Excel_RDR

Three Ripple-Down Rule demonstrators in Excel

Quick Start Guide

This distribution: The zip file contains Excel_SCRDR.xlsm, Excel_MCRDR.xlsm and Excel_GRDR.xlsm Each file is preloaded with the zoo dataset. A few other UCIrvine data sets in Excel format are included and this readme file

Before Opening: Excel RDR uses the Personal Macro Workbook and this must be open. This program writes its own VBA code, so you if are using Windows Excel you first have to change Excel trust setting to allow access to the VBA project object model and to allow macros to run. See below on opening the Personal Macro Workbook and changing Excel trust settings

Opening: On opening one of the programs program on a Mac you will be asked to enable macros. For a new knowledge base you will be asked indicate the number of attributes a case has, but this can be changed at any stage.

Cases worksheet: Contains the cases, one case per row, for which rules will be written. Any edit changes including changing attribute names, adding and deleting columns should be propagated to other sheets. You should be prompted that the name of the last column is *conclusion* but check just in case before leaving this sheet. Any insertion or deletion of columns or changing headers should be done on this sheet to be propagated to the other sheets

Double-clicking on a row (case) shows the user options. When cases are run after choosing one of the run option, If the second last column is named *target,* rows where the inference conclusion differs from the target will be coloured red.

Rule builder worksheet: To build a rule access this sheet by double clicking a case on the case sheet and selecting add rule. The *operator* and *value* fields can be edited to construct rule conditions for the case. Standard operators are: *is, contains*, and the arithmetic operators, =, => *etc*. User-defined operators can also be used as described below. Some sample user-defined functions are contained in the user funcs module in the Personal Workbook. Double-clicking a case shows its rule-trace.

Rules worksheet: Shows all rules plus the rule trace for the case in the top panel. This case can be edited to see how the rule trace changes. Double-clicking a rule will put the rule's cornerstone case in the top panel and show the trace.

Cornerstone cases: Double-Clicking will show the rule-trace for that cornerstone case

Statistics: gives some simple statistics not easily accessible otherwise

Bugs

If you have more than one Excel_RDR workbook open at time, you may run into problems closing a workbook. To fix, delete reference to RDR from the Personal Workbook references – even better: open one Excel_RDR workbook at a time. Unknown bugs: there are probably many

Introduction

Introduction to Ripple-Down Rules

RDR is an approach to building knowledge-based or expert systems, where a knowledge-based system is put into use processing cases without any rules or with minimal priming rules. The output is monitored and a rule is added to provide the correct output whenever the output is incorrect. In adding a rule, the user might be prompted to specify conditions that distinguish this case from previous cases, which had a different conclusion, but which will be evaluated by the new rule. Previous rules are never edited, retracted or deleted, all knowledge base maintenance and editing is managed by the addition of new correction rules. The RDR system determines where and how new rules are added into the knowledge base, rather than requiring rule builder (or knowledge engineer) decisions. This approach allows for extremely rapid and simple knowledge acquisition that can be carried out by domain end-users with minimal training. RDR systems, as products, services or used internally have been developed by Pacific Knowledge Systems[1], Ivis, Erudine Medscope, SeeGene, Hyundai MnSoft, Hyundai Steel, G & Net, BIT Computer, IPMS and Yawl

The three demonstrators

SCRDR only allows a single class for rule conclusions. That is, an animal can be classified as a mammal or a fish etc, but not as both. MCRDR allows for multiple classifications, e.g. an animal can be both a mammal and a quadruped etc. Neither SCRDR or MCRDR allows for heuristic classification, that is intermediate abstractions or conclusions given before the final conclusion is reached; however, Excel_SCRDR and Excel_MCRDR can both call external abstractions written in VBA. Excel_GRDR provides for intermediate abstraction and inference looping and can be used for construction as well as classification. This manual does not discuss the motivations for the different types of RDR or other variants of these; it is simply a manual on how to use them.

Purpose of these demonstrators

The ideas in RDR are extremely simple, but tend to be counter-intuitive, particularly for experienced developers, who tend to tend to assume, for example, that a tool for building and maintaining a knowledge base must include a knowledge base editor – which RDR does not include. The purpose of these demonstrator is to provide an opportunity for potential RDR users to experience why and how RDR works and also to conduct simple experiments evaluating the development costs and accuracy of an RDR knowledge base.

If users do want to do something a little more sophisticated in data manipulation an excel formula, which operates on the case data can be used as a conclusion.

Limitations of these demonstrators

These demonstrators are not intended for industrial use. An industrial RDR system will require a much faster inference engine, a customized interface and generally, integration with an existing information system. Finally bugs are highly likely to emerge. I have done manual testing on various versions of Excel, for both Macs and Windows, but since this has only been manually testing, and the programs are quite complex, it is highly likely (certain?) that you will run into bugs. If so please let me know

---

[1] http://pks.com.au The main focus of PKS for their generic RDR system is medical applications and they have a large user base. I have a small share-holding in PKS

The Excel RDR demonstrators

Excel_SCRDR

Single classification Ripple-Down Rules

The demonstrator provides five worksheet views.

- The **cases** sheet allows for data entry, with each row being a case. From this sheet the inference engine can be run on cases or a case can be selected for rule addition.
- The **rule builder** sheet provides an interface for building a rule for a case.
- The **rules** sheet shows the rules in the knowledge base and also a rule trace for the current case.
- The **cornerstone cases** sheet shows the cases for which rules have been added.
- The **statistics** sheet provides some simple statistics on the performance of an evolving knowledge base, which cannot be directly calculated from the other worksheets.

It also supports the use of rule conditions based on functions defined by user in VBA and stored in the Personal Macro Workbook.

Before opening Excel_SCRDR

Excel_SCRDR stores a compiled version of the rules in the Personal Macro Workbook and updates this every time a new rule is added. The Personal Macro Workbook is a workbook tied to a particular instantiation of Excel. If you have not used it before you will need to open it manually, but from then on it should always be open, but hidden, unless you have selected the Developer Tab.

A simple way to open it for the first time is by using the Developer Tab which has to be added to the ribbon. In Windows under the *file tab* select *options* and then *customize the ribbon* then under *main tabs* select the *developer tab*. On a Mac click on *preferences* under the *Excel* menu at the top left of the screen. Under *sharing and privacy* then select *ribbon* and then the *developer tab*. Then click the *developer tab*, which will give access to the VBA editor and allow the user to record a macro. Record a trivial macro and then when it comes to save, select the option to save it in the Personal Macro Workbook. You should then be able to run Excel_SCRDR.

This program writes its own VBA code, so if you are using Windows you first have to set Excel trust preferences to allow access to the VBA project object model. This is not necessary in Mac Excel 2011 or 2016. In Windows under the *file tab* select *options* and then select *Trust Center* and the *Trust Center Settings*. Then select *Macro Settings* and check the box for *Trust access to* the *VBA project object model* and also *enable all macros*. Then close all Office applications and restart Excel.

Note that access to the VBA project object model makes your machine very vulnerable, so be sure to turn off access to the *VBA project object model* and *enable all macros* before opening any Excel workbook from a non-trusted source.

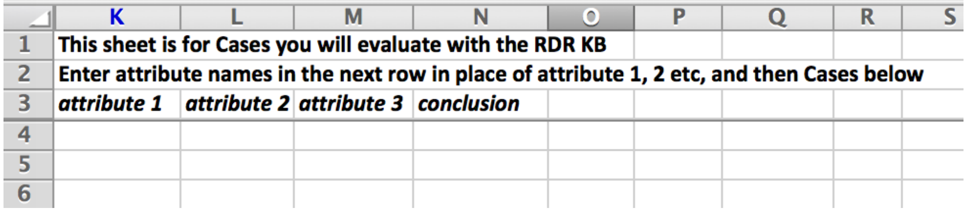
Opening Excel_SCRDR
On a Mac you will be asked to enable macros.

Then for a new knowledge base on Windows or Mac you will be asked indicate the number of attributes a case has, but this can be changed at any stage.

An **Excel_SCRDR** workbook obviously can be copied and renamed

If the user wants to start a new knowledge base they will be asked how many attributes are need to for a case.

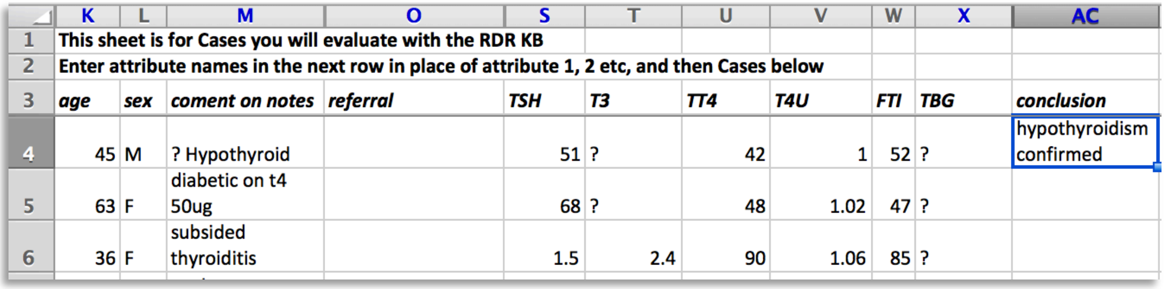
Cases Worksheet
If the user specified 3 attributes, they would be shown this screen:



	K	L	M	N	O	P	Q	R	S
1	**This sheet is for Cases you will evaluate with the RDR KB**								
2	**Enter attribute names in the next row in place of attribute 1, 2 etc, and then Cases below**								
3	***attribute 1***	***attribute 2***	***attribute 3***	***conclusion***					
4									
5									
6									

The user can add or edit attribute names at any stage and paste data including attribute names to this sheet. Such changes, as well as inserting, deleting or hiding columns, changing column widths or word-wrapping (applied to the whole column) will also be applied to the other relevant worksheets. The final column must be labeled ***conclusion*** and you should be warned if it is not, but check anyway before leaving this sheet.   A user will be warned if they try to delete a column for attribute that has been used in a rule condition, as any such change will make a the rule more general covering more cases, perhaps inappropriately. **Note**: if you want to change the conclusion column after you have already added rules, you should do this by inserting or deleting columns before the conclusion column. Editing or overriding the conclusion column itself may mess up your rules. All such changes will be propagated from the cases worksheet to the other worksheets

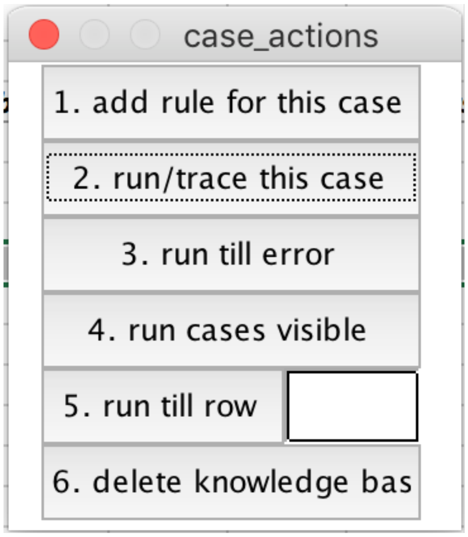
E.g. an example of a thyroid case database:



	K	L	M	O	S	T	U	V	W	X	AC
1	**This sheet is for Cases you will evaluate with the RDR KB**										
2	**Enter attribute names in the next row in place of attribute 1, 2 etc, and then Cases below**										
3	***age***	***sex***	***coment on notes***	***referral***	***TSH***	***T3***	***TT4***	***T4U***	***FTI***	***TBG***	***conclusion***
4	45	M	? Hypothyroid		51	?	42	1	52	?	hypothyroidism confirmed
5	63	F	diabetic on t4 50ug		68	?	48	1.02	47	?	
6	36	F	subsided thyroiditis		1.5	2.4	90	1.06	85	?	

On this sheet a row corresponds to a case and the conclusion column is where the RDR conclusion for the case will be added. Columns can be freely added, e.g. for using Excel formulae to create new attributes.

If inference fails perhaps because of a type difference between the rule condition and the attribute value, the conclusion should be an error message indicating the first rule at which an error occurred and inference was aborted.



Double clicking *on a row on the cases sheet shows this menu.*

1. *add rule for this case* copies the case to the rule builder worksheet
2. *run/trace this case* adds the conclusion for the case and takes the user to the rules worksheet to show a rule trace for the case.
3. *run till error* is applicable only if the second last column is labeled <u>target</u>. Cases will be processed until there is a mismatch between the <u>conclusion</u> and the <u>target</u>
4. *run cases visible* does exactly this but colours any errors red, if a <u>target</u> column is used
5. *run till row* inference will continue until the row number is reached or if no row number is entered, until all cases are processed. Inference should stop on a further double click, but this is unreliable, so run a small number of cases before doing large numbers. There can be very large differences in inference speed between different versions of Excel and different machines.
6. *delete knowledge base* enables you to delete all the rules you have added and all current case conclusions, but retain the case data.

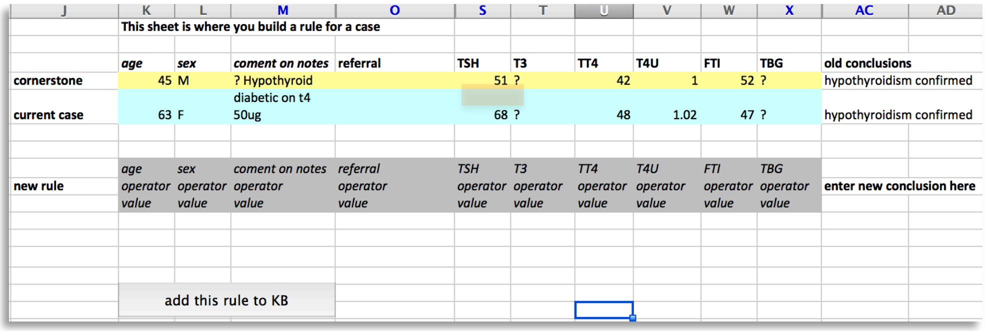
Data sets from sources such as the UCIrvine repository often have the target concept or label specified. Ensure this is in the second last column and labeled ***target***, for the system to identify incorrectly classified cases. In industrial applications there will be some sort of sequence of cases processed by an RDR knowledge base and monitored by a user and when an incorrect output is observed a new rule is added to give the correct output. To mimic this without the effort of manual checking each case use UCIrvine cases or similar and the *run till error* option, adding a rule to deal with each error

Excel_SCRDR Rule Builder worksheet

This worksheet is where rules are built for cases, and a central feature of RDR is that rules are built for cases, rather than by asking the expert for important information for the domain. There are a few general tactics to bear in mind when building RDR rules for cases

- As for any rule system, over-specialized rules with many conditions will result in more rules being added than required. With RDR there is simply no point in making rules over-specific.
- With RDR, over-generalized rules are not a problem as they are very rapidly corrected with future cases and rules
- Although over-generalized rules can be rapidly added without much thought, it is a mistake to add stupid rules specifying irrelevant conditions. E.g. If a patient's sex was irrelevant to a medical knowledge base, starting off with an "IF female" rule, could result in repeating the same rules for both males and females.

If from the pop-up menu *2. add rule for this case* clicked for row 5 in the example above, the system would go to the rule builder worksheet and show:



J	K	L	M	O	S	T	U	V	W	X	AC	AD
	This sheet is where you build a rule for a case											
	age	*sex*	*coment on notes*	referral	TSH	T3	TT4	T4U	FTI	TBG	old conclusions	
cornerstone	45	M	? Hypothyroid		51	?	42	1	52	?	hypothyroidism confirmed	
current case	63	F	diabetic on t4 50ug		68	?	48	1.02	47	?	hypothyroidism confirmed	
	age	*sex*	*coment on notes*	*referral*	*TSH*	*T3*	*TT4*	*T4U*	*FTI*	*TBG*		
new rule	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	enter new conclusion here	
	value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*		
	add this rule to KB											

The case for which the rule is to be added is shown in blue, together with the current conclusion, which the user has decided is wrong for this case. The rule giving this wrong conclusion was added for a specific case, which is shown in yellow and orange. Cases for which past rules were added are known as <u>cornerstone cases</u>. Attributes used in rules reaching the conclusion are coloured orange for the cornerstone case (the same attributes are also used for the case of course). Double-clicking the case will show its current rule trace on the rules worksheet – which is the same rule trace as for the cornerstone cases. It is not necessary to know this, but it may be of interest.

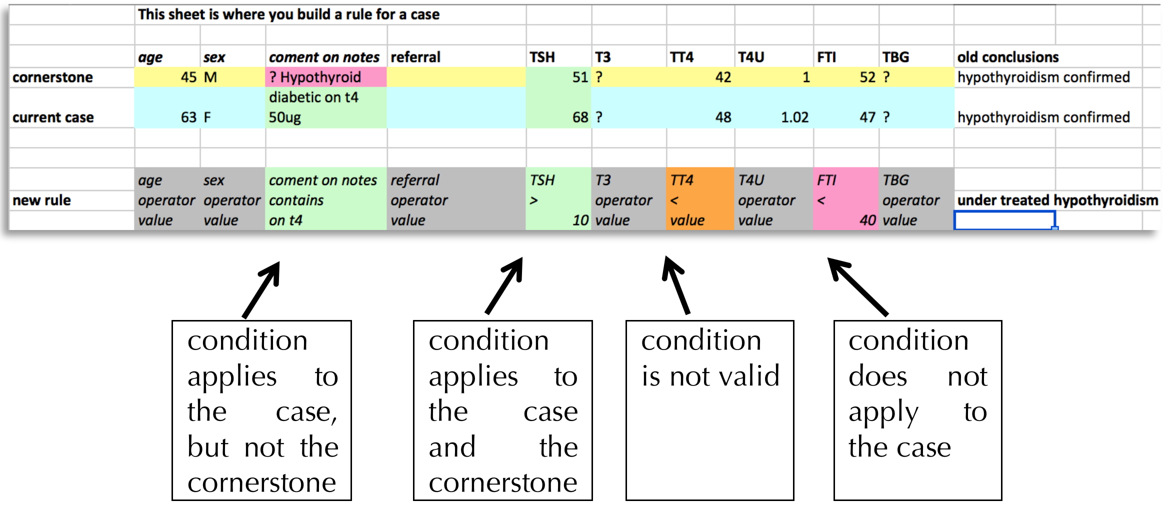
A rule condition for an attribute is constructed by replacing 'operator' with one of the standard operators provided:

- the numerical operators >, < , >=, <=, =
- the case-independent Boolean/text operator 'is'
- the case-independent text operator 'contains'

The user then replaces the 'value' field with appropriate value. The following colour scheme is used to show the status of a rule condition as it is created:

Brown	the rule condition is invalid, and cannot be evaluated against the case.
Green	the condition is valid and applies to the case. The cornerstone case may be red or green depending on whether the condition also fires on the cornerstone case
Red	the condition is valid, but the rule cannot fire as the condition is not true

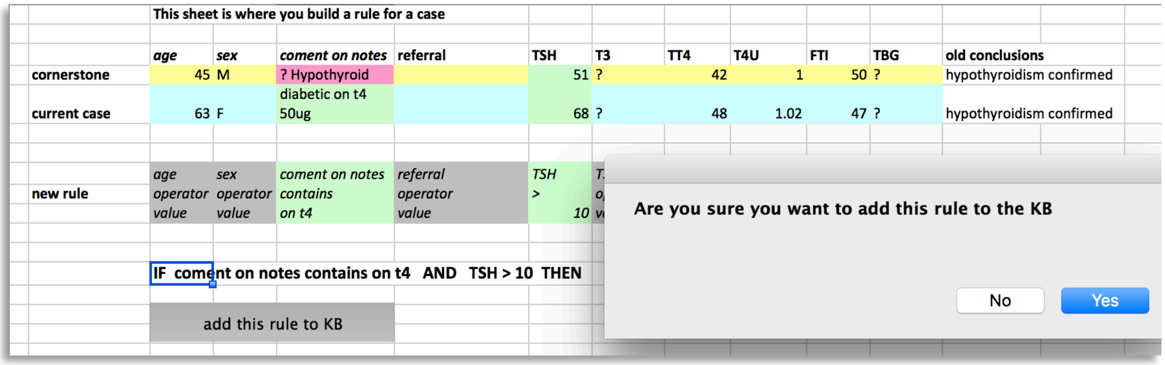
The following rule could be created for the case above:



	This sheet is where you build a rule for a case											
	age	*sex*	*coment on notes*	referral	TSH	T3	TT4	T4U	FTI	TBG	old conclusions	
cornerstone	45	M	? Hypothyroid		51	?	42	1	52	?	hypothyroidism confirmed	
current case	63	F	diabetic on t4 50ug		68	?	48	1.02	47	?	hypothyroidism confirmed	
	age	*sex*	*coment on notes*	*referral*	*TSH*	*T3*	*TT4*	*T4U*	*FTI*	*TBG*		
new rule	*operator*	*operator*	*contains*	*operator*	*>*	*operator*	*<*	*operator*	*<*	*operator*	under treated hypothyroidism	
	value	*value*	*on t4*	*value*	*10*	*value*	*value*	*value*	*40*	*value*		

condition applies to the case, but not the cornerstone
condition applies to the case and the cornerstone
condition is not valid
condition does not apply to the case

The user will be prevented from adding this rule to the knowledge base until the invalid (TT4) condition and inapplicable condition (FTI) are fixed or removed. After the rule is corrected and the add rule button is clicked the user will see:



This sheet is where you build a rule for a case

	age	sex	coment on notes	referral	TSH	T3	TT4	T4U	FTI	TBG	old conclusions
cornerstone	45	M	? Hypothyroid		51	?	42	1	50	?	hypothyroidism confirmed
current case	63	F	diabetic on t4 50ug		68	?	48	1.02	47	?	hypothyroidism confirmed

new rule	age operator value	sex operator value	coment on notes contains on t4	referral operator value	TSH > 10

IF coment on notes contains on t4 AND TSH > 10 THEN

Are you sure you want to add this rule to the KB

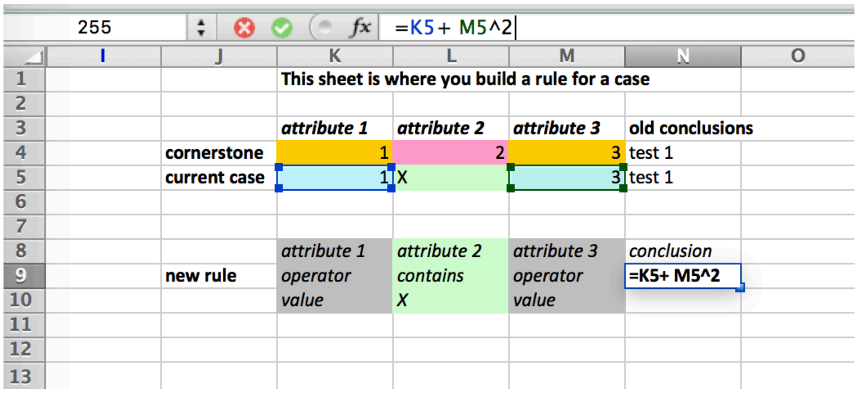
A range of other message may be shown including:

- If the rule fires the cornerstone case, the user will be asked to specify a rule condition that will fail on the cornerstone case or to confirm that the new rule should apply to the cornerstone case. This is rarely required as what it generally means is that the user does not like the wording of the conclusion and wants to change this wording for the new case and the conclusion. As discussed below, a user can directly edit the wording for a conclusion at any stage as this does not change the concept covered by the rule, only the wording for the label.
- If the user adds, edits or deletes a conclusion, but does not add any rule conditions, the system will query whether the user is adding or changing a default conclusion.

<u>Unless you want to develop you own operators, develop abstractions etc you can skip the following sections and go straight to the **rules worksheet** section</u>

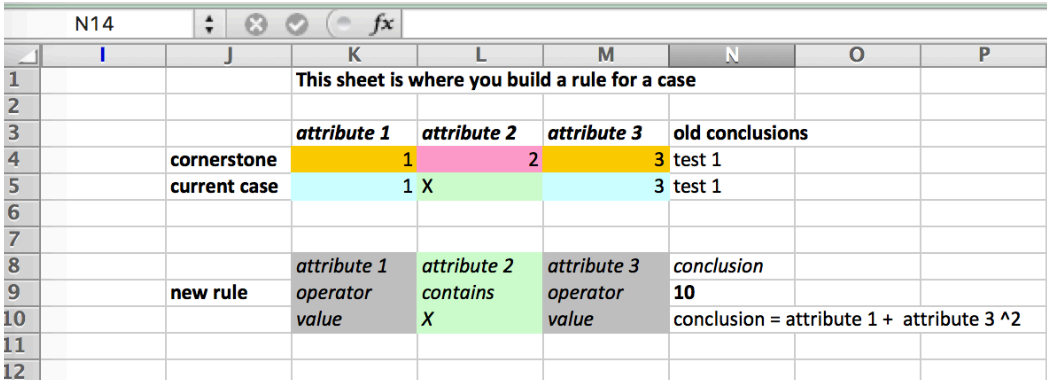
Formula conclusions

For normal knowledge base system application a conclusion is a string. However, for data manipulation it may be desirable that the conclusion be function generates a conclusion be applying whatever legal Excel formula is desired to various attributes for the case. As a demo, **Excel_SCRDR** does not require this feature, it is included for experimental purposes. Simply start the conclusion with an **=** sign and then click on the required cells in the **current case** row in the standard way one clicks on cells. An example is shown in the first image below. The standard practice in Excel is followed where as the formula is entered into the conclusion it shows in the formula panel above with any cells used in the formula highlighted. The second image shows what happens after return is clicked. The conclusion now shows the value of the conclusion for the case, and under it is an independent representation of the formula that will be applied to future cases.



255 =K5+ M5^2

	I	J	K	L	M	N	O
1			This sheet is where you build a rule for a case				
2							
3			*attribute 1*	*attribute 2*	*attribute 3*	old conclusions	
4		cornerstone	1	2	3	test 1	
5		current case	1	X	3	test 1	
6							
7							
8			*attribute 1*	*attribute 2*	*attribute 3*	*conclusion*	
9		new rule	*operator*	*contains*	*operator*	=K5+ M5^2	
10			*value*	*X*	*value*		
11							
12							
13							



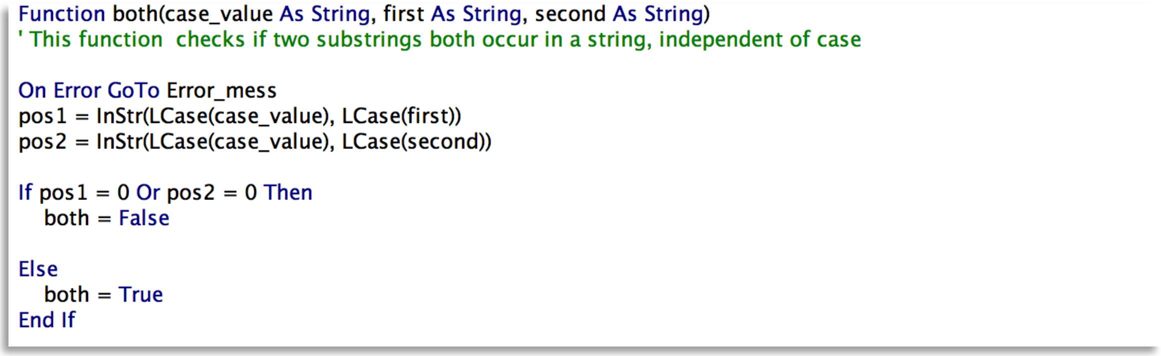
N14

	I	J	K	L	M	N	O	P
1			This sheet is where you build a rule for a case					
2								
3			*attribute 1*	*attribute 2*	*attribute 3*	old conclusions		
4		cornerstone	1	2	3	test 1		
5		current case	1	X	3	test 1		
6								
7								
8			*attribute 1*	*attribute 2*	*attribute 3*	*conclusion*		
9		new rule	*operator*	*contains*	*operator*	10		
10			*value*	*X*	*value*	conclusion = attribute 1 + attribute 3 ^2		
11								
12								

If a value cannot be calculated for a formula conclusion, the conclusion is shown as **#VALUE** in the normal way

Other rule conditions

The limited number of rule conditions provided, should be sufficient for most evaluation purposes; however, new operators can be defined as VBA functions in a module in the Personal Marco Workbook. The file my_rdr_macros provided in the zip file gives some examples of user-defined functions. These are also loaded automatically whenever **Excel_SCRDR** is loaded and has to be overwritten if you want to use your own. The current file includes functions: *both, before, just_before, is_missing, is_something, is_not, on_thyroxine*. You can replace there or alternatively all other modules with you functions to the Personal Macro Workbook. The function *both* is shown below.



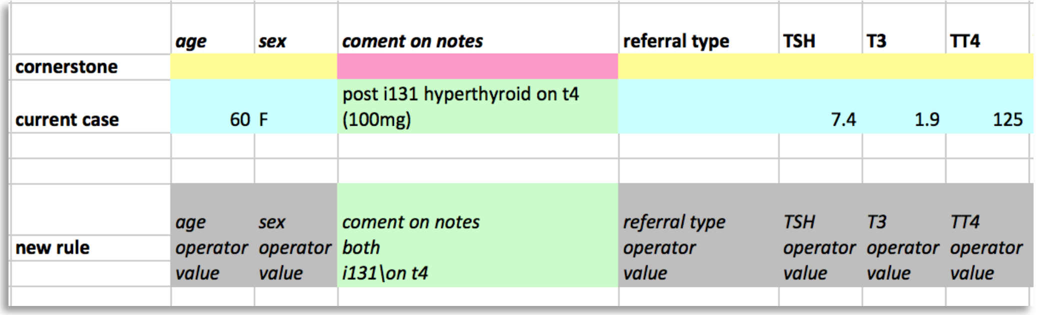
```
Function both(case_value As String, first As String, second As String)
' This function  checks if two substrings both occur in a string, independent of case

On Error GoTo Error_mess
pos1 = InStr(LCase(case_value), LCase(first))
pos2 = InStr(LCase(case_value), LCase(second))

If pos1 = 0 Or pos2 = 0 Then
   both = False

Else
   both = True
End If
```

This is called as follows:



	age	sex	coment on notes	referral type	TSH	T3	TT4
cornerstone							
current case	60	F	post i131 hyperthyroid on t4 (100mg)		7.4	1.9	125
	age	sex	coment on notes	referral type	TSH	T3	TT4
new rule	operator	operator	both	operator	operator	operator	operator
	value	value	i131\on t4	value	value	value	value

The function name is entered into the *operator* cell, and the values to be passed to the function are place in the value cell separated by "\". The VBA function *both* has three parameters. The first parameter is reserved for the case attribute value, here 'post i131 hyperthyroid on t4 (100mg)', while the other two parameters values passed in this example are 'i131' and 'on t4'. Up to three parameters (as well as the case attribute value) can be passed to used-defined functions. If a function is not called correctly, the condition will be coloured orange indicating an invalid condition. If you are using a function that only requires the case value and not a condition value; e.g. on_thyroxine is_something you will need to delete *value,* so that field is empty.

Abstractions (heuristic classification)

SCRDR only allows for simple classification; i.e. there is no intermediate abstraction step before rules are applied. If some sort of intermediate abstraction is required even for demo purposes users can:

- Make copies of a column on the data worksheet, so multiple rule conditions can be applied to the same attribute,
- add a column with an abstraction defined using an Excel formula
- add a user-defined function to a module in the Personal Macro Workbook. For example in the thyroid domain show here, there are many phrases that can occur in the 'comment on notes' field indicating the patient is on thyroid hormone replacement therapy, such as: 'on t4', 'on thyroxine', 'replacement', 'hypo rx', 'rx', 'orox' etc. A demonstration function *on_thyroxine* is included in my_rdr_macros. The only feedback you get that a function has not been called correctly is that the rule condition is coloured orange.

Ideally abstractions should be specified by rules (which are also developed incrementally) rather than VBA code or similar. This requires repeat inference and is shown in the Excel_GRDR systems and is also described in various RDR papers. Industrial-strength systems such as PKS RippleDown tend to have some sort of abstraction and repeat inference facilitiy.

Multi-dimensional data

This demonstration program is limited to simple attribute-value data and cannot directly deal with temporal data such a patient's accumulated pathology results or other multi-dimensional data. The only way to deal with such data with this program is to add columns for abstractions such as maximum, minimum, increasing. This is not a satisfactory solution if such abstractions are required for each of a large number of attributes. Industrial systems such as PKS RippleDown handle such requirements by calculating or inferring temporal abstractions on the fly.

If a user really wanted to explore using this demo program for more complex data with knowledge bases calling each other, a function called *run_this_case* when passed a row number referring to the Cases worksheet will return the conclusion for that case as a string. The header for this function is:

```
Function run_this_case(ByVal row As Integer) As String
```

And it can be called from another worksheet in the same directory using:

```
Dim conclusion As String
Conclusion = Application.Run("'excel_rdr.xlsm'!run_this_case", row_no)
```

Where `excel_rdr.xlsm` is replaced by whatever your copy of this workbook is called.

A better solution is to copy the kb_compiled macro and any user macros to whatever workbook you want to user it with. It is entirely self-contained using only standard VBA. The knowledge base is called as a function returning the conclusion. The data for a case being processed is contained in a global variable **rdr.case_array** which corresponds to a row in the **cases** worksheet

The Excel_SCRDR rules worksheet

SCRDR is essentially a binary tree with a rule at each node. This form of RDR was used in PEIRS a large early RDR medical expert[2]. As in PEIRS such SCRDR binary trees tend to be very unbalanced with far more new rules than correction rules. This sort of binary tree of rules can be represented as linked production rules[3]. Normal production rule knowledge bases contain no information about the order in which rules will be evaluated, as this is determined by the inference engine with its various conflict resolution strategies. And one of the reasons that these systems are difficult to maintain is that it is difficult for the knowledge engineer to anticipate the order in which a rules will be processed after editing a knowledge base. With linked production rules each rule contains information about which rule be processed next. The standard production rule representation is simply:

```
if [condition] then [action-list]
```

giving no information about evaluation order. Linked production rules are represented as:

```
 if [condition] then [case action-list],
                      [inference action]
else [inference action]
```

where the **`case`** `action-list` are simply the normal inference actions of asserting or retracting a conclusion. The **`inference`** `actions` simply specify which rule the inference engine next evaluates. The rules worksheet shows these inference actions on the left,

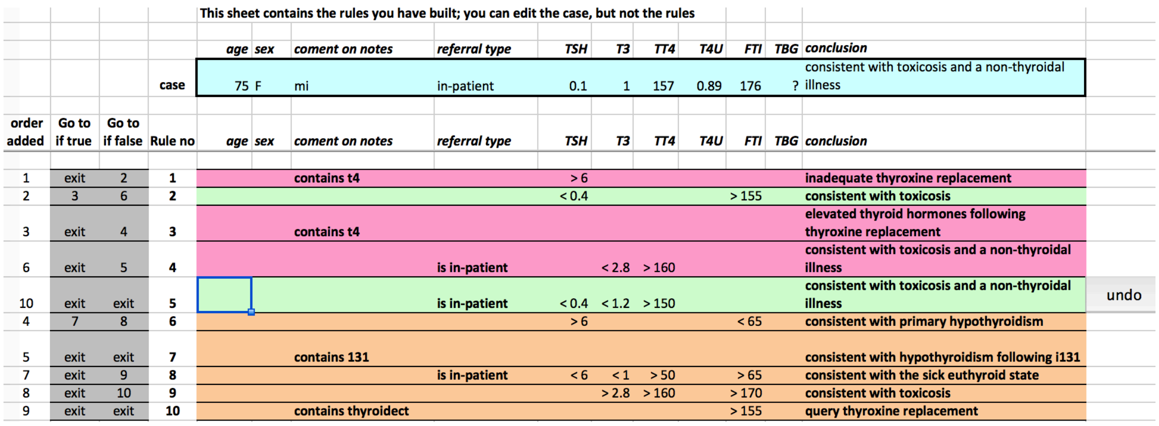
---

[2] Edwards, G., Compton, P., Malor, R., Srinivasan, A., & Lazarus, L. (1993). PEIRS: a pathologist maintained expert system for the interpretation of chemical pathology reports. *Pathology, 25*, 27-34.

[3] Compton, Paul, Yang Sok Kim, and Byeong Ho Kang. "Linked production rules: controlling inference with knowledge." In *2014 Pacific Rim Knowledge Acquisition Workshop (PKAW 2014)*, Springer. pp. 84-98. 2014.

specifying which rule to go to depending on whether the rule on the same line evaluates true of false.

This is all fairly trivial, but it is important in that all RDR structures can be represented as linked production rules, and this will be seen for both Excel_MCRDR and Excel_GRDR. That is, for these various methods, inferencing is exactly the same, independent of the type of RDR – each rule specifies the next rule to be evaluated. The different types of RDR are just different ways or organizing the rules at knowledge acquisition time.



This sheet contains the rules you have built; you can edit the case, but not the rules

	age	sex	coment on notes	referral type	TSH	T3	TT4	T4U	FTI	TBG	conclusion
case	75	F	mi	in-patient	0.1	1	157	0.89	176	?	consistent with toxicosis and a non-thyroidal illness

order added	Go to if true	Go to if false	Rule no	age	sex	coment on notes	referral type	TSH	T3	TT4	T4U	FTI	TBG	conclusion
1	exit	2	1			contains t4		> 6						inadequate thyroxine replacement
2	3	6	2					< 0.4				> 155		consistent with toxicosis
3	exit	4	3			contains t4								elevated thyroid hormones following thyroxine replacement
6	exit	5	4				is in-patient		< 2.8	> 160				consistent with toxicosis and a non-thyroidal illness
10	exit	exit	5				is in-patient	< 0.4	< 1.2	> 150				consistent with toxicosis and a non-thyroidal illness
4	7	8	6					> 6				< 65		consistent with primary hypothyroidism
5	exit	exit	7			contains 131								consistent with hypothyroidism following i131
7	exit	9	8				is in-patient	< 6	< 1	> 50		> 65		consistent with the sick euthyroid state
8	exit	10	9						> 2.8	> 160		> 170		consistent with toxicosis
9	exit	exit	10			contains thyroidect						> 155		query thyroxine replacement

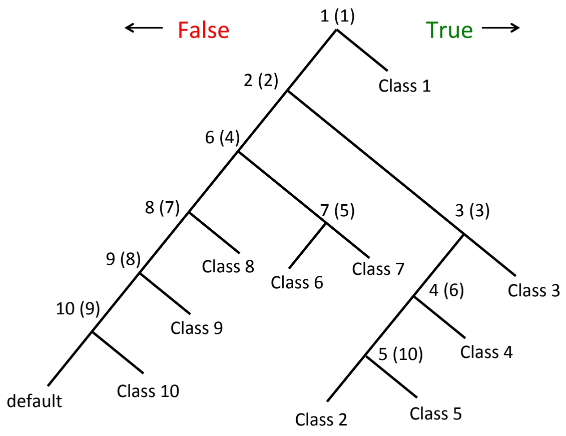
In the example above each row shows a single rule. Rule conditions are expressed in the same way as on the rule builder sheet except that the operator and value are shown in the same cell separated by a space. Note that if the conclusion is a function the application of the function to that case will be shown for the rule, regardless of whether the rule has fired.

The current case is in the upper panel, and the rule colouring shows the rule trace for that case:

Brown	the rule was not evaluated for this case.
Green	the rule has been evaluated and applies to the case
Red	the rule has been evaluated but did not apply to the case

The conclusion for the case, shown in the upper panel comes from the last rule that applied to the case (i.e. the last green rule)

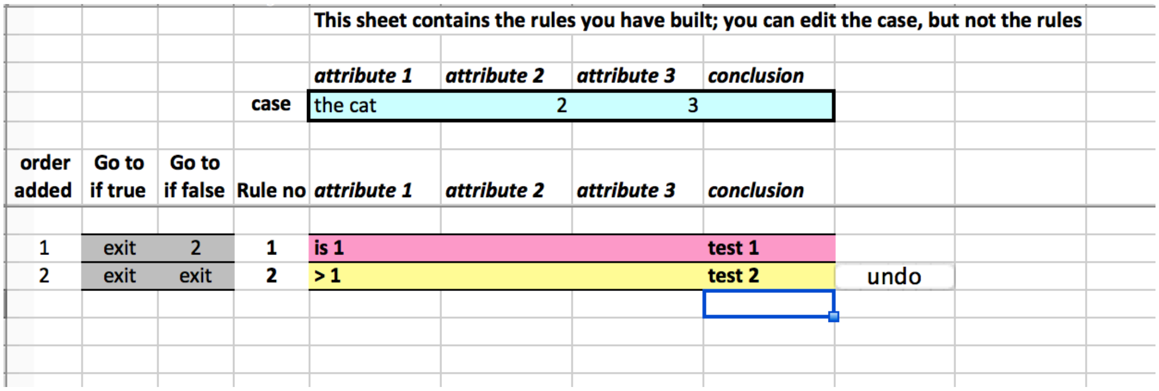
The rules above correspond to the following binary tree:



Rule no (order added)
Class X is the conclusion of Rule X

The rule number gives the evaluation order which is a depth-first traversal of the tree, true branches first. When a rule is added, it is added to the point at which the case exited the tree. In the example above rule 5 (the 10th rule to be added) was added because the conclusion from rule 2 given for the case was incorrect. This conclusion was given because neither rule 3 or 4 fired for the case and so the rule 5 was added to the false branch of rule 4 (the exit point). However, none of this need concern the user as RDR organizes rule placement automatically. Other RDR strategies have different rule structures, but again handled automatically.

If a rule cannot be evaluated, for example because of a type conflict between the rule condition and the attribute value the rule will be coloured yellow. In the example below the first rule condition **is 1** assumes the value for attribute 1 should be a string. It is false for the case shown with value **the cat** for attribute 1. However the second rule expects a numerical value for attribute 1. If this is not the case inference stops at this rule and it is coloured yellow. The conclusion on the case sheet will also show and error occurred at rule 2.



This sheet contains the rules you have built; you can edit the case, but not the rules

	attribute 1	attribute 2	attribute 3	conclusion
case	the cat	2	3	

order added	Go to if true	Go to if false	Rule no	attribute 1	attribute 2	attribute 3	conclusion	
1	exit	2	1	is 1			test 1	
2	exit	exit	2	> 1			test 2	undo

Rules worksheet actions

- The case in the upper panel can be edited and the rule trace for the edited case will be shown. This case can also be double-clicked to show the trace
- Double-clicking one of the rules will show the cornerstone case for that rule in the case panel and its rule trace
- The conclusions for the rules (and the default conclusion) can be edited, but not the rule body. That is the rule continues to cover the same concept, but the name or description of the concept can be edited, for cosmetic purposes
- There is an undo button next to the last rule added. There is no need for an undo capability with RDR, but with an unfamiliar demo a user may make a mistake in clicking the add rule button before a rule is properly formed and may wish to undo this. Only the last rule can be undone.

Cornerstone case worksheet

This worksheet shows the cases for which rules were added, in the order in which they were added. Double-clicking a case shows its rule trace of the rules worksheet. If a cornerstone is changed by a later rule (and this will have been flagged to the user), that case is shown as brown and the last conclusion for this case will shown as well as the current conclusion.

Statistics worksheet

This worksheet shows:

- The row number of the case for a rule is added. If the cases are processed sequentially and rules added only when a conclusion is incorrect or missing, recording the frequency with which rules are added compared to cases processed can be used as a measure of evolving accuracy
- A graph of rule number (i.e. order in which the rules were added) against case (row) number is also shown. The slope of the graph at any stage is a measure of accuracy at that stage.
- The time taken to add each rule is also shown. This is the elapsed time from when the user clicks add rule until the user is returned to the cases screen after rule addition. The time recorded will also include time away from the computer during rule addition.
- The number of cases processed for which each rule provided the conclusion is also shown. It doesn't matter how often a case is rerun, it is only counted once in this statistic.
- Assuming the second last column is the target conclusion, the number of cases where the RDR conclusion and the target conclusion is the same is shown for each rule.
-

These statistics are included because they cannot be readily derived after a user has built a knowledge base. It is assumed that users may wish to run completely unseen test cases etc and use standard Excel formulae to derive other statistics

Test data

Any attribute-value data can be used as test data. If a user doesn't have their own data set of interest, standard machine-learning data sets for classification can be used, e.g. from the UCIrvine machine learning repository[4]. Such data sets have the advantage that they often include the target conclusion. A few UCIrvine data sets have been converted to Excel workbooks and are included here. Apart from the Zoo data set it is unlikely users will know appropriate rules for such data sets; however, this can be provide a demonstration of another advantage of RDR. As long as a user tries to guess at reasonable rules, an RDR system must eventually converge towards a providing correct conclusions for a domain – even random guesses at rules will converge, albeit very slowly. A risk with these data sets; however, is that there can be errors in some cases in either the target label or the data, which makes guessing rules more problematic. Because these data sets were prepared for machine learning data they tend to use simplified attribute representations, whereas in the real-world example shown in the screen shots above some attributes are informal text.

A knowledge-base has been started for the zoo data set. The attributes in the Zoo data set are nearly all 0 or 1 to facilitate machine learning. Supposedly this data set does have errors, but if you build a complete knowledge base for it, about 10 rules, you will find there are in fact two erroneous cases.

---

[4] http://archive.ics.uci.edu/ml/index.php

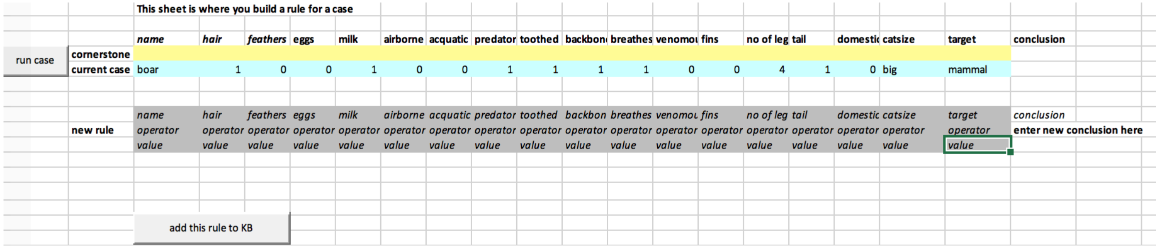
Excel_MCRDR

Multiple classification Ripple-Down Rules

The material on the SCRDR demonstrator should be read first, as the following covers only features that differ from SCRDR

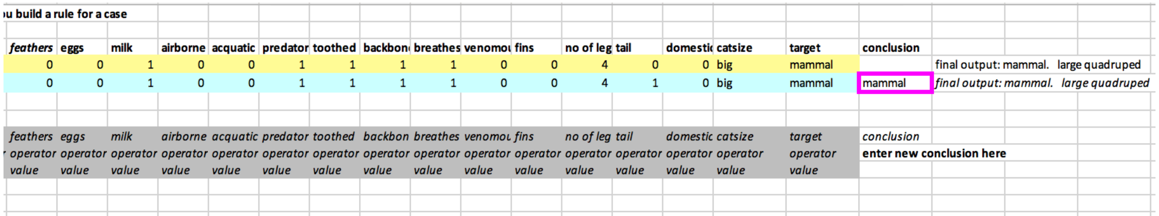
Excel_MCRDR Rule Builder worksheet

The initial difference from SCRDR is that there is a **run case** button to the left of the screen and the case is not run until this is clicked.

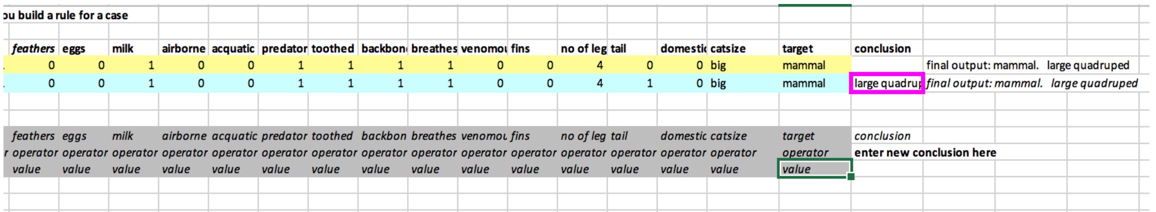


		This sheet is where you build a rule for a case																		
		name	*hair*	*feathers*	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion
	cornerstone																			
	current case	boar	1	0	0	1	0	0	1	1	1	1	0	0	4	1	0	big	mammal	
		name	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*predator*	*toothed*	*backbon*	*breathes*	*venomou*	*fins*	*no of leg*	*tail*	*domestic*	*catsize*	*target*	*conclusion*
	new rule	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	**enter new conclusion here**
		value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	

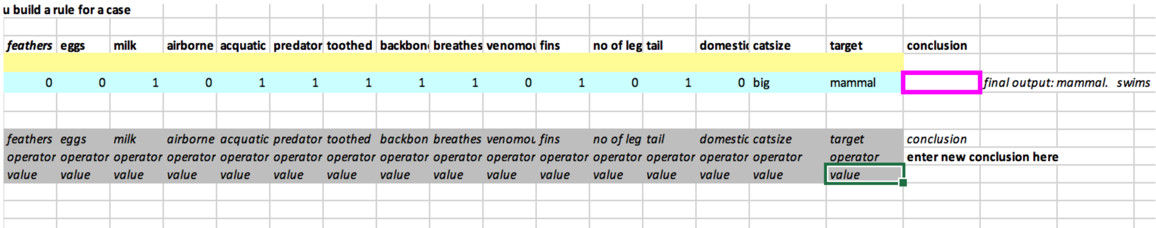
Each time the **run case** button is clicked the next of the multiple conclusions to be given will show in the case conclusion field with a red border, as shown below. After a blank conclusion is shown, cycling through the various multiple conclusions will commence again. Throughout, to the right of the conclusion cell the final complete output is shown.



u build a rule for a case																	
feathers	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
0	0	1	0	0	1	1	1	1	0	0	4	0	0	big	mammal		final output: mammal. large quadruped
0	0	1	0	0	1	1	1	1	0	0	4	1	0	big	mammal	mammal	*final output: mammal. large quadruped*
feathers	*eggs*	*milk*	*airborne*	*acquatic*	*predator*	*toothed*	*backbon*	*breathes*	*venomou*	*fins*	*no of leg*	*tail*	*domestic*	*catsize*	*target*	*conclusion*	
operator	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	**enter new conclusion here**	
value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*		



ou build a rule for a case																	
feathers	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
0	0	1	0	0	1	1	1	1	0	0	4	0	0	big	mammal		final output: mammal. large quadruped
0	0	1	0	0	1	1	1	1	0	0	4	1	0	big	mammal	large quadrup	*final output: mammal. large quadruped*
feathers	*eggs*	*milk*	*airborne*	*acquatic*	*predator*	*toothed*	*backbon*	*breathes*	*venomou*	*fins*	*no of leg*	*tail*	*domestic*	*catsize*	*target*	*conclusion*	
operator	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	**enter new conclusion here**	
value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*		

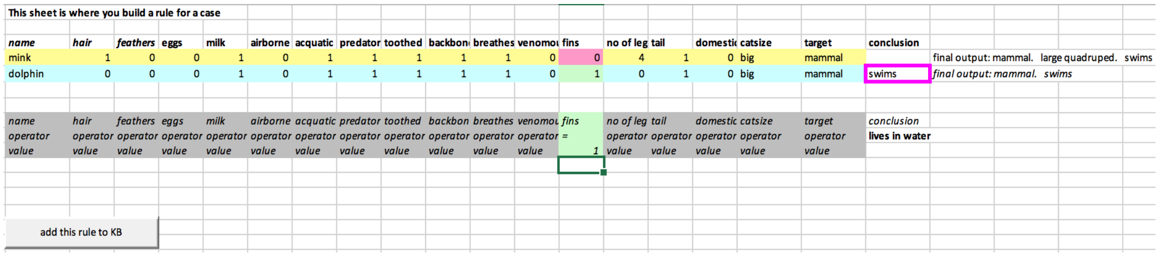


u build a rule for a case																	
feathers	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
0	0	1	0	1	1	1	1	1	0	1	0	1	0	big	mammal		*final output: mammal. swims*
feathers	*eggs*	*milk*	*airborne*	*acquatic*	*predator*	*toothed*	*backbon*	*breathes*	*venomou*	*fins*	*no of leg*	*tail*	*domestic*	*catsize*	*target*	*conclusion*	
operator	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	**enter new conclusion here**	
value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*		

A rule is added in exactly the same way as for SCRDR, but there are three different scenarios:

1. If there is a conclusion in the red-bordered conclusion box and a rule is added with a empty conclusion, it is assumed that this is meant to be a stopping rule which simply prevents that conclusion from being given. The cornerstone case for the rule giving the conclusion to be stopped is shown in the normal way
2. If the red-bordered conclusion box is empty and the new rule has a conclusion, it will be assumed that an extra rule to give this conclusion is to be added. In this case all the cornerstone cases will be checked against the new rule. The process will stop for any cornerstone case that fires the rule. The user can accept that the conclusion should apply to the case or add a further condition to the rule to prevent it firing on the cornerstone case
3. If a rule is entered which gives a conclusion and there is already a conclusion in the red-bordered conclusion box, then both a stopping rule to stop the old conclusion being given will be added, and a new rule to give the new conclusion. The system initially assumes the new rule is the same as the stopping rule, but as the stopping rule is likely to be over-general, further conditions will probably be added while checking the cornerstone cases.

In the following example a porpoise has been classified as mammal and swims. The user might decide this is inappropriate as a porpoise doesn't just swim but lives in water, and as can be seen from the cornerstone case, the original rule was constructed for a mammal that lives near water and swims, rather than lives in water. An (overgeneral) stopping rule might have only the condition fins = 1, so the user clicks the **add this rule to KB button**

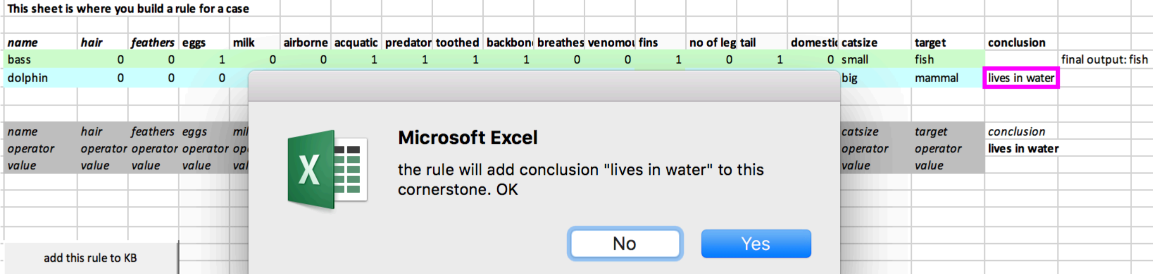


This sheet is where you build a rule for a case

name	*hair*	*feathers*	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
mink	1	0	0	1	0	1	1	1	1	1	0	0	4	1	0	big	mammal		final output: mammal. large quadruped. swims
dolphin	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	big	mammal	swims	*final output: mammal. swims*

name	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*predator*	*toothed*	*backbon*	*breathes*	*venomou*	*fins*	*no of leg*	*tail*	*domestic*	*catsize*	*target*	*conclusion*
operator	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*operator*	*=*	*operator*	*operator*	*operator*	*operator*	*operator*	**lives in water**
value	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*value*	*1*	*value*	*value*	*value*	*value*	*value*	

After the stopping rule is added the system moves to automatically check the cornerstone cases, and stops at the case below.



This sheet is where you build a rule for a case

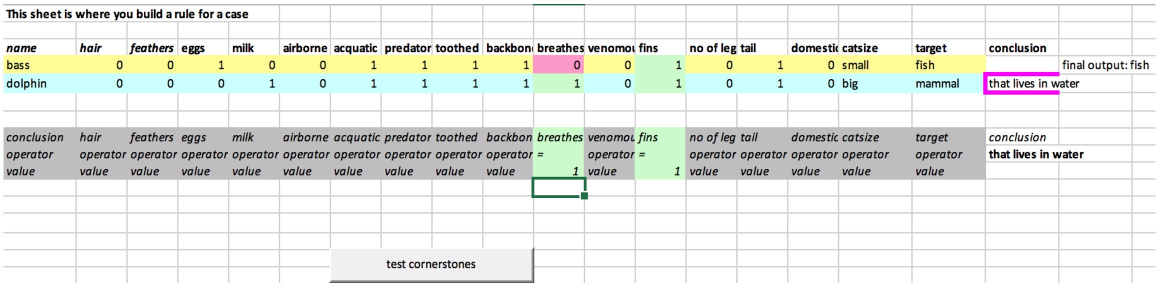
name	*hair*	*feathers*	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	small	fish		final output: fish
dolphin	0	0	0													big	mammal	lives in water	

name	*hair*	*feathers*	*eggs*	*catsize*	*target*	*conclusion*
operator	*operator*	*operator*	*operator*	*operator*	*operator*	**lives in water**
value	*value*	*value*	*value*	*value*	*value*	

Microsoft Excel

the rule will add conclusion "lives in water" to this cornerstone. OK

We could allow lives in water to apply to apply to a fish, but it would be pretty redundant to have a bass classified as fish. lives in water so we click **no** and add the further rule condition breathes = 1 to exclude fish, as below:



This sheet is where you build a rule for a case

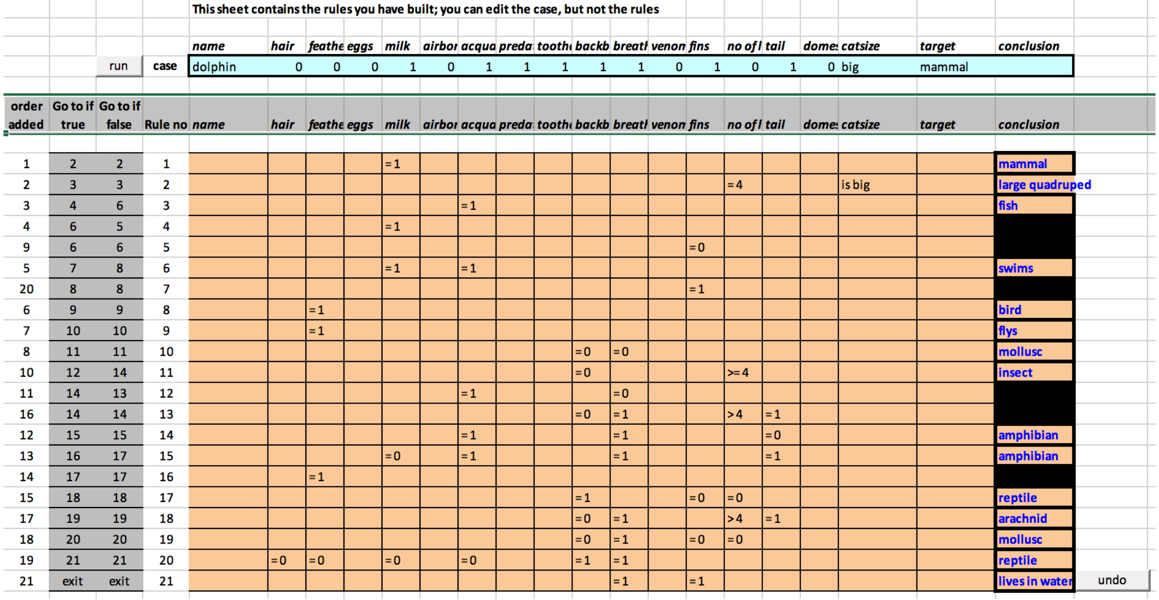
name	hair	feathers	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion	
bass	0	0	1	0	0	1	1	1	1	0	0	1	0	1	0	small	fish		final output: fish
dolphin	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	big	mammal	that lives in water	

conclusion	hair	feathers	eggs	milk	airborne	acquatic	predator	toothed	backbon	breathes	venomou	fins	no of leg	tail	domestic	catsize	target	conclusion
operator	operator	operator	operator	operator	operator	operator	operator	operator	operator	=	operator	=	operator	operator	operator	operator	operator	that lives in water
value	value	value	value	value	value	value	value	value	value	1	value	1	value	value	value	value	value	

If we click **test cornerstones** the system will continue to test further cornerstones, stopping each time one of the cornerstone cases fires the rule. When all cornerstones have been tested the rule is added to the knowledge base

Excel_MCRDR Rules worksheet

The only difference in the use of this worksheet from Excel_SCRDR is that when this page is opened, no rules will have been fired; you have to click the **run** button.

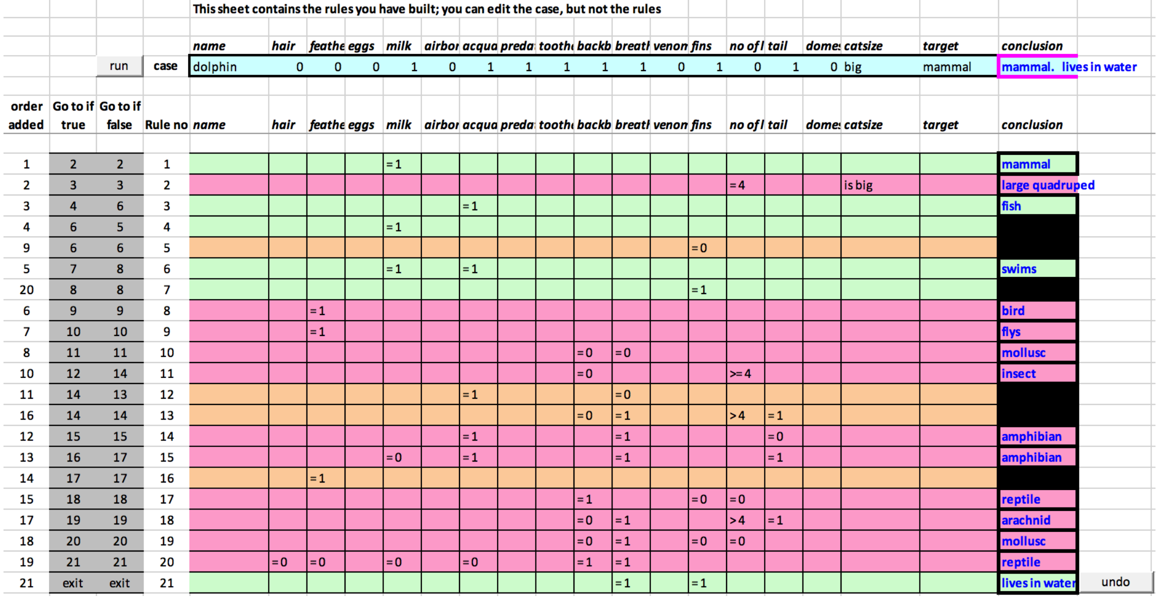


This sheet contains the rules you have built; you can edit the case, but not the rules

		name	hair	feathe	eggs	milk	airbor	acqua	preda	tooth	backb	breatl	venon	fins	no of l	tail	domes	catsize	target	conclusion
run	case	dolphin	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	big	mammal	

order added	Go to if true	Go to if false	Rule no	name	hair	feathe	eggs	milk	airbor	acqua	preda	tooth	backb	breatl	venon	fins	no of l	tail	domes	catsize	target	conclusion
1	2	2	1					=1														mammal
2	3	3	2														=4			is big		large quadruped
3	4	6	3							=1												fish
4	6	5	4					=1														
9	6	6	5													=0						
5	7	8	6					=1		=1												swims
20	8	8	7													=1						
6	9	9	8			=1																bird
7	10	10	9			=1																flys
8	11	11	10										=0	=0								mollusc
10	12	14	11										=0				>=4					insect
11	14	13	12							=1				=0								
16	14	14	13										=0	=1			>4	=1				
12	15	15	14							=1				=1				=0				amphibian
13	16	17	15					=0		=1				=1				=1				amphibian
14	17	17	16			=1																
15	18	18	17										=1			=0	=0					reptile
17	19	19	18										=0	=1			>4	=1				arachnid
18	20	20	19										=0	=1		=0	=0					mollusc
19	21	21	20		=0	=0		=0		=0			=1	=1								reptile
21	exit	exit	21											=1		=1						lives in water

After clicking the run button



This sheet contains the rules you have built; you can edit the case, but not the rules

		name	hair	feathe	eggs	milk	airbor	acqua	preda	tooth	backb	breatl	venon	fins	no of l	tail	domes	catsize	target	conclusion
run	case	dolphin	0	0	0	1	0	1	1	1	1	1	0	1	0	1	0	big	mammal	mammal. lives in water

order added	Go to if true	Go to if false	Rule no	name	hair	feathe	eggs	milk	airbor	acqua	preda	tooth	backb	breatl	venon	fins	no of l	tail	domes	catsize	target	conclusion
1	2	2	1					=1														mammal
2	3	3	2														=4			is big		large quadruped
3	4	6	3							=1												fish
4	6	5	4					=1														
9	6	6	5													=0						
5	7	8	6					=1		=1												swims
20	8	8	7													=1						
6	9	9	8			=1																bird
7	10	10	9			=1																flys
8	11	11	10										=0	=0								mollusc
10	12	14	11										=0				>=4					insect
11	14	13	12							=1				=0								
16	14	14	13										=0	=1			>4	=1				
12	15	15	14							=1				=1				=0				amphibian
13	16	17	15					=0		=1				=1				=1				amphibian
14	17	17	16			=1																
15	18	18	17										=1			=0	=0					reptile
17	19	19	18										=0	=1			>4	=1				arachnid
18	20	20	19										=0	=1		=0	=0					mollusc
19	21	21	20		=0	=0		=0		=0			=1	=1								reptile
21	exit	exit	21											=1		=1						lives in water

Firstly the conclusion given **mammal. lives in water** is a composite of the two conclusions, from rule 1 and rule 21. With MCRDR many conclusions may be produced. For simplicity in this demonstrator, they are simply concatenated together, separated by a full stop and space. In a real application, whatever is done with the multiple conclusions, whether they are concatenated together or treated separate is done outside the MCRDR system. In a pathology system where each conclusion is likely to be sentence, concatenating them as sentences works well, as in this example works well. However it is likely that in some cases essentially the same conclusion might be given twice, so there needs to be some way of managing this.

Next, the knowledge base is exited only after the last rule is evaluated, whereas with SCRDR whenever a rule fires that is not refined by a further rule, inference ceases. It can be seen from the <u>go to if false</u> and <u>go to if true</u> columns, that for most rules, the next rule is evaluated whether or not the rule fires.

Stopping rules are those shown with a black conclusion field. If we take rule 6, if it fails to fire, inference goes to the next rule with a conclusion, rule 8. If rule 6 does fire, then rule 7 a stopping rule is evaluated. Rule 11 has two stopping rules. With more than one stopping rule, each is evaluated in turn until either one fires, stopping the conclusion from the previous rule being given or until the next rule with a conclusion is reached.

Excel_MCRDR cases worksheet

With SCRDR, if the second last column was labelled target, any case where the conclusion disagreed with the target would be labelled red. With an MCRDR conclusion perhaps being a composite of a number of conclusions, this no longer applies. To help provide some advice with MCRDR, a case is coloured red if the conclusion does not <u>contain</u> the target. The idea is that the conclusion should at least include the UCIrvine target, as well as perhaps other conclusions. To turn this off, simply relabel the target column.

Excel_GRDR

General Ripple-Down Rules

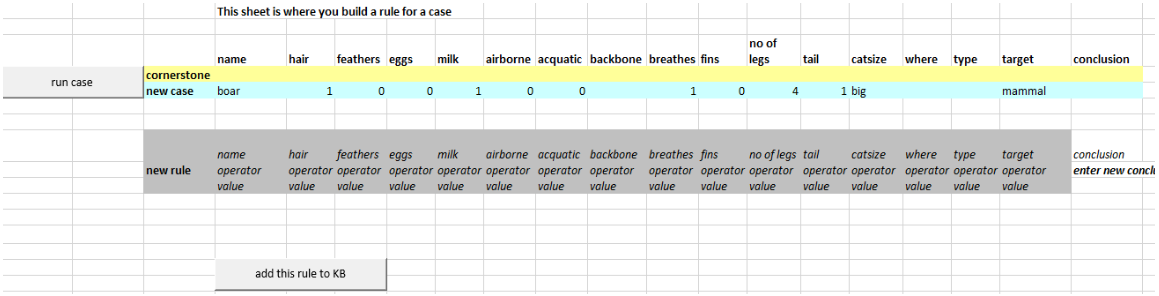
The most obvious feature of GRDR is that it allows for construction tasks, whereas SCRDR and MCRDR only allow for classification tasks, either single or multiple classification. GDR is also of value in classification tasks where it is often desirable to use inference conclusions as conditions in other rules. The essential features of GRDR are:

1. A rule can assert a conclusion for any variable with a missing value.
2. Once a variable has a value, either as part of the original data, or asserted by a rule it cannot be changed by inference.
    a. This avoids any possibility of inference cycles. The essential idea is that if a rule gives a wrong conclusion, this is a mistake and should be corrected by a refinement rule, rather than being changed by inference
3. Inference repeats starting from the first rule whenever a conclusion is asserted for a variable (with a missing value)
    a. The motivation is that new rules are added after inference on previous rules is complete, so it is best to evaluate a rule added after inference on previous rules is complete.
4. In this implementation, there is a special case that there is no repeat inference when a conclusion is reached for the variable **conclusion**. This is not essential but allows for appropriate composition of final output within GRDR
5. Inference stops after the final rule has been evaluated

The simplest implementation of GRDR, similar to the version of MCRDR used here, is to use stopping rules as well as new rules, so that whenever a rule gives a wrong conclusion to be replaced by a different conclusion, a stopping rule is added to the rule and a new rule giving the new conclusion is added after the last rule. In this implementation instead there is an SCRDR tree for every conclusion variable for which a rule is written. Various SCRDR sub-trees are interspersed across the knowledge base, so that when a new conclusion, rather than a correction, is added for a variable, another SCRDR sub-tree is started for that variable.

Excel_GRDR Rule builder worksheet

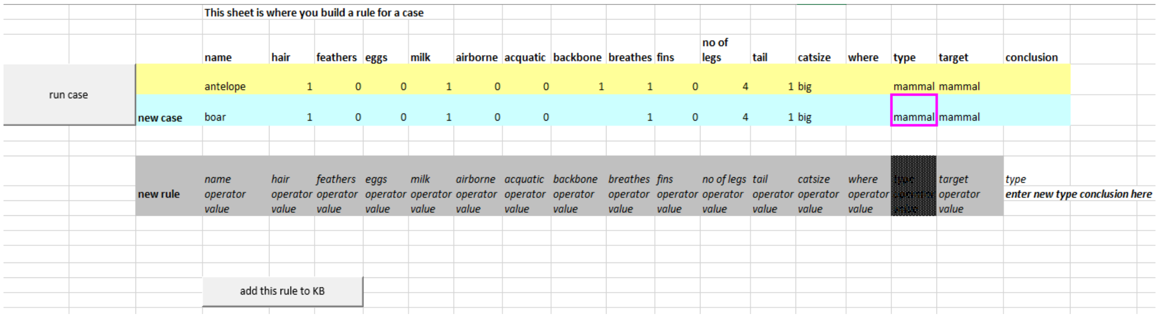
Similarly to MCRDR the rule builder has a **run case** button. Initially no conclusion is given.



This sheet is where you build a rule for a case

		name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
run case	**cornerstone**																	
	new case	boar	1	0	0	1	0	0		1	0	4	1	big			mammal	
	new rule	*name* *operator* *value*	*hair* *operator* *value*	*feathers* *operator* *value*	*eggs* *operator* *value*	*milk* *operator* *value*	*airborne* *operator* *value*	*acquatic* *operator* *value*	*backbone* *operator* *value*	*breathes* *operator* *value*	*fins* *operator* *value*	*no of legs* *operator* *value*	*tail* *operator* *value*	*catsize* *operator* *value*	*where* *operator* *value*	*type* *operator* *value*	*target* *operator* *value*	*conclusion* ***enter new concl***

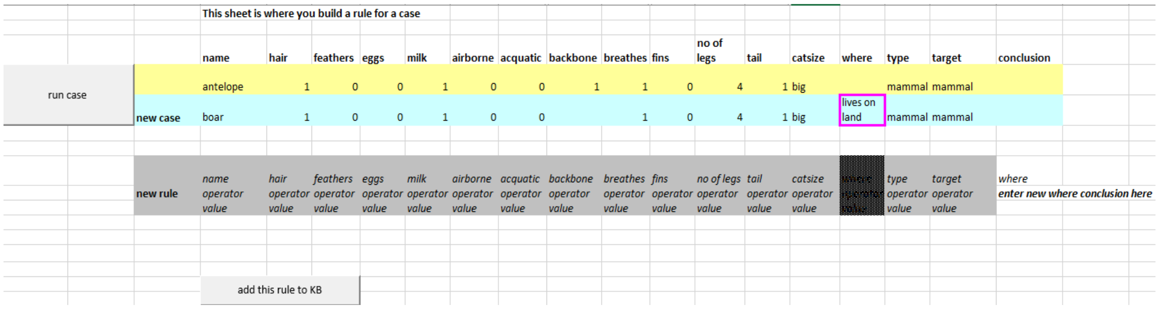
Each time **run case** is clicked a conclusion will appear, with a red border. The next two screen shots show the two conclusions made for this case added one by one.



This sheet is where you build a rule for a case

	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
	antelope	1	0	0	1	0	0	1	1	0	4	1	big		mammal	mammal	
new case	boar	1	0	0	1	0	0		1	0	4	1	big		mammal	mammal	
new rule	*name operator value*	*hair operator value*	*feathers operator value*	*eggs operator value*	*milk operator value*	*airborne operator value*	*acquatic operator value*	*backbone operator value*	*breathes operator value*	*fins operator value*	*no of legs operator value*	*tail operator value*	*catsize operator value*	*where operator value*	*type operator value*	*target operator value*	*type* ***enter new type conclusion here***

When the **run case** button is clicked again the conclusion <u>mammal</u> is the red-bordered cell is assigned permanently to the case, in the same way as piece of data, and cannot be changed by inference. When the **run case** button is clicked again the next conclusion is made

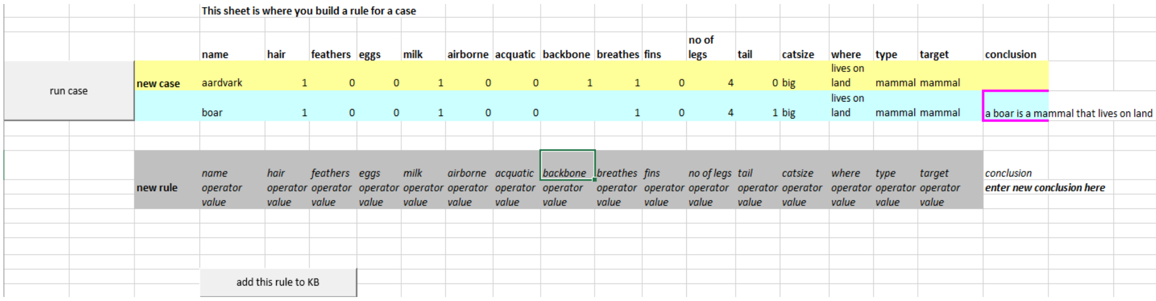


This sheet is where you build a rule for a case

	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
	antelope	1	0	0	1	0	0	1	1	0	4	1	big		mammal	mammal	
new case	boar	1	0	0	1	0	0		1	0	4	1	big	lives on land	mammal	mammal	
new rule	*name operator value*	*hair operator value*	*feathers operator value*	*eggs operator value*	*milk operator value*	*airborne operator value*	*acquatic operator value*	*backbone operator value*	*breathes operator value*	*fins operator value*	*no of legs operator value*	*tail operator value*	*catsize operator value*	*where operator value*	*type operator value*	*target operator value*	*where* ***enter new where conclusion here***

The two conclusions in the examples above are for two extra variables, **type** and **where,** and in this case a boar is of the type <u>animal</u> that <u>lives on land</u>. At any stage when a conclusion has just been given a rule can be added to change that conclusion in the normal SCRDR fashion. The operator and value fields below the conclusion are greyed out to show that they cannot be used for a rule conditions; other variables have to be used as rule conditions to give a new conclusion for **type** or **where.** When a red border is shown around one of these intermediate conclusions, a rule can be added only to change that conclusion.

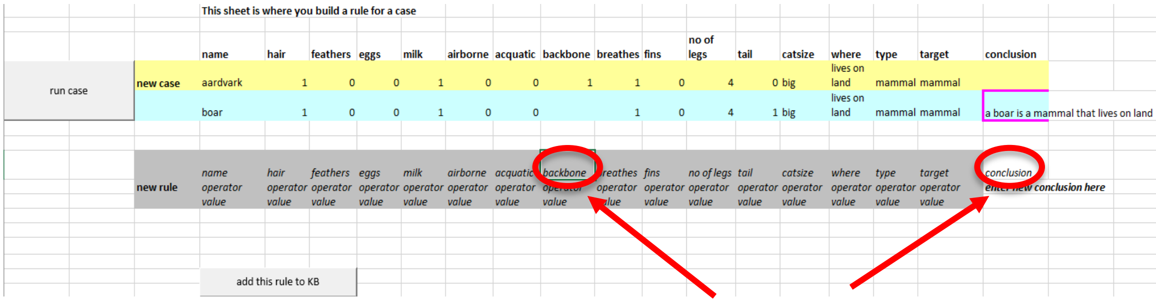
If the **run case** button is clicked until there is a red box around the conclusion attribute, then either a rule for final conclusion can be added, or changed, or a rule for any variable that does not have a value can be added. If the **run case** button is clicked again, the conclusions are cycled through again.



This sheet is where you build a rule for a case

	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
new case	aardvark	1	0	0	1	0	0	1	1	0	4	0	big	lives on land	mammal	mammal	
	boar	1	0	0	1	0	0		1	0	4	1	big	lives on land	mammal	mammal	a boar is a mammal that lives on land
new rule	*name operator value*	*hair operator value*	*feathers operator value*	*eggs operator value*	*milk operator value*	*airborne operator value*	*acquatic operator value*	*backbone operator value*	*breathes operator value*	*fins operator value*	*no of legs operator value*	*tail operator value*	*catsize operator value*	*where operator value*	*type operator value*	*target operator value*	*conclusion* ***enter new conclusion here***

In the example above the conclusion variable has a red border (and a conclusion is given) so can add a rule for the variable **backbone** as the value for this variable is missing. As shown below, we can select **backbone** as a variable for which a rule will assign a conclusion by either double clicking in the cell labelled **backbone** in the grey new rule section of the sheet or by changing <u>conclusion</u> to <u>backbone</u> above **enter new conclusion**. variable for the case will have a red border with the operators below shaded. A rule can then be added to give a **backbone** conclusion

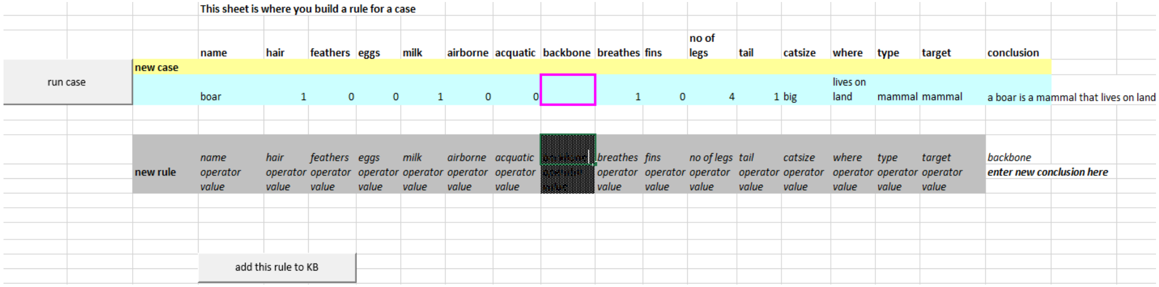


This sheet is where you build a rule for a case

		name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
run case	new case	aardvark	1	0	0	1	0	0	1	1	0	4	0	big	lives on land	mammal	mammal	
		boar	1	0	0	1	0	0		1	0	4	1	big	lives on land	mammal	mammal	a boar is a mammal that lives on land
	new rule	*name* *operator* *value*	*hair* *operator* *value*	*feathers* *operator* *value*	*eggs* *operator* *value*	*milk* *operator* *value*	*airborne* *operator* *value*	*acquatic* *operator* *value*	*backbone* *operator* *value*	*breathes* *operator* *value*	*fins* *operator* *value*	*no of legs* *operator* *value*	*tail* *operator* *value*	*catsize* *operator* *value*	*where* *operator* *value*	*type* *operator* *value*	*target* *operator* *value*	*conclusion* ***enter new conclusion here***

Double click or edit

After double-clicking or editing, the following is shown. Note that in the conclusion column, the conclusion is now for **backbone.**



This sheet is where you build a rule for a case

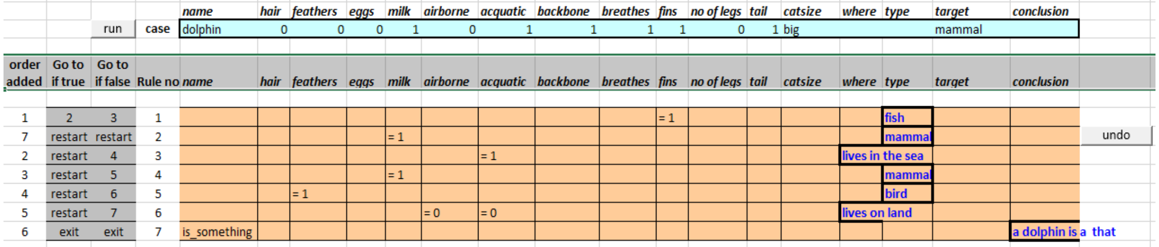
		name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
run case	new case																	
		boar	1	0	0	1	0	0		1	0	4	1	big	lives on land	mammal	mammal	a boar is a mammal that lives on land
	new rule	*name* *operator* *value*	*hair* *operator* *value*	*feathers* *operator* *value*	*eggs* *operator* *value*	*milk* *operator* *value*	*airborne* *operator* *value*	*acquatic* *operator* *value*		*breathes* *operator* *value*	*fins* *operator* *value*	*no of legs* *operator* *value*	*tail* *operator* *value*	*catsize* *operator* *value*	*where* *operator* *value*	*type* *operator* *value*	*target* *operator* *value*	*backbone* ***enter new conclusion here***

At any time before a rule is added, another variable with a missing value can be double-clicked or the conclusion name edited. On the other hand if nothing is clicked or edited, i.e. the final conclusion still has red border, indicating it was the last conclusion made, it is assumed that the new rule applies to the final conclusion. Also at any stage the run button can be clicked to continue cycling through conclusions.

Excel_GRDR rules worksheet

The rules worksheet is essentially the same as for SCRDR and MCRDR, but with three additions

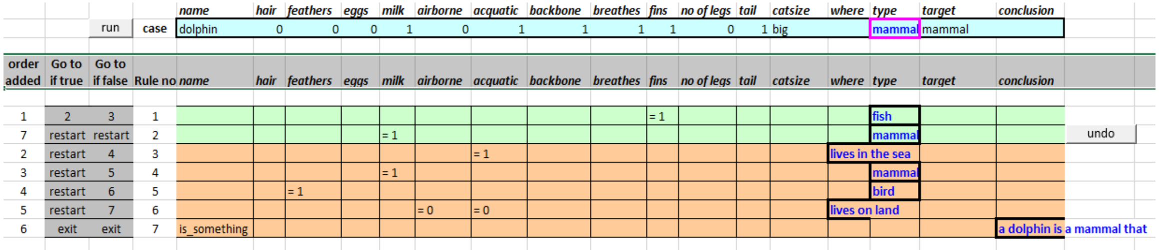
- Rule conclusions are shown with a black border as any variable may be the conclusion of a rule
- The <u>go to if true</u> and <u>go to if false</u> columns which for Excel_SCRDR and Excel_MCRDR show either a <u>rule number</u> or <u>exit</u> , may also include **restart** indicating that inference returns to the first rule
- When **run** is clicked only one conclusion is shown, and the conclusions are cycled through one by one, with the previous conclusions now being considered as data



				name	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*backbone*	*breathes*	*fins*	*no of legs*	*tail*	*catsize*	*where*	*type*	*target*	*conclusion*
		run	case	dolphin	0	0	0	1	0	1	1	1	1	0	1	big			mammal	

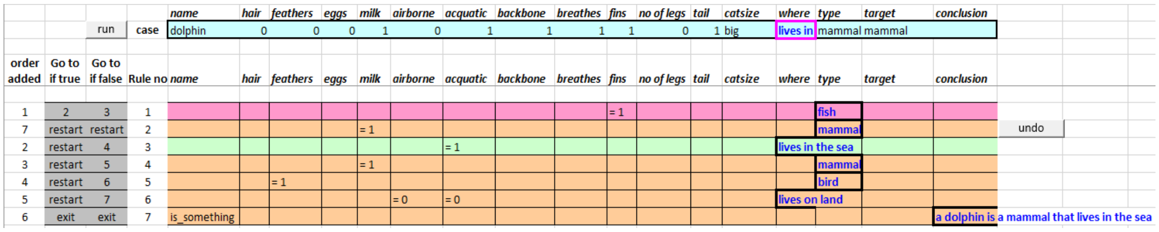
order added	Go to if true	Go to if false	Rule no	*name*	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*backbone*	*breathes*	*fins*	*no of legs*	*tail*	*catsize*	*where*	*type*	*target*	*conclusion*
1	2	3	1										= 1					fish		
7	restart	restart	2					= 1										mammal		
2	restart	4	3							= 1							lives in the sea			
3	restart	5	4					= 1										mammal		
4	restart	6	5			= 1												bird		
5	restart	7	6						= 0	= 0							lives on land			
6	exit	exit	7	is_something																a dolphin is a that

When the case is a dolphin, after **run** is clicked the conclusion **mammal** is given. Note that rule 1 fired giving fish, but that rule 2 (the 7th rule added) is a refinement rule correcting **fish** to **mammal.** Note also that rule 2 goes to restart whether it fires or not. This is because it is only evaluated (and can fail) when rule 1 is true, so that rule knowledge base needs to be re-evaluated with the fact **mammal** now asserted



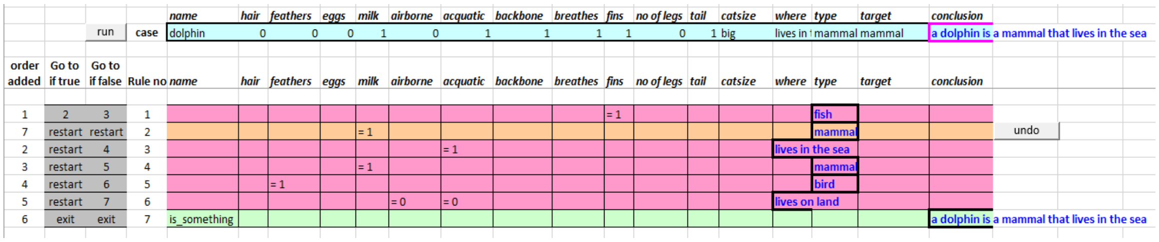
				name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
		run	case	dolphin	0	0	0	1	0	1	1	1	1	0	1	big		mammal	mammal		
order added	Go to if true	Go to if false	Rule no	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
1	2	3	1										= 1					fish			
7	restart	restart	2					= 1										mammal			undo
2	restart	4	3							= 1							lives in the sea				
3	restart	5	4					= 1										mammal			
4	restart	6	5			= 1												bird			
5	restart	7	6						= 0	= 0							lives on land				
6	exit	exit	7	is_something																a dolphin is a mammal that	

When **run** is clicked again, **lives in the sea** is asserted. Note that rule 1 now fails, because there is already a conclusion for **type**, namely **mammal**, and a rule fails that tries to assert a conclusion for variable that already has a value.



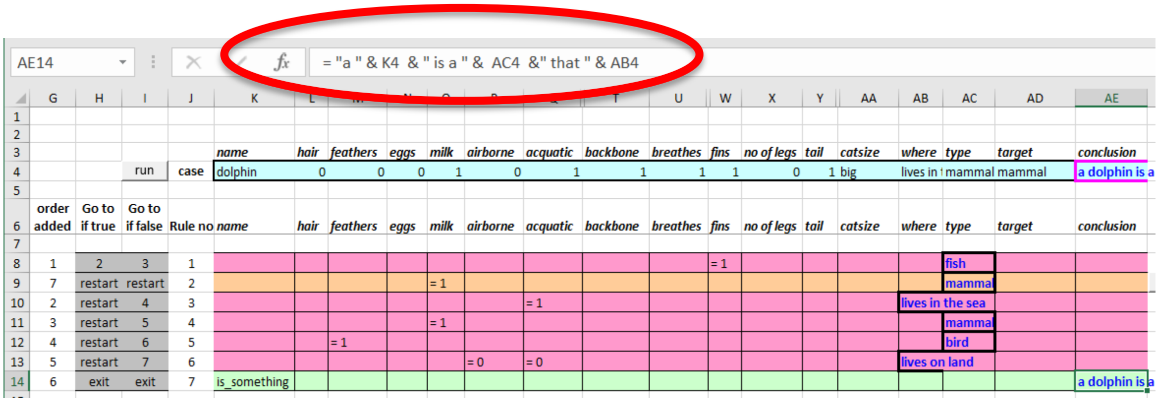
				name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
		run	case	dolphin	0	0	0	1	0	1	1	1	1	0	1	big	lives in	mammal	mammal		
order added	Go to if true	Go to if false	Rule no	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
1	2	3	1										= 1					fish			
7	restart	restart	2					= 1										mammal			undo
2	restart	4	3							= 1							lives in the sea				
3	restart	5	4					= 1										mammal			
4	restart	6	5			= 1												bird			
5	restart	7	6						= 0	= 0							lives on land				
6	exit	exit	7	is_something																a dolphin is a mammal that lives in the sea	

When **run** is clicked again, the final conclusion is given and inference is complete. Clicking **run** again, does not recommence inference with the **where** and **type** variables empty. They have to be manually deleted, or the case rerun from the cases worksheet.



				name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
		run	case	dolphin	0	0	0	1	0	1	1	1	1	0	1	big	lives in	mammal	mammal	a dolphin is a mammal that lives in the sea	
order added	Go to if true	Go to if false	Rule no	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion	
1	2	3	1										= 1					fish			
7	restart	restart	2					= 1										mammal			undo
2	restart	4	3							= 1							lives in the sea				
3	restart	5	4					= 1										mammal			
4	restart	6	5			= 1												bird			
5	restart	7	6						= 0	= 0							lives on land				
6	exit	exit	7	is_something																a dolphin is a mammal that lives in the sea	

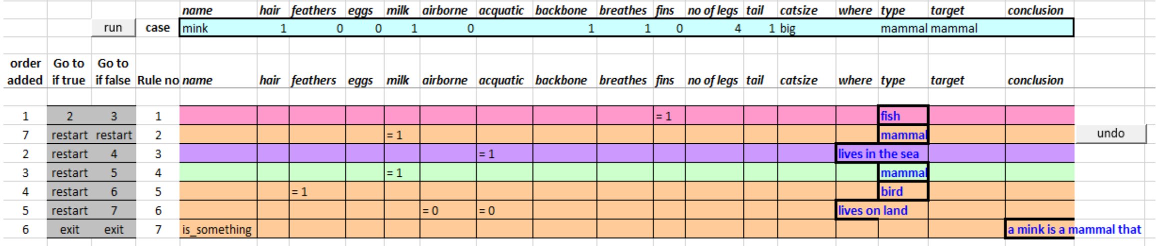
This particular final conclusion is an example of a rule conclusion that is a function. This is why the earlier screen shots show partial conclusions for rule 7, as the data is incomplete. The conclusion field for rule 7 contains the formula circled below



`= "a " & K4  & " is a " &  AC4  &" that " & AB4`

	G	H	I	J	K	L						T	U	W	X	Y	AA	AB	AC	AD	AE
1																					
2																					
3					name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
4			run	case	dolphin	0	0	0	1	0	1	1	1	1	0	1	big	lives in	mammal	mammal	a dolphin is a
5																					
6	order added	Go to if true	Go to if false	Rule no	name	hair	feathers	eggs	milk	airborne	acquatic	backbone	breathes	fins	no of legs	tail	catsize	where	type	target	conclusion
7																					
8	1	2	3	1										= 1					fish		
9	7	restart	restart	2					= 1										mammal		
10	2	restart	4	3							= 1							lives in the sea			
11	3	restart	5	4					= 1										mammal		
12	4	restart	6	5			= 1												bird		
13	5	restart	7	6						= 0	= 0							lives on land			
14	6	exit	exit	7	is_something																a dolphin is a

A further colour is used for the Excel_ GRDR rules worksheet. As for Excel_SCRDR and Excel_MCRDR a rule will fail, if data for a rule condition is missing. With Excel_GRDR rules that fail because of missing are coloured purple. The idea is to alert the user that they may wish to add another rule to so the data is not missing and the previous rule will fire.

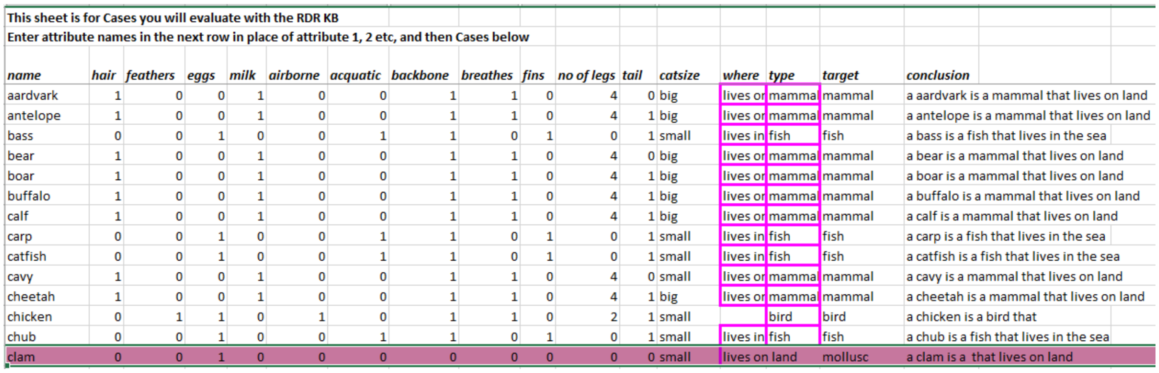


				name	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*backbone*	*breathes*	*fins*	*no of legs*	*tail*	*catsize*	*where*	*type*	*target*	*conclusion*	
		run	case	mink	1	0	0	1	0		1	1	0	4	1	big		mammal	mammal		
order added	Go to if true	Go to if false	Rule no	*name*	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*backbone*	*breathes*	*fins*	*no of legs*	*tail*	*catsize*	*where*	*type*	*target*	*conclusion*	
1	2	3	1										= 1					fish			
7	restart	restart	2					= 1										mammal			undo
2	restart	4	3							= 1							lives in the sea				
3	restart	5	4					= 1										mammal			
4	restart	6	5			= 1												bird			
5	restart	7	6						= 0	= 0							lives on land				
6	exit	exit	7	is_something																a mink is a mammal that	

Excel_GRDR cases worksheet

The Excel_GRDR cases worksheet has an extra feature, that the conclusions for any intermediate rules (not giving the final conclusion) have a red border. Using the rules above and running until an error occurs we get the following. Inference stops on mollusk, because the conclusion does not contain mollusk. Note also that the conclusion for chicken is incomplete, because no rule giving a **where** conclusion fired.

When a case is double-clicked to carry out inference or build a rule etc, these temporary conclusions are removed from the data, so that the original data is rerun. If any of the red-bordered cells are changed, the red border disappears and the change is considered permanent. Also when <u>delete knowledge base</u> is selected, as well as rules and cornerstone cases these temporary conclusions will be removed.



This sheet is for Cases you will evaluate with the RDR KB
Enter attribute names in the next row in place of attribute 1, 2 etc, and then Cases below

name	*hair*	*feathers*	*eggs*	*milk*	*airborne*	*acquatic*	*backbone*	*breathes*	*fins*	*no of legs*	*tail*	*catsize*	*where*	*type*	*target*	*conclusion*
aardvark	1	0	0	1	0	0	1	1	0	4	0	big	lives or	mammal	mammal	a aardvark is a mammal that lives on land
antelope	1	0	0	1	0	0	1	1	0	4	1	big	lives or	mammal	mammal	a antelope is a mammal that lives on land
bass	0	0	1	0	0	1	1	0	1	0	1	small	lives in	fish	fish	a bass is a fish that lives in the sea
bear	1	0	0	1	0	0	1	1	0	4	0	big	lives or	mammal	mammal	a bear is a mammal that lives on land
boar	1	0	0	1	0	0	1	1	0	4	1	big	lives or	mammal	mammal	a boar is a mammal that lives on land
buffalo	1	0	0	1	0	0	1	1	0	4	1	big	lives or	mammal	mammal	a buffalo is a mammal that lives on land
calf	1	0	0	1	0	0	1	1	0	4	1	big	lives or	mammal	mammal	a calf is a mammal that lives on land
carp	0	0	1	0	0	1	1	0	1	0	1	small	lives in	fish	fish	a carp is a fish that lives in the sea
catfish	0	0	1	0	0	1	1	0	1	0	1	small	lives in	fish	fish	a catfish is a fish that lives in the sea
cavy	1	0	0	1	0	0	1	1	0	4	0	small	lives or	mammal	mammal	a cavy is a mammal that lives on land
cheetah	1	0	0	1	0	0	1	1	0	4	1	big	lives or	mammal	mammal	a cheetah is a mammal that lives on land
chicken	0	1	1	0	1	0	1	1	0	2	1	small		bird	bird	a chicken is a bird that
chub	0	0	1	0	0	1	1	0	1	0	1	small	lives in	fish	fish	a chub is a fish that lives in the sea
clam	0	0	1	0	0	0	0	0	0	0	0	small	lives on land		mollusc	a clam is a that lives on land

Notices

Copyright and IP

You are free to make and distribute copies of this software as supplied, but you are not permitted to modify the source code. The reasons the code is not open source are:

1. The purpose of making the demonstration RDR systems available is to try to make clear the ideas behind RDR so a developer can build their own RDR.
2. Everything a develop needs to know to build their own RDR system is visible on the various worksheets.
3. It would be a waste of time and effort to adapt the code here.
    a. Firstly the code is very Excel specific
    b. Secondly, the code is poorly written, so that even if someone wanted to write an Excel-based RDR system with VBA, it would be far more productive to write their own

To try to get a reasonable inference speed when processing a number of cases, compiled versions of the rules, with less Excel-specific coding are found in the KB_compiled module of the Personal Macro Workbook. This is not protected and you are free to inspect or export these for any knowledge bases you might build.

Acknowledgements

You are free to use these programs for any fruitful purpose; however, in any report, media release, or other communication of results from using this software, the use of this software must be acknowledged and where appropriate one or more relevant RDR publications should be cited.

Errors and error messages

Inference errors such as a type difference between the attribute value and rule condition for that attribute will be indicated in the rule conclusion. These are likely to be common in normal dirty real world data

Any errors in user defined function will crash the program, but the error message should make it obvious that you have a problem with one of your functions.

Other errors are certain to occur, as these programs have only been tested mannually. If so I would appreciate it you can send me as much information as you can about the error. Mail: (p.compton@unsw.edu.au)