P8106_group2recovery_secondaryanalysis

Yimin Chen (yc4195), Yang Yi (yy3307), Qingyue Zhuo (qz2493)

Contents

Import and data manipulation

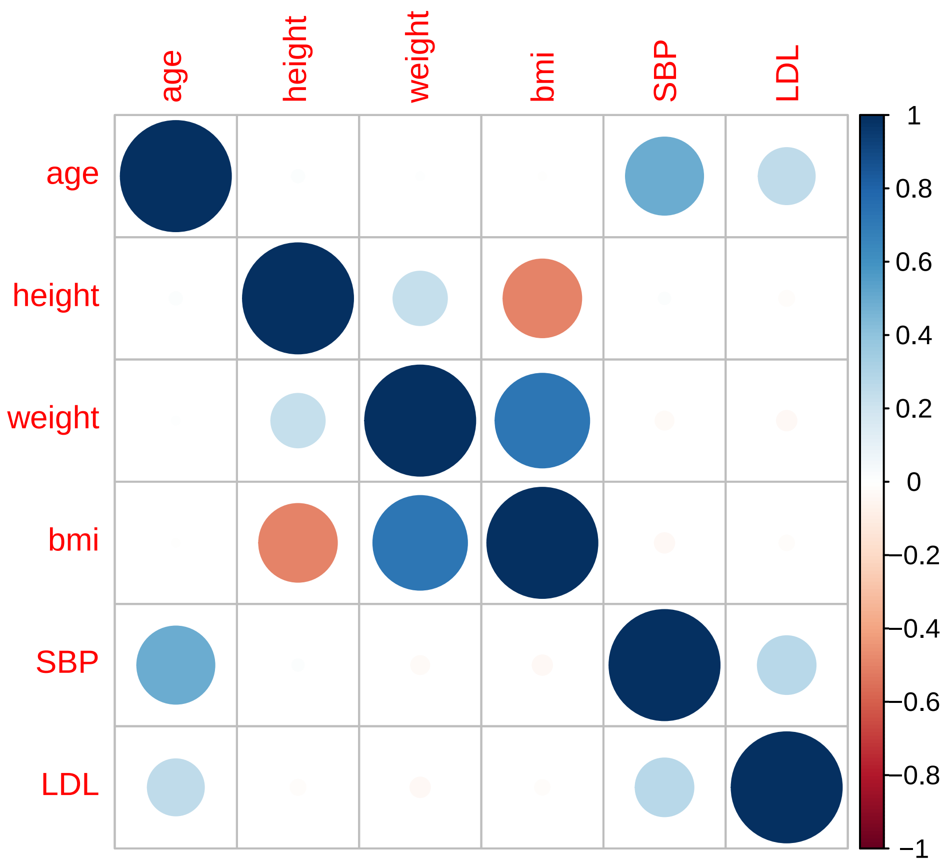
```
# Load recovery.RData environment
load("./recovery.Rdata")

dat %>% na.omit()

# dat1 draw a random sample of 2000 participants Uni:3307
set.seed(3307)

dat1 = dat[sample(1:10000, 2000),]

dat1 =
  dat1[, -1] %>%
  mutate(
    recovery_time = as.factor(
      case_when(recovery_time <= 30 ~ "long", recovery_time > 30 ~ "short")
    ),
    gender = as.factor(gender),
    race = as.factor(race),
    smoking = as.factor(smoking),
```

```r
    hypertension = as.factor(hypertension),
    diabetes = as.factor(diabetes),
    vaccine = as.factor(vaccine),
    severity = as.factor(severity),
    study = as.factor(
      case_when(study == "A" ~ 1, study == "B" ~ 2, study == "C" ~ 3)
      )
    )

# dat2 draw a random sample of 2000 participants Uni:2493
set.seed(2493)

dat2 = dat[sample(1:10000, 2000),]

dat2 =
  dat2[, -1] %>%
  mutate(
    recovery_time = as.factor(
      case_when(recovery_time <= 30 ~ "long", recovery_time > 30 ~ "short")
    ),
    gender = as.factor(gender),
    race = as.factor(race),
    smoking = as.factor(smoking),
    hypertension = as.factor(hypertension),
    diabetes = as.factor(diabetes),
    vaccine = as.factor(vaccine),
    severity = as.factor(severity),
    study = as.factor(
      case_when(study == "A" ~ 1, study == "B" ~ 2, study == "C" ~ 3)
      )
    )

# Merged dataset with unique observation
covid_dat = rbind(dat1, dat2) %>%
  unique()

covid_dat2 = model.matrix(recovery_time ~ ., covid_dat)[, -1]

# Partition dataset into two parts: training data (70%) and test data (30%)
rowTrain = createDataPartition(y = covid_dat$recovery_time, p = 0.7, list = FALSE)

trainData = covid_dat[rowTrain, ]
testData = covid_dat[-rowTrain, ]

ctrl1 = trainControl(method = "repeatedcv", number = 10, repeats = 5)
ctrl2 = trainControl(method = "cv",
                          classProbs = TRUE,
                          summaryFunction = twoClassSummary)
```

Data visualization

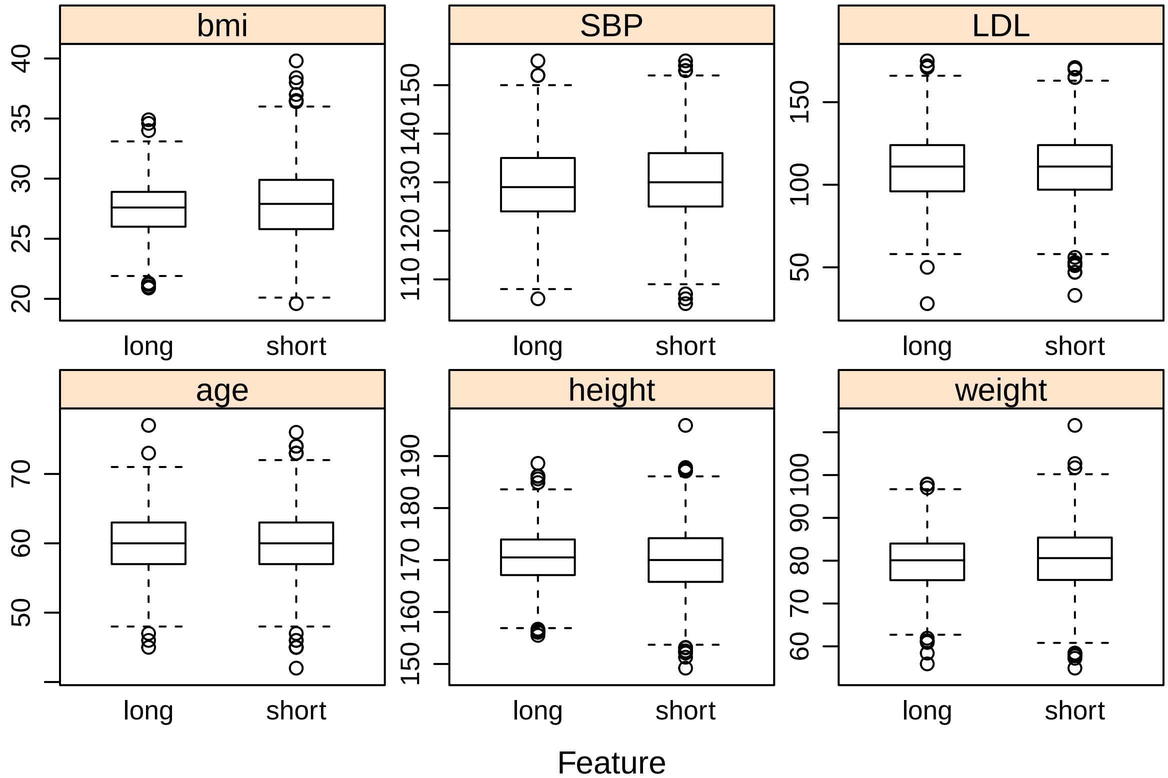
Correlation plot

```
corr_dat = covid_dat[rowTrain,] %>%
  dplyr::select('age', 'height', 'weight', 'bmi', 'SBP', 'LDL')
corrplot(cor(corr_dat), method = "circle", type = "full")
```



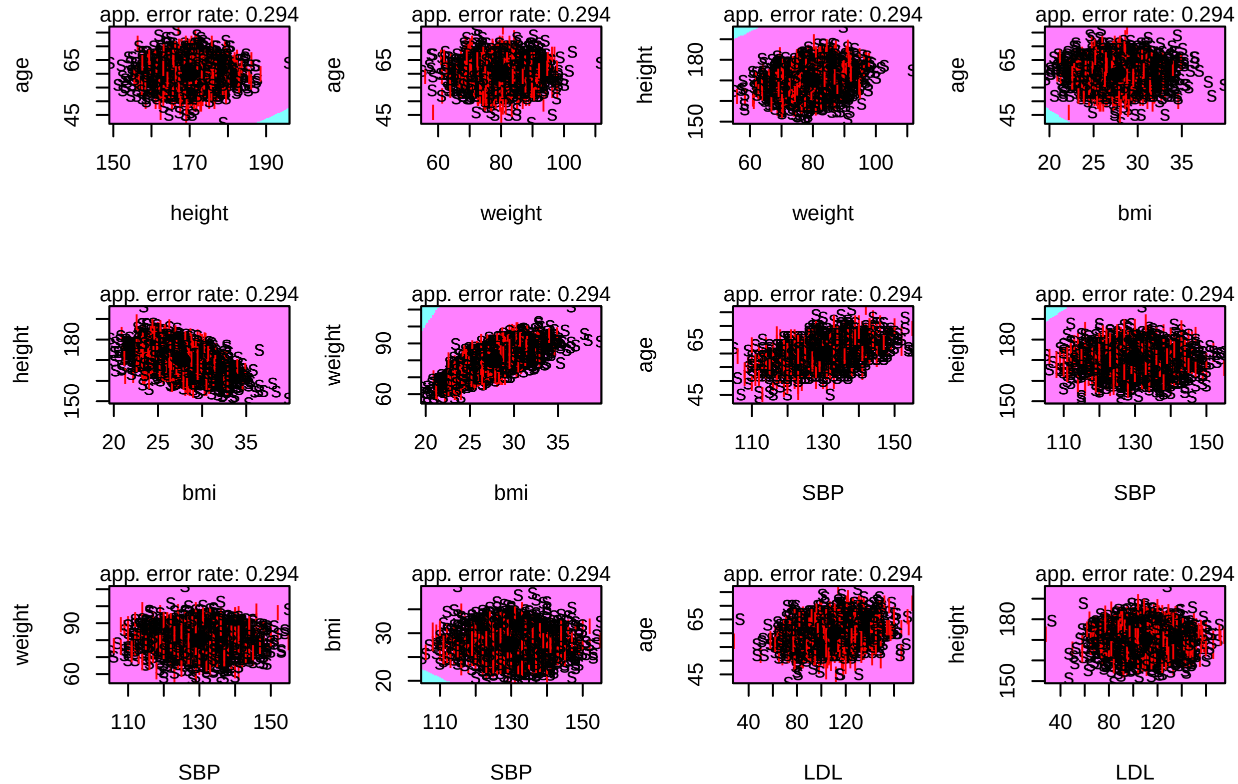
Feature plot

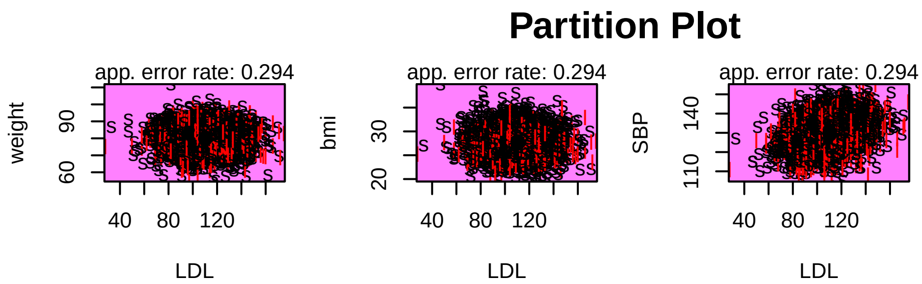
```
vis_trdat = trainData %>%
  dplyr::select('age', 'height', 'weight', 'bmi', 'SBP', 'LDL', 'recovery_time')

theme1 = transparentTheme(trans = .4)
trellis.par.set(theme1)

featurePlot(x = vis_trdat[, 1:6],
            y = vis_trdat[, 7],
            scales = list(x = list(relation = "free"),
                          y = list(relation = "free")),
            plot = "box", pch = "|",
            auto.key = list(columns = 2))
```



Partition plot

```
partimat(recovery_time ~ age + height + weight + bmi + SBP + LDL, data = covid_dat, subset = rowTrain, m
```


app. error rate: 0.294
age
height
app. error rate: 0.294
age
weight
app. error rate: 0.294
height
weight
app. error rate: 0.294
age
bmi
app. error rate: 0.294
height
bmi
app. error rate: 0.294
weight
bmi
app. error rate: 0.294
age
SBP
app. error rate: 0.294
height
SBP
app. error rate: 0.294
weight
SBP
app. error rate: 0.294
bmi
SBP
app. error rate: 0.294
age
LDL
app. error rate: 0.294
height
LDL



Model training

classification - classification tree: L11 - glm + penalized logistice regreesion L8 - GAM L8 - MARS L8 - QDA L9 - LDA L9 - Navie Bayes L9 - random forest L12 - boosting L12 - support vecotr machines L13

LDA

```
set.seed(2)

model.lda <- train(x = covid_dat2[rowTrain,],
                   y = covid_dat$recovery_time[rowTrain],
                   method = "lda",
                   metric = "ROC",
                   trControl = ctrl2)
```

QDA

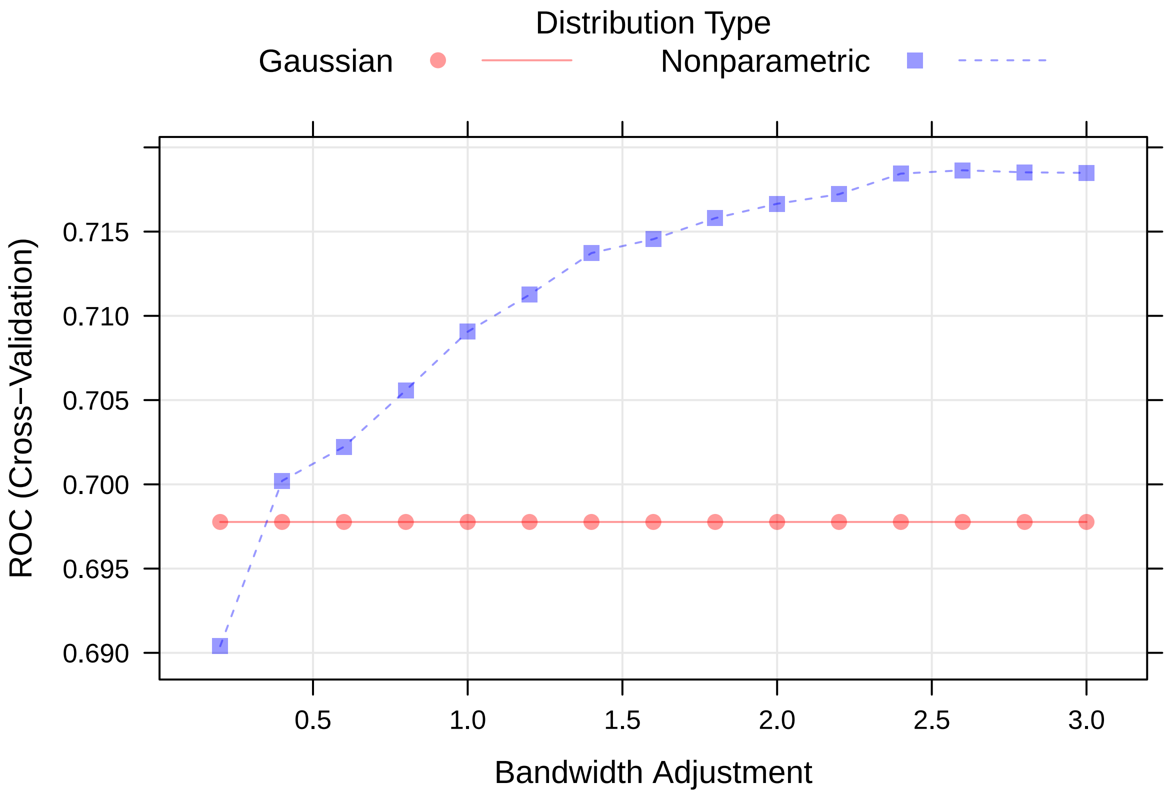
```
set.seed(2)
model.qda <- train(x = covid_dat2[rowTrain,],
                   y = covid_dat$recovery_time[rowTrain],
```

```
                   method = "qda",
                   metric = "ROC",
                   trControl = ctrl2)
```

Naive Bayes (NB)

There is one practical issue with the NB classifier when nonparametric estimators are used. When a new data point includes a feature value that never occurs for some response class, the posterior probability can become zero. To avoid this, we increase the count of the value with a zero occurrence to a small value, so that the overall probability doesn't become zero. In practice, a value of one or two is a common choice. This correction is called "Laplace Correction," and is implemented via the parameter `fL`. The parameter `adjust` adjusts the bandwidths of the kernel density estimates, and a larger value means a more flexible estimate.

```
nbGrid <- expand.grid(usekernel = c(FALSE,TRUE),
                      fL = 1,
                      adjust = seq(.2, 3, by = .2))

set.seed(2)
model.nb <- train(x = covid_dat2[rowTrain,],
                  y = covid_dat$recovery_time[rowTrain],
                  method = "nb",
                  tuneGrid = nbGrid,
                  metric = "ROC",
                  trControl = ctrl2)

plot(model.nb)
```




```r
res <- resamples(list(LDA = model.lda, QDA = model.qda, NB = model.nb))
summary(res)
```

```
##
## Call:
## summary.resamples(object = res)
##
## Models: LDA, QDA, NB
## Number of resamples: 10
##
## ROC
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.6982994 0.7092113 0.7204289 0.7236334 0.7385363 0.7509869    0
## QDA 0.6825470 0.6870515 0.7021897 0.7067165 0.7212097 0.7436988    0
## NB  0.6882730 0.7120171 0.7177622 0.7186405 0.7223240 0.7519739    0
##
## Sens
##          Min.    1st Qu.     Median       Mean    3rd Qu.       Max. NA's
## LDA 0.1891892 0.26013514 0.27702703 0.26525361 0.29489078 0.31081081    0
## QDA 0.5270270 0.55743243 0.60135135 0.59546464 0.63175676 0.67567568    0
## NB  0.0000000 0.01351351 0.01351351 0.01488338 0.02369493 0.02702703    0
##
## Spec
##          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
## LDA 0.8587571 0.8997175 0.9239351 0.9170507 0.9324890 0.9606742    0
```

```
## QDA 0.6440678 0.7090395 0.7211325 0.7185298 0.7299562 0.7683616    0
## NB  0.9887006 0.9957627 1.0000000 0.9977401 1.0000000 1.0000000    0
```

test set performance.

```
lda.pred <- predict(model.lda, newdata = covid_dat2[-rowTrain,], type = "prob")[,2]
nb.pred <- predict(model.nb, newdata = covid_dat2[-rowTrain,], type = "prob")[,2]
qda.pred <- predict(model.qda, newdata = covid_dat2[-rowTrain,], type = "prob")[,2]


roc.lda <- roc(covid_dat$recovery_time[-rowTrain], lda.pred)
roc.nb <- roc(covid_dat$recovery_time[-rowTrain], nb.pred)
roc.qda <- roc(covid_dat$recovery_time[-rowTrain], qda.pred)


auc <- c(roc.lda$auc[1], roc.qda$auc[1], roc.nb$auc[1])

plot(roc.lda, legacy.axes = TRUE)
plot(roc.qda, col = 2, add = TRUE)
plot(roc.nb, col = 3, add = TRUE)

modelNames <- c("lda","qda","nb")
legend("bottomright", legend = paste0(modelNames, ": ", round(auc,3)),
       col = 1:3, lwd = 2)
```