Data Analysis on US Car Accidents

Yuanhao zhu

May 30, 2020

Abstract: United States is one of the contries who have the largest amount of automobiles in the world. Prosperous as it is, massive number of car accidents took place there every year. There has been discussion upon the reason of car accidents, scientist crave to find a way to predict and control accidents around the country.This article focus on a rough analysis on US accidents data, along with brief model based on linear regression and neuron network. With the powerful statistic tool "R", data visualization could be implemented with ease. Presented with intuitive visualized data plots and corresponding explaination, this article is capable of delineating the US accidents comprehensively.The following simple mathmatical model based on linear regression and random forest could provide some insight about relation between weather, road condition, and some aspect of the accidents.

Contents

1 Overview

1.1 Background

According to a research data publicated at statista[1], there are about 273 millions of passenger cars in the US in 2018. Taking into account the fact that the population of US in 2018 is 327 millions, there are about 8 out of every 10 US civilian who own a car. Indeed, if you live in a place like the US, a proper auto-mobile is essential to everyday life, since in US the residential area is commonly far from the center city. As the population increases, there are more cars on the road, which lead to the problem that the control of car accidents could become a major challenge. Estimated over 6 millions of them each year[2], the car accidents in the US could not only cause great economical cost, but also delay the everyday commuting traffic. Many data scientist had been craving to find a way to predict the occurance of accidents that impact the traffic heavily, some real time predicting system had been developed. With the help of the powerful statistic tool like Python and R, we might be able to find a solution to control the car accidents and make everyone a better commuting traffic.

1.2 Dataset Introduction

We use the US accident[3] dataset publish by a US data scientist Sobhan Moosavi with the paper[4] that explains how the dataset is constructed and the details about the attribute.According the paper, the source of the dataset comes mainly from two map data api, the MapQuest realtime traffic data collector and the Microsoft Bing realtime traffic data collector.The data from these collectors are captured by a variety of entities - the US and state departments of transportation, law enforcement agencies, traffic cameras, and traffic sensors within the road-networks. The dataset contains about 3 million records.

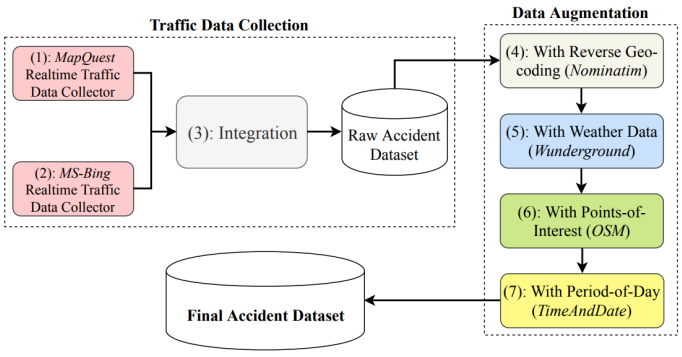

Figure 1: The dataset construction process[4]

The data was integrated and augmented after obtained from data collectors. The coordinates of the accidents are augmented through reverse-geocoding, the Point of interest is extracted by regular expression from the accident descriptions. The weather data was obtained from Weather Underground API collected from the nearest airport weather station and conbined with the accidents records to provide some context about accidents.

1.3 Dataset Format

The dataset has the following attributes

Table 1: Dataset attributes

Attribute	Definition
ID	Unique identifier for each accident
$Source$	Indicates source of the accident report
TMC	Traffic Message Channel (TMC) code
$Severity$	Shows the severity of the accident
$StartTime$	Shows start time of the accident in local time zone.
$StartLatitude$	Shows latitude of the start point.
$StartLongtitude$	Shows longitude of the start point.
$Distance(miles)$	The length of the road extent affected by the accident.
$Description$	Shows natural language description of the accident.
$WeatherTimestamp$	Shows the time-stamp of weather observation.
$Temperature(F)$	Shows the temperature (in Fahrenheit).
$WindChill(F)$	Shows the wind chill(apparent temperature) (in Fahrenheit).
$Humidity(\%)$	Shows the humidity (in percentage).
$Pressure(in)$	Shows the air pressure (in inches).
$Visibility(mi)$	Shows visibility (in miles).
$WindSpeed(mph)$	Shows the wind speed
$Precipitation(in)$	Shows precipitation amount in inches, if there is any.
$WeatherCondition$	Shows the weather condition
$POInotations$	Indicate the presence of various road conditions

Explaination upon some of these attribute is necessary

Severity, unlike what most people would concern, indicate how badly the accident had impact the traffic. There are four levels of severity in the dataset from 1 to 4. Larger severity means that longer delay is caused by the accident.For instance, if the severity is 1 than 2 or 3 minutes would be the delay time.

TMC, which is a method use by some countries including US among traffic police to do quick reports on the accidents. Each TMC code represent a specific event that had slow down the traffic.

Wind chill is a temperature that how you actually feel outdoor, and is affected by the wind speed. If the wind speed is higher, then the evaporation would be quicker so the temperature you actually feel would be lower.

Pressure is measured by barometer, so the it is presented in inches. If we wish to convert it to pascals, we need to multiply the value by 3386.53.

POI notations are series of boolean values indicating the presence of point of interest(POI). It tells the road condition of the accident location such as the presence of junction, crossing, traffic signals.

2 Data Visualizations

2.1 Accidents Locations Summary

We employee ggplot2[6] in R[7] to visualize the data and managed to plot the accidents on map of the US.

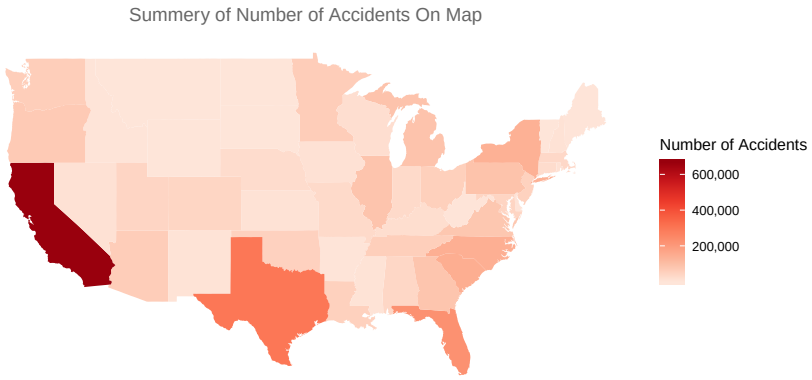

Figure 2: Accidents locations on map

We can see in figure2 that California has the most number of accident, and that Texas and Florida is also high in accidents numbers. We learn that california is the most populous state, consequently the number of car accidents should be higher that other states. Main reason for that is the affluent IT industry and massive amount of people from not only US but all over the world gather in California to persue high salary.

We then look into California to see which cites has the high car accidents

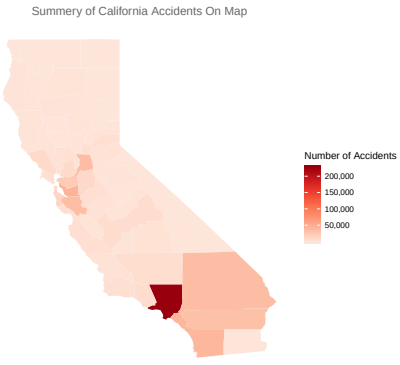

Figure 3: Accidents locations of California

It is apparent that most accidents happen in Los Angeles and San Francisco,which proved the statement that most people come to California to persue IT jobs. Most IT engineer got high salary and live outside the center of the city where their companies

locates, so driving a car to work is necessary for most San Francisco, which might lead to the fact that a lot more accidents would happen here. After the visualization, we might need to warn the government of these two cities to come up with and idea to control the car accidents.

2.2 Accident Tendency

For top 3 states who have the highest number of accidents, we managed to plot the tendency of accidents over 2016 to 2019

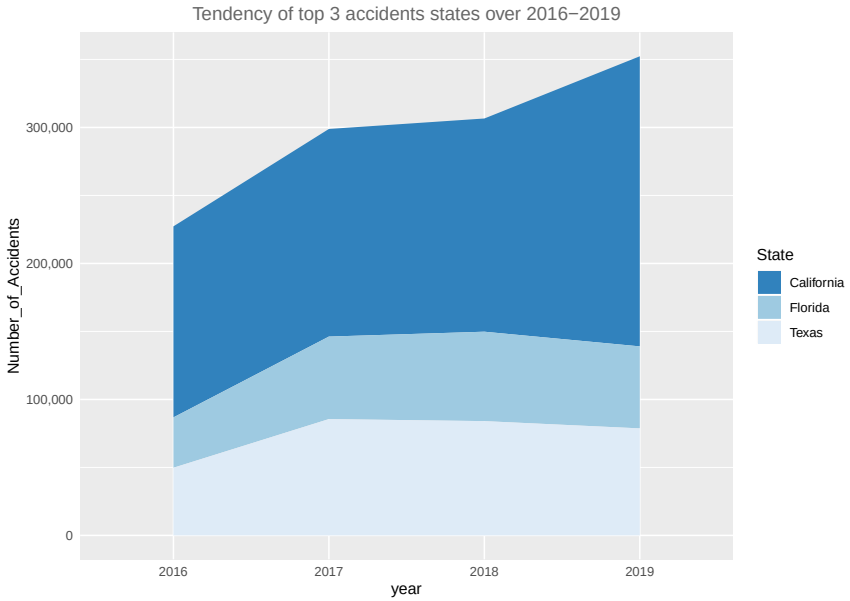

Figure 4: Accident tendency over years

We can see from the figure that for California the number of accidents keeps climbing from 2016 to 2019, which correspond the fact that the amount of auto-mobile in this particular state increases over last few years[5]. More cars would definitely cause more accidents.

While for Texas and Florida, the amounts of accidents are prone to stable.On the one hand, the population of these two states do not grow as sharply as CA, so that there would not be gargantuan increase on the amount of motor vehicles. On the other hand, it might be correlated with the fact that population in CA is more hetero-ethnical. People with different culture and background could have diverse driving habits, which might lead to more occurance of car accidents.

Then we map the accident to the time of the day and examine the tendency of it on figure5.

It is apparent that the period of 7 am to 8 am and the period of 4 pm to 5 pm is the time of the day that most accidents happens. Most people go to work at around 7 to 8 am and go back to home at 4 to 5 pm, which explains why most accidents where recorded at these period of the day. Moreover, it seems accidents are more likely to occur in the morning, which we interpret it as the result of people being too hurrying on their way to work and the result of not completed awake. If you are going to be late, you would drive faster and thus the probability of accident could increase. It could also be noticed that during the day time around 12 am is the period when the

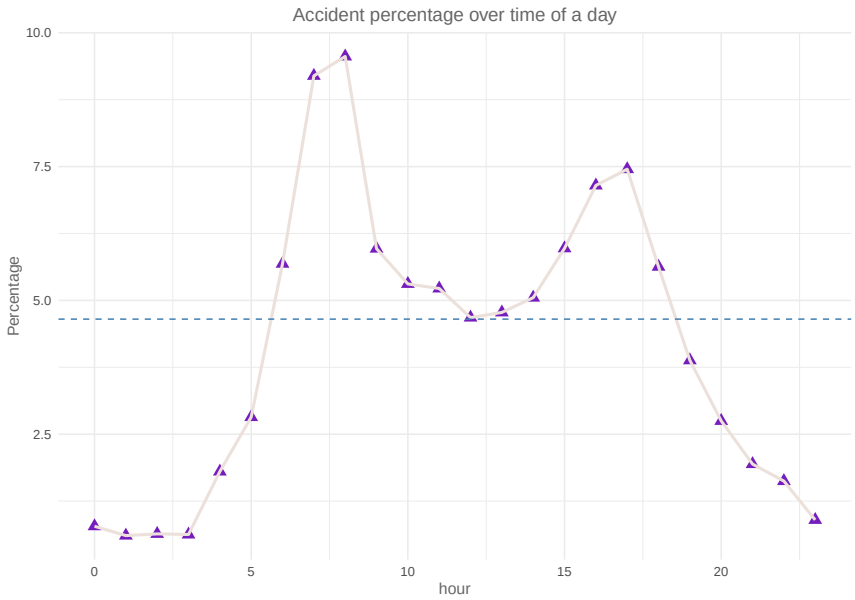

Figure 5: Accident tendency over time of the day

least accident happens. It probably occur for most people stay at their work place so that there won't be to many auto-mobiles on the road.

3 Analysis on the Factors of Severity

3.1 Data cleaning

For the dataset we obtain, it is necessary to clean the data for further use. We firstly convert the raw data into proper type.(Date conversion was implemented by lubridate[8])

We look into the dataset POI notations and found that only Amenity, Crossing, Junction, Traffic Signal is acceptablly balanced, the rest of POI notation is extreme in-balanced so we convert only four POI notations above as the possible x and discard the rest.

Table 2: Dataset attribute type conversion

Attribute	Before(type)	After(type)
StartTime	Characters	POSIXct Time
WeatherTimestamp	Characters	POSIXct Time
POIAmenity	Characters	Boolean
POICrossing	Characters	Boolean
POIJunction	Characters	Boolean
POITrafficSignal	Characters	Boolean

Then we examine the NA values in each columns by Naniar package[9], the results are presented in figure 6

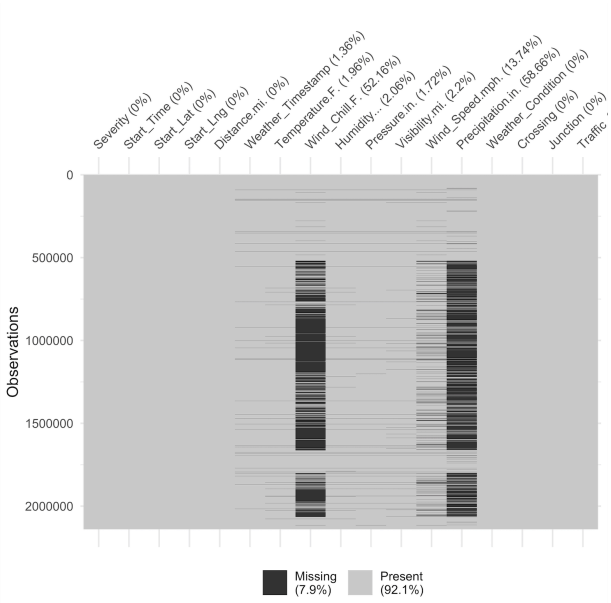

Figure 6: Missing values of each columns

From the figure 6 we find out that there are missing values in temperature, humidity, wind chill, and precipitation columns, which are mainly weather condition data.

We thus employ package Zoo[10] to fill NA values in the dataset. The na.fill function in Zoo package will fill the NA values with the nearest non-NA values in the dataset or with interior linear interpolation. The adventage of this method is that NA values could be fitted with reasonable values.When the NA value has a neighboring non-NA value, then use its value to replace the NA is reasonable. While if the NA value is in the mid of nowhere, then the linear interpolation could fit the NA value with linear stimulation.

3.2 Quantifying Accident Severity

The dataset had presented us with the accident severity, however, the level indicator of the severity is not enough for us to train a linear model. The reason behind is that most accident are level 1 or 2 accidents, which will lead to the fact that the training set could not be diverse enough in severity level. In a word, the severity levels are inbalanced. We solve this problem by converting the severity level into the actual delay time that it might cause in average. In order to achieve that, we obtain an estimated relation between the severity level and the delay time from the dataset author.

Table 3: Severity and Delay

Severity level	Delay time
1	2 minutes and 30 seconds
2	3 minutes and 15 seconds
3	8 minutes
4	18 minutes

With the correlation between severity level and the delay time, we are able to aug-

ment the severity level with actual delay time.However, directly map the severity level to the data above won't solve problem. In order to maximize the diversity of delay time, we randomly choose a delay time for a particular accident with respect to its severity level. We convert the corresponding delay minutes into intervals measured in seconds by empirical analysis, then we randomly choose a delay time(seconds) for a specific accident. The quantification algorithm is illustrated below.

```
Begin
  class <- data["Severity"]
  if(class == 1){
    Delay <- RandomlyChooseFrom(0,150)
  }
  else if(class == 2){
    Delay <- RandomlyChooseFrom(150,480)
  }
  else if(class == 3){
   Delay <- RandomlyChooseFrom(480,1080)
  }
  else if(class == 4){
    Delay <- RandomlyChooseFrom(1080,1800)
  }
  return(Delay)
End
```

3.3 Correlation between variables

We use the "cor" function to calculate the correlation coefficient, and the spearman method is employed. The spearman correlation is a nonparametric measure of rank correlation (statistical dependence between the rankings of two variables). It assesses how well the relationship between two variables can be described using a monotonic function[12].

The spearman coefficent of two variable is defined as below.

$$r_s = \rho_{rg_X, rh_Y} = \frac{cov(rg_X, rh_Y)}{\sigma_{rg_X} \sigma rg_Y} \tag{1}$$

where:

- *rho* denotes the usual Pearson correlation coefficient, but applied to the rank variables,
- $cov(rg_X, rh_Y)$ is the covariance of the rank variables,
- σ_{rg_X} and σrg_Y are the standard deviations of the rank variables.

The spearman coefficient could provide us with a brief insight on the correlations amoung the dataset variables, and the results are presented with corrplot package[11] in figure 7

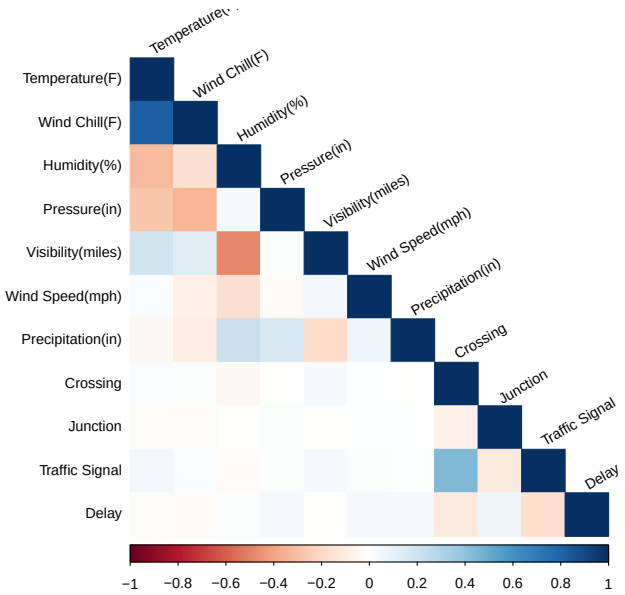

Figure 7: Correlation plot of the variables

3.4 Linear Regression Fit

Linear Regression is a powerful statistical tool that allow us to build multi-variable linear model for a given dataset. Compared with more contemperary method such as neuron network, the linear regression fit requires less computation power. At the same time, the result coefficient could give us insight on how the variable affect the dependent variable.

Given a set of $x_{11}, x_{12}, x_{13}, \cdots, x_{ji}$ data, the linear regression is defined as follow:

$$y_i^p = \beta_0^p + \beta_0^p x_{1i} + \beta_2^p x_{2i} + \cdots + \beta_k^p x_{ki} \tag{2}$$

where:

- y_i^p is the predicted value of i th observation
- x_{ji} is the jth x value of i th observation
- β_0^p is the intercept, which indicate the predicted y when other variables are 0
- β_0^j is the coefficient of jth control variable(x)

The idea is to minimize the differece between the real observatoin value and the predicted value. Firstly, we select all variable in the correlation plot as the control variable and the delay time as the dependent variable to fit a model. Considering the fact that wind chill is directly affected by wind speed, we take them as interaction items. The first model is presented below

$$\begin{aligned} Delay \leftarrow\ & Temperature.F. + WindChill : WindSpeed + Pressure.in. + Precipitation \\ & + Humidity + Visibility + Crossing + Junction + TrafficSignal \end{aligned}$$

The training set is randomly select 80% of the whole dataset and the rest was used as validation set

Then we use "summary" function to examine the results of the first fit:

```
Call:
lm(formula = Delay ~ Temperature.F. + Wind_Chill.F.:Wind_Speed.mph. +
    Pressure.in. + Precipitation.in. + Humidity... + Visibility.mi. +
    Crossing + Junction + Traffic_Signal, data = accident.data.clean)

Residuals:
     Min       1Q   Median       3Q      Max
-1395.69  -205.07   -81.54   129.63  1481.50

Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                     3.498e+02  7.289e+00   47.99   <2e-16 ***
Temperature.F.                 -4.077e-01  1.235e-02  -33.02   <2e-16 ***
Pressure.in.                    4.871e+00  2.418e-01   20.14   <2e-16 ***
Precipitation.in.               6.037e+00  4.724e-01   12.78   <2e-16 ***
Humidity...                     2.220e-01  9.843e-03   22.55   <2e-16 ***
Visibility.mi.                  9.111e-01  7.075e-02   12.88   <2e-16 ***
CrossingTRUE                   -5.265e+01  8.571e-01  -61.43   <2e-16 ***
JunctionTRUE                    5.164e+01  7.726e-01   66.84   <2e-16 ***
Traffic_SignalTRUE             -1.146e+02  5.942e-01 -192.81   <2e-16 ***
Wind_Chill.F.:Wind_Speed.mph.   2.318e-02  7.045e-04   32.90   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 291 on 2139610 degrees of freedom
Multiple R-squared:  0.03481, Adjusted R-squared:  0.03481
F-statistic:  8574 on 9 and 2139610 DF,  p-value: < 2.2e-16
```

We find out that all variables are significant, but we still need to exclude one particular controlling variable. It is noticed that the visibility has a postive effect on the length of the delay time, which does not meet the empirical understanding upon the relationship and the visibility. It is commonly agreed that if the visibility is low then more severe accident might happen.

The abnomaly on the coefficient of visibility could be explained as that fact that most light accidents actually happens when there are traffic jams, where most cars are in close range. Under this specific circumstance, visibility is no longer the fact that would lead to an accident, on the other hand, the density of the cars should be the one that counts. Additionlly, most part of the accidents in the dataset are light accidents, hence the influence of visibility on accidents might be mislead by the inbalance of the dataset. In order to make the model more persuasive, we should exclude the visibility out of the controlling variables.

The new model is presented in equation below:

$$Delay \leftarrow Temperature.F. + WindChill : WindSpeed + Pressure.in. + Precipitation + Humidity + Crossing + Junction + TrafficSignal$$

Again, we use "summary" function to get the result of the fit:

```
Call:
lm(formula = Delay ~ Temperature.F. + Wind_Chill.F.:Wind_Speed.mph. +
    Pressure.in. + Precipitation.in. + Humidity... + Crossing +
    Junction + Traffic_Signal, data = accident.data.clean)

Residuals:
```

```
      Min        1Q    Median        3Q       Max
-1389.08   -205.09    -81.55    129.64   1480.83

Coefficients:
                                 Estimate Std. Error t value Pr(>|t|)
(Intercept)                     3.594e+02  7.251e+00   49.57   <2e-16 ***
Temperature.F.                 -3.995e-01  1.233e-02  -32.40   <2e-16 ***
Pressure.in.                    4.908e+00  2.418e-01   20.30   <2e-16 ***
Precipitation.in.               6.011e+00  4.724e-01   12.72   <2e-16 ***
Humidity...                     1.781e-01  9.235e-03   19.28   <2e-16 ***
CrossingTRUE                   -5.249e+01  8.570e-01  -61.24   <2e-16 ***
JunctionTRUE                    5.162e+01  7.726e-01   66.81   <2e-16 ***
Traffic_SignalTRUE             -1.145e+02  5.942e-01 -192.71   <2e-16 ***
Wind_Chill.F.:Wind_Speed.mph.   2.304e-02  7.044e-04   32.70   <2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 291 on 2139611 degrees of freedom
Multiple R-squared:  0.03474, Adjusted R-squared:  0.03473
F-statistic:  9624 on 8 and 2139611 DF,  p-value: < 2.2e-16
```

To validate the exclusion of the visibility is feasible, we use Chi-Square test to implement the validation. Pearson's chi-squared test is used to determine whether there is a statistically significant difference between the expected frequencies and the observed frequencies in one or more categories of a contingency table. In this case, we examine if there is a significant difference between the two model's prediction. Using the prediction results, we examine the predicted values of the two models to test if there is a different.

First, we have the following assumption:

- $H0$: Two models are not related with each other
- $H1$: Two models are related with each other

Then we call the built-in Chi-square test function in R to examine the hypothesis, the result is:

```

  Pearson's Chi-squared test

data:  df.pred$lm.pred.1 and df.pred$lm.pred.2
X-squared = 23690000, df = 22477072, p-value < 2.2e-16
```

We can see that the p-value is extremely low, less than 0.05 significance level. So we reject the null hypothesis and conclude that the two models have a significant relationship, which means that the exclusion of visibility is feasible.

3.5 Conclusion

From the linear regression fit, we can learn that the temperature, crossing, traffic signal have the negtative influence on the delay time. Pressure, precipitation, humidity, junction, wind chill and wind speed, have a positive effect on delay time.The result meets the empirical analysis that when temperture is low and precipitation increases, car the probability of car accident would grow. For the low temperature and raining

could reduce the friction of the road. Meanwhile, if there is a crossing or junction, traffic signal helps decrease the likelihood of the occurrance of car accidents.We can infer that when the road condition is complex, like the junction and the crossing, a traffic signal helps regulate the cars, so that drivers are passing through the area with guidance. Consequantly, the traffic signal could be an essential infrastructure for complex traffic locations.

4 Time Prediction of Long Delay Accidents using Random Forest

4.1 Time gap conversion for dependent variable

Firstly, we define a long delay accident to be the accidents which has a severity level of 3 to 4. We use "subset" function to seperate them from the original cleaned data set and randomly chose 80% of the data as the trainning set.

Outputting a timestamp from nowhere could be way too difficult, however, we can simplify the problem using time gap. From the dataset author, we learn that instead of at the time when the accident happens, the weather data is actually measured in 15 minutes before or after the accident. Hence, we convert the gap between the time when the weather data is measure and the time when the accident happens to seconds as the dependent variable.

Additionally, we need to take the absolute value of the result. Since when it comes to predicting, when we have the weather data at a specific time point, only the positive timegap would make sense so that we can predict how long would the next long delay accident happen. We present the conversion as following.

The point is this:**If we get a set of weather data at time stamp t_1, we get an estimated time gap G from the model, which means that from t_1, after G seconds, there could be a long delay accident occurance.**

$$TimeGap = seconds(abs(accidentStartTime - WeatherTimeStamp)) \tag{3}$$

4.2 Random Forest

Random forest is also known as the random decision forest. The random forest is consist of a number of decision tree, which in the regression case, outputs the mean prediction of the input. The decision tree, on the other hand, is a popular machine learing algorithm that uses the data structure consist of decision node and the path to its children. Random forest algorithms involve sampling sample units and variables to generate a large number of decision trees. For each sample unit.That is, all decision trees are classified in turn. The mode category of all decision tree prediction categories is predicted by the random forest. The same as the category of this unit.[13]

For a dataset with N sample units and M variables, the random forest act like below:

- Sample N units with putting back and generate large amount of decision tree

- Randomly select $m < M$ variables at each node as the node-spliting candidate where the m should be constant in every node.
- Completed generate all decision trees, no branch would be cut.
- The type of terminal node is decided by the mode of the node
- For a new obervation, use all the tree generated to classsify it, where the result is decided by majority principle.

Then we firstly select all variables in the coefficient matrix as the controlling variable and train the model. Since the random forest trainning requires massive memory space, it is not possible to use all the data so we reduce the size of the training set to 50000.

The model is presented as below:

$$TimeGap \leftarrow Temperature.F. + WindChill : WindSpeed + Pressure.in. + Precipitation + Humidity + Crossing + Junction + TrafficSignal$$

Then we call the function "randomForest"[14] to train a model, the result is as following:

```
1
2 Call:
3  randomForest(formula = TimeGap ~ Temperature.F. + Pressure.in. +
     Precipitation.in. + Humidity... + Wind_Speed.mph. + Junction +
     Traffic_Signal, data = accident.train)
4                Type of random forest: regression
5                      Number of trees: 500
6 No. of variables tried at each split: 2
7
8           Mean of squared residuals: 3785769
9                     Var explained: 1
```

Where we can learn that the random forest model is struggling to fit the data, with a mean of squared residuals of 3785769, and that only 1% of the variables are explained. We persume that the result is not good enough becasue random forest is not good at fitting multiple variable regression mission. Hence we need to transfer the continuous time gap into classifications to give the random forest better preliminaries.

4.3 Dependent variable refinement

It is already clear that the time gap between the weather timestamp and the accident time is less than 20 minutes, then we assume that put those data into 3 bins as classes could be a reasonable solution to discrete the time gap.

The algorithm is described as following:

```
1  gap <- as.numeric(data["TimeGap"])
2  if(gap <= 300){
3    bin <-  1
4  }
```

```
  else if((gap > 300) && (gap <= 600)){
    bin <- 2
  }
  else {
    bin <- 3
  }
  return(bin)
```

Then we apply the random forest again. The result is as following:

```
  Call:
 randomForest(formula = bin ~ Temperature.F. + Wind_Chill.F. +      Wind
    _Speed.mph. + Pressure.in. + Precipitation.in. + Humidity... +
    Visibility.mi. + Crossing + Junction + Traffic_Signal, data =
    accident.train)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 50.15%
Confusion matrix:
     1    2     3 class.error
1 2435 1025  9788   0.8161987
2 1689  893  8873   0.9220428
3 2381 1317 21599   0.1461833
```

Now the random forest is making classification on the interval that we produce.On the one hand this model might not give out a specific time when the next long delay accident might happen, on the other hand it improved the correctness with the output of a rougher results.The refined dependent variable helped the dataset to fit the random forest. We could get the long delay prediction on the scale of 5 minutes ahead of its occurance.

4.4 Prediction results and performance evaluation

Using the refined random forest fit, we can get the prediction results on the validation set presented as following.

The predicted/actual value matrix is as follwing:

```
       Predicted
Actual    1    2    3
     1  286  109  818
     2  224   88  701
     3  362  181 2231
```

From the matrix we can learn that the model is best at predicting the level 3 and level 1 accidents, which means that for the accidents that might not happen in the next 10 minutes, we can have our prediction presented. For those long delay accidents that is happening in 5 minutes, we could have the the precision about 23%. We than use Receiver Operating Characteristic curve to evalute the model. In this case, we have more than two classes in our model, so multi-class ROC evaluation is needed.

We use the "multiclass.roc" function in the pROC package[15] to calculate the average AUC. The AUC stands for area under the curve, which is the area size that under the ROC curve. The AUC value is the indicator of how well the mode could distinguish between classes.

The result is presented as following:

```
Call:
multiclass.roc.default(response = accident.test$bin, predictor = tm.pred
    )

Data: multivariate predictor tm.pred with 3 levels of accident.test$bin:
    1, 2, 3.
Multi-class area under the curve: 0.5462
```

Where the average AUC is 0.5462, meaning that our model is able to distinguish just over 50% of the data give.

Next, we use the "importance" function to examine which variables are important in the random forest we just contructed.

The results are presented below:

```
                   MeanDecreaseGini
Temperature.F.             4569.5046
Wind_Chill.F.              5403.8011
Wind_Speed.mph.            3726.7086
Pressure.in.               5514.3888
Precipitation.in.          2385.8355
Humidity...                4221.8433
Visibility.mi.             1322.5229
Crossing                    137.9778
Junction                    521.4034
Traffic_Signal              333.2018
```

We can learn that the weather data including temperature, wind chill, pressure, are essential when making a prediction. The results meet the empirical analysis, when we get the weather data, we can infer the condition of the roads and thus predict the time interval of how long until the next long delay accidents might happen in a particular location.

5 Conclusion

Traffic accidents are major problem of modern cities. Maybe just a "touch" from your car to others might cause a light accident. The traffic accident as related to the number of motor vehicles in a particular region, the more the cars there are, the more accidents it would be. In our analysis, the California has way more accident happen from 2016 to 2019, for it has the largest number of cars registered in US. Next, we analyzed the time of the accidents and it turned out that most accidents happen on your way to work. We propose that taking public traffic like subways and bus could essentailly mitigate the situation. We also found out some critical factor that influence how bad a specific accidents could impact traffic. It is noticed that lower temperature could increase the severity and the presence of junction could increase the delay time of an accdient. In general, driving in an overcast weather with raining and gust could form the obstruction that stops you from getting a hot dinner. Using random forest, we were able to construct a rough model predicting the time of the long delay accidents occurance. If we could get the access of weather data at a specific location, we can estimate the time of accident with the weather data and the time that we get these data.

6 Appendix:Data Visualization Code List

```r
#Set the workspace
workspace <- "~/Downloads/R/us_traffic_accidents"
setwd(workspace)

#Read in the data file
accident.data.ori <- read.csv("./US_Accidents_Dec19.csv")
accident.data <- accident.data.ori
accident.data$Start_Time <- parse_date_time(accident.data$Start_Time,"ymd HMS")
accident.data$End_Time <- parse_date_time(accident.data$End_Time,"ymd HMS")


#Data visualization

#Visualizing the percentage of the TMC Type
TMC_df <- data.frame(table(accident.data$TMC))
names(TMC_df)[1] <- "TMC_code"
names(TMC_df)[2] <- "Frequency"

#Cut the TMC "Accidents"

TMC_df <- subset(TMC_df,TMC_code != 201)
other.freq.count <- 0
for(i in  (1 : nrow(TMC_df))){
  if(TMC_df[i,"Frequency"] < 11163){
    other.freq.count <- other.freq.count + TMC_df[i,"Frequency"]
  }
}
other.freq.count
TMC_df <- subset(TMC_df, Frequency >= 11163)
TMC_df <- rbind(TMC_df,data.frame(TMC_code = "others",Frequency = other.freq.count))
TMC_df
TMC_colors <- c("#888888", "#E69F00", "#56B4E9", "#009E73",
                "#F0E442", "#0072B2", "#D55E00")
TMC_pie  <- ggplot(TMC_df, aes(x="", y=Frequency, fill=TMC_code)) + geom_bar(width = 1, stat = "identity") + coord_polar("y", start=0) + scale_fill_manual(values = TMC_colors,limits = c("241","others","245","244","229","222","203"),labels = c("accident(s). Right lane blocked","others","accident(s). Two lanes blocked","accident(s). Hard shoulder blocked","accident(s). Slow traffic","accident(s). Queuing traffic","multi-vehicle accident (involving Q vehicles)")) + theme_void() + labs(x = NULL,y = NULL, fill = NULL, title = "TMC Code Summary") + geom_text(aes(label = paste0(round((Frequency/sum(TMC_df$Frequency))*100), "%")), position = position_stack(vjust = 0.5),color = "black")
TMC_pie <- TMC_pie + theme_classic() + theme(axis.line = element_blank(),
                                             axis.text = element_blank(),
                                             axis.ticks = element_blank(),
                                             plot.title = element_text(hjust = 0.5, color = "#666666"))
TMC_pie


```

```r
#Visualizing the Accidents on map state-wide
map.df <- data.table(subset(accident.data,select = c("Start_Lng","Start_Lat","Zipcode","County","City","State")))
map.df.back <- map.df
map.df <- subset(map.df, select = c("State"))
state.accident.count <- data.table(table(map.df))
names(state.accident.count)[1] <- "State"
names(state.accident.count)[2] <- "Number_of_Accidents"


#Get the map data and convert the state names
us_map <- map_data("state")
us_map$region <- state2abbr(us_map$region)
us_map <- data.table(us_map)
names(us_map)[4] <- "order"
names(us_map)[5] <- "State"
names(us_map)[6] <- "County"
#setkey(us_map,State,County)
head(us_map)

#Merge the count and the map_data
map.data.plot <- merge(us_map,state.accident.count,by = "State")
head(map.data.plot)

#Get the plot
map.plot <- ggplot(map.data.plot, aes(long, lat)) +
  geom_polygon(aes(group = group,fill = Number_of_Accidents)) +
  coord_map()
map.plot <- map.plot + labs(fill = "Number of Accidents",title = "Summery of Number of Accidents On Map",axis = "")
map.plot <- map.plot + theme_classic() +theme(plot.title = element_text(hjust = 0.5, color = "#666666"),axis.text = element_blank(),
                                axis.ticks = element_blank(),axis.line = element_blank(),axis.title = element_blank())
#Convert Scientific numbers to readable numbers
map.plot <- map.plot + scale_fill_distiller(palette = "Reds",labels = comma,direction = 1)
map.plot

#Visualize the CA state
ca.df <- data.table(subset(accident.data,State == "CA",select = c("Start_Lng","Start_Lat","Zipcode","County","City","State")))
head(ca.df)
#Convert the format of the county column so that they could be merged with map data
ca.df$County <- as.character(cs.df$County)

#Get the accident count
ca.accident.count <- data.table(table(ca.df$County))
names(ca.accident.count)[1] <- "County"
names(ca.accident.count)[2] <- "Number_of_Accidents"


#Get the map data and convert the state names
ca_map <- map_data("county",region = "california")
ca_map$region <- state2abbr(ca_map$region)
ca_map <- data.table(ca_map)
#names(ca_map)[4] <- "order"
names(ca_map)[5] <- "State"
```

```
names(ca_map)[6] <- "County"
#setkey(us_map,State,County)
head(ca_map)
unique(ca_map$County)
#Capitalize every first character of each word so that they could be merged
ca_map$County <- stri_trans_totitle(ca_map$County)
unique(ca_map$County)

#Merge the data for CA state
ca.data.plot <- merge(ca_map,ca.accident.count,by = "County")
head(ca.data.plot)

#Get the plot
ca.plot <- ggplot(ca.data.plot, aes(long, lat)) +
  geom_polygon(aes(group = group,fill = Number_of_Accidents)) +
  coord_map()
ca.plot <- ca.plot + labs(fill = "Number of Accidents",title = "Summery of California Accidents On Map",axis = "")
ca.plot <- ca.plot + theme_classic() +theme(plot.title = element_text(hjust = 0.5, color = "#666666"),axis.text = element_blank(),
                                            axis.ticks = element_blank(),axis.line = element_blank(),axis.title = element_blank())
#Convert Scientific numbers to readable numbers
ca.plot <- ca.plot + scale_fill_distiller(palette = "Reds",labels = comma,direction = 1)
ca.plot


#Plot the accidents based on zipcode
zip.df <- data.table(subset(accident.data,select = c("Start_Lng","Start_Lat","Zipcode","County")))
names(zip.df)[1] <- "Lng"
names(zip.df)[2] <- "Lat"
table(zip.df$Zipcode)

#Filled Line plot for top 3 states with highest accident counts
line.top.three.dt <- data.table(subset(accident.data,select = c("State","Start_Time")))
line.top.three.dt$Start_Time <- year(line.top.three.dt$Start_Time)
line.top.three.dt <- data.table(table(line.top.three.dt))
head(line.top.three.dt)
names(line.top.three.dt)[3] <- "Number_of_Accidents"

#Sort the data and get the top three
state.accident.count <- state.accident.count[with(state.accident.count, order(-Number_of_Accidents)),]
top_three_state <- state.accident.count[1:3,]
line.top.three.dt <- subset(line.top.three.dt,State == top_three_state[1,State] | State == top_three_state[2,State] | State == top_three_state[3,State])
#Exclude years with no records
line.top.three.dt <- line.top.three.dt[4:15,]
names(line.top.three.dt)[2] <- "year"
area.year.plot <- ggplot(line.top.three.dt,aes(x = year,y = Number_of_Accidents))  +  geom_area(aes(group = State,fill = State))
            + scale_fill_brewer(direction = -1,labels = c("California", "Florida"  ,  "Texas")) + scale_y_continuous(labels = comma)
```

```
area.year.plot <- area.year.plot + theme(plot.title = element_text(hjust = 0.5, color = "#666666"),axis.ticks = element_blank())
                 + labs(title = "Tendency of top 3 accidents states over 2016-2019")
area.year.plot

#Get the data by year
us.accident.dt <- data.table(subset(accident.data,select = "Start_Time"))
us.accident.dt$Start_Time <- year(us.accident.dt$Start_Time)
names(us.accident.dt)[1] <- "year"

#Get the count of the accidents
us.tendency.count <- data.table(table(us.accident.dt))
us.tendency.count <- us.tendency.count[2:5,]
names(us.tendency.count)[1] <- "year"
names(us.tendency.count)[2] <- "Number_of_Accidents"

us.tendency.plot <- ggplot(us.tendency.count,aes(x = year,y = Number_of_Accidents,fill = year)) + geom_bar(stat="identity",width = 0.7) + geom_text(aes(label=Number_of_Accidents), vjust=1.6, color="white", size=5) + scale_fill_manual(values=c("#999999", "#E69F00", "#56B4E9", "#f69ca6"))
us.tendency.plot <- us.tendency.plot + theme_minimal() + labs(title = "Number of Accidents in US over 2016-2019") + theme(plot.title = element_text(hjust = 0.5, color = "#666666"),axis.title = element_text(color = "#666666"))
us.tendency.plot

#Visiualize the number of accidents within different visiblity

#Get the data
visibility.dt <- data.table(subset(accident.data,Severity > 3,select = "Visibility.mi."))
visibility.count <- data.table(table(visibility.dt))
names(visibility.count)[1] <- "Visibility"
names(visibility.count)[2] <- "Number_of_Accidents"
visibility.count
visibility.count <- data.table(subset(visibility.count,Number_of_Accidents > 2000 & Number_of_Accidents < 230000))
visibility.count$Visibility <- as.numeric(visibility.count$Visibility)

visibility.count <- visibility.count[order(Visibility),]
#plot
vis.count.plot <- ggplot(visibility.count,aes(x = Visibility,y = Number_of_Accidents)) + geom_point() + geom_smooth(color = "#522492",fill = "#e6d1bf") + theme_minimal() + labs(title = "Rough relationship between Visibilty and Number of Accidents") + xlab( "Visibility(miles)") + theme(plot.title = element_text(hjust = 0.5, color = "#666666"),axis.title = element_text(color = "#666666"))
vis.count.plot
visibility.count

#Correlation plot
cor.dt <- subset(accident.data,select = -c(ID,Source,End_Lat,End_Lng,Description,Number))
cor.dt <- subset(cor.dt,select = c(TMC,Severity,Start_Lat,Start_Lng,Distance.mi.,Temperature.F.,Wind_Chill.F.,Humidity...,Pressure.in.,Visibility.mi.,Wind_Speed.mph.,Precipitation.in.))
```

```r
cor.dt <- na.locf(cor.dt)
head(cor.dt)
cor.rst <- cor(cor.dt, method="spearman")
View(cor.rst)
rownames(cor.rst) <- c("Temperature(F)","Wind Chill(F)","Humidity(%)","Pressure(in)","Visibility(miles)","Wind Speed(mph)","Precipitation(in)","Crossing","Junction","Traffic Signal","Delay")
colnames(cor.rst) <- rownames(cor.rst)
corrplot(cor.rst, type="lower", tl.col="black", tl.srt=30, method="color", tl.cex=0.8, diag=TRUE, tl.pos="ld")

#Percentage of accidents over time of the day
time.dt <- subset(accident.data,select = "Start_Time")
time.dt$Start_Time <- hour(time.dt$Start_Time)
head(time.dt)

time.count <- data.table(table(time.dt))
names(time.count)[1] <- "hour"
names(time.count)[2] <- "count"
time.count$hour <- as.numeric(time.count$hour)
time.count$count <- time.count$count / (sum(time.count$count))
time.count

time.plot.day <- ggplot(time.count,aes(x = hour,y = count * 100,group = 1)) + geom_point(color = "#761dbe",shape = 17,size = 3)
              + geom_line(size = 1,color = "#ece0da") + scale_y_continuous(labels = comma) + ylab("Percentage")
time.plot.day <- time.plot.day + theme_minimal() + geom_hline(yintercept = 4.65,linetype = "dashed",color = "steelblue")
              + labs(title = "Accident percentage over time of a day")
              + theme(plot.title = element_text(hjust = 0.5,color = "#666666"),axis.title = element_text(color = "#666666"))
time.plot.day
```

7 Appendix:Model Training Code List

```r
workspace <- "~/Downloads/R/us_traffic_accidents"
setwd(workspace)

#Read in the data file
accident.data.ori <- read.csv("./US_Accidents_Dec19.csv", stringsAsFactors = FALSE)
accident.data <- accident.data.ori
View(accident.data)

#Convert the values to proper types
accident.data$Start_Time <- parse_date_time(accident.data$Start_Time,"ymd HMS")
accident.data$End_Time <- parse_date_time(accident.data$Start_Time,"ymd HMS")
accident.data$Weather_Timestamp <- parse_date_time(accident.data$Weather_Timestamp,"ymd HMS")
accident.data$Amenity <- as.logical(accident.data$Amenity)
accident.data$Crossing <- as.logical(accident.data$Crossing)
accident.data$Junction <- as.logical(accident.data$Junction)
accident.data$Traffic_Signal <- as.logical(accident.data$Traffic_Signal)
ori.names <- colnames(accident.data)

```

```r
#Choose part of the columns
demandedrow <- c(ori.names[4],ori.names[5],ori.names[7],ori.names[8],ori.names[11],ori.names[23],
                 ori.names[24],ori.names[25],ori.names[26],ori.names[27],ori.names[28],ori.names[30],ori.names[31],ori.names[32],ori.names[35],ori.names[37],ori.names[44])
accident.data.clean <- subset(accident.data,select = demandedrow)
accident.data.clean <- subset(accident.data.clean, Start_Time > as.Date("2017-9-1"))

#Visualize missing values
tiff("missing.tiff",height = 600*5 ,width = 600*5,res = 500,compression = "lzw")
vis_miss(accident.data.clean,warn_large_data = FALSE)
dev.off()
#Fill NAs
accident.data.clean$Wind_Chill.F. <- na.fill(accident.data.clean$Wind_Chill.F.,fill = "extend")
accident.data.clean$Wind_Speed.mph. <- na.fill(accident.data.clean$Wind_Speed.mph.,fill = "extend")
accident.data.clean$Precipitation.in. <- na.fill(accident.data.clean$Precipitation.in.,fill = "extend")
accident.data.clean$Temperature.F. <- na.fill(accident.data.clean$Temperature.F.,fill = "extend")
accident.data.clean$Visibility.mi. <- na.fill(accident.data.clean$Visibility.mi.,fill = "extend")
accident.data.clean$Humidity... <- na.fill(accident.data.clean$Humidity...,fill = "extend")
accident.data.clean$Pressure.in. <- na.fill(accident.data.clean$Pressure.in.,fill = "extend")

accident.data.clean.back <- accident.data.clean

#Function to map the severity to delay time
accident.data.clean$Delay <- 0
severity2time <- function(data){
  class <- data["Severity"]
  if(class == 1){
    #The class 1
    out <- floor(runif(1,0,150))
  }
  else if(class == 2){
    out <- floor(runif(1,150,480))
  }
  else if(class == 3){
   out <- floor(runif(1,480,1080))
  }
  else if(class == 4){
    out <- floor(runif(1,1080,1800))
  }
  return(out)
}

accident.data.clean$Delay <- apply(accident.data.clean,1,FUN = severity2time)
#accident.data.clean$Weather_Condition <- apply(accident.data.clean,1,FUN = condition2level)
accident.data.clean.back <- accident.data.clean
#Discard Weather type column
```

```
accident.data.clean <- accident.data.clean[,-14]

#Get the correlation matrix
cor.mtx <- cor(accident.data.clean[,7:17],method = "spearman")
View(cor.mtx)
rownames(cor.mtx) <- c("Temperature(F)","Wind Chill(F)","Humidity(%)","Pressure(in)","Visibility(miles)","Wind Speed(mph)","Precipitation(in)","Crossing","Junction","Traffic Signal","Delay")
colnames(cor.mtx) <- rownames(cor.mtx)
corrplot(cor.mtx, type="lower", tl.col="black", tl.srt=30, method="color", tl.cex=0.8, diag=TRUE, tl.pos="ld")

weathercol <- accident.data.clean$Weather_Condition

#Get the trainning set and validation set
dt.v <- accident.data.clean
train_index <- sample(1:nrow(dt.v), 0.8 * nrow(dt.v))
test_index <- setdiff(1:nrow(dt.v), train_index)

accident.data.train.lm <- dt.v[train_index,]
accident.data.test.lm <- dt.v[test_index,]
#The data is incomplete before 2017-9-1
lm_model_1 <- lm(Delay ~ Temperature.F.+ Wind_Chill.F.:Wind_Speed.mph. + Pressure.in.+ Precipitation.in. + Humidity...+ Visibility.mi.+ Crossing+Junction+Traffic_Signal
                 ,data = accident.data.train.lm)
lm_model_2 <- lm(Delay ~ Temperature.F.+ Wind_Chill.F.:Wind_Speed.mph. + Pressure.in.+ Precipitation.in. + Humidity...+Crossing+Junction+Traffic_Signal
                   ,data = accident.data.train.lm)

summary(gvlma(lm_model_2))
confint(lm_model_2)
outlierTest(lm_model_2)
accident.data.clean <- accident.data.clean[-1482314,]
vif(lm_model)
#Make prediction
lm.pred.1 <- predict(lm_model_1,accident.data.test.lm)
lm.pred.2 <- predict(lm_model_2,accident.data.test.lm)

#Make the Chi-square test
df.pred <- data.frame(lm.pred.1,lm.pred.2)

#Cut the size to 50000 for the chisqure test size limit
df.pred <- df.pred[1:5000,]
chisq.test(df.pred$lm.pred.1,df.pred$lm.pred.2)

#Prediction model for time we use the weather timestamp as the start point of prediction
accident.data.clean$TimeGap <- 0
accident.data.clean$TimeGap <- abs(accident.data.clean$Start_Time - accident.data.clean$Weather_Timestamp)
accident.data.clean$TimeGap <- as.numeric(accident.data.clean$TimeGap)
accident.data.clean$TimeGap <- na.fill(accident.data.clean$TimeGap,fill = "extend")

#We study accident with severity 3-4 only
df.svt <- subset(accident.data.clean, Severity > 2)
#Get the training set through random selection
```

```
train_index <- sample(1:nrow(df.svt), 0.8 * nrow(df.svt))
test_index <- setdiff(1:nrow(df.svt), train_index)

accident.data.train <- df.svt[train_index,]
accident.data.test <- df.svt[test_index,]

cor(accident.data.clean$Temperature.F.,accident.data.clean$TimeGap)

accident.train <- subset(accident.data.train, TimeGap <= 1700)
accident.train <- accident.train[1:50000,]
accident.test <- accident.data.test[1:5000,]

diff2bin <- function(data){
  gap <- as.numeric(data["TimeGap"])
  if(gap <= 300){
    bin <-  1
  }
  else if((gap > 300) && (gap <= 600)){
    bin <- 2
  }
  else {
    bin <- 3
  }
  return(bin)
}
accident.train$bin <- apply(accident.train,1,FUN = diff2bin)
accident.test$bin <- apply(accident.test,1,FUN = diff2bin)
accident.train.back <- accident.train
accident.test.back <- accident.test
accident.test <- accident.test[,-2]
accident.test <- accident.test[,-5]
table(accident.train$bin)
accident.train$bin <- as.factor(accident.train$bin)
accident.test$bin <- as.factor(accident.test$bin)
time_model <- randomForest(bin ~ Temperature.F.+ Wind_Chill.F.+Wind_
    Speed.mph. +Pressure.in.+ Precipitation.in. + Humidity...+ Visibility
    .mi.+Crossing+Junction+Traffic_Signal
                    ,data = accident.train)
summary(time_model)
print(time_model)
plot(time_model)

tm.pred <- predict(time_model,accident.test,type = "prob")
perf <- table(accident.test$bin,tm.pred,dnn = c("Actual","Predicted"))
perf
multiclass.roc(accident.test$bin,tm.pred)

importance(time_model)
```

End of Appendix

References

[1] https://www.statista.com/statistics/183505/number-of-vehicles-in-the-united-states-since-1990/

[2] https://www.thewanderingrv.com/car-accident-statistics/

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

[4] 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.

[5] https://www.statista.com/statistics/859950/vehicles-in-operation-by-quarter-united-states/

[6] H. Wickham. ggplot2: Elegant Graphics for Data Analysis. Springer-Verlag New York, 2016.

[7] R Core Team (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL https://www.R-project.org/.

[8] Garrett Grolemund, Hadley Wickham (2011). Dates and Times Made Easy with lubridate. Journal of Statistical Software, 40(3), 1-25. URL http://www.jstatsoft.org/v40/i03/.

[9] Nicholas Tierney, Di Cook, Miles McBain and Colin Fay (2020). naniar: Data Structures, Summaries, and Visualisations for Missing Data. R package version 0.5.1. https://CRAN.R-project.org/package=naniar

[10] Achim Zeileis and Gabor Grothendieck (2005). zoo: S3 Infrastructure for Regular and Irregular Time Series. Journal of Statistical Software, 14(6), 1-27. doi:10.18637/jss.v014.i06

[11] Taiyun Wei and Viliam Simko (2017). R package "corrplot": Visualization of a Correlation Matrix (Version 0.84). Available from https://github.com/taiyun/corrplot

[12] https://en.wikipedia.org/wiki/Spearman%27s_rank_correlation_coefficient

[13] Robert Kabacoff. 2015. R in Action: Data Analysis and Graphics with R. Manning Publications Co., USA.

[14] A. Liaw and M. Wiener (2002). Classification and Regression by randomForest. R News 2(3), 18–22.

[15] Xavier Robin, Natacha Turck, Alexandre Hainard, Natalia Tiberti,Frédérique Lisacek, Jean-Charles Sanchez and Markus Müller (2011).pROC: an open-source package for R and S+ to analyze and compare ROCcurves.