

Github link: <https://github.com/yunlinwadonwang/ml-final-project>

Model link: <https://drive.google.com/drive/folders/1oKaxiLyL4Oi3-eD9PAg74Nc-7Zt4AFV?usp=sharing>

## Result

Submission and Description		Private Score	Public Score	Selected
	109550038.csv Complete (after deadline) · now	0.58997	0.58592	<input type="checkbox"/>

## Reference

<https://www.kaggle.com/competitions/tabular-playground-series-aug-2022/discussion/349810>

將 product code 排列組合，區分出 train data 和 valid data

## Introduction and Methodology

### Pre-processing:

使用 sklearn 的 SimpleImputer，用平均值填入缺少的值

使用 category\_encoders 的 WOEEncoder，透過 weight of evidence 對類別型特徵進行編碼

### Training:

透過 product code 排列組合產生 train data 和 valid data

使用 sklearn 的 LogisticRegression 作為 model

### Fit and predict

使用 sklearn 的 roc\_auc\_score 對 predict 評分

Save 10 models and woencoder

### Inferencing:

Load 10 models and pre-fit woencoder

Predict

## Different Approaches

我有嘗試使用神經網路，架構是樹狀結構

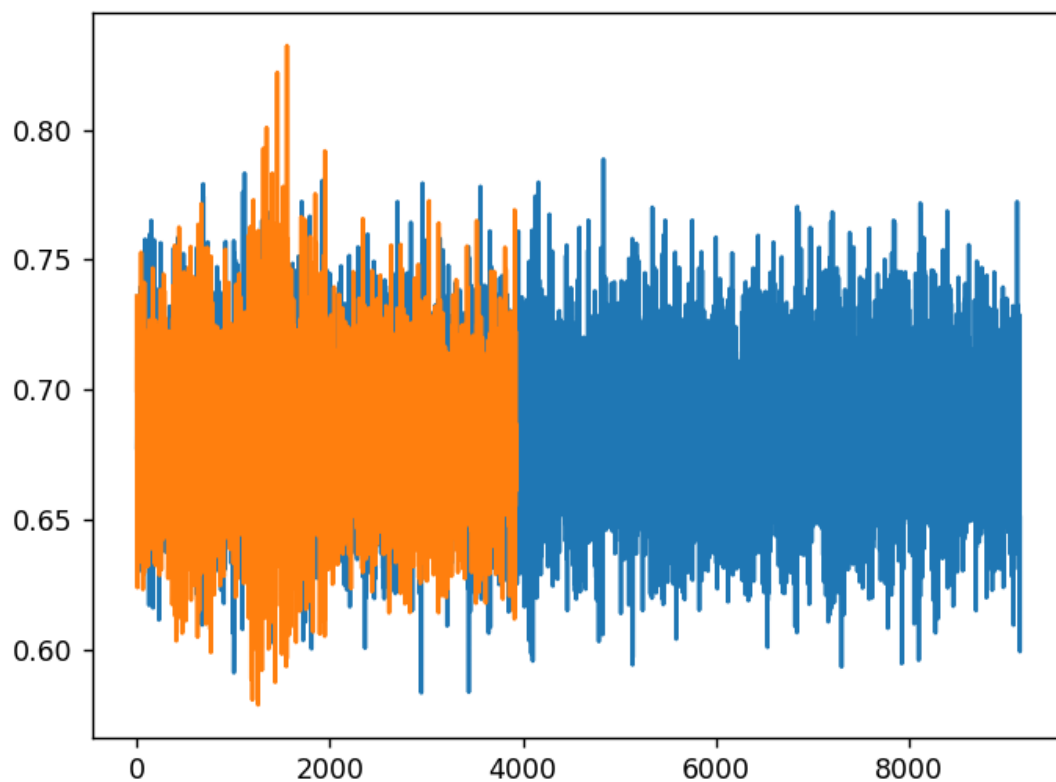
```

self.layer1 = nn.Sequential(
    nn.Linear(22, 22), nn.ReLU(), nn.BatchNorm1d(22), nn.Dropout(p=0.2), nn.Linear(22, 22), nn.ReLU())
mlist_leaf = []
mlist = []










def nn_tree(input, bias, depth):
    if depth <= 0:
        mlist_leaf.append(nn.Sequential(nn.Linear(input, input-bias), nn.ReLU(), nn.Linear(input -
                                          bias, input-bias-1), nn.ReLU(), nn.BatchNorm1d(input-bias-1)))
    return
    mlist.append(nn.Sequential(
        nn.Linear(input, input-bias), nn.ReLU()))
    nn_tree(input-bias, 1, depth-1)
    nn_tree(input-bias, 2, depth-1)
nn_tree(22, 1, 3)
nn_tree(22, 2, 3)
self.layers = nn.ModuleList(mlist)
self.layer_leaf = nn.ModuleList(mlist_leaf)
self.layer2 = nn.Sequential(
    nn.Linear(240, 22), nn.ReLU(), nn.Linear(22, 1), nn.Sigmoid())

```

不過結果並不理想，loss 沒有收斂的跡象，會一直來回震盪



嘗試過只用一層也無法收斂，又發現 label 分布不平均，所以試著做資料前處理，還是沒有太大的進展，於是放棄換其他方法

	<b>submission.csv</b> Complete (after deadline) · 2d ago	<b>0.56332</b>
	<b>submission.csv</b> Complete (after deadline) · 2d ago	<b>0.54081</b>
	<b>submission.csv</b> Complete (after deadline) · 2d ago	<b>0.55365</b>
	<b>submission.csv</b> Complete (after deadline) · 2d ago	<b>0.56577</b>
	<b>submission.csv</b> Complete (after deadline) · 3d ago	<b>0.55378</b>
	<b>submission.csv</b> Complete (after deadline) · 3d ago	<b>0.56389</b>
	<b>submission.csv</b> Complete (after deadline) · 3d ago	<b>0.56528</b>
	<b>submission.csv</b> Complete (after deadline) · 3d ago	<b>0.55852</b>
	<b>submission.csv</b> Complete (after deadline) · 3d ago	<b>0.55396</b>

## Summary

這次作業花了很多時間在訓練神經網路，然而成效都不是很好，因此我明白了神經網路有其侷限性，並不是萬能的，或者說許多調整方面非常困難且麻煩，相較之下，這次使用 **LogisticRegression** 的成效好，且訓練速度快，讓我了解遇到不同問題需要使用不同的 **model**。