Final Project: Auto Claim Fraud

Mary Donovan Martello

5/28/2020

## The following is the initial data set for this project:

# import data  
claims\_with\_Outlier = read.csv("FradulentInsuranceClaims.csv")  
str(claims\_with\_Outlier)

## 'data.frame': 1000 obs. of 40 variables:  
## $ months\_as\_customer : int 328 228 134 256 228 256 137 165 27 212 ...  
## $ age : int 48 42 29 41 44 39 34 37 33 42 ...  
## $ policy\_number : int 521585 342868 687698 227811 367455 104594 413978 429027 485665 636550 ...  
## $ policy\_bind\_date : Factor w/ 951 levels "1990-01-08","1990-01-27",..: 941 636 414 20 923 647 402 4 292 822 ...  
## $ policy\_state : Factor w/ 3 levels "IL","IN","OH": 3 2 3 1 1 3 2 1 1 1 ...  
## $ policy\_csl : Factor w/ 3 levels "100/300","250/500",..: 2 2 1 2 3 2 2 1 1 1 ...  
## $ policy\_deductable : int 1000 2000 2000 2000 1000 1000 1000 1000 500 500 ...  
## $ policy\_annual\_premium : num 1407 1197 1413 1416 1584 ...  
## $ umbrella\_limit : int 0 5000000 5000000 6000000 6000000 0 0 0 0 0 ...  
## $ insured\_zip : int 466132 468176 430632 608117 610706 478456 441716 603195 601734 600983 ...  
## $ insured\_sex : Factor w/ 2 levels "FEMALE","MALE": 2 2 1 1 2 1 2 2 1 2 ...  
## $ insured\_education\_level : Factor w/ 7 levels "Associate","College",..: 6 6 7 7 1 7 7 1 7 7 ...  
## $ insured\_occupation : Factor w/ 14 levels "adm-clerical",..: 3 7 12 2 12 13 10 13 8 9 ...  
## $ insured\_hobbies : Factor w/ 20 levels "base-jumping",..: 18 16 3 3 3 4 3 1 10 5 ...  
## $ insured\_relationship : Factor w/ 6 levels "husband","not-in-family",..: 1 3 4 5 5 5 1 5 4 6 ...  
## $ capital.gains : int 53300 0 35100 48900 66000 0 0 0 0 0 ...  
## $ capital.loss : int 0 0 0 -62400 -46000 0 -77000 0 0 -39300 ...  
## $ incident\_date : Factor w/ 60 levels "2015-01-01","2015-01-02",..: 25 21 53 10 48 2 13 58 30 5 ...  
## $ incident\_type : Factor w/ 4 levels "Multi-vehicle Collision",..: 3 4 1 3 4 1 1 1 3 3 ...  
## $ collision\_type : Factor w/ 4 levels "?","Front Collision",..: 4 1 3 2 1 3 2 2 2 3 ...  
## $ incident\_severity : Factor w/ 4 levels "Major Damage",..: 1 2 2 1 2 1 2 3 3 3 ...  
## $ authorities\_contacted : Factor w/ 5 levels "Ambulance","Fire",..: 5 5 5 5 3 2 5 5 5 4 ...  
## $ incident\_state : Factor w/ 7 levels "NC","NY","OH",..: 5 6 2 3 2 5 2 6 7 1 ...  
## $ incident\_city : Factor w/ 7 levels "Arlington","Columbus",..: 2 6 2 1 1 1 7 2 1 3 ...  
## $ incident\_location : Factor w/ 1000 levels "1012 5th Lane",..: 997 629 686 670 221 892 540 277 430 225 ...  
## $ incident\_hour\_of\_the\_day : int 5 8 7 5 20 19 0 23 21 14 ...  
## $ number\_of\_vehicles\_involved: int 1 1 3 1 1 3 3 3 1 1 ...  
## $ property\_damage : Factor w/ 3 levels "?","NO","YES": 3 1 2 1 2 2 1 1 2 2 ...  
## $ bodily\_injuries : int 1 0 2 1 0 0 0 2 1 2 ...  
## $ witnesses : int 2 0 3 2 1 2 0 2 1 1 ...  
## $ police\_report\_available : Factor w/ 3 levels "?","NO","YES": 3 1 2 2 2 2 1 3 3 1 ...  
## $ total\_claim\_amount : int 71610 5070 34650 63400 6500 64100 78650 51590 27700 42300 ...  
## $ injury\_claim : int 6510 780 7700 6340 1300 6410 21450 9380 2770 4700 ...  
## $ property\_claim : int 13020 780 3850 6340 650 6410 7150 9380 2770 4700 ...  
## $ vehicle\_claim : int 52080 3510 23100 50720 4550 51280 50050 32830 22160 32900 ...  
## $ auto\_make : Factor w/ 14 levels "Accura","Audi",..: 11 9 5 4 1 11 10 2 13 11 ...  
## $ auto\_model : Factor w/ 39 levels "3 Series","92x",..: 2 13 31 34 32 4 30 6 9 2 ...  
## $ auto\_year : int 2004 2007 2007 2014 2009 2003 2012 2015 2012 1996 ...  
## $ fraud\_reported : Factor w/ 2 levels "N","Y": 2 2 1 2 1 2 1 1 1 1 ...  
## $ X\_c39 : logi NA NA NA NA NA NA ...

## Data Cleaning

In my exploratory data analysis, I used simple histograms and counts to look for outliers. None of the histograms identified any outliers. However, the counts() for umbrella\_limit shows an obvious outlier of a negative 1 million:

count(claims\_with\_Outlier$umbrella\_limit)

## x freq  
## 1 -1000000 1  
## 2 0 798  
## 3 2000000 3  
## 4 3000000 12  
## 5 4000000 39  
## 6 5000000 46  
## 7 6000000 57  
## 8 7000000 29  
## 9 8000000 8  
## 10 9000000 5  
## 11 10000000 2

I filtered out the negative 1 million observation for the intial data set and named the dataframe object “claims”:

# Filter to remove the outliers  
claims <- claims\_with\_Outlier %>%   
 filter(umbrella\_limit >= 0)

My Final Report includes my discussion on data exploration and data cleaning. This is the code for the data cleaning:

# Remove empty variable/column  
claims <- Filter(function(x)!all(is.na(x)), claims)  
  
# separate policy\_csl variable into two variables and convert them to int data type  
claims <- separate(claims, "policy\_csl", into=c("cslBodily","cslProp"), sep="/")  
claims$cslBodily <- as.integer(claims$cslBodily)  
claims$cslProp <- as.integer(claims$cslProp)  
  
#convert two variables to date data type  
claims$incident\_date <- as.Date(claims$incident\_date)  
claims$policy\_bind\_date <- as.Date(claims$policy\_bind\_date)  
  
# add a new variable to hold calculation of the difference of weeks between to date variables  
claims$weeks\_bf\_incident<- difftime(claims$incident\_date,claims$policy\_bind\_date, units = c("weeks"))  
  
# remove two columns that produced the new weeks\_bf\_incident variable from the dataframe  
claims$incident\_date <- NULL  
claims$policy\_bind\_date <- NULL  
  
# convert months to weeks in another variable to be consistent with weeks\_bf\_incident variable  
claims$months\_as\_customer <- claims$months\_as\_customer \* 4.33  
  
# change name of variable to reflect that it is now measured in weeks  
names(claims)[names(claims) == "months\_as\_customer"] <- "weeks\_as\_customer"  
  
# three variables have "?" instead of NA; replace ? with one of the other factors  
claims$collision\_type <- gsub("?","Rear Collision",claims$collision\_type, fixed = TRUE)  
claims$property\_damage <- gsub("?","YES",claims$property\_damage, fixed = TRUE)  
claims$police\_report\_available <- gsub("?","NO",claims$police\_report\_available, fixed = TRUE)  
  
# convert the data type back to factors   
claims$collision\_type <- as.factor(claims$collision\_type)  
claims$property\_damage <- as.factor(claims$property\_damage)  
claims$police\_report\_available <- as.factor(claims$police\_report\_available)

## Categorical Variables

In order to make use of the several factored variables in the data set, I needed to convert the factor type variables to numeric type variables. However, for some of the analysis I still needed the categorical variables to be factor data types. Thus, instead of converting these categorical variables back and forth between factors and numeric I created a copy of the dataframe and converted the categorical variables to numeric in the copied dataframe:

# make copy of dataframe to use numeric categorical variables  
claimsNum <- claims  
  
# add variables for converted factor to numeric  
claimsNum$policy\_stateNum <- as.numeric(claims$policy\_state)  
claimsNum$insured\_sexNum <- as.numeric(claims$insured\_sex)  
claimsNum$insured\_education\_levelNum <- as.numeric(claims$insured\_education\_level)  
claimsNum$insured\_occupationNum <- as.numeric(claims$insured\_occupation)  
claimsNum$insured\_hobbiesNum <- as.numeric(claims$insured\_hobbies)  
claimsNum$insured\_relationshipNum <- as.numeric(claims$insured\_relationship)  
claimsNum$incident\_typeNum <- as.numeric(claims$incident\_type)  
claimsNum$collision\_typeNum <- as.numeric(claims$collision\_type)  
claimsNum$incident\_severityNum <- as.numeric(claims$incident\_severity)  
claimsNum$authorities\_contactedNum <- as.numeric(claims$authorities\_contacted)  
claimsNum$incident\_stateNum <- as.numeric(claims$incident\_state)  
claimsNum$incident\_cityNum <- as.numeric(claims$incident\_city)  
claimsNum$property\_damageNum <- as.numeric(claims$property\_damage)  
claimsNum$police\_report\_availableNum <- as.numeric(claims$police\_report\_available)  
claimsNum$auto\_makeNum <- as.numeric(claims$auto\_make)  
claimsNum$auto\_modelNum <- as.numeric(claims$auto\_model)  
claimsNum$fraud\_reportedNum <- as.numeric(claims$fraud\_reported)  
claimsNum$weeks\_bf\_incident <- as.numeric(claimsNum$weeks\_bf\_incident)

The following is the clean dataframe with converted categorical variables:

str(claimsNum)

## 'data.frame': 999 obs. of 56 variables:  
## $ weeks\_as\_customer : num 1420 987 580 1108 987 ...  
## $ age : int 48 42 29 41 44 39 34 37 33 42 ...  
## $ policy\_number : int 521585 342868 687698 227811 367455 104594 413978 429027 485665 636550 ...  
## $ policy\_state : Factor w/ 3 levels "IL","IN","OH": 3 2 3 1 1 3 2 1 1 1 ...  
## $ cslBodily : int 250 250 100 250 500 250 250 100 100 100 ...  
## $ cslProp : int 500 500 300 500 1000 500 500 300 300 300 ...  
## $ policy\_deductable : int 1000 2000 2000 2000 1000 1000 1000 1000 500 500 ...  
## $ policy\_annual\_premium : num 1407 1197 1413 1416 1584 ...  
## $ umbrella\_limit : int 0 5000000 5000000 6000000 6000000 0 0 0 0 0 ...  
## $ insured\_zip : int 466132 468176 430632 608117 610706 478456 441716 603195 601734 600983 ...  
## $ insured\_sex : Factor w/ 2 levels "FEMALE","MALE": 2 2 1 1 2 1 2 2 1 2 ...  
## $ insured\_education\_level : Factor w/ 7 levels "Associate","College",..: 6 6 7 7 1 7 7 1 7 7 ...  
## $ insured\_occupation : Factor w/ 14 levels "adm-clerical",..: 3 7 12 2 12 13 10 13 8 9 ...  
## $ insured\_hobbies : Factor w/ 20 levels "base-jumping",..: 18 16 3 3 3 4 3 1 10 5 ...  
## $ insured\_relationship : Factor w/ 6 levels "husband","not-in-family",..: 1 3 4 5 5 5 1 5 4 6 ...  
## $ capital.gains : int 53300 0 35100 48900 66000 0 0 0 0 0 ...  
## $ capital.loss : int 0 0 0 -62400 -46000 0 -77000 0 0 -39300 ...  
## $ incident\_type : Factor w/ 4 levels "Multi-vehicle Collision",..: 3 4 1 3 4 1 1 1 3 3 ...  
## $ collision\_type : Factor w/ 3 levels "Front Collision",..: 3 2 2 1 2 2 1 1 1 2 ...  
## $ incident\_severity : Factor w/ 4 levels "Major Damage",..: 1 2 2 1 2 1 2 3 3 3 ...  
## $ authorities\_contacted : Factor w/ 5 levels "Ambulance","Fire",..: 5 5 5 5 3 2 5 5 5 4 ...  
## $ incident\_state : Factor w/ 7 levels "NC","NY","OH",..: 5 6 2 3 2 5 2 6 7 1 ...  
## $ incident\_city : Factor w/ 7 levels "Arlington","Columbus",..: 2 6 2 1 1 1 7 2 1 3 ...  
## $ incident\_location : Factor w/ 1000 levels "1012 5th Lane",..: 997 629 686 670 221 892 540 277 430 225 ...  
## $ incident\_hour\_of\_the\_day : int 5 8 7 5 20 19 0 23 21 14 ...  
## $ number\_of\_vehicles\_involved: int 1 1 3 1 1 3 3 3 1 1 ...  
## $ property\_damage : Factor w/ 2 levels "NO","YES": 2 2 1 2 1 1 2 2 1 1 ...  
## $ bodily\_injuries : int 1 0 2 1 0 0 0 2 1 2 ...  
## $ witnesses : int 2 0 3 2 1 2 0 2 1 1 ...  
## $ police\_report\_available : Factor w/ 2 levels "NO","YES": 2 1 1 1 1 1 1 2 2 1 ...  
## $ total\_claim\_amount : int 71610 5070 34650 63400 6500 64100 78650 51590 27700 42300 ...  
## $ injury\_claim : int 6510 780 7700 6340 1300 6410 21450 9380 2770 4700 ...  
## $ property\_claim : int 13020 780 3850 6340 650 6410 7150 9380 2770 4700 ...  
## $ vehicle\_claim : int 52080 3510 23100 50720 4550 51280 50050 32830 22160 32900 ...  
## $ auto\_make : Factor w/ 14 levels "Accura","Audi",..: 11 9 5 4 1 11 10 2 13 11 ...  
## $ auto\_model : Factor w/ 39 levels "3 Series","92x",..: 2 13 31 34 32 4 30 6 9 2 ...  
## $ auto\_year : int 2004 2007 2007 2014 2009 2003 2012 2015 2012 1996 ...  
## $ fraud\_reported : Factor w/ 2 levels "N","Y": 2 2 1 2 1 2 1 1 1 1 ...  
## $ weeks\_bf\_incident : num 14.3 447.1 754.6 1285.1 36.6 ...  
## $ policy\_stateNum : num 3 2 3 1 1 3 2 1 1 1 ...  
## $ insured\_sexNum : num 2 2 1 1 2 1 2 2 1 2 ...  
## $ insured\_education\_levelNum : num 6 6 7 7 1 7 7 1 7 7 ...  
## $ insured\_occupationNum : num 3 7 12 2 12 13 10 13 8 9 ...  
## $ insured\_hobbiesNum : num 18 16 3 3 3 4 3 1 10 5 ...  
## $ insured\_relationshipNum : num 1 3 4 5 5 5 1 5 4 6 ...  
## $ incident\_typeNum : num 3 4 1 3 4 1 1 1 3 3 ...  
## $ collision\_typeNum : num 3 2 2 1 2 2 1 1 1 2 ...  
## $ incident\_severityNum : num 1 2 2 1 2 1 2 3 3 3 ...  
## $ authorities\_contactedNum : num 5 5 5 5 3 2 5 5 5 4 ...  
## $ incident\_stateNum : num 5 6 2 3 2 5 2 6 7 1 ...  
## $ incident\_cityNum : num 2 6 2 1 1 1 7 2 1 3 ...  
## $ property\_damageNum : num 2 2 1 2 1 1 2 2 1 1 ...  
## $ police\_report\_availableNum : num 2 1 1 1 1 1 1 2 2 1 ...  
## $ auto\_makeNum : num 11 9 5 4 1 11 10 2 13 11 ...  
## $ auto\_modelNum : num 2 13 31 34 32 4 30 6 9 2 ...  
## $ fraud\_reportedNum : num 2 2 1 2 1 2 1 1 1 1 ...

## Fraud Reported Subsets

I created subsets of both dataframes based on the fraud\_reported variable. The fraud\_reported variable is either Y or N to indicate if a fraud was reported on the auto insurance claim transaction. The subsets are useful for identifying differences in the data and thus identifying variables influencing fraud:

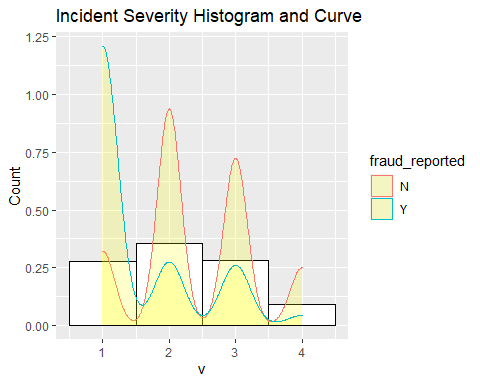
# Subsets   
noFraudSubsection <- subset(claims, fraud\_reported == "N")  
yesFraudSubsection <- subset(claims, fraud\_reported == "Y")  
noFraudSubsectionNum <- subset(claimsNum, fraud\_reported == "N")  
yesFraudSubsectionNum <- subset(claimsNum, fraud\_reported == "Y")

## Histograms to Identify Influencing Variables

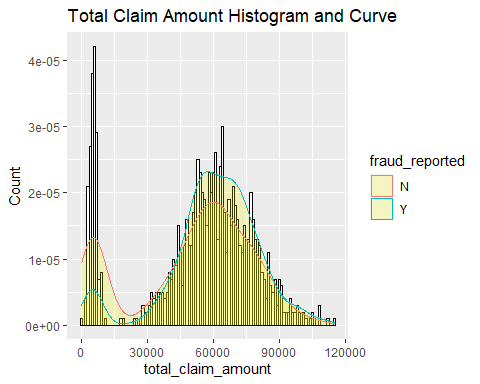
I used histograms with normal curves and colored by the fraud\_reported variable to look for variables that might be an indication of a fraudulent auto claim. Many of the histograms showed the same curve for the fraud\_reported variable, and thus did not appear to identify a difference for fraudulent claims. However, some of the histograms show a difference in the curve for fraud\_reported = Y versus fraud\_reported = N. These may indicate variables that can identify fraudulent claims.

The fraud\_reported == “N” subset has 752 observations but the fraud\_reported == “Y” subset has only 247 observations. If each subset had the same proportional number of values for a variable, then the histogram’s red curve (for N) would always be proportionally higher than the green curve (for Y). However, for some variables the green curve (for Y) was much higher than the red curve. The following histograms for incident\_severity, total\_claim\_amount and weeks\_bf\_incident demonstrate this discrepancy.

# Histogram of the incident\_severityNum variable with normal curve  
ggplot(claimsNum, aes(x=incident\_severityNum, col = fraud\_reported)) +   
 geom\_histogram(binwidth = 1, aes(y=..density..), colour="black", fill="white") +  
 geom\_density(alpha=.2, fill="yellow") +  
 labs(title="Incident Severity Histogram and Curve", x="v", y = "Count")

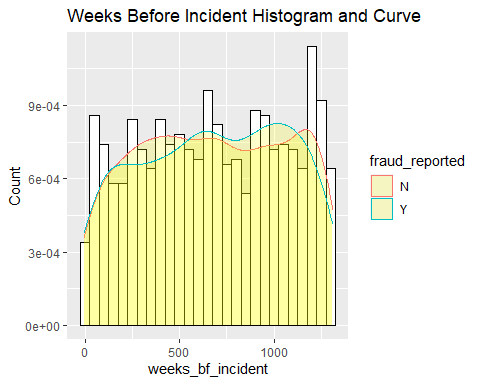


ggplot(claims, aes(x=total\_claim\_amount, col = fraud\_reported)) +   
 geom\_histogram(binwidth = 1000, aes(y=..density..), colour="black", fill="white") +  
 geom\_density(alpha=.2, fill="yellow") +  
 labs(title="Total Claim Amount Histogram and Curve", x="total\_claim\_amount", y = "Count")



ggplot(claims, aes(x=weeks\_bf\_incident, col = fraud\_reported)) +   
 geom\_histogram(binwidth = 50, aes(y=..density..), colour="black", fill="white") +  
 geom\_density(alpha=.2, fill="yellow") +  
 labs(title="Weeks Before Incident Histogram and Curve", x="weeks\_bf\_incident", y = "Count")

## Don't know how to automatically pick scale for object of type difftime. Defaulting to continuous.



## Correlation

My outcome variable is fraud\_reported. In order to determine which variables will explain this outcome variable, I ran correlation functions to try to discover some correlation between fraud\_reported and other variables. Fraud\_reported is a discrete dichotomous variable. Point-biserial correlation is used when one of the variables is a discrete dichotomous variable. The point-biserial correlation is equivalent to calculating the Pearson correlation between a continuous and a dichotomous variable. Therefore, I can use the standard cor.test function in R, which will output the correlation, a 95% confidence interval, and an independent t-test with associated p-value.

The p-values from the correlation test is the significance level of the t-test. If the p-value is less than 5% then we can conclude that the correlation is significant. The variables that have p-values that are 6% or less are total\_claim\_amount, incident\_severity, and umbrella\_limit.

cor.test(claimsNum$total\_claim\_amount, claimsNum$fraud\_reportedNum, method = "pearson")

##   
## Pearson's product-moment correlation  
##   
## data: claimsNum$total\_claim\_amount and claimsNum$fraud\_reportedNum  
## t = 5.2751, df = 997, p-value = 1.627e-07  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## 0.1038180 0.2245105  
## sample estimates:  
## cor   
## 0.164781

cor.test(claimsNum$umbrella\_limit, claimsNum$fraud\_reportedNum, method = "pearson")

##   
## Pearson's product-moment correlation  
##   
## data: claimsNum$umbrella\_limit and claimsNum$fraud\_reportedNum  
## t = 1.8386, df = 997, p-value = 0.06627  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.003907198 0.119723455  
## sample estimates:  
## cor   
## 0.05813101

cor.test(claimsNum$incident\_severityNum, claimsNum$fraud\_reportedNum, method = "pearson")

##   
## Pearson's product-moment correlation  
##   
## data: claimsNum$incident\_severityNum and claimsNum$fraud\_reportedNum  
## t = -14.073, df = 997, p-value < 2.2e-16  
## alternative hypothesis: true correlation is not equal to 0  
## 95 percent confidence interval:  
## -0.4575715 -0.3540160  
## sample estimates:  
## cor   
## -0.4071011

Given the results of the correlation tests, I ran partial correlation on the fraud\_reported and incident-severity variables, with total\_claim\_amount and umbrella\_limit as controlled variables. These correlations are stronger than any of the two-variable correlations.

claimsPC1 <- claimsNum[, c("fraud\_reportedNum", "total\_claim\_amount", "incident\_severityNum")]  
pc <- pcor(c("fraud\_reportedNum", "incident\_severityNum", "total\_claim\_amount"), var(claimsPC1))  
pc

## [1] -0.3778517

claimsPC2 <- claimsNum[, c("fraud\_reportedNum", "umbrella\_limit", "incident\_severityNum")]  
pc <- pcor(c("fraud\_reportedNum", "incident\_severityNum", "umbrella\_limit"), var(claimsPC2))  
pc

## [1] -0.4082894

claimsPC3 <- claimsNum[, c("fraud\_reportedNum", "umbrella\_limit", "total\_claim\_amount", "incident\_severityNum")]  
pc <- pcor(c("fraud\_reportedNum", "incident\_severityNum", "umbrella\_limit", "total\_claim\_amount"), var(claimsPC3))  
pc

## [1] -0.3782586

## Logistic Regression

My outcome variable is a binary categorical variable and therefore I am using logistic regression to try to determine explanatory variables to predict the fraud\_reported variable. Note: I am able to use the categorical variables in the regression model as I converted them to numeric data types.

I first included all of the variables in the formula with the exception of (1) the three variables with a “?” as a value because I have no means of deciding on meaningful replacement values and (2) the variables that caused multicollinearity. There was a multicollinearity problem with including all the variables. I tested for multicollinearity with the vif() function. When the vif() function shows a multicollinearity problem, you can see which variables are the cause of the problem with the alias() function. I removed the variables causing the multicollinearity problem and the three “?” variables, and ran a logistic regression model with the remaining variables (the “Permissible Variables Model”).

The accuracy result of the Permissible Variables Model is 80.58%:

claimsMR10 <- claimsNum[, c(1:3, 5:10, 16:17, 25:26, 28:29, 31:31, 34:34, 37:46, 48:51, 54:55)]  
logModel <- glm(fraud\_reported ~ ., data = claimsMR10, family = binomial(), maxit = 100)  
  
# make predictions  
predlogModel <- predict(logModel, type = "response")  
  
# make confusion matrix  
confMatrix <- table(Actual\_value = claimsNum$fraud\_reported,   
 Predicted\_Value = predlogModel > 0.50)  
confMatrix

## Predicted\_Value  
## Actual\_value FALSE TRUE  
## N 697 55  
## Y 139 108

# calculate accuracy  
(confMatrix[[1,1]] + confMatrix[[2,2]]) / sum(confMatrix) \* 100

## [1] 80.58058

z-statistics for Permissible Variables Model:

summary(logModel)

##   
## Call:  
## glm(formula = fraud\_reported ~ ., family = binomial(), data = claimsMR10,   
## maxit = 100)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.6515 -0.6892 -0.4288 -0.1291 2.9967   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) 2.609e+00 2.800e+01 0.093 0.9258   
## weeks\_as\_customer 2.540e-04 4.501e-04 0.564 0.5726   
## age -1.723e-02 2.416e-02 -0.713 0.4759   
## policy\_number -4.042e-07 3.301e-07 -1.224 0.2208   
## cslBodily 3.788e-03 4.988e-03 0.759 0.4476   
## cslProp -2.418e-03 2.808e-03 -0.861 0.3891   
## policy\_deductable 3.756e-05 1.378e-04 0.273 0.7852   
## policy\_annual\_premium -2.049e-04 3.432e-04 -0.597 0.5505   
## umbrella\_limit 7.138e-08 3.509e-08 2.034 0.0419 \*   
## insured\_zip 4.011e-07 1.171e-06 0.342 0.7321   
## capital.gains -1.693e-06 3.005e-06 -0.563 0.5731   
## capital.loss -3.109e-06 2.985e-06 -1.042 0.2976   
## incident\_hour\_of\_the\_day -8.649e-03 1.221e-02 -0.708 0.4788   
## number\_of\_vehicles\_involved -1.756e-02 2.134e-01 -0.082 0.9344   
## bodily\_injuries 7.674e-02 1.026e-01 0.748 0.4544   
## witnesses 1.064e-01 7.563e-02 1.406 0.1597   
## total\_claim\_amount -1.911e-05 1.680e-05 -1.137 0.2555   
## vehicle\_claim 3.745e-05 2.336e-05 1.604 0.1088   
## auto\_year -2.874e-04 1.391e-02 -0.021 0.9835   
## weeks\_bf\_incident -3.630e-05 2.188e-04 -0.166 0.8682   
## policy\_stateNum 9.433e-02 1.015e-01 0.929 0.3529   
## insured\_sexNum 1.111e-01 1.688e-01 0.658 0.5103   
## insured\_education\_levelNum 1.934e-02 4.302e-02 0.450 0.6530   
## insured\_occupationNum 4.271e-03 2.076e-02 0.206 0.8370   
## insured\_hobbiesNum -2.426e-02 1.493e-02 -1.624 0.1043   
## insured\_relationshipNum 2.943e-02 5.076e-02 0.580 0.5620   
## incident\_typeNum 1.272e-02 2.118e-01 0.060 0.9521   
## incident\_severityNum -1.311e+00 1.164e-01 -11.261 <2e-16 \*\*\*  
## authorities\_contactedNum 2.312e-02 5.486e-02 0.421 0.6735   
## incident\_stateNum -3.217e-02 3.922e-02 -0.820 0.4121   
## incident\_cityNum -7.100e-02 4.104e-02 -1.730 0.0836 .   
## auto\_makeNum -1.991e-02 2.131e-02 -0.934 0.3503   
## auto\_modelNum -3.834e-03 7.736e-03 -0.496 0.6202   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1117.46 on 998 degrees of freedom  
## Residual deviance: 897.68 on 966 degrees of freedom  
## AIC: 963.68  
##   
## Number of Fisher Scoring iterations: 5

Odds ratio for Permissible Variables Model:

# calculate odds ratio  
exp(coef(logModel))

## (Intercept) weeks\_as\_customer   
## 13.5852170 1.0002540   
## age policy\_number   
## 0.9829214 0.9999996   
## cslBodily cslProp   
## 1.0037947 0.9975851   
## policy\_deductable policy\_annual\_premium   
## 1.0000376 0.9997951   
## umbrella\_limit insured\_zip   
## 1.0000001 1.0000004   
## capital.gains capital.loss   
## 0.9999983 0.9999969   
## incident\_hour\_of\_the\_day number\_of\_vehicles\_involved   
## 0.9913880 0.9825962   
## bodily\_injuries witnesses   
## 1.0797609 1.1122237   
## total\_claim\_amount vehicle\_claim   
## 0.9999809 1.0000375   
## auto\_year weeks\_bf\_incident   
## 0.9997127 0.9999637   
## policy\_stateNum insured\_sexNum   
## 1.0989244 1.1175149   
## insured\_education\_levelNum insured\_occupationNum   
## 1.0195295 1.0042804   
## insured\_hobbiesNum insured\_relationshipNum   
## 0.9760329 1.0298719   
## incident\_typeNum incident\_severityNum   
## 1.0128043 0.2696240   
## authorities\_contactedNum incident\_stateNum   
## 1.0233847 0.9683421   
## incident\_cityNum auto\_makeNum   
## 0.9314576 0.9802911   
## auto\_modelNum   
## 0.9961738

Next, I tried to improve on the accuracy of the logistic regression model by using only some of the Permissible Variables in the formula. I reviewed the odds ratio from the Permissible Variable Model and only included the variables that had an odds ratio of 1 or greater. However, the accuracy from this model decreased to 75.18%.

Because incident\_severity showed to be significant in the counts, histogram and correlation analysis I added incident\_severity back to the model even though it only showed an odds ratio of 0.273. The accuracy results of this model (the “Odds Ratio + Severity Model”) is 81.48%:

claimsMR8 <- claimsNum[, c(38:38, 1:1, 5:5, 7:7, 9:10, 20:20, 28:29, 34:34, 40:43, 45:46, 49:49)]  
logModel <- glm(fraud\_reported ~ ., data = claimsMR8, family = binomial(), maxit = 100)  
# make predictions  
predlogModel <- predict(logModel, type = "response")  
  
# make confusion matrix  
confMatrix <- table(Actual\_value = claimsNum$fraud\_reported,   
 Predicted\_Value = predlogModel > 0.50)  
confMatrix

## Predicted\_Value  
## Actual\_value FALSE TRUE  
## N 651 101  
## Y 84 163

# calculate accuracy  
(confMatrix[[1,1]] + confMatrix[[2,2]]) / sum(confMatrix) \* 100

## [1] 81.48148

I tried various other combinations of variables for the logistic regression model and the highest accuracy I found is the 81.48% for the “Odds Ratio + Severity Model”.

## Classification Model

Classification algorithms can be used predict categorical outcomes. My research question is ultimately determining the outcome of the categorical fraud\_reported variable and thus I am using a k nearest neighbor model to try to predict the fraud\_reported variable.

Because the dataframe has many categorical variables, I used one-hot encoding to convert the factors to dummy variables so that they may be used in the training data and test data in the machine learning model.

The following is the code to create dummy variables for the categorical variables:

# Converting every categorical variable to numerical using dummy variables  
dummy <- dummyVars(" ~ .", data=claims, fullRank = T)  
claimsDummy <- data.frame(predict(dummy, newdata = claims))  
# Converting the dependent variable back to categorical   
claimsDummy$fraud\_reported.Y <- as.factor(claimsDummy$fraud\_reported.Y)

The classification model I am using is the k nearest neighbor model. The following fits a k nearest neighbor model for the claimsDummy data set:

set.seed(123)  
# data splicing  
claimsDummyOutcome <- claimsDummy[, c("fraud\_reported.Y")]  
split <- sample.split(claimsDummyOutcome, SplitRatio = 0.65)  
train <- subset(claimsDummy, split == "TRUE")  
test <- subset(claimsDummy, split == "FALSE")  
# 144 is the outcome variable  
train\_claims <- claimsDummy[split == "TRUE", 144]  
test\_claims <- claimsDummy[split == "FALSE", 144]  
# knn model  
knn.roundup <- knn(train=train, test=test, cl=train\_claims, k=5)  
#Calculate the proportion of correct classification   
ACC.roundup <- 100 \* sum(test\_claims == knn.roundup)/NROW(test\_claims)  
ACC.roundup

## [1] 99.71347

# Check prediction against actual value in tabular form   
confMatrixUp <- table(knn.roundup, test\_claims)  
confMatrixUp

## test\_claims  
## knn.roundup 0 1  
## 0 348 1

However, please note that the accuracy result and confusion matrix that displays in the Rmd file (above) does not match the accuracy result and confusion matrix in the R file. The accuarcy result for this same code was 71.06017% and the confusion matrix was as follows:

# test\_claims  
#knn.roundup 0 1  
# 0 238 76  
# 1 25 10

I have tried to figure out why the Rmd displays different results from the R file in the console of RStudio, but I cannot fix it. The Rmd results shown above appear incorrect because they do not show the last line of the confusion matrix and has an accuracy result of 99%+, which is very much higher than anything RStudio is producing in any of the scenarios that I ran.