Car Accidents in the U.S

Uncovering Trends and Potential Risk Factors

Alejandro Mora & Pedro Pereira

Introduction

Car accidents are common hazards throughout the whole world. There are vast land masses with roads, and in turn, there are many cars with drivers, taking the risk of driving. Since each car or land vehicle is operated by an individual that has different skill sets in driving, there is that possibility of unawareness of driving safety, and awareness of surrounding hazards. Each situation regarding car accidents is different, the one thing that can be done, is mitigation of these accidents.

The more accidents that can be prevented, the more lives that can be saved as well as property damage. A thing to look at is how are these accidents connected and what correlations can be found. These can help with starting on what to include in driving safety lessons and help with building/installing safer road signage if needed.

According to the National Highway Traffic Safety Administration, “Fifty-seven percent of fatal crashes in 2017 involved only one vehicle” (nhtsa.gov). Based off this statistic from what was mentioned before, most fatal accidents aren’t caused by one car to another. There might be an underlying environment factors regarding the crashes. This being weather, road conditions and maybe time of day factors. They can be highly correlated, however there is no guarantee that they are one hundred percent causes for accidents or severities, but they do help. The possibility of outlying factors can be possible too, but it is hard to judge which one could the only underlying cause, and it could possibly be a combination. All factors must be looked at in order to get a good picture of what is happening.

Looking further into the issue regarding whether looking at accidents due to causes of the environment around them, most underlying factors can come from within the car. Harry Brown from the Brown Firm states there are leading causes for car crashes that aren’t looked at a lot (Brown 2018). These include, “Distracted driving, Fatigue, Driving while intoxicated, and aggressive driving” (Brown 2018). People do spend a lot of their time driving, not paying attention to the road whether it’s purposefully or not. Many things come up along your starting point to your destination. From being too tired, too inpatient, or not realizing how dangerous their situation is at the time, anything can be a factor. More investigations need to go into factors like these, and into things surrounding the vehicle to raise awareness.

Analysis

Tools

Due to the nature of the data the Python and R programming languages were used to analyze the data. Below is the sample code of all necessary libraries for conducting the analysis.

Python

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report
from sklearn import metrics
from sklearn.naive_bayes import GaussianNB
from sklearn import tree
from sklearn.tree import plot_tree
```

R

```r
library(tidyverse)
library(ggmap)
library(magrittr)
library(zoo)
library(zipcode)
library(viridis)
library(rgdal)
library(arules)
library(arulesViz)
library(kernlab)
library(e1071)
library(caret)
library(yardstick)
library(grid)
library(gridExtra)
library(FactoMineR)
```

Methodology

Statistical exploratory analysis and visual exploratory analysis techniques were used to identify high-level insights about the data. Unsupervised machine learning methods were used to derive more granular insights and correlations. Supervised machine learning methods were use generate in-depth insights and predictions.

About the Data

The data was sourced from Kaggle.com. It contains records of accidents in the United States. These records were obtained from the MapQuest and Bing API, from February 2016 to June 2020. The dataset contains approximately 3.5 million rows, with 49 variables. The image below shows a sample of the first 5 rows of the dataset.

	ID	Source	TMC	Severity	Start_Time	End_Time	Start_Lat	Start_Lng	End_Lat	End_Lng	...	Roundabout	Station	Stop	Traffic_Calming	Traffic_Signal	Turning_Loop
0	A-1	MapQuest	201.0	3	2016-02-08 05:46:00	2016-02-08 11:00:00	39.865147	-84.058723	NaN	NaN	...	False	False	False	False	False	False
1	A-2	MapQuest	201.0	2	2016-02-08 06:07:59	2016-02-08 06:37:59	39.928059	-82.831184	NaN	NaN	...	False	False	False	False	False	False
2	A-3	MapQuest	201.0	2	2016-02-08 06:49:27	2016-02-08 07:19:27	39.063148	-84.032608	NaN	NaN	...	False	False	False	False	True	False
3	A-4	MapQuest	201.0	3	2016-02-08 07:23:34	2016-02-08 07:53:34	39.747753	-84.205582	NaN	NaN	...	False	False	False	False	False	False
4	A-5	MapQuest	201.0	2	2016-02-08 07:39:07	2016-02-08 08:09:07	39.627781	-84.188354	NaN	NaN	...	False	False	False	False	True	False

5 rows × 49 columns

The variables types were diverse. There were 13 Boolean variables, 14 float, 1 integer, and 21 as factor. Based on the variable descriptions, and the purpose of this analysis 12 of the variables were dropped. Reducing the number of variables to 37. The code sample below shows the variables that were dropped.

```
delcols = ['ID','TMC','Source', 'End_Lat', 'End_Lng', 'Number', 'Street', 'Airport_Code', 'Weather_Timestamp', 'Civil_Twilight',
           'Nautical_Twilight', 'Astronomical_Twilight']
df.drop(delcols, axis=1, inplace=True)
```

Out the remaining variables 14 had null values, as shown below.

```
Severity                0
Start_Time              0
End_Time                0
Start_Lat               0
Start_Lng               0
Distance(mi)            0
Description             1
Side                    0
City                  112
County                  0
State                   0
Zipcode              1069
Country                 0
Timezone             3880
Temperature(F)      65732
Wind_Chill(F)     1868249
Humidity(%)         69687
Pressure(in)        55882
Visibility(mi)      75856
Wind_Direction      58874
Wind_Speed(mph)    454609
Precipitation(in) 2025874
Weather_Condition   76138
Amenity                 0
Bump                    0
Crossing                0
Give_Way                0
Junction                0
No_Exit                 0
Railway                 0
Roundabout              0
Station                 0
Stop                    0
Traffic_Calming         0
Traffic_Signal          0
Turning_Loop            0
Sunrise_Sunset        115
```

	Severity	Start_Lat	Start_Lng	Distance(mi)	Temperature(F)	Wind_Chill(F)	Humidity(%)	Pressure(in)	Visibility(mi)	Wind_Speed(mph)	Precipitation(in)
count	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.447885e+06	1.645368e+06	3.443930e+06	3.457735e+06	3.437761e+06	3.059008e+06	1.487743e+06
mean	2.339929e+00	3.654195e+01	-9.579151e+01	2.816167e-01	6.193512e+01	5.355730e+01	6.511427e+01	2.974463e+01	9.122644e+00	8.219025e+00	1.598256e-02
std	5.521935e-01	4.883520e+00	1.736877e+01	1.550134e+00	1.862106e+01	2.377334e+01	2.275558e+01	8.319758e-01	2.885879e+00	5.262847e+00	1.928262e-01
min	1.000000e+00	2.455527e+01	-1.246238e+02	0.000000e+00	-8.900000e+01	-8.900000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	2.000000e+00	3.363784e+01	-1.174418e+02	0.000000e+00	5.000000e+01	3.570000e+01	4.800000e+01	2.973000e+01	1.000000e+01	5.000000e+00	0.000000e+00
50%	2.000000e+00	3.591687e+01	-9.102601e+01	0.000000e+00	6.400000e+01	5.700000e+01	6.700000e+01	2.995000e+01	1.000000e+01	7.000000e+00	0.000000e+00
75%	3.000000e+00	4.032217e+01	-8.093299e+01	1.000000e-02	7.590000e+01	7.200000e+01	8.400000e+01	3.009000e+01	1.000000e+01	1.150000e+01	0.000000e+00
max	4.000000e+00	4.900220e+01	-6.711317e+01	3.336300e+02	1.706000e+02	1.150000e+02	1.000000e+02	5.774000e+01	1.400000e+02	9.840000e+02	2.500000e+01

The missing values were replaced with median value of the column, as the mean is susceptible to outliers. This is especially true for the weather variables, as there is significant weather variance in the U.S. The result of replacing the missing values with the median is below. Variables that were not numeric and had missing values were omitted as they did not represent a significant portion of the dataset.

```
Severity                0
Start_Time              0
End_Time                0
Start_Lat               0
Start_Lng               0
Distance(mi)            0
Description             1
Side                    0
City                  112
County                  0
State                   0
Zipcode              1069
Country                 0
Timezone             3880
Temperature(F)          0
Wind_Chill(F)           0
Humidity(%)             0
Pressure(in)            0
Visibility(mi)          0
Wind_Direction      58874
Wind_Speed(mph)         0
Precipitation(in)       0
Weather_Condition   76138
Amenity                 0
Bump                    0
Crossing                0
Give_Way                0
Junction                0
No_Exit                 0
Railway                 0
Roundabout              0
Station                 0
Stop                    0
Traffic_Calming         0
Traffic_Signal          0
Turning_Loop            0
Sunrise_Sunset        115
```

There were no apparent outliers in the data that required additional investigation. Any outliers in the numerical variables were due to normal weather variance in the 4-year period covered by the dataset. The table below shows the statistical summary of the data after replacing the missing values.

	Severity	Start_Lat	Start_Lng	Distance(mi)	Temperature(F)	Wind_Chill(F)	Humidity(%)	Pressure(in)	Visibility(mi)	Wind_Speed(mph)	Precipitation(in)
count	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06
mean	2.339929e+00	3.654195e+01	-9.579151e+01	2.816167e-01	6.197375e+01	5.538784e+01	6.515167e+01	2.974790e+01	9.141586e+00	8.061301e+00	6.767368e-03
std	5.521935e-01	4.883520e+00	1.736877e+01	1.550134e+00	1.844818e+01	1.635883e+01	2.253032e+01	8.257331e-01	2.857404e+00	4.927607e+00	1.257220e-01
min	1.000000e+00	2.455527e+01	-1.246238e+02	0.000000e+00	-8.900000e+01	-8.900000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	2.000000e+00	3.363784e+01	-1.174418e+02	0.000000e+00	5.000000e+01	5.700000e+01	4.900000e+01	2.974000e+01	1.000000e+01	5.000000e+00	0.000000e+00
50%	2.000000e+00	3.591687e+01	-9.102601e+01	0.000000e+00	6.400000e+01	5.700000e+01	6.700000e+01	2.995000e+01	1.000000e+01	7.000000e+00	0.000000e+00
75%	3.000000e+00	4.032217e+01	-8.093299e+01	1.000000e-02	7.570000e+01	5.700000e+01	8.400000e+01	3.009000e+01	1.000000e+01	1.040000e+01	0.000000e+00
max	4.000000e+00	4.900220e+01	-6.711317e+01	3.336300e+02	1.706000e+02	1.150000e+02	1.000000e+02	5.774000e+01	1.400000e+02	9.840000e+02	2.500000e+01

After correcting missing values, and checking for outliers, some additional variable modification and creation was required. First the Boolean values representing the physical environment variables were converted to binary in order to conduct arithmetic operations on them and for easier interpretation by machine learning models. Additionally, a variable calculating the duration of the accident was created by subtracting the end time from the start time. This value is the total time it took for the accident to be cleared from the highway in hours. To account for outliers, any value above 5 was replaced with the median, the same was done for any negative values. The code sample below displays the method used to accomplish this.

```
#Replacing boolean values
df.replace({True:1,False:0}, inplace=True)
#Converting start and end time to datetime data type
df.Start_Time =  pd.to_datetime(df.Start_Time)
df.End_Time =  pd.to_datetime(df.End_Time)
#Converting severity to factor
df.Severity = df.Severity.astype(object)
#Calculating duration in hours
df['Duration'] = df.End_Time - df.Start_Time
df['Duration'] = df['Duration'] / np.timedelta64(1, 'h')
#Replacing outliers
df.loc[df['Duration'] > 5, 'Duration'] = 0.74
df.loc[df['Duration'] < 0, 'Duration'] = 0.74
```

Exploring the Data

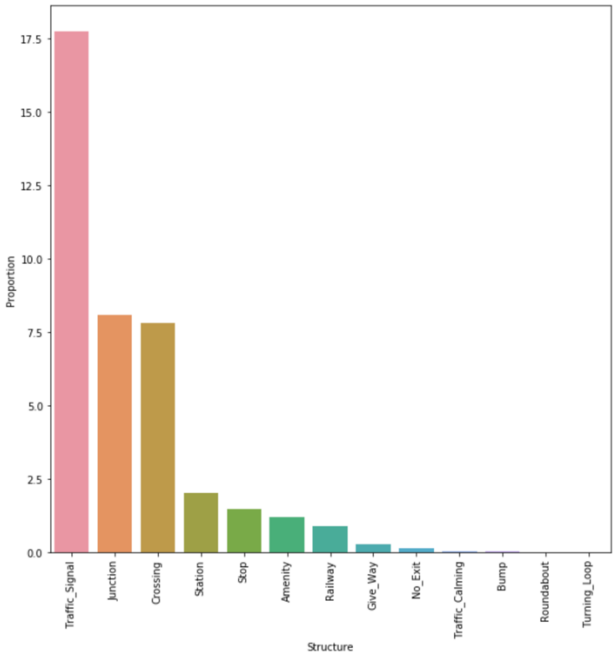
The accident severity variable describes the seriousness of an accident on a scale of 1 – 4, with 1 being the least severe. The majority of observations were categorized as severity 2.

```
df.Severity.value_counts(normalize = True)
```

```
2    0.675432
3    0.284298
4    0.031967
1    0.008303
```

There were 12 binary variables that recorded whether the corresponding physical structure was present. As shown by the table and plot below, traffic signal was the most common physical environment variable.

Structure	Frequency	Proportion
Traffic_Signal	623623	17.748747
Junction	284449	8.095618
Crossing	274526	7.813202
Station	70321	2.001385
Stop	51976	1.479273
Amenity	42082	1.197683
Railway	31175	0.887262
Give_Way	9564	0.272198
No_Exit	4384	0.124772
Traffic_Calming	1401	0.039873
Bump	606	0.017247
Roundabout	184	0.005237
Turning_Loop	0	0.000000



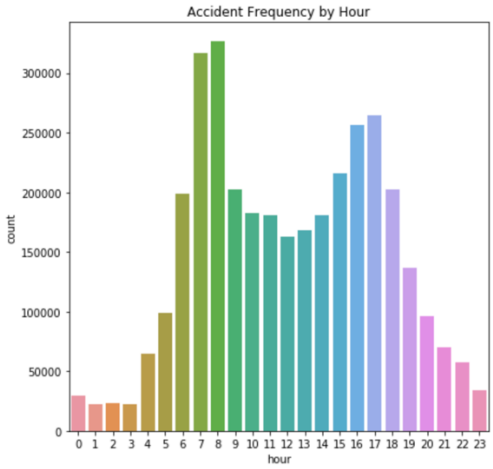
The remaining variables were factors. The table below displays the number of unique values for each variable. There are over 400000 zip codes in the dataset, and over 11000 cities.

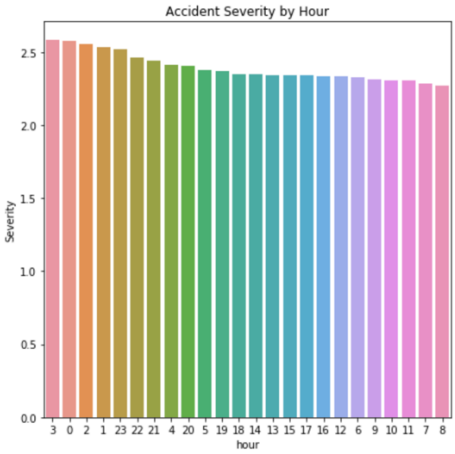
feature	nunique
Country	1
Side	3
Timezone	4
State	49
Weather_Condition	127
County	1724
City	11895
Zipcode	418780

There was significant variability in the weather variables as shown in the table below. This is likely due to the different weather types in the U.S. However, data collection errors can also be responsible for the variability.

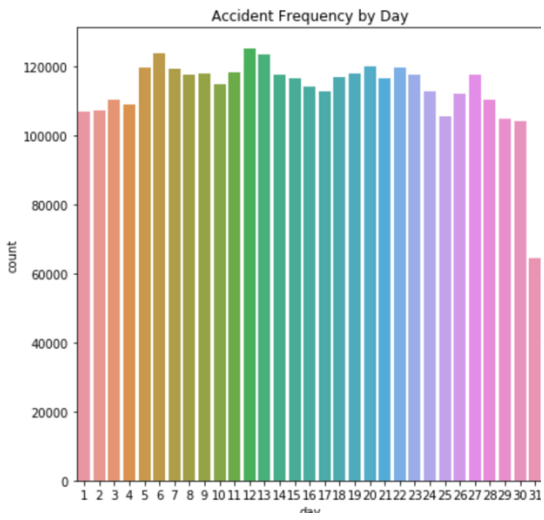
	Temperature(F)	Wind_Chill(F)	Humidity(%)	Pressure(in)	Visibility(mi)	Wind_Speed(mph)	Precipitation(in)
count	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06	3.513617e+06
mean	6.197375e+01	5.538784e+01	6.515167e+01	2.974790e+01	9.141586e+00	8.061301e+00	6.767368e-03
std	1.844818e+01	1.635883e+01	2.253032e+01	8.257331e-01	2.857404e+00	4.927607e+00	1.257220e-01
min	-8.900000e+01	-8.900000e+01	1.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00	0.000000e+00
25%	5.000000e+01	5.700000e+01	4.900000e+01	2.974000e+01	1.000000e+01	5.000000e+00	0.000000e+00
50%	6.400000e+01	5.700000e+01	6.700000e+01	2.995000e+01	1.000000e+01	7.000000e+00	0.000000e+00
75%	7.570000e+01	5.700000e+01	8.400000e+01	3.009000e+01	1.000000e+01	1.040000e+01	0.000000e+00
max	1.706000e+02	1.150000e+02	1.000000e+02	5.774000e+01	1.400000e+02	9.840000e+02	2.500000e+01

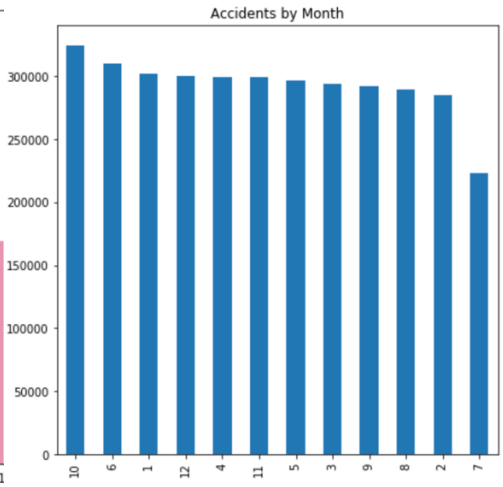
Exploring Accident Seasonality





Most accidents occur during commuting times, 7:00am – 8:00am and 4:00pm – 5:00pm. However, there is considerably more accidents in the morning commute than an in the afternoon commute. Accident severity is higher in the late night and early mornings. From approximately 9:00pm until 3:00am the median accident severity is higher. This is potentially due to fatigued drivers and increased chances of drunk drivers on the road. There was no apparent trend on accident frequency month-to-month or day-to-day. The graph below shows an apparent drop in accidents on the 31st. However, this is due to all months not having a 31st day. There is a notable drop in accidents in July, which was not expected as people typically take time off during the summer. This would likely lead to more cars on the road and therefore more accidents.





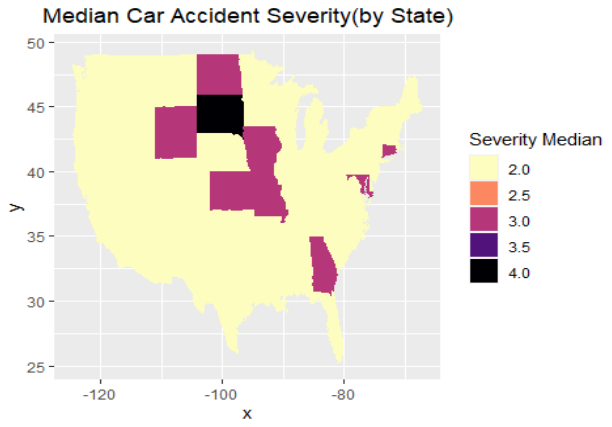
Exploring Accident Geography

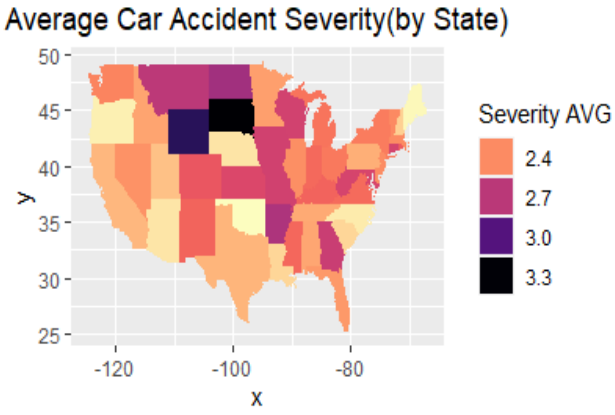
The dataset was heavily biased towards the state of California, making up nearly a quarter of the total observations. The top 5 states with accidents were California, Texas, Florida, South Carolina and North Carolina. 3 out of these 5 states are populous, so it was expected for them to have a

high number of accidents. Adding population data from the Census Bureau, the proportion of accidents to population was calculated for each state. The top 5 states based on this new metric was very different. As shown on the table below, these states had very small sample sizes.

State	Start_Lat	Pop	prop
SD	61	87480525	143411.0
ND	44	7573795	17213.0
WV	2381	181111625	7607.0
MS	6585	298340425	4531.0
KS	7939	291105875	3667.0

Exploring the median and mean accident severity in each state, showed a concentration of higher values in the Midwest. States like the Dakotas, Wyoming, and Iowa all had accident severities of 3 or above on average.





Generating Correlations from Physical Environment Variables

Association Rules Mining

Association Rule Mining can help with correlations between multiple variables. This model uses distance relationships between the variables to see which ones are the closest in relationship. It calculates support, confidence, and lift to see what the most significant rules are. The support means how much data is taken by making the rules from the data which is important when looking at the size of data that is present.

```
#Create a Data frame for Association Rule Mining using physical environments an
d the accident ID itself
AccEnvData <- data.frame(Accidents$ID, phys_env)
colnames(AccEnvData)[1] <- "ID"
#Remove all rows if all environments are FALSE points in data
AccEnvData <- AccEnvData[!(AccEnvData$Traffic_Signal=="False" & AccEnvData$Traf
```

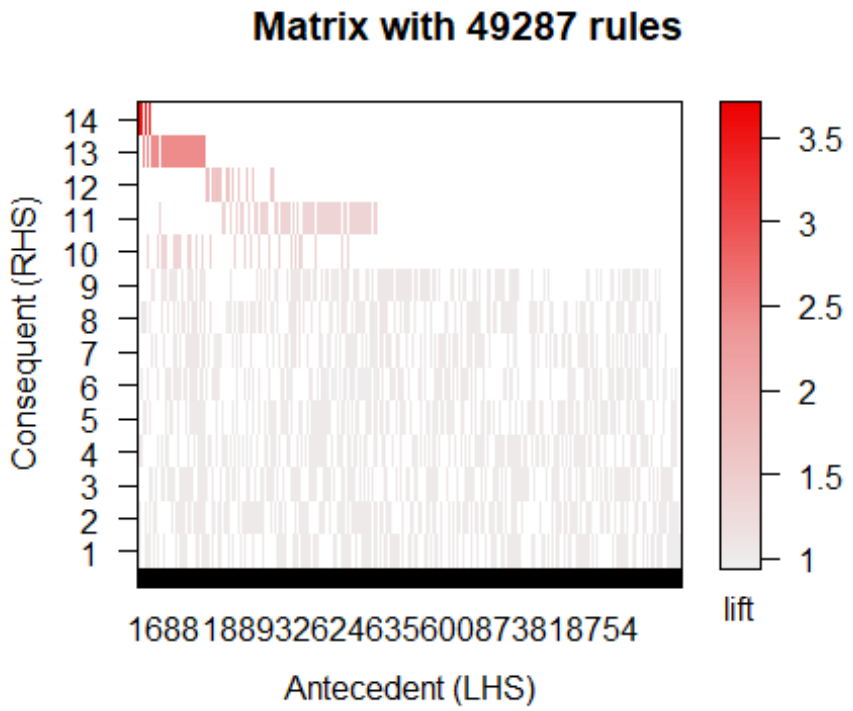
```
fic_Calming== "False" & AccEnvData$Stop=="False" & AccEnvData$Station == "False
" & AccEnvData$Roundabout== "False" & AccEnvData$Railway== "False" & AccEnvData
$No_Exit== "False" & phys_env$Junction== "False" & AccEnvData$Give_Way== "False
" & AccEnvData$Crossing== "False" & AccEnvData$Bump== "False" & AccEnvData$Amen
ity== "False"),]
```

Data Must be discretized to use Association Rule Mining

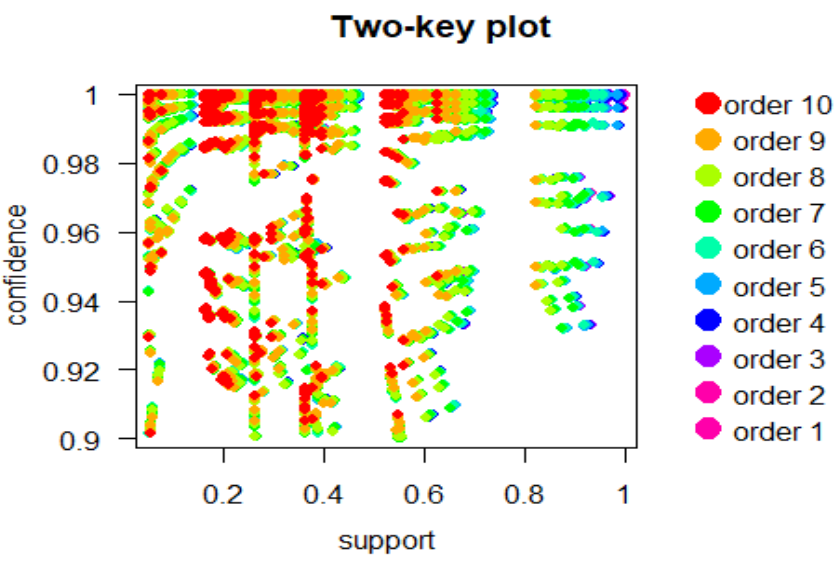
```
#Convert all physical environment factors to binary
AccEnvData[2:13] <- lapply(AccEnvData[2:13],factor)
str(AccEnvData)
```

Create the Association Rules using the Apriori algorithm with 0.05 support and .90 confidence as parameters. These parameters are set to these values based on the size of the data. The more observations that are present, the more memory the rules will take up on the working station. The support will have to be lowered to account for the large data set.

```
Accident_rules1 <- apriori(AccEnvData, parameter = list(support = 0.05, confide
nce = 0.90))

#Sort the rules based on support
Accident_rules1 <- sort(Accident_rules1, by = "support")

## set of 49287 rules

#plot Rules using a matrix format to visualize
plot(Accident_rules1,method="matrix")
```



```
#A plot that show the distribution of rules based on support and confidence
plot(Accident_rules1,method="two-key plot", jitter=0)
```



The rules can be viewed based on highest lift counts to see what rules show the most significance.

```
inspect(head(sort(Accident_rules1, by = "lift"), 3))
```

```
##     lhs                          rhs             support confidence coverage
lift  count
## [1] {Traffic_Signal=False,
##      Traffic_Calming=False,
##      Stop=False,
##      Station=False,
##      Railway=False,
##      No_Exit=False,
##      Give_Way=False,
##      Crossing=False,
##      Amenity=False}         => {Junction=True}  0.2626     0.9999   0.2626
3.694 275898
## [2] {Traffic_Signal=False,
##      Stop=False,
##      Station=False,
##      Railway=False,
##      No_Exit=False,
##      Give_Way=False,
##      Crossing=False,
##      Bump=False,
##      Amenity=False}         => {Junction=True}  0.2626     0.9988   0.2629
3.690 275935
## [3] {Traffic_Signal=False,
##      Stop=False,
##      Station=False,
##      Roundabout=False,
```

```
##      Railway=False,
##      No_Exit=False,
##      Give_Way=False,
##      Crossing=False,
##      Amenity=False}         => {Junction=True}  0.2625     0.9978   0.2631
3.686 275863
```

Association Rules Mining Results

Looking at the Association Rules, there were 49,287 rules created based on a support level of 0.30. The highest confidence level that was able to be created was 0.9998695 and a very high lift level of 3.693811. The mixed variables provided top rules having correlations of junctions being present and also having a false value for traffic signals. Other variables present that show high correlation are false values for stop signs and traffic calming being false. This shows within accidents, junctions show the highest areas for accidents with having no presence of traffic signals or stop signs and heavy traffic

Predicting Accident Severity

Naïve Bayes

Regarding Naive Bayes Models, the predictions are based on one variable having effect on another variable. In this case regarding physical environment factors, we can compare severity of accidents and the effect the environment factors have on the severity.

Creating a data frame for Naive Bayes

```
AccEnvData <- data.frame(Accidents$ID, phys_env)
colnames(AccEnvData)[1] <- "ID"
#Add Severity, take out ID
AccEnvData <- cbind(AccEnvData, Accidents$Severity)
names(AccEnvData)[14] <- "Severity"
AccEnvData <- AccEnvData[,-1]
#Remove all rows if all environments are FALSE points in data
#Convert all variables to factors
AccEnvData[1:13] <- lapply(AccEnvData[1:13],factor)
```

Unlike Association Rule mining that can use large data sets, with the Naive Bayes model being used and the size of the data, the must be broken down into a sample to save memory space.

```
#split the whole set and take 4% of the data
SampleSplit <- sample(nrow(AccEnvData), nrow(AccEnvData)*.04)
SampleDF <- AccEnvData[SampleSplit,]
```

There is only 4% of the data being used for this model. In order to make sure the model can receive its best results and reduce biased results, k-fold cross-validation can be used. A 3-fold cross-validation will be used for this model to reduce the possibilities of biased results. #Holdout 3-fold cross validation

```
#Create a hold-out variable to split the data
HoldOutData <- split(sample(1:nrow(SampleDF)), 1:3)

#Create training and test data sets for models
##3 sets must be created as there are 3-fold indexes of the data
##Data can all be put together after all sets are run by the models.
Acc_Train1 <- SampleDF[-HoldOutData[[1]],]
Acc_Test1 <- SampleDF[HoldOutData[[1]],]
.
. Down to three data sets
#Remove test data sets' "ID"
Nolab_Test1 <- Acc_Test1[-c(13)]
Nolab_Test2 <- Acc_Test2[-c(13)]
Nolab_Test3 <- Acc_Test3[-c(13)]
#Create a label variable
Labels1 <- Acc_Test1$Severity
Labels2 <- Acc_Test2$Severity
Labels3 <- Acc_Test3$Severity
```

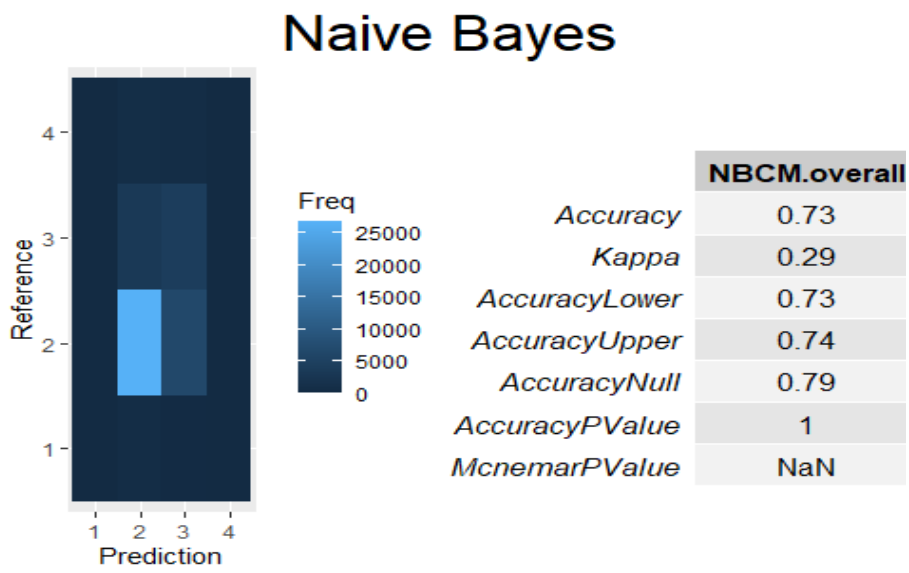
Start by creating the first model based on the first training data set created and creating the next two on their respective training and test data sets.

```
#Create a Naive Bayes Model using Digit_Train1
NBAcc1 <- naiveBayes(Acc_Train1$Severity~.,data = Acc_Train1)
#Create a Naive Bayes Model using Digit_Train2
NBAcc2 <- naiveBayes(Acc_Train2$Severity~ .,data = Acc_Train2)
#Create a Naive Bayes Model using Digit_Train3
NBAcc3 <- naiveBayes(Acc_Train3$Severity~ .,data = Acc_Train3)
#Use the first model to test on Nolab_Test1
NB_Predict1 <- predict(NBAcc1, Nolab_Test1)
#Use the model to test on Digit_Test2
NB_Predict2 <- predict(NBAcc2, Nolab_Test2)
#Use the model to test on Digit_Test3
NB_Predict3 <- predict(NBAcc3, Nolab_Test3)
```

The models can be aggregated to get the final results of the cross-validations.

```
#Combine all Naive Bayes Predictions and Labels
NBResults <- c(NB_Predict1, NB_Predict2, NB_Predict3)
NBLabels <- c(Labels1, Labels2, Labels3)
```

```
#Insert Results into a confusion matrix
NBCM <- confusionMatrix(as.factor(NBResults), as.factor(NBLabels))

#Create data frames for visualization
NBtable <- data.frame(NBCM$table)
statsNB1 <- data.frame(NBCM$overall)
statsNB1$NBCM.overall <- round(statsNB1$NBCM.overall, 2)
#Create heatmap for visualization
NBPlot <- ggplot(NBtable, aes(x = Prediction, y = Reference, fill = Freq)) + ge
om_tile()
#Create a statistic legend for Naive Bayes
stats1 <- tableGrob(statsNB1)
#Show original Confusion Matrix
table(NBResults,NBLabels)

##          NBLabels
## NBResults     1     2     3     4
##         2   526 26688  3122   595
##         3    88  6384  4084   546

#Input visuals and statistics legend together
grid.arrange(NBPlot, stats1, nrow = 1, ncol = 2, top = textGrob("Naive Bayes",
gp=gpar(fontsize=25, font=1)))
```



	NBCM.overall
Accuracy	0.73
Kappa	0.29
AccuracyLower	0.73
AccuracyUpper	0.74
AccuracyNull	0.79
AccuracyPValue	1
McnemarPValue	NaN

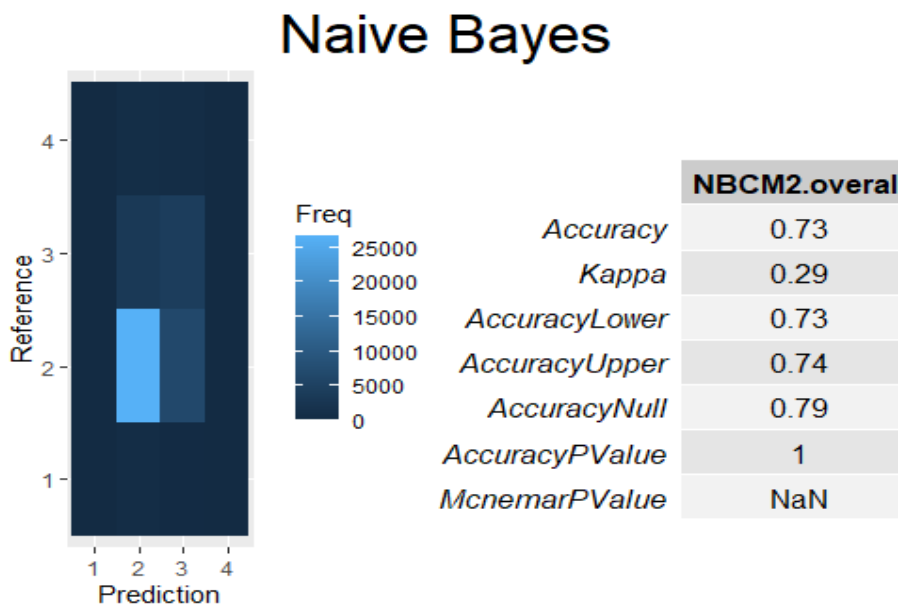
From the association rules, seeing that Junctions and Traffic_Signal are high correlations to accidents, there can be a more specific model created. Severity will be predicted just on those two variables instead of all the physical environment variables.

```
#Create a Naive Bayes Model using Digit_Train1
NBAcc4 <- naiveBayes(Acc_Train1$Severity~Junction + Traffic_Signal,data = Acc_T
rain1)

#Use the first model to test on Nolab_Test1
NB_Predict4 <- predict(NBAcc4, Nolab_Test1)
```

```
.
.
. Down to three models using the algorithm and respective data sets
```

#Add all results together(NB)

```
##           NBLabels2
## NBResults2     1     2     3     4
##          2   525 26621  3100   592
##          3    89  6451  4106   549
```



	NBCM2.overal
Accuracy	0.73
Kappa	0.29
AccuracyLower	0.73
AccuracyUpper	0.74
AccuracyNull	0.79
AccuracyPValue	1
McnemarPValue	NaN

Naïve Bayes Results

Looking within the Naive Bayes Models, the results including all physical environment variables showed an accuracy of 73% with a Kappa value being only .29. Given the data set size of the sample the Kappa not being too low isn't a bad thing considering it is not based on the whole data set. Looking at the Naive Bayes model with just Junction and Traffic Signal being provided as x variables, the results were similar in predicting accident severity based on these variables. There were many correctly predicted accidents of severity being 2 on a scale of 1-4

Random Forest Classifier

The foundation blocks of a random forest classifier are decision trees. These fictional trees work as a flowchart in which the most important feature is the top node. The tree begins to branch out and split on different nodes based on an information gain calculation. The tree stops splitting after the information gain is minimal and each data point has been classified into a category. The random forest classifier expands on this functionality by utilizing multiple decision trees. Each decision tree gives a classification output for each data point. The random forest classifier aggregates the

counts and chooses the majority class as the output for each data point. This algorithm is very well equipped to handle a classification problem like this one. The random forest classifier will attempt to classify each data point into 1 of the 4 accident severity categories.

Due to the size of the data, a 70% randomized sample of the data was taken.

```
#Creating sample of the data
df = accidents.sample(frac= 0.70, replace=False, random_state=1)
```

Based on the knowledge of the data acquired so far and multiple experiments, many of the variables were dropped. The data was split into independent (X) and dependent (y) variables. The dependent variable in this case is the accident severity. Some of the variables required one-hot encoding which creates a variable for each possible discrete value or category in the pre-processed data.

```
#Splitting the data into training and testing sets with 20% reserved for test
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
```

After the data was split, the model was trained, and predictions were generated. The model was configured to use 20 decision trees with a max depth of 20 per tee. This was found to be the optimal parameters in this experiment. The model achieved 78% accuracy.

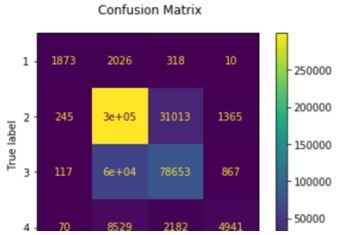
```
#Training the model
from sklearn.ensemble import RandomForestClassifier
rf = RandomForestClassifier(n_estimators=20, max_depth = 20, random_state=0).fit(X_train, y_train)
#Making predictions
y_pred_test = rf.predict(X_test)
#Displaying model accuracy
round(rf.score(X_test, y_test), 4)
```

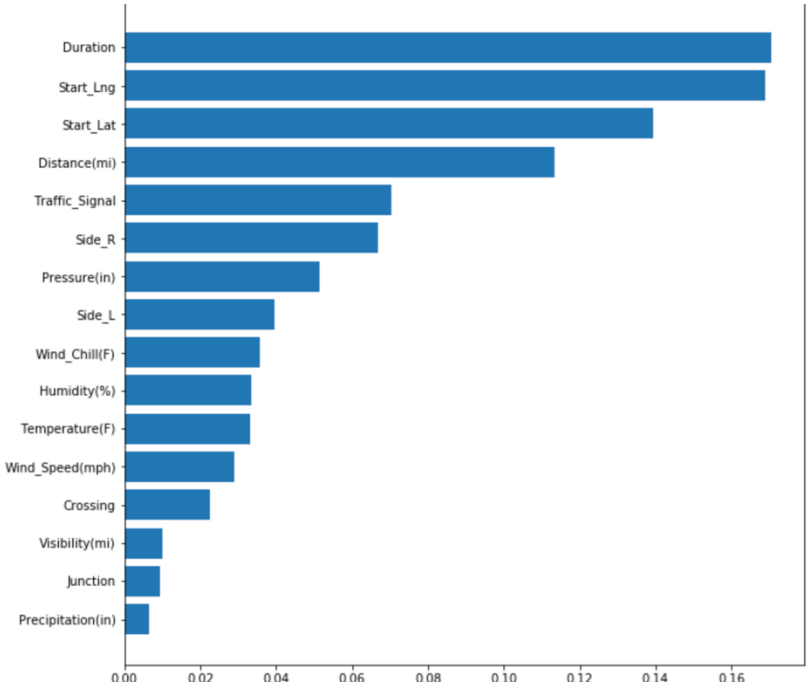
```
0.7829
```

The plots below display more detailed metrics of the model and a bar graph of the most significant variables for the model.

```
#Displaying more detailed metrics and visual confusion matrix
from sklearn import metrics
print("Classification report for classifier %s:\n%s\n"
      % (rf, metrics.classification_report(y_test, y_pred_test)))
disp = metrics.plot_confusion_matrix(rf, X_test, y_test)
disp.figure_.suptitle("Confusion Matrix")
print("Confusion matrix:\n%s" % disp.confusion_matrix)

plt.show()
```

```
Classification report for classifier RandomForestClassifier(max_depth=20, n_estimators=20, random_state=0):
              precision    recall  f1-score   support

           1       0.81      0.44      0.57      4227
           2       0.81      0.90      0.85    332294
           3       0.70      0.56      0.62    139664
           4       0.69      0.31      0.43     15722

    accuracy                           0.78    491907
   macro avg       0.75      0.56      0.62    491907
weighted avg       0.77      0.78      0.77    491907


Confusion matrix:
[[  1873   2026    318     10]
 [   245 299671  31013   1365]
 [   117  60027  78653    867]
 [    70   8529   2182   4941]]
```





Random Forest Classifier Results

The random forest classifier yielded a respectable accuracy score. However, upon further inspection, the model did not perform as well across all classes. Predicting whether an accident would be level 2 severity was achieved with a high level of accuracy with an 0.85 f1 score in this class. Predicting whether an accident would be level 4 severity was much less accurate with an

0.43 f1 score. It is important to note that class 2 was the most frequent class in the dataset, so the model had more data to learn from. It is also a fair assumption that higher severity accidents are inherently more complex. The model's performance was very impressive considering the size of the data. It took less than 45 seconds to train the model and make predictions. It appears that the duration of the accident, and the accident's location had the most impact on the model. The most significant structural variable was the presence of a traffic signal, while the most significant environmental variable was air pressure.

Predicting Accident Duration

The accident duration variable was not an organic variable in the dataset. As such, this variable was created by subtracting the accident end time from the accident start time. Due to the variability of this variable, a discreet measure of weather the accident caused a severe delay or not was created. This was done by creating a binary column delineating a 1 for a severe delay and a 0 for non-severe delay. The severe delay threshold was the accident duration median. The median accident duration was 0.75 hours or 45 minutes. Therefore, if the accident was longer than 45 minutes it was a severe delay, otherwise it was not.

Using this newly created variable, the problem was transformed into a binary classification problem. The objective was to predict whether an accident would cause a severe delay.

Decision Tree Classifier

The same data sample from the random forest classifier was used. However, the dependent and independent variables had to be slightly adjusted. The independent variables are all the same except that accident duration was removed, the dependent variable was created from it. The dependent variable was the previously created binary column severe delay.

```
#Dependent variable
X = df.drop(columns=['Severity', 'City', 'County', 'Zipcode', 'Country', 'Weather_Condition', 'Wind_Direction',
                     'Roundabout', 'Bump', 'Turning_Loop', 'Traffic_Calming', 'State', 'Sunrise_Sunset', 'Timezone',
                     'Stop', 'Amenity', 'Give_Way', 'No_Exit', 'Station', 'Railway', 'Duration', 'Severe_Delay' ], axis = 1)
#Independent columns
y = df.Severe_Delay
#One hot encoding variables
X = pd.get_dummies(X)
```

```
#Splitting the data into training and testing set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 42)
```

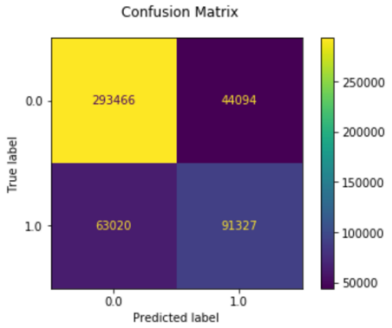
The model was trained using a max depth of 3. This was found to be the most accurate and best performing setting for this experiment. The model yielded 78% accuracy.

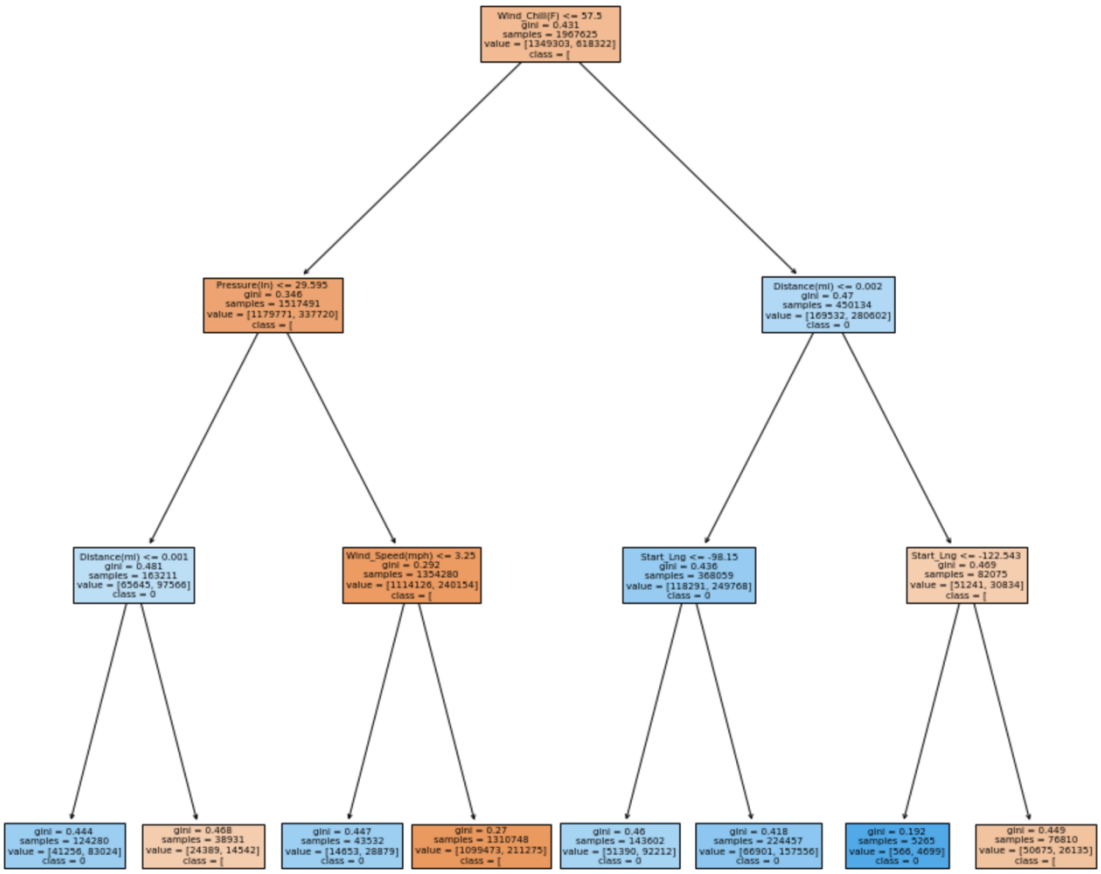
```
#Training model and making predictions
tree = tree.DecisionTreeClassifier(max_depth = 3)
y_pred = tree.fit(X_train, y_train).predict(X_test)
```

More detailed model metrics and the decision tree are visualized below.

```
Classification report for classifier DecisionTreeClassifier(max_depth=3):
              precision    recall  f1-score   support

         0.0       0.82      0.87      0.85    337560
         1.0       0.67      0.59      0.63    154347

    accuracy                           0.78    491907
   macro avg       0.75      0.73      0.74    491907
weighted avg       0.78      0.78      0.78    491907
```

```
Confusion matrix:
[[293466  44094]
 [ 63020  91327]]
```





Decision Tree Classifier Results

The decision tree achieved an 0.85 f1 score for the 0 class and an f1 score of 0.63 for the 1 class. Meaning that the model performed better when predicting if an accident would not cause a severe delay than when predicting otherwise. The decision tree was also very fast to train and make predictions, clocking in at less than 20 seconds. Based on the tree visualization, the most important variable was wind chill, followed by pressure and distance. The accident location was also important but not as much as the previously mentioned variables. Notably, the structure variables were not present in the tree.

Conclusion

Findings

Nationally, car accidents appear to be more prominent in states with a lower population, concentrating on mainly the Midwest. These states are also more likely to have more severe accidents. There is no evident significant seasonality month-to-month. However, there is hourly seasonality. Accidents are more frequent during the typical commuting hours of 8:00am – 9:00am and 4:00pm – 5:00pm. Additionally, the hours of 9:00pm – 3:00am appear to be the most dangerous, with a higher mean and median severity value. The severity of an accident can be accurately predicted; mainly based on the accident's duration, location, weather conditions, and presence of a traffic signal. An accident's duration above or below a specified threshold can be accurately predicted by considering weather conditions and location.

Limitations

The size of the dataset was a limiting factor in conducting diverse experiments. Some of the methods were limited to utilizing smaller samples of the data to conduct reproducible experiments. Methods such as support vector machines could not be adequately explored as they required significant computing power for a dataset of this size. Additionally, the dataset contained an overrepresentation of some regions and underrepresentation of others. This made it difficult to explore the data regionally as the conclusions would likely not be accurate. The data may have been a convenience sample, as it likely contained data collected from areas with wide use of the MapQuest and Bind API.

Why it Matters

As humans, the more comfortable that we get while performing a task, the more likely you are to make mistakes by diverging attention to other things. People over the age 15 in most places are on the road, driving, and forget that it is a dangerous task to perform. There are measures that have made driving better over the last decade, but there is still a problem in the roads or environment. Things can only be mitigated and not stopped when it comes to the rules of the road as far as surrounding forces. Finding common issues in accidents and finding areas of improvement can further prevent terrible things in the near future.

Future Research

Future research would require additional data, from more diverse sources. This additional data would require additional computing power to process it. In order to draw more conclusive results, the severity variable would either need to be under-sampled for overrepresented classes or oversampled for underrepresented classes. Creating more balance classes would reduce model bias, and lead to better performing models.

References:

Administration, N. (n.d.). FARS Encyclopedia. Retrieved November 15, 2020, from https://www-fars.nhtsa.dot.gov/Main/index.aspx

Brown, H., Jr. (2018). Leading Causes of Car Accidents With Statistics. Retrieved November 12, 2020, from https://www.jdsupra.com/legalnews/leading-causes-of-car-accidents-with-60370/

Dataset:

Moosavi, Sobhan, Mohammad Hossein Samavatian, Srinivasan Parthasarathy, and Rajiv Ramnath. "A Countrywide Traffic Accident Dataset.", 2019.

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