Lecture 16 Notes

2024-03-21

Loading the data

```
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')
```

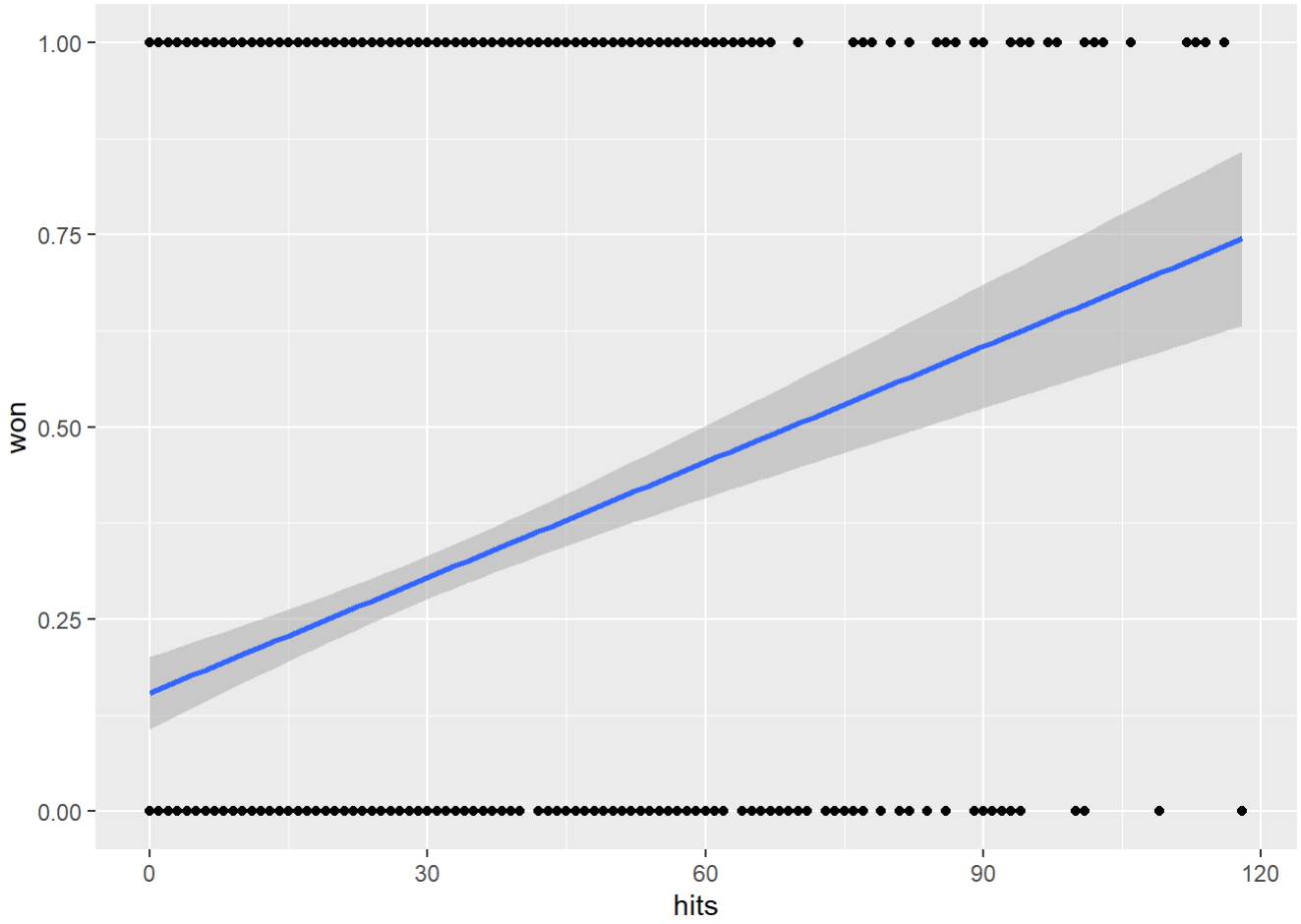
Regression

```
m <- lm(won ~ mental_state + hits,fn)
summary(m)
```

```
##
## Call:
## lm(formula = won ~ mental_state + hits, data = fn)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8074 -0.3192 -0.1952  0.5214  0.9144
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.0855560  0.0277550   3.083  0.00211 **
## mental_statesober 0.1339652  0.0285516   4.692  3.1e-06 ***
## hits              0.0049820  0.0006277   7.937  5.8e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4413 on 954 degrees of freedom
## Multiple R-squared:  0.08239,    Adjusted R-squared:  0.08047
## F-statistic: 42.83 on 2 and 954 DF,  p-value: < 2.2e-16
```

```
fn %>%
  ggplot(aes(x = hits,y = won)) +
  geom_point() +
  geom_smooth(method = 'lm')
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

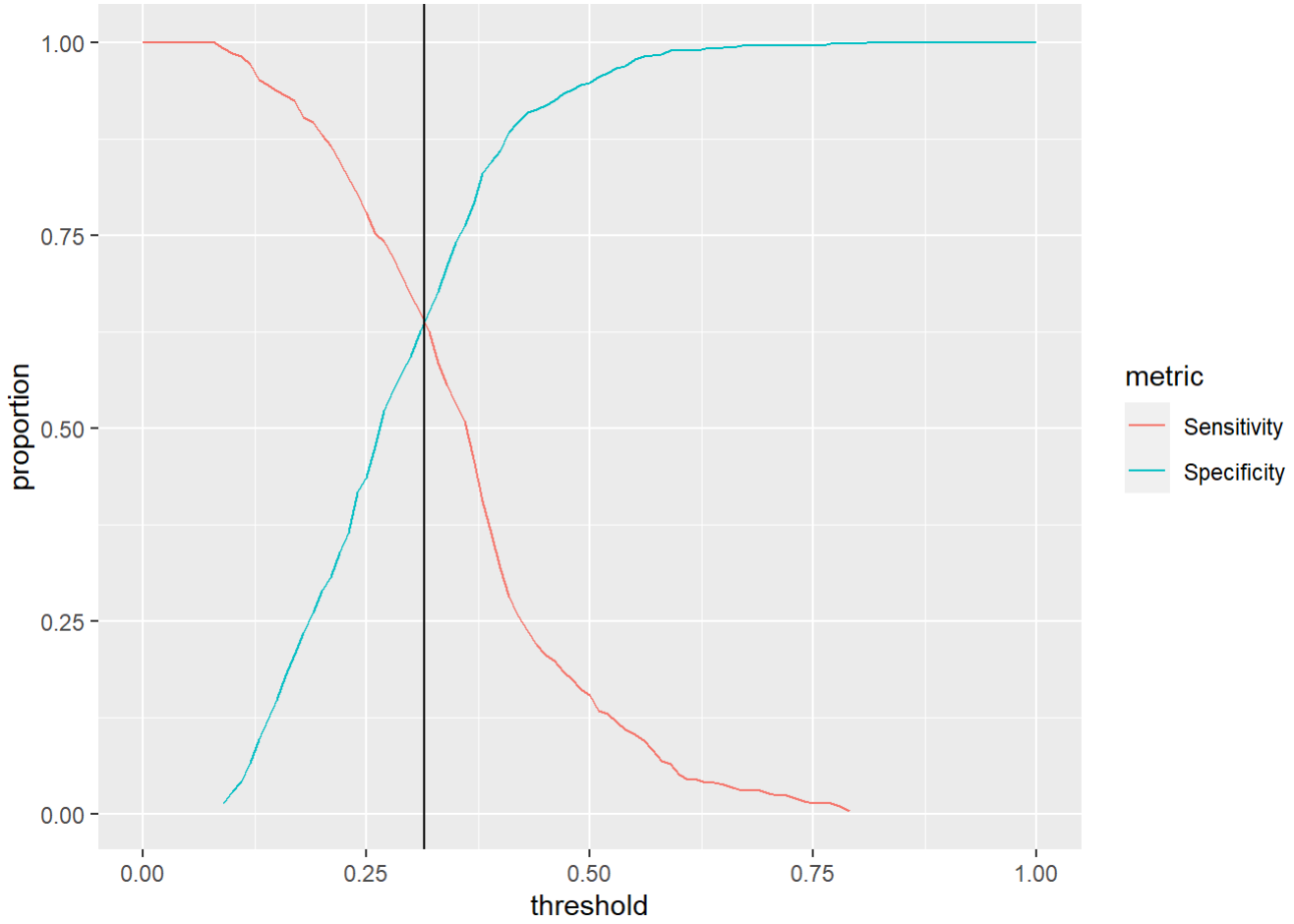



```
# Create probability predictions
fn %>%
  mutate(prob_win = predict(m)) %>%
  select(won,prob_win) %>%
  mutate(pred_win = ifelse(prob_win > .3,1,0)) %>%
  group_by(won,pred_win) %>%
  summarise(nGames = n()) %>%
  group_by(won) %>%
  mutate(tot_games = sum(nGames)) %>%
  ungroup() %>%
  mutate(proportion = nGames / tot_games)
```

```
## `summarise()` has grouped output by 'won'. You can override using the `.groups`
## argument.
```

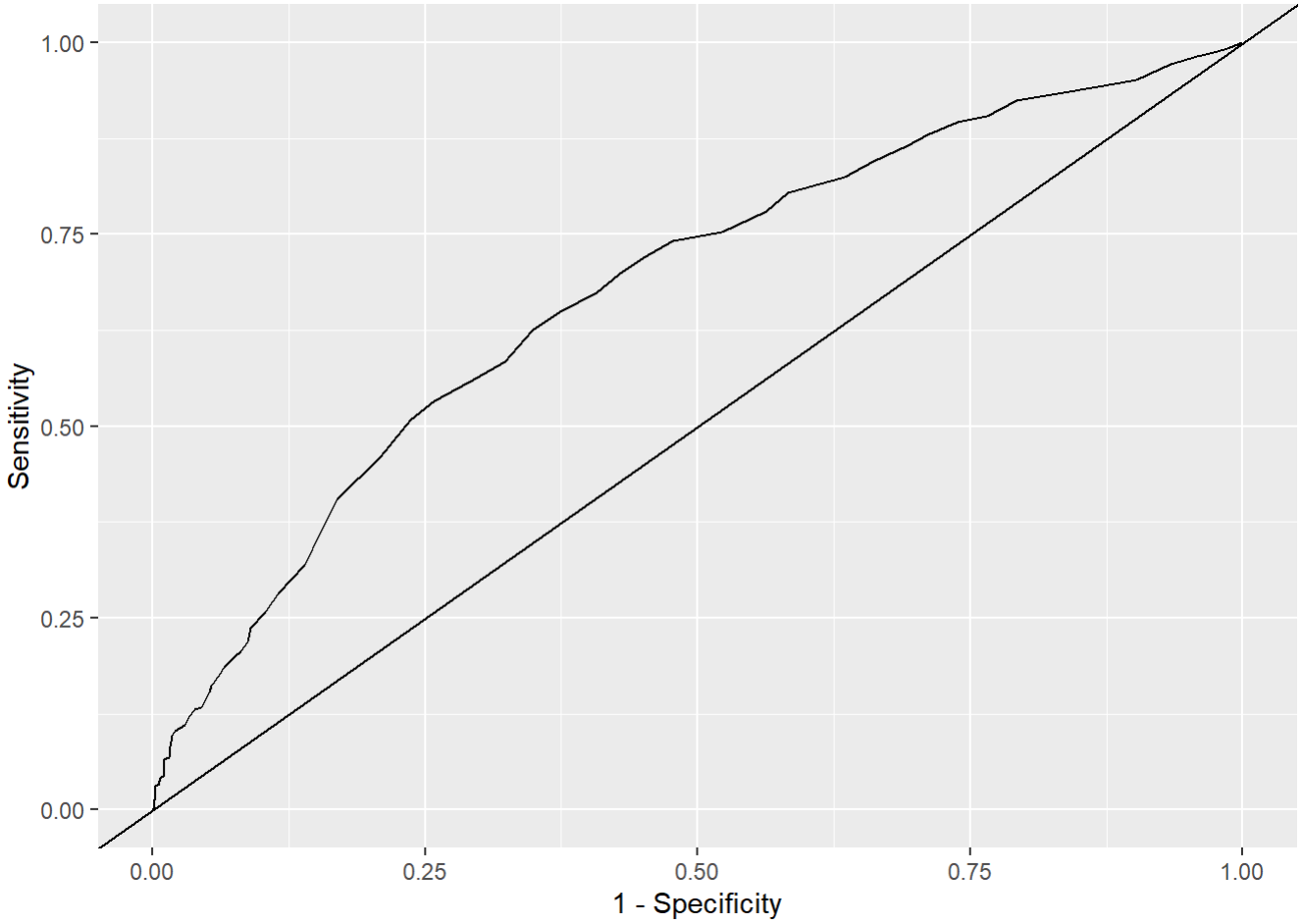
```
## # A tibble: 4 × 5
##     won pred_win nGames tot_games proportion
##   <dbl>    <dbl>  <int>     <int>      <dbl>
## 1     0        0    395       666      0.593
## 2     0        1    271       666      0.407
## 3     1        0     95       291      0.326
## 4     1        1    196       291      0.674
```

```
# Make it efficient with a loop()
threshRes <- NULL
for(i in seq(0,1,by = .01)) {
  tmp <- fn %>%
  mutate(prob_win = predict(m)) %>%
  select(won,prob_win) %>%
  mutate(pred_win = ifelse(prob_win > i,1,0)) %>%
  group_by(won,pred_win) %>%
  summarise(nGames = n(),.groups = 'drop') %>%
  group_by(won) %>%
  mutate(tot_games = sum(nGames)) %>%
  ungroup() %>%
  mutate(proportion = nGames / tot_games)

  threshRes <- threshRes %>%
    bind_rows(tmp %>%
                mutate(threshold = i))
}

threshRes %>%
  mutate(metric = ifelse(won == 1 & pred_win == 1,'Sensitivity',
                         ifelse(won == 0 & pred_win == 0,'Specificity',NA))) %>%
  drop_na(metric) %>%
  ggplot(aes(x = threshold,y = proportion,color = metric)) +
  geom_line() +
  geom_vline(xintercept = .315)
```



Receiver Operator Curve (ROC)

```
threshRes %>%
  mutate(metric = ifelse(won == 1 & pred_win == 1,'Sensitivity',
                         ifelse(won == 0 & pred_win == 0,'Specificity',NA))) %>%
  drop_na(metric) %>%
  select(proportion,metric,threshold) %>%
  spread(metric,proportion,fill = 0) %>%
  arrange(desc(Specificity),Sensitivity) %>%
  ggplot(aes(x = 1 - Specificity,
             y = Sensitivity)) +
  geom_line() +
  geom_abline(intercept = 0,slope = 1)
```



Calculate AUC (Area Under the Curve)

```
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()
## • Dig deeper into tidy modeling with R at https://www.tmwr.org
```

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

roc_auc(toEval,truth,prob_win)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.677
```

More X variables

```
m2 <- lm(won ~ mental_state + hits + damage_taken + distance_traveled + head_shots + rev
ives,fn)

toEval2 <- fn %>%
  mutate(prob_win = predict(m2)) %>%
  mutate(truth = factor(won,levels = c('1','0')))

roc_auc(toEval2,truth,prob_win)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.817
```

Logit function

```
m3 <- glm(won ~ mental_state + hits + damage_taken + distance_traveled + head_shots + re
vives,
          fn,
          family = binomial(link = 'logit'))

toEval3 <- fn %>%
  mutate(prob_win = predict(m3,type = 'response')) %>%
  mutate(truth = factor(won,levels = c('1','0')))

roc_auc(toEval3,truth,prob_win)
```

```
## # A tibble: 1 × 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 roc_auc binary         0.817
```