Problem 9.3 Predicting Prices of Used Cars (Regression Trees). The file

ToyotaCorolla.csv contains the data on used cars (Toyota Corolla) on sale

during late summer of 2004 in the Netherlands. It has 1436 records containing

details on 38 attributes, including Price, Age, Kilometers, HP, and other

specifications. The goal is to predict the price of a used Toyota Corolla

based on its specifications. (The example in Section 9.7 is a subset of this

dataset).

Data Preprocessing. Split the data into training (60%), and validation (40%)

datasets.

car.df <- read.csv("C:/Users/sergi/Documents/dmclass/DMBAdata/ToyotaCorolla.csv")

str(car.df)

here you want to get you samples set up

preprocess - we will use the same method from before

set.seed(1)

we do this to be able to reproduce the same results since we are

randomly sampling

get the percentage of train here it is 0.6

percnt.of.data<- 0.6

train.index <- sample(c(1:dim(car.df)[1]), dim(car.df)[1]\*percnt.of.data)

the validation is the remaining rcord

valid.index <- setdiff(c(1:dim(car.df)[1]), train.index)

We now use the indexes set up to reference the rows we are sampling

train.df <- car.df[train.index, ]

valid.df <- car.df[valid.index, ]

9.3.a Run a regression tree (RT) with outcome variable Price and predictors

Age\_08\_04, KM, Fuel\_Type, HP, Automatic, Doors, Quarterly\_Tax, Mfr\_Guarantee,

Guarantee\_Period, Airco, Automatic\_Airco, CD\_Player, Powered\_Windows,

Sport\_Model, and Tow\_Bar. Keep the minimum number of records in a terminal

node to 1, maximum number of tree levels to 30, and cp = 0:001, to make the

run least restrictive.

based on the parameters defined the regression tree call will be:

tr <- rpart(Price ~ Age\_08\_04 + KM + Fuel\_Type +

HP + Automatic + Doors + Quarterly\_Tax +

Mfr\_Guarantee + Guarantee\_Period + Airco +

Automatic\_airco + CD\_Player + Powered\_Windows +

Sport\_Model + Tow\_Bar, data = train.df,

the following is the difference from the generic form

method = "anova", minbucket = 1, maxdepth = 30, cp = 0.001)

generate the tree

prp(tr)

the following code generates the rules

tr

9.3.a.i Which appear to be the three or four most important car specifications

for predicting the car's price?

using str we can identify the components of the tree

str(tr)

among other things we can get the variable importance

variable importance

we take the transpose of the "vector" to get the row and the

transpose of the gives us column vector with the variable labels

t(t(tr$variable.importance))

note your results may be different based on the random seed for your system

> t(t(tr$variable.importance))

[,1]

Age\_08\_04 9885745349

Automatic\_airco 2987651311

KM 2886703313

Quarterly\_Tax 1885088412

HP 1665897142

Guarantee\_Period 430364046

CD\_Player 357332146

Fuel\_Type 206914123

Airco 122474786

Powered\_Windows 109250988

Doors 63966015

Mfr\_Guarantee 21300178

Automatic 19776457

Sport\_Model 3395399

From the regression tree output briefly discuss the top five below:

[Note: It has been pointed out that there is a value in the cc variable - 16,000 -

that is probably a data input error. The solutions have been prepared without

correcting this error, but a solution that includes correcting this error to

1600 would also be fine.

9.3.a.ii Compare the prediction errors of the training and validation sets by

examining their RMS error and by plotting the two boxplots. How does the

predictive performance of the validation set compare to the training set?

Why does this occur?

errors

need to use the forecast and ggplot2 libraries

you have most likely installed them

install.packages("forecast")

install.packages("ggplot2)

library(forecast)

library(ggplot2)

training accuracy

using the code for accuracy

we check the training accuracy - the results are those based on the errors

your results may be different due to the differences in sampling

accuracy(predict(tr, train.df), train.df$Price)

> accuracy(predict(tr, train.df), train.df$Price)

ME RMSE MAE MPE MAPE

Test set -7.061715e-14 986.5121 760.0899 -1.006552 7.671931

to get the validation accuracy use use the valid.df$Price

accuracy(predict(tr, valid.df), valid.df$Price)

> accuracy(predict(tr, valid.df), valid.df$Price)

ME RMSE MAE MPE MAPE

Test set 11.42479 1192.877 907.2528 -1.008939 9.038641

to get the the errors subtract the prediction from the actual

train.err <- predict(tr, train.df) - train.df$Price

valid.err <- predict(tr, valid.df) - valid.df$Price

create a data frame from the training and validation errors

we lable the errors with associate labels - Training and Validation

err <- data.frame(Error = c(train.err, valid.err),

Set = c(rep("Training", length(train.err)),

rep("Validation", length(valid.err))))

the ggplot will use the error data frame for boxplots

ggplot(err, aes(x = Set, y = Error)) + geom\_boxplot()

Briefly discuss the differences between training and validation outcomes

9.3.a.iii How might we achieve better validation predictive performance at

the expense of training performance?

Provide your answers here

9.3.a.iv Create a less deep tree by leaving the arguments cp, minbucket, and

maxdepth at their defaults. Compared to the deeper tree, what is the

predictive performance on the validation set?

shallower tree -use the same as above but omit the

parameters after "method" call it tr.shallow

enter your code here

prp(tr.shallow)

determine the training accuracy of the prunned tree

training accuracy of prunned tree based on the similar code

above use tr.shallow instead of tf

output

ME RMSE MAE MPE MAPE

Test set 2.11238e-13 1396.864 1010.652 -1.735082 9.987881

do the same for the validation

the output

ME RMSE MAE MPE MAPE

Test set 35.55491 1341.264 1022.343 -1.05491 9.829217

We see that, as expected, compared to the deeper tree, the best pruned tree

performs worse on the training set (RMSE=1397 compared to 987 for the deep tree;

MAE=1010 vs. 760), and not so bad on the validation set (RMSE=1341 compared to

1192; MAE=1022 vs. 907)

9.3.b Let us see the effect of turning the price variable into a categorical

variable. First, create a new variable that categorizes price into 20 bins.

Now repartition the data keeping Binned\_Price instead of Price. Run a

classification tree with the same set of input variables as in the RT, and

with Binned\_Price as the output variable. As in the less deep regression

tree, leave the arguments cp, minbucket, and maxdepth at their defaults

create the bins based on the car prices

categorical price

bins <- seq(min(car.df$Price),

max(car.df$Price),

(max(car.df$Price) - min(car.df$Price))/20)

determine the number of bins based on the min and max prices

lets see the bins

bins

the following uses bincode to determine the bin assignments

if you vie Binned\_Price you can see the assignments

Binned\_Price <- .bincode(car.df$Price,

bins,

include.lowest = TRUE)

we convert the Binned\_Price to factors for classification

Binned\_Price <- as.factor(Binned\_Price)

Binned\_Price

we add the binned price to the data frame based on the

indexes this will allow us to identify the training and validation

data frames

train.df$Binned\_Price <- Binned\_Price[train.index]

valid.df$Binned\_Price <- Binned\_Price[valid.index]

9.3.b.i Compare the tree generated by the CT with the generated by the less

deep RT. Are they different? (Look at structure, the top predictors, size of

tree, etc.) Why?

use the same a 9.3.a.iv but omit the "method parameter and

the target is Binned\_Price

assuming your result is called tr.binned

we can vie the classification tree

classification tree

prp(tr.binned)

generate the tree rules like before

tr.binned

view the regression tree similarly

regression tree

prp(tr)

tr

briefly discuss the differences between the classification tress

and the regression trees

9.3.b.ii Predict the price, using the less deep RT and the CT, of a used

Toyota Corolla with the specifications listed in Table 9.6.

for this we create a data frame for the new record

predict price

new.record <- data.frame(Age\_08\_04 = 77,

KM = 117000,

Fuel\_Type = "Petrol",

HP = 110,

Automatic = 0,

Doors = 5,

Quarterly\_Tax = 100,

Mfr\_Guarantee = 0,

Guarantee\_Period = 3,

Airco = 1,

Automatic\_airco = 0,

CD\_Player = 0,

Powered\_Windows = 0,

Sport\_Model = 0,

Tow\_Bar = 1)

using regression tree

price.tr <- predict(tr, newdata = new.record)

price.tr

> price.tr

1

7318.056

using classificaation tree

since we have bins we want to bin the prediction

price.tr.bin <- bins[predict(tr.binned, newdata = new.record, type = "class")]

price.tr.bin

> price.tr.bin

[1] 7165

discuss the two outputs