Ensemble Techniques - Neo Zhao & Andrew Sen - CS4375

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.5     v purrr   0.3.4
## v tibble  3.1.8     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'tibble' was built under R version 4.1.3
```

```
## Warning: package 'tidyr' was built under R version 4.1.3
```

```
## Warning: package 'readr' was built under R version 4.1.3
```

```
## Warning: package 'purrr' was built under R version 4.1.3
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
## Warning: package 'forcats' was built under R version 4.1.3
```

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

```
library(dplyr)
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.1.3
```

```
library(mccr)
```

```
## Warning: package 'mccr' was built under R version 4.1.3
```

```
library(ISLR)
```

```
## Warning: package 'ISLR' was built under R version 4.1.3
```

```r
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.1.3
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(tree)
```

```
## Warning: package 'tree' was built under R version 4.1.3
```

```r
library(rpart)
library(e1071)
```

```
## Warning: package 'e1071' was built under R version 4.1.3
```

```r
# Source: https://www.kaggle.com/datasets/budnyak/wine-rating-and-price?select=Red.csv

# Red, White, Rose, and Sparkling wine are all from the same dataset; however, separated by type

# Red Total: 8666
Red <- read.csv("Red.csv")

# White Total: 3764
White <- read.csv("White.csv")

# Rose Total: 397
Rose <- read.csv("Rose.csv")

# Sparkling Total: 1007
Sparkling <- read.csv("Sparkling.csv")

# Combine the datasets together, Total: 13058
totalWine <- rbind(data = Red, data = White, data = Rose, data = Sparkling)

# Rename ï..Name to just Name
names(totalWine)[1] <- "Name"

# Omit Names, Winery, & Region Column
totalWine <- subset(totalWine, select = -c(Name, Winery, Region))

# Omit all records where Year = N.V.
totalWine <- subset(totalWine, totalWine$Year != "N.V.")

# Omit all records before 2000s
totalWine <- subset(totalWine, totalWine$Year >= 2000)
```

```r
# Make the Year from chr -> num
totalWine$Year <- as.numeric(totalWine$Year)

# Set Country from chr -> factor
totalWine$Country <- as.factor(totalWine$Country)

totalWine$Rating <- round(totalWine$Rating / 0.5) * 0.5

totalWine$Rating[totalWine$Rating == 0.5] <- 1
totalWine$Rating[totalWine$Rating == 1] <- 2
totalWine$Rating[totalWine$Rating == 1.5] <- 3
totalWine$Rating[totalWine$Rating == 2] <- 4
totalWine$Rating[totalWine$Rating == 2.5] <- 5
totalWine$Rating[totalWine$Rating == 3] <- 6
totalWine$Rating[totalWine$Rating == 3.5] <- 7
totalWine$Rating[totalWine$Rating == 4] <- 8
totalWine$Rating[totalWine$Rating == 4.5] <- 9
totalWine$Rating[totalWine$Rating == 5] <- 10

# Reorder Columns
totalWine <- totalWine[,c(1,2,3,5,4)]

# train/test
set.seed(1234)
i <- sample(1:nrow(totalWine), nrow(totalWine)*0.8, replace=FALSE)
train <- totalWine[i,]
test <- totalWine[-i,]
```

Random Forest

```r
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.1.3

## randomForest 4.7-1.1

## 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
```

```
set.seed(512)
train_copy <- train
train_copy$Rating <- as.factor(train_copy$Rating)
start_time <- proc.time()
rf <- randomForest(Rating ~ ., data = train_copy, importance = TRUE)
end_time <- proc.time()

summary(rf)
```

```
##                 Length Class  Mode
## call               4  -none- call
## type               1  -none- character
## predicted      10426  factor numeric
## err.rate        3000  -none- numeric
## confusion         30  -none- numeric
## votes          52130  matrix numeric
## oob.times      10426  -none- numeric
## classes            5  -none- character
## importance        28  -none- numeric
## importanceSD      24  -none- numeric
## localImportance    0  -none- NULL
## proximity          0  -none- NULL
## ntree              1  -none- numeric
## mtry               1  -none- numeric
## forest            14  -none- list
## y              10426  factor numeric
## test               0  -none- NULL
## inbag              0  -none- NULL
## terms              3  terms  call
```

```
print(paste("Training time: ", (end_time-start_time)[[3]], "s"))
```

```
## [1] "Training time:  109.22 s"
```

```
pred <- predict(rf, newdata = test, type = "response")
pred <- as.character(pred)
acc_rf <- mean(pred == test$Rating)
print(paste("Accuracy = ", acc_rf))
```

```
## [1] "Accuracy =  0.688147295742232"
```

XGBoost

```
library(xgboost)
```

```
## Warning: package 'xgboost' was built under R version 4.1.3
```

```
##
## Attaching package: 'xgboost'
```

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

```
# need to convert dataframe to matrix
train_matrix <- data.matrix(train)

start_time <- proc.time()
xg <- xgboost(data=train_matrix, label=train$Rating, nrounds=100, objective='multi:softprob', num_class=
```

```
## [1]  train-mlogloss:0.969228
## [2]  train-mlogloss:0.662372
## [3]  train-mlogloss:0.470101
## [4]  train-mlogloss:0.339450
## [5]  train-mlogloss:0.247427
## [6]  train-mlogloss:0.181403
## [7]  train-mlogloss:0.133493
## [8]  train-mlogloss:0.098468
## [9]  train-mlogloss:0.072790
## [10] train-mlogloss:0.053897
## [11] train-mlogloss:0.039966
## [12] train-mlogloss:0.029679
## [13] train-mlogloss:0.022076
## [14] train-mlogloss:0.016451
## [15] train-mlogloss:0.012308
## [16] train-mlogloss:0.009241
## [17] train-mlogloss:0.006968
## [18] train-mlogloss:0.005263
## [19] train-mlogloss:0.004014
## [20] train-mlogloss:0.003086
## [21] train-mlogloss:0.002420
## [22] train-mlogloss:0.001917
## [23] train-mlogloss:0.001545
## [24] train-mlogloss:0.001267
## [25] train-mlogloss:0.001056
## [26] train-mlogloss:0.000908
## [27] train-mlogloss:0.000778
## [28] train-mlogloss:0.000696
## [29] train-mlogloss:0.000635
## [30] train-mlogloss:0.000610
## [31] train-mlogloss:0.000590
## [32] train-mlogloss:0.000573
## [33] train-mlogloss:0.000559
## [34] train-mlogloss:0.000546
## [35] train-mlogloss:0.000535
## [36] train-mlogloss:0.000525
## [37] train-mlogloss:0.000517
## [38] train-mlogloss:0.000498
## [39] train-mlogloss:0.000491
## [40] train-mlogloss:0.000485
## [41] train-mlogloss:0.000478
## [42] train-mlogloss:0.000472
## [43] train-mlogloss:0.000467
## [44] train-mlogloss:0.000462
```

```
## [45] train-mlogloss:0.000457
## [46] train-mlogloss:0.000452
## [47] train-mlogloss:0.000448
## [48] train-mlogloss:0.000444
## [49] train-mlogloss:0.000440
## [50] train-mlogloss:0.000436
## [51] train-mlogloss:0.000432
## [52] train-mlogloss:0.000429
## [53] train-mlogloss:0.000426
## [54] train-mlogloss:0.000423
## [55] train-mlogloss:0.000419
## [56] train-mlogloss:0.000417
## [57] train-mlogloss:0.000414
## [58] train-mlogloss:0.000411
## [59] train-mlogloss:0.000409
## [60] train-mlogloss:0.000406
## [61] train-mlogloss:0.000404
## [62] train-mlogloss:0.000401
## [63] train-mlogloss:0.000399
## [64] train-mlogloss:0.000397
## [65] train-mlogloss:0.000395
## [66] train-mlogloss:0.000393
## [67] train-mlogloss:0.000391
## [68] train-mlogloss:0.000389
## [69] train-mlogloss:0.000388
## [70] train-mlogloss:0.000386
## [71] train-mlogloss:0.000384
## [72] train-mlogloss:0.000383
## [73] train-mlogloss:0.000381
## [74] train-mlogloss:0.000380
## [75] train-mlogloss:0.000378
## [76] train-mlogloss:0.000377
## [77] train-mlogloss:0.000375
## [78] train-mlogloss:0.000374
## [79] train-mlogloss:0.000372
## [80] train-mlogloss:0.000371
## [81] train-mlogloss:0.000370
## [82] train-mlogloss:0.000368
## [83] train-mlogloss:0.000367
## [84] train-mlogloss:0.000366
## [85] train-mlogloss:0.000365
## [86] train-mlogloss:0.000364
## [87] train-mlogloss:0.000363
## [88] train-mlogloss:0.000362
## [89] train-mlogloss:0.000360
## [90] train-mlogloss:0.000359
## [91] train-mlogloss:0.000358
## [92] train-mlogloss:0.000357
## [93] train-mlogloss:0.000356
## [94] train-mlogloss:0.000355
## [95] train-mlogloss:0.000355
## [96] train-mlogloss:0.000354
## [97] train-mlogloss:0.000353
## [98] train-mlogloss:0.000352
```

```
## [99] train-mlogloss:0.000351
## [100]    train-mlogloss:0.000350
```

```
end_time <- proc.time()

summary(xg)
```

```
##                Length Class              Mode
## handle              1 xgb.Booster.handle externalptr
## raw            793552 -none-             raw
## niter               1 -none-             numeric
## evaluation_log      2 data.table         list
## call               15 -none-             call
## params              3 -none-             list
## callbacks           2 -none-             list
## feature_names       5 -none-             character
## nfeatures           1 -none-             numeric
```

```
print(paste("Training time: ", (end_time-start_time)[[3]], "s"))
```

```
## [1] "Training time:  0.780000000000001 s"
```

```
test_matrix <- data.matrix(test)

# get probability of each class for each x
probs <- predict(xg, test_matrix, reshape=TRUE)
pred <- rep(NA, dim(probs)[1])

# take most likely values for predictions
for (i in 1:dim(probs)[1]) {
  pred[i] <- which.max(probs[i,]) - 1
}

acc_xg <- mean(pred==test$Rating)
print(paste("accuracy=", acc_xg))
```

```
## [1] "accuracy= 1"
```

Adaboost

```
library(adabag)
```

```
## Warning: package 'adabag' was built under R version 4.1.3
```

```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 4.1.3
```

```
##
## Attaching package: 'foreach'

## The following objects are masked from 'package:purrr':
##
##     accumulate, when

## Loading required package: doParallel

## Warning: package 'doParallel' was built under R version 4.1.3

## Loading required package: iterators

## Warning: package 'iterators' was built under R version 4.1.3

## Loading required package: parallel
```

```
# limited to 2000 rows because adaboost is slow
set.seed(1234)
i <- sample(1:nrow(train), 2000, replace=FALSE)
train_sample <- train[i,]

train_sample$Rating <- as.factor(train_sample$Rating)

start_time <- proc.time()
adab <- boosting(Rating~., data=train_sample, boos=TRUE, mfinal=10)
end_time <- proc.time()

summary(adab)
```

```
##            Length Class   Mode
## formula         3 formula call
## trees          10 -none-  list
## weights        10 -none-  numeric
## votes       10000 -none-  numeric
## prob        10000 -none-  numeric
## class        2000 -none-  character
## importance      4 -none-  numeric
## terms           3 terms   call
## call            5 -none-  call
```

```
print(paste("Training time: ", (end_time-start_time)[[3]], "s"))
```

```
## [1] "Training time:  29.14 s"
```

```
pred <- predict(adab, newdata=test, type="response")
pred$class <- as.integer(pred$class)

acc_adab <- mean(pred$class==test$Rating)
print(paste("accuracy=", acc_adab))
```

```
## [1] "accuracy= 0.70042194092827"
```