MEMORANDUM — Topic Introduction
To: Big Data Analytics students
From: Prof. Roger Bohn

Subject: Week 4 on Linear Regression and using R

Date: April 23, 2018.

This week we have 3 learning goals. It will take the entire week to do them.

1. Linear regression for prediction. How prediction differs from hypothesis testing.
2. Showing how to use R instead of, or in conjunction with, Rattle. We will be converting gradually over to R.
3. Many specific tricks and issues that come up with linear regression, such as word equations and creating interactive variables. Several memos are attached at the end of this assignment, with tips on these topics. View these as supplements to the textbook.

8

Correlation multiplots

??

Handling nonlinear causes

??

MEMORANDUM
To: BDA students
From: Prof. Roger Bohn
Subject: Homework for week 4, days 7 and 8: linear regression models
Date: April 23, 2018

In weeks 2 and 3 we looked at *classification* models, which predict a binary outcome. This week we look at more familiar terrain, linear models based on least squares analysis, used to predict *continuous* outcomes.

This assignment brings up a number of issues that we will revisit later in the course. For example, how to decide what variables to include. In your write-up, explicitly list "open issues" that you have not resolved to your own satisfaction. In all data mining projects, there are more ideas you *could* investigate than you actually have time for. You cannot do everything!

Reading:
The assigned reading for this homework is Gareth James et al, *An Introduction to Statistical Learning with Applications in R.* (Abbreviated ISLR .) It's available from . Review Chapter 3 "Linear Regression," section 3.2 which should be familiar, and read section 3.3 on variations on the basic model linear. Also read DMBA main textbook, only sections 6.1 and 6.2
 You can get away with reading only one of the texts, if you are already familiar with logistic regression.

Total material for this problem set:
1) Assigned reading in two textbooks,.
2) The dataset itself, about 30,000 automobiles. On the course web site.
3) Documentation of that dataset. On the fuel economy.gov web site.
4) This assignment.
5) Memo on how I cleaned the data.
6) Your old textbooks on OLS regression. In particular, you may want to study the discussions of dummy variables and variable interactions.

Problem set part A: Introduction

This homework uses data I pulled from the EPA web site, on car mileage. The data is in a CSV Excel file. You may need to play with the file before you import it into Rattle, such as removing the first few rows. Other data manipulation can be done inside Rattle (or even more, inside R). For example, some columns are redundant. The file can be accessed from the t data sets page for the course,https://bda2020.wordpress.com/data-sets/ . It's 12 MB so it might take some time to download. The spreadsheet has various information in it, and you will need to "clean it up" a little, either in Excel or in Rattle. (See discussion below.) Import it into Rattle as a .csv file.

Be sure to read the EPA's description of the variables, at http://www.fueleconomy.gov/feg/ws/index.shtml#vehicle . The current (2018) version of this description may be slightly different than the data, which was captured in 2016. I have color coded some variables to consider in your analysis, and you can fine-tune the list of included and excluded variables yourself.

As with the Toyota Corolla data, pay attention to whether a variable like Cylinders should be categorical (a set of dummy variables), or continuous. Here is the basic concept to use: Do you think that going from 4 to 6 cylinders will have approximately the same effect as going from 6 to 8? Both are a   change of 2 "units." If you think the answer is YES, then make cylinders a numerical variable. If not, make it a category. (You could even do it both ways.)

Your approach might be especially important for "year." You certainly don't want to create a category for each year, but on the other hand, the rate of improvement in gasoline mileage may be very different in the early years, than in the later years. See part "C" below for some ideas on how to handle this.

By the way, Rattle and R handle categorical variables internally. You do not need to manually convert them to dummy variables, as you do in Stata.

Problem set part B **Which variables to include?**

With the CART algorithm, we made a big point of "letting the model choose which variables to include." That worked because CART decides on variable importance, and as it "goes down the tree" it always picks variables that seem to be important. So when CART stops, many variables may still be excluded.

That does not work with simple linear models! In a linear model, *all* variables that are allowed will automatically get some coefficient assigned to them. So if your model includes a lot of "garbage can variables" such as the number of storks or the number of sunspots, it will overfit. *Unlike in CART models, one of the principal ways to tune a linear model is to add interaction variables and subtract other variables.*

An example to investigate if you have time: Should you include or exclude the "car model" variable, column AW? For some brands, there are only a few cars of each model, so this risks overfitting. You can find out whether including or excluding is better by using the validation RMS error. (R and Rattle will handle a multi-level category like this by automatically creating a large number of 0-1 dummy variables.)

See part E at the bottom: I actually did this part for you, mostly.

Problem set part C: Estimate fuel efficiency

Estimate the model for predicting Miles Per Gallon (MPG), using a training set of approximately 70 percent and a validation set of approximately 30 percent of the data.

- Make sure to exclude variables that are another version of the gasoline efficiency, such as city and highway mileage. You may decide to exclude other columns as well.
- Clearly show *one* example of the equation estimated by your model. You do not have to show every version that you try.
- Rewrite your best model *as a word equation* such as MPG = 3.1 x Engine displacement - 4.5* # of gears +{0 if 2012 year, 1.5 if 2014 year, -.3 if 2013 year} + {0 if rear wheel drive, 1.19 if front wheel drive, 4.33 if 4 wheel drive, 5.82 if all wheel drive}
- Give the units for each variable. (e.g. tons or pounds).
- What else does the training output tell you?

When the target is a continuous variable, Rattle uses conventional OLS. The Rattle book by Williams does not talk about it much because it's covered in other courses. Just go ahead and click on "Linear" in the "Model" tab.
Use the *validation* data to decide if your model is working well. Report the predicted RMS error (on the validation data) as your "model performance" estimate for your best model. Also try to calculate the Mean Percentage Distribution, using R.

Problem set part D: variable interactions.

Give some thought to transformations of the predictor variables. List the transformations you end up choosing. Good transformations, such as creating interactions among certain variables, can greatly improve predictions. This is discussed briefly in the textbook assignments, but everything you learned in QM still applies. (In the CART tree method, there is no need to create interactions, since the CART algorithm figures them out automatically. That is one of the big advantages of some non-linear algorithms.)

The basic criterion for whether to include an interaction is "Does the model fit get better with that interaction in the model?" You can think about the set of "all possible interactions" as an additional hundred (very roughly) variables that you could decide to include in the model, or not.

Effect of year: In particular, we know that over the years cars got "more efficient" due to more aerodynamic body designs, more efficient engines, higher tire pressures, lighter materials, and many other changes. Certainly "year" must be included as a predictor variable. But it's also likely that the effects of some of the predictor variables will change from year to year. For example, front-wheel drive is generally more efficient, but the difference between front and rear-wheel drive may change over time.

The "poor person's approach" to this would be to set up separate runs of the model, by decade or 5 year interval. For example, run the model for 2000 to 2004, and 2010 to 2014, and you will expect to see different betas on some coefficients. Does such a division of the data set also give better (lower MSE) prediction error?

The theoretically better way to deal with this is to create year dummy variables, perhaps in 5 year intervals. For clarity, you could name these Y84, Y89, Y94, etc. (If 1984 is the first year in the data set.) Then create interactions between the most important predictors, and these dummy variables. This will allow, for example, the effect of gasoline type to be different in 1994 than it is in 2004 (different slope of the partial derivative).

What variables would you like to have in your analysis, that are not in the data set?

Part E: How I edited the data set.

I have done my own version of sections B and D. Please see memo titled "Importing and filtering data, using EPA mileage as an example" . Run through these steps, and you should end up with something very close to the revised EPA data set. Several versions of the data are available as Excel files on https://bda2020.wordpress.com/data-sets/

Introductory reading on linear models for Linear and logistic regression.

You are already familiar with linear regression (such as Ordinary Least Squares OLS), which assumes that the result is the weighted sum of the explanatory variables, and fitting the model means figuring out the best weights (beta coefficients). In it, the output is a continuous variable. It then leads to *logistic regression* (where the output is a binary variable).

The key idea of "linear models" is that the final result is that 1) the final outcome is the *sum* of contributions from multiple causes, and 2) the contribution from each cause is *proportional* to the magnitude of the cause. This means, for example, that sharp changes in an effect cannot be modeled.

Introducing nonlinearities in linear models. Fortunately, there is a way around this restriction in many cases, although it is sometimes painful. You can *create new variables* to pick up nonlinear effects. One method is dummy variables; make sure you understand how they work. Another method is that you can create composite variables, such as:

```
Var35  = (Variable 3 ) x (Variable 5)     #Interaction effect
Var44  = (Variable 4)^2                   # quadratic effect
```

These new variables are then added to the right hand side of possible predictors. The reason this is painful is that you can create a *lot* of new variables. For example if you start with 15 variables and you want to include *all* two-variable interactions, that requires creating (15 * 14) = 210 new variables, for a total of 225 variables. So this method is practical mainly when you have domain knowledge that suggests certain interactions may exist.

Measuring the results of models: Although you are familiar with linear regression from statistics, our view of "what is a good fit" is substantially different in data mining than in conventional econometrics. The main differences are:

o We don't bother to measure the errors in the residuals from the data we used for fitting, because they can be dramatically too optimistic (e.g. with over-fitting). Instead, we look at the errors in the *validation sample*, the data that was *not* used to fit the model.

o Root mean squared error, the most common error measure for continuous outcomes, is only

one way to measure errors. It heavily penalizes large errors, compared with small errors. Other good measures are mean absolute error, and mean absolute percent error. See Chapter 5 in the main textbook DMBA-R.
o We don't care (much) about the t-statistic for individual coefficients of the model. t statistics only answer one question, and it's usually not an important question in data mining. Mainly, we care about the quality of the *overall* forecast. Choosing which variables to include in the analysis is a long topic (we will discuss in the future), but we may well choose to include some variables that are "not statistically significant" by the usual t measures, because they improve the overall prediction, as confirmed by validation. We do sometimes care about the estimated Beta coefficients for individual causal variable.

Be sure you are familiar with using dummy variables as explanatory variables. In R these are usually referred to as categorical variables (since they show which category an observation fits into) or *factors.*

On logistic classification models, ie. predicting a binary outcome. This is called "linear classification modeling" in data mining, and "logistic regression" in most of the statistical world, It's an extension of the OLS-style regression model to handle discrete outcomes.
Logistic regression is covered in Chapter 10 of DMBA+R, and Chapter 4.3 of ISLR book. If you have never encountered it before, read these especially carefully. The core concept is that we map from a specially constructed *continuous* variable, to the binary outcome we really care about. Then we try to predict the continuous outcome, using a linear model.

On linear OLS-related regression. (Predicting a *continuous* outcome.)
I don't like the way the main textbook handles linear regression, so I'm not passing out that chapter. I will cover it in lectures, later in the course.
Readings will be assigned later. But you can look at Gareth James et al, ISLR *An Introduction to Statistical Learning with Applications in R."* (Supplementary textbook.) It's available from Springerlink, at https://www.springer.com/gp/products/springerlink. Review section 3.2, and read section 3.3 on variations on the basic linear model.

You can use Rattle to do linear regression with discrete or continuous outcomes, although the DMRR book does not explain it in detail. Just click the appropriate **linear** buttons in the "model" tab.

Version April 12, 2017

MEMORANDUM BohnDA-gram
To: Students in Big Data Analytics
Subject: Importing and filtering data, using EPA mileage as an example
From: Prof. Roger Bohn
Date: April 17, 2016 updated april 23, 2018

The EPA data on auto mileage was provided in a spreadsheet, that you can convert into a CSV file. The original file, and the converted one from this memo, are accessible at https://bda2020.wordpress.com/data-sets/

You know how to clean up data like this using Excel and using Rattle. This memo shows how to do it using raw R.
The cleaned data, with some variables that I added, is now in a file called *EPA*

mileage 1984-16 cleanRB.csv You are welcome to use it.

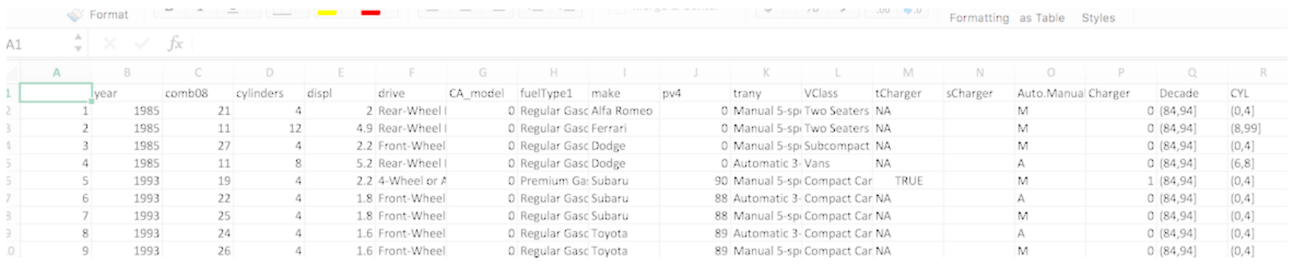
The main changes I made were:

Cutting out many columns

Adding some categorical variables, such as a year category and a cylinders category. Also a “Charger” variable which has 3 levels and includes both supercharger and turbocharger.

Removing all the electric cars, which were messing up some of the results. Only 3 fuel types remain: Diesel, Premium, and Regular (gasoline).

After saving the file as .CSV, I read it into Excel to check it. This is what it looks like.



	year	comb08	cylinders	displ	drive	CA_model	fuelType1	make	pv4	trany	VClass	tCharger	sCharger	Auto.Manual	Charger	Decade	CYL
1	1985	21	4	2	Rear-Wheel I	0	Regular Gasc	Alfa Romeo	0	Manual 5-sp	Two Seaters	NA		M	0	(84,94]	(0,4]
2	1985	11	12	4.9	Rear-Wheel I	0	Regular Gasc	Ferrari	0	Manual 5-sp	Two Seaters	NA		M	0	(84,94]	(8,99]
3	1985	27	4	2.2	Front-Wheel	0	Regular Gasc	Dodge	0	Manual 5-sp	Subcompact	NA		M	0	(84,94]	(0,4]
4	1985	11	8	5.2	Rear-Wheel I	0	Regular Gasc	Dodge	0	Automatic 3-	Vans	NA		A	0	(84,94]	(6,8]
5	1993	19	4	2.2	4-Wheel or A	0	Premium Ga:	Subaru	90	Manual 5-sp	Compact Car	TRUE		M	1	(84,94]	(0,4]
6	1993	22	4	1.8	Front-Wheel	0	Regular Gasc	Subaru	88	Automatic 3-	Compact Car	NA		A	0	(84,94]	(0,4]
7	1993	25	4	1.8	Front-Wheel	0	Regular Gasc	Subaru	88	Manual 5-sp	Compact Car	NA		M	0	(84,94]	(0,4]
8	1993	24	4	1.6	Front-Wheel	0	Regular Gasc	Toyota	89	Automatic 3-	Compact Car	NA		A	0	(84,94]	(0,4]
9	1993	26	4	1.6	Front-Wheel	0	Regular Gasc	Toyota	89	Manual 5-sp	Compact Car	NA		M	0	(84,94]	(0,4]

R code used to edit the EPA file.

I have added row numbers to make it easier to read. Obviously, undo them before pasting into R.

1. setwd("~/Documents/REB docs '12/Teaching/Big data IRGN 452/R data dir/EPA fuel efficiency 2015") # The directory will differ for you. Use wherever you store your data files for the BDA course
2. EPA.DATA.vehicles.86.to.15.shrunk <- read.csv("~/Documents/REB docs '12/Teaching/Big data IRGN 452/R data dir/EPA fuel efficiency 2015/EPA DATA vehicles 86 to 15 shrunk.csv") **#done automatically using the Tools/Import Dataset menu in Rstudio**. File name will change slightly
3. View(EPA.DATA.vehicles.86.to.15.shrunk)
4. Df <- EPA.DATA.vehicles.86.to.15.shrunk str(Df)
5. table(Df$fuelType1) **#simple one-way table of occurrences**
6.
7.
8. OK.fuel <- Df$fuelType1 == "Diesel"
9. sum(OK.fuel) **#counts number of occurrences, to check that OK.fuel is what I want.**
10. OK.fuelP <- Df$fuelType1 == "Premium Gasoline"
11. sum(OK.fuelP)
12. OK.fuelR <- Df$fuelType1 == "Regular Gasoline"
13. sum(OK.fuelR)

14. OK.fuel <- OK.fuel | OK.fuelP |OK.fuelR   **#combine all 3 fuel types, and only those fuel types**
15. sum(OK.fuel)
16. dim (Df2 <- Df[OK.fuel,]) **# This line is poor coding. It creates Df2, and reads its size, in the same line.**
17. summary(Df2$year)
18. Breaks <- c(1994, 2004, 2014, 2017).
19. Yr.cat <- cut(Df2$year, Breaks)   # cut is a clever function which converts numbers into factors, Use Help in Rstudio to see details.
20. head(Yr.cat)
21. Yr.cat <- cut(Df2$year-1900, Breaks)
22. head(Yr.cat)
23. Breaks <- c(1984,Breaks)
24. Yr.cat <- cut((Df2$year-1900), Breaks-1900)
25. Df2$Decade <- Yr.cat
26. Df3 <-Df2 **# Now let's get rid of some columns**
27. Df3 <- subset(Df3, select = -id) **#This is a way to get rid of columns by name, not number. Note the minus sign!**
28. Df3 <- subset(DF3, select = c(-city08,-co2TailpipeGpm, -fuelCost08))
29. Df3 <- subset(Df3, select = c(-city08,-co2TailpipeGpm, -fuelCost08))
30. Df3 <- subset(Df3, select = c(-highway08, -id.1, -year.1, -youSaveSpend))
31.
32. Df3 <- subset(Df3, select = c(-engId, -eng_dscr, -model))
33. sum(Df3$Auto.Manual == "") # what is mysterious third transmission? Only 2 cars have it.
34. CYL <- cut(Df3$cylinders, c(0,4,6,8,99)) **#categorical variable for number of cylinders: 4, 5+6, 7+8, and 9 or higher**
35. Df3$CYL <- CYL
36. write.csv(Df3,file="EPA clean") **#writes results to a file, in your working directory.**
37. summary(Df3) **#this is always a good idea when loading any data.**
38. str(Df3) **#so is this. It's more concise, and shows actual values better.**
39. head(Df3) **# this shows the first 6 rows**

R tip: Relabeling and re-ordering categorical variables.

R dataframes are designed to work well with categorical data. You can change the names to make them clearer. For example instead of 1/2, use M/F or Male/Female. Instead of age groups 1, 2, 3…, use "0-5", "6-10", etc.

R refers to such categories as *factors*. In fact much of the time R works with categorical data (and dummy variables), the data is actually stored as factors. Factors can be either ordered (ordinal data), or unordered.

> In R, each possible value of a categorical variable is called a *level*. A vector of levels is called a *factor*. Factors fit very cleanly into the vector orientation of R, and they are used in powerful ways for processing data and building statistical models. …

> In many situations it is not necessary to call *factor* explicitly. When an R function requires a factor, it usually converts your data to a factor automatically. The *table* function, for instance, works only on factors, so it routinely converts its inputs to factors without asking. You must explicitly create a factor variable when you want to specify the full set of levels or when you want to control the ordering of levels. [From:Section 5.4 of *The R Cookbook*, by Paul Teeter. This excerpt available on Google books.]

So, learning how to set up factors can save a lot of time for plotting and other purposes. A good tutorial introduction is http://www.r-bloggers.com/data-types-part-3-factors/

MEMORANDUM - R language

To: Students in Big Data Analytics

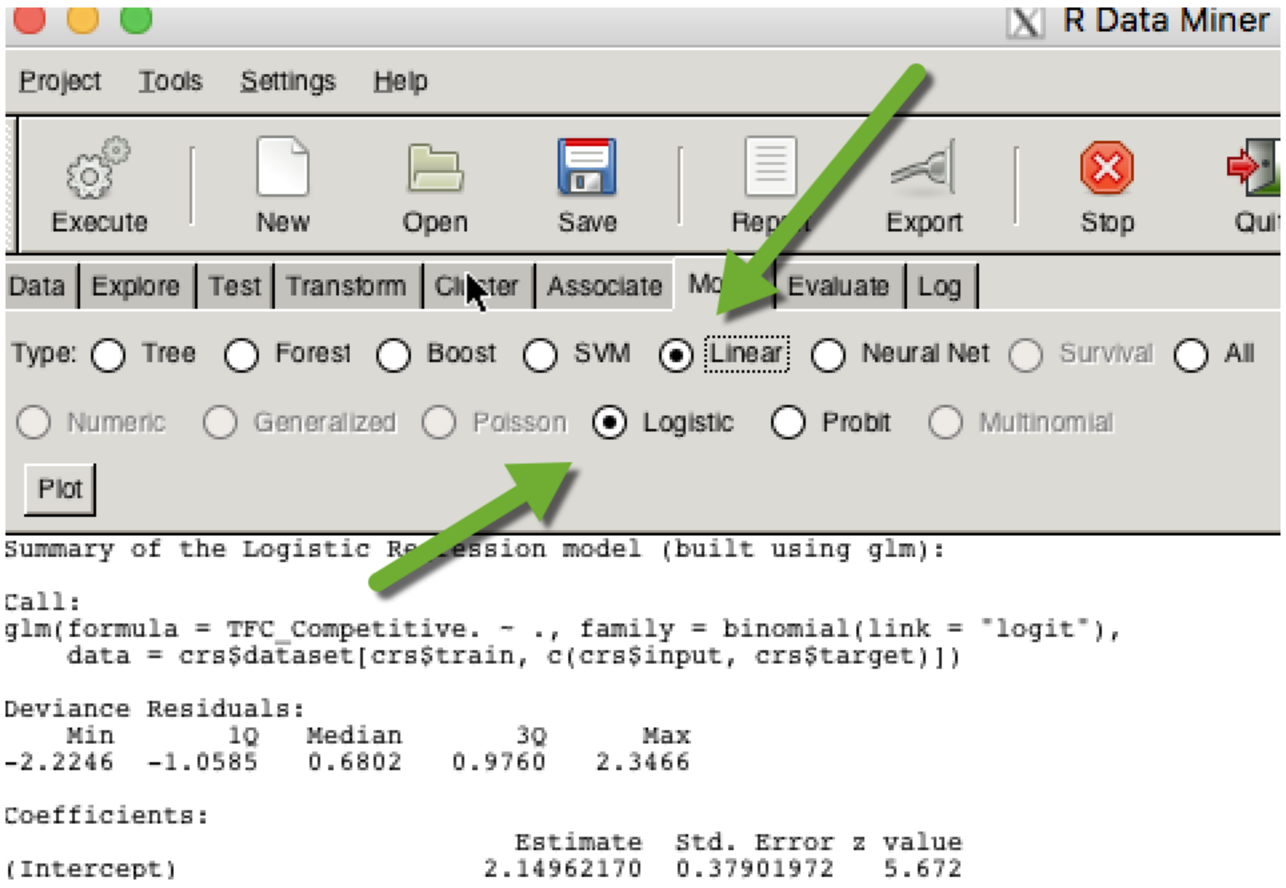
Subject: Linear models and variable transformations in Rattle

From: Prof. Roger Bohn   Rbohn@ucsd.edu
Date: April 16, 2018. Draft 0.1

Rattle has chapters devoted to different kinds of models including CART, Random Forest, and SVM (state vector machines). There is no chapter specifically about running linear models, however.

Logistic regression (Linear classification) in Rattle

Fortunately it is easy. On the MODEL tab, click the button for "Linear." If the Target variable is binary, as it is for eBay, then the buttons on the next row will let you choose two methods of classification. Choose logistic, as shown by the green arrows in the diagram.


R Data Miner
Project Tools Settings Help
Execute New Open Save Report Export Stop Quit
Data Explore Test Transform Cluster Associate Model Evaluate Log
Plot

Type: ( ) Tree ( ) Forest ( ) Boost ( ) SVM [✓ Linear] ( ) Neural Net ( ) Survival ( ) All

( ) Numeric ( ) Generalized ( ) Poisson [✓ Logistic] ( ) Probit ( ) Multinomial

```
Summary of the Logistic Regression model (built using glm):

Call:
glm(formula = TFC_Competitive. ~ ., family = binomial(link = "logit"),
    data = crs$dataset[crs$train, c(crs$input, crs$target)])

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.2246  -1.0585   0.6802   0.9760   2.3466

Coefficients:
                              Estimate  Std. Error z value
(Intercept)                 2.14962170  0.37901972   5.672
```

By the way, whenever you run a model in Rattle, the output report includes the actual R code that created your results. For linear classification, this was:

```
glm(formula = TFC_Competitive. ~ ., family = binomial(link =
"logit"),
    data = crs$dataset[crs$train, c(crs$input, crs$target)])
```

This will be mysterious right now, but it's time to start looking at pieces of R code.

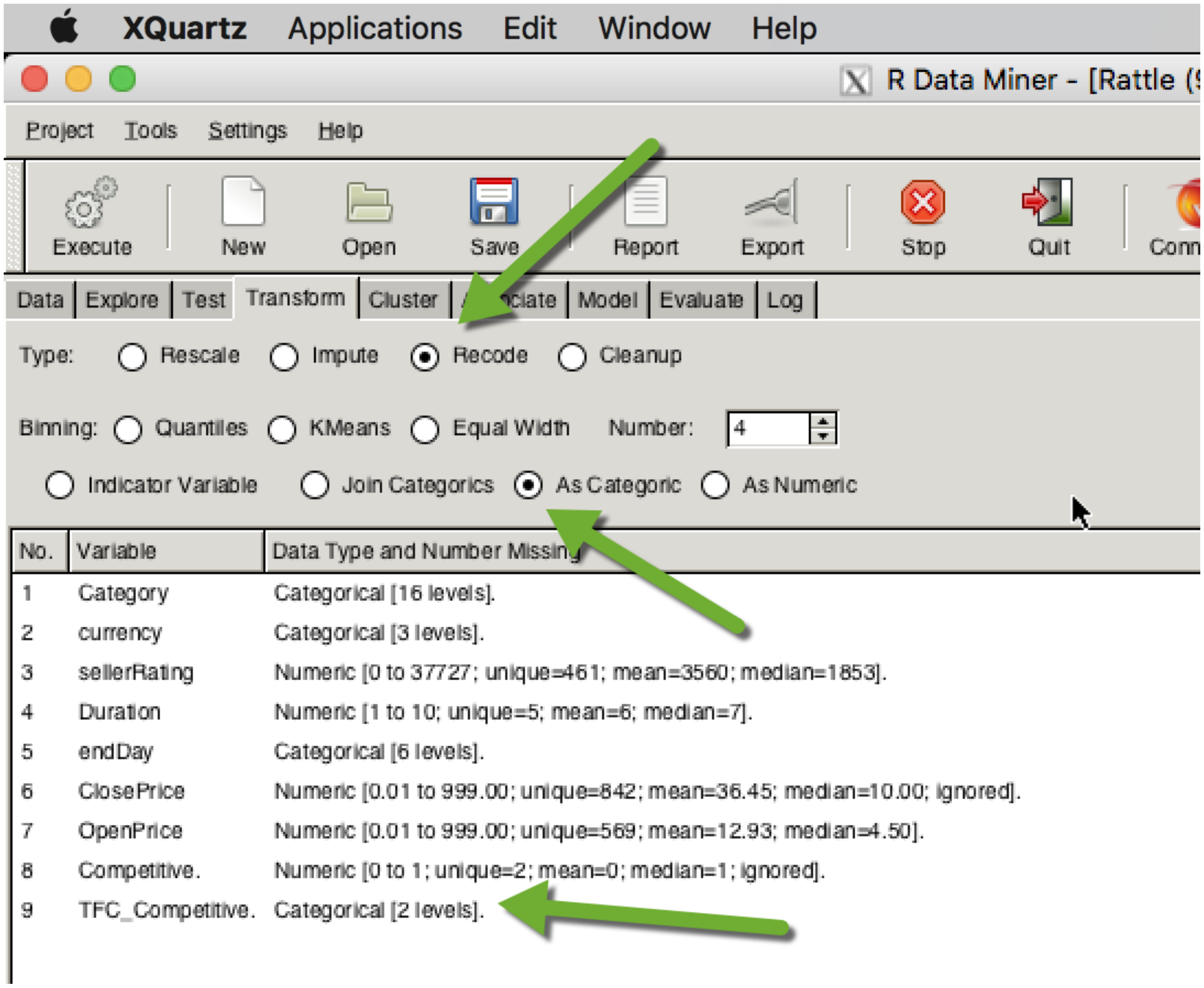
- 'glm' is an R function that does linear models.
- Family = binomial specifies that the output being predicted is 0/1.
- crs$dataset is what Rattle names its data matrix, after it has been imported into R. So
  **crs$dataset**[crs$train,] selects all the rows in the data frame that are in the crs$train list of random row numbers.
- A complete list of the R code that Rattle ran is in the LOG tab. After the model runs, you can go into RStudio (or R) and look at each of the variables that was created.

As always after you run the model, go to the EVALUATE tab and select the types of reports you want to see. Again, see the appropriate chapter in the Rattle textbook. (Chapter 15)

If your target variable was continuous, then your choices of types of models will change to appropriate methods of linear prediction.

Variable transformations in Rattle

You can do some variable transformation in Rattle. Please see Chapter 7 in the Rattle textbook for details. The figure below show how to transform the Competitive variable from a numeric to a categorical. You use the Recode button. When you hit *Execute*, it creates the new variable, with a modified version of the original name. (3rd green arrow.) If you go back to the DATA tab, you will see that the original variable has now been marked “Ignore.”



Type: ○ Rescale ○ Impute [● Recode] ○ Cleanup

Binning: ○ Quantiles ○ KMeans ○ Equal Width Number: 4

○ Indicator Variable ○ Join Categorics [● As Categoric] ○ As Numeric

No.	Variable	Data Type and Number Missing
1	Category	Categorical [16 levels].
2	currency	Categorical [3 levels].
3	sellerRating	Numeric [0 to 37727; unique=461; mean=3560; median=1853].
4	Duration	Numeric [1 to 10; unique=5; mean=6; median=7].
5	endDay	Categorical [6 levels].
6	ClosePrice	Numeric [0.01 to 999.00; unique=842; mean=36.45; median=10.00; ignored].
7	OpenPrice	Numeric [0.01 to 999.00; unique=569; mean=12.93; median=4.50].
8	Competitive.	Numeric [0 to 1; unique=2; mean=0; median=1; ignored].
9	TFC_Competitive.	Categorical [2 levels].

There are various other transformation you can do, such a taking the log of a numeric variable. Again, see the textbook chapter for details. The homework asks you to combine some of the eBay Categories.

Rattle does not allow you to create variables that are functions of two or more original variables. For example, you cannot create a new variable that is the ratio of two original variables. For that, you can use Excel (for now), or R/RStudio in a few weeks.