模型比較報告書(MASTER-Transformer v.s. XGBoost Regressor)

I. 定義名詞

NN模型：使用論文上的MASTER-Transformer架構，使用186個多因子時間序列訓練。

XGBoost模型：這裡採用XGBoost Regressor，對186個因子進行multicollinearity test，檢查多重共線性，去除VIF>10的特徵，然後將篩選後的因子擴增時間滯後的因子，接著篩選特徵，讓模型泛化能力更佳。

XGBoost PCA模型 ： 同樣採用XGBoost Regressor，同樣檢查多重共線性，但是不直接刪除特徵，而是採用PCA降維，將特徵群壓縮到較小的線性空間，篩選出能解釋99%的特徵群。

三者都是採用相同的交易策略(long前30好，short最差30個)，方便比較兩種模型，T0買入股票，T4賣出(以上決策過程都已經有將股票排名的分數shift過，避免未來資訊洩漏)。

這裡使用alphalens的方式是，把模型在T0預測T1買入T4賣出的標準化報酬率，當作是因子，拿來與實際T1買入T4賣出的真實報酬率去做分析，劃分成10個Quantile。

II. 篩選特徵(共線性)

(A) XGBoost模型篩選方法

計算VIF（Variance Inflation Factor），衡量特徵之間的共線性程度。VIF的計算方式，是用線性回歸的統計指標，衡量多重共線性程度，背後的數學核心是Fig. 1.，其中這裡的R_i^2是，除了第 i 個特徵以外，所有其餘特徵對第 i 個特徵進行線性回歸後的決定係數。

$$\mathrm{VIF}_i = \frac{1}{1 - R_i^2}$$

Fig. 1. VIF公式

逐次檢定，每次只刪除一個VIF大於10的特徵，避免誤刪有用的特徵，直到剩餘特徵的VIF都小於10為止。最後篩選出第[0, 1, 5, 8, 12, 13, 16, 20, 27, 28, 34, 35, 41, 45, 46, 47, 51, 61, 62, 66, 67, 69, 72, 76, 83, 87, 88, 92, 93, 94, 99, 102, 105, 106, 113, 114, 116, 119, 120, 121, 122, 123, 124, 125, 126, 128, 129, 130, 132, 133, 134, 137, 139, 140, 145, 150, 154, 158, 162, 164, 165, 167, 170, 172, 174, 175, 176, 178, 179, 181, 182, 183, 184, 185]個特徵，總共74個。

(B) XGBoost PCA模型篩選方法

XGBoost PCA篩選方式不同於XGBoost，不直接捨棄特徵，採用PCA找出新的正交軸，將每一個主成分所能解釋的變異量，畫成累計百分比圖，如Fig. 2.，這裡篩選的條件是篩選出累計能解釋99%變異量的特徵群，而不是用elbow point去當作門檻，因為用elbow point去篩選會去除過多的特徵，如Fig. 2.。

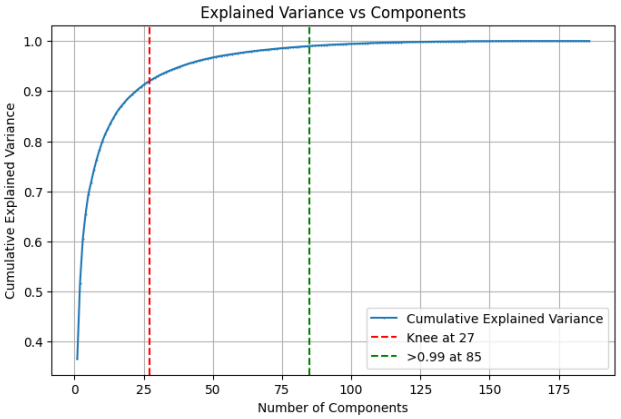

Fig. 2. PCA主成分累計百分比圖

III. 擴增時間序列因子

將原有的MultiIndex DataFrame，多增加新的因子，方法如下：增加時間滯後的資料，新增過去八天的因子資料，意思是，在第T日，不但會有第T日的74個因子資料，還會有第T-1日到第T-8日的74個因子的資料(與MASTER-Transformer相呼應，因為MASTER-Transformer也是輸入第T-1日到T-8日的因子資料去做預測)。

由於XGBoost Regressor模型不能輸入時間序列的資料，因此將MultiIndex DataFrame的結構(row: Date, col: (factor, ticker))，攤平成一般二維的DataFrame，保持時間順序以及股票代號順序。將時間序列預測的問題，化為橫截面預測的問題，同時也不會有未來資訊洩漏的疑慮。

攤平的過程及結果如Fig. 3.到Fig. 4.所示。

[11]:

factor									fact
ticker	1101	1102	1103	1104	1108	1109	1110	1201	1203
Date									
2020-04-15	0.013958	-0.201672	-0.339740	-0.584696	-0.549632	-0.282272	-1.546238	-0.109791	-0.737367
2020-04-16	0.034026	0.337675	-0.107776	-0.674801	-0.589977	-0.445789	-1.597296	-0.105350	-0.415152
2020-04-17	0.378489	-0.099463	-0.494212	-0.707160	-1.345811	-0.533943	-0.167658	1.729352	0.140102
2020-04-20	0.922095	-0.021827	-0.405039	-0.417830	-0.645849	-0.588843	-1.197003	0.675610	1.284421
2020-04-21	0.327030	-1.084917	-0.440371	-0.226221	-0.522327	-0.407719	-0.876101	0.524946	1.006333
...	...	...	...	...	...	...	...	...	...

Fig. 3. MultiIndex DataFrame示意圖

[84]:

	factor	factor_0_shift0	factor_1_shift0	factor_5_shift0	factor_8_shift0	factor_12_shift0	factor_13_shif
0	0.013958	1.135203	1.174119	0.000000	0.000000	0.0000
1	-0.201672	-0.350124	0.537934	0.000000	0.000000	0.0000
2	-0.339740	-0.699315	-0.468685	0.000000	0.000000	0.0000
3	-0.584696	-1.273606	-0.191595	0.000000	0.000000	0.0000
4	-0.549632	-1.323156	-0.783332	0.000000	0.000000	0.0000
...	...	...	...	...	...	
1075039	-0.615998	-0.665086	-0.321606	-0.854870	-1.042193	-0.2086
1075040	0.893574	1.109722	0.964520	0.636865	0.547572	0.7592
1075041	-1.079307	-0.934317	-0.797304	-1.064585	-1.158710	-0.8740
1075042	-0.858004	-0.710468	-0.510363	-0.656977	-0.669673	-0.4353
1075043	-0.922699	-1.438412	-0.972356	-0.845021	-0.720793	-1.1657

1075044 rows × 666 columns

Fig. 4. MultiIndex DataFrame攤平後示意圖

IV. 篩選特徵(特徵重要程度)

以下使用4種篩選特徵的方法，FRegression、XGBoost regressor、LASSO、ElasticNet：

(1) FRegression : 每個特徵X_i個別對目標變數 y 做單變數　線性回歸，然後使用F-statistic 檢定該特徵與目標的線性相關性是否顯著。

(2)XGBoost regressor : 每次split時，計算當前節點的 loss，嘗試用每個特徵去切這個節點 ，選擇能讓 loss 減少最多的特徵來切，將「loss 降低量」累積到該特徵的 gain，等所有樹跑完後，把 gain 對每個特徵加總做平均，得到特徵重要程度的分數。

(3)LASSO : 自動把無用的特徵係數壓成 0，如Fig. 4.所示。

$$\text{Loss} = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha\sum_{j=1}^{p}|\beta_j|$$

- 第一項是普通的均方誤差 loss
- 第二項是 L1 正則化項，強迫所有權重 β_j 趨近於 0
- α 控制壓縮強度（越大 → 越多係數變 0）

Fig. 5. LASSO篩選特徵圖(alpha=0.1)

(4) ElasticNet : 將 Lasso（L1）和 Ridge（L2）結合起來的線性模型。

L1（Lasso）部分會讓「不重要的特徵」的係數壓到 0 → 特徵被剃除

L2（Ridge）部分則幫助模型在有 共線性 的情況下更穩定、不誤刪全部特徵

如

$$\text{Loss} = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \alpha\left[\rho\sum_{j}|\beta_j| + (1-\rho)\sum_{j}\beta_j^2\right]$$

- α：正則化強度（越大，懲罰越重）
- ρ：L1 與 L2 的混合比（0=純 L2，1=純 L1）

Fig. 6. ElasticNet篩選特徵圖(alpha=0.01, rho=0.7)

V. 篩選特徵數量

排名出特徵重要程度後，將特徵重要度排序，形成曲線，找出**最大曲率點**(elbow point)，意思是邊際效益遞減最快的點，如Fig. 7.中藍色重要度曲線以及紅色直線交會點，選擇最大曲率點之前的特徵，會是最具經濟價值的特徵群。

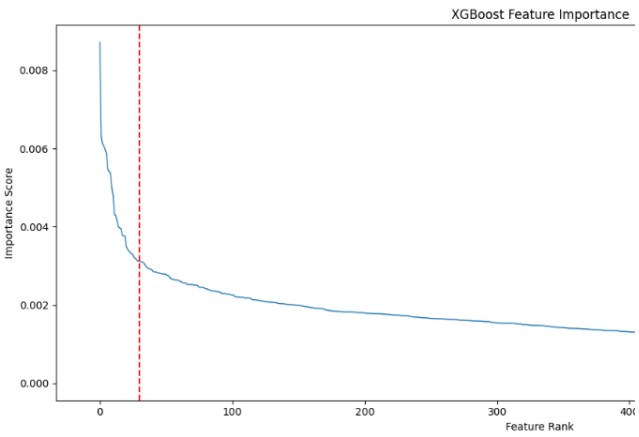

Fig. 7. Elbow point篩選圖

VI. TimeSeriesSplit cross-validation

由於時間序列本身前後相依的結構，使用普通的Cross Validation會造成未來資訊洩漏，因此需要採用TimeSeriesSplit cross-validation，如Fig. 8.。在驗證時總共切了10份Fold，XGBoost模型和XGBoost PCA模型都各自做TimeSeriesSplit cross-validation。

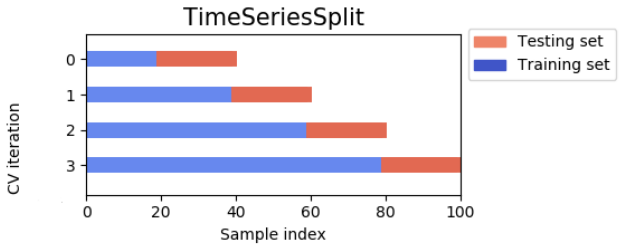

Fig. 8. TimeSeriesSplit cross-validation示意圖

在每一個Fold中，用PCA壓縮多重共線性的維度，保留99%變異後，可以從Fig. 9. 觀察出，PCA合併完的主成分維度都落在一個不大的區間，代表有多重共線性的特徵較為固定，不會隨著時間而有劇烈的變動，因此用PCA壓縮維度，較可以不用擔心train set和validation set的多重共線性分布會有極大的差異，造成用PCA不適當。

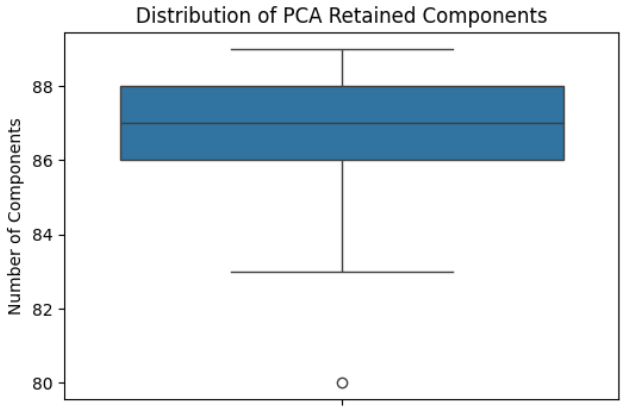

Fig. 9. PCA主成分數量分布圖

每一個Fold都會篩選出不同的特徵群(意思是每一種篩選方法的每一個Fold，篩出來的特徵都不一樣，是滾動式篩選及訓練，確保與時俱進)，並且用篩選後的特徵群，用來預測testing set，計算出MAE當作衡量指標，最終將10個MAE取平均、標準差、風險後平均，評估4個模型的表現，結果如Fig. 10.和Fig. 11.。

Selector	MAE Mean	MAE Std	MAE Mean / Std
Elastic	0.645783	0.010307	62.656174
LASSO	0.646263	0.009918	65.161759
FRegression	0.646868	0.010789	59.954126
XGB	0.646966	0.011948	54.147126

Fig. 10. XGBoost模型(MAE mean, std, risk adjusted mean)

Selector	MAE Mean	MAE Std	MAE Mean / Std
LASSO	0.645199	0.009801	65.829817
FRegression	0.645215	0.010141	63.625243
Elastic	0.645258	0.010306	62.607741
XGB	0.645369	0.009890	65.252973

Fig. 11. XGBoost PCA模型(MAE mean, std, risk adjusted mean)

VII. 統計檢定模型效能

(模型內部比較)

(A) Friedman 檢定

接著用Friedman Test分別對XGBoost模型和XGBoost PCA模型，各自檢定用FRegression、XGBoost、LASSO、ElasticNet四種篩選方法，MAE的表現是否有顯著差異。

XGBoost 模型： Friedman statistic: 1.4400，p-value: 0.6962 > 0.01，MAE並無顯著差異。

XGBoost PCA 模型： Friedman statistic: 0.1200，p-value: 0.9893 > 0.01，MAE並無顯著差異。

(B) Nemenyi Post-hoc 檢定

然後用Nemenyi Post-hoc 檢定，各自查看XGBoost模型與XGBoost PCA模型，用四種不同的特徵篩選方式，MAE表現是否有差異，如Fig. 12.和Fig. 13.，可以看出，不論是XGBoost模型還是XGBoost PCA模型，任意特徵篩選方式，兩兩之間並無顯著效能差異，因此在回測時，可以採用任意一種方法，接下來會採用FRegression作為篩選特徵的方法，因為篩選速度最快，計算量最小，計算速度：FRegression > XGBoost > LASSO > Elastic。

	XGB	FRegression	LASSO	Elastic
XGB	1.000000	1.000000	0.985723	0.726349
FRegression	1.000000	1.000000	0.985723	0.726349
LASSO	0.985723	0.985723	1.000000	0.899884
Elastic	0.726349	0.726349	0.899884	1.000000

Fig. 12. (XGBoost) Nemenyi Post-hoc 檢定表

	XGB	FRegression	LASSO	Elastic
XGB	1.000000	0.998155	1.000000	0.998155
FRegression	0.998155	1.000000	0.998155	0.985723
LASSO	1.000000	0.998155	1.000000	0.998155
Elastic	0.998155	0.985723	0.998155	1.000000

Fig. 13. (XGBoost PCA) Nemenyi Post-hoc 檢定表

VIII. 統計檢定模型效能

(模型間比較)

(A) Paired t-test

(XGBoost PCA模型 v.s. XGBoost模型)

XGBoost: p-value = 0.1207 => 不顯著

FRegression: p-value = 0.0229 => 顯著

LASSO: p-value = 0.1550 => 不顯著

ElasticNet: p-value = 0.3209 => 不顯著

(B) Wilcoxon signed-rank 檢定

(XGBoost PCA模型 v.s. XGBoost模型)

XGB : p-value = 0.1611 => 不顯著

FRegression : p-value = 0.0312 => 顯著

LASSO : p-value = 0.1875 => 不顯著

ElasticNet : p-value = 0.3125 => 不顯著

Wilcoxon signed-rank test是非參數檢定，不依賴於常態假設。

從Paired t-test和Wilcoxon signed-rank 檢定，可以看出，在FRegression 篩選方法下，XGBoost PCA模型的MAE顯著小於XGBoost模型，代表XGBoost PCA模型在FRegression方法下，表現勝過於XGBoost 模型。

IX. 滾動式回測過程

用過去約4.5年的資料當作訓練集，用於未來三個月的預測，每經過3個月汰除最久之前3個月的資料，並且新加入最近過去3個月的資料當作是新的訓練集，避免過去資料分布與現在分布相差過大，造成失真，進行滾動式訓練及回測(每3個月重新訓練模型)。

X. PCA結果

最一開始是使用線性PCA去降維，然而效果卻如同Fig. 14.以及Fig. 15.，效果不盡理想，懷疑是因為線性降維的關係，導致無法金融市場上的非線性關係，造成丟失有用資訊。

所以接著使用Kernel PCA採用非線性方法，將資料投影到高維空間中，由於資料量太多，直接使用原始的Kernel PCA會造成Kernel Matrix達到TB資料級別，所以這裡採用替代方案，使用Nystroem 方法。

Nystroem 方法是一種將非線性核技巧轉換為線性特徵表示的近似方法，抽樣部分資料來近似整體的核矩陣，以此提升運算效率並降低記憶體消耗。

原本Kernel Matrix的時間複雜度和空間複雜度都是$O(n^2)$，用了Nystroem 方法，變成O(m · n)，其中m是抽樣筆數，然而效果也是不盡理想，也約莫如同Fig. 14.以及Fig. 15.，可能原因應該是金融市場的時間變動性，用過去的資料找到PCA的方向，然後用於現有的資料上，會造成沒有考慮到金融市場的分佈已經悄然改變。

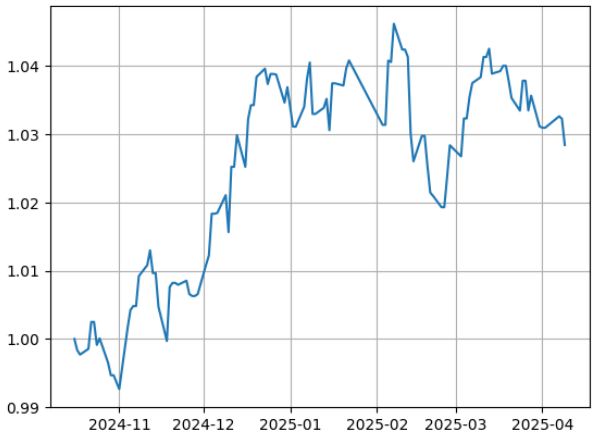

Fig. 14. PCA報酬圖1

Fig. 15. PCA報酬圖2

XI. PCA反思

然而PCA並非一無是處，從Fig. 15.以及Little_Algo_Trader的Patreon訂閱版文章「量化交易中的PCA：不只是降為那麼簡單」，可以得知，PCA可以抓取市場轉折點，當PCA負荷因子有所改變時，便可以得知市場已經開始轉變，從Fig. 15.看出，再2025年2月中有一個非常明顯的轉折點，若回頭看PCA負荷因子，便可以看出有所轉變，回頭看那一段時間的加權指數也可以看出，確實是在那個時候市場開始有了明顯的轉變。

所以PCA不用於直接壓縮資料餵給模型，而是適合偵測市場或是股票的轉折點(資料分布變化)。

XII. IC值比較

從Fig. 16. 可以看出，因子預測力，NN模型表現比XGBoost模型好，但NN模型標準差比XGBoost高，Risk Adjusted IC也是NN模型表現較好。p-value表示，NN模型的因子預測力，不是偶然的，而是真正有效。

	Metric	XGBoost模型	NN模型
0	IC Mean	0.006	0.066
1	IC Std.	0.057	0.110
2	Risk Adjusted IC	0.107	0.598
3	t-stat(IC)	1.177	6.242
4	p-value(IC)	0.242	0.000
5	IC skew	-0.410	-0.064
6	IC kurtosis	0.411	-0.238

Fig. 16. IC值表格比較圖

XIII. Alpha以及Beta比較

從Fig. 17.來看，對於alpha表現，NN模型表現比XGBoost模型好，兩者的beta則是XGBoost較小，兩者beta都與市場整體輕微負相關，再區分Quantile上，不論是Top Quantile Return還是Bottom Quantile Return，都是NN模型表現較好，最後的Spread(bps)也必然是原始attention表現出色。

	Metric	XGBoost模型	NN模型
0	Ann. alpha	0.037	0.078
1	beta	-0.022	-0.132
2	Top Quantile Return (bps)	15.255	27.984
3	Bottom Quantile Return (bps)	-16.994	-39.621
4	Spread (bps)	32.249	67.605

Fig. 17. Alpha和Beta以及Quantile比較圖

XIV. Turnover比較

從Fig. 18.來看，Q1~Q10的Turnover整體來說，XGBoost模型比NN模型低， Mean Factor Rank Autocorrelation也是比NN模型好。

	Metric	XGBoost模型	NN模型
0	Quantile 1 Mean Turnover	0.366	0.341
1	Quantile 2 Mean Turnover	0.468	0.629
2	Quantile 3 Mean Turnover	0.513	0.713
3	Quantile 4 Mean Turnover	0.540	0.763
4	Quantile 5 Mean Turnover	0.551	0.779
5	Quantile 6 Mean Turnover	0.553	0.775
6	Quantile 7 Mean Turnover	0.539	0.742
7	Quantile 8 Mean Turnover	0.539	0.698
8	Quantile 9 Mean Turnover	0.454	0.628
9	Quantile 10 Mean Turnover	0.296	0.403
10	Mean Factor Rank Autocorrelation	0.794	0.743

Fig. 18. Turnover比較

XV. 交易策略報酬結果(單利計算)

這裡都是採用最簡單的交易策略，T0決定買入的股票，持有期間T1~T4，T4賣出獲利。

從Fig. 19. 和 Fig. 20.來看，NN模型比起XGBoost模型，整體報酬以及報酬穩定度來說，都是比較好的，而且抗跌的能力也較佳。

Fig. 19.XGBoost模型

Fig. 20. NN模型

XVI. 比較報酬Quantile

以下的Quantile是計算根據算出的分數排名，劃分成Q1~Q10去買入，**當日報酬是採計T1買入T4賣出的報酬率**(相當於時間是5倍速流逝，這種方式只是單純看預測五日後的預測是否準確，以及模型是否能區分好壞股票，所以不是實際交易的報酬)，而且這裡採計純粹以**採計T1買入T4賣出的報酬率相加**，所以會出現負值。

從Fig. 21. 和 Fig. 22.可以看出，XGBoost模型的Q1~Q10的Quantile曲線，和 NN模型Q1~Q10的Quantile曲線分得很開，但是NN模型比XGBoost模型分得更開。

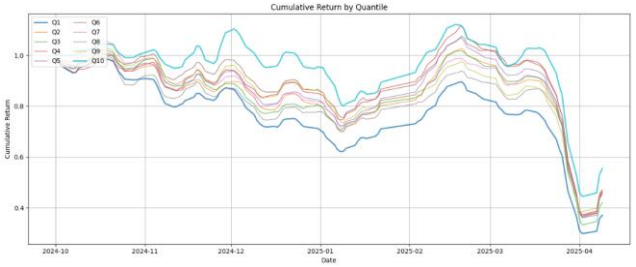

Fig. 21. XGBoost模型 (Cumulative Quantile)

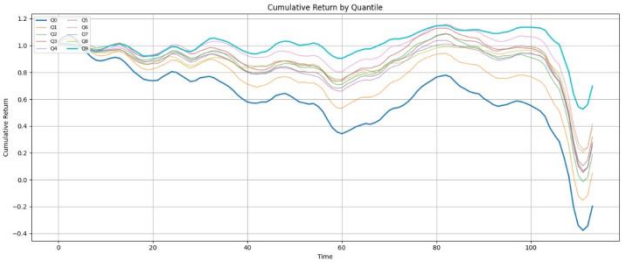

Fig. 22. NN模型 (Cumulative Quantile)

XVII. 訓練難易度及解釋性

整體來說，XGBoost是傳統機器學習模型，待配篩選特徵，比NN模型好訓練，而且好解釋。

XVIII. 數值收斂

模型預測的是標準化後報酬率，因此valid loss在不同的loss function底下有不同的標準。

MSE : valid loss < 1 才算是好模型

因為標準化後報酬率滿足mean=0, std=1，最沒有預測力的模型，什麼都不做，只輸出mean，意思是指輸出0的話。以常態分佈來說：

$$\mu = 0, \quad \sigma = 1$$

$$Z \sim \mathcal{N}(\mu, \sigma^2)$$

$$\mathbb{E}[(Z - \mu)^2] = \sigma^2 = 1$$

所以在MSE底下，valid loss < 1 才算是好。

MAE : valid loss < 0.797 才算是好模型

以常態分佈來說：

$$\mu = 0, \quad \sigma = 1$$

$$Z \sim \mathcal{N}(\mu, \sigma^2) = \mathcal{N}(0, 1)$$

$$\mathbb{E}[|Z - \mu|] = \sigma \cdot \sqrt{\frac{2}{\pi}} \approx 0.797$$

所以在MAE底下，valid loss < 0.797 才算是好。

對於MAE，NN模型和XGBoost模型都可以落在0.67以下，都比0.797這個門檻還小。

XIX. 結果

以回測結果來說，NN模型的穩定性以及報酬都比XGBoost模型高，然而XGBoost經過種種篩選，表現也不俗，可以再經過改善獲得更好的效果。

XX. 啟發

從PCA XGBoost模型的例子來看，PCA XGBoost模型的MAE相當低，但是結果未必是好的，所以可以知道MAE不應該是訓練模型時的唯一最佳化考量，因此在MASTER-Transformer那一篇論文中純粹以MSE、MAE去做最佳化是有一些問題的，應該要搭配其他指標，例如Sharpe ratio等等才能選到最好的模型，MAE < 1、MSE < 0.797只是最基本的門檻。