



Music Recommender System

Group Member:

徐博揚 藍子軒 陳應中 呂函庭

Teacher:

丁德天 老師

Contents of our Research



1.研究背景與動機



2.研究方法與流程



3.結果探討



4.未來展望



Background & Motivation

在資訊發達的現代，想找音樂只需要輸入關鍵字，便能從輕易地完成閱聽人的需求，然而在有些時候使用者可能想要找到符合他喜好之新的歌曲，音樂推薦系統便能滿足此需求。本專題透過，以下的資料向閱聽人推薦符合他們喜好的音樂作品。



1.歌曲長度



2.作詞作曲者



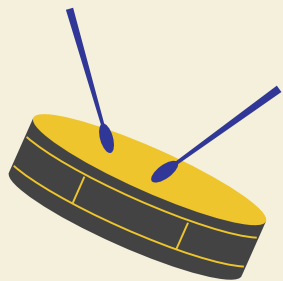
3.使用者年齡



4.註冊日期



5.軟體介面使用習慣





Data Description

資料集取自Kaggle上
WSDM-KKBOX的音樂推薦
挑戰賽，內容分成：

train.csv
test.csv
sample_submission.csv
song.csv
members.csv
song_extra_info.csv



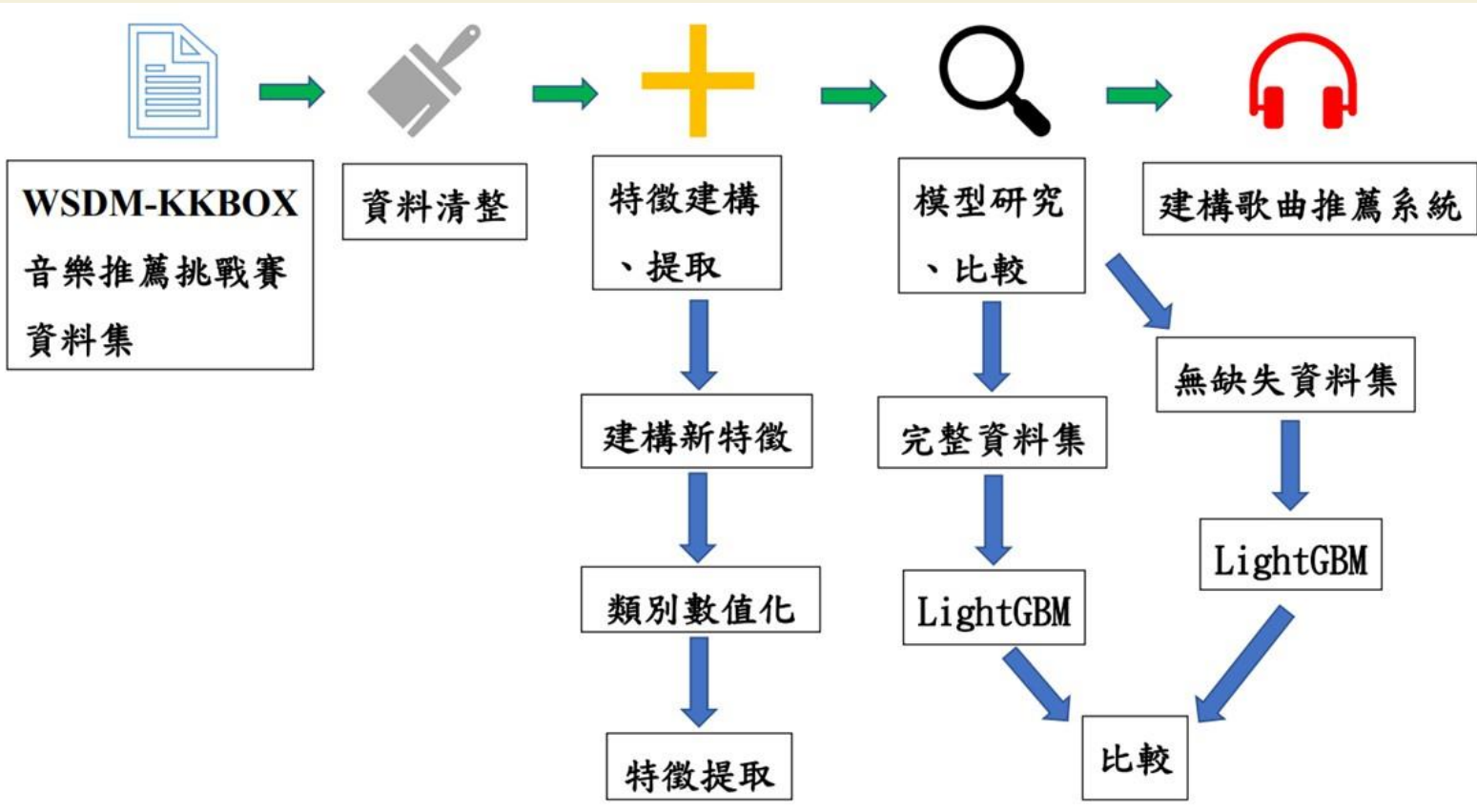
train.csv: KKBOX提供的訓練集，用戶每聽一首歌會記錄成一筆資料記在其中。
(User Based)

song.csv: 記錄歌曲相關資訊的資料集，內容包含歌曲長度、歌曲語言、歌手、歌曲風格等資訊。
(Item Based)

members.csv: 記錄KKBOX用戶相關資訊的資料集，內容包含性別、年齡、居住城市、註冊時間等資訊。
(User Based)

song_extra_info.csv: 記錄歌曲相關的額外資訊，提供歌曲名稱及isrc國際標準錄音代碼。
(Item Based)

Flow Chart





Method

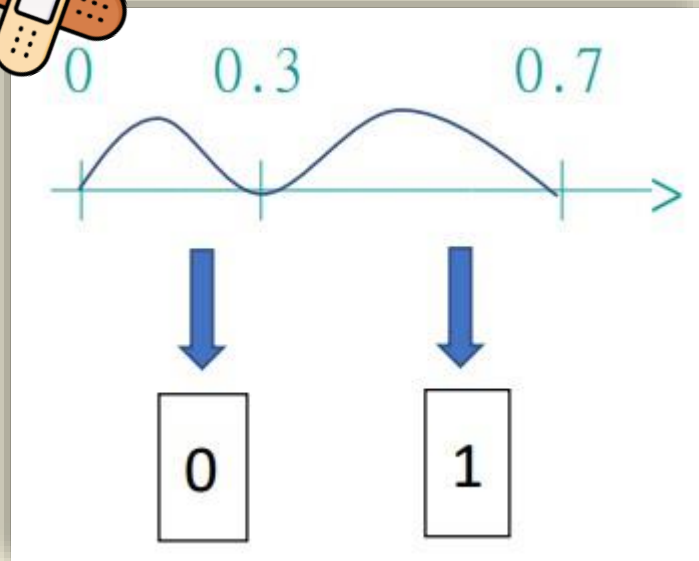
01 Histogram

02 Leaf_wise

03 GOSS

04 EFB

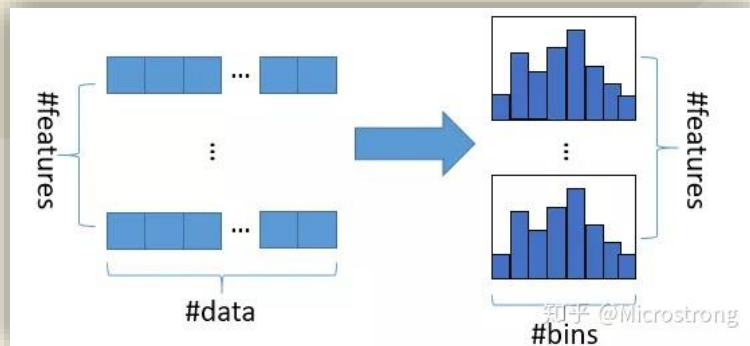
Histogram

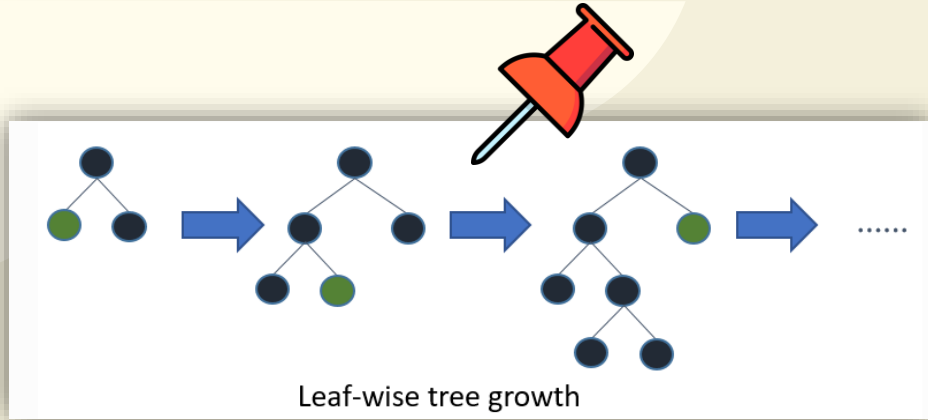


直方圖算法基本思想:

走訪數據時，根據離散化後的值作為索引直方圖中累積統計量，最後根據直方圖的離散值，尋找最優分割點。

舉例來說:[0,0.3)→0 [0.3,0.7)→1
就是將某個區間的數據映射到離散的數據值。





Leaf-Wise的主要思想為，

尋找分裂增益最大的子節點作為分裂點

優點在於分裂次數相同的情況下

1. 有更好的精確度
2. 降低更多誤差

LEAF-WISE



當樣本過小時會產生**過擬合(Overfitting)**的問題

必須設置**最大樹深(max_depth)**以防止這個問題

GOSS

GOSS是一個樣本的採樣算法，其主要思想為：

保留梯度較大的數據



若直接將所有梯度較小的數據都丟棄，勢必會影響數據的總體分布

作法：

保留絕對值最大的 a 個數據

在梯度值較小的數據中，選取 b 個，並乘以 $(1-a)/b$ 的常數

Algorithm 2: Gradient-based One-Side Sampling

Input: I : training data, d : iterations

Input: a : sampling ratio of large gradient data

Input: b : sampling ratio of small gradient data

Input: $loss$: loss function, L : weak learner

$models \leftarrow \{\}$, $fact \leftarrow \frac{1-a}{b}$

$topN \leftarrow a \times \text{len}(I)$, $randN \leftarrow b \times \text{len}(I)$

for $i = 1$ **to** d **do**

$preds \leftarrow models.predict(I)$

$g \leftarrow loss(I, preds)$, $w \leftarrow \{1, 1, \dots\}$

$sorted \leftarrow \text{GetSortedIndices}(\text{abs}(g))$

$topSet \leftarrow sorted[1:topN]$

$randSet \leftarrow \text{RandomPick}(sorted[topN:\text{len}(I)], randN)$

$usedSet \leftarrow topSet + randSet$

$w[randSet] \times = fact$ \triangleright Assign weight $fact$ to the small gradient data.

$newModel \leftarrow L(I[usedSet], -g[usedSet],$

$w[usedSet])$

$models.append(newModel)$

高維數據通常是**稀疏矩陣**具有非常多的0值，因而造成**兩個問題**！

1. 記憶體不足
2. 時間複雜度上的負擔

EFB的作用就是將**互斥特徵**綁在一起形成一個特徵，達到**降維**，以解決稀疏矩陣造成的問題





RESULT



New Feature

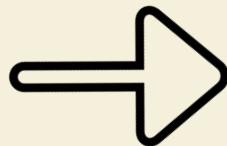
1.



2.

train中target等於1的值

歌曲被播放次數



歌曲被重複收聽率

New Feature

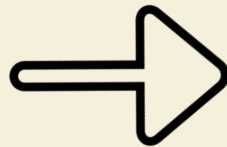
3.

註冊時間

registration_init_time

註冊到期

expiration_date

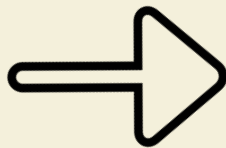


日期格式

註冊時間

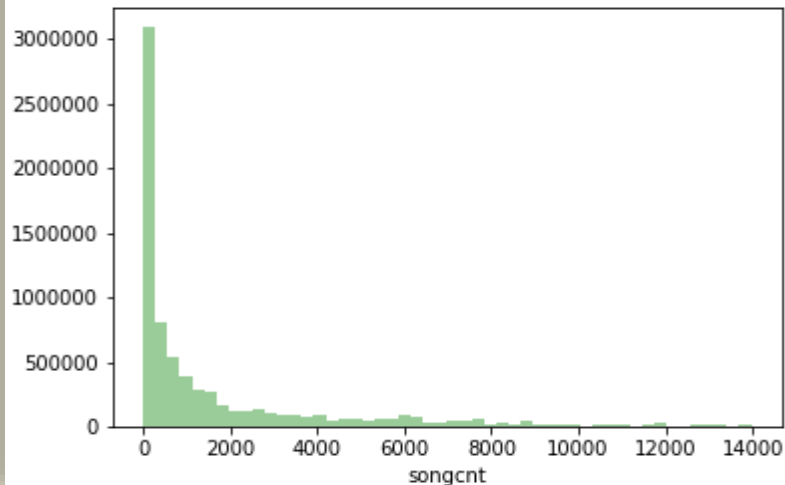
相減

註冊到期



註冊天數

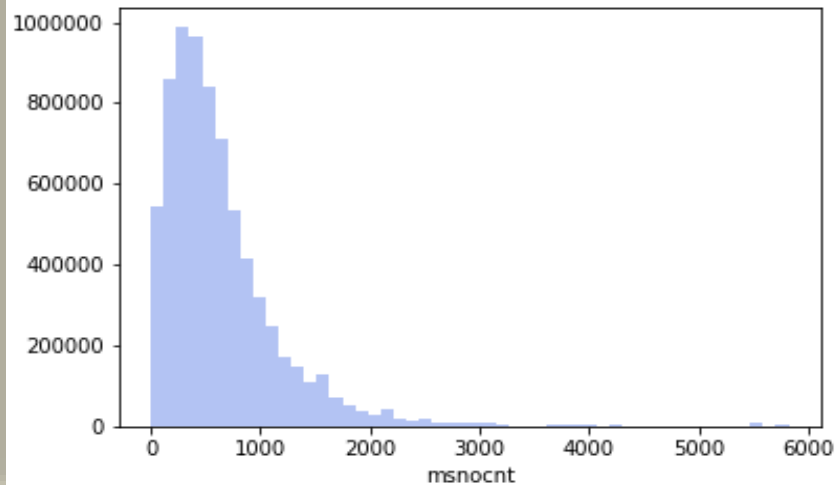
registration_days



Songcnt的長條圖以及四分位數

下四分位數	70
中四分位數	467
上四分位數	1893
IQR	1823

*Descriptive
statistics*

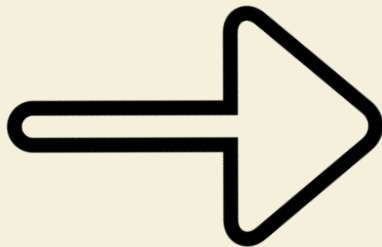


msnocnt的長條圖以及四分位數

下四分位數	286
中四分位數	510
上四分位數	838
IQR	552

Descriptive statistics

	city	target
0	A	0
1	A	1
2	A	1
3	B	0
4	B	1
5	C	1



	city	target
0	0.6667	0
1	0.6667	1
2	0.6667	1
3	0.5	0
4	0.5	1
5	1	1

類別資料平均範例示意圖



Classification Model



1. **Df_train_0(gbdt)**: 保留完整無缺失的資料的資料集，並且特徵欄位多了性別(gender)、年齡(bd)與作詞者(lyricist)，使用lightgbm，基礎學習模型選擇傳統梯度提升決策樹。



2. **Df_train_0(goss)**: 無缺失值的資料集，並且特徵欄位多了性別(gender)、年齡(bd)與作詞者(lyricist)，使用lightgbm，基礎學習模型選擇使用Gradient-based One-Side Sampling的梯度提升決策樹。



3. **Df_train_0(LR)**: 無缺失值的資料集，並且特徵欄位多了性別(gender)、年齡(bd)與作詞者(lyricist)，使用Logistic Regression演算法作分析。

Classification Model



1. `Df_train_full(gbdt)`: 使用了全部資料的資料集，使用 `lightgbm`，基礎學習模型選擇傳統梯度提升決策樹。



2. `Df_train_full(goss)`: 使用了全部資料的資料集，基礎學習模型選擇使用Gradient-based One-Side Sampling的梯度提升決策樹。



3. `Df_train_full(LR)`: 使用了全部資料的資料集，使用 `Logistic Regression`演算法作分析。

Parameter Design



1. **learning_rate**:指定一個數值為學習率
2. **num_rounds**:指定一個數值為迭代次數
3. **max_depth**:最大樹深
4. **num_leaves**:代表一棵樹上葉節點數目
5. **min_data_in_leaf**:一個葉節點包含的最少樣本數量
6. **bagging_fraction**:每次迭代選擇樣本數量百分比
7. **bagging_freq**:每幾次迭代次數執行bagging
8. **feature_fraction**:每次迭代選擇特徵數量百分比

初始參數

	無缺失資料集 (DF_TRAIN_0)		完整資料集 (DF_TRAIN_FULL)	
	goss	gbdt	goss	gbdt
LEARNING_RATE	0.1	0.1	0.1	0.1
MAX_DEPTH	21	21	19	19
NUM_LEAVES	100	100	100	100
NUM_ROUNDS	1000	1000	1000	1000
MIN_DATA_IN_LEAF	50	50	50	50
BAGGING_FRACTION		0.8		0.8
BAGGING_FREQ		2		2
FEATURE_FRACTION	0.8	0.8	0.8	0.8

Model Analysis

最佳參數

	無缺失資料集 (DF_TRAIN_0)		完整資料集 (DF_TRAIN_FULL)	
	goss	gbdt	goss	gbdt
LEARNING_RATE	0.1	0.1	0.1	0.1
MAX_DEPTH	24	26	32	29
NUM_LEAVES	450	477	533	9045
NUM_ROUNDS	262	999	985	301
MIN_DATA_IN_LEAF	160	45	70	22
BAGGING_FRACTION		1.0		1.0
BAGGING_FREQ		45		40
FEATURE_FRACTION	1.0	1.0	1.0	0.8





Model Comparison



	<u>precision</u>	<u>recall</u>	<u>F1-score</u>	<u>Training Accuracy</u>	<u>Testing Accuracy</u>
Df_train_0(<u>goss</u>)	0.74	0.72	0.73	0.748	0.715
Df_train_0(<u>gbdt</u>)	0.75	0.73	0.74	0.850	0.726
Df_train_0(LR)	0.70	0.75	0.72	0.692	0.692
Df_train_full(<u>goss</u>)	0.71	0.72	0.71	0.772	0.707
Df_train_full(<u>gbdt</u>)	0.74	0.74	0.74	0.895	0.741
Df_train_full(LR)	0.70	0.71	0.70	0.698	0.699

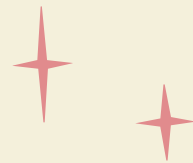












Our Prospect



Conclusion



	過擬合 的問題	預測 能力
LightGBM		
羅吉斯回歸		

	預測 能力	花費 時間
DF_train_0		
DF_train_full		

Limitation & Prospect

1. 以使用者依賴程度或以歌曲為目標進行預測。
2. 利用此研究的分析結果進行反向分析，取得推薦資料建立更有效的客戶特徵。

Reference

1. Kaggle資料集

<https://www.kaggle.com/c/kkbox-music-recommendation-challenge/data>

2. XGBoost與LightGBM演算法

<https://moread.cc/2544500.html>

3. 機器學習與XGBoost

<https://moread.cc/2544500.html>

4. 深度學習

<https://medium.com/@baubibi/速記ai課程-深度學習入門--68e27912ce30>

5. XGBoost

<https://medium.com/jameslearningnote/資料分析-機器學習-第5-2講-kaggle機器學習競賽神器xgboost介紹-1c8f55cffc>

Reference

6. 特徵工程

<https://ithelp.ithome.com.tw/users/20111826/ironman/173>

7. lightgbm使用

http://www.huaxiaozhuan.com/%E5%B7%A5%E5%85%B7/lightgbm/chapters/lightgbm_usage.html

8. LightGBM調參筆記

<https://www.twblogs.net/a/5be215942b717720b51cce01>

9. LightGBM+gridsearchcv调参

<https://zhuanlan.zhihu.com/p/76206257>

10. 基於大數據個性化音樂推薦算法分析（附代碼github地址）

<https://www.twblogs.net/a/5ee6b192f96db16fcf62f881>

Reference

11. 決策樹的進化 (ID3、C4.5、CART、GBDT、RF、DART、lambdaMART、XGBoost、lightGBM)

<https://www.itread01.com/content/1545255739.html>

12. 資料探勘-分類器的ROC曲線及相關指標 (ROC、AUC、ACC) 詳解

<https://www.itread01.com/content/1547058444.html>

13. 商品类别推荐系统：LightGBM模型

<https://blog.csdn.net/wong2016/article/details/89288531>

14. KKBOX 歌曲推薦系統

<https://wenwender.wordpress.com/2019/04/03/kkbox-%E6%AD%8C%E6%9B%B2%E6%8E%A8%E8%96%A6%E7%B3%BB%E7%B5%B1/>

15. WSDM CUP 2018 Call-for-Participants Music Recommendation & Churn Prediction

<http://www.wsdm-conference.org/2018/call-for-participants.html>

Reference

16. lightgbm調參+gridsearchcv

<https://yunglinchang.blogspot.com/2018/12/lightgbm-gridsearchcv-feat-categorical.html>

17. LightGBM explained 系列 histogram-based algorithm是什麼?

<https://yunglinchang.blogspot.com/2019/05/lgb-histogram-based-algor.html>

18. LightGBM explained 系列 Exclusive Feature Bundling

<https://yunglinchang.blogspot.com/2019/07/lightgbm-efb.html>

19. LightGBM explained 系列 Gradient-based One-Side Sampling(GOSS)是甚麼?

<https://yunglinchang.blogspot.com/2019/07/lightgbm-goss.html>

20. XGboost模型

<https://yuanc1.github.io/2019/05/17/ml/XGboost%E6%A8%A1%E5%9E%8B/>



THANKS