random Forest

Angel Chen

12/7/2018

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------- tidyverse 1.2.1 --

NA
NA
NA
NA

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

```
setwd("~/Downloads")
dat_raw = read.csv("data.csv")
dat_raw = dat_raw[, 1:32]
```

```
dat_covar = dat_raw %>% select(-id, -diagnosis)
dat_label = cbind(dat_raw$diagnosis) %>% as.data.frame
```

```
set.seed(123)
index = sample(1:nrow(dat_raw), size = trunc(.8 * nrow(dat_raw)))
dat_raw$diagnosis = as.integer(factor(dat_raw$diagnosis))-1
Train <- dat_raw[index,-1]
Test <- dat_raw[-index,-1]
```

```
#implement random forest
library(randomForest)
```

```
## randomForest 4.6-12

## 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
```

```
library(caret)
```

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

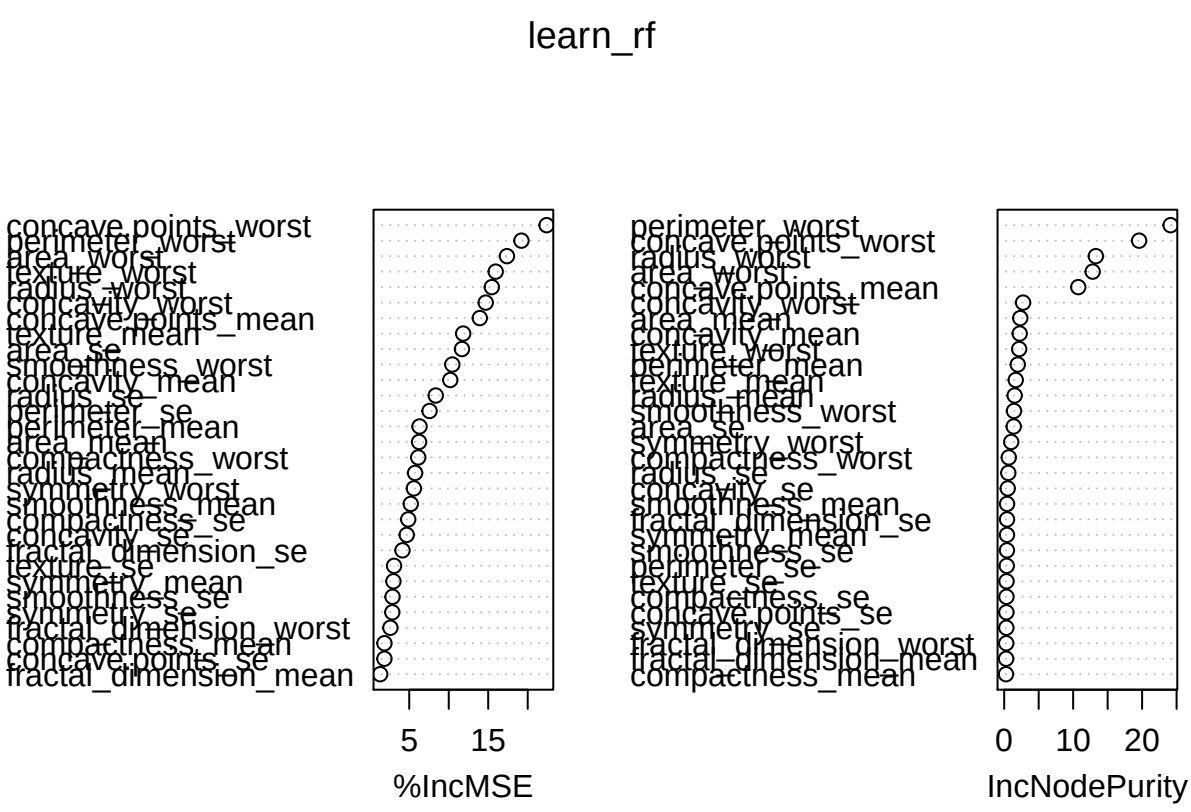
```
##
## Attaching package: 'caret'
```

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

```
learn_rf <- randomForest(diagnosis~., data=Train, ntree=500, proximity=T, importance=T)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
#variance importance plot
varImpPlot(learn_rf)
```




```
#append predicted values onto test set to plot confusion matrix
Test$predicted <- round(predict(learn_rf ,Test),0)

plotConfusionMatrix <- function(testset, sSubtitle) {
    tst <- data.frame(testset$predicted, testset$diagnosis)
    opts <- c("Predicted", "True")
```

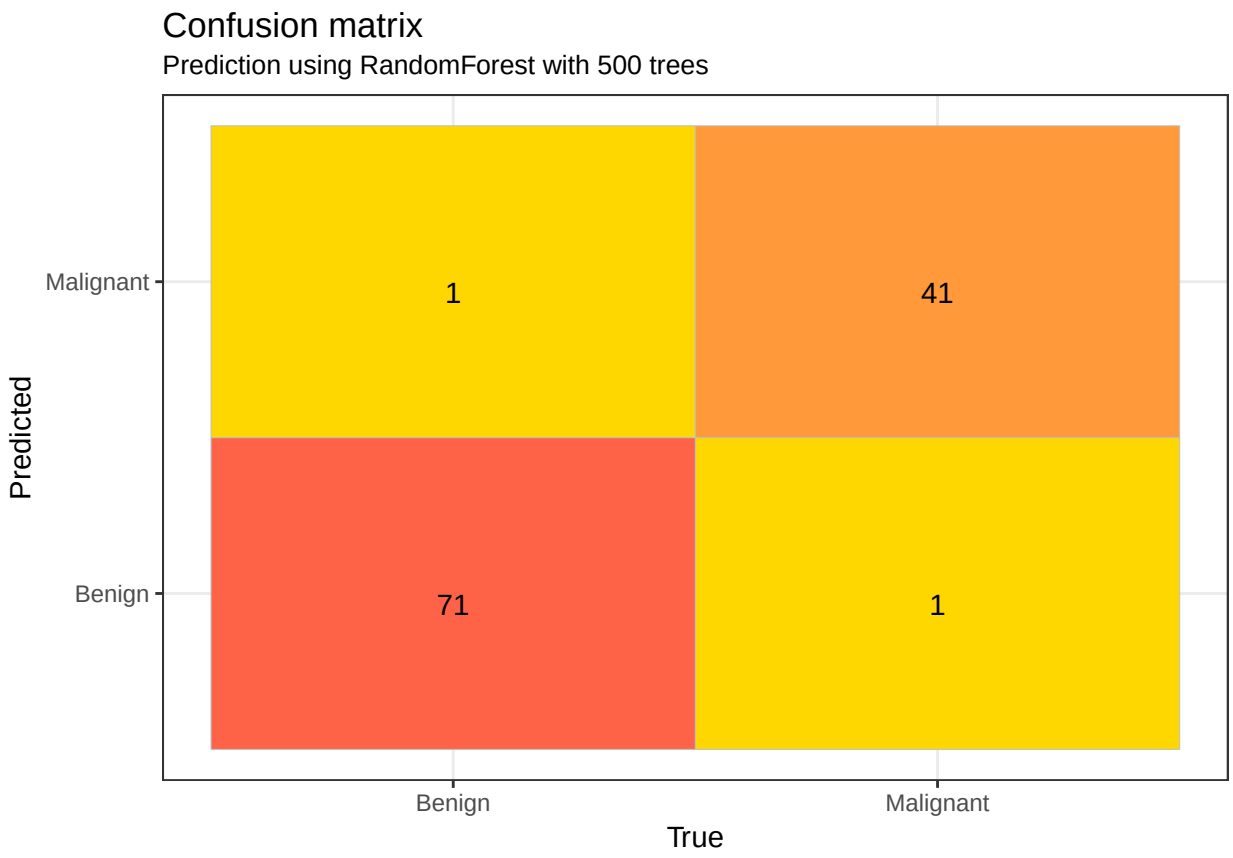
```
    names(tst) <- opts
    cf <- plyr::count(tst)
    cf[opts][cf[opts]==0] <- "Benign"
    cf[opts][cf[opts]==1] <- "Malignant"
    ggplot(data =  cf, mapping = aes(x = True, y = Predicted)) +
      labs(title = "Confusion matrix", subtitle = sSubtitle) +
      geom_tile(aes(fill = freq), colour = "grey") +
      geom_text(aes(label = sprintf("%1.0f", freq)), vjust = 1) +
      scale_fill_gradient(low = "gold", high = "tomato") +
      theme_bw() + theme(legend.position = "none")
}

#calculate and plot AUC
library(pROC)
```

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

##
## Attaching package: 'pROC'

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

```
print(sprintf("Area under curve (AUC) : %.3f",auc(Test$diagnosis, Test$predicted)))
```

```
## [1] "Area under curve (AUC) : 0.981"
```

```
plotConfusionMatrix(Test,"Prediction using RandomForest with 500 trees")
```




```
#check prediction accuracy
```

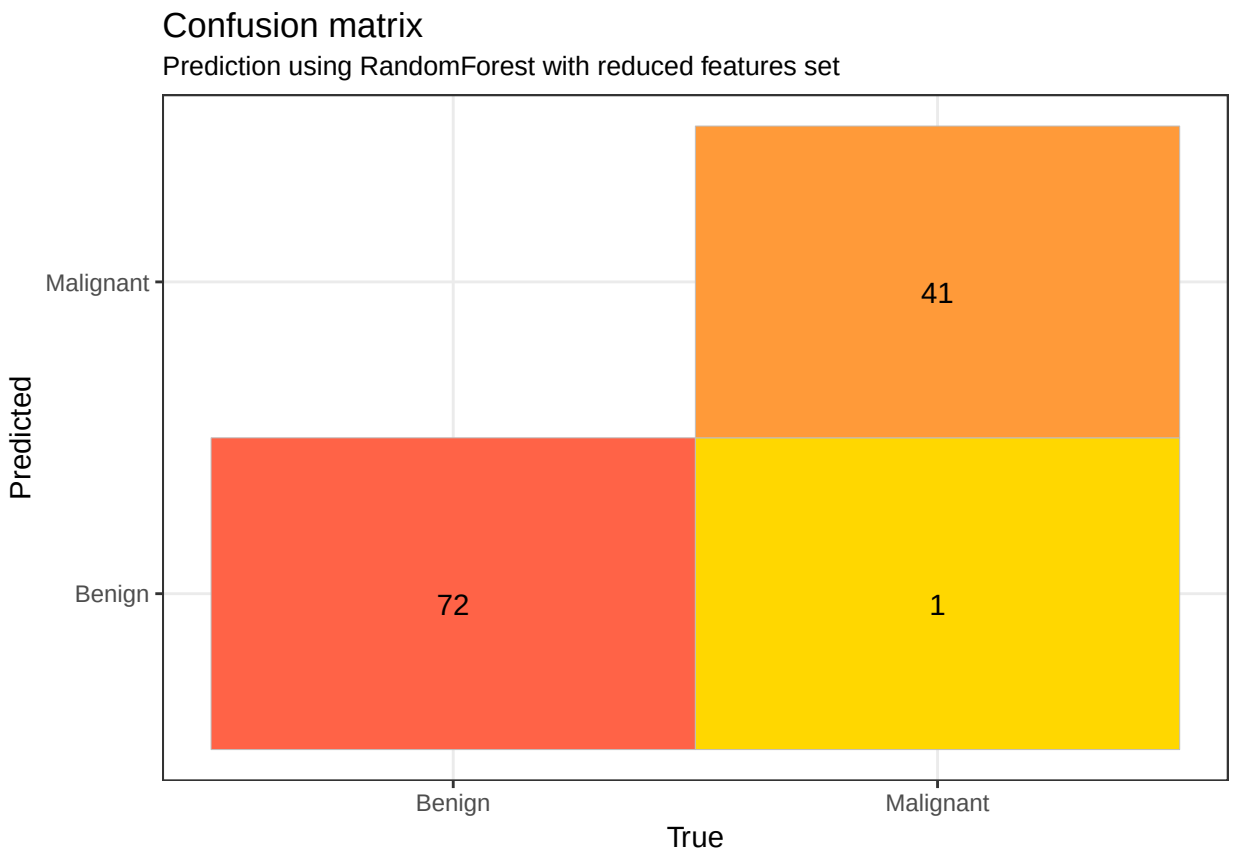
Refine covariates according to the result of the varimpplot

```
features_list <-  c("perimeter_worst", "area_worst", "concave.points_worst", "radius_worst", "concavity
 "radius_mean", "radius_se", "perimeter_mean", "perimeter_se", "compactness_worst", "smoothness_worst",



#define train and validation set

Train_red = Train[,features_list]

Test_red =  Test[,features_list]

#training
train_red_rf <- randomForest(diagnosis ~.,Train_red,ntree=500,importance=T)
```

```
## Warning in randomForest.default(m, y, ...): The response has five or fewer
## unique values. Are you sure you want to do regression?
```

```
#get prediction
Test_red$predicted <- round(predict(train_red_rf ,Test_red),0)
```

```r
plotConfusionMatrix(Test_red,"Prediction using RandomForest with reduced features set")
```




```r
#increase in true negative
#calculate AUC
print(sprintf("Area under curve (AUC) : %.3f",auc(Test_red$diagnosis, Test_red$predicted)))
```

```
## [1] "Area under curve (AUC) : 0.988"
```

```r
#reducing the features only increased prediction rate by 0.007
```