以DNN預測商業貨品配送是否能準時到達

Use DNN to predict whether commercial goods delivery can arrive on time

大綱

OUTLINE

01 背景介紹
Background

02 資料集描述
Data set

03 資料前處理
Pre-processing

04 模型介紹
Model

05 結論與未來發展
Conclusion

Part 01

背景介紹

Background Introduction

5W1H
問題定義

WHY

預測商品是否能準時送達，針對可能逾期的高風險訂單進行監控

WHO

國際電子商務公司的客戶

WHAT

客戶資料與該商品訂單資訊

WHEN

預計配送日期是否準時送達

WHERE

全球

HOW

資料分析、深度學習、機器學習

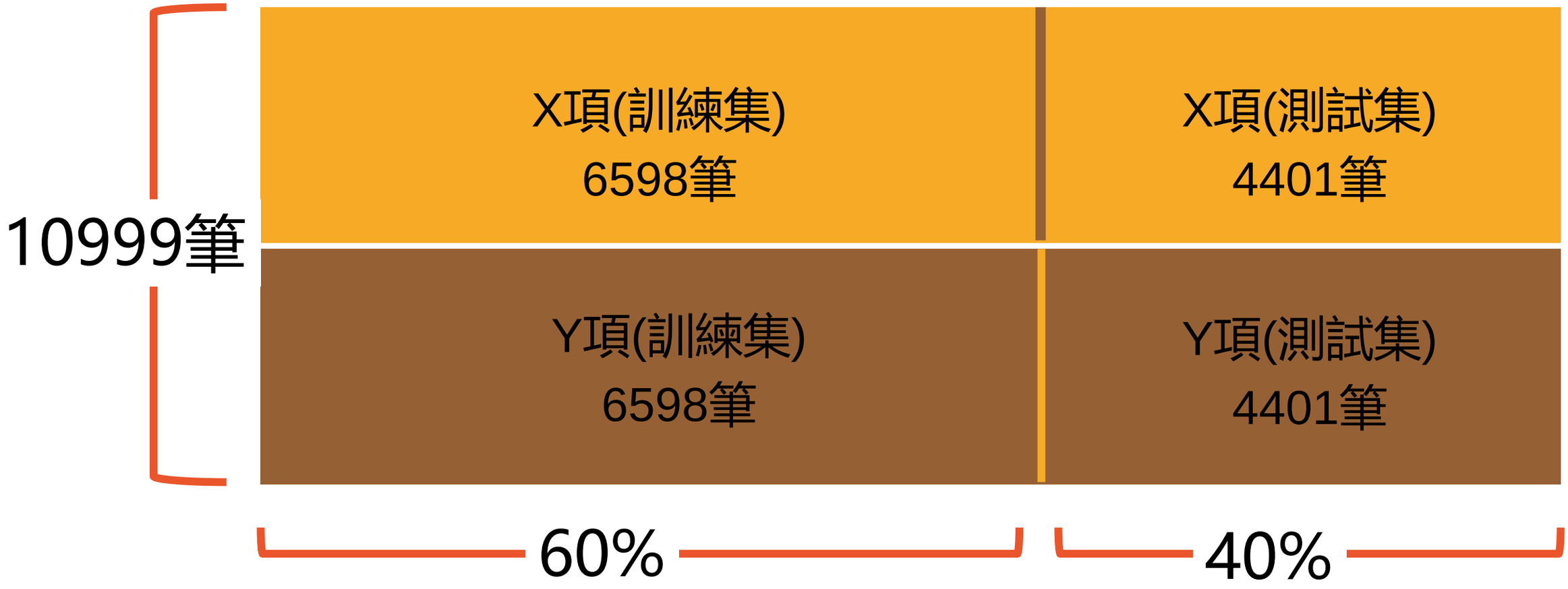
Part 02

資料集描述

CLICK TO ADD CAPTION TEXT

資料集

From Kaggle



特徵值

Features

ID(客戶編號) - 編號1到10999以隨機的方式散佈在兩個資料集中。			
	Feature	Description	Content
X1	Warehouse_block	倉庫位置	分為A、B、C、D、F等位置
X2	Mode_of_Shipment	公司運輸產品的方式	Ship, Flight, Road
X3	Customer_care_calls	查詢貨件查詢來電次數	來電次數
X4	Customer_rating	顧客為公司評價	1(評價低)~5(評價高)
X5	Cost_of_the_Product	商品的成本	以美元計價
X6	Prior_purchases	該顧客先前購買次數	購買次數
X7	Product_importance	公司產品的重要程度	low, medium, high
X8	Gender	客戶性別	M, F
X9	Discount_offered	該特定產品提供的折扣	折價金額(美元)
X10	Weight_in_gms	產品重量	重量(公克)
Y	Reached.on.Time_Y.N	是否準時送達	1(No), 0(Yes)

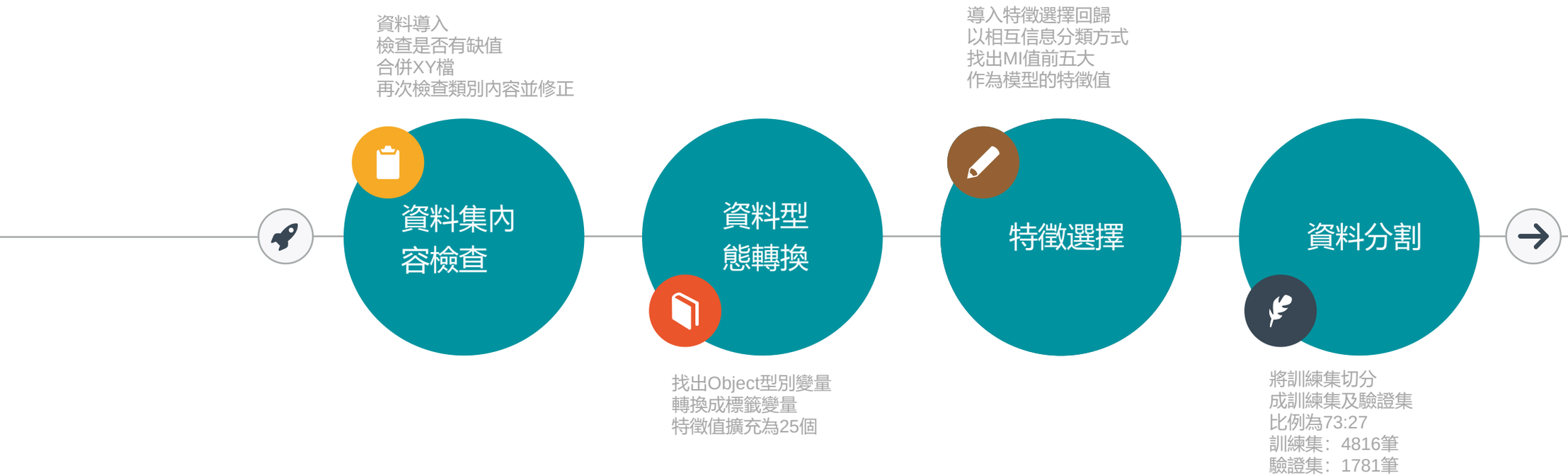
Part 03

資料前處理

Data pre-processing

資料前處理

Data pre-processing



資料前處理-資料集內容檢查

Data pre-processing

- 資料導入

Importing datasets

```
[140] xtr_data=pd.read_csv('/content/X_train.csv')
      xte_data=pd.read_csv('/content/X_test.csv')
      ytr_data=pd.read_csv('/content/y_train.csv')
      yte_data=pd.read_csv('/content/y_test.csv')
```

- 合併XY檔

```
[688] xtr_data=pd.merge(xtr_data,ytr_data,   on='ID',how='outer')
      xte_data=pd.merge(xte_data,yte_data,   on='ID',how='outer')
```

- 檢查類別內容並修正

```
[297] xtr_data['Customer_care_calls']=xtr_data['Customer_care_calls'].replace('$7',7)
      xte_data['Customer_care_calls']=xte_data['Customer_care_calls'].replace('$7',7)
```

- 檢查是否有缺值

Exploring the datasets

```
xtr_data.head()
```

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discount_offered	Weight_in_gms
0	6045	A	Flight	4	3	266	5	high	F	5	1590
1	44	F	Ship	3	1	174	2	low	M	44	1556
2	7940	F	Road	4	1	154	10	high	M	10	5674
3	1596	F	Ship	4	3	158	3	medium	F	27	1207
4	4395	A	Flight	5	3	175	3	low	M	7	4833

```
[142] xte_data.head()
```

	ID	Warehouse_block	Mode_of_Shipment	Customer_care_calls	Customer_rating	Cost_of_the_Product	Prior_purchases	Product_importance	Gender	Discount_offered	Weight_in_gms
0	6811	D	Ship	5	2	259	5	low	F	7	1032
1	4320	F	Ship	3	5	133	3	medium	F	4	5902
2	5732	F	Road	3	4	191	5	medium	F	4	4243
3	7429	D	Ship	4	2	221	3	low	M	10	4126
4	2191	D	Flight	4	5	230	2	low	F	38	2890

```
[144] xtr_data.describe()
```

	ID	Customer_rating	Cost_of_the_Product	Prior_purchases	Discount_offered	Weight_in_gms
count	6598.000000	6598.000000	6598.000000	6598.000000	6598.000000	6598.000000
mean	5476.977266	2.991361	210.393149	3.577751	13.353592	3604.191119
std	3172.946154	1.409624	48.258089	1.511394	16.187267	1635.697627
min	1.000000	1.000000	96.000000	2.000000	1.000000	1001.000000
25%	2731.250000	2.000000	170.000000	3.000000	4.000000	1834.250000
50%	5476.000000	3.000000	214.000000	3.000000	7.000000	4119.500000
75%	8187.750000	4.000000	251.000000	4.000000	10.000000	5027.500000
max	10998.000000	5.000000	310.000000	10.000000	65.000000	7684.000000

資料前處理-資料型態轉換

Data pre-processing

- 找出Object型別變量

```
[157] col_list=[]
      for  column  in  xtr_data:
             if  xtr_data[column].dtype=='object':  #object型別
                    print(column,xtr_data[column].dtype)
                    col_list.append(column)
```

```
Warehouse_block object
Mode_of_Shipment object
Customer_care_calls object
Product_importance object
Gender object
```

- 轉換成標籤變量後特徵值擴充為25個

```
xtr_data.columns
```

```
Index(['Customer_rating', 'Cost_of_the_Product', 'Prior_purchases',
       'Discount_offered', 'Weight_in_gms', 'Reached.on.Time_Y.N',
       'Warehouse_block_A', 'Warehouse_block_B', 'Warehouse_block_C',
       'Warehouse_block_D', 'Warehouse_block_F', 'Mode_of_Shipment_Flight',
       'Mode_of_Shipment_Road', 'Mode_of_Shipment_Ship',
       'Customer_care_calls_7', 'Customer_care_calls_2',
       'Customer_care_calls_3', 'Customer_care_calls_4',
       'Customer_care_calls_5', 'Customer_care_calls_6',
       'Product_importance_high', 'Product_importance_low',
       'Product_importance_medium', 'Gender_F', 'Gender_M'],
      dtype='object')
```

資料前處理-特徵選擇

Data pre-processing

- 導入特徵選擇回歸以相互信息分類方式

```
[700] from  sklearn.feature_selection  import  mutual_info_classif
      mi_score=mutual_info_classif(xtr_data.drop('Reached.on.Time_Y.N',axis=1),xtr_data['Reached.on.Time_Y.N'])
      mi_score=pd.Series(mi_score*100,index=xtr_data.drop('Reached.on.Time_Y.N',axis=1).columns)
      mi_score=mi_score.sort_values(ascending=False)
      mi_score

      Discount_offered             15.745423
      Weight_in_gms                13.422614
      Customer_care_calls_6         0.993544
      Cost_of_the_Product           0.824798
      Customer_care_calls_3         0.741838
      Warehouse_block_B             0.695440
      Prior_purchases               0.605218
      Customer_care_calls_2         0.218103
      Warehouse_block_F             0.168214
      Gender_M                      0.121725
      Mode_of_Shipment_Ship         0.100416
      Gender_F                      0.079125
      Warehouse_block_C             0.065903
      Warehouse_block_D             0.000000
      Warehouse_block_A             0.000000
      Mode_of_Shipment_Road         0.000000
      Mode_of_Shipment_Flight       0.000000
      Customer_care_calls_7         0.000000
      Customer_care_calls_4         0.000000
      Customer_care_calls_5         0.000000
      Product_importance_high       0.000000
      Product_importance_low        0.000000
      Product_importance_medium     0.000000
      Customer_rating               0.000000
      dtype: float64
```

- 找出MI值前五大作為模型的特徵值

```
[701] top_fea=mi_score.index[:5]
      top_fea

      Index(['Discount_offered', 'Weight_in_gms', 'Customer_care_calls_6',
             'Cost_of_the_Product', 'Customer_care_calls_3'],
            dtype='object')
```

資料前處理-資料分割

Data pre-processing

- 將訓練集切分成訓練集及驗證集(73：27 = 4816筆：1781筆)

Splitting the data

```
[217] from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import StandardScaler
      xtr_sc=StandardScaler().fit_transform(xtr_data[top_fea])
      xte_sc=StandardScaler().fit_transform(xte_data[top_fea])
      xtr,xval,ytr,yval=train_test_split(xtr_sc,xtr_data['Reached.on.Time_Y.N'],random_state=108,test_size=0.27)
```

Part 04

模型介紹

Model introduction

模型介紹-Deep Neural Networking(DNN)

Model introduction

投入因子為MI值前五大的特徵值

1. Discount_offered (產品折扣)
2. Weight_in_gms(產品重量)
3. Customer_care_calls_6(客戶來電6次)
4. Cost_of_the_Product (產品成本價格)
5. Customer_care_calls_3(客戶來電3次)

模型結構

六層隱藏層

模型參數

Dropout = 0.35
Activation = relu
Optimizer = adam
Batch_size = 50
Epochs = 100

訓練集效果

Accuracy = 0.6736

測試集效果

Accuracy = 0.6703

模型介紹- Parameter setting of DNN

Model introduction

MSE		模型的隱藏層層數		
		4層	5層	6層
Dropout	0.3	0.1858	0.1868	0.1848
	0.35	0.1847	0.1864	**0.1827**
	0.4	0.1850	0.1863	0.1851

模型介紹- Parameter setting of DNN

Model introduction

```
model=keras.Sequential([
        layers.Dense(164,activation='relu',input_shape=(5,)),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(228,activation='relu'),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(248,activation='relu'),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(268,activation='relu'),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(168,activation='relu'),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(94,activation='relu'),
        layers.BatchNormalization(),
        layers.Dropout(0.35),
        layers.Dense(1,activation='sigmoid')
])

model.compile(optimizer='adam',loss='binary_crossentropy',metrics='mean_squared_error')

call=callbacks.EarlyStopping(patience=12,min_delta=0.0001,restore_best_weights=True)
history=model.fit(xtr,ytr,batch_size=50,epochs=100,validation_data=(xval,yval),callbacks=call)
```

- Dropout = 0.35
- Activation = relu
- Optimizer = adam
- Batch_size = 50
- Epochs = 100

模型介紹- DNN模型測試結果

Model introduction

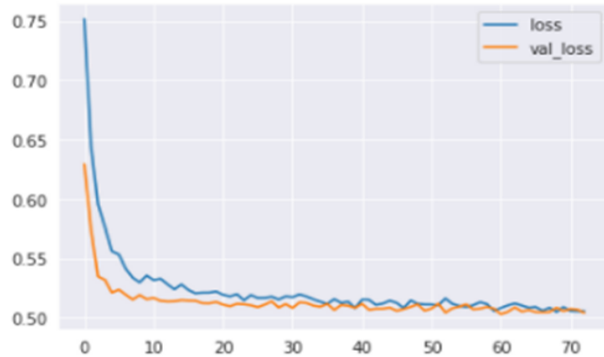
```
his.loc[:,['accuracy','val_accuracy']].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5738e2e290>
```

訓練集
Accuracy = 0.6736
模型訓練成功!

```
his.loc[:,['loss','val_loss']].plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f57421304d0>
```



訓練集
Loss = 0.5013

```
model.evaluate(xte_sc,yte_data)
```

```
138/138 [==============================] - 0s 2ms/step - loss: 0.5087 - accuracy: 0.6703
[0.5086554288864136, 0.6703022122383118]
```

測試集
Accuracy = 0.6703
Loss = 0.5087

模型介紹-RandomForest (RF)

Model introduction

模型結構

max_depth = 5

模型參數

n_estimators = 100
Criterion = entropy
random_state = 0

訓練集效果

Accuracy = 0.6936
AUC = 0.7742

測試集效果

Accuracy = 0.6673
AUC = 0.7326

模型介紹- Parameter setting of RF

Model introduction

AUC		Max_Depth		
		3	4	5
Criterion	Entropy	0.7687	0.7686	**0.7742**
	Gini	0.7694	0.7672	0.7727

模型介紹- Parameter setting of RF

Model introduction

```
rm = RandomForestClassifier(n_estimators=100, criterion='entropy', max_depth=5, random_state=0)
rm.fit(X_train_model, y_train_model)

val_pred = rm.predict(X_val)
val_pred_proba = rm.predict_proba(X_val)[:,1]

print(accuracy_score(Y_val, val_pred))
print('AUC : ', roc_auc_score(Y_val, val_pred_proba))
```

- N_estimators = 100
- Criterion = Entropy
- Max_depth = 5
- Random_state = 0

模型介紹- RF模型測試結果

Model introduction

```
0.6936026936026936
AUC :  0.7742494082797655
```

訓練集
AUC = 0.7742
Accuracy = 0.6936
模型訓練成功!

```
roc_auc_score(true,  pred)
0.7326434463598039

[76] print(accuracy_score(true,  test_val_pred))
0.667348329925017
```

測試集
AUC = 0.7326
Accuracy = 0.6673

Part 05

結論與未來發展

Conclusion and future development

結論與未來發展

Conclusion and future development

分析原始資料內容

原始資料集的收集越豐富越好，在應用的時候將不需要的項目剃除是相對容易，若是有缺少的項目資料需要補充卻是相當困難的。作業主題的選擇上是可以在網路上公開的資料集進行選擇，但實際應用則不一定可以任意挑選資料集，所以蒐集資料的過程就非常重要。

商務應用

業務來往的紀錄文件或是客戶填寫的個人資料就會是未來可以分析的資料來源，因此新客戶的資料填寫表格設計及資料庫後台的管理相對重要。

產線應用

現場可應用資料來源除了架設sensor進行即時的資料收集存取外，作業人員的工作流程所需表單也可盡量改以電子化結合MES系統，對於未來製程的資料分析會相對容易，降低紙本內容轉換成電子檔的人工處理過程。

THANK YOU

感謝聆聽