機器學習 _ 信用卡發放分析

聯成電腦 _ 機器學習與影像識別

目錄

問題定義：

人力判斷客戶的特徵是否符合發卡資格是非常耗時耗力的，用機器學習的方式預測是否將信用卡發放給給定特徵的人，既快速又準確。

5W1H ：

Who ：

信用卡公司分析工程師、想要申請信用卡的客戶、・・・等。

What ：

問題：分析和預測哪些客戶更有可能獲得信用卡發放。

數據：包括客戶的收入、支出、信用記錄、就業情況、居住狀況等信息。 模型目標：建立一個能夠根據客戶資料預測信用卡批核結果的機器學習模型

When ：

模型更新：不能拿過於久遠的數據預測預測當今人類是否符合標準。

判斷發卡時機：當客戶提交信用卡申請時，模型能預測其批核結果。

Where ：

電腦、雲端平台・・・等。

Why ：

風險管理：預測信用風險。 客戶體驗：用機器能快速準確的批核過程提升客戶滿意度。

How ：

數據預處理：特徵工程、檢查缺失值、分析數據合理性、正規化・・・等。

模型選擇：選擇合適的機器學習算法。 訓練和評估：使用歷史數據訓練模型並使用測試數據做評估。

資料來源：

資料說明：

此資料集來自《Econometric Analysis》這本書，是一個申辦信用卡客戶資料的虛擬資料集。此資料集如如表一所示所示擁有 12 個欄位，且總共有 1,319 筆資料。

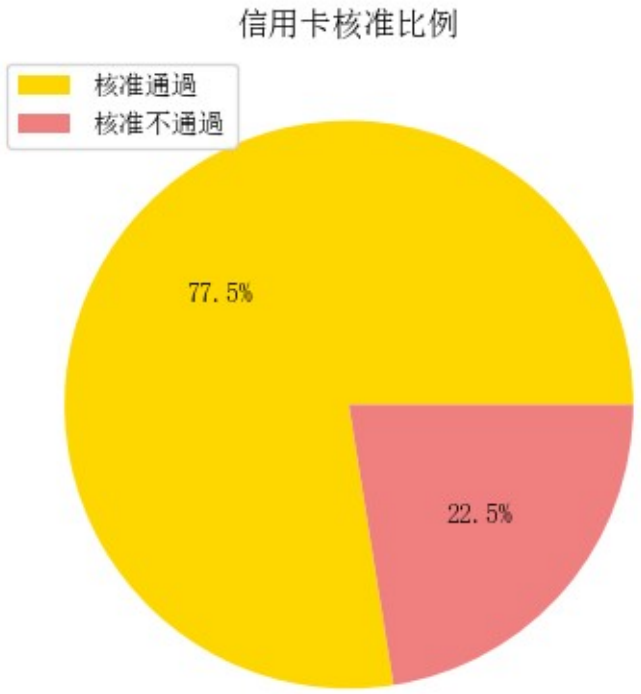
URL:https://www.kaggle.com/datasets/dansbecker/aer-credit-card-data/code

特徵介紹：

	欄位說明
Card	代表信用卡的核準結果，為本次專案的 目標變數
Reports	信用不良紀錄次數
Age	年齡
Income	年收入 (單位 : 萬)
Share	信用卡消費金額占年收入之比率
Expenditure	信用卡消費金額
Owner	申請人是否擁有房地產
Selfemp	申請人是否自行創業
Dependents	申請人扶養人數
Months	申請人登記資料中居住地址的居住時長
Majorcards	申請人目前持有信用卡數量
Active	活躍中的信用帳戶數量

1. 信用卡核准狀況 *(Card)*

信用卡核准狀況欄位為本次專案的目標欄位。
如下圖所示，可觀察到資料集的核准辦卡人數是遠遠大於不核准的人數。

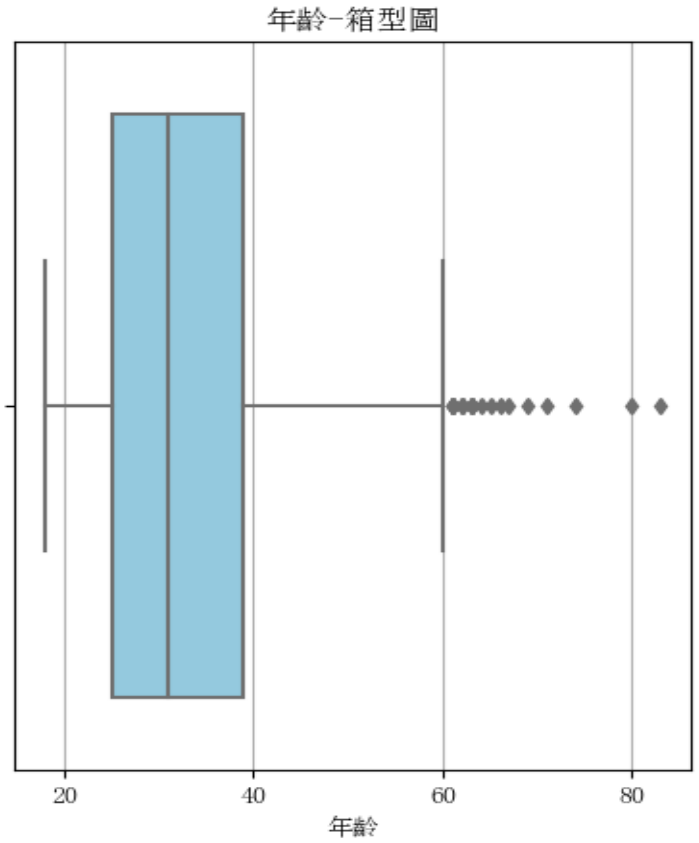


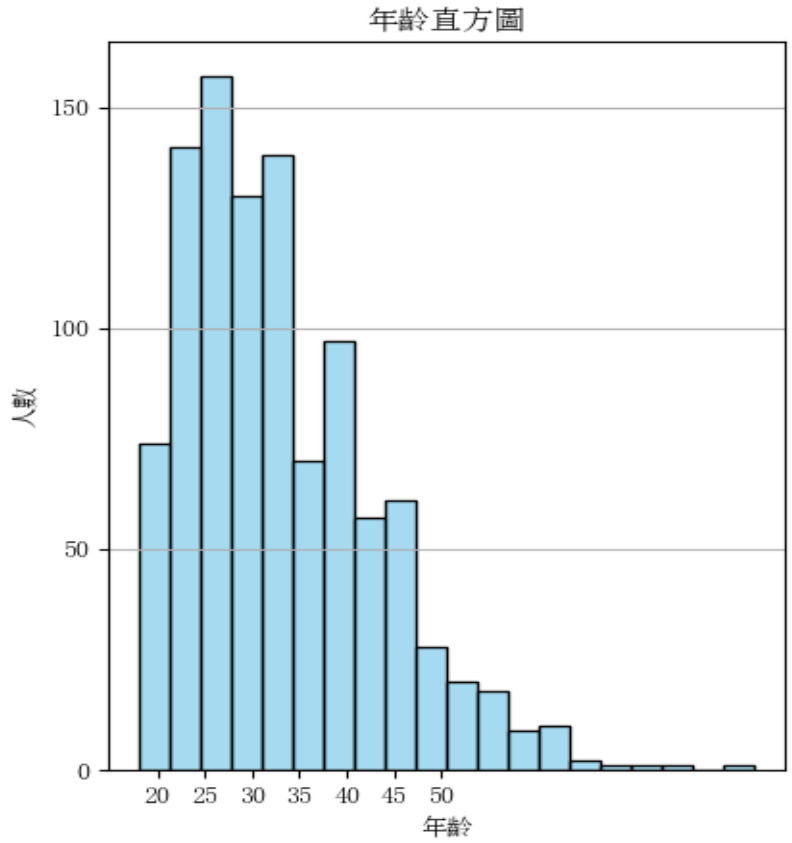
2-1. 年齡 *(Age)*

在年齡方面，如下圖所示。
可得知大部分信用卡申辦者的年齡會落在 25~34 歲之間。

❶ 平均數 : 33.38 歲　　❷眾數 : 25 歲

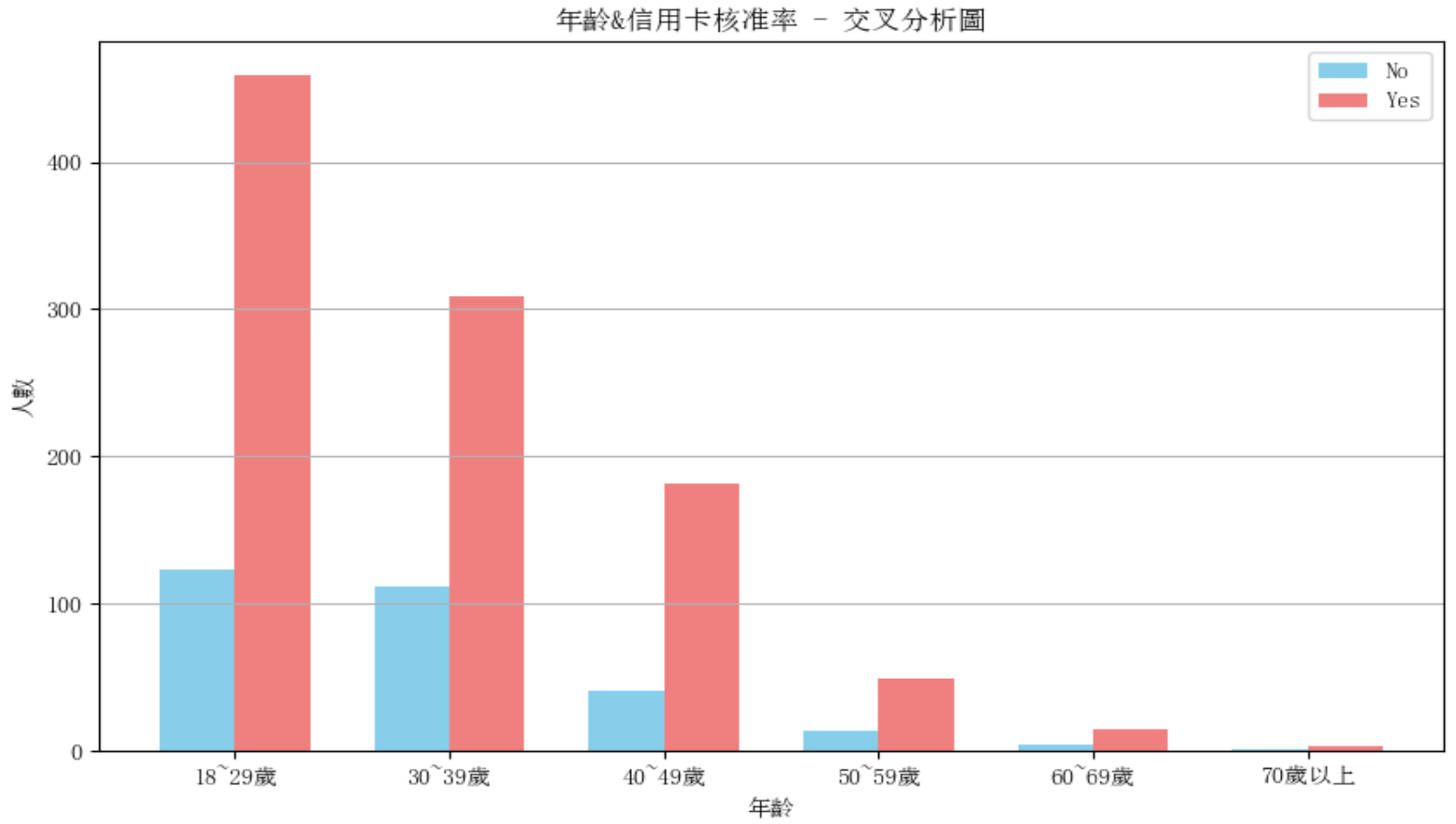
❸中位數 : 31 歲





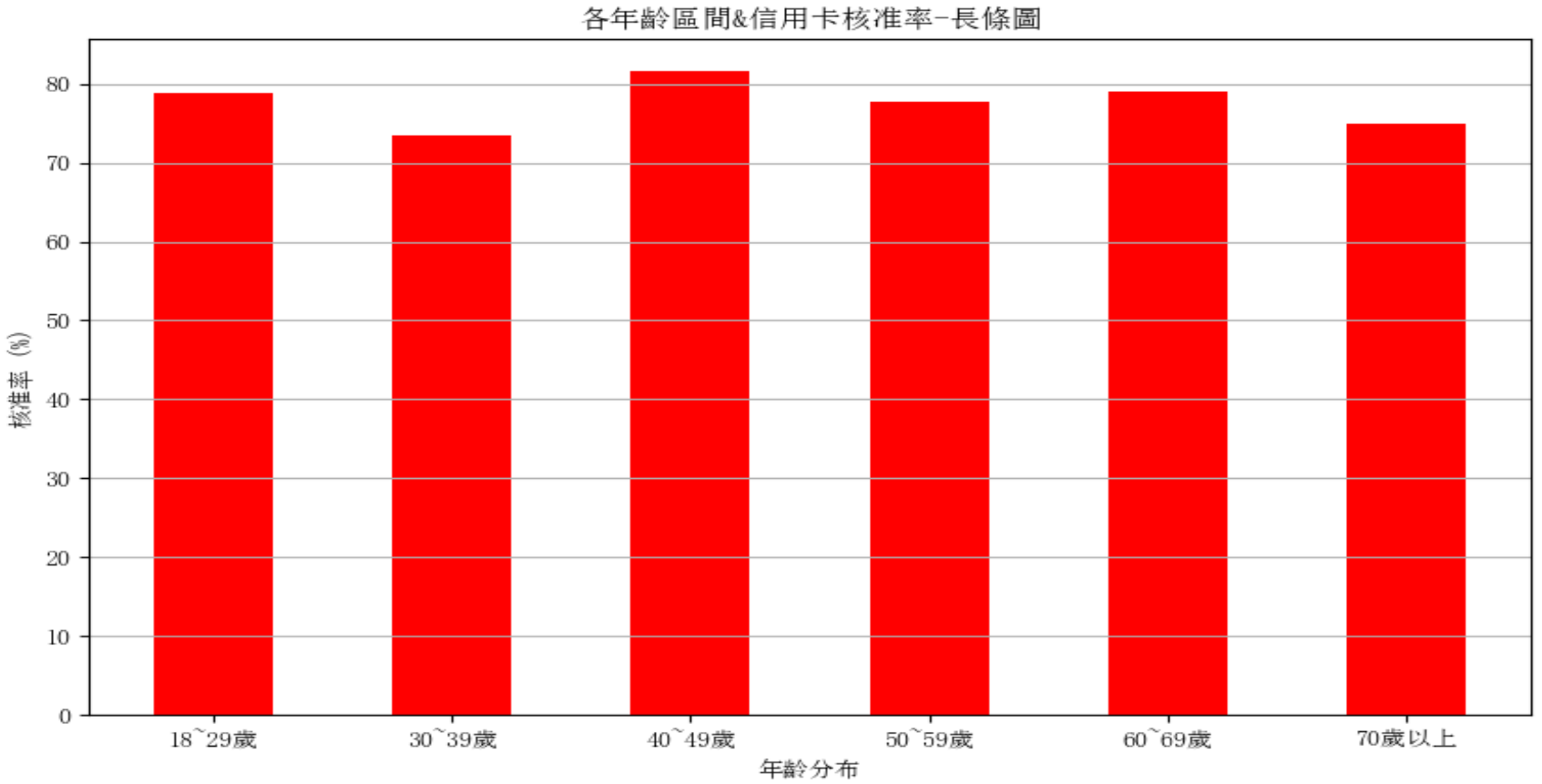
2-2. 年齡 *(Age)*

為了讓年齡與目標欄位能夠從視覺化中看出更有意義的關係。
將年齡欄位與目標欄位經過交叉分析後。
如下圖所示可得知越年輕的區間信用卡核卡數會越高，但是被拒絕發卡的次數也偏高。



2-3. 年齡 *(Age)*

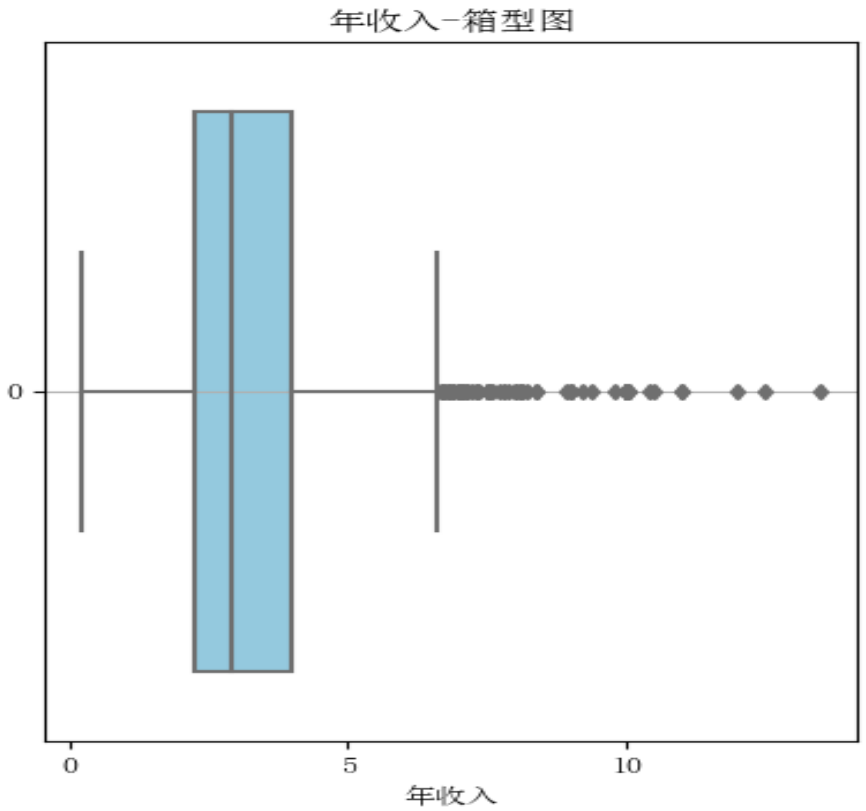
為了知道年齡跟核卡率是否有絕對的關係。
將各年齡區間的成功核卡率繪製成長條圖做比較。
如下圖所示可得知年齡與核准率並沒有呈現正向關係。
所以由此推斷年齡有較高的可能性不會是核卡通過的標準。

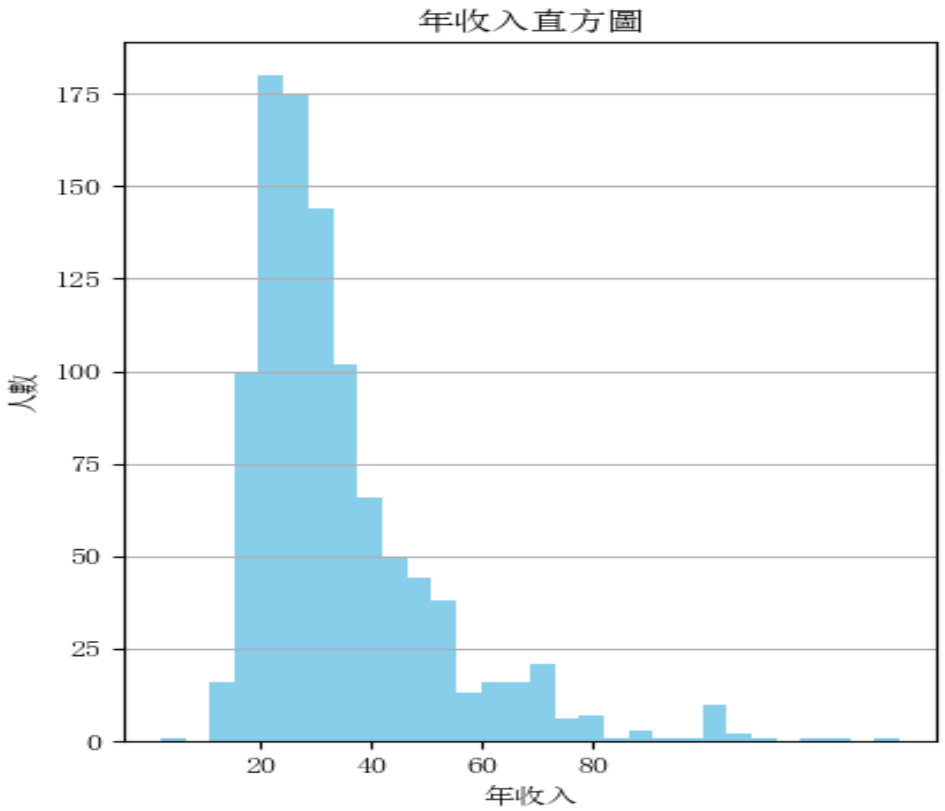


3-1. 年收入 *(income)*

在年收入方面，如下圖所示。
的數據知曉大部分的信用卡申辦者的年收入會落在 29K~33.6K 之間。

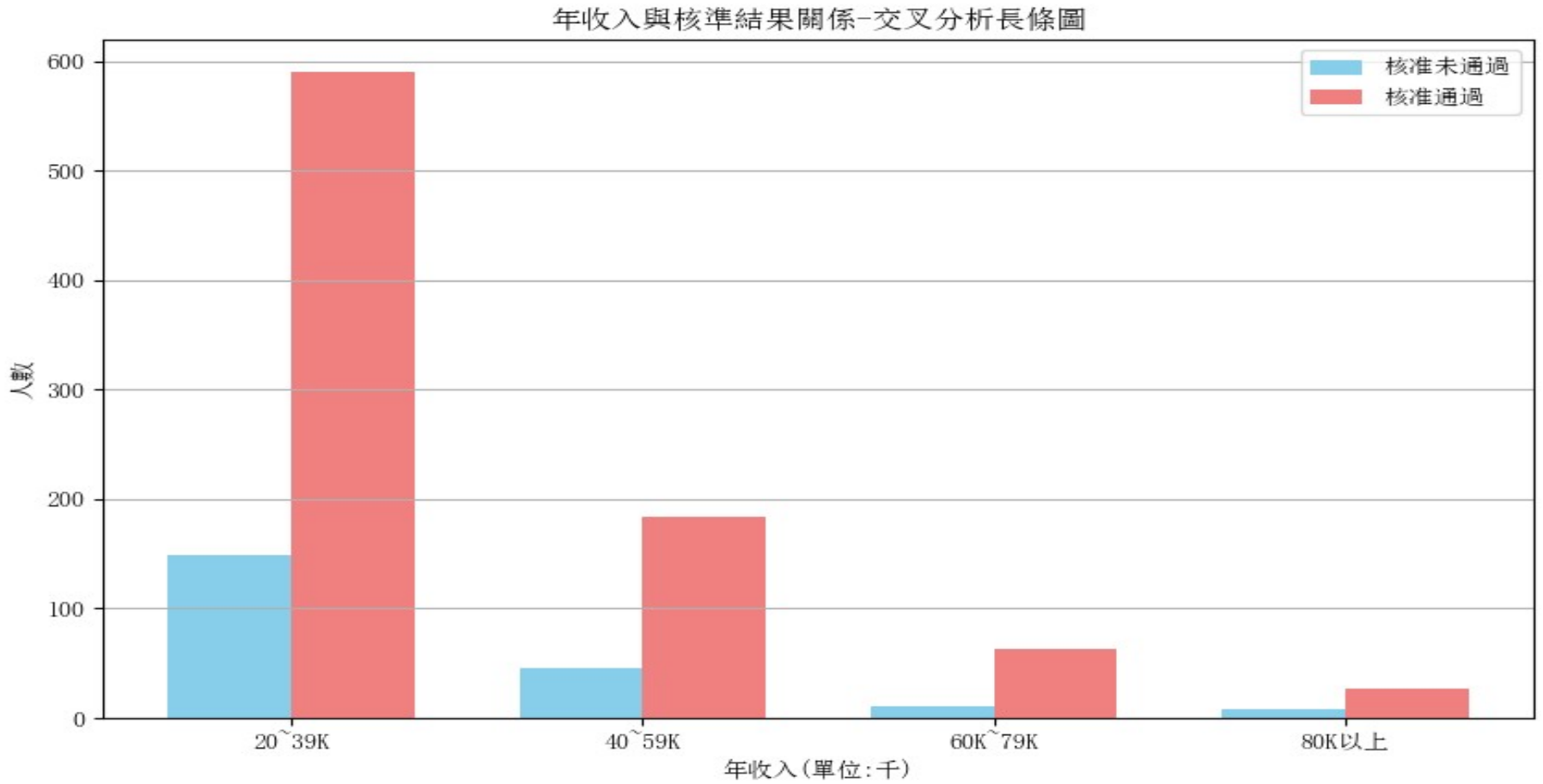
❶ 平均數 : 33.6K　　❷ 眾數 : 30K　　❸ 中位數　: 29K





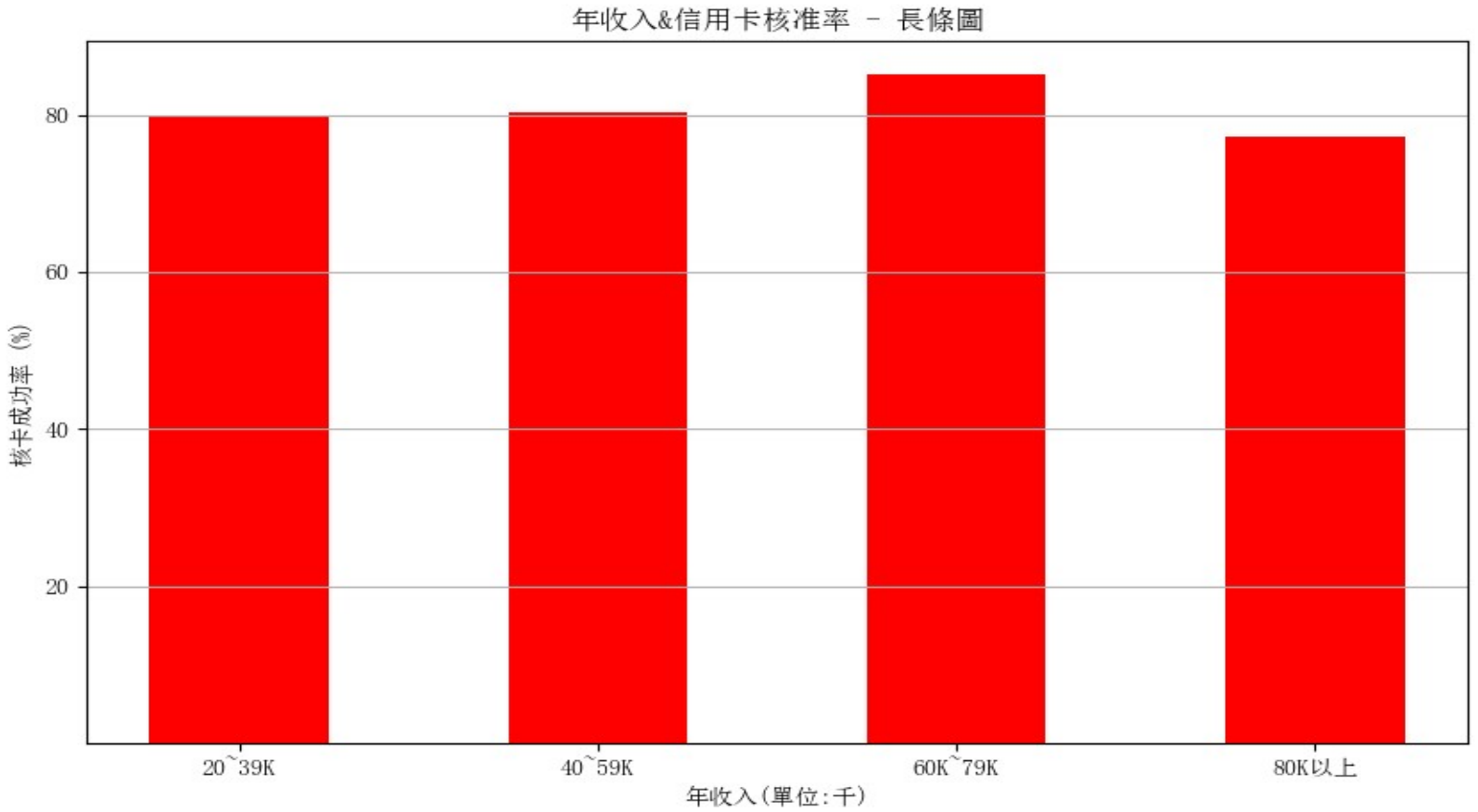
3-2. 年收入 *(income)*

為了讓年收入與目標欄位能夠從視覺化中看出更有意義的關係。
將年收入欄位與目標欄位經過交叉分析過後。
如下圖所式，可觀察到年收入介於 29K~33.6K 這個區間的核卡通過人數最高，但是拒絕核卡的人數也最高。



3-3. 年收入 *(income)*

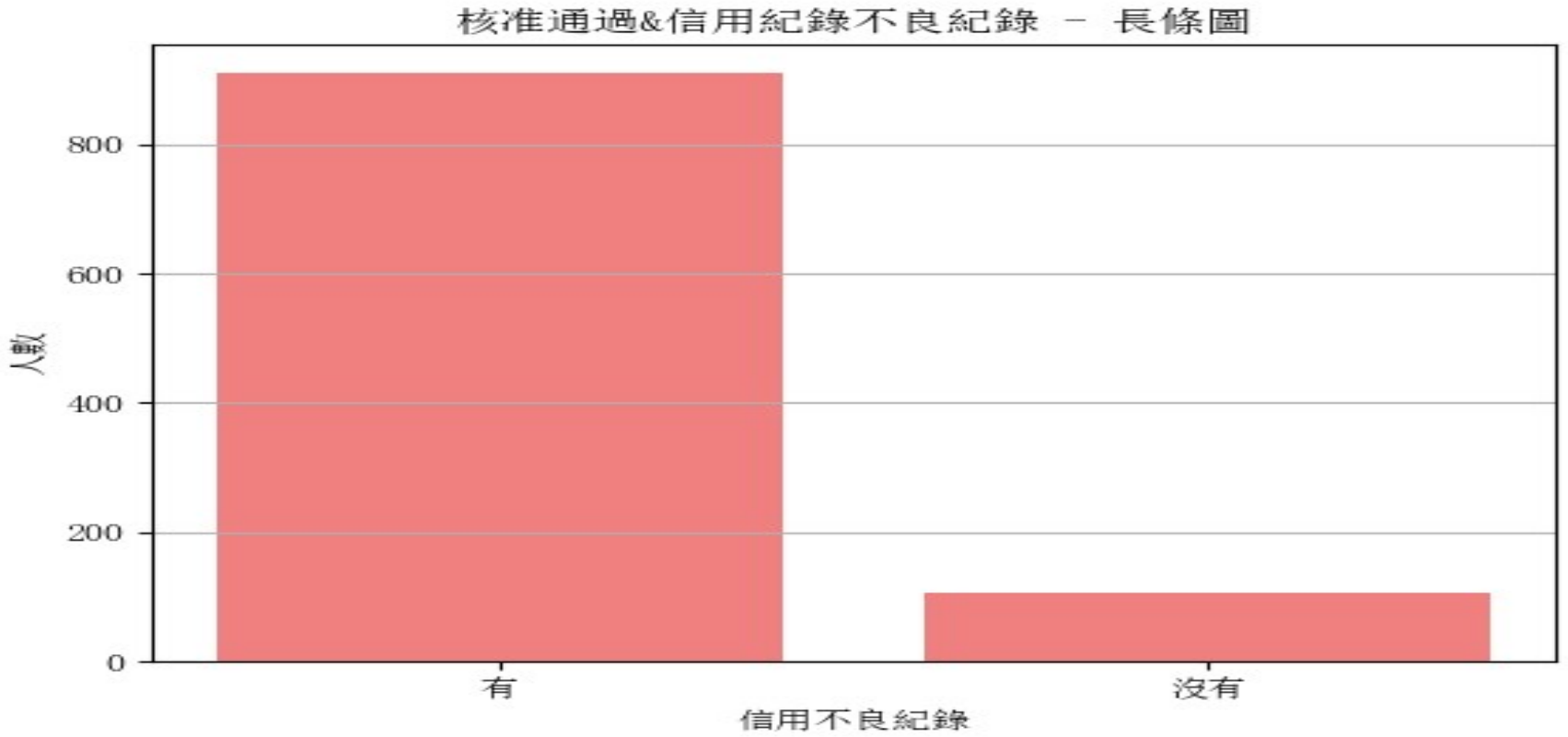
- 為了瞭解年收入跟核卡率是否有絕對的關係。
  將各年收入區間的成功核卡率繪製成長條圖做比較。
  如下圖所示，可觀察到各年收入區間與核准率並沒有呈現正向關係，所以由此我們推斷年齡收入有很高的機率不會是核卡通過的標準。



4-1. 信用不良紀錄 *(Report)*

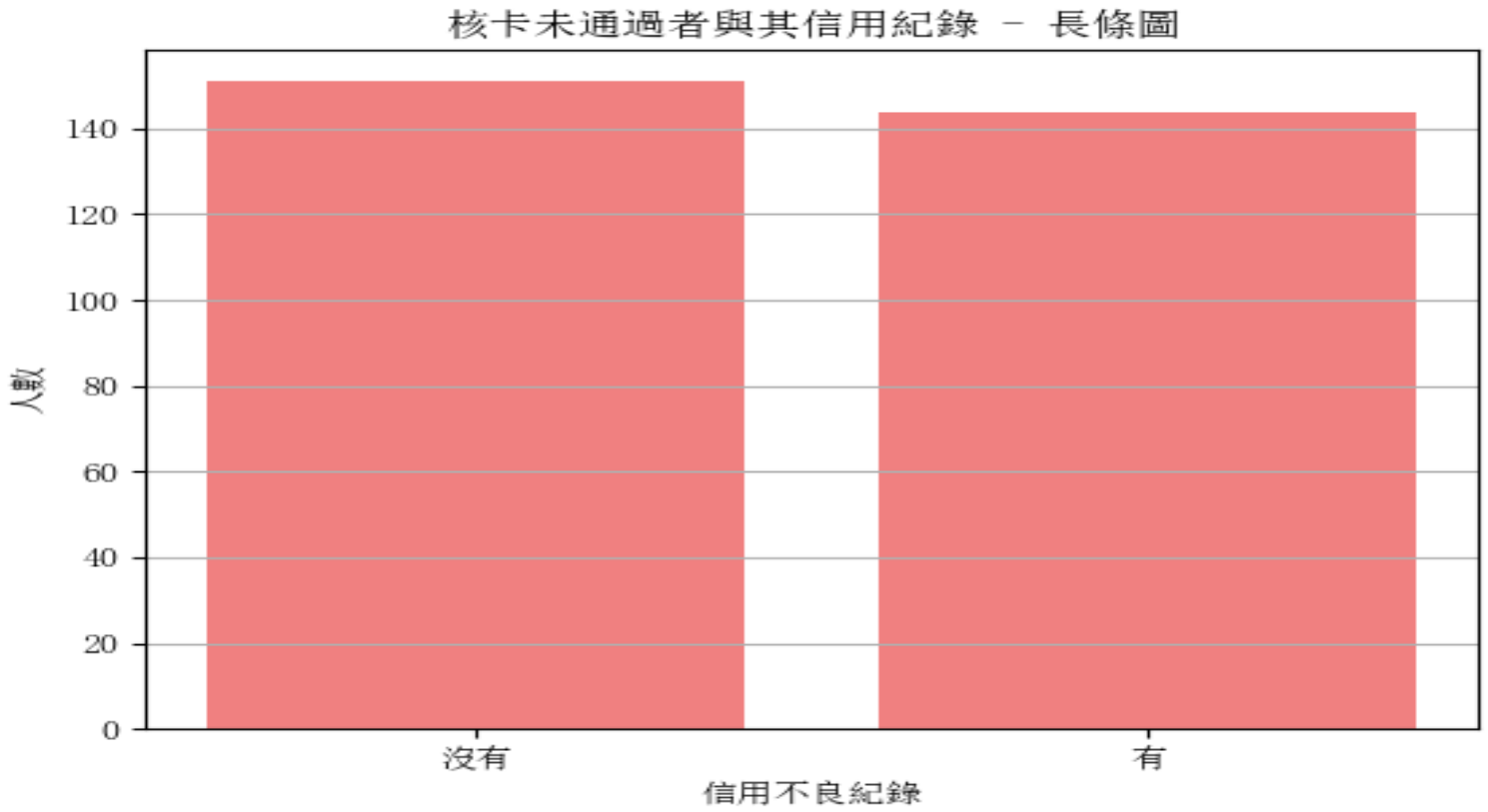
- 在信用不良紀錄方面，為了要知道通過申辦者與遭拒申辦者的信用不良紀錄是否有差別。將信用不良紀錄欄位做統計分析，如下圖所示。
- 可觀察到通過審核的申辦者中有 910 人有信用不良紀錄， 107 人沒有信用不良紀錄，可由此推斷就算申辦者有不良紀錄還是可以通過信用卡審核。

❶ 有信用不良紀錄者 : 910 人　　❷沒有信用不良紀錄者 : 107 人



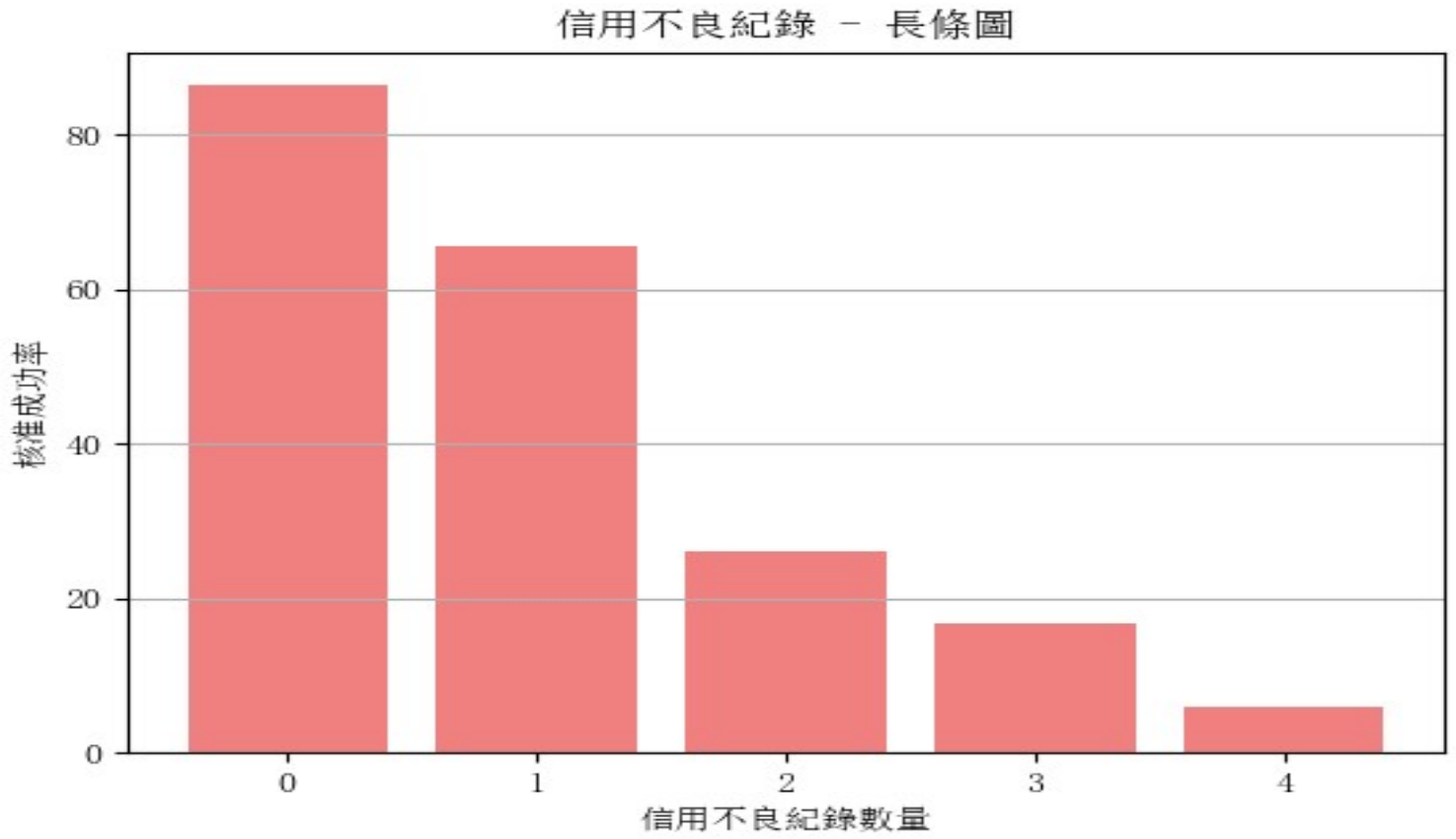
4-2. 信用不良紀錄 *(Report)*

- 在核卡遭拒方面，如下圖所示。
- 可觀察到核卡遭拒的申辦者中有 151 人沒有信用不良紀錄，由此可推斷申辦者是否有信用不良紀錄都有機會通過申請。



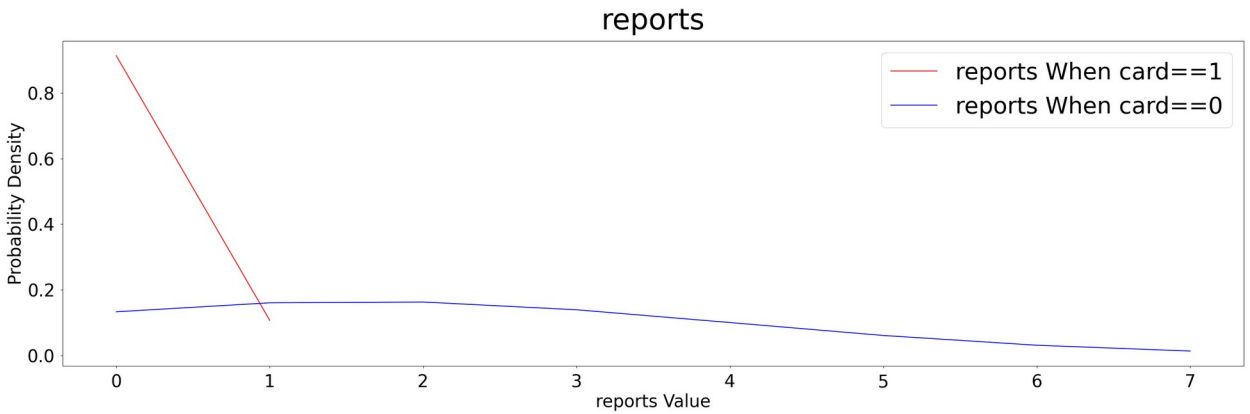
4-3. 信用不良紀錄 *(Report)*

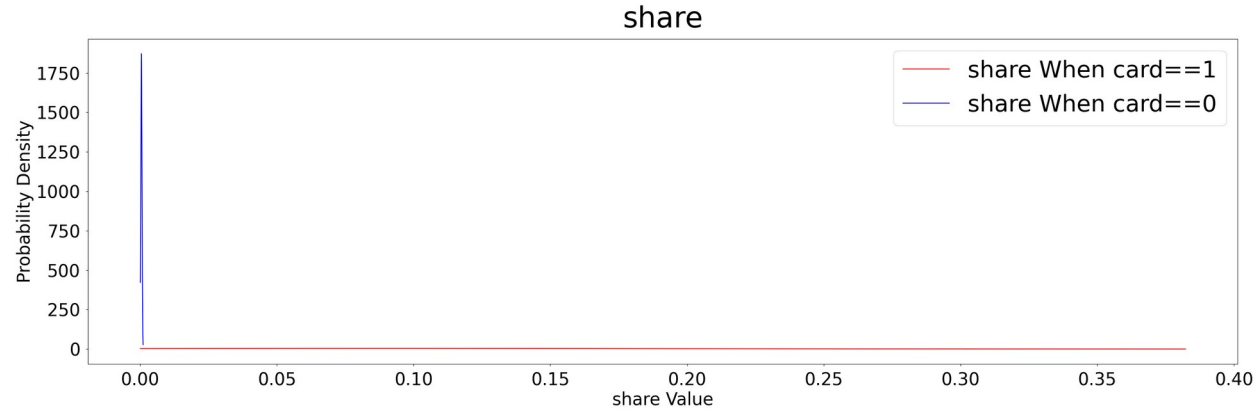
為了讓信用不良紀錄欄位與目標欄位能夠從視覺化中看出更有意義的關係。
將信用不良紀錄欄位與目標欄位經過交叉分析後，如下圖所示。
可得知信用不良紀錄在越少的情況下，申辦者通過信用審核的機會比較高。

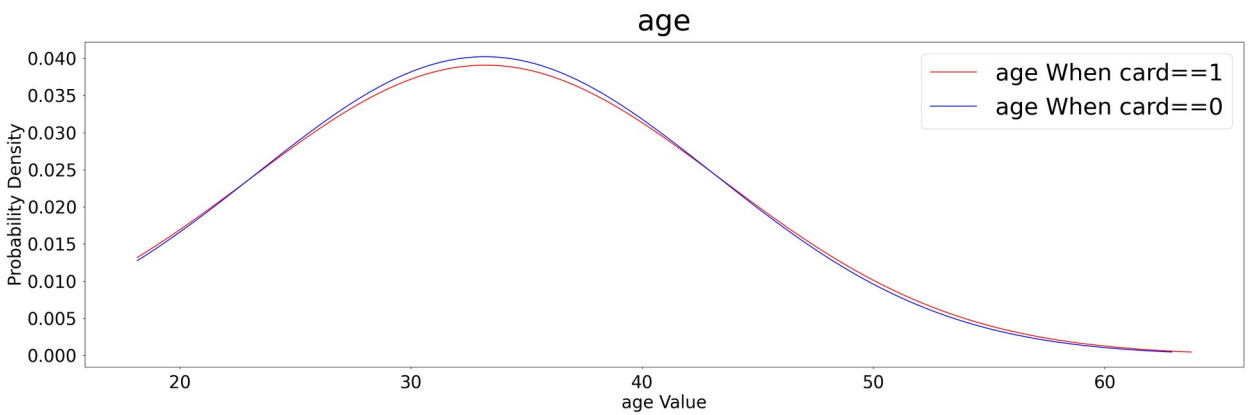


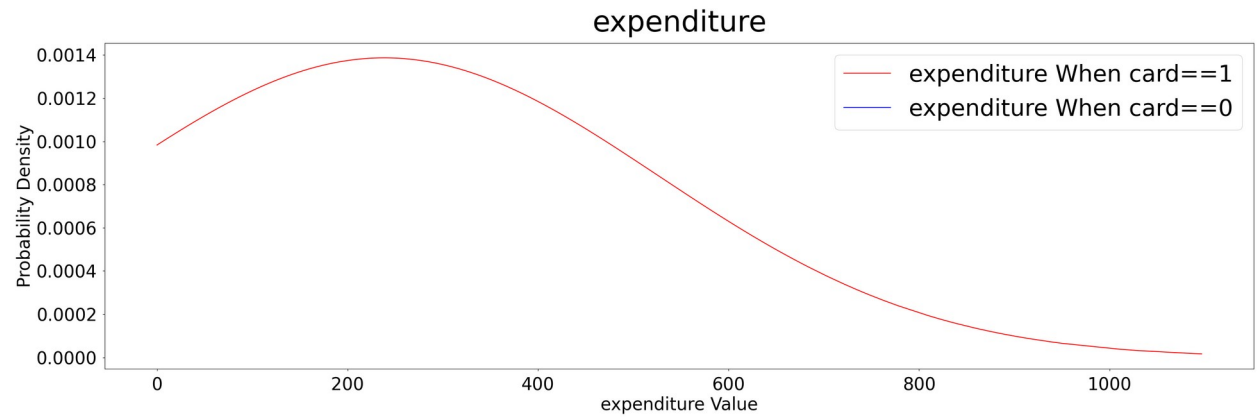
特徵分析 – 概率分布曲線 (取值範圍根據三西格瑪原則)

分布密度函數 =(1/std*(2*π)**0.5)*np.exp(-0.5*(value-mean)/std)**2)

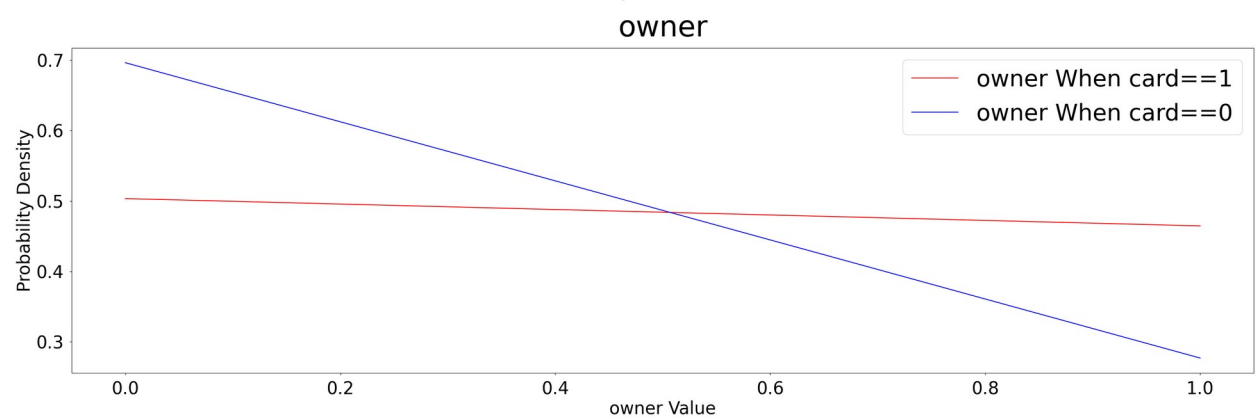

reports
reports When card==1
reports When card==0
Probability Density
reports Value

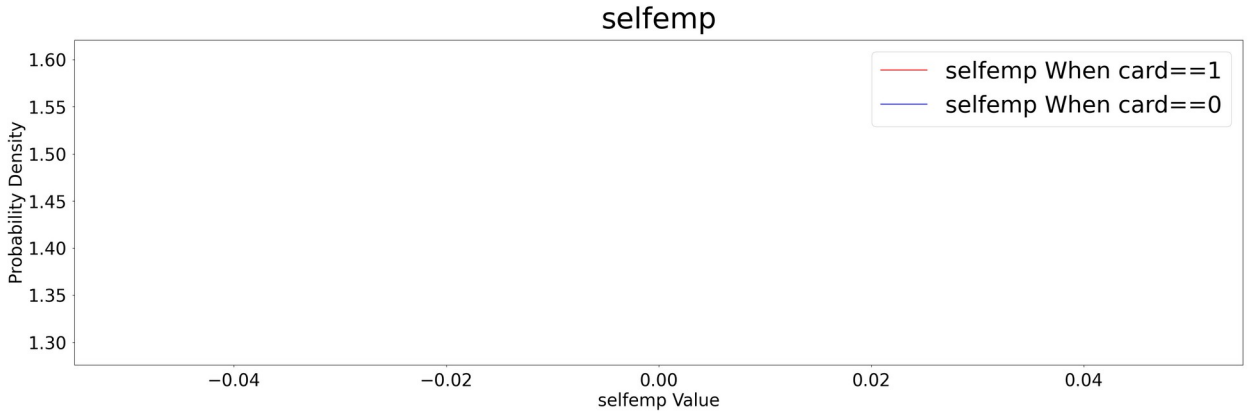

share
share When card==1
share When card==0
Probability Density
share Value

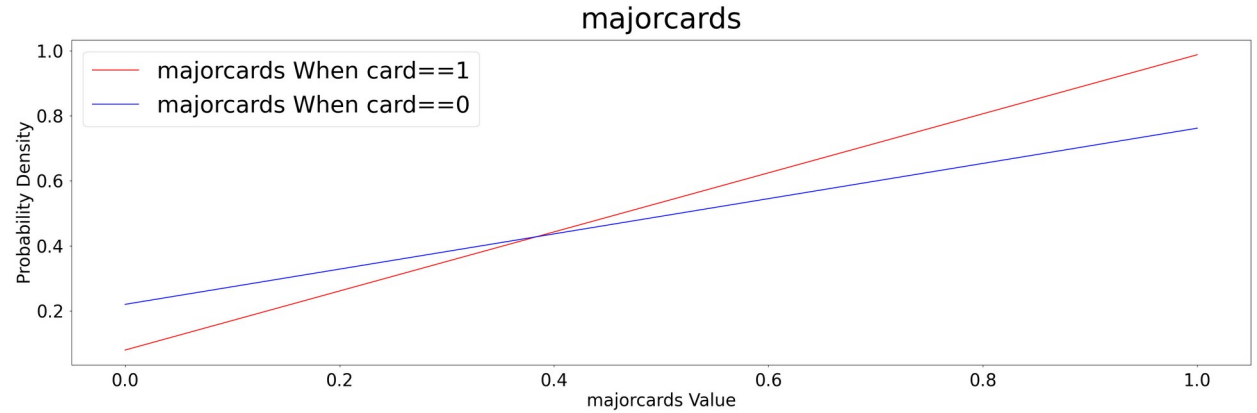

age
age When card==1
age When card==0
Probability Density
age Value

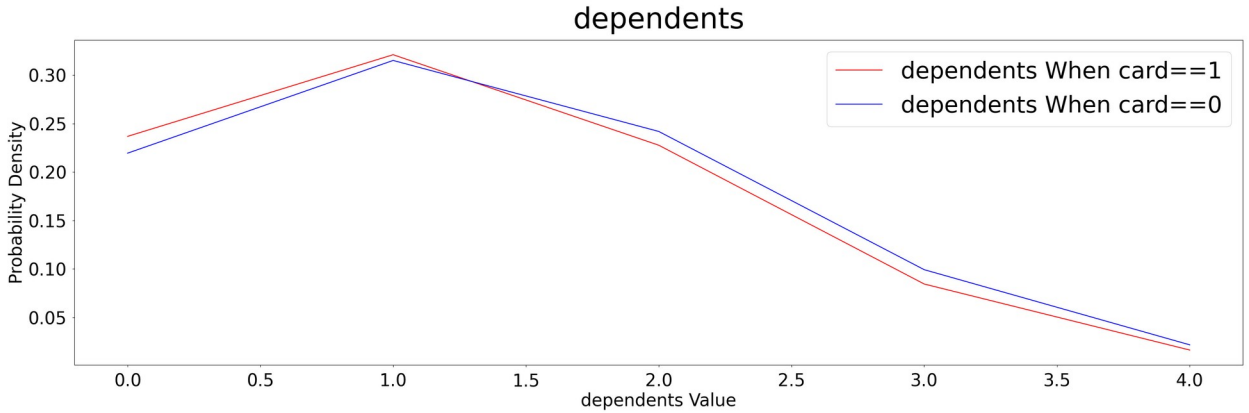

expenditure
expenditure When card==1
expenditure When card==0
Probability Density
expenditure Value

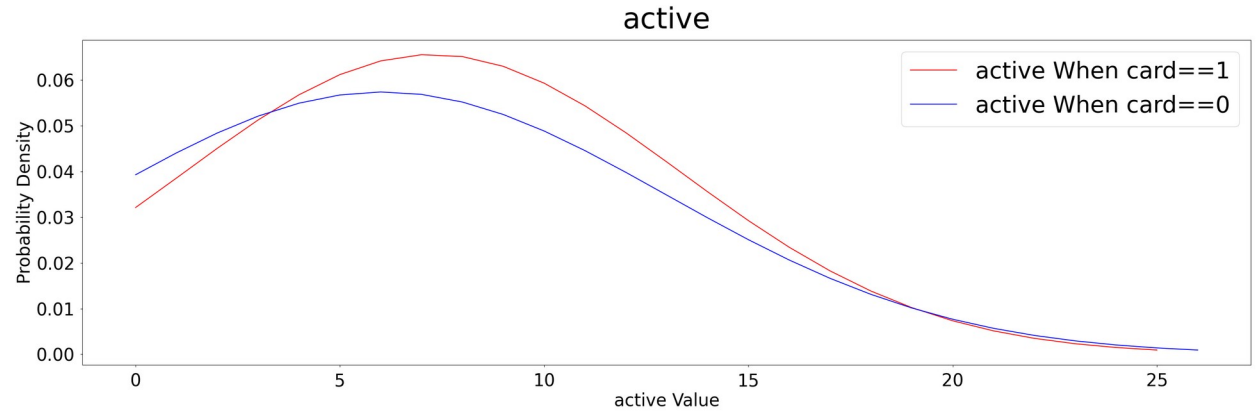

income
income When card==1
income When card==0
Probability Density
income Value

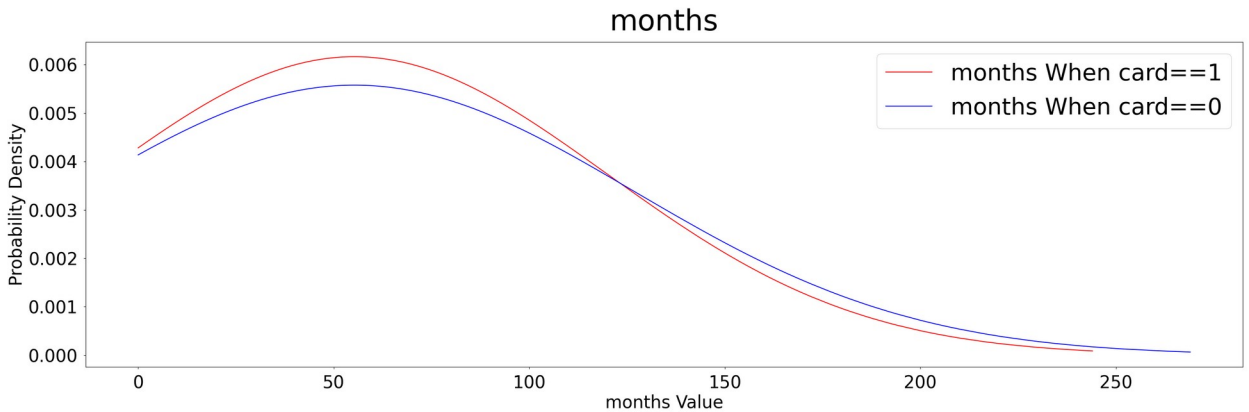

owner
owner When card==1
owner When card==0
Probability Density
owner Value


selfemp
selfemp When card==1
selfemp When card==0
Probability Density
selfemp Value


majorcards
majorcards When card==1
majorcards When card==0
Probability Density
majorcards Value


dependents
dependents When card==1
dependents When card==0
Probability Density
dependents Value


active
active When card==1
active When card==0
Probability Density
active Value


months
months When card==1
months When card==0
Probability Density
months Value

申辦者特徵

最後，根據上方不斷地分析數據，大概可歸納出會成功核准的信用卡申辦者特徵：

1. 信用不良紀錄：　4 筆以下
2. 年齡：青壯年族群
3. 年收入：　33,000 上下
4. 信用卡消費金額 : 大於 5 元
5. 信用卡支出占比 : 大於 1%
6. 是否擁有房地產 : 有
7. 是否非創業人士 : 否
8. 扶養人數 : 0~1 人
9. 居住時間 : 1~5 年
10. 主卡數量 : 擁有 1 張以上信用卡
11. 活躍信用帳戶數量 : 7 個上下

特徵工程：

整體無缺失值

	欄位說明
Card	離散型資料 ; 為目標欄位修改字串內容成數執行資料 {“yes”:1, “no”:0}
Reports	連續型資料 ; z-score 正規化
Age	連續型資料 ; 刪除不合理數據 ex. 年齡 0 歲且有年薪
Income	連續型資料 ; 天花板地板法刪除更改離群值、 z-score 正規化
Share	連續型資料 ; 天花板地板法刪除更改離群值、 z-score 正規化
Expenditure	連續型資料 ; 天花板地板法刪除更改離群值、 z-score 正規化
Owner	離散型資料 ; OneHotEncoding
Selfemp	離散型資料 ; OneHotEncoding
Dependents	連續型資料 ; z-score 正規化
Months	連續型資料 ;
Majorcards	離散型資料 ; 以當前資料沒有 0/1 以上的值，轉換擁有 / 非擁有
Active	連續型資料 ;

選擇模型：

決策樹 / 高斯貝葉斯分類 / 柏努力貝葉斯分類 / 邏輯迴歸・・・等

公式準備 - 邏輯回歸

Sigmoid = 1/(1+ np.exp(-(Θ0+Θ1*W1+Θ2*W2))# 要做二元預測的 (0~1)

Cost = 1/2Σ(y_true - y_pre)^2 # 計算差距的

w = w + LearningRate*Σ(y_true – y_pre)x # 更新的截距加總

訓練模型 - 邏輯回歸

```
w: [ 4.41141026  4.14620362  1.72514216 19.97404803 -2.35413003  3.0282689
  1.05314094  0.92026118 -0.56784273 -3.2003988   2.29294845 -0.34026095
 -0.57296248  0.78822294 -1.51451922 -1.29674422  2.04959389  0.10885131
  3.64988005 -2.8218259   2.53854015]
b: -0.04623181115712358
```

- `data_numeric = pd.read_csv("Data/Data1.csv")`
- `data_numeric = data_numeric.astype(float)`
- `features = data_numeric.drop(['核卡狀況'], axis=1)`
- `y_true = data_numeric['核卡狀況']`
- `X_train, X_test, y_train, y_test = train_test_split(features, y_true, test_size=0.2)`

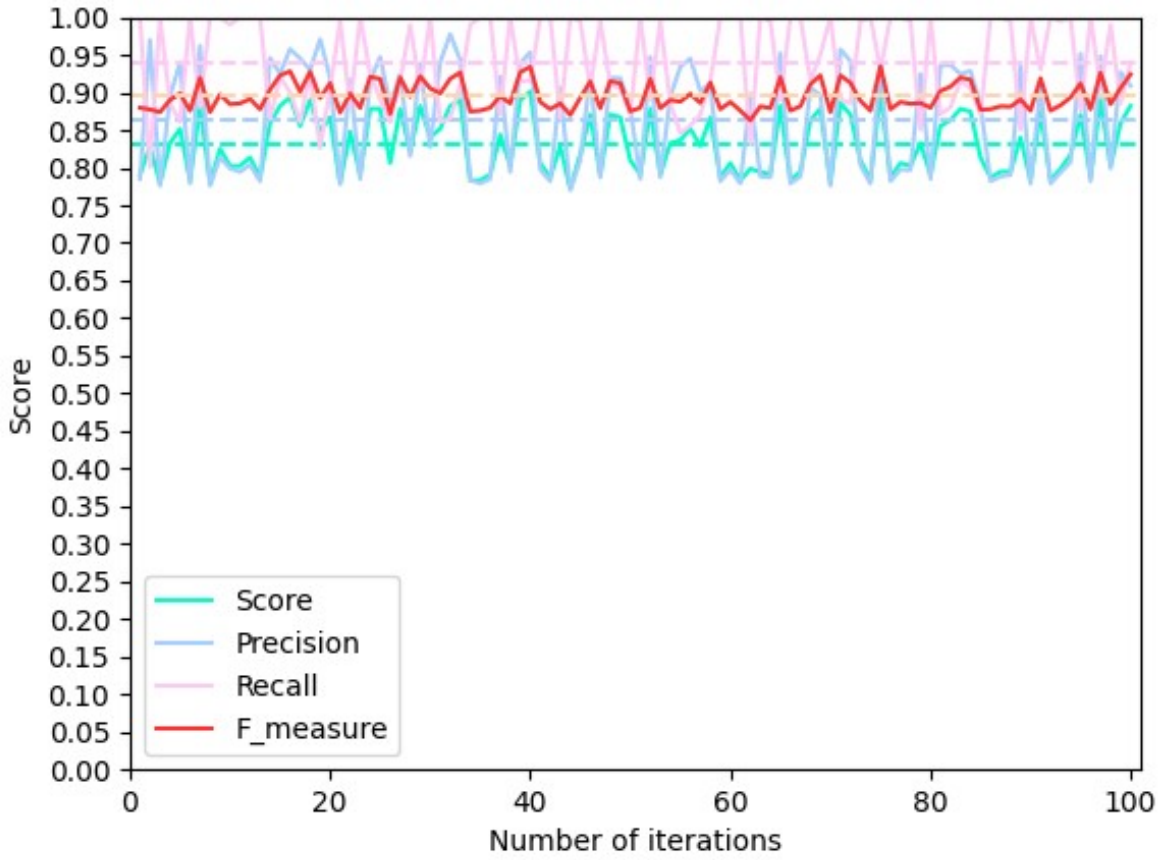
模型自：《Python_Machine_Learning》 book by Sebastian Raschka 、 Vahid Mirjalili

- `GR = LogisticR(learning_rate=0.020999999999999998, iter=730)`
- `GR.fit(X_train, y_train)# 擬合與訓練`
- `pre = GR.predict(X_test)# 預測模型分數`
- `matrix(y_test, pre)# 以混淆矩陣分析預測好壞`

```
parameter={"learning_rate":[i for i in np.arange(0.001,0.1,0.01)], "iter":[i for i in
np.arange(100,1000)]}
gs = GridSearch(LogisticR, "LogisticR", parameter)
gs.fit(X_train,y_train, X_test, y_test)
```

GridSearch : https://github.com/RCK10274/Issue_Credit_Card/blob/OAO/GridSearch.py

分析結果



First Try

分數浮動大但整體平均皆為0.80~0.95

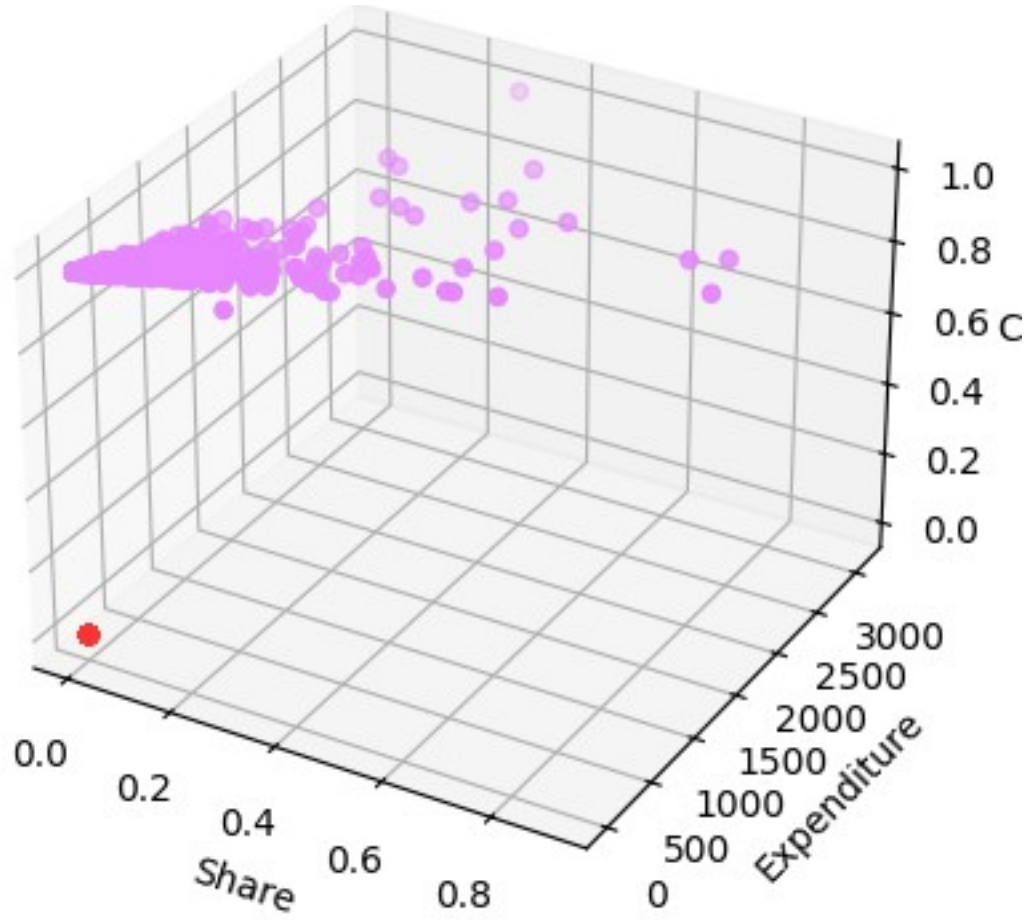
	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.830342	0.863455	0.939602	0.895262
2	std	0.039380	0.072544	0.059727	0.018757
3	min	0.771863	0.770115	0.800995	0.863049
4	25%	0.790875	0.787402	0.885572	0.879121
5	50%	0.826996	0.889155	0.930348	0.889444
6	75%	0.870722	0.934156	1.000000	0.912903
7	max	0.901141	0.977528	1.000000	0.935000

有正相關的兩組特徵：收支比 [share] 、每月信用卡支出平均 [expenditure]

```
print(feature_df1.iloc[:, 3][target1==1].shape[0])
print(feature_df1.iloc[:, 3][target1==0].shape[0])
print(feature_df1.iloc[:, 4][target1==1].shape[0])
print(feature_df1.iloc[:, 4][target1==0].shape[0])
```

✓ 0.0s

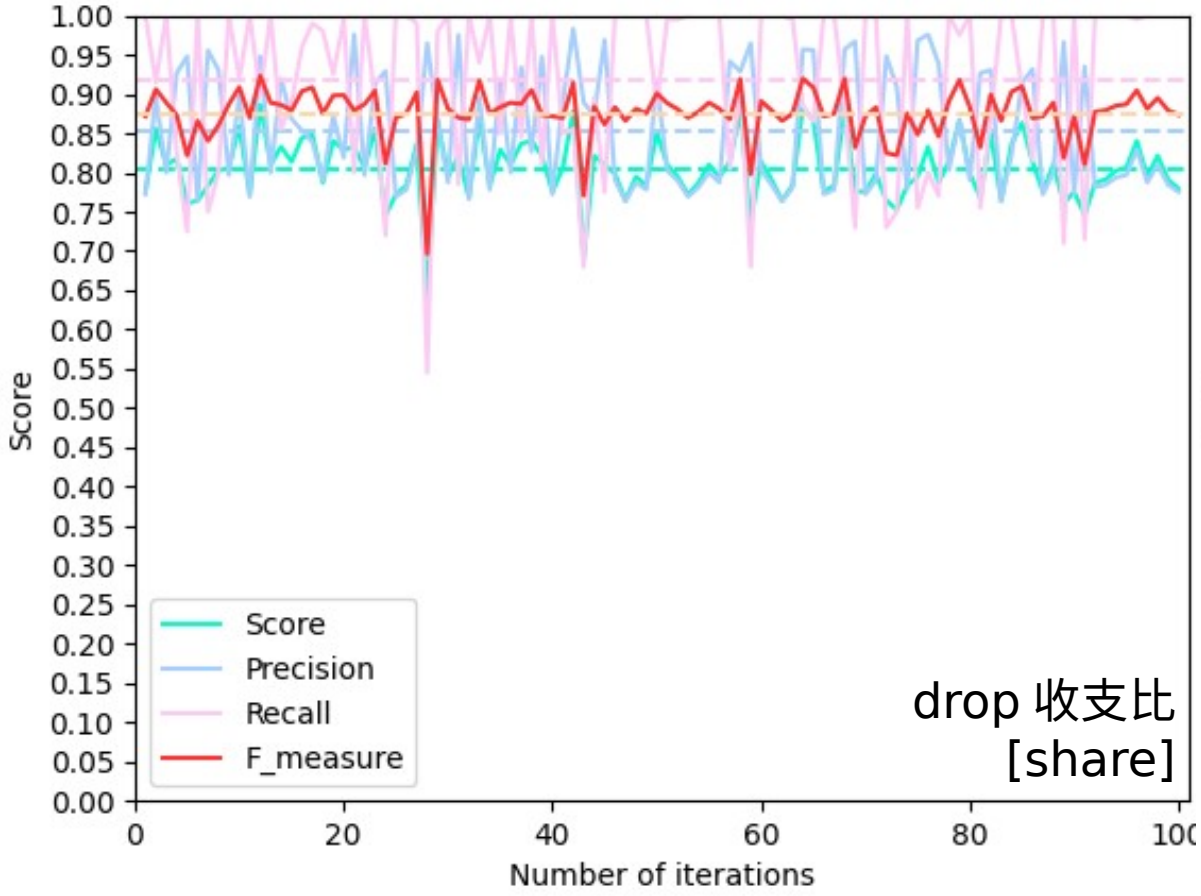
```
1017
295
1017
295
```

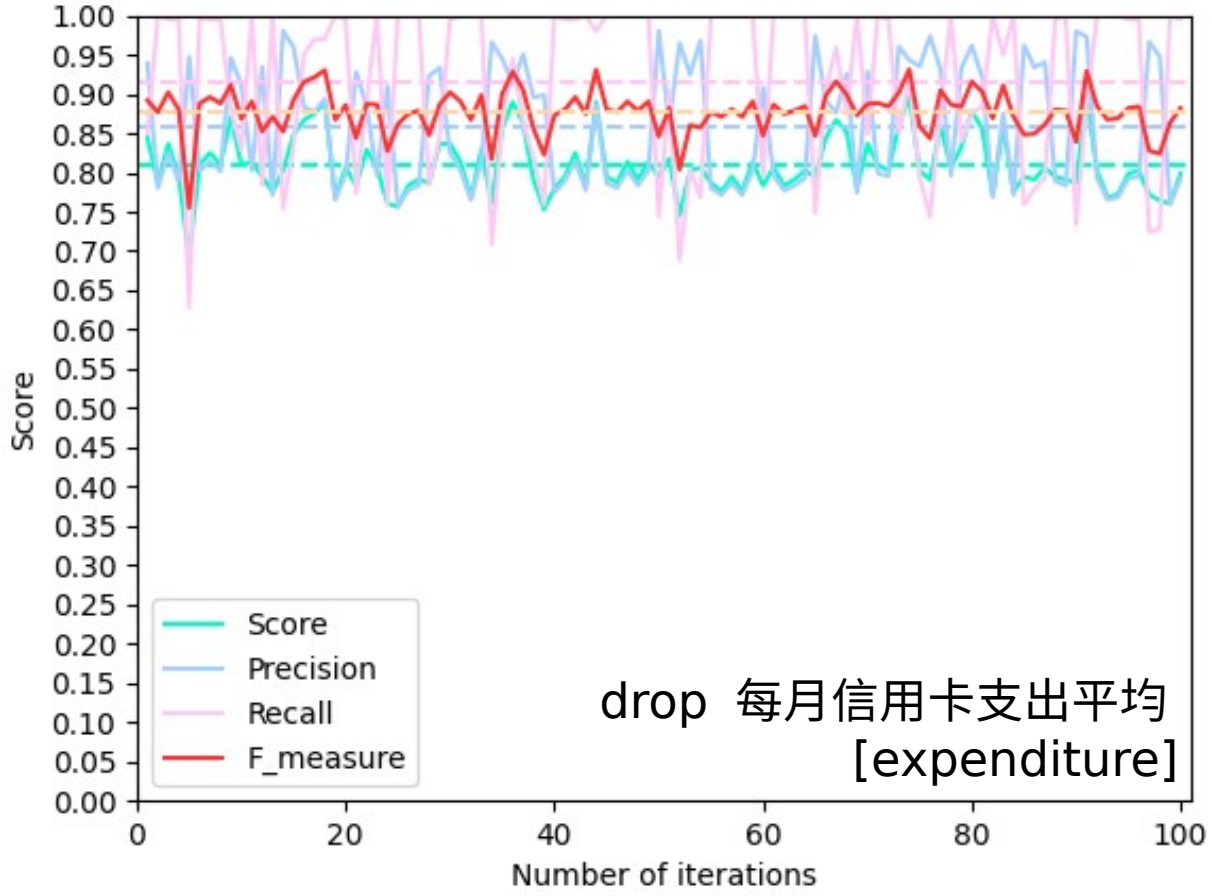


Second Try

	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.804449	0.853248	0.918200	0.875693
2	std	0.040429	0.075241	0.107072	0.032933
3	min	0.638783	0.763359	0.545000	0.696486
4	25%	0.778517	0.784314	0.850000	0.869565
5	50%	0.802281	0.824746	0.995000	0.879121
6	75%	0.832700	0.929332	1.000000	0.890869
7	max	0.885932	0.982759	1.000000	0.923077

	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.808365	0.858291	0.915427	0.877362
2	std	0.037470	0.076432	0.102967	0.028358
3	min	0.692015	0.756654	0.628141	0.755287
4	25%	0.786122	0.788274	0.837940	0.866162
5	50%	0.802281	0.825000	0.994975	0.880265
6	75%	0.829848	0.933434	1.000000	0.890740
7	max	0.897338	0.980392	1.000000	0.931298





分析優化：數值型資料分箱

修改內容

card

Target 目標欄位

income

連續數值型特徵

share

連續數值型特徵

age_18~30/age_30~50/age_50~

年齡分 3 類 18~30 30~50 50~

owner_no/owner_yes

有無房地產 one_hot

selfemp_no/selfemp_yes

有無創業 one_hot

majorcards_0/majorcards_>=1

持有的主卡是大於等於 1 、或是小於 1

reports_equal and greater then 1 /reports_less than 1

紀錄＞ 1 與紀錄小於 1

dependents_0~1/dependents_1~

分有無撫養人

months_60~/months_~50/months_50～60

居住時間分 3 類

active_<7/active_>=7

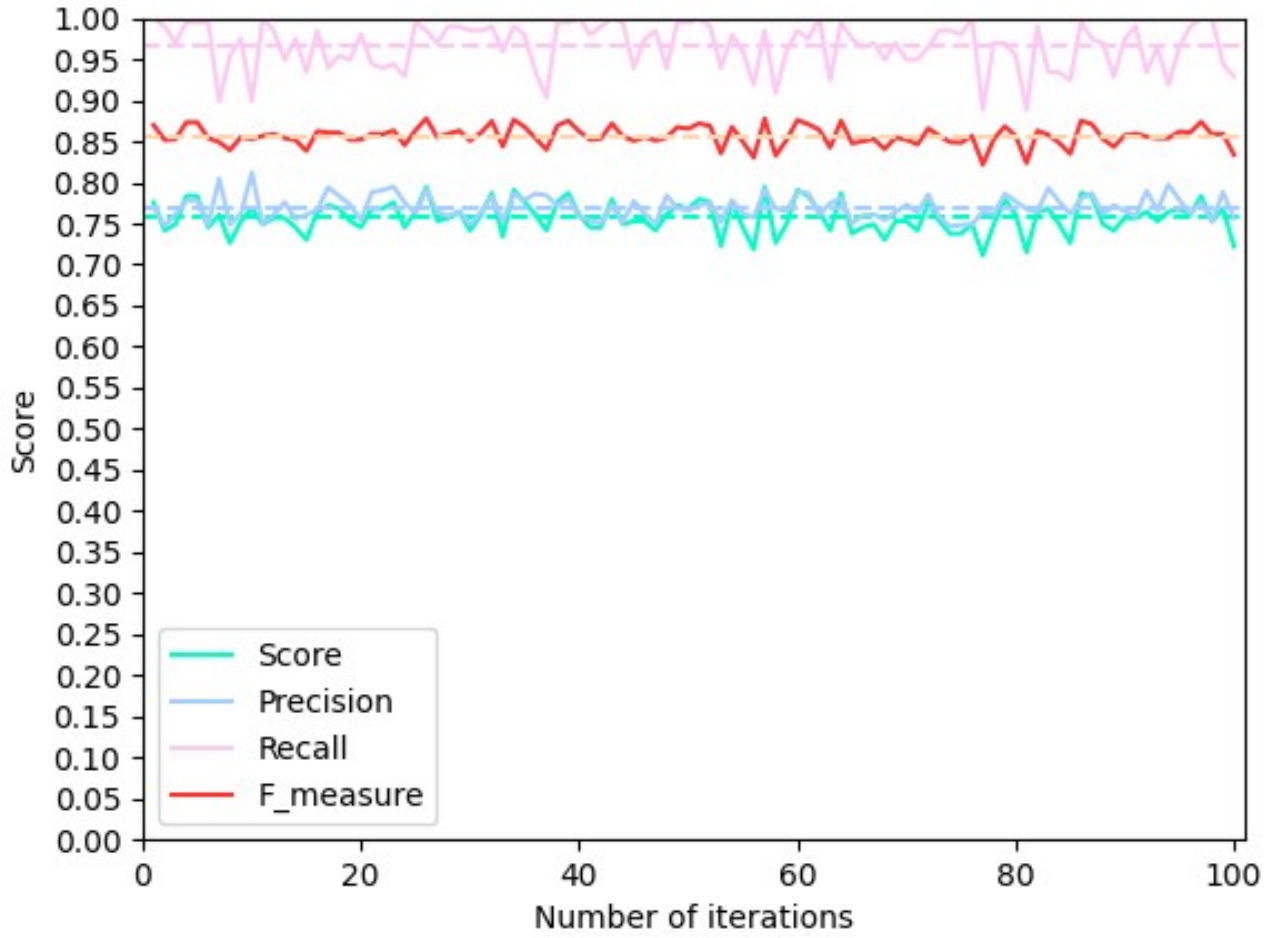
活躍帳戶 one_hot_endcoding 成大於等 7 或小於 7

分箱特定數值特徵後

Third Try

有變穩定的趨勢，但預測 True 的正確率比 False 高太多了。

	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.757452	0.769655	0.965990	0.856367
2	std	0.018871	0.014624	0.028954	0.011910
3	min	0.711027	0.746154	0.888325	0.821596
4	25%	0.745247	0.759689	0.944162	0.850877
5	50%	0.756654	0.766464	0.974619	0.855855
6	75%	0.769011	0.780876	0.989848	0.863586
7	max	0.794677	0.811927	1.000000	0.877828



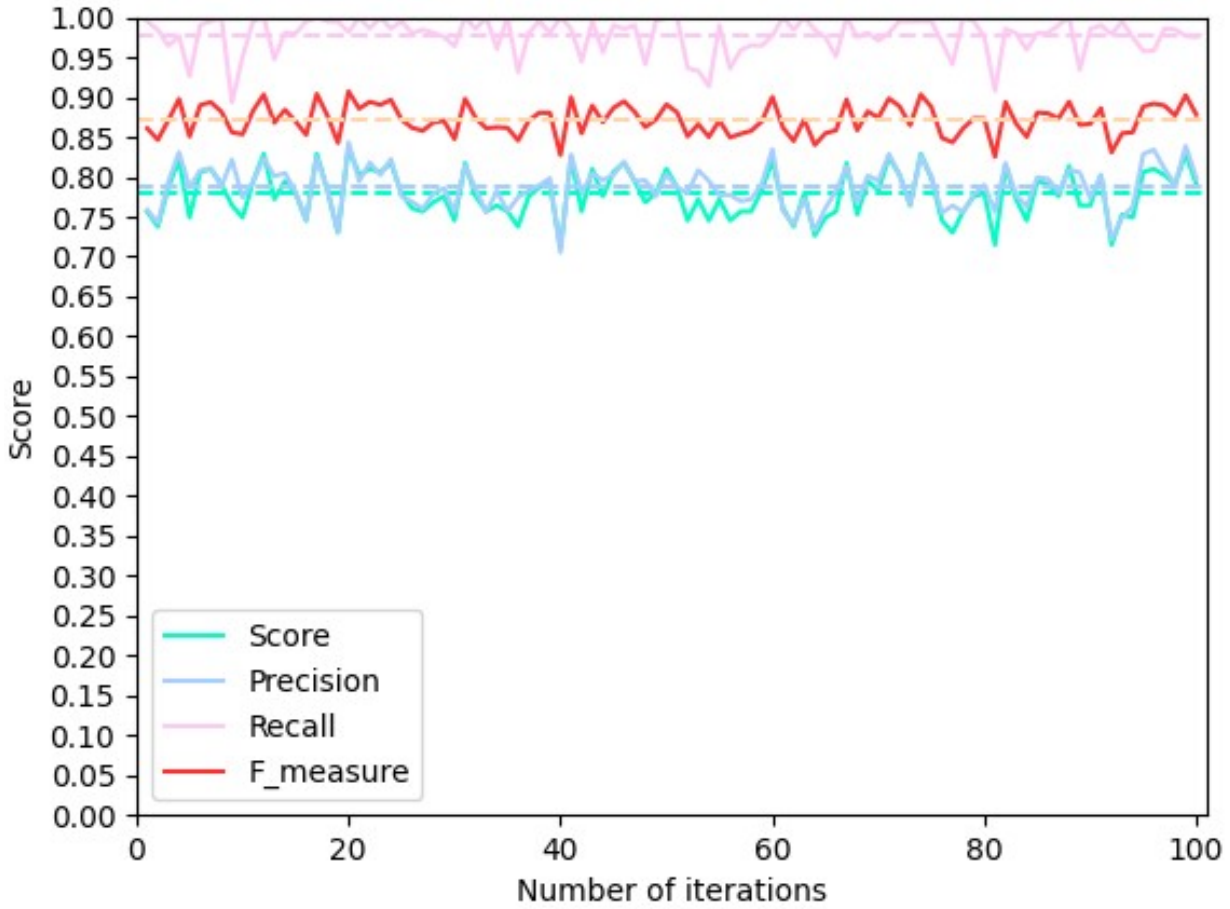
再分箱－年收入［income］：

- 將　income 分箱為　>=4 與　<4

Fourth Try

似乎跌代的起伏更高了

	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.778479	0.787457	0.977283	0.871769
2	std	0.029663	0.027195	0.022971	0.018902
3	min	0.711027	0.705426	0.893204	0.825175
4	25%	0.756654	0.771387	0.968054	0.857778
5	50%	0.777567	0.788867	0.984334	0.871325
6	75%	0.802281	0.804172	0.994994	0.887706
7	max	0.836502	0.843373	1.000000	0.907127



檢討原因

樣本不足：

資料總比數為 1312 筆資料，目標欄位 True 與 False 占比約 4 ： 1 ，False 筆數少於 True 太多，在計算 False 上沒有足夠的數據分析。

解決方法：

再找一組資料能夠補上 False 的資料

DataSet-credit_Card

- https://www.kaggle.com/datasets/vishwas199728/credit-card/data

串接重疊特徵

聯徵紀錄 [reports] > 收支比 [share] > 每月信用卡支出平均 [expenditure] > 房地產 [owner] > 持有主卡 [majorcards] > 年收入 [income] >

活躍帳戶 [active] > 創業人士 [selfemp] > 扶養人數 [dependents] > 年齡 [age] > 居住時間 [months]

Credit_Card 資料重疊特徵：

［Annual_income］=>［income］

［Birthday_count］=>［age］

［Family_Members］=>［dependents］

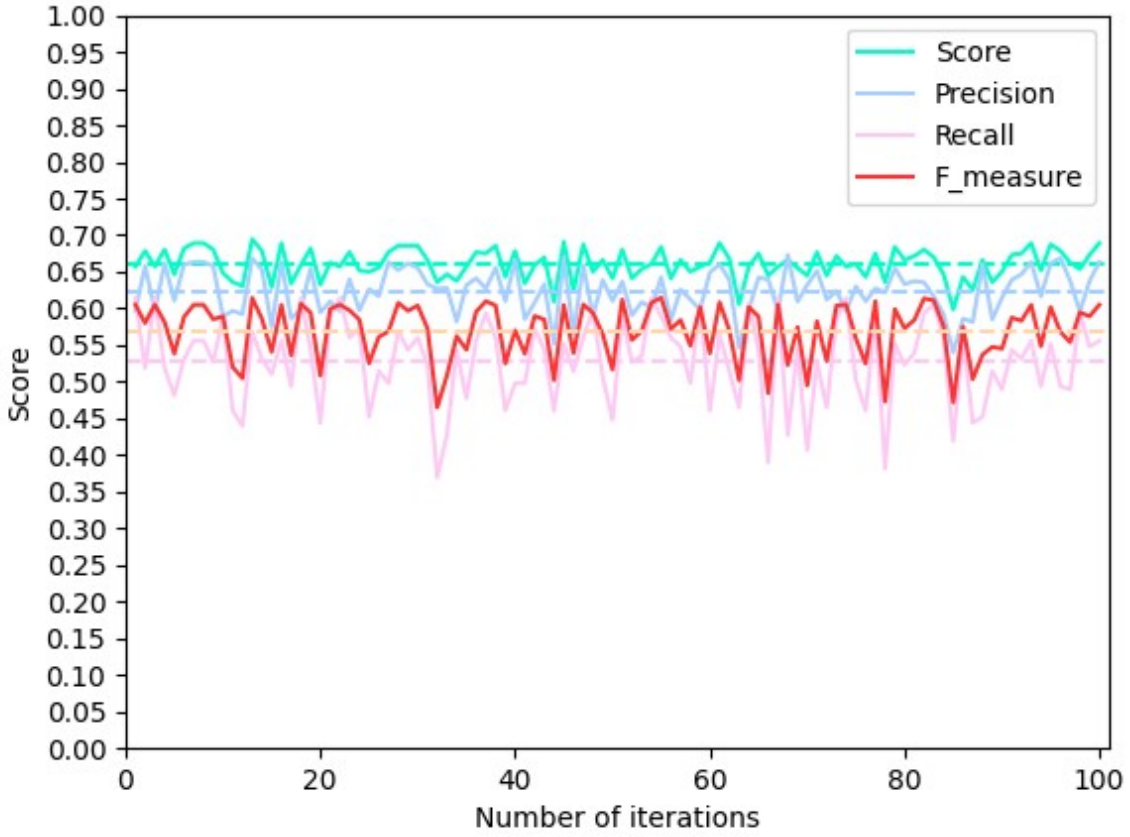
［Propert_Owner］=>［owner］ ・・・・・單位一致後執行同一 EDA 處理

串接資料集後

Fifth Try

都掉下去了，特徵不足？

	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.660888	0.623999	0.527469	0.569653
2	std	0.020639	0.030802	0.058221	0.038069
3	min	0.598579	0.540107	0.369295	0.464752
4	25%	0.646536	0.597245	0.493776	0.542966
5	50%	0.661634	0.627193	0.541494	0.582856
6	75%	0.678508	0.652551	0.564315	0.603898
7	max	0.694494	0.673203	0.614108	0.614719



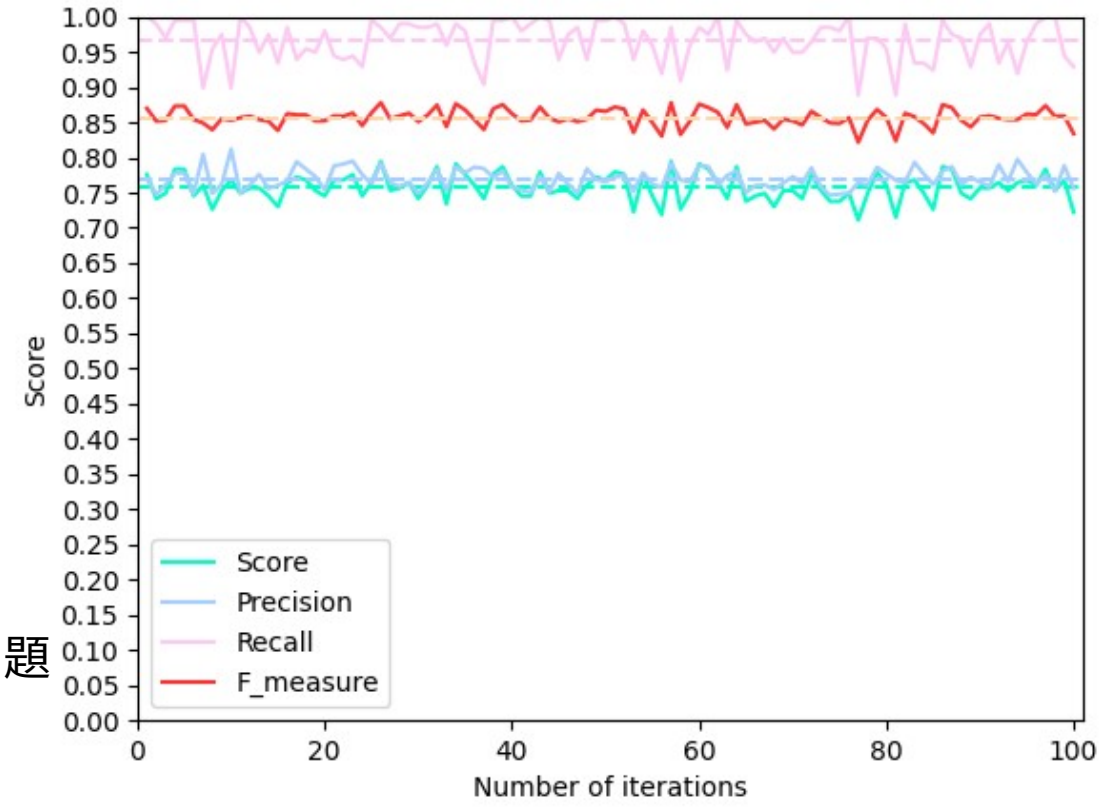
結論

以第 3 次訓練結果是最好的，但依然有所不足，

Recall 高且 Score 的分數相對低

顯示模型整體偏向 True-Target

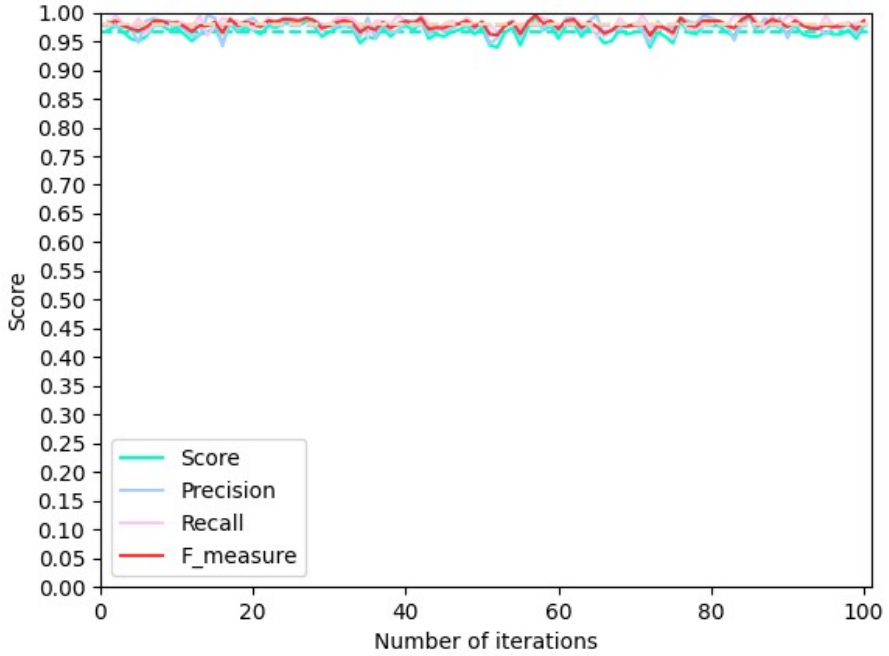
如何優化模型或是模型的適合度會是之後探討的問題



比較 sklearn 的其他模型 資料與 EDA 同 Third Try

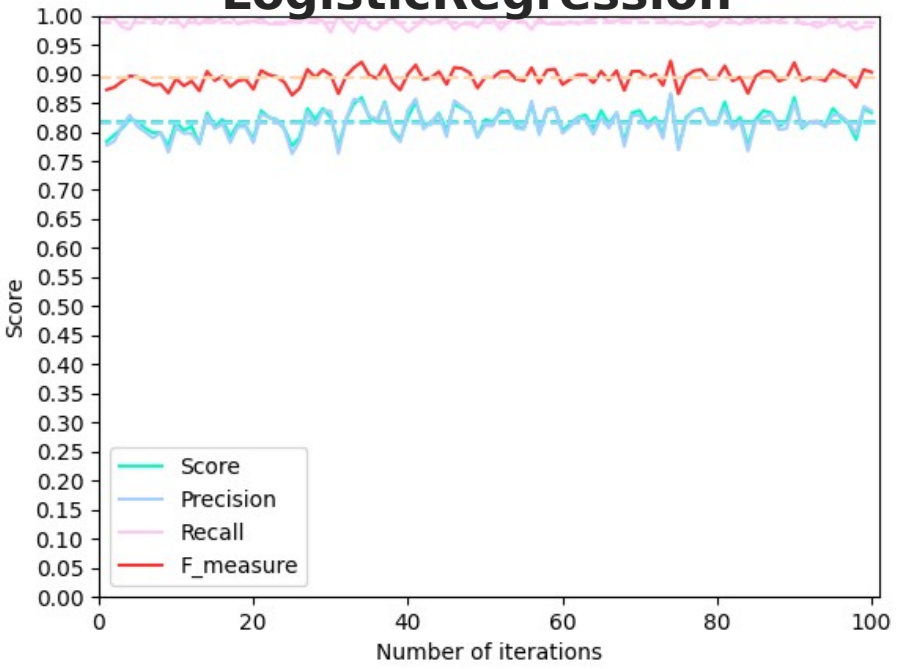
	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.966958	0.976741	0.980874	0.978735
2	std	0.011319	0.012307	0.009542	0.007439
3	min	0.939163	0.941463	0.954774	0.959799
4	25%	0.958175	0.970297	0.975308	0.973933
5	50%	0.967681	0.980149	0.980582	0.979460
6	75%	0.977186	0.985646	0.989664	0.985019
7	max	0.992395	0.995327	1.000000	0.995169

DecisionTrees



	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.817833	0.814682	0.989081	0.893232
2	std	0.020785	0.022940	0.007475	0.013133
3	min	0.771863	0.762295	0.971292	0.863109
4	25%	0.806084	0.799597	0.985056	0.886554
5	50%	0.817490	0.816733	0.990025	0.893333
6	75%	0.832700	0.830645	0.994987	0.903930
7	max	0.863118	0.865854	1.000000	0.922078

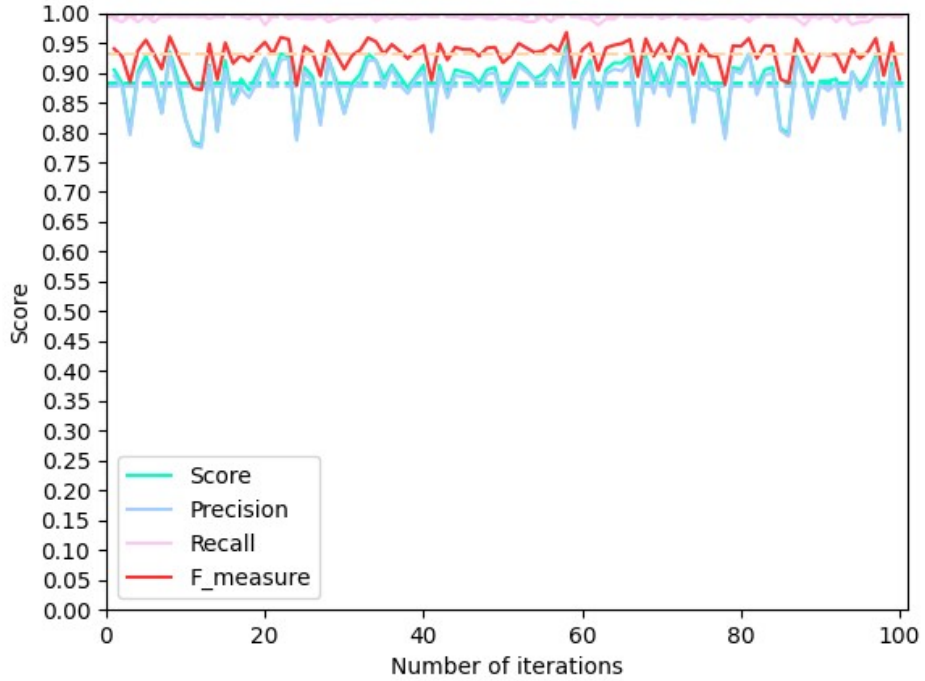
LogisticRegression



比較 sklearn 的其他模型 資料與 EDA 同 Third Try

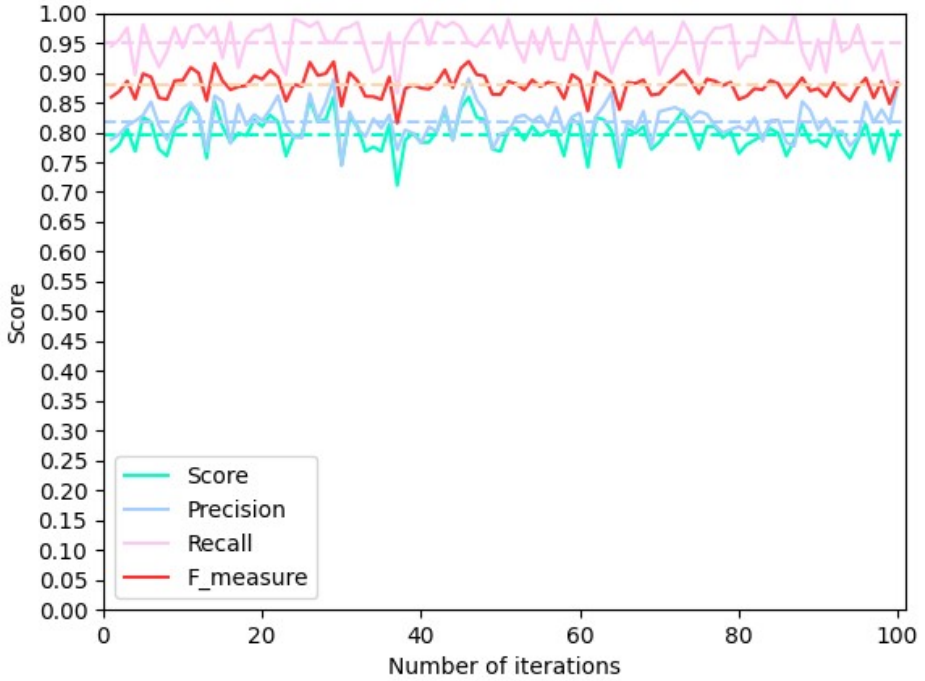
	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.883270	0.875820	0.992548	0.930002
2	std	0.041947	0.040743	0.005010	0.023111
3	min	0.779468	0.774704	0.979167	0.871111
4	25%	0.873574	0.859798	0.990037	0.922375
5	50%	0.893536	0.884352	0.994949	0.936073
6	75%	0.912548	0.905259	0.995204	0.946860
7	max	0.946768	0.937220	1.000000	0.967593

GaussianNB



	Unnamed: 0	score	Precision	Recall	f_measure
0	count	100.000000	100.000000	100.000000	100.000000
1	mean	0.796996	0.818251	0.950275	0.878762
2	std	0.027349	0.027892	0.030794	0.018813
3	min	0.711027	0.744856	0.865979	0.815534
4	25%	0.778517	0.799799	0.931000	0.864391
5	50%	0.802281	0.819338	0.954676	0.881275
6	75%	0.813688	0.836352	0.976247	0.890990
7	max	0.859316	0.889831	1.000000	0.919037

BurnulliNB



全部代碼：

- https://github.com/RCK10274/Issue_Credit_Card

結語

在本次的「機器學習 - 信用卡發放分析」項目中，我們運用了數據處理技術和機器學習算法來預測和分析哪些因素影響信用卡的發放。

通過對模型的訓練和驗證，我們不僅增強了對信用卡發放機制的理解，同時也展示了數據引響結果的過程。

作為結束時的反思，我們需要更多的知識應對數據分析，也需要加強對數學模型的理解，且隨著市場環境和客戶行為的變化，模型也需要不斷地進行更新和調整。此外，模型的應用應當伴隨著持續的監督與評估，以確保其預測的有效性和公平性。　最後，我們對我們機器學習的指導老師感謝，感謝各方提供的知識與技術，期待在未來產生更多的創新和價值。

編輯自　陳弘斌與黃勝鴻團隊

日期： 2024/03/21

END