Different Models

Zhuocheng Lin

3/22/2020

1. Packages

```
library(tidyverse)
library(modelr)
library(caret)
library(ROSE)
library(randomForest)
library(glmnet)
library(rpart)
```

2. Data pre-processing

(1) Read data

```
df <- read_csv("./tidy.csv", col_types = cols(.default = col_character())) %>%
  type_convert()
```

(2) Specify factors

```
df_format <- df %>%
  mutate(TMC = factor(TMC), Severity = factor(Severity), Year = factor(Year), Wday = factor(Wday)) %>%
  mutate_if(is.logical, factor) %>%
  mutate_if(is.character, factor)
```

(3) Narrow down to one State

```
df_format %>% count(State) %>% arrange(desc(n))
```

```
## # A tibble: 49 x 2
##    State      n
##    <fct>  <int>
##  1 CA    650285
##  2 TX    291281
##  3 FL    221148
##  4 SC    143606
##  5 NC    141397
##  6 NY    136288
##  7 PA     89120
##  8 MI     88488
```

```
##  9 IL     86105
## 10 GA     82547
## # ... with 39 more rows
```

```r
# choose TX as the target State
df_TX <- df_format %>% filter(State == "TX") %>% select(-State)
```

(4) Remove unuseful variables

```r
# remove variables with only 1 distinct value
df_TX %>% summarise_all(~ n_distinct(.)) %>%
  pivot_longer(everything(), names_to = "variable", values_to = "n") %>% filter(n == 1)
```

```
## # A tibble: 1 x 2
##   variable         n
##   <chr>        <int>
## 1 Turning_Loop     1
```

```r
df_TX <- df_TX %>% select(-Turning_Loop)
```

(5) Drop weather condition levels

```r
# some Weather_Condition levels only have a few observations
# which can be a problem when we try to build a model
df_TX %>% count(Weather_Condition) %>% filter(n < 20) %>% select(Weather_Condition)
```

```
## # A tibble: 18 x 1
##    Weather_Condition
##    <fct>
##  1 Blowing Dust
##  2 Drizzle and Fog
##  3 Haze / Windy
##  4 Heavy Drizzle
##  5 Heavy T-Storm / Windy
##  6 Light Drizzle / Windy
##  7 Light Freezing Fog
##  8 Light Haze
##  9 Light Ice Pellets
## 10 Light Rain Showers
## 11 Light Snow / Windy
## 12 N/A Precipitation
## 13 Rain Showers
## 14 Sand
## 15 Showers in the Vicinity
## 16 Smoke
## 17 Thunder / Windy
## 18 Wintry Mix
```

```r
drop_weather <- df_TX %>% count(Weather_Condition) %>% filter(n < 20) %>% select(Weather_Condition)
drop_weather <- drop_weather$Weather_Condition %>% unlist()
df_TX <- df_TX %>% filter(!(Weather_Condition %in% drop_weather))
df_TX <- df_TX %>% mutate(Weather_Condition = factor(Weather_Condition))
```

(6) Add new labels

```r
# group level 3 and 4 together, as "Severe"
# group level 1 and 2 together, as "Not Severe"
df_label <- df_TX %>%
  mutate("Status" = factor(ifelse(Severity == "3" | Severity == "4", "Severe", "Not Severe"),
                           levels = c("Not Severe", "Severe")))
df_label %>% select(Severity, Status)
```

```
## # A tibble: 291,156 x 2
##    Severity Status
##    <fct>    <fct>
##  1 2        Not Severe
##  2 2        Not Severe
##  3 2        Not Severe
##  4 2        Not Severe
##  5 3        Severe
##  6 2        Not Severe
##  7 2        Not Severe
##  8 3        Severe
##  9 3        Severe
## 10 2        Not Severe
## # ... with 291,146 more rows
```

(7) Near Zero-Variance Predictors

```r
# these variable may become zero-variance when the data are split into subsets
# remove them
nzv <- nearZeroVar(df_label, saveMetrics = T)
nzv[nzv$nzv,]
```

```
##                   freqRatio percentUnique zeroVar  nzv
## Visibility         21.90956   0.015112174   FALSE TRUE
## Amenity            52.04354   0.000686917   FALSE TRUE
## Bump             5598.15385   0.000686917   FALSE TRUE
## Give_Way          176.10219   0.000686917   FALSE TRUE
## No_Exit           969.52000   0.000686917   FALSE TRUE
## Railway           106.71587   0.000686917   FALSE TRUE
## Roundabout      26467.72727   0.000686917   FALSE TRUE
## Station            62.64066   0.000686917   FALSE TRUE
## Stop               66.13304   0.000686917   FALSE TRUE
## Traffic_Calming  2598.60714   0.000686917   FALSE TRUE
```

```r
nzv_cols <- rownames(nzv[nzv$nzv,])
df_label <- df_label %>%
  select(-nzv_cols)
```

(8) Partition

```r
set.seed(1)
df_parts <- resample_partition(df_label, c(train = 0.6, valid = 0.2, test = 0.2))
train <- as_tibble(df_parts$train)
valid <- as_tibble(df_parts$valid)
test <- as_tibble(df_parts$test)
```

```r
# check Weather_Condition levels
# train should have more levels than valid and test
tr <- train %>% select(Weather_Condition) %>% distinct()
va <- valid %>% select(Weather_Condition) %>% distinct()
te <- test %>% select(Weather_Condition) %>% distinct()
setdiff(va, tr)
```

```
## # A tibble: 0 x 1
## # ... with 1 variable: Weather_Condition <fct>
```

```r
setdiff(te, tr)
```

```
## # A tibble: 0 x 1
## # ... with 1 variable: Weather_Condition <fct>
```

(9) Sampling

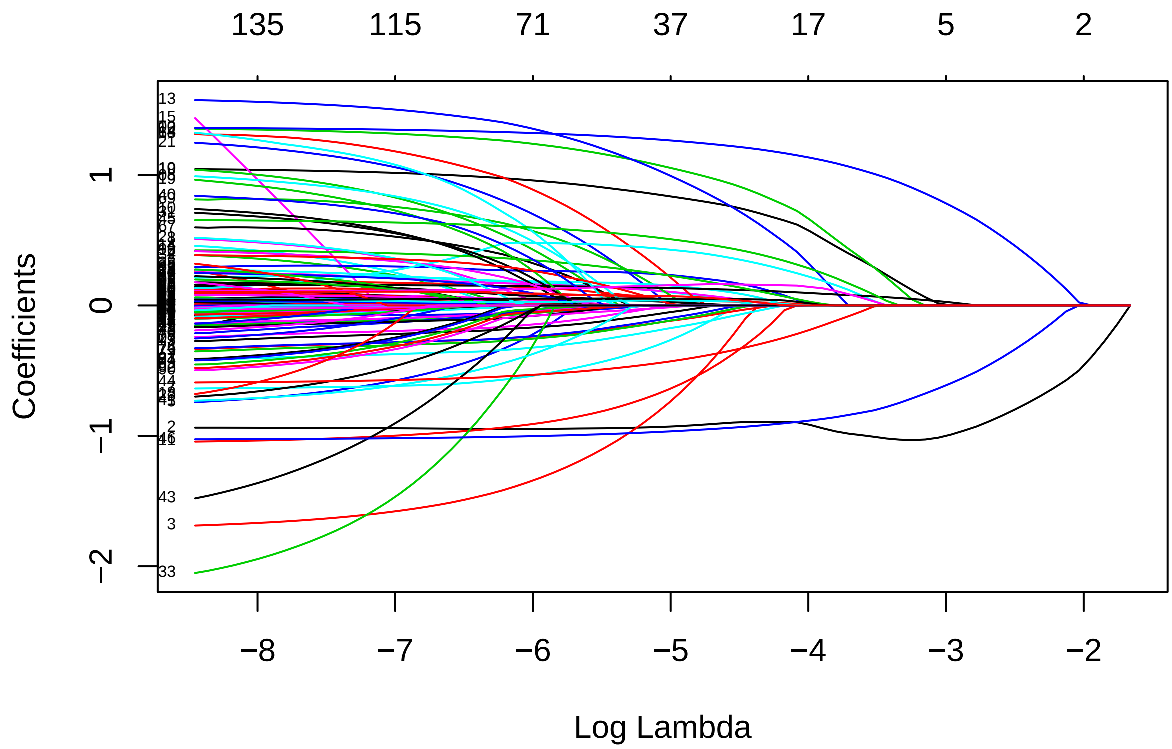
```r
new_train <- ovun.sample(Status ~ .,
                         data = train %>% select(-Severity),
                         method = "both", p = 0.5, N = 90000)$data %>% as_tibble()
table(new_train$Status)
```

```
##
## Not Severe     Severe
##      44978      45022
```

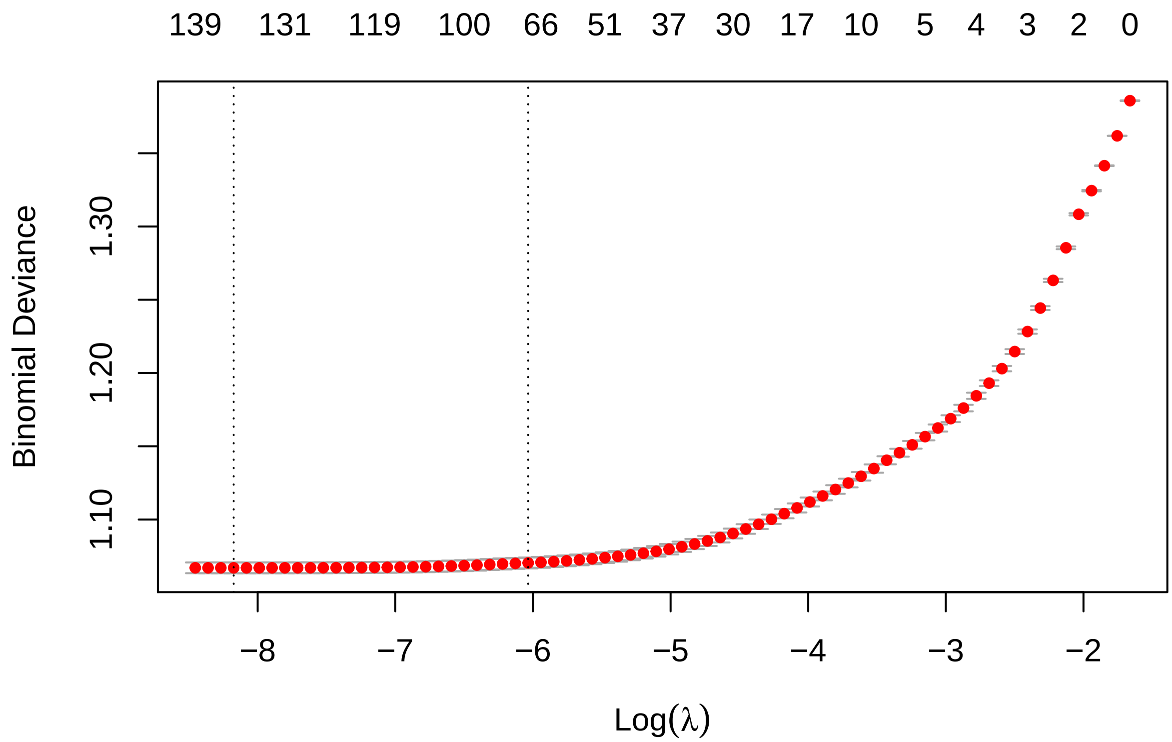
3. Use different models to fit the data

(1) Sparse Logistic regression

```r
x <- model.matrix(Status ~ ., data = new_train)
model_total <- glmnet(x, new_train$Status, family = "binomial")
plot(model_total, xvar = "lambda", label = T)
```




```
model_lambda <- cv.glmnet(x, new_train$Status, family = "binomial")
plot(model_lambda)
```




```r
# use the best lambda
valid_pred <- valid %>%
  mutate("pred" = predict(model_lambda,
                          newx = model.matrix(Status ~ ., data = valid %>% select(-Severity)),
                          s = "lambda.min", type = "response")[,1]) %>%
  mutate("pred" = ifelse(pred > 0.5, "Severe", "Not Severe"))
valid_pred %>% select(Status, pred)
```

```
## # A tibble: 58,231 x 2
##    Status     pred
##    <fct>      <chr>
##  1 Not Severe Not Severe
##  2 Not Severe Not Severe
##  3 Not Severe Not Severe
##  4 Severe     Severe
##  5 Severe     Severe
##  6 Not Severe Not Severe
##  7 Not Severe Severe
##  8 Not Severe Not Severe
##  9 Not Severe Not Severe
## 10 Not Severe Not Severe
## # ... with 58,221 more rows
```

```r
table(valid$Status)
```

```
##
## Not Severe     Severe
```

```
##      42154      16077
```

```
confusionMatrix(table(valid_pred$Status, valid_pred$pred))
```

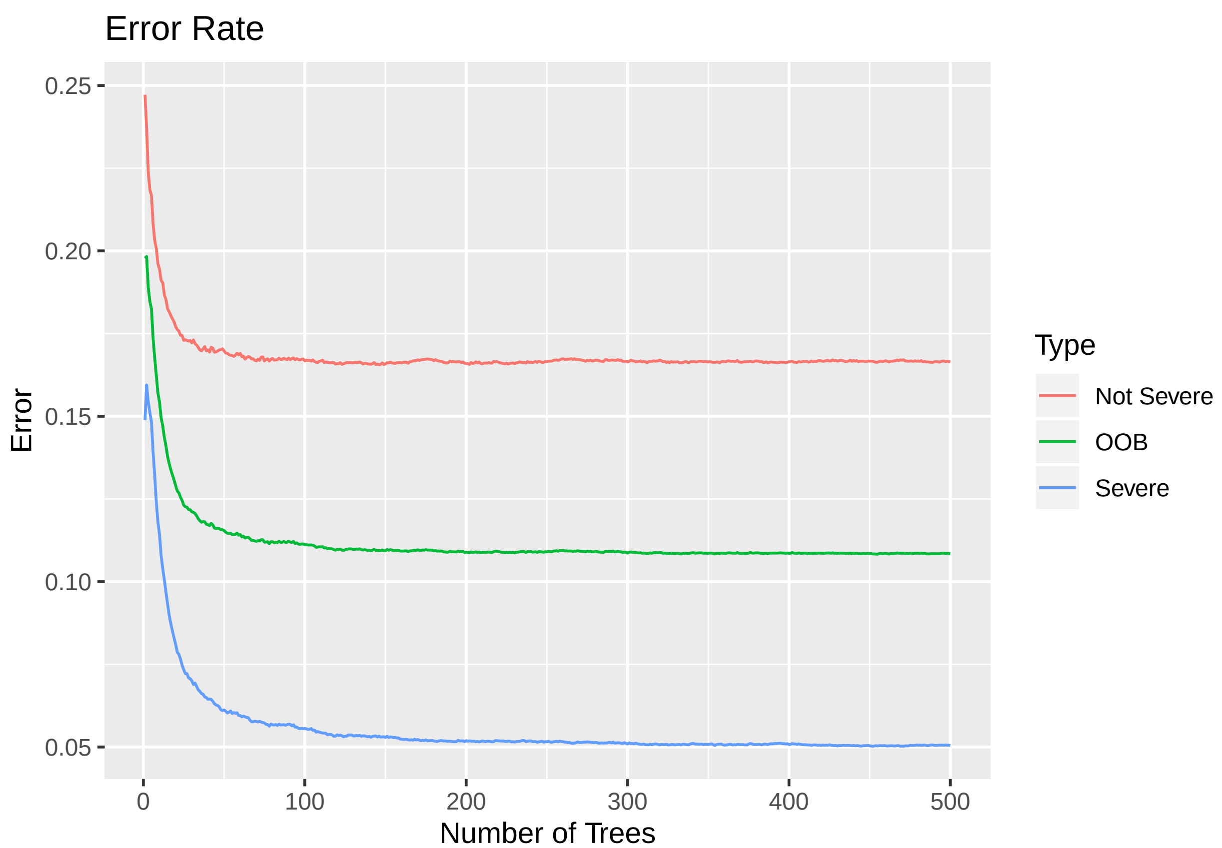
```
## Confusion Matrix and Statistics
##
##
##                Not Severe Severe
##    Not Severe      31410  10744
##    Severe           4332  11745
##
##                  Accuracy : 0.7411
##                    95% CI : (0.7375, 0.7447)
##     No Information Rate : 0.6138
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                     Kappa : 0.4234
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##               Sensitivity : 0.8788
##               Specificity : 0.5223
##            Pos Pred Value : 0.7451
##            Neg Pred Value : 0.7305
##                Prevalence : 0.6138
##            Detection Rate : 0.5394
##      Detection Prevalence : 0.7239
##         Balanced Accuracy : 0.7005
##
##          'Positive' Class : Not Severe
##
```

(2) Random forest

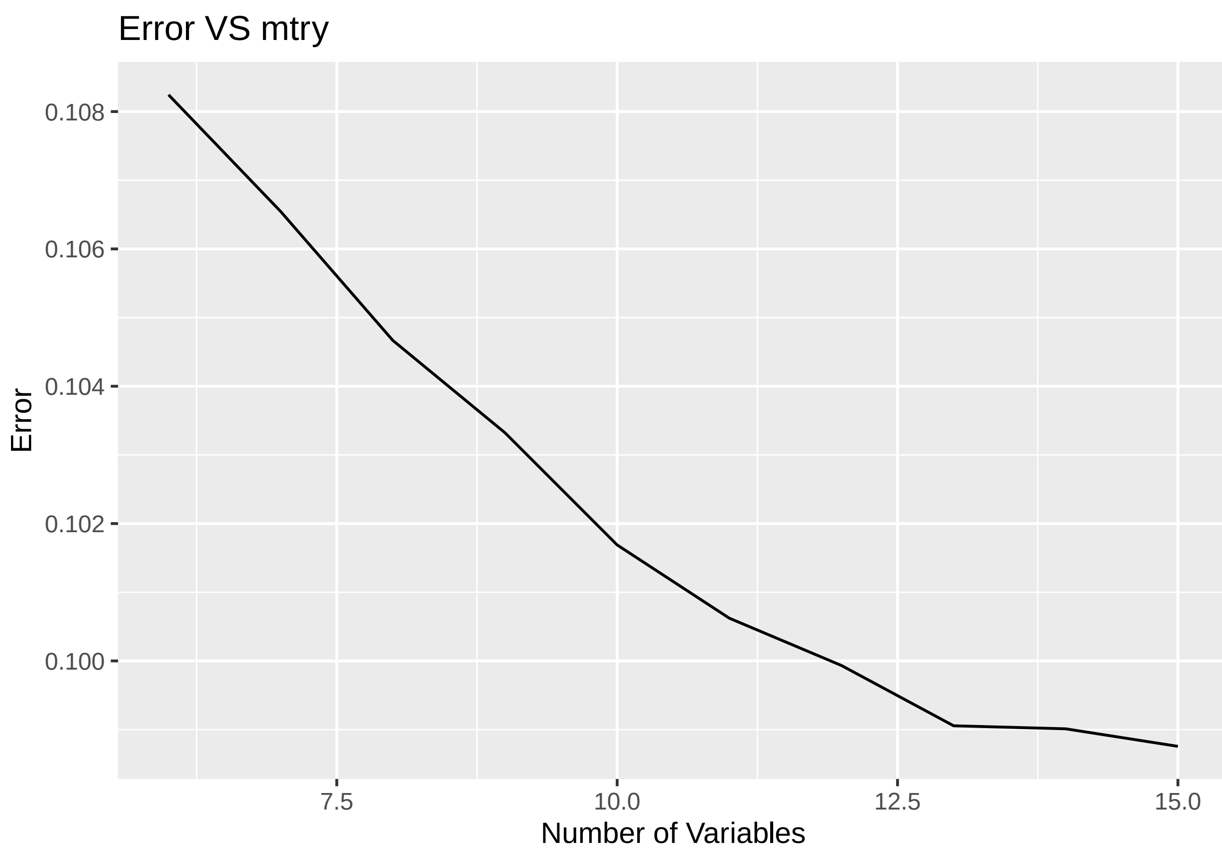
```
model <- randomForest(Status ~ ., data = new_train, mtry = 6, ntree = 500)

# see if ntree = 500 is enough
error_data <- model$err.rate %>%
  as_tibble() %>%
  mutate("Trees" = seq_along(OOB)) %>%
  pivot_longer(cols = 1:3, names_to = "Type", values_to = "Error")

ggplot(error_data, aes(Trees, Error, color = Type)) +
  geom_line() +
  labs(x = "Number of Trees",
       title = "Error Rate")
```




```r
# try different mtry
oob_values <- vector(length = 10)
for (i in 1:10) {
  temp_model <- randomForest(Status ~ ., data = new_train, mtry = (i + 5))
  oob_values[i] <- temp_model$err.rate[nrow(temp_model$err.rate), 1]
}
ggplot(tibble("Error" = oob_values), aes(x = 6:15, y = Error)) +
  geom_line(aes(group = 1)) +
  labs(x = "Number of Variables",
       title = "Error VS mtry")
```




```r
# choose mtry = 12 as the best model
best_model <- randomForest(Status ~ ., data = new_train, mtry = 12, ntree = 500)

valid_pred_rf <- valid %>%
  add_predictions(best_model)
table(valid_pred_rf$Status)
```

```
##
## Not Severe     Severe
##      42154      16077
```

```r
confusionMatrix(valid_pred_rf$Status, valid_pred_rf$pred)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction   Not Severe Severe
##   Not Severe      33096   9058
##   Severe           2143  13934
##
##                Accuracy : 0.8076
##                  95% CI : (0.8044, 0.8108)
##     No Information Rate : 0.6052
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5753
##
```

```
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9392
##             Specificity : 0.6060
##          Pos Pred Value : 0.7851
##          Neg Pred Value : 0.8667
##              Prevalence : 0.6052
##          Detection Rate : 0.5684
##    Detection Prevalence : 0.7239
##       Balanced Accuracy : 0.7726
##
##        'Positive' Class : Not Severe
##
```

(3) Decision tree

```r
model_decision <- rpart(Status ~ ., data = new_train, method = "class")

valid_pred_dc <- valid %>%
  mutate(pred = predict(model_decision, valid, type = "class"))
confusionMatrix(table(valid_pred_dc$Status, valid_pred_dc$pred))
```

```
## Confusion Matrix and Statistics
##
##
##               Not Severe Severe
##   Not Severe      30795  11359
##   Severe           3894  12183
##
##                Accuracy : 0.7381
##                  95% CI : (0.7345, 0.7416)
##     No Information Rate : 0.5957
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.427
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.8877
##             Specificity : 0.5175
##          Pos Pred Value : 0.7305
##          Neg Pred Value : 0.7578
##              Prevalence : 0.5957
##          Detection Rate : 0.5288
##    Detection Prevalence : 0.7239
##       Balanced Accuracy : 0.7026
##
##        'Positive' Class : Not Severe
##
```