ex 10 mixed effects models

2025-03-17

Exercise 10: Mixed effects

This homework assignment is designed to give you practice fitting and interpreting mixed effects models.

We will be using the LexicalData.csv and Items.csv files from the Homework/lexDat folder in the class GitHub repository again.

This data is a subset of the English Lexicon Project database. It provides the reaction times (in milliseconds) of many subjects as they are presented with letter strings and asked to decide, as quickly and as accurately as possible, whether the letter string is a word or not. The Items.csv provides characteristics of the words used, namely frequency (how common is this word?) and length (how many letters?). Unlike in the previous homework, there isn't any missing data in the LexicalData.csv file.

Data courtesy of Balota, D.A., Yap, M.J., Cortese, M.J., Hutchison, K.A., Kessler, B., Loftis, B., Neely, J.H., Nelson, D.L., Simpson, G.B., & Treiman, R. (2007). The English Lexicon Project. Behavior Research Methods, 39, 445-459.

1. Loading and formatting the data (1 point)

Load in data from the LexicalData.csv and Items.csv files. As in the previous homeworks, remove the commas from the reaction times and convert them from strings to numbers. Use left_join to add word characteristics Length and Log_Freq_Hal from Items to LexicalData.

Note: the Freq_HAL variable in Items.csv has a similar formatting issue, using string values with commas. We're not going to worry about fixing this since we're only using Log_Freq_HAL, which is the natural log transformation of Freq_HAL, in this homework.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
list.files()
```

```
##  [1] "data-as-objects-and-architectures.pdf"
##  [2] "davis-troller_hw10.pdf"
##  [3] "davis-troller_hw10.Rmd"
```

```
##  [4] "davisi-troller_hw10.Rmd"
##  [5] "Guest_Martin_2022.pdf"
##  [6] "hw10.Rmd"
##  [7] "IMG_3885.jpeg"
##  [8] "lexDat"
##  [9] "LexicalData_toclean.csv"
## [10] "models-as-testable-hypotheses.pdf"
## [11] "Popper_1959.pdf"
## [12] "techniques-for-data-cleansing.ipynb"
## [13] "techniques-for-data-cleansing.pdf"
## [14] "visualization-through-human-eyes.pdf"
```

```
setwd("lexDat")
lex_data <- read.csv("LexicalData.csv")
items <- read.csv("Items.csv")


clean_df = lex_data
clean_df$D_RT = gsub(",","",clean_df$D_RT)# want to remove commas gsub
clean_df$D_RT = as.numeric(clean_df$D_RT) #turn D_RT into number

head(clean_df)
```

```
##   Sub_ID Trial Type D_RT      D_Word Outlier D_Zscore
## 1    157     1    1  710      browse   false   -0.437
## 2     67     1    1 1094 refrigerant   false    0.825
## 3    120     1    1  587     gaining   false   -0.645
## 4     21     1    1  984   cheerless   false    0.025
## 5    236     1    1  577    pattered   false   -0.763
## 6    236     2    1  715    conjures   false   -0.364
```

```
items <- items %>%
  select(Word, Length, Log_Freq_HAL)

lex_data_clean <- left_join(clean_df, items, by = c("D_Word" = "Word"))%>%
  drop_na()

head(lex_data_clean)
```

```
##   Sub_ID Trial Type D_RT      D_Word Outlier D_Zscore Length Log_Freq_HAL
## 1    157     1    1  710      browse   false   -0.437      6        8.856
## 2     67     1    1 1094 refrigerant   false    0.825     11        4.644
## 3    120     1    1  587     gaining   false   -0.645      7        8.304
## 4     21     1    1  984   cheerless   false    0.025      9        2.639
## 5    236     1    1  577    pattered   false   -0.763      8        1.386
## 6    236     2    1  715    conjures   false   -0.364      8        5.268
```

2. Model fitting (4 points) First, fit a linear model with Log_Freq_HAL and Length as predictors, and D_RT as the output. Include an interaction term. Use summary() to look at the model output.

```r
lmmodel <- lm(D_RT ~ Log_Freq_HAL * Length, data = lex_data_clean)
summary(lmmodel)
```

```
##
## Call:
## lm(formula = D_RT ~ Log_Freq_HAL * Length, data = lex_data_clean)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1118.01  -205.23   -86.95    90.77  3147.07
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)         610.1903    14.6678  41.601  < 2e-16 ***
## Log_Freq_HAL         -6.0239     1.9678  -3.061  0.00221 **
## Length               47.7531     1.6368  29.175  < 2e-16 ***
## Log_Freq_HAL:Length  -2.9421     0.2348 -12.528  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 359.1 on 62606 degrees of freedom
## Multiple R-squared:  0.09473,    Adjusted R-squared:  0.09469
## F-statistic:  2184 on 3 and 62606 DF,  p-value: < 2.2e-16
```

Now, install lme4 using install.packages() and then load the library.

```r
#install.packages("lme4")
library(lme4)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

Now fit a mixed effects model that includes the same predictors as the linear model above, as well as random intercepts for Sub_ID (i.e., cases where subject ID shifts the RT mean). Use summary() to look at the model output.

```r
mixed_model <- lmer(D_RT ~ Log_Freq_HAL * Length + (1 | Sub_ID), data = lex_data_clean)
summary(mixed_model)
```

```
## Linear mixed model fit by REML ['lmerMod']
## Formula: D_RT ~ Log_Freq_HAL * Length + (1 | Sub_ID)
##    Data: lex_data_clean
##
## REML criterion at convergence: 888235.6
##
```

```
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.5058 -0.5472 -0.1568  0.3103 10.7381
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Sub_ID   (Intercept) 46333    215.3
##  Residual             82978    288.1
## Number of obs: 62610, groups:  Sub_ID, 299
##
## Fixed effects:
##                     Estimate Std. Error t value
## (Intercept)         616.8445    17.1522  35.963
## Log_Freq_HAL         -7.4374     1.5830  -4.698
## Length               47.7477     1.3162  36.277
## Log_Freq_HAL:Length  -2.8778     0.1888 -15.239
##
## Correlation of Fixed Effects:
##             (Intr) Lg_F_HAL Length
## Log_Frq_HAL -0.645
## Length      -0.656  0.917
## Lg_Fr_HAL:L  0.582 -0.942   -0.923
```

3. Model Assessment (4 point) Compare the three t-values for the fixed effects and the mixed effects models. How do they differ, and why? Fixed effects: Mixed effects: log -3.061 -4.698 length 29.175 36.277 interaction -12.528 -15.239

The mixed effect t-values are all a little more extreme, further away from 0. This is likely the case because the mixed model reduces residual noise.

Use the Aikaike Information Criterion (AIC) to compare these two models. Which one is better?

```
AIC(mixed_model)
```

```
## [1] 888247.6
```

```
AIC(lmmodel)
```

```
## [1] 914436.4
```

The lower AIC is the better model. Therefore, the mixed model is the better model even after accounting for the complexity of the model.

4. Reflection (1 point) What other random effects could be controlled for in this data set?

If data are collected in blocks or trials over time, we could control for learning or fatigue effects as the task continues.