


Introduction to Machine Learning Final Project Report

Name: 劉家達 StudentID: 0713332

1. [Github link](#)
2. Performance

Submission and Description	Private Score ⓘ	Public Score ⓘ	Selected
 0713332_submission.csv Complete (after deadline) · now	0.5915	0.58911	<input type="checkbox"/>

3. Reference Code:

- 關於主要的方法，我參考了 Kaggle discussion panel 的[這篇](#)的方法，達到比原作者更好的 performance (0.59105 vs. ours 0.5915)。
- 關於 AUC/ROC，我參考了[這篇文章](#)。
- 關於 ensemble model 的 post-processing，我參考了[這篇文章](#)。

4. Introduction: 這次的競賽是給很多數值或類別的 features，然後預測 failure 的機率。跟以往分類問題不同的是，這次是要預測失敗的機率，評估的指標是 AUC/ROC (即比較 TPR/FPR 之間的變化，ROC 曲線下的面積越大代表效能越好)。原本想說可以用分類問題的模型著手，loss function 可以使用 Cross Entropy，但要預測得是機率，因此嘗試使用 Sklearn 中 Logistic Regression 的模型的 predict_prob 進行預測，但效果不佳 (連 baseline 都沒達到)，初估是因為使用了太多沒什麼用的 features，導致模型學不起來。後來使用了 blending of Logistic

Regression 的方法加上一些前處理、特徵工程和後處理，使得模型效能得到不少的提升。

5. Methodology:

- Data Pre-processing

- i. Missing Values: 將 measurement_3, measurement_4, measurement_5 的 missing value 變成 binary 的 feature，希望讓模型學到「某個」產品有某個缺失值與「結果會失敗」之間的關聯性
- ii. Loading: 因為這個特徵是偏度比較大的數據，因此用 log1p 函數進行轉換，使其服從高斯分佈，這樣會對我們後續的分類結果得到一個更好的結果
- iii. Dummy variables: attribute_0 and attribute_1 是類別，所以用成新的 feature，類似 one hot code 的概念，跟第一點的目的類似
- iv. 填補缺失值
 1. 找到 measurement_17 有缺失值的 data，然後根據不同的 product 對 measurement_17 進行回歸預測，然後填入
 2. 對剩下的所有有缺失值的 measurement，用 KNNImputer 進行缺失值的預測

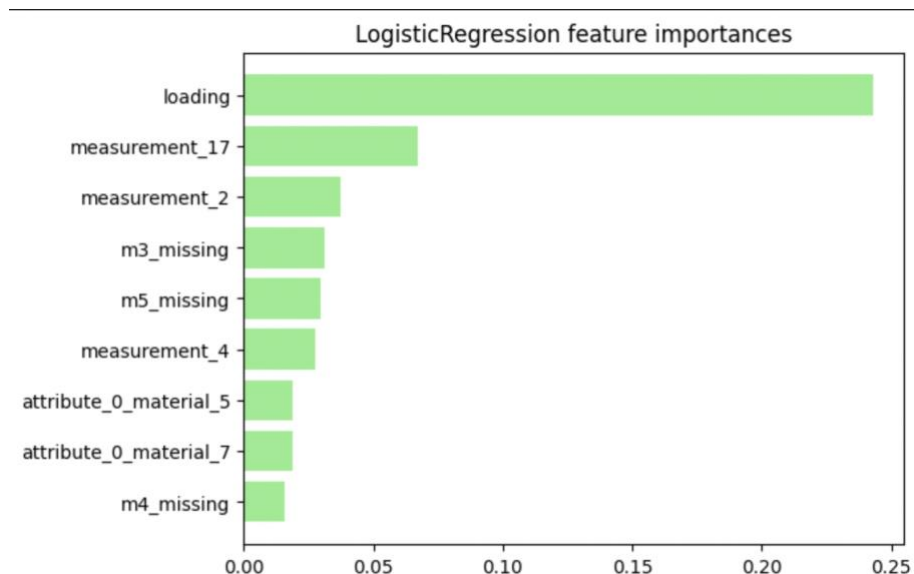
- Feature Engineering

- i. 標準化: 使用 sklearn 的 StandardScaler 進行

1. Training data: fit + transform
2. Validation data: 只有 transform
3. Testing data: 只有 transform

- Model Architecture

- i. 根據一開始的實驗觀察, 'loading' , 'measurement_17' , 'measurement_2' 是我認為比較重要的 features, 因此我將他們列為 default features, 其餘我根據一開始「使用所有特徵的模型」的 feature importance 選出了前十重要的 candidate features。再將所有 candidate features 的排列組合加上 default features 的模型分別 train 一次, 找到令 average auc score 最高的排列組合。Feature Importance 的圖如下:



- ii. Training 過程中的 cross validation 使用 Stratified K Fold，定義是分層採樣交叉切分，確保訓練集，測試集中各類別樣本的比例與原是數據集中是相同的。
- Hyperparameters (Grid Search)
 - i. 在 Logistic Regression 中，我對 C 跟 Solver 這兩個參數進行了校調，因為我覺得 max iteration 設成 1000 基本上都會收斂（前處理的優點），所以就用找到的 best hyperparameter 再將模型訓練一次，得到的 Average auc 是 0.592 左右，但老實說差距不大，只是實驗性的作法。BTW 後面的 ensemble 模型皆是以此組 hyperparameter 進行訓練
- Training Models
 - i. 這次的模型使用多個 logistic regression 的 blending，分別用不同的特徵，以及相同的超參數，和 cross validation 做訓練
 - ii. 原本作者的方法使用四個 logistic regression 模型合併，但我覺得四個模型比較難做後續權重參數的調整（因組合太多，難以辨認哪個模型的貢獻較顯著），因此我只採用了三個模型進行 blending

Model	Feature Selection	Missing Value Feature	Number of Parameters
Model 1 (Total 9 features)	['loading', 'measurement_17', 'measurement_2', 'm3_missing', 'm5_missing',	Many	Many

	'measurement_4', 'attribute_0_material_5', 'attribute_0_material_7','m4_missing']		
Model 2 (Total 4 features)	['measurement_1', 'measurement_2', 'loading', 'measurement_17']	Not many	Not many
Model 3 (Total 5 features)	['m3_missing', 'm5_missing', 'measurement_2', 'loading', 'measurement_17']	Many	Not many

iii. 我的想法是，以「missing 開頭的 feature」和「參數數量的多寡」選擇的分水嶺：比如第一個模型有三個 missing，第二個模型沒有 missing，第三個模型有兩個 missing 等等。如此就可以綜合模型在不同情況下的表現

- Data Post-processing


- 將預測出來的機率填入 submission，分別以 'lr0'，'lr1'，'lr2' 欄來表示
- 接著用基於這個比賽的 metrics，我參考網路上使用 rankdata 的概念去提升 ensemble model 的表現，實驗結果表示確實可以提升 0.0001~0.00015 左右的表現

- Ensemble of three logistic regression models

- 在經過多次實驗性調整參數後，以 $\frac{0.4 \times \text{rank0} + 0.3 \times \text{rank1} + 0.3 \times \text{rank2}}{3}$ 當作預測的數值，最終在 private score 上獲得 0.5915 的成績

6. Summary

- 這次參與 Kaggle 的競賽跟以往的作業最不同的一點就是完全不限方法，以及怎麼處理資料，讓一切東西都有了不確定性及潛力，但比較可惜的是開始嘗試的時間有些晚，有一些課堂上教的深度學習的方法沒有用上。不過有把更多的心力擺在資料前處理跟特徵工程上，是我覺得學到最多的地方，有時候一個好的特徵真的比得過十個 dummy 特徵，希望未來自己還有機會參與像 Kaggle 這種競賽。還要講更深入的心得的話，就是我覺得「有時候方向比努力更重要」，參考別人的方法雖然會限制自己的想像力，但對於時間有限的時候可以少走一些彎路，在這個效率至上的時代還是挺必要的。但適時還是要給自己多一些想象力，才能真正在研究的領域找到新的道路。

Submission and Description	Private Score ⓘ	Public Score ⓘ	Selected
 0713332_submission.csv Complete (after deadline) · now	0.5915	0.58911	<input type="checkbox"/>