Yanying Wu
Professor: Dr. Carsten Eickhoff
3/1/2021

Report of Machine Learning in Readmission Prediction Case

Introduction

In this report, a machine learning classifier to predict the Hospital's readmission's rate within 30 days is built. With the correct prediction (0 for not readmission and 1 for readmission) of the readmission rate will lead to approximately 76% of readmissions are avoidable. Also, it reduced the economic burden, which appropriately 43.1 billion per year. Our team build the classifier (Readmission.py), which is ready to train and evaluate. Here is a warming: the running time is going to be very long.

Random Forest (RF) ML algorithm is a type of bootstrap aggregation method in ensemble learning. Random Forest builds forest consisting of many decision trees, which is an algorism based on if-then-else rule for calculation. Every tree is independent and comes up with final decision of the testing sample. In the forest, the majority votes of class win, which play as a result of such random forest.

The readmission data set is huge, consisting of 28,861 rows (raw data) and 68 columns (features). Therefore, the most attractive reason for using random forest is that the model can handle huge data set with high dimensionality (many features). With Random Forest, we do not need to particularly select features for analysis for its significances. RF model can distinguish the significant levels of features, considering as one of the dimensionality reduction method. Also, in the script with readmission.csv dataset, it already implemented with Naïve Bayes, which is a supervised learning algorithm with the assumption that all features are independent and have no effect of each other. Therefore, each feature are relatively independent and it is good for using RF. What is more, with the sampling and pruning method used in construction, the likelihood of overfitting is significantly reduced and increase accuracy of the result. Most importantly, the result in our case is binarization, which only contain 0 and 1 for the readmission case. Random forest is a relatively easy model to build to solve classification with relatively accurate output. Last but not least, in limited time, the model needs to be trained with huge amount of data set, and Random forest allow us to easily build method of parallelization to accelerate the training speed and improve the scalability.

Therefore, the RF model is used as the major algorithm method. K-fold Cross Validation (K-CV) method is for evaluation. In the following construction part, each method and algorithm will be detailly explained.

Construction Details of the Model and Details' Explanations

In our readmission case, randomly select samples for every decision tree is used. Bootstrap Sampling is drawing sample data over times with replacement (Samples reappear in the next drawn sample sets). Every tree gets random independent dataset, including n data rows among all N rows of dataset (in the fit_multi_subset function) depend on the value of a certain features, and m features among all M columns of features (in the fit function). The use of the random two-way sampling (both rows and columns) method makes sure the randomness in each sample set and reduce over-fitting, which lead to high error rate in making decision.

The following step is finding split (find_split function in the code) locations in every decision tree. In the previous step, m features out of total M features are randomly chosen for each tree. Therefore, one out of m attributes is randomly chosen as the split attribute of the nodes and root.

Also, Classification and regression tree (CART) is used. Gini impurity, which use Gini function (in the code) to calculate, is used in CART to measure the likelyhood of being incorrectly labeled in randomly selected sample dataset. Gini index is ranging from 0 to 1, and the larger number the more complexity the sample is. Therefore, the best fit is that make the Gini_Gain as small as possible. By running through every feature and threshold, the Gini score calculate and store (in find_split function) for classification of left and right branches. Then the samples in the decision tree are split according to the Gini score. While the building tree passes the setting maximum node numbers, the blenches are cut and returned to the mode of features in both left and right blenches. This purpose of pruning, which means cutting additional blenches) is to reduce overfitting and reduce the training time. Following, a large number of trees are constructed to form a random forest.

For evaluation, K-CV method split the dataset into k folds, including k-1 folds as training set and 1 as testing set. K-CV effectively test the learning model and avoid the situation of over-learning and under-learning. In this case, we use K fold for validation of the model and prevention of overfitting. What is more, we also use the provided evaluate function with the calculation of F1 score (evaluate_return Function) for our own track of the performances of our model.

The parameter, including the number of trees in each forest, the nodes in each tree, and the number of k-folds. Another script with sklearn library, which is an advance learning library consisting of the Random Forest Classifier. The hyperparameter function is used to test out different number of parameters and compared their model evaluated score, and then to select the best parameter to test on our model. Finally, we choose 200 trees, 30 nodes and 3 folds to test in our model because of the balance between the speed of operation and the evaluation scores of outputs in Random Forest.

Further Improvement Discussion

One of the biggest problems with Random Forest model is that while dealing with noisy dataset, which means the dataset has irrelevant information (features), the model will have overfitting situation, which lead to a fault conclusion. In our case, this situation could also happen since 68 features are provided and not everyone is significant. Therefore, feature selection, which is a process of reducing the number of inputs (features) in training the model, is needed for improvement. One of the solutions for the feature selection, which might be more suitable for readmission case is that adding some new features, which might be the combination of the original features. For example, in the readmission case, the features about marital status have seven original features, including divorce, life partner, married, separated, single, unknown, and widowed. Those features can be simplified into two features, including single and not_single. With the feature selection, the training time can be significantly shortened, and the likelihood of overfitting reduced, and therefore lead to a more accurate prediction of classes.

What is more, for our model design, one of the shorting coming is that the pruning process is done by cutting the blenches as maximum number reached and returning to the mode of features in both sides. The pruning method belong one of the pre-pruning methods might lead to lower F1 scores in output (less accurate output) and high risk of overfitting. The better solution is method Reduced Error Pruning (REP) in post-pruning in readmission case. Post pruning is that after the generation of the over-fitted decision tree, pruning will be done for a simplified version of tree. For the first step, in a decision tree, replace a leaf with a subtree S. If the new-forming error (D), which formed between new tree and the original one, is equal to or less than the error generated by S, the subtree S will replace with a leaf (the blench is cut). Repeat the process in nodes of single tree until every tree is simplified. This method might take longer time, but it significantly improves the accuracy of prediction.

For our model design, we put randomly 1/3 of features into every tree, and do not test the effect of different numbers of feature in each tree, and therefore, get the best parameter of features in each tree. By choosing the idea parameter, the speed of the model training will improve, and the resting errors will be reduced. For testing the idea number, the sklearn.ensemble.RandomForestRegressor is used to write in another script. Then, max_features means the maximum features in every tree. Use the script to try some values to see the RF score, and then decide the parameter.

Also, the biggest problem with our model is that training the model is very time consuming. One of the models to consider is multithread function. Several tree can be trained all the same time in parallel regard of the CPU capability. The other solution will be using CUDA python. CUDA is a parallel computing model developed by Nvidia. CUDA speed up the speed by using GPU for computing parallelization. The method of using the CUDA Python is to install the CUDA Toolkit on the system with CUDA capable GPUs. The CUDA should be ready to use and just do some tiny change in our code, the tree can be trained in parallel with each other's, and the speed of training forest will be improved exponentially.

Conclusion

In this readmission case, the use of Random Forest, including randomly selecting bootstrap sample, using the CART (Gini impurity) to make split, pruning ideally give a better performance and F1 grade, compared with the baseline system given. The biggest problem is very time-consuming to run and the situation of overfitting. With the improvement in feature selection, REP, the modification of parameter, and adding parallel computing models, the model will have more accurate result and be train easily.