz import Source
from IPython.display import Image
#import export_graphviz to visualize the decision tree
from sklearn.tree import export_graphviz
dot_data = export_graphviz(clf, out_file=None,
                           filled=True, rounded=True,
                           special_characters=True)
graph = Source(dot_data)
graph.render('decision_tree')
```

Out[110…

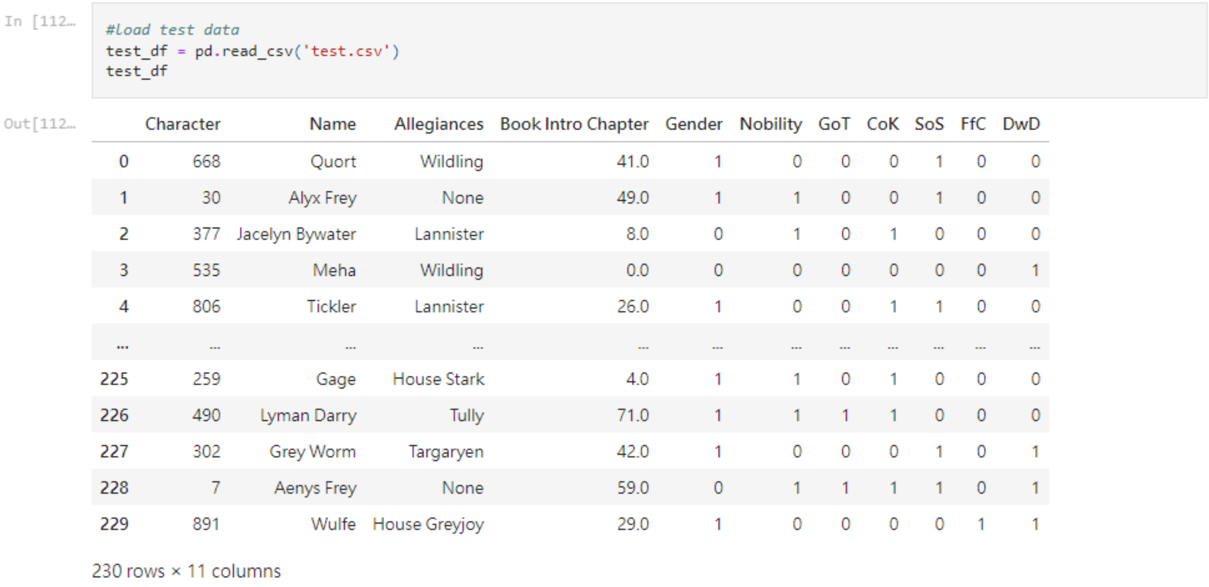
```
'decision_tree.pdf'
```

In [111…

```
graph
```

Out[111…

- 以DataFrame載入測試用的資料(之後要提交到Kaggle)



In [112…

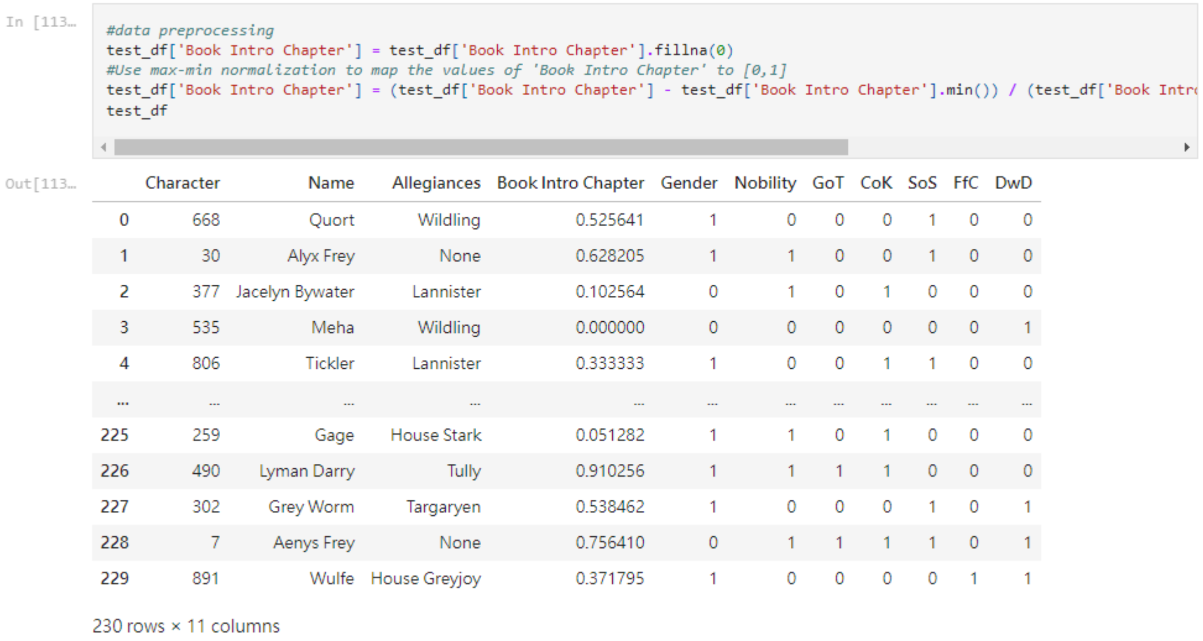
```
#load test data
test_df = pd.read_csv('test.csv')
test_df
```

Out[112…

	Character	Name	Allegiances	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD
0	668	Quort	Wildling	41.0	1	0	0	0	1	0	0
1	30	Alyx Frey	None	49.0	1	1	0	0	1	0	0
2	377	Jacelyn Bywater	Lannister	8.0	0	1	0	1	0	0	0
3	535	Meha	Wildling	0.0	0	0	0	0	0	0	1
4	806	Tickler	Lannister	26.0	1	0	0	1	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...
225	259	Gage	House Stark	4.0	1	1	0	1	0	0	0
226	490	Lyman Darry	Tully	71.0	1	1	1	1	0	0	0
227	302	Grey Worm	Targaryen	42.0	1	0	0	0	1	0	1
228	7	Aenys Frey	None	59.0	0	1	1	1	1	0	1
229	891	Wulfe	House Greyjoy	29.0	1	0	0	0	0	1	1

230 rows × 11 columns

- 像前面用一樣的方法處理資料的缺失值和做Min-Max的處理



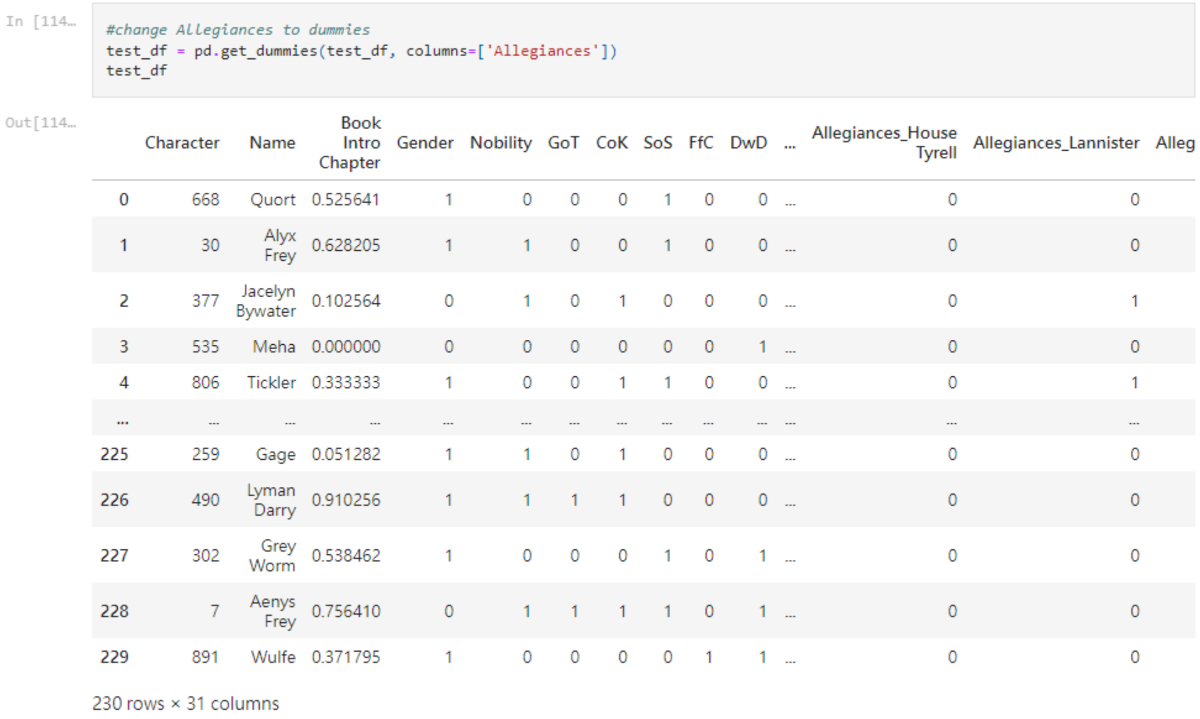
```
In [113…
#data preprocessing
test_df['Book Intro Chapter'] = test_df['Book Intro Chapter'].fillna(0)
#Use max-min normalization to map the values of 'Book Intro Chapter' to [0,1]
test_df['Book Intro Chapter'] = (test_df['Book Intro Chapter'] - test_df['Book Intro Chapter'].min()) / (test_df['Book Intr
test_df
```

Out[113…

	Character	Name	Allegiances	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD
0	668	Quort	Wildling	0.525641	1	0	0	0	1	0	0
1	30	Alyx Frey	None	0.628205	1	1	0	0	1	0	0
2	377	Jacelyn Bywater	Lannister	0.102564	0	1	0	1	0	0	0
3	535	Meha	Wildling	0.000000	0	0	0	0	0	0	1
4	806	Tickler	Lannister	0.333333	1	0	0	1	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...
225	259	Gage	House Stark	0.051282	1	1	0	1	0	0	0
226	490	Lyman Darry	Tully	0.910256	1	1	1	1	0	0	0
227	302	Grey Worm	Targaryen	0.538462	1	0	0	0	1	0	1
228	7	Aenys Frey	None	0.756410	0	1	1	1	1	0	1
229	891	Wulfe	House Greyjoy	0.371795	1	0	0	0	0	1	1

230 rows × 11 columns

- 像前面一樣針對類別資料，按照原始的屬性值，建立許多新的二元值欄位(0、1)



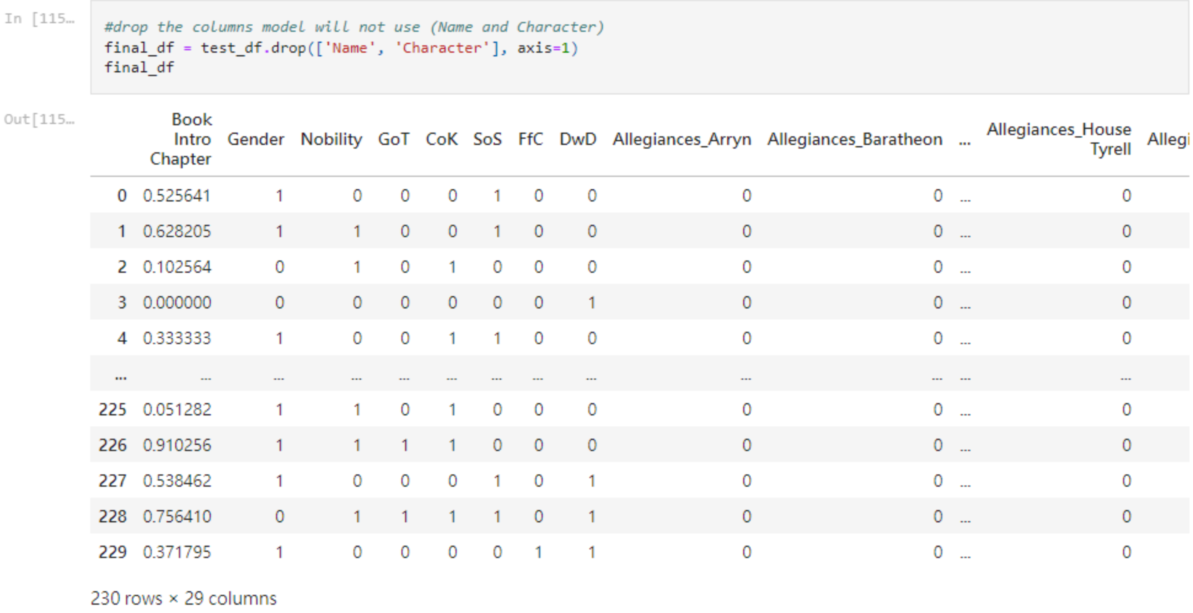
```
In [114…
#change Allegiances to dummies
test_df = pd.get_dummies(test_df, columns=['Allegiances'])
test_df
```

Out[114…

	Character	Name	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	...	Allegiances_House Tyrell	Allegiances_Lannister	Alleg
0	668	Quort	0.525641	1	0	0	0	1	0	0	...	0	0	
1	30	Alyx Frey	0.628205	1	1	0	0	1	0	0	...	0	0	
2	377	Jacelyn Bywater	0.102564	0	1	0	1	0	0	0	...	0	1	
3	535	Meha	0.000000	0	0	0	0	0	0	1	...	0	0	
4	806	Tickler	0.333333	1	0	0	1	1	0	0	...	0	1	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	
225	259	Gage	0.051282	1	1	0	1	0	0	0	...	0	0	
226	490	Lyman Darry	0.910256	1	1	1	1	0	0	0	...	0	0	
227	302	Grey Worm	0.538462	1	0	0	0	1	0	1	...	0	0	
228	7	Aenys Frey	0.756410	0	1	1	1	1	0	1	...	0	0	
229	891	Wulfe	0.371795	1	0	0	0	0	1	1	...	0	0	

230 rows × 31 columns

- 刪除預測不需要使用到的資料欄位，並產生final_df



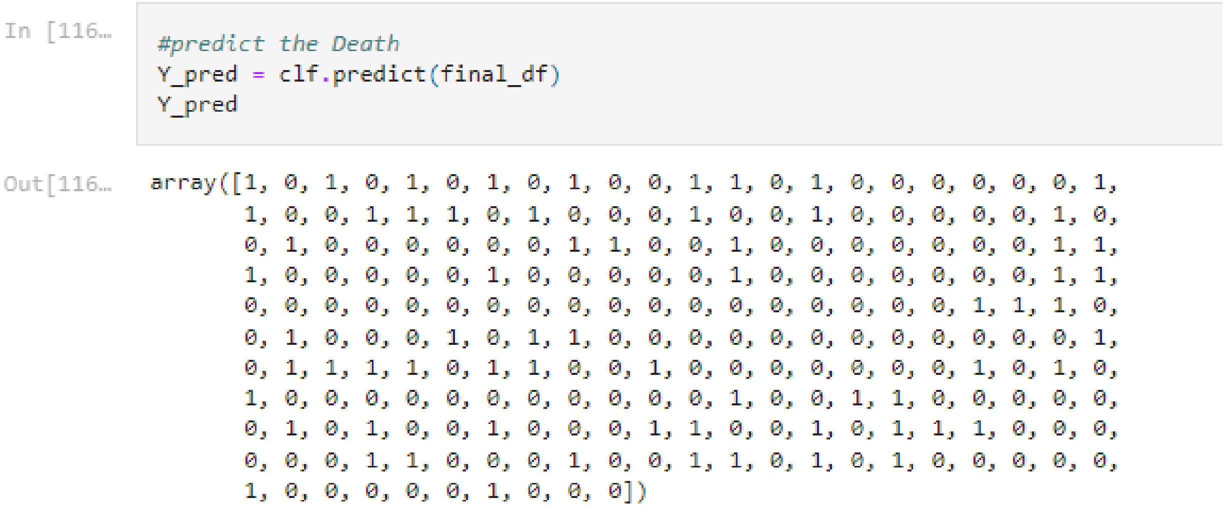
In [115…
```
#drop the columns model will not use (Name and Character)
final_df = test_df.drop(['Name', 'Character'], axis=1)
final_df
```

Out[115…

	Book Intro Chapter	Gender	Nobility	GoT	CoK	SoS	FfC	DwD	Allegiances_Arryn	Allegiances_Baratheon	...	Allegiances_House Tyrell	Allegi
0	0.525641	1	0	0	0	1	0	0	0	0	...	0	
1	0.628205	1	1	0	0	1	0	0	0	0	...	0	
2	0.102564	0	1	0	1	0	0	0	0	0	...	0	
3	0.000000	0	0	0	0	0	0	1	0	0	...	0	
4	0.333333	1	0	0	1	1	0	0	0	0	...	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	
225	0.051282	1	1	0	1	0	0	0	0	0	...	0	
226	0.910256	1	1	1	1	0	0	0	0	0	...	0	
227	0.538462	1	0	0	0	1	0	1	0	0	...	0	
228	0.756410	0	1	1	1	1	0	1	0	0	...	0	
229	0.371795	1	0	0	0	0	1	1	0	0	...	0	

230 rows × 29 columns

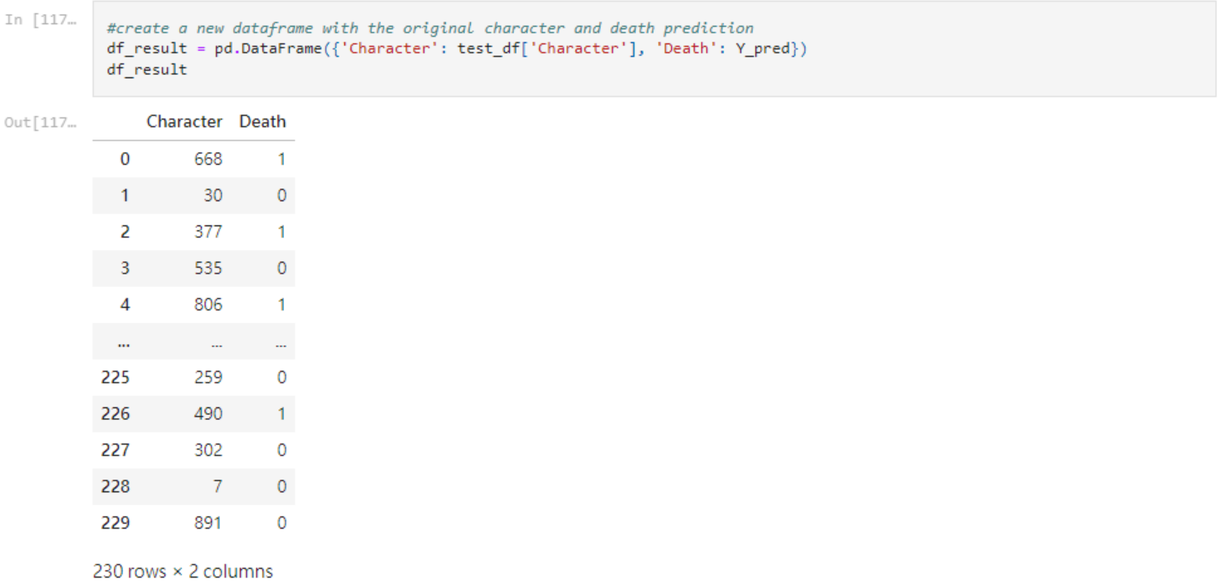
- 將final_df用剛剛訓練出來的Decision Tree進行"Death"的預測，並印出預測結果



In [116…
```
#predict the Death
Y_pred = clf.predict(final_df)
Y_pred
```

Out[116…
```
array([1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1,
       1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0,
       0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
       0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0,
       0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 1, 0, 0, 0])
```

- 將預測結果補上對應的"Character"欄位，並輸出成可以上傳到Kaggle的CSV檔

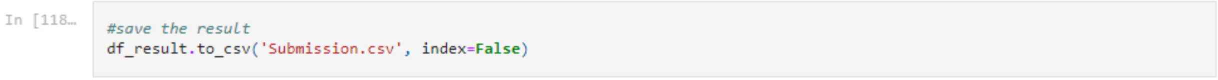


In [117…
```
#create a new dataframe with the original character and death prediction
df_result = pd.DataFrame({'Character': test_df['Character'], 'Death': Y_pred})
df_result
```

Out[117…

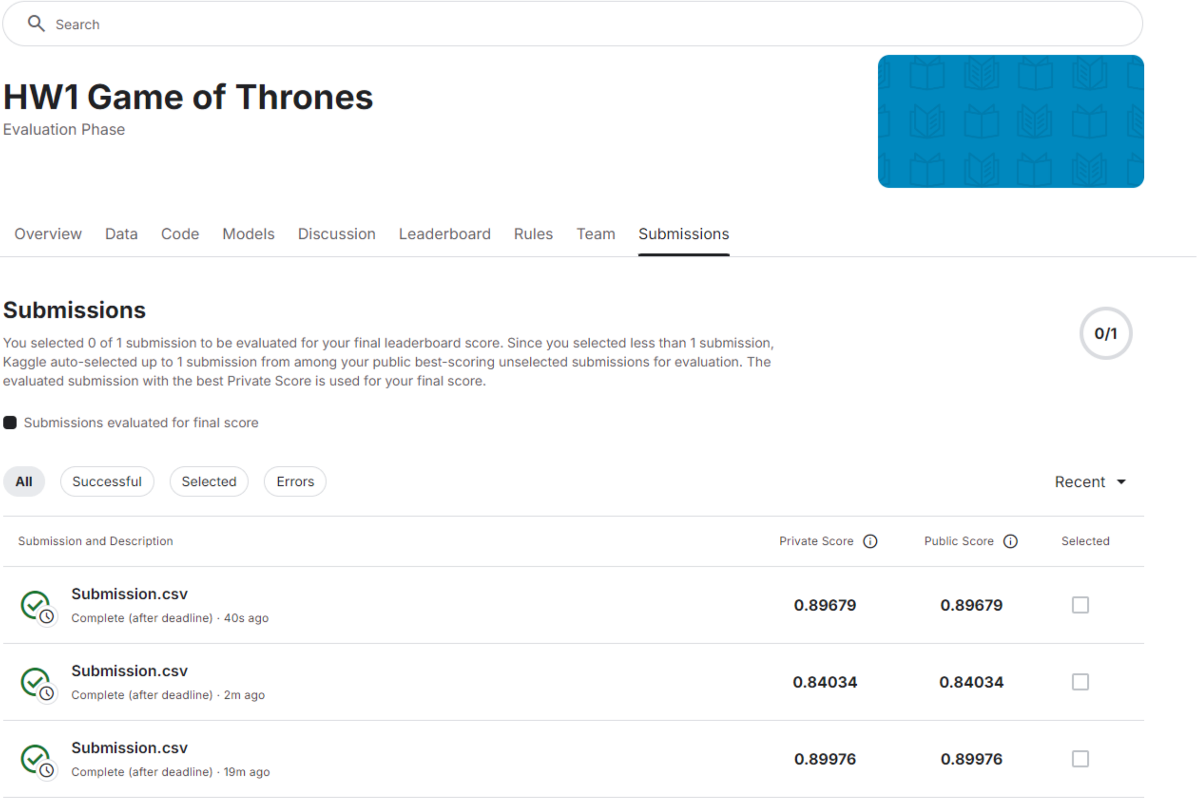
	Character	Death
0	668	1
1	30	0
2	377	1
3	535	0
4	806	1
...	...	...
225	259	0
226	490	1
227	302	0
228	7	0
229	891	0

230 rows × 2 columns



In [118…
```
#save the result
df_result.to_csv('Submission.csv', index=False)
```

- 以下是將預測結果上傳到Kaggle的結果，當中最好的表現為0.89976



HW1 Game of Thrones

Evaluation Phase

Submissions

You selected 0 of 1 submission to be evaluated for your final leaderboard score. Since you selected less than 1 submission, Kaggle auto-selected up to 1 submission from among your public best-scoring unselected submissions for evaluation. The evaluated submission with the best Private Score is used for your final score.

0/1

■ Submissions evaluated for final score

Submission and Description	Private Score	Public Score	Selected
Submission.csv Complete (after deadline) · 40s ago	0.89679	0.89679	□
Submission.csv Complete (after deadline) · 2m ago	0.84034	0.84034	□
Submission.csv Complete (after deadline) · 19m ago	0.89976	0.89976	□