Assignment Prediction Model

Manousos Emmmanouil Theodosiou [6686311], Group: GG Force

11-01-2021

```
library(MASS)
library(class)
library(ISLR)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x dplyr::select() masks MASS::select()
```

```
library(rpart)
library(rpart.plot)
library(randomForest)
```

```
## randomForest 4.6-14

## Type rfNews() to see new features/changes/bug fixes.

##
## Attaching package: 'randomForest'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

```r
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

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

Load the dataset "accidents_2012_to_2014.csv"

```r
accidents <-
  read_csv("accidents_2012_to_2014.csv") %>%
  distinct()  #Removes 34147 duplicate rows
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   .default = col_double(),
##   Accident_Index = col_character(),
##   Date = col_character(),
##   Time = col_time(format = ""),
##   `Local_Authority_(Highway)` = col_character(),
##   Road_Type = col_character(),
##   Junction_Detail = col_logical(),
##   Junction_Control = col_character(),
##   `Pedestrian_Crossing-Human_Control` = col_character(),
##   `Pedestrian_Crossing-Physical_Facilities` = col_character(),
##   Light_Conditions = col_character(),
##   Weather_Conditions = col_character(),
##   Road_Surface_Conditions = col_character(),
##   Special_Conditions_at_Site = col_character(),
##   Carriageway_Hazards = col_character(),
##   Did_Police_Officer_Attend_Scene_of_Accident = col_character(),
##   LSOA_of_Accident_Location = col_character()
## )
## i Use `spec()` for the full column specifications.
```

```r
accidents <- accidents %>%
  mutate(Accident_Index = 1:nrow(accidents)) #Re-index. Previous index had lots of dupl

#View(accidents)

#distinct(accidents, Road_Type)
#distinct(accidents, Weather_Conditions)
#distinct(accidents, Light_Conditions)
#distinct(accidents, Road_Surface_Conditions)
#distinct(accidents, Special_Conditions_at_Site)
#distinct(accidents, Carriageway_Hazards)
```

Introduction and Description of the Data set

For this assignment we have used a dataset of UK Traffic Accidents found on kaggle. (https://www.kaggle.com/daveianhickey/2000-16-traffic-flow-england-scotland-wales/data?select=accidents_2012_to_2014.csv). Given the size of the dataset (over 600 MB) we decided to focus only on the timeframe 2012 to 2014. This still gives us over 450.000 data entries to work with. The data is very well structured and requires no dropping of data due to incomplete records. The data set contains information coming from police reports on major car accidents throughout the UK. In all there are 33 columns containing location information (coordinates, police force and administrative area), date information (date, time, day of the week, year), information on the road where the accident took place (weather, type of road, overpasses, junctions) and the severity of the accident (severity, casualties). Further descriptions for the columns can be found in the file 7752_road-accident-safety-data-guide.xls. Given this breadth of ordinal, categorical and numerical information, there are many different questions that can be looked into. We have, however, focused on predicting the accident severity based on a combination of the predictor variables. Given the large amount of data it should also be possible to split the data set into many individual sets and tune the parameters of the given predictions.

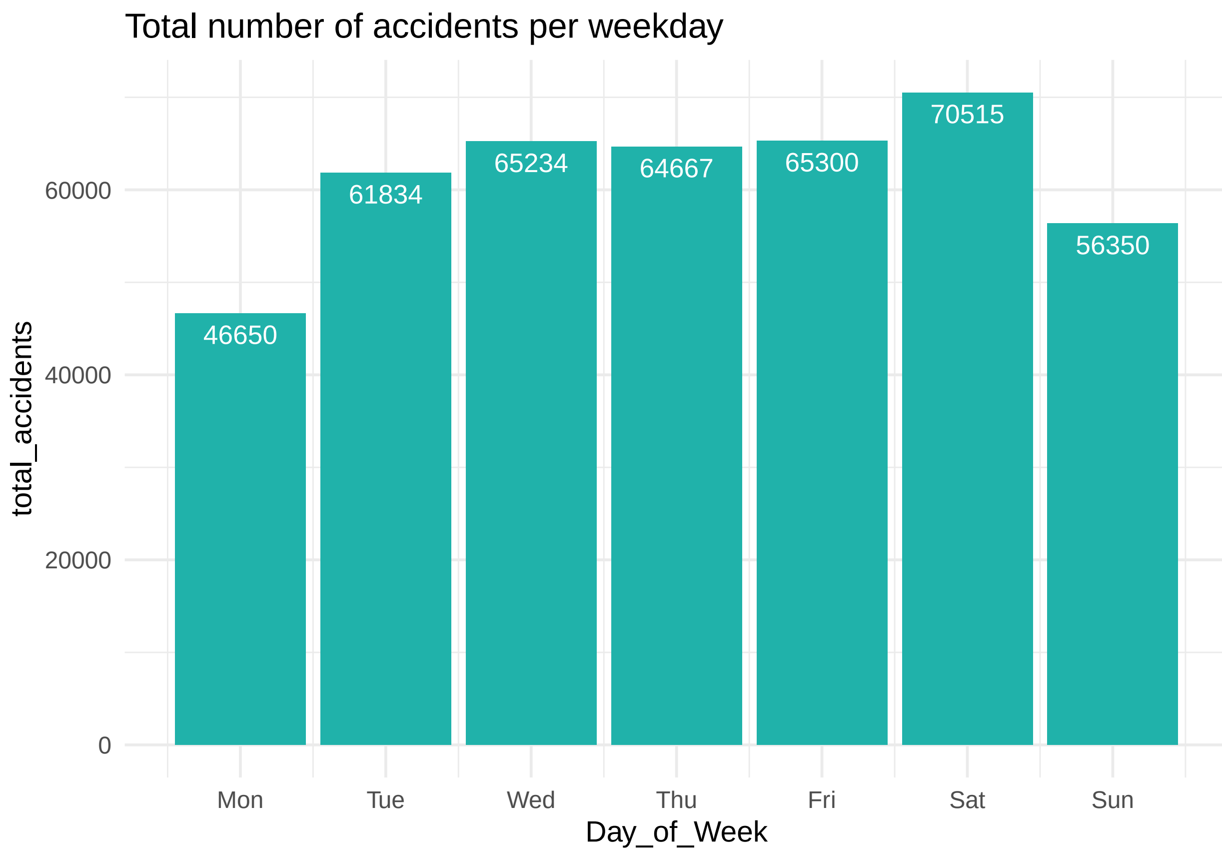
Exploration

By looking at the dataset it can be seen that in 2012-2014 the most accidents occur on Saturday whereas the least number of accidents take place on Monday. (1 is Monday, 2 is Tuesday, etc).

```r
accidents %>%
  group_by(Day_of_Week) %>%
  summarize(total_accidents=n_distinct(Accident_Index)) %>%
    ggplot(aes(x=Day_of_Week, y=total_accidents)) +
    geom_bar(stat="identity", fill="light seagreen")+
```

```
  geom_text(aes(label=total_accidents), vjust=1.6, color="white", size=3.5)+
  ggtitle("Total number of accidents per weekday") +
  scale_x_continuous(breaks = 1:7, labels = c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat'
  theme_minimal()
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

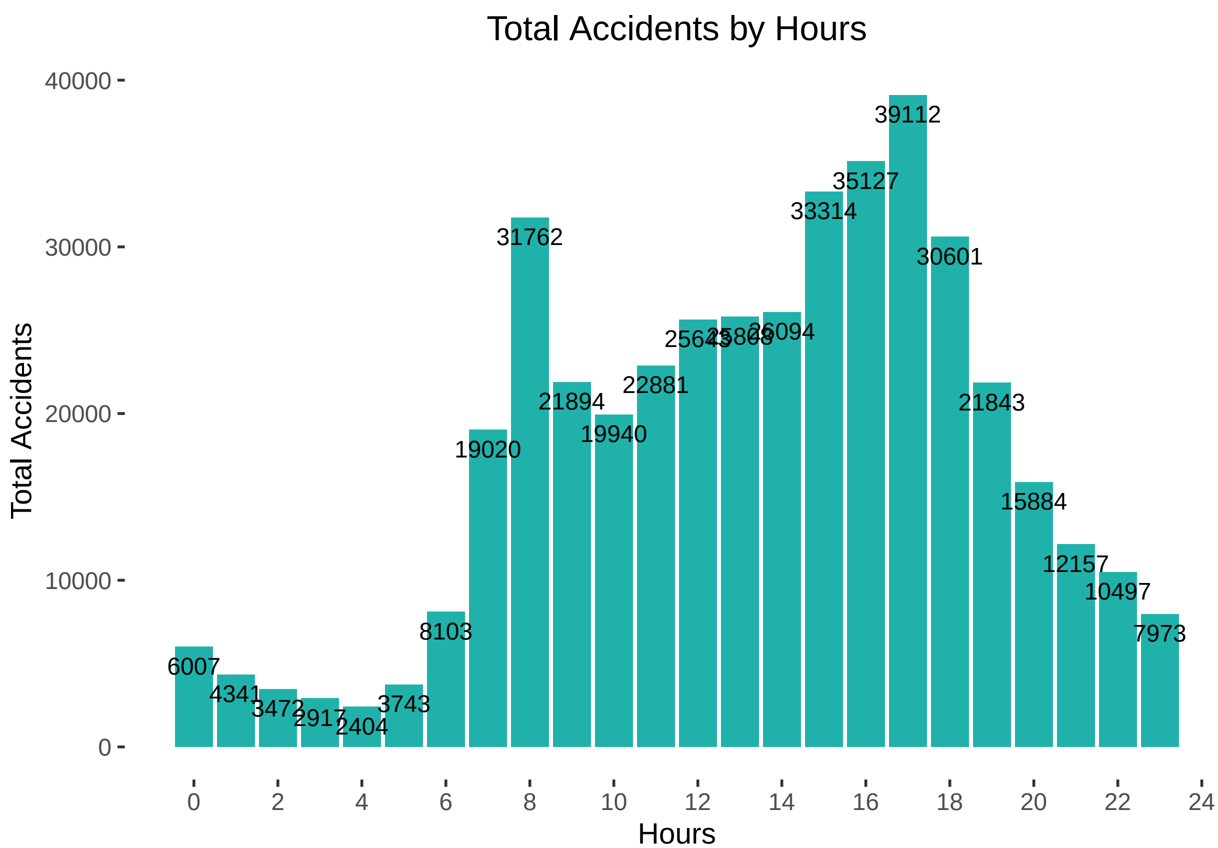


We can also visualise the number of accidents by hour. We can see that most accidents take place around 5pm and 8am. This could make sense since people having 9 to 5 jobs are more likely to have an accident on their way to or from their jobs.

```
accidents %>%
  mutate(time_slot = as.numeric(substr(Time,0,2))) %>%
  group_by(time_slot) %>%
  summarize(total_accidents=n_distinct(Accident_Index)) %>%
    ggplot(aes(x=time_slot, y=total_accidents)) +
    geom_bar(stat="identity", fill="light seagreen")+
    geom_text(aes(label=total_accidents), vjust=1.6, color="black", size=3)+
    scale_x_continuous(breaks = round(seq(0, 24, by = 2),0)) +
    ggtitle("Total Accidents by Hours") +
```

```r
    xlab("Hours") + ylab("Total Accidents")+
    theme(plot.title = element_text(hjust = 0.5), panel.background = element_blank())
```

```
## `summarise()` ungrouping output (override with `.groups` argument)

## Warning: Removed 1 rows containing missing values (position_stack).

## Warning: Removed 1 rows containing missing values (geom_text).
```



Here, we will analyse the probability of having an accident (1 for Fatal, 2 for Serious, 3 for Slight) in a given hour.

```r
accidents <- accidents %>%
  mutate(time_slot = as.numeric(substr(Time,0,2)))

prop.table(table(accidents$time_slot, accidents$Accident_Severity), 1)
```

```
##
##                    1            2            3
```

```
##   0  0.024138505 0.190444481 0.785417013
##   1  0.028104123 0.194655609 0.777240267
##   2  0.027361751 0.212845622 0.759792627
##   3  0.034624614 0.213232773 0.752142612
##   4  0.036189684 0.188851913 0.774958403
##   5  0.030189687 0.185679936 0.784130377
##   6  0.016537085 0.161298285 0.822164630
##   7  0.010252366 0.137486856 0.852260778
##   8  0.005383792 0.114381966 0.880234242
##   9  0.008175756 0.120032886 0.871791358
##   10 0.012337011 0.135055165 0.852607823
##   11 0.011450548 0.132205760 0.856343691
##   12 0.008618336 0.132355809 0.859025855
##   13 0.010306882 0.134648171 0.855044947
##   14 0.009734038 0.138116042 0.852149920
##   15 0.009485502 0.140961758 0.849552741
##   16 0.009650696 0.143023885 0.847325419
##   17 0.009025363 0.135994068 0.854980569
##   18 0.009215385 0.142413647 0.848370968
##   19 0.012177814 0.148377054 0.839445131
##   20 0.011206245 0.156950390 0.831843364
##   21 0.015299827 0.166735214 0.817964958
##   22 0.020100981 0.169476993 0.810422025
##   23 0.022576195 0.173711276 0.803712530
```

Now we will choose only the relevant predictors for the response variable Accident_Severity and make a dataframe.

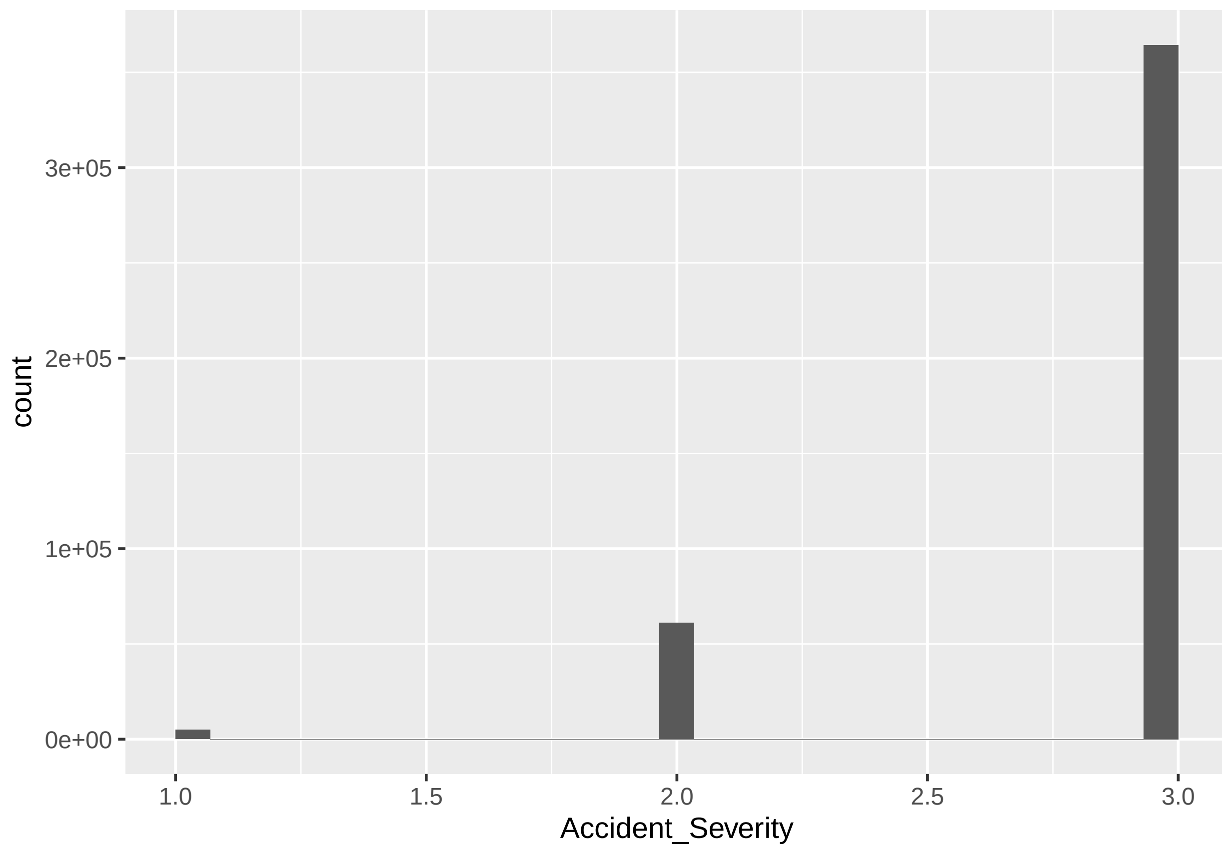
```r
variables <- names(accidents) %in% c("Accident_Severity", "Number_of_Vehicles",
                                      "Number_of_Casualties", "Day_of_Week",
                                      "time_slot", "Road_Type",
                                      "Speed_limit", "Light_Conditions",
                                      "Weather_Conditions", "Road_Surface_Conditions"
                                      )
accidents <- data.frame(accidents[variables])
rm(variables)
```

Initially we ran all our models on the unfiltered data, but soon it became clear that the classes are very disproportionately distributed as can be seen in the graph below.

Prior probabilities of groups: 1 2 3 0.01141174 0.14371515 0.84487311

```r
ggplot(accidents, aes(x=Accident_Severity)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We decided then to downsample to make classes more equal. The new distribution:

Prior probabilities of groups: 1 2 3 0.1666667 0.3333333 0.5000000

```
# Downsampling to make classes equally largge


sev_1 <- accidents %>%
  filter( Accident_Severity == 1)

sev_2 <- accidents %>%
  filter( Accident_Severity == 2)

sev_3 <- accidents %>%
  filter( Accident_Severity == 3)

dim(sev_1)
```

```
## [1] 4903   10
```

```r
dim(sev_2)
```

```
## [1] 61201    10
```

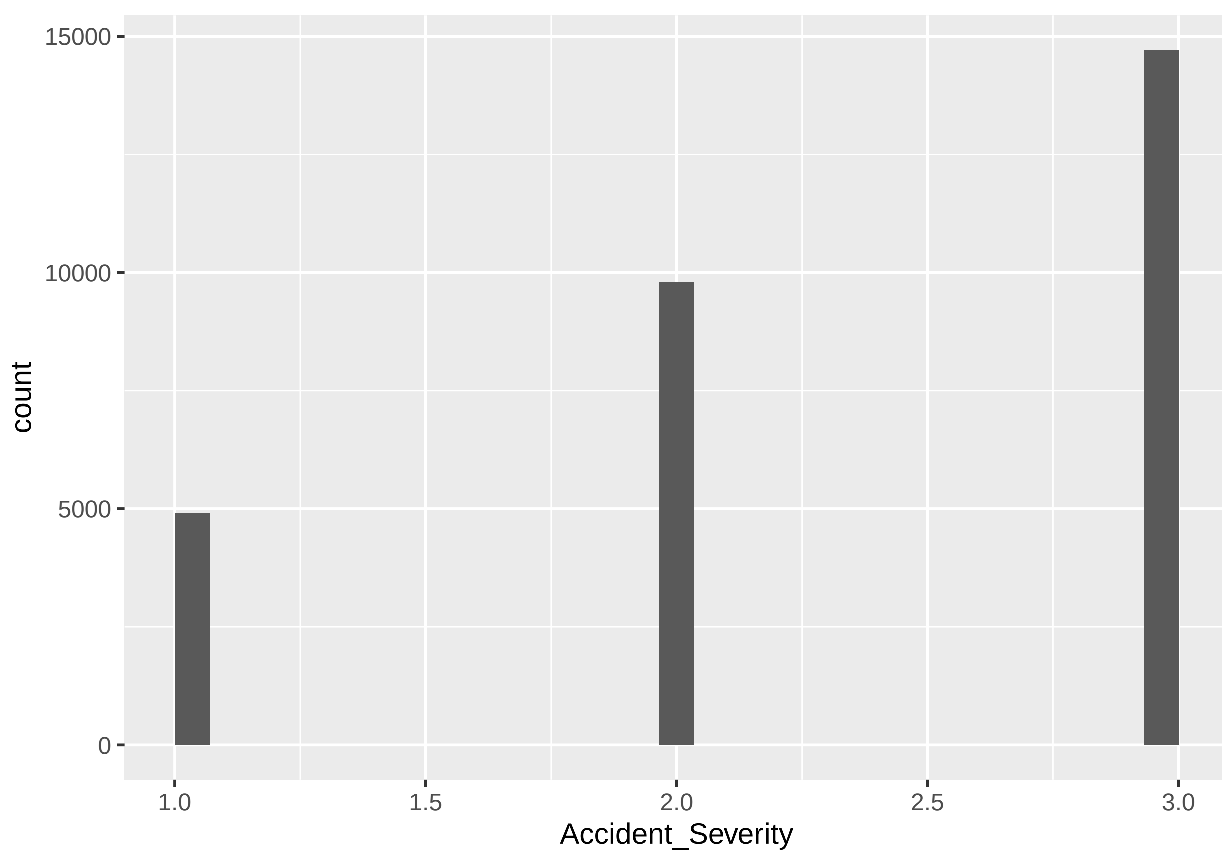
```r
dim(sev_3)
```

```
## [1] 364446     10
```

```r
set.seed(5)

accidents_balanced <- rbind(sev_1, sample_n(sev_2, 2 * nrow(sev_1)), sample_n(sev_3,  3
rm(sev_1, sev_2, sev_3)
```

```r
ggplot(accidents_balanced, aes(x=Accident_Severity)) + geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



We experimented with different distributions. Such as sampling all of the classes equally, or not sampling at all. However, when sampling equally, the accuraccy suffers immensely, as most of the observations will be of class 3, but will be missclassified due to the dissproportionate weight on class 1. If we don't rebalance at all, the accuraccy goes up very high, however, this is only due to the unequal distribution. A 'dumb classifier' that puts everything in class 3 would already have an accuracy of 84%. Yet, since class one means fatal accidents, it's important that these don't get underpredicted since these have the highest impact. It is for this reason that we shifted the distribution to a 1:2:3 balancing, so that this class does not get underpredicted.

Check the type of each column by running the sapply() function. Changed Accident_Severity to factor, making the tree plots behave better.

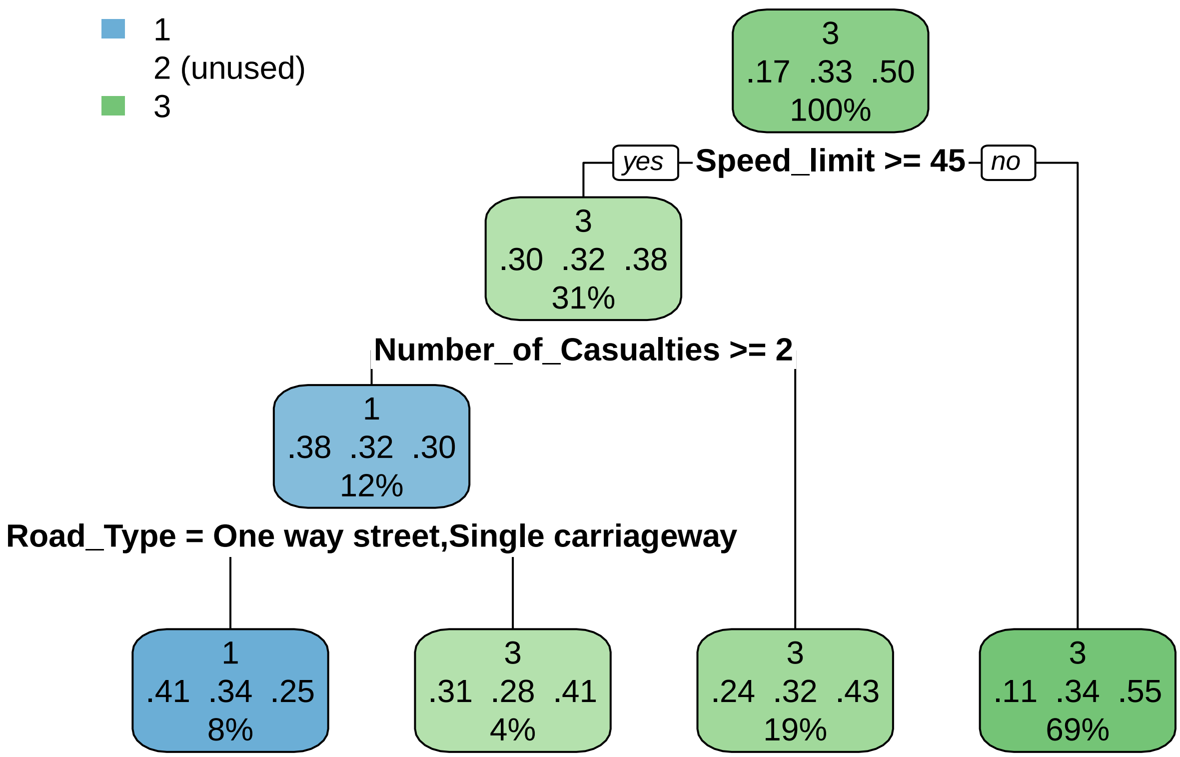
```
accidents_balanced$Accident_Severity <- as.factor(accidents_balanced$Accident_Severity)
accidents$Accident_Severity <- as.factor(accidents$Accident_Severity)

sapply(accidents_balanced, class)
```

```
##       Accident_Severity      Number_of_Vehicles    Number_of_Casualties
##                "factor"               "numeric"               "numeric"
##             Day_of_Week               Road_Type             Speed_limit
##               "numeric"             "character"               "numeric"
##        Light_Conditions      Weather_Conditions Road_Surface_Conditions
##             "character"             "character"             "character"
##               time_slot
##               "numeric"
```

Make classification trees using the rpart() function.

Using all predictors:

```
set.seed(1)
tree_mod <- rpart(Accident_Severity ~ ., data = accidents_balanced,
                            control=rpart.control(minsplit=20, cp=0.005))
rpart.plot(tree_mod)
```



The classifcationt tree above is well behaved and setting the minsplit and cp values as they are, it is still possible to read the chart as well as making sure not to overfit on the training data.

Random Forest for classification

In addition to the tree classifier we want to fit a random forest on all of the chosen variables.

```
rf_mod <- randomForest(Accident_Severity ~ ., data = accidents_balanced, importance = TR
rf_mod
```

```
##
## Call:
##  randomForest(formula = Accident_Severity ~ ., data = accidents_balanced,      import
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 48.08%
## Confusion matrix:
```

```
##      1    2     3 class.error
## 1 1321 1635  1942   0.7302981
## 2 1080 2768  5943   0.7172914
## 3  841 2686 11165   0.2400626
```

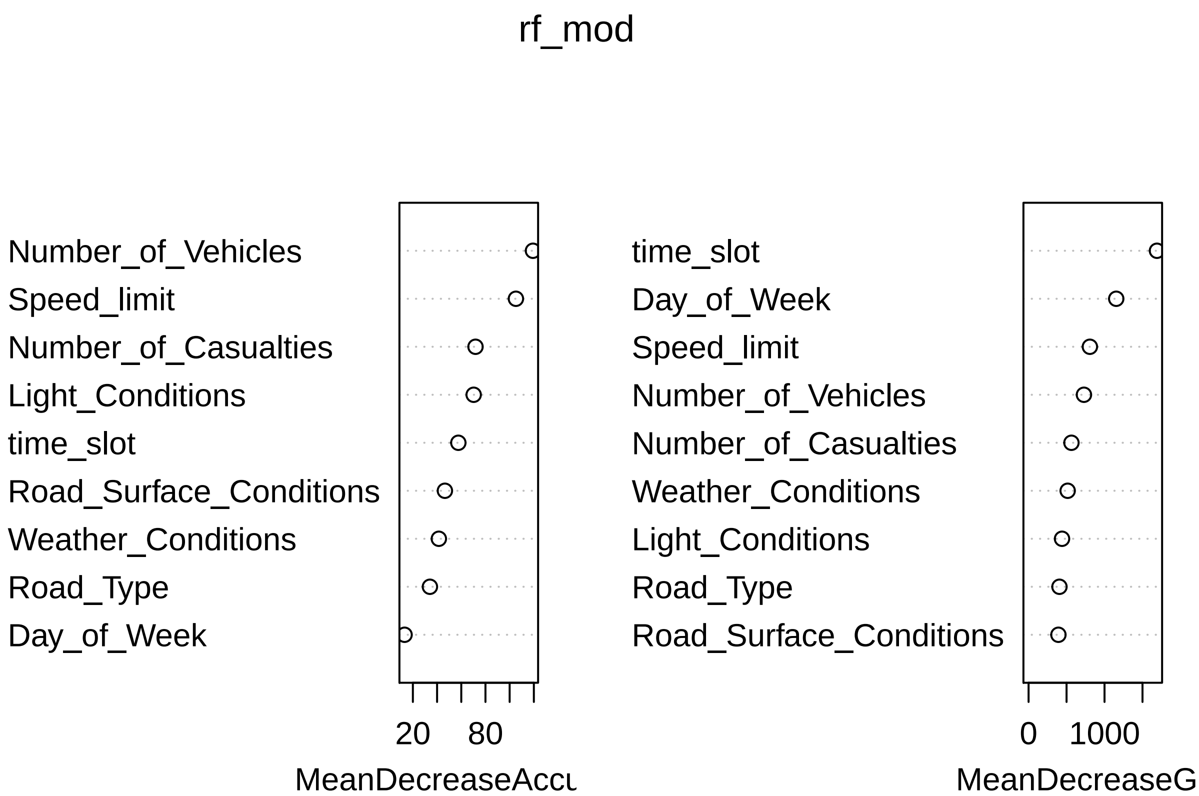
Importance

```r
importance(rf_mod)
```

```
##                                  1           2         3 MeanDecreaseAccuracy
## Number_of_Vehicles       42.872691  45.5098809 118.11490            119.26605
## Number_of_Casualties     61.680777  17.8634760  30.14713             71.78429
## Day_of_Week               6.791339   0.8585354  13.72394             13.13640
## Road_Type                18.574633  -6.8561100  28.07522             34.08654
## Speed_limit             107.250543 -13.7514600  65.84254            105.17277
## Light_Conditions         30.764369 -16.2210910  59.89421             70.24553
## Weather_Conditions       -1.079939 -21.0458705  47.06572             41.45329
## Road_Surface_Conditions   4.593091 -13.8299951  48.19219             46.47734
## time_slot                 4.662587  -0.8010361  52.41819             57.58710
##                         MeanDecreaseGini
## Number_of_Vehicles              728.7777
## Number_of_Casualties            564.2432
## Day_of_Week                    1154.2524
## Road_Type                       406.5733
## Speed_limit                     807.1141
## Light_Conditions                440.3480
## Weather_Conditions              514.9026
## Road_Surface_Conditions         392.8388
## time_slot                      1690.4434
```

Plot importance

```r
varImpPlot(rf_mod)
```



The charts above indicate that … [INTERPRETATION] Discuss gini index etc.?

Next we will look at the prediction accuracy for the training data set:

```r
prediction_rf_train <- predict(rf_mod, newdata = accidents_balanced, na.action = na.excl


conf_rf <- table(predicted = prediction_rf_train, true = accidents_balanced$Accident_Sev

conf_rf
```

```
##          true
## predicted     1     2     3
##         1  2679   318   358
##         2   900  5289  1413
##         3  1319  4184 12921
```

```r
acc <- (sum(conf_rf[1,1] + conf_rf[2,2] + conf_rf[3,3]) / sum(conf_rf))

## Print 'accuracy' result (sum of diagonal entries/sum of all entries)
paste(round(acc*100, 2), "%", sep="")
```

```
## [1] "71.1%"
```

The accuracy rate is good considering the unbalanced data sets and the compromises we had to make. Class 1 is largely classified correctly, but better results would cause a lower accuracy, as the other classes will suffer considerably.

Now we look at all of the data:

```r
prediction_rf_test <- predict(rf_mod, newdata = accidents, na.action = na.exclude)


conf_test_rf <- table(predicted = prediction_rf_test, true = accidents$Accident_Severity

conf_test_rf
```

```
##          true
## predicted      1      2      3
##         1   2680   5827  20396
##         2    900  20098  66152
##         3   1318  35182 277242
```

```r
acc <- (sum(conf_test_rf[1,1] + conf_test_rf[2,2] + conf_test_rf[3,3]) / sum(conf_test_r

## Print 'accuracy' result (sum of diagonal entires/sum of all entries)
paste(round(acc*100, 2), "%", sep="")
```

```
## [1] "69.81%"
```

Interestingly, the accuracy for the whole data set is actually higher, given the data imbalance.

LDA Analysis

In addition to the random forest, we also take a look at an LDA model to predict the severity. Here we picked out the most important factors to base our formula on:

```r
# Fit lda model, i.e. calculate model parameters, using 'integer' variables only, we w
lda_mod <- lda(Accident_Severity ~ Number_of_Vehicles + Number_of_Casualties + Day_of_We


lda_mod
```

```
## Call:
## lda(Accident_Severity ~ Number_of_Vehicles + Number_of_Casualties +
##     Day_of_Week + Speed_limit + Light_Conditions + Weather_Conditions,
##     data = accidents_balanced)
##
## Prior probabilities of groups:
##         1         2         3
## 0.1666667 0.3333333 0.5000000
##
## Group means:
##   Number_of_Vehicles Number_of_Casualties Day_of_Week Speed_limit
## 1           1.753620             1.791556    4.078115    48.04405
## 2           1.688966             1.408831    4.150214    39.89904
## 3           1.858046             1.314297    4.133796    38.04337
##   Light_ConditionsDarkness: Street lighting unknown
## 1                                        0.01060575
## 2                                        0.01611258
## 3                                        0.01543273
##   Light_ConditionsDarkness: Street lights present and lit
## 1                                               0.2000816
## 2                                               0.2055884
## 3                                               0.1964783
##   Light_ConditionsDarkness: Street lights present but unlit
## 1                                               0.010197838
## 2                                               0.005200897
## 3                                               0.005438847
##   Light_ConditionsDaylight: Street light present
## 1                                      0.6012645
## 2                                      0.6979400
## 3                                      0.7331566
##   Weather_ConditionsFine without high winds Weather_ConditionsFog or mist
## 1                                 0.8366306                   0.009382011
## 2                                 0.8222517                   0.006628595
## 3                                 0.7996465                   0.006118703
##   Weather_ConditionsOther Weather_ConditionsRaining with high winds
## 1              0.01162554                                0.01550071
## 2              0.01458291                                0.01560269
## 3              0.01869604                                0.01556870
```

```
##   Weather_ConditionsRaining without high winds
## 1                                    0.0966755
## 2                                    0.1076892
## 3                                    0.1260453
##   Weather_ConditionsSnowing with high winds
## 1                              0.0006118703
## 2                              0.0015296757
## 3                              0.0016996397
##   Weather_ConditionsSnowing without high winds Weather_ConditionsUnknown
## 1                                  0.004691006               0.009993881
## 2                                  0.003569243               0.014684887
## 3                                  0.005914746               0.017336325
##
## Coefficients of linear discriminants:
##                                                                 LD1
## Number_of_Vehicles                                       0.46868394
## Number_of_Casualties                                    -0.48084618
## Day_of_Week                                              0.01312473
## Speed_limit                                             -0.04688459
## Light_ConditionsDarkness: Street lighting unknown        1.22360491
## Light_ConditionsDarkness: Street lights present and lit   0.64837144
## Light_ConditionsDarkness: Street lights present but unlit 0.15943793
## Light_ConditionsDaylight: Street light present           1.18576580
## Weather_ConditionsFine without high winds                0.28586663
## Weather_ConditionsFog or mist                            0.86272132
## Weather_ConditionsOther                                  1.10254531
## Weather_ConditionsRaining with high winds                0.88888114
## Weather_ConditionsRaining without high winds             0.81152366
## Weather_ConditionsSnowing with high winds                2.25696294
## Weather_ConditionsSnowing without high winds             1.11784145
## Weather_ConditionsUnknown                                0.64132707
##                                                                 LD2
## Number_of_Vehicles                                       1.22949738
## Number_of_Casualties                                    -0.07312953
## Day_of_Week                                             -0.03311254
## Speed_limit                                              0.01103909
## Light_ConditionsDarkness: Street lighting unknown       -1.24016807
## Light_ConditionsDarkness: Street lights present and lit  -0.69846092
## Light_ConditionsDarkness: Street lights present but unlit 0.66185949
## Light_ConditionsDaylight: Street light present          -0.66774554
## Weather_ConditionsFine without high winds                1.09806115
## Weather_ConditionsFog or mist                            1.34891599
## Weather_ConditionsOther                                  1.79429874
## Weather_ConditionsRaining with high winds                1.13290535
## Weather_ConditionsRaining without high winds             1.63502005
```

```
## Weather_ConditionsSnowing with high winds                        0.58093919
## Weather_ConditionsSnowing without high winds                     3.05033157
## Weather_ConditionsUnknown                                        1.59469064
##
## Proportion of trace:
##    LD1    LD2
## 0.9211 0.0789
```

7.

```
## Create a confusion matrix and assess model performance on the 'test' data set

## Use test data set
accident_pred <- predict(lda_mod, accidents_balanced)

## Now use the 'class' feature to assess performance
lda_class <- accident_pred$class


## Calculate the accuracy (diagonal entries) for this table
conf_mat <- table(predicted = lda_class, true = accidents_balanced$Accident_Severity)

conf_mat
```

```
##          true
## predicted     1     2     3
##         1  1264   991   918
##         2   518   639   528
##         3  3121  8176 13263
```

```
lda_acc <- (sum(conf_mat[1,1] + conf_mat[2,2] + conf_mat[3,3]) / sum(conf_mat))

## Print 'accuracy' result (sum of diagonal entires/sum of all entries)
paste(round(lda_acc*100, 2), "%", sep="")
```

```
## [1] "51.55%"
```

Compare with full data set:

```
## Create a confusion matrix and assess model performance on the 'test' data set

## Use test data set
```

```r
accidents_pred <- predict(lda_mod, accidents, na.action = na.exclude)

## Now use the 'class' feature to assess performance
lda_class <- accidents_pred$class


## Calculate the accuracy (diagonal entries) for this table
conf_mat <- table(predicted = lda_class, true = accidents$Accident_Severity)

conf_mat
```

```
##          true
## predicted      1      2      3
##         1   1264   5973  21840
##         2    518   4305  13969
##         3   3121  50923 328637
```

```r
lda_acc <- (sum(conf_mat[1,1] + conf_mat[2,2] + conf_mat[3,3]) / sum(conf_mat))

## Print 'accuracy' result (sum of diagonal entires/sum of all entries)
paste(round(lda_acc*100, 2), "%", sep="")
```

```
## [1] "77.62%"
```

Conclusion

This dataset posed us to some challenges. Most apparently because the classes are very unbalanced: a large majority of accidents is not severe and belongs to class 3. Therefore, it was difficult to train the models to classify severe accidents accurately. The result is that our classifiers did not perform better than a dummy classifier. Upsampling of the minority class to obtain equally large classes did not resolve this problem. This made the models predict the minority class more often, but did not improve accuracy.

Reduce number of predictors in RF

```r
library(splitstackshape)
vars <- names(accidents) %in% c("Accident_Severity", "Number_of_Vehicles", "Speed_limit"
                                "Number_of_Casualties", "time_slot", "Day_of_Week")
accidents_small <- accidents[vars]
rm(vars)
```

```r
accidents_small$Day_of_Week <- as.factor(accidents_small$Day_of_Week)
accidents_small$time_slot <- as.factor(accidents_small$time_slot)
accidents$Day_of_Week <- as.factor(accidents$Day_of_Week)
accidents$time_slot <- as.factor(accidents$time_slot)

accidents_small$Accident_Severity <- recode(accidents_small$Accident_Severity,
                                            '1'= "Severe", '2'= "Medium", '3'= "Light")

accidents_balanced_small <- accidents_small %>%
  stratified(., group = "Accident_Severity",
             size = c(Severe = 4900, Medium = 9800, Light = 14700),
             replace = FALSE)

rf_mod_small <- randomForest(Accident_Severity ~ ., data = accidents_balanced_small,
                             importance = TRUE, na.action = na.exclude)
rf_mod_small
```

```
##
## Call:
##  randomForest(formula = Accident_Severity ~ ., data = accidents_balanced_small,
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 48.4%
## Confusion matrix:
##        Severe Medium Light class.error
## Severe   1010   1469  2420   0.7938355
## Medium    796   2433  6570   0.7517094
## Light     689   2286 11725   0.2023810
```

```r
prediction_rf_small <- predict(rf_mod_small, newdata = accidents, na.action = na.exclude
conf_rf <- table(predicted = prediction_rf_small, true = accidents$Accident_Severity)
conf_rf
```

```
##          true
## predicted      1      2      3
##    Severe   1804   4554  15778
##    Medium   1036  16053  53731
##    Light    2062  40588 294931
```

```r
acc <- (sum(conf_rf[1,1] + conf_rf[2,2] + conf_rf[3,3]) / sum(conf_rf))
paste(round(acc*100, 2), "%", sep="")
```

```
## [1] "72.65%"
```

Reducing the number of predictors in the RF model results in less accurate predictions, even though the used predictors here are the most important.