CSE 780

Noah Frank, 400050264

Assignment 3
Car Evaluation - Classification Tree Analysis

2021-02-26

frankn1@mcmaster.ca

1 Data Description

1.1 Data Source

The data used for this assignment is the ‘Car Evaluation Data Set’ retrieved from the UCI Machine Learning Repository.[1] The data is comprised of information regarding the evaluation of car acceptability across 1728 instances. The data set was originally derived from a hierarchical decision model, but will be used for classification task in this project.

1.2 Variable Details

The data set is made up of 6 predictor variables and 1 evaluation variable. All of the features are categorical variables, and the evaluation is a categorical variable denoting the evaluated acceptability of a car. This data set has no missing values for any instances. The data set also has has perfectly balanced predictor variables evenly split between either 3 or 4 factors, however the evaluation variable is unbalanced. The variables used in the analysis are described here:

- **Buying Price:** Cost of purchase (Low, Med, High, Very High)
- **Maintenance Price:** Cost of maintenance (Low, Med, High, Very High)
- **Doors:** Number of doors (2, 3, 4, 5+)
- **Persons:** Seating capacity (2, 4, 5+)
- **Luggage Boot:** Size of luggage boot (Small, Medium, Big)
- **Safety:** Estimated safety rating (Low, Medium, High)
- **Evaluation:** Evaluated acceptability of car (Unacceptable, Acceptable, Good, Very Good)

1.3 Descriptive Analysis

As previously stated, all variables of the data set are categorical in nature. The variables with 4 categories are evenly balanced with 432 of the 1728 instances per category. Similarly, the variables with 3 categories are evenly balanced with 576 of the 1728 instances per category. For example, the safety rating is distributed evenly between low, medium, and high, with 576 observations in each category.

Table 1: Unbalanced Class Distribution

Class	Count	Percentage
Unacceptable	1210	70.023 %
Acceptable	384	22.222 %
Good	69	3.993 %
Very Good	65	3.762 %

However, the output variable car evaluation, is not balanced between classes, as can be seen in Table 1. The ‘unacceptable’ class makes up a majority of the instances at 70%, with the ‘good’ and ‘very good’ classes making up a combined approximate 8 % of the data set. We can see how the data set is heavily skewed to the lower evaluated acceptability classes. In Fig. 1 we can see a bar plot of the safety level counts along with the evaluation class in which they were observed. This shows a good example of a feature for which model decisions might be made easier as all low safety ratings are associated with ‘unacceptable’ evaluations, and all ‘very good’ evaluations are associated with high safety rating.

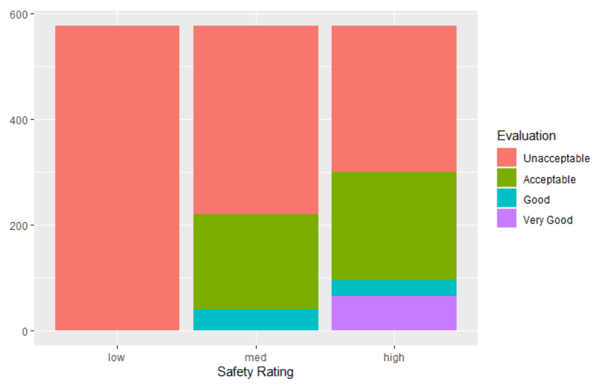

Figure 1: Counts of Safety Levels by Evaluation Class

Similarly to the safety rating, the number of occupants can aid in model decision, as all the cars with a capacity of 2 are evaluated as unacceptable. Conversely, some features such as number of doors have much more even distributions across evaluations classes as seen in Fig. 2. These may not help the model with decision making as much as the previously mentioned factors. The remaining variables’ distributions lie somewhere between that of safety rating and number of doors. The nature of this data set, at a glance, seems to make it a solid contender for decision trees.

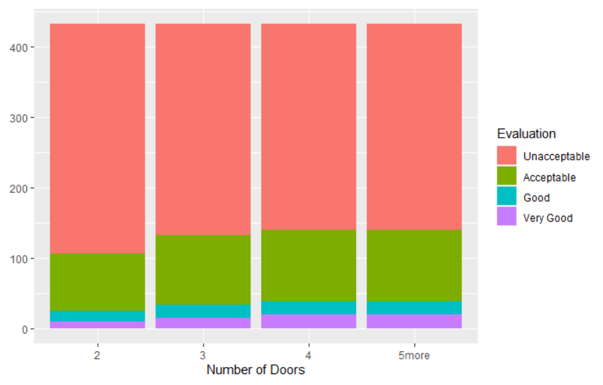

Figure 2: Counts of Door Numbers by Evaluation Class

[1] UCI Machine Learning Repository. *Car Evaluation Data Set*. 1997. URL: https://archive.ics.uci.edu/ml/datasets/Car+Evaluation.

2 Problem Description

The aim of this analysis is to train a classification tree in order to accurately predict the acceptability of an unknown car given its predictor variables. Specifically, the analysis will also look at how different classification tree methods will affect the results of the model. Through the analysis, we will also determine which variables are the greatest and worst predictors for the evaluated acceptability.

3 Technique Description

Classification is used in order to predict the category to which a new observation belongs based on previously observed data. Classification is often done using both a training and a test set, containing labelled observations and unlabelled observations, respectively.[2] Measures of classification success that can be used are classification accuracy and the Rand index. The classification rate is given by the ratio of correctly classified observations over all the observations in the test set. The Rand index is more flexible than classification rate, and can be used in both supervised and unsupervised learning. The Rand index (RI) is given by:

$$RI = \frac{TP+TN}{TP+TN+FP+FN}$$

where TP is the amount of true positives, TN is true negatives, FP is false positives, and FN is false negatives.

The adjusted Rand index (ARI) corrects the Rand index by reducing agreement by chance, and is given by:

$$ARI = \frac{RI - \text{Expected}RI}{\text{Max}RI - \text{Expected}RI}$$

The ARI usually ranges from a value of 0 to 1, for random classification to perfect classification, respectively.

In order to classify the data set for this analysis, we first look at the classification tree, also called a decision tree.[3] A classification tree is composed of nodes and branches which will allow us to produce a predicted class using the predictor variables of a data set. Starting from the root node and through each intermediate node, the data is evaluated according to a feature and a rule on which it will be split, causing a 'decision' in the model. After terminal nodes are the final nodes in the tree, in which a class is predicted based on the path leading to that node. The decision on the splitting feature and rule is based on the Gini index, given by:

$$\sum_{g=1}^{G} \hat{p}_{mg}(1 - \hat{p}_{mg}),$$

where $\hat{p}_{mg}$ is the proportion of observations from class g in node m.

Single classification trees have the drawback of having high variability between training sets, and over fitting to the training data. We will be using ensemble methods to decrease the variation and increase the performance of our classifier.[4] With bagging, also called bootstrap aggregating, we use bootstrapping to re-sample the entire training set with repetition, and fit a classification tree to each ensemble. Combining the prediction from all the trees will lead to a classification estimate with less variance than the single classification tree, taking the majority vote of estimates. The performance of the classifier can be measured by the misclassification rate (MR) or ARI. The bagging method can be adjusted by tuning the number of trees parameter.

A slightly modified version of bagging can be carried out where we do not consider all predictors at every split in the tree. This ensemble method is called random forests, and is used to diversify the trees used in bagging by forcing the splitting to use different variables depending on the variable sampled and the re-sampling of the training set for each tree. At each split, for each tree, a different number of predictors, $\mathcal{M}$, is considered. The random forest method can be adjusted by tuning the number of trees and the number of predictors at each split. A value for $\mathcal{M}$ often used is: $\mathcal{M} = \sqrt{p}$, where p is the number of predictors.

Finally, the last ensemble method explore here is boosting. In boosting, trees are grown sequentially with respect to each other, rather than independently and then combined. The output of the first tree will be the input to the second tree, and so on. The goal of boosting is to pass on the tree's output and a weight relative to it's accuracy in order to ensemble a weighted sum. The goal is to minimize the overall error according to these weights in an iterative manner. Thus, boosting takes multiple weak learners and can form a strong learner. This method can be adjusted by tuning certain parameters including: The number of trees, the shrinkage parameter, and the depth of each tree.

4 Result Description

The classification tree analysis described in Section 3 was carried out using the 'randomForest', 'gbm', 'tree', and 'e1071' packages in R. Before implementing models, the data was split into training and testing sets each comprised of half the original data set. The testing sets were ensured to be relatively balanced, as shown in Table 2.

[2]Prof. Sharon McNicholas. *CSE780: Introduction to Classification*. McMaster University, School of Computational Science & Engineering. 2021.

[3]Prof. Sharon McNicholas. *CSE780: Classification & Regression Trees*. McMaster University, School of Computational Science & Engineering. 2021.

[4]Prof. Sharon McNicholas. *CSE780: Bagging, Random Forests, and Boosting*. McMaster University, School of Computational Science & Engineering. 2021.

Table 2: Near Stratified Test/Train Data Split

Subset	unacc	accc	good	vgood
Train	601	201	30	32
Test	609	183	39	33

The first model, used as a baseline to compare other models is the basic classification tree. When applied to the testing set, the simple tree had a MR of 0.0776 and an ARI of 0.8280. Using bagging, with a default number of trees as 500, the resulting measures were reduced to 0.0417 MR and a 0.9084 ARI. Tuning the number of trees in the ensemble lead to find the optimal model at 300 trees, resulting in the same MR of 0.0417, but an improved ARI of 0.9085. The improvement of ARI is marginal, however reducing the number of trees leads to faster computation times. In addition, from Fig 3. we can see the variable importance based on the decrease in Gini Index when those variables are removed from the model as a split option. The safety rating and number of persons appear as the most important variables, while number of doors is the least important according to the bagging model.

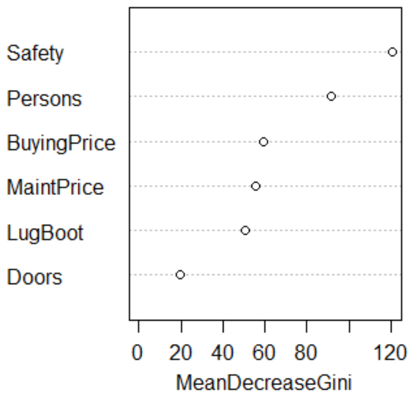

Figure 3: Bagging Model Variable Importance scored by Gini Index

Next, we look at the performance of random forest beginning with the default model with 500 trees and a starting $\mathcal{M}$ value of 3. This default random forest model results in a MR of 0.0567 and an ARI of 0.8724. After parameter tuning, the optimal values were found to be 300 for the number of trees and a $\mathcal{M}$ of 5 resulting in a MR of 0.0428 and an ARI of 0.9043. As the number of trees increased or decreased away from 300, the ARI decreases. Also, a $\mathcal{M}$ of 1 was the worst at all number of trees, increasing the ARI as $\mathcal{M}$ increased towards 6. However, at 6 the model would instead just be a bagging model. The variable importance was the same as in Fig. 3, and overall the model performed worse than the bagging model.

Finally, we can look at the boosting model starting with a default of 100 trees to fit, default shrinkage of 0.1, and default depth of 1. This default model which serves as a pre-tuning reference has a MR of 0.1215 and an ARI of 0.7423. After tuning, the optimal parameters found were: 3900 trees to fit, shrinkage of 0.1, depth of 3. These new parameters lead to a major increase in the model performance, resulting in a MR of 0.0139 and an ARI of 0.9722. This result was found testing on 3900 trees, even though according to the 'gbm' package the optimal number of trees to test on would be 83, which actually resulted in a MR of 0.0567 and an ARI of 0.8817. Overfitting did not seem to cause a problem when tested on 3900 training iterations and 3900 testing iterations. However when attempted with 10000 training iterations, testing iterations above 5000 lead to a decrease in the ARI, showing that overfitting must be watched for when testing boosting models.

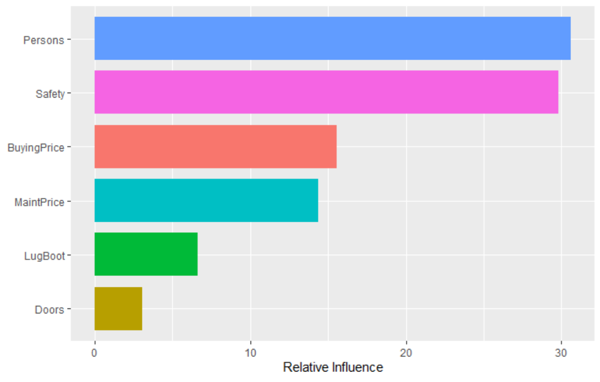

Figure 4: Decrease in Training Deviance over Iterations

In Fig. 4, we can see the relative influence of each variable showing similar results from Fig. 3. However in the boosting model more of the reduction in loss can be attributed to the car capacity than the safety rating. Finally, the results of the models can be summarized in Table 3.

Table 3: Tuned Model Results

Model	MR	ARI
Tree	0.0776	0.8280
Random Forest	0.0428	0.9043
Bagging	0.0417	0.9085
Boosting	0.0139	0.9722

5 Conclusions

This analysis of classification trees and related ensemble methods showed the advantage of these ensemble methods. As seen in Table 3, all ensemble methods performed better than the single classification tree, with the boosting method improving the ARI of the classification tree by nearly 0.15 points. Based on it's performance with the test data, these models should be able to confidently classify new data points. Furthermore, through the classification tree analysis, we have uncovered information on variable importance and influence from this data set. The most important factors in determining the acceptability of a car are the number of people it can seat, as well as the safety rating. Following these factors, it is shown that lower price, for both buying and maintenance also contribute to a better evaluation. Finally, the amount of boot space and number of doors have the least importance when distinguishing between the acceptability of cars.