Win Probability

Connor Train

2024-07-22

```
library(nflreadr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(caTools)
library(ROCR)
```

```
pbp_data <- load_pbp(2023)
```

```
# Add the winner column

pbp_data <- pbp_data %>%
  mutate(winner = ifelse(total_home_score > total_away_score, home_team, away_team))

# Create outcome variable

pbp_data <- pbp_data %>%
  mutate(poswins = as.factor(ifelse(winner == posteam, "Yes", "No")))
```

```
# Step 3: Filter and select specified variables
filtered_pbp <- pbp_data %>%
  filter(
    qtr <= 4 &
    !is.na(poswins) &
    play_type != "no_play" &
    !is.na(play_type) &
    !is.na(down)
  ) %>%
```

```
  select(
    game_id, game_date, posteam, home_team, away_team, yardline_100, qtr,
    game_seconds_remaining, poswins, down, ydstogo, score_differential,
    home_wp, away_wp, wp, desc
  )
```

```
# Step 4: Split Data into Training and Testing Sets (if desired)
set.seed(123)  # For reproducibility
split <- sample.split(filtered_pbp$poswins, SplitRatio = 0.7)
training_set <- subset(filtered_pbp, split == TRUE)
testing_set <- subset(filtered_pbp, split == FALSE)
```

```
# Step 5: Build the Logistic Regression Model
model <- glm(
  poswins ~ qtr + down + ydstogo + game_seconds_remaining + yardline_100 + score_differential,
  data = training_set,
  family = binomial
)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
# Step 6: Evaluate the Model
summary(model)
```

```
##
## Call:
## glm(formula = poswins ~ qtr + down + ydstogo + game_seconds_remaining +
##     yardline_100 + score_differential, family = binomial, data = training_set)
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              4.753e-01  4.310e-01   1.103    0.270
## qtr                      8.729e-02  9.981e-02   0.875    0.382
## down                     1.179e-01  2.654e-02   4.442 8.93e-06 ***
## ydstogo                  1.112e-03  6.787e-03   0.164    0.870
## game_seconds_remaining -1.869e-05  9.970e-05  -0.188    0.851
## yardline_100           -1.157e-02  1.089e-03 -10.631  < 2e-16 ***
## score_differential      1.314e+00  3.000e-02  43.807  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 37865.5  on 27474  degrees of freedom
## Residual deviance:  9107.5  on 27468  degrees of freedom
## AIC: 9121.5
##
## Number of Fisher Scoring iterations: 10
```

```
# Predict on the testing set
predicted_probs <- predict(model, newdata = testing_set, type = "response")
predicted_classes <- ifelse(predicted_probs > 0.5, "Yes", "No")
```

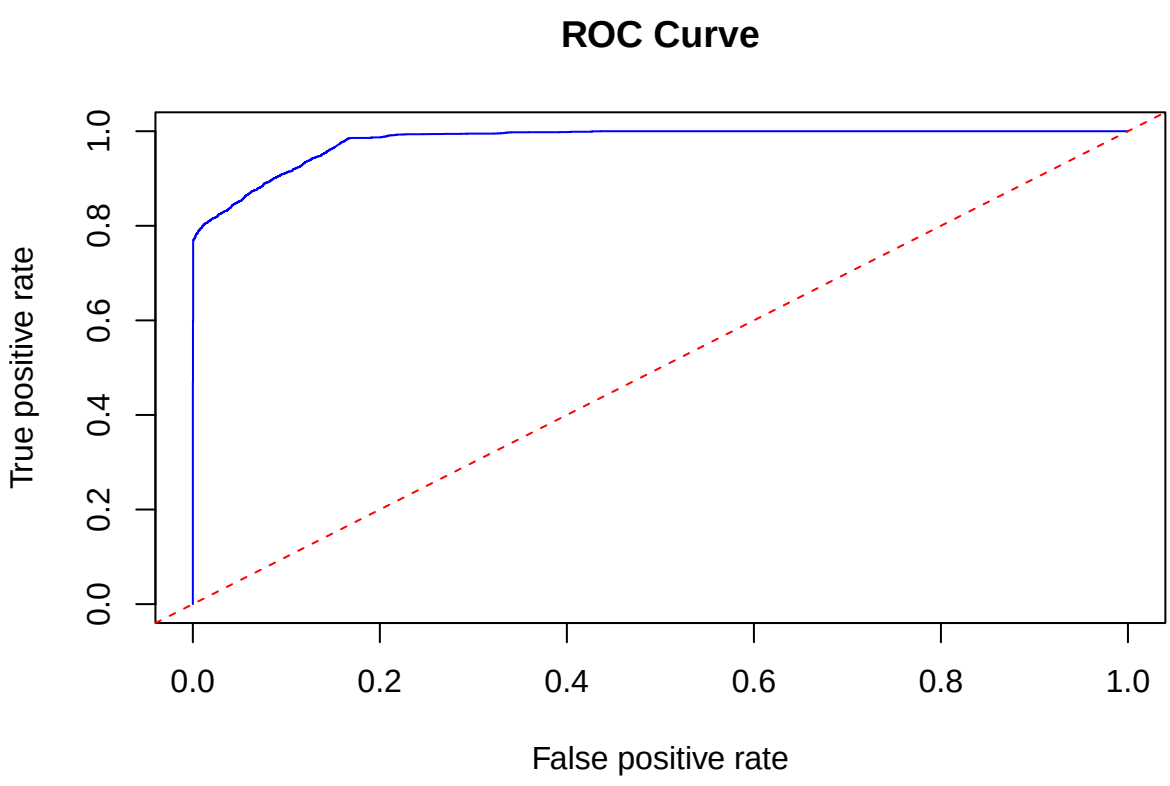
```r
# Calculate accuracy
accuracy <- mean(predicted_classes == testing_set$poswins)
cat("Model Accuracy: ", round(accuracy, 3), "\n")
```

```
## Model Accuracy:  0.904
```

```r
# Additional metrics (e.g., AUC)
pred <- prediction(predicted_probs, testing_set$poswins)
perf <- performance(pred, "tpr", "fpr")
auc <- performance(pred, "auc")@y.values[[1]]
cat("AUC: ", round(auc, 3), "\n")
```

```
## AUC:  0.98
```

```r
# Plot ROC Curve
plot(perf, col = "blue", main = "ROC Curve")
abline(a = 0, b = 1, lty = 2, col = "red")
```



###Question 2

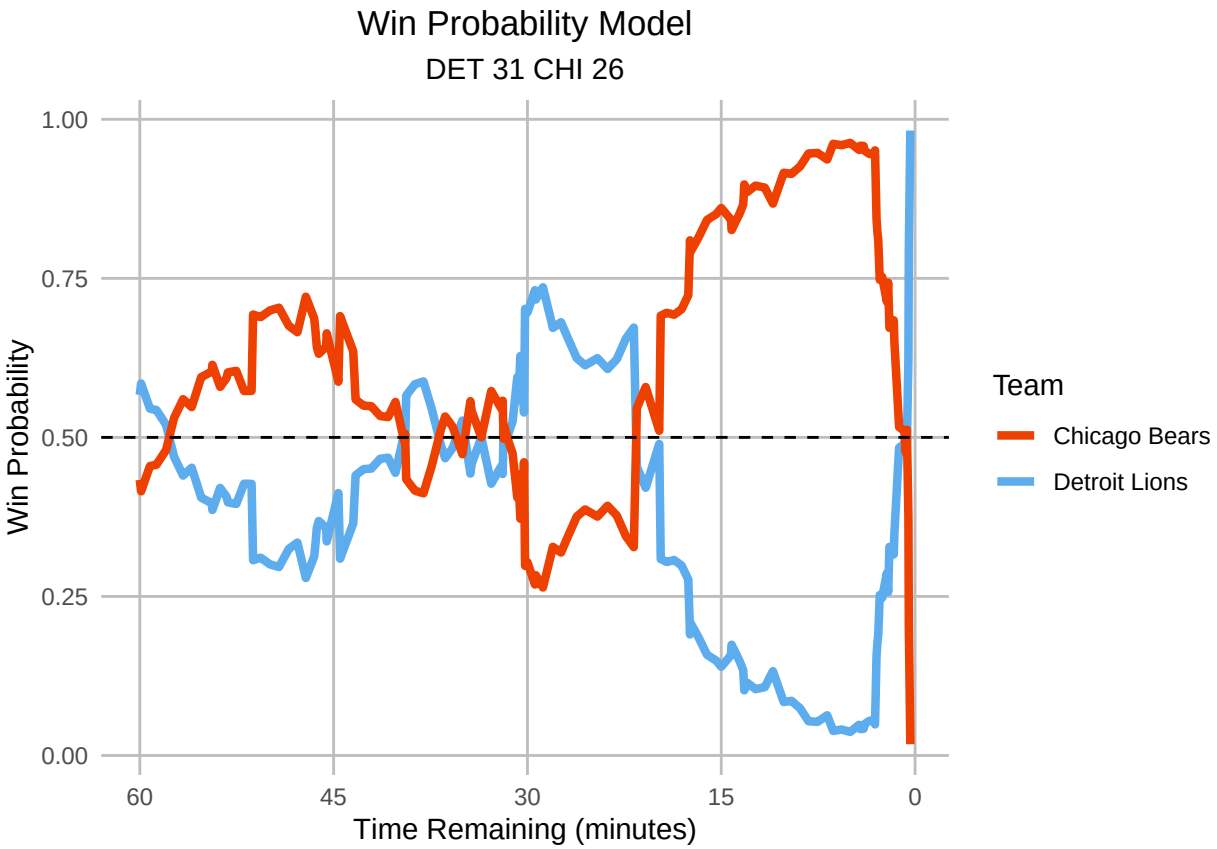
```r
# Filter for the specific game_id '2023_11_CHI_DET'
specific_game <- filtered_pbp %>%
  filter(game_id == '2023_11_CHI_DET') %>%
```

```r
  mutate(
    time_remaining = game_seconds_remaining / 60  # Convert to minutes
  )

# Plot win probabilities for the specific game
ggplot(specific_game) +
  geom_line(aes(x = time_remaining, y = home_wp, color = "Detroit Lions"), size = 1.5) +
  geom_line(aes(x = time_remaining, y = away_wp, color = "Chicago Bears"), size = 1.5) +
  scale_color_manual(values = c("Detroit Lions" = "steelblue2", "Chicago Bears" = "orangered2")) +
  labs(
    title = "Win Probability Model",
    subtitle = "DET 31 CHI 26",
    x = "Time Remaining (minutes)",
    y = "Win Probability",
    color = "Team"
  ) +
  theme_minimal() +
  theme(
    plot.title = element_text(hjust = 0.5),
    plot.subtitle = element_text(hjust = 0.5),
    axis.text.x = element_text(hjust = .5),
    panel.grid.major = element_line(color = "grey", size = 0.5),
    panel.grid.minor = element_blank()
  ) +
  scale_x_reverse(breaks = seq(0, 60, by = 15), labels = c("0", "15", "30", "45", "60")) +
  geom_hline(yintercept = 0.5, linetype = "dashed", color = "black")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
## Warning: The ‘size‘ argument of ‘element_line()‘ is deprecated as of ggplot2 3.4.0.
## i Please use the ‘linewidth‘ argument instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```


Win Probability Model
DET 31 CHI 26
1.00
0.75
0.50
0.25
0.00
Win Probability
60
45
30
15
0
Time Remaining (minutes)
Team
Chicago Bears
Detroit Lions