GitHub link of your code

Reference if you used any code from other resources
I implemented some of the useful ideas in the following references:
https://www.kaggle.com/code/takanashihumbert/tps-aug22-9th-solution
https://www.kaggle.com/code/lgregory/optimize-score-7cf675/notebook
https://www.kaggle.com/code/vishnu123/tps-aug-22-top-2-logistic-regression-cv-fe/notebook
https://www.kaggle.com/code/trixytea/tps-aug-22-threshold-random/data?select=submission_no_splits.csv

Brief introduction
In this final project, I use logistic regression with some of the pre-processing methods such as SMOTE and WoEncoder, and kfold method to improve the accuracy of my models.

Methodology (Data pre-process, Model architecture, Hyperparameters, ...)
During the pre-processing step, I created two additional features called missing_3 and missing_5. I found the idea in Kaggle discussion that the missing product testing data might actually mean something important, and according to the analysis, the missing data might have larger effects on measurement 3 and 5.

After that, I separate features into category features and numerical features. Then, calculate the top 4 correlated features of each numerical features. Noticed that after spending lots of time adjusting correlated features, I followed the standard protocol in many references with top scores to set the correlated features of each product code in measurement 17.

Then, I use HuberRegressor to fill in missing column in each sample while the correlated features of the target feature are not missing. Otherwise, use KNNImputer with number of neighbors larger than 20.

After that, crate normalization function to standardize the numerical features to achieve better results. Also, encode the category features by WoEEncoder, though, I did not use them due to missing category features.

Then, implement Logistic Regression with StratifiedKfold because the data is highly class-imbalanced and we want to do oversampling with SMOTE. StratifiedKfold can

preserved the ratio of each class. Noticed that in each kfold, I use different kind of features, so it will be something like ensemble method.

Finally, save the models with pickle and inference by those models.

```
num_split = 2
skf = StratifiedKFold(n_splits=num_split, shuffle=True, random_state=0)
sm = SMOTE(random_state = 42, n_jobs = -1)
```

```
clf = LogisticRegression(max_iter=1000, C=0.0001, penalty='l2',solver='newton-cg')
clf = pickle.load(open(f'clf{n}_{i}.pickle', "rb"))
print(f"Successfully load clf{n}_{i}.pickle")
```

Summary

I found ensemble method extremely helpful of getting better score. I once weighted the prediction of 0.059002 and 0.059017, I get 0.59028, but I forgot how to reproduce. So, I will just submit the prediction with private score accuracy 0.059026.

Comparisons of different approaches

I found out that using SMOTE or not can highly affect the probabilities of the predictions. For example, without SMOTE, the probabilities tend to be lower (0.4) and with SMOT, the probabilities tend to be higher (0.55).
Therefore, I performed interpolation and get a better results with probabilities in the middle.

The state-of-the-art methods for this task

I think the state-of-the-art methods in this task is Logistic Regression. Neural network methods have less accuracy.

Thorough experimental results

Actually, I think kfold with smaller k might have better performance. (k=2 is better than k=5)

Ablation studies of different parameters

As stated above, with and without SMOTE affects the average probability value.
Also, KNNimputer will have less performance when the number of nearest neighbor is too low (k=3, 5). When k is larger than 20 (k=20~100), the accuracy stay the same and stop increasing.

Interesting findings or novel features engineering

I create additional features for missing measurement 3 and 5, performing

interpolations and ensemble methods.