```
##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)#shows the structure of the data

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

# 4. Change all categorical variables to factors
#DO THIS to transform categorical variables; from vectors 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: 70% for training, 30% for testing
library(caret)#we load caret package which we will use here
set.seed(1)
#next line splits data into testing and training data
train_index <- createDataPartition(carAuction$IsBadBuy, p=0.7, list=FALSE)
datTrain <- carAuction[train_index,]
datTest <- carAuction[-train_index,]

# 6. Check the rows and proportion of target variable for both training and
```

```
#testing datasets
nrow(datTrain)#checks number of rows in traning data
nrow(datTest)
prop.table(table(datTrain$IsBadBuy))#shows proportions BadBuy = Yes and NO
prop.table(table(datTest$IsBadBuy))

# 7. Build NB models with laplace = 1
library(e1071)#we will use e1071 it includes NB model
#library means "load the package"
library(rminer)#we will use rminer package to evaluate the model performance
model <- naiveBayes(IsBadBuy~.,data=datTrain, laplace = 1)#isbadbuy our target
#variable, we will only use training data to build our naiveBayes model.
model#when we run this model it gives us all the a-priory and conditional
#probabilities

  # a. What are the prior probabilities of IsBadBuy = No and IsBadBuy = Yes?
#No: 0.8704471
#Yes:0.1295529
  # b. What are the conditional probabilities of
#P(WheelType = unkwnWheel|IsBadBuy = Yes) and P(WheelType = unkwnWheel|
IsBadBuy = No)?
#P(WheelType = unkwnWheel|IsBadBuy = Yes) = 0.267837541
#P(WheelType = unkwnWheel|IsBadBuy = No)? = 0.013938996

  # c. For a new car X = (WheelType = unkwnWheel, Auction = OTHER, Color =
GOLD)
#, we can calculate (the problem is asking to calculate probab of badbuy = NO,
#given these conditions)

  # P(IsBadBuy = No|X) ??? P(X|IsBadBuy = No) * P(IsBadBuy = No) =
#Answer:
##probablity of bad buy given these conditions can be simplified:
#= P(WheelType = unkwnWheel|IsBadBuy = No)*P(Auction = OTHER|IsBadBuy = No)*
#*P(Color = GOLD|IsBadBuy = No)*P(IsBadBuy = No) =
# = 0.013938996*0.2443825*0.069067103*0.8704471 = 0.0002047931
#
  # P(IsBadBuy = Yes|X) ??? P(X|IsBadBuy = Yes) * P(IsBadBuy = Yes) =
#Answer:
#= P(WheelType = unkwnWheel|IsBadBuy = Yes)*P(Auction = OTHER|IsBadBuy = Yes)*
#*P(Color = GOLD|IsBadBuy = Yes)*P(IsBadBuy = Yes) =
#0.267837541*0.2219780*0.081256771*0.1295529 = 0.0006258757
# What is the prediction result based on your calculation?
#Answer:
#Our conclusion is, based on the conditions, the probability that this care is
BADBUY=Y
#is greater than BadBuy=No, so we predict that this car is BadBuy = Yes.


# 8. Make predictions on both training and testing sets
#Note: for this task we using naivebase model to predict on the training
dataset
#and we will save predictino results on datTrain and Test respectfully
prediction_on_train <- predict(model, datTrain)
prediction_on_test <- predict(model, datTest)

# 9. Results comparison
```

```
mmetric(datTrain$IsBadBuy,prediction_on_train, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train,metric=c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test,metric=c("ACC","PRECISION","TPR","F1"))

  # a. Does the naive bayes model have better performance on training set or
testing set? why?
#Note: first precision, first recall, first Fmeasure are for BadBuy=No class
#Answer:
#Training set has better overall performance based on accuracy. (accuracy
higher
#on the training dataset).
#Naive base model has a higher precision and overall performance on both
#classes (yes and no)for the
#training dataset rather than for test dataset. However, the model has lower
#value for the recall (TPR1) for BadBuy = No, than on the test dataset.

  # b. Does the naive bayes model have better performance on majority
#(IsBadBuy = 'No') or minority class (IsBadBuy = 'Yes')? why?
#Note: Only use the TESTING data to compare the performance of the classes.
#Answer:
#It has much better performance on the majority (BADBUY = NO) than on
minority.
#IT has higher percesion, recall and f measure for badbuy NO class.

 # c. How many bad buy cars are identified by the naive bayes model in testing
data?
#Note: look at the confusion matrix;
#rows are actual cars in the data, columns are the predictions.
#This question is asking how many we predict are badbuys that actually are
#badbuys.
#Answer:
#We correctly predicted 101 cars as bad buy YES.

  # d. How many cars are predicted as bad buy in testing data?
#Again in the testing data, if the naive bayes predicts a car as bad buy,
#what is the probability that such prediction is correct?
#Note: look at confusion matrix;
#Answer: Overall we predicted that 222 cars were bad buy equal YES.
#THe probability is the precision #Precision2 (101/222) on the test dataset.
#Probability is 45.49550.

# 10. Build decision tree model with cp = 0.0001, maxdepth = 1.
library(rpart)
library(rpart.plot)#we uuse rpart package because we can specify depths and
#it will be easire for our decison trees.
rpart_model <- rpart(IsBadBuy~.,data = datTrain,control = rpart.control(cp =
0.0001, maxdepth = 1))
rpart.plot(rpart_model)
rpart_model

# 11. Make predictions on both training and testing sets
prediction_on_train_rpart <- predict(rpart_model, datTrain)
prediction_on_test_rpart <- predict(rpart_model, datTest)

# 12. Compare the evaluation results on training and testing sets
```

```
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test_rpart, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart,metric =
c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test_rpart,metric=c("ACC","PRECISION","TPR","F1"))


# 13. Build decision tree model with cp = 0.0001, maxdepth = 2.
#NOte: change the name of the model.
rpart_model2 <- rpart(IsBadBuy~.,data = datTrain,control = rpart.control(cp =
0.0001, maxdepth = 2))
rpart.plot(rpart_model2)
rpart_model2

# 14. Make predictions on both training and testing sets
#Also change the name of rpart
prediction_on_train_rpart2 <- predict(rpart_model2, datTrain)
prediction_on_test_rpart2 <- predict(rpart_model2, datTest)

# 15. Compare the evaluation results on training and testing sets
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart2, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test_rpart2, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart2,metric =
c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test_rpart2,metric=c("ACC","PRECISION","TPR","F1"))

# 16. Build decision tree model with cp = 0.0001, maxdepth = 3.
rpart_model3 <- rpart(IsBadBuy~.,data = datTrain,control = rpart.control(cp =
0.0001, maxdepth = 3))
rpart.plot(rpart_model3)
rpart_model3


# 17. Make predictions on both training and testing sets
prediction_on_train_rpart3 <- predict(rpart_model3, datTrain)
prediction_on_test_rpart3 <- predict(rpart_model3, datTest)

# 18. Compare the evaluation results on training and testing sets
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart3, metric="CONF")
mmetric(datTest$IsBadBuy,prediction_on_test_rpart3, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart3,metric =
c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test_rpart3,metric=c("ACC","PRECISION","TPR","F1"))

# 19. Build decision tree model with cp = 0.0001, maxdepth = 4.
rpart_model4 <- rpart(IsBadBuy~.,data = datTrain,control = rpart.control(cp =
0.0001, maxdepth = 4))
rpart.plot(rpart_model4)
rpart_model4

# 20. Make predictions on both training and testing sets
prediction_on_train_rpart4 <- predict(rpart_model4, datTrain)
prediction_on_test_rpart4 <- predict(rpart_model4, datTest)

# 21. Compare the evaluation results on training and testing sets
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart4, metric="CONF")
```

```
mmetric(datTest$IsBadBuy,prediction_on_test_rpart4, metric="CONF")
mmetric(datTrain$IsBadBuy,prediction_on_train_rpart4,metric =
c("ACC","PRECISION","TPR","F1"))
mmetric(datTest$IsBadBuy,prediction_on_test_rpart4,metric=c("ACC","PRECISION","TPR","F1"))



  #Compare the performances of decision tree model with different maxdepth
  # a. Does the decision tree model with maxdepth = 2 generalize well on
#the testing set? why?
#Note: comapre accuracy of the second tree with the other ones.
#Answer:
#We can see that the training and testing results for 2nd deciion tree are
#pretty close; it menas that this tree generalized well on the training
#and testing data.
#We can see that as the tree becomes bigger, the traiing performance will
always
#increase.
#For the tesing, we can see that first it increasees a littile bit for the
#testing dataset with increase in tree size. Then 2nd tree and 3d tree, the
#accuracy is equal. Then it decreases for the 4th decison tree.
#We have the highest accuracy for the 2nd tree (matched with 3d)

  # b. If you are a car dealer, which decision tree model you will use, and
why?
#Answer:
#When making such decison it is good to look at true positive rate (aka Recal
#value or TPR). You can look at hte TPR and which tree can best help you
#identify badBUy car so you can avoid them. First tree, has the highest
#recall value TPR2 on the testing data (24.48).It means that it can
#corectly identify and avoid badbuy cars.
```