Innovista Classifier Documentation

The classifier Innovista has developed seeks to simplify the user experience by utilizing UI based menus and algorithms to determine how to classify user input and present the results. It is designed to be used by individuals with minimal knowledge of R code.

The classifier offers the user the option to input a vector containing the inputs or manually type the input in the UI prompt. In both cases, if the input contains *NA*s or strings, the function will stop and issue and error message, listing the index of the *NA*s or strings, enabling the user to easily identify and rectify the issue.

The classifier must be trained on a dataset each time it is run. The user is given the option to select from a list. This list is generated based on matching file names in the current working directory. If the user wishes to use a dataset in a different directory, or one that is not listed, the user can select the 'other' option and input the file name manually. The classifier will then search for the file in the current directory, and if it is not found, prompt the user to change the directory temporarily. This only fires once, so if the user fails to provide the correct file name or directory, the function will fail.

The classifier determines the type of classification algorithm to use based on the structure of the training dataset. For single feature datasets (b and c type datasets) the classifier will use KNN to predict the class of the input value(s). For 8 feature datasets (o type datasets), the classifier will use multinomial logistic regression to predict the class of the input based on the 8 feature values provided.

Due to the size of certain results exceeding max print in R, we opted to automatically assign the results to a variable in the global environment. This variable is named 'results.*' where * = a number. The classifier will always select the highest * and increase it by 1 each time a new result is generated. I.e., if the user manually creates a variable named 'results.7', and the classifier is run for the first time, the variable 'results.8' will be created in the global environment.

If KNN is used to predict classes, and the input is greater than 1, then the function will provide a graph with the count of the predicted classes. If the user also provides a vector of 'true values' (for the sake of testing accuracy), the function will provide 3 graphs which contain the true and false predictions for each class, and a pie chart with the total accuracy.

Calling and Set Up:

1. The input is stored as a variable for ease of use, but manual input is an option as well. Here are the vectors used in the examples.

```
var_b<-0.85154492

var_o<-c(0.8178090360475355, 0.9786911275448799,
         0.003926835,  0.812027417,
         0.005202518,  0.8969672067652326,
         0.048520391,  0.022907260957708543)
var_of<-c(NA, '0.9786911275448799',
          0.003926835, 0.812027417,
          0.005202518, 0.8969672067652326,
          0.048520391, 'a string')


var_c<-read.csv('c0.csv', header = F)
c1<-var_c$V2
c2<-var_c$V1
```

2. The function is called as such, no parameters, no assignment to a variable:

```
classify()
```

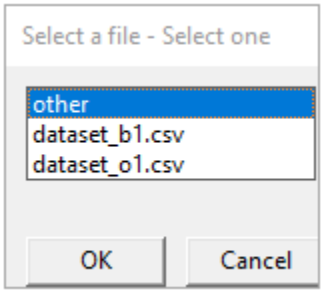
3. The function requires certain packages to run. If these packages are not all installed and loaded, you will be asked to do so. Enter 1 and the function will install and load all packages, 0 and the function will terminate.

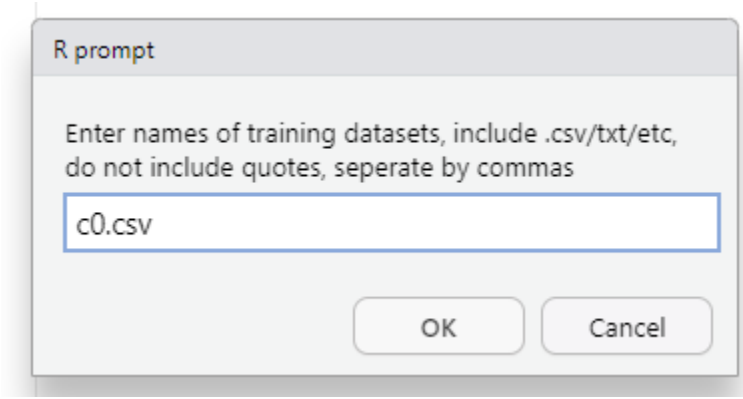
```
> classify()
Install and load required packages? 1 for yes, 0 to exit:
```

4. You must wait untill all packages are loaded and installed before making any more selections in the menu. Your screen might look different if you have never installed these packages before. Depending on your computer and internet connection, this may take some time.

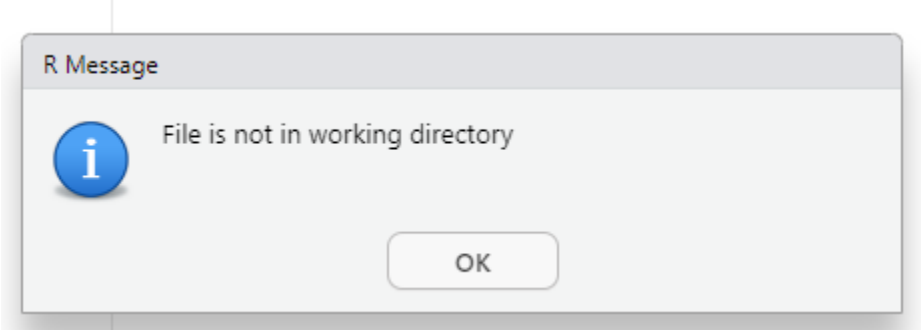
```
Loading required package: nnet
Loading required package: svDialogs
Loading required package: stringr
Loading required package: class
Loading required package: ggplot2
Loading required package: patchwork
```

5. You are prompted to choose from the matching files in your current directory. Since my c*.csv datasets are stored in a subfolder, they do not appear here in the menu. To select a dataset not shown, use 'other'



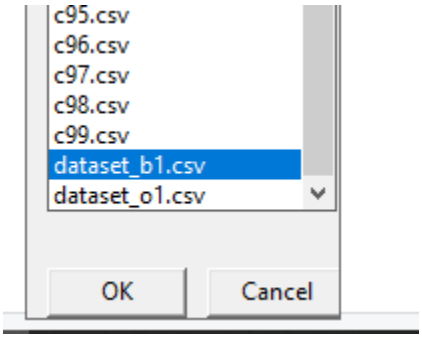


6. If this screen appears, your file is not in the current directory, and you will be prompted to change is temporarily (when the function is done, it will return to what it was set at before running the function). Press ok and use the UI to change your directory.

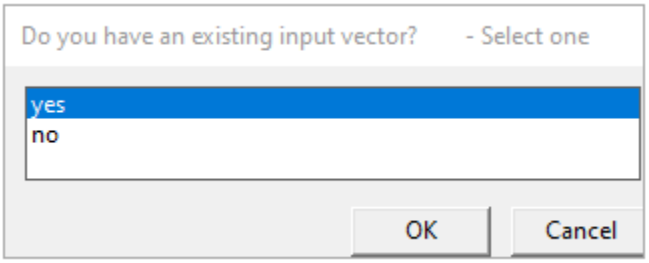


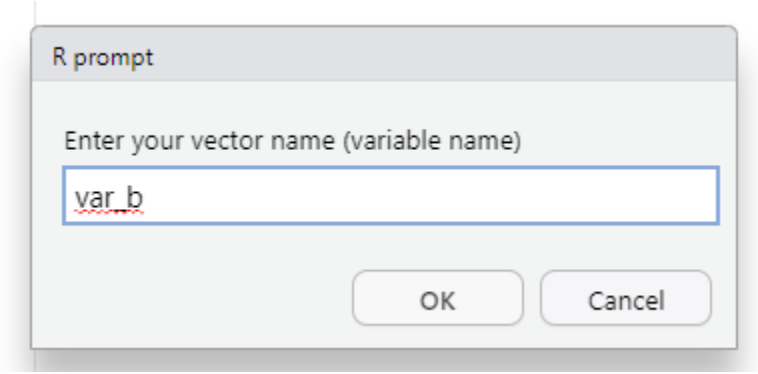
B Type Classification

1. Select the b type dataset, the example shows the default name, use the ‘other’ option if your dataset is under a different name or if it does not show up in the list.

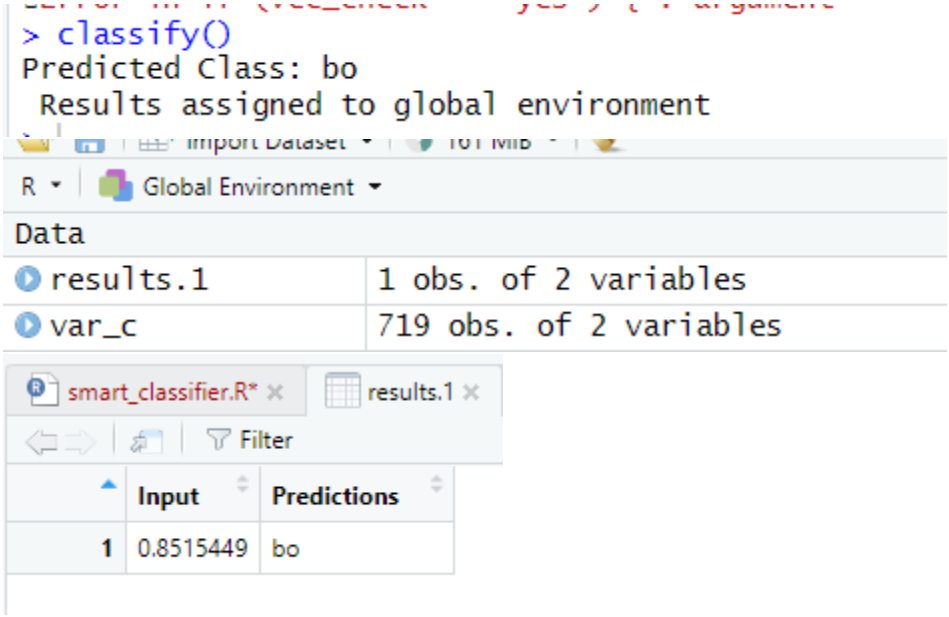


2. For this example, I stored the inputs as variables before, so I will be using this option instead of manual input.



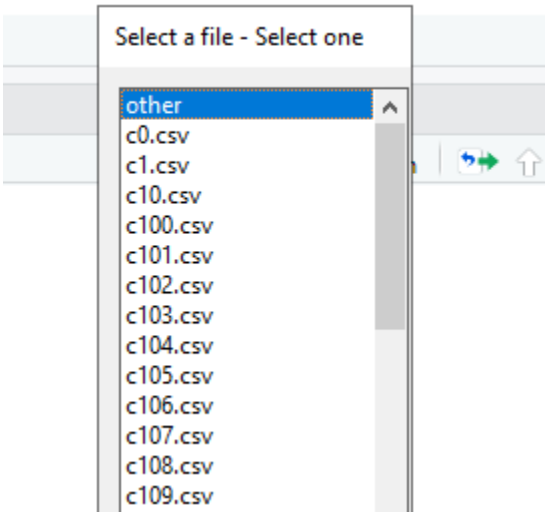


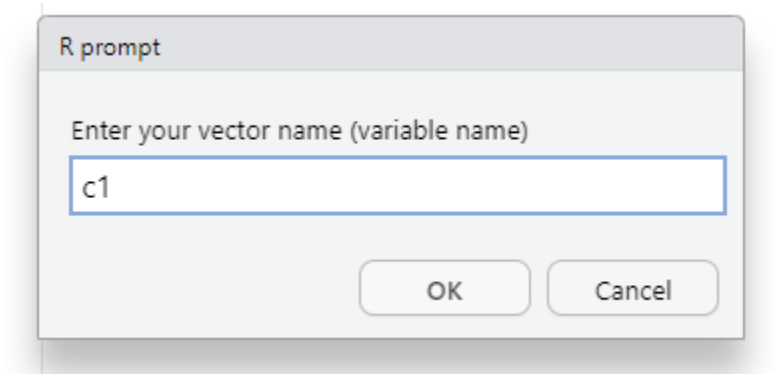
3. A message with the results is given, the results are also stored as a variable in the global environment as a data frame should you need to access them.



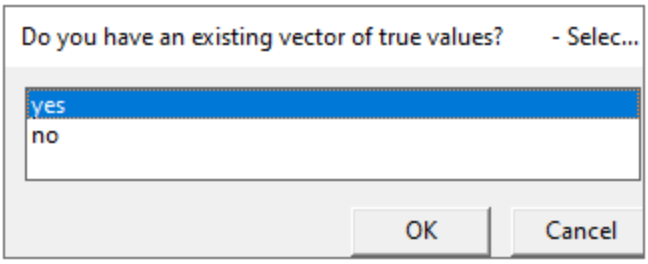
C Type Classification

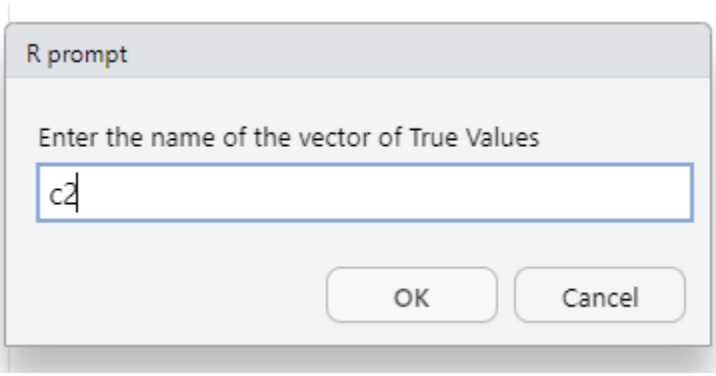
1. Select the dataset you wish the train the algorithm on. In this case I am using the c0 set, and will be using the c0 values column as my input, and the c0 class column as my vector of true values (refer to example vectors image).



2. 

3.


Do you have an existing vector of true values?
- Selec...
yes
no
OK
Cancel


R prompt
Enter the name of the vector of True Values
c2
OK
Cancel

4. A results message is provided, a variable is assigned to the global environment, and graphs are created.

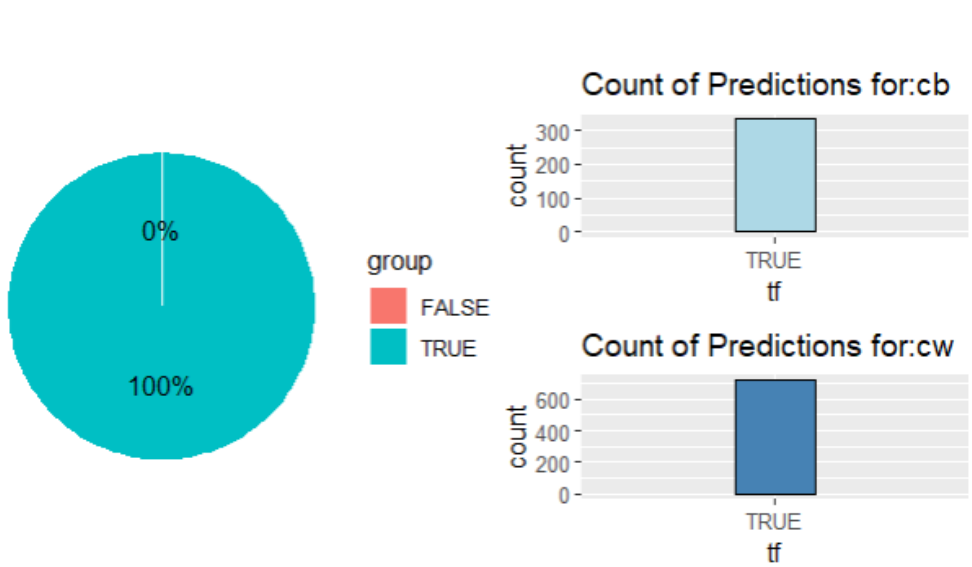
```
> classify()
Accuracy: 100%
 Results assigned to global environment
```

Data	
results.1	1 obs. of 2 variables
results.2	719 obs. of 4 variables
var_c	719 obs. of 2 variables

smart_classifier.R* | results.2 | results.1

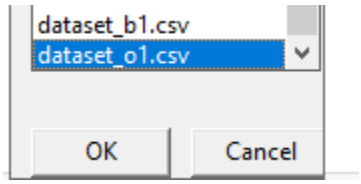
Filter

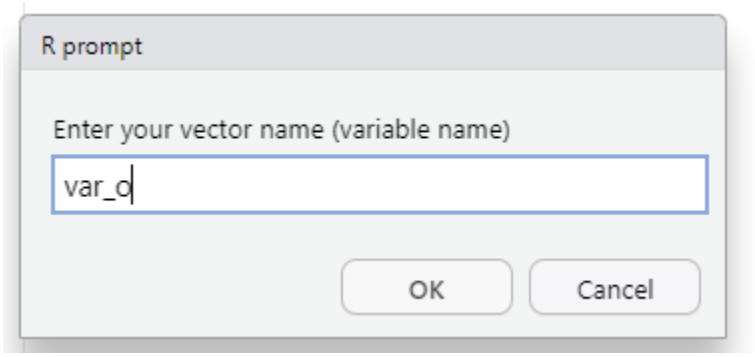
	Input	Predictions	True_Values	tf
1	147.98694	cw	cw	TRUE
2	149.10147	cw	cw	TRUE
3	109.58099	cb	cb	TRUE
4	152.26065	cw	cw	TRUE
5	157.14501	cw	cw	TRUE
6	152.66917	cw	cw	TRUE
7	148.93920	cw	cw	TRUE



O Type Classification

1. Select appropriate training dataset.



2. 

3.
```
> classify()
Results assigned to global environment
 Predicted Class: o6
```

Values	
c1	num [1:719] 148 149 110 152 157 ...
c2	chr [1:719] "cw" "cw" "cb" "cw" "cw" "cw" "cw"
results.3	Factor w/ 10 levels "o0","o1","o2",..: 7

(results.3 is saved to global environment)

- If I were to use vector var_of I would receive this error message:

```
> classify()
Strings or NAs at positions:
 1 8
Error in classify() :
>
```

Recall that var_of had a string at index 8, and an NA at index 1, which caused errors. However, the number entered as a string at index 2 was converted to a numeric value.

```
var_of<-c(NA, '0.9786911275448799',
          0.003926835, 0.812027417,
          0.005202518, 0.8969672067652326,
          0.048520391, 'a string')
```