

109550042_Final Report

GitHub link of your code

https://github.com/roylin506/2022_ML


Model link

https://drive.google.com/file/d/1yEhxdebyoRrfivguMSxipaB37qoo0WmU/view?usp=share_link

Brief introduction

用 PyTorch 實做 neural network，output dimension 為 1，也就是 failure 之機率。

最佳結果

Submission and Description	Private Score ⓘ	Public Score ⓘ	Selected
 109550042.csv Complete (after deadline) · 2h ago · DNN drop = 0.1 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 40	0.59065	0.57916	<input type="checkbox"/>

可以透過 inference 檔案復現分數。

Methodology

Data pre-process

- 把 column = 'ID' 丟掉。
- 把不是數值的 value 轉成數值
 - 在 column = 'attribute_0', 'attribute_1'，把 'matreial_N' 轉換成 N。
 - 在 column = 'product_code'，把大寫英文字母轉成其 ASCII code 的三倍。
- 把所有非 float 的 column 變成 float。

- 把 missing value 變成該 column 的 median 值。

Model architecture

- 6 層 Deep Neural Network。
- hidden dimension 設定為 512, 2048, 1024, 512, 32。
 - activation function = ReLU()
- output dimension 設定為 1。
 - activation function = sigmoid()
- 在某些層 dropout 避免 overfitting。
- loss function 使用 `nn.BCELoss()`。
- optimizer 使用 `torch.optim.Adam(model.parameters(), lr=LR)`
- LR 設為 1e-6。

```

# set parameters
drop1, drop2, drop3, drop4 = 0.1, 0.15, 0.15, 0.1
dim1, dim2, dim3, dim4, dim5 = 512, 2048, 1024, 512, 32

# Create Model
class mymodel(nn.Module):
    def __init__(self):
        super(mymodel, self).__init__()
        self.model = nn.Sequential(nn.Linear(24, dim1),
                                    nn.ReLU(),
                                    nn.Linear(dim1, dim2),
                                    nn.ReLU(),
                                    nn.Dropout(drop1),
                                    nn.Linear(dim2, dim3),
                                    nn.ReLU(),
                                    nn.Dropout(drop2),
                                    nn.Linear(dim3, dim4),
                                    nn.ReLU(),
                                    nn.Dropout(drop3),
                                    nn.Linear(dim4, dim5),
                                    nn.ReLU(),
                                    nn.Dropout(drop4),
                                    nn.Linear(dim5, 1),
                                    nn.Sigmoid())

    def forward(self, x):
        logits = self.model(x)
        return logits

```

Hyperparameters

- LR 設為 1e-6。
- 調整 hidden dimension。
 - 最終設定為 512, 2048, 1024, 512, 32。
- 調整 dropout 比率。
- 最終設定為 0.1, 0.15, 0.15, 0.1。









Summary

用神經網路做二元分類，在調整過程中，發現網路架構對預測結果影響很大，前前後後測試了好幾種架構，層數從2層到6層都試過。以及層數過深容易有 overfitting 的現象，但加上 dropout 後可以獲得改善。還有 learning rate 太大時無法收斂等等，算是更熟悉了神經網路的實做細節。








Bonus

Comparisons of different approaches

- The state-of-the-art methods for this task
- 除了使用神經網路，也使用了其他機器學習模型，例如：
 - 線性的 Linear Regression, Logistic Regression
 - tree-based 的 Decision Tree 與 Random Forest
- 初次嘗試時，線性模型相較於 tree-based model 表現更好。
- 於是開始針對線性模型優化。經過非常非常非常非常久的嘗試，以及多次的 hyperparameter tuning，線性方法最多只能到 0.58894，距離 baseline 不到 0.001。
- 圖為 hyperparameter tuning 時在 Kaggle 上獲得的分數，並上傳訊息紀錄 hyperparameter。
 - （只擷取一小部份）

Overview	Data	Code	Discussion	Leaderboard	Rules	Team	Submissions	Late Submission	...
	109550042.csv						0.58858	0.58096	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 18 lbfgs									
	109550042.csv						0.58703	0.58153	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 18 liblinear									
	109550042.csv						0.58703	0.58153	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 18 liblinear									
	109550042.csv						0.5854	0.58191	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 17 newton-cg									
	109550042.csv						0.58703	0.58153	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 17 liblinear									
	109550042.csv						0.58858	0.58096	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 17 lbfgs									
	109550042.csv						0.58892	0.58198	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 16 saga									
	109550042.csv						0.58856	0.58187	<input type="checkbox"/>
Complete (after deadline) · 9h ago · Logistic 16 sag									

- 以及後來使用 DNN 之 hyperparameter tuning
 - 列出在調整的各種參數




Submission and Description	Private Score ^①	Public Score ^②	Selected
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58977	0.58343	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58784	0.5797	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58933	0.58252	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58714	0.57694	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58914	0.58137	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58896	0.58336	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 1h ago · DNN drop = 0.1 0.15 0.15 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 35	0.58931	0.58364	<input type="checkbox"/>

Comprehensive related works survey

- 去比賽的討論區看了一輪，發現許多人使用 Linear model，並且在上面比較各方法，以及細節調整所產生的結果。
- 後續換成神經網路架構時就沒有再上去討論區逛逛ㄌ。

Thorough experimental results

- 附上一些調整過程與數據

 109550042.csv Complete (after deadline) · 3h ago · DNN drop = 0.3 0.35 0.3 0, dim = 512 2048 1024 512 32, 8e-05, 60	0.58989	0.58231	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 3h ago · DNN 0.3 0.35 512 2048 1024 5e-05 80	0.5898	0.58273	<input type="checkbox"/>
 109550042.csv Complete (after deadline) · 2h ago · DNN drop = 0.1 0.1 0.1 0.1, dim = 512 2048 1024 512 32, batch = 500, 1e-06, 40	0.58792	0.58176	<input type="checkbox"/>

- 主要針對以下參數做調整：
 - model

- logistic regression
- linear regression
- DNN
- For DNN
 - layer 數
 - 層數小時表現比較糟糕
 - 層數增加時，分數會稍微提升
 - overfitting 的問題要透過 dropout 減緩
 - hidden dimension
 - 從 32 到 2048 都試過
 - 不同層數也會有差
 - 試出來的結果是，第一層跟最後一層 hidden dimension 設成較小，對於分數有幫助。
 - Drop rate
 - 越大可以減緩 overfitting
 - 但太大會丟掉太多訓練結果導致模型預測準確率低。
 - 每層的 drop rate 也會不一樣。
 - 最好的 drop rate 介於0.1 到 0.2 之間。
 - batch size
 - 主要影響到 stochastic gradient descent 的收斂方向準確度。
 - 也連帶影響到 learning rate 的設置。
 - 最後調成 batch size = 500, learning rate = 1e-6。

Interesting findings or novel features engineering

- 對 data scale 做處理。
 - 把 value 對應到差不多的值域範圍，但幫助不大，後來就捨棄。

- 也試過對 output 機率的 scale 做調整，
 - 對機率做 線性調整、標準化、等倍率放大等等，但分數都一樣，也捨棄。
- 平行化執行
 - 這蠻有趣的，因為上傳 Kaggle 時都要等大約五秒，在作業初期訓練很快時，等待 Kaggle 時會花一半左右的時間。
 - 因此我寫了平行程式，讓另一個執行緒幫我送檔案給 Kaggle，而原本的執行緒就可以繼續用新的參數做訓練。
 - 但他動起來怪怪的，有時後會卡住，所以還是以平行失敗收尾。
- cuda
 - 1050ti 就超快喔（可能我還沒被更強顯卡寵壞）