# code switching

Memoo Guo

2025-05-01

```r
# --- 0. Setup ---
library(readr)
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
```

```r
library(tidyr)

# --- 1. Load and Prepare Data ---
file_path <- "/Users/jack/Downloads/Chinese switching.txt" #
data <- read_csv(file_path, col_types = cols(
  Language = col_factor(),
  Speaker = col_factor(),
  Word = col_character(),
  POS = col_factor()
))

# --- 2. Feature Engineering ---
# Create features based on lagged values (t-1)
data <- data %>%
  mutate(
    # Previous Language State (S_{t-1})
    Prev_Language = lag(Language, default = first(Language)),
    # Indicator for Previous State being L2 (using '2' as L2 identifier based on data)
    Prev_State_L2 = ifelse(Prev_Language == "2", 1, 0),
    # Contextual Feature (xt): Previous POS tag using the refined 'POS' column
    Prev_POS = lag(POS, default = first(POS))
    # Add other features from xt if needed (e.g., Current_Speaker = Speaker)
  )

# Prepare the target variable: Is the current state L2?
```

```r
data <- data %>%
  mutate(Current_State_L2 = ifelse(Language == "2", 1, 0))

# Select relevant columns for the logistic regression model
# Using Prev_POS (refined tags) and Speaker as example contextual features (xt)
model_data <- data %>%
  select(Current_State_L2, Prev_State_L2, Prev_POS, Speaker)

# --- 3. Fit the Logistic Regression Model for Transitions ---

# Model P(S_t = L2 | S_{t-1}, xt) using logistic regression
# Target: Current_State_L2 (0 or 1)
# Predictors: Intercept (alpha), Prev_POS (beta part), Speaker (beta part),
#             Prev_State_L2 (gamma part)
transition_model <- glm(Current_State_L2 ~ 1 + Prev_POS + Speaker + Prev_State_L2,
                        data = model_data,
                        family = binomial(link = "logit"))

# --- 4. Examine Model Output ---
summary(transition_model)
```

```
##
## Call:
## glm(formula = Current_State_L2 ~ 1 + Prev_POS + Speaker + Prev_State_L2,
##     family = binomial(link = "logit"), data = model_data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -2.5126     1.2220  -2.056   0.0398 *
## Prev_POSNOUN    0.9208     1.2672   0.727   0.4675
## Prev_POSADJ     1.3508     1.3739   0.983   0.3255
## Prev_POSADV     0.6763     1.2952   0.522   0.6016
## Prev_POSVERB    0.2175     1.2569   0.173   0.8626
## Prev_POSPREP    0.5691     1.3145   0.433   0.6650
## Prev_POSCONJ    1.5338     1.5115   1.015   0.3102
## Prev_POSPRON   -0.3640     1.2939  -0.281   0.7785
## Prev_POSDET     0.8480     1.3140   0.645   0.5187
## Prev_POSMODAL -12.0535   882.7442  -0.014   0.9891
## Speaker2       -0.8295     0.3649  -2.273   0.0230 *
## Prev_State_L2   4.7209     0.3955  11.935   <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: 527.97  on 380  degrees of freedom
## Residual deviance: 233.85  on 369  degrees of freedom
## AIC: 257.85
##
## Number of Fisher Scoring iterations: 13
```

This is the main model attempting to predict the probability of the current word being L2 (Current_State_L2 = 1) using the previous word's POS tag (Prev_POS), the current speaker (Speaker), and

whether the previous word was L2 (Prev_State_L2).

- Prev_State_L2: Coefficient 4.7209, p < 2e-16 (***). Highly significant and positive. This confirms a strong language persistence effect. If the previous word was L2, the odds of the current word being L2 are much higher ($e^{4.72} \approx 112$ times higher), holding other variables constant.
- Speaker2: Coefficient -0.8295, p = 0.023 (*). Significant and negative. Speaker 2 is less likely to use L2 than Speaker 1 (the reference), holding other factors constant.
- (Intercept): Coefficient -2.5126, p = 0.0398 (*). Significant and negative. There's a baseline tendency towards L1 (English) when the previous word was L1, the speaker is Speaker 1, and the previous POS was the reference level.
- Problematic Predictor: Prev_POSMODAL has a huge standard error (882.7) and non-significant p-value. This indicates potential issues like quasi-complete separation or very few instances of Prev_POS == MODAL, making its effect unreliable or possibly indicating an error in the input data (as "MODAL" wasn't in the final agreed list of 9 tags).

```
# --- 5. Calculate Transition Probabilities ---
newdata <- data.frame(Prev_POS = factor("NOUN", levels=levels(model_data$Prev_POS)),
                      Speaker = factor("1", levels=levels(model_data$Speaker)),
                      Prev_State_L2 = 0)
# Case: Previous state was L1
prob_L2_given_L1 <- predict(transition_model, newdata = newdata, type = "response")
print(paste("Predicted P(St=L2 | St-1=L1, xt):", round(prob_L2_given_L1, 3)))
```

```
## [1] "Predicted P(St=L2 | St-1=L1, xt): 0.169"
```

```
newdata$Prev_State_L2 = 1
# Case: Previous state was L2
prob_L2_given_L2 <- predict(transition_model, newdata = newdata, type = "response")
print(paste("Predicted P(St=L2 | St-1=L2, xt):", round(prob_L2_given_L2, 3)))
```

```
## [1] "Predicted P(St=L2 | St-1=L2, xt): 0.958"
```

```
# Construct the time-dependent transition matrix At based on these probabilities:
A_t = matrix(c(1 - prob_L2_given_L1, prob_L2_given_L1,
               1 - prob_L2_given_L2, prob_L2_given_L2),
             nrow=2, byrow=FALSE)
rownames(A_t) <- c("L1_to", "L2_to")
colnames(A_t) <- c("Prev_L1", "Prev_L2")
print(A_t)
```

```
##         Prev_L1    Prev_L2
## L1_to 0.8308682 0.04192387
## L2_to 0.1691318 0.95807613
```

Column 1 (Prev_L1): If the previous word was L1, the chance of staying L1 is ~83.1%, chance of switching to L2 is ~16.9%.

Column 2 (Prev_L2): If the previous word was L2, the chance of switching to L1 is ~4.2%, chance of staying L2 is ~95.8%.

```r
simpler_model <- glm(Current_State_L2 ~ 1 + Speaker + Prev_State_L2,
                     data = model_data, # Use the same data
                     family = binomial(link = "logit"))
summary(simpler_model)
```

```
##
## Call:
## glm(formula = Current_State_L2 ~ 1 + Speaker + Prev_State_L2,
##     family = binomial(link = "logit"), data = model_data)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.8674     0.2757  -6.772 1.27e-11 ***
## Speaker2       -0.8114     0.3587  -2.262   0.0237 *
## Prev_State_L2   4.4985     0.3587  12.541  < 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: 527.97  on 380  degrees of freedom
## Residual deviance: 241.86  on 378  degrees of freedom
## AIC: 247.86
##
## Number of Fisher Scoring iterations: 5
```