MAT 8414 – Categorical Data Analysis

Final Project Report

A CUMULATIVE LOGIT REGRESSION MODEL TO EXPLORE SEVERITY LEVELS OF CAR ACCIDENTS IN PENNSYLVANIA, 2019

Thach Nguyen

INTRODUCTION

Car accidents happen all year round and can cause massive damage to both individuals and the society in terms of health and economy. Not only does a car accident put people in a health jeopardy, being it physical or mental, but it also brings about economic and social tolls where involved parties may have to pay for vehicle repair, insurance claims, lawsuit settlements, or local authorities may incur additional expenses in mobilizing police forces, ambulances, etc. to resolve the situation, or general economic productivity loss due to traffic jams, re-routing, and suchlike impacts.

Because of those potential damages resulting from car accidents, local authorities and civil engineering researchers may be interested in exploring what factors are associated with car accident severity, so as to implement plans to control the levels, protect lives and enhance economies. This is the motivation for my project, where I will apply a cumulative logit regression model on car accident severity and explore parameter estimates, hopefully being able to find interesting associations.

DATASET DESCRIPTION

The dataset that I use originally has 49 variables, including my targeted response variable *Severity*. The other explanatory variables describe information surrounding specific accidents, such as detailed time and date, state, county, city, zip code, longitude/latitude coordinates, weather conditions, wind speed, temperature, etc. They also include a variety of indicator variables showing whether the accident is in close proximity to particular civil infrastructures like airport, train station, traffic signal, stop sign, pedestrian crossing, etc. If any link is found, it can be useful in future urban planning where city planners consider the proximity of streets and such infrastructure in minimizing car accident frequency as well as severity.

My response variable is *Severity*, which has 4 ordinal levels. Level 1 indicates no impact or minor impact on traffic, and level 4 indicates a major or serious impact on traffic, possibly causing several-hour traffic jams. *Severity* here only refers to the impact on traffic, as this is the main point of this dataset and this project. It does not have any implication on casualty or human harm.

One limitation of this dataset is it only records and shows information on the *surrounding settings* of accidents (weather, street, location, etc.). It will be much more useful if it also collects other *personal settings* of the accidents, such as car speed at the time of accident, how many passengers were on the car, car brands, car age, driver race/sex/age/health conditions, injury/casualty, etc. With more personal information, better conclusions can be inferred from models to explain accident severity.

Originally, I aimed at building a model from the whole dataset (almost 3 million car accidents during 2015-2019 in the contiguous US). Because of limited machine capacity and potentially long time to process such huge amount of data, I decided to filter out only accidents in Pennsylvania during 2019, thus narrowing my project down to data in PA, 2019 only.

ACKNOWLEDGEMENT

As per the request from the authors of the dataset, I would like to include here information regarding their initial research papers that engender the collection and compilation of the data:

Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. “A Countrywide Traffic Accident Dataset.”, arXiv preprint arXiv:1906.05409 (2019).

Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, Radu Teodorescu, and Rajiv Ramnath. “Accident Risk Prediction based on Heterogeneous Sparse Data: New Dataset and Insights.” In proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2019.

DATA CLEANING AND PREPARATION

I broke down the *Start_Time* into smaller time variables: year, month, day, weekday, hour period of the day. It is easier to filter data with these smaller variables, as well as easier to interpret later when we obtain parameter estimates.

For some variables, I used common sense or simple Google checks to remove absurd values (800 mph wind speed, 192 F temperature, -5 mile visibility, etc.). However, some other variables are beyond my knowledge (air pressure, humidity, wind chills) so I can only accept their values as is. There could be some miscoded values in the variables when the authors of the dataset pull the data from their secondary sources which also pull information from various other sources. Trying to verify the data is a big challenge.

In my model, I imputed the missing values with median for numerical variables and mode for categorical variables. Many research papers have pointed out that this approach may be quite inappropriate as it pulls the data distribution towards the median/mode and introduces some bias into the predicting model. Besides, missing values are not always at random, there can be specific reasons behind it, thus making pure imputation an un-informed and biased task. I tried to perform imputation by random forest approach, which will build a random forest model for each variable using all of the remaining ones as predictors and then predict the missing values based on other observed values. This approach is more scientific, but it takes too much time. I ran the code for more than 1 hour and still could not receive results, while median/mode imputation took only 5 seconds.

SUMMARY OF MODEL AND INTERPRETATION

I built a random forest model with 501 trees and 10 predictors for each split, then used the top 30 variables ranked by random forest algorithm based on mean decreased accuracy as potential predictors in my cumulative logit model. The random forest ranking shows that *Street_Type* is the most important variable in predicting level of car accident severity. Cumulative logit model with proportional odds provides some interesting results regarding *Street_Type*:

- The estimated odds of a car accident being in the lower severity level rather than higher severity when it happens on a highway is 0.151 times the estimated odds if it happens on a regular road, holding all other variables constant.
- The estimated odds of a car accident being in the lower severity level rather than higher severity when it happens on a highway is 0.183 times the estimated odds if it happens on a city street, holding all other variables constant.

Based on such results, we can predict that a car accident on a highway is estimated more likely to cause major traffic jams than one on a regular road/city street does, which makes sense intuitively as usually highway allows higher speed limit and larger vehicle transportation, so accidents there are often more serious and cause much more damage/traffic impact. However, an issue I have with this variable is that the type of street seems arbitrary at the choice of local authorities without much systematic process. What is considered “Street” in one city may be named “Road” in a rural town; large city streets are sometimes named “Boulevard” or “Avenue” but streets of same size, again, may be named just “Road” or “Street” in other places. Such arbitrary choice of names and street types can lead to misleading interpretation of model estimates.

The random forest model has accuracy of 74%, while the cumulative logit model with proportional odds achieves 81%, so at this point, the cumulative logit model is better in terms of prediction accuracy.

LIMITATIONS/CHALLENGES OF THE PROJECT

As explained earlier in the report, a limitation of my project was that I had to narrow down the scale of model to work on PA, 2019 data only. With a more powerful machine, the project can be extended to cover the entire contiguous US. This can also lead to further challenges in consolidating some categorical variables that have a large number of levels. For example, in the original dataset, variable *Zipcode* has 377,152 levels, *City* has 11,685 levels, *Weather_Condition* has 120 levels, etc. In my project, I use *combine.levels()* to combine low-frequency levels into a more general group. For a larger and more diverse dataset, this approach may be too conservative and may leave out peculiar information.

Another limitation is that I cannot perform a proportional odds test on my model to check whether such assumption is reasonable. After some improvement in data cleaning and variable selection, I was able to run a non-proportional odds model but R failed to provide log-likelihood or deviance information, so I cannot use likelihood ratio test to compare the 2 models. The error message in R when it failed to provide log-likelihood value was that the non-proportional odds model was not full-ranked and currently the *vglm()* function in *VGAM* package only supports full-ranked models. I tried to use *rrvglm()* function which specializes in reduced-ranked models, but it also failed. Full-ranked/Reduced-ranked models are beyond my knowledge so this issue remains unsolved for now.

Test of goodness of fit based on deviance would be inappropriate for my model as it has some continuous predictors, so the deviance does not have an approximate chi-squared distribution. R also failed to run the Hosmer-Lemeshow test, notifying that the test is only applicable to binary response.

FUTURE/FURTHER RESEARCH IDEAS

Running my model/project on a more powerful machine can resolve some of the challenges. It can also extend the project to cover the entire contiguous US, which then can possibly provide comparison between states, counties, etc.

More advanced techniques can be researched and utilized in my model to resolve the challenges. For example, I tried looking up methods to combine levels, such as weight-of-evidence binning, decision tree binning, etc., but most of them are built for binary response or simple linear regression models. There could be R packages that are specialized for ordinal response, which can be applied in this model to combine the levels with a more scientific method than the low-frequency one. If the processing machine is powerful enough, perhaps we do not need to consolidate the levels of categorical variables at all and just run the original whole dataset.

Perhaps there could be a way to perform a Hosmer-Lemeshow test on ordinal categorical response to test goodness-of-fit of model. I found that some researchers used Lipsitz test and Pulkstenis-Robinson test for this situation, but these methods are beyond my knowledge. If we could apply them to my model, perhaps we can further improve the model knowing whether or not it lacks fit. I took a look at the parameter estimates for all continuous variables in my model:

Start_Lat	**Start_Lng**	**Temperature**	**Wind_Chill**	**Humidity**
-2.1834e-01	1.2966e-01	-9.5416e-03	-5.9461e-06	-7.4938e-04
Pressure	**Visibility**	**Wind_Speed**	**Precipitation**	**Log_Distance**
-4.2152e-01	-3.4108e-02	-7.9831e-03	-7.2784e-01	-2.2646e-01

They are very close to 0 so I suspect these continuous variables are insignificant in the model. If we can run multiple likelihood ratio tests to check significance of them and it turns out none of them are statistically significant enough, then we can drop them and it would enable us to run goodness of fit test based on deviance as by then our model would only include categorical predictors.

We can also improve the analysis if the dataset is more particular in each observation, i.e. I think the provided variables are a little general. In the future, if the authors decide to include further detailed information such as driver profiles, area profiles, speed limit, speed at time of accident, street conditions, street facilities (median, tree, ditch, fence, painted strips, etc.) it could substantially help in model building and prediction tuning.

Other advanced models can be considered instead of just cumulative logit regression model with proportional odds. In my project, I ran a random forest model and found its accuracy slightly below that from a cumulative model. However, accuracy is not the only measure when it comes to model assessment/comparison. We need to look further into other measures such as precision, F-1 scores, ROC curves, etc. to decide which model is better suited to the dataset and goal of research. I could further improve my random forest model by tuning the number of trees, number of predictors at each split, upsampling/downsampling approach to tackle class imbalance. Machine learning or neural network models are also possible choices for predictive analytics.

Appendix: R codes

```
library(knitr)
library(dplyr)
library(ggplot2)
library(VGAM)
library(readr)
library(lubridate)
library(caTools)
library(imputeMissings)
library(leaps)
library(bestglm)
library(randomForest)
library(Hmisc)
library(caret)
library(mctest)
#Read in the original dataset
accident <- read_csv("C:/James Laptop/M.S. Applied Statistics/MAT 8414 Categorical Data Analysis - Dr. Bernhardt/Final Pr
oject/US_Accidents_Dec19.csv")
#Remove some variables that are not useful
accident2 <- accident %>% select(-c("End_Lat", "End_Lng", "Description", "ID", "Number", "Country", "Airport_Code", "Weat
her_Timestamp", "Source", "Timezone", "County", "City"))
#Break time variables into separate smaller time variables
accident2 <- accident2 %>%
  mutate(Year=year(Start_Time),
         Month=month(Start_Time, label=T, abbr=T),
         Day=day(Start_Time),
         Weekday=wday(Start_Time, label=T, abbr=T),
         Start_Time_Hour_th=hour(Start_Time)+1) %>%
  select(-Start_Time, -End_Time)
accident2$Month <- as.factor(accident2$Month)
accident2$Weekday <- as.factor(accident2$Weekday)
accident2$Day <- as.factor(accident2$Day)
accident2$Start_Time_Hour_th <- as.factor(accident2$Start_Time_Hour_th)
#Filter out only PA 2019 data to narrow down our analysis
accident2 %>% filter(Year==2019, State=="PA") %>%
  select(-Year, -State) -> accident2
#Reformat the Wind_Direction variable to avoid duplicated levels
accident2$Wind_Direction <- accident2$Wind_Direction %>%
  gsub(pattern="CALM", replacement="Calm", fixed=T) %>%
  gsub(pattern="East", replacement="E", fixed=T) %>%
  gsub(pattern="North", replacement="N", fixed=T) %>%
  gsub(pattern="South", replacement="S", fixed=T) %>%
  gsub(pattern="West", replacement="W", fixed=T) %>%
  gsub(pattern="VAR", replacement="Variable", fixed=T)
#Transform Street variable into Street_Type for better grouping and interpretation
accident2$Street <- ifelse(substr(accident2$Street, start=1, stop=2)=="I-", "Highway",
                           ifelse(substr(accident2$Street, start=nchar(accident2$Street)-2+1, stop=nchar(accident2$Street
))=="Rd", "Road",
                                  ifelse(substr(accident2$Street, start=nchar(accident2$Street)-3+1, stop=nchar(accident2
$Street))=="Ave", "Avenue",
                                         ifelse(substr(accident2$Street, start=nchar(accident2$Street)-2+1, stop=nchar(ac
cident2$Street))=="Ln", "Lane",
                                                ifelse(substr(accident2$Street, start=nchar(accident2$Street)-4+1, stop=n
char(accident2$Street))=="Blvd", "Boulevard",
                                                       ifelse(substr(accident2$Street, start=nchar(accident2$Street)-2+1,
 stop=nchar(accident2$Street))=="Rd", "Road",
                                                              ifelse(substr(accident2$Street, start=nchar(accident2$Stree
t)-2+1, stop=nchar(accident2$Street))=="Dr", "Lane",
                                                                     ifelse(substr(accident2$Street, start=nchar(accident
2$Street)-2+1, stop=nchar(accident2$Street))=="St", "Street", "Road"))))))))
accident2 %>% mutate(Street_Type=Street) %>% select(-Street) -> accident2
accident2$Street_Type <- as.factor(accident2$Street_Type)
#Change the name of variables to drop the unit in parentheses
accident2 %>% rename(
  Distance = `Distance(mi)`,
  Temperature = `Temperature(F)`,
  Wind_Chill = `Wind_Chill(F)`,
  Humidity = `Humidity(%)`,
  Pressure = `Pressure(in)`,
```

```r
  Visibility = `Visibility(mi)`,
  Wind_Speed = `Wind_Speed(mph)`,
  Precipitation = `Precipitation(in)`) -> accident2
#Set some variables into categorical format
accident2 %>% select(
  c(TMC, Zipcode, Severity,
    names(accident2[sapply(accident2, is.character)]))) -> reformat
reformat <- sapply(reformat, as.factor)
accident2 %>% select(
  -c(TMC, Zipcode, Severity,
     names(accident2[sapply(accident2, is.character)]))) -> noreformat
accident2 <- cbind(reformat, noreformat)
#Check amount of missing values
for (i in 1:length(names(accident2))){
  print(paste(names(accident2)[i], " ", sum(is.na(accident2[,i])), " ", round(sum(is.na(accident2[,i]))/20099*100, digits
=4), "%"))
}
#Check histogram of "Distance" variable
hist(accident2$Distance) #Looks skewed, may need log-transformation
quantile(accident2$Distance, probs=.99) #Unusual/unlikely value
#Cut off the unusually high values of "Distance" by using 99th percentile
accident2$Distance <- ifelse(
  accident2$Distance>quantile(accident2$Distance, probs=.99),
  quantile(accident2$Distance, probs=.99),
  accident2$Distance)
#Add some tiny values to the distance variable to avoid excessive amount of 0 values, which does not really make sense fo
r car accidents
#Log-transform distance to make it less skewed and improve the variable
#Then drop original distance variable
accident2 %>%
  mutate(Log_Distance = log(accident2$Distance+runif(length(accident2$Distance), min=0, max=0.01))) %>%
  select(-Distance) -> accident2
#Check histogram of "Distance" variable again
hist(accident2$Log_Distance)
#Check histogram of "Temperature" to detect skewness/unusual values
hist(accident2$Temperature)
#Check histogram of "Wind_Chill" to detect skewness/unusual values
hist(accident2$Wind_Chill)
#Check histogram of "Wind_Speed"
hist(accident2$Wind_Speed)
#Check histogram/quantiles of "Visibility"
hist(accident2$Visibility)
quantile(accident2$Visibility, na.rm=T)
#Cut off the unusual values of "Visibility" on both ends by using 99th and 1st percentiles
accident2$Visibility <- ifelse(
  accident2$Visibility>quantile(accident2$Visibility, na.rm=T, probs=.99),
  quantile(accident2$Visibility, na.rm=T, probs=.99),
  ifelse(
    accident2$Visibility<quantile(accident2$Visibility, na.rm=T, probs=.01),
    quantile(accident2$Visibility, na.rm=T, probs=.01),
    accident2$Visibility))
#Check histogram/quantiles of "Visibility" again
hist(accident2$Visibility)
quantile(accident2$Visibility, na.rm=T)
#Combine levels
accident2 %>% mutate(
  Zipcode.comb = combine.levels(accident2$Zipcode, minlev=.003),
  Weather.comb = combine.levels(accident2$Weather_Condition, minlev=.005)) %>%
  select(-Zipcode, -Weather_Condition) -> accident2
accident.imp <- accident2
num.var <- sapply(accident.imp, is.numeric)
#Set seed to split the dataset into training and validation sets
#Training 80% and validation 20%
#Split dataset based on response variable: Severity
set.seed(267)
split = sample.split(accident.imp$Severity, SplitRatio = 0.8)
#Imputation
impu <- imputeMissings::compute(accident.imp[split,], method="median/mode")
accident.imp <- imputeMissings::impute(accident.imp, object=impu)
#Random Forest model
accident.md.train <- as.data.frame(accident.imp[split,])
accident.md.valid <- as.data.frame(accident.imp[!split,])
RF <- randomForest(Severity~., data=accident.md.train,
                   ntree=501,
                   importance=TRUE,
                   mtry=10)
predictionRF <- predict(RF, newdata=accident.md.valid, type="response")
confusionMatrix(predictionRF, accident.md.valid$Severity)
#Check variable importance plot from Random Forest model
varImpPlot(RF)
#Pick out the top 30 variables based on MeanDecreaseAccuracy for cumulative logit model
logit.df <- as.data.frame(cbind(rownames(importance(RF)[,5:6]), importance(RF)[,5:6]), stringsAsFactors=F)
logit.df %>% rename(Variable=V1) %>%
  mutate(MeanDecreaseAccuracy=as.numeric(MeanDecreaseAccuracy)) %>%
  select(-MeanDecreaseGini) %>%
  arrange(desc(MeanDecreaseAccuracy)) -> logit.df
accident.cumu.train <- accident.md.train %>%
  select(Severity, head(logit.df,30)[,1])
accident.cumu.valid <- accident.md.valid %>%
  select(Severity, head(logit.df,30)[,1])
#Recode the response variable to ensure the model utilizes its ordinality
accident.cumu.train$Severity <- factor(accident.cumu.train$Severity, levels=c(1,2,3,4), ordered=T)
accident.cumu.valid$Severity <- factor(accident.cumu.valid$Severity, levels=c(1,2,3,4), ordered=T)
#Cumulative model with proportional odds
fitcumu <- vglm(Severity~.,
                family=cumulative(parallel=T), data=accident.cumu.train)
predcumu <- predict(fitcumu, newdata=accident.cumu.valid, type="response")
predcumu <- as.data.frame(predcumu)
predcumu.class <- base::apply(predcumu, 1, function(x) which.max(x))
predcumu.class <- factor(predcumu.class, levels=c(1,2,3,4), ordered=T)
confusionMatrix(predcumu.class, accident.cumu.valid$Severity)
#Cumulative model WITHOUT proportional odds
fitcumu.nopo <- vglm(Severity~.,
                     family=cumulative(parallel=F), data=accident.cumu.train)
predcumu.nopo <- predict(fitcumu.nopo, newdata=accident.cumu.valid, type="response")
predcumu.nopo <- as.data.frame(predcumu.nopo)
predcumu.classnopo <- base::apply(predcumu.nopo, 1, function(x) which.max(x))
predcumu.classnopo <- factor(predcumu.classnopo, levels=c(1,2,3,4), ordered=T)
confusionMatrix(predcumu.classnopo, accident.cumu.valid$Severity)
#Log-likelihood
logLik.vlm(fitcumu)
df.residual_vlm(fitcumu)
#Coefficients and estimated odds ratio
fitcumu.coef <- Coef(fitcumu)
head(sort(fitcumu.coef), 15)
tail(sort(fitcumu.coef), 15)
#Checking coefficients related to hours of a day
accident.imp$Start_Time_Hour_th <- factor(accident.imp$Start_Time_Hour_th, levels=c(1:24), ordered=T)
plot(accident.imp$Start_Time_Hour_th, xlab="Hour periods", ylab="Freq")
sort(fitcumu.coef[startsWith(names(fitcumu.coef),"Start_Time")])
exp(unname(fitcumu.coef["Start_Time_Hour_th9"]-fitcumu.coef["Start_Time_Hour_th21"]))
exp(unname(fitcumu.coef["Start_Time_Hour_th8"]-fitcumu.coef["Start_Time_Hour_th20"]))
#Checking coefficients related to street types
sort(fitcumu.coef[startsWith(names(fitcumu.coef),"Street_Type")])
exp(unname(fitcumu.coef["Street_TypeHighway"]-fitcumu.coef["Street_TypeRoad"]))
exp(unname(fitcumu.coef["Street_TypeHighway"]-fitcumu.coef["Street_TypeStreet"]))
#Some other estimated odds ratios
exp(fitcumu.coef["Temperature"])
exp(fitcumu.coef["Wind_Speed"])
exp(fitcumu.coef["Traffic_SignalTRUE"])
exp(fitcumu.coef["StopTRUE"])
exp(fitcumu.coef["CrossingTRUE"])
#Numeric variable coefficients
fitcumu.coef[names(accident.imp)[sapply(accident.imp, is.numeric)]]
```