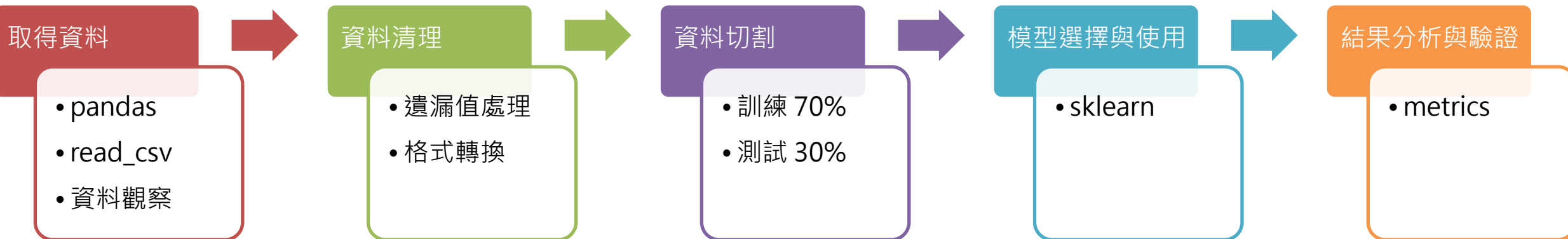


機器學習運作流程



練習一：房價預測



Linear Regression

練習二：鐵達尼號生存預測



Logistic Regression

練習：房價預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

```
#import modules
```

```
import pandas as pd
```

```
import numpy as np
```

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

```
;%matplotlib inline
```

```
import seaborn as sns
```

```
#import dataset
```

```
df = pd.read_csv("data/Housing_Dataset_Sample.csv")
```

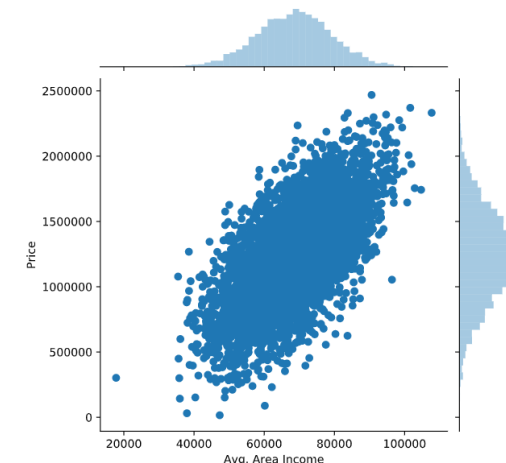
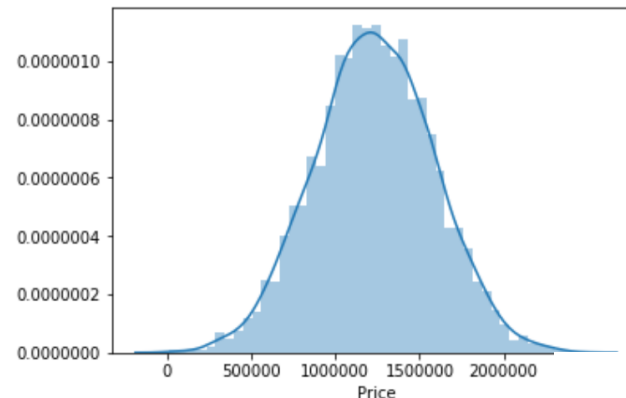
```
#observing dataset
```

```
df.head()
```

```
df.describe().T
```

```
sns.distplot(df['Price'])
```

```
sns.jointplot(df['Avg. Area Income'],df['Price'])
```



練習：房價預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

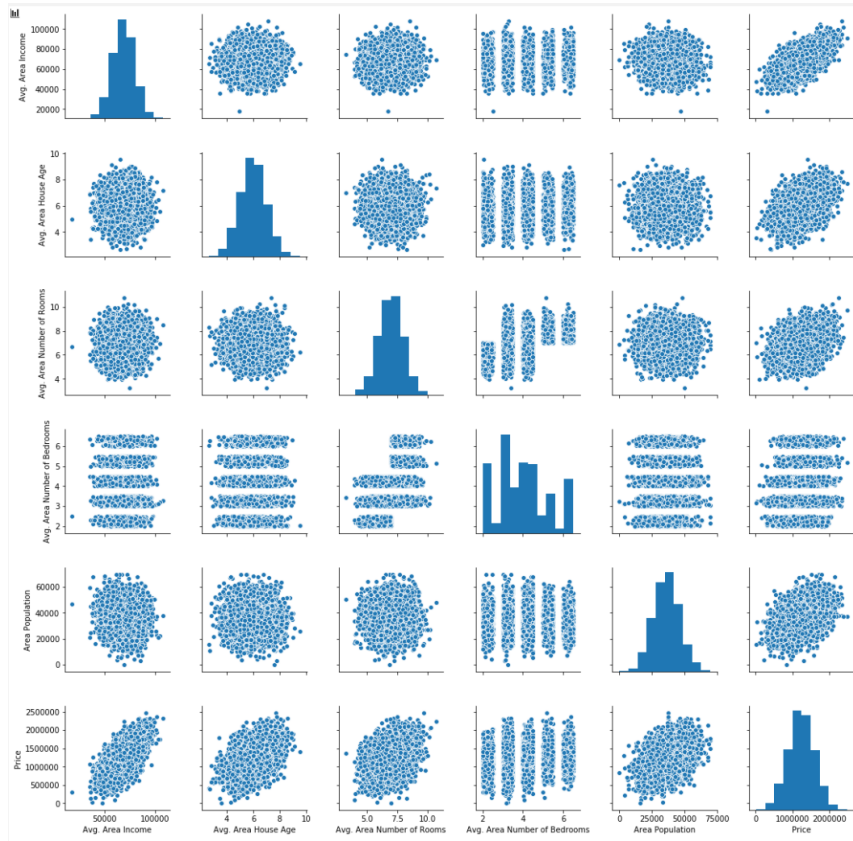
模型選擇與使用

- sklearn

結果分析與驗證

- metrics

`sns.pairplot(df)`



練習：房價預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

#prepare to train model

#X是所有可能的影響變因

#取得所有的列的0,1,2,3,4欄位

```
X = df.iloc[:, :5]
```

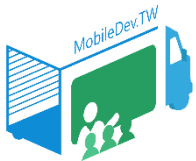
#y是目標值

```
y = df['Price']
```

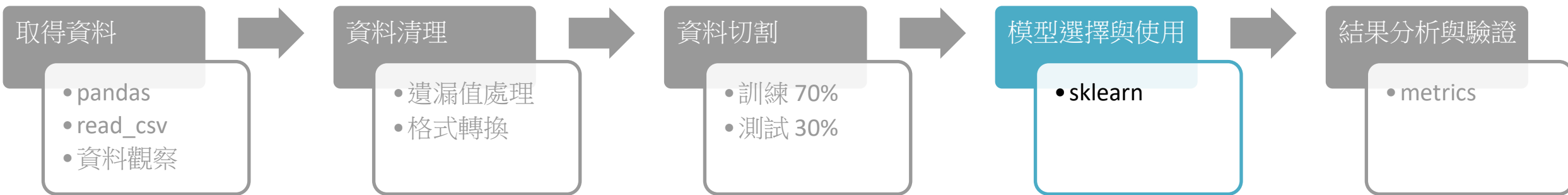
#split to training data & testing data

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=54)
```



練習：房價預測



#using linear regression model

```
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(X_train, y_train)
```

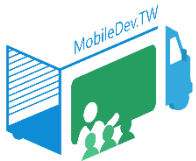
#get the result

```
predictions = reg.predict(X_test)
predictions
```

[24] predictions



```
array([ 614607.96220755, 1849444.80372635, 1118945.08884266, ...,
        834789.03428584, 1787928.10906905, 1455422.23696488])
```



練習：房價預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

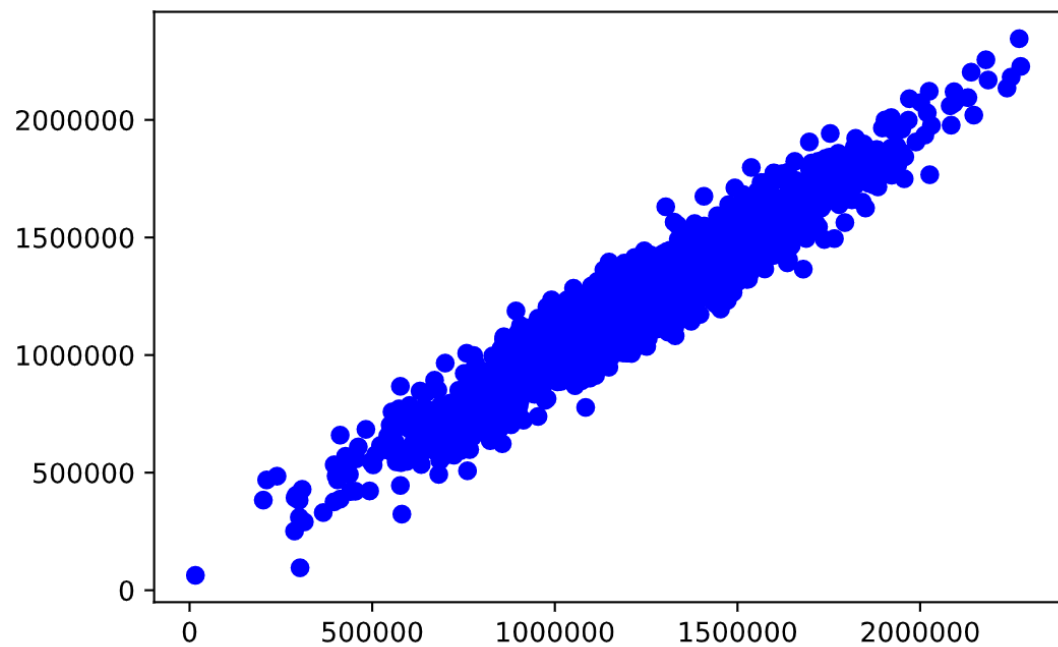
- sklearn

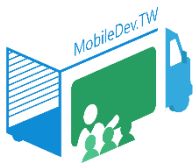
結果分析與驗證

- metrics

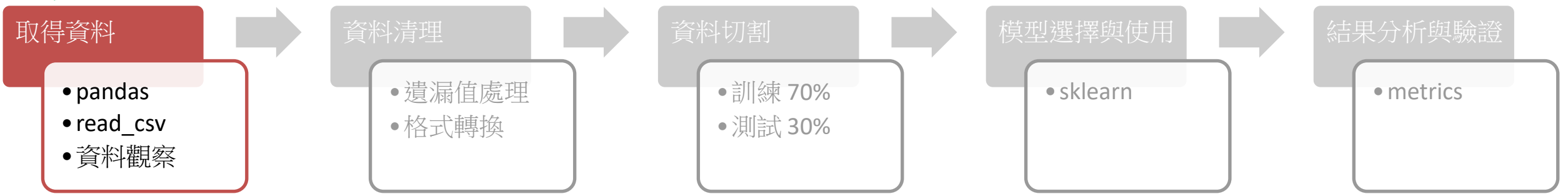
```
from sklearn.metrics import r2_score  
r2_score(y_test, predictions)  
plt.scatter(y_test, predictions, color='blue')
```

0.9216604865707106





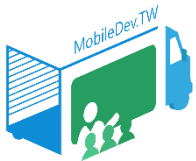
練習：鐵達尼號生存預測



```
#import modules
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

#import dataset
df = pd.read_csv("data/train_data_titanic.csv")
df.head()
df.info()
```

功能變數名稱	說明
PassengerId	乘客編號
Survived	是否存活(0 : 否、1 : 是)
Pclass	船票等級(1等、2等、3等)
Name	乘客姓名
Sex	性別
Age	年齡
Sibsp	有多少兄弟姊妹/配偶在船上
Parch	有多少父母/小孩在船上
Ticket	船票編號
Fare	票價
Cabin	艙房編號
Embarked	登船港口 C 瑟堡 Q 皇后鎮 S南安普頓



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

df.head()

	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



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

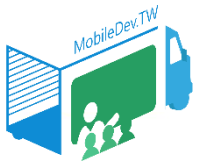
- sklearn

結果分析與驗證

- metrics

df.info()

```
<class 'pandas.core.frame.DataFrame'>
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
```



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

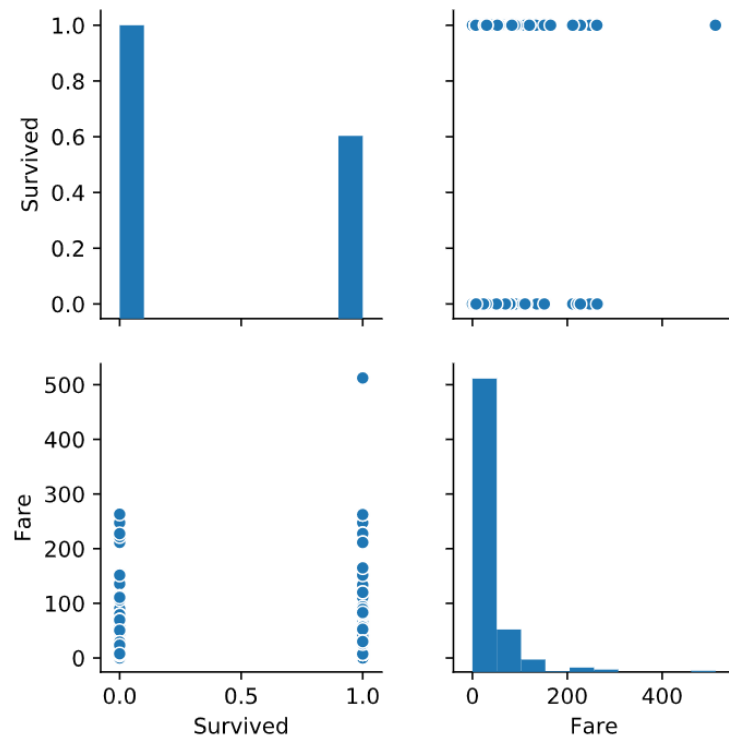
- metrics

#Remove the columns model will not use

```
df.drop(['Name', 'Ticket'], axis=1, inplace=True)  
df.head()
```

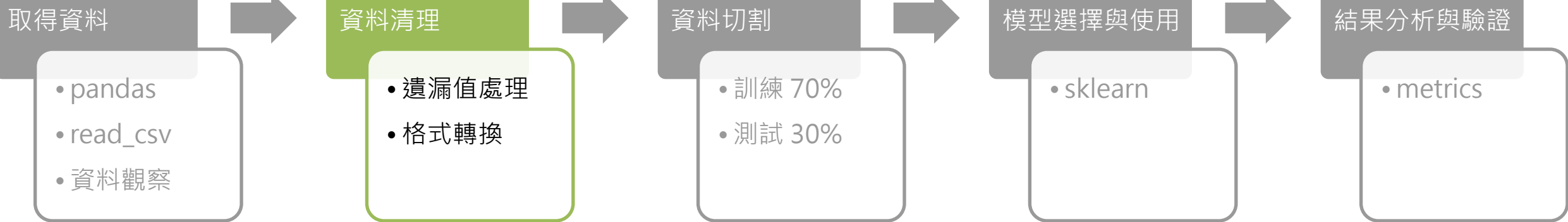
```
sns.pairplot(df[['Survived', 'Fare']], dropna=True)
```

練習：請嘗試觀察
其他欄位與Survived之間的關聯





練習：鐵達尼號生存預測



#Remove the columns model will not use

```
df.drop(['Name', 'Ticket'], axis=1, inplace=True)
df.head()
```

```
sns.pairplot(df[['Survived', 'Fare']], dropna=True)
```

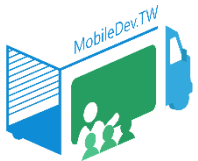
#data observing

```
df.groupby('Survived').mean()
```

存活者
平均年齡稍低一些!
票價平均較高一些!

```
df.groupby('Survived').mean()
```

	PassengerId	Pclass	Age	SibSp	Parch	Fare
Survived						
0	447.016393	2.531876	30.626179	0.553734	0.329690	22.117887
1	444.368421	1.950292	28.343690	0.473684	0.464912	48.395408



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

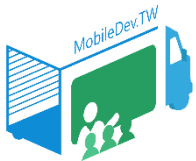
#data observing

```
df['SibSp'].value_counts()  
df['Parch'].value_counts()  
df['Sex'].value_counts()
```

```
0    608  
1    209  
2     28  
4     18  
3     16  
8       7  
5        5  
Name: SibSp, dtype: int64
```

```
0     678  
1     118  
2      80  
5        5  
3         5  
4         4  
6         1  
Name: Parch, dtype: int64
```

```
male      577  
female   314  
Name: Sex, dtype: int64
```



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

#Handle missing values

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

```
len(df)
```

```
len(df)/2
```

```
df.isnull().sum()>(len(df)/2)
```

```
[23] df.isnull().sum()
```



PassengerId	0
Survived	0
Pclass	0
Sex	0
Age	177
SibSp	0
Parch	0
Fare	0
Cabin	687
Embarked	2
dtype:	int64

```
[24] len(df)
```



891

```
[25] len(df)/2
```

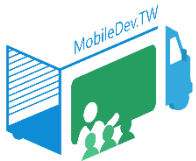


445.5

```
[26] df.isnull().sum()>len(df)/2
```



PassengerId	False
Survived	False
Pclass	False
Sex	False
Age	False
SibSp	False
Parch	False
Fare	False
Cabin	True
Embarked	False
dtype:	bool



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

#Cabin has too many missing values

```
df.drop('Cabin', axis=1, inplace=True)
```

```
df.head()
```

```
df['Age'].isnull().value_counts()
```

#Age is also have some missing values

```
df.groupby('Sex')['Age'].median().plot(kind='bar')
```

#缺失值男生就用男生的中位數(29)、女生就用女生的中位數(27)來填補

```
df['Age'] = df.groupby('Sex')['Age'].apply(lambda x: x.fillna(x.median()))
```

```
[38] df['Age'].isnull().value_counts()
```

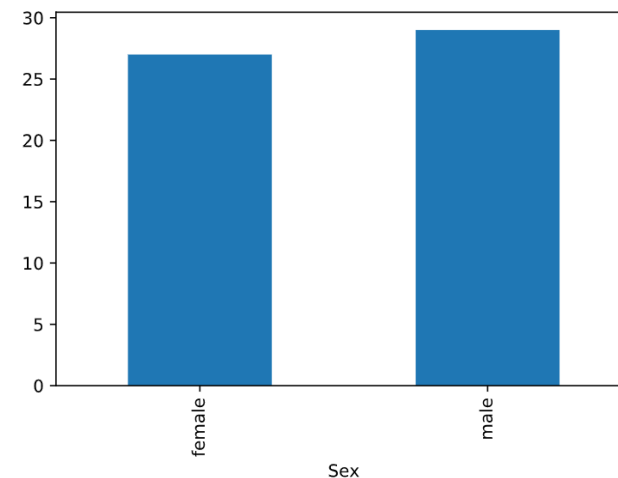


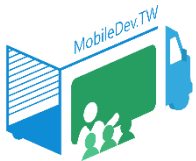
```
False    714
True      177
Name: Age, dtype: int64
```

```
[40] df.groupby('Sex')['Age'].median()
```



```
Sex
female    27.0
male      29.0
Name: Age, dtype: float64
```





練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

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

#發現還有Embarked還有缺2個

```
df['Embarked'].value_counts()
```

#找出第一個次數最多的，發現是S

```
df['Embarked'].value_counts().idxmax()
```

```
df['Embarked'].fillna(df['Embarked'].value_counts().idxmax(),inplace=True)
```

```
df['Embarked'].value_counts()
```

```
[45] df.isnull().sum()
```



```
PassengerId    0
Survived        0
Pclass         0
Sex            0
Age           0
SibSp         0
Parch         0
Fare          0
Embarked       2
dtype: int64
```

```
[46] df['Embarked'].value_counts()
```



```
S      644
C      168
Q       77
Name: Embarked, dtype: int64
```

加起來889筆 (少2筆)

```
[50] df['Embarked'].value_counts()
```



```
S      646
C      168
Q       77
Name: Embarked, dtype: int64
```

填補後

練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

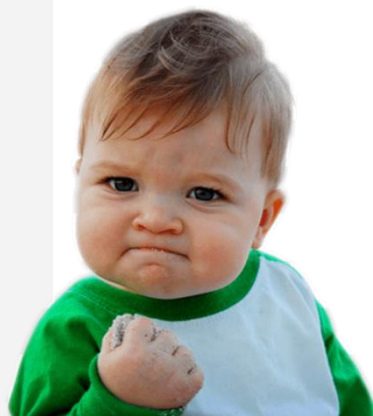
#所有缺失值搞定!

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

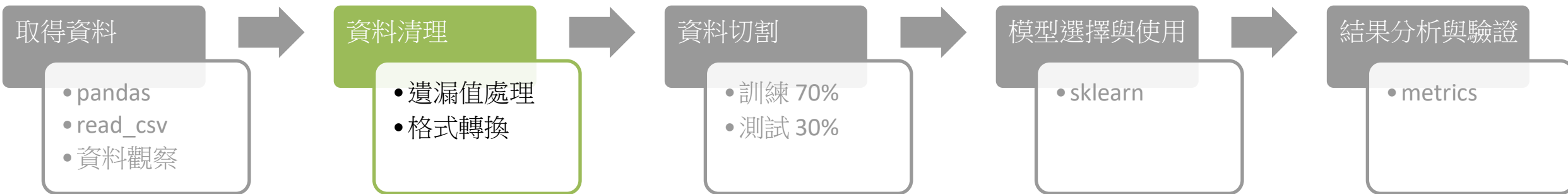
```
[51] df.isnull().sum()
```



PassengerId	0
Survived	0
Pclass	0
Sex	0
Age	0
SibSp	0
Parch	0
Fare	0
Embarked	0
dtype:	int64



練習：鐵達尼號生存預測



#將Sex, Embarked進行轉換

#Sex轉換成是否為男生、是否為女生，Embarked轉換為是否為S、是否為C、是否為Q

```
df = pd.get_dummies(data=df, columns=['Sex', 'Embarked'])
```

```
df.head()
```

#是否為男生與是否為女生只要留一個就好，留下是否為男生

```
df.drop(['Sex_female'], axis=1, inplace=True)
```

```
df.head()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	Sex_male	Embarked_C	Embarked_Q	Embarked_S
0	1	0	3	22.0	1	0	7.2500	1	0	0	1
1	2	1	1	38.0	1	0	71.2833	0	1	0	0
2	3	1	3	26.0	0	0	7.9250	0	0	0	1
3	4	1	1	35.0	1	0	53.1000	0	0	0	1
4	5	0	3	35.0	0	0	8.0500	1	0	0	1



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

模型選擇與使用

- sklearn

結果分析與驗證

- metrics

```
df.corr()
```

```
#Prepare training data
```

```
#把Survived, Pclass丟掉
```

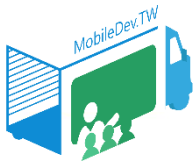
```
X = df.drop(['Survived', 'Pclass'], axis=1)
```

```
y = df['Survived']
```

```
#split to training data & testing data
```

```
from sklearn.model_selection import train_test_split
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=67)
```



練習：鐵達尼號生存預測

取得資料

- pandas
- read_csv
- 資料觀察

資料清理

- 遺漏值處理
- 格式轉換

資料切割

- 訓練 70%
- 測試 30%

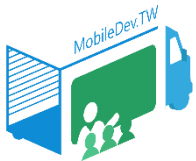
模型選擇與使用

- sklearn

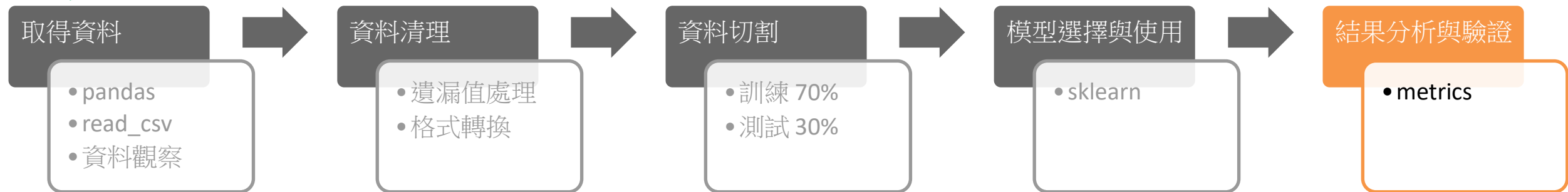
結果分析與驗證

- metrics

```
#using Logistic regression model
from sklearn.linear_model import LogisticRegression
lr = LogisticRegression()
lr.fit(X_train, y_train)
predictions = lr.predict(X_test)
```



練習：鐵達尼號生存預測



#Evaluate

```
from sklearn.metrics import confusion_matrix, accuracy_score, recall_score, precision_score
accuracy_score(y_test, predictions) 0.832089552238806
recall_score(y_test, predictions) 0.7264150943396226
precision_score(y_test, predictions) 0.8279569892473119
```

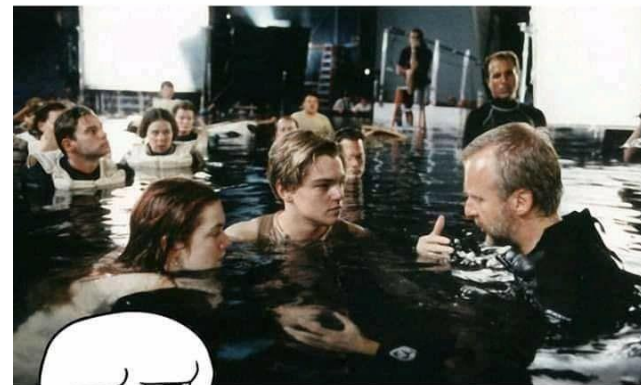
```
pd.DataFrame(confusion_matrix(y_test, predictions), columns=['Predict not Survived',
'Predict Survived'], index=['True not Survived', 'True Survived'])
```

	Predict not Survived	Predict Survived
True not Survived	146	16
True Survived	29	77

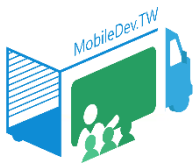
延伸練習 - 鐵達尼號生存密碼

- 請仿造上述練習，探索還有哪些因素跟生存可能有關連性
 - 乘客姓名、船票編號、艙房編號、家屬人數...意想不到的越好
 - 走完五個步驟(觀察->清理->切割->模型->分析)
 - 最後寫上你分析完後的結論
- 評分標準
 - 創意思考 40%
 - 流程完整度 40%
 - 結論分析 20%

Titanic was filmed in pool
and the water was warm



**My whole life
was a lie!**



常見評量方式

- 迴歸
 - mean_squared_error
 - mean_absolute_error
 - explained_variance_score
 - r2_score
- 分類
 - Precision
 - Recall
 - F1 Score
 - Accuracy

常見評量方式

n = 100	預測為No		預測為Yes	
實際上是 No	TN	35	FP	15 (Type I Error)
實際上是 Yes	FN	5 (Type II Error)	TP	45

Precision 準確率 = $\frac{\text{模型預測為Yes且實際上為Yes}}{\text{模型預測為Yes的個數}}$

Recall 召回率 = $\frac{\text{實際上為Yes而模型也預測為Yes}}{\text{實際上為Yes的所有個數}}$

F1 Score = $2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$

Accuracy 精準率 = $\frac{\text{模型預測為Yes且實際上為Yes} + \text{模型預測為No且實際上為No}}{\text{所有預測的個數}}$

使用時間

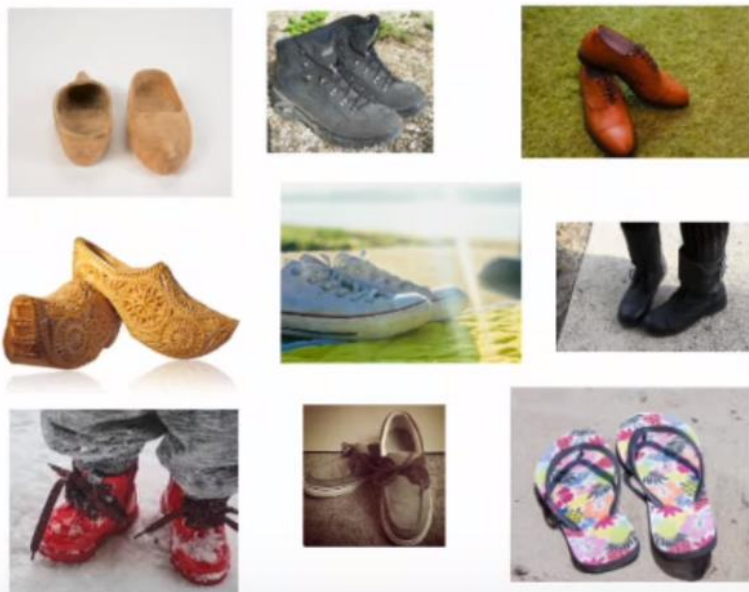
- 機率為Yes或No比例相當時，大多數可用Accuracy
 - 因為當Yes或No明顯比例偏高時，就全部猜那一邊Accuracy會大幅提升
- 怕Type I Error的，要用Precision
 - Type I Error 就是預測為Yes但實際為No
 - 例如門禁系統把陌生人當成自家人
- 怕Type II Error的，要用Recall
 - Type II Error 就是預測為No但實際為Yes
 - 例如廣告投放判斷不是潛在客戶但結果却是潛在客戶
- F1 Score 可以避免Precision & Recall的極端誤差

訓練資料

- 多元化

The Right Data

I want to train a model to detect shoes



Training Data





訓練資料

- 越真實越好，實驗室的資料有時太乾淨反而脫離現實
- 多樣且平均分散，避免資料僅來自部分類別，造成偏差
 - 各種可能的情境
 - 真實世界的影像(不是在攝影棚/錄音室/實驗室內拍攝的)
 - 有背景噪音的聲音資料
 - 各種風格的手寫字
 - 不同種類的資要，但盡可能蒐集數量均衡一些
 - 確認標註的正確性

模型評估

- 80%訓練+20%測試或80%測試+10%驗證+10%測試
- 訓練資料與測試資料都盡量平均分散分配，效果最佳

The model is evaluated against a test set of data



***NOTE: Test data is always separated from the training data**

Precision & Recall

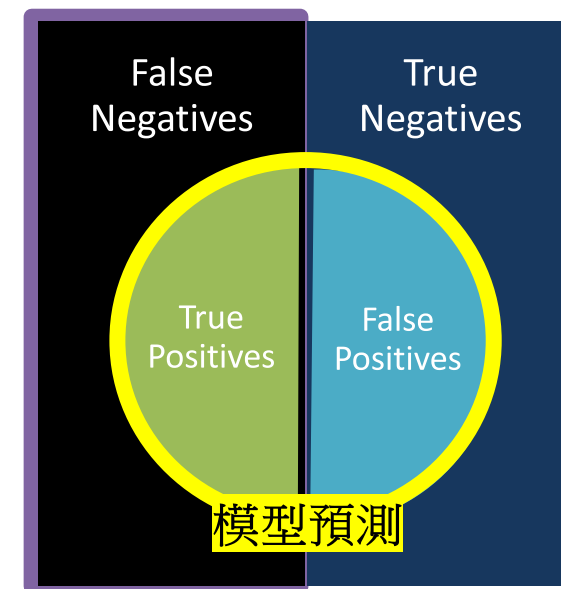
- Precision – 準確率(你的模型判斷是對的中，有多少真的是對的)
- Recall – 召回率(真的是對的的項目中，你的模型找到幾個)
- 準確率是從模型的角度出發、召回率是用真實的狀況來看

$$\text{Precision 準確率} = \frac{\text{模型預測為Yes且實際上為Yes}}{\text{模型預測為Yes的個數}} =$$



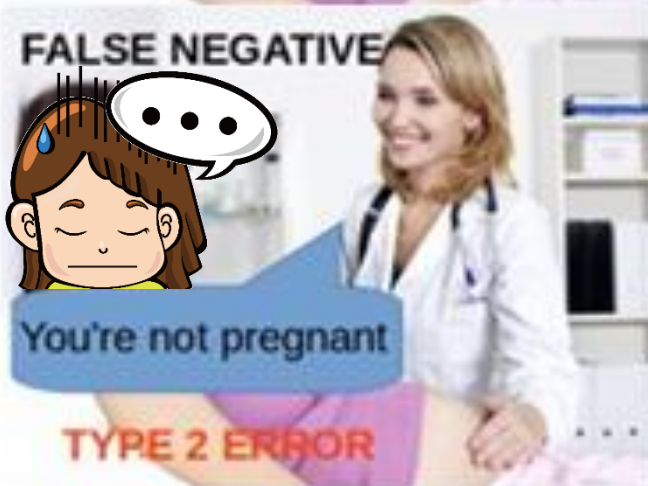



$$\text{Recall 召回率} = \frac{\text{實際上為Yes而模型也預測為Yes}}{\text{實際上為Yes的所有個數}} =$$

Ground Truth Positives



範例：是否懷孕的判斷

		Actual Values	
		1	0
Predicted Values	1	<p>TRUE POSITIVE</p> 	<p>FALSE POSITIVE</p> 
	0	<p>FALSE NEGATIVE</p> 	<p>TRUE NEGATIVE</p> 

Recall & Precision練習

模型預測結果



Cat



Dog



Cat

Precision 準確率 = $\frac{\text{模型預測為Yes且實際上為Yes}}{\text{模型預測為Yes的個數}}$

Precision for Cat = _____

Precision for Dog = _____

Precision for Mouse = _____

Precision for Whole Model = $\frac{\text{每一種類別的精確率加總}}{\text{類別數}}$

= _____

Recall & Precision練習

模型預測結果



Cat



Dog



Cat

Recall 召回率 = $\frac{\text{實際上為Yes而模型也預測為Yes}}{\text{實際上為Yes的所有個數}}$

Recall for Cat = _____

Recall for Dog = _____

Recall for Mouse = _____

Recall for Whole Model = $\frac{\text{每一種類別的召回率加總}}{\text{類別數}}$

= _____

Summary

- 面對問題首先釐清是分類/數值預測/時間序列等類型，再決定方法
- 了解數據特徵、確認遺失值狀況，選擇合適方式填補或直接捨去
- 選擇演算法、選擇評估方式，過程中亦可搭配視覺化方式進行探索

