Prediction on Diabetes Patient Readmission via Tree-based Algorithms

Data Mining Techniques Course Project

Weiqi Weng, Yang Cai, Jiaji Dong, Hang Shang
College of Computer and Information Science
College of Engineering
Northeastern University

Introduction

In this project, we aim to build a tree-based model to predict whether a diabetic patient will be readmitted within 30 days given the patient's background and diagnostic information. First we do data preprocessing. We form the data set into a more approachable style and fix imbalance within label with a combination of Synthetic Minority Over-sampling Technique (SMOTE) and Tomek Link. Also we split the whole data set into training, validation and testing set. Then we run Gird Search Cross Validation and build four different tree-based models which perform similarly well on the training set. They are Decision Tree, AdaBoosted Decision Trees, Random Forest and Feature-selected Random Forest. Next we perform n-fold Cross Validation with $n = 5, 6, \cdots, 50$ and draw error bar plot to evaluate the model performance. For feature-selected random forest, we plot a bubble chart to further study its performance in terms of accuracy and time consumption. Finally we run the best model on testing set, compute confusion matrix and plot out statistics including sensitivity, specificity, F1 score and precision to evaluate the final model.

Algorithm

Data Preprocessing

To begin with, 4 features are dropped. Encounter ID is dropped since it's only an identity. Weight is dropped since 97% weight data are missing. Similarly, Payer Code and Medical Specialty, missing 52% and 53% respectively, are also dropped. Patient Number reveals that one patient may have several encounter recorded. Patient's readmission is affected and somehow decided by the previous diagnosis and treatment. In other words, one patient's duplicate encounters are dependent. To simplify the problem, we remove the duplicates and keep only the first encounter of each patient. Numerical features are all integers within a small range which turns out to be fine for tree-based models. Binomial features are encoded as binary.

Some nominal features are especially treated. Samples with 'unknown' (question mark in the data set) race are revised as 'other'. Admission Type ID 'Not Mapped' are merged into 'NULL'. Discharge Disposition ID 'Not Mapped' and 'Unknown/Invalid' are merged into 'NULL'. Admission Source ID 'NULL', 'Unknown/Invalid' and 'Not Mapped' are treated as 'Not Available'. As to Diagnosis 1, 2 and 3, we group the Diagnosis Code, icd9 code, according to which part of body the diagnosis is focused on. For example, 390 to 459 plus 785 are classified as Circulatory and 460 to 519 plus 786 are classified as Respiratory. (Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore, 2014) Then dummy indicators are added for each nominal feature.

Finally, we move to tackle the imbalance within label feature. After cleaning the data set, we have 90409 non-readmitted samples and 11357 readmitted samples, which means a model predicting all given sample as non-readmitted will result in a 88.84% accuracy on this data set. We fix this problem with a

combination of Synthetic Minority Over-sampling Technique (SMOTE) and Tomek Link, which is basically doing under-sampling and over-sampling at the same time. SMOTE plus Tomek Link is appropriate since it takes good control of the data set size. It increases the size to 110869 which is an proportional level. The balanced data set contains readmitted samples with a proportion of 41.2% and is divided into training, validation and testing set with 60%, 20% and 20% of the whole data set.

Decision Tree

The processed data set has 176 features including dummy indicators, of which only 8 are numerical. Decision tree comes naturally to this kind of feature setting since it's very powerful to crack data set with nominal features as the majority. We run Grid Search Cross Validation, which runs 5-fold Cross Validation for each hyper-parameter combination on training set, to select the maximum depth of decision tree (1 to 20) leading to best mean accuracy. In detail, for each maximum depth, we do a 5-fold cross validation, compute mean accuracy and find the value corresponding to best mean accuracy. Technically we can expand the full tree without pruning but in this way the model is taking risk of overfitting which can be verified in Cross Validation. The result shows that maximum depth 20 corresponds to best mean accuracy 86.35%.

AdaBoosted Decision Trees

One way to improve the model performance is ensemble method. In this part, we ensemble Decision Trees with AdaBoost Algorithm and run Grid Search Cross Validation to select the maximum depth, 1 to 20, of a single Decision Tree and the number of trees boosted, 50 to 500 with a step 50, leading to best mean accuracy. It has been revealed that maximum depth 20 of a single Decision Tree and base classifier size 400 leads to best mean training accuracy 90.82%. However, a drawback of AdaBoosted Decision Trees is time consumption. It takes almost two days for us to finish searching for best hyper-parameter combination.

Random Forest

In this part, we try to promote the model performance with Random Forest, which can be viewed as another form of ensemble. Just like the previous two methods, Grid Search Cross Validation helps again to figure out maximum depth of a single decision tree and the number of trees in the forest leading to best mean accuracy. The result shows that maximum depth 20 and forest size 350 leads to best mean training accuracy 89.82%.

Feature-selected Random Forest

Another way to kook at this problem is through feature selection. We have such a large feature pool that redundant and useless feature is inevitable. In this part, we perform a pipeline-style algorithm. We first train a decision tree and use the tree to do feature selection based on feature importance which is computed via Gini reduction. Then we feed the selected features into a Random Forest. As usual, Grid Search Cross Validation is performed to locate maximum depth (1 to 20) of the feature-selection Decision Tree and the number of trees in the forest (50 to 500 with a step 50) contributing to best mean accuracy. The framework is elaborated as follow.

```
procedure FEATURE-SELECTED-RANDOM-FOREST(data, range of max depth D, range of forest size N)
Input: data frame, range of max depth, range of number of trees in the forest
Output: the optimal max depth d* and forest size n*

    for all d ∈ D do:
        train a Decision Tree with d
        compute feature importance, select features and transform the given data
        for all n ∈ N do:
            train a Random Forest with n on the transformed data
            run 5-fold Cross Validation with the model
            compute mean accuracy
            update best mean accuracy due to different (d, n) combination
        end for
    end for
return the best mean accuracy, d* and n*

end procedure
```

In our case, feature selection varies because we get the feature importance corresponding to best mean accuracy which is related to a Random Forest model. However, in most cases, the process returns that 9 features are selected. They are Time in Hospital, Number of Procedures, Number of Medications, Number of Emergency, Number of Diagnoses, Indicator of whether the patient is transferred to home with health service, Indicator of whether the patient's age is between 50 and 60, Indicator of whether the patient's HbA1c test result is above 7 and Indicator of whether the patient's insulin status is down. Additionally, the best model has a mean accuracy of 92.47% with max depth 20 of each Decision Tree and forest size 400. Compared to Random Forest, we have 50 more base estimators but we lower the feature size from 176 to 9.

Cross Validation

In order to figure out which model performs better for our problem in general, we run k-fold Cross Validation with $k = 5, 10, \cdots, 50$, compute mean accuracy, standard deviation and report error bar plot for each model on the validation set.

For Decision Tree, we can see from Figure 1 that the mean CV accuracy fluctuates from 84% to 85%.

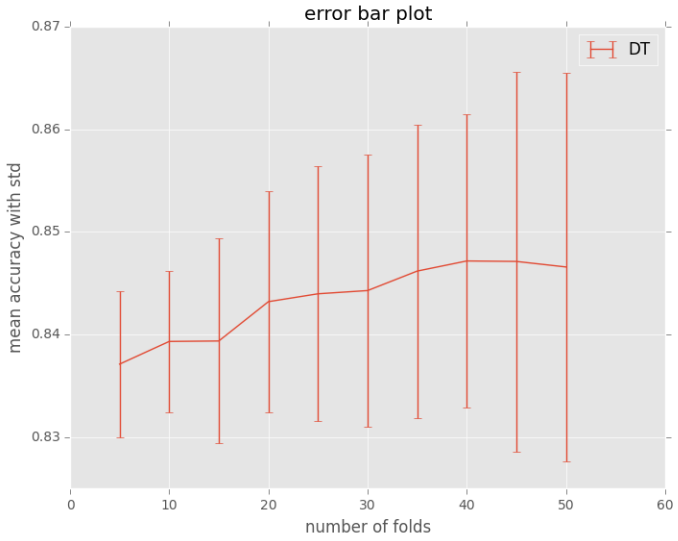

Figure 1. error bar of Decision Tree

AdaBoosted Decision Tree is more powerful than Decision Tree. It achieves an mean accuracy of 93.00% to 93.50%. Under some circumstance, the accuracy is approaching 94.50%. However, the accuracy fluctuates and is relatively unstable.

Figure 1. error bar of AdaBoosted Decision Tree

From the following Figure 3, we can see that the mean CV accuracy of Random Forest generally falls into $[0.855, 0.885]$.

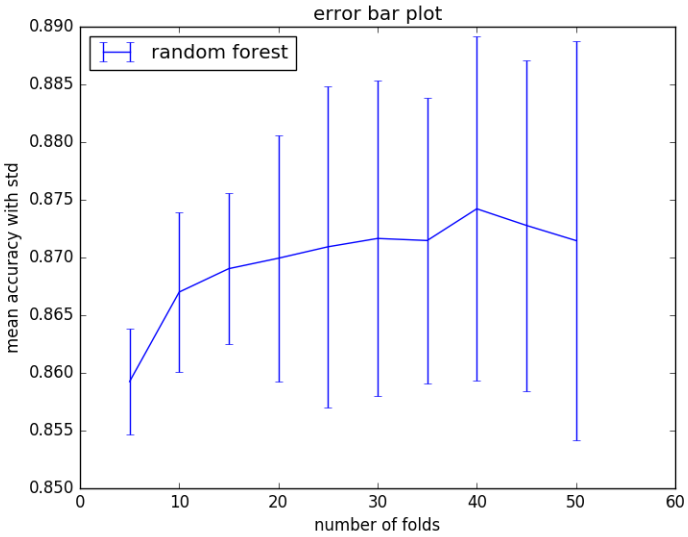

Figure 3. error bar of Random Forest

As to Figure 4, we can see that the mean CV accuracy of Feature-selected Random Forest generally falls into $[0.885, 0.890]$.

Figure 4. error bar of Feature-selected Random Forest

Each of the 4 models has its own pros and cons.

- Decision Tree: simple, fast; risk of overfitting, less accurate
- AdaBoosted Decision Tree: most accurate; much time-consuming, relatively unstable
- Random Forest: straightforward, generalizable; less accurate
- Feature-selected Random Forest: simple in terms of feature size, stable, fast; less accurate

Making a balance between performance and simplicity, we bring the last model to further testing, evaluation and discussion. The model also sheds some light on the data set.

Testing, Evaluation and Discussion

In this part, we first merge training set and validation set, which brings a new training data set containing 88695 samples. Then we pick out 9 selected features and train Feature-selected Random Forest on the merged training set to get training accuracy. Since accuracy is sensitive to the random effect, we run 30 epochs of estimation for $k = 5, 6, \cdots, 34$. In epoch k, we train the model, compute accuracy for k times and then get the mean training accuracy. To get an overall view of the model performance on the merged set, we plot the mean accuracy as a function of epoch in Figure 5. We can see that as we run more epochs, the mean accuracy plateaus at 91.80%.

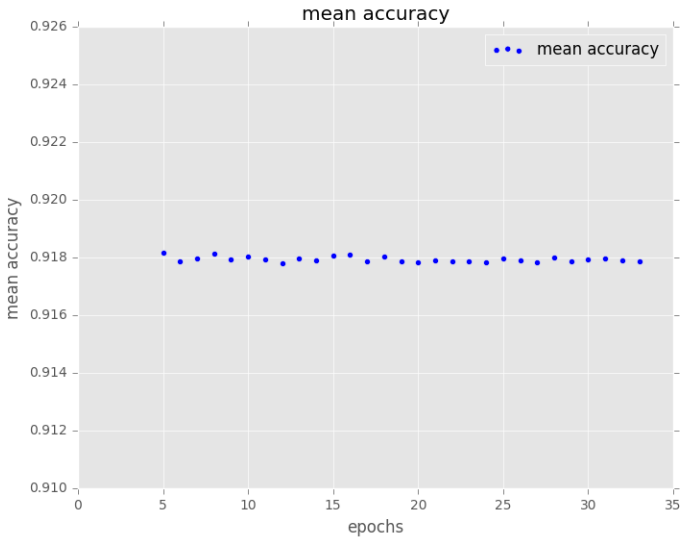

Figure 5. mean training accuracy of Feature-selected Random Forest

In addition, we draw a bubble chart for Feature-selected Random Forest as follow. The maximum depth of a single Decision Tree is mapped to x axis while size of the forest is mapped to y. The radius of each circle represents mean Cross Validation accuracy during Grid Search. Also the color of each circle measures how much time does it take for the algorithm to finish CV. The darker the color is, the more time training takes.

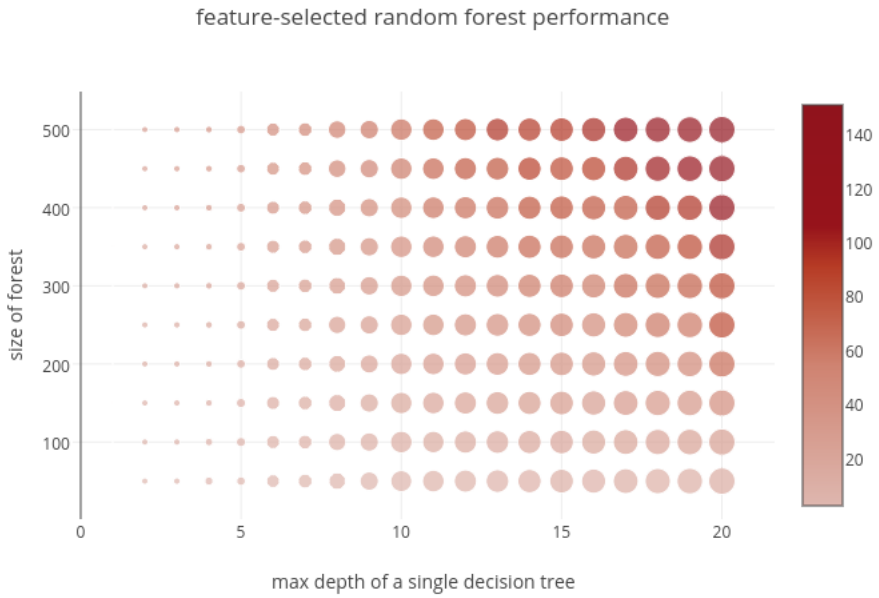

Figure 6. bubble chart of Feature-selected Random Forest

Figure 6 tells that

- For a fixed forest size, both accuracy and time-consumption increase as the max depth of a single tree grows.
- For a fixed max depth of a single tree, time-consumption increases as the forest size grows. However, accuracy increases insignificantly when the forest size grows. In this way, increasing the max depth of a single tree is relatively more effective than increasing the forest size in terms of accuracy improvement.
- the right part of graph reveals that training time can be saved by decreasing forest size and increasing max depth of a single tree at the cost of losing a little accuracy.

Then we run the model on testing data set, compute confusion matrix and plot sensitivity and specificity as a function of epoch. The model gets 94.14% accuracy on the testing set and the confusion matrix is presented as follow.

	Truth 0	Truth 1
Prediction 0	12935	2158
Prediction 1	121	6960

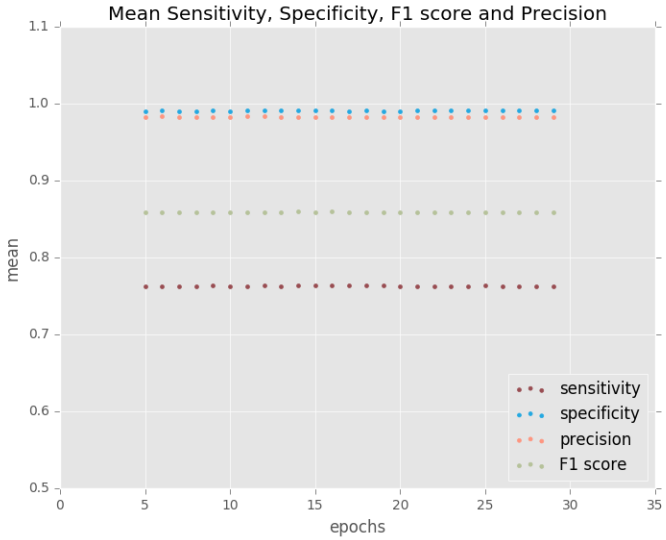

Figure 6. sensitivity, specificity, F1 score and precision

Again accuracy is sensitive to the random effect, we run 30 epochs for $k = 5, 6, \cdots, 34$. In epoch k, we compute sensitivity, specificity, F1 score and precision. Also we plot them as a function of epoch in Figure 7. Sensitivity converges to 0.77 while specificity is approximately 0.98. F1 score fluctuates around 0.86 and precision plateaus at 0.97. We can see that the model is similarly accurate when dealing with readmitted and non-readmitted samples. The performance is satisfactory, robust and stable, not to mention the significantly narrowed feature size.

In addition, the feature selection process based on Gini Reduction is to some extent consistent with work of Beata Strack's group. (Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore, 2014) In their paper, they have studied the effect of HbA1c test on diabetic patient's readmission while in our model HbA1c test result is often (not necessarily always due to random choice of feature) selected as a feature when building random forest. This means HbA1c test result really plays an critical role in readmission prediction. Similarly, according to feature selection result, people are encouraged to pay more attention to Time in Hospital, Number of Procedures, Number of Medications, Number of Emergency and Number of Diagnoses. The number of discharge type can be reduced with patients transferred to home with health service remained. Age group, 50 to 60, should be stressed since it's more informative. These features are more descriptive so at least we should make sure the high data quality of these features.

1 Reference

[1]Beata Strack, Jonathan P. DeShazo, Chris Gennings, Juan L. Olmo, Sebastian Ventura, Krzysztof J. Cios, and John N. Clore, "Impact of HbA1c Measurement on Hospital Readmission Rates: Analysis of 70,000 Clinical Database Patient Records", BioMed Research International, vol. 2014, Article ID 781670, 11 pages, 2014.