Latent Class Analysis with MplusAutomation

Dina Arch

2025-02-13

Contents

Mixture Modeling with `MplusAutomation`

Welcome! This will be a collection of resources that will teach you how to apply mixture modeling using Mplus(**?**) and `MplusAutomation`(**?**)! These resources will serve as a comprehensive guide to understanding and applying LCA using Mplus and its automation capabilities with `MplusAutomation`. Here, you will learn from start to finish how to apply mixture modeling using Mplus with the `MplusAutomation` package.

Stay in touch!

- Please visit our website to learn more about the IMMERSE fellowship.
- Visit our GitHub account to access all the IMMERSE training materials.
- Follow us on BlueSky and X to stay-up-to date on our fellowship!

Acknowledgements

The Institute of Mixture Modeling for Equity-Oriented Researchers, Scholars, and Educators (IMMERSE) is an IES funded training grant (R305B220021) to support education scholars in integrating mixture modeling into their research.

How to reference this workshop: Institute of Mixture Modeling for Equity-Oriented Researchers, Scholars, and Educators (2025). IMMERSE Online Resources (IES No. 305B220021). Institute of Education Sciences. https://bookdown.org/education_immerse/lca-bookdown/

Chapter 1

Introduction to R and RStudio

This walkthrough is presented by the IMMERSE team and will go through some common tasks carried out in R. There are many free resources available to get started with R and RStudio. One of our favorites is *R for Data Science*.

- *R*(**?**) is a free, open-source programming language and environment widely used for statistical computing, data analysis, and data visualization.
- *RStudio*(**?**) is an integrated development environment (IDE) for R, providing an intuitive interface that makes coding, visualization, and project management more accessible.
- *Mplus*(**?**) is a statistical modeling program used for analyzing complex data, such as latent variable models, structural equation modeling, and growth modeling. This book uses an R package called `MplusAutomation` to automate the process of running models, extracting results, and generating data visualizations.

1.1 Installation

1.1.1 Step 0: Install R, RStudio, and Mplus

Here you will find a guide to installing both R and R Studio. You can also install Mplus here.

Note: The installation of Mplus requires a paid license with the mixture add-on. IMMERSE fellows will be given their own copy of Mplus for use during the one year training.

1.2 Set-up

1.2.1 Step 1: Create a new R-project in RStudio

R-projects help us organize our folders , filepaths, and scripts. To create a new R project:

- File –> New Project…

Click "New Directory" –> New Project –> Name your project

1.2.2 Step 2: Create an R-markdown document

An R-markdown file provides an authoring framework for data science that allows us to organize our reports using texts and code chunks. This document you are reading was made using R-markdown!

To create an R-markdown:

- File –> New File –> R Markdown…

In the window that pops up, give the R-markdown a title such as "**Introduction to R and RStudio**" Click "OK." You should see a new markdown with some example text and code chunks. We want a clean document to start off with so delete everything from line 10 down. Go ahead and save this document in your R Project folder.

1.2.3 Step 3: Load packages

Your first code chunk in any given markdown should be the packages you will be using. To insert a code chunk, etiher use the keyboard shortcut ctrl + alt + i or Code –> Insert Chunk or click the green box with the letter C on it. There are a few packages we want our markdown to read in:

```r
library(psych) # describe()
library(here) #helps with filepaths
library(gt) # create tables
library(tidyverse) #collection of R packages designed for data science
```

As a reminder, if a function does not work and you receive an error like this: `could not find function "random_function"`; or if you try to load a package and you receive an error like this: `there is no package called `random_package``, then you will need to install the package using `install.packages("random_package")` in the console (the bottom-left window in R studio). Once you have installed the package you will *never* need to install it again, however you must *always* load in the packages at the beginning of your R markdown using `library(random_package)`, as shown in this document.

The style of code and package we will be using is called `tidyverse`(**?**) . Most functions are within the `tidyverse` package and if not, I've indicated the packages used in the code chunk above.

1.3 Explore the data

1.3.1 Step 4: Read in data

To demonstrate mixture modeling in the training program and online resource components of the IES grant we utilize the *Civil Rights Data Collection (CRDC)* (CRDC) data repository. The CRDC is a federally mandated school-level data collection effort that occurs every other year. This public data is currently available for selected latent class indicators across 4 years (2011, 2013, 2015, 2017) and all US states. In this example, we use the Arizona state sample. We utilize six focal indicators which constitute the latent class model in our example; three variables which report on harassment/bullying in schools based on disability, race, or sex, and three variables on full-time equivalent school staff

LCA indicators[1]

Name	Label
leaid	District Identification Code
ncessch	School Identification Code
report_dis	Number of students harassed or bullied on the basis of disability
report_race	Number of students harassed or bullied on the basis of race, color, or n
report_sex	Number of students harassed or bullied on the basis of sex
counselors_fte	Number of full time equivalent counselors hired as school staff
report_sex	Number of full time equivalent psychologists hired as school staff
counselors_fte	Number of full time equivalent law enforcement officers hired as school

[1] Civil Rights Data Collection (CRDC)

hires (counselor, psychologist, law enforcement). This data source also includes covariates on a variety of subjects and distal outcomes reported in 2018 such as math/reading assessments and graduation rates.

To read in data in R:

```
data <- read_csv(here("data", "crdc_lca_data.csv"))
```

Ways to view data in R:

1. click on the data in your Global Environment (upper right pane) or use...

```
View(data)
```

2. `summary()` gives basic summary statistics & shows number of NA values (great for checking that data has been read in correctly)

```
summary(data)
#>     leaid             ncessch            report_dis    
#>  Length:2027        Length:2027        Min.   :0.0000  
#>  Class :character   Class :character   1st Qu.:0.0000  
#>  Mode  :character   Mode  :character   Median :0.0000  
#>                                        Mean   :0.0425  
#>                                        3rd Qu.:0.0000  
#>                                        Max.   :1.0000  
#>                                        NA's   :27      
#>   report_race      report_sex   counselors_fte
#>  Min.   :0.000   Min.   :0.00   Min.   :0.0000  
#>  1st Qu.:0.000   1st Qu.:0.00   1st Qu.:0.0000  
#>  Median :0.000   Median :0.00   Median :0.0000  
#>  Mean   :0.103   Mean   :0.17   Mean   :0.4595  
#>  3rd Qu.:0.000   3rd Qu.:0.00   3rd Qu.:1.0000  
#>  Max.   :1.000   Max.   :1.00   Max.   :1.0000  
#>  NA's   :27      NA's   :27     NA's   :27      
#>    psych_fte          law_fte      
#>  Min.   :0.0000   Min.   :0.0000  
#>  1st Qu.:0.0000   1st Qu.:0.0000  
#>  Median :0.0000   Median :0.0000  
#>  Mean   :0.4742   Mean   :0.1255  
#>  3rd Qu.:1.0000   3rd Qu.:0.0000  
#>  Max.   :1.0000   Max.   :1.0000  
#>  NA's   :30       NA's   :27      
```

3. `names()` provides a list of column names. Very useful if you don't have them memorized!

```
names(data)
#> [1] "leaid"          "ncessch"        "report_dis"    
#> [4] "report_race"    "report_sex"     "counselors_fte"
#> [7] "psych_fte"      "law_fte"
```

4. head() prints the top 6 rows of the dataframe

```
head(data)
#> # A tibble: 6 x 8
#>   leaid   ncessch      report_dis report_race report_sex
#>   <chr>   <chr>             <dbl>       <dbl>      <dbl>
#> 1 0400001 040000100120          0           0          0
#> 2 0400001 040000100616          0           0          1
#> 3 0400001 040000101204          0           0          1
#> 4 0400001 040000101871          0           1          1
#> 5 0400001 040000101872          0           0          0
#> 6 0400001 040000102344          0           0          0
#> # i 3 more variables: counselors_fte <dbl>,
#> #   psych_fte <dbl>, law_fte <dbl>
```

1.3.2 Step 5: Descriptive Statistics

Let's look at descriptive statistics for each variable. Because looking at the ID variables' (`leaid`) and (`necessch`) descriptives is unnecessary, we use `select()` to remove the variable by using the minus (`-`) sign:

```
data %>%
  select(-leaid, -ncessch) %>%
  summary()
#>    report_dis      report_race      report_sex  
#>  Min.   :0.0000   Min.   :0.000   Min.   :0.00  
#>  1st Qu.:0.0000   1st Qu.:0.000   1st Qu.:0.00  
#>  Median :0.0000   Median :0.000   Median :0.00  
#>  Mean   :0.0425   Mean   :0.103   Mean   :0.17  
#>  3rd Qu.:0.0000   3rd Qu.:0.000   3rd Qu.:0.00  
#>  Max.   :1.0000   Max.   :1.000   Max.   :1.00  
#>  NA's   :27       NA's   :27      NA's   :27    
#>  counselors_fte     psych_fte         law_fte      
#>  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
#>  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000  
#>  Median :0.0000   Median :0.0000   Median :0.0000  
#>  Mean   :0.4595   Mean   :0.4742   Mean   :0.1255  
#>  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:0.0000  
#>  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000  
#>  NA's   :27       NA's   :30       NA's   :27     
```

Alternatively, we can use the `psych::describe()` function to give more information:

```
data %>%
  select(-leaid, -ncessch) %>%
  describe()
#>                vars    n mean   sd median trimmed mad min
#> report_dis        1 2000 0.04 0.20      0    0.00   0   0
#> report_race       2 2000 0.10 0.30      0    0.00   0   0
#> report_sex        3 2000 0.17 0.38      0    0.09   0   0
#> counselors_fte    4 2000 0.46 0.50      0    0.45   0   0
#> psych_fte         5 1997 0.47 0.50      0    0.47   0   0
#> law_fte           6 2000 0.13 0.33      0    0.03   0   0
#>                max range skew kurtosis   se
#> report_dis       1     1 4.53    18.55 0.00
#> report_race      1     1 2.61     4.82 0.01
#> report_sex       1     1 1.76     1.08 0.01
#> counselors_fte   1     1 0.16    -1.97 0.01
#> psych_fte        1     1 0.10    -1.99 0.01
#> law_fte          1     1 2.26     3.11 0.01
```

What if we want to look at a subset of the data? For example, what if we want to subset the data to observe a specific school district? (`leaid`) We can use `tidyverse::filter()` to subset the data using certain criteria.

```
data %>%
  filter(leaid == "0408800") %>%
  describe()
#>                 vars  n  mean    sd median trimmed   mad min
#> leaid*             1 86  1.00  0.00    1.0    1.00  0.00   1
#> ncessch*           2 86 43.50 24.97   43.5   43.50 31.88   1
#> report_dis         3 86  0.05  0.21    0.0    0.00  0.00   0
#> report_race        4 86  0.15  0.36    0.0    0.07  0.00   0
#> report_sex         5 86  0.19  0.39    0.0    0.11  0.00   0
#> counselors_fte     6 86  0.95  0.21    1.0    1.00  0.00   0
#> psych_fte          7 86  0.19  0.39    0.0    0.11  0.00   0
#> law_fte            8 86  0.14  0.35    0.0    0.06  0.00   0
#>                 max range  skew kurtosis   se
#> leaid*            1     0   NaN      NaN 0.00
#> ncessch*         86    85  0.00    -1.24 2.69
#> report_dis        1     1  4.23    16.10 0.02
#> report_race       1     1  1.91     1.68 0.04
#> report_sex        1     1  1.59     0.52 0.04
#> counselors_fte    1     1 -4.23    16.10 0.02
#> psych_fte         1     1  1.59     0.52 0.04
#> law_fte           1     1  2.04     2.21 0.04


#You can use any operator to filter: >, <, ==, >=, etc.
```

Since we have binary data (0,1), it would be helpful to look at the proportions:

```
data %>%
  drop_na() %>%
  pivot_longer(report_dis:law_fte, names_to = "variable") %>%
  group_by(variable) %>%
  summarise(prop = sum(value)/n(),
            n = n()) %>%
  arrange(desc(prop))
#> # A tibble: 6 x 3
#>   variable         prop     n
#>   <chr>           <dbl> <int>
#> 1 psych_fte      0.481   1970
#> 2 counselors_fte 0.459   1970
#> 3 report_sex     0.173   1970
#> 4 law_fte        0.127   1970
#> 5 report_race    0.105   1970
#> 6 report_dis     0.0431  1970
```

Chapter 2

Introduction to `MplusAutomation`

`MplusAutomation`(**?**) is designed to streamline the use of Mplus, a powerful statistical software for modeling complex data developed by Muthen and Muten (https://www.statmodel.com). With `MplusAutomation`, researchers can automate the process of estimating latent variable models, running batches of models, extracting results, and generating data visualizations - all within the R environment.

WHAT?

- `MplusAutomation` is an `R` package
- It "wraps around" the `Mplus` program
- Requires both `R` & `Mplus` software
- Requires learning some basics of 2 programming languages
- Car metaphor: R/Rstudio is the *steering wheel or dashboard* & Mplus is the *engine*

WHY?

- `MplusAutomation` can provide clearly organized work procedures in which every research decision can be documented in a single place

- Increase reproducibility, organization, efficiency, and transparency

---

HOW?

- The interface for MplusAutomation is entirely within R-Studio. You do not need to open Mplus
- The code presented will be very repetitive by design

Below is a template for `mplusObject()` & `mplusModeler()` functions. Use this template to run statistical models with Mplus.

```r
m_template  <- mplusObject(

  TITLE =
    "",

  VARIABLE =
    "",

  ANALYSIS =
    "",

  PLOT =
    "",

  OUTPUT =
    "",

  usevariables = colnames(),
  rdata =  )

m_template_fit <- mplusModeler(m_template,
                  dataout=here("", ".dat"),
                  modelout=here("", ".inp"),
                  check=TRUE, run = TRUE, hashfilename = FALSE)
```

Chapter 3

LCA Enumeration

Example: Bullying in Schools

To demonstrate mixture modeling in the training program and online resource components of the IES grant we utilize the *Civil Rights Data Collection (CRDC)*(**?**) data repository. The CRDC is a federally mandated school-level data collection effort that occurs every other year. This public data is currently available for selected latent class indicators across 4 years (2011, 2013, 2015, 2017) and all US states. In this example, we use the Arizona state sample. We utilize six focal indicators which constitute the latent class model in our example; three variables which report on harassment/bullying in schools based on disability, race, or sex, and three variables on full-time equivalent school staff hires (counselor, psychologist, law enforcement). This data source also includes covariates on a variety of subjects and distal outcomes reported in 2018 such as math/reading assessments and graduation rates.

Load packages

```r
library(tidyverse)
library(haven)
library(glue)
library(MplusAutomation)
library(here)
```

LCA indicators[1]

Name	Label
leaid	District Identification Code
ncessch	School Identification Code
report_dis	Number of students harassed or bullied on the basis of disability
report_race	Number of students harassed or bullied on the basis of race, color, or n
report_sex	Number of students harassed or bullied on the basis of sex
counselors_fte	Number of full time equivalent counselors hired as school staff
psych_fte	Number of full time equivalent psychologists hired as school staff
law_fte	Number of full time equivalent law enforcement officers hired as school

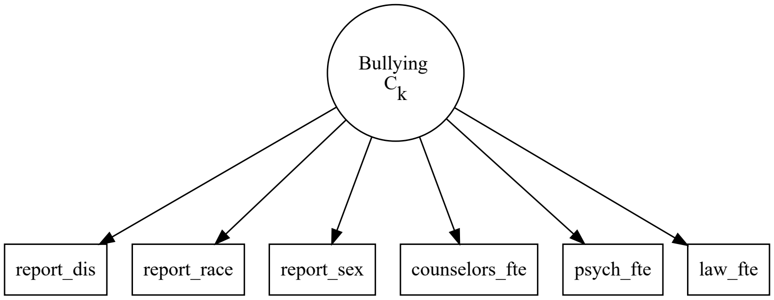
[1] Civil Rights Data Collection (CRDC)

```r
library(janitor)
library(gt)
library(cowplot)
library(DiagrammeR)
```

3.1 Variable Description

Variables have been transformed to be dichotomous indicators using the following coding strategy

Harassment and bullying count variables are recoded `1` if the school reported at least one incident of harassment (`0` indicates no reported incidents). On the original scale reported by the CDRC staff variables for full time equivalent employees (FTE) are represented as `1` and part time employees are represented by values between `1` and `0`. Schools with greater than one staff of the designated type are represented by values greater than 1. All values greater than zero were recorded as `1s` (e.g., `.5`, `1`,`3`) indicating that the school has a staff present on campus at least part time. Schools with no staff of the designated type are indicated as `0` for the dichotomous variable.


Bullying
C_k
report_dis
report_race
report_sex
counselors_fte
psych_fte
law_fte

3.2 Prepare Data

```
df_bully <- read_csv(here("data", "crdc_lca_data.csv")) %>%
  clean_names() %>%
  dplyr::select(report_dis, report_race, report_sex, counselors_fte, psych_fte, law_fte
```

3.3 Descriptive Statistics

```
# Set up data to find proportions of binary indicators
ds <- df_bully %>%
  pivot_longer(c(report_dis, report_race, report_sex, counselors_fte, psych_fte, law_ft


# Create table of variables and counts, then find proportions and round to 3 decimal p
prop_df <- ds %>%
  count(variable, value) %>%
  group_by(variable) %>%
  mutate(prop = n / sum(n)) %>%
  ungroup() %>%
  mutate(prop = round(prop, 3))


# Make it a gt() table
prop_table <- prop_df %>%
  gt(groupname_col = "variable", rowname_col = "value") %>%
  tab_stubhead(label = md("*Values*")) %>%
  tab_header(
    md(
      "Variable Proportions"
    )
  ) %>%
  cols_label(
    variable = md("*Variable*"),
    value = md("*Value*"),
    n = md("*N*"),
    prop = md("*Proportion*")
  )

prop_table
```

Variable Proportions

Values	*N*	*Proportion*
counselors_fte		
0	1081	0.533
1	919	0.453
NA	27	0.013
law_fte		
0	1749	0.863
1	251	0.124
NA	27	0.013
psych_fte		
0	1050	0.518
1	947	0.467
NA	30	0.015
report_dis		
0	1915	0.945
1	85	0.042
NA	27	0.013
report_race		
0	1794	0.885
1	206	0.102
NA	27	0.013
report_sex		
0	1660	0.819
1	340	0.168
NA	27	0.013

Save as image

```r
gtsave(prop_table, here("figures", "prop_table.png"))
```

3.4 Enumeration

This code uses the `mplusObject` function in the `MplusAutomation` package and saves all model runs in the `enum` folder.

```
lca_6  <- lapply(1:6, function(k) {
  lca_enum  <- mplusObject(

    TITLE = glue("{k}-Class"),

    VARIABLE = glue(
    "categorical = report_dis-law_fte;
     usevar = report_dis-law_fte;
     classes = c({k}); "),

  ANALYSIS =
   "estimator = mlr;
    type = mixture;
    starts = 200 100;
    processors = 10;",

  OUTPUT = "sampstat residual tech11 tech14;",

  PLOT =
    "type = plot3;
    series = report_dis-law_fte(*);",

  usevariables = colnames(df_bully),
  rdata = df_bully)

lca_enum_fit <- mplusModeler(lca_enum,
                            dataout=glue(here("enum", "bully.dat")),
                            modelout=glue(here("enum", "c{k}_bully.inp")) ,
                            check=TRUE, run = TRUE, hashfilename = FALSE)
})
```

IMPORTANT: Before moving forward, make sure to open each output document to ensure models were estimated normally.

3.5 Table of Fit

First, extract data:

```
#
output_bully <- readModels(here("enum"), filefilter = "bully", quiet = TRUE)

enum_extract <- LatexSummaryTable(
  output_bully,
  keepCols = c(
    "Title",
    "Parameters",
    "LL",
    "BIC",
    "aBIC",
    "BLRT_PValue",
    "T11_VLMR_PValue",
    "Observations"
  ),
  sortBy = "Title"
)


allFit <- enum_extract %>%
  mutate(CAIC = -2 * LL + Parameters * (log(Observations) + 1)) %>%
  mutate(AWE = -2 * LL + 2 * Parameters * (log(Observations) + 1.5)) %>%
  mutate(SIC = -.5 * BIC) %>%
  mutate(expSIC = exp(SIC - max(SIC))) %>%
  mutate(BF = exp(SIC - lead(SIC))) %>%
  mutate(cmPk = expSIC / sum(expSIC)) %>%
  dplyr::select(1:5, 9:10, 6:7, 13, 14) %>%
  arrange(Parameters)
```

Then, create table:

```
fit_table1 <- allFit %>%
  gt() %>%
  tab_header(title = md("**Model Fit Summary Table**")) %>%
  cols_label(
    Title = "Classes",
    Parameters = md("Par"),
    LL = md("*LL*"),
    T11_VLMR_PValue = "VLMR",
    BLRT_PValue = "BLRT",
    BF = md("BF"),
    cmPk = md("*cmPk*")
  ) %>%
  tab_footnote(
    footnote = md(
      "*Note.* Par = Parameters; *LL* = model log likelihood;
BIC = Bayesian information criterion;
aBIC = sample size adjusted BIC; CAIC = consistent Akaike information criterion;
AWE = approximate weight of evidence criterion;
BLRT = bootstrapped likelihood ratio test p-value;
VLMR = Vuong-Lo-Mendell-Rubin adjusted likelihood ratio test p-value;
*cmPk* = approximate correct model probability."
    ),
locations = cells_title()
  ) %>%
  tab_options(column_labels.font.weight = "bold") %>%
  fmt_number(c(3:7),
             decimals = 2) %>%
  sub_missing(1:11,
              missing_text = "--") %>%
  fmt(
    c(8:9, 11),
    fns = function(x)
      ifelse(x < 0.001, "<.001",
             scales::number(x, accuracy = .01))
  ) %>%
  fmt(
    10,
    fns = function (x)
      ifelse(x > 100, ">100",
             scales::number(x, accuracy = .01))
  ) %>%
  tab_style(
    style = list(
      cell_text(weight = "bold")
      ),
    locations = list(cells_body(
     columns = BIC,
     row = BIC == min(BIC[c(1:6)]) # Change this to the number of classes you are eval
    ),
    cells_body(
     columns = aBIC,
     row = aBIC == min(aBIC[1:6])
    ),
    cells_body(
     columns = CAIC,
     row = CAIC == min(CAIC[1:6])
    ),
    cells_body(
     columns = AWE,
```

Model Fit Summary Table[1]

Classes	Par	*LL*	BIC	aBIC	CAIC	AWE	BLRT	VLMR	
1-Class	6	-5,443.41	10,932.50	10,913.44	10,938.50	10,996.19	–	–	
2-Class	13	-5,194.14	10,487.26	10,445.96	10,500.26	10,625.24	<.001	<.001	
3-Class	20	-5,122.48	**10,397.24**	**10,333.70**	**10,417.24**	**10,609.53**	<.001	<.001	>
4-Class	27	-5,111.76	10,429.10	10,343.32	10,456.10	10,715.69	**<.001**	**0.01**	>
5-Class	34	-5,105.59	10,470.07	10,362.04	10,504.06	10,830.95	0.29	0.18	>
6-Class	41	-5,099.88	10,511.95	10,381.69	10,552.95	10,947.14	0.38	0.18	

[1] *Note.* Par = Parameters; *LL* = model log likelihood; BIC = Bayesian information criterion; aBIC = sample size adjusted BIC; CAIC = consistent Akaike information criterion; AWE = approximate weight of evidence criterion; BLRT = bootstrapped likelihood ratio test p-value; VLMR = Vuong-Lo-Mendell-Rubin adjusted likelihood ratio test p-value; *cmPk* = approximate correct model probability.

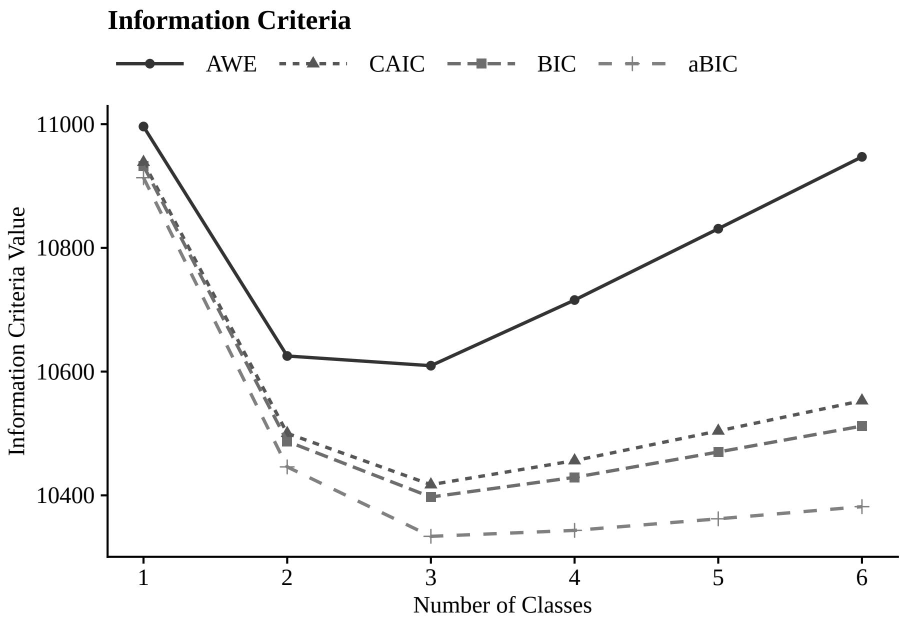
```
     row = AWE == min(AWE[1:6])
    ),
    cells_body(
     columns = cmPk,
     row =  cmPk == max(cmPk[1:6])
     ),
    cells_body(
     columns = BF,
     row =  BF > 10),
    cells_body(
     columns =  T11_VLMR_PValue,
     row =  ifelse(T11_VLMR_PValue < .05 & lead(T11_VLMR_PValue) > .05, T11_VLMR_PValue < .05, NA
    cells_body(
     columns =  BLRT_PValue,
     row =  ifelse(BLRT_PValue < .05 & lead(BLRT_PValue) > .05, BLRT_PValue < .05, NA))
  )
)

fit_table1
```

Save table

```r
gtsave(fit_table1, here("figures", "fit_table.png"))
```

3.6 Information Criteria Plot

```r
allFit %>%
  dplyr::select(2:7) %>%
  rowid_to_column() %>%
  pivot_longer(`BIC`:`AWE`,
               names_to = "Index",
               values_to = "ic_value") %>%
  mutate(Index = factor(Index,
                        levels = c ("AWE", "CAIC", "BIC", "aBIC"))) %>%
  ggplot(aes(
    x = rowid,
    y = ic_value,
    color = Index,
    shape = Index,
    group = Index,
    lty = Index
  )) +
  geom_point(size = 2.0) + geom_line(linewidth = .8) +
  scale_x_continuous(breaks = 1:nrow(allFit)) +
  scale_colour_grey(end = .5) +
  theme_cowplot() +
  labs(x = "Number of Classes", y = "Information Criteria Value", title = "Information
  theme(
    text = element_text(family = "serif", size = 12),
    legend.text = element_text(family="serif", size=12),
    legend.key.width = unit(3, "line"),
    legend.title = element_blank(),
    legend.position = "top"
  )
```



Save figure

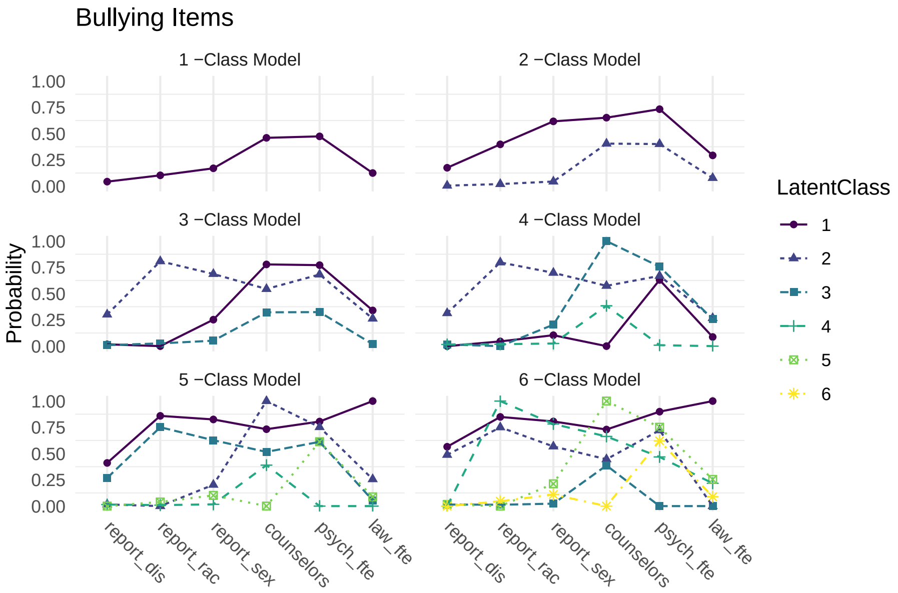
```
ggsave(here("figures", "info_criteria.png"), dpi=300, height=5, width=7, units="in")
```

3.7 Compare Class Solutions

Compare probability plots for $K = 1 : 6$ class solutions

```
model_results <- data.frame()

for (i in 1:length(output_bully)) {

  temp <- output_bully[[i]]$parameters$probability.scale %>%
    mutate(model = paste(i,"-Class Model"))

  model_results <- rbind(model_results, temp)
}
```

```
rm(temp)

compare_plot <-
  model_results %>%
  filter(category == 2) %>%
  dplyr::select(est, model, LatentClass, param) %>%
  mutate(param = as.factor(str_to_lower(param)))

compare_plot$param <- fct_inorder(compare_plot$param)

ggplot(
  compare_plot,
  aes(
    x = param,
    y = est,
    color = LatentClass,
    shape = LatentClass,
    group = LatentClass,
    lty = LatentClass
  )
) +
  geom_point() +
  geom_line() +
  scale_colour_viridis_d() +
  facet_wrap( ~ model, ncol = 2) +
  labs(title = "Bullying Items",
       x = " ", y = "Probability") +
  theme_minimal() +
  theme(panel.grid.major.y = element_blank(),
                          axis.text.x = element_text(angle = -45, hjust = -.1))
```



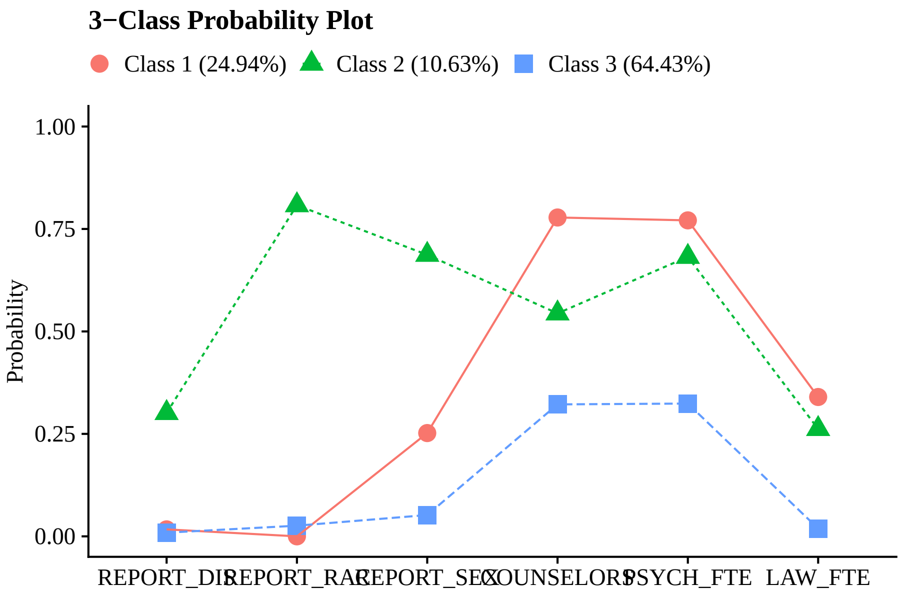
Save figure:

```
ggsave(here("figures", "compare_kclass_plot.png"), dpi=300, height=5, width=7, units="in")
```

3.8 3-Class Probability Plot

Use the `plot_lca` function provided in the folder to plot the item probability plot. This function requires one argument: - `model_name`: The name of the Mplus `readModels` object (e.g., `output_bully$c3_bully.out`)

```
source("plot_lca.txt")

plot_lca(model_name = output_bully$c3_bully.out)
```



Save figure:

```
ggsave(here("figures", "C3_bully_LCA_Plot.png"), dpi="retina", height=5, width=7, units
```

3.9 Observed Response Patterns

Save response frequencies for the 3-class model from the previous lab with `response is _____.dat` under `SAVEDATA`.

```
patterns  <- mplusObject(

  TITLE = "C3 LCA - Save response patterns",

  VARIABLE =
  "categorical = report_dis-law_fte;
   usevar =  report_dis-law_fte;
   classes = c(3);",
  ANALYSIS =
   "estimator = mlr;
    type = mixture;
    starts = 0;
    processors = 10;
    optseed = 802779;",

  SAVEDATA =
   "File=savedata.dat;
    Save=cprob;

    ! Code to save response frequency data

    response is resp_patterns.dat;",

  OUTPUT = "residual patterns tech11 tech14",

  usevariables = colnames(df_bully),
  rdata = df_bully)

patterns_fit <- mplusModeler(patterns,
                dataout=here("mplus", "bully.dat"),
                modelout=here("mplus", "patterns.inp") ,
                check=TRUE, run = TRUE, hashfilename = FALSE)
```

Read in observed response pattern data and relabel the columns

```
# Read in response frequency data that we just created:
patterns <- read_table(here("mplus", "resp_patterns.dat"),
                        col_names=FALSE, na = "*")

# Extract the column names
names <- names(readModels(here("mplus", "patterns.out"))[['savedata']])

# Add the names back to the dataset
colnames(patterns) <- c("Frequency", names)
```

Create a table with the top 5 unconditional response pattern, then top of conditional response pattern for each modal class assignment

```r
# Order responses by highest frequency
order_highest <- patterns %>%
  arrange(desc(Frequency))

# Loop `patterns` data to list top 5 conditional response patterns for each class
loop_cond  <- lapply(1:max(patterns$C), function(k) {
order_cond <- patterns %>%
  filter(C == k) %>%
  arrange(desc(Frequency)) %>%
  head(5)
  })

# Convert loop into data frame
table_data <- as.data.frame(bind_rows(loop_cond))

# Combine unconditional and conditional responses patterns
response_patterns <-  rbind(order_highest[1:5,], table_data)
```

Finally, use `{gt}` to make a nicely formatted table

```r
resp_table <- response_patterns %>%
  gt() %>%
    tab_header(
    title = "Observed Response Patterns",
    subtitle = html("Response patterns, estimated frequencies, estimated posterior clas
    tab_source_note(
    source_note = md("Data Source: **Civil Rights Data Collection (CRDC)**")) %>%
    cols_label(
      Frequency = html("<i>f</i><sub>r</sub>"),
    REPORT_D = "Harrassment: Disability",
    REPORT_R = "Harrassment: Race",
    REPORT_S = "Harrassment: Sex",
    COUNSELO = "Staff: Counselor",
    PSYCH_FT = "Staff: Psychologist",
    LAW_FTE = "Staff: Law Enforcement",
    CPROB1 = html("P<sub><i>k</i></sub>=1"),
    CPROB2 = html("P<sub><i>k</i></sub>=2"),
    CPROB3 = html("P<sub><i>k</i></sub>=3"),
    C = md("*k*")) %>%
  tab_row_group(
    label = "Unconditional response patterns",
    rows = 1:5) %>%
  tab_row_group(
    label = md("*k* = 1 Conditional response patterns"),
    rows = 6:10) %>% #EDIT THESE VALUES BASED ON THE LAST COLUMN
  tab_row_group(
    label = md("*k* = 2 Conditional response patterns"),
    rows = 11:15)  %>% #EDIT THESE VALUES BASED ON THE LAST COLUMN
  tab_row_group(
    label = md("*k* = 3 Conditional response patterns"),
    rows = 16:20)  %>% #EDIT THESE VALUES BASED ON THE LAST COLUMN
    row_group_order(
      groups = c("Unconditional response patterns",
                 md("*k* = 1 Conditional response patterns"),
                 md("*k* = 2 Conditional response patterns"),
                 md("*k* = 3 Conditional response patterns"))) %>%
    tab_footnote(
    footnote = html(
      "<i>Note.</i> <i>f</i><sub>r</sub> = response pattern frequency; P<sub><i>k</i></sub> = pos
    )
  ) %>%
  cols_align(align = "center") %>%
  opt_align_table_header(align = "left") %>%
  gt::tab_options(table.font.names = "Times New Roman")

resp_table
```

Save table:

```
gtsave(resp_table, here("figures","resp_table.png"))
```

3.10 Classification Diagnostics

Use Mplus to calculate k-class confidence intervals (Note: Change the synax to make your chosen *k*-class model):

```
classification  <- mplusObject(

  TITLE = "C3 LCA - Calculated k-Class 95% CI",

  VARIABLE =
    "categorical = report_dis-law_fte;
```

Observed Response Patterns

Response patterns, estimated frequencies, estimated posterior class probabilities and modal assignments

<i>f</i><sub>r</sub>	Harrassment: Disability	Harrassment: Race	Harrassmen
Unconditional response patterns			
525	0	0	0
299	0	0	0
293	0	0	0
251	0	0	0
75	0	0	0
$k = 1$ Conditional response patterns			
251	0	0	0
75	0	0	0
72	0	0	1
38	0	0	1
34	0	0	0
$k = 2$ Conditional response patterns			
24	0	1	0
20	0	1	1
19	0	1	1
18	0	1	1
12	0	1	1
$k = 3$ Conditional response patterns			
525	0	0	0
299	0	0	0
293	0	0	0
36	0	0	1
27	0	0	0

<i>Note.</i> <i>f</i><sub>r</sub> = response pattern frequency; P<sub><i>k</i></sub> = posterior class probabilities Data Source: **Civil Rights Data Collection (CRDC)**

```
   usevar =  report_dis-law_fte;
   classes = c(3);",

  ANALYSIS =
    "estimator = ml;
    type = mixture;
    starts = 0;
    processors = 10;
    optseed = 802779;
    bootstrap = 1000;",

  MODEL =
    "
  !CHANGE THIS SECTION TO YOUR CHOSEN k-CLASS MODEL

  %OVERALL%
  [C#1](c1);

  [C#2](C2);

  Model Constraint:
  New(p1 p2 p3);

  p1 = exp(c1)/(1+exp(c1)+exp(c2));
  p2 = exp(c2)/(1+exp(c1)+exp(c2));
  p3 = 1/(1+exp(c1)+exp(c2));",


  OUTPUT = "cinterval(bcbootstrap)",

  usevariables = colnames(df_bully),
  rdata = df_bully)

classification_fit <- mplusModeler(classification,
                dataout=here("mplus", "bully.dat"),
                modelout=here("mplus", "class.inp") ,
                check=TRUE, run = TRUE, hashfilename = FALSE)
```

Note: Ensure that the classes did not shift during this step (i.e., Class 1 in the enumeration run is now Class 4). Evaluate output and compare the class counts and proportions for the latent classes. Using the OPTSEED function ensures replication of the best loglikelihood value run.

Read in the 3-class model:

```
# Read in the 3-class model and extract information needed
output_bully <- readModels(here("mplus", "class.out"))

# Entropy
entropy <- c(output_bully$summaries$Entropy, rep(NA, output_bully$summaries$NLatentClas

# 95% k-Class and k-class 95% Confidence Intervals
k_ci <- output_bully$parameters$ci.unstandardized %>%
  filter(paramHeader == "New.Additional.Parameters") %>%
  unite(CI, c(low2.5,up2.5), sep=", ", remove = TRUE) %>%
  mutate(CI = paste0("[", CI, "]")) %>%
  rename(kclass=est) %>%
  dplyr::select(kclass, CI)

# AvePPk = Average Latent Class Probabilities for Most Likely Latent Class Membership
avePPk <- tibble(avePPk = diag(output_bully$class_counts$avgProbs.mostLikely))

# mcaPk = modal class assignment proportion
mcaPk <- round(output_bully$class_counts$mostLikely,3) %>%
  mutate(model = paste0("Class ", class)) %>%
  add_column(avePPk, k_ci) %>%
  rename(mcaPk = proportion) %>%
  dplyr::select(model, kclass, CI, mcaPk, avePPk)

# OCCk = odds of correct classification
OCCk <- mcaPk %>%
  mutate(OCCk = round((avePPk/(1-avePPk))/(kclass/(1-kclass)),3))

# Put everything together
class_table <- data.frame(OCCk, entropy)
```

Now, use {gt} to make a nicely formatted table

```
class_table <- class_table %>%
  gt() %>%
    tab_header(
    title = "Model Classification Diagnostics for the 3-Class Solution") %>%
    cols_label(
      model = md("*k*-Class"),
      kclass = md("*k*-Class Proportions"),
      CI = "95% CI",
      mcaPk = html("McaP<sub>k</sub>"),
      avePPk = md("AvePP<sub>k</sub>"),
```

Model Classification Diagnostics for the 3-Class Solution

k-Class	*k*-Class Proportions	95% CI	McaP<sub>k</sub>	AvePPk	OCCk	Entropy
Class 1	0.249	[0.166, 0.329]	0.282	0.675	6.264	0.635
Class 2	0.106	[0.083, 0.136]	0.095	0.904	79.420	
Class 3	0.644	[0.561, 0.731]	0.623	0.893	4.614	

<i>Note.</i> McaP<sub>k</sub> = Modal class assignment proportion; AvePP<sub>k</sub> = Average posterior class probabilities; OCC<sub>k</sub> = Odds of correct classification;

```
      OCCk = md("OCC<sub>k</sub>"),
      entropy = "Entropy") %>%
    sub_missing(7,
              missing_text = "") %>%
    tab_footnote(
    footnote = html(
      "<i>Note.</i> McaP<sub>k</sub> = Modal class assignment proportion; AvePP<sub>k</sub> = Ave
    )
  ) %>%
  cols_align(align = "center") %>%
  opt_align_table_header(align = "left") %>%
  gt::tab_options(table.font.names = "Times New Roman")

class_table
```