Road Safety - Group 4

Mar 2019

Project Overview

This project tried to identify the features that are predictive regarding the severity of traffic Accidents in United Kingdom (2012). The data set we use contains information about traffic accidents and the condition on sites. This report completes the following aspects:

1. Preparation

- load the data set
- load the required packages (Please comment out the install part to download them)

2. Data processing and Preliminary Analysis of the data set

- Upsampling data set
- Preliminary Feature Selection

3. Feature Selection

- Correlation Analysis with Chi-square test
- Information gain
- Forward step searching based on AIC
- Random Forest

4. Feature Visualization

5. Predicting Accident Severity

- Random Forest
- SVM
- Logistic Regression
- Naive Bayes

6.Conclusion and Recommendations

Preparation Stage

Load the Required Package

Please comment out the code as needed.

```
# install.packages("ggplot2")
# install.packages("lubridate")
# install.packages("devtools")
# install.packages("knitr")
# install.packages("caret")
# install.packages("plyr")
# install.packages("dplyr")
# install.packages("kableExtra")
# install.packages("mlbench")
# install.packages("e1071")
# install.packages("randomForest")
# install.packages("fastDummies")
# install.packages("bestglm")
# install.packages("leaps")
#install.packages("klaR")
#install.packages("AppliedPredictiveModeling")
# install.packages("MASS")

require(devtools)
require(ggplot2)
require(lubridate)
install_github("dkahle/ggmap")
library(ggmap)
register_google(key = "AIzaSyCCWzW2nOzzBl03FokcZKTclevanlDCfq8") # this is the API key to create
the heatmap
require(knitr)
require(caret)
require(plyr)
require(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.5.2
```

```
library(kableExtra)
```

```
## Warning: package 'kableExtra' was built under R version 3.5.2
```

```
require(mlbench)
require(e1071)
```

```
## Warning: package 'e1071' was built under R version 3.5.2
```

```
require(leaps)
require(bestglm)
```

```
## Warning in library(package, lib.loc = lib.loc, character.only = TRUE,
## logical.return = TRUE, : there is no package called 'bestglm'
```

```r
require(fastDummies)
library(randomForest)
library(AppliedPredictiveModeling)
library(klaR)
require(MASS)
```

Load the Dataset

```r
# Load the orginal dataset and the recoded dataset
raw.road <- read.csv("DfTRoadSafety_Accidents_2012.csv")
road<-read.csv("clean_original.csv")
```

Data Processing and Preliminary Analysis

Before running analysis, this part completes the following three tasks:

1. Remove all the NAs from the data set. The remove of NAs did not significantly reduce the number of records in the data set and it did not considerably change the distribution of the accident severity.
2. Pre-selecting features that will not be used in the following analysis. During this stage, three types of the features were excluded from the data set: the indicators of police force authority, local authority, road number, and location. We remove these features since they are unique for each individual accident record. For location, as indicated by the heatmap below, most accidents concentrate on big cities, where there are more vehicles and road more accidents. Therefore, Location cannot be used to explain why severe accidents occur.
3. Recoded all the categorical variables to the actual values rather than numbers.

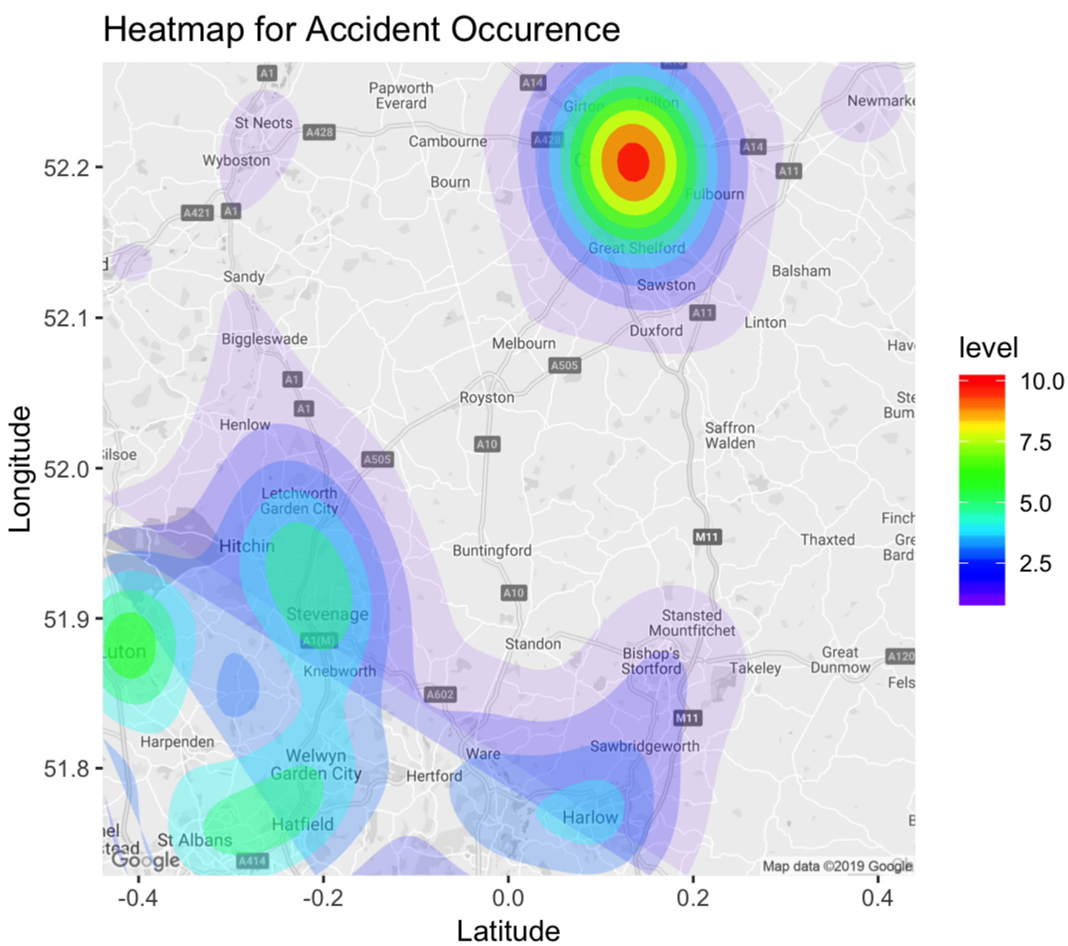
Notes: part of the codes in this section is commented out since the data we load in are already recoded

```r
# remove all the NAs
raw.road <- na.exclude(raw.road)
road<-na.omit(road)
```

As indicated by the heatmap, most accident took place in major cities. There is not significant different in terms of the distribution of the severe and not severe accident.

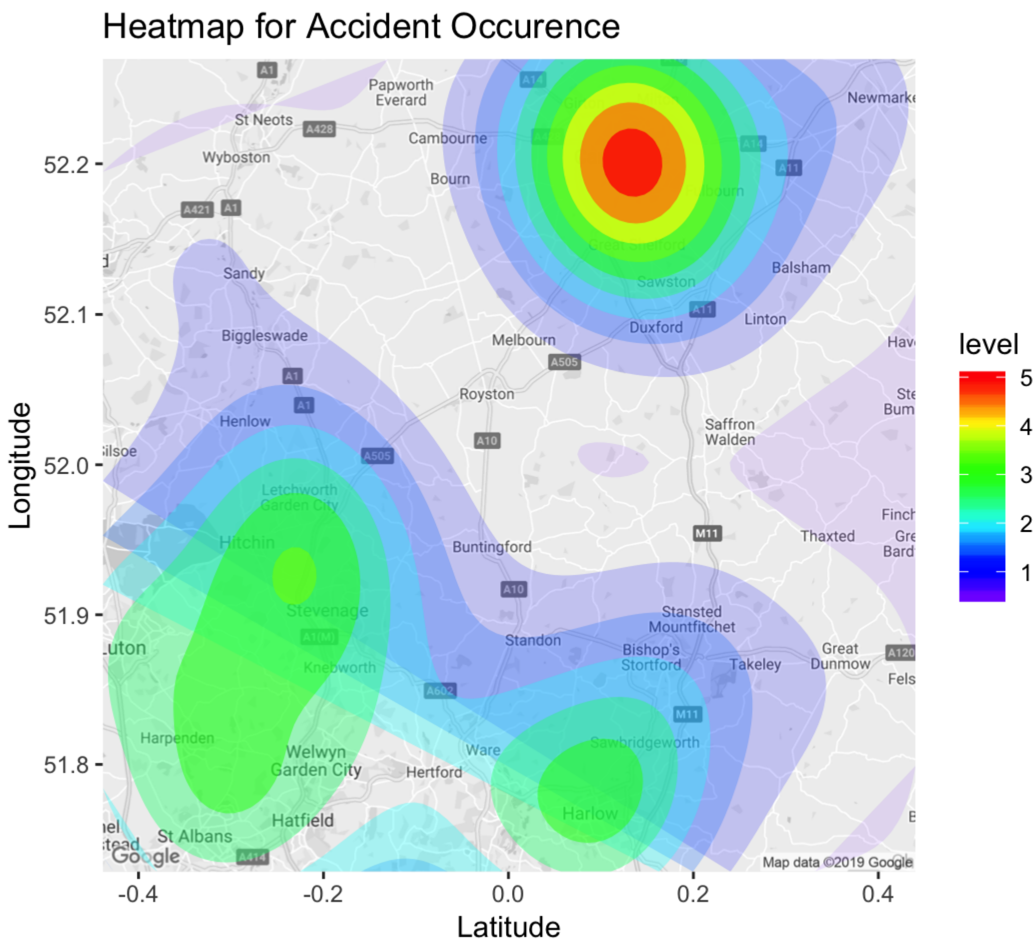
```
# Feature Pre - selection
# create the heatmap showing the geographical distribution of the accident
# define a function to create the heatmap
gg_heatmap <- function(dataset,lon,lat){
  g <- ggmap(get_map(location =c(0,52), zoom=10, color =c("bw"),maptype="roadmap", source="googl
e"))
  g <- g + scale_fill_gradientn(colours=rev(rainbow(100, start=0, end=0.75)))
  g <- g + stat_density2d(data=dataset, aes(x = lon, y = lat,fill = ..level..,alpha=..level..),
                           geom = 'polygon')
  g <- g +   scale_alpha_continuous(guide="none",range=c(.1,.8))
  g <- g+labs(title="Heatmap for Accident Occurence")
  g<-g+labs(x="Latitude")
  g<-g+labs(y="Longitude")
  print (g)
  return(g)
}
accident_map <- gg_heatmap(raw.road, as.numeric(as.character(raw.road$Longitude)),as.numeric(as.
character((raw.road$Latitude))))
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=52,0&zoom=10&size=640x640&scal
e=2&maptype=roadmap&language=en-EN&key=xxx
```



```
raw.road.severe <- subset(raw.road, raw.road$Accident_Severity !="3")
accident_map_severe <- gg_heatmap(raw.road.severe, as.numeric(as.character(raw.road.severe$Longi
tude)),as.numeric(as.character((raw.road.severe$Latitude))))
```

```
## Source : https://maps.googleapis.com/maps/api/staticmap?center=52,0&zoom=10&size=640x640&scal
e=2&maptype=roadmap&language=en-EN&key=xxx
```




```
# select the columns and recoded the dataset
sub_road <- raw.road[,c(1,10,11,12,15,17,18,19,20,21,
                        23,24,25,26,27,28,29,30,31)]

# with the data dictionary we reoded all the variables with a loop

# read the variable dictionary into r, this gives us all value - category pair
# The following chunk is commented out since the data we load in has already been recoded
# for( i in c(3,5,6,8,9,10,11,12,13,14,15,16,17,18,19)){
# dict<-read.csv("variable_dictionary.csv") # this is a data dictionary we created for pair up t
he values
# dict_sub<-subset(dict, dict$var == i)
# coded<-c()
# for(n in(1:nrow(sub_road))){
#       coded[n]<-as.character((dict_sub$label[which(dict_sub$code== sub_road[n,i])]))}
# sub_road[,i] <- coded }
```

```
# This part deals with date and time related
# # generate indicator for each hour
# sub_road<- separate(sub_road,Time,into = c("hour","minitue"), sep = ":")
# sub_road<- sub_road[,-5]
#
# # create the indicator of month
# sub_road<-separate(sub_road, Date, into = c("Year","Month","Day"),sep = "/")
# sub_road<-sub_road[,-c(2,4)]
```

Exploring the Dataset

As shown in the following result, the data set we have contains 84628 observations and 23 columns. However, it is worth noticing that in terms of the distribution of the response variable:

1. The distribution of the severity level is so unbalanced that less than 10% of the record are actually highly severe, which is the output level we are interested in.
2. We will not have enough variability for feature selection and model training.

Low	Medium	High
86.2%	13.1%	7.1%

To address the problem, we have two potential solutions:

- Upsampling both the low and medium level. However, this impose too strong assumption on the prior probability of different traffic accident, in reality, we won’t expect accident with different severity have same likelihood. Also, it further increases the size of the data set.
- Combining the Medium and High levels “Severe” and treat “low” as not severe and upsample the severe level. Not too strong assumption on the distribution of different accident.

Therefore, we combine the low and medium level and upsample the severe category. After the transformation, we have 75,517 rows for each category.

```
# Basic infomation
dim(road)
```

```
## [1] 84628    23
```

```
#The distribution of the output variables
kable(with(road,prop.table(table(Severity))), digit =3)
```

Severity	Freq
high	0.007
low	0.862
medium	0.131

```
# combine the high and meduim and upsample
road$Severity<-as.factor(ifelse(road$Severity == "low", "Not_Severe","Severe"))
road.balanced <- upSample(road,road$Severity,yname = "Class") # this would be used for the follo
wing analysis
road.balanced<- road.balanced[,-which(names(road.balanced) %in% c("X","Vehicles","Casualties","C
lass","Accident_Index"))]
```

Feature Selection and Visualization

This part of the analysis completes feature selection and visualization. For feature selection, we use correlation analysis, information gain, step AIC score and random forest models.

** Correlation Analysis Approach ** Before correlation analysis, feature engineering was implemented to reduce the levels of the categorical variables. We combined certain levels of certain categorical variable that are similar in nature and have only limited observations in our dataset.

```
# Combined levels
road.balanced$X1st_Road_Class<- mapvalues(road.balanced$X1st_Road_Class, from = levels(road.bala
nced$X1st_Road_Class),
                                      to = c("A","A","B","C","Motorway","Unclassified"))
road.balanced$X2nd_Road_Class<- mapvalues(road.balanced$X2nd_Road_Class, from = levels(road.bala
nced$X2nd_Road_Class),
                                      to = c("A","A","B","C","Motorway","Unclassified"))

road.balanced$Special_Conditions_at_Site<-mapvalues(road.balanced$Special_Conditions_at_Site,
                                                    from = levels(road.balanced$Special_Conditio
ns_at_Site),
                                                    to =c("Auto signal defective","Auto signal d
efective","Mud", "None","Oil or
                                                           diesel","Road sign or marking defecti
ve or obscured","Road surface
                                                           defective","Roadworks"))
road.balanced$Carriageway_Hazards<-mapvalues(road.balanced$Carriageway_Hazards, from = levels(ro
ad.balanced$Carriageway_Hazards),
                                             to = c("Objects in Carriageway","None","Objects on
 Road",
                                                    "Objects in Carriageway","Previous Accident"
,"Objects on Road"))
road.balanced$Did_Police_Officer_Attend_Scene_of_Accident<- mapvalues(road.balanced$Did_Police_O
fficer_Attend_Scene_of_Accident,
                                                                      levels(road.balanced$Did_P
olice_Officer_Attend_Scene_of_Accident),
                                                                      to=c("No","Yes","No"))
#write.csv(road.balanced, file = "balanced_2level.csv")
```

```
road$X1st_Road_Class<- mapvalues(road$X1st_Road_Class, from = levels(road$X1st_Road_Class),
                                 to = c("A","A","B","C","Motorway","Unclassified"))
road$X2nd_Road_Class<- mapvalues(road$X2nd_Road_Class, from = levels(road$X2nd_Road_Class),
                                 to = c("A","A","B","C","Motorway","Unclassified"))

road$Special_Conditions_at_Site<-mapvalues(road$Special_Conditions_at_Site,
                                                    from = levels(road$Special_Conditions_at_Sit
e),
                                                    to =c("Auto signal defective","Auto signal d
efective","Mud", "None","Oil or
                                                          diesel","Road sign or marking defecti
ve or obscured","Road surface
                                                          defective","Roadworks"))
road$Carriageway_Hazards<-mapvalues(road$Carriageway_Hazards, from = levels(road$Carriageway_Haz
ards),
                                              to = c("Objects in Carriageway","None","Objects on
 Road",
                                                    "Objects in Carriageway","Previous Accident"
,"Objects on Road"))
road$Did_Police_Officer_Attend_Scene_of_Accident<- mapvalues(road$Did_Police_Officer_Attend_Scen
e_of_Accident,
                                                                      levels(road$Did_Police_Of
ficer_Attend_Scene_of_Accident),
                                                                     to=c("No","Yes","No"))
```

To analyze the correlation of two categorical variables, this project uses chi-square test. The function takes the frequency table between two variables and calculate x^2 statistics and the corresponding p_value. The following loop with return the combinations that is correlated

Based on the result, “Pedestrian_Crossing.human_control” turns out to be correlated with several other variables. Additionally, the levels of this variable are None, other, school , which is not very meaningful. Therefore, we think it is reasonable to remove this column from the dataset.

```
# we use a loop to populate the frequency table and calculate the chi-square test
chi.result <- c()
for(i in c(1:ncol(road.balanced))){
  for(n in c(1:ncol(road.balanced))){
    result <- table(road.balanced[,i],road.balanced[,n])
    if(chisq.test(result)$p.value > 0.05)
    { chi.result <-c(chi.result,paste(names(road.balanced[c(i,n)])))}
  }}

chi.result

# remove the column
road.balanced<-road.balanced[,-which(names(road.balanced) =="Pedestrian_Crossing.Human_Control"
)]
```

Information Gain Approach

This part first define the function to caluate the information gain of each individual features on the severity of traffic accident.

```
# define the function to calculate information gain
entropy <- function(x) {
  H <- 0
  freq.x <- as.data.frame(table(x))$Freq

  if (sum(freq.x) ==0){
    H <- 0 # Case shows up when calculating conditional entropies
    return (H)
  }
  p <- freq.x/sum(freq.x)
  p <- p[p > 0] # Discard zero entries (because 0 log 0 = 0)

  # *********** Edit me ***********
  H=0
  for (i in c(1:length(p))){
    prob <- p[i]
    H= H + (-prob*log2(prob))

  }
  # *********** End EDIT ***********

  return(H)
}

info.gain <- function(x,y){
  IG <- 0
  Hyx<-0
  # get the entropy of y alone
  Hy<-entropy(y)
  # get the condition entropy of y given x

  for(i in levels(factor(x))){
    px<-sum(x==i)/length(x) # p(x=i)
    Hyx<-Hyx + entropy(y[which(x==i)])*px
  }

  IG <-Hy-Hyx
  return(IG)
}
```

In the output, the highlighted rows represent the important features for predicting the severity of the accident. We draw a red line to separate the most predictive features as we identified a gap in information gains.

```
# calculate infomation gain of each individual variable
# remove all the response variables
road.info <- road.balanced

# calculate the infomation gain of each of the variables
infomation.gain <- c()

for (i in (1:17)){
  x <- road.info[,i]
  y <- road.info$Severity
  infomation.gain[i]<-round(info.gain(x,y),5)}

result<-data.frame(Var=names(road.info[(1:17)]),Information.gain = infomation.gain )

# We sort the date from the highest infomation gain from the lowest
result<-arrange(result,desc(Information.gain))

# formate the output table
kable(result,caption = "Infomation Gain Result") %>%
  kable_styling("striped", full_width = F) %>%
  row_spec(1:7, bold = T, color = "white", background = "#D7D3D4")
```

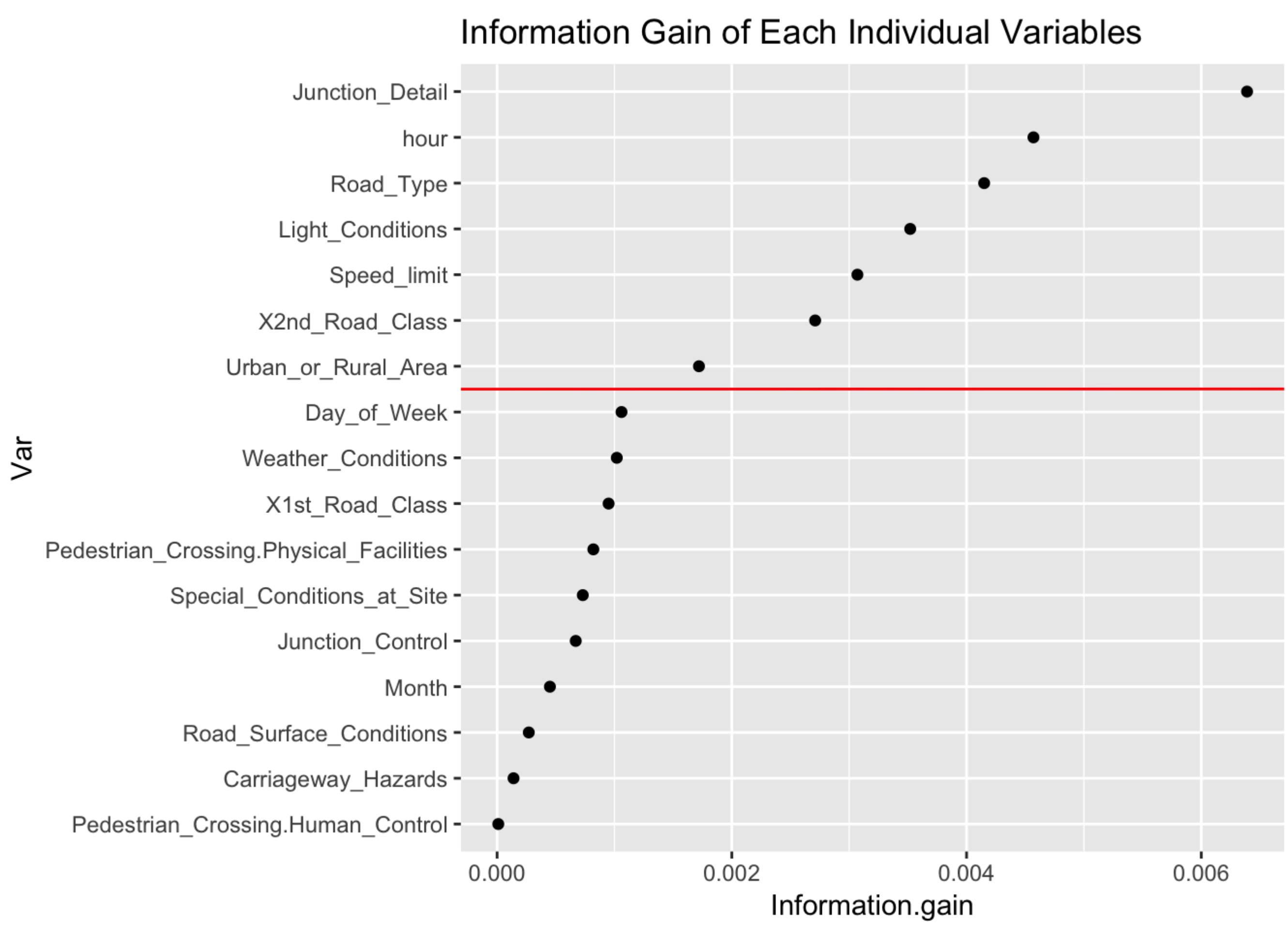
Infomation Gain Result

Var	Information.gain
Junction_Detail	**0.00639**
hour	**0.00457**
Road_Type	**0.00415**
Light_Conditions	**0.00352**
Speed_limit	**0.00307**
X2nd_Road_Class	**0.00271**
Urban_or_Rural_Area	**0.00172**
Day_of_Week	0.00106
Weather_Conditions	0.00102
X1st_Road_Class	0.00095
Pedestrian_Crossing.Physical_Facilities	0.00082
Special_Conditions_at_Site	0.00073
Junction_Control	0.00067

Var	Information.gain
Month	0.00045
Road_Surface_Conditions	0.00027
Carriageway_Hazards	0.00014
Pedestrian_Crossing.Human_Control	0.00001

```
# to better illustrate the difference, we also create a plot
result$Var<-reorder(result$Var,result$Information.gain, mean)
ggplot(data = result, mapping = aes(y = Var,x=Information.gain))+geom_point()+geom_line()+labs(t
itle = "Information Gain of Each Individual Variables") + geom_abline(intercept = 10.5,color =
"red")
```

```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```



Feature selection based on foward stepwise selection (Logistic Regression)

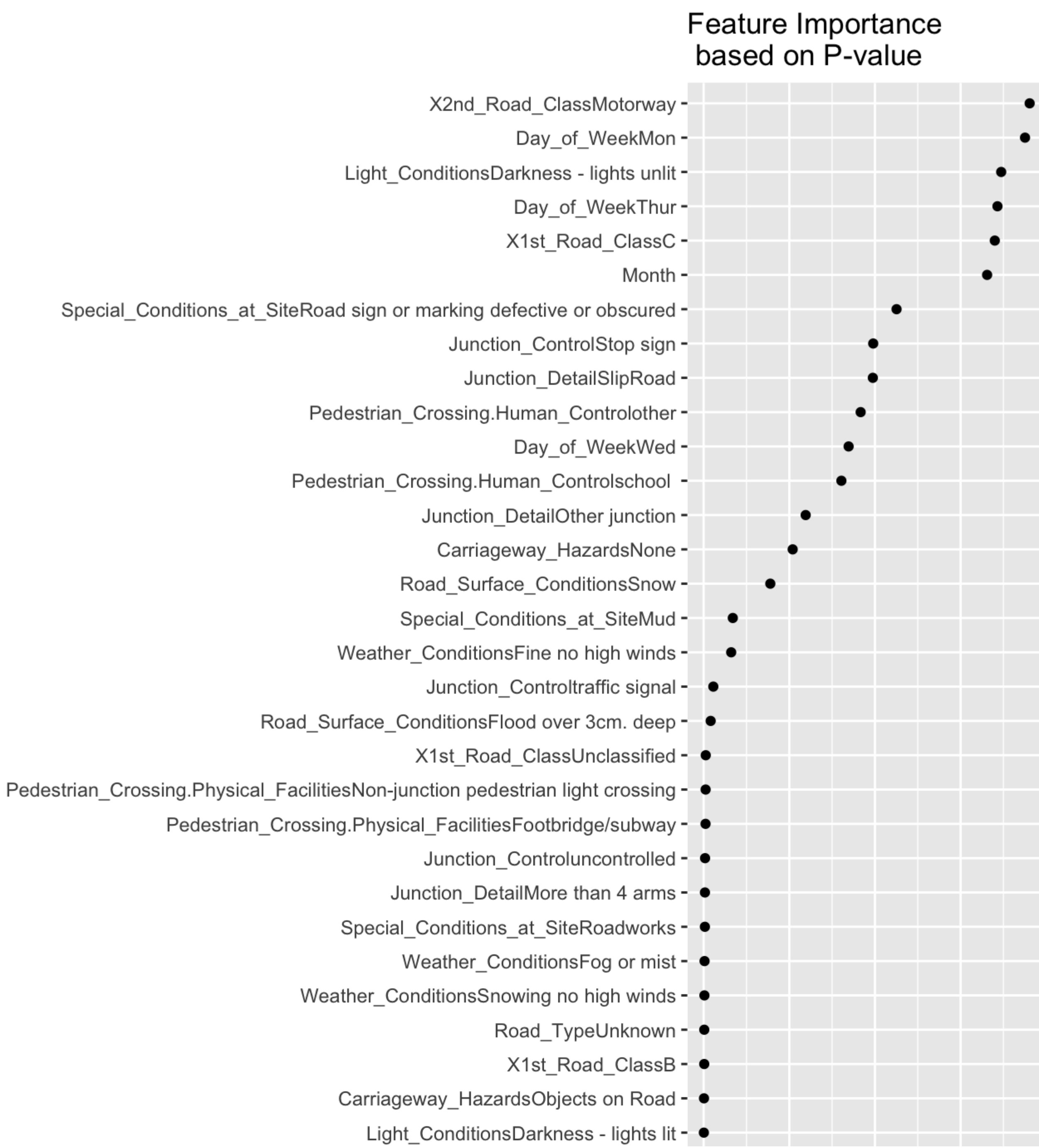
The third approach is forward step wise searching with the stepAIC function. This provide us an easy and interpretable way to analyze feature importance. The regsubset() function covered during class is not applicable here since the leaps algorithm cannot be used for logistic regression with requires maximum likelihood.

The first method we use here is **stepAIC()**, starting from the full model, this function is only able to remove one variable “month” to improve the AIC.

source: http://www2.uaem.mx/r-mirror/web/packages/bestglm/vignettes/bestglm.pdf (http://www2.uaem.mx/r-mirror/web/packages/bestglm/vignettes/bestglm.pdf); https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4842399/ (https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4842399/)

```
# visualize the result
full.result$var<-reorder(full.result$var,full.result$p.value, mean)
ggplot(data = full.result, mapping = aes(y=var,x=p.value)) + geom_point()+geom_line()+labs(title
= "Feature Importance \n based on P-value")
```

```
## geom_path: Each group consists of only one observation. Do you need to
## adjust the group aesthetic?
```




```
# get step AIC
forward.select <- stepAIC(full,direction = "backward")
forward.select$anova
```

** Feature Selection based on Random Forest **

With Caret package, this part of the project used random foreast for classification and calcualte the contribution of different features

Note: This is only a demo of the method with 1000 observations. The complete result is a little different

```
set.seed(2019)
# Devide the dataset into train set and test set
road.train <- road.balanced
road.train.id<-sample(c(1:nrow(road.balanced)),1000)
road.train<-na.omit(road.balanced[road.train.id,])

# define the control using a random selection function
control <- rfeControl(functions=rfFuncs, method="cv", number=10)

# run the RFE algorithm
rfe.feature <- rfe(road.train[,c(1:16)], road.train[,17], sizes=c(1:10), rfeControl=control)

# Get the most important predictors from the result
predictors(rfe.feature)
plot(rfe.feature, type=c("g", "o"))
```

Variable Visualization

Based on previous analysis, we think the following features could be informative in predicting the severity of traffic accident and we divided them into the following categories:

1. Road related factors: Speed limit, road class, and junction detail
2. External factors: light condition and weather conditions
3. People-related factors: policy attendance
4. Time related factors: hour and day of week
5. Location Related factor: urban or rural areas

After selecting the relevant variables, this part tried to identify some interesting patterns. The challenge for us in this part is that all the features and target in this dataset are categorical variables.

The distribution of time related features

In terms of day of the week, as indicated by the following graph, there is no significant differences in the distribution of severe and not severe group among different time within hour. As expected, we see more accidents and also more severe accidents during the rush hours.

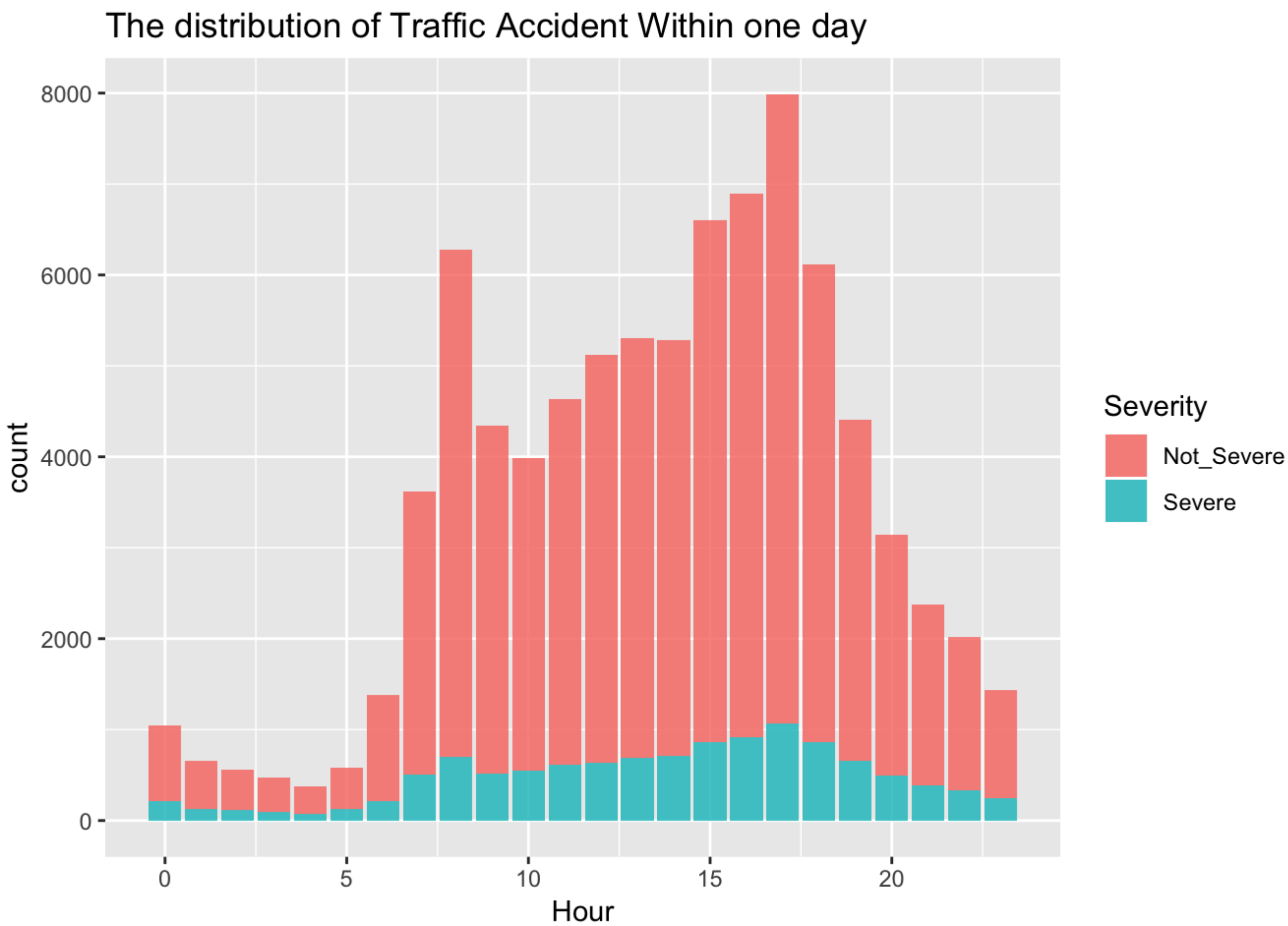
When it comes to the different day of one week, the distribution of accident within one days seem to be different during weekends. More specifically, the number of severe accidents seems to be more uniformly distributed during the day as opposed to have peak numbers at certain time.

```
# load the dataset
road.balanced <-read.csv("balanced_2level.csv")
road.unbalanced <-read.csv("final_unbalanced.csv")

# prepare the dataset
road.balanced<-road.balanced[,-c(1,2,9)]
road.unbalanced<-road.unbalanced[, which(names(road.unbalanced) %in% names(road.balanced))]
```

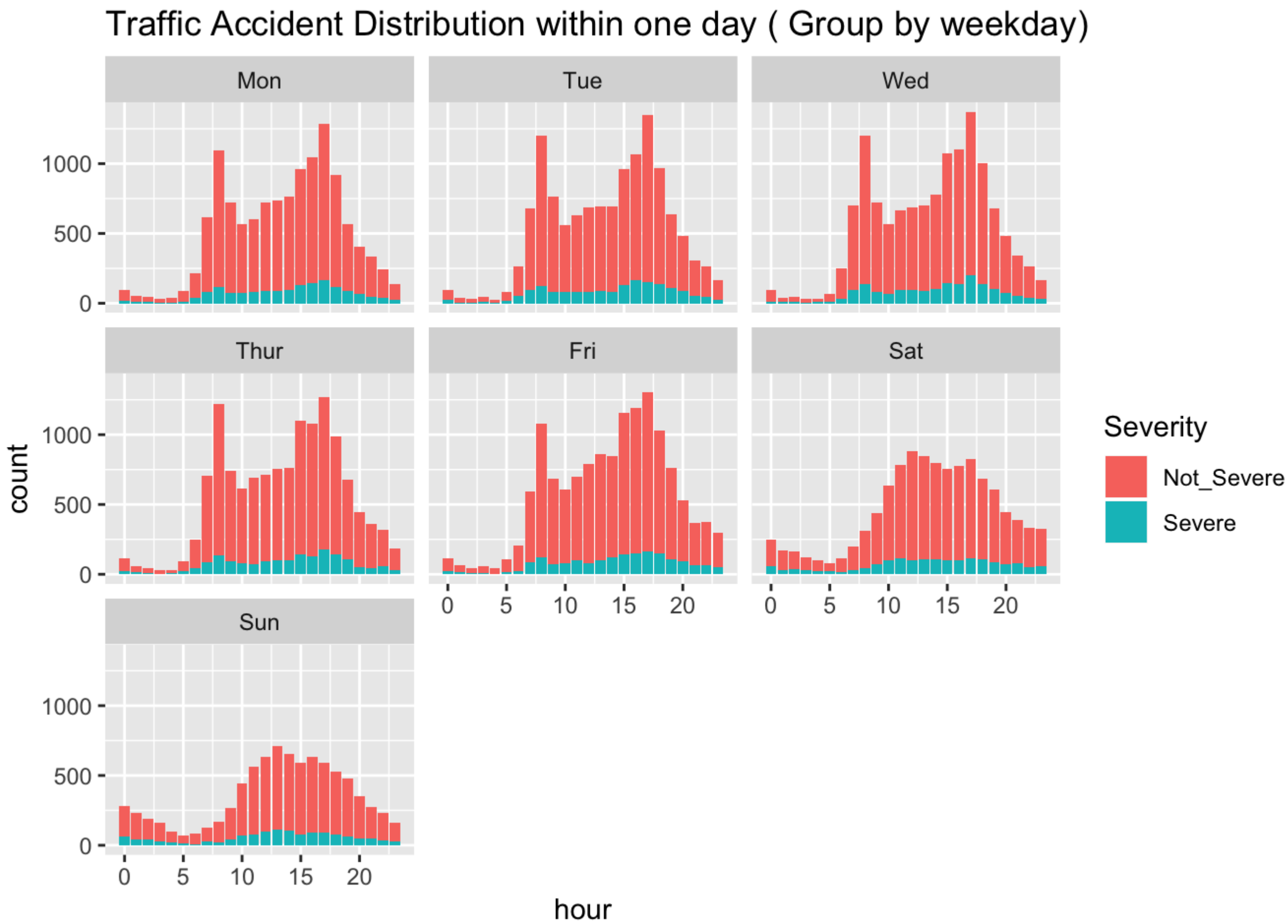
```
# Visualize tims related features
ggplot(data=road.unbalanced, mapping = aes(x=road.unbalanced$hour, fill = Severity))+geom_histog
ram(stat="count", alpha= 0.8) +labs(title = "The distribution of Traffic Accident Within one da
y") + xlab("Hour")
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```




```
# The distribution in different weekday
road.unbalanced$Day_of_Week<-factor(road.unbalanced$Day_of_Week, levels = c("Mon","Tue","Wed","T
hur","Fri","Sat","Sun"))
ggplot(data=road.unbalanced, mapping = aes(x=road.unbalanced$hour, fill = Severity))+geom_histog
ram(stat="count") + facet_wrap('Day_of_Week') + labs( title = "Traffic Accident Distribution wit
hin one day ( Group by weekday)") + xlab("hour")
```

```
## Warning: Ignoring unknown parameters: binwidth, bins, pad
```

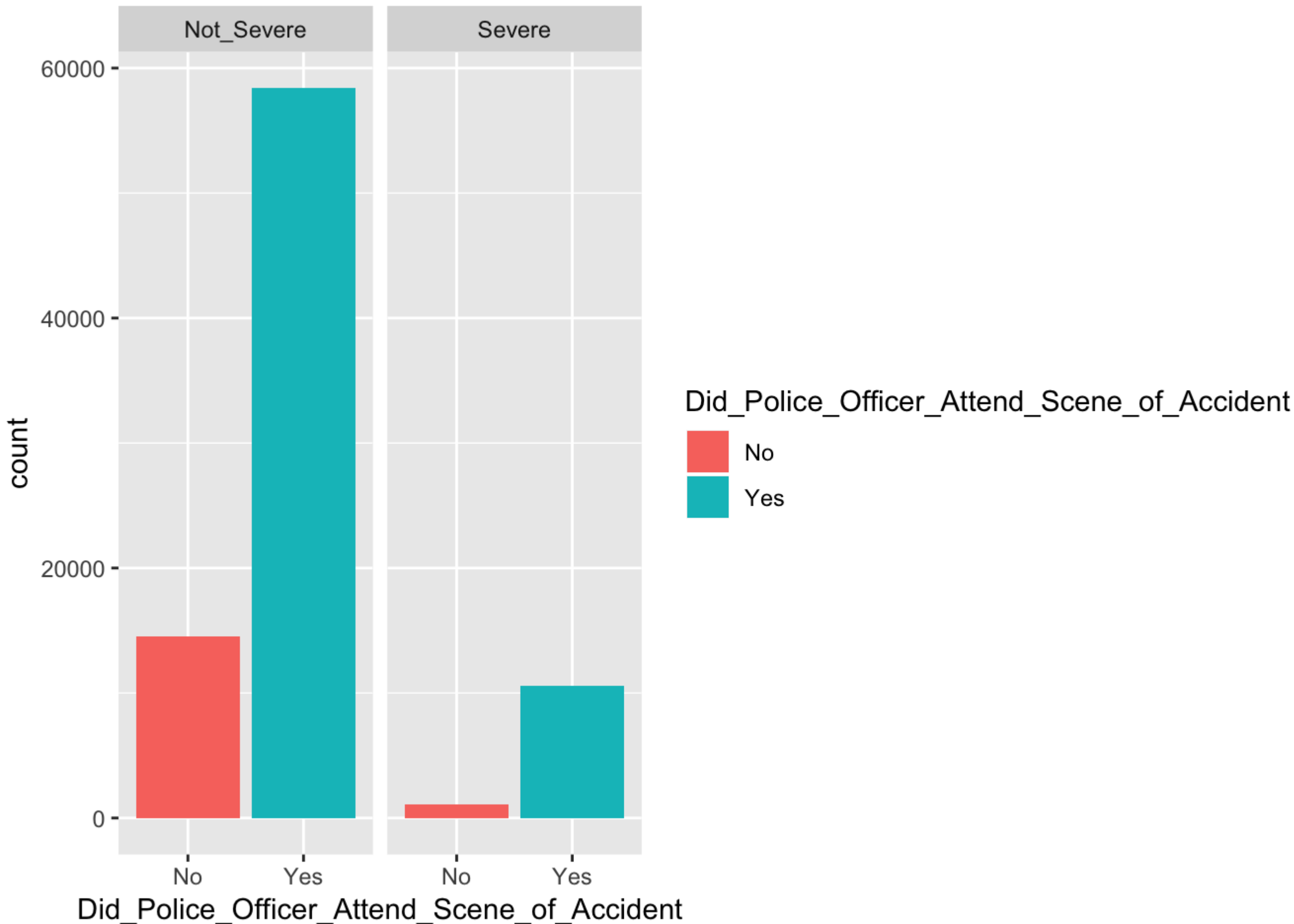


Visulize Police Force For severe traffic accident, when compare it is more likely that police will attend the sences of the accident.

```
require(plyr)
kable((table(road.unbalanced$Severity,road.unbalanced$Did_Police_Officer_Attend_Scene_of_Acciden
t)))
```

	No	Yes
Not_Severe	14552	58376
Severe	1086	10614

```
ggplot(data = road.unbalanced, mapping = aes(x= Did_Police_Officer_Attend_Scene_of_Accident ,fil
l =Did_Police_Officer_Attend_Scene_of_Accident)) + geom_bar( ) + facet_wrap('Severity')
```



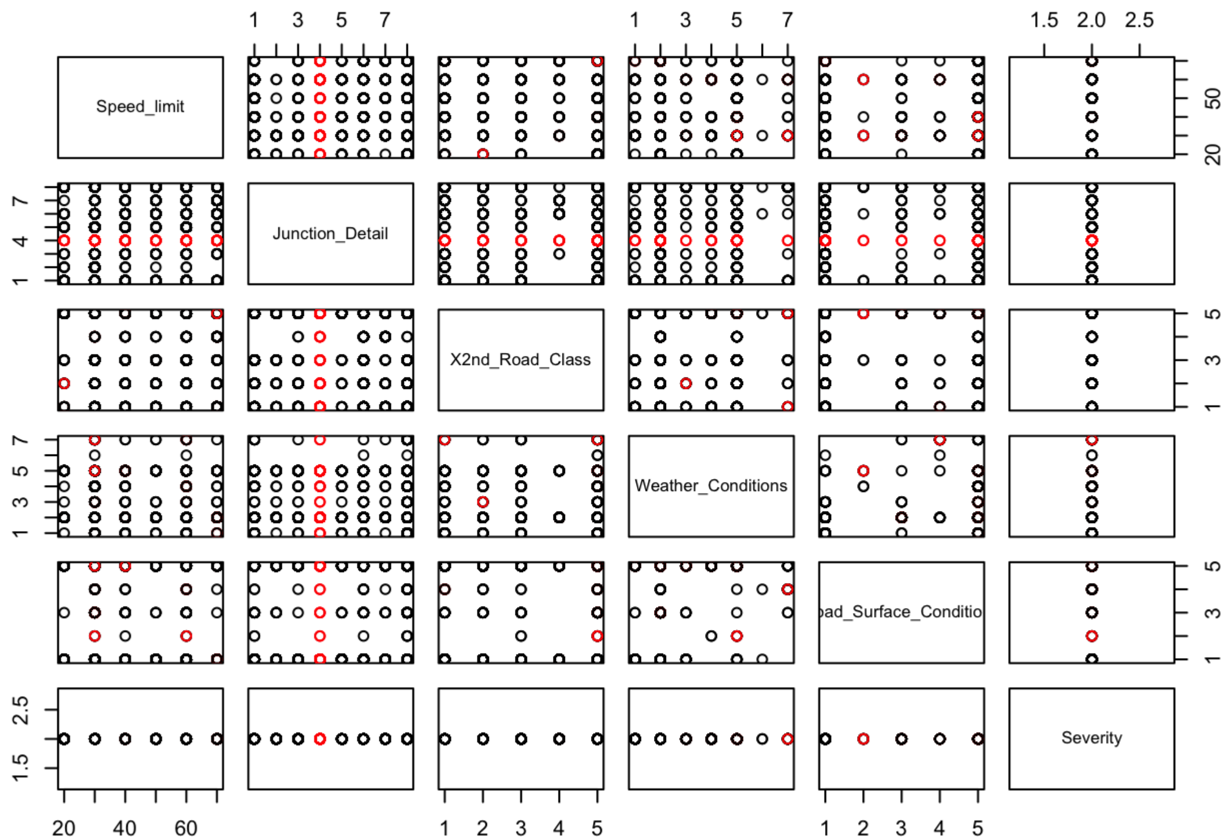
Visualize the clusterred result

The following graph tries to visualize other categorical features with kmode( ) function. This function is for clustering data based on categorical features. Please note that different colors do not represent different severity. Rather, they are the two clusters determined by the algorithms.

```
# select the columns
tst.road<- subset(road.unbalanced, road.unbalanced$Severity == "Severe", select = c(5,6,7,11,12,
17))

# train the model
set.seed(2019)
id<-sample(1:nrow(tst.road))
tst.road<-tst.road[id,]
k.road<-kmodes(tst.road, 2, iter.max = 10, weighted = FALSE, fast = TRUE)

# plot the result
plot(tst.road,col = k.road$cluster)
points(k.road$modes)
```



Predictive Analysis

This part of the report completed the predictive analysis based on the result of feature selection for the project. In this part, we tried to a) find the most predictive model regarding the traffic accident severity and b) confirm our feature selection result

Major classification models including random Forest, naive bayes, and random forest are adopted here.Additionally, svm was also considered for classification task. However, svm uses gradient descent that works best with continuous variables and it assumes that the observations can be divided by a linear hyperplane, which is not applicable for our dataset (with all categorical variables). Therefore, svm was not implemented for this project.

Preparation Stage

```
# load the dataset
road.balanced <-read.csv("balanced_2level.csv")
road.unbalanced <-read.csv("final_unbalanced.csv")

# prepare the dataset
road.balanced<-road.balanced[,-c(1,2,9)]
road.unbalanced<-road.unbalanced[, which(names(road.unbalanced) %in% names(road.balanced))]

# load the package
#install.packages("randomForest")
library(randomForest)
#install.packages("caret")
library(caret)
#install.packages("e1071")
library(e1071)
#install.packages("pROC")
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
#install.packages("knitr")
library(knitr)
library(plyr)
#install.packages("ggplot2")
library(ggplot2)
```

Naive Bayes

The first model we tried is naive bayes. We chose this model for following reasons:

- it takes prior probability into consideration
- it handles categorical variables elegantly
- the algorithm is efficient

However, due to the extremely unbalanced distribution between the severe and not severe categories. The performance of naive bayes model was considerably limited by this prior distribution. As we see a accuracy higher than the default classifier with an extremely low sensitivity, therefore, we don’t think naive bayes is a proper model for predicting severe accident.

Define a function that trains a naive bayes model and do cross validation

```
set.seed(2019)
# define the function
nb.cv<-function(data){
  cv.accuracy <-c()
  cv.sensitivity <-c()
  all.id <- c(1:nrow(data))
      for(i in c(1:10)){
      tst.id<-sample(all.id,1600)
      tst.set <- data[tst.id,]
      trn.set<- data[-tst.id,]
      nb.full.road <-naiveBayes(Severity~., data=trn.set)
      nb.full.pred <-predict(nb.full.road, newdata=tst.set, type = "class")
       result<-data.frame(true = tst.set$Severity, pred.tst = nb.full.pred)
      cv.accuracy <-c(cv.accuracy, mean(result$true == result$pred.tst))
      cv.sensitivity <-c(cv.sensitivity,
       sum(result$true =="Severe"&result$pred.tst=="Severe")/length(which(result$true=="Severe"
)))
      all.id<-all.id[-tst.id]}
  final.result <- data.frame(cv.accuracy, cv.sensitivity)
  return(final.result)}
```

Call the function on the unbalanced dataset

```
nb.cv(road.unbalanced)
```

```
##    cv.accuracy cv.sensitivity
## 1     0.878750    0.000000000
## 2     0.866250    0.004694836
## 3     0.861875    0.000000000
## 4     0.867500    0.009389671
## 5     0.861250    0.000000000
## 6     0.852500    0.008510638
## 7     0.866250    0.004716981
## 8     0.868750    0.009478673
## 9     0.859375    0.004424779
## 10    0.871875    0.004901961
```

Random Forest

Train the random Forest model with the balanced dataset

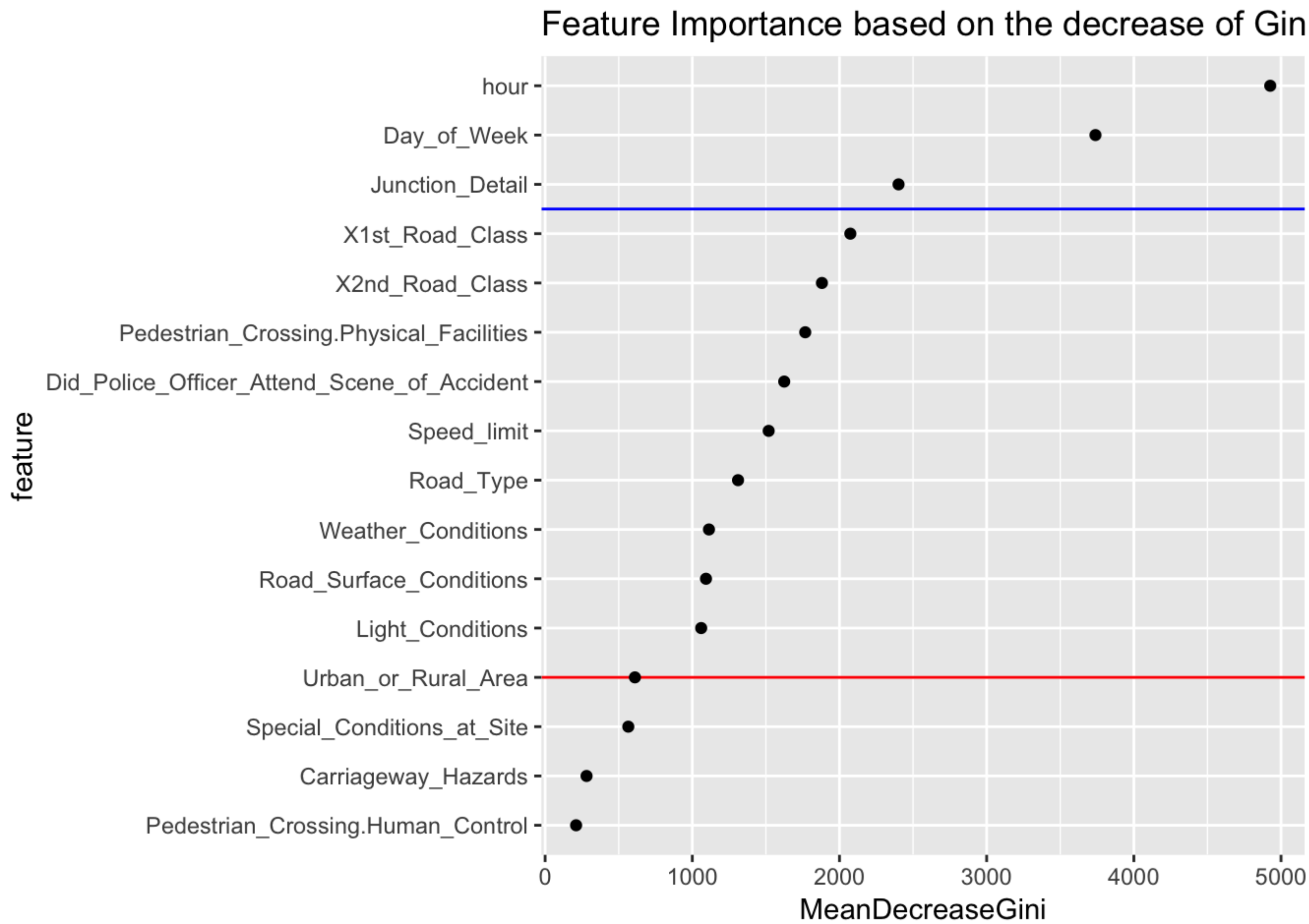
The following part trained an random forest model, tested the model and get feature importance.

```
# train the randome forest model with the balanced dataset]
set.seed(2019)
forest.road.full <- randomForest(Severity ~.,
                                 data = na.omit(road.balanced),
                                 mtry = round(sqrt(ncol(road.balanced))-1),
                                 ntrees = 500)
# get variable importance
importance<- as.data.frame(importance(forest.road.full,type = 2))
importance$feature <- rownames(importance)
rownames(importance)<-c(1:16)
kable(arrange(importance[,c(2,1)],desc(MeanDecreaseGini)))
```

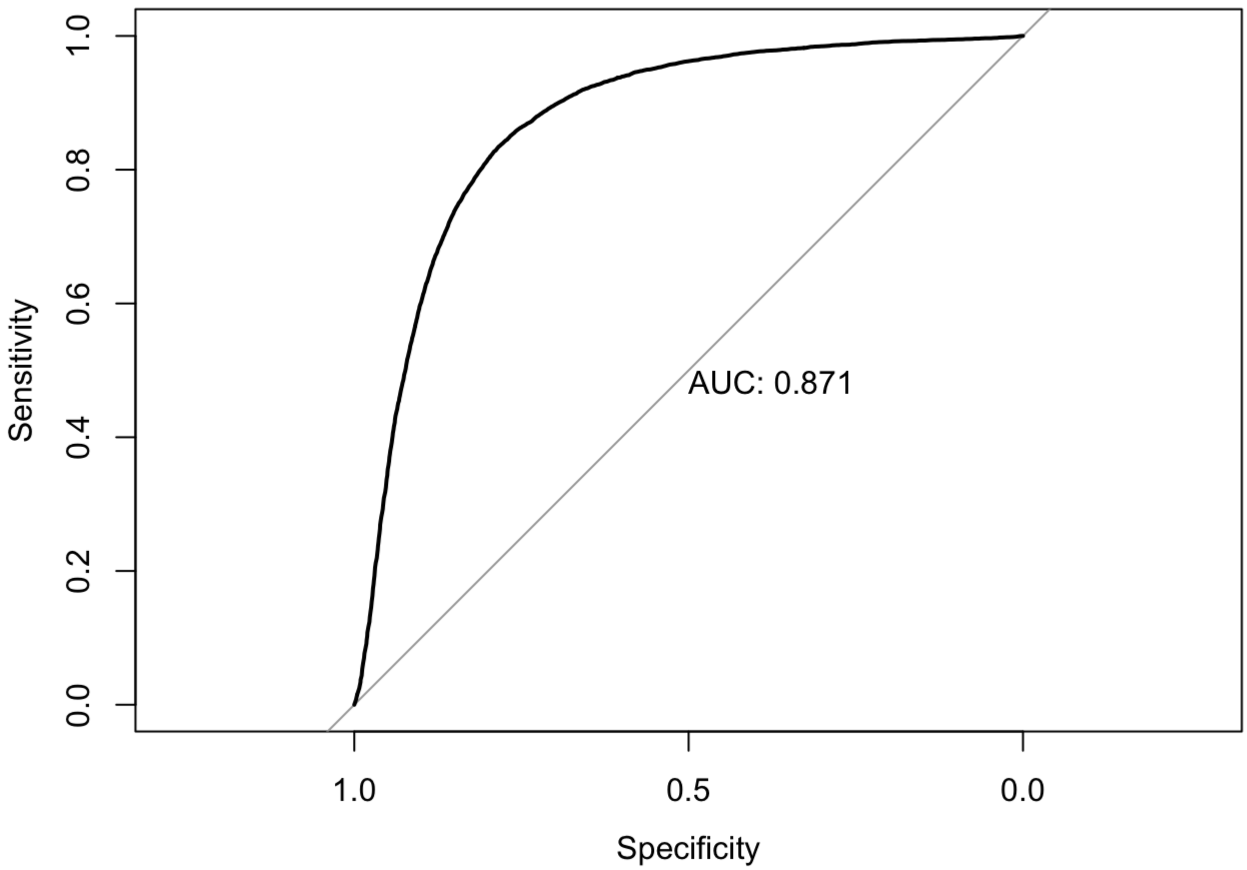
feature	MeanDecreaseGini
hour	4926.6709
Day_of_Week	3739.1937
Junction_Detail	2401.3582
X1st_Road_Class	2073.8298
X2nd_Road_Class	1880.6990
Pedestrian_Crossing.Physical_Facilities	1767.3322
Did_Police_Officer_Attend_Scene_of_Accident	1624.5878
Speed_limit	1518.8790
Road_Type	1310.6837
Weather_Conditions	1113.2497
Road_Surface_Conditions	1093.0159
Light_Conditions	1060.3385
Urban_or_Rural_Area	610.1785
Special_Conditions_at_Site	565.0536
Carriageway_Hazards	281.5902
Pedestrian_Crossing.Human_Control	210.5528

```
importance$feature<-reorder(importance$feature,importance$MeanDecreaseGini,mean)

# plot feature importance
ggplot(data = importance,mapping = aes(x=MeanDecreaseGini, y = feature))+geom_abline(intercept =
4, slope = 0, color= "red")+geom_abline(intercept = 13.5, slope = 0, color= "blue")+geom_point()
+labs(title ="Feature Importance based on the decrease of Gini")
```




```
# plot the roc curve of the model
pred.forest.full <-predict(forest.road.full, newdata = na.omit(road.unbalanced), type = "prob")
full.roc <- roc(road.unbalanced$Severity ~ pred.forest.full[,2],plot=TRUE,print.auc = TRUE)
```




```
full.roc
```

```
##
## Call:
## roc.formula(formula = road.unbalanced$Severity ~ pred.forest.full[,     2], plot = TRUE, prin
t.auc = TRUE)
##
## Data: pred.forest.full[, 2] in 72928 controls (road.unbalanced$Severity Not_Severe) < 11700 c
ases (road.unbalanced$Severity Severe).
## Area under the curve: 0.8713
```

Cross validation with the unbalanced dataset

To see to what extent the random forest model can be applied to the real-life setting where the distribution of severe and not severe accident is not balanced. We defined a 10-fold cross validation function that calculates the accuracy and sensitivity of the model.

Based on the result of cross validation, the predicting accuracy is about 75% with sensitivity rate of 12%. In this regard, the performance of a random forest model is not as good as the default classifier.

For the feature selection part, features below the red line could be regarded as not informative, which is consistent with our findings in the feature selection part. Features above the blue line should be considered as important. On the other hand, the order of the important features seems to differ from previous analysis. This could be explained by the fact that the decrease in Gini is measure at each branch of the tree while previously, we are taking all features into consideration at the same time.

```
set.seed(2019)

# define the function
cv.rf <- function(data){
  cv.accuracy<-c()
  cv.sensitivity <-c()
  for(i in c(1:10)){
  tst.id <- sample(c(1:nrow(data)), 8000)
  tst.set <- data[tst.id,]
  pred.tst <- predict(forest.road.full, newdata = tst.set , type = "class")
  result<-data.frame(true = tst.set$Severity, pred_forest = pred.tst)
  cv.accuracy <-c(cv.accuracy, mean(result$true == result$pred_forest))
  cv.sensitivity <-c(cv.sensitivity,
                     sum(result$true =="Severe" & result$pred_forest =="Severe")/nrow(result))
  data<-data[-tst.id,]}
  final.result <-data.frame(cv.accuracy,cv.sensitivity)
  return(final.result)}

# get the result of cross validation
cv.rf(road.unbalanced)
```

```
##    cv.accuracy cv.sensitivity
## 1     0.732625       0.120375
## 2     0.729125       0.122750
## 3     0.741625       0.125000
## 4     0.739375       0.123250
## 5     0.730500       0.122875
## 6     0.744125       0.124000
## 7     0.742125       0.122750
## 8     0.734750       0.119000
## 9     0.733250       0.122125
## 10    0.747375       0.124250
```

Logistic Regression

The last model we use is logistic regression. We first regress the outcomes on all the features then on the selected sub group. Based on the result of previous analysis, we divide the variables into different importance levels.

- High Importance Features:“Junction_Detail”,“hour”,“Light_Conditions”,“Speed_Limit”
- Meduim Importance Features:“X1st_Road_Class”,“X2nd_Road_Class”,“Road_Type”
- Low Importance Features:“Did_Police_Officer_Attend_Scene_of_Accident”,“Road_Surface_Conditions”,“Day_of_Week”)

We first trained the model and ran 10-fold cross validation. Based on the prediction error, the features selected are informative about the severity of the accidents as we did not identify significant decrease in predicting accuracy as we reduce the number of predictors in the regression model.

Additionally, it is worth noticing that the predicting accuracy of logistic regression is 88%, which is slightly higher than the default classifier. Therefore, we could conclude that the features we chose are useful.

```
#install.packages("boot")
library(boot)
```

```
##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:lattice':
##
##     melanoma
```

```
# create the vector to store the variables
var.1 <- c("Junction_Detail","hour","Light_Conditions","Speed_Limit")
var.2 <- c("Junction_Detail","hour","Light_Conditions","Speed_Limit","X1st_Road_Class","X2nd_Roa
d_Class","Road_Type")
var.3 <-c("Junction_Detail","hour","Light_Conditions","Speed_Limit","X1st_Road_Class","X2nd_Road
_Class","Road_Type","Did_Police_Officer_Attend_Scene_of_Accident","Road_Surface_Conditions","Day
_of_Week")

# with a for loop, we go through the three models
set.seed(9)

CV.Error <-c()
for (var in list(var.1,var.2,var.3)){
  formulaString <- as.formula(paste0("Severity ~ ", var))
  glm.fit <- glm(formulaString , data = road.unbalanced, family = "binomial", control = list(max
it = 50))
  cv10.error = cv.glm(data = road.unbalanced,  glmfit = glm.fit, K=10)
  CV.Error<-c(CV.Error,cv10.error$delta[2])}

log.result <- data.frame( Var = c("high","High + Medium","High + Meduim + Low" ),
                          Prediction.Error = CV.Error)
kable(log.result)
```

Var	Prediction.Error
high	0.1186825
High + Medium	0.1186902
High + Meduim + Low	0.1186851

Conclusion

Feature Selection and predictive analysis

Based on previous analysis, we think the following features are informative about traffic accidents, and we divided them into different levels based on their importance as demonstrated in the following list:

- High Importance Features:“Junction_Detail”,“hour”,“Light_Conditions”,“Speed_Limit”
- Meduim Importance Features:“X1st_Road_Class”,“X2nd_Road_Class”,“Road_Type”
- Low Importance Features:“Did_Police_Officer_Attend_Scene_of_Accident”,“Road_Surface_Conditions”,“Day_of_Week”)

Additionally, location seems to be highly related to high-severity accidents as well. However, after visualizing the distribution of accident, this could be caused by the busy traffic conditions in major cities and urban areas.

With these features, we ran predictive analysis (randomforest, naive bayes and logistic regression) trying to differentiate traffic accidents of different severity. Among the three models, logistic regression yields the highest predicting accuracy higher than the default classifier. The performance of the logistic regression did not decrease significantly when we only include the most important features.

Recommendations

Road Conditions

From a practical perspective, following measures could be adopted to avoid high-severity traffic accidents:

- improve the light condition, especially in the areas with no lighting. As shown in the following table, darkness is positively correlated with the occurrence of high-severity accidents.
- Speed limits are also important in reducing severe traffic accidents, but these two features are not perfectly correlated. When speed limit is 60, we found the highest percentage of severe accident. This could be explained by the fact that severe accidents are unlikely to happen when the speed of vehicles is low (low speed limit). On the other hand, when the speed of vehicles is high (speed limit = 70), drivers are generally more concentrated on driving, and we could expect the road conditions are generally good. Therefore, when the speed limit is somewhere in the middle, severe accidents might happen. To address this problem, the transportation department could consider to reduce the speed limits of roads with high record of traffic accidents. Also, use warning signs along the road to keep drivers concentrated.
- For road conditions as we would expected, wet/damp, forest/ice, and snow condition may lead to severe traffic accident.

```
require(plyr)
require(knitr)
# Light condition
road.unbalanced <- na.exclude(road.unbalanced)
kable(table(road.unbalanced$Severity,road.unbalanced$Light_Conditions))
```

	Darkness - lighting unknown	Darkness - lights lit	Darkness - lights unlit	Darkness - no lighting	Daylight
Not_Severe	1176	15459	339	1240	54714
Severe	190	2944	57	351	8158

```
# speed limit
speed <- table(road.unbalanced$Speed_limit,road.unbalanced$Severity)
speed$percentage <- speed[,2]/(speed[,1] + speed[,2])
speed$percentage
```

```
##        20        30        40        50        60        70
## 0.1395349 0.1320442 0.1446176 0.1392666 0.1893165 0.1255498
```

```
# road surface condition
surface <- ddply(road.unbalanced,"Road_Surface_Conditions", summarise, Count=sum(Severity=="Seve
re"), Percentage = mean(Severity == "Severe"))
arrange(surface,desc(Percentage))
```

```
##   Road_Surface_Conditions Count Percentage
## 1                     Dry  8339  0.1406382
## 2              Wet or damp  3201  0.1331143
## 3             Frost or ice   123  0.1255102
## 4                     Snow    28  0.1233480
## 5    Flood over 3cm. deep     9  0.1125000
```

Police Jurisdiction

According to previous analysis, certain types of roads and junctions might need more police force to reduce severe accidents.

For Junction detail, high severity accidents are more likely to happen in rather complex road junctions like crossroads, roads with more than 4 arms, private drive or entrance, and T/staggered road. In particular, more police force should be deployed to Staggered T road where we have most severe accidents in terms of both the absolute count and percentage. Also, policy jurisdiction barely exists for private entrance and drive way, which is another area that could potentially be improved as well.

For road types, more police force is need in carriageway and one-way road. Naturally one-way road is regarded as a rather safe road type due to its simple structure. Our finding indicates that extra attention might be needed as well.

```
require(knitr)
require(plyr)
require(dplyr)

# Junction Details
junction <- as.data.frame(ddply(road.unbalanced,"Junction_Detail", summarise, Count = sum(Severi
ty =="Severe"),
                                Percentage = mean(Severity =="Severe")))
junction <- arrange(junction,desc(Percentage))
kable(junction,digits = 5)
```

Junction_Detail	Count	Percentage
T/staggered	6945	0.15366
Private drive or entrance	636	0.15132
More than 4 arms	273	0.13943
Other junction	391	0.13797
Crossroads	1850	0.13215
SlipRoad	242	0.11646
Mini-roundabout	162	0.09963
Roundabout	1201	0.09432

```
# Road Type
road.type <- as.data.frame(ddply(road.unbalanced,"Road_Type", summarise, Count = sum(Severity ==
"Severe"),
                                Percentage = mean(Severity =="Severe")))
road.type <- arrange(road.type,desc(Percentage))
kable(road.type,digits = 5)
```

Road_Type	Count	Percentage
Single carriageway	9357	0.14738

Road_Type	Count	Percentage
One way	237	0.14434
Dual carriageway	1045	0.12248
Roundabout	946	0.09783
Slip road	98	0.09032
Unknown	17	0.08057

On the whole, Predicting the severity of traffic accidents is a difficult task. The happen of severe accidents tend to be the result of a combination of undesirable conditions and bad luck. Based on previous analysis, when several undesirable external conditions like darkness, wet road, complex junction detail exist at the same time same place, the transportation department should pay extra attention to preventing severe traffic accidents.

Reflections on the Projects

After completing the project, our group found the following aspects especially challenging:

- We only have variables describing the external conditions. Equally important is information about the vehicles and the drivers in analyzing traffic accidents in predicting the severity of traffic accidents.
- All the features given by the data set are categorical variables which considerable limits our choices in classification algorithms and effective visualization methods.
- The distribution of outcome variable is so unbalanced that 1) the prior possibility considerably constrained the performance of naive bayes model; 2) the data set cannot provide enough information about the high-severity traffic accidents. Therefore, it would be better of we could have more data about the severe accident so that we could have more variability.