Variable-Specific Entropy Contribution

Tihomir Asparouhov and Bengt Muthén

June 19, 2018

In latent class analysis it is useful to evaluate a measurement instrument in terms of how well it identifies the latent classes. This is typically done by computing the entropy

$$E = 1 + \frac{1}{N \log(k)} \left(\sum_{i=1}^{N} \sum_{k=1}^{K} P(C = k|U_i) \log(P(C = k|U_i)) \right)$$

where C is the latent variable, K is the number of classes, N is the sample size and U_i is the vector of all latent class indicator variables and the probabilities $P(C = k|U_i)$ are computed from the estimated model. The larger the entropy is the more clear the latent class identification is. The entropy value is between 0 and 1. Entropy with values approaching 1 indicate clear separation of the classes.

The entropy evaluates the quality of the measurement instrument as a whole. Often there is a need to evaluate the quality of individual items. Here we define the univariate entropy which can be computed for each latent class indicator U_j

$$E_j = 1 + \frac{1}{N \log(k)} \left(\sum_{i=1}^{N} \sum_{k=1}^{K} P(C = k|U_{ij}) \log(P(C = k|U_{ij})) \right)$$

where $P(C = k|U_{ij})$ is also computed from the estimated model. The univariate entropies E_j are directly comparable among each other. The higher E_j is the more informative the indicator U_j is in identifying the latent classes. Latent class indicators with univariate entropies near 0 can probably be removed from the model as they do not provide much information about the latent class variable. On the other hand latent class indicators with large univariate entropies can be considered good indicators. Thus the univariate entropy can be used in evaluating specific items within a latent class measurement instrument. Note here that the univariate entropy is not additive, i.e., the total entropy for an LCA model is not the sum of the univariate entropies for all of the items.

We illustrate the univariate entropy using example 7.3 from the user's guide but we treat all of the variables in the data file as class indicators. In order to obtain the univariate entropy the option **ENTROPY** has to be added to the **OUTPUT** command.

```
Univariate Entropy
   U1            U2            U3            U4            X1
   ____          ____          ____          ____          ____
  0.553         0.545         0.577         0.567         0.909

Univariate Entropy
   X2            X3            X4            X5            X6
   ____          ____          ____          ____          ____
  0.935         0.535         0.526         0.167         0.164

Univariate Entropy
   X7            X8            X9            X10
   ____          ____          ____          ____
  0.501         0.561         0.913         0.934
```

From this output one can conclude that the best class indicators are X1, X2, X9 and X10 while the worst class indicators are X5 and X6. This result is accurate. The data generation is described in mcex7.3.inp file in the Mplus installation directory. The variables X5 and X6 are independent of the latent class variable and carry no information about it, while the variables X1, X2, X9 and X10 have a mean of -2 in one of the classes and 2 in the other and thus are highly correlated with the latent class variable.

In simple 2 or 3 class latent variable models it could be easy to evaluate an indicator simply by considering the estimated parameters, however, in more complex models this might be hard. The univariate entropy value gives a new and simple method for evaluating the quality of a latent class indicator. In Mplus Version 7.3 this feature is available for continuous and categorical indicators.

The univariate entropy has the advantage over other indicator specific tests because it is directly comparable between indicators, due to the fact that values are all on the same entropy scale. A disadvantage of the univariate entropy is that it doesn't provide statistical significance. For completeness, we will review here the indicator specific tests that can be used as an alternative to the univariate entropy and can also provide statistical significance. The three methods are: the model constraint command (Z-test), the model test command (Wald test), and the LRT (likelihood ratio test) test.

If there are only two classes in the model and each indicator has two parameters, the **model constraint** command can be used to construct indicator specific tests by computing the differences of the class specific parameters for each indicator. Using the same example as the one considered earlier, the model constraint command is specified as follows

```
DATA: FILE IS ex7.3.dat;
VARIABLE: NAMES ARE u1-u4 x1-x10;
          CLASSES = c (2);
          CATEGORICAL = u1-u4;
ANALYSIS: TYPE = MIXTURE;

model:
%overall%
%C#1%
[u1$1-u4$1] (m1-m4); [x1-x10] (m5-m14);
%C#2%
[u1$1-u4$1] (mm1-mm4); [x1-x10] (mm5-mm14);

model constraint: new(d1-d14); do(1,14) d#=m#-mm#;
```

The results for this example are as follows

```
New/Additional Parameters
   D1                -4.082      0.319    -12.814      0.000
   D2                -3.994      0.308    -12.964      0.000
   D3                 4.296      0.336     12.796      0.000
   D4                 4.254      0.341     12.465      0.000
   D5                 4.038      0.107     37.915      0.000
   D6                 4.029      0.100     40.098      0.000
   D7                 1.971      0.097     20.311      0.000
   D8                 1.829      0.095     19.349      0.000
   D9                 0.135      0.108      1.246      0.213
   D10                0.026      0.100      0.263      0.793
   D11               -1.781      0.107    -16.624      0.000
   D12               -1.982      0.094    -21.132      0.000
   D13               -3.824      0.102    -37.594      0.000
   D14               -4.129      0.097    -42.554      0.000
```

These results indicate that the univariate entropy is in agreement with the Z-

score(third column) in these difference tests.

If there are more than two parameter per indicator, such as for example, more than 2 classes, or the case when not only the means but also the variances are class specific, it is necessary to use the command **Model Test**. Using the same example as before the following input file illustrates how this is done for the variable X1 when both the means and the variances are class specific.

```
DATA: FILE IS ex7.3.dat;
VARIABLE: NAMES ARE u1-u4 x1-x10;
          CLASSES = c (2);
          CATEGORICAL = u1-u4;
ANALYSIS: TYPE = MIXTURE;
model:
%overall%
%C#1%
[u1$1-u4$1] (m1-m4); [x1-x10] (m5-m14); x1-x10 (v5-v14);
%C#2%
[u1$1-u4$1] (mm1-mm4); [x1-x10] (mm5-mm14); x1-x10 (vv5-vv14);

model test:
0=m5-mm5;
0=v5-vv5;
```

The results are given in the following section of the output file

```
Wald Test of Parameter Constraints

          Value                          1453.144
          Degrees of Freedom                2
          P-Value                         0.0000
```

The **Model Test** command can be used for one variable at a time. The key point here is that model test is specified to test the equality across class for all parameters related to the indicator variable.

The LRT test can also be used to test the indicator specific effect on the class formation. We illustrate this for the X1 indicator in the above example. We estimate the model first without any constraints as follows

```
DATA: FILE IS ex7.3.dat;
VARIABLE: NAMES ARE u1-u4 x1-x10;
          CLASSES = c (2);
          CATEGORICAL = u1-u4;
ANALYSIS: TYPE = MIXTURE;
model:
%overall%
%C#1%
[u1$1-u4$1]; [x1-x10];
%C#2%
[u1$1-u4$1]; [x1-x10];
```

We then estimate the LCA model without the indicator X1 by holding equal across class the parameters of X1. That way, X1 will not contribute any information to the class formation.

```
DATA: FILE IS ex7.3.dat;
VARIABLE: NAMES ARE u1-u4 x1-x10;
          CLASSES = c (2);
          CATEGORICAL = u1-u4;
ANALYSIS: TYPE = MIXTURE;
model:
%overall%
%C#1%
[u1$1-u4$1]; [x1] (1); [x2-x10];
%C#2%
[u1$1-u4$1]; [x1] (1); [x2-x10];
```

The log-likelihood for the first model is -8071.281, and for the second it is -8408.071. We can then compute the LRT test as 2(-8071.281 - (-8408.071))=673.580 which is a significant result (DF=1 in this case). Note that in the second run it is important to include X1 in the model with class invariant parameters, rather than exclude it from the model completely, becasue that way the above two models are nested and can be used to obtain the LRT test. With the LRT test this process is performed for each variable separately.