MATH 3338: Mathematical Modeling (Fall 2019)

Instructor: Dr. Hoa Nguyen
Project Advisor: Courtney Rohde, ACAS

Project Title
Using GLMs to Estimate Auto Insurance Losses by Policy Characteristics

Phase II Information

Task 1: Simplify the GLM *(3-5 Hours)*

In Phase I, we ran the first GLM. This was a very basic model. We just let R estimate the parameters that produced the best fit on the training data set. However, this is not necessarily the best model.

There may be variables that don't add much predictive power to the model and should be removed. There may be categorical variables that have levels which are not significantly different from each other, and should be grouped. There may be continuous variables that should have a curve fit through them, instead of letting R estimate the parameters for each level separately.

These types of changes are called simplifications. It is important to note that making changes like this will make the model perform "worse" on the training data set, in the sense that the predicted values will be farther away from the observed values in general. This is because we are putting restrictions on how R is allowed to make the parameter estimates. We are forcing some to be the same, in the case of a grouping, or we are forcing a linear relationship among levels in the case of fitting a curve. Even by removing an insignificant variable we will reduce model performance, since even an insignificant variable will explain some of the deviation from the observed values by pure chance. We know that such a variable is actually picking up random noise, not true explanatory signal, and this is why we must simplify the model.

Simplifying the model reduces model performance on the training data set, but it will improve model performance on data the model has never seen before. This is because simplifying the model helps it better distinguish the signal from the noise. If the model is over-fit, and is explaining the noise in the training data set, then it will perform poorly on the testing data set. The noise in the testing data is different, and won't follow the same (random) patterns that the model saw in the training data. If the model is mostly explaining real signal, it will still make good predictions on the testing data, which comes from the same distribution as the training data.

We want the model to perform very well on unknown data, since the model will be used to rate future policies before we know anything about the losses they will experience. We need to charge those policies accurate premiums to avoid adverse selection.
I want you all to explore some different ways to simplify your model. You may independently come to the same simplification decisions on some variables, which is ok. This is to be expected, since there are so few variables (each with relatively few levels) in our data. In fact, if you have similarly simplified models, it may make it even easier to propose new rating factors as a group, since the indicated factors from the models will be similar.

Note: You can ignore the linear model for all of Phase II. We may come back to it in Phase III when we are evaluating model performance. If you want, you can comment out the 'lm' parts of the code by adding #'s to the front of those lines. Then they will not even run.

Option 1: Remove a variable that is not predictive

When you use the summary() command to view the results of a model you just ran, you can use the statistics in the columns to the right of the parameter estimates to determine whether each parameter is adding value to your model. Generally, the larger the standard errors are, the less confidence the model has in the parameter estimate. You can divide the standard error by the parameter estimate (not the exponentiated factor) to get the "standard error percent." I am cautious of standard error % values greater than 0.5.

The next column contains the t-statistic. This number is not so easy to interpret on its own, but the idea is that more extreme values of the t-statistic suggest that the parameter is more significant. R compares the t-statistic to the values of a t distribution, to get the probabilities in the right-most column. These probabilities are also called p-values, and they represent the probability that the parameter is no different than 0. You recall that 0 is the parameter assigned to the base level. so if you have a high p-value, R is saying that parameter is not significantly different than the base. So a low p-value is better. I like to use a 5% (0.05) threshold for the p-value.

For the Phase 1 model, almost all of the p-values are small. when R displays a p-value like "< 2e-16", it's using scientific notation to denote 0.00000000000000022, which is basically 0. Only the "Unknown" level of NCD has a high p-value. This suggests that the "Unknown" level should be grouped with the base.

For a variable with only one parameter being estimated, like Gender or Protected NCD, the p-value gives enough information to determine whether the variable is significant. For a variable with more levels, these p-values don't give us any information about whether the non-base levels are significantly different from each other. We need to be able to measure this in order to simplify the categorical variables. I'll come back to this in the next section.

Suppose in the meantime that you have a simplification you like, on any type of variable. You can do a chi-square test on that variable to determine whether the whole variable adds anything to the model. (This is a different question than whether the individual levels' parameter estimates are different from 0). To run a chi square test, fit the model INCLUDING the variable in question as "model1." Then fit the model WITHOUT the variable in question as "model2." Then use the command:

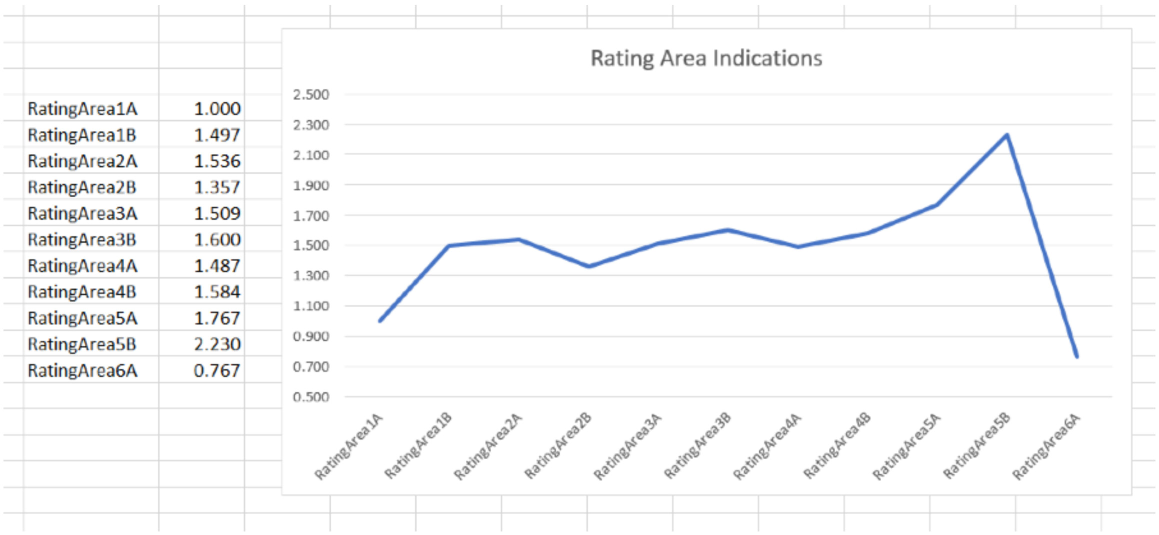
```
anova(model1,model2, test="Chisq").
```

"Anova" is short for "analysis of variance." The null hypothesis of this test is that model1 and model2 are the same, so a low chi squared percent (given in the last column as "Pr(>Chi)"), suggests that the models are different, and your variable IS adding value. I like to use a chi squared percent threshold of 5% when deciding whether to include a variable in my model.

Test all variables for predictiveness using the chi-squared test.

<u>Option 2: Further group a categorical variable</u>

How to group a categorical variable is a more difficult question to answer. A good place to start is to plot the indicated factors from your model on a line graph in Excel. Here I've plotted some estimates for the Rating Area factors:



RatingArea1A	1.000
RatingArea1B	1.497
RatingArea2A	1.536
RatingArea2B	1.357
RatingArea3A	1.509
RatingArea3B	1.600
RatingArea4A	1.487
RatingArea4B	1.584
RatingArea5A	1.767
RatingArea5B	2.230
RatingArea6A	0.767

By plotting the estimates, I observe that the 3A factor is very similar to the 4A factor, but they are both quite different from the base level (and their p-values from the model summary tell me they should not be grouped with the base level). I wonder if they should be grouped with each other. I need to test whether the parameter estimates are statistically different.

In statistics, this type of test is called a "Contrast." You will need to install another package in R: multcomp. And add the "library(multcomp)" command to the beginning of your R script, where you put the others. This library must load for us to run this test. Then after fitting the model you want to evaluate, you will run three lines with this format:

```
k <-matrix(c(0,0,0,0,0,1,0,-1,0,0,0,0,0,0,0,0,0,0,0,0),1)
t <-glht(model1, linfct = k)
summary(t)
```

You will only need to change the text in red, depending on what you are testing.

For this example, I am using the Phase I model except that I removed "as.factor" from VehicleAge, and I named it "model1" (you will put the name of your fitted model in the second line, as I have done). This is the model which produced the Rating Area estimates in my graph above. R estimates 20 parameters for this model (including the intercept). The vector defined in the first line has one component for each of the 20 parameters in the model, and they correspond to the order the parameters are listed in the summary screen. So if we want to compare the parameters for RatingArea3A and RatingArea4A (the 6th and 8th parameters listed by the summary() command, respectively), we put 0s in all other spaces, and we put a 1 and -1 in the 6th and 8th spaces, as I've done above.

Essentially, R is going to multiply this vector by the "matrix" formed by your model variables, and it will test whether (1 * parameter1) + [(-1) * parameter2] is any different than 0. If it is, we can conclude that the parameters are different. In the result printed by R, the value under "Pr(>|z|)" is the probability that the parameters are the same. So yet again, low probabilities are better, and 5% is an acceptable threshold for including the additional parameter. If the probability is above 5%, I recommend grouping together the levels you were testing.

For a model with a different number of estimated parameters, you will need to change the number of components of the vector in line 1 to match.

Visit this page for a more comprehensive (and technical) discussion of testing contrasts.

You can run this test for any pair of parameters estimated by the model. Not all pairs will make sense, however. You can only use these results if you are comparing parameters for different levels of the SAME VARIABLE. that is, you can compare two of the rating area levels as in the example, but you should not compare the NCD3 estimate to the GenderMale estimate. That doesn't really make sense.

Since the model doesn't make a parameter estimate for the base level, the contrast test can only compare non-base levels (which is exactly what we want, since we already have the p-value for comparing each level to the base).

If you have decided to group two levels, you will need to go into your data and create a grouped version of that variable's column. I recommend creating a new column - don't overwrite the original.

You can do this lots of ways, but one way might be to use a combination of the IF and OR functions in Excel. For example: IF(OR(B4="1",B4="2"),"1and2",B4). You can google the functions for more information. If you use functions to create a new column, you will have to copy that column and "paste values" over itself, since the .csv format can't store formulas.

Give the new column a new name. Column names and level names should be descriptive (not just "version1" "version2", etc), and they should not use spaces or special characters. Then include the name of the grouped column in your model formula in R, replacing the original column name. When run, the model will produce a single parameter estimate for the levels you grouped together. Remember to re-save your data as a .csv, not as an excel workbook, after you make changes.

I recommend grouping one pair at a time, iteratively fitting the model and testing to see if further grouping is necessary.

Note: There may be cases where you want to group levels that do not have similar parameter estimates. Consider your plot of the NCD factors. The "Unknown" estimate is the farthest from 1.0, but you recall that its p-value with the base level was quite high, suggesting they should be grouped together. Can you think of a reason why parameter estimates might be far apart, yet the model can't confidently say they are statistically different from each other?

Test as many pairs of levels for the categorical variables as you want, until you believe all of your parameter estimates are statistically significant on their own.

<u>Option 3: Fit a curve to a continuous variable</u>

This simplification may be the easiest to accomplish – just remove "as.factor" from Vehicle age. The single parameter estimate produced by the model is the **differential**, or the percent change between factors of adjacent levels, when exponentiated. That is:

Vehicle Age Factor = e^(β * Vehicle Age Value) = [e^(β)]^Vehicle Age Value

So vehicle age 1 gets the factor e^{β}, age 2 gets $(e^{\beta})^2$, and so forth, each successive level being multiplied by e^{β} again. This is essentially fitting an exponential curve (also called a **variate**) to the Vehicle Age variable.

The only other variable that could potentially be appropriately fit with a variate is NCD. R will not naturally do this however, because there are non-numerical level names in the data. If you want to test a variate on it, you'd have to assign those levels an x-value. Frequently we group Unknowns or error values with the base. 4+ could be tested to see if it's significantly different from 3 (perhaps using the methods described for categorical variables) and if not, they could be grouped. You could add a numerical version of the NCD column to your data, and include that in your model formula instead of the original. Then R would fit a curve to it.

<u>Final Comments on Simplification:</u>

I expect the simplification process to be the most time-consuming part of the project. Keep a log in Excel of the tests you performed and their results. This way you can keep track of what you've already tested. Record the types of tests you performed on which variables or combination of levels, the resulting statistics, and your modeling decisions. The format of your log is up to you.

The final deliverable for this task is your model log. Be sure to include your name, a description of the simplification you chose for each variable, the chi squared percent for the variable with that simplification, and the formula for your simplified model.

I also want you to include a graph of each variable's final simplification in your model log. For each variable, use one graph to plot the factors from the final simplification AND the factors from our Phase 1 model. Your simplification should reasonably track with the un-simplified estimates.

Task 2: Select Proposed Rating Factors *(30-60 Minutes)*

I am uploading a workbook called Rating.xlsx. The data in this workbook is similar to the data we used to build the model, except that the model data was a historical data set (from 2013 and 2014), and this data represents the policies we currently have on the books – just under 50,000 of them. These are the policies that will renew onto our new proposed base rate and factors.

I want you to use the 'Rels' tab of the Rating workbook to choose some tentative proposed factors. You will need to fill in your indicated factors from the simplified model you made in Task 1. You can start by (somewhat arbitrarily) selecting factors somewhere between the current factors I have given and the model indicated factors.

Before we settle on a final proposal for the new factors, we need to thoroughly understand how it will impact our current book of business. The first step in this analysis is to calculate new current and proposed premiums.

Similar to calculating fitted model values in Phase I, we calculate premiums by multiplying the appropriate relativities together for each record in the dataset. The main difference between calculating fitted values and premiums is that we do not use the intercept term from the model. Instead, we use the base rate, which represents the amount of premium an average driver would pay (i.e. average means that all relativities equal 1). For this project, the current base rate is **$500**. Recall that the formula for calculating an insurance premium is

Base Rate * Relativity_1 * Relativity_2 * … * Relativity_n

(We will continue to suppose that there are no expense fees.) Fill in the empty columns on the 'Business Data' tab of the Ratings workbook to calculate the current and proposed premiums for the active auto policies.

Since your indicated factors may be different (if you made different simplification decisions), it is ok if you propose different selected factors at this stage. In Phase III, I will ask you all to compare your modeling decisions and selection rationale, and come up with a joint proposal.

Task 3: Calculate the New Base Rate *(30-45 Minutes)*

We don't just need new factors, we also need a new base rate.

In Phase I, we built an indication for the insurance company that represented the overall rate change the company needs to implement in order to achieve its target profit. Due to the changes that we are making to the factors, the current total premium under the proposed rates is no longer the same as the total premium under the current rates. To correct for this, and to achieve the overall rate change indicated in Phase I, we must raise or lower the base rate. After changing the relativities, students must **off-balance** the rates to satisfy:

Total Proposed Premium = (1+rate change%) * Total Current Premium

Note that you can break down the premiums as follows:

$\sum_{\text{all policies}}$ (New Base Rate*New factors) = (1+rate change %) * $\sum_{\text{all policies}}$ (Old Base Rate*Old factors)

Solve this equation for "New Base Rate" to obtain the off-balance formula, and use it to calculate the proposed base rate.

Your proposed base rate can be entered to the right of the data on the 'Business Data' tab.

In Phase III we will further analyze the **dislocation** (difference between current and proposed premium at an individual policy level) produced by changing your rating algorithm. The **pivot table** on the 'Segments' tab will be used in Phase III. If you are unsatisfied with your dislocation analysis at that time (perhaps the new premiums are not competitive enough on key segments, which we will discuss later), you can make adjustments to your proposed factors and re-off-balance the base rate.

The deliverable for Tasks 2 and 3 is a filled-in version of the “Rating” workbook. The results will differ based on what your final indicated factors were and what factor selections you made, but the base rate must be properly off-balanced to achieve the indicated rate change given those factors.