I. Random Forest

1. Machine Learning Models Screening

Dataset was split into training and test set a fashion of 70%:30%. K-fold cross validation with 10 folds and 3 times repetition was implemented to the data to validate the model's performance. Then, a variety of machine learning models with default parameters were trained and fitted the data. Macro F1 score was chosen as the metric for this imbalanced and multi-classification problem. Macro F1 score is well-suited metric because it put equal weight to each class, so it can capture how well the model performs to predict minority classes. Otherwise, if weighted-average F1 score is used, a nearly perfect F1 score will be observed which failed to validate the model for predicting minority classes. A boxplot as illustrated in **Fig.1,** showcases the performance of each model. It appears ensemble learning models (included Random Forest, Gradient Boosting and Stacking) outperformed other models.

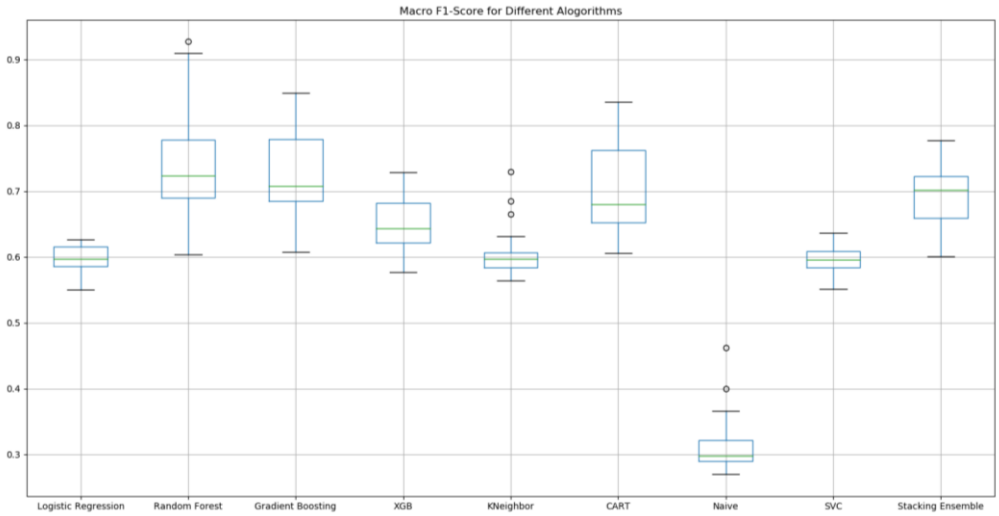


Figure 1: Machine Learning algorithm screening to show the promising model for this dataset

II. Workflow

An overview of workflow is demonstrated in the **Fig.2**below. The processed data was split into Dataset and Test Dataset. Dataset then subjected to K-Fold cross validation to split into training and validation dataset. Random Forest was used to train and fit training dataset. Because the dataset in this project is highly imbalanced and multiclass, different resampling techniques were implemented to see if they can enhance the performance of the models. Finally, random grid search and grid search were carried out to tune models' hypermeters.

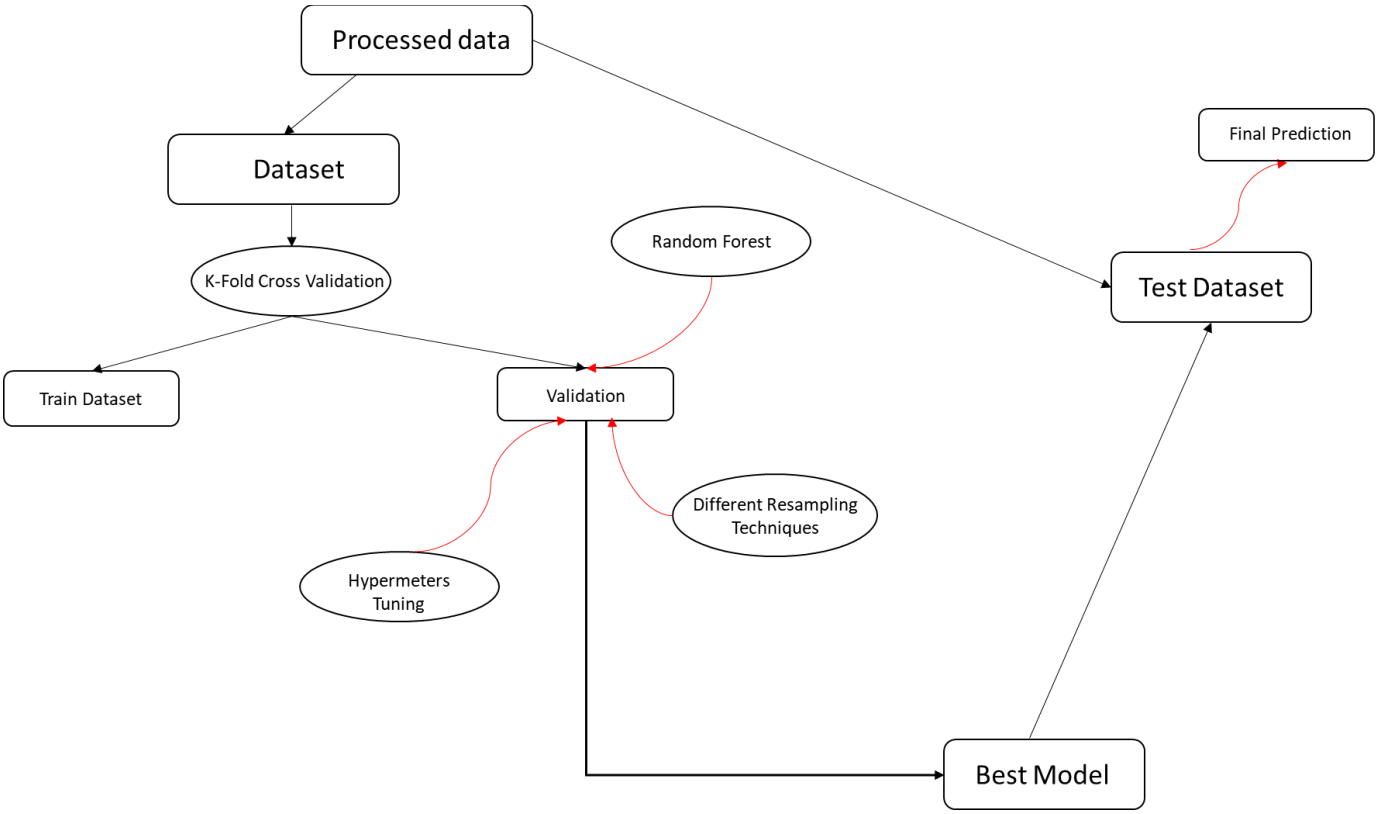


Figure 2: An overview of methodology implemented in this project.

III. Random Forest

1. Baseline Model

A random forest with default parameters was used to train with the train dataset. Its performance was evaluated by using the model to predict for the test dataset. As illustrated in **Fig.3**, the model obtained weighted average f1-score of 0.97. However, if the individual f1-score for predicting each class was examined, it shows that model can predict class 1, 2,4, and 5 with high accuracy. However, it does relatively poorer job to predict class 0,3, and 6. This is expected because the dataset is highly imbalanced. Therefore, macro average f1-score was chosen as the metric to evaluate the model's performance because it put equal weights for each class.

```
In [6]: print(classification_report(test_pred_baseline,y_test))
              precision    recall  f1-score   support

           0       0.17      1.00      0.29         1
           1       0.96      0.94      0.95       408
           2       0.99      0.95      0.97       622
           3       0.48      0.56      0.52        52
           4       0.99      1.00      0.99       864
           5       0.99      1.00      1.00      1112
           6       0.25      1.00      0.40         3

    accuracy                           0.97      3062
   macro avg       0.69      0.92      0.73      3062
weighted avg       0.98      0.97      0.97      3062
```

Figure 3: The classification report for the baseline random forest.

2. Features Extraction and Features Engineer

Features importance from original dataset was calculated by using Random Forest. As shown in **Fig.4**, "VSH", "PHI", and "W_Tar" were the most three important features in this dataset. Later, two different datasets that derived from the original dataset was used to train and fit two separate baseline random forest model independently. The difference between these two datasets is that one contains the most two important features while the other dataset consists of the three most important features.

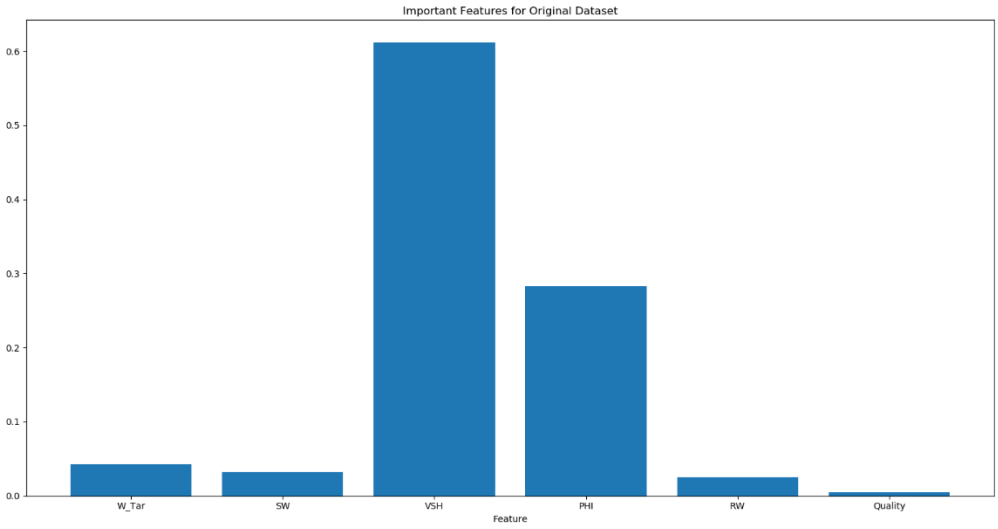


Figure 4: Important features of original dataset.

Consequently, two engineer features, volume of sand (VS) and hydrocarbon saturation (Shc), were computed by the equation below.

$$VS= 1- VSH$$

$$Shc=1-SW$$

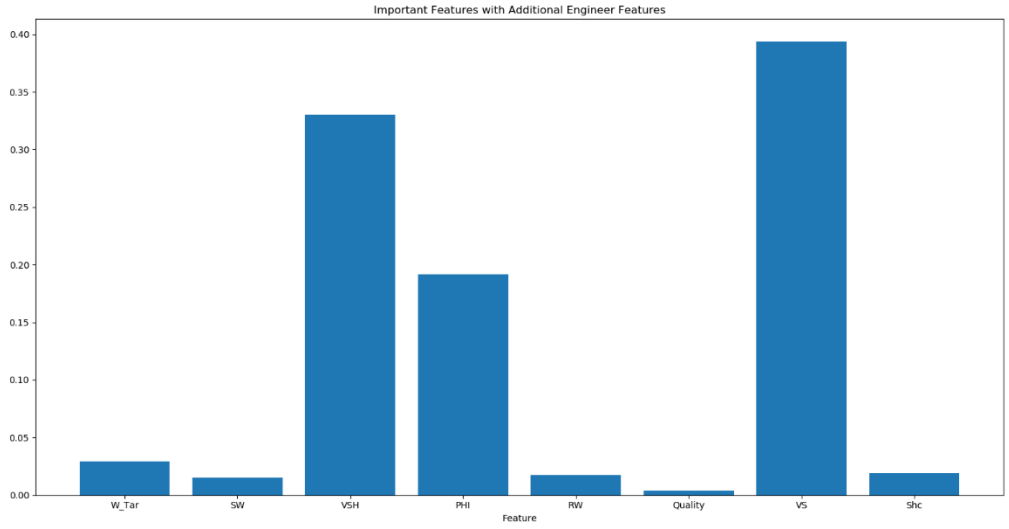


Figure 12: Important features after addition of engineer features.

Feature importance was computed again with this additional engineer features as illustrated in **Fig.5**. Noticed that VS is even more important that PHI and VSH. Shc has about the same importance as SW. A baseline random forest model was trained with this newly dataset. A summary of model with different choice and number of features were shown in **Fig.13**. Because there is not significantly difference in model's performance when fit the model with original dataset or with engineer features; so, the original dataset is used for the rest of this study.

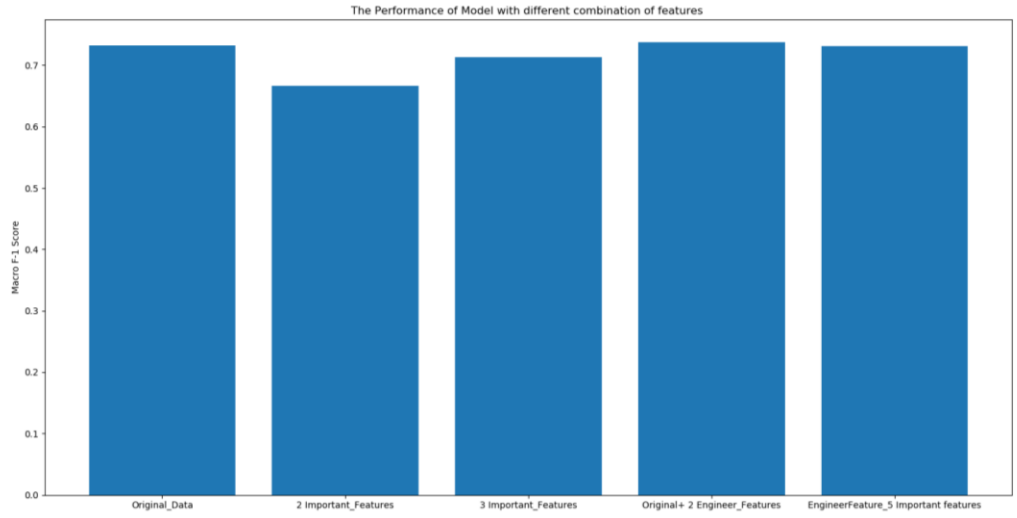


Figure 5: A comparison of model's performance with and without engineer features

3. Handle Imbalanced Dataset – Resampling Techniques

Four different oversampling methods, one undersampling method and 3 combination method were investigated to see if these techniques can enhance the model performance. RandomOverSampler, SMOTE, ADASYN, and SMOTEENN are oversampling methods. RandomUnderSampler is the only one undersampling technique considered for this project. Two combination methods include SMOTETomek ,SMOTE-RandomUnderSampler and RandomOverSampler-RandomUnderSampler.

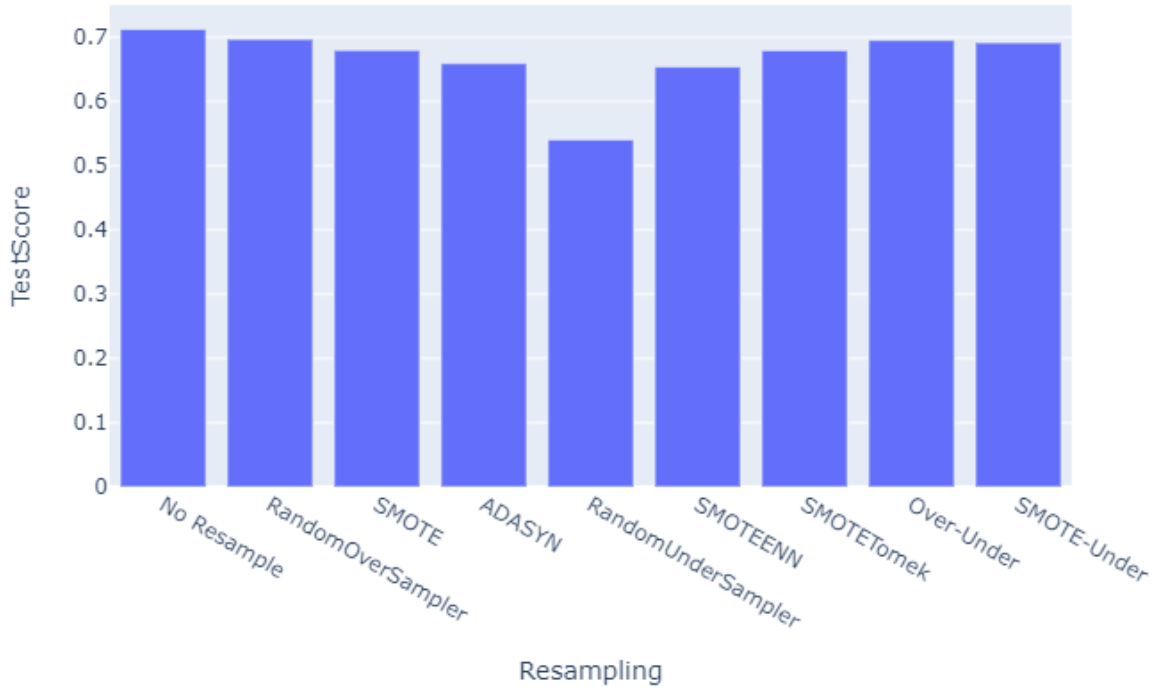
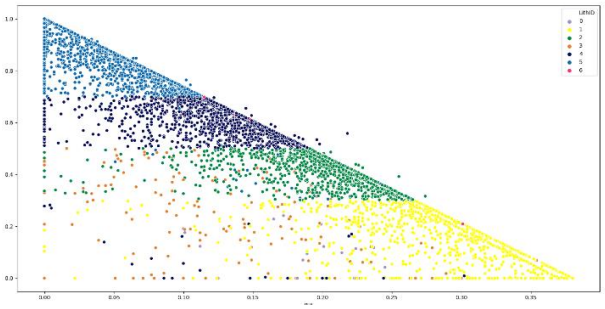


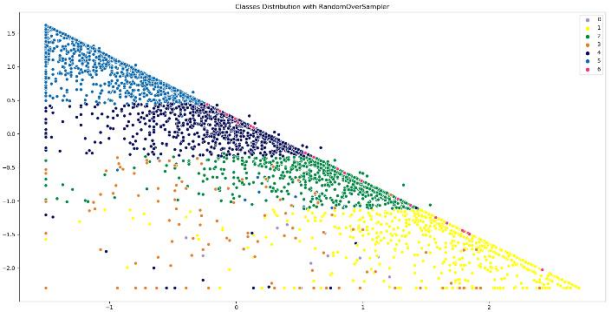
Figure 6: The comparison of model's performance in which different resampling techniques were applied to the original data.

Stratified K-Fold cross validation is utilized to split data into training and validation sets. Then resampling technique was applied to the training set only and use the validation set to evaluate the model. As **Fig.6** illustrated, the model seems to perform the best with the original data rather than the resampling data. RandomUnderSampler technique performed the worst. **Fig.7** shows how the data looks like after applying resampling technique. These imagines showed that minority classes in this project is very difficult to classify as its instances can spread out from one class cluster to another class cluster. This is understandable as the physical properties of these minority classes are very similar to these majority classes. Therefore, the resampling technique for this dataset does not improve this random forest model.

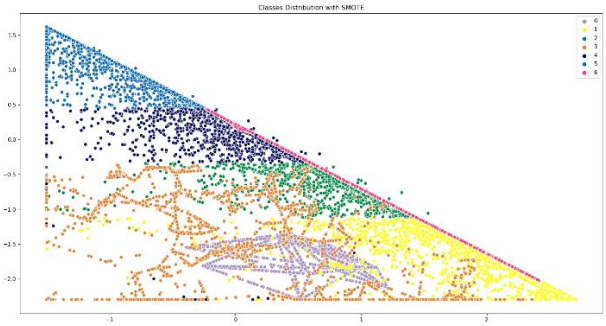
Original Data



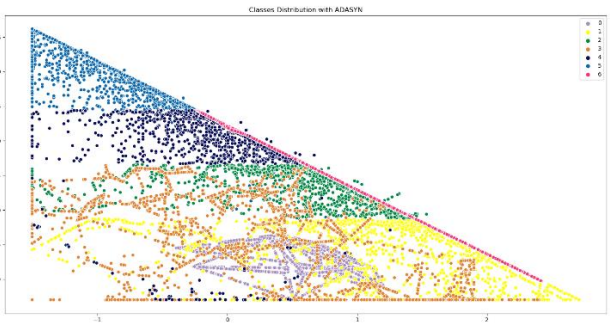
RandomOverSampler



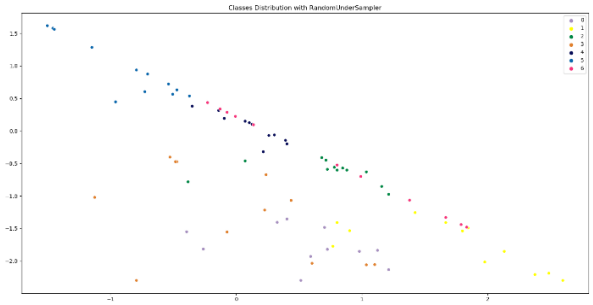
SMOTE



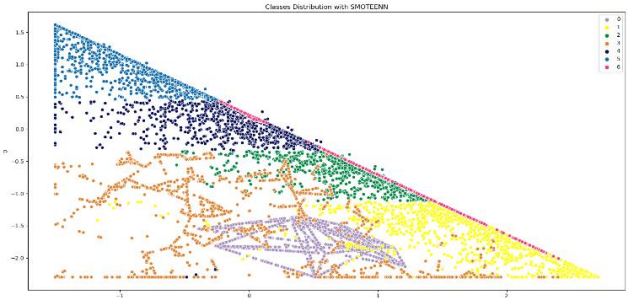
ADASYN



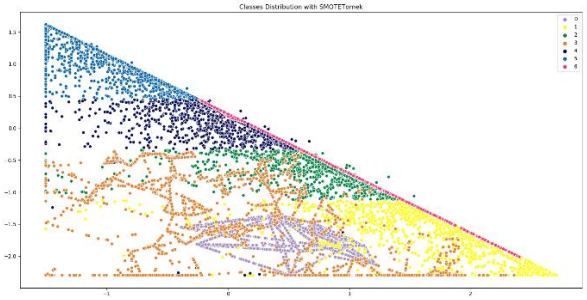
RandomUnderSampler



SMOTEENN



SMOTETomek



RandomOverSampler-RandomUnderSampler

SMOTE-RandomUnderSampler

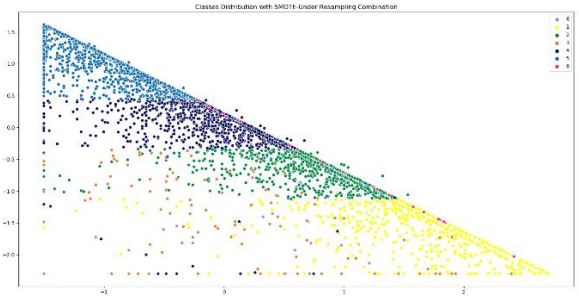


Figure 7: The Effect of Different Resampling Technique to the Original Data

1. Random Forest Hyper-Parameters Tuning

Hyper-parameters were fine-tuned by first apply RandomizedSearchCV to narrow the possible range of values for each random forest's hyper-parameters. Then, GridSearchCV was used to find the optimized model that then verified against the holdout dataset to evaluate its performance. **Table 1** summaries a set of hyper-parameters for this model:

Table 1: A Summary of hyper-parameters used for the final random forest model

Random Forest	N_estimators	Max_Depth	Max_Feature	Min_Sample_Leaf	Min_Sample_Split	Bootstrap	F1- Score for holdout
	557	50	2	1	3	False	0.74

A learning curve of the model with the finetuned hyper-parameters was constructed, as shown in **Fig. 8.** Within probably the first 200 samples of dataset, the F1-score increased steeply from 0 to 0.5. The overall trend is the continuous increase of F1-score metric as more samples were used. Therefore, the model's performance will be improved as more data are collected and used.

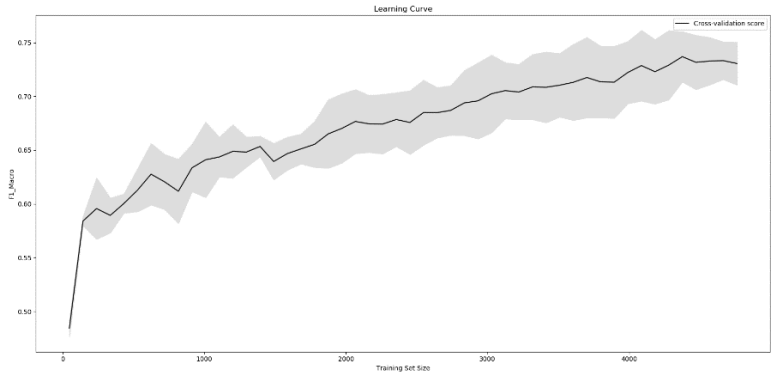


Figure 8: The learning curve of the random forest model with fine-tune hyper-parameters