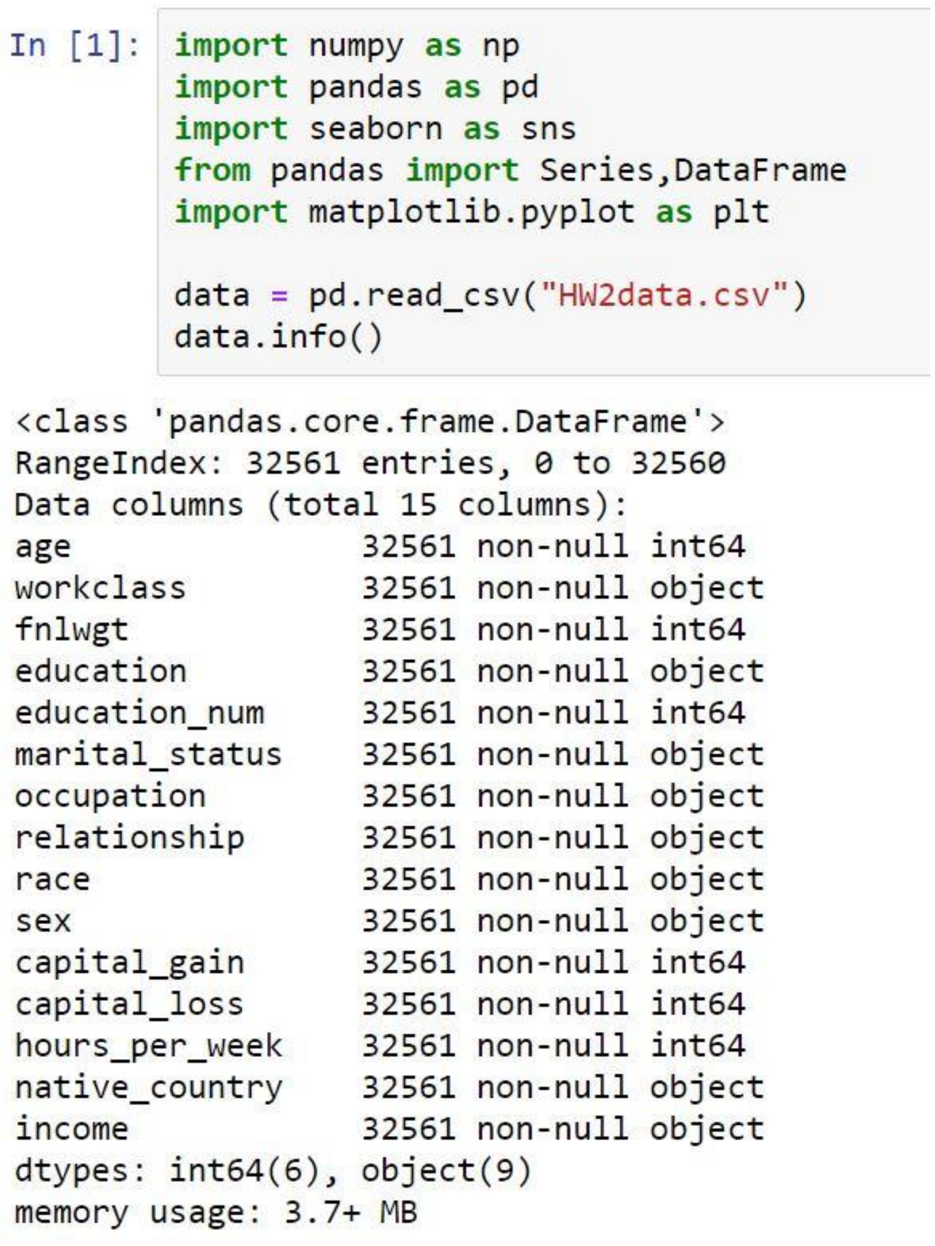
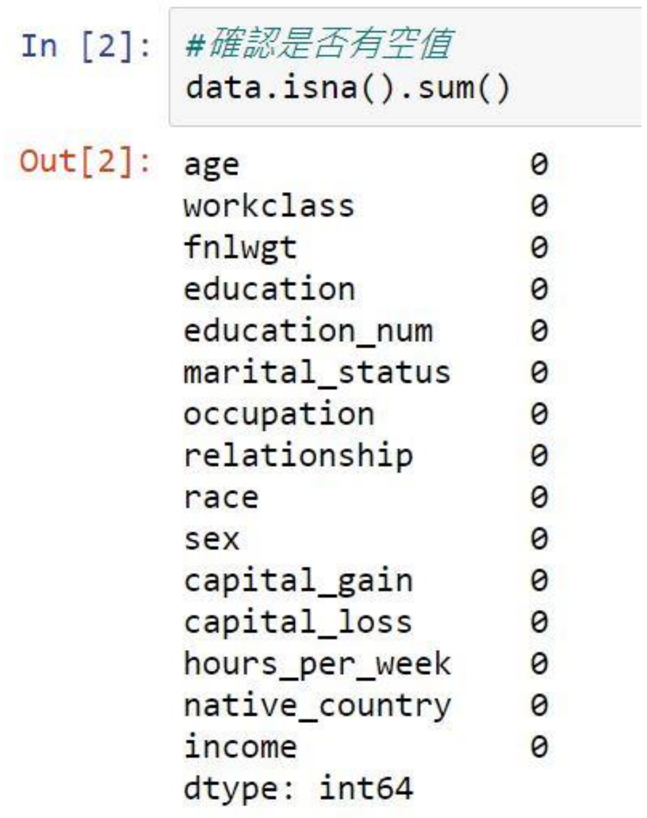
先 import 可能會用到的套件，接著利用 pandas 套件將資料讀取進來，透過 info()來看 HW2data.csv 的資訊，如欄位的型態(object、float64…等)、資料的數量…等。



```
In [1]: import numpy as np
        import pandas as pd
        import seaborn as sns
        from pandas import Series,DataFrame
        import matplotlib.pyplot as plt

        data = pd.read_csv("HW2data.csv")
        data.info()
```

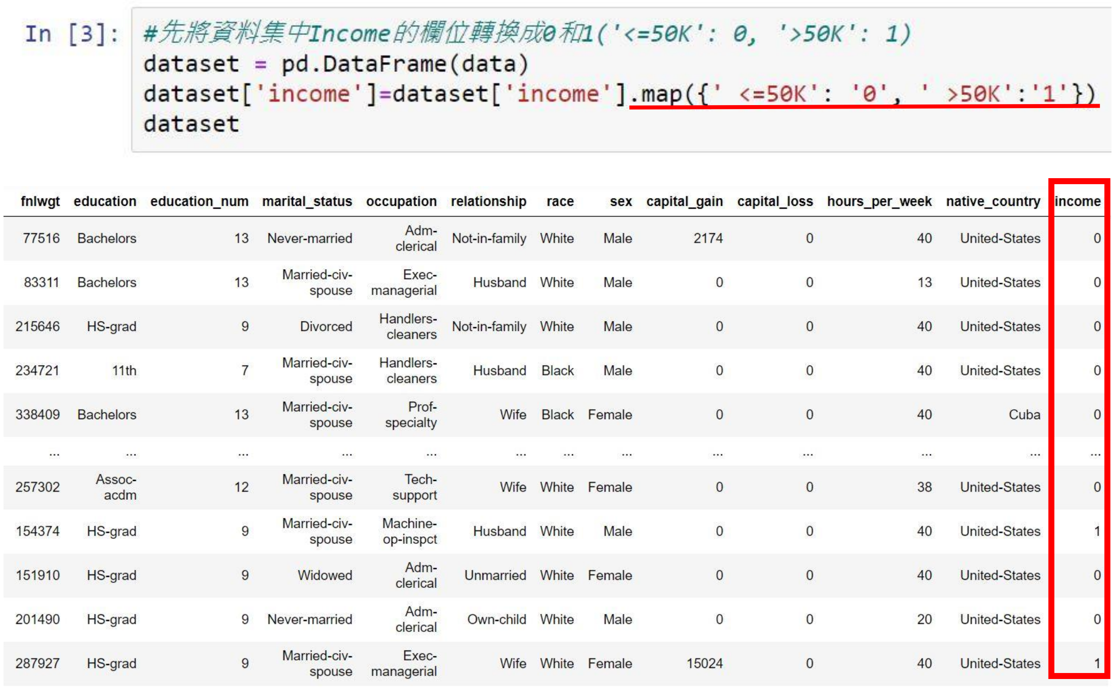
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 32561 entries, 0 to 32560
Data columns (total 15 columns):
age                  32561 non-null int64
workclass            32561 non-null object
fnlwgt               32561 non-null int64
education            32561 non-null object
education_num        32561 non-null int64
marital_status       32561 non-null object
occupation           32561 non-null object
relationship         32561 non-null object
race                 32561 non-null object
sex                  32561 non-null object
capital_gain         32561 non-null int64
capital_loss         32561 non-null int64
hours_per_week       32561 non-null int64
native_country       32561 non-null object
income               32561 non-null object
dtypes: int64(6), object(9)
memory usage: 3.7+ MB
```

再來進入資料前處理的階段；先利用 isna().sum()來看哪些欄位是有空值存在，可以看出此資料集沒有空值的存在。



```
In [2]: #確認是否有空值
        data.isna().sum()

Out[2]: age               0
        workclass         0
        fnlwgt            0
        education         0
        education_num     0
        marital_status    0
        occupation        0
        relationship      0
        race              0
        sex               0
        capital_gain      0
        capital_loss      0
        hours_per_week    0
        native_country    0
        income            0
        dtype: int64
```

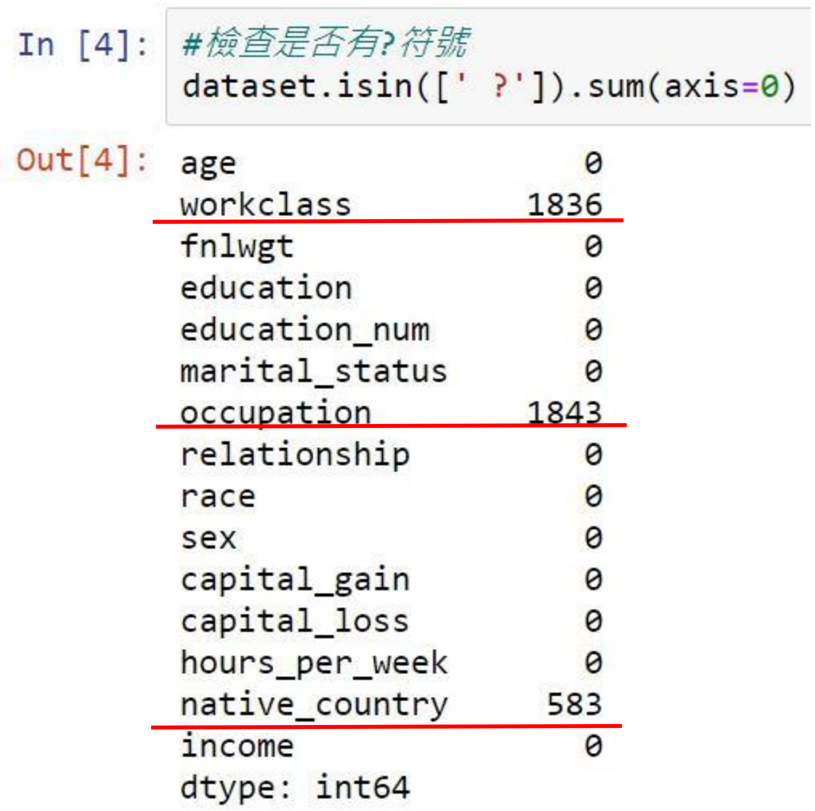
將資料集中 Income 的欄位中原本的值轉換成 0 和 1('<=50K': 0, '>50K': 1)



```
In [3]: #先將資料集中Income的欄位轉換成0和1('<=50K': 0, '>50K': 1)
        dataset = pd.DataFrame(data)
        dataset['income']=dataset['income'].map({' <=50K': '0', ' >50K':'1'})
        dataset
```

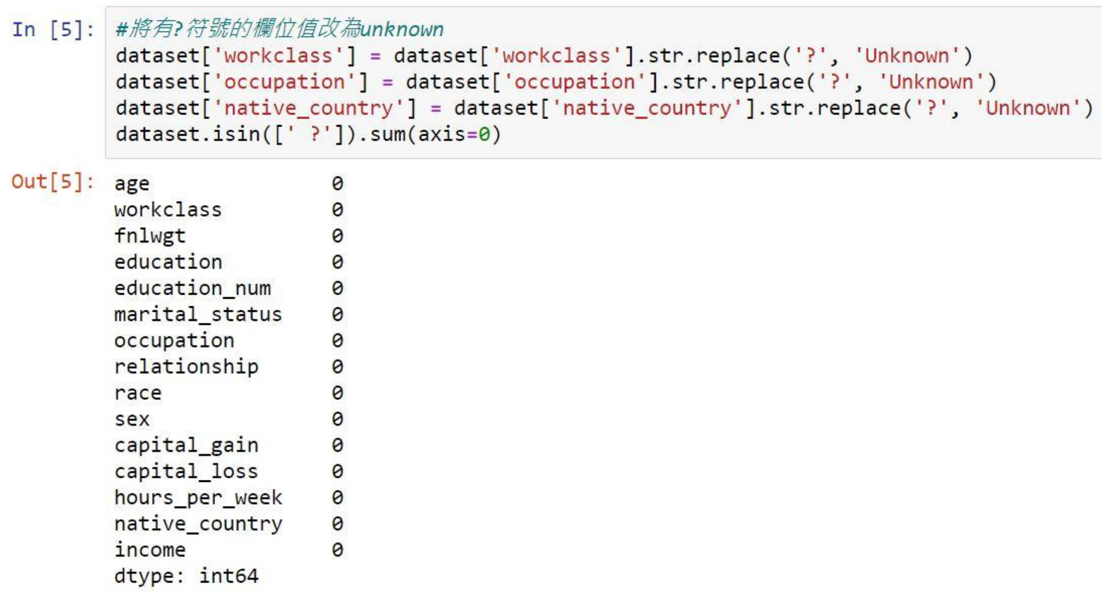
fnlwgt	education	education_num	marital_status	occupation	relationship	race	sex	capital_gain	capital_loss	hours_per_week	native_country	income
77516	Bachelors	13	Never-married	Adm-clerical	Not-in-family	White	Male	2174	0	40	United-States	0
83311	Bachelors	13	Married-civ-spouse	Exec-managerial	Husband	White	Male	0	0	13	United-States	0
215646	HS-grad	9	Divorced	Handlers-cleaners	Not-in-family	White	Male	0	0	40	United-States	0
234721	11th	7	Married-civ-spouse	Handlers-cleaners	Husband	Black	Male	0	0	40	United-States	0
338409	Bachelors	13	Married-civ-spouse	Prof-specialty	Wife	Black	Female	0	0	40	Cuba	0
...	...	...	...	...	...	...	...	...	...	...	...	...
257302	Assoc-acdm	12	Married-civ-spouse	Tech-support	Wife	White	Female	0	0	38	United-States	0
154374	HS-grad	9	Married-civ-spouse	Machine-op-inspct	Husband	White	Male	0	0	40	United-States	1
151910	HS-grad	9	Widowed	Adm-clerical	Unmarried	White	Female	0	0	40	United-States	0
201490	HS-grad	9	Never-married	Adm-clerical	Own-child	White	Male	0	0	20	United-States	0
287927	HS-grad	9	Married-civ-spouse	Exec-managerial	Wife	White	Female	15024	0	40	United-States	1

雖然資料集沒有空值，但是有?的符號，代表該欄位的值是未知，所以要檢查哪些屬性有?符號



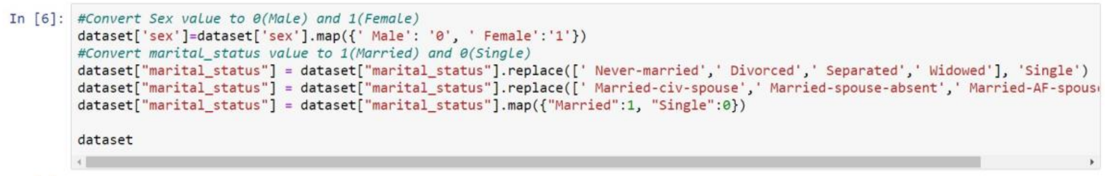
```
In [4]:  #檢查是否有?符號
         dataset.isin([' ?']).sum(axis=0)

Out[4]:  age                  0
         workclass         1836
         fnlwgt               0
         education            0
         education_num        0
         marital_status       0
         occupation        1843
         relationship         0
         race                 0
         sex                  0
         capital_gain         0
         capital_loss         0
         hours_per_week       0
         native_country     583
         income               0
         dtype: int64
```

將有?符號的屬性欄位值改為 unknown

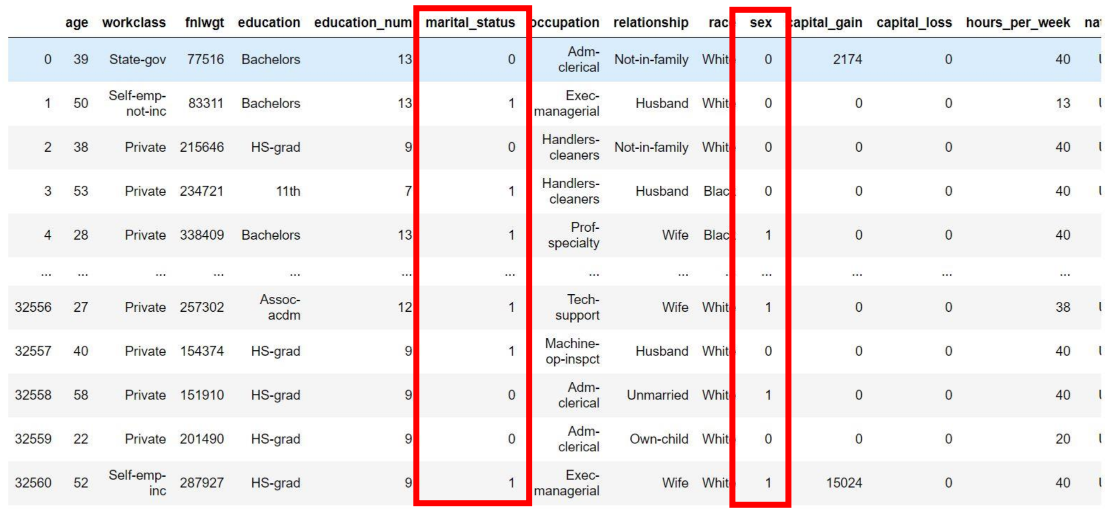


```
In [5]:  #將有?符號的欄位值改為unknown
         dataset['workclass'] = dataset['workclass'].str.replace('?', 'Unknown')
         dataset['occupation'] = dataset['occupation'].str.replace('?', 'Unknown')
         dataset['native_country'] = dataset['native_country'].str.replace('?', 'Unknown')
         dataset.isin([' ?']).sum(axis=0)

Out[5]:  age               0
         workclass         0
         fnlwgt            0
         education         0
         education_num     0
         marital_status    0
         occupation        0
         relationship      0
         race              0
         sex               0
         capital_gain      0
         capital_loss      0
         hours_per_week    0
         native_country    0
         income            0
         dtype: int64
```

接著將一些欄位原本屬性為 object 的值轉成 0 和 1；sex 的欄位中，將 Male 轉成 0，Female 轉成 1；marital_status 的欄位中，先將 Never-married、Divorced、Separated、Widowed 用 Single 表示，然後 Married-civ-spouse、Married-spouse-absent、Married-AF-spouse 用 Married 表示，再將 Married 轉成 1，Single 轉成 0



```
In [6]: #Convert Sex value to 0(Male) and 1(Female)
        dataset['sex']=dataset['sex'].map({' Male': '0', ' Female':'1'})
        #Convert marital_status value to 1(Married) and 0(Single)
        dataset["marital_status"] = dataset["marital_status"].replace([' Never-married',' Divorced',' Separated',' Widowed'], 'Single')
        dataset["marital_status"] = dataset["marital_status"].replace([' Married-civ-spouse',' Married-spouse-absent',' Married-AF-spous
        dataset["marital_status"] = dataset["marital_status"].map({"Married":1, "Single":0})

        dataset
```



	age	workclass	fnlwgt	education	education_num	marital_status	occupation	relationship	rac	sex	capital_gain	capital_loss	hours_per_week	na
0	39	State-gov	77516	Bachelors	13	0	Adm-clerical	Not-in-family	Whit	0	2174	0	40	U
1	50	Self-emp-not-inc	83311	Bachelors	13	1	Exec-managerial	Husband	Whit	0	0	0	13	U
2	38	Private	215646	HS-grad	9	0	Handlers-cleaners	Not-in-family	Whit	0	0	0	40	U
3	53	Private	234721	11th	7	1	Handlers-cleaners	Husband	Blac	0	0	0	40	U
4	28	Private	338409	Bachelors	13	1	Prof-specialty	Wife	Blac	1	0	0	40	
...	...	...	...	...	...	...	...	...	.	...	...	...	...	
32556	27	Private	257302	Assoc-acdm	12	1	Tech-support	Wife	Whit	1	0	0	38	U
32557	40	Private	154374	HS-grad	9	1	Machine-op-inspct	Husband	Whit	0	0	0	40	U
32558	58	Private	151910	HS-grad	9	0	Adm-clerical	Unmarried	Whit	1	0	0	40	U
32559	22	Private	201490	HS-grad	9	0	Adm-clerical	Own-child	Whit	0	0	0	20	U
32560	52	Self-emp-inc	287927	HS-grad	9	1	Exec-managerial	Wife	Whit	1	15024	0	40	U

接著將無助於分類的欄位 drop 掉

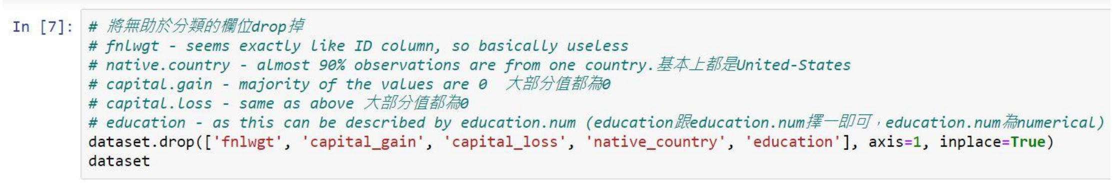
# fnlwgt － 類似 ID，故可 drop 掉

# native.country － 幾乎 90%都是 United-States，較無助於分析

# capital.gain － 大部分值都為 0，較無助於分析

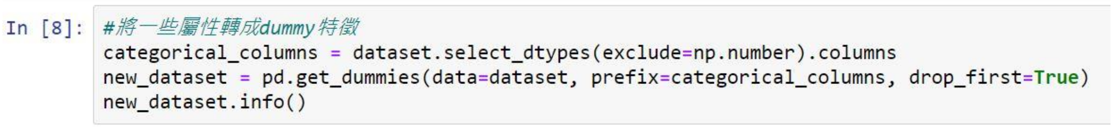
# capital.loss － 大部分值都為 0，較無助於分析

# education － education 跟 education.num 擇一即可，因 education.num 為 numerical，所以選擇 drop 掉 education



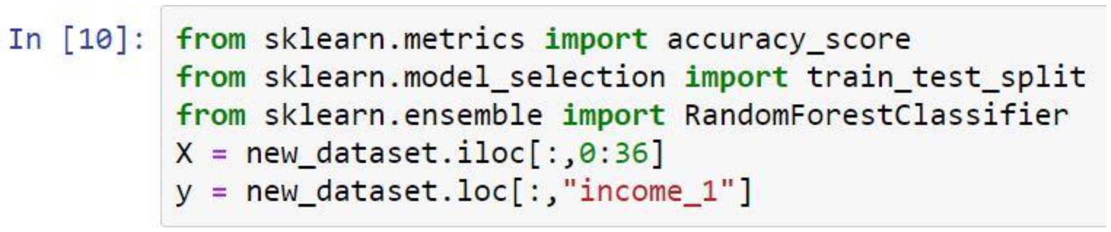
```
In [7]: # 將無助於分類的欄位drop掉
        # fnlwgt - seems exactly like ID column, so basically useless
        # native.country - almost 90% observations are from one country.基本上都是United-States
        # capital.gain - majority of the values are 0  大部分值都為0
        # capital.loss - same as above 大部分值都為0
        # education - as this can be described by education.num (education跟education.num擇一即可，education.num為numerical)
        dataset.drop(['fnlwgt', 'capital_gain', 'capital_loss', 'native_country', 'education'], axis=1, inplace=True)
        dataset
```

將一些屬性轉成 dummy 特徵



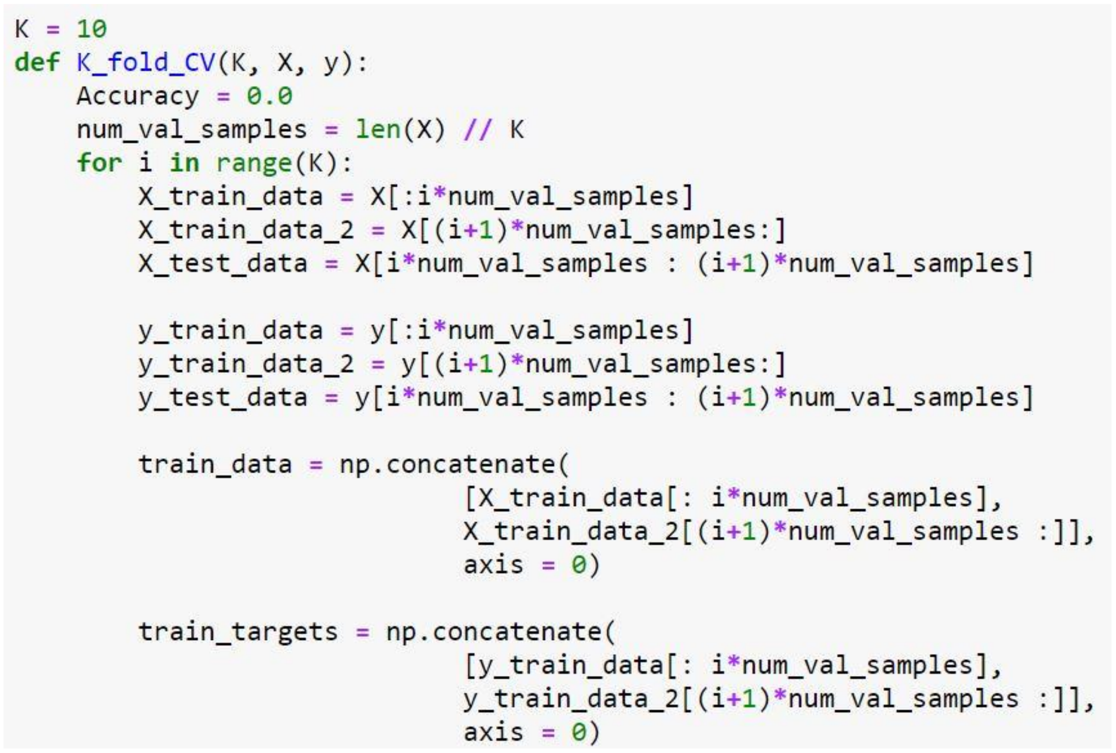
```
In [8]: #將一些屬性轉成dummy特徵
        categorical_columns = dataset.select_dtypes(exclude=np.number).columns
        new_dataset = pd.get_dummies(data=dataset, prefix=categorical_columns, drop_first=True)
        new_dataset.info()
```

Income 欄位為目標

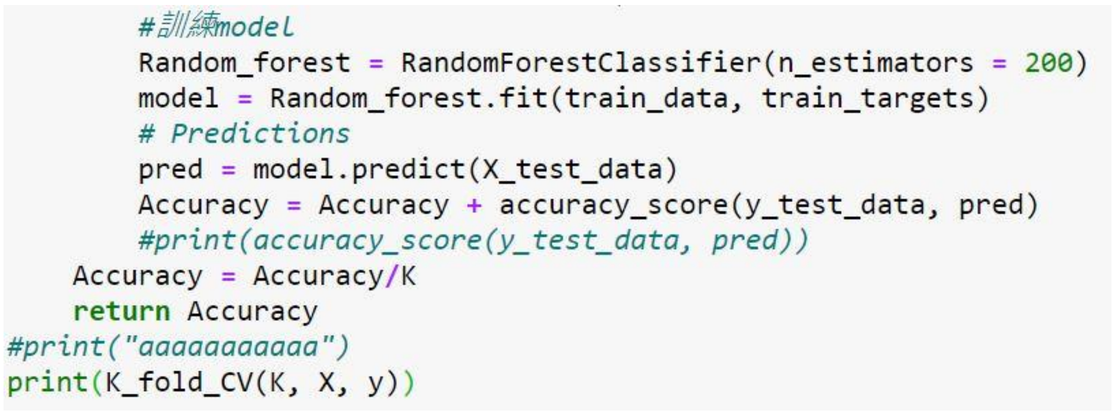


```
In [10]: from sklearn.metrics import accuracy_score
         from sklearn.model_selection import train_test_split
         from sklearn.ensemble import RandomForestClassifier
         X = new_dataset.iloc[:,0:36]
         y = new_dataset.loc[:,"income_1"]
```

自行撰寫 function 進行 k-fold cross-validation(不可使用套件)並計算 Accuracy，例如資料有 100 筆，testing set 在本次 iteration 取第 1 到 10 筆，則 training set 為第 11 到 100 筆；下次 testing set 為 11~20，training set 為 21~100 & 1~10；會使用 Random Forest 進行分類

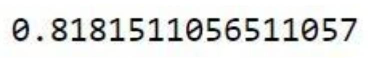


```
K = 10
def K_fold_CV(K, X, y):
    Accuracy = 0.0
    num_val_samples = len(X) // K
    for i in range(K):
        X_train_data = X[:i*num_val_samples]
        X_train_data_2 = X[(i+1)*num_val_samples:]
        X_test_data = X[i*num_val_samples : (i+1)*num_val_samples]

        y_train_data = y[:i*num_val_samples]
        y_train_data_2 = y[(i+1)*num_val_samples:]
        y_test_data = y[i*num_val_samples : (i+1)*num_val_samples]

        train_data = np.concatenate(
                             [X_train_data[: i*num_val_samples],
                             X_train_data_2[(i+1)*num_val_samples :]],
                             axis = 0)

        train_targets = np.concatenate(
                             [y_train_data[: i*num_val_samples],
                             y_train_data_2[(i+1)*num_val_samples :]],
                             axis = 0)
```



```
        #訓練model
        Random_forest = RandomForestClassifier(n_estimators = 200)
        model = Random_forest.fit(train_data, train_targets)
        # Predictions
        pred = model.predict(X_test_data)
        Accuracy = Accuracy + accuracy_score(y_test_data, pred)
        #print(accuracy_score(y_test_data, pred))
    Accuracy = Accuracy/K
    return Accuracy
#print("aaaaaaaaaaa")
print(K_fold_CV(K, X, y))
```

最後平均下來的準確率大概 81.8%



```
0.8181511056511057
```