(HarvardX PH125.9x)
Data Science: Capstone

Capstone Project

Classification of Car Evaluation

March 2022

submitted by Tobias Sveaas

Contents

Introduction

In this project we use machine learning to determine a classification model that predicts whether a car is in an "acceptable" or "unacceptable" condition based on different attributes of the car. As doing this manually can be time consuming.

For this project we determine the effectiveness of the model based on the accuracy of it (% of correct classifications). A decision tree model was the final model used in this project and resulted in an accuracy of 95.1%.

The car data set was collected from the UCI machine learning repository (and can be downloaded via https://archive.ics.uci.edu/ml/machine-learning-databases/car/)

Load packages and libraries needed in the project

First we install and load in the packages and libraries we will be using for the project

```
library(tidyverse)
library(caret)
library(data.table)
library(ggplot2)
library(ggthemes)
library(dplyr)
library(stringr)
library(rpart.plot)
library(randomForest)
```

Data cleaning

We load in cars data set and simpilfy the car acceptability classification from “unacceptable”, “acceptable”, “good” and “Verygood”. To just unacceptable and acceptable by combing “acceptable”,“good” and “Verygood” into just “acceptable”. This is to make it a binary problem on determining whether the car is acceptable or unacceptable.

We then split the data into two sets, the edx set (80%) which we will be using to create our model and the final validation set (20%) which we will be using only for the final test of our model.

Data Exploration

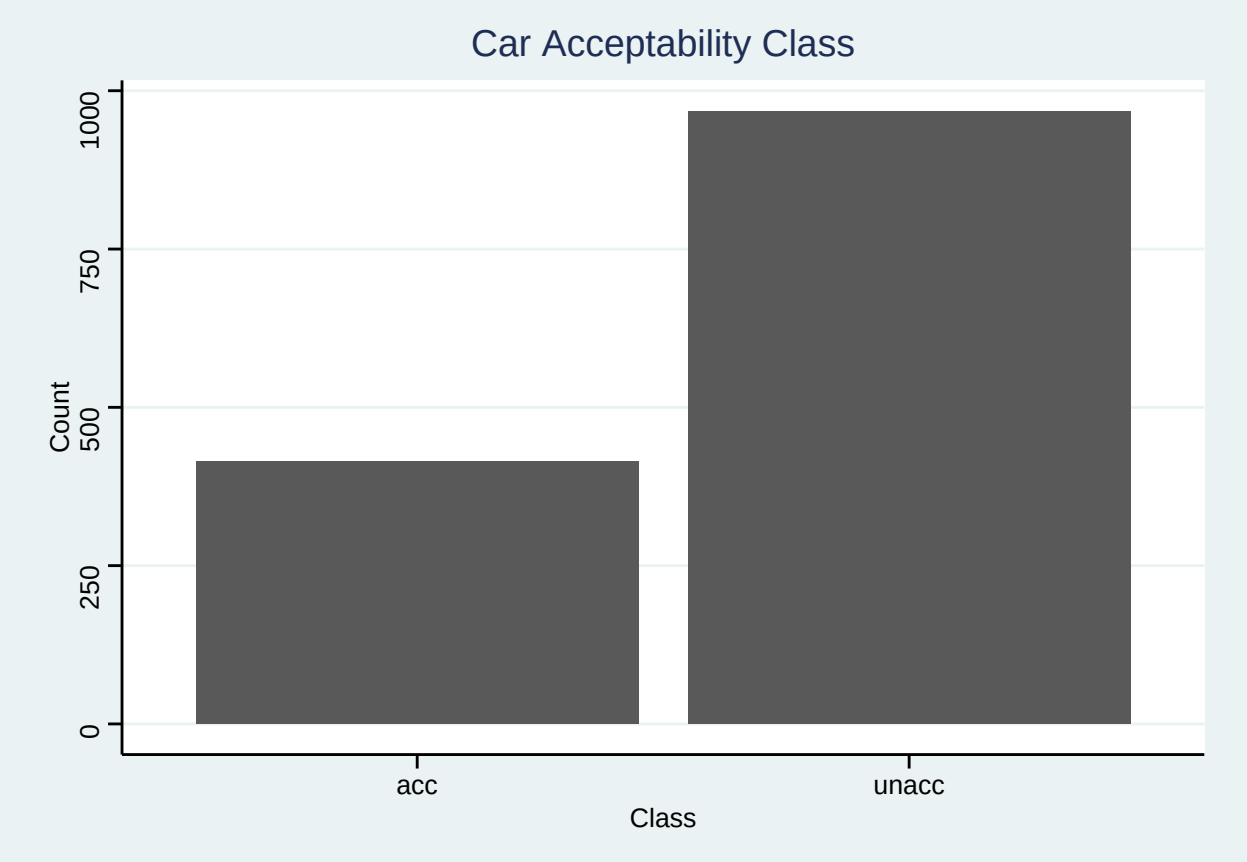
Exploratory analysis is done on the edx data set to get an overview of the main characteristics of the data set.

The data set contains 1382 car sample observations with 7 variables. The variables feature names, descriptions and classifications are listed below: *class: Car acceptability classification (acceptable = acc, unacceptable = unacc), this is our dependent variable and what our model will attempt to predict* buying_price: Cost of the car to buy (low, med, high, vhigh) *maint_cost: Maintenance cost of the car (low, med, high, vhigh)* doors: Number of doors on the car (2,3,4,5 or more) *person_capacity: Number of person the car can carry (2, 4, more)* lug_boot: The size of the luggage boot (small, med, big) *safety: Safety level of the car (low, med, high)

Data Analysis and Visualisation

Here we further analyze the data through visualization to explore the relationship each of our variables have with the car acceptability classification

Lets first see the distribution between acceptable and unacceptable cars

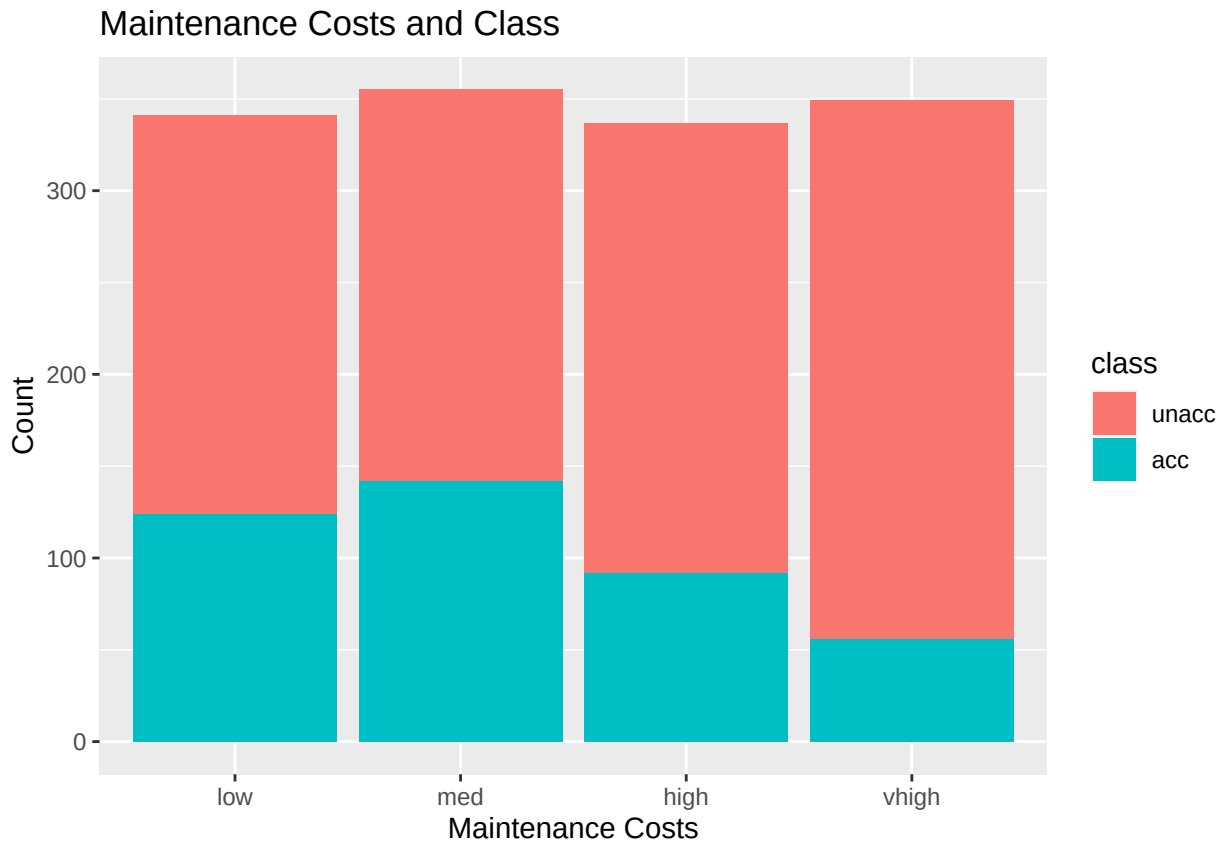


We see that there are more than twice as many unacceptable cars than acceptable cars

Buying Price

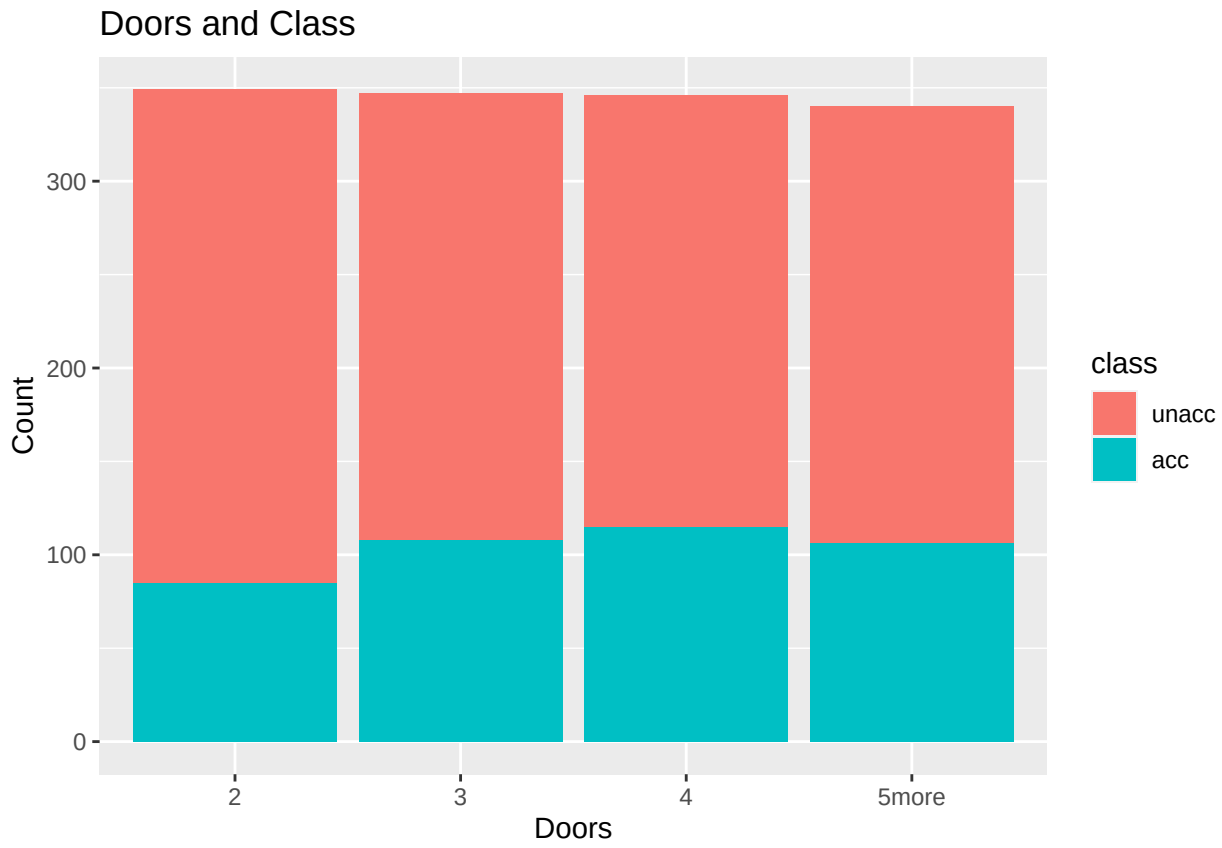
We can see a relationship with buying price in that a higher proportion of cars are acceptable at lower prices

Maintenace Costs



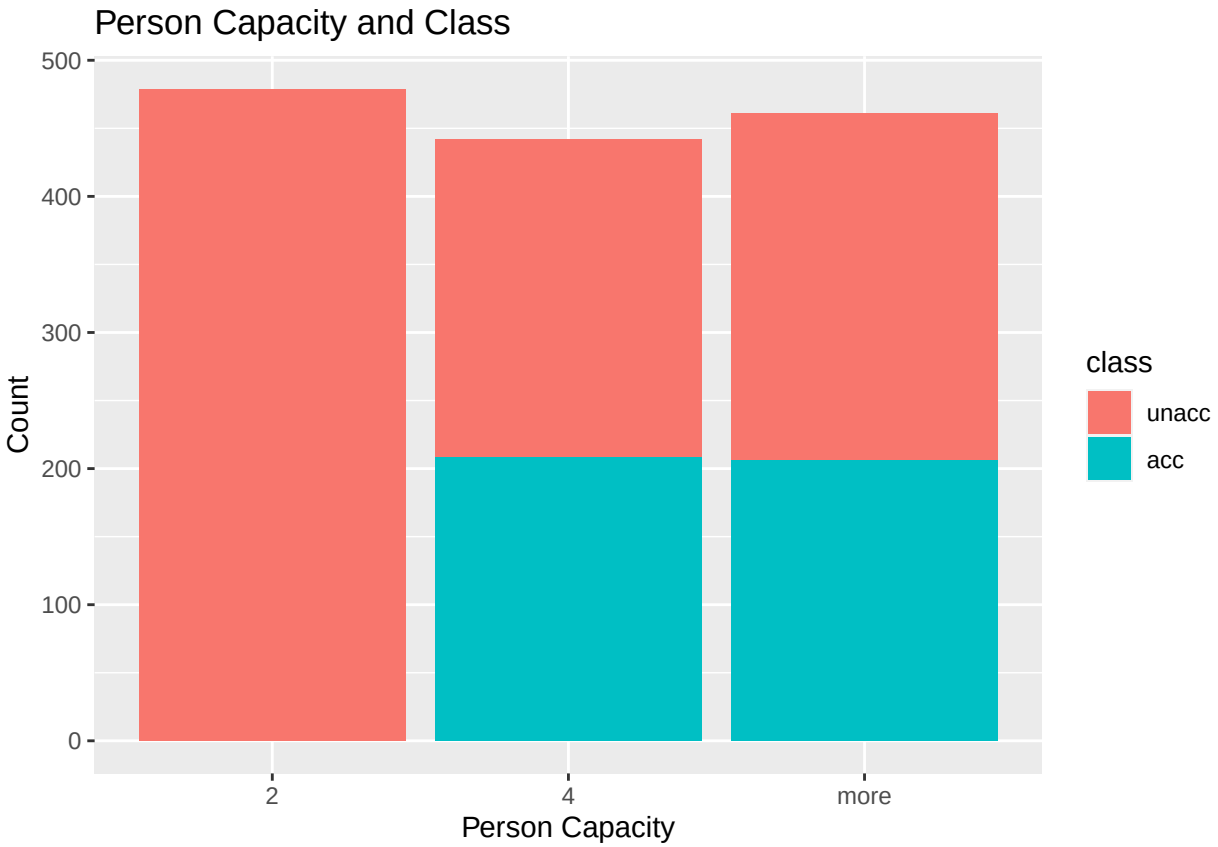
We can see that cars with vhigh maintenance costs are less likely to be acceptable but low, med and high are fairly similar

Number of doors



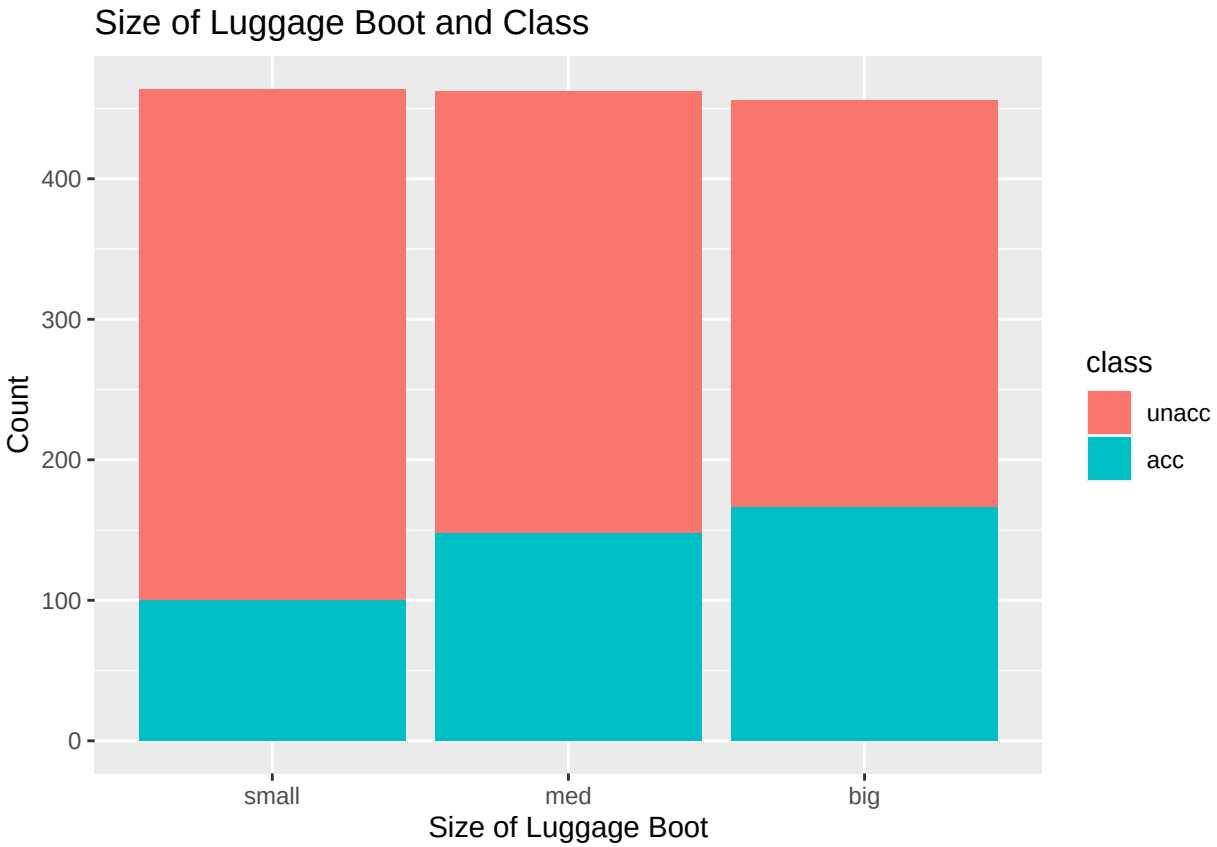
We don't see much effect the number of doors have on the acceptability of the car.

Person Capacity



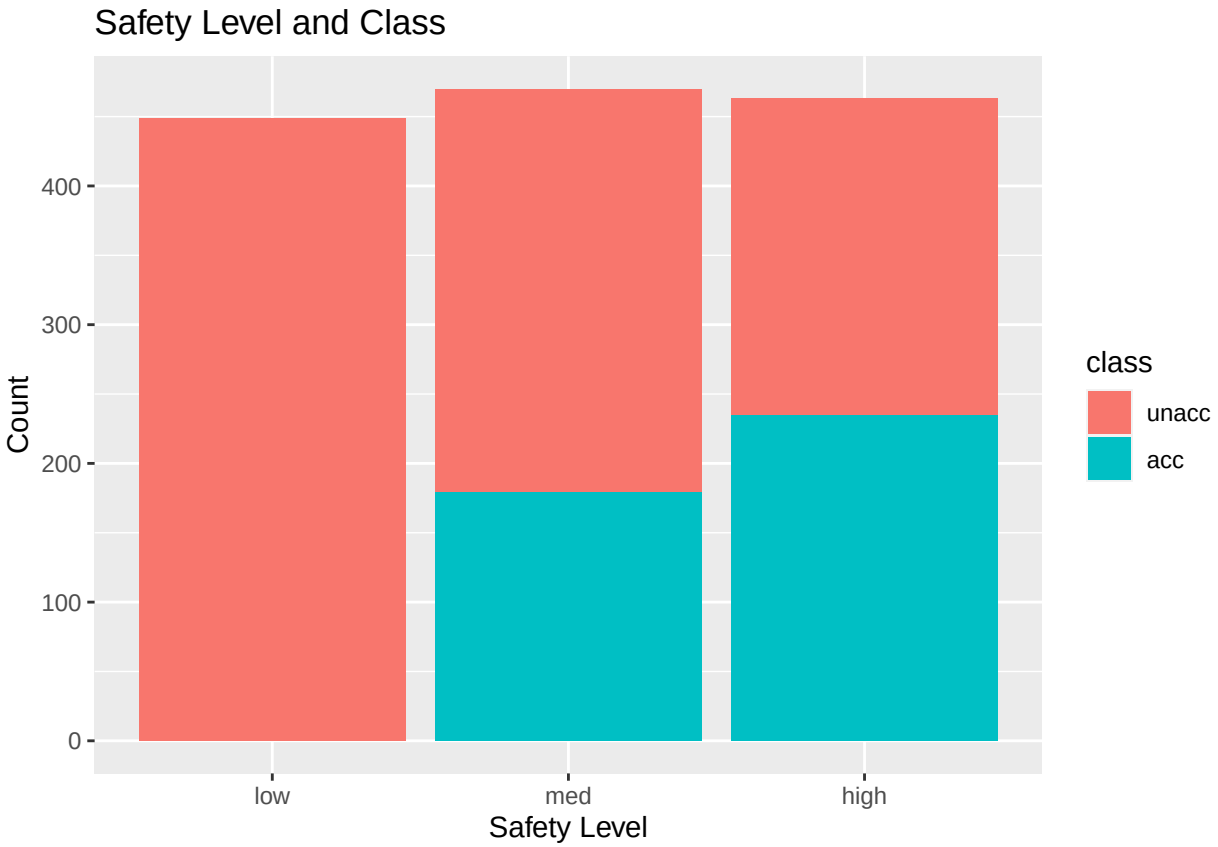
We see that all cars that seat only 2 people are unacceptable with 4 and more being equally acceptable. So this will likely be an important variable in our prediction model.

Size of Luggage Boot



We see a slight trend with cars having a larger boot size being more acceptable than cars with a smaller boot size

Safety Level



We see that all cars with a low safety level being unacceptable and that cars with a higher safety level being more acceptable. So this will likely be an important variable in our prediction model.

Creating the Prediction Model

After the data analysis I have determined that a decision tree model will be a suitable model for this project. A decision tree is a predictive model that benefits from a number of binary rules to calculate the classification of the desired variable.

Decision trees generate classification models in tree form. This form helps to the understand the decision hierarchy and relations between attributes by visualizing the possible outcomes of each attribute as a branch of a tree.

One of the benefits of the decision tree model is that it is easy to interpret the classification process. However in order to maximize the accuracy of our model we will also try a random forest model, which loses the interpretability of the decision model.

The edx set will be further split into training (80%) and test set (20%) in order to find the best model.

Building Decision tree model

USing the rpart and caret package we train the decision tree model using cross validation with 10 folds repeated 3 times.

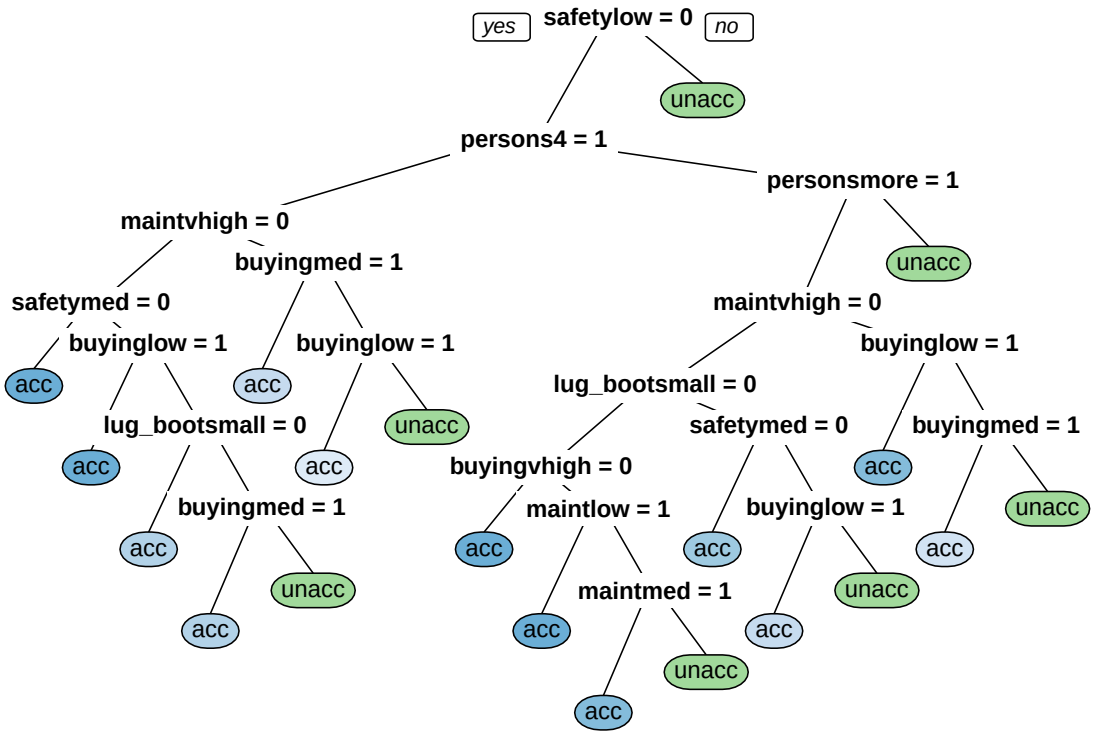
```
# Building the decision tree model
# Do repeated cross validation with 10 folds 3 times
train_control <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

# Create the decision tree using train function
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
decision_tree <- train(class ~ ., data = train_set, method = "rpart",
                       trControl = train_control,
                       tuneLength = 10)
```

Now we can plot the Decision tree created to visualize the process using the prp function

```
# Plot the decision tree model
prp(decision_tree$finalModel, box.palette = "auto",
    faclen = 0, # print fullt factor names
    varlen = 0) # write full names in branches
```



Next we will use our model to predict the test set and examine the confusion matrix.

```
# Use the model to predict test set
test_pred_rpart <- predict(decision_tree, test_set)
confusionMatrix(test_pred_rpart, as.factor(test_set$class))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc unacc
##      acc    80    10
##      unacc   3   184
##
##                Accuracy : 0.9531
##                  95% CI : (0.9211, 0.9748)
##     No Information Rate : 0.7004
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.8908
```

```
##
##  Mcnemar's Test P-Value : 0.09609
##
##             Sensitivity : 0.9639
##             Specificity : 0.9485
##          Pos Pred Value : 0.8889
##          Neg Pred Value : 0.9840
##              Prevalence : 0.2996
##          Detection Rate : 0.2888
##    Detection Prevalence : 0.3249
##       Balanced Accuracy : 0.9562
##
##        'Positive' Class : acc
##
```

We get an accuracy of 95.3% with this model which is very good.

Random forest model

Lets see if we can get an increase accuracy by building a random forest model. A random forest model consists of a large number of decision trees that operate as an ensemble. Each individual tree in a random forest outputs a class prediction and the class with the most votes becomes our models prediction.

The random forest does lose the interpretability of the single decision tree model but hopefully makes up for it with increased accuracy.

We build the random forest model using the train function tuning it with different mtry (number of variables randomly sampled at each split) and ntrees (number of trees to grow) values.

```
# Build random forest model
tunegrid <- expand.grid(mtry = c(1:5), ntrees = c(50,100,150,200))
set.seed(1, sample.kind="Rounding")
```

```
## Warning in set.seed(1, sample.kind = "Rounding"): non-uniform 'Rounding' sampler
## used
```

```
random_forest <- train(class ~ ., data = train_set, method = "rf",
                               metric = "Accuracy",
                               tunegrid = tunegrid,
                               trControl = train_control)
```

Test the model on the test set and examine the confusion matrix

```
# USe random forest model to predict test set
test_pred_rf <- predict(random_forest, test_set)
confusionMatrix(test_pred_rf, as.factor(test_set$class))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc unacc
##      acc    72     6
```

```
##      unacc  11   188
##
##                Accuracy : 0.9386
##                  95% CI : (0.9036, 0.9638)
##     No Information Rate : 0.7004
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.8512
##
##  Mcnemar's Test P-Value : 0.332
##
##             Sensitivity : 0.8675
##             Specificity : 0.9691
##          Pos Pred Value : 0.9231
##          Neg Pred Value : 0.9447
##              Prevalence : 0.2996
##          Detection Rate : 0.2599
##    Detection Prevalence : 0.2816
##       Balanced Accuracy : 0.9183
##
##        'Positive' Class : acc
##
```

Our random forest model gets an accuracy of 0.9386 which is less than our decision tree model with the sensitivity (true positive rate) being the most significant decrease from 96.4% to 86.8%

Results

After determining that the single decision tree model gave us the best results in terms of accuracy we will apply this model to predict our validation set to determine our accuracy.

```
# Use decision tree model on the validation set
valid_pred <- predict(decision_tree, validation)
confusionMatrix(valid_pred, as.factor(validation$class))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc unacc
##      acc   104    17
##      unacc   0   225
##
##                Accuracy : 0.9509
##                  95% CI : (0.9225, 0.9711)
##     No Information Rate : 0.6994
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8883
##
##  Mcnemar's Test P-Value : 0.0001042
##
##             Sensitivity : 1.0000
##             Specificity : 0.9298
##          Pos Pred Value : 0.8595
##          Neg Pred Value : 1.0000
##              Prevalence : 0.3006
##          Detection Rate : 0.3006
##    Detection Prevalence : 0.3497
##       Balanced Accuracy : 0.9649
##
##        'Positive' Class : acc
##
```

An accuracy of 95.1% is achieved.

Conclusion

THe model described in this report successfully predicts the acceptability of of car with 95.1% accuracy.

The report decision tree model in order to obtain the classification which also has the benefit of being easily interpretable.

Some areas that could have been improved could have been to include more values in the tuning parameter for the random forest method which should net us a model with increased accuracy. Another thing could be to treat some of the variables as ordinal rather than just categorical such as buying price or number of doors to aid the model.