TracHack Project Report

Team name:

RE&CLA

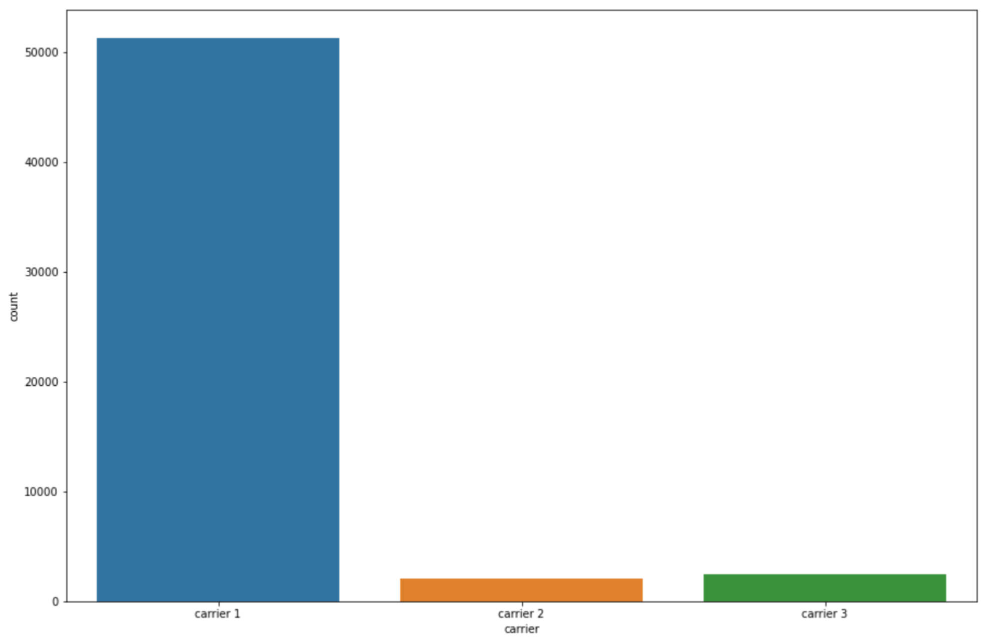
Team members:

Guohuan Li (z5315754)

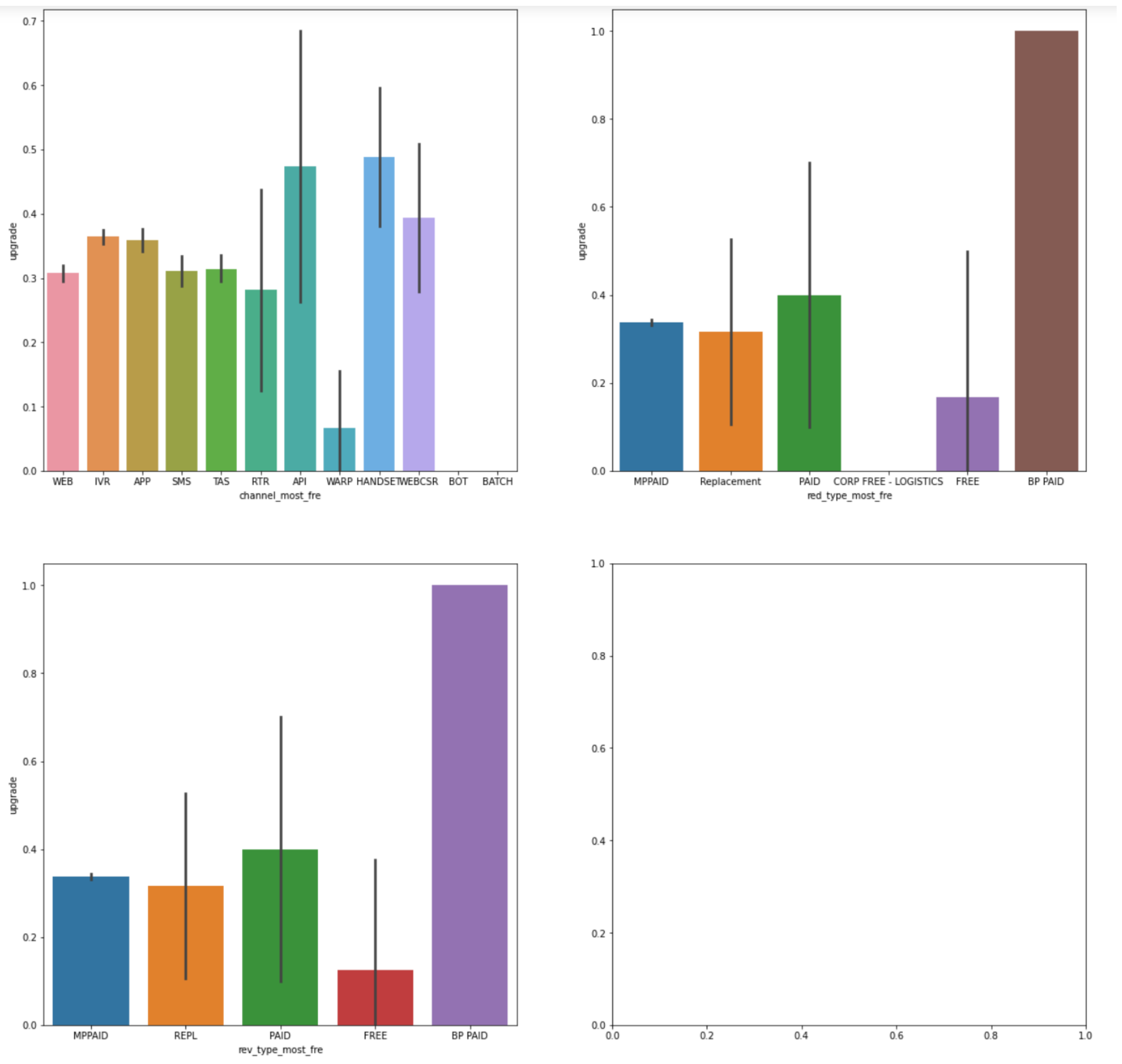
Introduction

Exploratory Data Analysis

EDA is an important part for understanding the overview of the data before doing data cleaning and feature engineering. In this project, we are given 9 different datasets and they have different data distribution. For example, most of the features in phone_info dataset is categorical features, while most features are numerical in network_usage. Meanwhile, most of the datasets include missing values and outliers with different degree. For target values dataset upgrades, the ratio between 'yes' and 'no' is roughly 1:4, which indicates the imbalance of targets distribution exists. For observing the distribution of categorical features and numerical features, we use bar plots for variable count and variable versus target to have a better understanding. For instance, plot bar plots for counting the variable in customer_info, we can easily see that carrier1 is dominated in the carrier features while in plan name the spread is roughly even. Extreme biased features may not give much useful information to differentiate the target values.



Since we have transferred the binary target into 0 and 1 represent, plotting variable versus the target can easily shows us what kinds of variables have larger ratio in upgrading. The following plots are some of the categorical features in redemption dataset after generating new features. We can see that in most frequent channel type usage, those people using handset and API have the larger percentage in choosing upgrade than other variables and same as other 2 plots. These plots can give us the directly understanding of what kinds of variables in different features contribute most in people choosing upgrade.



Also, when looking into the target values versus some missing features values, like 'plan name' in customer information, it appears that those ids contain missing values are having high ratio in upgrades. This is a sign that those missing values is informative for predicting the targets and can make use of them by treating them as a new variable in those features.

There are also some features that contain redundant meaning. Most of them are categorical features such as os_name, os_version and os_family in phone information dataset. In this situation, we can leave one of them instead of feeding all into the model. For intuition of the different features' relationship between target, we can guess that some feature may have positive or negative affect in deciding whether the person will choose to upgrade their plans or not. Like the person with less suspension records, using the mobile data in a large amount with newest popular phone brand may be more likely to choose to upgrade the plan. All this can help in the later feature generation to create useful new features.

Methodology

Data cleaning

Data cleaning is an essential part before doing feature engineering. It includes dealing with outliers, missing values, rare value in categorical features and scaling so on.

For dealing with outliers in numerical features, one of the methods can be considered is box plot and calculate lower fence and upper fence based on IQR and threshold. Setting fences based on 0.25 and 0.75 quantile of the data with 1.5 IQR is a common standard for detecting the outliers. Since we consider the extreme values may contain useful information in predict the targets, we set our threshold a high value, such as 5 IQR to leave more values representative. After detecting the outliers, one method called windsorization which replaces the outliers with upper fence and lower fence based on their positions in the range of that feature. Another method also can be applied is log transform which makes the extreme value become near to majority data range. This method is preferred since it remains the representation of the outlier in some degree instead of replacing by other values. Dealing with the outliers is helpful in the following feature selection step that using statistical tests which are under strong assumption on data distribution and very sensitive to the outliers.

For dealing with missing values, it should be careful whether we decide to drop it or not. In order to avoid losing any potential useful information, we decided to fill with meaningful values. For numerical features, we can fill the missing values by the mean or median, for categorical values, they can be represented by the most frequent shows values or treating them as a new variable call 'ismiss'. In most of our features, we implement in those methods. However, we must consider the features under the situation that cannot be simply replace by the normal values. For example, after merging suspension datasets with target datasets and generate the feature of suspension count, those id do not have any suspension records should replace by 0, instead of mean or median. Because for intuitive thinking, the person without any suspension is more likely to upgrade the plan. This way can be more representative for these kinds of people. Meanwhile, as mentioned before, some missing variable in categorical features values have higher ratio in choosing 'yes' so replace them as 'ismiss' may be helpful for providing more information for model prediction.

Dealing with the rare value in categorical features is like dealing with the outliers in numerical features. Some categorical features may have lots of unique variables with only few amounts. For example, the device types or OS names in phone information contain lots of unique variables. There are certain number of people still using old or not much popular devices. This causes the features contain lots of variables only occupy very small amount of the users. If we not dealing with them and directly feed into out model, it will make the dataset less representative with lots of noise and make the model tend to overfit the data. Therefore, we set a threshold which is 0.02, for detecting the rare values and grouped them as ‘other’. This also decrease the dimension of the features significantly if we want to do the one-hot encoding for these features.

Finally, scaling is also crucial for preparing the data before feeding into the model. Using scaling can transform the data into similar range which will prevent some numerical features dominant the contribution in distance based one just because their range is large. We have considered three popular scaling methods: Minmax scaler, Standardize scaler and Robust scaler. All these methods can transfer the data into a certain range but first two methods also put outliers into account when doing the scaling. This will make the scaled value into a very small range if the outliers are in large number and decrease the value difference between normal values, which will give the model harder to extract the different from the data. Meanwhile, Robust scaler can separate the outliers while doing the scaling, which makes the normal values represent their difference and make the outliers stood out. Since we have processed the outliers and want the data to be more representative of their different, minmax scale after log transformation is applied in here.

Feature engineering

After finishing the data pre-processing, Feature engineering is a way to mining useful information and generate new feature based on the current raw data. Good feature engineering that generates representative features can help making our model improve the performance in prediction.

For datasets that has unique line id, it does not require any group by method so simple calculation can be met. For example, in customer datasets, we generate a new feature by simply calculating the number of days between the redemption date and first activation date.

For the datasets with multiple repeated ids, it means the data contains multiple number activity of the person. In this case, we can use group by method to group all the activity by the line id and then calculate the count, mean, sum or most frequent variable that appears in the features. For example, in redemption dataset we can group by id and return the redemption count of the person, taking the mean of the person’s gross revenue or return the most frequent type of paying methods and so on. All these methods can give us a nice summary of the features.

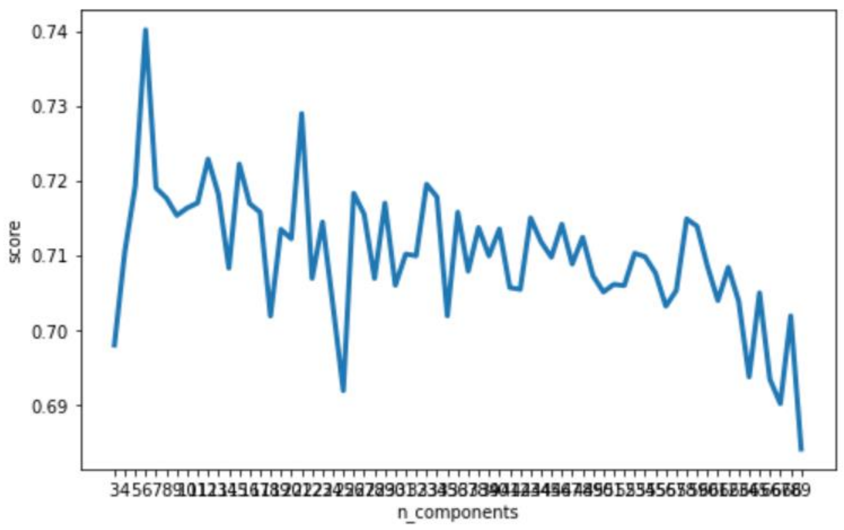
We can also generate the new features by calculating the ratio or doing subtraction. For instance, in network usage domestic dataset, we calculated the ratio between the mean of mms in and mms out data, giving us a mean ratio between these two features. This feature can be a great reflection of how the mms out and in ratio for the user and may provide useful information in separating the target if the ratio actually contributes to the choice of upgrade.

Categorical features encoding

It is essential for encoding the categorical features into numerical represent before it can feed into different models. In this project, we have implemented four different encoding methods: One-hot encoding, hash encoding, ordinal encoding, and binary encoding. One-hot encoding is a very popular encode method for nominal features since it is simple and not introduce the misleading order of the features while encoding so it is good for distance-based models. However, it also suffers dimension explosion when facing the features with lots of unique values since it represents. Therefore, it is not a good method of generate the features feeding into tree-based models.

Ordinal encoding is for encoding the features with natural order. This technique can also encode the features that has no natural order but only useful in tree-based model such as Decision Tree or Random Forest. Feeding these data into distance-based models may cause a misleading of the prediction.

Hash encoding can be used for feature dimension reduction since it can customize the number of the dimensions of the features. Here is the plot of f1 score versus the number of components we introduce to the features for RF model. We can see that number 6 generate the highest f1 score thus 6 is the most fitted number for hash encoding in this model.

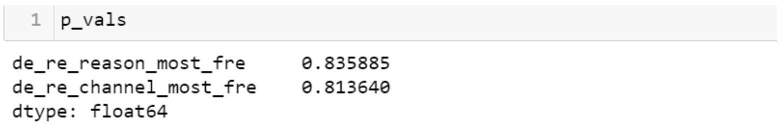


Binary encoding is another technique for encoding ordinal features. It works like the one-hot encoding but using binary number to represent the variables. Therefore, it will not introduce natural order but also not much suffer from dimension explosion as the one-hot encoding.

Feature selection

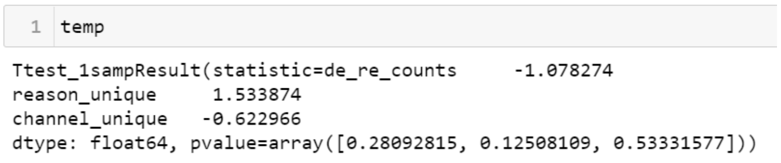
There are bunch of techniques for conducting feature selections. The ways we implement here includes statistical valid method such as chi-square test, student t-test and correlation plot, as well as non-statistical valid method like feature importance in Random Forest Classifier.

For conducting chi-test or t-test for features versus the target values, it is importance to clarify what kinds of null hypothesis we are trying to reject. For chi-test in categorical features, the null hypothesis is the relationship of the target values and the specific categorical features are independent. For t-test in numerical features, the null hypothesis is the means of 2 target types are same. The significant we set here is 0.05. For instance, we conducted chi-test and t-test for the features in dea_rea_info dataset, which is the merge between deactivations and reactivations data. The results come out that none of the numerical features and categorical features have significant p-values, which may indicate that these features may provide no useful information in differentiate the target. However, these tests are under strong assumption such as independence and distribution of the data. Also, these tests are sensitive to the outliers. Therefore, we must be very careful in inspecting the outcomes before we accept can the results.



```
p_vals
```

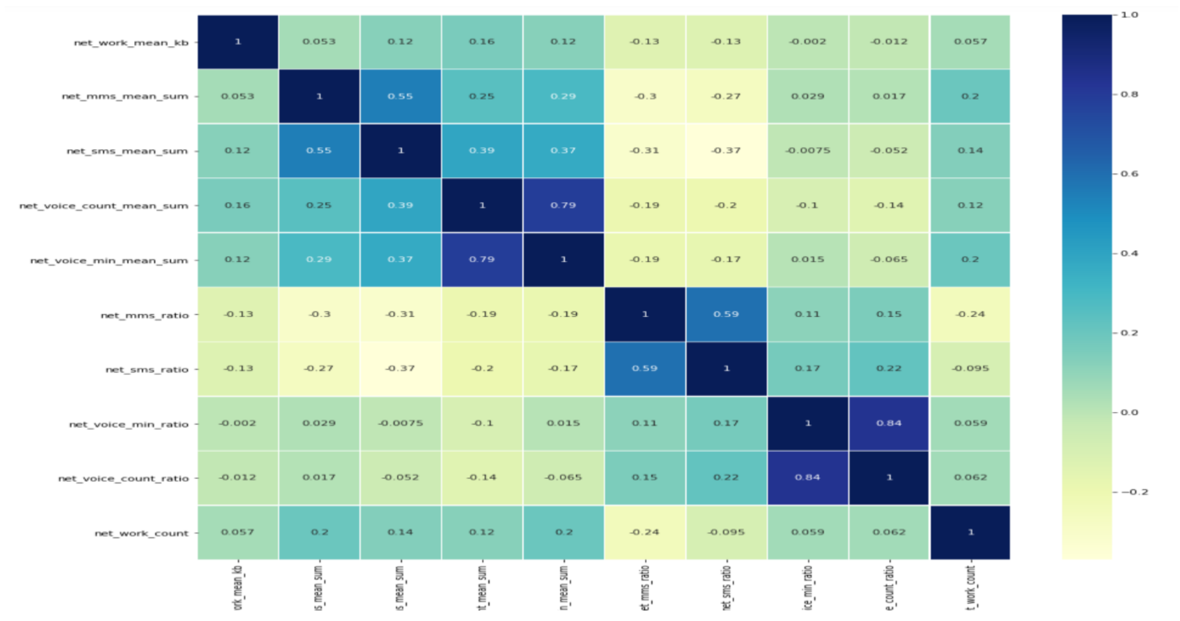
```
de_re_reason_most_fre     0.835885
de_re_channel_most_fre    0.813640
dtype: float64
```



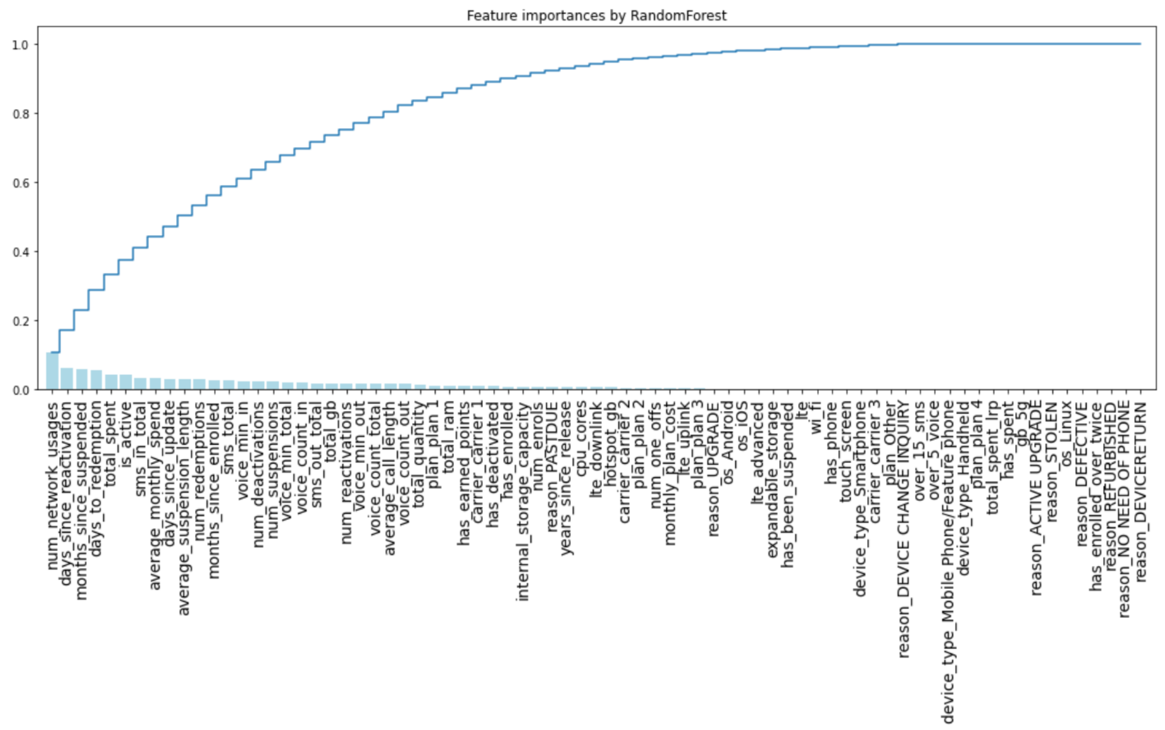
```
temp
```

```
Ttest_1sampResult(statistic=de_re_counts     -1.078274
reason_unique     1.533874
channel_unique   -0.622966
dtype: float64, pvalue=array([0.28092815, 0.12508109, 0.53331577]))
```

Another method we tried is using correlation plot to see the correlation between different features. When two features with high correlation in each other, that indicates that those features have strong linear relationship and may contribution similar result in predicting the target. Eliminate linear related features can help generating a better adjust R-square score model. Also, leaving only one of those features to feed the models can reduce the dimension of the data and introduce less potential noise from the features. For example, the following plot is the correlation relationship between different features in network information dataset after doing feature engineering. We can see that none most of the new generated features have small correlation score which indicate they can provide information for separate the target in various aspects. Some features have relative high correlation score but since we set the threshold as 0.9 so these features are remaining for the model training to prevent losing any potential useful information.



Another useful feature selection method is importance features in Random Forest Classification. This method is very useful to reduce the dimension of the features and select important features feeding into other tree-based models such as xgboost. However, this method is an impurity-based feature selection method which tends to select the features have large unique values. Therefore, the numerical features are preferred for this method and the results may be misleading. For the following result of the feature importance of the top 24 importance new features, we can see that most of the numerical features take top score of the importance while only few of the categorical features are listing in the top place. One categorical feature is_active is in top 5 position so wo consider this feature is a very strong feature in prediction. The categorical features encoding using here is ordinal encoding since the RF suffer from one-hot encoding and tree-based models do not affected by introducing the model.



For final feature selections because most of the ML models have built-in feature selection methods, so the features selected by the statistic tests only useful in parametric models such as logistic regression. For the following training, it seems that boosting methods performed best in this project so using the features in RFC feature importance as a reference would be a good choice.

Model training and tuning

Model training and tuning is the final part in the machine learning pipeline and giving the feedback of how we can improve the above methods. Here we tried different models like classic ML models Logistic Regression, SVM, KNN and so on, as well as ensemble learning models such as RF, xgboost, adaboost and so on. For tuning, using GridsearchCV is good for finding the best parameter. For training non-tree-based model, using binary or one-hot encoding then feed the features into the models and using GridsearchCV for getting the best parameters. For training boosting models, first we feed all the features with ordinal encoding or one-hot encoding in categorical features to the RF with GridsearchCV, then compare the results of using all features and top 20 features in different models.

Results

For model training, we have implemented different models of both traditional ML models (like KNN, SVM, LR) or ensemble learning models (Adaboost, Random Forest, XGboost, LightGBM) using different format of the features. For traditional ML models, the overall f1 scores are lower than the ensemble learning models using the same features, in average 0.1 different. The average f1 scores of the boosting models are 0.87 after tuning and the best model we got is LightGBM with 0.9 f1 score. We also tried using majority vote with 3 best performance ensemble models: LGBM, XGboost and Gradientboost and the f1 score is around 0.89. So, we decided to pick the LightGBM with its best parameter we got as our final submitted model.

Discussion

Comparing the results of different model performance, it is clear that ensemble learning models tend to having much better performances compared to the traditional ML models. This indicates that how powerful and robust the models are in dealing with datasets that contain complicated information.

Also, after trying different features, encoding methods and machine learning models. It appears that one-hot encoding will lower the performance in tree-based model compares to ordinal encoding or binary encoding. It is because the features space will become sparser and the power of representation of important features will be reduced.

Moreover, it is not always the case that using more features will increase the performance of the models because useless features can introduce more noise than contribution to separate the target. Like feeding all features into RF and using top 25 features importance selected features, the later model gets a better f1 score than the first one. However, we also should not rely on such feature selection methods too much. Such as statistical tests can not fit into the statistic invalid models and RF feature importance can bias to the numerical features only because they tend to have much more unique values. Keep trying different combination of the features is the only way to get the most powerful features and a better performance model.

Conclusion

In this project, our team have implemented different techniques in data cleaning, feature generation, feature encoding and model training. What we realize is how important the data mining in bringing up an outstanding performance in training the models. It is worth to do a thorough and comprehensive analysis of the data before generating any new features, instead of blindly generating new features. Moreover, the powerful performance of different ensemble models really stands out compares to traditional ML models in this classification problem. If we have more time, we will explore deeper into the dataset to try to mine more useful information of the data and may generate more strong features which can further improve our model performance.

Reference

Yimeng-Zhang-feature-engineering-and-feature-selection, Yimeng Zhang,https://github.com/Yimeng-Zhang/feature-engineering-and-feature-selection

Encoding categorical variables, kiwidamien, https://kiwidamien.github.io/encoding-categorical-variables.html

The prevention and handling of the missing data, Hyun Kang, https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3668100/