random-froest-modeling

Dorothy Leung

12/7/2019

Import Library

```
library(dplyr) # data manipulation
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caTools) # splits
library(ggplot2) # plot graph
library(randomForest) # Random Forest
```

```
## randomForest 4.6-14
```

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

```
##
## Attaching package: 'randomForest'
```

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

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

```
library(rpart)
library(caret)
```

```
## Loading required package: lattice
```

```
OSR2 <- function(predictions, test, train) {
  SSE <- sum((test - predictions)^2)
  SST <- sum((test - mean(train))^2)
  r2 <- 1 - SSE/SST
  return(r2)
}
```

```
suicides <- read.csv("suicide_rates_no_na.csv")
head(suicides)
```

```
##   HDI.for.year         age       country      country.year
## 1        0.841   75+ years United States United States1985
## 2        0.841 55-74 years United States United States1985
## 3        0.841 25-34 years United States United States1985
## 4        0.841 35-54 years United States United States1985
## 5        0.841 15-24 years United States United States1985
## 6        0.841 35-54 years United States United States1985
##   gdp_per_capita....      generation population    sex suicides.1
00k.pop
## 1              19693 G.I. Generation    4064000   male
53.57
## 2              19693 G.I. Generation   17971000   male
29.50
## 3              19693         Boomers   20986000   male
24.46
## 4              19693          Silent   26589000   male
22.77
## 5              19693    Generation X   19962000   male
21.38
## 6              19693          Silent   27763000 female
7.58
##   suicides_no year gdp_for_year....
## 1        2177 1985     4.346734e+12
## 2        5302 1985     4.346734e+12
## 3        5134 1985     4.346734e+12
## 4        6053 1985     4.346734e+12
## 5        4267 1985     4.346734e+12
## 6        2105 1985     4.346734e+12
```

```
dim(suicides)
```

```
## [1] 744  12
```

```
colnames(suicides)
```

```
##  [1] "HDI.for.year"       "age"                "country"
##  [4] "country.year"       "gdp_per_capita...." "generation"
##  [7] "population"         "sex"                "suicides.100k.pop
"
## [10] "suicides_no"        "year"               "gdp_for_year...."
```

```
suicides.us <- suicides %>% filter(country == "United States") %>% s
elect(suicides.100k.pop, HDI.for.year, gdp_per_capita...., year, gdp
_for_year...., sex)
```

```
suicides.jap <- suicides %>% filter(country == "Japan") %>% select(s
uicides.100k.pop, HDI.for.year, gdp_per_capita...., year, gdp_for_ye
ar...., sex)
```

```
suicides.us <- suicides.us %>% arrange(year)
suicides.jap <- suicides.jap %>% arrange(year)
```

```
# checking nan values in the dataframe
sapply(suicides.us, function(x)all(any(is.na(x))))
```

```
##  suicides.100k.pop       HDI.for.year gdp_per_capita....
##              FALSE               FALSE              FALSE
##               year   gdp_for_year....                sex
##              FALSE               FALSE              FALSE
```

```
sapply(suicides.jap, function(x)all(any(is.na(x))))
```

```
##  suicides.100k.pop       HDI.for.year gdp_per_capita....
##              FALSE               FALSE              FALSE
##               year   gdp_for_year....                sex
##              FALSE               FALSE              FALSE
```

0.85 for us nan 0.8025 for japan nan

```
# suicdes.us <- suicides.us %>% replace(., is.na(.), 0.85)
# suicides.jap <- suicides.jap %>% replace(., is.na(.), 0.8025)
```

Split Data

```
set.seed(377)

# US dataset
us.train.ids <- sample(nrow(suicides.us), 0.70*nrow(suicides.us)) #
70/30 split
us.train <- suicides.us[us.train.ids, ]
us.test <- suicides.us[-us.train.ids, ]

# Japan dataset
jap.train.ids <- sample(nrow(suicides.jap), 0.70*nrow(suicides.jap))
# 70/30 split
jap.train <- suicides.jap[jap.train.ids, ]
jap.test <- suicides.jap[-jap.train.ids, ]
```

```
# ===== Random forest on US dataset =====
set.seed(377)
mod.rf.us <- randomForest(suicides.100k.pop ~ ., data = us.train, mt
ry = 5, nodesize = 5, ntree = 500)
## mtry: Number of variables randomly sampled as candidates at each
split.
## nodesize: Minimum size of terminal nodes.
## ntree: Number of trees to grow.
```

```
pred.rf.us <- predict(mod.rf.us, newdata = us.test) # just to illust
rate
pred.rf.us[1:5]
```

```
##         6        10        13        14        15
##  5.027277  5.027277 23.300426 23.300426 23.300426
```

```
importance(mod.rf.us)
```

```
##                      IncNodePurity
## HDI.for.year              2191.199
## gdp_per_capita....        1734.562
## year                      1794.989
## gdp_for_year....          1652.286
## sex                      23179.523
```

```
train.rf.us <- train(suicides.100k.pop ~ .,
                     data = us.train,
                     method = "rf",
                     tuneGrid = data.frame(mtry=1:5),
                     trControl = trainControl(method="cv", number=5,
verboseIter = TRUE),
                     metric = "RMSE")
```

```
## + Fold1: mtry=1
## - Fold1: mtry=1
## + Fold1: mtry=2
## - Fold1: mtry=2
## + Fold1: mtry=3
## - Fold1: mtry=3
## + Fold1: mtry=4
## - Fold1: mtry=4
## + Fold1: mtry=5
## - Fold1: mtry=5
## + Fold2: mtry=1
## - Fold2: mtry=1
## + Fold2: mtry=2
## - Fold2: mtry=2
## + Fold2: mtry=3
## - Fold2: mtry=3
## + Fold2: mtry=4
## - Fold2: mtry=4
## + Fold2: mtry=5
## - Fold2: mtry=5
## + Fold3: mtry=1
## - Fold3: mtry=1
## + Fold3: mtry=2
## - Fold3: mtry=2
## + Fold3: mtry=3
```

```
## - Fold3: mtry=3
## + Fold3: mtry=4
## - Fold3: mtry=4
## + Fold3: mtry=5
## - Fold3: mtry=5
## + Fold4: mtry=1
## - Fold4: mtry=1
## + Fold4: mtry=2
## - Fold4: mtry=2
## + Fold4: mtry=3
## - Fold4: mtry=3
## + Fold4: mtry=4
## - Fold4: mtry=4
## + Fold4: mtry=5
## - Fold4: mtry=5
## + Fold5: mtry=1
## - Fold5: mtry=1
## + Fold5: mtry=2
## - Fold5: mtry=2
## + Fold5: mtry=3
## - Fold5: mtry=3
## + Fold5: mtry=4
## - Fold5: mtry=4
## + Fold5: mtry=5
## - Fold5: mtry=5
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 1 on full training set
```

```
train.rf.us$results
```

```
##   mtry     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1    1 11.03930 0.3477123 7.113677 1.356406  0.1333967 0.9094582
## 2    2 11.48702 0.3266464 7.208276 1.836438  0.1440097 1.2967554
## 3    3 11.53989 0.3335956 7.264496 1.961984  0.1476955 1.4355032
## 4    4 11.50868 0.3379362 7.259775 2.013350  0.1495024 1.4588513
## 5    5 11.53750 0.3364317 7.289534 2.014691  0.1491697 1.4687086
```

```
train.rf.us
```

```
## Random Forest
##
## 260 samples
##   5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 208, 208, 208, 208, 208
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##   1     11.03930  0.3477123  7.113677
##   2     11.48702  0.3266464  7.208276
##   3     11.53989  0.3335956  7.264496
##   4     11.50868  0.3379362  7.259775
##   5     11.53750  0.3364317  7.289534
##
## RMSE was used to select the optimal model using the smallest valu
e.
## The final value used for the model was mtry = 1.
```

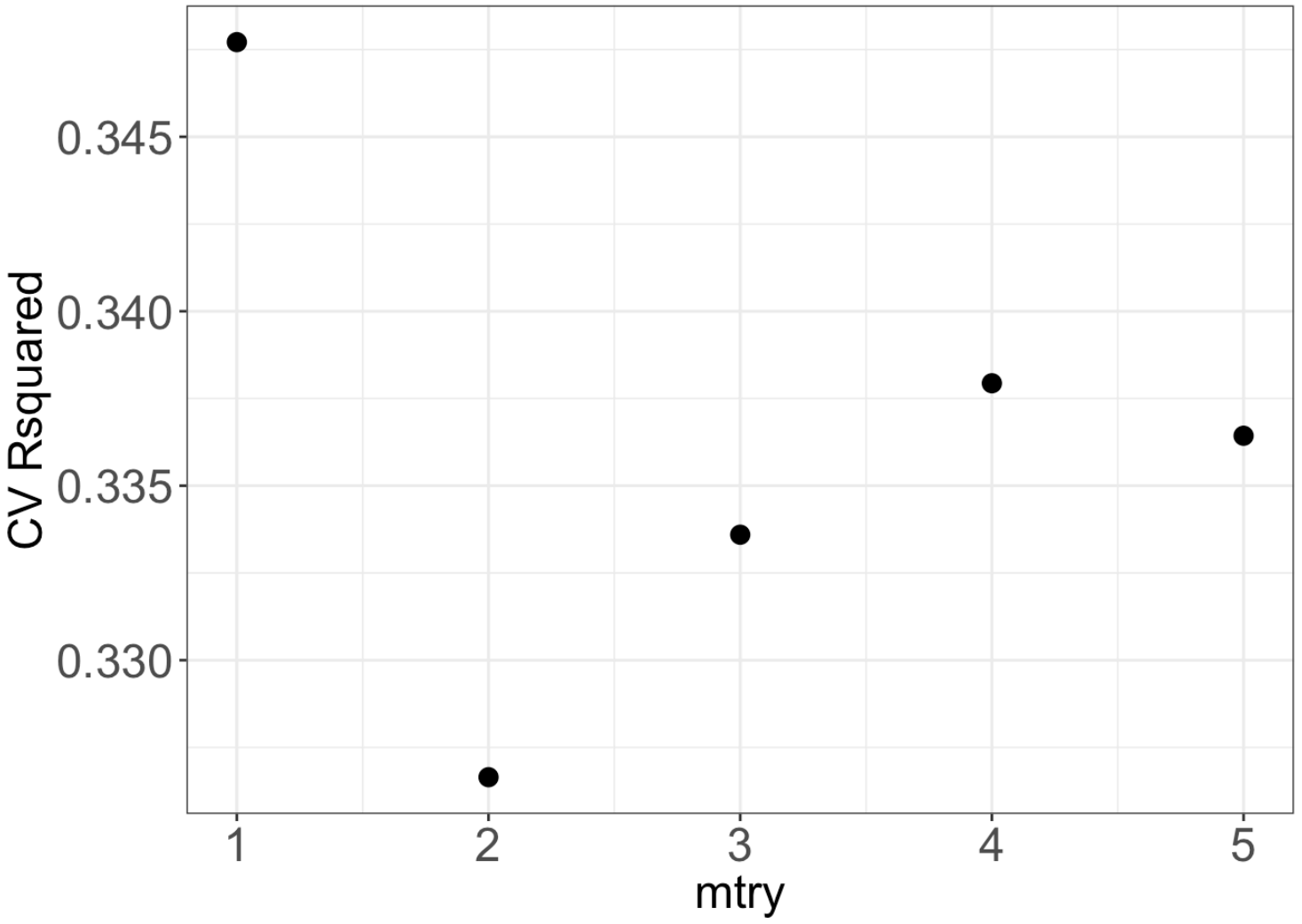
```
best.rf.us <- train.rf.us$finalModel
```

```
us.test.oo = as.data.frame(model.matrix(suicides.100k.pop ~ . + 0, d
ata = us.test))
```

```
pred.best.rf.us <- predict(best.rf.us, newdata = us.test.oo)
pred.best.rf.us[1:5]
```

```
##        6       10       13       14       15
## 10.07762 10.07762 20.37441 20.37441 20.37441
```

```
ggplot(train.rf.us$results, aes(x = mtry, y = Rsquared)) + geom_poin
t(size = 3) +
  ylab("CV Rsquared") + theme_bw() + theme(axis.title=element_text(s
ize=18), axis.text=element_text(size=18))
```




```
# ===== Random forest on Japan dataset =====
set.seed(377)
mod.rf.jap <- randomForest(suicides.100k.pop ~ ., data = jap.train,
mtry = 5, nodesize = 5, ntree = 500)
## mtry: Number of variables randomly sampled as candidates at each
split.
## nodesize: Minimum size of terminal nodes.
## ntree: Number of trees to grow.
```

```
pred.rf.jap <- predict(mod.rf.jap, newdata = jap.test) # just to ill
ustrate
pred.rf.jap[1:5]
```

```
##        2        6        7       10       11
## 10.85131 10.85131 10.85131 10.85131 38.04045
```

```
importance(mod.rf.jap)
```

```
##                     IncNodePurity
## HDI.for.year             4407.085
## gdp_per_capita....       4119.129
## year                     5793.620
## gdp_for_year....         4005.144
## sex                     15177.759
```

```
train.rf.jap <- train(suicides.100k.pop ~ .,
                      data = jap.train,
                      method = "rf",
                      tuneGrid = data.frame(mtry=1:5),
                      trControl = trainControl(method="cv", number=5
, verboseIter = TRUE),
                      metric = "RMSE")
```

```
## + Fold1: mtry=1
## - Fold1: mtry=1
## + Fold1: mtry=2
## - Fold1: mtry=2
## + Fold1: mtry=3
## - Fold1: mtry=3
## + Fold1: mtry=4
## - Fold1: mtry=4
## + Fold1: mtry=5
## - Fold1: mtry=5
## + Fold2: mtry=1
## - Fold2: mtry=1
## + Fold2: mtry=2
## - Fold2: mtry=2
## + Fold2: mtry=3
## - Fold2: mtry=3
## + Fold2: mtry=4
## - Fold2: mtry=4
## + Fold2: mtry=5
## - Fold2: mtry=5
## + Fold3: mtry=1
## - Fold3: mtry=1
```

```
## + Fold3: mtry=2
## - Fold3: mtry=2
## + Fold3: mtry=3
## - Fold3: mtry=3
## + Fold3: mtry=4
## - Fold3: mtry=4
## + Fold3: mtry=5
## - Fold3: mtry=5
## + Fold4: mtry=1
## - Fold4: mtry=1
## + Fold4: mtry=2
## - Fold4: mtry=2
## + Fold4: mtry=3
## - Fold4: mtry=3
## + Fold4: mtry=4
## - Fold4: mtry=4
## + Fold4: mtry=5
## - Fold4: mtry=5
## + Fold5: mtry=1
## - Fold5: mtry=1
## + Fold5: mtry=2
## - Fold5: mtry=2
## + Fold5: mtry=3
## - Fold5: mtry=3
## + Fold5: mtry=4
## - Fold5: mtry=4
## + Fold5: mtry=5
## - Fold5: mtry=5
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 1 on full training set
```

```
train.rf.jap$results
```

```
##   mtry     RMSE   Rsquared      MAE   RMSESD RsquaredSD    MAESD
## 1    1 16.66777 0.08689917 13.27282 1.357727  0.1079083 1.130086
## 2    2 17.91068 0.06229225 14.27061 1.908774  0.1063650 1.658064
## 3    3 18.04017 0.06254494 14.36435 1.979002  0.1055922 1.730786
## 4    4 18.00821 0.06373223 14.31217 1.927709  0.1038813 1.734757
## 5    5 18.08593 0.05998491 14.39434 1.930205  0.1014877 1.761266
```

```
train.rf.jap
```

```
## Random Forest
##
## 260 samples
##   5 predictor
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 208, 208, 208, 208, 208
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared    MAE
##   1     16.66777  0.08689917  13.27282
##   2     17.91068  0.06229225  14.27061
##   3     18.04017  0.06254494  14.36435
##   4     18.00821  0.06373223  14.31217
##   5     18.08593  0.05998491  14.39434
##
## RMSE was used to select the optimal model using the smallest valu
e.
## The final value used for the model was mtry = 1.
```

```
best.rf.jap <- train.rf.jap$finalModel
```

```
jap.test.oo = as.data.frame(model.matrix(suicides.100k.pop ~ . + 0,
data = jap.test))
```

```
pred.best.rf.jap <- predict(best.rf.jap, newdata = jap.test.oo)
pred.best.rf.jap[1:5]
```

```
##        2        6        7       10       11
## 25.95633 25.95633 25.95633 25.95633 35.12114
```

```
ggplot(train.rf.jap$results, aes(x = mtry, y = Rsquared)) + geom_poi
nt(size = 3) +
  ylab("CV Rsquared") + theme_bw() + theme(axis.title=element_text(s
ize=18), axis.text=element_text(size=18))
```