EDA – Contributing Factors to Crashes

Several categories of factors contributing in the accidents are recorded in FARS dataset. Speeding, alcohol, drugs, driver distractions, drivers' vision obstruction, and vehicle level contributing circumstances are some examples. Perform a comprehensive review of the contributing factors, analyze the related tables in the dataset, and identify the most common contributing factors in each category.

```
library(MASS)
library(plyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:MASS':
##
##     select
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(caret)
```

```
## Loading required package: lattice
```

```
setwd("/Users/jackfrancis/Dropbox/Heuristics_Optimization/IISE_Data_Analytics/Jack_Code")
```

Initial Thoughts

- Probably a relationship between speeding and fatal crashes.

Code Information:

Table	CSV Name	FARS Data Dictionary Location
Driver Distractions	DISTRACT	545
Driver Impaired	DRIMPAIR	465
Vehicle Defects Contributing to Crash	FACTOR	510
Driver Manuevered to Avoid prior to Crash	MANEUVER	542
Non-Motorist Contributing Circumstances	NMCRASH	734

Table	CSV Name	FARS Data Dictionary Location
Non-Motorist Impaired	NMIMPAIR	743
Non-Motorist Action	NMPRIOR	730
Motorist Violations	VIOLATN	459
Driver Vision Obscured	VISION	539
Damaged Areas	DAMAGE	371

Case 1: Speeding

Important columns are in VEHICLE.csv. SPEEDREL denotes if the crash was speeding related. TRAV_SP is the speed of the driver and VSPD_LIM is speed limit of road crash occured on.

SPEEDREL Codes	Attributes
0	No
2	Yes, Racing
3	Yes, Exceeded Speed Limit
4	Yes, Too Fast for Conditions
5	Yes, Specifics Unknown
8	No Driver Present
9	Unknown

- Note: It may be interesting to analyze the relationship between speed limit and number of crashes.

```
vehicle_df = read.csv("../FARS_Data/FARS2018NationalCSV/VEHICLE.csv")
```

```
frequency_speed = as.data.frame(table(vehicle_df$SPEEDREL))
colnames(frequency_speed) = c("Code", "Freq")
frequency_speed[order(frequency_speed$Freq, decreasing = TRUE), ]
```

```
##   Code  Freq
## 1    0 41190
## 4    4  3850
## 3    3  3432
## 7    9  1735
## 5    5  1245
## 6    8   342
## 2    2    78
```

First, I wanted to investigate the total number of fatal accidents that are speeding related

```
# Percentage of accidents that were speed related
# If two vehicles are in the same crash, only the ones which were speeding will have a speed_rel > 0.
speed_rel <- vehicle_df$SPEEDREL
speed_rel <- mapvalues(speed_rel, from = c(0, 2, 3, 4, 5, 8, 9), to = c(0, 1, 1, 1, 1, 2, 2))
# Some missing values can be collected by comparing the travel speed to the speed limit (2077 -> 481)
for (i in 1:length(speed_rel)) {
    if (speed_rel[i] == 2) {
        if (vehicle_df$TRAV_SP[i] < 151) {
            speed_rel[i] = ifelse((vehicle_df$TRAV_SP[i] - vehicle_df$VSPD_LIM[i]) > 0, 1, 0)
        } else {
            speed_rel[i] = 0
        }
    }
```

```
}

# Now identify which accidents had at least one accident where speed was related
speed_df = data.frame(cbind("Speed_Related" = speed_rel, "State_Case" = vehicle_df$ST_CASE))
accident_speed = aggregate(x = speed_df[c("Speed_Related")], by = speed_df[c("State_Case")], FUN = max)
accident_speed_percentage = sum(accident_speed$Speed_Related)/length(accident_speed$State_Case)
print(sprintf("In Fatal Crashes, Speed is related %s%% of the time", round(accident_speed_percentage, di
```

```
## [1] "In Fatal Crashes, Speed is related 25.29% of the time"
```

Next, of these accidents how are they distributed among the four types of speeding classified by FARS. One note is that there are accidents with multiple types, so the total percentage of the accidents caused by each type will be over 100.

```
# Want to check how many accidents are a result of 1) racing 2) Exceeding Speed Limit 3) Speeding in ba
# First, check if there are multiple wihtin the same accident
speed_rel <- vehicle_df$SPEEDREL
speed_rel <- mapvalues(speed_rel, from = c(0, 2, 3, 4, 5, 8, 9), to = c(0, 1, 2, 3, 4, 0, 0))
speed_df = data.frame(cbind("Speed_Related" = speed_rel, "State_Case" = vehicle_df$ST_CASE))
types_of_speeding = aggregate(x = speed_df[c("Speed_Related")], by = speed_df[c("State_Case")], FUN = su
max(types_of_speeding$Speed_Related)
```

```
## [1] 30
```

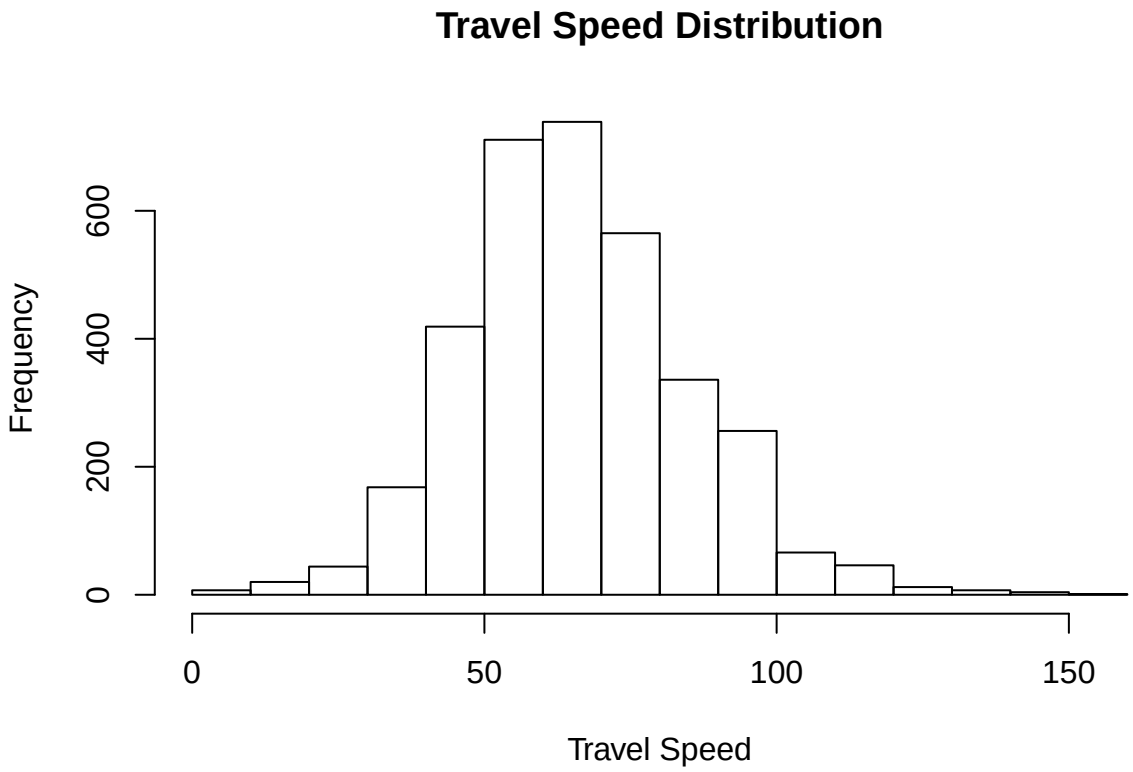
```
# Since there are multiple speeding related for each, we can get the count of state cases joined by the
speed_df = data.frame(cbind("Speed_Related" = speed_rel, "State_Case" = vehicle_df$ST_CASE))
types_of_speeding = aggregate(x = speed_df[c("State_Case")], by = speed_df[c("Speed_Related")], FUN = le

# Calculate percentages (Not necessarily equal to 100 because one case can have multiple types)
total_accidents = length(unique(vehicle_df$ST_CASE))
total_speeding_related_accidents = sum(accident_speed$Speed_Related)
racing_accidents_percentage = types_of_speeding$State_Case[2] / total_speeding_related_accidents * 100
exceed_limit_accidents_percentage = types_of_speeding$State_Case[3] / total_speeding_related_accidents *
bad_cond_accidents_percentage = types_of_speeding$State_Case[4] / total_speeding_related_accidents * 100
other_accidents_percentage = types_of_speeding$State_Case[5] / total_speeding_related_accidents * 100

# Add the percentages to the total_table
speeding_related_accidents = data.frame(matrix(ncol = 3, nrow = 4))
colnames(speeding_related_accidents) = c("Type of Speeding", "Number of Accidents", "Percentage of all A
speeding_related_accidents[,1] = c("Racing", "Exceeded Speed Limit", "Speeding in Bad Conditions", "Othe
speeding_related_accidents[,2] = types_of_speeding$State_Case[2:5]
speeding_related_accidents[,3] = c(round(racing_accidents_percentage, digits = 4),
                                   round(exceed_limit_accidents_percentage, digits = 4),
                                   round(bad_cond_accidents_percentage, digits = 4),
                                   round(other_accidents_percentage, digits = 4))

# This is slightly over 100 because some accidents had more than one.
total_percentage = racing_accidents_percentage + exceed_limit_accidents_percentage +
  bad_cond_accidents_percentage + other_accidents_percentage
```

A few interesting ways to slice the speeding data. First, let's look at the distribution of travel speed for vehicles in a fatal crash. This should have tails on both ends, because cars that were going very slow may have been hit by cars going very fast. Second, by looking at the speed limit of the roads the crashes occur on, we can identify the types of roads that most commonly have fatal crashes. My initial assumption is that there will be large clumpings around 55 and 70, since these are common speed limits for the highway and interstate, respectively. Finally, by comparing the travel speed to the speed limit, we can identify, on average,

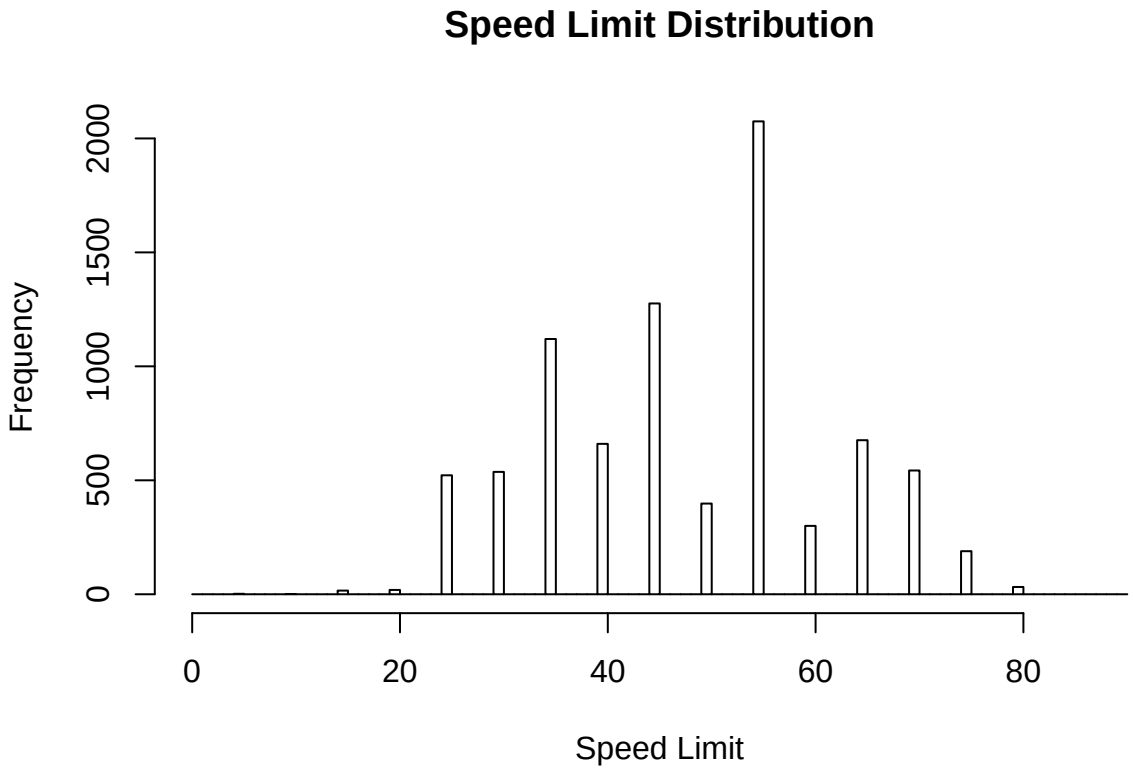
how fast were drivers going over the speed limit before being involved in a fatal crash.

Data coders round all speeds to the nearest integer, so we don't have to worry about decimals here.

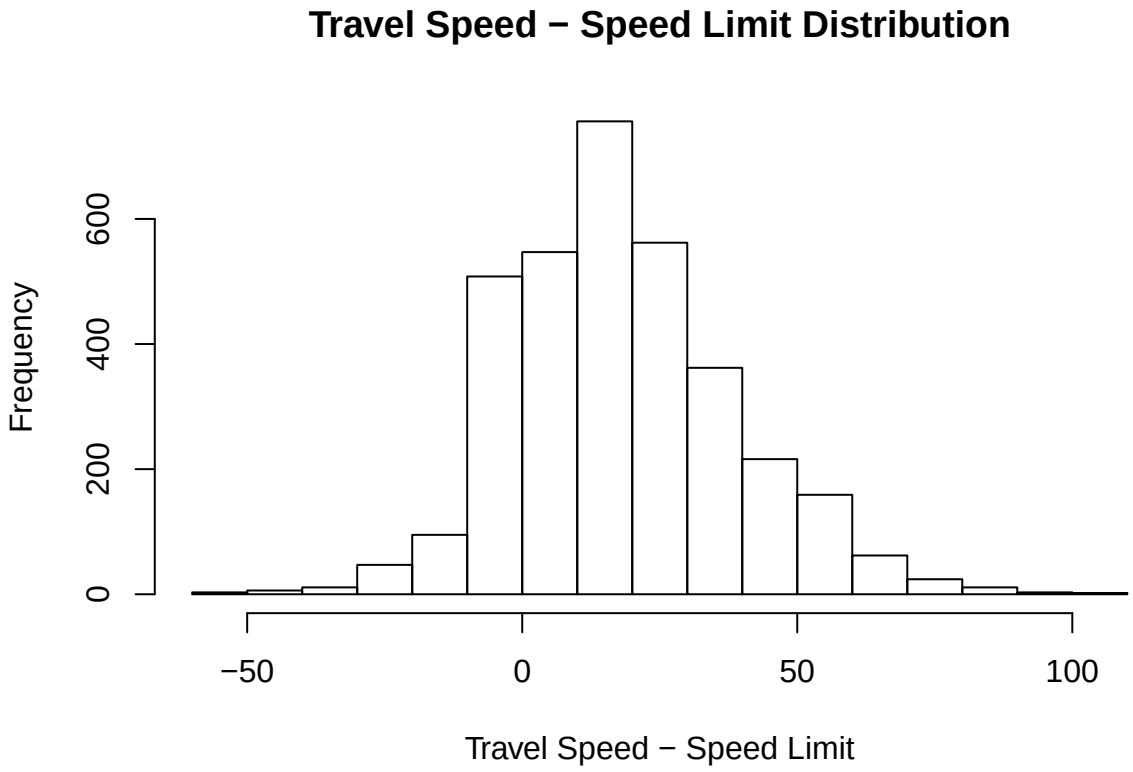
```
speed_rel <- mapvalues(speed_rel, from = c(0, 1, 2, 3, 4), to = c(0, 1, 1,1,1))

# Distribution of how fast drivers were going
travel_speed_distribution = vehicle_df$TRAV_SP[speed_rel == 1]
travel_speed_distribution = travel_speed_distribution[travel_speed_distribution < 152]
travel_speed_distribution = travel_speed_distribution[travel_speed_distribution > 0]
hist(travel_speed_distribution, main = "Travel Speed Distribution", xlab = "Travel Speed")
```




```
# Distribution of the speed limit on roads
speed_limit_distribution = vehicle_df$VSPD_LIM[speed_rel == 1]
speed_limit_distribution = speed_limit_distribution[speed_limit_distribution < 96]
speed_limit_distribution = speed_limit_distribution[speed_limit_distribution > 4]
hist(speed_limit_distribution, breaks = seq(0,90,1), main = "Speed Limit Distribution", xlab = "Speed Li
```




```r
# Distribution of the difference between the driver speed and speed limit.
speeding_amount_distribution = data.frame(cbind(vehicle_df$TRAV_SP, vehicle_df$VSPD_LIM))
speeding_amount_distribution = speeding_amount_distribution[speed_rel == 1,]
speeding_amount_distribution = speeding_amount_distribution[speeding_amount_distribution$X1 < 152,]
speeding_amount_distribution = speeding_amount_distribution[speeding_amount_distribution$X1 > 0,]
speeding_amount_distribution = speeding_amount_distribution[speeding_amount_distribution$X2 < 96,]
speeding_amount_distribution = speeding_amount_distribution[speeding_amount_distribution$X2 > 4,]
hist(speeding_amount_distribution$X1 - speeding_amount_distribution$X2,
      main = "Travel Speed - Speed Limit Distribution", xlab = "Travel Speed - Speed Limit")
```



Case 2: Alcohol

Important columns are in PERSON.csv. DRINKING denotes whether the officer believed alcohol was involved (0: No, 1: Yes, 8: Not Rep, 9: Unknown). Other important columns are: ALC_DET (method by which police made determination), ALC_STATUS (Whether test was administered 0:No, 2:Yes), ATST_TYP (Type of Test), ALC_RES (Test Result)

ALC_DET Codes	Attributes
1	Evidential Test (breath, blood, urine)
2	Preliminary Breath Test (PBT)
3	Behavioral
4	Passive Alcohol Sensor (PAS)
5	Observed
8	Other (e.g., Saliva test)
9	Not Reported

ALC_RES Codes	Attributes
000-939	Actual Value
940	.94 or Greater
996	Test Not Given
997	AC Test Performed, Results Unknown
998	Positive Reading with No Actual Value
995	Not Reported
999	Reported as Unknown if Tested

```
person_df = read.csv("../FARS_Data/FARS2018NationalCSV/PERSON.csv")
accident_df = read.csv("../FARS_Data/FARS2018NationalCSV/Accident.csv")
# Only get important columns
alc_res = person_df$ALC_RES
drinking = person_df$DRINKING
accident_drinking = accident_df$DRUNK_DR
```

A couple interesting questions for alcohol consumption and fatalities. First, how often are drunk people involved in fatal accidents. A side note to this question direclty involves whether the driver was drunk. Second, is there a relationship between how drunk (i.e. BAC level) and fatalities?

There are a few ways to get drunk driver statistics. First, is using the accident table column "DR_DRINK". This column provides the number of drunk drivers in each accident. Another way is to use the person table column "ALC_RES" and test if the BAC of the driver (i.e. seat pos == 11) is greater than 0.8 (the federal limit). The limit in Utah is slighlty lower (0.5), but only 59 out of 8644 drunk driving cases are in Utah, so using the 0.8 benchmark should be sufficient. Let's compare the number of cases using each method.

Look at who is more likely to die in accidents involved in drinking.

```
# Using Accident table. Need to do > 0, because some accidents have more than one drunk driver
accident_drinking = accident_df$DRUNK_DR
num_accident_drinking = accident_drinking[accident_drinking > 0]

# Total number of drunk drivers
sum(num_accident_drinking)
```

```
## [1] 8644
```

```
# Total accidents involving at least one drunk driver
length(num_accident_drinking)
```

```
## [1] 8379
```

```
# Number of drunk drivers using Person BAC > 0.8 guideline
num_person_drinking = alc_res[person_df$SEAT_POS == 11]
num_person_drinking = num_person_drinking[num_person_drinking < 941]
num_person_drinking = num_person_drinking[num_person_drinking > 80]

# Total number of drunk drivers
length(num_person_drinking)
```

```
## [1] 5752
```

```
# Number of accidents drunk drivers were involved using Person BAC > 0.8 guideline
num_person_accident_drinking = data.frame(cbind(alc_res, person_df$ST_CASE))
num_person_accident_drinking = num_person_accident_drinking[person_df$SEAT_POS == 11,]
num_person_accident_drinking = num_person_accident_drinking[num_person_accident_drinking$alc_res > 80,]
num_person_accident_drinking = num_person_accident_drinking[num_person_accident_drinking$alc_res < 941,]
num_person_accident_drinking = aggregate(x = num_person_accident_drinking[c("alc_res")],
                                         by =num_person_accident_drinking[c("V2")], FUN = length)

# Total accidents involving at least one drunk driver
nrow(num_person_accident_drinking)
```

```
## [1] 5666
```

So based on the accident table, there were 8644 drunk drivers, which were spread across 8379 accidents. Based on the person table, using BAC as a guideline, in total there were 5752 drunk drivers, which were

spread across 5666 accidents.

Another interesting line of analysis is to compare how often the police expected drinking was involved, compared to the actual number of drivers that were legally intoxicated

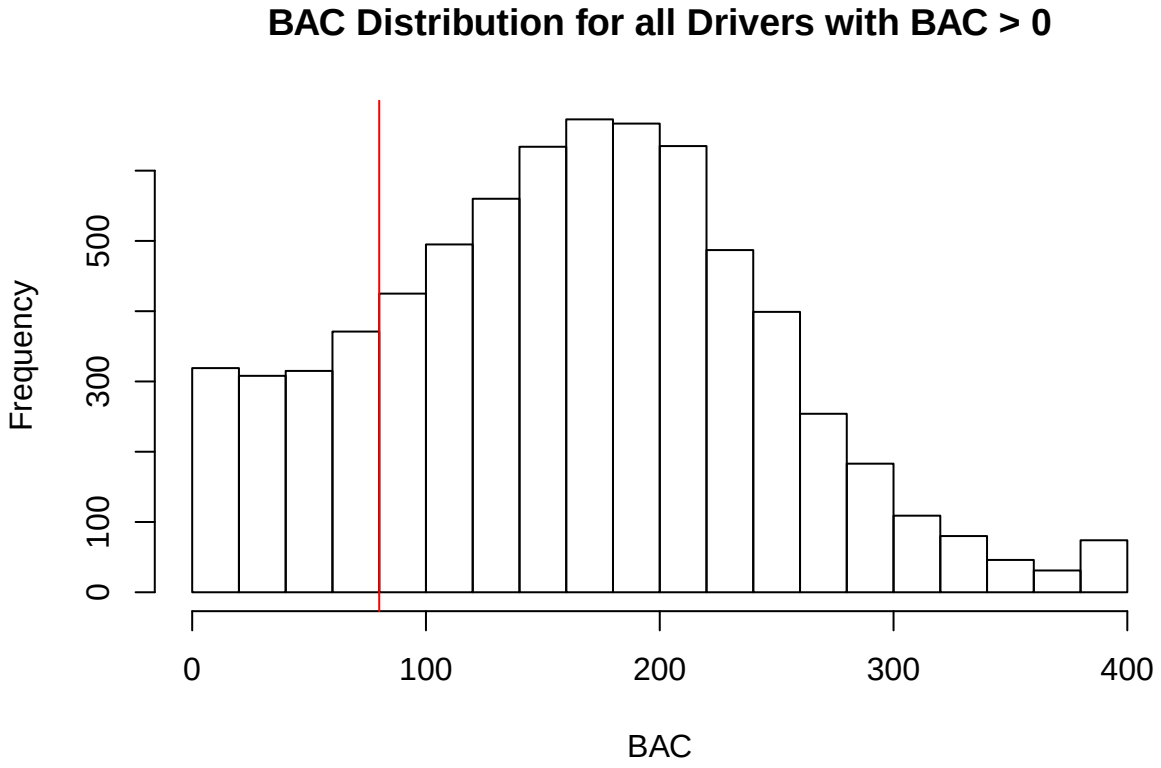
```
# Number of accidents drunk drivers were involved using Person BAC > 0.8 guideline
num_person_accident_drinking = data.frame(cbind(drinking, person_df$ST_CASE))
num_person_accident_drinking = num_person_accident_drinking[person_df$SEAT_POS == 11,]
num_person_accident_drinking = num_person_accident_drinking[num_person_accident_drinking$drinking == 1,]
num_person_accident_drinking = aggregate(x = num_person_accident_drinking[c("drinking")],
                                         by = num_person_accident_drinking[c("V2")], FUN = length)

# Total accidents involving at least one drunk driver
nrow(num_person_accident_drinking)
```

```
## [1] 6112
```

Surprisingly, there were roughly 500 accidents where police believed alcohol was involved, but the driver did not have a BAC over 0.8. It is possible that other occupants in the vehicle were drunk (there are 1350 of these cases by modifying SEAT_POS != 11 above) or the driver was distracted by alcohol, but not drunk. However, we are not given further information about why the police believed alcohol was involved, so this is merely speculation.

Examining the BAC of drivers who were involved in fatal accidents may lead to interesting relationships.

```
# For all people in the person set. Generally, most people are not drinking.
alc_res_all = alc_res[person_df$SEAT_POS == 11]
alc_res_all = alc_res_all[alc_res_all < 941]
# Only 45 cases are greater than 400, so these are moved to 400 for ease of plotting the histogram
alc_res_all[alc_res_all > 400] = 400
# There are 14268 cases where the BAC is 0, so these are removed to show the histogram more clearly
alc_res_non_zero = alc_res_all[alc_res_all > 0]
hist(alc_res_non_zero, main = "BAC Distribution for all Drivers with BAC > 0", xlab = "BAC", breaks = s
abline(v = 80, col = 'red')
```




```
# Total percentage of drivers over a BAC of 0.8
drunk_percentage = length(alc_res_all[alc_res_all > 80]) / length(alc_res_all)
print(sprintf("Number of drivers in fatal crashes who had BAC over the legal limit of 0.8 is %s%%", rou
```

```
## [1] "Number of drivers in fatal crashes who had BAC over the legal limit of 0.8 is 26.96%"
```

When there are fatalities from drunk driving, there are more often drunk people than sober people. This may be related to drunk people riding home together. The BAC follows pretty close to a normal distribution, which is somewhat surprising to me. I expected a large area below 200, and much less area to the right of 200.

Case 3: Drugs

Data is in both DRUGS.csv and PERSON.csv, need to check that similar data is in both. Using DRUGS.csv right now. For DRUGS.csv make sure to group by person number. One person can have multiple drugs. From PERSON.csv, use DRUGS (0: No, 1:Yes, 8: Not Reported, 9:Unknown) DRUG_DET (Type of Test 1:Evidential, 2:Expert, 3:Behavioral, 7:Other, 8:NR) DSTATUS (Test Given 0:No, 2:Yes, 8:NR, 9:Unknown). DRUGS.csv use DRUGSPEC (Type of Test pg 657) DRUGRES (Drug Test Result)

DRUGRES Codes	Attributes
000	Test Not Given
001	Tested, No Drugs Found/Negative
100-295	Narcotic*
300-399	Depressant*
400-495	Stimulant*
500-595	Hallucinogen*
600-695	Cannabinoid*
700-795	Phencyclidine (PCP)*
800-895	Anabolic Steroid*
900-995	Inhalant*

DRUGRES Codes	Attributes
996	Other Drug
997	Tested for Drugs, Results Unknown
998	Tested for Drugs, Drugs Found, Type Unknown/Positive
095	Not Reported
999	Reported as Unknown If Tested for Drugs

```
drug_df = read.csv("../FARS_Data/FARS2018NationalCSV/DRUGS.csv")
person_df = read.csv("../FARS_Data/FARS2018NationalCSV/PERSON.csv")
# Need to join these based on state case and state number. Group DRUGS_DF by person number first. I thi
```

Need to join drug df and person df based on st_case, veh_no, per_no to get the seat pos of each person. That way the drugged drivers can be extracted.

```
reduced_person_df = data.frame(cbind("ST_CASE" = person_df$ST_CASE,
                                     "VEH_NO" = person_df$VEH_NO,
                                     "PER_NO" = person_df$PER_NO,
                                     "SEAT_POS" = person_df$SEAT_POS))
person_drugs_df = merge(x = drug_df, y = reduced_person_df, by = c("VEH_NO", "PER_NO", "ST_CASE"), all =
```

Now, filter based on only drivers and map all drug values to the categories provided by FARS NHTSA

```
person_drugs_df = person_drugs_df[person_drugs_df$SEAT_POS == 11,]
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = 1, to = 2)
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = c(0, 997, 95, 999), to = rep(1, 4))
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(100, 295, by = 1), to = rep(3,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(300, 399, by = 1), to = rep(4,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(400, 495, by = 1), to = rep(5,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(500, 595, by = 1), to = rep(6,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(600, 695, by = 1), to = rep(7,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(700, 795, by = 1), to = rep(8,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(800, 895, by = 1), to = rep(9,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = seq(900, 995, by = 1), to = rep(10,
person_drugs_df$DRUGRES = mapvalues(person_drugs_df$DRUGRES, from = c(996, 998), to = rep(11, 2))
```

```
drug_summary = data.frame(matrix(nrow = 11, ncol = 3))
colnames(drug_summary) = c("Drug Group", "Number of Drugged Drivers", "Percentage of Drugged Drivers")

drug_summary[,1] = c("Unknown", "No Drugs Found", "Narcotic","Depressant","Stimulant","Hallucinogen",
                     "Cannabinoid","Phencyclidine (PCP)", "Anabolic Steroid","Inhalant", "Other"

for (i in seq(1, 11, by = 1)){
  subtotal = person_drugs_df$DRUGRES[person_drugs_df$DRUGRES == i]
  drug_summary[i, 2] = length(subtotal)
  drug_summary[i, 3] = round(length(subtotal) * 100 / 51872, digits = 2)
}
```

Case 4: Driver Distractions

Data is in DISTRACT.csv. Codes are below.

Distractions

Codes	Attributes
00	Not Distracted
03	By Other Occupant(s)
04	By a Moving Object in Vehicle
05	While Talking or Listening to Cellular Phone
06	While Manipulating Cellular Phone
07	Adjusting Audio or Climate Controls
09	While Using Other Component/Controls Integral to Vehicle
10	While Using or Reaching For Device/Object Brought Into Vehicle
12	Distracted by Outside Person, Object, or Event
13	Eating or Drinking
14	Smoking Related
15	Other Cellular Phone Related
16	No Driver Present/Unknown if Driver Present
17	Distraction/Inattention
18	Distraction/Careless
19	Careless/Inattentive
92	Distraction (Distracted), Details Unknown
93	Inattention (Inattentive), Details Unknown
96	Not Reported
97	Lost in Thought/Day Dreaming
98	Other Distraction
99	Reported as Unknown if Distracted

```
distraction_df = read.csv("../FARS_Data/FARS2018NationalCSV/DISTRACT.csv")
driver_distraction = distraction_df$MDRDSTRD
# Replace 5,6, 15 with 2; 3,4,7,9,10 with 3 etc.
```

```
frequency_distraction = as.data.frame(table(driver_distraction))
colnames(frequency_distraction) = c("Code", "Freq")
frequency_distraction[order(frequency_distraction$Freq, decreasing = TRUE), ]
```

```
##    Code  Freq
## 19   96 24259
## 1     0 17533
## 22   99  7050
## 18   93  1012
## 13   16   342
## 14   17   308
## 17   92   263
## 9    12   187
## 21   98   153
## 2     3   126
## 12   15   125
## 5     6   116
## 4     5   114
## 8    10   102
## 16   19    48
## 10   13    45
## 7     9    33
## 6     7    29
## 20   97    17
## 3     4    12
```

```
## 11   14     9
## 15   18     6
```

I think these can be grouped into the following groups.

Codes	Attributes
00	Not Distracted
05, 06, 15	Cell Phone Distraction
03, 04, 07, 09, 10	In-Car Distraction not cell phone
13, 14	Food / Smoking Related
12, 17, 18, 19, 92, 93, 97, 98	Other Distraction
16, 96, 99	Unknown if Distracted

There are a few different ways to group the data, but some of the commmon distractions I hear about from the media are cell-phone, food/drink, and other in-car distractions.

Some drivers had multiple distractions. Specifically, there were 51872 vehicles involved in crashes and 51889 distractions. Since there are more distractions than vehicles, the percentage of each type will sum to slightly more than 100

```
driver_distraction = distraction_df$MDRDSTRD
driver_distraction = mapvalues(driver_distraction, from = c(0), to = c(1))
driver_distraction = mapvalues(driver_distraction, from = c(5, 6, 15), to = rep(2, 3))
driver_distraction = mapvalues(driver_distraction, from = c(3, 4, 7, 9, 10), to = rep(3, 5))
driver_distraction = mapvalues(driver_distraction, from = c(13, 14), to = rep(4, 2))
driver_distraction = mapvalues(driver_distraction, from = c(12, 17, 18, 19, 92, 93, 97, 98), to = rep(5
driver_distraction = mapvalues(driver_distraction, from = c(16, 96, 99), to = rep(6, 3))

distraction_summary = data.frame(matrix(nrow = 6, ncol = 3))
colnames(distraction_summary) = c("Distraction Group", "Total Number of Vehicles Distracted", "Percenta

distraction_summary[,1] = c("Not Distracted", "Cell Phone Distraction", "Other In-Car Distraction",
                            "Food/Smoking Distraction", "Other Distraction", "Unknown if Distracted")

for (i in seq(1, 6, by = 1)){
  subtotal = driver_distraction[driver_distraction == i]
  distraction_summary[i, 2] = length(subtotal)
  distraction_summary[i, 3] = round(length(subtotal) * 100 / 51872, digits = 2)
}
```

Case 5: Driver's Vision Obstruction

Codes	Attributes
00	No Obstruction Noted
01	Rain, Snow, Fog, Smoke, Sand, Dust
02	Reflected Glare, Bright Sunlight, Headlights
03	Curve, Hill, or Other Roadway Design Feature
04	Building, Billboard, Other Structure
05	Trees, Crops, Vegetation
06	In-Transport Motor Vehicle (including load)
07	Not In-Transport Motor Vehicle (parked/working)
08	Splash or Spray of Passing Vehicle

Codes	Attributes
09	Inadequate Defrost or Defog System
10	Inadequate Vehicle Lighting System
11	Obstruction Interior to the Vehicle
12	External Mirrors
13	Broken or Improperly Cleaned Windshield
14	Obstructing Angles on Vehicle
95	No Driver Present/Unknown if Driver Present
97	Vision Obscured – No Details
98	Other Visual Obstruction
99	Reported as Unknown

```
vision_df = read.csv("../FARS_Data/FARS2018NationalCSV/VISION.csv")
# Think about grouping htese together better
```

```
frequency_vision = as.data.frame(table(vision_df$MVISOBSC))
colnames(frequency_vision) = c("Code", "Freq")
frequency_vision[order(frequency_vision$Freq, decreasing = TRUE), ]
```

```
##    Code  Freq
## 1     0 48264
## 19   99  1726
## 2     1   457
## 16   95   342
## 3     2   256
## 18   98   227
## 7     6   214
## 4     3   137
## 17   97   104
## 8     7    86
## 6     5    68
## 5     4    15
## 15   14    10
## 14   13     9
## 12   11     8
## 10    9     3
## 9     8     2
## 11   10     2
## 13   12     2
```

From the frequency table, we see that the most common values are no obstruction and unknown data. For the codes that correspond to obstructions, weather related obstructions, light-based obstructions, and other motor vehicle obstructions are the most common.

First, let's look at how many vehicles involved in fatal crashes were distracted. Some vehicles had multiple distractions so these need to be merged together.

Note: Easiest way to look at data is to create one row for each vision obstruction and divide the frequency by 51872, which is the total number of drivers. This does lead to the percentages summing to over 100%, but does identify how often each type of obstruction is

```
reduced_vehicle_df = data.frame(cbind("ST_CASE" = vehicle_df$ST_CASE,
                                      "VEH_NO" = vehicle_df$VEH_NO))
vision_vehicle_df = merge(x = vision_df, y = reduced_vehicle_df, by = c("VEH_NO", "ST_CASE"), all = TRU
```

```r
vision_vehicle_df = vision_vehicle_df[vision_vehicle_df$MVISOBSC > 0,]
vision_vehicle_df = vision_vehicle_df[vision_vehicle_df$MVISOBSC != 95,]
vision_vehicle_df = vision_vehicle_df[vision_vehicle_df$MVISOBSC != 99,]

# Total number of vision obstructions after removing 1) no obstruction (0) 2) Unknown Driver (95) and 3
nrow(vision_vehicle_df)
```

```
## [1] 1600
```

There are quite a few vision obstruction types, but relatively few vehicles (i.e. 3%) have any vision obstruction at all. To help identify the major causes of vision obstruction, we group the vision obstruction types as follows:

Codes	Attributes
00	No Obstruction Noted
01	Rain, Snow, Fog, Smoke, Sand, Dust
02	Reflected Glare, Bright Sunlight, Headlights
03	Curve, Hill, or Other Roadway Design Feature
04	Building, Billboard, Other Structure
05	Trees, Crops, Vegetation
06	In-Transport Motor Vehicle (including load)
07	Not In-Transport Motor Vehicle (parked/working)
08	Splash or Spray of Passing Vehicle
09	Inadequate Defrost or Defog System
10	Inadequate Vehicle Lighting System
11	Obstruction Interior to the Vehicle
12	External Mirrors
13	Broken or Improperly Cleaned Windshield
14	Obstructing Angles on Vehicle
95	No Driver Present/Unknown if Driver Present
97	Vision Obscured – No Details
98	Other Visual Obstruction
99	Reported as Unknown

Category	Group
1	No obstruction (00)
2	Weather Related Obstructions (01, 02)
3	Other Exterior Obstructions (03, 04, 05, 06, 07, 08)
4	Vehicle Related Obstructions (09, 10, 11, 12, 13, 14)
5	Other/Unknwon Obstructions (95, 97, 98, 99)

```r
# Weird order to make sure values don't get overwritten
vision_vehicle_df = merge(x = vision_df, y = reduced_vehicle_df, by = c("VEH_NO", "ST_CASE"), all = TRU
vision_vehicle_df$MVISOBSC = mapvalues(vision_vehicle_df$MVISOBSC, from = c(seq(3,8,by=1)), to = rep(3,
vision_vehicle_df$MVISOBSC = mapvalues(vision_vehicle_df$MVISOBSC, from = c(95,97,98,99), to = rep(5,4)
vision_vehicle_df$MVISOBSC = mapvalues(vision_vehicle_df$MVISOBSC, from = c(seq(9,14,by=1)), to = rep(4
vision_vehicle_df$MVISOBSC = mapvalues(vision_vehicle_df$MVISOBSC, from = c(1,2), to = rep(2,2))
vision_vehicle_df$MVISOBSC = mapvalues(vision_vehicle_df$MVISOBSC, from = c(0), to = c(1))
```

```r
vision_summary = data.frame(matrix(nrow = 5, ncol = 3))
colnames(vision_summary) = c("Vision Obstruction", "Number of Vision Obstructions",
```

```
                              "Percentage of Vision Obstructions")

vision_summary[,1] = c("No Obstruction", "Weather Related Obstruction", "Other Exterior Obstruction",
                       "Vehicle Related Obstrcution","Other/Unknown Obstruction")

for (i in seq(1, 5, by = 1)){
  subtotal = vision_vehicle_df$MVISOBSC[vision_vehicle_df$MVISOBSC == i]
  vision_summary[i, 2] = length(subtotal)
  vision_summary[i, 3] = round(length(subtotal) * 100 / 51872, digits = 2)
}
```

```
# First find the rows with
reduced_vision_df = data.frame(cbind("ST_CASE" = vision_df$ST_CASE,
                                     "STATE" = vision_df$STATE,
                                     "VEH_NO" = vision_df$VEH_NO))
print(sum(as.numeric(duplicated(reduced_vision_df))))
```

```
## [1] 60
```

```
print(which(duplicated(reduced_vision_df)))
```

```
##  [1]   973  2273  3378  3379  4822  5303  5942  6146  8054  8056  9053 15867
## [13] 18150 19562 21527 21618 21675 26040 28880 29192 29194 29257 30214 30683
## [25] 36093 38437 38438 39451 39452 39453 39606 39608 39664 39680 39765 39767
## [37] 39779 39781 45416 46612 46980 47035 48583 50989 50990 51015 51016 51018
## [49] 51019 51026 51033 51046 51114 51392 51550 51604 51684 51688 51705 51786
```

```
print(vision_df$MVISOBSC[duplicated(reduced_vision_df)])
```

```
##  [1]  3 13  2 98 98 13  5 13  7  7  5  2 98  7 98  2 98  5  7  5  5  8  7  2  2
## [26] 11 98  2  4  5  5  5  6 14  3  3  5  5  6 97 98 98 11 10 97 14 98 14 98  3
## [51]  7  3 11  5 14  3 98  2  8 98
```

```
obs_vehicle = vision_df[vision_df$MVISOBSC > 0,]
obs_vehicle = obs_vehicle[obs_vehicle$MVISOBSC != 95,]
obs_vehicle = obs_vehicle[obs_vehicle$MVISOBSC != 99,]
nrow(obs_vehicle)
```

```
## [1] 1600
```

```
nrow(obs_vehicle) / nrow(vision_df)
```

```
## [1] 0.03080952
```

Case 6: Summarizing Figures

```
# Get percentage of accidents where each factor was involved

# Speeding
speed_rel <- vehicle_df$SPEEDREL
speed_rel <- mapvalues(speed_rel, from = c(0, 2, 3, 4, 5, 8, 9), to = c(1, 2, 2, 2, 2, 0, 0))
speed_df = data.frame(cbind("Speed_Related" = speed_rel, "State_Case" = vehicle_df$ST_CASE))
types_of_speeding = aggregate(x = speed_df[c("Speed_Related")], by = speed_df[c("State_Case")], FUN = ma

# Speeding Results
unknown_speeding = length(types_of_speeding[types_of_speeding == 0])
```

```r
known_no_speeding = length(types_of_speeding[types_of_speeding == 1])
known_speeding = length(types_of_speeding[types_of_speeding == 2])

# Alcohol
known_drinking = length(accident_df$DRUNK_DR[accident_df$DRUNK_DR > 0])
known_no_drinking = length(accident_df$DRUNK_DR[accident_df$DRUNK_DR == 0])

# Drugged Driving
drug_use = as.data.frame(person_drugs_df$DRUGRES)
colnames(drug_use) = c("Drugs_Found")
drug_use$Drugs_Found <- mapvalues(drug_use$Drugs_Found, from = c(seq(1,11,1)), to = c(22,23,rep(24,9)))
drug_df = data.frame(cbind("Drug_Related" = drug_use$Drugs_Found, "State_Case" = person_drugs_df$ST_CASE
types_of_drugs = aggregate(x = drug_df[c("Drug_Related")], by = drug_df[c("State_Case")], FUN = max)


# Drug Results
unknown_drugs = length(types_of_drugs$Drug_Related[types_of_drugs$Drug_Related == 22])
known_no_drugs = length(types_of_drugs$Drug_Related[types_of_drugs$Drug_Related == 23])
known_drugs = length(types_of_drugs$Drug_Related[types_of_drugs$Drug_Related == 24])
# There are 86 cases where drugs are not reported these are added to the unknown
unknown_drugs = unknown_drugs + 86

# Distraction
distraction_use = as.data.frame(driver_distraction)
colnames(distraction_use) = c("Distractions")
distraction_use$Distractions = mapvalues(distraction_use$Distractions, from = c(seq(1,6,1)), to = c(21,
distraction_summary_df = data.frame(cbind("Distraction" = distraction_use$Distractions, "State_Case" = c
types_of_distractions = aggregate(x = distraction_summary_df[c("Distraction")], by = distraction_summary

# Distraction Results
unknown_distraction = length(types_of_distractions$Distraction[types_of_distractions$Distraction == 20])
known_no_distraction = length(types_of_distractions$Distraction[types_of_distractions$Distraction == 21]
known_distraction = length(types_of_distractions$Distraction[types_of_distractions$Distraction == 22])


# Obstruction Results
obstruction_use = as.data.frame(vision_vehicle_df$MVISOBSC)
colnames(obstruction_use) = c("Obstruction")
obstruction_use$Obstruction = mapvalues(obstruction_use$Obstruction, from = c(seq(1,5,1)), to = c(21, re
obstruction_summary_df = data.frame(cbind("Obstruction" = obstruction_use$Obstruction, "State_Case" = vi
types_of_obstructions = aggregate(x = obstruction_summary_df[c("Obstruction")], by = obstruction_summary

# Distraction Results
unknown_obstruction = length(types_of_obstructions$Obstruction[types_of_obstructions$Obstruction == 20])
known_no_obstruction = length(types_of_obstructions$Obstruction[types_of_obstructions$Obstruction == 21]
known_obstruction = length(types_of_obstructions$Obstruction[types_of_obstructions$Obstruction == 22])
```

```r
# Now turn the above into a table and then a stacked barplot
speeding_data = c(known_speeding, known_no_speeding, unknown_speeding)
drinking_data = c(known_drinking, known_no_drinking, 0)
drug_data = c(known_drugs, known_no_drugs, unknown_drugs)
distraction_data = c(known_distraction, known_no_distraction, unknown_distraction)
obstruction_data = c(known_obstruction, known_no_obstruction, unknown_obstruction)
```

```r
test_four = data.frame(rbind(speeding_data, drinking_data, drug_data, distraction_data, obstruction_data

summary_info_crashes = c(0,0,0)
named_columns = c("Speeding", "Drinking", "Drugs", "Distraction", "Obstruction")
named_columns = c(seq(1,5,1))
named_rows = c("Related", "Not Related", "Unknown")

for (i in 1:nrow(test_four)){
  for (j in 1:ncol(test_four)){
    summary_info_crashes = rbind(summary_info_crashes, c(named_columns[i], named_rows[j], test_four[i,j]
  }
}

summary_info_crashes = data.frame(summary_info_crashes)
summary_info_crashes$X3 = as.numeric(as.matrix(summary_info_crashes$X3))
summary_info_crashes = summary_info_crashes[-1,]
summary_info_crashes = summary_info_crashes[-6,]
row.names(summary_info_crashes) <- 1:nrow(summary_info_crashes)

summary_plot = ggplot(summary_info_crashes, aes(fill=factor(X2, levels = c("Unknown", "Not Related", "Re
  geom_bar(position="stack", stat="identity") +
  xlab("Causes of Fatal Accidents") +
  # names(c("Sunday", "Monday", "Tuesday", "Wed.", "Thursday", "Friday", "Saturday")) +
  ylab("Percentage of Accidents") +
  #ggtitle("Percentage of Accidents Where Each Cause was Related") +
  scale_x_discrete(breaks=1:5, labels=c("Speeding", "Drinking", "Drugs", "Distractions", "Obstructions"
  theme(legend.position="bottom", legend.direction="horizontal", legend.title = element_blank()) +
  guides(fill = guide_legend(reverse = TRUE)) +
  theme(plot.title = element_text(size = 16, hjust = 0.5),
        axis.text = element_text(size = 12),
        axis.title = element_text(size = 14, face="bold"),
        legend.text=element_text(size=12))

png("Q2_Results_Summary.png")
print(summary_plot)
dev.off()
```

```
## pdf
##   2
```