MD\_Drivers\_Crash

Rachel Rolle

2024-05-25

Data Source: [Data\_Info](https://catalog.data.gov/dataset/crash-reporting-drivers-data)

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.3 ✔ readr 2.1.4  
## ✔ forcats 1.0.0 ✔ stringr 1.5.0  
## ✔ ggplot2 3.4.4 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.2 ✔ tidyr 1.3.0  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

## Brief Exploration of the Data

crashes <- read.csv("Crash\_Reporting\_Drivers\_Data.csv")  
  
dim(crashes) #172105 rows and 43 columns

## [1] 172105 43

crashes <-na.omit(crashes)  
dim(crashes) #172071 rows and 43 columns

## [1] 172071 43

#date <- crashes %>%  
 #mutate(DATE\_TIME = as.Date(Crash.Date.Time, format = "%m/%d/%Y %H:%M:%S"))  
  
# Find the oldest date  
#oldest\_date <- min(date$DATE\_TIME)  
  
# Find the newest date  
#newest\_date <- max(date$DATE\_TIME)  
  
# Print the results  
#print(paste("Oldest date:", oldest\_date)) , 2015-01-01  
#print(paste("Newest date:", newest\_date)) , 2024-01-03

## Data Cleaning

* Removing classes that were Unknown, Other … to increase validity.
* Reduced variable’s, Substance.Abuse & Injury.Severity, classes using case\_when()
* Selected only 12 out of the 42 variable to simplify problem

library(lubridate)  
set.seed(42)  
is\_unwanted <- function(x) {  
 x %in% c("UNKNOWN", "Unknown", "OTHER", "(Other)")  
}  
#print(colnames(crashes))  
crashes <- crashes %>%  
 filter(!if\_any(everything(), is\_unwanted)) %>%  
 mutate(  
 DATE\_TIME = as.POSIXct(Crash.Date.Time, format = "%m/%d/%Y %H:%M:%S"),  
 Substance.Abuse = case\_when(  
 Driver.Substance.Abuse %in% c("ALCOHOL PRESENT", "ILLEGAL DRUG PRESENT", "MEDICATION PRESENT", "COMBINED SUBSTANCE PRESENT") ~ "Yes",  
 Driver.Substance.Abuse %in% c("ALCOHOL CONTRIBUTED", "ILLEGAL DRUG CONTRIBUTED", "MEDICATION CONTRIBUTED", "COMBINATION CONTRIBUTED") ~ "Yes",  
 TRUE ~ "No"  
 ),  
 Injury.Severity = case\_when(  
 Injury.Severity %in% c("NO APPARENT INJURY") ~ "NO INJURY",  
 Injury.Severity %in% c("SUSPECTED SERIOUS INJURY", "FATAL INJURY", "POSSIBLE INJURY", "SUSPECTED MINOR INJURY") ~ "INJURY"  
 ),  
 across(c(Injury.Severity, Vehicle.Body.Type, Driver.At.Fault, Substance.Abuse, Traffic.Control, Weather, Equipment.Problems, Collision.Type), as.factor)  
 )%>%  
 mutate(  
 DATE = as.Date(DATE\_TIME),  
 TIME = hms::as\_hms(format(DATE\_TIME, "%H:%M:%S"))  
 )%>%  
 select(Injury.Severity, Speed.Limit, Vehicle.Body.Type, Driver.At.Fault, Substance.Abuse, Traffic.Control, Weather, Equipment.Problems, Collision.Type, TIME, Latitude, Longitude)  
  
dim(crashes) #104174 rows & 12 columns

## [1] 104174 12

head(crashes)

## Injury.Severity Speed.Limit Vehicle.Body.Type Driver.At.Fault  
## 1 NO INJURY 35 PICKUP TRUCK Yes  
## 2 NO INJURY 40 PASSENGER CAR Yes  
## 3 NO INJURY 35 (SPORT) UTILITY VEHICLE Yes  
## 4 NO INJURY 30 TRANSIT BUS No  
## 5 NO INJURY 25 (SPORT) UTILITY VEHICLE Yes  
## 6 NO INJURY 35 VAN No  
## Substance.Abuse Traffic.Control Weather Equipment.Problems  
## 1 No TRAFFIC SIGNAL CLEAR NO MISUSE  
## 2 No NO CONTROLS CLEAR NO MISUSE  
## 3 No TRAFFIC SIGNAL CLOUDY NO MISUSE  
## 4 No NO CONTROLS CLEAR NO MISUSE  
## 5 No NO CONTROLS CLEAR NO MISUSE  
## 6 No TRAFFIC SIGNAL CLEAR NO MISUSE  
## Collision.Type TIME Latitude Longitude  
## 1 STRAIGHT MOVEMENT ANGLE 03:10:00 39.10954 -77.07581  
## 2 STRAIGHT MOVEMENT ANGLE 12:10:00 39.19015 -77.26677  
## 3 HEAD ON LEFT TURN 06:10:00 39.17256 -77.20375  
## 4 SAME DIR REAR END 07:40:00 38.99450 -77.02733  
## 5 SAME DIR REAR END 08:28:00 39.00811 -77.02016  
## 6 SAME DIR REAR END 05:00:00 39.08380 -76.95131

## Need to Undersample Injury Status

* Under sampled majority class (no injury)

library(caret)

## Warning: package 'caret' was built under R version 4.3.3

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

set.seed(42)  
  
for (col in names(crashes)) {  
 # Check if the column is a factor  
 if (is.factor(crashes[[col]])) {  
 # Check if "N/A" is a level in the factor  
 if ("N/A" %in% levels(crashes[[col]])) {  
 # Remove the "N/A" level from the factor  
 crashes[[col]] <- droplevels(crashes[[col]], exclude = "N/A")  
 }  
 }  
}  
  
# Remove rows with any NA values  
crashes <- na.omit(crashes)  
  
# Calculate the number of samples in the minority class  
minority\_count <- sum(crashes$Injury.Severity == "INJURY")  
  
# Sample randomly from the majority class to match the number of samples in the minority class  
crashes <- rbind(  
 crashes[crashes$Injury.Severity == "INJURY", ],  
 crashes[sample(which(crashes$Injury.Severity == "NO INJURY"), minority\_count), ]  
)  
  
dim(crashes) #29094 rows & 12 columns

## [1] 29094 12

head(crashes)

## Injury.Severity Speed.Limit Vehicle.Body.Type Driver.At.Fault  
## 12 INJURY 40 MOTORCYCLE Yes  
## 15 INJURY 25 PASSENGER CAR Yes  
## 16 INJURY 35 PASSENGER CAR Yes  
## 17 INJURY 35 PASSENGER CAR No  
## 22 INJURY 35 PASSENGER CAR No  
## 23 INJURY 35 PASSENGER CAR No  
## Substance.Abuse Traffic.Control Weather Equipment.Problems  
## 12 No NO CONTROLS CLEAR NO MISUSE  
## 15 No FLASHING TRAFFIC SIGNAL CLEAR NO MISUSE  
## 16 No NO CONTROLS CLEAR NO MISUSE  
## 17 No TRAFFIC SIGNAL CLEAR NO MISUSE  
## 22 No NO CONTROLS CLEAR NO MISUSE  
## 23 No TRAFFIC SIGNAL CLOUDY NO MISUSE  
## Collision.Type TIME Latitude Longitude  
## 12 STRAIGHT MOVEMENT ANGLE 04:45:00 39.24856 -77.23393  
## 15 STRAIGHT MOVEMENT ANGLE 12:57:00 39.04309 -77.05200  
## 16 SAME DIR REAR END 09:25:00 39.11174 -76.92952  
## 17 ANGLE MEETS LEFT TURN 11:53:00 38.96675 -77.10589  
## 22 SAME DIR REAR END 09:25:00 39.11174 -76.92952  
## 23 STRAIGHT MOVEMENT ANGLE 06:25:00 39.19813 -77.24508

summary(crashes)

## Injury.Severity Speed.Limit Vehicle.Body.Type  
## INJURY :14547 Min. : 0.00 PASSENGER CAR :21435   
## NO INJURY:14547 1st Qu.:30.00 (SPORT) UTILITY VEHICLE: 3170   
## Median :35.00 PICKUP TRUCK : 944   
## Mean :35.14 VAN : 884   
## 3rd Qu.:40.00 TRANSIT BUS : 439   
## Max. :70.00 SCHOOL BUS : 321   
## (Other) : 1901   
## Driver.At.Fault Substance.Abuse Traffic.Control   
## No :17307 No :28457 TRAFFIC SIGNAL :13215   
## Yes:11787 Yes: 637 NO CONTROLS :12133   
## STOP SIGN : 2847   
## FLASHING TRAFFIC SIGNAL: 525   
## YIELD SIGN : 285   
## PERSON : 50   
## (Other) : 39   
## Weather Equipment.Problems  
## CLEAR :21095 NO MISUSE :29016   
## RAINING : 4068 AIR BAG FAILED : 61   
## CLOUDY : 3455 BELT(S) MISUSED : 7   
## SNOW : 232 BELTS/ANCHORS BROKE: 5   
## FOGGY : 138 FACING WRONG WAY : 2   
## WINTRY MIX: 46 STRAP/TETHER LOOSE : 2   
## (Other) : 60 (Other) : 1   
## Collision.Type TIME Latitude   
## SAME DIR REAR END :10747 Length:29094 Min. :38.78   
## STRAIGHT MOVEMENT ANGLE : 7305 Class1:hms 1st Qu.:39.03   
## HEAD ON LEFT TURN : 3321 Class2:difftime Median :39.08   
## SINGLE VEHICLE : 2173 Mode :numeric Mean :39.09   
## SAME DIRECTION SIDESWIPE: 2019 3rd Qu.:39.15   
## HEAD ON : 807 Max. :39.72   
## (Other) : 2722   
## Longitude   
## Min. :-79.49   
## 1st Qu.:-77.20   
## Median :-77.11   
## Mean :-77.12   
## 3rd Qu.:-77.05   
## Max. :-75.98   
##

## Train and Test Split

set.seed(42)  
  
trainIndex <- sample(1:nrow(crashes), size = 0.8 \* nrow(crashes))  
  
trainData <- crashes[trainIndex, ]  
testData <- crashes[-trainIndex, ]  
  
dim(trainData) #23275 rows and 12 columns

## [1] 23275 12

dim(testData) #5819 rows and 12 columns

## [1] 5819 12

# Exploratory Data Analysis

str(trainData) #Checking variable types

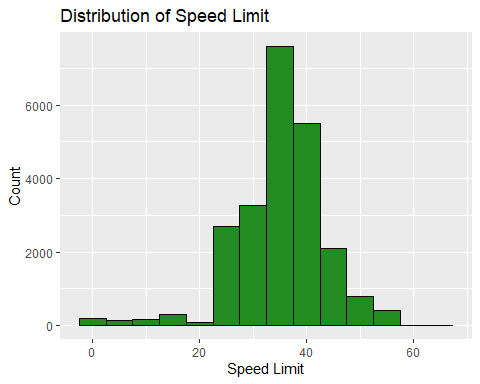
## 'data.frame': 23275 obs. of 12 variables:  
## $ Injury.Severity : Factor w/ 2 levels "INJURY","NO INJURY": 2 2 2 2 1 1 2 1 1 1 ...  
## $ Speed.Limit : int 30 30 15 45 30 30 35 45 40 55 ...  
## $ Vehicle.Body.Type : Factor w/ 28 levels "(SPORT) UTILITY VEHICLE",..: 18 18 9 18 18 15 28 18 18 18 ...  
## $ Driver.At.Fault : Factor w/ 2 levels "No","Yes": 2 1 2 1 1 1 1 1 2 1 ...  
## $ Substance.Abuse : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Traffic.Control : Factor w/ 9 levels "FLASHING TRAFFIC SIGNAL",..: 2 6 2 2 2 2 6 2 6 7 ...  
## $ Weather : Factor w/ 10 levels "BLOWING SAND, SOIL, DIRT",..: 3 3 3 3 4 3 3 3 3 3 ...  
## $ Equipment.Problems: Factor w/ 8 levels "AIR BAG FAILED",..: 5 5 5 5 5 5 5 5 5 5 ...  
## $ Collision.Type : Factor w/ 16 levels "ANGLE MEETS LEFT HEAD ON",..: 9 16 15 9 9 16 16 9 5 16 ...  
## $ TIME : 'hms' num 09:09:00 02:58:00 08:10:00 10:40:00 ...  
## ..- attr(\*, "units")= chr "secs"  
## $ Latitude : num 39 39.2 39.2 39.1 39.1 ...  
## $ Longitude : num -77 -77.2 -77.2 -77.1 -77.2 ...  
## - attr(\*, "na.action")= 'omit' Named int [1:33595] 7 8 18 24 31 33 35 36 37 40 ...  
## ..- attr(\*, "names")= chr [1:33595] "7" "8" "18" "24" ...

summary(trainData)

## Injury.Severity Speed.Limit Vehicle.Body.Type  
## INJURY :11609 Min. : 0.00 PASSENGER CAR :17108   
## NO INJURY:11666 1st Qu.:30.00 (SPORT) UTILITY VEHICLE: 2524   
## Median :35.00 PICKUP TRUCK : 762   
## Mean :35.15 VAN : 734   
## 3rd Qu.:40.00 TRANSIT BUS : 353   
## Max. :65.00 SCHOOL BUS : 257   
## (Other) : 1537   
## Driver.At.Fault Substance.Abuse Traffic.Control   
## No :13877 No :22765 TRAFFIC SIGNAL :10563   
## Yes: 9398 Yes: 510 NO CONTROLS : 9730   
## STOP SIGN : 2257   
## FLASHING TRAFFIC SIGNAL: 418   
## YIELD SIGN : 233   
## PERSON : 44   
## (Other) : 30   
## Weather Equipment.Problems  
## CLEAR :16893 NO MISUSE :23213   
## RAINING : 3200 AIR BAG FAILED : 45   
## CLOUDY : 2803 BELT(S) MISUSED : 7   
## SNOW : 187 BELTS/ANCHORS BROKE: 5   
## FOGGY : 108 FACING WRONG WAY : 2   
## WINTRY MIX: 37 STRAP/TETHER LOOSE : 2   
## (Other) : 47 (Other) : 1   
## Collision.Type TIME Latitude   
## SAME DIR REAR END :8629 Length:23275 Min. :38.78   
## STRAIGHT MOVEMENT ANGLE :5801 Class1:hms 1st Qu.:39.03   
## HEAD ON LEFT TURN :2674 Class2:difftime Median :39.08   
## SINGLE VEHICLE :1748 Mode :numeric Mean :39.09   
## SAME DIRECTION SIDESWIPE:1618 3rd Qu.:39.15   
## HEAD ON : 645 Max. :39.72   
## (Other) :2160   
## Longitude   
## Min. :-79.49   
## 1st Qu.:-77.20   
## Median :-77.11   
## Mean :-77.12   
## 3rd Qu.:-77.05   
## Max. :-75.98   
##

### Numerical Variables

# Histogram for Speed Limit  
ggplot(trainData, aes(x = Speed.Limit)) +  
 geom\_histogram(binwidth = 5, fill = "forestgreen", color = "black") +  
 labs(title = "Distribution of Speed Limit", x = "Speed Limit", y = "Count")

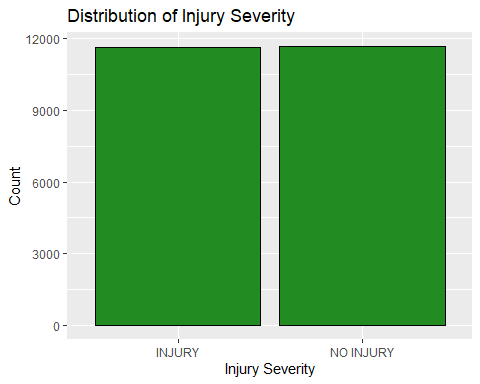


# Summary statistics for Speed Limit  
summary(trainData$Speed.Limit)

## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## 0.00 30.00 35.00 35.15 40.00 65.00

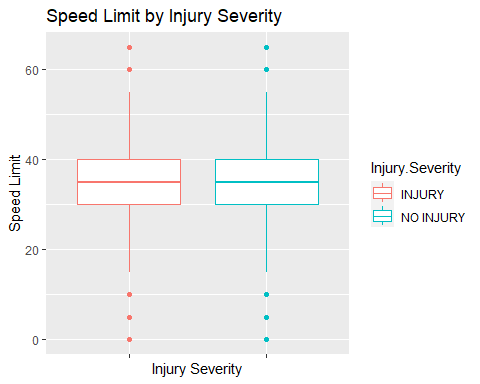
### Categorical Variables

# Bar plot for Injury Severity: Classes are now sampled  
ggplot(trainData, aes(x = Injury.Severity)) +  
 geom\_bar(fill = "forestgreen", color = "black") +  
 labs(title = "Distribution of Injury Severity", x = "Injury Severity", y = "Count")



## Speed vs Injury Severity

# Box plot for Speed Limit by Injury Severity  
ggplot(trainData, aes(x = Injury.Severity, y = Speed.Limit, color = Injury.Severity)) +  
 geom\_boxplot() +  
 labs(title = "Speed Limit by Injury Severity", x = "Injury Severity", y = "Speed Limit",  
 x = NULL,  
 y = "Count")+  
 theme(axis.text.x = element\_blank())

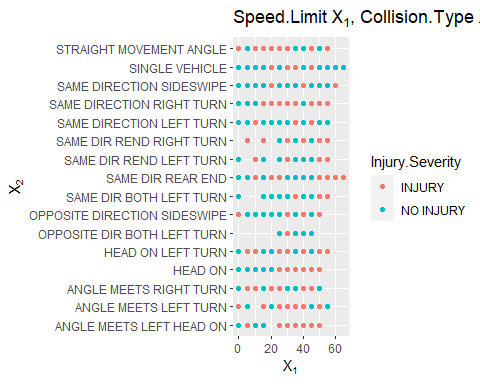


## Plot and Color code by Categorical Response Variable

More classes: Vehicle Type, Traffic Control, Weather, Equipment Problems, Collison Type

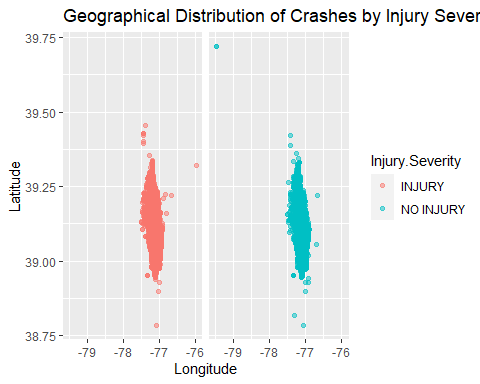
Simple (Yes or No Response): Driver.At.Fault, Substance.Abuse

ggplot(trainData, aes(Speed.Limit, Collision.Type, color = Injury.Severity))+  
 geom\_point()+  
 labs(title = latex2exp::TeX("Speed.Limit $X\_1$, Collision.Type $X\_2$ & Response Y with 2 Cat."))+   
 xlab(latex2exp::TeX("$X\_1$")) +  
 ylab(latex2exp::TeX("$X\_2$"))



## Geospatial Analysis

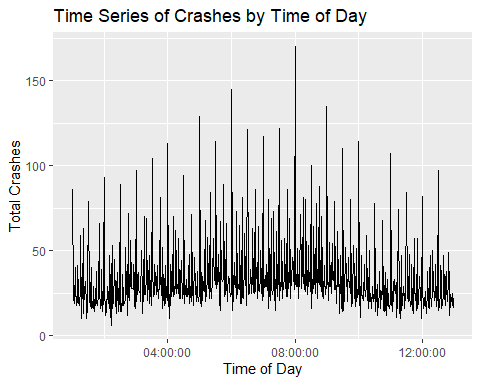
ggplot(crashes, aes(x = Longitude, y = Latitude)) +  
 geom\_point(aes(color = Injury.Severity), alpha = 0.5) +  
 labs(title = "Geographical Distribution of Crashes by Injury Severity", x = "Longitude", y = "Latitude")+  
 facet\_wrap(~ Injury.Severity)+  
 theme(strip.text = element\_blank())



# Fatal injury and serious injury seems to be congested into one area not spread out so proactive measures should be a focus on this area

## Time Series Analysis -go back

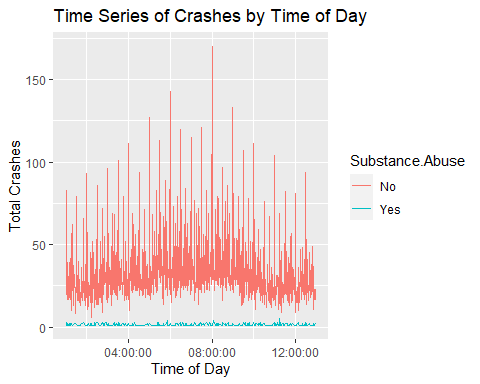
crashes\_by\_date <- trainData %>%  
 group\_by(TIME) %>%  
 summarise(Total\_Crashes = n())%>%  
 arrange(TIME)  
  
# Time series plot  
ggplot(crashes\_by\_date, aes(x = TIME, y = Total\_Crashes)) +  
 geom\_line() +  
 labs(title = "Time Series of Crashes by Time of Day", x = "Time of Day", y = "Total Crashes") # more people are driving during the day so naturally you will see more crashes



#Substance Abuse  
crashes\_by\_time <- trainData %>%  
 group\_by(TIME, Substance.Abuse) %>%  
 summarise(Total\_Crashes = n())

## `summarise()` has grouped output by 'TIME'. You can override using the  
## `.groups` argument.

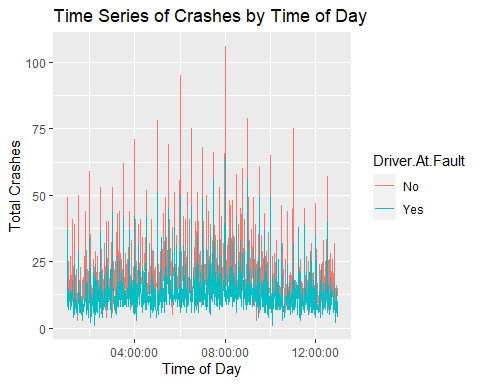
# Time series plot  
ggplot(crashes\_by\_time, aes(x = TIME, y = Total\_Crashes, color = Substance.Abuse)) +  
 geom\_line() +  
 labs(title = "Time Series of Crashes by Time of Day", x = "Time of Day", y = "Total Crashes")



#Driver at Fault  
  
crashes\_by\_time <- trainData %>%  
 group\_by(TIME, Driver.At.Fault) %>%  
 summarise(Total\_Crashes = n())

## `summarise()` has grouped output by 'TIME'. You can override using the  
## `.groups` argument.

# Time series plot  
ggplot(crashes\_by\_time, aes(x = TIME, y = Total\_Crashes, color = Driver.At.Fault)) +  
 geom\_line() +  
 labs(title = "Time Series of Crashes by Time of Day", x = "Time of Day", y = "Total Crashes")



## Contingency Table & Chisquared Test

## Convert to dummy variables to be further assessed

From the correlation matrix, there seems to be no collinearity between variables.

library(caret)  
  
# Convert qualitative variables to dummy variables  
set.seed(42)  
dummy\_model <- dummyVars(~ . - 1 - Speed.Limit - TIME - Latitude -Longitude, data = trainData, fullRank = TRUE)  
  
trainData\_dummy <- predict(dummy\_model, newdata = trainData)  
  
tdummy\_model <- dummyVars(~ . - 1 - Speed.Limit - TIME - Latitude -Longitude, data = trainData, fullRank = TRUE)  
  
testData\_dummy <- predict(dummy\_model, newdata = testData)  
  
# Include original variables back into training dataset  
trainData\_final <- cbind(trainData[, c("Speed.Limit")], trainData\_dummy)  
  
# Include original variables back into testing dataset  
testData\_final <- cbind(testData[, c("Speed.Limit")], testData\_dummy)  
  
# Calculate correlations  
cor\_matrix <- cor(trainData\_final)

## Warning in cor(trainData\_final): the standard deviation is zero

#print(cor\_matrix)

## Logistic Regression

#injury.glm <- glm(Injury.Severity~., data =crashes, family= binomial)  
#AIC<- step(injury.glm, direction = "both") #AIC  
  
#BIC <- step(injury.glm, direction="both", k=log(length(injury.glm$fitted.values))) #BIC  
  
  
#May be the best for a complex model  
AIC\_formula <- (Injury.Severity ~ Speed.Limit + Vehicle.Body.Type + Driver.At.Fault +   
 Substance.Abuse + Traffic.Control + Collision.Type + Latitude) #7  
  
# Simpler Formula  
BIC\_formula <- (Injury.Severity ~ Speed.Limit + Vehicle.Body.Type + Driver.At.Fault +   
 Collision.Type + Latitude) #5

Compare Logistic Regression to new formula *There is statistical significance of adding more variables to the improve the model*

red <- glm(BIC\_formula, family = binomial, data = trainData)  
full <- glm(AIC\_formula, family= binomial, data= trainData)  
  
aout <- anova(red, full, test= "Chisq")  
aout

## Analysis of Deviance Table  
##   
## Model 1: Injury.Severity ~ Speed.Limit + Vehicle.Body.Type + Driver.At.Fault +   
## Collision.Type + Latitude  
## Model 2: Injury.Severity ~ Speed.Limit + Vehicle.Body.Type + Driver.At.Fault +   
## Substance.Abuse + Traffic.Control + Collision.Type + Latitude  
## Resid. Df Resid. Dev Df Deviance Pr(>Chi)   
## 1 23229 29748   
## 2 23220 29725 9 22.968 0.006268 \*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# Correcting the family parameter and removing duplicate exclusions  
injury.glm <- glm(AIC\_formula, family = binomial, data = trainData)  
  
# Display the summary of the fitted model  
glm\_results <- summary(injury.glm)  
  
  
p\_values = glm\_results$coefficients[, "Pr(>|z|)"]   
  
  
cbind(p\_values)

## p\_values  
## (Intercept) 1.601787e-15  
## Speed.Limit 1.323334e-29  
## Vehicle.Body.TypeALL TERRAIN VEHICLE (ATV) 8.820046e-01  
## Vehicle.Body.TypeAMBULANCE/EMERGENCY 1.537902e-02  
## Vehicle.Body.TypeAMBULANCE/NON EMERGENCY 2.272190e-02  
## Vehicle.Body.TypeAUTOCYCLE 9.309829e-01  
## Vehicle.Body.TypeCARGO VAN/LIGHT TRUCK 2 AXLES (OVER 10,000LBS (4,536 KG)) 2.864154e-07  
## Vehicle.Body.TypeCROSS COUNTRY BUS 9.565989e-01  
## Vehicle.Body.TypeFARM VEHICLE 9.358631e-01  
## Vehicle.Body.TypeFIRE VEHICLE/EMERGENCY 4.477869e-04  
## Vehicle.Body.TypeFIRE VEHICLE/NON EMERGENCY 2.268627e-02  
## Vehicle.Body.TypeLIMOUSINE 9.562289e-01  
## Vehicle.Body.TypeLOW SPEED VEHICLE 7.292997e-01  
## Vehicle.Body.TypeMEDIUM/HEAVY TRUCKS 3 AXLES (OVER 10,000LBS (4,536KG)) 2.924763e-05  
## Vehicle.Body.TypeMOPED 1.170455e-02  
## Vehicle.Body.TypeMOTORCYCLE 2.708015e-17  
## Vehicle.Body.TypeOTHER BUS 1.624254e-02  
## Vehicle.Body.TypeOTHER LIGHT TRUCKS (10,000LBS (4,536KG) OR LESS) 5.318914e-08  
## Vehicle.Body.TypePASSENGER CAR 3.611654e-01  
## Vehicle.Body.TypePICKUP TRUCK 7.677777e-11  
## Vehicle.Body.TypePOLICE VEHICLE/EMERGENCY 2.521390e-03  
## Vehicle.Body.TypePOLICE VEHICLE/NON EMERGENCY 5.083143e-08  
## Vehicle.Body.TypeRECREATIONAL VEHICLE 5.961438e-01  
## Vehicle.Body.TypeSCHOOL BUS 9.053211e-20  
## Vehicle.Body.TypeSNOWMOBILE 1.028502e-01  
## Vehicle.Body.TypeSTATION WAGON 1.093586e-01  
## Vehicle.Body.TypeTRANSIT BUS 5.494371e-19  
## Vehicle.Body.TypeTRUCK TRACTOR 3.414244e-05  
## Vehicle.Body.TypeVAN 1.409033e-02  
## Driver.At.FaultYes 1.135817e-123  
## Substance.AbuseYes 1.916535e-01  
## Traffic.ControlNO CONTROLS 4.515547e-02  
## Traffic.ControlPERSON 3.972892e-02  
## Traffic.ControlRAILWAY CROSSING DEVICE 9.514337e-01  
## Traffic.ControlSCHOOL ZONE SIGN DEVICE 9.359397e-01  
## Traffic.ControlSTOP SIGN 4.069068e-02  
## Traffic.ControlTRAFFIC SIGNAL 2.150561e-01  
## Traffic.ControlWARNING SIGN 6.754618e-01  
## Traffic.ControlYIELD SIGN 5.884878e-01  
## Collision.TypeANGLE MEETS LEFT TURN 4.440335e-01  
## Collision.TypeANGLE MEETS RIGHT TURN 6.991627e-02  
## Collision.TypeHEAD ON 2.595158e-01  
## Collision.TypeHEAD ON LEFT TURN 9.008507e-01  
## Collision.TypeOPPOSITE DIR BOTH LEFT TURN 1.419216e-02  
## Collision.TypeOPPOSITE DIRECTION SIDESWIPE 3.687341e-02  
## Collision.TypeSAME DIR BOTH LEFT TURN 5.531521e-06  
## Collision.TypeSAME DIR REAR END 4.513076e-03  
## Collision.TypeSAME DIR REND LEFT TURN 6.215482e-02  
## Collision.TypeSAME DIR REND RIGHT TURN 1.164195e-01  
## Collision.TypeSAME DIRECTION LEFT TURN 2.338951e-07  
## Collision.TypeSAME DIRECTION RIGHT TURN 2.033208e-06  
## Collision.TypeSAME DIRECTION SIDESWIPE 2.355625e-12  
## Collision.TypeSINGLE VEHICLE 7.333285e-01  
## Collision.TypeSTRAIGHT MOVEMENT ANGLE 5.729355e-01  
## Latitude 1.418173e-15

# Fit and Tune Each Model

## Logistic Regression

### Cross Validation

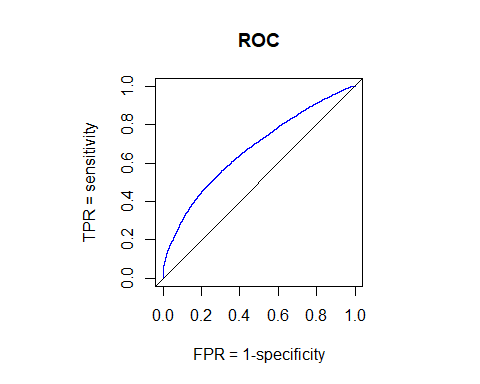
set.seed(42)  
# Define your training control  
train\_control <- trainControl(method = "cv", number = 5, savePredictions = "final")  
  
# Logistic Regression  
# Define the model  
model\_logit <- train(AIC\_formula, data = crashes, method = "glm", trControl = train\_control, family = binomial)  
  
# Access the cross-validated predictions  
predictions <- model\_logit$pred  
  
# Create a confusion matrix  
conf\_matrix <- confusionMatrix(predictions$pred, predictions$obs)  
  
# Print the confusion matrix  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction INJURY NO INJURY  
## INJURY 10045 6410  
## NO INJURY 4502 8137  
##   
## Accuracy : 0.6249   
## 95% CI : (0.6193, 0.6305)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.2499   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.6905   
## Specificity : 0.5594   
## Pos Pred Value : 0.6105   
## Neg Pred Value : 0.6438   
## Prevalence : 0.5000   
## Detection Rate : 0.3453   
## Detection Prevalence : 0.5656   
## Balanced Accuracy : 0.6249   
##   
## 'Positive' Class : INJURY   
##

#Accuracy: 0.6249  
#Sensitivity: 0.6905  
#Specificity: 0.5594

### ROC Curve Threshold

set.seed(42)  
injury.glm <- glm(AIC\_formula, family = binomial, data=trainData)  
  
#summary(injury.glm)  
  
  
roc.analysis <-function (object, newdata = NULL, newplot=TRUE)  
{  
if (is.null(newdata)) {  
pi.tp <- object$fitted[object$y == 1]  
pi.tn <- object$fitted[object$y == 0]  
}  
else {  
pi.tp <- predict(object, newdata, type = "response")[newdata$y == 1]  
pi.tn <- predict(object, newdata, type = "response")[newdata$y == 0]  
}  
pi.all <- sort(c(pi.tp, pi.tn))  
sens <- rep(1, length(pi.all)+1)  
specc <- rep(1, length(pi.all)+1)  
for (i in 1:length(pi.all)) {  
sens[i+1] <- mean(pi.tp >= pi.all[i], na.rm = T)  
specc[i+1] <- mean(pi.tn >= pi.all[i], na.rm = T)  
}  
npoints <- length(sens)  
area <- sum(0.5 \* (sens[-1] + sens[-npoints]) \* (specc[-npoints] -  
specc[-1]))  
lift <- (sens - specc)[-1]  
cutoff <- pi.all[lift == max(lift)][1]  
sensopt <- sens[-1][lift == max(lift)][1]  
specopt <- 1 - specc[-1][lift == max(lift)][1]  
par(pty="s")  
if (newplot){  
plot(specc, sens, col = "blue", xlim = c(0, 1), ylim = c(0, 1), type = "s",  
xlab = "FPR = 1-specificity", ylab = "TPR = sensitivity", main="ROC")  
abline(0, 1)  
}  
else lines(specc, sens, type="s", lty=2, col=2)  
list(pihat=as.vector(pi.all), sens=as.vector(sens[-1]),  
spec=as.vector(1-specc[-1]), area = area, cutoff = cutoff,  
sensopt = sensopt, specopt = specopt)  
}  
  
  
train.ROC <- roc.analysis(injury.glm)  
testData$Direction <- 1\*(testData$Injury.Severity == 1)  
test.ROC <- roc.analysis(injury.glm, newdata=testData, newplot=F)



train.ROC[(4:7)]

## $area  
## [1] 0.6726821  
##   
## $cutoff  
## 42838   
## 0.5018894   
##   
## $sensopt  
## [1] 0.5603463  
##   
## $specopt  
## [1] 0.6935137

set.seed(42)  
test\_prob <- predict(injury.glm, newdata=testData, type="response")  
test\_pred <- ifelse(test\_prob > 0.5018894, "No Injury", "Injury")  
  
# Create the confusion matrix  
confusion\_matrix <- prop.table(table(Actual = testData$Injury.Severity, Predicted= test\_pred))  
print(confusion\_matrix)

## Predicted  
## Actual Injury No Injury  
## INJURY 0.3457639 0.1591339  
## NO INJURY 0.2134387 0.2816635

#Classification  
log.class <- sum(diag(confusion\_matrix)) / sum(confusion\_matrix)  
print(paste("Classification Rate:", log.class)) #0.626

## [1] "Classification Rate: 0.627427393022856"

log.error <- 1-log.class  
print(paste("Error rate:", log.error)) #0.378

## [1] "Error rate: 0.372572606977144"

#Plot results

# Combine the actual and predicted data  
plot\_data <- data.frame(Actual = testData$Injury.Severity,   
 Predicted = test\_pred,   
 Probability = test\_prob)  
  
# Plot  
ggplot(plot\_data, aes(x = Probability, y = Actual)) +  
 geom\_jitter(aes(color = Predicted), width = 0.02, height = 0.1, alpha = 0.2) +  
 geom\_vline(xintercept = 0.5018894, linetype = "dashed", color = "red") +  
 labs(title = "Logistic Regression Results", x = "Predicted Probability", y = "Actual Outcome") +  
 theme\_minimal()

 ## QDA

#### Cross Validation

Accuracy: 0.5504 Sensitivity: 0.6944 Specificity: 0.4064

set.seed(42)  
# Define your training control  
train\_control <- trainControl(method = "cv", number = 5, savePredictions = "final")  
  
# Train the QDA model  
qda\_model <- train(Injury.Severity ~ Latitude + TIME + Speed.Limit, data = crashes, method = "qda", trControl = train\_control)  
  
# Access the cross-validated predictions  
predictions <- qda\_model$pred  
  
# Create a confusion matrix  
conf\_matrix <- confusionMatrix(predictions$pred, predictions$obs)  
  
# Print the confusion matrix  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction INJURY NO INJURY  
## INJURY 10101 8635  
## NO INJURY 4446 5912  
##   
## Accuracy : 0.5504   
## 95% CI : (0.5447, 0.5561)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.1008   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.6944   
## Specificity : 0.4064   
## Pos Pred Value : 0.5391   
## Neg Pred Value : 0.5708   
## Prevalence : 0.5000   
## Detection Rate : 0.3472   
## Detection Prevalence : 0.6440   
## Balanced Accuracy : 0.5504   
##   
## 'Positive' Class : INJURY   
##

#Train-Test-Split

Accuracy: 0.548891562124076 Sensitivity: 0.3354528 Specificity: 0.2134387

# Load the MASS package for QDA  
library(MASS)

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:dplyr':  
##   
## select

library(caret)  
set.seed(42)  
  
# Train the QDA model  
qda\_model <- qda(Injury.Severity ~ Latitude + TIME + Speed.Limit, data = trainData)  
  
# Predictions on new data  
pred.test <- predict(qda\_model, newdata = testData)$class  
  
# Create the confusion matrix  
confusion\_matrix <- prop.table(table(Actual = testData$Injury.Severity, Predicted = pred.test))  
print(confusion\_matrix)

## Predicted  
## Actual INJURY NO INJURY  
## INJURY 0.3354528 0.1694449  
## NO INJURY 0.2816635 0.2134387

# Calculate the classification rate  
qda.class <- mean(pred.test == testData$Injury.Severity)  
print(paste("Classification Rate:", qda.class))

## [1] "Classification Rate: 0.548891562124076"

# Calculate the error rate  
qda.error <- 1 - qda.class  
print(paste("Error rate:", qda.error))

## [1] "Error rate: 0.451108437875924"

## Random Forest

### Cross Validation

Sensitivity: 0.7637 Specificity: 0.4688 Accuracy: 0.6162

set.seed(42)  
# Define your training control  
train\_control <- trainControl(method = "cv", number = 5, savePredictions= "final")  
  
#Model  
model\_rf <- train(AIC\_formula, data = crashes, trControl = train\_control, method = "rf")  
  
  
# Access the cross-validated predictions  
predictions <- model\_rf$pred  
  
# Create a confusion matrix  
conf\_matrix <- confusionMatrix(predictions$pred, predictions$obs)  
  
# Print the confusion matrix  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction INJURY NO INJURY  
## INJURY 11110 7728  
## NO INJURY 3437 6819  
##   
## Accuracy : 0.6162   
## 95% CI : (0.6106, 0.6218)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.2325   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.7637   
## Specificity : 0.4688   
## Pos Pred Value : 0.5898   
## Neg Pred Value : 0.6649   
## Prevalence : 0.5000   
## Detection Rate : 0.3819   
## Detection Prevalence : 0.6475   
## Balanced Accuracy : 0.6162   
##   
## 'Positive' Class : INJURY   
##

### Train Test Split

Accuracy = 0.6343 Sensitivity = 0.7444 Specificity = 0.5220

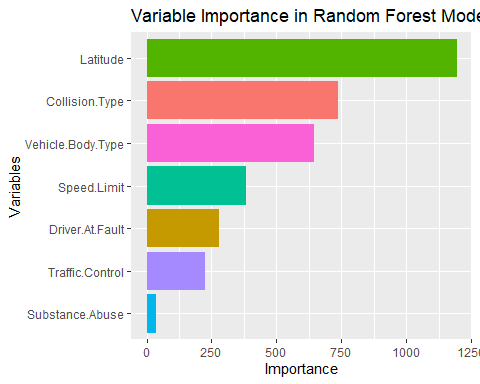
# Load necessary libraries  
library(randomForest)  
library(caret)  
library(ggplot2)  
  
# Set seed for reproducibility  
set.seed(42)  
  
# Assuming AIC\_formula, trainData, and testData are already defined  
# Train the random forest model  
model\_rf <- randomForest(AIC\_formula, data = trainData, mtry = 2)  
  
# Predict on test data  
pred\_rf <- predict(model\_rf, newdata = testData)  
  
# Calculate confusion matrix  
rf\_conf\_matrix <- confusionMatrix(pred\_rf, testData$Injury.Severity)  
print(rf\_conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction INJURY NO INJURY  
## INJURY 2187 1377  
## NO INJURY 751 1504  
##   
## Accuracy : 0.6343   
## 95% CI : (0.6218, 0.6467)  
## No Information Rate : 0.5049   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.267   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.7444   
## Specificity : 0.5220   
## Pos Pred Value : 0.6136   
## Neg Pred Value : 0.6670   
## Prevalence : 0.5049   
## Detection Rate : 0.3758   
## Detection Prevalence : 0.6125   
## Balanced Accuracy : 0.6332   
##   
## 'Positive' Class : INJURY   
##

# Show variable importance  
importance\_rf <- importance(model\_rf)  
print(importance\_rf)

## MeanDecreaseGini  
## Speed.Limit 382.32136  
## Vehicle.Body.Type 644.60069  
## Driver.At.Fault 280.63934  
## Substance.Abuse 35.24251  
## Traffic.Control 223.57823  
## Collision.Type 738.74994  
## Latitude 1194.09299

# Convert importance to a data frame for plotting  
importance\_df <- data.frame(Variable = rownames(importance\_rf), Importance = importance\_rf[, 1])  
  
# Plot variable importance  
ggplot(importance\_df, aes(x = reorder(Variable, Importance), y = Importance, fill = Variable)) +  
 geom\_bar(stat = "identity") +  
 coord\_flip() +  
 xlab("Variables") +  
 ylab("Importance") +  
 ggtitle("Variable Importance in Random Forest Model") +  
 theme(legend.position = "none")



## Support Vector Machine

* SVM takes a lot of computational power train/test split would not be useful Svm model:

The final values used for the model were sigma = (0.04408933) and C = 1.

### Cross Validation

Accuracy: 0.6257 Sensitivity: 0.7433 Specificity: 0.5080

set.seed(42)  
# Define your training control  
train\_control <- trainControl(method = "cv", number = 5, savePredictions = "final")  
  
# Train the SVM model  
svm\_model <- train(AIC\_formula, data = crashes, method = "svmRadial", trControl = train\_control)  
  
# Access the cross-validated predictions  
predictions <- svm\_model$pred  
  
# Create a confusion matrix  
conf\_matrix <- confusionMatrix(predictions$pred, predictions$obs)  
  
# Print the confusion matrix  
print(conf\_matrix)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction INJURY NO INJURY  
## INJURY 10813 7157  
## NO INJURY 3734 7390  
##   
## Accuracy : 0.6257   
## 95% CI : (0.6201, 0.6312)  
## No Information Rate : 0.5   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.2513   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.7433   
## Specificity : 0.5080   
## Pos Pred Value : 0.6017   
## Neg Pred Value : 0.6643   
## Prevalence : 0.5000   
## Detection Rate : 0.3717   
## Detection Prevalence : 0.6177   
## Balanced Accuracy : 0.6257   
##   
## 'Positive' Class : INJURY   
##