Practice 2: Regression

Nimalan Subramanian

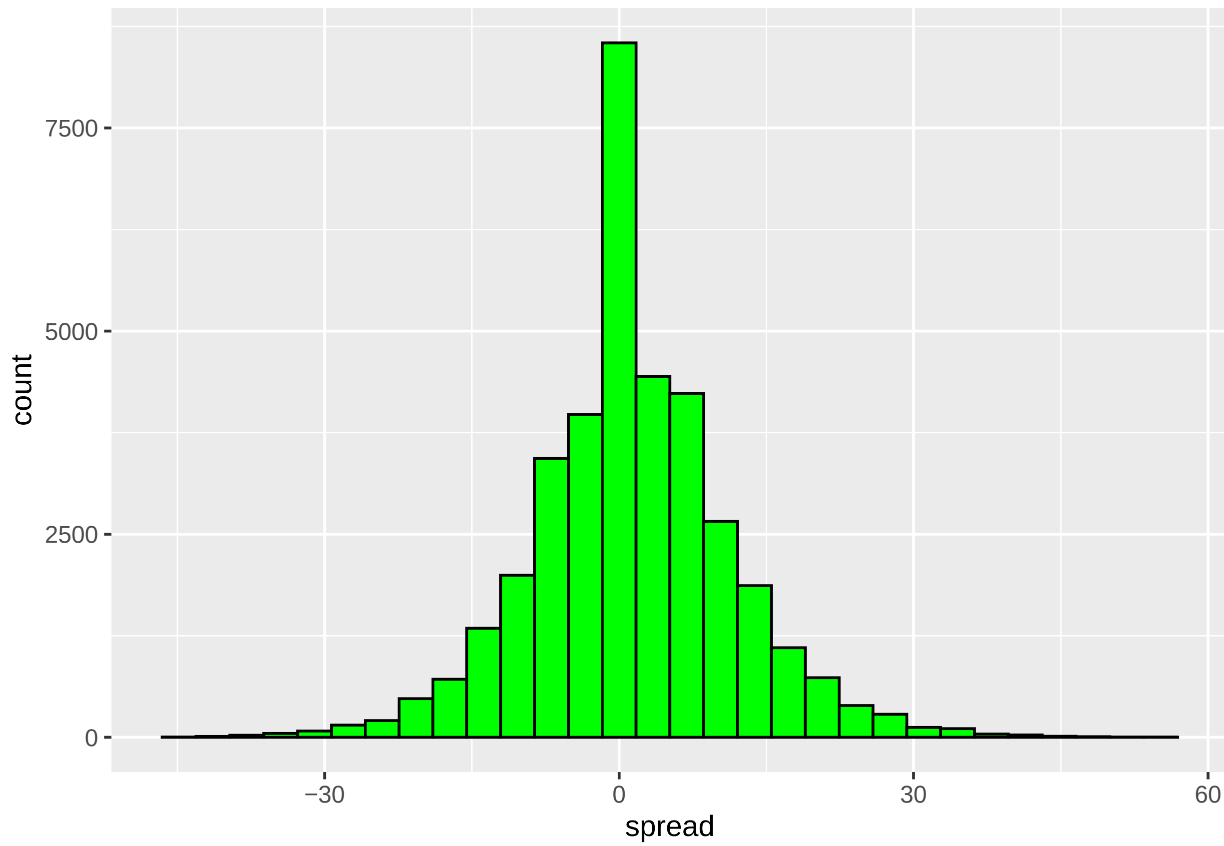
March 19, 2024

Linear Regression

```
# Import data
kick_data <- read.csv("kicking.csv")

# Add spread variable
kick_data <- kick_data %>%
  mutate(spread = home_score_pre - visiting_score_pre)

# Create histogram for spread
ggplot(kick_data, aes(x = spread)) +
  geom_histogram(bins = 30, fill = "green", color = "black")
```



```r
library(broom)
```

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

```r
broom::tidy()
```

```
## # A tibble: 0 x 0
```

```r
broom::glance()
```

```
## # A tibble: 0 x 0
```

```r
# Create linear regression model for spread knowing only home, away, quarter, and spread
spread_linear <- lm(spread ~ factor(home_team) + factor(away_team) + quarter + yardline, data = kick_da

# Create table for coefficients and p-values
coeff_pvals <- tidy(spread_linear)

# Get R-squared and adjusted R-squared values
linear_glance <- glance(spread_linear)
rsquare_adjrsquare <- linear_glance %>%
  select(r.squared, adj.r.squared)

coeff_pvals
```

```
## # A tibble: 65 x 5
##    term                                  estimate std.error statistic  p.value
##    <chr>                                    <dbl>     <dbl>     <dbl>    <dbl>
##  1 (Intercept)                             -0.147     0.446    -0.329 7.42e- 1
##  2 factor(home_team)Atlanta Falcons         3.32      0.419     7.91  2.57e-15
##  3 factor(home_team)Baltimore Ravens        4.05      0.428     9.48  2.78e-21
##  4 factor(home_team)Buffalo Bills           0.822     0.428     1.92  5.51e- 2
##  5 factor(home_team)Carolina Panthers       1.20      0.426     2.82  4.80e- 3
##  6 factor(home_team)Chicago Bears           1.08      0.425     2.55  1.09e- 2
##  7 factor(home_team)Cincinnati Bengals      1.84      0.424     4.34  1.46e- 5
##  8 factor(home_team)Cleveland Browns       -0.204     0.433    -0.471 6.38e- 1
##  9 factor(home_team)Dallas Cowboys          1.36      0.417     3.26  1.13e- 3
## 10 factor(home_team)Denver Broncos          1.47      0.423     3.47  5.22e- 4
## # i 55 more rows
```

```r
rsquare_adjrsquare
```

```
## # A tibble: 1 x 2
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1    0.0494        0.0477
```

```r
# Add interaction between home and away teams to previous model
spreadlin2 <- lm(spread ~ factor(home_team) * factor(away_team) + quarter + yardline, data = kick_data)

# Get R-squared and adjusted R-squared of new model
linglance2 <- glance(spreadlin2)
rsq_adjrsq2 <- linglance2 %>%
  select(r.squared, adj.r.squared)

rsq_adjrsq2
```

```
## # A tibble: 1 x 2
```

```
##   r.squared adj.r.squared
##       <dbl>         <dbl>
## 1     0.183         0.160
```

```r
# Split data into 80-20 training testing split
set.seed(123)
create_train_test <- function(data, size = 0.8, train = TRUE) {
    n_row = nrow(data)
    total_row = size * n_row
    train_sample <- 1: total_row
    if (train == TRUE) {
        return (data[train_sample, ])
    } else {
        return (data[-train_sample, ])
    }
}

kick_train <- create_train_test(kick_data, 0.8, train = TRUE)
kick_test <- create_train_test(kick_data, 0.8, train = FALSE)

# Fit models on training data
train_mod1 <- lm(spread ~ factor(home_team) + factor(away_team) + quarter + yardline, data = kick_train)
train_mod2 <- lm(spread ~ factor(home_team) * factor(away_team) + quarter + yardline, data = kick_train)

# Predict spread with testing data
test_pred1 <- predict(train_mod1, newdata = kick_test)
test_pred2 <- predict(train_mod2, newdata = kick_test)
```

```
## Warning in predict.lm(train_mod2, newdata = kick_test): prediction from
## rank-deficient fit; attr(*, "non-estim") has doubtful cases
```

```r
# Calculate RMSE and MAD
rmse1 <- sqrt(mean((test_pred1 - kick_test$spread)^2))
mad1 <- mean(abs(test_pred1 - kick_test$spread))

rmse2 <- sqrt(mean((test_pred2 - kick_test$spread)^2))
mad2 <- mean(abs(test_pred2 - kick_test$spread))

# Make table to house RMSE and MAD values
results <- data.frame(
  Model = c("Without Interaction", "With Interaction"),
  RMSE = c(rmse1, rmse2),
  MAD = c(mad1, mad2)
)

results
```

```
##                 Model     RMSE      MAD
## 1 Without Interaction 10.20359 7.754810
## 2    With Interaction 11.08710 8.528837
```

Logistical Regression

```r
# Subset kick data for only field goals and only keep kickers with more than 100 field goals
field_goals <- kick_data %>%
```

```r
  filter(play_type == "Field Goal")

kickers_fg <- field_goals %>%
  group_by(kicker_name) %>%
  filter(n() > 100) %>%
  ungroup()

# Fit logistic regression model to predict probability of success
log_mod <- glm(scored ~ yardline + quarter + factor(kicker_name) + spread, data = kickers_fg, family =
log_tidy <- tidy(log_mod)

log_tidy
```

```
## # A tibble: 63 x 5
##    term                           estimate std.error statistic   p.value
##    <chr>                             <dbl>     <dbl>     <dbl>     <dbl>
##  1 (Intercept)                      4.49      0.185      24.3   7.56e-131
##  2 yardline                        -0.114     0.00320   -35.6   6.57e-278
##  3 quarter                          0.0302    0.0241      1.25  2.10e-  1
##  4 factor(kicker_name)B.Cundiff    -0.854     0.248      -3.45  5.60e-  4
##  5 factor(kicker_name)B.McManus    -0.279     0.269      -1.04  2.99e-  1
##  6 factor(kicker_name)B.Walsh      -0.0612    0.270      -0.227 8.21e-  1
##  7 factor(kicker_name)C.Barth      -0.144     0.261      -0.551 5.82e-  1
##  8 factor(kicker_name)C.Boswell     0.103     0.317       0.326 7.44e-  1
##  9 factor(kicker_name)C.Catanzaro  -0.268     0.294      -0.911 3.62e-  1
## 10 factor(kicker_name)C.Parkey     -0.404     0.306      -1.32  1.87e-  1
## # i 53 more rows
```

```r
# Create confusion matrix for predicted and actual field goals
fg_pred <- predict(log_mod, type = "response")
pred_class <- ifelse(fg_pred > 0.5, 1, 0)
actual_fg <- kickers_fg$scored
conf_mat <- table(Predicted = pred_class, Actual = actual_fg)

conf_mat
```

```
##          Actual
## Predicted     0     1
##         0   592   100
##         1  1816 11544
```

```r
# Add empty column for predictions
kickers_fg$predicted_scored <- NA

# Assign each obs to a fold
set.seed(123)
kickers_fg$fold <- sample(1:10, nrow(kickers_fg), replace = TRUE)

# Perform 10-fold CV
for (fold in 1:10) {
  fg_train <- kickers_fg[kickers_fg$fold != fold, ]
  fg_test <- kickers_fg[kickers_fg$fold == fold, ]

  fg_mod <- glm(scored ~ yardline + quarter + factor(kicker_name) + spread, data = fg_train, family = "
```

```r
  fg_pred_prob <- predict(fg_mod, newdata = fg_test, type = "response")
  fg_pred_class <- ifelse(fg_pred_prob > 0.5, 1, 0)

  kickers_fg$predicted_scored[kickers_fg$fold == fold] <- fg_pred_class
}

# Check to make sure there are no NA values in prediction column
# sum(is.na(kickers_fg$predicted_scored))

# Store actual and predicted fg results in variables
fg_actual <- kickers_fg$scored
fg_predicted <- kickers_fg$predicted_scored

# Create confusion matrix
fg_conf_mat <- table(Actual = fg_actual, Predicted = fg_predicted)

fg_conf_mat
```

```
##       Predicted
## Actual     0     1
##      0   592  1816
##      1   104 11540
```