Finding thresholds - Latent Class Analysis (LCA)

MM4DBER

Preparation

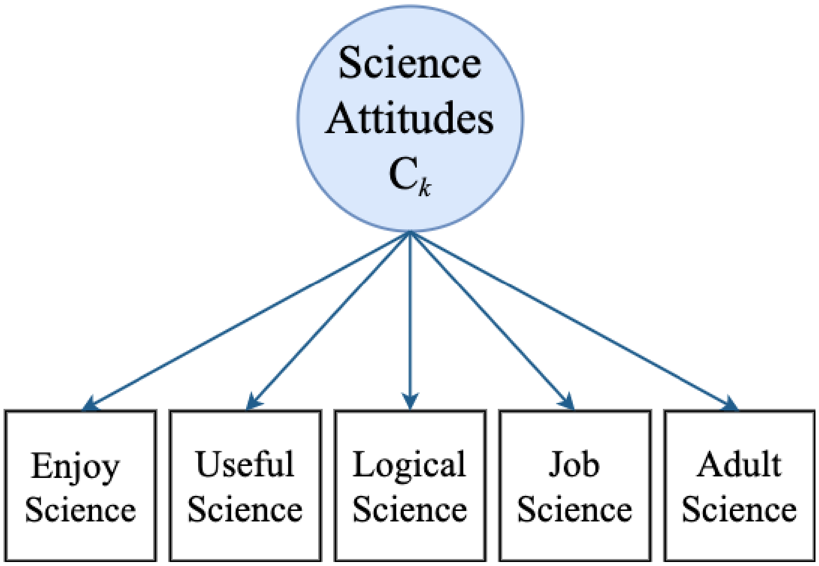
Data source:

Public-use data, *The Longitudinal Survey of American Youth* (**LSAY**): See documentation here

---

Load packages

```
library(tidyverse)
library(haven)
library(glue)
library(MplusAutomation)
library(here)
library(janitor)
library(gt)
library(reshape2)
library(cowplot)
library(poLCA)
library(linguisticsdown)
```



Load data

```
lsay_data <- read_csv("https://garberadamc.github.io/project-site/data/lca_lsay_sci.csv",
                      na = c("9999", "9999.00")) %>%
  clean_names() %>%
  dplyr::select(1:5, Female=female, Enjoy = ab39m, Useful = ab39t,
```

View LCA indicators

Science Attitudes[1]

1

Name	Label	Values
LCA Indicators		
Enjoy	I enjoy science	0 = Disagree, 1 = Agree
Useful	Science useful in everday problems	0 = Disagree, 1 = Agree
Logical	Science helps logical thinkng	0 = Disagree, 1 = Agree
Job	Need science for a good job	0 = Disagree, 1 = Agree
Adult	Will use science often as an adult	0 = Disagree, 1 = Agree
Covariate		
Female	Reported gender	0 = Male, 1 = Female

[1] Longitudinal Study of American Youth

2.1 Enumeration:

Estimate K-class models with 1 through 6 classes.

Run enumeration using `mplusObject` method

```
lca_k1_6  <- lapply(1:6, function(k) {

  lca_enum  <- mplusObject(

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

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

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

  OUTPUT = "sampstat residual tech11 tech14;",

  PLOT =
    "type = plot3;
    series = Enjoy-Adult(*);",

  usevariables = colnames(lsay_data),
  rdata = lsay_data)

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

Plot Final Model - Conditional Item Probability Plot

This syntax creates a function called `plot_lca_function` that requires 7 arguments (inputs):

- `model_name`: name of Mplus model object (e.g., `model_step1`)
- `item_num`: the number of items in LCA measurement model (e.g., `5`)
- `class_num`: the number of classes (k) in LCA model (e.g., `3`)
- `item_labels`: the item labels for x-axis (e.g., `c("Enjoy","Useful","Logical","Job","Adult")`)
- `class_labels`: the class label names (e.g., `c("Adaptive Coping","Externalizing Behavior","No Coping")`)
- `class_legend_order` = change the order that class names are listed in the plot legend (e.g., `c(2,1,3)`)

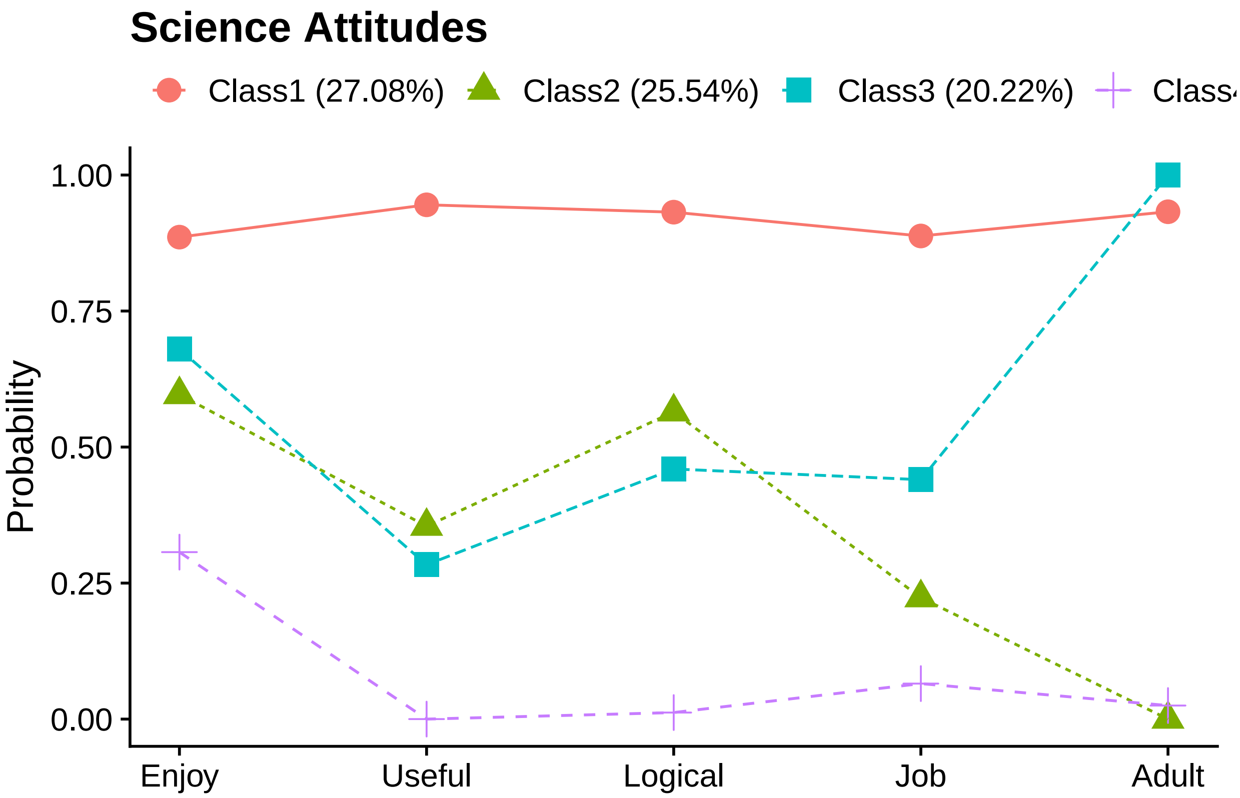
- `plot_title`: include the title of the plot here (e.g., `"LCA Posterior Probability Plot"`)

```r
plot_lca_function <- function(model_name,item_num,class_num,item_labels,
                              class_labels,class_legend_order,plot_title){

mplus_model <- as.data.frame(model_name$gh5$means_and_variances_data$estimated_probs$values)
plot_data <- mplus_model[seq(2, 2*item_num, 2),]

c_size <- as.data.frame(model_name$class_counts$modelEstimated$proportion)
colnames(c_size) <- paste0("cs")
c_size <- c_size %>% mutate(cs = round(cs*100, 2))
colnames(plot_data) <- paste0(class_labels, glue(" ({c_size[1:class_num,]}%)"))
plot_data <- plot_data %>% relocate(class_legend_order)

plot_data <- cbind(Var = paste0("U", 1:item_num), plot_data)
plot_data$Var <- factor(plot_data$Var,
               labels = item_labels)
plot_data$Var <- fct_inorder(plot_data$Var)

pd_long_data <- melt(plot_data, id.vars = "Var")

# This syntax uses the data.frame created above to produce the plot with `ggplot()`

p <- pd_long_data %>%
  ggplot(aes(x = as.integer(Var), y = value,
  shape = variable, colour = variable, lty = variable)) +
  geom_point(size = 4) + geom_line() +
  scale_x_continuous("", breaks = 1:item_num, labels = plot_data$Var) +
  labs(title = plot_title, y = "Probability") +
  theme_cowplot() +
  theme(legend.title = element_blank(),
        legend.position = "top")

p
return(p)
}
```

```r
model_c4 <- readModels(here("enum_lsay", "c4_lca.out"), quiet = TRUE)
```

Run the `plot_lca_function` by specifying each input (***Figure 1***)

```r
plot_lca_function(
  model_name = model_c4,
  item_num = 5,
  class_num = 4,
  item_labels = c("Enjoy", "Useful", "Logical", "Job", "Adult"),
  class_labels = c("Class1","Class2","Class3","Class4"),
  class_legend_order = c(1,2,3,4),
  plot_title = "Science Attitudes"
  )
```



```
ggsave(here("figures","C4_LSAY_Plot.png"), dpi=300, height=4, width=6, units="in")
```

Run model with covariate

```
  lca_enum  <- mplusObject(

    TITLE = "Class 4 model with covariate (female)",

    VARIABLE =
    "categorical = Enjoy-Adult;
     usevar = Enjoy-Adult;

     classes = c(4);
     auxiliary = Female (r3step);",

  ANALYSIS =
   "estimator = mlr;
    type = mixture;
    starts = 500 200;
    processors = 10;",

  OUTPUT = "sampstat residual tech11 tech14;",

  PLOT =
    "type = plot3;
    series = Enjoy-Adult(*);",

  usevariables = colnames(lsay_data),
  rdata = lsay_data)

lca_enum_fit <- mplusModeler(lca_enum,
                            dataout=here("cov_c4", "lca_lsay.dat"),
                            modelout=here("cov_c4", "c4_lca_cov.inp") ,
                            check=TRUE, run = TRUE, hashfilename = FALSE)
```