Assignment1: Decision Trees

Xiao Liu

September 17, 2014

1 Dataset Description

The dataset is from UCI machine learning directories. These are real word data collected by Paulo Cortez (Univ. Minho) and Sérgio Moro (ISCTE-IUL) [3]. These real-world data were collected from a Portuguese marketing campaign related with bank deposit subscription. The business goal is to find a model that can explain success of a contact, i.e. if the client subscribes the deposit. Such model can increase campaign efficiency by identifying the main characteristics that affect success, helping in a better management of the available resources (e.g. human effort, phone calls, time) and selection of a high quality and affordable set of potential buying customers. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product (bank term deposit) would be (or not) subscribed.

There are 45211 instances in the original data set and we applied 10%, that is 4521 instances, for this task by random sampling. Basically, this is a binary classification task that we use in total 16 attributes to predict "**Whether a person will subscribe the deposit**". The 16 input variables/attributes and their details are listed in Table 1.

Table 1: Attributes and Description

Attributes: Descriptions
age : numeric
type of job : categorical: "admin.","unknown",...,"unemployed"
marital status : categorical: "married","divorced","single"
education :categorical : "unknown","secondary","primary","tertiary"
has credit in default? : binary: "yes","no"
average yearly balance, in euros : numeric
has housing loan? : binary: "yes","no"
contact communication type : categorical: "unknown","telephone","cellular"
last contact day of the month : numeric
last contact month of year : categorical: "jan", "feb", ..., "nov", "dec"
last contact duration, in seconds : numeric
number of contacts performed during this campaign : numeric
number of days that passed by after the client was last contacted : numeric
number of contacts performed before this campaign : numeric
outcome of the previous campaign : categorical: "unknown","other","failure","success"

2 Model Fit

We conducted experiments to evaluate the performance of the decision trees under varying conditions for classifying the people into two categories: "yes" for "Subscription" and "No" for "Non-Subscription". In order to build a prediction model and test it, we separate the data into the

“Training Set” and “Testing Set” by ratio of 7:3. We first fitted the data in the training set with an unbounded tree by setting up the elements in the model as shown in Table 2. Then we used cross-validation to prune this large tree with 47 splits based on the the relationship between the corresponding errors and the number of splits.

2.1 Full Tree

To grow a full tree, our algorithm made use of the Gini criterion for choosing splits, with 2 observations necessary in each split. The default setting for the parameters are indicated in Table 2. We use 70% of the data, that is 3181 instances, from the complete data set for training this tree. As a result, we grow a tree with 47 splits to achieve the stopping criteria, that is Complexity Parameter=0 as we configured.

Table 2: Settings

Goodness of splits	Gini Index
Stopping criteria	All leaves are pure or all leaves contain less than 2 samples.
Class assignment	Yes or No

Table 3: Splits Results

Complexity Parameter	nsplit	error	xerror	xstd
0.029255319	7	0.8617021	0.9654255	0.04769294
0.018617021	9	0.8324468	0.8936170	0.04610426
0.013962766	11	0.8138298	0.8989362	0.04622502
0.013297872	13	0.7579787	0.9095745	0.04646501
0.010638298	**17**	**0.7313830**	**0.8989362**	**0.04622502**
0.009308511	19	0.7101064	0.9308511	0.04693905
0.007978723	22	0.6914894	0.9361702	0.04705634
0.006648936	24	0.6755319	0.9202128	0.04670301
0.005319149	29	0.6622340	0.9202128	0.04670301

2.2 Pruned Tree

As can be seen in Table 3, pruning of the decision tree is necessary since the tree is overfit. To make it clear, we also plot the relationship between the Relative Error and number of Split in Figure 2. Obviously, we can find that, the Relative Error gets the minimum value when there are in total 17 splits and the corresponding CP value is 0.010638298. Taking advantage of this information, we pruned the full tree and demonstrate the pruned tree in Figure 3. In the pruned tree, there are in total 17 splits which is much less than the fully developed tree.

2.3 Random Forest

Bagging and boosting methods can increase the performance of classifiers. Random forests use a modified tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. The reason for doing this is the correlation of the trees in an ordinary bootstrap sample: if one or a few features are very strong predictors for the response variable (target output), these features will be selected in many of the B trees, causing them to become correlated [1]. We used random forest classifier as our bagging method to elevate the final predictions accuracy.

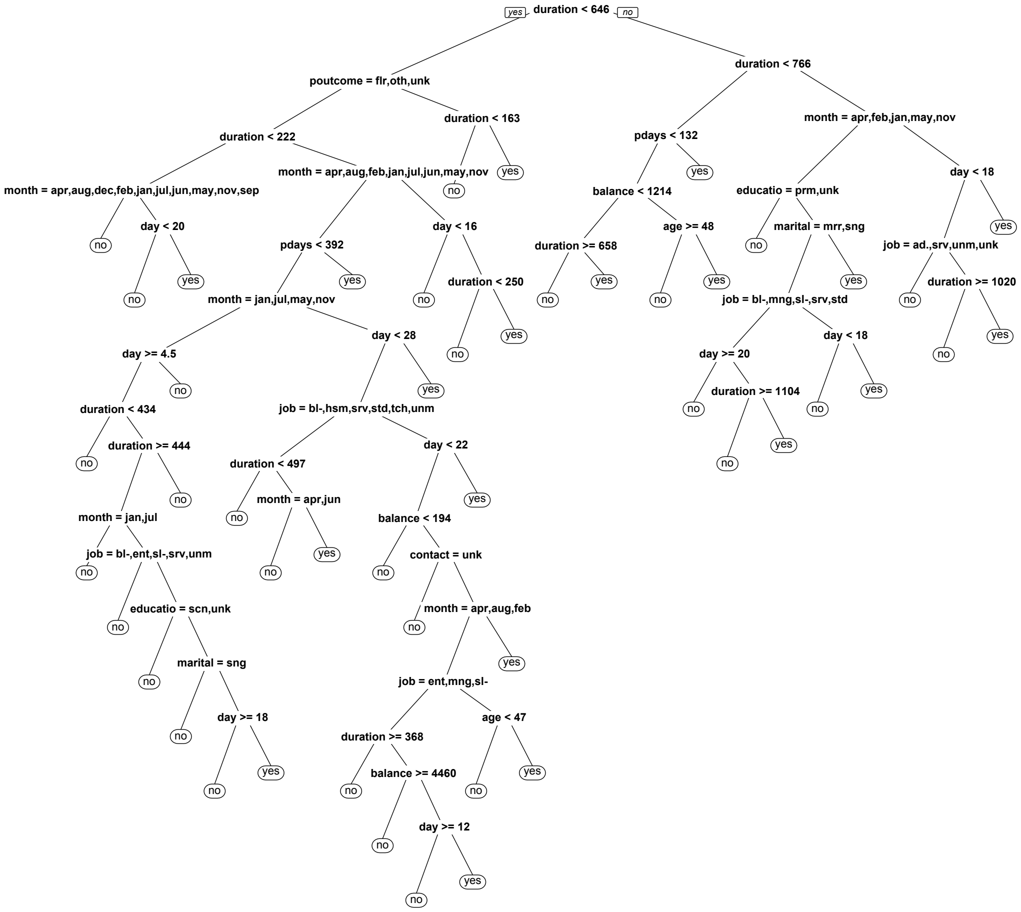


Figure 1: Full Tree

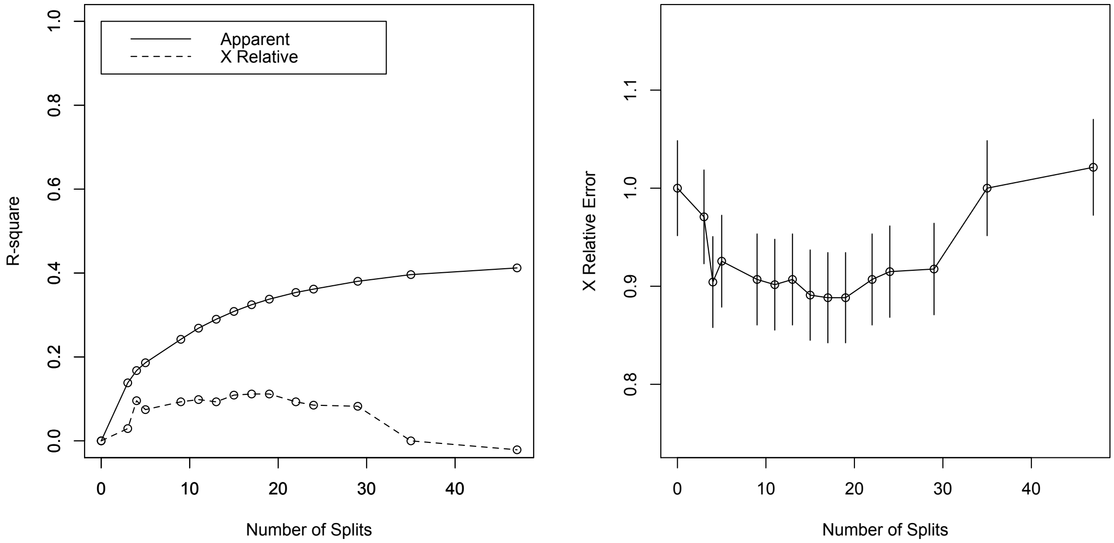


Figure 2: Relative Error vs #Splits

We configured the random forest algorithm by setting up prediction trees range from 10 to 100 to grow in each round. We got the importance of every predictor by observing the MeanDecreaseGini in Table 5 when we set the number of trees to 100.

Table 4: Importance of Predictors

Predictor	MeanDecreaseGini
age	81.987620
job	69.103244
marital	20.575953
education	24.121349
default	2.635483
balance	85.434797
housing	13.073510
loan	6.257758
contact	16.837081
day	77.674435
month	105.590479
duration	257.728635
campaign	32.924787
pdays	39.741963
previous	22.322606
poutcome	54.182151

2.4 Adaboost

We adopted Adaboost as our boosting method to train a stronger learner. We did 50 iterations in the Adaptive Boosting and the model fitted shows a train error of 5.5% which is no doubly a better

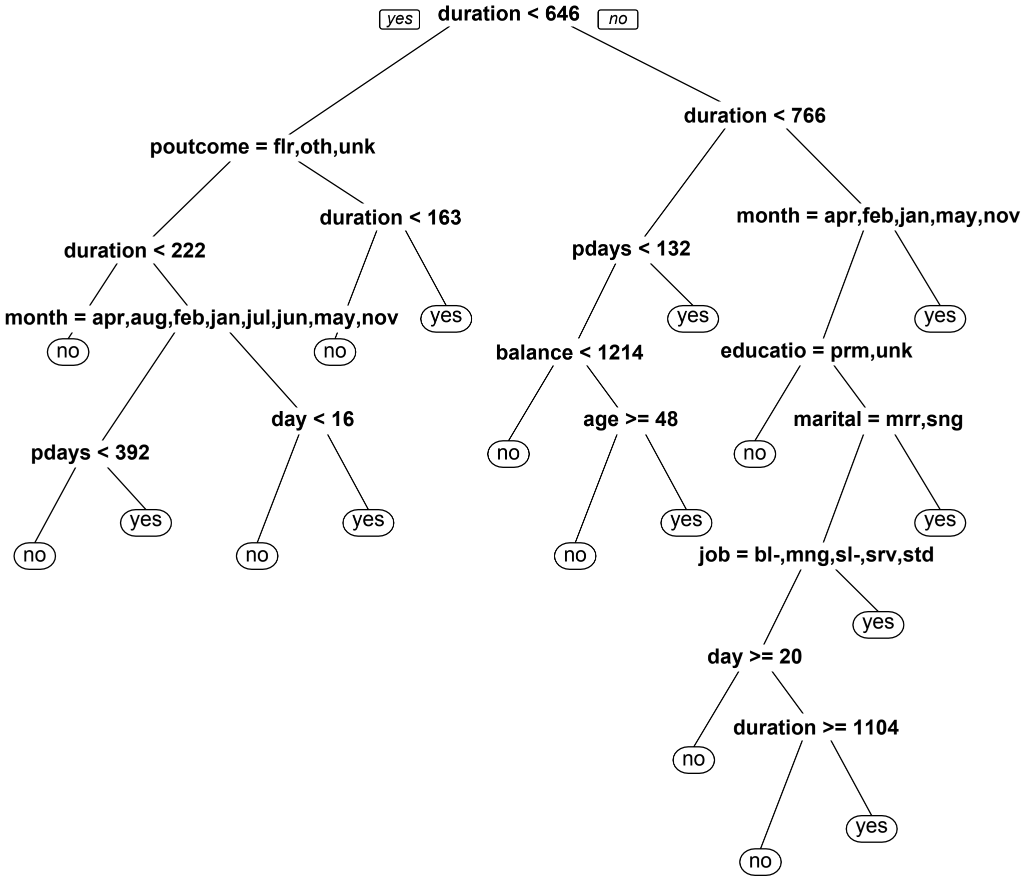
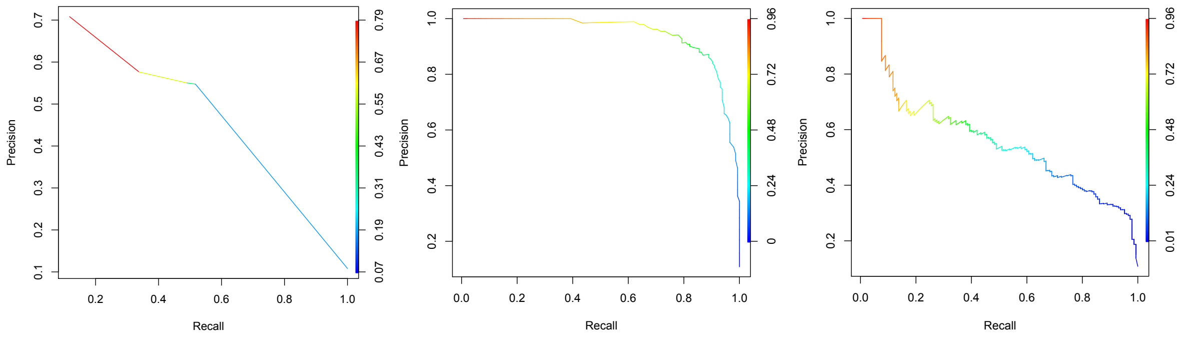


Figure 3: Pruned Tree



(a) Pruned Tree

(b) Random Forest

(c) Adaboost

Figure 4: Precision/Recall Curves

result than the former decision tree.

3 Evaluation

To evaluate the models fitted using the training data, we tested with the remaining 30% data to each model in each round and we adopted the 10-fold cross-validation to make the result more convincing. We demonstrate the performance of the models in terms of (1) Average Accuracy, (2) Precision/ Recall Curves, and (3) ROC curves.

3.1 Average Accuracy

Predictive accuracy refers to the ability of the model to correctly predict the class label of new or previously unseen data. In our models, we get the average accuracy from 10 rounds of cross-validation and each of them are the numbers of attributions correctly classified. The average accuracies of the (i) Pruned Decision Tree, (ii) Random Forest, and (iii) Adaboost are shown in Table **??**.

According to the table which indicates that the latter two models demonstrate higher average accuracy, we can conclude with the statement: bagging and boosting elevate the performance of the classic decision trees with higher predictive accuracies.

Table 5: Importance of Predictors

Model	Average Accuracy	Std. Dev. (+/-)
Decision Tree	90.15%	8.03%
Random Forest	96.97%	0.57%
Adaboost	96.96%	0.62%

3.2 Precision/Recall Curves

Precision and Recall are two measurements of great importance in data mining technologies. Precision is the probability that a (randomly selected) retrieved document is relevant. Recall is the probability that a (randomly selected) relevant document is retrieved in a search. We use Precision/Recall Curves which is a built-in measure in the ROCR package in R.

The three curves are depicted in Figure 3.2.

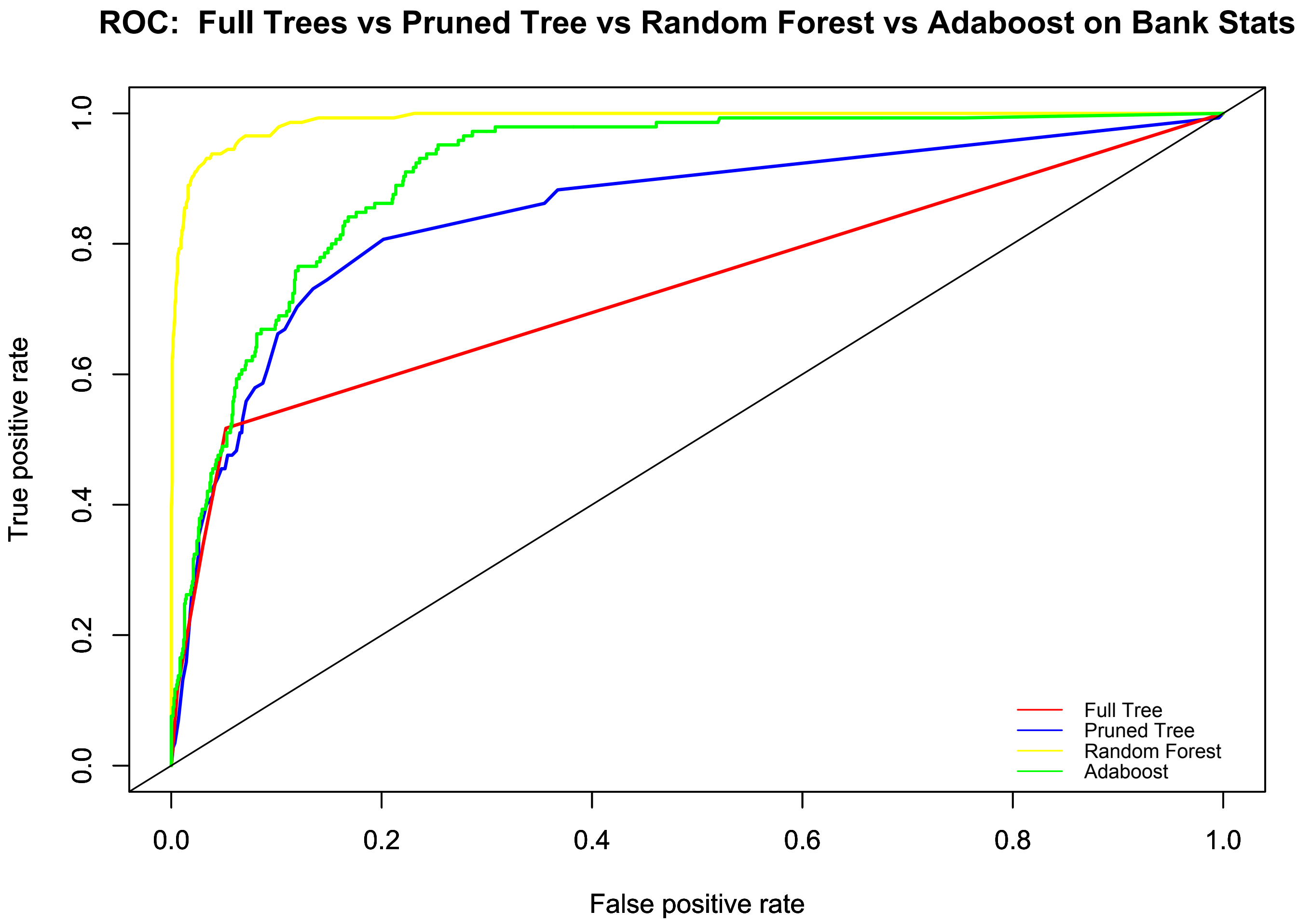


Figure 5: ROC Curves for each model

3.3 ROC Curve

ROC curve is a graphical plot which illustrates the performance of a binary classifier system as its discrimination threshold is varied [2]. A ROC space is defined by FPR and TPR as x and y axes respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). Since our task is to do a binary classification, we adopted this measurement for the evaluation as shown in Figure 5. As a result, these plots demonstrate that the random forest algorithm outperforms every other algorithm by a fairly wide margin.

4 Conclusion

In this task, we applied the classic data mining method, decision tree to a bank marketing dataset that predicts a user will subscribe the deposit or not based on a set a predictors collected by the contact campaign. We first built a full tree by using the Gini Criterion as the goodness of split and the CP value as well as the least number to split as the stopping criteria. We pruned the tree using cross-validation and get a 17 split tree out of a 47-splits-full tree. We also applied the Random

Forest and Adaboost as the bagging and boosting methods that assist with the model construction.

To evaluate the models we construct, we adopted the Average Accuracy, Precision/Recall Curves, and ROC curves. Random Forest and Adaboost outperform than the classic Decision Tree.

References

[1] R. Bryll, R. Gutierrez-Osuna, and F. Quek. Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets. *Pattern recognition*, 36(6):1291–1302, 2003.

[2] D. M. Green, J. A. Swets, et al. *Signal detection theory and psychophysics*, volume 1. Wiley New York, 1966.

[3] S. Moro, R. Laureano, and P. Cortez. Using data mining for bank direct marketing: An application of the crisp-dm methodology. 2011.