Vignette for the ePort package: Academic report generation for statistics instructors

Xiaoyue Cheng, Dianne Cook, Lindsay Rutter, Amy Froelich

`ePort` version 0.0.1 , 2015-09-17

Contents

[1]This LaTeX vignette document is created using the R function `Sweave` on the R package `ePort`. It is automatically downloaded with the package and can be accessed with the R command `vignette("ePort")`.

Summary

The `ePort` package provides tools for course instructors to generate electronic reports regarding student performance. Instructors can produce reports immediately after homework assignment deadlines, and use them to better understand student performance throughout the teaching semester. The goal is to allow instructors to assess and improve upon their teaching approaches in a fast response cycle.

The tools in this package will be especially beneficial for users who supervise large introductory courses. These courses often consist of multiple topics (groups of learning outcomes) that are taught by multiple instructors across multiple sections (groups of students). To accomodate the various ways that student performance can be examined for such courses, the package can generate various reports that can compare within and between topics and sections.

At its simplest, a report can be generated for one topic and one section. This would allow course coordinators to determine how well a particular section performed on a particular topic.

Reports can also be generated for one topic across multiple sections, with output format that allows course coordinators to quickly determine how well and consistently the multiple sections performed on a particular topic. This could be particularly insightful in cases where discrepancies in student performance are discovered between sections, especially if different instructors and/or teaching methods are being used across the sections.

In addition, reports can be generated for one unit (group of topics), either within one section or between multiple sections. This allows coordinators to assess student performance across all the learning outcomes of the combined topics that form the unit, and to quantify the consistency of students perform across sections.

Both short and long versions of reports can be generated for any of the aforementioned scenarios. Short versions of reports provide brief summarizations of student performance without regard to individual problems, whereas long versions of reports provide detailed summarizations of student performance for each problem in the assignment. Hence, reports can also be used to confirm the suitability of assigned problems. For instance, in some courses, problems that assess the same learning outcome and are intended to be of equal difficulty levels are grouped into a problem set, and each student is assigned a random subset from this set of problems. However, sometimes there may be an unexpected discrepancy in student performance between problems in a given problem set, indicating an unintended discrepancy in the clearness or difficulty level of the problems to which students are randomly assigned. This package will allow users to efficiently find and fix such issues.

Introduction

Installation

R is a open source software project for statistical computing, and can be freely downloaded from the Comprehensive R Archive Network (CRAN) website. The link to contributed documentation on the CRAN website offers practical resources for an introduction to R , in several languages. After downloading and installing R , the installation of additional packages is straightforward. To install the `ePort` package from R , use the command:

```
> install.packages("ePort")
```

The ePort package should now be successfully installed. Next, to render it accessible to the current R session, simply type:

```
> library(ePort)
```

To access help pages with example syntax and documentation for the available functions of the ePort package, please type:

```
> help(package="ePort")
```

To access more detailed information about a specific function in the ePort package, use the following help command on that function, such as:

```
> help(merge_section)
```

The above command will return the help file for the function. The help file often includes freestanding example syntax to illustrate how function commands are executed. In the case of the function, the example syntax is the following three lines, which can be pasted directly into an R session.

```
> # Fill later
```

Preprocessing pipeline

```
> key_htm = "/Users/lindz/ePort/data/KeyFiles/Topic06.Questions.htm"
> # generate a clean answer key with paths to the plots in questions
> refine_key(key_htm)
> # get the path/name of the new answer key
> key_txt = gsub("htm$","txt",key_htm)
> # Get the path for a data file
> datapath = "/Users/lindz/ePort/data/DataFiles/Topic06.AB.csv"
> # Get the path for the learning objective
> LOpath = "/Users/lindz/ePort/data/OutcomeFiles/Topic06.Outcomes.txt"
> rewrite_data(datapath)
> report_routine(key_txt,datafile=datapath,rewrite=FALSE,LOfile=LOpath,knitfile="/Users/lindz/ePort/i
```

```
> #report_routine(key_txt,datapath,rewrite=FALSE,LOfile=LOpath,knitfile="/Users/lindz/ePort/inst/Rnw/
> #report_routine(key_txt,datafile=datapath,rewrite=FALSE,LOfile=LOpath,knitfile="/Users/lindz/ePort/
```

```
> #data(sbGeneal)
> #head(sbGeneal)
> #dim(sbGeneal)
> #str(sbGeneal)
```

Generating Reports

The `ePort` package offers several functions that result in useful reports. Below is a brief introduction to some of the available functions.

One section one topic

The `ePort` package offers several functions that you can use to obtain information for individual vertices. First, the function `isParent()` can return a logical variable to indicate whether or not the second variety is a parent of the first variety.

```
> #isParent("Young","Essex",sbGeneal)
> #isParent("Essex","Young",sbGeneal)
```

We see that "Essex" is a parent of "Young", and not vice-versa. In some cases, you may wish to obtain a complete list of all the parents of a given variety. This can be achieved using the `getParent()` function:

```
> #getParent("Young",sbGeneal)
> #getParent("Tokyo",sbGeneal)
> #getYear("Tokyo", sbGeneal)
```

We learn from this that "Essex" is not the only parent of "Young"; "Young" also has a parent "Davis". We also see that "Tokyo" does not have any documented parents in this dataset, and has an older year of introduction than other varieties we have examined thusfar. Likewise, in other cases, you may wish to obtain a complete list of all the children of a given variety. This can be achieved using the `getChild()` function:

```
> #getChild("Tokyo",sbGeneal)
> #getChild("Ogden",sbGeneal)
```

We find that even though the "Tokyo" variety is a grandparent of the dataset, it only has two children, "Ogden" and "Volstate". However, one of its children, "Ogden", produced Sexprlength(getChild("Ogden",sbGeneal)) children.

If we want to obtain a list that contains more than just one generation past or previous to a given variety, then we can use the `getAncestors()` and `getDescendants()` functions, where we specify the number of generations we wish to view. This will return a data frame to us with the labels of each ancestor or descendant, along with the number of generations each one is from the given variety.

If we only look at one generation of ancestors of the "Young" variety, we should see the same information we did earlier when we used the `getParent()` function of the Young variety.

Multiple section one topic

Say you have a pair of vertices, and you wish to determine the degree of the shortest path between them, where edges represent parent-child relationships. You can accomplish that with the `getDegree()` function.

```
> #getDegree("Tokyo", "Ogden", ig, sbGeneal)
> #getDegree("Tokyo", "Holladay", ig, sbGeneal)
```

One section Multiple topics

Until this point, the vignette has introduced functions that return lists, data frames, and statistics about the genealogical dataset. However, the `ePort` package also contains visualization tools for genealogical datasets. Access to various types of visual plots and diagrams of the lineage can allow genealogical researchers to more efficiently and accurately explore an otherwise complicated data structure. Below, we introduce functions in `ePort` that produce visual outputs of the dataset.

Multiple Sections Multiple Topics

One visualization tool, `plotAncDes()`, allows the user to view the ancestors and descendants of a given variety. The inputted variety is highlighted in the center of the plot, ancestors are displayed to the left of the center, and descendants are displayed to the right of the center. The further left or right from the center, the larger the number of generations that particular ancestor/descendant is from the inputted and centered variety.

As such, this plotting command does not provide visual information about specific years associated with each related variety (as is done in some of the visualization tools introduced later), but it does group all varieties from each generation group onto the same position of the horizontal axis. Here, we specify that we want to plot 5 ancestor generations and 4 descendant generations of the variety "Lee":

Short reports

Brief summarizations

Low-scoring students

Long reports

Individual problem summarizations

Clustering analysis

Conclusions

The `ePort` package offers various plotting tools that can assist those studying genealogical lineages in the data exploration phases. As each plot comes with its advantages and disadvantages, we recommend for users to explore several of the available visualization tools.

This vignette briefly introduced some of the capabilities of the `ePort` package. Inevitably, new approaches will necessitate new features in subsequent versions and might reveal unforeseen bugs. Please send comments, suggestions, questions, and bug reports to `lrutter@iastate.edu`.