重大不實表達

B05702117 涂譽寶
B05702075 張文毓
B05702057 李季澄

（一）資料前處理

由於這次專案是使用飯店的訂單記錄作為訓練模型的資料，其中包含了 ID、hotel、lead time、arrival_date_year、country、meal、company、assigned_room_type、agent、customer 等特徵資料。透過 Kaggle 上的敘述仔細了解各個特徵後，我們發現有很大一部份的特徵屬於類別型特徵（例如：hotel、country、agent 等等），因此在資料前處理上，我們對所有可能影響模型預測結果的特徵做 One-hot encoding，包含 country、market_segment、distribution channel、customer_type、agent、 meal 等，並排除了 ID 這項明顯無法協助預測的特徵。而之所以不在第一個步驟就刪去大量特徵是因為我們沒有旅遊業相關的 domain knowledge，即使依據常識我們認為有些特徵對預測的結果影響不大，都還是先保留這些特徵。舉例來說，許多人可能會覺得 country 和取消訂單不會有關係，但缺乏相關實務知識的我們其實不知道民族性是否會是影響用戶取消訂單的重要因素。因此，我們的做法是在建立最初的模型後，試著將可能不重要的特徵拿掉，或是改變處理特徵的方式，並根據模型在驗證資料及上的表現來決定是否刪除變數及要採用什麼樣的特徵處理方式。值得注意的是，為了預測 revenue，我們選擇建立兩個模型，分別預測 adr 與 is_canceled，[i]而因為要預測 adr 及 is_canceled 所需的特徵會有所不同，上述資料處理的過程我們也是分別進行，使兩個模型最後使用的特徵略有不同。

在特徵處理的部分，我們嘗試了許多不同的技巧。首先是 NULL 值的部分，我們嘗試以該欄的眾數及平均值取代 NULL，最後以平均值取代 NULL 的方式在多數的變數上有比較好的效果。除了 NULL 值外，在 company 及 agent 這兩個特徵，我們發現它的資料非常分散，分別有數百種可能的值，而我們的做法便是將這些出現次數較低或者較不重要[ii]的值都歸類為『其他』，實測上在部分特徵上能達到不錯的效果。另外，我們推測總天數及總人數等特徵也會影響模型預測，因此我們將 stays_in_weekend_nights 與 stays_in_week_nights 相加為『總天數』，也將adults、children、babies 相加為『總人數』，利用原本的特徵提供給了模型更多的資訊。最後在清理資料的過程中，我們發現有部分資料的總人數及總天數為 0，這樣的情況以常識來說並不合理，但因為在測試資料集中也出現了同樣的情況，我們擔心將這些資料移除後會影響模型遇到這類資料時的處理能力，最後仍決定保留這些異常資料。另外，為了確保測試資料與訓練資料的一致性，我們在前處理的一開始就將兩組資料串接在一起，對所有資料同時做 One-hot encoding 以及特徵處理，確保不會遺漏任何的類別（避免

[i] 我們實際測試發現直接預測 revenue 的效果不佳。

[ii] 判斷方式詳見第四部分 Neural Network: Feature Selection。

有些類別特徵所包含的值只出現在訓練資料集，例如測試資料集的資料不一定有包含到所有 agent 的編號），也確保在計算眾數及平均值時能夠取得合理的值。

至於 revenue 的部分，由於助教並沒有提供 revenue 的算法，我們便試著把同一天的 revenue = adr * (stays_in_week_nights + stays_in_weekend_nights) * (1 - is_canceled) 加總在一起，並除以 10000 取整數作為最後的 label，和 train_label.csv 比對後所有的結果完全符合，因此我們推測這樣的計算方式是正確的。

（二）模型驗證

為了驗證模型的好壞，我們也測試了不同的驗證方式。起初，我們選擇將資料打亂並切分成五份做交叉驗證。然而，經過仔細思考後，我們認為這樣的驗證方式並不能有效地達到我們所預想的效果。原因在於，測試時我們是用 2015、2016 及 2017 年初的資料訓練，並預測 2017 年年中的資料，也就是說我們的模型應該要有能用過去（訓練資料）預測未來（測試資料）的能力。不過在原本打亂資料做交叉驗證的情況下，我們同時用了過去、現在、未來的資料來預測，造成模型的能力被高估，因此，我們後來改以 2015、2016 年的資料做訓練資料集、2017 年初的資料做驗證資料集。實測發現，在預測 is_canceled 的部分，最初交叉驗證的情況下準確率約為 90%，改用時間序列驗證的後準確率約為 83%，可見原本的驗證方式嚴重高估了模型能力。在確認模型的驗證方式後，我們便開始調整各種資料前處理的細節以及模型的各個參數，最後透過 validation loss 選出最好的模型及參數組合，並使用完整的訓練資料（2015、2016 及 2017 年初）再做一次訓練得到最終模型。

（三）分類模型 - is_canceled

為了預測用戶是否會取消訂單，我們建立並測試了幾種不同的分類模型，包含 Logistic Regression、Support Vector Machine、XGBoost及Random Forest，以下簡述模型效果。

(1) Logistic Regression：此模型的複雜度較低，在執行更進一步的特徵工程之前，模型在訓練資料集及驗證資料集上的表現皆不佳，有 underfitting 的問題。

(2) Support Vector Machine：在完成初步的特徵工程及資料集分割後，訓練資料集約有十萬筆資料，每筆資料有近千個特徵。而在 Scikit-learn 套件中，Nonlinear Support Vector Machine 訓練模型時的時間複雜度介於 O(Sample_Num^2 * Feature_Num) 與 O(Sample_Num^3 * Feature_Num) 之間。因此，此模型的訓練極為耗時。

(3) XGBoost：此模型能夠調整的參數較多，我們依據 Kaggle 上多數人調整此模型的方式，將參數設為 n_estimator=300、learning_rate=0.3、max_depth=15、sub_sample=1、reg_lambda=0.1、col_sample_bytree=0.1、min_child_weight=0.1，希望能夠在保有模型複雜度的同時避免 overfitting，模型在驗證資料集上的準確率為81%。

(4) Random Forest：此模型原理和 XGBoost 相似，也有許多相似的參數要調整，不過此模型的優點在於訓練速度較快，約只要 XGBoost 一半的時間便能訓練完畢，且模型下限低，通常只要訓練資料集能達到 99% 的正確率，測試資料集的表現便不會太差，模型最初在驗證資料集上的表現約為82%。

考量到四種模型的訓練時間、驗證成果、使用難易度及可解釋性，我們最終選擇了 Random Forest。在上述四種模型中，該模型的訓練速度排名第二，在驗證集上的準確率排名第一，使用難易度並列第一，且最容易解釋給機器學習的初學者聽，以下簡述我們採用這些標準選擇模型的原因。

之所以要追求模型訓練速度是因為在缺乏 domain Knowledge 的情況下，我們需要反覆測試才能知道哪些特徵以及處理特徵的方式適合用來預測用戶的取消行為。使用難易度的部分，因為 Random Forest 的下限低，參數只要在一個合理的範圍，表現都不會太差，相當大的程度上避免了因為不熟悉參數調整方式而使模型表現不佳的情況，同時大大縮短了需要在調參數上所花費的時間。最後，可解釋性的部分，在商業世界中，當我們想透過資料科學協助決策時，往往需要將技術內容呈現給上層主管聽，而隨機森林中的決策樹及各樹投票過程皆非常符合現實生活中人類群體作出決策的邏輯，這一特性使得它在商業世界中受到廣泛喜愛。

（四）迴歸模型 - adr

預測 adr 的部分，我們分別使用了 Random Forest Regressor 及 Neural Network 兩種模型進行預測，由於 Neural Network 表現的好壞牽涉到架構、epoch 數量、optimizer 以及 learning rate 的選擇，若初期便直接以 Neural Network 進行預測，當模型表現很差時，我們很難判別是資料前處理的部分沒做好還是模型的架構及各個參數沒有調整好，因此我們先以 Random Forest 的預測作為 baseline，後續 Neural Network 的參數調整皆以打敗 Random Forest 為目標。

(1) Random Forest：在未經任何特徵篩選的狀況下，Random Forest 在最初的驗證資料集已經可以達到 MSE 410 的成績（參數設定：n_estimator=200, max_depth=None, min_sample_leaf=5, random_state=0, criterion=”mse”），而此模型除了作為初步的 Baseline 外，更可以幫助我們透過訓練好的模型了解每個特徵對於預測 adr 的重要性[i]。

(2) Neural Network (Feature Selection & Normalization)：如同前提到的，我們可以透過 Random Forest Regressor 了解每個特徵對於預測 adr 的重要程度。經過觀察，可以發現有一大部分的 agent 與 country 在做完 one-hot encoding後對於 adr 的 importance 為 0，因此有嘗試在前處理時，將這些 importance為零的 agent 和 country 歸類為 “Others”後再進行 One-hot encoding，然而我們發現這樣子做的效果並沒有比直接全部做 One-hot encoding 好。我們認為這是因為 Neural Network 原本就會做許

[i] 從 Scikit-learn 中的 feature importance 可看出每個特徵中的重要程度。

多特徵篩選，只要層數跟 units 數足夠，模型就可能會有足夠的能力納入重要的特徵進行預測。

在做完 one-hot encoding 後，我們的訓練資料中就存在著一些尺度相較於其他欄位大很多的特徵，像 lead_time 通常是3位數字，相較於通常為個位數的 adults，以及眾多做完 one-hot encoding 後只有 0/1 的欄位大很多。在Andrew Ng的課程[i]中有提到，透過 normalization 可以讓 Neural Network 做梯度下降時的等高線更對稱，讓梯度下降收斂的速度更快。然而，我們發現做 normalization 後會使 Neural Network 的表現變差，推測是因為訓練資料大部分都是 0/1 的類別型資料。對資料做 normalization 後可能會使連續行變數跟類別型變數之間差異更不顯著，因此造成模型訓練效果表現不佳。也有另一種說法是我們只應該對連續型變數做 normalization ，但礙於本次專案仍有許多面向要嘗試，時間考量下，我們並沒有再多測試其他 normalization 方式，而是選擇先不使用 normalization 的技巧。

(3) Neural Network (Hyperparameter Tuning)：我們先就模型的層數及 units 數去做調整，讓第一層由 [256, 512, 1024] 個 units 開始往下每層 ÷ 2 或 ÷ 4 的遞減 (ex. 1st layer=1024 units, 2nd layer=512 units, …)，並分別嘗試了 2到 5 層四種不同的層數，同時以 Mean Squared Error 作為 loss function 並以 validation loss 的變化來設定最終訓練的 epoch 數作為模型 early stopping 的條件 (挑選validation mse最低者)。而當 validation mse 很接近時，我們會選擇 training mse 最高的。因為我們認為使用Neural Network這種比較複雜的模型，選擇E_in 較高的比較不會有 overfitting 的問題。

在選定模型架構後，我們則也嘗試了 SGD 以及 Adam 兩種 optimizer 並分別進行 learning rate 的 tuning。由於 SGD 在我們試了不同的 learning rate 及momentum 表現得都比 Adam 差，所以選定以 Adam 作為 optimizer (模型最終參數：Layer_Num=6, Unit_Num=[1024, 256, 64, 16, 4, 1], Optimizer=Adam, Learning_Rate=0.005, Batch_size=16)，最終該模型在驗證資料集上的可達到的最低 MSE 為 350。[ii]

(五) 特殊技巧

(1) Soft Label [1]：在預測 revenue 時，最基本的方式是將分類模型預測的 is_canceled 與迴歸模型預測的 adr 相乘。然而，我們其實還可以直接使用分類模型所預測的取消機率與 adr 相乘。由於原本的 Hard Label 會將 0.5 以上的值都視為 1，以下的值都視為 0，這樣的做法有可能造成我們最終算出的結果和模型實際上的預測有很大的差異。因此我們改將 Soft Label 與 adr 相乘，實測上在 Public Score Board 上分數為 0.34，為當時所有結果中最高者。

---

[i] Normalizing Inputs (C2W1L09), Andrew Ng. https://www.youtube.com/watch?v=FDCfw-YqWTE

[ii] 我們亦曾測試用 Neural Network 預測 is_canceled，但實測結果兩類模型在分類上表現差不多，Random Forest 又比較快，所以最後採用 Random Forest 來預測 is_canceled。

(2) Early Stopping By Training Loss：在訓練 Neural Network 的過程中，我們往往會透過觀察 Validation Loss 的變動來決定我們最終要訓練幾個 Epoch，然而，這樣的做法我們可能會遇到的問題是由於訓練資料集的大小改變[i]，最佳的 Epoch 數也會改變。有一個折衷的做法是，將使用訓練測試集訓練的模型作為最終模型，放棄使用剩餘的驗證資料集。這個做法的好處是能讓我們使用最適當的 Epoch 數，缺點是我們必須放棄一部份的資料。而為了找到最適當的 Epoch 數且不放棄任何可訓練的資料，我們觀察 Validation Loss 達到最低時 Training Loss 的值為何，並以此 Loss 當作最終模型（使用所有資料）訓練時 Training Loss 要達到的目標。邏輯上我們將 Training Loss 視為模型擬合資料的程度，並假設相同的擬合程度可使模型在測試資料集上有近似的表現，實測發現這樣的技巧確實能使模型表現提升[ii]。

Ensemble：Random Forest 演算法中原本就實現了 Decision Tree 的投票，然而，我們還可以進一步將幾個不同的 Random Forest 模型，甚至是各種模型結合，形成一個更不容易 overfitting 且集個模型優點於一身的模型[iii]。舉例來說，原本在預測 revenue 時，我們有選擇 Hard Label 與 Soft Label 兩種做法。雖然兩種做法預測出的 revenue 差異不大，但最終的 Label 仍可能會有所不同。像是在 2017/4/1 這天，使用 Hard Label 時，模型預測的 revenue 為 29000，而使用 Soft Label 時，模型預測的 Revenue 為 32000，因此將兩值除以 10000 取整數後的 Label 分別為 2 與3。在這種情況下，我們可以將 (29000 + 32000) / 2 = 31000 當作最後的 revenue，Label 為 3。邏輯上，我們可以將這兩個 revenue與 30000 的距離當作他們對於各自 Label 的信心，因為 32000 比 29000 距離 30000 更遠，因此他對於 3 這個結果更有信心。除此之外，我們也曾思考過只使用 2016 及 2017 年的資料做訓練是否會比使用全部資料訓練來得好（推測日期接近的資料，分布也會比較接近），因此我們也建立了只使用 2016 及 2017 年資料訓練的模型，並將其與原本的模型結合在一起。

另外，在訓練 Neural Network 時，我們觀察到 Validation Loss 的浮動非常大，雖然大致上隨著訓練過程會漸漸減少，但也會伴隨幾次不小的跳動。為了避免我們最後選擇的 adr 剛好是訓練過程中收斂比較差的結果，我們重複訓練了 30 次最終的模型（相同參數與資料），並將 30 次的結果中的離群值剔除，剩餘的值取平均來作為最後的 adr，如此一來，就能過濾掉掉訓練過程中收斂較差的結果，同時透過 ensemble 來避免 overfitting 的發生。

最終，我們透過上述三個特殊技巧，使模型從最初 Public Score Board 上的 MAE=0.4 達到了最終的 MAE=0.28（第三名）。

[i] 最終模型會使用所有能用的資料。

[ii] Public Score Board 上的分數進步 0.2 左右。

[iii] 我們亦曾將 Random Forest 與其他模型結合，例如：XGBoost，不過最終模型不佳。推測是因為原本的 XGBoost 等模型表現上都還不夠好，因此加入它們只是拖累了最後的結果。

（六）建議

(1) 模型：考量到模型參數調整的難易度，如果專案需要在很短的時間內完成的話，會建議以 Random Forest 進行 adr 及 is_canceled 的預測，除了可以快速的提供一個不錯的成績外，演算法的決策過程除了可以預測每天的 revenue 之外，更可以讓管理階層了解哪些因素會影響每筆訂單能賺取的收入以及客戶取消的機率。然而，如果需要更精進模型的預測的話，Random Forest演算法的調整就不如 Neural Network 有彈性。也因此，在此次有充足專案進行時間的狀況下，我們可以透過調整 Neural Network 的架構與參數，幫助我們得到更精確地預測。然而相反的，我們便很難透過 Neural Network 的結果去得到特別的商業 insight。

(2) 變數選擇：在最終的 Private Score Board 上我們的表現並不如在 Public Score Board 上那麼亮眼，不過相較於許多前面的組別，我們掉的名次已經很少，推測是因為我們透過了許多 ensemble 來避免 overfitting，在選擇變數的過程中也很重視驗證資料集的表現。至於表現變差的原因，我們認為可能是因為我們最終還是使用了太多無關的特徵來做預測，這些特徵也許只是剛好在驗證資料集上表現好，但沒有辦法真正地幫助我們預測。如果要避免這樣的情形，我認為實務上應該要與旅遊業專業人士合作，討論 general case 下，哪些變數對於 adr 及 is_canceled 會有影響，提升模型的泛化能力。

（七）小組分工

我們將一週劃分為三個區段，三位負責人輪流對模型進行調整，並盡量每天都上傳五次結果到 Score Board。而在各時段結束後，負責人也要寫一份簡單的報告，讓接手的同學能延續前一位的成果。如此一來，除了能避免浪費時間及上傳次數測試較差的作法外，也能確保之後的模型都有保留那些較好的作法。

三人主要負責時間及工作內容如下：

組員	主要工作	輪班時段
張文毓	資料前處理、模型驗證	星期一、五
涂譽寶	特徵篩選、迴歸模型	星期二、三、四
李季澄	分類模型、特殊技巧	星期六、日

（七）參考資料

[1] Geoggrey Hinton, Oriol Vinyals, & Jeff Dean. 2015. Distilling the Knowledge in a Neural Network. arXiv preprint arXiv: 1503.02531 (2015).