Summary

Shawn Garbett

2020-05-07

Our team here at Vanderbilt replies quite a bit on exploratory data analysis, and thus summary statistics are an important first step. One of the tools we've brought to bear on this is `tangram` which allows for customized statistical tables. Customization aside, the question is how can I quickly get a set of summary statistics into an Rmd document quickly and painlessly using `tangram`?

Turns out it's easy! Statistics are presentable at the console or the same table layout is available in HTML5, LaTeX or RTF and styleable to your choice of styles.

History

`tangram` was originally developed as a replacement to `Hmisc::summaryM` as we needed to be able to index values and trace them through a Colloboration process. However to do that, a framework was invented that allowed for complete user customization at each step of the pipeline: Parsing, Transformation, and Rendering. Parsing takes an R formula and generates an Abstract Syntax Tree (AST) which contains pieces of a data frame. This in an of itself is a useable piece outside of table generation. Getting this AST generation to match exactly the syntax of R formulas is ongoing. Care has been taken to make sure that no semantics are present in the AST, i.e. the meaning of anything in side a formula is not coded into the AST representation. Semantic meaning is given by the transform specified. The default being `summaryM`. The cross product of each term on each side creates a set of rows and columns that are passed to a transform given by list of list by data type. Data typing is also a customizable part of a transform bundle. The final abstract table object is renderable to a wide variety of formats, and once again this is user customizable. See the trend here, everything is overrideable such that my opinions about how things should be done are not enforced upon the end user. My opinions on statistics or summaries are not a limiting factor of the library.

Finding out the interfaces inside this package has been quite a journey of discovery. An early proof of concept has been going through refactoring, which each effort deleting more and more code and increasing functionality with each pass. The interface between these layers has begun to stablize and refactors in areas aren't spilling over in major ways. To this end, the internal representation is a modified version of Markdown. There's still a lot of work to do to fully realize the vision, but what's available now is useful, and useful is the most important piece of any model or tool.

Quick Summaries!

It by default will take a data frame, and if there exist columns that are of class `cell` it will render directly as a table. Otherwise, it will generate a summary versus an intercept model.

```
tangram(iris, id="iris1", style="nejm")
```

	N	Overall
		(N=150)
Sepal.Length	150	
Median (interquartile range)		5.80 (4.30—5.10—6.40—7.90)

Range		4.30—7.90
Mean±SD		5.84±0.83
Sepal.Width	150	
Median (interquartile range)		3.00 (2.00—2.80—3.31—4.40)
Range		2.00—4.40
Mean±SD		3.06±0.44
Petal.Length	150	
Median (interquartile range)		4.35 (1.00—1.59—5.10—6.90)
Range		1.00—6.90
Mean±SD		3.76±1.77
Petal.Width	150	
Median (interquartile range)		1.30 (0.10—0.30—1.80—2.50)
Range		0.10—2.50
Mean±SD		1.20±0.76
Species	150	
setosa		50/150 (33.333)
versicolor		50/150 (33.333)
virginica		50/150 (33.333)

N is the number of non-missing value. [1]Kruskal-Wallis. [2]Pearson. [3]Wilcoxon.

It's clear that a better breakdown model is possible since `Species` is a factor. One can then switch to a formula interface.

```
tangram(Species ~ Petal.Length + Petal.Width + Sepal.Length + Sepal.Width,
      iris, "iris2", caption="Iris Stats", style="nejm")
```

Table 2: Iris Stats

Table 2: Iris Stats

	N	setosa (N=50)	versicolor (N=50)	virginica (N=50)
Petal.Length	150			
Median (interquartile range)		1.50 (1.40—1.60)	4.35 (4.00—4.60)	5.55 (5.10—5.90)
Range		1.00—1.90	3.00—5.10	4.50—6.90
Petal.Width	150			
Median (interquartile range)		0.200 (0.200—0.300)	1.30 (1.20—1.50)	2.00 (1.80—2.30)
Range		0.100—0.600	1.00—1.80	1.40—2.50
Sepal.Length	150			
Median (interquartile range)		5.00 (4.80—5.20)	5.90 (5.60—6.30)	6.50 (6.20—6.92)
Range		4.30—5.80	4.90—7.00	4.90—7.90
Sepal.Width	150			
Median (interquartile range)		3.40 (3.19—3.70)	2.80 (2.50—3.00)	3.00 (2.80—3.20)
Range		2.30—4.40	2.00—3.40	2.20—3.80

N is the number of non-missing value. [1]Kruskal-Wallis. [2]Pearson. [3]Wilcoxon.

Maybe one really prefers using `dplyr` so let's go with that.

```
iris %>%
  group_by(Species) %>%
  summarise(Mean=mean(Petal.Length),
            Median=median(Petal.Length),
            SD=round(sd(Petal.Length),3) ) %>%
  tangram("iris3", caption="Petal Length (dplyr example)", style="nejm", id="iris3")
```

Table 3: Petal Length (dplyr example)

Table 3: Petal Length (dplyr example)			
iris3			
setosa	1.462	1.5	0.174
versicolor	4.26	4.35	0.47
virginica	5.552	5.55	0.552