大數據資料分析實作

確認資料

博雅(科技)課程

確認資料

- 建立模型的第一步就是**確認資料**。
- 想要建立**高正確性**的模型，就必須以正確的方式確認訓練資料的狀態。

- **確認方式**大致可分為以下2種：
  1. 利用**資料框(Dataframe)**的功能，以**數值**或**統計**方式進行分析。檢查有無缺失值、計算欄位值的數量，以及確認平均值與標準差等統計量。
  2. 利用**matplotlib**與**seaborn**的**繪圖**功能，以**視覺化**進行分析確認。繪製各欄位的直方圖或瞭解2個欄位關係的散佈圖等。

實作練習：鐵達尼號資料集

- 分析資料前，要先有分析資料的對象，本次練習使用「鐵達尼號資料集」
- 資料集內容為鐵達尼號郵輪的乘客名單，包含乘客的基本欄位，以及各乘客於郵輪沉沒後是否生還等欄位資料。

- **選擇此資料集的原因**：
  1. 資料的欄位數量不會太多
  2. 含有缺失值（需確認缺失值有哪些，以便之後處理）
  3. 混合數值欄位和字串欄位，正好適合用來嘗試各種統計處理

實作練習：Titanic資料集欄位名稱與意義

欄位名稱	代表意思	說明
survival	生還	0：死亡；1：存活
pclass	艙等	1：一等艙/2：二等艙/3：三等艙
sex	性別	male：男性/female：女性
age	年齡	
sibsp	手足與配偶數	同乘的兄弟姊妹與配偶數
parch	父母與子女數	同乘的父母與子女數
fare	票價	
embarked	乘船港代碼	C：Cherbourg/Q：Queenstown/S：Southampton
class	艙等名	First：一等艙/Second：二等艙/Third：三等艙
who	男女兒童	man：男性/woman：女性/child：兒童
adult_male	成人男子	True/False
deck	甲板	房艙號碼首字母（A到G）
embark_town	乘船港	Southampton/Cherbourg/Queenstown
alive	生還與否	yes/no
alone	單身	True/False

課堂實作範例操作流程

實作步驟：安裝需要的套件

- !pip install pandas：這行指令用來安裝 pandas 套件。pandas 是一個強大的 Python 資料分析工具庫，廣泛**用於資料處理和分析**，尤其是在**結構化資料**（如表格）上。

- !pip install japanize-matplotlib | tail -n 1：這行指令安裝 japanize-matplotlib 套件，這個套件用於讓 matplotlib（一個繪圖工具庫）能顯示日文或其他語言的字符，這裡的目的是**支援中文顯示**。

```
!pip install pandas
!pip install japanize-matplotlib | tail -n 1
```

補充說明

數據分析的第一步！

- 開啟數據分析的大門，要使用python做數據分析，其實現有很多很好用的package可以讓我們輕易的上手
- 介紹數據分析會使用到的幾個package，包含：
  - Pandas
  - Numpy
  - Matplotlib

補充說明

- Pandas、Numpy與Matplotlib構成了資料科學的強大基礎
- Pandas是一個基於Numpy的package，在處理數據方面非常的好用簡單，透過標籤和索引，Pandas讓我們可以非常輕易的處理數據
- Numpy是一個提供矩陣運算非常好用的工具，具備平行處理的能力，可以將操作動作一次套用在大型陣列上，幫助我們做更多方法建立多維數據以及矩陣運算，像是Pandas就是建立在Numpy的基礎延伸的套件
- Matplotlib是Python繪圖的它包含了大量的工具，可以使用這些工具創建各種圖形，包括簡單的散點圖、直方圖，甚至是三維圖形，將資料轉成圖表，在python的數據分析中會經常使用Matplotlib完成數據可視化的工作

實作步驟：載入所需的 Python 模組

- import pandas as pd：載入 pandas 模組並將其命名為 pd，通常用於**資料處理與分析**。
- import numpy as np：載入 numpy，一個支援大規模數據運算的套件，常用於**數字運算和數據處理**。

```
import pandas as pd
import numpy as np
```

實作步驟：載入所需的 Python 模組

- import matplotlib.pyplot as plt：載入 matplotlib 的 pyplot 模組，這是用來**繪圖**的工具庫。
- import japanize_matplotlib：載入 japanize_matplotlib，使 matplotlib 能顯示**日文或中文字符**。

```
import matplotlib.pyplot as plt
import japanize_matplotlib
```

實作步驟：設定numpy和pandas顯示選項

- np.set_printoptions(suppress=True, precision=4)：設定 **numpy** 顯示**浮點數的格式**。suppress=True 使得數字較小（接近零）時不顯示科學記號，precision=4 使得浮點數顯示到小數點後四位。
- pd.options.display.float_format = '{:.4f}'.format：設定 **pandas** 顯示**浮點數的格式**，將浮點數顯示為小數點後四位。
- plt.rcParams["font.size"] = 14：設定 **matplotlib** 繪圖的**字型大小**為 14。

```
np.set_printoptions(suppress=True, precision=4)
pd.options.display.float_format = '{:.4f}'.format
plt.rcParams["font.size"] = 14
```

實作步驟：載入Titanic資料集並顯示

- import seaborn as sns：載入 **seaborn** 套件，這是一個基於 matplotlib 的資料視覺化庫。
- df_titanic = sns.**load_dataset**(“titanic”)：使用 seaborn 載入 Titanic 數據集，此數據集包含了 Titanic 事件中的乘客資訊。
- print(df_titanic.head())：顯示 Titanic 數據集的前五行，以便快速了解資料的結構。

```
import seaborn as sns
df_titanic = sns.load_dataset("titanic")
print(df_titanic.head())
```

實作步驟：更改資料集的欄位名稱

- 此段程式碼是將 Titanic 數據集的欄位名稱更改為更具中文描述的名稱。這樣可以更容易理解每個欄位的內容。
- df_titanic.columns = columns_t：是將 columns_t 中定義的中文欄位名稱賦予給 df_titanic 的欄位名稱。

```
columns_t = ['生還','艙等','性別','年齡','手足與配偶數',
'父母與子女數','票價','乘船港代碼','艙等名','男女兒童',
'成人男子','甲板','乘船港','生還與否','單身']

df_titanic.columns = columns_t
print(df_titanic.head())
```

實作步驟：檢查缺失值

- df_titanic.**isnull**()：會檢查每一個欄位中的缺失值，返回一個布林值 DataFrame。
- .sum()：會統計每一個欄位中缺失值的數量，顯示**缺失值的總數**。

```
print(df_titanic.isnull().sum())
```

實作步驟：計算欄位值的數量

- df_titanic['乘船港'].**value_counts**()：計算〝乘船港〞欄位中每個值的出現次數。
- df_titanic['生還與否'].value_counts()：計算〝生還與否〞欄位中每個值的出現次數。
- 這有助於了解不同港口和生還狀況的分佈情況。

```
print(df_titanic['乘船港'].value_counts())
print()
print(df_titanic['生還與否'].value_counts())
print()
```

實作步驟：確認統計資訊

- df_titanic.**describe**()：顯示資料集中數值型欄位的**統計描述**（如平均值、標準差、最小值、最大值等），以了解各個**數值型欄位**的**分佈情況**。
  - count：每個欄位的有效數據數量（非缺失值）
  - mean：每個數值欄位的平均值。
  - std：標準差，反映數據的變異性。
  - min：每個欄位的最小值。
  - 25%：第25百分位數（即25%的數據小於此值）
  - 50%：中位數（即 50% 的數據小於此值）。
  - 75%：第75百分位數（即75%的數據小於此值）
  - max: 每個欄位的最大值。

```
display(df_titanic.describe())
```

實作步驟：聚合函式使用 - 依性別分組求均值

- df_titanic.**groupby**(‘性別’)：將 Titanic 資料集按照〝性別〞欄位進行**分組**。也就是將資料分為男性和女性兩組。
- .mean(numeric_only=True)：在每一組中，分別計算所有**數值型**欄位的**平均值**（不包含非數值型欄位，如〝性別〞欄位）。此方法會顯示每個性別組別的數值型欄位的平均值（如年齡、票價等）。

```
display(df_titanic.groupby('性別').mean(numeric_only=True))
```

- 此行程式是將資料按照〝性別〞分組，並計算每組中的數值欄位的平均值，這樣有助於**比較不同性別群體**的平均資料。

實作步驟：繪製數值型資料的直條圖

- columns_n = [...]：主要定義〝columns_n〞清單中所指定的欄位名稱，都是數值型欄位，將用於繪製直條圖。
- plt.rcParams[‘figure.figsize’] = (10,10)：設定圖形的大小。

```
columns_n = ['生還','艙等','年齡','手足與配偶數','父母與子女數','票價']
plt.rcParams['figure.figsize'] = (10,10)
```

實作步驟：繪製數值型資料的直條圖

- df_titanic[columns_n].hist()：對 df_titanic 中〝columns_n〞清單中的欄位（如 生還、艙等 等）繪製直條圖。而直條圖會顯示每個數值欄位的數據分佈情況。【補充說明：特別是對於**連續型數據**，直條圖可幫助**理解數據的分佈**。】
- plt.show()：顯示繪製的圖形。這是 matplotlib 中的基本方法，用來將所有繪製的圖形顯示出來。

```
df_titanic[columns_n].hist()
plt.show()
```

實作步驟：繪製分類(非數值型)資料的直條圖

- 透過 for 迴圈來執行〝columns_c〞清單中的每一個欄位名稱。**enumerate**() 函數會返回每個欄位名稱及其索引（i）。
- ax = plt.subplot(2, 2, i+1)：用來建立一個 2x2 的子圖（總共 4 個圖形區域）。i+1 是為了確保每個直條圖都顯示在正確的位置。

```
columns_c = ['性別','乘船港','艙等名','成人男子']
plt.rcParams['figure.figsize'] = (8,8)
for i, name in enumerate(columns_c):
    ax = plt.subplot(2, 2, i+1)
```

實作步驟：繪製分類(非數值型)資料的直條圖

- df_titanic[**name**].**value_counts**()：用來計算指定欄位（例如 性別）中各類別的出現次數。
- .plot(kind='bar')：將計算出的分類數據繪製為直條圖（**kind='bar'** 表示直條圖）。
- title=name：用「當前欄位名稱」設定為圖的標題。
- ax=ax 設定該直條圖繪製到對應的子圖區域。

```
df_titanic[name].value_counts().plot(kind='bar', title=name, ax=ax)
```

補充說明：ax子圖區域

- 在 matplotlib 中繪製多個圖形時，會使用 子圖（subplot） 來將一個畫布（Figure）分割成多個小區域，每個區域中顯示一個獨立的圖形。
- 在 matplotlib 中，ax 代表每個子圖的座標軸對象（Axes），也就是圖表的繪製區域。

```
ax = plt.subplot(2, 2, i+1)
```

- plt.subplot()：用來建立子圖，程式為 **plt.subplot(nrows, ncols, index)**。表示將畫布分割成 **nrows 行**和 **ncols 列**的子區域，並將繪圖位置放在 **index 所指定的位置**。
  - nrows=2：表示子圖有 2 行。
  - ncols=2：表示子圖有 2 列。
  - i+1：index 用來指定當前繪圖的位置，從 1 開始計數。

補充說明：子圖 (Subplot) 的排列

- 當 plt.subplot (2, 2, i+1) 被使用時，畫布會被分成了 2 行 2 列的區域，即一共 4 個區域。

- i+1 的變化會讓每個直條圖分別顯示在 4 個區域中。

  ✓第一次循環：i=0 → plt.subplot (2, 2, 1) 會將圖形繪製在第一個子圖（左上角）

  ✓第二次循環：i=1 → plt.subplot (2, 2, 2) 會將圖形繪製在第二個子圖（右上角）

  ✓第三次循環：i=2 → plt.subplot (2, 2, 3) 會將圖形繪製在第三個子圖（左下角）

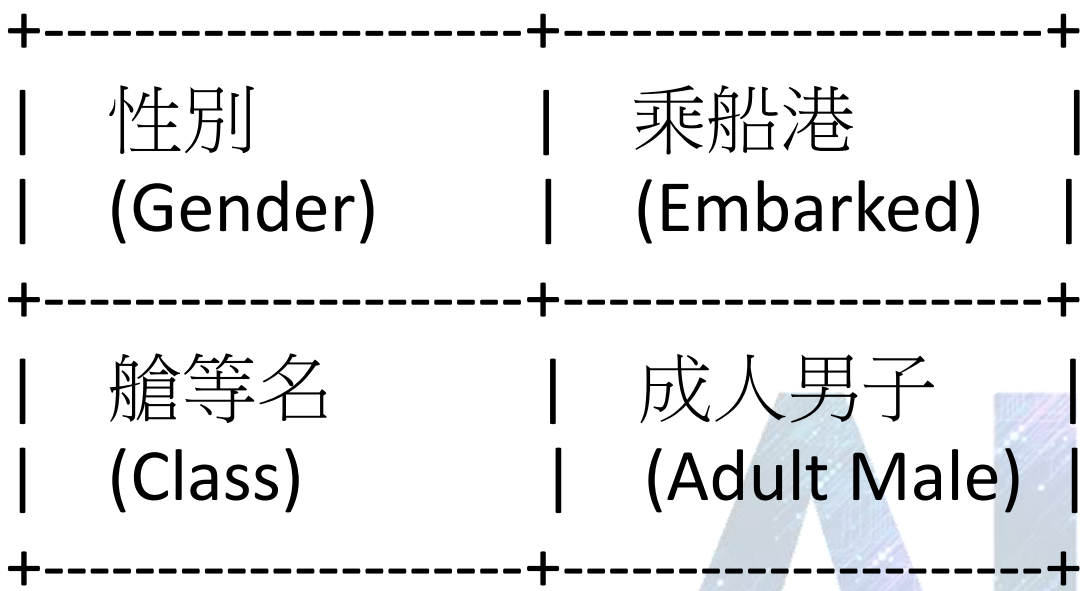
  ✓第四次循環：i=3 → plt.subplot (2, 2, 4) 會將圖形繪製在第四個子圖（右下角）

補充說明：子圖 (Subplot) 的排列

- 指定columns_c = [‘性別’, ‘乘船港’, ‘艙等名’, ‘成人男子’]，將會得到如下 2x2 的排列，分別顯示這些欄位的直條圖

- ax 代表子圖的區域對象。而 **plt.subplot(nrows, ncols, index)** 用來將畫布**分割成多個子區域**，並在指定的位置繪製圖形。

- 子圖功能非常有用，尤其是在同一個畫布中顯示多個圖形時，可以方便**比較不同變數或不同數據**的**視覺化**效果。



實作步驟：繪製分類(非數值型)資料的直條圖

- plt.**tight_layout**()：**自動調整**子圖之間的**間距**，以確保不會重疊，讓圖表顯示得更加清晰。
- plt.show()：顯示繪製的圖形。

```
plt.tight_layout()
plt.show()
```

- 這幾行程式碼是用來繪製〝分類〞欄位（如 性別、乘船港 等）的直條圖，顯示每個類別的頻率次數分佈情況。同時使用子圖讓這些圖形在一個畫布中顯示，以便於比較。

「大數據資料分析實作」課程