Daavid Moseley
Final Examination – Machine Learning & Data Mining
Date: May 11th, 2015

An analysis of combined ratio models

Overview

In this final examination/project, we attempt to model the response variable **CR** (combined ratio) by utilizing a multitude of data mining techniques. The modeling of such data has proven to be quite strenuous, however, due to our lack of choice regarding predictor variables. In the past, models were designed with a wide range of predictor variables available to us. This liberality was extremely beneficial since several of our models gave very strong correlation matrices, good prediction errors, agreeable histograms and exhibited strong fits to our response variable overall. Our decision to choose any predictor variable was severely limited in this case and, as such, most of the models presented in this paper are less than satisfactory. Nonetheless, we discuss each model in depth and provide the rationale behind our modeling procedures.

Model 1.a

Before actually modeling the data, scaling and centering procedures were performed. Utilizing the knowledge gleaned from the midterm examination, in addition to the other examples in class, the decision was made to "clean up" the data accordingly. The **CR** column noticeably had zeroes in certain parts, so I made the decision to add one to each value of **CR** and then used the log function. This idea was used in the midterm, so I made the choice to use it here. After this, I scaled/centered the data and assessed the resulting histogram. Before the transformation, **CR**'s distribution did not look recognizable; after the transformation, however, **CR** actually resembled a statistical distribution. Confidently, the other variables were transformed accordingly to provide a potential, strong relationship with **CR**. Not only did we do this in the midterm examination, but doing so gave us better histograms to work with and the data was distributed nicely. The code contains comments regarding which histograms look normal, which don't, etc. For the sake of brevity, I do not include those comments in this section.

After appropriately transforming the variables, I combined the mostly unchanged data with the transformed variables and then made the decision to create indicator variables. The code for this change was taken directly from the midterm and applied to **State, New_Renewal, MGU** and **Underwriter**. Principle component analysis was utilized on the scaled/centered quantitative variables, as well. The PCA components, in addition to the indicator variables, were then combined with the original data set to provide a diversity. The final data frame, therefore, contained four different variables: Raw data, scaled data, PCA components and indicator variables. A correlation matrix was constructed to analyze if certain variables could be used as good predictors for SC_LOG_CR. The correlation matrix did not display any subset of "good" predictor variables, but **EE_Count** was recorded as one of the highest ones at approximately .119 correlation.

Stepwise Variable Selection / Linear Model – Model 1.a

From this point forth, modeling took the center stage of the project. Analyzing previous other examples discussed in class and the midterm, the choice was made to use a stepwise variable selection linear model with the scaled version of **CR** acting as the response variable and develop training/testing data. Therefore, I used the **lm** function to create a null.mod using the data from **D.train**. Afterward, a model was built, making sure to eliminate **Total_GWP**, **Total_Exp**, **Claims** and **CR** from the list of predictor variables. Associated versions – i.e., scaled versions – of those aforementioned variables were also eliminated as predictor variables. The R code comments on this.

The model quite swiftly and, unsurprisingly, gave a lackluster adjusted R^2 value of **0.0651** and a residual standard error of **.9759**. The code for the model can be found below:

```
set.seed(11291988)
train.index <- sample(1:nrow(D),round(0.6*nrow(D)),replace = FALSE); length(train.index)
D.train <- D[train.index,]; dim(D.train)
D.test <- D[-train.index,]; dim(D.test)
head(D.train)
null.mod <- lm(D.train$SC_LOG_CR ~ 1, data = D.train); ## Response=SC_LOG_CR
summary(null.mod)
head(D.train)
head(D_2)

##This is Model #1.##
##Very poor model.##

colnames(D.train)
colnames(D)
dim(D.train)
xnam <- colnames(D)[c(14:20, 22:30, 31:121)]
xnam
head(xnam)
full.fmla <- as.formula(paste("SC_LOG_CR ~", paste(xnam, collapse="+")))
full.fmla
mod1 <- step(null.mod, full.fmla, trace=1000, k=2)
summary(mod1)
## Our R-squared = 0.06051.  Our residual standard error = .9759.  This is a poor model.##
```

Unimpressed, the decision was made to utilize the BIC criterion as opposed to the AIC criterion. This made the model worse and was ultimately not included in the final draft of the code. The data frame **D** was amended several times to change the testing & training data, but the end result was still the same or very nearly similar. At one point, **D** only featured scaled variables, indicator variables and PCA components and the results were the same. The code contains references to these changes via variable names of **D_2**, **D3**, etc. These versions of D were not included in the final code, as they provided absolutely no benefit in improving the model(s). That being said, however, the skeletal system of those ideas remain within the R code to show evidence that other attempts were made.

The histogram of the prediction error was noticeably reminiscent of the scaled version of **CR** and the summary of the prediction error squared did not showcase the evidence of a good model. See below.

```
> summary((D.test$SC_LOG_CR-y_1.hat)^2)
       V1
 Min.   : 0.0000
 1st Qu.: 0.1446
 Median : 0.4660
 Mean   : 0.9211
 3rd Qu.: 0.9347
 Max.   :20.3892
```

Lasso Regression – Model 1.a

Moving onward, the choice was made to utilize either a Lasso or Ridge regression model. Using code from both the midterm & several R projects, the Lasso regression model was created. A 10-fold cross validation was used for lambda. The result was largely the same – the histogram of the prediction error mirrored that of the former model, and the summary gave very similar values. See below:

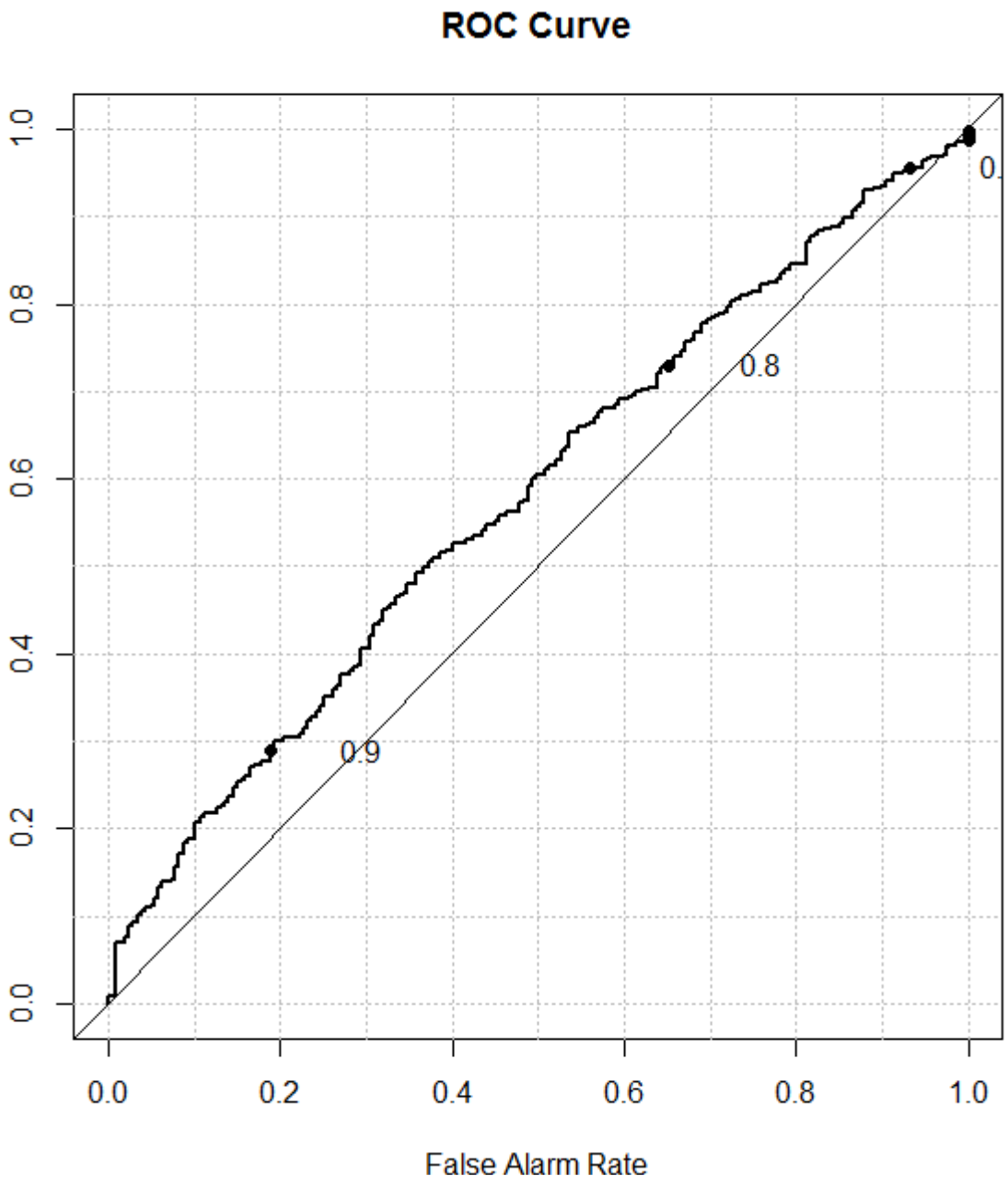
```
> summary((D.test$SC_LOG_CR - y_2.hat)^2)
       s0
 Min.   : 0.000002
 1st Qu.: 0.145822
 Median : 0.538512
 Mean   : 0.935443
 3rd Qu.: 0.884910
 Max.   :18.571247
```

GEV Modeling – Model 1.a

Analyzing the scaled version of CR once again, a GEV fit was incorporated to see if it was a good model. Surprisingly, GEV seemed to model the data almost perfectly. The probability plot, QQ plot and return level plot all gave confident results. The density plot seemed very spot on, and the code displayed a convergence to zero; hence, all signs points toward this being a great model. On inspecting the observed mean and estimated mean, however, I found a significant difference. The estimated mean of the GEV model gave us **0.06643**, whereas the observed mean is approximately **3.83688e-17**. In previous models, the difference was never this severe. In the GEV fit model in class, both the observed and estimated means were very much related to one another; a negligible difference existed. This is much more than negligible. Nonetheless, I've made the decision to include it as a model since the PP/QQ and return level plots are significant. Predictor variables were incorporated in GEV modeling, but this did nothing – the PP/QQ and return level plots did not change significantly, if at all. Unable to do anything else substantial with this GEV model and the others, attention was turned to Model 2.

Model 2

A logistic regression model was implemented, with the weights = **D.train$Total_GWP/100000** Other versions Total_GWP of Total_GWP were implemented, but they gave horrific results. This weight resulted in a mediocre ROC curve, as shown below.

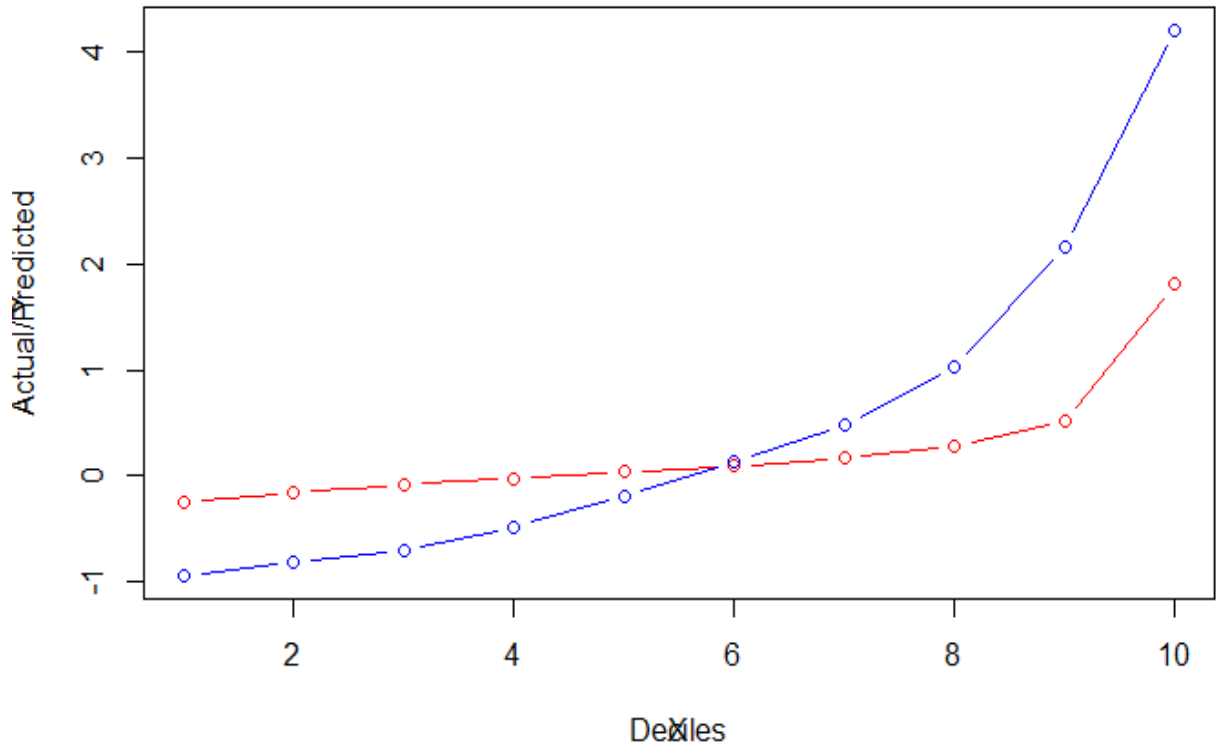


Another logistic regression model without weights was implemented for the sake of curiousity and the ROC curve was marginally higher, but this difference did not give any deep conceptual or theoretical insight since the change was only marginal.

Discussion/Conclusion

The quantitative predictor variables were mostly abysmal for the models in this project. In a separate R file, the author created a few models that used **Claims** as a predictor variable and the models improved drastically. **Total_GWP** was also used as a predictor and the models showcased, again, immense improvement. This makes sense, however, since both **Claims** and **Total_GWP** are used in the calculation for **CR**. The indicator variables used in this project were surprisingly helpful in modeling the data, but they didn't design a significantly better model. The lift chart was, by far, the hardest code to implement and the author is still unsure if the idea for the algorithm is correct. The methodology for the lift chart was to separate the data in deciles, find the mean of each decile and then implement the graphing algorithm suggested in the email. The chart that was produced looked similar to a lift chart viewed online, but many of the lift charts online are concerned with binomial predictive modeling. In the R file, I also provide a section for a Gains chart which is completely different from the other Gains charts online, but it is one of the first R packages associated with lift/gains charts.

LIFT CHART



Overall, the exam/project was difficult and very frustrating due to the data not being able to fit any model correctly. Without the certain predictive variables, the author learned that models can be extremely difficult to design. In addition, the lift chart proved to be a difficult task to code.