Prediction and Simulation for Multi-Level Models

Andrew Zieffler

Educational Psychology

University of Minnesota

Driven to Discover℠

Prediction for Conventional Regression

1. Specify a matrix of the values for the predictors
2. Compute the predicted values (y-hat)
3. Simulate using the predicted data (to obtain prediction intervals, or to simulate a new set of Y's)

Read in the NFL data and fit the model

```
Lfci ~ ageStadium + I(ageStadium ^ 2) + LcoachYrswTeam
```

```
Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     6.225e+00  6.122e-02 101.691  < 2e-16 ***
ageStadium     -1.442e-02  3.694e-03  -3.903 0.000544 ***
I(ageStadium^2) 1.803e-04  4.528e-05   3.983 0.000440 ***
LcoachYrswTeam  8.150e-02  2.859e-02   2.851 0.008097 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1335 on 28 degrees of freedom
Multiple R-squared:  0.4625,	Adjusted R-squared:  0.4049
F-statistic:  8.03 on 3 and 28 DF,  p-value: 0.0005141
```

Predict for 2016 Vikings and Packers

1. Specify a matrix of the values for the predictors

```
ageStadium = 7
LcoachYrswTeam = log(2 + 1) = 1.098612

ageStadium = 59
LcoachYrswTeam = log(10 + 1) = 2.397895
```

```
> myData = data.frame(
     ageStadium = c(7, 59),
     LcoachYrswTeam = c(1.098612, 2.397895)
     )

> myData
  ageStadium LcoachYrswTeam
1          7       1.098612
2         59       2.397895
```

2. Compute the predicted values (y-hat)

```
> predict(lm.a, newdata = myData)

       1        2
6.222686 6.197595
```

```
> myMatrix = cbind(
     rep(1, 2),
     c(7, 59),
     c(49, 3481),
     c(1.098612, 2.397895)
     )

> myMatrix
     [,1] [,2] [,3]     [,4]
[1,]    1    7   49 1.098612
[2,]    1   59 3481 2.397895

> coef1 = matrix(coef(lm.a))
> coef1
              [,1]
[1,]  6.2252510003
[2,] -0.0144202338
[3,]  0.0001803219
[4,]  0.0815037059

> myMatrix %*% coef1
         [,1]
[1,] 6.222686
[2,] 6.197595
```

An alternative method for obtaining the fitted values is via matrix algebra

$$\hat{y} = \tilde{\mathbf{X}}\hat{\beta}$$

3. Simulate using the predicted data (to obtain prediction intervals, or to simulate a new set of Y's)

```
> mySim = sim(lm.a, 1)
> mySim
An object of class "sim"
Slot "coef":
        [,1]        [,2]         [,3]       [,4]
[1,] 6.20946 -0.01157016 0.0001438249 0.07384828

Slot "sigma":
[1] 0.1174558
```

sim() accounts for model uncertainty by sampling the regression coefficients

```
> myMatrix %*% mySim@coef[1, ]
         [,1]
[1,] 6.216647
[2,] 6.204556
```

Use the coefficients to compute the simulated y-hats

```
> myMatrix %*% mySim@coef[1, ] + rnorm(2, mean = 0, sd = mySim@sigma)
         [,1]
[1,] 6.216647
[2,] 6.204556
```

Use the rnorm() function to account for prediction uncertainty

Multilevel Models

Read in and Prepare Data for these Notes

```
# Load foreign package to be able to read in SPSS data
> library(foreign)

# Read in the level-1 (player-level) data
> nbaL1 = read.spss(file = "http://www.tc.umn.edu/~zief0002/data/nbaLevel1.sav",
    to.data.frame = TRUE)

# Read in the level-2 (team-level) data
> nbaL2 = read.spss(file = "http://www.tc.umn.edu/~zief0002/data/nbaLevel2.sav",
    to.data.frame = TRUE)

# Merge nbaL2 into nbaL1 using the Team_ID variable
> nba = merge(nbaL1, nbaL2, by = "Team_ID")

# Load libraries
> library(lme4)
> library(dplyr)
```

Group Mean Center the Level-1 Predictor

```
# Compute the mean for each team
> teams = nba %.%
    group_by(Team_ID) %.%
    summarise(teamMean = mean(Shots_on_five))

> head(teams)

  Team_ID meanShots
1      01       3.0
2      02       3.7
3      03       3.3
4      04       3.3
5      05       1.5
6      06       2.7

# Merge the team means with the nba data frame
> nba = merge(nba, teams, by = “Team_ID”)

# Compute the group mean deviations
> nba$SO5 = nba$Shots_on_five - nba$teamMean
```

Fit the Model

Level-1: $Y_{ij} = \beta_0^* + \beta_1^*(\text{SO5}) + \epsilon_{ij}$ where $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$

Level-2: $\beta_0^* = \beta_{00} + \beta_{01}(\text{CE}) + b_{0j}$
$\beta_1^* = \beta_{10} + b_{1j}$ where $\begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_0^2 & 0 \\ 0 & \sigma_1^2 \end{bmatrix}\right)$

Composite: $Y_{ij} = \beta_{00} + \beta_{01}(\text{CE}) + \beta_{10}(\text{SO5}) + [b_{0j} + b_{1j}(\text{SO5}) + \epsilon_{ij}]$

```
# Fit the model using REML for better variance estimates
> lmer.1 = lmer(Life_Satisfaction ~ 1 + SO5 + Coach_Experience +
    (1 | Team_ID) + (0 + SO5 | Team_ID), data = nba)
```

```
> sigma.hat(lmer.1)

$sigma
$sigma$data
[1] 2.149366

$sigma$Team_ID
(Intercept)
   1.635307

$sigma$Team_ID.1
      SO5
0.7961247


$cors
$cors$data
[1] NA

$cors$Team_ID
[1] NA

$cors$Team_ID.1
[1] NA
```

`sigma.hat()` from the **arm** library produces the residual error estimates (as SDs and correlations)

These are the same estimates from the Std.Dev. column of the summary() output

```
> coef(lmer.1)

$Team_ID
   (Intercept)      SO5 Coach_Experience
01    6.840518 3.280400         4.784303
02    6.405044 2.940759         4.784303
03    9.344543 3.086668         4.784303
04    6.957849 3.238191         4.784303
05    5.705497 2.565480         4.784303
06    6.228794 3.748913         4.784303
07    3.695694 2.201679         4.784303
08    4.685157 3.402976         4.784303
09    6.857489 2.351816         4.784303
10    4.056297 3.284135         4.784303
11    7.624062 3.656969         4.784303
12    3.630631 3.946794         4.784303
13    6.011951 2.381322         4.784303
14    4.039330 1.777575         4.784303
15    4.647044 2.341654         4.784303
16    4.679447 3.120067         4.784303
17    4.297697 2.182346         4.784303
18    4.441286 2.655086         4.784303
19    4.699144 3.251258         4.784303
20    3.866828 3.412657         4.784303
21    3.819930 3.439804         4.784303
22    6.402150 2.437140         4.784303
23    5.448320 2.477962         4.784303
24    3.728949 2.172226         4.784303
25    6.241073 3.162423         4.784303
26    5.482091 2.453972         4.784303
27    6.645762 3.747956         4.784303
28    5.747616 3.300996         4.784303
29    6.968676 2.259027         4.784303
30    2.727369 2.657018         4.784303
```

`coef()` produces the group-level (team) regression coefficients

Prediction for a new observation from an existing group

```
> coef(lmer.1)$Team_ID[10, ]

   (Intercept)      SO5 Coach_Experience
10    4.056297 3.284135         4.784303
```

Get the 10th team's regression coefficients

```
> team.betas = as.matrix(coef(lmer.1)$Team_ID[10, ])
> team.betas

   (Intercept)      SO5 Coach_Experience
10    4.056297 3.284135         4.784303
```

Put those coefficients into a vector

```
> myMatrix = cbind(1, -1.2, 2)

> myMatrix
     [,1] [,2] [,3]
[1,]    1 -1.2    2
```

Set up a matrix of X's to predict from (team 10's values)

SO5 = -1.2
CE = 2

Need to account for the within-group variation

```
> yhat = myMatrix %*% t(team.betas)
> yhat
           10
[1,] 9.68394
```

team.betas is a 1x3 vector, so we need to transpose it

```
> rnorm(1, yhat, within.team.error)
[1] 8.743602
```

Prediction for a new observation from Team 10

```
> mySim = rnorm(1000, yhat, within.team.error)

> quantile(mySim, probs = c(0.025, 0.975))
     2.5%     97.5%
 5.816525 13.671626
```

Prediction interval for Team 10

```
> within.team.error = sigma.hat(lmer.1)$sigma$data

$sigma
$sigma$data
[1] 2.149366

$sigma$Team_ID
(Intercept)
   1.635307

$sigma$Team_ID.1
      SO5
0.7961247


$cors
$cors$data
[1] NA

$cors$Team_ID
[1] NA

$cors$Team_ID.1
[1] NA
```

Prediction for a new observation from a new group

Multilevel Models to Gelman's Notation

Level-1: $Y_{ij} = \beta_0^* + \beta_1^*(\mathrm{SO5}) + \epsilon_{ij}$ where $\epsilon_{ij} \sim N(0, \sigma_\epsilon^2)$

Level-2: $\beta_0^* = \beta_{00} + \beta_{01}(\mathrm{CE}) + b_{0j}$
$\beta_1^* = \beta_{10} + b_{1j}$ where $\begin{bmatrix} b_0 \\ b_1 \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma_0^2 & 0 \\ 0 & \sigma_1^2 \end{bmatrix}\right)$

$$Y_{ij} = \mathcal{N}\left(\beta_0^* + \beta_1^*(\mathrm{SO5}), \sigma_y^2\right)$$

$$\beta_0^* \sim \left(\beta_{00} + \beta_{01}(\mathrm{CE}), \sigma_{b_{0j}}^2\right)$$

$$\beta_1^* \sim \left(\beta_{10}, \sigma_{b_{1j}}^2\right)$$

Need to account for the within-group variation and between-group variation

$$\beta_0^* \sim \left(\beta_{00} + \beta_{01}(\mathrm{CE}), \sigma^2_{b_{0j}}\right)$$

$$\beta_1^* \sim \left(\beta_{10}, \sigma^2_{b_{1j}}\right)$$

Use these to randomly sample a B^*_0 and B^*_1

$$Y_{ij} = \mathcal{N}\left(\beta_0^* + \beta_1^*(\mathrm{SO5}), \sigma_y^2\right)$$

Use the sampled B^*_0 and B^*_1 along with a randomly sampled error to compute a Y

$$\beta_0^* \sim \left(\beta_{00} + \beta_{01}(\mathrm{CE}), \sigma^2_{b_{0j}}\right)$$

$$\beta_1^* \sim \left(\beta_{10}, \sigma^2_{b_{1j}}\right)$$

```
> Coach_Experience = mean(nba$Coach_Experience)
```

Assume team's CE = average CE

```
> b00 = fixef(lmer.1)["(Intercept)"]
> b01 = fixef(lmer.1)["Coach_Experience"]

> s0j = sigma.hat(lmer.1)$sigma$Team_ID

> bstar0 = rnorm(1, mean = (b00 + b01*Coach_Experience), sd = s0j)
> bstar0
[1] 12.27907
```

Simulate a B^*_0

```
> b10 = fixef(lmer.1)["SO5"]

> s1j = sigma.hat(lmer.1)$sigma$Team_ID.1

> bstar1 = rnorm(1, mean = (b10), sd = s1j)
> bstar1
[1] 2.833721
```

Simulate a B^*_1

$$Y_{ij} = \mathcal{N}\left(\beta_0^* + \beta_1^*(\mathrm{SO5}), \sigma_y^2\right)$$

```
> SO5 = mean(nba$SO5)
```

Assume team's SO5 = average SO5

```
> within.team.error = sigma.hat(lmer.1)$sigma$data

> Y = rnorm(1, mean = (bstar0 + bstar1*SO5), sd = within.team.error)
> Y
[1] 11.46452
```

Simulate a Y