智慧化企業整合 Intelligent Integration of Enterprise
消費者價值模型與預測

108034533 李岱諭
國立清華大學工業工程與工程管理學系

摘要

近年來網路購物的興起，帶動了電商產業的蓬勃發展，新型消費型態所產生的電子訂單不僅能夠幫助客戶資料庫的建立，亦能透過每筆訂單來進行消費者分類與行為預測。本研究藉由消費者價值模型之建立，將顧客依照購買習慣與金額的不同進行分群，接著以此資料集為基準，結合機器學習與深度學習之方法建立模型以進行預測，使企業得以從過往的顧客行為來預測未來之消費模式，省去不必要之廣告與行銷成本，同時提升消費者滿意程度。

一、緒論

1.1 研究背景

隨著電子商務的時代來臨，網路購物的市場規模日漸擴大，上網購物儼然已成為現代人的新型消費型態。其所產生的龐大顧客資料隱藏著大量的商機與意義，需要依靠資料分析賦予數字價值，同時，良好的顧客關係管理亦能替企業帶來更多的營收。

1.2 研究動機與目的

該如何從大量的資料中找出對企業具有價值的部份並以此作為預測消費者行為的依據正是電商業者的目標，透過 RFM 模型得以將顧客進行分群，並藉由深度學習的方法，根據過往的顧客購買行為資料進行分析與預測，推測顧客未來可能的行為並將顧客分類，使企業能夠以此為基準對潛在顧客購買行為進行預測，省去不必要的廣告行銷預算，在提升顧客購買意願的同時亦能夠降低企業成本，消費者亦能迅速地找到符合自己需求的商品，達成雙贏的局面，以下將透過 5W1H 進行更進一步的問題定義：

Who?	各類型企業，如：家電、超市、運輸、證券公司
What?	消費者價值模型與行為預測
Why?	降低企業成本，優化 CRM，掌握顧客需求
Where?	企業顧客資料庫
When?	顧客購買前
How?	資料分析、機器學習、深度學習

二、文獻探討

2.1 RFM 模型

RFM 模型是由喬治·卡利南（George Cullinan）於 1961 年所提出，他指出

在顧客資料庫裡，有三項指標能夠提供企業判斷顧客價值，分別為：最近一次消費(Recency)、消費頻率(Frequency)、消費金額(Monetary)，而這三項指標可對應至「R:新客」(近期有消費的人)、「F:常客」(常常來消費的人)、「M:貴客」(消費金額大的人)，也就是對於企業最有價值的客戶。RFM 模型在 CRM 中廣泛地被提及，精準地描繪了顧客輪廓，判斷顧客價值，且適用於各類型企業，使之進行企業流程重建並針對特定客群進行精準行銷與廣告投放。

2.2 MLP

Multilayer Perceptron，多層感知機，又稱人工神經網路（ANN，Artificial Neural Network)，除了最底層的輸入層與最後的輸出層，中間還包含多個隱層。MLP 所有的引數就是各層之間的連線權重(w)以及偏置(b)，並從中找到最佳的數值。

2.3 SVC

Support Vector Machine，支持向量機，是一種機器學習中監督式學習的方法，可以廣泛的應用於統計分類(classification)和回歸分析(regression analysis)。SVC 在資料庫中透過一些特徵分類不同的資料，使距離兩個類別的邊界可以達到最大產生最佳的超平面。

2.4 Random Forest

隨機森林，由多個決策樹組成，每一棵樹之間並沒有關聯，當有一個新的輸入樣本進入時，就讓森林中的每一棵決策樹分別進行判斷，決定此樣本應屬於哪一類；對於行取樣，採用**有放回**的方式，使得 over-fitting 的情況不易產生，此外，由於隨機性的引入，在資料集上的表現相當良好。

三、資料分析

3.1 資料前處理

3.1.1 數據來源

為了建立合適的模型，此次的資料集與數據來源為 Kaggle，為一數據建模和數據分析之競賽平台，提供統計學家和數據挖掘專家資料以產生最好的模型。本研究採用 Kaggle 中之一銷售獨特禮品公司之跨國數據集，其中包含 2010 年 12 月 1 日至 2011 年 12 月 9 日之間在英國註冊的線上零售的所有交易紀錄，共計 541909 筆。

3.1.2 資料欄位

接著進行各欄位資料所代表的資訊及意涵釐清，以使後續資料處理與模型分析得以順利進行。此資料表中共有 6 個欄位分別為：Stock Code、Description、Quantity、Invoice Date、Unit Price、Customer ID、

Country。

3.1.3 數據處理

(1)將資料庫導入，並利用 pandas 的 head()來檢查數據是否正確匯入。

```
#import the database
data = pd.read_csv('data.csv', encoding="ISO-8859-1", dtype={'CustomerID': str,'InvoiceID': str})
data.head()
```

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	12/1/2010 8:26	2.55	17850	United Kingdom
1	536365	71053	WHITE METAL LANTERN	6	12/1/2010 8:26	3.39	17850	United Kingdom
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	12/1/2010 8:26	2.75	17850	United Kingdom
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	12/1/2010 8:26	3.39	17850	United Kingdom
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	12/1/2010 8:26	3.39	17850	United Kingdom

(2)檢視資料集中個欄位的資訊，由此可知，資料表共有 541909 筆數據，而各欄位的缺失值亦可由此看出，除了 Quantity 及 Customer ID 兩個欄位之外，其餘欄位皆無遺漏值，因此，將此兩欄位的缺失值與重複值刪除以進行後續之研究。

```
#check the data information
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 541909 entries, 0 to 541908
Data columns (total 8 columns):
InvoiceNo      541909 non-null object
StockCode      541909 non-null object
Description    540455 non-null object
Quantity       541909 non-null int64
InvoiceDate    541909 non-null object
UnitPrice      541909 non-null float64
CustomerID     406829 non-null object
Country        541909 non-null object
dtypes: float64(1), int64(1), object(6)
memory usage: 33.1+ MB
```

(3)接著，由於資料集中存在特殊商品編碼，本研究亦將其從資料集中刪除。

```
#check the special code in stockcode
list_special_codes = df_cleaned[df_cleaned['StockCode'].str.contains('^[a-zA-Z]+', regex=True)]['StockCode'].unique()
list_special_codes
```

```
array(['POST', 'D', 'C2', 'M', 'BANK CHARGES', 'PADS', 'DOT'],
      dtype=object)
```

(4)將 Country 欄位以字母為順序進行排列後，由字串形式改為數字以利進行數據分析。

```
l = [i for i in range(37)]
dict(zip(list(le.classes_), l))
```

```
{'Australia': 0,
 'Austria': 1,
 'Bahrain': 2,
 'Belgium': 3,
 'Brazil': 4,
 'Canada': 5,
 'Channel Islands': 6,
 'Cyprus': 7,
 'Czech Republic': 8,
 'Denmark': 9,
 'EIRE': 10,
 'European Community': 11,
```

(5)新增 total price 欄位以估計消費者價值。

```
# Total price feature
df_cleaned['TotalPrice'] = df_cleaned['UnitPrice'] * (df_cleaned['Quantity'] - df_cleaned['QuantityCanceled'])
df_cleaned.head(5)
```

3.2 RFM 模型：顧客分群

在資料處理完畢後，接著以此資料集進行 RFM 模型的建立。為了建構出 RFM 模型中的三大要素：Recency、Frequency、Monetary，本研究分別將資料進行以下處理：

(1) Recency 欄位建立

為了找出近期有消費的顧客，先找出資料集中購買時間最近和最遠的日期，並以此為判斷標準，將時間設為此資料集所涵蓋的時間區段最近的日期(2011/12/10)，以此時間點為基準進行顧客最近一次購買的日數推算，並以此項指標進行評分與分類。

```
#RFM model
df_cleaned['InvoiceDate'].min()
```

```
'1/10/2011 10:32'
```

```
df_cleaned['InvoiceDate'].max()
```

```
'9/9/2011 9:52'
```

```
NOW = dt.datetime(2011,12,10)
df_cleaned['InvoiceDate'] = pd.to_datetime(df_cleaned['InvoiceDate'])
```

```
custom_aggregation = {}
custom_aggregation["InvoiceDate"] = lambda x:x.iloc[0]
custom_aggregation["CustomerID"] = lambda x:x.iloc[0]
custom_aggregation["TotalPrice"] = "sum"

rfmTable = df_cleaned.groupby("InvoiceNo").agg(custom_aggregation)
```

```
rfmTable["Recency"] = NOW - rfmTable["InvoiceDate"]
rfmTable["Recency"] = pd.to_timedelta(rfmTable["Recency"]).astype("timedelta64[D]")
```

(2) Frequency & Monetary 欄位建立

Frequency 由 Customer ID 進行判斷，計算方式為加總同一個顧客的總購買次數；而 Monetary 則是同一個顧客的所有購買資料中購買金額之加總，其運算方式如下：

```
#construct the RFM feature
custom_aggregation = {}

custom_aggregation["Recency"] = ["min", "max"]
custom_aggregation["InvoiceDate"] = lambda x: len(x)
custom_aggregation["TotalPrice"] = "sum"

rfmTable_final = rfmTable.groupby("CustomerID").agg(custom_aggregation)
```

```
#show the result of RFM table
rfmTable_final.columns = ["min_recency", "max_recency", "frequency", "monetary_value"]
rfmTable_final.head(5)
```

CustomerID	min_recency	max_recency	frequency	monetary_value
12346	325.0	325.0	1	0.00
12347	2.0	367.0	7	4310.00
12348	75.0	358.0	4	1437.24
12349	18.0	18.0	1	1457.55
12350	310.0	310.0	1	294.40

(3)進行顧客價值評分

取得各消費者的三項評估指標後，為了進行各項指標的評分與顧客分類，因此依各項指標的百分比將其四等分，需要注意的是，由於 Recency 是透過最大日期減去最小日期而得，故 Recency 的值越小代表越近期有購買紀錄，分數也會越高；而 Frequency 和 Monetary 則是數值越大就越高分，將各欄位轉換為評分制度的過程與最終表格如下圖所示：

```
#define the segmentation of each score, each score is divided to categories
def RScore(x,p,d):
    if x <= d[p][0.25]:
        return 1
    elif x <= d[p][0.50]:
        return 2
    elif x <= d[p][0.75]:
        return 3
    else:
        return 4

def FMScore(x,p,d):
    if x <= d[p][0.25]:
        return 4
    elif x <= d[p][0.50]:
        return 3
    elif x <= d[p][0.75]:
        return 2
    else:
        return 1
```

```
segmented_rfm['r_quartile'] = segmented_rfm['min_recency'].apply(RScore, args=('min_recency',quantiles,))
segmented_rfm['f_quartile'] = segmented_rfm['frequency'].apply(FMScore, args=('frequency',quantiles,))
segmented_rfm['m_quartile'] = segmented_rfm['monetary_value'].apply(FMScore, args=('monetary_value',quantiles,))
segmented_rfm.head()
```

CustomerID	min_recency	max_recency	frequency	monetary_value	r_quartile	f_quartile	m_quartile
12346	325.0	325.0	1	0.00	4	4	4
12347	2.0	367.0	7	4310.00	1	1	1
12348	75.0	358.0	4	1437.24	3	2	2
12349	18.0	18.0	1	1457.55	2	4	2
12350	310.0	310.0	1	294.40	4	4	4

經由轉換與計算後而得的 RFM 分數可用來進行顧客分群，區分出對於企業經營最有價值的顧客，例如：最頻繁購買的客群、購買次數不多但單價高的客群等，並以此為依據進行不同類型的消費者策略制定，節省不必要的支出，可直接鎖定目標客群進行投放與行銷，既能有效提升客單價與轉換率，亦可使消費者的需求能精準地被滿足，提升消費者滿意程度。

四、模型訓練

透過前章所建立的 RFM 模型對消費者進行三個面向的分類與評分後，已經可以將過往的顧客資料進行數據分析以使後續之廣告投放與行銷推廣效益最大化，本章將以此為基礎，根據最終資料集內之資訊進行模型的建立及訓練，將資料集以 80:20 的比例分為訓練與測試部分，透過不同的學習方式與參數調整，以期達到最佳預測準確度。

首先，本研究以虛擬分類器做為最初的預測準確度基準，透過 DummyClassifier 函式進行訓練，所得到的準確度僅有 0.280，因此本研究接著使用 MLP、SVC 和 Random Forest 三種模型進行準確度的優化，透過各項參數的調整訓練過程如下：

4.1 MLP (Multilayer Perceptron)

```
#MLP
from sklearn.neural_network import MLPClassifier
mlp = MLPClassifier(hidden_layer_sizes=(64,64,64),
                    activation='logistic',
                    solver='adam',
                    batch_size='auto',
                    learning_rate='constant',
                    learning_rate_init=0.001,
                    max_iter=10,
                    random_state=0)
mlp.fit(X_train, y_train)
print("Train accuracy of MLP: {:.3f}".format(mlp.score(X_train, y_train)))
print("Test accuracy of MLP: {:.3f}".format(mlp.score(X_test, y_test)))
```

首先，使用多層感知器作為訓練的模型，本研究以 sklearn 套件中分類器的各個參數進行修改，由上圖可看見其能調整之參數共有 8 項，在此將以

activation、solver、max iteration 此三項參數進行調整以觀察準確率之改變，並以最大化準確度為訓練目標。

模型初始參數值為 activation function=logistic、solver=SGD、max iteration=auto，訓練及測試資料集的準確率皆為 0.358。在訓練的過程中，先調整了 solver，改為 adam 後，準確率分別提升至 0.575 與 0.555，接著調整了 max iteration 為 100，此時的準確率已可提升至 0.864 與 0.826，最後針對 activation function 進行調整，由 logistic 改為 relu 後，最終準確率可達 0.999 與 0.912。

4.2 Linear SVC

```
#Linear SVC
from sklearn.svm import LinearSVC
lsvc = LinearSVC(
                random_state=None,
                max_iter=10
                )
svc.fit(X_train, y_train)
print("Train accuracy of SVC: {:.3f}".format(svc.score(X_train, y_train)))
print("Test accuracy of SVC: {:.3f}".format(svc.score(X_test, y_test)))
```

本研究使用之第二種機器學習方法為 SVM 中的 linea
rSVC，套用 scikit-learn 導入函式，雖然 Linear SVC 只能使用預設之 activation function，但其具備良好的適配性，且相較於 kNN，僅需較少的樣本數即可用來建立分類模型，因此本研究仍然使用此模型作為方法之一，而由其訓練的結果準確率 0.954 與 0.931 亦證明此模型於本研究中具備良好的預測能力。

4.3 Random Forest

```
#Random Forest
from sklearn.ensemble import RandomForestClassifier
rfc=RandomForestClassifier(max_features=None, criterion='gini', max_depth=None,
                           random_state=0, n_estimators = 100)
param_grid = {
    'n_estimators' : [10, 50, 100],
    'max_features' : ['auto', 'sqrt', 'log2'],
    'max_depth' : [2, 4],
    'criterion' :['gini', 'entropy']
}
rfc=RandomForestClassifier(random_state=0, n_estimators = 100,
                         criterion='entropy', max_depth=3, max_features='auto')
rfc.fit(X_train, y_train)
print("Train accuracy of RFC: {:.3f}".format(rfc.score(X_train, y_train)))
print("Test accuracy of RFC: {:.3f}".format(rfc.score(X_test, y_test)))
```

最後，本研究以隨機森林模型進行訓練，運用其實現簡單、精準度高、抗過擬合能力強的特性來預測。在貢獻大小(criterion)的部分，通常使用基尼指數（Gini index）或者袋外數據（OOB）錯誤率作為評估指標來衡量；在弱學習器

的最大迭代次數(n_estimators)，則需避免過低或過高時產生的欠擬合與過擬合情況發生。

在此，主要針對 n_estimators、criterion、max_depth 三項參數進行調整。最初，以調整 max_depth 觀察其準確率的變化而發現當 max_depth 值越大時，準越率亦越高，而為了避免 overfitting 之情況產生，我們將 max_depth 調整至 3 以進行後續的模型訓練；接著，我們分別以 gini 及 entropy 兩個 creterion 對應到 max_estimators 分別為 10、50、100 時的準越度進行比較與觀察，其結果整理如下表所示：

	10	50	100
gini	0.817	0.840	0.843
entropy	0.826	0.857	0.865

由表中可看出，當 max_estimators 的數值增加時，準確度亦有小幅度的提升，而在 entropy 的表現又略優於 gini，以整體性來看，各數值間差異不大，而最後以準確率為判斷標準，將 criterion 設為 entropy，max_estimators 設為 100，得到最終準確率為 0.865。

4.4 結果分析

在本章中，一共使用了 MLP、SVC 和 Random Forest 等三種模型進行準確度的優化，根據各模型特性進行調整參數的過程時，也發現了準確率隨著參數改變時的影響，最終得到之準確率分別為 0.999、0.954、0.865，雖可能存在 overfitting 的問題，但皆有不錯的預測準確度，可供企業作為未來顧客消費行為的評斷。

五、結論與未來發展

5.1 結論

於本研究中，先透過過去的顧客資料集進行資料欄位的篩選與釐清，刪去異常值與缺失值後，將各指標以資料集為母體進行評分，根據 RFM 模型內的三項指標：Recency、Frequency、Monetary 進行數值與分數間的轉換後得到消費者價值模型，並以此為最終資料集，進行後續之模型建立與訓練的資料來源。本次選用的機器學習方法共有三項，在發現運用 dummy classifier 的預測準確率低落後，首先使用多層感知器進行訓練，而訓練後的準確率亦有大幅地提升。接著，分別使用 SVC 與隨機森林模型進行預測，所得之準確率皆大於 0.85，由結果可得各模型在訓練後的表現皆相當良好，得以適切地進行消費者價值預測，為後續分析提供良好的決策方向。

5.2 未來展望

本次模型僅就本資料集內的消費者價值進行分類與預測，在資料筆數與時間限制下，未能涉及多店間的消費者資料數據庫，亦尚未考量全面性的行銷手

法與廣告類型。若未來得以進一步地深入研究，無論是取得連鎖店各分店間的顧客資料進行交互分析，或是針對同類型商店進行消費者的購物行為預測，都將能使此模型的建構更為全面與完善，同時，在資料筆數增加以及資料來源更多元的情況下，也能使預測準確率更為精準亦更具參考價值，提供企業在思考廣告投放方向與行銷手法時，得以省去不必要之成本與時間浪費，成功提升轉換率，為企業帶來更大的收益。

六、參考資料

- https://wiki.mbalib.com/zh-tw/RFM%E6%A8%A1%E5%9E%8B
- https://www.itread01.com/content/1542288247.html
- https://yourgene.pixnet.net/blog/post/115169530-%E5%88%86%E9%A1%9E%E5%B7%A5%E5%85%B7(1)---%E6%94%AF%E6%8C%81%E5%90%91%E9%87%8F%E6%A9%9F(support-vector-machine)
- https://www.itread01.com/content/1549579879.html