Classifying car price ranges

Sivert M. Skarning

Mai 2019

Contents

1 Introduction

1.1 Problem set

Today everything is moving online. Advertising your car by using services online is very common. So how much is your car worth? How do you categorize the price range of your car? This project focuses on how to predict the value of cars, based the car features.

The data set chosen for this project is the car evaluation data set created by Marko Bohanec. This data set contains information about 1728 cars.

- buying: vhigh, high, med, low.
- maint: vhigh, high, med, low.
- doors: 2, 3, 4, 5more.
- persons: 2, 4, more.
- lug-boot: small, med, big.
- safety: low, med, high.
- Acceptability: unnac, acc, good, v-good

Based on maintenance costs, number of doors, how many persons that fits in the car, boot size, acceptability and safety rating we will try to make a model that predicts the buying price of the car. The model will use 6 predictors to classify which class the car is in. The predictors are maintenance, number of doors, number of persons that can fit in the car, luggage size, acceptability and safety.

1.2 Data set

Usage The car evaluation has been referenced in many scientific papers. Amongst the most noteworthy are:

- MML Inference of Decision Graphs with Multi-way Joins and Dynamic Attributes [1]
- Stopping Criterion for Boosting-Based Data Reduction Techniques: from Binary to Multi class Problem [2]
- Impact of learning set quality and size on decision tree performance [3]

The data set was used as an example for displaying multi attribute decision-making [5].

Generation According to Bohanec the Car evaluation data set was created from a hierarchical decision model. This model was created for the demonstration of a decision making software called DEX [4].

2 Data Pre-processing

2.1 Feature Extraction

Data reduction is about removing predictors to get a smaller and simpler data-set to work with. I decided to use PCA for feature exctraction in order to simplify my data set and still capture much of the variance in the data-set. The way PCA works is that it finds predictors that are highly correlated and tries to create linear combinations of them. These linear combinations capture all of the needed information.

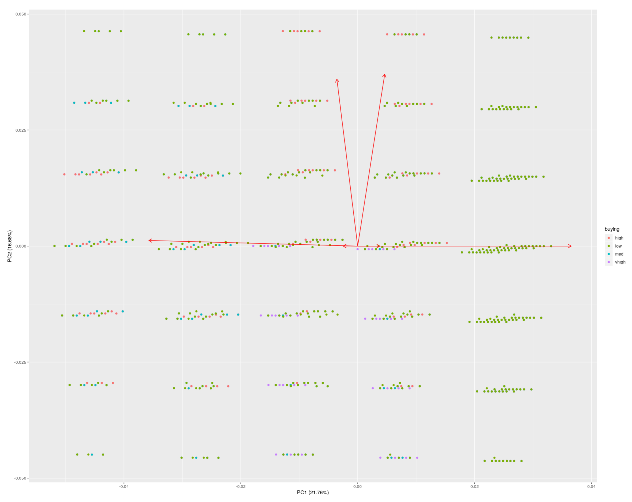


Figure 1: PCA plot

As said in the book [7] having predictors that are not correlated can be advantegous. As you can se on the PCA plot there are not any clusters that sperates the classes. When trying to train the model using PCA, I get an error rate of 65%. This is roughly 10% worse than without PCA. The result might be because the pca has generated componenents that does not capture the relevant data [7]

2.2 Between Predictor Correlation

As i talked about in de Feature Extraction subsection, to high of a correlation is always good for the model accuracy. To remove predictors that are highly correlated we can also use the following algorithm [7]:

- Set a treshold
- Calculate the correlation matrix of the predictors.
- Find the highest correlation
- Determine the average correlation between A and all the other variables as well as B.
- Remove the predictors with the highest average correlation.
- Repeat this until no predictor correlations are above the treshold you hava set.

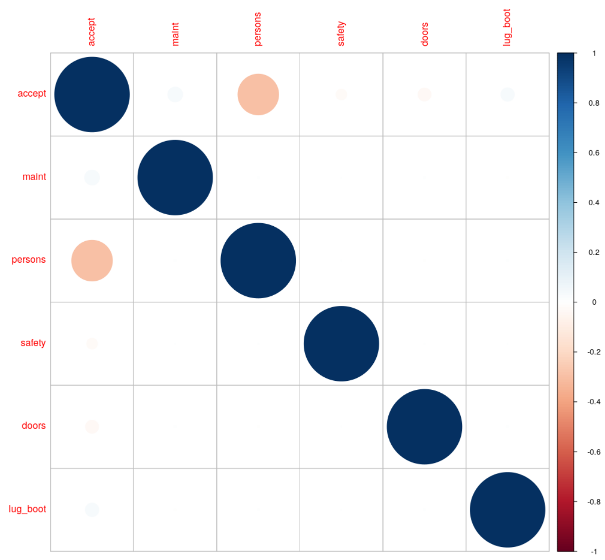


Figure 2: Between predictor correlation

In the car evaluation data set there were no predictors that had a correlation high enough to remove it. This algorithm wil not yield any result for the model.

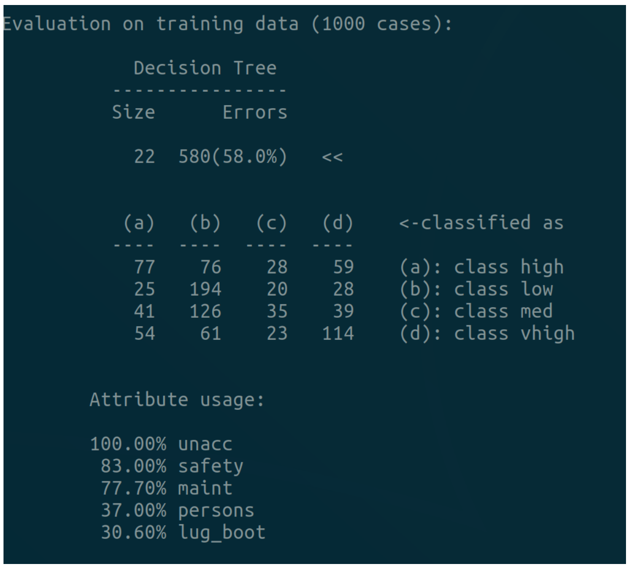
3 Classification

3.1 C5.0

I decided to use C5.0 for the classification. It is free and easy use. C5.0 is an algorithm to produce decision trees for classification purposes. For configuring the data and running the algorithm I used R. In R you can use the C5.0 package to generate basic tree-models and rule-based models. In this chapter I will present my findings when using this algorithm on the car evaluation data set. To help me get started with R and the C5.0 algorithm I used this guide [6].

3.2 Findings

When running the C5.0 algorithm on the car evaluation dateset we found that the accuracy was not very high. We got a 55 percent error rate when using all the predictors. The summary of the first run can bee seen in figure 3 and the decision tree generated can be seen in figure 4.



```
Evaluation on training data (1000 cases):

            Decision Tree
          ----------------
          Size      Errors

            22  580(58.0%)   <<


           (a)   (b)   (c)   (d)    <-classified as
          ----  ----  ----  ----
            77    76    28    59    (a): class high
            25   194    20    28    (b): class low
            41   126    35    39    (c): class med
            54    61    23   114    (d): class vhigh


        Attribute usage:

        100.00% unacc
         83.00% safety
         77.70% maint
         37.00% persons
         30.60% lug_boot
```

Figure 3: tree-summary

3.3 Trees and rules

When comparing trees in figure 3 and rules in figure 5 we see that the error rate is similar. The tree has a slightly better error rate. This is expected

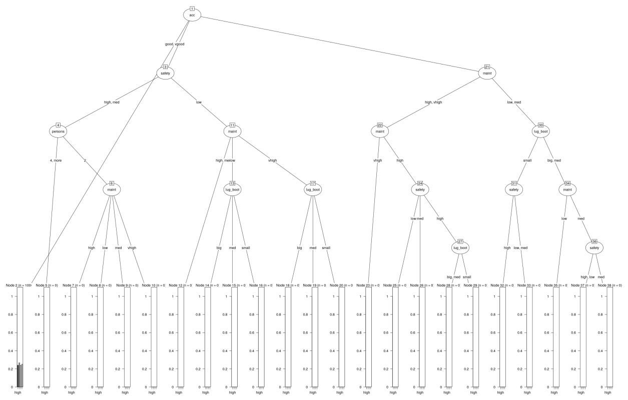
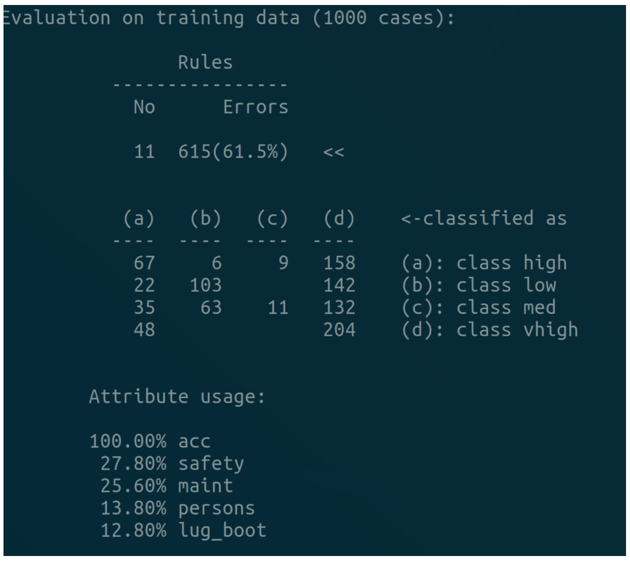


Figure 4: Decision-tree

because rules are a simplified tree. The rules consists of if-then rules. The rules makes it easier to read, however it might yield a slightly worse result.



```
Evaluation on training data (1000 cases):

              Rules
        ----------------
          No      Errors

          11  615(61.5%)   <<


         (a)   (b)   (c)   (d)    <-classified as
        ----  ----  ----  ----
          67     6     9   158    (a): class high
          22   103         142    (b): class low
          35    63    11   132    (c): class med
          48               204    (d): class vhigh


       Attribute usage:

       100.00% acc
        27.80% safety
        25.60% maint
        13.80% persons
        12.80% lug_boot
```

Figure 5: Rules

Rules When generating rules from the tree model we get 11 rules or 11 paths through the tree. Some rule are not very sensible like rule number 10 which states that if the car acceptability is unacceptable then the price range is very high. Some rules are more logical like rule 5 that states that if it has medium safety, a small boot size and its in acceptable condition it will be be priced in a low price range. Rule 9 also states that if the maintenance costs are medium, it has a big boot and its in acceptable condition it will be priced very high which is very true when it comes to big station wagons or SUV's. This means that some of the rules have logic that someone with domain knowledge can recognize.

3.4 Testing the model

When I started the building of the model i split the data set. A thousand rows of data was reserved for training the data and 728 rows was reserved for testing. When using the predict function C5.0 on the test data set we get the results displayed in figure 6.

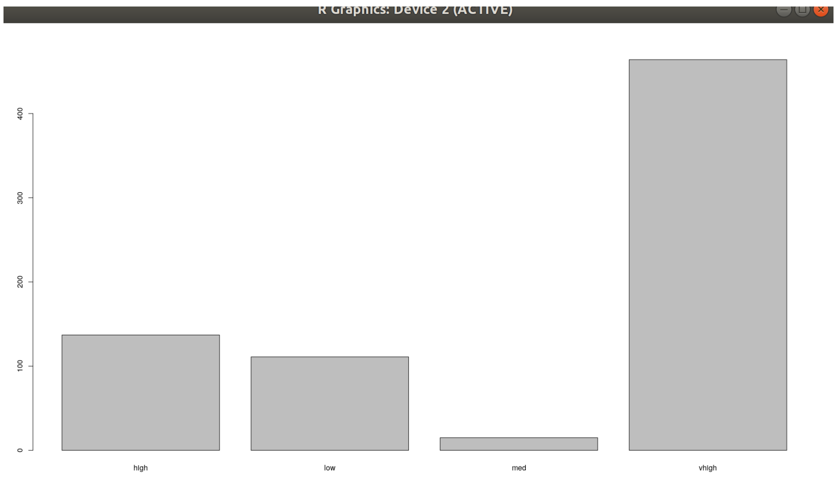
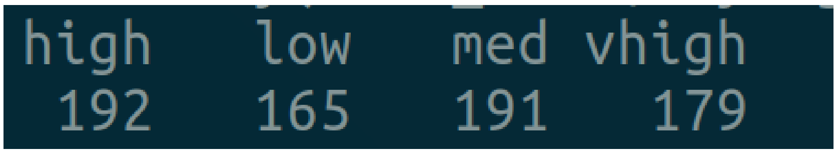


Figure 6: Prediction on training set with the rule model

As you can see in figure 6 the number of cars classified in the v-high price range, way exceed the number of cars that exceed the medium price range. This is expected when you look at the rules of the model. Three ways thorough the tree gives a classification of v-high, but only one classifies it as medium.

Figure 7 shows the actual distribution.



```
high   low   med vhigh
 192   165   191   179
```

Figure 7: Actual distribution of test data

3.5 Winnowing, boosting and pruning

Boosting Boosting is an algorithm where multiple model are created. When classifying, all of the models vote for which classification the row will be. I tried to enable boosting for my data set, but it showed up with an error that said that the last classifier was not accurate enough.

Winnowing Winnowing is an algorithm that exclude attributes that does not provide any value able information. When i applied the C5.0 algorithm with winnowing enabled it winnowed three attribute, the number of doors, the number of persons that can fit in the car and the boot size. This had a negative effect on the model and the error rate increased with 4.3 percent.

4 Scikit Learn and dection trees

4.1 Trees

Tree-based models consists of one or more nested if-then statements for the predictors that partition the data.A rule is a set of if-then conditions that have been collapsed into independent conditions.A rule is defined by the tree. A rule is defined as a distinct path through a tree. For the tree, a new sample can only travel down a single path through the tree defined by these rules.Let's consider an example provided in the book, Applied Predictive Modelling, Chapter 8:If Predictor Ais bigger or equals1.7 then If predictor B is smaller or equals 202.1 then outcome =1.3. Else outcome =5.6Else outcome = 2.5.

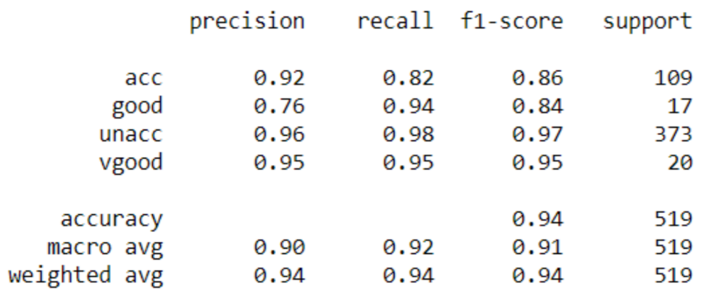
Trees are more sensitive to slight change of parameters. In such cases rules perform rigidly then trees. Rules have there own limitations representing many categories which can be done most effectively by trees. A 2.5 B 5.6 1.3 Our data set is taken from http://archive.ics.uci.edu/ml/datasets/Car+Evaluation.The dataset is about predicting the model of car (acceptable, unacceptable, very god, good are the class which represents the model). On the basis of features like comfort, buying price, maintenance price, number of doors, number of passengers, luggage boot size, the class of the car is predicted.We model this data set using R and python separately. Here we will discuss aboutthe finding of dataset by using python and decision tree algorithm for classifying the class.In original data set column name was missing. We assign name to each column. During Exploratory Data Analysis, we find that the number of doors were assigned as 5, which is quite unpractical. And, number of People were mentionedas '5more'. It was replaced by an integer value of 6. We used Decision Tree Classifier, which is a function of Scikit learn library.

4.2 Testing

We split the data set into training and testing set in the proportion of 70% and 30% respectively. For the object type attributes of column, dummies variable wasused before fitting the data into Decision tree classifier.

4.3 Result

In the figure below you can se the achieved result.



```
              precision    recall  f1-score   support

         acc       0.92      0.82      0.86       109
        good       0.76      0.94      0.84        17
       unacc       0.96      0.98      0.97       373
       vgood       0.95      0.95      0.95        20

    accuracy                           0.94       519
   macro avg       0.90      0.92      0.91       519
weighted avg       0.94      0.94      0.94       519
```

Figure 8: Confusion matrix

5 Conclusion

In this project i hoped to achieve much greater accuracy than what I did. I have tried the C5.0 algorithm with several configurations and additional algorithms. The best result I was able to achieve was 58 percent error rate. 58 percent in this domain is not satisfactory for the use cases of this model. I have worked systematically Troy try to improve it without no real breakthrough. In order to improve my model I will try other algorithms like XgBoost. I will also try other data set about cars in order to get more predictors to work with.

References

[1] peter J Tan, David L Towe, 2003 *MML Inference of Decision Graphs with Multi-way Joins and Dynamic Attributes* .

[2] Marc Sebban, Richard Nock, Stéphane Lallich, 2002 *Stopping Criterion for Boosting-Based Data Reduction Techniques: from Binary to Multi-class Problems.*

[3] M. Sebban, R. Nock2, J.H. Chauchat and R. Rakotomalala *Impact of learning set quality and size on decision tree performance*

[4] Marko Bohanec *http://archive.ics.uci.edu/ml/datasets/Car+Evaluation*

[5] Marko Bohanec, Vladislav Rajkovic *Knowledge acquisition and explanation for multi-attribute decision making*

[6] Rulequest research *C5.0: An Informal Tutorial*

[7] Max Khun *Applied Predictive Modeling*