

Machine Learning Final Project

Team name : MLFighter

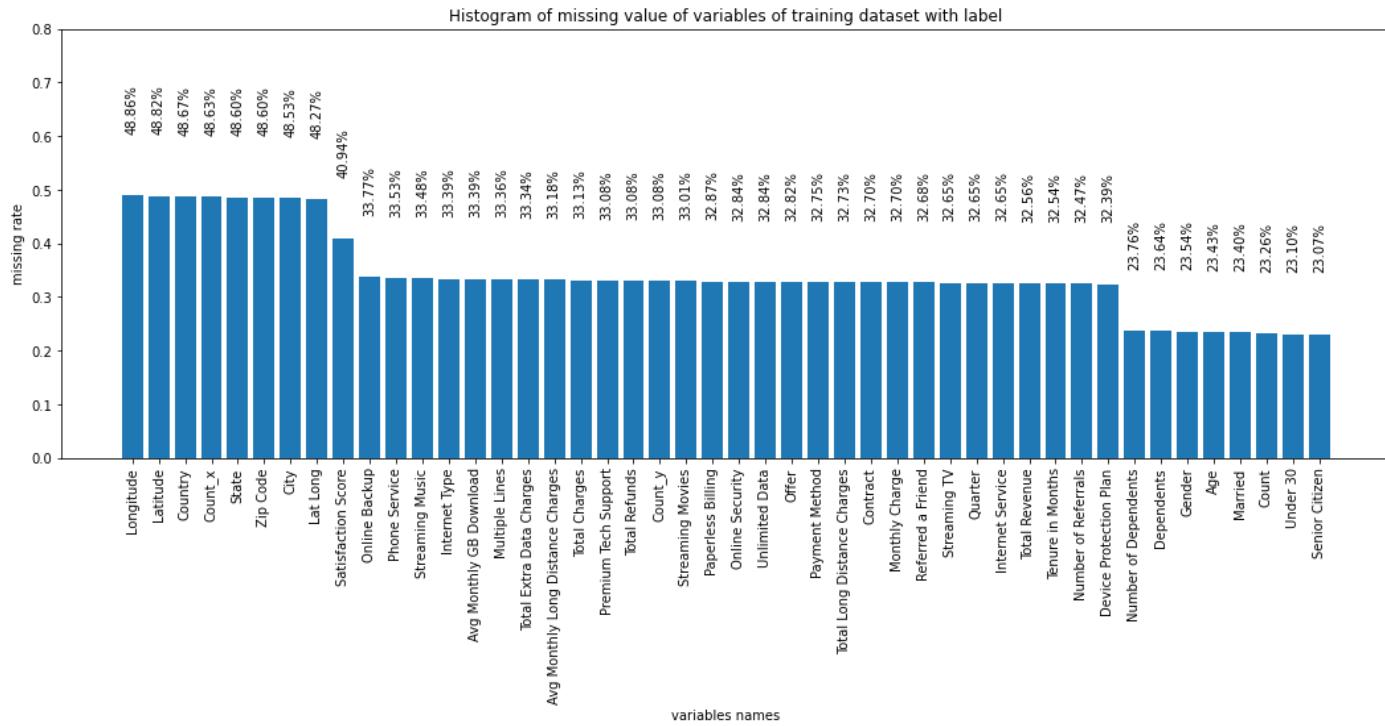
Team member : R10522801 吳政彥、R08945015 許逸翔、R10521235 范淳皓

1. Preliminary Data Analysis

在對資料進行前處理前，我們先簡易觀察資料有哪些特性，以決定資料前處理的方向。

(1) 缺失值分析

首先檢查訓練資料有標記Churn Category的客戶中(共4226筆)的各特徵有多少比例的缺失值。



(2) 各類別分佈分析

接著檢查我們拿到的dataset是balanced還是unbalanced的，下表為分析有標記Churn Category的4226筆客戶的結果，顯示為我們拿到的dataset是unbalanced的。

No Churn	Dissatisfaction	Competitor	Attitude	Price	Other
73.8%	4.2%	11.6%	4.8%	2.9%	2.7%

2. Pre-processing Data

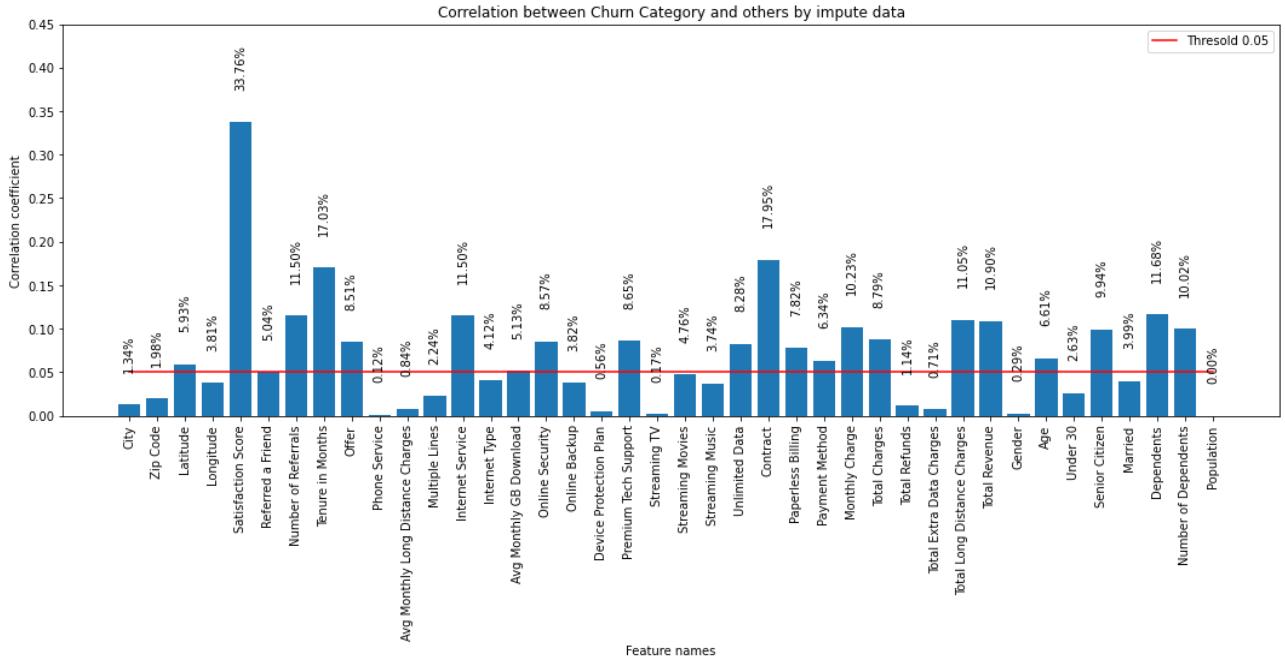
我們藉由上述所觀察到的資料特性，對資料進行前處理，以利模型的訓練。

(1) 缺失資料填補

將特徵分成數值類與類別類，數值類如Age、Monthly Charge等，會取中位數來填補缺失值；而類別類如Gender、Under 30等，會使用出現次數最多的類別(眾數)來填補。

(2) 特徵選取

透過個特徵與Churn Category相關性分析 [1], 將相關係數(pearson correlation coefficient)低於0.05的特徵剔除。



(3) 平衡資料數量

為克服dataset是unbalanced的挑戰, 我們使用Synthesized Minority Oversampling Technique (SMOTE) [2] 的過採樣技術使各數量較少的類別能補到數量最多的類別數量。

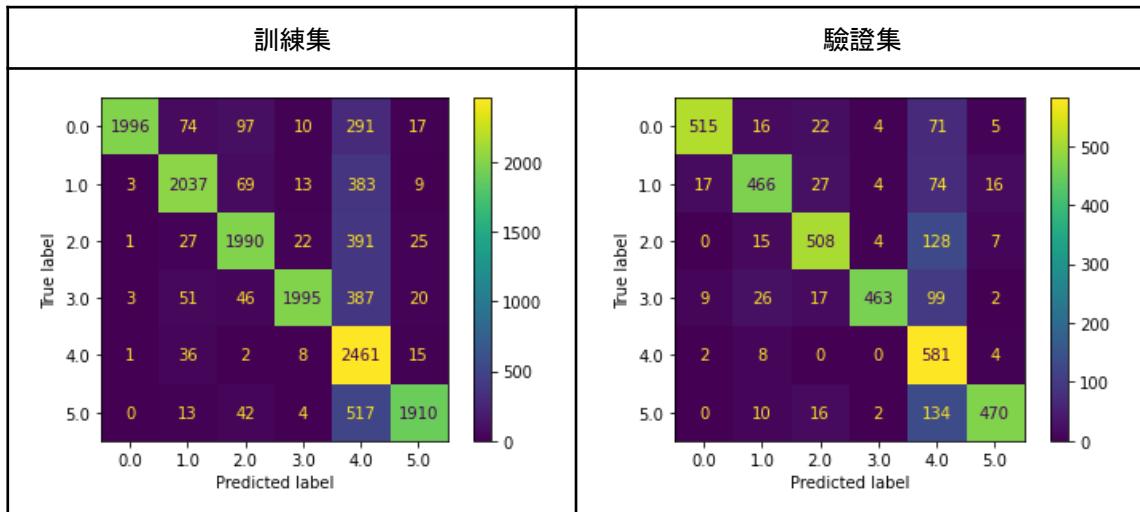
(4) 將訓練資料分成訓練集與驗證集

有標記Churn Category的訓練資料共有4226筆, 經過過採樣後有15590筆, 因此將這些資料隨機打亂後依照8:2的比例分成訓練集與驗證集。

3. Training Models

(1) SVM

使用Sklearn中的SVC [3] 模型並結合GradSearchCV來找最佳化參數, GridSearch 設定5-Fold並以F1 macro當指標, 找到參數的Kernel為rbf, Gamma為0.1, C為10。訓練後的模型在訓練集與驗證集的混淆矩陣如下圖所示:



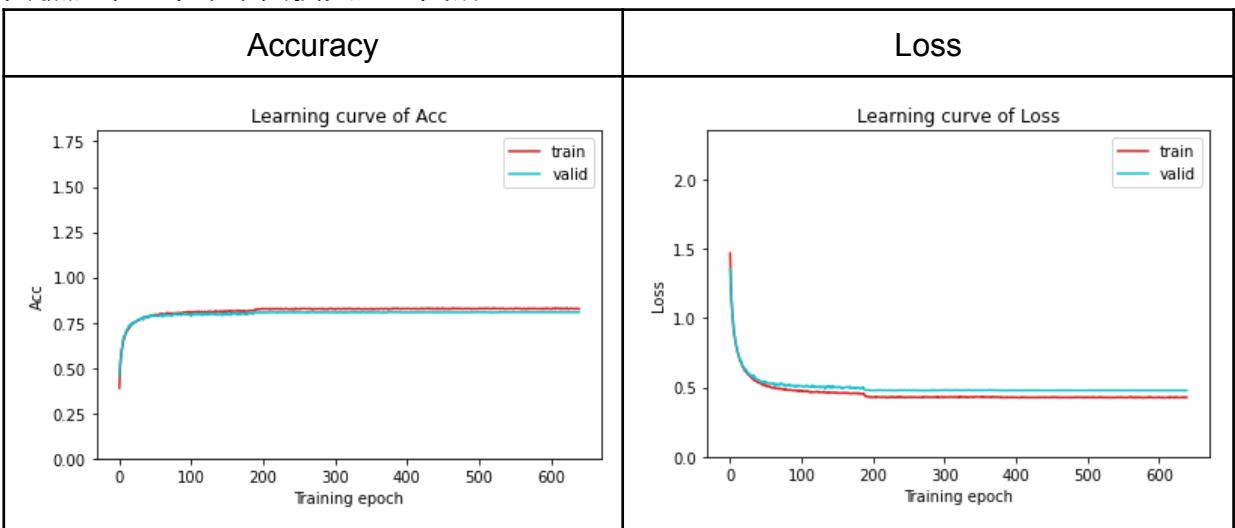
(註: 0是Attitude, 1是Competitor, 2是Dissatisfaction, 3是No Churn, 4是Other, 5是Price)

(2) DNN

使用Pytorch建立深度神經網路，網路架構與模型的超參數如下：

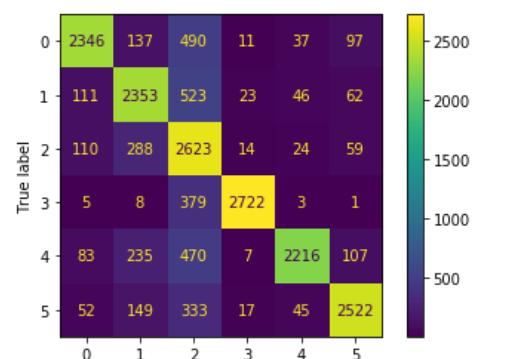
DNN架構		超參數
<pre> Layer (type) Output Shape Param # Linear-1 [-1, 4226, 256] 5,888 LeakyReLU-2 [-1, 4226, 256] 0 Dropout-3 [-1, 4226, 256] 0 Linear-4 [-1, 4226, 64] 16,448 LeakyReLU-5 [-1, 4226, 64] 0 Dropout-6 [-1, 4226, 64] 0 Linear-7 [-1, 4226, 6] 390 Total params: 22,726 Trainable params: 22,726 Non-trainable params: 0 Input size (MB): 0.35 Forward/backward pass size (MB): 31.15 Params size (MB): 0.09 Estimated Total Size (MB): 31.59 </pre>		<p>epoch : 3000 batch size : 64 optimizer : Adam learning rate : 0.001 weight dacay : 0.0005 early stop : 300 loss function : Cross-entropy learning scheduler : StepLR</p>

設定StepLR每迭代200次就會使學習率變小10倍，且在每次迭代中有使用Pseudo Labeling此技術，將訓練資料中未標註的資料輸入至訓練中的模型先預測此類別，若模型預測此類別機率有高於0.95則加入至訓練資料中一同訓練，訓練過程訓練集與驗證集的準確率與損失如下圖所示：



(3) XGBoost

使用XGBoost classifier [4] 結合SMOTE來處理資料，超參數與訓練分類結果如下：

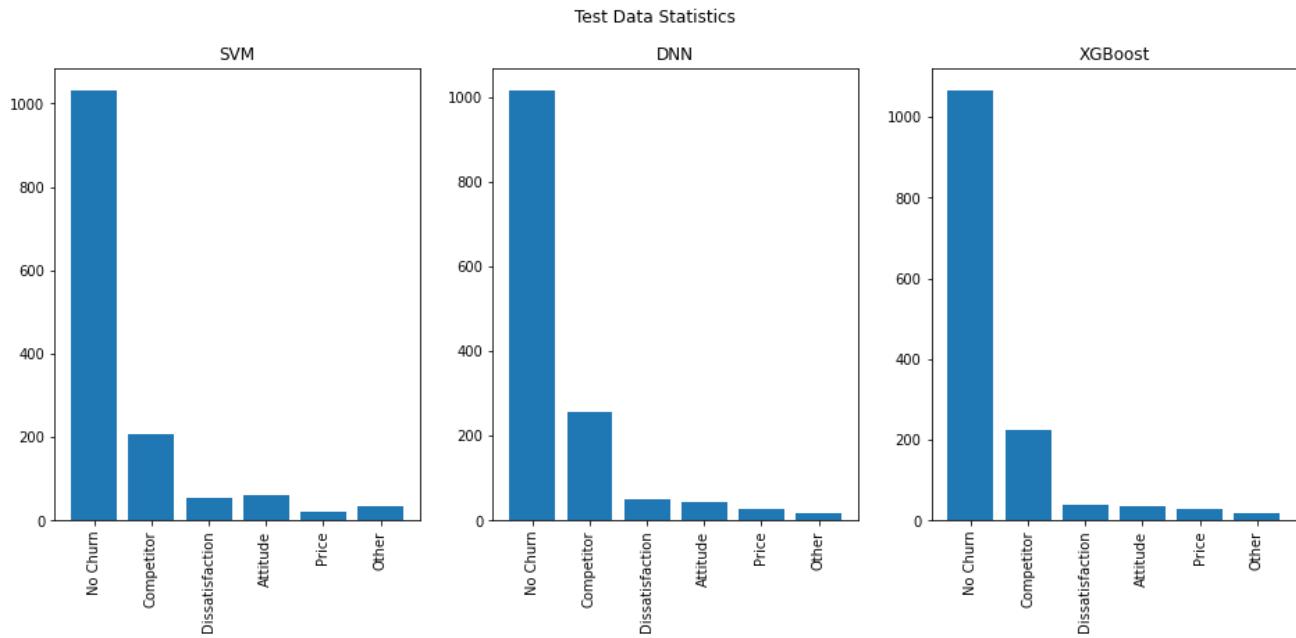
模型超參數	訓練集混淆矩陣																																																	
<pre> classifier = XGBClassifier classifier(learning_rate = 0.1, n_estimators=10, max_depth=5, min_child_weight=1, gamma=0, subsample=0.5, colsample_bytree=0.8, scale_pos_weight=1, use_label_encoder =False, eval_metric='mlogloss') </pre>	 <table border="1"> <caption>Confusion Matrix Data</caption> <thead> <tr> <th>Predicted label \ True label</th> <th>0</th> <th>1</th> <th>2</th> <th>3</th> <th>4</th> <th>5</th> </tr> </thead> <tbody> <tr> <th>0</th> <td>2346</td> <td>137</td> <td>490</td> <td>11</td> <td>37</td> <td>97</td> </tr> <tr> <th>1</th> <td>111</td> <td>2353</td> <td>523</td> <td>23</td> <td>46</td> <td>62</td> </tr> <tr> <th>2</th> <td>110</td> <td>288</td> <td>2623</td> <td>14</td> <td>24</td> <td>59</td> </tr> <tr> <th>3</th> <td>5</td> <td>8</td> <td>379</td> <td>2722</td> <td>3</td> <td>1</td> </tr> <tr> <th>4</th> <td>83</td> <td>235</td> <td>470</td> <td>7</td> <td>2216</td> <td>107</td> </tr> <tr> <th>5</th> <td>52</td> <td>149</td> <td>333</td> <td>17</td> <td>45</td> <td>2522</td> </tr> </tbody> </table>	Predicted label \ True label	0	1	2	3	4	5	0	2346	137	490	11	37	97	1	111	2353	523	23	46	62	2	110	288	2623	14	24	59	3	5	8	379	2722	3	1	4	83	235	470	7	2216	107	5	52	149	333	17	45	2522
Predicted label \ True label	0	1	2	3	4	5																																												
0	2346	137	490	11	37	97																																												
1	111	2353	523	23	46	62																																												
2	110	288	2623	14	24	59																																												
3	5	8	379	2722	3	1																																												
4	83	235	470	7	2216	107																																												
5	52	149	333	17	45	2522																																												

並使用Sklearn中的classification_report [5] 來驗證訓練結果：

	pre	rec	spe	f1	geo	iba	sup
0	0.25	0.14	0.98	0.18	0.37	0.12	36
1	0.44	0.52	0.92	0.48	0.69	0.45	97
2	0.05	0.14	0.84	0.07	0.34	0.11	44
3	0.93	0.84	0.83	0.89	0.84	0.70	622
4	0.20	0.08	0.99	0.12	0.29	0.08	24
5	0.20	0.09	0.99	0.12	0.29	0.08	23
avg / total	0.76	0.70	0.86	0.72	0.74	0.58	846

4. Model Comparison

比較三種方法在測試資料集上的分類數量如下圖所示：

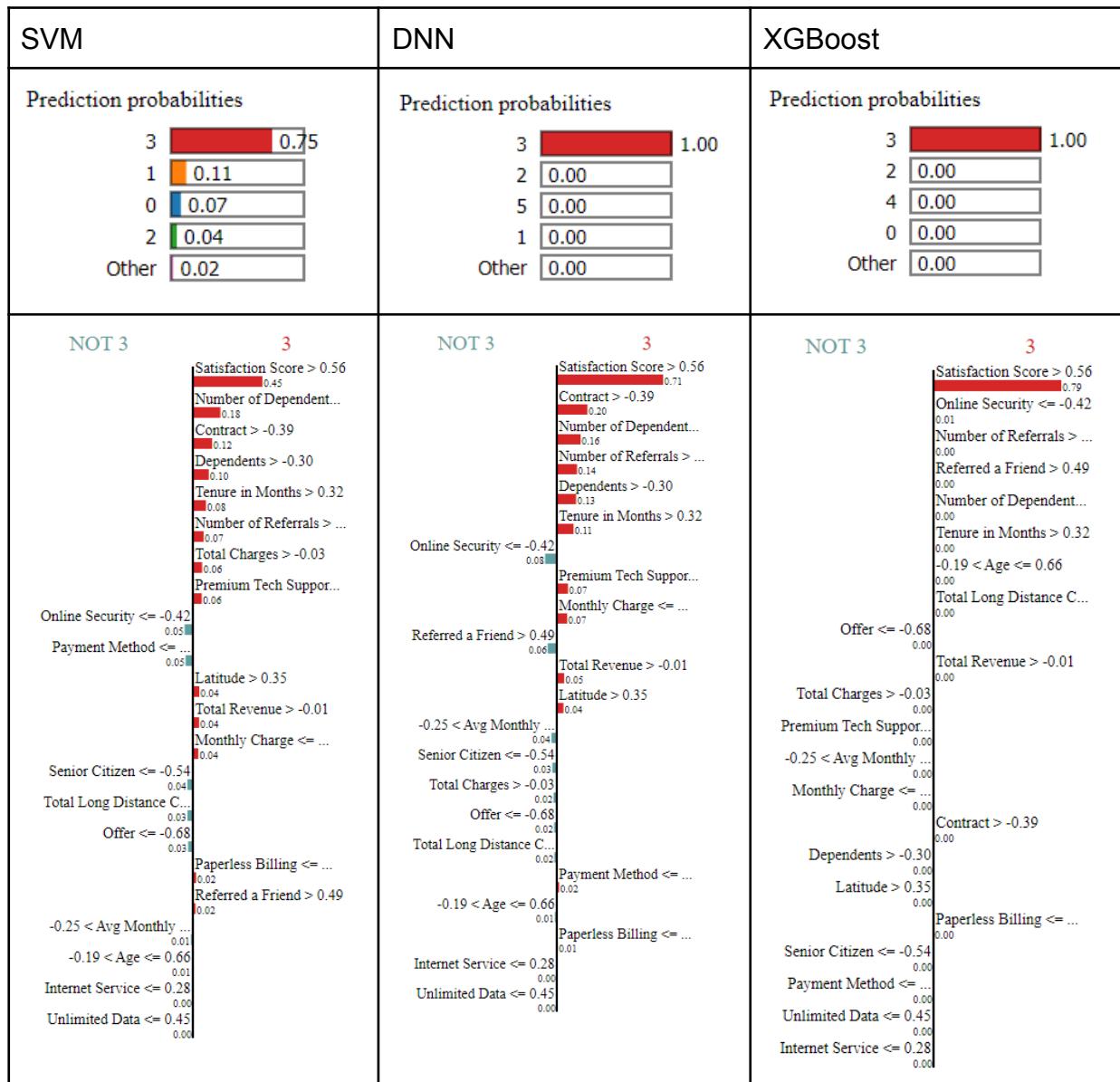


下表為三種模型的訓練時間、訓練集與驗證集上的zero one error與F1 score以及在Kaggle上的Public score與Private score：

	SVM	DNN	XGBoost
Trainng time (s)	72.11	394.727	21.355
$E_{in}^{0/1}$	0.17219	0.16838	0.11974
f1 score (in)	0.83823	0.83936	0.88834
$E_{val}^{0/1}$	0.19749	0.19081	0.1852
f1 score (val)	0.81114	0.81612	0.82165
	SVM	DNN	XGBoost

Public score	0.29708	0.28625	0.29149
Private score	0.28119	0.28633	0.33071

根據上表分析可以發現XGBoost在資料上的表現較其他兩這好，且訓練時間也是最短的。為了更深入比較三種訓練方法的差別，並想了解為了模型得到了此結果，我們使用Local Interpretable Model-agnostic Explanations (LIME) [6] 來做分析，透過LIME可以得知複雜模型中各特徵如何貢獻將某個單筆資料分類到特定的類別，因此我們選擇使用客戶ID為0454-OKRCT的資料分別輸入給三個模型來分析權重的變化，此資料的真實類別是No Churn。



(註: 0是Attitude, 1是Competitor, 2是Dissatisfaction, 3是No Churn, 4是Other, 5是Price)

三種模型預測此客戶的類別皆為No Churn，不過SVM只有75%的機率認為是No Churn，而DNN與XGBoost則是100%的肯定。在特徵權重的變化圖中可以發現SVM與DNN有其他特徵給與正向貢獻來支持此客戶類別為No Churn，而XGBoost則幾乎以

Satisfaction Score來判定，在資料前處理中我們發現相關係數前三高的特徵依序為 Satisfaction Score、Contract、Tenure in Months，從權重變化圖中可以發現也都是 Satisfaction Score分到的權重最多。這次分類只有1409筆客戶的資料，若要分類更多筆的話，應該會以XGBoost為優先，不僅訓練時間短、誤差小，且Kaggle上的得分也是最高。

5. Work Loads

組員	工作內容
許逸翔	建立SVM模型、結果分析
吳政彥	建立DNN模型、資料前處理、結果分析
范淳皓	建立XGBoost模型、結果分析

6. References

[1] Correlation matrix analysis:

<https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.corr.html>

[2] SMOTE:

https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html

[3] SVC: <https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html>

[4] XGBoost : <https://xgboost.readthedocs.io/en/stable/>

[5] Classification_report:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

[6] LIME : <https://github.com/marcotcr/lime>