```
##Data description

###
---------------------------------------------------------------------------
# This dataset contains information of cars purchased at the Auction.
# We will use this file to predict the quality of buying decisions and
visualize decision processes.

# VARIABLE DESCRIPTIONS:
#Auction: Auction provider at which the  vehicle was purchased
#Color: Vehicle Color
#IsBadBuy: Identifies if the kicked vehicle was an avoidable purchase
#MMRCurrentAuctionAveragePrice: Acquisition price for this vehicle in average
condition as of current day
#Size: The size category of the vehicle (Compact, SUV, etc.)
#TopThreeAmericanName:Identifies if the manufacturer is one of the top three
American manufacturers
#VehBCost: Acquisition cost paid for the vehicle at time of purchase
#VehicleAge: The Years elapsed since the manufacturer's year
#VehOdo: The vehicles odometer reading
#WarrantyCost: Warranty price (term=36month  and millage=36K)
#WheelType: The vehicle wheel type description (Alloy, Covers)
###
---------------------------------------------------------------------------


# 1. Import the datadet
carAuction <- read.csv(file = "carAuction.csv", stringsAsFactors = FALSE)

# 2. str() shows the structure of data
str(carAuction)

# 3. summary() shows the mean and the five-number statistics indicating the
spread of each column's values
summary(carAuction)

# 4. Change all categorical variables to factors
carAuction$Auction <- factor(carAuction$Auction)
carAuction$Color <- factor(carAuction$Color)
carAuction$IsBadBuy <- factor(carAuction$IsBadBuy)
carAuction$Size <- factor(carAuction$Size)
carAuction$TopThreeAmericanName <- factor(carAuction$TopThreeAmericanName)
carAuction$WheelType <- factor(carAuction$WheelType)
str(carAuction)
summary(carAuction)

# 5. Partition the dataset for Decision Tree model
#caret package includes particioning function to split data
library(caret)#this function loads the package
set.seed(1)
#if you select train_index it will show the numbers of rows which will go into
#training dataset, those that not included will go into testing
train_index <- createDataPartition(carAuction$IsBadBuy, p=0.7, list=FALSE)
#p=7 means that 70% of data will go into training dataset and 30% for testing
datTrain <- carAuction[train_index,]#training
datTest <- carAuction[-train_index,]#testing; negative train_index means
```

```
#that rows that are not included into training will go here.

# 6. Check the rows and porportion of target variable for both training and
testing datasets
nrow(datTrain)
nrow(datTest)
prop.table(table(datTrain$IsBadBuy))#distribution of the target variable
#in the training
prop.table(table(datTest$IsBadBuy))#distrib. in testing.

# 7. Build C50 models
#includes C5 function that can build a decision three model based on car
auction dataset
#rminer package help
library(C50)#load the package first! C5.0 included in this package.
library(rminer)
model <- C5.0(IsBadBuy~.,data=datTrain)#ISBadbuy is our target variable
#the one we want to predict. data=datTrain is we specify our training dataset.
#"model" is the name of our deicison three model
#it will show the number of variables (predictors)which we will use to
#predict
#three size shows the number of leaf nodes
model
dev.off()
plot(model)#!!!This shows the following error:
#Error in .Call.graphics(C_palette2, .Call(C_palette2, NULL)) :
#invalid graphics state
#I had to run this dev.off()to solve the error

summary(model)

# 8. Make predictions on both training and testing sets
prediction_on_train <- predict(model, datTrain)#make predictions on the train
#data set first.
prediction_on_test <- predict(model, datTest)#we are applying deicison tree
#model on the training dataset.

# 9. Results comparison
mmetric(datTrain$IsBadBuy,prediction_on_train, metric="CONF")#first we want to
#look at confusion matrix

#The row sum 1 will show number of badbuyvalue out of 7001 are equal to NO.
#Second row sum will show how many bad buy equal to YES. ROws are REAL VALUES.
#COLUMN SUMS show how many cars we PREDICT badbuy eqal to no and yes. Ex: we
#predict 245 cars would be badbuy equal to yes. So 32 cars we predicted that
#badbuy eqal to yes, but they actually are good (badbuy no). 6275 correct
predic
mmetric(datTrain$IsBadBuy,prediction_on_train,c("ACC","PRECISION","TPR","F1"))
#Precision shows accuracy of predictions on training data set.
#Proportion of cars that we predicted correctly. Sum of correct / total
sampels
#Precition1 shows proporiton of predicted badbuy "no"cars. It represents
confidence.
#Precision2 shows proportion of predicted badbuy equal to yes.
#TPR1 shows that Model corectly identified 6062 bad buy "no" cars which is
99%.
```

```
#BUt, TPR2 shows for bad buy equal to yes
#we only identified 213 cars which is 23% (since real number is 907;
907/213=23%
#F11 and F12 showing the means among precions and tprs (recall value). they
give
#overal performance.
#next two lines are prediction results on testing data
mmetric(datTest$IsBadBuy,prediction_on_test, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test,metric=c("ACC","PRECISION","TPR","F1"))

# 10. Build C50 decision tree with CF = 0.5
#CF 0.5 is a confidence factr. The bigger it is the bigger the three is.
#Default value is 0.25, so this one is bigger.
model2 <- C5.0(IsBadBuy~.,data=datTrain, control = C5.0Control(CF=0.5))
dev.off()
plot(model2)#!!!This shows the following error:
summary(model2)

# 11. Make predictions on both training and testing sets
prediction_on_train2 <- predict(model2, datTrain)
prediction_on_test2 <- predict(model2, datTest)

# 12. Compare the evaluation results on training and testing sets
mmetric(datTrain$IsBadBuy,prediction_on_train2, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train2,c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test2, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test2,metric=c("ACC","PRECISION","TPR","F1"))

  # a. Does the decision tree model have better performance on training set or
#testing set? why?
#The performance of this decison three model is better on the training
dataset.
#Accuracy, precison, and recall are higher on the training dataset.

  # b. Does the decision tree model have better performance on majority
#(IsBadBuy = 'No') or minority class (IsBadBuy = 'Yes')? why?
#Yes, this model has better perforamcne on majority class, because precion,
#recall, and means (f11)are higher for the majority class (badbuy = NO)

# 13. Build C50 decision tree with 8 predictors (removing WheelType and
Auction) and set CF = 0.3
model3 <- C5.0(IsBadBuy~.,data=datTrain[,c(-1,-11)], control =
C5.0Control(CF=0.3))
dev.off()
plot(model3)#!!!This shows the following error:
summary(model3)


  # a. How many decision nodes and how many leaf nodes are in the tree?
#we have 6 deicison nodes and 8 leaf nodes.
  # b. Compare it to the C50 tree generated in task 7,  is it more or less
complex? Give reasons for your answer.
#This model has 8 leaf nodes, while the first one only had 3 leaf nodes.
#so this model is more complex.
  # c. What is the predictor that first splits the tree? How the decision tree
selects the first predictor to split?
```

```
#The first variable that splits is Vehiclecost (VehBCost). The reason its
#chosen as the first variable to split the decison tree is because it
#provides the highest information gain. It can lower the entropy the most.
  # d. Find one path in the tree to a leaf node that is classified to IsBadBuy
= 'Yes'. What is this path/rule's misclassification error rate?
#look at the summary on console. If VehBCost is less than 4010 and color is in
#brown or purple, then Badbuy is equal to YES.

# 14. Make predictions on training and testing sets
prediction_on_train3 <- predict(model3, datTrain)
prediction_on_test3 <- predict(model3, datTest)


# 15. Generate the evaluation results on training and testing sets
mmetric(datTrain$IsBadBuy,prediction_on_train3, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train3,c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test3, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test3,metric=c("ACC","PRECISION","TPR","F1"))


  # a. On the testing set, how many bad buy cars are predicted as Not bad buy?
#On the testing set: 2990 cars are PREDICTED BadBuy = "NO".
  # b. Compared to the decision tree model generated in task 7,
#which one (model in task 7 or task 13) has better performance, and why?

#model 1 is more accurate (89 vs 86%). F12, TPR2 are considerably higher in
#model 1 as well. They show overall performance and number of correct
predictons
#for badbuy equal to yes.
#Model 1 is definately better than the current model.
```