Predict Car Acceptability

Yogesh Kumar Malkoti

May 24, 2019

Introduction

This report is part of the capstone project of the EdX course 'HarvardX: PH125.9x Data Science: Capstone - Choose Your own project'. Its goal is to demonstrate that the student acquired skills with the R programming language in the field of datascience to actually solve realworld problems. The task is to analyse a Car Evaluation Data Set called 'cars' which contains 1728 data about car's criteria. The insights from this analysis are used to generate & Predict Car Acceptability which are compared with the actual ratings to check the quality of the prediction algorithm.

Summary

The report is split in three sections. First, the dataset is loaded and preformated for further analysis. Second, an exploratory datan alysis helps to understand the structure of the dataset. Finally, a machine learning algorithm creates predictions which are then exported for a final test. We haev four class of outcomes available which we have to predict from 6 predictors. A 'Random Forest' approach was used a final model after ensembling through variuos regression algorithms. This algorithm achieved an accuracy of 91% with decent Sensitivity and Specificity across the classes.

Data Set Information:

Car Evaluation Database was derived from a simple hierarchical decision model originally developed for the demonstration of DEX, M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.). The model evaluates cars according to the following concept structure:

Car Acceptability Factors.

- PRICE: overall price
  buying: buying price
  maint: price of the maintenance

- TECH technical characteristics
  COMFORT: comfort
  doors: number of doors
  persons: capacity in terms of persons to carry
  lug_boot: the size of luggage boot
  Safety: estimated safety of the car

Input attributes are printed in lowercase. Besides the target concept (CAR), the model includes three intermediate concepts: PRICE, TECH, COMFORT. Every concept is in the original model related to its lower level descendants by a set of examples.

The Car Evaluation Database contains examples with the structural information removed, i.e., directly relates CAR to the six input attributes: buying, maint, doors, persons, lug_boot, safety.

Because of known underlying concept structure, this database may be particularly useful for testing constructive induction and structure discovery methods.

Attribute Information:

Class Values:

unacc, acc, good, vgood

Attributes:

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.

Random Forest

Method Description

In Random Forests the idea is to decorrelate the several trees which are generated by the different bootstrapped samples from training Data. And then we simply reduce the Variance in the Trees by averaging them. Averaging the Trees helps us to reduce the variance and also improve the Performance of Decision Trees on Test Set and eventually avoid Overfitting. The idea is to build lots of Trees in such a way to make the Correlation between the Trees smaller.The effect of using all predictors is that each tree uses different predictors to split data at various times. This means that 2 trees generated on same training data will have randomly different variables selected at each split, hence this is how the trees will get de-correlated and will be independent of each other. Another great thing about Random Forests and Bagging is that we can keep on adding more and more big bushy trees and that won't hurt us because at the end we are just going to average them out which will reduce the variance by the factor of the number of Trees.

Step 1) Download MovieLens Data

I am using below url for the cars data set.

http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data

The dataset 'cars' gets split into a training-testset called 'train' and a set for validation purposes called 'validation'.

```
carfile<-"http://archive.ics.uci.edu/ml/machine-learning-databases/car/car.data"

if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse

## -- Attaching packages --------------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.1       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0

## -- Conflicts ------------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if(!require(Rborist)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: Rborist

## Rborist 0.1-17

## Type RboristNews() to see new features/changes/bug fixes.
```

```r
if(!require(ggpubr)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: ggpubr

## Loading required package: magrittr

##
## Attaching package: 'magrittr'

## The following object is masked from 'package:purrr':
##
##     set_names

## The following object is masked from 'package:tidyr':
##
##     extract
```

```r
dl <- tempfile()
download.file(carfile, dl)
cars <- str_split_fixed(readLines(dl), ",", 7)
colnames(cars) <- c("buying", "maint", "door", "persons", "lug_boot", "safety", "decision")

summary(cars)
```

```
##    buying      maint        door      persons     lug_boot    safety
##  high :432   high :432   2    :432   2   :576   big  :576   high:576
##  low  :432   low  :432   3    :432   4   :576   med  :576   low :576
##  med  :432   med  :432   4    :432   more:576   small:576   med :576
##  vhigh:432   vhigh:432   5more:432
##   decision
##  acc  : 384
##  good :  69
##  unacc:1210
##  vgood:  65
```

```r
car_frame<-as.data.frame(cars)

## lets create train and test data set out of it.

#y<-car_frame$decision

car_matrix<- createDataPartition(y=car_frame$decision, times = 1, p = 0.5, list = FALSE)

train<-car_frame[-car_matrix,]
```

```
validation<-car_frame[car_matrix,]

rm(dl)
```

Step 2) Exploratory Data Analysis

This sections helps to understand the structure of the cars dataset in order to use the insights for a better prediction of acceptance.

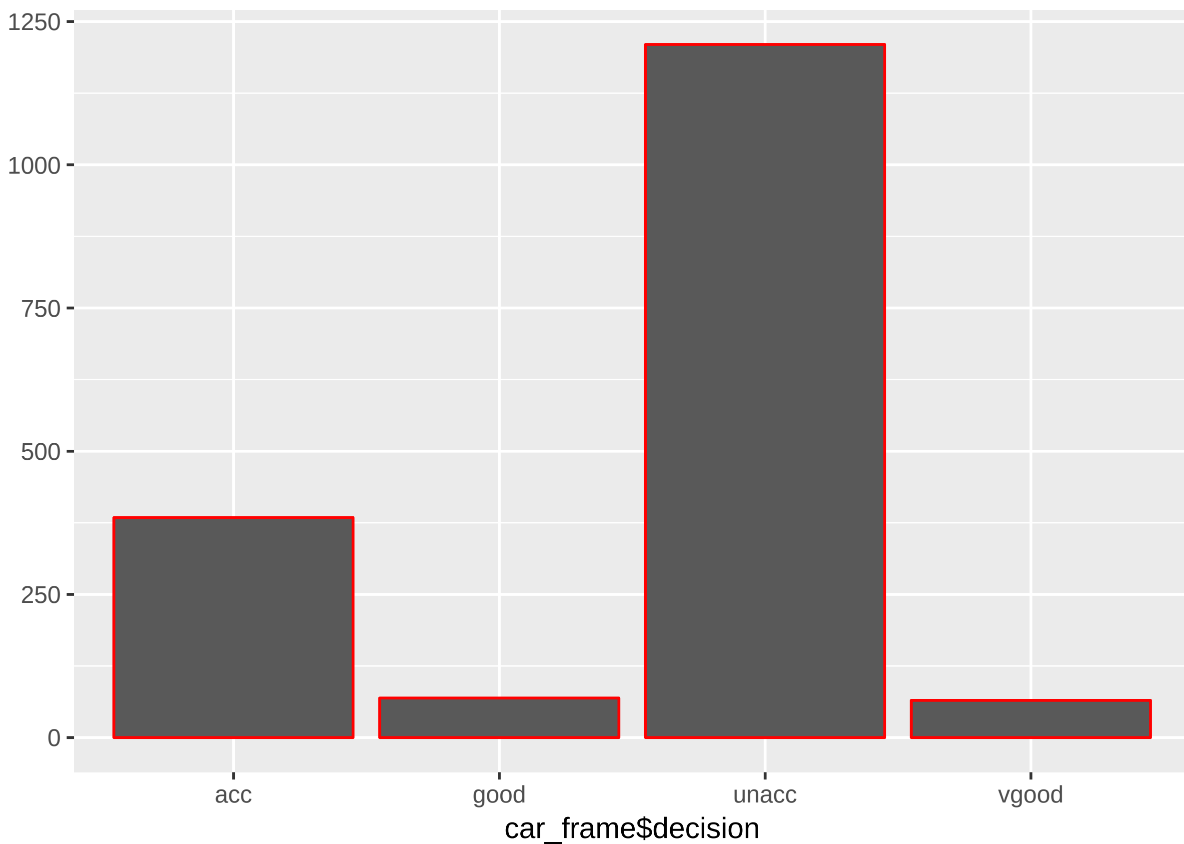
Let's find number of rows and columns in the train dataset.

```
paste('The Cars dataset has',nrow(train),'rows and',ncol(train),'columns.')
```

```
## [1] "The Cars dataset has 863 rows and 7 columns."
```

As we can see the train sample quit low so ML algorithms can be run on home run laptop/desktop.

```
qplot(car_frame$decision, col = I("red"))
```



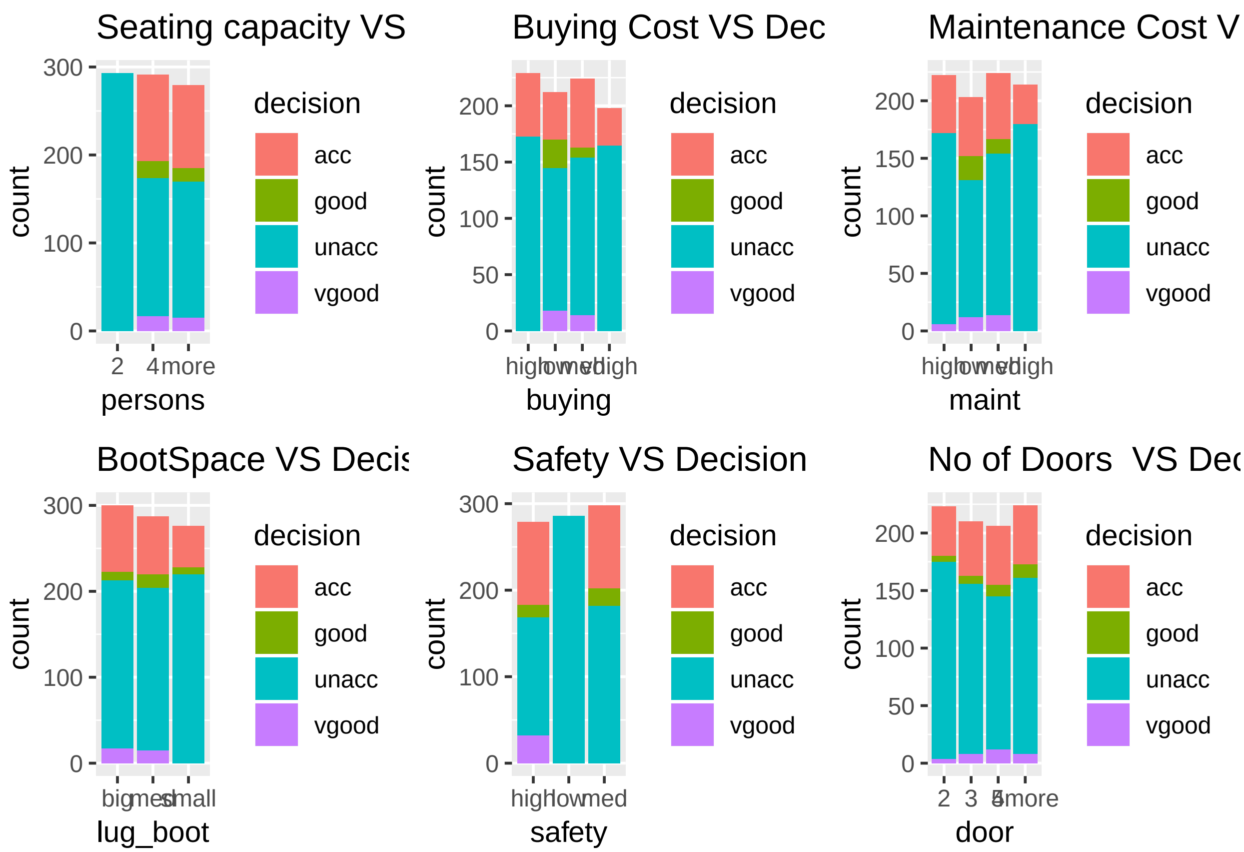

```
table(car_frame$decision)
```

```
##
##   acc  good unacc vgood
##   384    69  1210    65
```

We can see the data set contains most of the cceptability related to "unacc" class set.

```
plot_person<-train %>% ggplot(aes(x = persons, fill = decision)) + geom_bar() + ggtitle("Seating capaci
plot_buying<-train %>% ggplot(aes(x = buying, fill = decision)) + geom_bar() + ggtitle("Buying Cost VS I
plot_maint<-train %>% ggplot(aes(x = maint, fill = decision)) + geom_bar() + ggtitle("Maintenance Cost V
plot_logboot<-train %>% ggplot(aes(x = lug_boot, fill = decision)) + geom_bar() + ggtitle("BootSpace VS
plot_safety<-train %>% ggplot(aes(x = safety, fill = decision)) + geom_bar() + ggtitle("Safety VS Decis
plot_door<-train %>% ggplot(aes(x = door, fill = decision)) + geom_bar() + ggtitle("No of Doors  VS Dec


ggarrange(plot_person, plot_buying, plot_maint, plot_logboot, plot_safety, plot_door, widths = 1:1)
```



From this plot we can see that Seating Capacity and Safety are the two major factors along with other predcitors while making a decision on car acceptance. Hence we need to get higher sensitivity over these two factors than others.

Results

The challenge was to get the highest accuracy, along with higher sensitivity on two predictoes namely "Capacity" and "Safety". Here the outcome is a four class output with 6 predictors numeric as well as non numeric. Also from expolratory analysis of data we can see there is decision based approach of user while buying the vehicle depends on this factor. I tried a varietey of machine learning regression algorithms on the train data using ensebled approach and figured out Random Forest produng the best accuracy over train set of alost 99%. I then trained the RF model on train set and used on validation test with accuracy of 91% with very good sensitity.

Train the model with Random forest algorithm.

```
rtrain<-train(decision ~ ., method = "rf", data = train)
```

```
decision_predict<- predict(rtrain, train)
```

```
confusionMatrix(decision_predict, train$decision)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc good unacc vgood
##      acc   192    0     0     0
##      good    0   34     0     0
##      unacc   0    0   605     0
##      vgood   0    0     0    32
##
## Overall Statistics
##
##                Accuracy : 1
##                  95% CI : (0.9957, 1)
##     No Information Rate : 0.701
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: acc Class: good Class: unacc Class: vgood
## Sensitivity              1.0000      1.0000        1.000      1.00000
## Specificity              1.0000      1.0000        1.000      1.00000
## Pos Pred Value           1.0000      1.0000        1.000      1.00000
## Neg Pred Value           1.0000      1.0000        1.000      1.00000
## Prevalence               0.2225      0.0394        0.701      0.03708
## Detection Rate           0.2225      0.0394        0.701      0.03708
## Detection Prevalence     0.2225      0.0394        0.701      0.03708
## Balanced Accuracy        1.0000      1.0000        1.000      1.00000
```

As we can see the accruacy is about % along with sesitivty for person and securoty predictors.

Now lets use this trained model on validation test.

```
decision_predict<- predict(rtrain, validation)
```

```
confusionMatrix(decision_predict, validation$decision)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction acc good unacc vgood
##      acc   165   12    24     5
##      good    9   21     0     6
##      unacc  17    0   581     0
##      vgood   1    2     0    22
##
```

```
## Overall Statistics
##
##                Accuracy : 0.9121
##                  95% CI : (0.8913, 0.9302)
##     No Information Rate : 0.6994
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.8093
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: acc Class: good Class: unacc Class: vgood
## Sensitivity              0.8594     0.60000       0.9603      0.66667
## Specificity              0.9391     0.98193       0.9346      0.99639
## Pos Pred Value           0.8010     0.58333       0.9716      0.88000
## Neg Pred Value           0.9590     0.98311       0.9101      0.98690
## Prevalence               0.2220     0.04046       0.6994      0.03815
## Detection Rate           0.1908     0.02428       0.6717      0.02543
## Detection Prevalence     0.2382     0.04162       0.6913      0.02890
## Balanced Accuracy        0.8992     0.79096       0.9475      0.83153
```

Export Predictions

```r
# Ratings will go into the CSV submission file below:

write.csv(validation %>% mutate(predicted_decision = decision_predict),
          "car_predict.csv", na = "", row.names=FALSE)
```

Conclusion

The aim of the project was to predict the acceptance of cars given a six factors from a data containing it. We accuracy is measured as absolute difference between the predicted value and the acutal value. We used supervised learning approach to train the model woth Random Forest algorithm and got a pretty high accuracy.