Author: Ludovic Henry

2.1/ Data Mining Techniques

a. supervised

b. supervised

c. supervised

d. unsupervised

e. supervised

f. supervised

g. supervised

h. supervised

2.2/ Data Partition

You can use the validation partition to tune the model, while you cannot do the same with the test partition. This is to avoid having the test partition influence the model, and to avoid an overfitting of the model on the test partition.

2.3/ Data Sample

The “OBS” column has constant increment of 8 which seems to indicate that the values weren’t randomly selected, but instead every 8th row was picked.

2.4/ Modeling Steps

Following the Modeling Process in chapter 2.6 of the book, the purpose has been determined, the data has been obtained, the data seems appropriate and reduced (the number of variables is fairly low, even if some could be redundant), the data mining task has been defined (“identify likely responders”), and the data has been partitioned. The next step in the process is now to choose the technique to apply for the model, and eventually try and compare multiple ones to identify the most performant.

Some outliers might still reside in the data (OBS 21 and 26 for example, with mortgages much greater than the income), but my lack of domain knowledge is limiting my ability to identify these as certain or uncertain outliers.

2.5/ Overfitting

By having zero error on the training partition, we create a model which is not necessarily able to correctly predict an outcome on a different data set or different partition. This is because it has been "overfitted" to the training partition, and is only able to predict outcome from what it has been trained on.

2.6/ Data Leakage

The model will have learned and will predict outcome based on both the demographic and the purchase data. If the latter is not present when using the model, it won't be able to predict an appropriate outcome since its decision will be based on variables not currently available.

2.11/ ToyoataCorolla.csv only "a."

a. Some examples of variables which seem correlated are:

- Age\_08\_04 and KM

- Age\_08\_04 and Pricew

- Fuel\_Type, CC and HP

- Fuel\_Type and KM

b.

i. You get a list containing every possible values, then attribute to each element in this list a unique dummy, and finally use these dummies to map back all the values in the original dataset.

df$Fuel\_Type\_Dummy <- df$Fuel\_Type %>% (function(v) return(match(v, unique(df$Fuel\_Type))))

ii. The training partition will be used to train the model, the validation partition will be used to compare different models and their respective performance (it will act as a feedback loop for the training partition), and the test partition will be used to verify the performance of the model without having any influence on the model picked.

# This is to guarantee we select the same elements everytime, even if randomly

set.seed(1)

# Randomly shuffle the data

df.random <- sample(df)

# Pick the first 50% of elements

df.training <- dfRandom[(1):(floor(.5 \* nrow(df))),]

# Pick the next 30% of elements

df.validation <- dfRandom[(floor(.5 \* nrow(df)) + 1):(floor(.8 \* nrow(df))),]

# Pick the last 20% of elements

df.test <- dfRandom[(floor(.8 \* nrow(df)) + 1):(nrow(df)),]