The package decision-table*

S. Vandevelde, F. Pantigny
s.vandevelde@kuleuven.be
fpantigny@wanadoo.fr

October 1, 2021

Abstract

The LaTeX package decision-table provides a command `\dmntable`, which allows for an easy way to generate decision tables in the Decision Model and Notation (DMN) format. (See Fig. 1) This package ensures consistency in the tables (i.e. fontsize), and is thus a better alternative to inserting tables via images. Besides pure DMN tables, it also supports the tables used in the cDMN and pDMN extensions, namely glossary and probability tables.

1 Description

The `decision-table` package allows for easy and straightforward generation of decision tables (with or without input columns) in the Decision Model and Notation (DMN) format, as demonstrated in Fig. 1. Additionally, it also supports glossary and goal tables as defined by cDMN[1], and probability tables as defined by pDMN[2]. Usage of this package ensures consistency in tables (i.e. font type, font size, color, ...), and is thus a better alternative to inserting tables via images.

BMI Level		
U	BMI	BMILevel
1	< 18.5	Underweight
2	[18.5..25]	Normal
3	(25..30]	Overweight
4	> 30	Obese

Figure 1: Example of a DMN table

2 Usage

In total, the `decision-table` package adds 5 new commands.

- `dmntable`: basic DMN table.
- `dmnoutputtable`: basic DMN table, but without input columns.
- `dmnglossarytable`: glossary table, as defined by cDMN.
- `goaltable`: goal table, as defined by cDMN.
- `pdmntable`: probability table, as defined by pDMN.
- `pdmnoutputtable`: probability table, but without input columns.

*This document corresponds to the version 0.0.4 of decision-table, at the date of 2021/10/01.
[1]https://cdmn.be
[2]Vandevelde S., et al., "A Table-Based Representation for Probabilistic Logic: Preliminary Result".

2.1 dmntable

The `dmntable` command generates standard DMN decision tables. This command expands into a `tabular`, so it can be used within a `table` or `figure` environment. Furthermore, this allows labels and captions to be added seamlessly, and allows placing multiple DMN tables in the same environment. These tables will automatically arrange themselves based on size.
The `dmntable` command has the following inputs:

- title
- hit policy
- input column headers
- output column headers
- the table values

The command is used as follows:

```
\dmntable{title}{hitpolicy}{input}{output}{values}
```

The input, output and cell values are split by a comma. It is not necessary to include the row numbers for the cell values. For example, Fig. 1 is generated by the following code:

```
\begin{figure}[H]
    \centering
    \dmntable{BMI Level}{U}{BMI}{BMILevel}
             {$< 18.5$, Underweight,
              $[18.5..25]$, Normal,
              $(25..30]$, Overweight,
              $> 30$, Obese}
    \caption{Example of a DMN table}
    \label{ex1}
\end{figure}
```

If a cell value contains multiple values (e.g. multiple string values), then accolades should be written around them. See the example shown in Fig. **??**.
It is also possible to have cells be multiline using the `makecell` package. This is useful in cases where large text causes page overflow.

```
\include{makecell} % This should be somewhere at the top of document
\begin{figure}[H]
    \centering
    \dmntable{BMI Level}{U}{BMI}{\makecell{BMI \\ Level}}
             {$< 18.5$, Underweight,
              $[18.5..25]$, Normal,
              $(25..30]$, Overweight,
              $> 30$, Obese}
    \caption{Example of a DMN table with multiline cell}
    \label{ex1}
\end{figure}
```

2.2 dmnoutputtable

Decision tables that have no input columns can be created using the `dmnoutputtable` command. Its usage is the same as the standard decision table, except that no input columns should be given. For example, the following code generates the table shown in Fig. 2.

```
\begin{figure}[H]
    \centering
    \dmnoutputtable{BMI Level}{U}{BMI}{5}
    \caption{Example of a DMN output table}
    \label{ex1}
\end{figure}
```

BMI Level	
U	BMI
1	5

Figure 2: Example of a DMN output table

2.3 dmnglossarytable

Glossary tables are added by using the `glossarytable` command. This command accepts three inputs: the title, the sub-titles of the columns, and the actual elements. See Fig. 3 for an example.

```
\begin{figure}[H]
    \centering
    \glossarytable{Type}{Name, DataType, Possible Values}
                        {Country, String, {Belgium, France, Germany}}
    \glossarytable{Function}{Name, DataType}{color of Country, Color}
    \glossarytable{Relation}{Name}{Country borders Country}
    \caption{Example glossary}
    \label{glos1}
\end{figure}
```

Type		
Name	**DataType**	**Possible Values**
Country	String	Belgium, France, Germany

Function	
Name	**DataType**
color of Country	Color

Relation
Name
Country borders Country

Figure 3: Example glossary

2.4 goaltable

Generating a `Goal` table is straightforward:

```
\goaltable{tablename}{values}
```

tablename
value1
value2

2.5 pdmntable

pDMN probability tables are similar to standard DMN tables, but with a few changes. The biggest change is the addition of probabilities. The syntax of `pdmntable` is as follows:

```
\pdmntable{title}{hitpolicy}{inputs}{outputs}{output values}{cell values}
```

For example, a table expressing the probabilities of an unbiased and biased dice throw can be expressed as follows:

```
\pdmntable{Throwing Dice}{Ch}{biased}{die value}
          {one, two, three, four, five, six}
          {No, $1/6$, $1/6$, $1/6$, $1/6$, $1/6$, $1/6$,
           Yes, $0.1$, $0.1$, $0.1$, $0.1$, $0.1$, $0.5$}
```

Throwing Dice							
Ch	biased	die value					
		one	two	three	four	five	six
1	No	1/6	1/6	1/6	1/6	1/6	1/6
2	Yes	0.1	0.1	0.1	0.1	0.1	0.5

2.6 pdmnoutputtable

The final type of table, `pdmnoutputtable`, is the same as the one shown in the previous subsection but without any input columns.

```
\pdmnoutputtable{tablename}{hitpolicy}{output}{outputvalue}{probabilities}
```

h1	
U	heads1
	Yes
1	0.5

3 Contributing

Contributions are always welcome. The project is hosted at https://gitlab.com/Vadevesi/dmn-tex

4 Implementation

<@@=dmn>

We give the traditional declaration of a package written with expl3:

```
\RequirePackage{l3keys2e}
\ProvidesExplPackage
  {decision-table}
  {\dmnfiledate}
  {\dmnfileversion}
  {Table of decision}
\RequirePackage { nicematrix }
```

We define the command `\dmntable` with the tools of xparse.

```
\NewDocumentCommand \dmntable { m m m m m }
  {
```

The clist (*comma separated list*) `\l_@@_input_clist` is for the list of the names of the input fields.

```
    \clist_clear_new:N \l_@@_input_clist
    \clist_set:Nn \l_@@_input_clist { #3 }
```

The clist `\l_@@_output_clist` is for the list of the names of the output fields.

```
    \clist_clear_new:N \l_@@_output_clist
    \clist_set:Nn \l_@@_output_clist { #4 }
```

The integer `\l_@@_input_int` is the number of the input fields.

```
    \int_zero_new:N \l_@@_input_int
    \int_set:Nn \l_@@_input_int { \clist_count:N \l_@@_input_clist }
```

The integer `\l_@@_output_int` is the number of the output fields.

```
    \int_zero_new:N \l_@@_output_int
    \int_set:Nn \l_@@_output_int { \clist_count:N \l_@@_output_clist }
```

The sequence `\l_@@_cells_seq` is the sequence of all the cells of the "body" of the tabular.

```
    \seq_clear_new:N \l_@@_cells_seq
    \seq_set_split:Nnn \l_@@_cells_seq { , } { #5 }
```

Now, we will begin the construction of the tabular (a `{NiceTabular}` of nicematrix).

The command `\use:x` will expand its argument. Indeed, the preamble of the `{NiceTabular}` (which has the same format as a preamble of `{tabular}` must be computed before the execution of the `\begin{NiceTabular}`.

```
    \use:x
      {
        \exp_not:N \begin { NiceTabular }
```

Here is the preamble of the tabular. The command `\prg_replicate:nn` is expandable and hence will be expanded by the `use:x`.

```
        { r \prg_replicate:nn { \l_@@_input_int + \l_@@_output_int } l }
```

Here is the list of options of the `{NiceTabular}` (a standard tabular of `{array}` don't have such list of options. Once again, we have to compute some quantities in this list of options before the execution of `\begin{NiceTabular}`.

```
        [
```

The key `hvlines-except-corners` will draw all the rules of the tabular, excepted in the (upper right) corner.

```
          hvlines-except-corners ,
```

The key `code-before` of `{NiceTabular}` contains instructions to color the cells *before* the rules (doing so, the resulting PDF gives better results in the PDF viewers).

```
          code-before =
```

First, a `\rectanglecolor` for the labels of the "input" fields. The command `\int_eval:n` is expandable and, hence, will be expanded by the `\use:x`. On the other side, we have to prevent the expansion of `\rectanglecolor` which, in fact, at that point is not defined (it will be defined by nicematrix after the construction of the array).

```
            \exp_not:N \rectanglecolor
              { blue!10!green!60!black!30 }
              { 2 - 2 }
              { 2 - \int_eval:n { \l_@@_input_int + 1 } }
```

A `\rectanglecolor` for the labels of the "output" fields.

```
            \exp_not:N \rectanglecolor
              { green!30!blue!15 }
              { 2 - \int_eval:n { \l_@@_input_int + 2 } }
              { 2 - \int_eval:n { \l_@@_input_int + \l_@@_output_int + 1 } }
        ]
      }
```

Now, we begin the body of the tabular (the environment `{NiceTabular}`).

The body begins by a `\multicolumn` for the title. However, we have to compute the number of cells of that `\multicolumn`. That's why we have to expand the first argument of the `\multicolumn` before executing the `\multicolumn`. However, we have to do that in an expandable way in order to prevent the functionnality of the `\multicolumn` (which internally give a `\omit` of TeX). That's why we have to use `\exp_args:Ne` (`\exp_args:Nx` would not do the job).

```
    \exp_args:Ne \multicolumn
      { \int_eval:n { \l_@@_input_int + 1 } }
      { l }
      { #1 } \\
```

`#2` is the *hit policy*.

```
    #2 &
```

Now, the fields ("input fields" and "output fields"). By using `\clist_use:Nn`, we replace the commas by ampersands (`&`).

```
    \clist_use:Nn \l_@@_input_clist { & } &
    \clist_use:Nn \l_@@_output_clist { & } \\
```

Now, all the rows corresponding to the rules. We begin a loop over all the cells with `\seq_map_inline:Nn`.

```
    1 &
    \seq_map_inline:Nn \l_@@_cells_seq
      {
```

`\c@jCol` and `\c@iRow` are counters provided by `{NiceTabular}` for the current column and the current row. If you are in the first column, we insert the number of rule.

```
        \int_compare:nT { \c@jCol = 0 } { \int_eval:n { \c@iRow - 1 } & }
```

Now, we add one composante of `\l_@@_cells_seq`.

```
        ##1
```

Before the following cell, we have, of course, to add `\\` (if we are at the end of the row) or `&` (elsewhere).

```
        \int_compare:nTF { \c@jCol = \l_@@_input_int + \l_@@_output_int + 1 }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }
```

The other two commands are simply more of the same.

```
\NewDocumentCommand \dmnoutputtable { m m m m }
  {
    \clist_clear_new:N \l__dmn_output_clist
    \clist_set:Nn \l__dmn_output_clist { #3 }
    \int_zero_new:N \l__dmn_output_int
    \int_set:Nn \l__dmn_output_int { \clist_count:N \l__dmn_output_clist }
    \seq_clear_new:N \l__dmn_cells_seq
    \seq_set_split:Nnn \l__dmn_cells_seq { , } { #4 }
    \use:x
      {
        \exp_not:N \begin { NiceTabular }
        { r \prg_replicate:nn { \l__dmn_output_int } l }
        [
          hvlines-except-corners ,
          code-before =
           \exp_not:N \rectanglecolor
             { blue!10!green!60!black!30 }
             { 2 - 2 }
             { 2 - \int_eval:n { 1 } }
           \exp_not:N \rectanglecolor
             { green!30!blue!15 }
             { 2 - \int_eval:n { 2 } }
             { 2 - \int_eval:n { \l__dmn_output_int + 1 } }
        ]
      }
    \exp_args:Ne \multicolumn
      { \int_eval:n { 1 } }
      { l }
      { #1 } \\
    #2 &
    \clist_use:Nn \l__dmn_output_clist { & } \\
```

```
    1 &
    \seq_map_inline:Nn \l__dmn_cells_seq
      {
        \int_compare:nT { \c@jCol = 0 } { \int_eval:n { \c@iRow - 1 } & }
        ##1
        \int_compare:nTF { \c@jCol = \l__dmn_output_int + 1 }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }

\NewDocumentCommand \glossarytable { m m m }
  {
    \clist_clear_new:N \l__clist
    \clist_set:Nn \l__clist { #2 }
    \int_zero_new:N \l__cint
    \int_set:Nn \l__cint { \clist_count:N \l__clist }
    \seq_clear_new:N \l__dmn_cells_seq
    \seq_set_split:Nnn \l__dmn_cells_seq { , } { #3 }
    \use:x
      {
        \exp_not:N \begin { NiceTabular }
        { c \prg_replicate:nn { \l__cint -1 } c }
        [
          hvlines-except-corners ,
          code-before =
           \exp_not:N \rectanglecolor
             { blue!30!green!10!red!40 }
             { 1 - \int_eval:n { 1 } }
             { 1 - \int_eval:n { \l__cint} }
        ]
      }
    \exp_args:Ne \multicolumn
      { \int_eval:n { \l__cint } }
      { c }
      { \textbf{#1} } \\
    \bf \clist_use:Nn \l__clist { & \bf } \\
    \seq_map_inline:Nn \l__dmn_cells_seq
      {
        \int_compare:nT { \c@jCol = 0 } { }
        ##1
        \int_compare:nTF { \c@jCol = \l__cint }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }
\NewDocumentCommand \pdmntable { m m m m m m}
  {
    \clist_clear_new:N \l__dmn_input_clist
    \clist_set:Nn \l__dmn_input_clist { #3 }
    \clist_clear_new:N \l__dmn_output_clist
```

```
\clist_set:Nn \l__dmn_output_clist { #4 }
\clist_clear_new:N \l__suboutput_clist
\clist_set:Nn \l__dmn_suboutput_clist { #5 }
\int_zero_new:N \l__dmn_input_int
\int_set:Nn \l__dmn_input_int { \clist_count:N \l__dmn_input_clist }
\int_zero_new:N \l__dmn_output_int
\int_set:Nn \l__dmn_output_int { \clist_count:N \l__dmn_output_clist }
\int_zero_new:N \l__dmn_suboutput_int
\int_set:Nn \l__dmn_suboutput_int { \clist_count:N \l__dmn_suboutput_clist }
\seq_clear_new:N \l__dmn_cells_seq
\seq_set_split:Nnn \l__dmn_cells_seq { , } { #6 }
\seq_clear_new:N \l__dmn_inputcells_seq
\seq_set_split:Nnn \l__dmn_inputcells_seq { , } { #3 }
\use:x
  {
    \exp_not:N \begin { NiceTabular }
    { r \prg_replicate:nn { \l__dmn_input_int + \l__dmn_suboutput_int  + 1} l }
    [
      hvlines-except-corners ,
      code-before =
       \exp_not:N \rectanglecolor
         { blue!10!green!60!black!30 }
         { 2 - 2 }
         { 2 - \int_eval:n { \l__dmn_input_int + 1 } }
       \exp_not:N \rectanglecolor
         { green!30!blue!15 }
         { 2 - \int_eval:n { \l__dmn_input_int + 2 } }
         { 2 - \int_eval:n { \l__dmn_input_int + \l__dmn_suboutput_int + 1 } }
      \exp_not:N \rectanglecolor
         { red!60!green!60!blue!15}
         { 3 - \int_eval:n { 1 } }
         { 3 - \int_eval:n { \l__dmn_input_int + \l__dmn_suboutput_int + 1 } }
    ]
  }
\exp_args:Ne \multicolumn
  { \int_eval:n { \l__dmn_input_int + 1 } }
  { l }
  { #1 } \\
#2 &
\clist_use:Nn \l__dmn_input_clist { & } &
\exp_args:Ne \multicolumn
  { \int_eval:n {\l__dmn_suboutput_int}}
  { c }
  { #4 } \\
\seq_map_inline:Nn \l__dmn_inputcells_seq
  { & }
& \clist_use:Nn \l__dmn_suboutput_clist { & } & \\
%\clist_use:Nn \l__dmn_output_clist { & } \\
1 &
\seq_map_inline:Nn \l__dmn_cells_seq
  {
    \int_compare:nT { \c@jCol = 0 } { \int_eval:n { \c@iRow - 2 } & }
    ##1
```

```
        \int_compare:nTF { \c@jCol = \l__dmn_input_int + \l__dmn_suboutput_int + 1 }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }
\NewDocumentCommand \pdmnoutputtable { m m m m m }
  {
    \clist_clear_new:N \l__dmn_output_clist
    \clist_set:Nn \l__dmn_output_clist { #3 }
    \clist_clear_new:N \l__suboutput_clist
    \clist_set:Nn \l__dmn_suboutput_clist { #4 }
    \int_zero_new:N \l__dmn_output_int
    \int_set:Nn \l__dmn_output_int { \clist_count:N \l__dmn_output_clist }
    \int_zero_new:N \l__dmn_suboutput_int
    \int_set:Nn \l__dmn_suboutput_int { \clist_count:N \l__dmn_suboutput_clist }
    \seq_clear_new:N \l__dmn_cells_seq
    \seq_set_split:Nnn \l__dmn_cells_seq { , } { #5 }
    \use:x
      {
        \exp_not:N \begin { NiceTabular }
        { r \prg_replicate:nn { \l__dmn_suboutput_int } l }
        [
          hvlines-except-corners ,
          code-before =
           \exp_not:N \rectanglecolor
             { blue!10!green!60!black!30 }
             { 2 - 2 }
             { 2 - \int_eval:n { 1 } }
           \exp_not:N \rectanglecolor
             { green!30!blue!15 }
             { 2 - \int_eval:n { 2 } }
             { 2 - \int_eval:n { \l__dmn_suboutput_int + 1 } }
          \exp_not:N \rectanglecolor
             { red!60!green!60!blue!15}
             { 3 - \int_eval:n { 1 } }
             { 3 - \int_eval:n { \l__dmn_suboutput_int + 1 } }
        ]
      }
    \exp_args:Ne \multicolumn
      { \int_eval:n { 1 } }
      { l }
      { #1 } \\
    #2 &
    \exp_args:Ne \multicolumn
      { \int_eval:n {\l__dmn_suboutput_int}}
      { c }
      { #3 } \\
    & \clist_use:Nn \l__dmn_suboutput_clist { & } \\
    1 &
    \seq_map_inline:Nn \l__dmn_cells_seq
      {
        \int_compare:nT { \c@jCol = 0 } { \int_eval:n { \c@iRow - 1 } & }
```

```
        ##1
        \int_compare:nTF { \c@jCol = \l__dmn_suboutput_int + 1 }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }

\NewDocumentCommand \goaltable { m m }
  {
    \seq_clear_new:N \l__dmn_cells_seq
    \seq_set_split:Nnn \l__dmn_cells_seq { , } { #2 }
    \use:x
      {
        \exp_not:N \begin { NiceTabular }
        { \prg_replicate:nn { 1 } c }
        [
          hvlines-except-corners ,
          code-before =
           \exp_not:N \rectanglecolor
             { blue!30!green!10!red!20 }
             { 1 - \int_eval:n { 1 } }
             { 1 - \int_eval:n { 1} }
        ]
      }
    \textbf{#1} \\
    %\exp_args:Ne \multicolumn
    %  { \int_eval:n { 1 } }
    %  { c }
    %  { \textbf{#1} } \\
    \seq_map_inline:Nn \l__dmn_cells_seq
      {
        \int_compare:nT { \c@jCol = 0 } { }
        ##1
        \int_compare:nTF { \c@jCol = 1 }
          { \\ }
          { & }
      }
    \end { NiceTabular }
  }

\endinput
%%
%% End of file `decision-table.sty'.
```

Contents