IIE

Final Project

# 消費者價值模型與預測

108034533 李岱諭

IIE
Final Project

# CONTENTS


1 Introduction

2 Data Pre-processing

3 Model Structure

4 Conclusion

# PART 01

## INTRODUCTION

# 01 - SCENARIO

消費型態改變
網購市場擴大

網路購物興起
電商時代來臨

大量交易紀錄
與消費者資料

# 01 – 5W1H

Why

降低企業成本
優化CRM
掌握顧客需求

What

消費者價值模型
與行為預測

When

顧客購買前

Where

企業顧客資料庫
與電商平台

Who

各類型企業
如：家電、超市、
運輸、證券公司

How

資料分析
機器學習
深度學習

# PART 02

## DATA PREPROCESSING

# 02 – Data Source

-資料來源：Kaggle
-期間： 2010年12月1日至2011年12月9日
-企業：跨國禮品公司
-資料筆數：共計541909筆

```
#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 |

# 02 – Data Preprocessing

```
#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
```

檢測是否存在遺漏值

```
#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)
```

刪去存在特殊編碼之商品購買紀錄

# 02 – Data Preprocessing

## 轉換country欄位數值

```python
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,
```

## 計算total price欄位

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

| | InvoiceNo | StockCode | Description | Quantity | InvoiceDate | UnitPrice | CustomerID | Country | QuantityCanceled | TotalPrice |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 536365 | 85123A | WHITE HANGING HEART T-LIGHT HOLDER | 6 | 12/1/2010 8:26 | 2.55 | 17850 | 35 | 0 | 15.30 |
| 1 | 536365 | 71053 | WHITE METAL LANTERN | 6 | 12/1/2010 8:26 | 3.39 | 17850 | 35 | 0 | 20.34 |
| 2 | 536365 | 84406B | CREAM CUPID HEARTS COAT HANGER | 8 | 12/1/2010 8:26 | 2.75 | 17850 | 35 | 0 | 22.00 |
| 3 | 536365 | 84029G | KNITTED UNION FLAG HOT WATER BOTTLE | 6 | 12/1/2010 8:26 | 3.39 | 17850 | 35 | 0 | 20.34 |
| 4 | 536365 | 84029E | RED WOOLLY HOTTIE WHITE HEART. | 6 | 12/1/2010 8:26 | 3.39 | 17850 | 35 | 0 | 20.34 |

# 02 – RFM Model

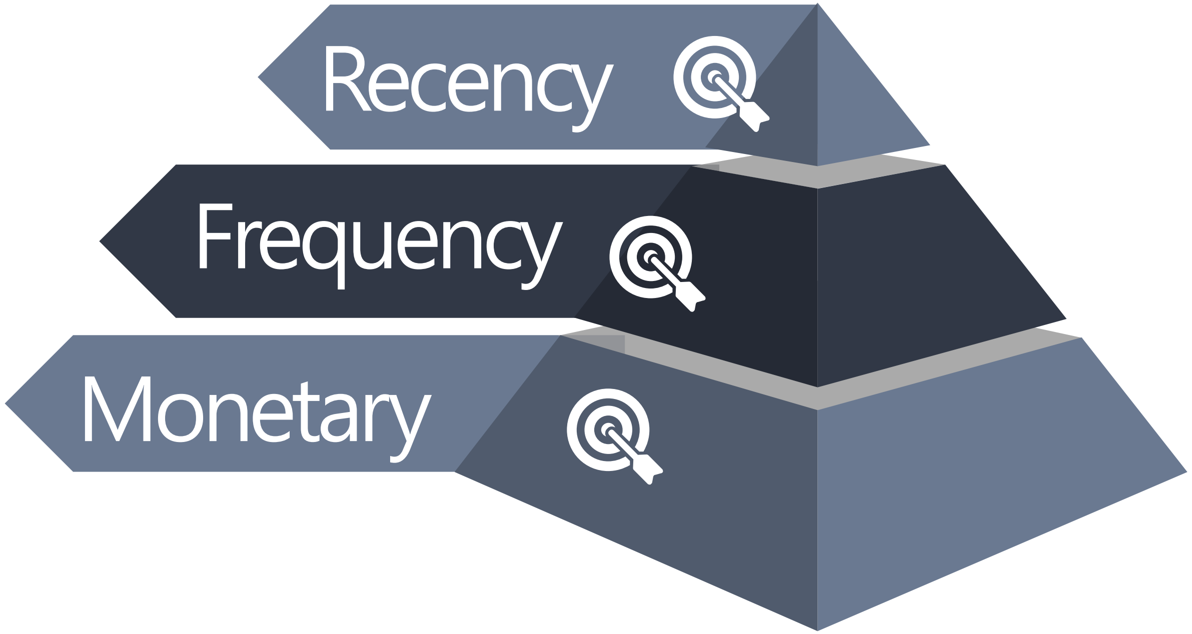
判斷顧客價值

「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）

George Cullinan,1961



# 02 – RFM Model

## 計算R欄位數值

```
#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]")
```

# 02 – RFM Model

## 計算F、M欄位數值

```
#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 |

# 02 – RFM Model

將RFM欄位轉換為分數
1為最佳值

```
#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
```

# 02 – RFM Model

```
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 |

以此分數進行消費者分類，並根據不同類型的客群進行精準行銷

# PART 03

## MODEL STRUCTURE

# 03 – MLP

```
#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)))
```

| PARAMETERS | activation=logistic<br>solver=SGD<br>max iteration=auto | solver=adam | max iteration=100 | activation=relu |
|---|---|---|---|---|
| ACCURACY | 0.358 | 0.555 | 0.826 | 0.919 |

調整參數：activation、solver、max iteration
準確率：0.358 -> 0.555 -> 0.826 -> 0.919

# 03 – 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)))
```

具備良好的適配性
相較於kNN，僅需較少的樣本數即可用來建立分類模型
準確率：0.931

# 03 – 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)))
```

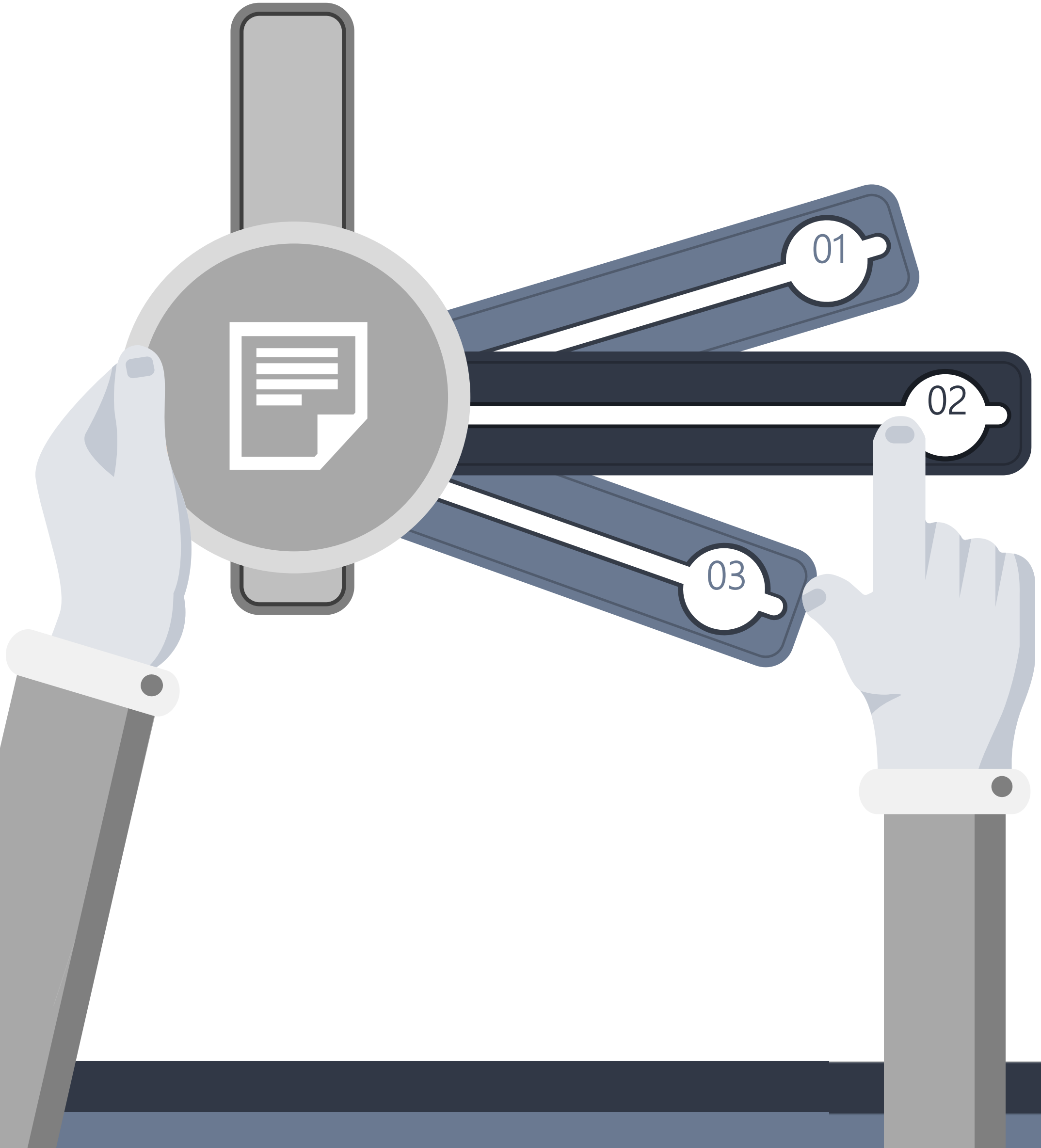
| | 10 | 50 | 100 |
|---|---|---|---|
| gini | 0.817 | 0.840 | 0.843 |
| entropy | 0.826 | 0.857 | 0.865 |

調整參數：criterion、max_estimators
準確率：0.865

# PART 04

## CONCLUSION

# 04 – Conclusion




## 資料前處理
01

刪除重複/缺失值
轉換數值
刪去不合理數值

## RFM模型
02

資料分析
欄位建立
顧客分群

## 預測模型
03

MLP
SVC
Random Forest

# 04 – Future Work

多店交叉預測

連鎖店分析

競品購物
行為預測

IIE

Final Project

# Thanks for listening