

109550035 楊賀弼

• GitHub link

<https://github.com/behebiiii/ML-Project>

• Model link

Model.pkl

 <https://drive.google.com/file/d/11gyniZNc9GogyBesy8P7BQPZbbIGEnOE/view?usp=sharing>

• Screenshot of my submissions

Submissions



You selected 0 of 2 submissions to be evaluated for your final leaderboard score. Since you selected less than 2 submission, Kaggle auto-selected up to 2 submissions from among your public best-scoring unselected submissions for evaluation. The evaluated submission with the best Private Score is used for your final score.

0/2

■ Submissions evaluated for final score

All Successful Selected Errors

Recent ▼

Submission and Description		Private Score ⓘ	Public Score ⓘ	Selected
	109550035.csv Complete (after deadline) · now	0.5913	0.58565	<input type="checkbox"/>
	submission.csv Complete (after deadline) · 8m ago	0.5913	0.58565	<input type="checkbox"/>

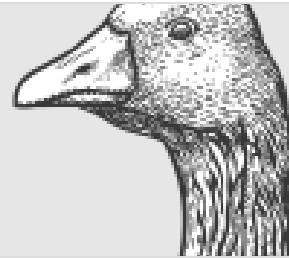
• Reference :

1.

TPSAUG22 EDA which makes sense ★★★★★

Explore and run machine learning code with Kaggle Notebooks |
Using data from Tabular Playground Series - Aug 2022

[k https://www.kaggle.com/code/ambrosm/tpsaug22-eda-which-makes-sense#Product-codes-and-attributes](https://www.kaggle.com/code/ambrosm/tpsaug22-eda-which-makes-sense#Product-codes-and-attributes)

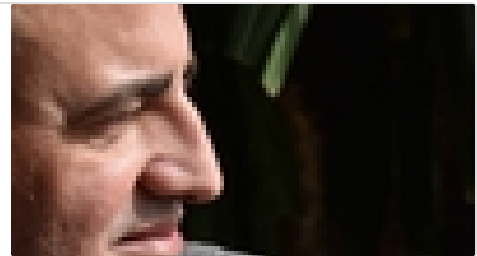


2.

LB 0.58978 Standing on the shoulder of giants

Explore and run machine learning code with Kaggle Notebooks |
Using data from Tabular Playground Series - Aug 2022

[k https://www.kaggle.com/code/maxsarmiento/lb-0-58978-standing-on-the-shoulder-of-giants?scriptVersionId=102785631](https://www.kaggle.com/code/maxsarmiento/lb-0-58978-standing-on-the-shoulder-of-giants?scriptVersionId=102785631)



• Brief introduction

針對這次作業我是用 Logistic Regression 的方式去實做的，在一開始的時候，我是有先參考 Reference 1 的內容(EDA)，去看說 這份資料集的內容到底是甚麼，然後才開始去實作內容。

在第一次做成功的預測之後，分數還沒達到baseline，所以我就上討論區看看大家都怎麼去improve的，所以我就看到 Reference 2 的文章，可以再利用 再加上 hidden features 來去把準確率提高。

詳細實作內容我有寫在註解裡。

• Methodology

資料前處理，將所有特徵標準化

```
scaler = RobustScaler()
```

因為看了 Reference 1 的分析 對於 attribute對failure的關西，只有 attribute 0,1比較重要，因為他們不是float 或是 int 所以做 one-hot encoded。

```
# one-hot encoded
Xt = pd.get_dummies(X_train, columns=['attribute_0'])
Xt.drop(["attribute_1"], axis=1, inplace=True)
```

再來就是必須把資料裡有空缺的補上，且針對不同的類別做不同的估測。

```
for feature in X_columns:
    if X_train[feature].dtype == "float":
        Xt[feature].fillna((X_train[feature].mean()), inplace=True)
    if X_train[feature].dtype == "object":
        Xt[feature].fillna((X_train[feature].mode()), inplace=True)
    if X_train[feature].dtype == "int":
        Xt[feature].fillna((X_train[feature].median()), inplace=True)
```

最後再做標準化

```
Xt= scaler.fit_transform(Xt)
```

Model

model 是用 Logistic Regression

```
model = LogisticRegression()
```

再利用 Grid Search 去調到 best parameters

```
param_grid = [
    {'penalty' : ['l1'],
     'C' : np.arange(0, 1, 0.01),
     'solver' : ['liblinear'],
     'max_iter' : [100, 500, 1000],
    }
]
```

```
clf = GridSearchCV(model, param_grid = param_grid, cv = 5, verbose=True, n_jobs=-1)
best_clf = clf.fit(Xt, y_train.values.ravel())
```

• Summary

做完這份作業後，我才了解到在做ML時對於資料集的分析是非常重要的，做預測並不難，但是你要找到對於預測有幫助的feature比較重要，其實也有一大部分時間是在看討論區裡面的人他們是怎麼發現，怎麼去找到這些可以去增進正確率的方式。也讓我了解到之後再面對這種feature一大堆的資料時，更可以知道怎麼下手。