巨量資料分析作業五

第一組 回不去的數統
薛丞棻、蔡宇媗、賴蓓瑩

資料介紹

10 筆資料的基本資訊:Financial、HTRU2、Bank、Wine、Shoppers、Crime都是應變數%POs低於20%的輕度或中度不平衡資料集。所有資料的分類皆為0或1的二分法。

	#Features	#Cat	#Num	Size	#Pos	%Pos	Task
Income	14	8	6	32561	7841	24.08%	Binary
Arcene	783	0	783	200	88	44.00%	Binary
Bank	16	9	7	45211	5289	11.70%	Binary
BlastChar	20	17	3	7043	1869	26.54%	Binary
Shoppers	17	2	15	12330	1908	15.47%	Binary
Shrutime	11	3	8	10000	2037	20.37%	Binary
HTRU2	8	0	8	17898	1639	9.16%	Binary
Wine	12	0	12	1599	217	13.57%	Binary
Financial	4	1	3	10000	333	3.33%	Binary
Crime	7	0	7	1200	237	19.73%	Binary

後面將用 5-fold validation 後的 Accuracy分析每一小題。

參數設定:

1. 訓練模型:以常用的XGBoost為基本，在部分題目要求下另加入其他模型。

名稱	使用參數
XGBClassifier	{'objective':'binary:logistic','max_depth': 4,'alpha': 10,'learning_rate':1.0,'n_estimators':100}
Randomforest	n_estimators=100
lightgbm	application='multiclass', boosting='gbdt', learning_rate=0.1, max_depth=-5, feature_fraction=0.5, random_state=42
MLP	hidden_layer_sizes = (256,128,64,32), activation="relu",max_iter=50, random_state=1

SVM	kernel='rbf',max_iter=10000

2. **重抽樣(resampling)**模型：僅列出有額外標示參數的，其餘未出現則為空白

名稱	使用參數
Nearmiss	**sampling_strategy = 'majority'**
Cluster Centroids	**voting='hard'**
ENN	**kind_sel="all"**
SMOTEENN	**smote = SMOTE(), enn = EditedNearestNeighbours(sampling_strategy='all')**
SMOTE TOMAK	**smote = SMOTE(), tomek = TomekLinks(sampling_strategy='majority')**

[P1] How does feature scaling (i.e., doing standardization or not) affect the performance?

以這邊經過 Label Encoding 為 Default 的資料來說，沒影響表現。

	標準化前	標準化後
Income	0.8626	0.8627
Arcene	0.7650	0.7650
Bank	0.6251	0.6251
BlastChar	0.7650	0.7647
Shoppers	0.8771	0.8771
Shrutime	0.8404	0.8404
HTRU2	0.9765	0.9765
Wine	0.8587	0.8587
Financial	0.9673	0.9673
Crime	0.9673	0.9673

[P2] When using tree‑based algorithms, will using one‑hot encoding for categorical features generate worse performance than using label encoding? Why?

交叉驗證為互有優勝。
可能在地區等類別數較多的資料，one-hot encoding 會使機器學習陷入 curse of dimensionaility，導致成效較不佳。

	Label Encoding	One Hot Encoding
Income	0.8626	0.8605
Arcene	0.7650	0.7650
Bank	0.6251	0.6141
BlastChar	0.7650	0.7671
Shoppers	0.8771	0.8770
Shrutime	0.8404	0.8428
HTRU2	0.9765	0.9751
Wine	0.8587	0.8674
Financial	0.9673	0.9710
Crime	0.9350	0.9420

[P3] Will feature binning provide performance improvement? When does binning be useful (which models or which kinds of datasets)? Which binning methods work better?

Frequency Binning 較 Equal Width Binning 佳，推測是部分數據分布較廣，Frequency Binning 較能顯示該筆數據特定特徵在整個資料集的排序位置或大小，亦能調整outlier，讓該特徵的數值差異變小。

	Equal Width Binning	Frequency Binning
Income	0.8357	0.8331
Arcene	0.7650	0.7650
Bank	0.6141	0.6215

BlastChar	0.7671	0.7680
Shoppers	0.8770	0.8756
Shrutime	0.8428	0.8389
HTRU2	0.9761	0.9740
Wine	0.8674	0.8630
Financial	0.9710	0.9455
Crime	0.9304	0.9350

[P4] Compare the performance of 6 different categorical feature encoding methods based on Random Forest, XGBoost, LightGBM, MLP, SVM. Which of the 6 encoding methods is better?

以大部分數據來說 XGBoost = LightGBM > Random Forest > MLP = SVM。

Label Encoding 後	XGBoost	Random Forest	Lightgbm	MLP	SVM
Income	0.8605	0.8537	0.8742	0.7891	0.7949
Arcene	0.7650	0.8300	0.8000	0.8400	0.7150
Bank	0.6251	0.6984	0.6533	-	-
BlastChar	0.7650	0.7898	0.7954	-	-
Shoppers	0.8771	0.8937	0.8940	0.8864	0.8475
Shrutime	0.8404	0.8612	0.864	0.742	0.7963
HTRU2	0.9740	0.9747	0.9764	0.9743	0.9769
Wine	0.8587	0.8662	0.8612	0.8649	0.8643
Financial	0.9673	0.9694	0.9715	0.9651	0.9667
Crime	0.8404	0.8612	0.864	0.742	0.7963

One hot	XGBoost	Random	Lightgbm	MLP	SVM

Encoding 後		Forest			
Income	0.8605	0.8537	0.8742	0.7891	0.7949
Arcene	0.7650	0.83000	0.8000	0.7450	0.7150
Bank	0.5856	0.7021	0.6636	-	-
BlastChar	0.7701	0.7838	0.7961	-	-
Shoppers	0.8758	0.8912	0.8906	0.8787	0.8921
Shrutime	0.8367	0.8616	0.8656	0.8132	0.7963
HTRU2	0.9751	0.9752	0.9766	0.9731	0.9764
Wine	0.8505	0.8674	0.8618	0.8374	0.8643
Financial	0.9706	0.9703	0.9702	0.9703	0.9708
Crime	0.8367	0.8616	0.8656	0.8132	0.7963

Frequency Encoding 後	XGBoost	Random Forest	Lightgbm	MLP	SVM
Income	0.8608	0.8589	0.8736	0.7785	0.7954
Arcene	0.7650	0.8300	0.8000	0.7450	0.7150
Bank	0.6143	0.6476	0.6307	-	-
BlastChar	0.7659	0.7889	0.7988	-	-
Shoppers	0.8789	0.8923	0.8917	0.8826	0.8890
Shrutime	0.8404	0.8621	0.8640	0.7014	0.7963
HTRU2	0.9707	0.9699	0.9699	0.9084	0.9699
Wine	0.8487	0.8687	0.8630	0.8593	0.8643
Financial	0.9673	0.9691	0.9715	0.9549	0.9667
Crime	0.8404	0.8621	0.864	0.7014	0.7963

Target	XGBoost	Random	Lightgbm	MLP	SVM

Encoding 後		Forest			
Income	0.8601	0.8592	0.8741	0.7930	0.7949
Arcene	0.7650	0.8300	0.8000	0.8550	0.7150
Bank	0.6063	0.6636	0.6455	-	-
BlastChar	0.7613	0.7911	0.7990	-	-
Shoppers	0.8760	0.8950	0.8931	0.8905	0.8475
Shrutime	0.8409	0.8633	0.8636	0.6849	0.7963
HTRU2	0.9707	0.9699	0.9699	0.9720	0.9702
Wine	0.8687	0.8837	0.8768	0.8581	0.8643
Financial	0.9673	0.969	0.9715	0.9483	0.9667
Crime	0.8409	0.8633	0.8636	0.6849	0.7963

Leave-one-out Encoding 後	XGBoost	Random Forest	Lightgbm	MLP	SVM
Income	0.8609	0.8599	0.8745	0.7934	0.7952
Arcene	0.7650	0.8100	0.8000	0.8500	0.7150
Bank	0.6189	0.6739	0.6584	-	-
BlastChar	0.7613	0.7909	0.7990	-	-
Shoppers	0.8760	0.8970	0.8931	0.8751	0.8475
Shrutime	0.8409	0.8635	0.8636	0.6849	0.7963
HTRU2	0.9704	0.9698	0.9700	0.9720	0.9702
Wine	0.8705	0.8824	0.8824	0.8649	0.8643
Financial	0.9673	0.9692	0.9715	0.9483	0.9667
Crime	0.8409	0.8635	0.8636	0.6849	0.7963

[P5] Which combinations of numerical and categorical feature transformation methods generally lead to better results?

丟棄有遺失值之樣本後，挑選常用的六個組合進行比較，發現 Standard Scalar 效果較佳，且 Label Encoder 效果又較 Target Encoding、LOO 佳。
以income來講，Standard Scalar 效果較佳，且 Label Encoder 效果又較 Target Encoding、LOO 佳，但 Telco 的效果差不多;數值型資料越多，越容易受Target跟LOO影響，其原因為可能遺失資訊。

[P6] If the number of possible categorical values of a feature is high, which encoding methods among target encoding, one‑hot encoding, and label encoding will have better performance? Why?

普遍來說，One-Hot Encoding 通常表現較差，Label Encoding 與 Target Encoding則是資料性質為連續型的多還是類別型的多。

	Label Encoding	**One‑Hot Encoding**	**Target Encoding**
Income	0.8605	0.8610	0.8606
Arcene	0.7650	0.7650	0.7650
Bank	0.6251	-	-
BlastChar	0.7650	0.7642	0.7613
Shoppers	0.8771	0.8756	0.8760
Shrutime	0.8404	0.8429	0.8409
HTRU2	0.9765	0.9766	0.9772
Wine	0.8587	0.8524	0.8681
Financial	0.9673	0.9652	0.9654
Crime	0.8404	0.8429	0.8409

[P7] Compare the classification performance of "doing nothing", 7 undersampling, 4 oversampling, 2 ensemble‑based methods in the presence of class imbalance. Which method works generally the best and the worst? Why?

表現最好的模型是 SMOTE+ENN，表現最差的模型是Nearmiss。

Undersampling: (Condensed NN因為訓練時間過久未納入)

	NearMiss	Cluster Centroids	Edited NN	NCR	Tomek Links	OSS
Income	0.8089	0.8360	0.8589	0.8570	0.8626	0.8619
Arcene	0.6884	0.7898	0.8074	0.8239	0.7397	0.7733
Bank	0.7703	0.6501	0.6371	0.6167	0.6171	0.7451
BlastChar	0.5688	0.7164	0.8568	0.8446	0.7868	0.7930
Shoppers	0.9342	0.6761	0.9188	0.9175	0.8874	0.8854
Shrutime	0.9099	0.7449	0.7912	0.7953	0.8304	0.8321
HTRU2	0.9335	0.9100	0.9840	0.9842	0.9782	0.9793
Wine	0.6751	0.7144	0.8687	0.8715	0.8656	0.8554
Financial	0.8694	0.8349	0.9649	0.9655	0.9665	0.9667
Crime	0.9573	0.7978	0.8144	0.9586	0.9472	0.9322

Oversampling: (Borderline‑SMOTE SVM 因時間關係未納入)

	SMOTE	Borderline‑SMOTE	ADASYN
Income	0.8759	0.8751	0.8710
Arcene	0.8081	0.8040	0.8038
Bank	0.7559	0.7486	0.8149
BlastChar	0.8227	0.8170	0.8148
Shoppers	0.8349	0.8349	0.8189
Shrutime	0.8306	0.8780	0.8402
HTRU2	0.9682	0.9879	0.9680
Wine	0.9038	0.9045	0.9018
Financial	0.9761	0.9782	0.9682
Crime	0.9487	0.9473	0.9323

Combined:

	SMOTE + ENN	SMOTE + Tomek Links
Income	0.9282	0.8801
Arcene	0.8826	0.7740
Bank	0.7523	-
BlastChar	0.9505	0.8278
Shoppers	0.9584	0.8511
Shrutime	0.8781	0.8402
HTRU2	0.9880	0.9680
Wine	0.9412	0.9021
Financial	0.9844	0.9788
Crime	0.9785	0.9412

[P8] Can you find SMOTE-based oversampling works better on which kinds of datasets (what does the data look like)? Why?

適合類別變數較多的資料，以 Crime與 HTRU_2這兩筆資料為例，Crime除了經緯度以數值資料形式作分析，其他皆為類別資料；HTRU_2則預設全部為連續資料。在 SMOTE-based oversampling下，即便Crime的%POs有 17.73%，HTRU_2 只有 9.16%，但 Crime在 SMOTE與 Borderline-SMOTE的準確率皆有97%以上，而 HTRU_2 在同樣 resampling 方法下只有95%~96% 左右。

[P9] Is a dataset's imbalance ratio (e.g., %Pos) related to choosing which resampling strategy for better performance? Any insights?

似乎不是 imbalance ratio在影響，大部分數據 Combined 的結果會最好，因為可以較接近原始資料筆數，再來是 Oversampling > Underssampling。

[P10] How do different ML algorithms (Random Forest, XGBoost, LightGBM, MLP, SVM) prefer different resampling strategies for better performance of imbalance classification? Describe any findings here.

如Arcene那種生物醫學的資料有較多特徵，使用MLP分析較佳，如果是一般20個以下的特徵使用Random Forest, XGBoost, LightGBM會較好。SVM則無論是哪一種資料表現都較差。

(XGBoost 在第七題已經實驗過了)

ENN	Random Forest	Lightgbm	MLP	SVM
Income	0.8519	0.8650	0.7250	0.6981
Arcene	0.9068	0.8941	0.9131	0.7763
Bank	0.7537	0.7014	-	-
BlastChar	0.8631	0.8664	-	-
Shoppers	0.9306	0.9305	0.9153	0.8389
Shrutime	0.8112	0.8180	0.6183	0.6719
HTRU2	0.9852	0.9853	0.9826	0.9787
Wine	0.8832	0.8740	0.8870	0.8343
Financial	0.9561	0.9658	0.9635	0.9635
Crime	0.9520	0.9426	0.8733	0.852

Tomek Links	Random Forest	Lightgbm	MLP	SVM
Income	0.8588	0.8650	0.7785	0.7798
Arcene	0.8214	0.8372	0.8214	0.6991
Bank	0.7044	0.6483	-	-
BlastChar	0.8076	0.8134	-	-
Shoppers	0.9028	0.8998	0.8962	0.8439
Shrutime	0.8525	0.8552	0.7378	0.7746
HTRU2	0.9808	0.9795	0.9770	0.9736
Wine	0.8751	0.8604	0.8681	0.8610
Financial	0.9576	0.9684	0.9661	0.9661
Crime	0.9370	0.9334	0.8692	0.8478

OSS	Random Forest	Lightgbm	MLP	SVM
Income	0.8579	0.8650	0.7699	0.7798
Arcene	0.8031	0.7980	0.7778	0.7236
Bank	0.7031	0.6534	-	-
BlastChar	0.8068	0.8116	-	-
Shoppers	0.9017	0.9005	0.8958	0.8432
Shrutime	0.8529	0.8544	0.6348	0.7742
HTRU2	0.9806	0.9789	0.9776	0.9735
Wine	0.8619	0.8632	0.8664	0.86
Financial	0.9573	0.9683	0.9660	0.9660
Crime	0.9268	0.9304	0.8573	0.8441

SMOTE	Random Forest	Lightgbm	MLP	SVM
Income	0.8794	0.8789	0.6026	0.5029
Arcene	0.8214	0.8573	0.9066	0.7501
Bank	0.7904	0.7695	-	-
BlastChar	88371	0.8283	-	-
Shoppers	0.9036	0.8223	0.8585	0.7209
Shrutime	0.8537	0.8487	0.5438	0.5730
HTRU2	0.9771	0.9614	0.9479	0.9258
Wine	0.9077	0.8976	0.7970	0.7329
Financial	0.9673	0.9664	0.4999	0.5406
Crime	0.9403	0.9535	0.8217	0.6685

Borderline-SMOTE	Random Forest	Lightgbm	MLP	SVM

Income	0.8787	0.8768	0.5437	0.5024
Arcene	0.8348	0.8885	0.8706	0.7410
Bank	0.8075	0.7764	-	-
BlastChar	0.8353	0.8304	-	-
Shoppers	0.9034	0.8600	0.8616	0.7182
Shrutime	0.8564	0.8471	0.5673	0.5853
HTRU2	0.9539	0.9477	0.9281	0.9205
Wine	0.9030	0.8979	0.7992	0.7358
Financial	0.9716	0.9747	0.5011	0.6093
Crime	0.9396	0.9480	0.8252	0.6733

SMOTE + ENN	**Random Forest**	**Lightgbm**	**MLP**	**SVM**
Income	0.9245	0.9218	0.6700	0.6278
Arcene	0.9308	0.9560	0.9373	0.8621
Bank	0.8678	0.8403	-	-
BlastChar	0.9505	0.9467	-	-
Shoppers	0.9645	0.9532	0.9278	0.7960
Shrutime	0.8630	0.8613	0.5907	0.6550
HTRU2	0.9906	0.9848	0.9703	0.9560
Wine	0.9406	0.9423	0.8333	0.7646
Financial	0.9761	0.9788	0.5059	0.6131
Crime	0.9791	0.9826	0.8626	0.7417

SMOTE + Tomek Links	**Random Forest**	**Lightgbm**	**MLP**	**SVM**

Income	0.8834	0.8823	0.8874	0.7298
Arcene	0.8473	0.8697	0.8875	0.7299
Bank	0.7910	0.7599	-	-
BlastChar	0.8408	0.8369	-	-
Shoppers	0.9074	0.8460	0.8664	0.7283
Shrutime	0.8564	0.8508	0.5250	0.5881
HTRU2	0.9789	0.9628	0.9462	0.9258
Wine	0.9096	0.9042	0.7675	0.7163
Financial	0.9096	0.9042	0.7675	0.7163
Crime	0.9739	0.9702	0.5330	0.5528

Datasets 來源
https://www.kaggle.com/lodetomasi1995/income-classification
https://archive.ics.uci.edu/ml/machine-learning-databases/arcene/
https://archive.ics.uci.edu/ml/datasets/bank+marketing
https://www.kaggle.com/blastchar/telco-customer-churn
https://archive.ics.uci.edu/ml/datasets/Online+Shoppers+Purchasing+Intention+Dataset
https://www.kaggle.com/shrutimechlearn/churn-modelling
https://www.kaggle.com/datasets/jinbonnie/crime-and-weed
https://www.kaggle.com/datasets/kmldas/loan-default-prediction?resource=download
https://archive.ics.uci.edu/ml/machine-learning-databases/00372/
https://www.kaggle.com/datasets/uciml/red-wine-quality-cortez-et-al-2009

參考資料
https://www.kaggle.com/datasets/ashkanranjbar/chicago-crime
https://ithelp.ithome.com.tw/articles/10235726
http://contrib.scikit-learn.org/category_encoders/leaveoneout.html