智慧化企業整合

預測 KKBOX 訂閱用戶流失率

Group2

組員:

黃怡菁 109034531

李旖庭 109034565

黃荻雅 109034543

黃浩銓 109034523

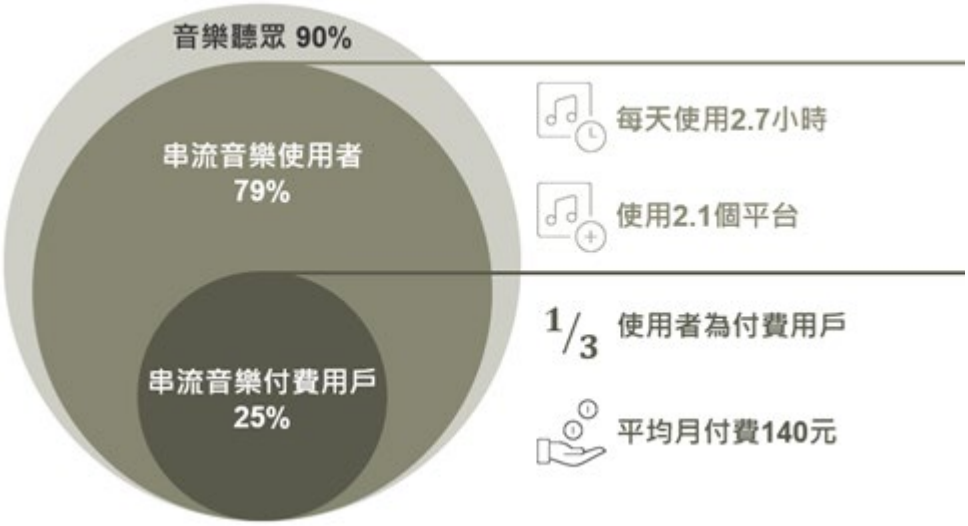
指導老師: 邱銘傳教授

目錄

一、 前言

1. 背景介紹

根據調查，線上串流音樂平台的使用率有 72.5%，觀察不同年齡的使用率後，發現 20 世代使用線上串流音樂平台比率明顯較其他年齡層更高，使用率高於整體 10%，超過八成有使用，如下圖所示。



資料來源: 台灣串流音樂市場概況分析（上）

KKBOX 作為台灣串流音樂的領導品牌，無論在知名度（66%）、目前使用普及（25%）或付費普及率（16%）都最高，換算起來 64%的用戶皆為付費用戶，居各音樂串流平台之冠。目前 KKBOX 付費用戶逾 200 萬，是主要收入來源，所以如何準確預估顧客持續付費使用率成為 KKBOX 的績效指標之一，也顯得格外重要。

2. 5W1H

KKBOX 亞太區董事總經理施盈良回憶當初首個合法付費聆聽的串流平台，即是有鑑於當時網路下載盛行、盜版猖獗，創作者無法收益，造成創作者的苦境，他說：「 KKBOX 首開先河與各家唱片公司談版權、創立音樂訂閱制的服務，把收入合理的分給唱片公司，用戶只要繳月費就可以聽到合法且多元的內容。」，目前 KKBOX 的服務範圍涵蓋台、港、澳、星、馬、日等地，會員已超過 1000 萬，而這當中付費會員已超過 200 萬人，並呈現持續成長的態勢。透過 5W1H 分析，了解的現況使 KKBOX 進一步了解用戶需求，在保持用戶活躍度上進一步採取營銷行動：

- Who ： 各大線上串流平台(Spotify、Netflix、KKBOX)

- What ： 各大線上串流平台訂閱用戶預測
- Why ： 訂閱用戶流失率難以預測，但卻至關重要
- When ： 平台欲評估顧客訂閱率時
- Where ： 平台行銷部門
- How ： 資料分析、深度學習、機器學習

二、 模型架構

1. 資料前處理

(1) 資料來源

資料是由 Kaggle 公開數據集中取得。Kaggle 是一個數據建模和數據分析競賽平台。企業和研究者可在此發布數據，提供統計學家和數據挖掘專家以此資料為基礎，進行競賽以產生最好的模型。此次使用的是 Kaggle 上的「KKBox's Churn Prediction Challenge」。由 KKBOX 提供用戶使用情況以此資料

來預測此用戶在到期日後 30 天後是否會繼續訂閱，如果沒有，則判定為流失客戶，總共有 886500 筆資料(用戶)。

(2) 資料欄位

接著我們釐清各欄位資料所代表的資訊及意涵，以使後續資料處理 與模型分析得以順利進行，本次資料欄位共有下列 9 項：

A. msno ： 使用者 id

B. is_churn ： 定義為這個用戶是否會在會員到期日後 30 天內繼續訂閱，以此來定義為客戶是否流失。如果 is_churn = 1，則表示客戶流失，is_churn = 0，則表示客戶沒有流失。

C. date ： 用戶使用 KKBOX 聽音樂的日期

D. num_25 ： 用來計算每首歌只播放完整時間的前 25%的數量

E. um_50： 用來計算每首歌只播放完整時間的 25%~50%的數量

F. num_75: 用來計算每首歌只播放完整時間的 50%~75%的數量

G. num_985 ： 用來計算每首歌只播放完整時間的 70%~98.5%的數量

H. num_100 ： 用來計算每首歌完整播放的數量

I. num_unq ： 用來計算總共有多少不同的歌曲是被用戶收聽的

J. Total_secs: 總播放時間

K. city: 註冊城市

L. bd: 年紀

M. gender: 性別

N. registered_via: 註冊方式

O. registration_init_time: 註冊時間

P. expiration_date: 到期時間

A. msno	使用者 id
B. is_churn	定義為這個用戶是否會在會員到期日後 30 天內繼續訂閱，以此來定義為客戶是否流失。如果 is_churn = 1，則表示客戶流失，is_churn = 0，則表示客戶沒有流失。
C. date	用戶使用 KKBOX 聽音樂的日期
D. num_25	用來計算每首歌只播放完整時間的前 25%的數量

E. um_50	用來計算每首歌只播放完整時間的 25%~50%的數量
F. num_75	來計算每首歌只播放完整時間的 50%~75%的數量
G. num_985	用來計算每首歌只播放完整時間的 70%~98.5%的數量
H. num_100	用來計算每首歌完整播放的數量
I. num_unq	用來計算總共有多少不同的歌曲是被用戶收聽的
J. Total_secs	總播放時間
K. city	城市
L. bd	年紀
M. gender	性別
N. registered_via	註冊方式
O. registration_init_time	註冊時間
P. expiration_date	到期時間

(3) 資料整合

```
%bq tables list
```

- iie-project2.kk_data.members_v3
- iie-project2.kk_data.sample_submission_v2
- iie-project2.kk_data.sample_submission_zero
- iie-project2.kk_data.train
- iie-project2.kk_data.train_v2
- iie-project2.kk_data.transactions
- iie-project2.kk_data.transactions_v2
- iie-project2.kk_data.user_logs
- iie-project2.kk_data.user_logs_v2

原始資料分別存在不同資料表裡，首先我們先將擁有相同資料欄位的資料表合併在一起。(因為一開始 kaggle 只提供一部份資料，_V2 是後續更新的資料，因此我們將其合併成一個以利後續分析)。

```
%bq  tables  list
```

- iie-project2.kk_data.members_v3
- iie-project2.kk_data.sample_submission_zero
- iie-project2.kk_data.train
- iie-project2.kk_data.transactions
- iie-project2.kk_data.user_label_201703
- iie-project2.kk_data.user_logs

2. 模型架構

(1) 數據分析

首先先從 User_Label 看，總共有 886500 筆資料，且有兩欄資料，一欄為 mson：表示用戶 id(此 id 為 KKBOX 隨機產生的一組數值，且互相獨立沒有重複)；另一欄為 is_churn：表示用戶是否流失，以 True 和 False 表示。

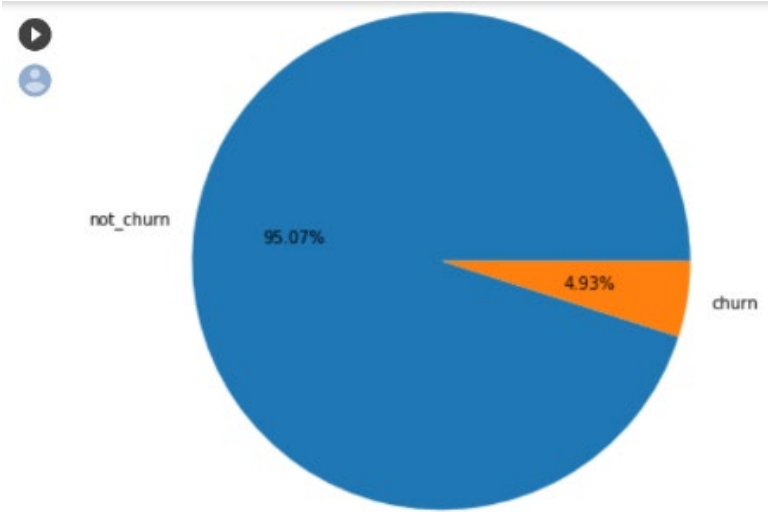
```
[ ]  churn_df.shape
```

```
(886500, 2)
```

```
churn_df.head()
```

	msno	is_churn
0	++4RuqBw0Ss6bQU4oMxaRlbBPoWzoEilZaxPM04Y4+U=	False
1	+/HS8LzrRGXolKbxRzDLqrmwuXqPOYixBlPXkyNcKNI=	False
2	+/g9O3USecrC8npzaFHXW/2XJ7fB80SineiUoCg7M6o=	False
3	+/namlXq+u3izRjHCFJV4MgqcXcLidZYszVsROOq/y4=	False
4	+0/X9tkmyHyet9X80G6GTrDFHnJqvai8d1ZPhayT0os=	False

並且進一步查看其用戶流失分布。



可以發現流失的用戶很少，這樣使得我們的資料集不平衡（沒有流失的用戶很明顯大於流失的用戶），導致我們對資料集訓練時需要調整類別權重，或是過濾掉一些資料。

(2) 接著再看 User_Logs 資料表，總共有 410502905 筆資料。

(rows: 410502905, iie-project2.kk_data.user_logs)

再進一步檢查是否空值，

	not_null_date	not_null_25	not_null_50	not_null_75	not_null_985	not_null_100	not_null_nuq	not_null_total_secs
0	410502905	410502905	410502905	410502905	410502905	410502905	410502905	410502905

發現沒有空值，代表所有列都是有數值不為 0。

因為此資料表(User_Logs)裡的有效資料數遠大於用戶數 886500，又發現 mson 用戶名稱是有重複出現的，代表這張資料表紀錄的是當天用戶聽歌情形。

	msno	date	num_25	num_50	num_75	num_985	num_100	num_unq	total_secs
1	8n5yuVdvBqR4aHr1hUx1FXoluxpZluW+kR/d0ounuYA=	20,160,512	24	2	1	1	53	61	15,019.365
2	d2GtIMHD45Ri5/Ksa86X4FYWCZMM3fZ4WRDKjbhp15A=	20,160,205	81	2	4	3	34	74	9,600.676
3	d2GtIMHD45Ri5/Ksa86X4FYWCZMM3fZ4WRDKjbhp15A=	20,161,124	37	4	4	2	48	83	12,778.431
4	d2GtIMHD45Ri5/Ksa86X4FYWCZMM3fZ4WRDKjbhp15A=	20,170,114	113	22	8	3	40	163	14,347.198
5	kb3qHtlz+K4Ume8TF4FQi9xwrTZqzFFvBDZsdYvyQ0A=	20,161,217	32	1	0	1	30	58	7,391.719
6	B/eZk3P+A98+vport4EL6KBRhYiioS+F1uVJSGmAUGw=	20,150,720	22	2	0	2	38	52	10,062.741
7	ysLUp9Ebqx3RrCNmZAOSmyW7kDZQafvyg7+Ge6lbG3Y=	20,150,717	38	6	5	4	66	87	20,265.215
8	ysLUp9Ebqx3RrCNmZAOSmyW7kDZQafvyg7+Ge6lbG3Y=	20,160,318	36	4	2	4	97	61	24,174.545
9	65MC0qTNLb/tG6fPv0IN7AzLqma4IkDHe15EB8TedA8=	20,150,411	42	24	13	6	85	131	26,109.762
10	aDyAkp8ZPYJUAVISQZ9oe1/2Ub1iDbq1Z9B6lBaTGRk=	20,150,107	23	12	3	7	60	90	19,544.564

接著再去分析每一欄的資料分布。

	feature	min	Q1	Q2	Q4	max
1	num_25	0.0	0.0	1.0	2.0	1710.0
0	num_50	0.0	0.0	2.0	7.0	18798.0
2	num_75	0.0	0.0	0.0	1.0	1690.0
3	num_985	0.0	0.0	0.0	1.0	2747.0
4	num_100	0.0	6.0	17.0	38.0	42004.0
5	num_unq	1.0	8.0	19.0	40.0	4925.0
6	total_sec	-9223372036854776.0	1894.0	4636.0	10228.0	9223372036854776.0

- 首先，我們從 total_secs 的結果可以看到有異常值的出現，在最大值和最小值的部分出現異常值。如果後續我們需要使用到這個特徵要進行資料清理，以免受到極端值影響。

- 接著我們從 25%, 50%, 75%的平均數來看，分別為 1, 2, 0，比例不高，我們可以推測其卡歌率不高。我們推測其原因是因為某種程度上有可能是使用者們滿意自己所聽的音樂。

- 此資料集根據官方描述是以一天為一筆，我們可以粗估使用者每天大約會聽 20 首歌。(因為平均一天聽 4636s，一首歌 4 min(240 s)，換算下來平均一天聽 4636 / 240 ≒ 20 首)

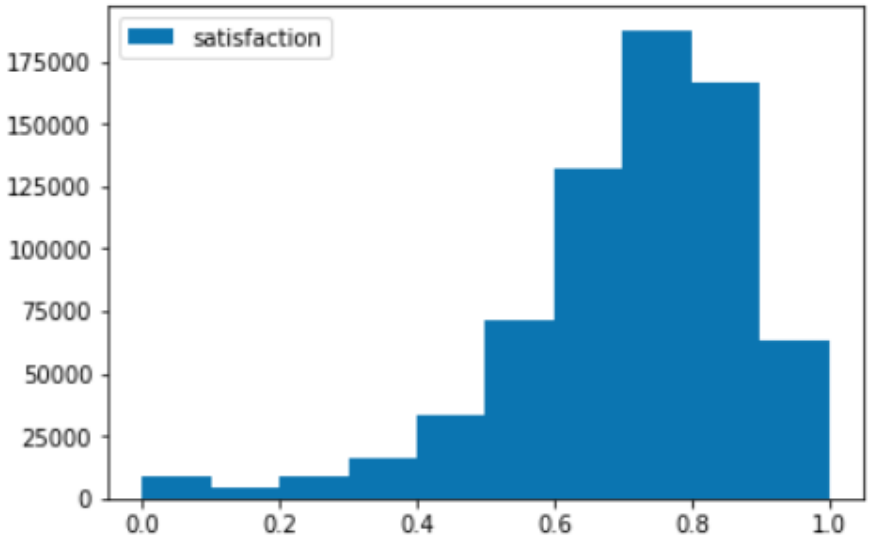
- 這裡的最大值都是值得懷疑的，官方文件寫說 user_logs 為每日紀錄，我們觀察 num_100 這一列，一首歌 4 分鐘也就是 240 秒，平均來說一天(86400)最多只有可能聽到 360 首完整撥放的歌曲。因此在後續使用到這項資料時需要進行資料處理或是有一些假設，來避免異常值的影響。

(3) 選擇特徵

- 因此綜合上述提到的分析，針對每一個使用者，計算其過去六個月以來聽歌 (98.5%+100%)的歌曲數/總聽歌曲數，並假設異常值比例不高(因為後來發現資料表中只有少數幾筆的資料是異常的絕大部分的資料是正常的，且如果我們為了這幾筆資料而去做資料處理導致時間，運算成本的增加，這樣子在實務上並不合乎成本)，所以我將其異常值當作 noise，並將特徵命名為使用者潛在滿意度(user_latent_satisfaction)。

- 針對每一個使用者，計算其過去六個月以來總共聽歌的天數，一條紀錄計算為一條，將特徵命名為聽歌天數(day_listen)

EDA(Exploratory Data Analysis)



- 我們以 is_churn==0 的資料，畫出 user_latent_satisfaction 的分布。從上面這張圖可以看到，滿意度較高的占大多數。

- 觀察訓練集和測試集的 Missing value

```
train_df.isnull().sum()

msno                0
is_churn            0
day_listen     153632
satisfaction   153632
dtype: int64
```

```
[ ] test_df.isnull().sum()

msno                0
is_churn            0
day_listen          0
satisfaction    599274
dtype: int64
```

- 由於我們發現如果只抽取一個月，缺失值比例會比較高，依序兩個月，三個月，缺失值比例會遞減，這對於我們機器學習模型來說，抽取越多個月表示資料將會越完整，但是，太久遠的資料對於目標函數的參考價值也會降低，因此這裡有兩個之間作取捨，

使用驗證集來測試抽取時間的參數應該幾個月比較好，最終測試得到的結果是抽取 6 個月的數據。

至於缺失的資料，我們將缺失值以-1 填入，做為一個特徵類別。(將其看成一個新的特徵)。

- 我們可以從右邊這張圖中看出隨著聽歌天數(day_listen)越多，則流失的機率越低(圖形顏色為粉色的)

- 我們可以從右邊這張圖中，我們將 day_listen 分為 5 個 level，然後分別去計算其比例，其數量佔比最少都有 17%以上，這使得流失率的準確度是可以相信的。

三、 模型比較

模型	優點	缺點

Random forest	1. 訓練可以並行化，對於大規模樣本的訓練具有速度的優勢 2. 由於進行隨機選擇劃分特徵列表，這樣在樣本維度較高的時候，仍然具有比較高的訓練效能 3. 由於存在隨機抽樣，訓練出來的模型方差小，泛化能力強 4. 對於部分特徵的缺失不敏感	1. 每個節點要選擇特徵數量和決策樹的數量，所以更難裝配 2. 在某些噪音比較大的特徵上容易陷入過擬合 3. 取值比較多的劃分特徵對決策會產生更大的影響，從而可能影響模型的效果
XG Boost	由於通過優化目標函數導出了增強樹，基本上可以用來解決幾乎所有可以寫出漸變的目標函數	1. 如果數據有noise，對過度擬合更敏感 2. 由於樹木是按順序建造的，因此培訓通常需要更長時間

四、 模型分析

- 模型一：Random forest-1

  參數：

```
print(df_train.columns)
print(df_sub.columns)
```

```
Index(['msno', 'is_churn', 'day_listen', 'user_latent_satisfactio
n',
       'day_listen_level'],
      dtype='object')
Index(['msno', 'is_churn', 'day_listen', 'user_latent_satisfactio
n',
       'day_listen_level'],
      dtype='object')
```

Random state：用來控制 forest 生成的模式，讓她不會固定只生成一種 tree

n_estimators：最大迭帶次數

min_samples_spl：內部節點再劃分所需最小樣本數

n_job：應用於 bagging

class_weigh：每個 label 的權重

Train(訓練集)： Test(測試集) = 8 : 2

模型建立:

```
model = RandomForestClassifier(random_state=2,n_estimators=300,
                               min_samples_split=0.05,n_jobs=-1,class_weight={0  :0.45,1  :0.55})
```

完整模型 code：

```
def model_training_rf(training_data,testing_data):
    # splits train and validation set
    X = training_data.drop(labels=['msno','is_churn'],axis=1)
    Y = training_data['is_churn']
    X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size=0.2,random_state = 2)
    # Training ~ 01:45s
    model = RandomForestClassifier(random_state=2,n_estimators=300,
                                                    min_samples_split=0.05,n_jobs=-1,class_weight={0 :0.45,1 :0.55})
    model.fit(X_train,Y_train)

    # caculating E_val

    model_probs = model.predict_proba(X_val)
        # [:,1] to show the prob to is_churn = 1
    model_val_score = log_loss(Y_val,model_probs[:,1])

    # predict on testing set
    model_pred_testing_set = model.predict_proba(testing_data.drop(labels=['msno','is_churn'],axis=1))
    model_pred_testing_set = model_pred_testing_set[:,1] # take out the prob if is_churn = 1
    submission = pd.DataFrame({"msno": testing_data.msno})
    submission.insert(1,column='is_churn',value=model_pred_testing_set)

    return model, model_val_score, submission
```

我們在近一步進行超參數的調整以及優化，如下表：

超參數	N_estimators	Min_sample_spl	log_loss	決策
entropy	**250**	**0.01**	**0.192628**	
entropy	**250**	**0.05**	**0.192535**	
entropy	**300**	**0.01**	**0.192628**	
entropy	**300**	**0.05**	**0.192535**	
gini	**250**	**0.01**	**0.192627**	
gini	**250**	**0.05**	**0.192534**	**v**
gini	**300**	**0.01**	**0.192627**	
gini	**300**	**0.05**	**0.192535**	

- 模型二：Random forest

  與模型依使用的參數一致，但是我們將資料做切分。

```
[ ]  #沒切分的資料建模
     orginal_day_listen_model,  orginal_day_listen_val_score,\
     orginal_day_listen_pred  =  model_training_rf(df_train[day_lis],df_sub[day_lis])


     #切分資料建模
     day_listen_bins_model,  day_listen_bins_val_score,\
     day_listen_bins_pred  =  model_training_rf(df_train[day_lis_bins],df_sub[day_lis_bins])
```

分別計算其 log_loss

```
print("Original  score  :",orginal_day_listen_val_score)
print("Bins  score  :",day_listen_bins_val_score)
```

```
Original score : 0.19352482255866896
Bins score : 0.1937750209986342
```

原始的 loss 比較小的原因可能是因為將資料做切分的同時過濾了 noise，但也同時刪掉了一些有價值的資訓，所以我們傾向選擇原始資料。所以後續在做 model 建立時一律使用原始資料。

- 模型三：XG Boost-1

參數：

```
model  =  xgb.XGBClassifier(learning_rate=0.08,  max_depth=4,n_estimators=300,\
                            subsample=0.5,  seed=2,missing=-1)
model.fit(X_train,  Y_train,eval_set=xgb_watchlist,eval_metric='logloss',
                    early_stopping_rounds=20,verbose=70)
```

Learning rate：每次迭帶的步長

Max_depth：為樹的最大深度

n_estimators：最大迭帶次數

subsample：控制對於每棵樹，隨機採樣的比例

seed：控制每次隨機數據的結果

missing：將數據中缺失的值已-1 為默認值

eval_metric：logloss 代表對於二元對數的損失

early_stopping_rounds：用來控制模型過度擬合

Train(訓練集)： Test(測試集) = 8 : 2

完整模型 code：

```
def model_training_xgb(training_data,testing_data):
    # splits train and validation set
    X = training_data.drop(labels=['msno','is_churn'],axis=1)
    Y = training_data['is_churn']
    X_train, X_val, Y_train, Y_val = train_test_split(X, Y, test_size=0.2,random_state = 2)
    # model
    xgb_watchlist = [(X_train, Y_train), (X_val, Y_val)]
    model = xgb.XGBClassifier(learning_rate=0.08, max_depth=4,n_estimators=300,\
                              subsample=0.5, seed=2,missing=-1)
    model.fit(X_train, Y_train,eval_set=xgb_watchlist,eval_metric='logloss',
                       early_stopping_rounds=20,verbose=70)
    # caculating E_val

    model_probs = model.predict_proba(X_val)
        # [:,1] to show the prob to is_churn = 1
    model_val_score = log_loss(Y_val,model_probs[:,1])

    # predict on testing set
    model_pred_testing_set = model.predict_proba(testing_data.drop(labels=['msno','is_churn'],axis=1))
    model_pred_testing_set = model_pred_testing_set[:,1] # take out the prob if is_churn = 1
    submission = pd.DataFrame({"msno": testing_data.msno})
    submission.insert(1,column='is_churn',value=model_pred_testing_set)

    return model, model_val_score, submission
```

我們在近一步進行超參數的調整以及優化，如下表：

Learning rate	Max_depth	log_loss	決策
0.06	**3**	**0.191613**	
0.06	**4**	**0.191613**	
0.06	**5**	**0.191613**	

0.08	3	0.191609	
0.08	4	0.191609	
0.08	5	0.191609	
0.1	3	0.1916081	v
0.1	4	0.1916082	
0.1	5	0.1916082	

```
rf_model,  rf_val_score,\
rf_pred  =  model_training_rf(train_df[parameters],test_df[parameters])

xgb_model,  xgb_val_score,\
xgb_pred  =  model_training_xgb(train_df[parameters],test_df[parameters])

#print  log_loss
print("log_loss  of  Random  Forest  :",rf_val_score)
print("log_loss  of  XGBoost  :",xgb_val_score)
```

```
[0]    validation_0-logloss:0.629574   validation_1-logloss:0.629556
Multiple eval metrics have been passed: 'validation_1-logloss' will be used for early stopping.

Will train until validation_1-logloss hasn't improved in 20 rounds.
[70]    validation_0-logloss:0.165156   validation_1-logloss:0.165436
Stopping. Best iteration:
[95]    validation_0-logloss:0.164936   validation_1-logloss:0.165339

log_loss of Random Forest : 0.16848600995079138
log_loss of XGBoost : 0.16535620513393134
```

我們比較使用 random_forest 和 XG BOOST 在使用新的特徵值

看出 XG BOOST 的 log_loss 較 Random forest 還要小，因此我們認

為 XG BOOST 比較好。

- 模型四：Random forest-2

  在原有的特徵下我們新加入一個特徵—「registered_via」

```
[ ]  parameters  =  ['msno','is_churn','day_listen',  'satisfaction','registered_via']
```

先以 model-1 相同參數下進行模型建立，然後再進行超參數的調整以及優化，如下表：

超參數	N_estimators	Min_sample_spl	log_loss	決策
entropy	**250**	**0.01**	**0.1659466**	
entropy	**250**	**0.05**	**0.167968**	
entropy	**300**	**0.01**	**0.165948**	
entropy	**300**	**0.05**	**0.167949**	
gini	**250**	**0.01**	**0.165933**	**v**
gini	**250**	**0.05**	**0.167862**	
gini	**300**	**0.01**	**0.165939**	
gini	**300**	**0.05**	**0.167868**	

比較前後有無加入新的特徵值，發現加入新的特徵可以有效減少我們的 log_loss。我們在進一步確認是否加入新的特徵也在 XGBOOST 可以達到有效的成果。

- 模型五：XG Boost-2

在原有模型二的特徵中加入新的特徵—「registered_via」

先以 model-1 相同參數下進行模型建立，然後再進行超參數的調整以及優化，如下表：

Learning rate	Max_depth	log_loss	決策
0.06	**3**	**0.164585**	
0.06	**4**	**0.164588**	
0.06	**5**	**0.164600**	
0.08	**3**	**0.164593**	
0.08	**4**	**0.164582**	**v**
0.08	**5**	**0.164602**	
0.1	**3**	**0.164615**	
0.1	**4**	**0.164595**	
0.1	**5**	**0.164614**	

比較前後有無加入新的特徵值，發現加入新的特徵可以有效減少我們的 log_loss，因此認定加入新的特徵是有效的。

五、 結果與討論

1. 結論

利用 kaggle 上的數據集來進一步作分析和訓練，對資料的合理性和正確性做過濾，XG Boost 每次構建一個決策樹，每一個新的樹都修正以前訓練過的決策樹所產生的錯誤，而隨機森林（RF）使用隨機數據樣本獨立訓練每棵樹，這種隨機性有助於使模型比單個決策樹更健壯。但 RF 不太可能過度擬合訓練數據，隨機森林中的模型調整比 XG Boost 更容易。在 RF 中，我們有兩個主要參數：每個節點要選擇的特徵數量和決策樹的數量。RF 比 XGB 更難裝配。在各有優缺點的情況下，我們實際

的成果顯示 XG BOOST 比較好，但可能在於我們資料的不健全或偏頗。不過在我們嘗試和觀察中，XG BOOST 的 loss 較 Random forest-1 還要小，因此我們認為 XG BOOST 比較好，雖然差異可能不大。

2. 未來展望

我們認為此模型的應用性很廣泛，舉凡分類、迴歸、特徵轉換和異常點檢測等都可以做延伸和應用，而我們此次的主題預測 KKBOX 訂閱用戶流失率，在資訊爆炸的時代，許多線上付費平台都能使用此模型進行預測並規劃下期行銷策略，也能測試如何鞏固顧客忠實度。