Lecture 17 Notes

2024-03-26

Introducing Random Forests with ranger

```
require(tidyverse)
```

```
## Loading required package: tidyverse
```

```
## Warning: package 'tidyverse' was built under R version 4.3.2
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✓ dplyr     1.1.2     ✓ readr     2.1.4
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.4     ✓ tibble    3.2.1
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ✓ purrr     1.0.1
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts t
o become errors
```

```
fn <- read_rds("https://github.com/jbisbee1/DS1000_S2024/raw/main/data/fn_cleaned_final.
rds")

form.perf <- 'won ~ hits + assists + accuracy + head_shots + damage_to_players'

form.games <- 'won ~ eliminations + revives + distance_traveled + materials_gathered'

form.context <- 'won ~ mental_state + startTime + gameIdSession'

form.full <- 'won ~ hits + assists + accuracy + head_shots + damage_to_players + elimina
tions + revives + distance_traveled + materials_gathered + mental_state + startTime + ga
meIdSession'
```

Start simple: lm()

```
m_perf <- lm(as.formula(form.perf),data = fn)

summary(m_perf)
```

```
##
## Call:
## lm(formula = as.formula(form.perf), data = fn)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7905 -0.2756 -0.1563  0.3429  1.0078
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        8.788e-02  3.768e-02   2.332 0.019893 *
## hits               6.962e-04  1.001e-03   0.695 0.487053
## assists            3.445e-02  1.020e-02   3.377 0.000764 ***
## accuracy          -4.164e-01  1.081e-01  -3.850 0.000126 ***
## head_shots        -4.808e-03  3.149e-03  -1.527 0.127057
## damage_to_players  4.728e-04  5.713e-05   8.275 4.31e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4191 on 951 degrees of freedom
## Multiple R-squared:  0.1752, Adjusted R-squared:  0.1708
## F-statistic:  40.4 on 5 and 951 DF,  p-value: < 2.2e-16
```

```
require(tidymodels)
```

```
## Loading required package: tidymodels
```

```
## ── Attaching packages ─────────────────────────────────────── tidymodels 1.1.1 ──
```

```
## ✓ broom        1.0.5     ✓ rsample      1.2.0
## ✓ dials        1.2.0     ✓ tune         1.1.2
## ✓ infer        1.0.5     ✓ workflows    1.1.3
## ✓ modeldata    1.2.0     ✓ workflowsets 1.0.1
## ✓ parsnip      1.1.1     ✓ yardstick    1.2.0
## ✓ recipes      1.0.8
```

```
## Warning: package 'scales' was built under R version 4.3.3
```

```
## ── Conflicts ────────────────────────────────────── tidymodels_conflicts() ──
## ✖ scales::discard() masks purrr::discard()
## ✖ dplyr::filter()   masks stats::filter()
## ✖ recipes::fixed()  masks stringr::fixed()
## ✖ dplyr::lag()      masks stats::lag()
## ✖ yardstick::spec() masks readr::spec()
## ✖ recipes::step()   masks stats::step()
## • Use suppressPackageStartupMessages() to eliminate package startup messages
```

```
toEval <- fn %>%
  mutate(prob_win = predict(m_perf),
         won = factor(won,levels =c('1','0')))

roc_auc(toEval,won,prob_win)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.744
```

```
m_games <- lm(as.formula(form.games),data = fn)
m_context <- lm(as.formula(form.context),data = fn)
m_full <- lm(as.formula(form.full),data = fn)

toEval <- fn %>%
  mutate(prob_win_perf = predict(m_perf),
         prob_win_games = predict(m_games),
         prob_win_context = predict(m_context),
         prob_win_full = predict(m_full),
         won = factor(won,levels =c('1','0')))

auc_perf <- roc_auc(toEval,won,prob_win_perf)
auc_games <- roc_auc(toEval,won,prob_win_games)
auc_context <- roc_auc(toEval,won,prob_win_context)
auc_full <- roc_auc(toEval,won,prob_win_full)

auc_perf
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.744
```

```
auc_games
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.800
```

```
auc_context
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.611
```

```
auc_full
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.826
```

```
# Using a logit instead
m_perf <- glm(as.formula(form.perf),data = fn,family = binomial(link = 'logit'))
m_games <- glm(as.formula(form.games),data = fn,family = binomial(link = 'logit'))
m_context <- glm(as.formula(form.context),data = fn,family = binomial(link = 'logit'))
m_full <- glm(as.formula(form.full),data = fn,family = binomial(link = 'logit'))

toEval <- fn %>%
  mutate(prob_win_perf = predict(m_perf,type = 'response'),
         prob_win_games = predict(m_games,type = 'response'),
         prob_win_context = predict(m_context,type = 'response'),
         prob_win_full = predict(m_full,type = 'response'),
         won = factor(won,levels =c('1','0')))

auc_perf_logit <- roc_auc(toEval,won,prob_win_perf)
auc_games_logit <- roc_auc(toEval,won,prob_win_games)
auc_context_logit <- roc_auc(toEval,won,prob_win_context)
auc_full_logit <- roc_auc(toEval,won,prob_win_full)

auc_perf
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.744
```

```
auc_perf_logit
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.746
```

```
auc_games
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.800
```

```
auc_games_logit
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.800
```

```
auc_context
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.611
```

```
auc_context_logit
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.611
```

```
auc_full
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.826
```

```
auc_full_logit
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.825
```

Ranger

```
require(ranger)
```

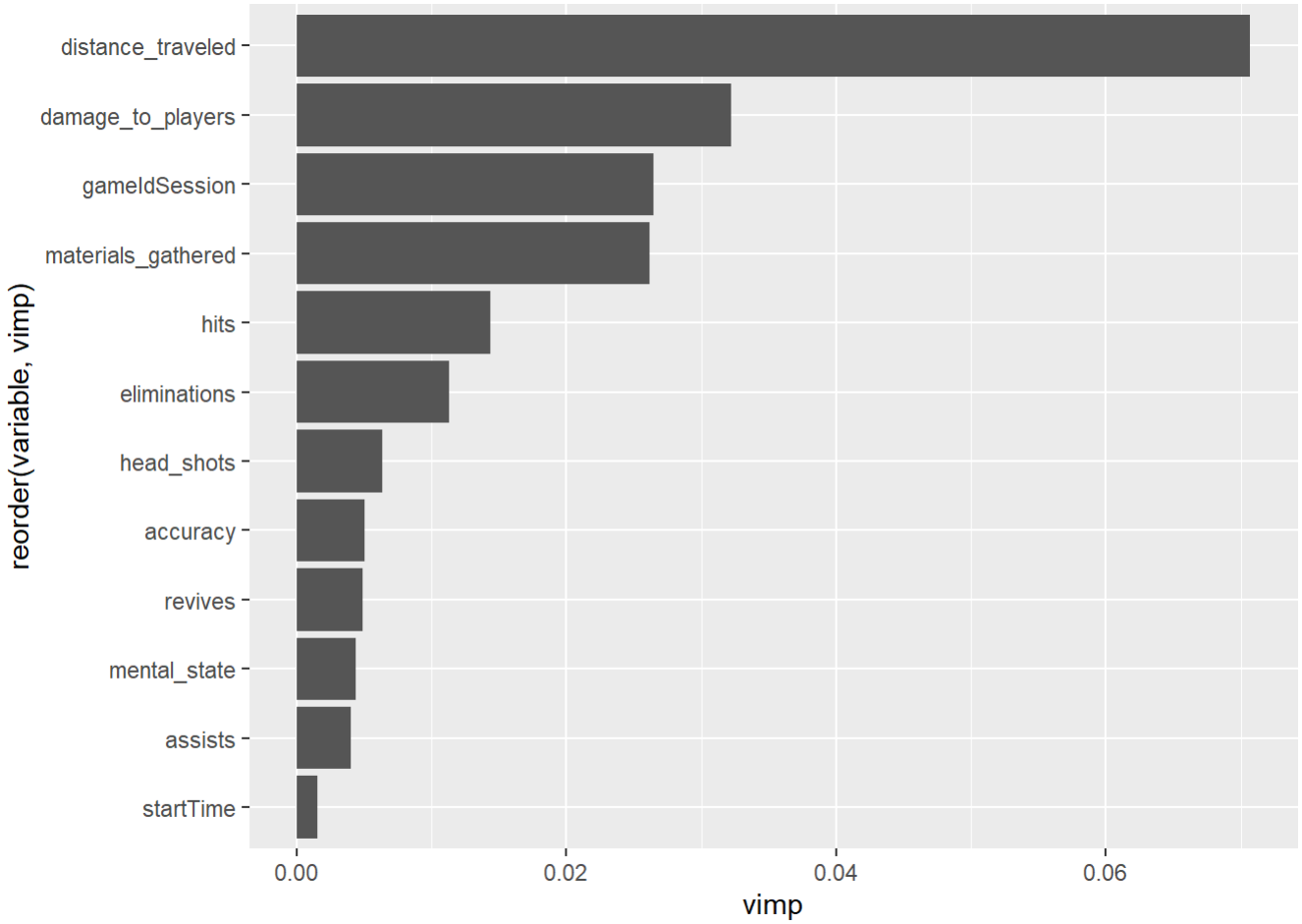
```
## Loading required package: ranger
```

```
## Warning: package 'ranger' was built under R version 4.3.3
```

```
m_ranger_full <- ranger(formula = as.formula(form.full),
                        data = fn,num.trees = 2000,
                        importance = 'permutation')


toEval <- fn %>%
  mutate(prob_win = m_ranger_full$predictions,
         won = factor(won,levels = c('1','0')))

auc_ranger <- roc_auc(toEval,won,prob_win)
```

Look at variable importance

```
toplot <- data.frame(vimp = m_ranger_full$variable.importance,
                     variable = names(m_ranger_full$variable.importance))

toplot %>%
  ggplot(aes(x = vimp,y = reorder(variable,vimp))) +
  geom_bar(stat = 'identity')
```

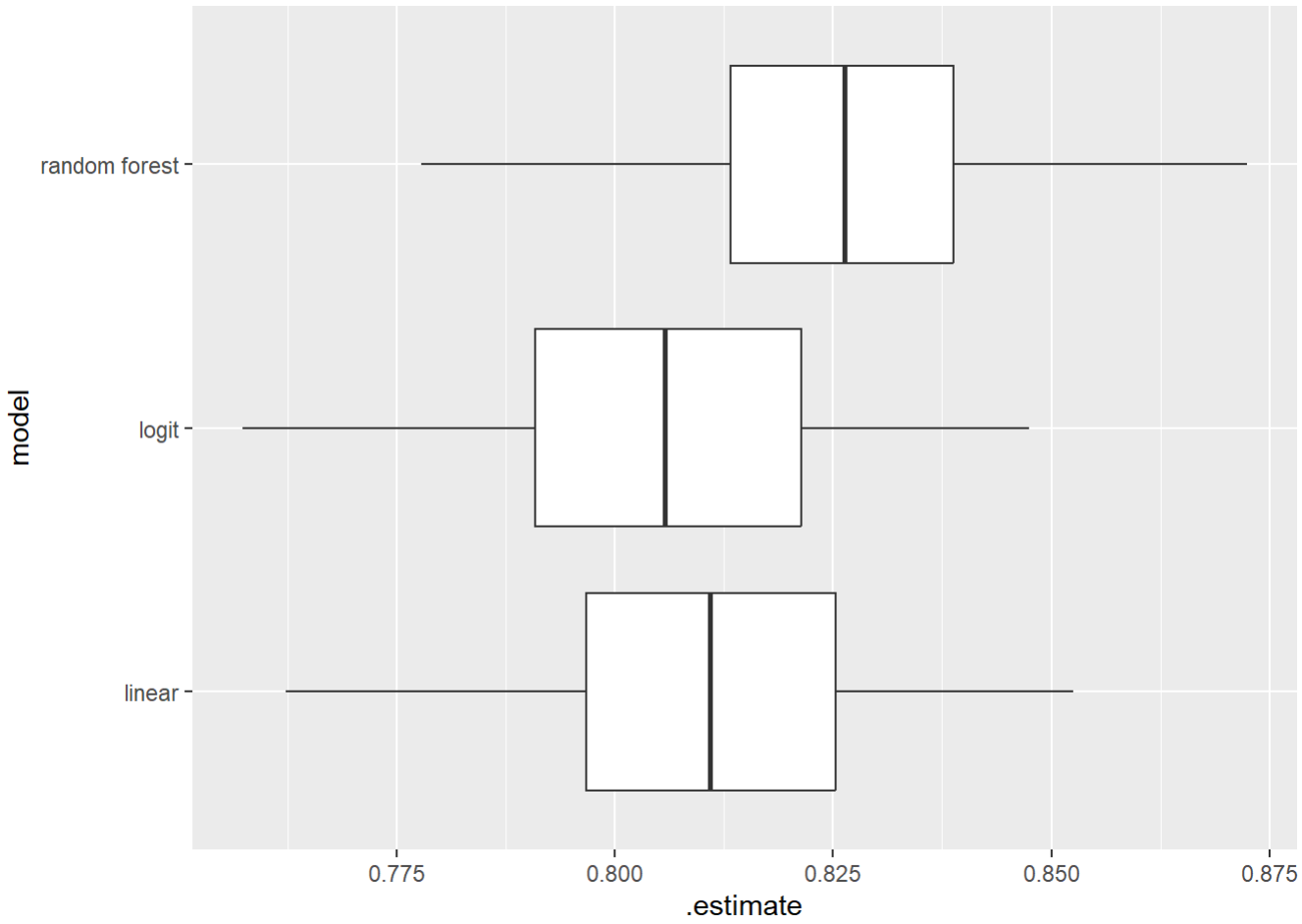


Cross validation with a random forest

```
set.seed(123)
cvRes <- NULL
for(i in 1:100) {
  inds <- sample(1:nrow(fn),size = round(nrow(fn)*.6),replace = F)
  train <- fn %>% slice(inds)
  test <- fn %>% slice(-inds)

  m_lm <- lm(as.formula(form.full),train)
  m_glm <- glm(as.formula(form.full),train,family = binomial(link = 'logit'))
  m_rf <- ranger(as.formula(form.full),train,importance = 'none')

  tmp_rf_preds <- predict(m_rf,data = test)

  toEval <- test %>%
    mutate(prob_win_lm = predict(m_lm,newdata = test),
           prob_win_glm = predict(m_glm,newdata = test,type = 'response'),
           prob_win_rf = tmp_rf_preds$predictions,
           won = factor(won,levels = c('1','0')))

  auc_lm <- roc_auc(toEval,won,prob_win_lm) %>% mutate(model = 'linear')
  auc_glm <- roc_auc(toEval,won,prob_win_glm) %>% mutate(model = 'logit')
  auc_rf <- roc_auc(toEval,won,prob_win_rf) %>% mutate(model = 'random forest')

  tmp <- auc_lm %>%
    bind_rows(auc_glm) %>%
    bind_rows(auc_rf)

  cvRes <- cvRes %>%
    bind_rows(tmp)
}

cvRes %>%
  group_by(model) %>%
  summarise(avg_auc = mean(.estimate))
```

```
## # A tibble: 3 × 2
##   model         avg_auc
##   <chr>           <dbl>
## 1 linear          0.811
## 2 logit           0.806
## 3 random forest   0.826
```

```
cvRes %>%
  ggplot(aes(x = .estimate,y = model)) +
  geom_boxplot()
```


random forest
logit
linear
model
0.775
0.800
0.825
0.850
0.875
.estimate