Fintech Final Project Report

- **Group 17**
  資工碩一 R11944024 黃秉茂 電機碩一 R11921113 陳品中
  資工碩一 R11922164 吳葦誠 資工碩一 R11922170 吳祐任

Data Pre-processing

Collecting

- n_day_range
  - To construct a comprehensive database for our analysis, we first identify all alert keys that have corresponding cust_id values and merge these into a single dataset. Next, we apply a date filter to the merged dataset, selecting only those data points that fall within a specific range of dates. This helps us to focus our analysis on a specific time period, ensuring that we are examining the most relevant information.
  - In our case, we have determined that a value of "n_day_range" equal to 5 strikes the best balance between having sufficient data to train a model and avoiding an excessively large dataset that may be difficult to process.
  - Comparison
    - If it is too small, the model may not have enough examples to learn from, resulting in poor performance. On the other hand
    - If it is too large, the dataset may become unwieldy, too large to process, and may contain a relatively small number of examples compared to the total size of the dataset. This may lead to the poor performance.
- merging
  - Due to the large volume of data, it is not practical to merge all of the CSV files into a single dataset at once. To address this challenge, we split the dataset into smaller batches and process each batch individually. This allows us to manage the data more efficiently and avoid overwhelming our memory with a dataset that is too large to handle at once.
  - After we have merged each batch of data with the appropriate alert keys and cust_id values, we append the resulting dataset to processed dataset. This allows us to gradually build up a comprehensive and fully-annotated dataset that includes all of the relevant information.
  - By splitting the data into smaller batches and appending the processed datasets, we can effectively manage and analyze the large volume of data in a systematic and efficient manner.
- data warranty
  - Each alert key in the dataset corresponds to at least one piece of data. It is possible for a single alert key to be associated with multiple data points, but there must be at least one data point for every alert key. Ensuring that every alert key has at least one corresponding data point is important for accurately analyzing and understanding the data.
- Label smoothing

  - binary label
    - a simple classification system where data is assigned to one of two categories, often represented as "1" or "0". Using a binary label allows us to quickly and easily categorize the data, but it may not capture all of the nuances and details of the dataset.
  - smoothing label
    - considering the proximity of the dates in labeling the data allows us to take into account the relative timing of the data points. FThis approach can provide a more detailed and nuanced understanding of the data, but it may also be more time-consuming and complex to implement.

  $$label = label * (1 - Clip(date_{alert\_key} - date_{cust\_info})/2))$$

Missing value

- Remove
  - To efficiently process and analyze the data, we first retrieve all relevant information using the alert keys as a starting point. Once we have gathered all of the data associated with the alert keys, we perform a thorough cleaning and preprocessing step to ensure that the data is of high quality.
  - Filtering
    - feature missing ratio
      - remove the entire features if they lack of 60% of data.
    - record missing ratio
      - remove the entire features if they lack of 60% of data.
    - days range
      - if the date of customer data are too far to the date of target alert key, we remove them.
  - Comparison
    - Remove missing ratio with 20%
      - Data is similar when only considering one data point with corresponding alert keys. Unfortunately, this causes a decrease in performance.
    - Remove missing ratio with 20%
      - only a very small amount of data will be removed, and we will impute a large amount of data, which may lead to unstable performance.
- Imputate
  - Numerical
    - median - easy manner with eough performance
    - KNN - good but very time-consuming
  - Categorical
    - most frequent - easy manner with eough performance
    - KNN - great but very time-consuming

Feature Selection and Pre-processing

- numerical
  - Normalization
    the mean of the numerical feature which is greater than 100
    - Standarlization
      $X_{std\_scaled} = \frac{X - X_{mean}}{X_{std}}$
    - MinMaxScaler
      $X_{mm\_scaled} = X_{std} * (max - min) + min$
- categorical
  - Labeling
    - Label Encoding
    - Onehot Encoding

Model

- XG boost
- Rabdom Forest Classifier
- K Nearest Neighbors
- Decision Tree
- SVM-Classification
- SVM-Regression
- DNN

Approaches

find parameters

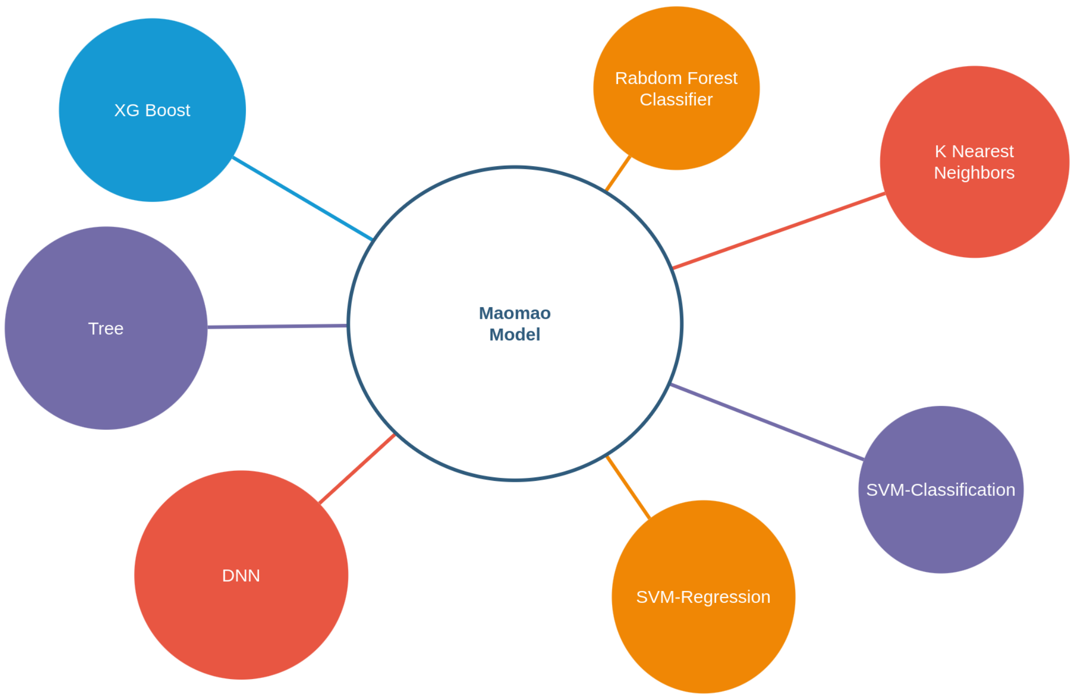
- GridSearchCV
  - find the best parameter with exhausted searching

imbalance data

- focal loss
  - focuses training on a sparse set of hard examples and prevents the vast number of easy negatives from overwhelming the detector during training
  - deal with imbalance data
    $FL(p_t) = \alpha(1 - p_t)^{\gamma} log(pt)$
    - α: imbalance data factor
    - γ: easy/hard sample

Post-processing

- Merging results predicted from different model
  - Weighted averaging probabilities with the same alert keys among different models
  - ignoring some results



Result

- Public Leaderboard

141	Strong Baseline	4	6	0.014859	12/26/2022 2:04:11 PM

  - publuc baseline

161	測試	2	5	0.011891	12/6/2022 3:40:31 AM

- Private Leaderboard

69	Strong Baseline	4	6	0.007057	12/26/2022 2:04:11 PM

Conclusion

- Removing missing values can improve performance, but if the amount of missing values removed is too low or too high, it can harm performance.
- Imputating missing value with KNN will do a great job, but it is too time-consuming. Thus, we back to imputate with median and the most frequent data, which are simple manner but leading to good enough performance
- Although we combine lots of models, we still give XGboost for the most attention. The reasons is as follwings. DNN tends to be over fitting when designed so complicated. SVM based are time-consuming. Tree based model like random forest and decision tree and KNN won't beat XGboost. In conclusion, we merge the result with different weights depend on their individual precision.

Discussion

- Since the data is filled with NaN, that is, it lacks of lots of data, the most important part is data preprocessing.
- Imputing missing values with KNN can be effective, but it is time-consuming. Therefore, we opt to impute missing values with the median and the most frequent data, which is a simple method but produces good enough performance.
- Although we consider multiple models, we give the most attention to XGboost. DNNs tend to overfit when they are too complex, SVM-based models are time-consuming, and tree-based models like random forests, decision trees, and KNN do not perform as well as XGboost. In summary, we combine the results of different models with different weights based on their individual precision.

work distribution

- 資工碩一 R11944024 黃秉茂 30%
- 電機碩一 R11921113 陳品中 20%
- 資工碩一 R11922164 吳葦誠 25%
- 資工碩一 R11922170 吳祐任 25%