**HTML Final Project**

**組員：b08902001 龍品瑞、r08945050 蔡宇晴、b10401097 李承祐**

**組別：第幾組**

1. **Data Proccessing**
2. **Merge Data**

用pandas.read\_csv()讀csv檔案，從Train\_IDs.csv和Test\_IDs.csv讀每筆data的標籤，用這些標籤去找demographics.csv、satisfaction.csv、services.csv、status.csv和location.csv裡面相對應的資料，然後把它們用pandas.merge()合在一起。

1. **Fill empty fields**

用一些field的特性為空的field填上數值。我們用Under 30來決定空的Age要填的數值，如果小於30歲，就從已有的資料中隨機抽小於30歲的年紀來填補資料，反之亦然。如果Dependents是No，就把Number of Dependents補零。把Lat Long的值拆開來補到Latitude和Longitude裡面。如果Internet Service是No，就把和網路有關的field(Avg Monthly GB Download、Online Security、Online Backup、Device Protection Plan、Premium Tech Support、Streaming TV、Streaming Movies、Streaming Music、Unlimited Data)全部補0或No。

1. **Remove Useless fields**

把前面已經利用過或是對training沒有幫助的field丟掉，包含Customer ID(保證merge好的資料和原來的train/test\_IDs.csv裡的一樣)、Under 30(前面用過)、Dependents(前面用過)、Contry(假設都在美國)、State(假設都在加州)、Lat Long(前面用過)、Quarter(假設都是Q3)、Internet Service(前面用過，不過這好像最好不要丟掉?)、所有的Count(假設都是1)。

1. **Remove empty data & transform binary fields**

把每個field都是空的data丟掉，把所有的Yes和No以及男女性別轉成1和-1，然後把空的資料從該field抽隨機值補上。把Churn Category的字串依照kaggle上的說明換成對應的數字，如果是test data就直接丟掉這個field。

1. **One hot encoding**

因為City是離散的值，所以我們把City用sklearn的OneHotEncoder轉成許多都是1和-1的field，每個field代表那筆data原來的City是否與該field的名字相同，所以每筆data在這些轉換出來的field中只會有一個是1，其他都是-1。OneHotEncoder的City來源是從Location.csv直接讀，因為training set和test set有的城市不一樣，用這個方法才能確保它看過所有城市。接著用pandas.get\_dummies直接把剩下的field用一樣的方法轉，因為透過工人智慧可以看出所有剩下的discrete value在兩個set中都有出現，所以用這個方法就能達到一樣的效果。

1. **Normalization**

將所有field標準化(減掉平均除以標準差)，讓每個field的比重相同，較不會發生偏差。

1. **Model**
2. **Logistic Regression**

第一個model，我們使用sklearn的LogisticRegression(solver="liblinear")作為baseline model。在kaggle上可以拿到public 0.26941, private 0.26404的成績。

1. **Gradient Boosting**

第二個model我們用sklearn的GradientBoostingClassifier， 使用前先用了cross validation決定參數sklearn.model\_selection.cross\_val\_score)，最後得到的參數是n\_estimators=45時，有最好的成績，拿到public 0.27870, private 0.30595。  
固定其他參數，只變動n\_estimator的結果:

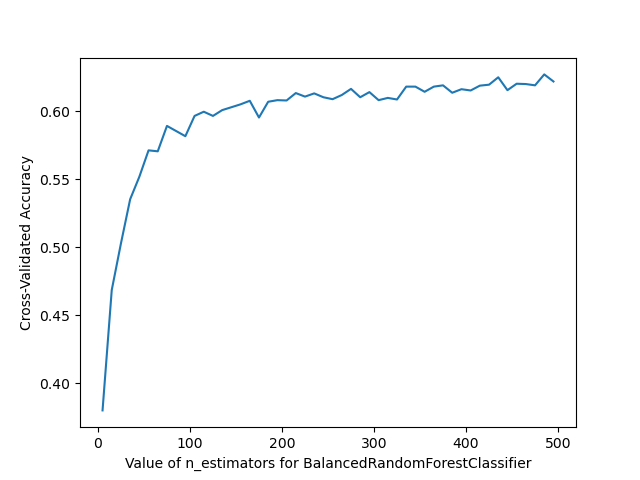


Fig 1.1

1. **Xgb Classifier**

第三個model我們使用xgboost的XGBClassifier，對test data進行cross validation來決定參數，在max\_depth=3, learning\_rate=0.09, n\_estimators=185, min\_child\_weight=5, subsample=0.8, cv=5的參數下，獲得最大平均accuracy–0.796，public 0.30741, private 0.30677的成績。

針對xgbc model的調整，設初始值默認值。然後對test set做5-cross validation，針對每一個變數實驗，使切出來的validation set的accuracy最高，再以此參數繼續實驗，以此作為最終submission訓練參數。

|  |  |
| --- | --- |
| Fig 2.1 | Fig 2.2 |
| Max\_depth為樹的最大深度，深度越小越不容易overfitting。 | Learning\_rate為學習率，控制每次疊代的布長，數值越小訓練越慢。 |
| Fig 2.3 | Fig 2.4 |
| n\_estimators為樹的數量，數量越小越不容易overfiiting。 | min\_child\_weight為最小葉子節點樣本權重和，數值越大，越容易避免學習到局部特徵，避免overfitting。 |
| Fig 2.5 |  |
| Subsample為每次建造一棵樹時使用的樣本比例，數值越小越能避免overfiiting。 |  |

我本來在使用xgbc時，發生e\_valid=0.2，卻在比賽中只拿到0.16的f-score，我認為這是在調參數時產生overfitting，在新增多項限制overfitting的參數，max-depth、n\_estimators、min\_child\_weight、subsample後，將overfitting的狀況降低，最終獲得0.3的f-score。

1. **Balanced RandomForest Classifier**

第四個model我們使用imblanced-learn的BalancedRandomForestclassifier。在n\_estimators=600, max\_depth=4參數設置下，cross-validation上獲得0.65的ACC，且在kaggle獲得public 0.27456, private 0.33861的成績。

我比較了n\_estimators在1至610且max\_depth均為4的cross-validation的acc。如下圖，可以發現隨著n\_estimators增加正確率有逐漸提升的趨勢，但是在約n\_estimators = 150時，acc會逐漸收斂在0.65。

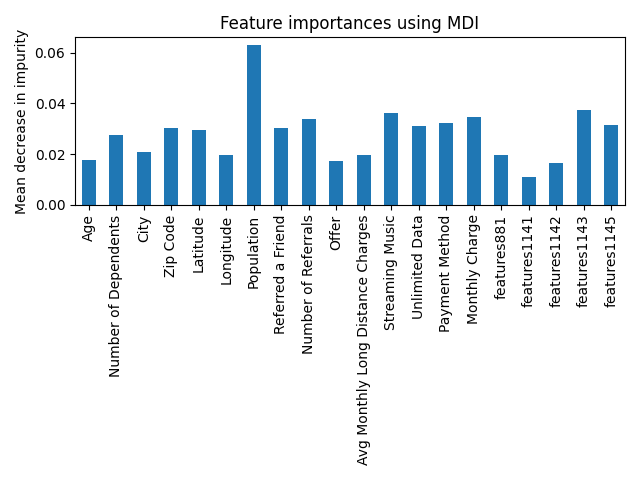
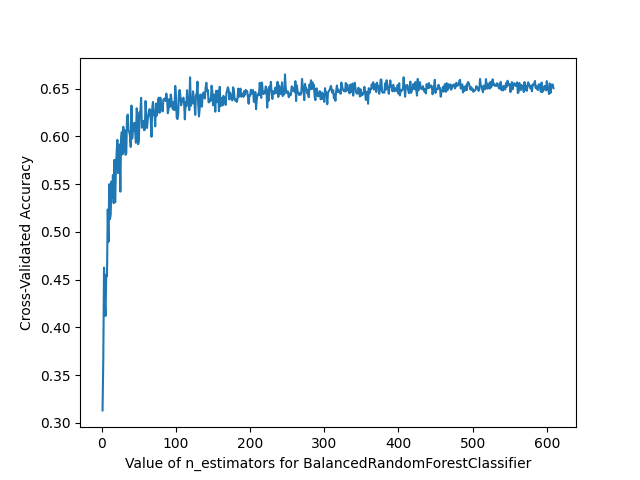


Fig 3.2

Fig 3.1

我認為每個資料集可能都會有最適合的n\_estimators，而這個數量是取決於資料集特性，並沒有一定的限制。此外，我based on mean decrease in impurity來決定各個特徵的重要性(Fig 3.2)，在這裡我們發現有趣的是Population竟然是在mean decrease in impurity評估下最重要的特徵。接著，我在Fig 3.2中選擇前20重要的特徵進行評估且depth均為4，n\_estimators為590~610，其結果分別為僅用前20重要的特徵進行分類(Fig 3.3) ,使用全部特徵進行分類( Fig3.4)，可以看出表現是選擇全部特徵會有較好的表現，我認為可以再對特徵進行更多的研究，或許某些特徵會造成分類結果變糟糕，但目前並沒有觀察出此現象。

原先我有整理了一份僅包含完整\_train\_ID(同時存在於"demographics", "location", "satisfaction", "services")的訓練集，兩千多筆。由於model表現不佳，故試著加入剩餘的訓練資料，並排除沒有Churn Category資料，總計四千多筆。接著將缺失值以個人的想法加以填值，例如年齡的部分以平均值，一些Yes/No缺失值，則是使用Churn Category相同的顧客的對應特徵進行填寫。並排除了我認為不重要的特徵(其中包含Population)。然而表現依然不佳。接著我使用品瑞的整理資料，其performance卻有大幅的改進。觀察此資料後發現，Population這個重要資訊並沒有在我的training features中。由於我對資料的不理解而造成失去重要特徵，因此我認為往後在此多種features的分類問題上，必須先觀察每一筆特徵的重要性再去適當的增減特徵，而不是先入為主的將資訊排除，畢竟不是任何特徵資訊都能經由人的先驗知識就決定的。

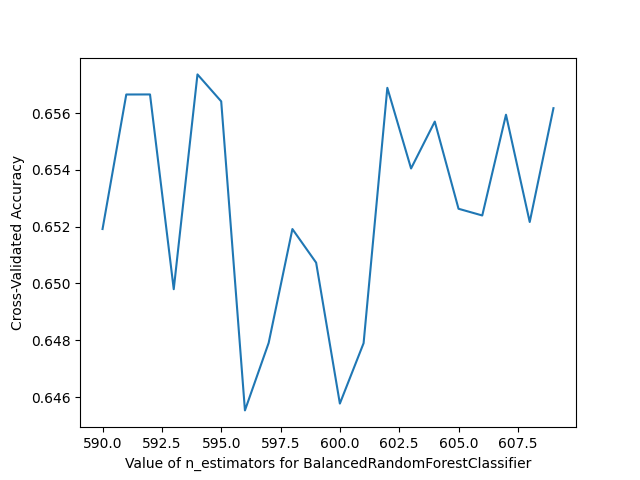
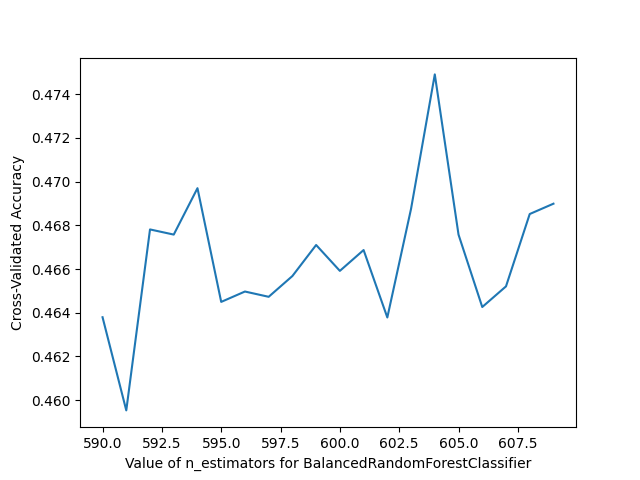


Fig 3.4

Fig 3.3

**參考資料**

1. [[資料分析&機器學習] 第5.2講: Kaggle機器學習競賽神器XGBoost介紹](https://medium.com/jameslearningnote/%E8%B3%87%E6%96%99%E5%88%86%E6%9E%90-%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92-%E7%AC%AC5-2%E8%AC%9B-kaggle%E6%A9%9F%E5%99%A8%E5%AD%B8%E7%BF%92%E7%AB%B6%E8%B3%BD%E7%A5%9E%E5%99%A8xgboost%E4%BB%8B%E7%B4%B9-1c8f55cffcc)
2. [機器學習 — Gradient Boosting](https://medium.com/@gary1346aa/%E6%A9%9F%E6%A2%B0%E5%AD%B8%E7%BF%92-gradient-boosting-1-272e8d7b17e1)
3. <https://machinelearningmastery.com/bagging-and-random-forest-for-imbalanced-classification/>
4. **分工情況**

|  |  |
| --- | --- |
| **姓名** | **工作內容** |
| **龍品瑞** | Preproccesing,Logistric regression,Gradient Boosting,撰寫報告 |
| **蔡宇晴** | Preproccesing, Balanced\_RandomForest\_Classifier,撰寫報告 |
| **李承祐** | Xgbc,撰寫報告 |