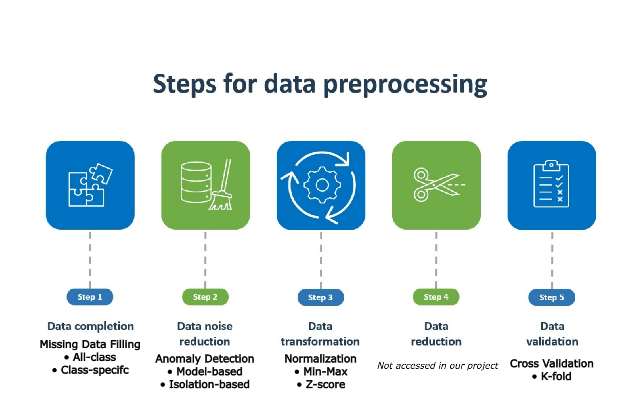
Data-Oriented Project Proposal

Tsung-Tse Tu

Faculty of Engineering, Architecture and Information Technology

The University of Queensland, Qld., 4072, Australia

1. Pre-processing approaches

*Fig 1. Steps for data processing (modified) [1]*

From the three techniques taught in class, imputation, normalization, and anomaly detection, our job is to find one that is necessary for the processing phase and implement the best practice under these techniques.

The imputation process is needed cause there’s a chunk of missing values in our dataset, without the potential values we couldn’t easily process the data. After a simple analysis of the dataset, each of the numeric columns has a missing percentage hovering around 9%-10% of all the missing data. From the equally separated pattern, we could try and impute the average of all values from the column they belong to; Similarly, the missing values of nominal columns tend to be around 9%-11% of all the missing data, so we can impute the most common feature in their column.

Anomaly detection is also a must-do because it’s hard to observe all these numbers, not to mention they’re in a fixed range. We perform it second because the result might influence what method we would use in the normalization process. Based on the lecture 7 slide [2], density-based, distance-based, and cluster-based are not that efficient when it comes to high dimensionality, so we will exclude them in future analyses. This leads us to model-based and isolation-based techniques. Model-based assumes that the data come from a regular Gaussian distribution, without enough background of these data points collected, it might not be the best method, but we will still try it in the cross-validation phase. Therefore, we will try importing the required library and fit the isolated forest model in our database to find any anomaly.

Finally, for the normalization process, there’s max-min and z-score normalization to consider. At first glance, all the data points seem to be in a fixed range, with numeric columns (0,1) and nominal columns (0,4), which is a great sign for implementing max-min normalization by using the MinMaxScaler; however, this method could easily be affected by outliers. So as a result, if we find outliers from our detection, we will go for z-score normalization by using StandardScaler

To further determine the optimal pre-processing combination, we first list down all the possible techniques we filtered, then we choose a cross-validation method to split our dataset into training and test sets. In our case, we will use the k-fold cross-validation method since it’s the most common one, and a 5-fold approach should be stable and promising enough for our small dataset. Lastly, we cycle through our potential combinations by repeating the CV procedure and evaluating the best strategy. Note that pipeline(scikit-learn) could be used to streamline the process of different combinations.

A diagram of a cross-section

Description automatically generated

*Fig 2. K-Fold Cross-Validations [3]*

1. Classification Techniques

Now we break down the procedure for applying the data's four classification techniques learned in lectures (decision tree, random forest, k-nearest neighbour and naïve Bayes). Before we delve into each technique, we will briefly outline the process of model selection.

First, consider different hyperparameters for each technique to train multiple models. Then, we cross-validate them with a given metric (possibly won’t consider accuracy since we seem to have an imbalanced class distribution). We may use ‘GridSearchCV’ or ‘RandomizedSearchCV’ (easier one) for hyperparameter tuning, and select the model with the best outcome, but also paying full attention to possible overfitting (compare training and validation data).

The decision tree method is a recursive procedure, we solve problems by breaking the problem into smaller pieces and solving it one by one. In other words, we use a flowchart structure tree model, where each internal node represents a test in an attribute, leaf nodes represent class labels, and branches represent the outcome. For hyperparameters, we would need ‘max\_depth’ [3, 5, 10, None] to represent the maximum depth of our tree; ‘min\_samples\_split’ [2, 5, 10] for the smallest number required to split an internal node; and ‘min\_samples\_leaf’ [1, 2, 4] to represent the least amount of samples needed for a leaf node, these two are important for adjusting overfitting scenarios. ‘criteria’ for the quality measure of a split. Note that ‘class\_weight’ might be a crucial parameter because we got an imbalanced class distribution.

Random forest is somewhat similar to the decision tree method, in which we create a forest of decision trees and integrate them. We will use ‘n\_estimators’ [50,100,200] to show the number of trees; max\_features [‘all’, ‘all\_sqrt’] for the best feature numbers to split; ‘max\_depth’ [3,5,10, None], min\_samples\_split’ [2, 5, 10], and ‘min\_samples\_leaf’ [1, 2, 4] as the other hyperparameters. Note that we can do ensemble learning of all four classification methods by combining all the output of the trees (majority voting), we need to ensure the overfitting problem. The weight of classifiers should also be considered if their a more important classifier than others.

A k-nearest neighbour, also known as KNN, is an instance-based algorithm that predicts samples based on the closest to the k-indicator. We choose ‘k’ usually by the elbow method. This technique could be highly effective against imbalanced data since the closer the neighbour is to the k-indicator, the more influence it had on the vote, but at the same time, we should make sure to process the normalization phase properly for the best result. We use ‘neighbours’ [3,5,11,19] to determine the number of neighbours we should consider and ‘weights’ [‘distance’, ‘uniform’] as distance-base for which the one closer to the k-indicator has more weight, otherwise uniform would do.

The last technique naïve Bayes is by using a probabilistic classifier based on Baye’s theorem assuming the independence between the features. From our numerical and nominal data, it’s best if we use different Baye’s functions for different feature types. Even though this technique doesn’t have parameters to handle imbalance, we can oversize (or undersize) our sample before running this technique to negate the imbalance effect. For our numeric data, we use’ var\_smoothing’ [1e-9, 1e-8, 1e-7] for Gaussian NB, and for nominal data, we use ‘alpha’ [0, 0.5, 1] for Bernoulli NB.

1. Evaluate and find the suitable metric

To evaluate our dataset, we use k-fold cross-validation. K=10 is a doable choice since we have 2180 data points, which means each fold would have 218 samples.

Because we have 10 folds, meaning each time a fold gets to be the validation set while the other 9 folds remain combined as the training set, we do this 10 times (each fold has a chance to be the validate set).

After all that, we add all the results and average the performance from each fold to get a single measurement. Now we can observe the model’s performance on unseen data.

Since we have an imbalanced dataset, we might as well use the F1-score or AUC-ROC for metrics. F1-score provides a balance between precision and recall, while AUC-ROC gives an aggregate measure of performance across all possible classification approaches. If the false positive and false negative errors have similar costs, the F1-score is preferable; otherwise, AUC-ROC would be better.

1. Remaining project timeline

The graph below is a timeline for part 2 of this project drawn by Draw.io (figure 3).

1. Acknowledgement

I used Grammarly to fix grammar and vocabulary errors and Draw.io for timeline graphing.

1. References

[1] Shehmir Javaid, Data Preprocessing in 2023: Importance & 5 Steps, digital image, accessed 15 September 2023, <https://research.aimultiple.com/data-preprocessing/>

[2] Miao Xu. "Lecture 7: ‘Anomaly Detection’ University of Queensland, Data Mining (INFS4203/7203), p.12-56. Accessed 15-September 2023.

[3] Prashant Sharma, published on February 10, 2022, digital image, accessed 15 September 2023, <https://www.analyticsvidhya.com/blog/2022/02/different-types-of-cross-validations-in-machine-learning/>

A screenshot of a computer

Description automatically generated

*Fig* ***3.*** *Project timeline*