智慧化企業整合

Project3

以 DNN 預測商業貨品配送是否能準時到達

110034584 林曉吟

目錄

一、 背景介紹

1. 情境描述

隨著線上購物發展與近年 COVID-19 疫情影響，消費者對於日常用品的採購，更傾向以網購取代實體店面的採買，物流業者所須配送的單量大幅提升，導致網購賣家或線上商場在各個購物節的期間來不及出貨到消費者端，因此希望藉由這次作業探討電商業者在商品出貨過程中，不同的因素對於商品是否能準時到達配送地進行預測，再藉由模型的預測結果針對可能逾期的高風險訂單進行監控，降低逾期風險以期提高客戶滿意度。

2. 問題定義

某國際電子商務公司，希望從客戶資料與該商品訂單資訊中找出與「準時配送」相關的因素；本次作業經由原始資料分析後，再以機器學習中的 DNN 演算法、隨機森林演算法進行模型訓練及比較預測結果的準確度。

5W1H	定義
What	客戶資料與該商品訂單資訊
Where	全球
When	預計配送日期是否準時送達
Who	國際電子商務公司的客戶
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
How	資料分析、深度學習、機器學習

表一、5W1H 問題定義

二、 資料集描述

此資料集的來源為一家國際電子商務公司的客戶訂單資料；原始資料分為四個檔案：訓練集 X 項(6598 筆)、訓練集 Y 項(6598 筆)、驗證集 X 項(4401 筆)與驗證集 Y 項(4401 筆)，共有 10999 筆。而其欄位扣除客戶 ID 後共有 10 個特徵值，敘述如下：

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)

表二、欄位特徵敘述表

三、 資料前處理

1. 資料集內容檢查

先將欲分析的資料導入，並確認資料是否有需要修正的內容。

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')
```

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

確認資料是否有缺值。

```
[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

個別合併 X 及 Y 的訓練集與驗證集檔案。

```
[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')
```

確認訓練集的資料筆數共 6598 筆後，檢查類別的資料型態是否需要重新修正類別標示。

- Warehouse_block：倉庫位置分為 A、B、C、D、F 等位置。

```
[145] xtr_data['Warehouse_block'].value_counts()

F    2262
B    1116
A    1090
D    1069
C    1061
Name: Warehouse_block, dtype: int64
```

- Mode_of_Shipment：產品運送方式分為 Ship, Flight and Road。

```
[146] xtr_data['Mode_of_Shipment'].value_counts()

Ship      4512
Flight    1066
Road      1020
Name: Mode_of_Shipment, dtype: int64
```

- Product_importance：產品重要程度分為 low, medium, high。

```
[147] xtr_data['Product_importance'].value_counts()

low       3162
medium    2866
high       570
Name: Product_importance, dtype: int64
```

- Gender：客戶性別分為 F, M。

```
[148] xtr_data['Gender'].value_counts()

F    3311
M    3287
Name: Gender, dtype: int64
```

- Customer_care_calls：客戶查詢貨件查詢來電次數。(將"$7"修改為"7")

```
[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)
```

```
xtr_data['Customer_care_calls'].value_counts()

4    2115
3    1919
5    1403
6     604
2     404
7     153
Name: Customer_care_calls, dtype: int64
```

2. 資料型態轉換

獲取陣列中元素的 Object 型別，接著得出客戶資訊列中的所有型別的唯一值，並將 Object 型別變量轉為標籤變量，例如：Gender 有 F 和 M 兩種，若轉為標籤變量之後會以 Gender_F 和 Gender_M 呈現，讓非數值資訊能以 0 或 1 來進行表示，以便後續相關性分析。

```
[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
```

```
for  i  in  col_list:
       print(xtr_data[i].value_counts())
```

```
F    2262
B    1116
A    1090
D    1069
C    1061
Name: Warehouse_block, dtype: int64
Ship      4512
Flight    1066
Road      1020
Name: Mode_of_Shipment, dtype: int64
4    2115
3    1919
5    1403
6     604
2     404
7     153
Name: Customer_care_calls, dtype: int64
low       3162
medium    2866
high       570
Name: Product_importance, dtype: int64
F    3311
M    3287
Name: Gender, dtype: int64
```

轉為標籤變量之後，從 10 個特徵值擴充為 25 個特徵值。

```
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')
```

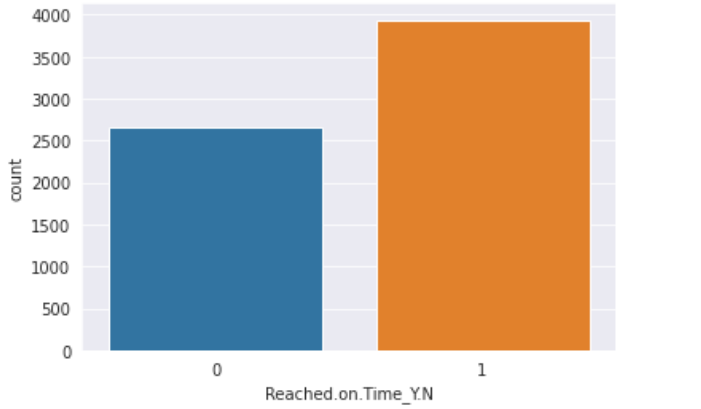
3. 數據視覺化分析

以可視化數據的分佈和趨勢分析數據資料，如下：

(1)商品未準時到達的比例較高。(0 代表準時，1 代表逾時。)

```
[480] sns.countplot(x=xtr_data['Reached.on.Time_Y.N'])
```

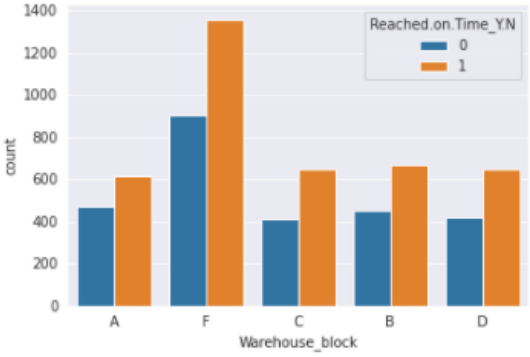
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5755fead10>
```



(2)產品從不同倉庫出貨比例，F 倉佔大宗。

```
[737] sns.countplot(x=xtr_data['Warehouse_block'],hue=xtr_data['Reached.on.Time_Y.N'])
```

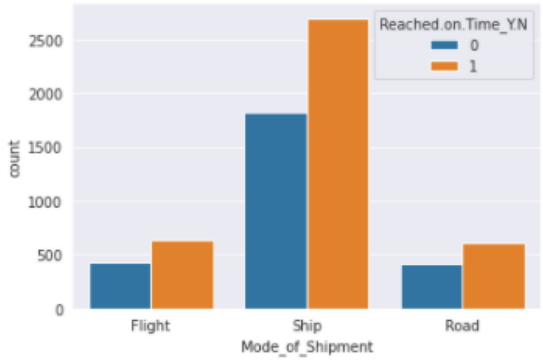
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5748c5da90>
```



(3) 客戶使用海運的比例很高。

```
sns.countplot(x=xtr_data['Mode_of_Shipment'],hue=xtr_data['Reached.on.Time_Y.N'])
```

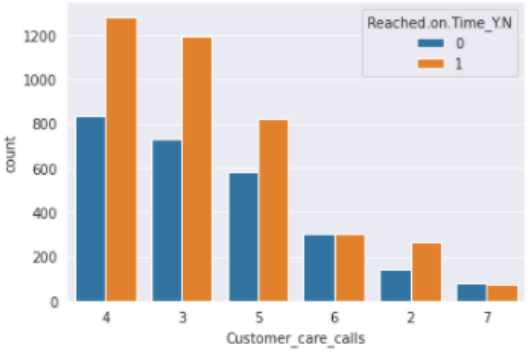
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f575ed56ad0>
```



(4)客戶查詢較多次(6、7 次)的貨品，準時抵達的次數比逾期高。

```
sns.countplot(x=xtr_data['Customer_care_calls'],hue=xtr_data['Reached.on.Time_Y.N'])
```

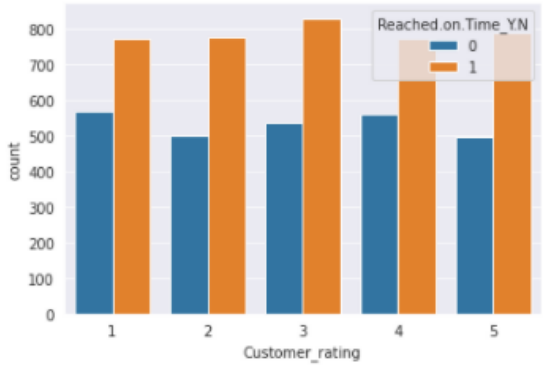
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f573d3513d0>
```



(5) 客戶評價對於運送是否準時相對較無影響。

```
[738] sns.countplot(x=xtr_data['Customer_rating'],hue=xtr_data['Reached.on.Time_Y.N'])
```

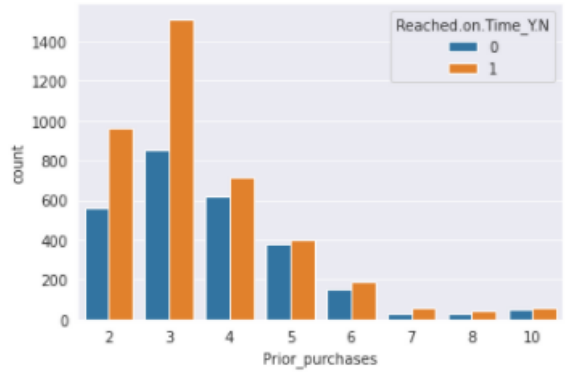
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f574d159d10>
```



(6)訂購次數多次的客戶，他們的貨品準時抵達的次數與逾期的次數接近。

```
[739] sns.countplot(x=xtr_data['Prior_purchases'],hue=xtr_data['Reached.on.Time_Y.N'])
```

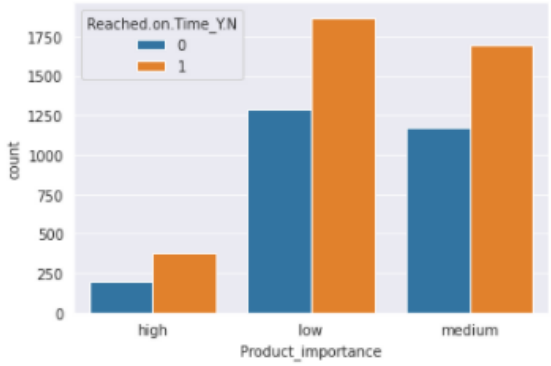
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f575f854450>
```



(7)公司產品的重要程度分級對於運送是否準時相對較無影響。

```
[742]
sns.countplot(x=xtr_data['Product_importance'],hue=xtr_data['Reached.on.Time_Y.N'])
```

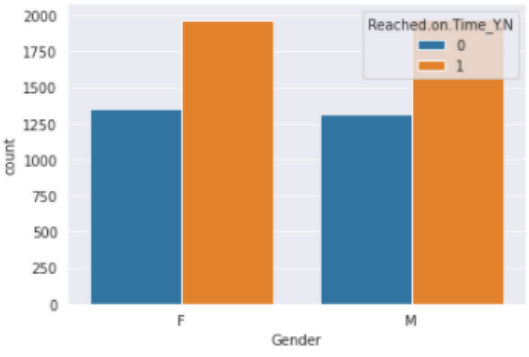
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f575ffb2e90>
```



(8)客戶性別對於運送是否準時相對較無影響。

```
[749] sns.countplot(x=xtr_data['Gender'],hue=xtr_data['Reached.on.Time_Y.N'])
```

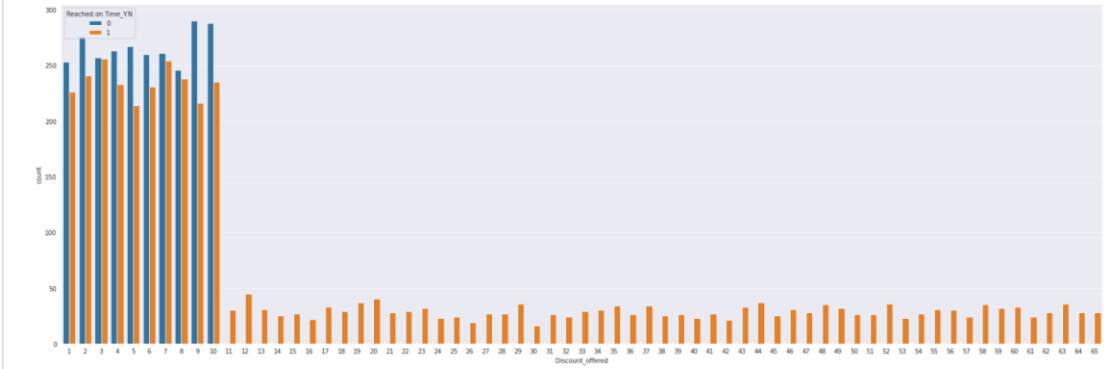
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f573a016c90>
```



(9)折扣少於 10 鎂的商品準時到達次數均高於逾期的次數。

```
plt.figure(figsize=(30,10))
sns.countplot(x=xtr_data['Discount_offered'],hue=xtr_data['Reached.on.Time_Y.N'])
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5742119510>
```



4. 特徵選擇

透過導入特徵選擇回歸中相互信息分類，可以在有多種特徵中找出質量最好的，也就是 MI 值大的特徵，並選擇其特徵作為預測數據。由下圖回歸結果可發現，獲得最大 MI 值得特徵也就是與 Reached.on.Time_Y.N (是否準時送達)相關性較大的特徵，前五大重要特徵值為 Discount_offered (產品折扣)、Weight_in_gms(產品重量)、Customer_care_calls_6(客戶來電 6 次)、Cost_of_the_Product (產品成本價格)以及 Customer_care_calls_3(客戶來電 3 次)。

```
[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
```

選擇前五大的 MI 值作為後續模型所使用的特徵值。

```
[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')
```

5. 資料分割

將訓練集資料切分為訓練以及測試兩個部分，訓練和測試的比例分別為 73：27，也就是訓練集為 4816 筆，測試集為 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)
```

四、 模型介紹

1. 投入因子

在特徵選擇所得出 5 個 MI 值較高的特徵值項目作為投入因子。

```
[522] 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)
```

2. 模型結構-DNN

本次作業使用 DNN(Deep neural networking)為基礎，以模型的層數(4 層、5 層、6 層)、Dropout 的比率(0.3、0.35、0.4)做為變數，再以 mean_squared_error(MSE)為評斷該模型預測準確度的標準。當 MSE 越小代表其所有預測結果與真實數據誤差平方和小，也代表模型預測較準確；而當 MSE 越大代表其預測與真實數據誤差平和大，也代表模型預測較不準確。從表三結果可知當模型為六層且 Dropout 的比率為 0.35 時可得到最小的 MSE，因此本作業決定以圖一的模型結構進行預測。

```
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)
```

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

表三、模型各變數的 MSE

```
Model: "sequential_22"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_139 (Dense)           (None, 164)               984

 batch_normalization_96 (Bat  (None, 164)              656
 chNormalization)

 dropout_117 (Dropout)       (None, 164)               0

 dense_140 (Dense)           (None, 228)               37620

 batch_normalization_97 (Bat  (None, 228)              912
 chNormalization)

 dropout_118 (Dropout)       (None, 228)               0

 dense_141 (Dense)           (None, 248)               56792

 batch_normalization_98 (Bat  (None, 248)              992
 chNormalization)

 dropout_119 (Dropout)       (None, 248)               0

 dense_142 (Dense)           (None, 268)               66732

 batch_normalization_99 (Bat  (None, 268)              1072
 chNormalization)

 dropout_120 (Dropout)       (None, 268)               0

 dense_143 (Dense)           (None, 168)               45192

 batch_normalization_100 (Ba  (None, 168)              672
 tchNormalization)

 dropout_121 (Dropout)       (None, 168)               0

 dense_144 (Dense)           (None, 94)                15886

 batch_normalization_101 (Ba  (None, 94)               376
 tchNormalization)

 dropout_122 (Dropout)       (None, 94)                0

 dense_145 (Dense)           (None, 1)                 95

=================================================================
Total params: 227,981
Trainable params: 225,641
Non-trainable params: 2,340
_________________________________________________________________
None
```

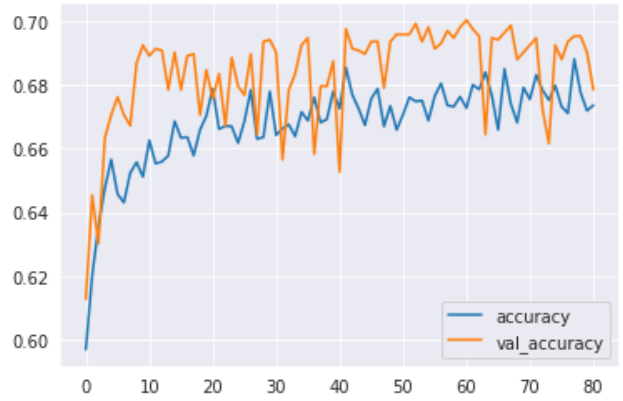
圖一、模型結構

3. DNN 模型測試結果

模型對於訓練集的 accuracy 值逐漸收斂至 0.6736，表示此模型訓練成功(準確度>0.5)。

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

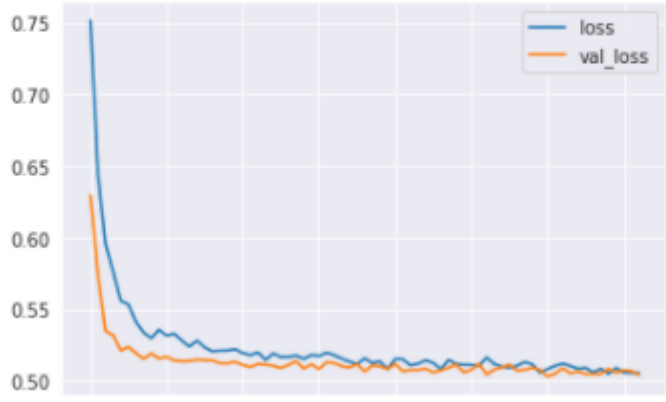
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f5738e2e290>
```



模型對於訓練集的 loss 值可以下降至 0.5013 並逐漸收斂，表示此模型訓練成功。

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

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



模型對於驗證集的 loss 值為 0.5087，accuracy 值為 0.6703，雖然準確度高於 0.5，但可以再嘗試其他模型確認是否可以提高。

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

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

4. 模型結構-RF

因為 DNN 的模型結果不是很滿意，再使用 RF(RandomForest)的模型的 max_depth 數(3、4、5)、criterion (entropy、gini)做為變數，再以 AUC 為評斷該模型預測準確度的標準。當 AUC 值越大代表其模型預測較準確。從表四結果可知當 max_depth 為 5，且 criterion 設定為 entropy 時可得到最佳的 AUC 值。

AUC		max_depth		
		3	4	5
criterion	entropy	0.7687	0.7686	**0.7742**
	gini	0.7694	0.7672	0.7727

表四、模型各變數的 AUC

最終選定使用 AUC 值最高的組合進行訓練：

- n_estimators=100
- criterion=entropy
- max_depth=5
- random_state=0

```
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))
```

訓練模型的 accuracy 值為 0.6936(大於 DNN 訓練集的 accuracy 值：0.6703)，其 AUC 值為：0.7742。

```
0.6936026936026936
AUC : 0.7742494082797655
```

5. RF 模型測試結果

模型對於驗證集的 accuracy 值為 0.6673(小於 DNN 驗證集的 accuracy 值：0.6703)，其 AUC 值為：0.7326，此模型訓練成功(準確度>0.5)，此模型準確度與 DNN 相比，未表現出預期優化的結果。

```
test_pred_proba = rm.predict_proba(x_test_dum)[:,1]
submission = pd.DataFrame({'ID': x_test['ID'], 'Reached.on.Time_Y.N': test_pred_proba})
submission.to_csv('result2.csv', index=False)
```

```
y_test = pd.read_csv('/content/y_test.csv')
result = pd.read_csv('/content/result2.csv')

true = y_test['Reached.on.Time_Y.N']
pred = result['Reached.on.Time_Y.N']

test_val_pred = rm.predict(x_test_dum)
```

```
roc_auc_score(true, pred)
0.7326434463598039
print(accuracy_score(true, test_val_pred))
0.667348329925017
```

五、 結論與未來發展

兩個模型的預測結果都不算非常好，雖然準確率有超過 0.5，但仍未達 0.7；從原始資料去分析可能的原因，我認為此資料集的變數相對少，若以之前修習過的物流管理課程內容來探討可能影響物流配送是否能及時送達的影響因子，在台灣配送目標地與倉儲間距離及轉運站設置點都是變因，但這個資料集的原始資料相對籠統無法判斷倉儲及配送地址間的距離可反映出配送所需的時間長度；再來是配送是否及時的定義，對於海陸空不同的運輸方式間，所預期的配送時間長度也會不同，彼此間對應的時間長度就存在差異(海>空>地)，且該因子也隱含距離的因素，但也可能在距離較長的狀況下因成本考量而採取較長時間的配送方式，因此這些不完整的內容因此這個不算完整的資料集，對於配送結果的預測模型，可提供的有效特徵值也有限。

經由此次作業，發現原始資料集的內容需要收集越豐富越好，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。因此在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。

商務應用對於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；而產線上的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。

這學期的課程內容所學習的知識很廣，雖然還沒辦法做出很厲害的機器學習模型對已有的資料進行預測，但讓我知道如何正確應用這些工具在職場上。

六、參考資料

1. 數據資料來源
   https://www.kaggle.com/kukuroo3/ecommerce-shipping-data-competition-form

2. 模型參考資料
   https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
   http://www.taroballz.com/2019/05/25/ML_RandomForest_Classifier/