On-hands Study of Tree-based Methods

Chenyang DONG
Department of Mathematics
cdongac@connect.ust.hk

Tsz Cheung LO
Department of Computer Science
tcloaa@connect.ust.hk

Jiacheng XIA
Department of Computer Science
jxiaab@connect.ust.hk

Abstract

For the second mini-project, we are analyzing the application of tree-based methods. The dataset used include American crime dataset from previous result and 2 in-class Kaggle competitions. By comparing the results from project 1 and the scores of competitors in Kaggle, we have found that in General Tree-based methods can give lower Mean-Squared-Error and the results can be further optimized by various techniques such as Bagging and Boosting. But the most suitable technique varies according to the features of the dataset.

1 Introduction

For the past few lectures we have gone through more methods for doing regression and classification analysis. Among them we decided to go for a study on tree-based methods for the second mini-project. We use tree-based methods both on the America Crime Dataset and the In-class Kaggle competitions, and we find that tree-based methods are straightforward to implement and can give satisfying performance. In this report we present the results for both tests.

The rest of the report is organized as following: Section 2 describes the crime dataset and our previous results in project 1 using Lasso, and make comparisons between Lasso and tree-based methods. Section 3 describes how we used tree-based methods for combinatoric-valued drug data, their results and model comparisons. Section 4 presents our results on the in-class Kaggle Competition: Binary OneDrug Data. In the end there is a concluding part summarizing our findings and comments on tree-based methods.

2 Crime Dataset and Previous Results

In this section we present the tree based analysis for the crime data. Firstly we describe of the results using regression tree analysis and further optimized by bagging and boosting. In the end the results were compared with Lasso which we used on same dataset for project 1.

2.1 The American Crime Dataset

This dataset [1] contains the statistics of crime rate of 7 types in American cities from 1970 to 1992. Besides the name and abbreviation attributes there are 23 features like city population size, number of police, income per capita etc. There are more than 1000 valid data with complete information in this form. Since the features are of different scales, we preprocessed the data by first centering it along the axis to the mean, and component-wise scaling to unit variance.

2.2 Mini-Project 1 Results

In the first mini-project, we used various methods to analyze the crime data, and we found that Lasso does best in terms of both prediction and feature selection. Since the seven kinds of crimes are analyzed seperately, we focus on the crime type "Larceny" and we observed that other crime types gave similar results. Lasso gave a test set mean-squared-error (MSE) for around 0.06. We would use this as a benchmark.

2.3 Regression Tree Analysis

We first did a regression tree analysis from the tree analysis. We randomly split 90% of the data as training set and rest as testing set and got a mean-squared error as 0.1157. As we can see in the graph, regression tree also did well in feature selection, picking out city population, age structure and police forces, which is consistent with our conclusion in project 1.

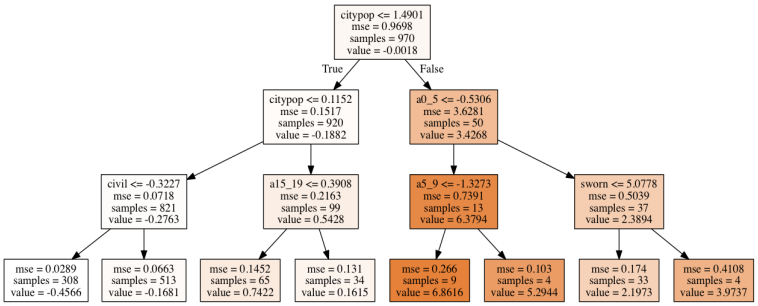

Figure 1: Regression tree on crime data

2.4 Boosting and Random Forest

To optimize the results from the regression tree, we tried random forest and gradient boosting methods. We found that gradient boosting gives a testing error of 0.04 and random forest using all features further reduces the results to 0.02. To compare the reason we plot the variable importance which indicates the reduction in Gini index if we eliminate this variable, and as shown random forest gives a good indication that *citypop* is the dominating feature. So in such kind of data where only one feature dominates, we can see that the Random Forest performs better.

2.5 Remarks

Despite tree-based methods generally perform better in the crime dataset, we can not yet conclude that tree-based methods would always perform better, since crime data is low dimension with only very few important features and they have different importance, and tree-based methods can intu-

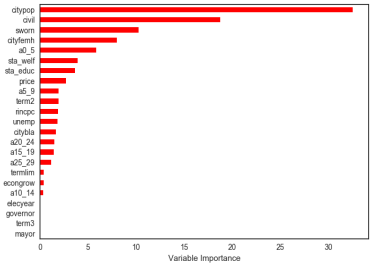

Figure 2: Importance from boosting

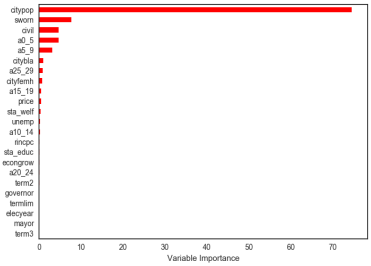

Figure 3: Importance from random forest

itively perform well in such cases. It remains to see the results in the in-Kaggle competition, whose data has some different features.

3 The Combinatorial Drug 20 Efficacy Data

In this section we concentrate on the analysis of tree-based methodologies. Main results of mean squared error are obtained through 5-fold cross validation. Regression tree, Bagging, Random Forest and Boosting are examined. Further comparison with Ridge and Lasso regression will be discussed as well. Finally, we will present the Kaggle competition testing results to elaborate on our empirical findings.

3.1 Data Description

The dataset contains 120 sample cell lines with 21 attributes in total and 20 instances are hidden for testing purpose. D1, D2, ..., D20 are 20 discrete features indicating random 4-level combinations of 20 types of drugs. The only real-valued attribute is the viability difference [1], which is the response variable in this study. Higher viability value implicates better efficacy of the drug combo.

3.2 Tree-based Model Analysis

3.2.1 Regression Tree

We performed regression tree analysis on the whole training dataset first and obtained a complicated regression tree with depth equal to 9. However, after we utilized 5-fold cross validation to simplify the tree and minimize the deviance related, we obtained a pruned tree with one split only. From the figure shown below, D3 is the variable with highest importance and thus the dataset actually has two main clusters separated by D3. However, despite from the intepretability, the cross-validation mean squared error of simple regression tree is indeed not satisfactory. After tree pruning, the overfitting problem was alleviated but the simple pattern of the pruned tree indicated that significant information loss exists in this model.

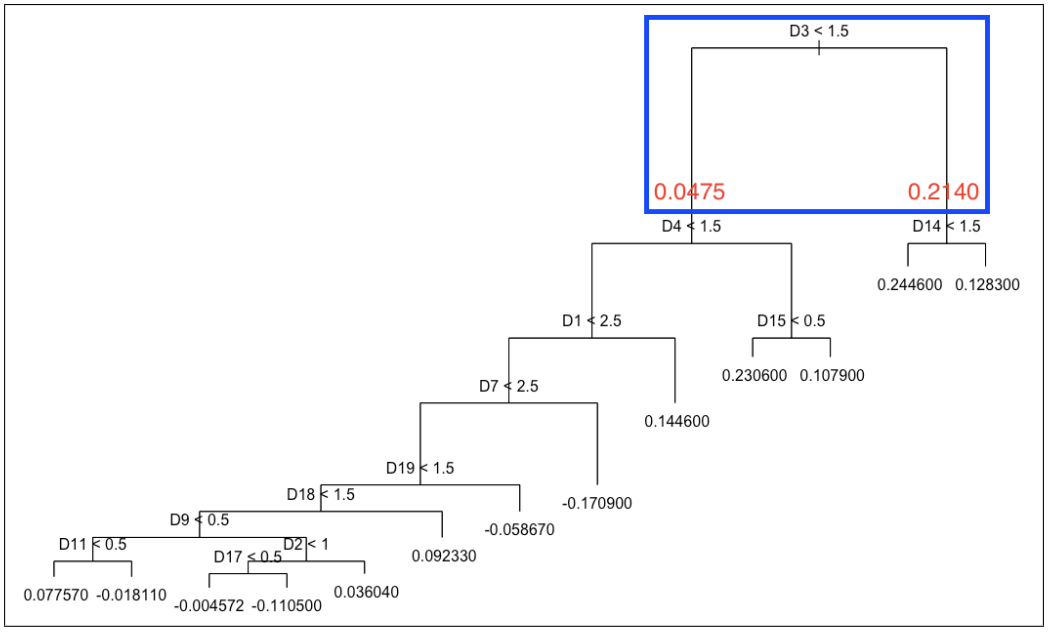

Figure 4: Full regression tree v.s. the pruned tree (within blue rectangle).

[1] Viability difference = normal cell - cancer cell, also known as therapeutic window.

3.2.2 Bagging and Random Forest

Due to the high intepretability of tree-based method but low prediction power, in this part we aim to check whether Bagging and Random Forest can be used to mitigate this problem and enhance the prediction power. Regression tree methods suffer a lot from high variance. Bootstrap aggregation for training datasets is one crucial technique for variance reduction. In this dataset, the number of instances = 100, which is moderately larger than the number of attributes = 20, so we expect that the variance reduction through random forest an bagging can be effective. In theory, random forest is expected to perform better than bagging because of decorrelation between random trees. Random forest achieves better variance reduction by reducing the predictor subset size. Three classical choice of the number of predictors are: $p, p/3, \sqrt{p}$. Therefore, we implement the random forest function three times with $mtry = 20, 7, 4$ respectively. Variable importance measure plots also show that D3 dominates other variables.

3.2.3 Boosting

Boosting to regression trees is another approach to improve the prediction accuracy. A large number of trees are built sequentially based on previous trees in order to fit the residuals iteratively. For this dataset, we set the shrinkage parameter $\lambda = 0.001$ small enough and the number of trees = 5000 to ensure the generalized boosted model is well learned. Aftering implementing the boosted model with $interaction\_depth = 1$ and the one with $interaction\_depth = 4$, the results are within our expectation. Both training error and CV error do not differ much after making the trees more complicated, which coincides with the one-split pruned tree previously mentioned.

3.2.4 Comparison with Linear Model

In order to compare tree-based models with linear models, we implemented Ridge and Lasso regression. Optimal shrinkage parameters are selected to minimize the 5-fold cross validation mean squared error. For Ridge, $\lambda^{\star}_{Ridge} = 0.02702$ and $MSE_{Ridge} = 0.01345$; For Lasso, $\lambda^{\star}_{Lasso} = 0.00864$ and $MSE_{Lasso} = 0.01352$.

3.3 Results and Discussion

In the following part, we will discuss our findings based on the results shown in Figure 5 and the testing performances in Kaggle competition.

	Training MSE	5-Fold CV MSE
Regression Tree (Unpruned)	0.00987	0.02408
Regression Tree (Pruned)	0.01754	0.01912
Bagging	0.00351	0.01798
Random Forest (mtry = 7)	0.00392	0.01789
Random Forest (mtry = 4)	0.00459	0.01800
Boosting (depth = 1)	0.01227	0.01707
Boosting (depth = 4)	0.01032	0.01693
Ridge	0.01587	0.01836
Lasso	0.01037	0.01416

Figure 5: Comparison between selected models.

This combinatorial drug 20 efficacy data is a typical dataset example with real-valued output and discrete inputs. According to the trade-off between interpretability and prediction accuracy, linear models are usually expected to have better performance than simple regression tree. More advanced tree-based methodologies may improve the accuracy significantly.

The original regression tree, which has tree depth equal to 9, suffers from overfitting problem with poor cross validation MSE. Although the pruned tree only involves D3, it outperforms the full regression tree anyway. The Kaggle testing MSE for pruned tree can reach 0.01655.

Bagging, as well as Random Forests, achieves excellent performance with respect to data training as shown in the result table. In terms of Kaggle testing, bagging and random forest (number of variables considered when splitting = 7) achieve a $MSE = 0.01066$ and a $MSE = 0.01079$ respectively. However, the percentage of variances explained by both approaches only range around 20%. Better techniques are needed to explain more of this dataset. From the perspective of decorrelation between individual trees, one may expect that Random Forests should display dominant patterns over Bagging according to the theory. A possible origin of this particular phenomenon is the number of important variables. As illustrated in Figure 6, out of twenty predictors, only D3 and D4 are considered to be important. However, in the setting of Random Forest, D3 and D4 are relatively less likely appear in an individual tree if the number of variables for splitting ($mtry$) is smaller. Thus it is more appropriate to apply Bagging for datasets with few important variables. Our empirical results actually corroborates the work led by Andy Liaw and Matthew Wiener [2].

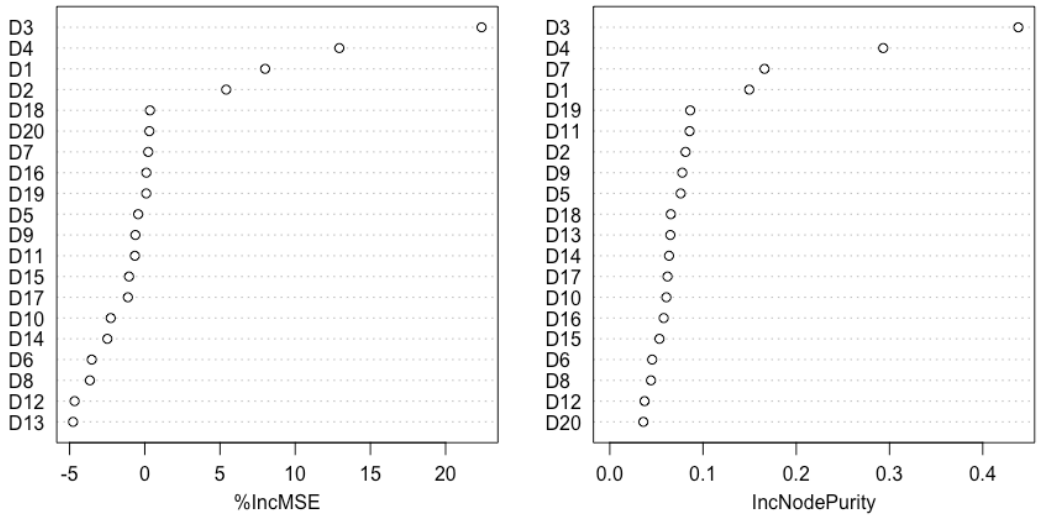

Figure 6: Variable importance plot.

For Boosting approach on regression trees, we achieved a testing $MSE = 0.01354$ in Kaggle using a model with $interaction\_depth = 4$. Since the prediction results coming from Boosting approach can be influenced by initial tree settings, we presume that the original tree structure generated by Boosting may be volatile. The inherent properties of dataset may be the reason why Boosting does not learn the model as satisfactory as Random Forests.

Linear models may give us the impression that they have higher prediction power than advanced tree-based tools as demonstrated in Part 3.2.4 and Figure 5. In fact, Lasso regression achieves a Kaggle $MSE = 0.01174$ slightly inferior to that of Bagging. We may conclude that for such dataset with real valued output and discrete inputs, there is no remarkable advantage of tree methods over linear models.

4 Results of Binary OneDrug

In Section 3, we have seen that, tree-based models have relatively satisfactory performance for discrete-valued input and real-valued output data. To further evaluate their robustness for similar dataset, we applied these tree-based models on the in-class Kaggle Competition: Binary DrugSensitivity2 to "compete" with other students' regression models.

In this section, we will first present results of Bagging, Random Forest and Boosting models on validation set. Then we will see how they perform on test set, i.e. in-class competition with ground truth. Since Section 3 has discussed a lot on the theory of tree-based models with similar dataset, we will not focus on the theoretical part but try to give straight results on these models.

4.1 Data Description

The OneDrug dataset [3] contains 642 cancer cell line samples, with 60 binary features as gene mutation status and 1 real-valued response, logarithmic IC50, to measure drug sensitivity. The basic problem is to predict the responses of these test samples based on their binary predictors/features.

Define $mtry$ as the number of random predictors that should be considered for each split of the tree. We are interested in the following 4 tree-based models:

1. Bagging
2. Random Forest with $mtry = \sqrt{p}$
3. Random Forest with $mtry =$ # of important variables
4. Boosting with $depth = 1$

4.2 Results on Validation Set

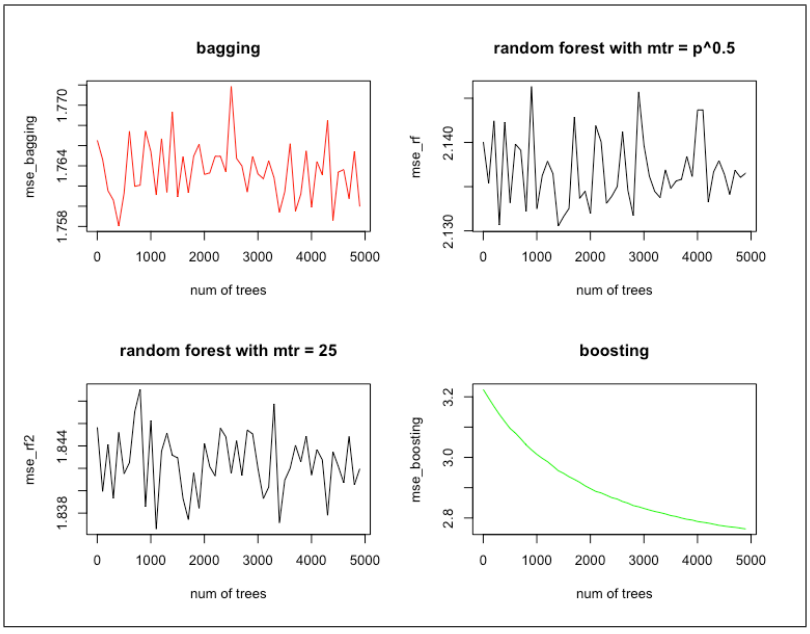

Figure 7: Mean Squared Error (MSE) v.s. Number of Trees

For this OneDrug dataset, except for the boosting model, it might be hard to observe the relationship between MSE and *Number of Trees*, but we are still able to see that, for this dataset, bagging performs better in the validation step. But what about the result for test set (Kaggle ground truth)?

4.3 Results on Test Set: Kaggle Competition

Table 1: Kaggle MSE & Ranking

Model	Kaggle MSE	Ranking (until 9 April, 2017)
Bagging, 5000 trees	3.12935	$6th$
Random Forest with $mtry = \sqrt{p}$, 5000 trees	3.16048	$8th$
Random Forest $mtry =$ # of important variables, 5000 trees	3.07900	$1st$
Boosting, 5000 trees	3.23467	$13th$

It is not surprising to see that, still, Bagging and Random Forest perform better than Boosting. But it is interesting to see that, Random Forest with $mtry =$ (# of important variables) performs much betther than with $mtry = \sqrt{p}$ and $mtry = p$ (Boosting). From the view of "Variable Importance", how to explain this result?

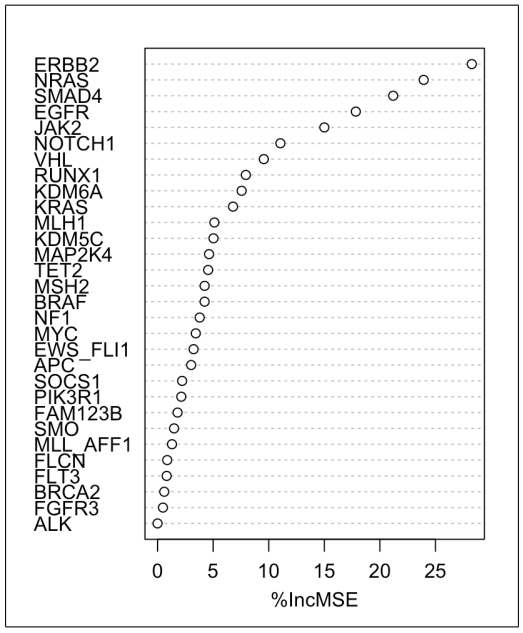

Figure 8: Increase in MSE when eliminating a variable

From the plot we can see that, there are 30 important variables (i.e %IncMSE ≥ 0). So comparing to $mtry = \sqrt{p} = 8$, $mtry = 25$ has a higher chance to include them as features in each tree iteration, while $mtry = p$ (Boosting) will include not only these important variables but also other "redundant" ones.

5 Conclusion

In this second mini-project, we apply several tree-based methods on 2 different types of datasets: American Crime dataset with real-valued input, 2 discrete-valued input datasets (ComboDrug and OneDrug). For American Crime dataset, we have found that in general, tree-based methods can give lower Mean-Squared-Error than traditional Linear Regression model via comparing the results from our mini-project 1.

For discrete-valued input data, we conclude that Bagging and Random Forest may perform better than other tree-based models like pruned trees, boosting, etc. Nevertheless, the most suitable technique should vary according to the features of the dataset.

To further study Bagging and Random Forest, we have found an interesting result: Since the only difference between these 2 models lies in the number of variables selected in each iteration, a concept "Variable Importance", a measure of the importance of the predictor variables, attracts our interest. From repeated experienments, the variable importance measures may vary from run to run, but the ranking of the importance is quite stable. Therefore, under the circumstance that only a small fraction of the features of the dataset are dominately important, Bagging is much more likely to perform better than Random Forest. This is quite reasonable since Bagging considers all predictors in each iteration when building the model.

Acknowledgments

We thank Professor Ed K. Chow, Ph.D. from Cancer Science Institute, NUS, for providing the Combinatorial Drug 20 Efficacy Data.

We thank Professor WANG Jiguang, Ph.D. and Dr. JIANG, Biaobin for providing the Drug Sensitivity dataset and initiating the Kaggle contest.

References

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Liaw, A., & Wiener, M. (2002). *Classification and regression by randomForest*. R news, 2(3), 18-22.

[3] Knijnenburg, T. A., Klau, G. W., Iorio, F., Garnett, M. J., McDermott, U., Shmulevich, I., & Wessels, L. F. (2016). *Logic models to predict continuous outputs based on binary inputs with an application to personalized cancer therapy*. Scientific reports, 6.