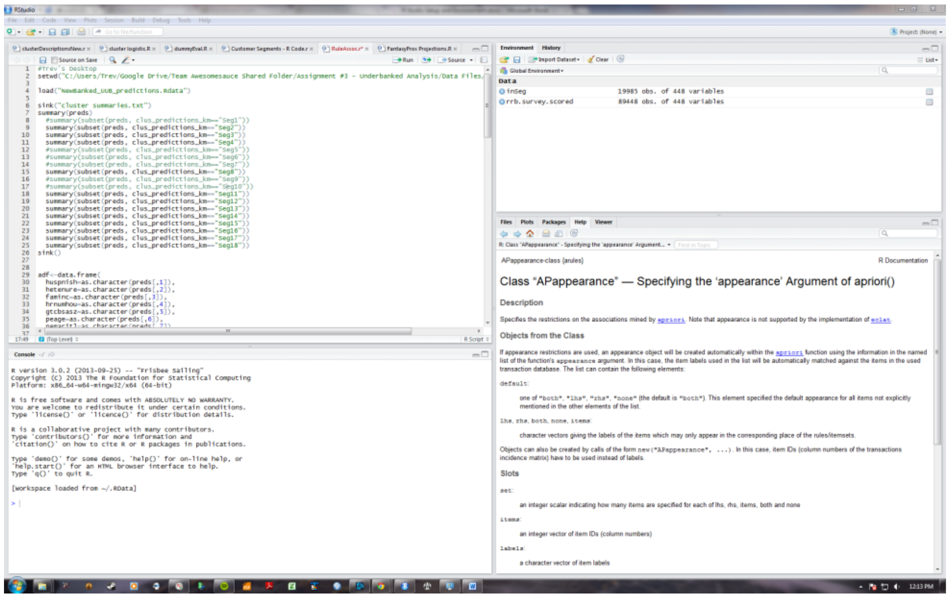
R Studio Feature Introduction

MSPA 401 – Introduction to Statistics

Northwestern University

The RStudio environment consists of multiple windows that help make development and analysis in R easier to come to grips with. The default display (with a file loaded) features four windows. There are tabs on each window that change the functionality of those windows. The placement of each window, and the tabs on each, can be changed from the Tools > Global Options menu. The pane layout area controls what goes where. By default, the code window is in the top left quadrant of the layout, the console window is in the bottom left, the environmental window is in the top right, and the file menu is in the bottom right.



```
#Trev's Desktop
setwd("C:/Users/Trev/Google Drive/Team Awesomesauce Shared Folder/Assignment #3 - Underbanked Analysis/Data Files/

load("NewBanked_UUB_predictions.Rdata")

sink("cluster summaries.txt")
summary(preds)
  #summary(subset(preds, clus_predictions_km=="Seg1"))
  summary(subset(preds, clus_predictions_km=="Seg2"))
  summary(subset(preds, clus_predictions_km=="Seg3"))
  summary(subset(preds, clus_predictions_km=="Seg4"))
  #summary(subset(preds, clus_predictions_km=="Seg5"))
  #summary(subset(preds, clus_predictions_km=="Seg6"))
  #summary(subset(preds, clus_predictions_km=="Seg7"))
  summary(subset(preds, clus_predictions_km=="Seg8"))
  #summary(subset(preds, clus_predictions_km=="Seg9"))
  #summary(subset(preds, clus_predictions_km=="Seg10"))
  summary(subset(preds, clus_predictions_km=="Seg11"))
  summary(subset(preds, clus_predictions_km=="Seg12"))
  summary(subset(preds, clus_predictions_km=="Seg13"))
  summary(subset(preds, clus_predictions_km=="Seg14"))
  summary(subset(preds, clus_predictions_km=="Seg15"))
  summary(subset(preds, clus_predictions_km=="Seg16"))
  summary(subset(preds, clus_predictions_km=="Seg17"))
  summary(subset(preds, clus_predictions_km=="Seg18"))
sink()


adf<-data.frame(
  huspnish=as.character(preds[,1]),
  hetenure=as.character(preds[,2]),
  faminc=as.character(preds[,3]),
  hrnumhou=as.character(preds[,4]),
  gtcbsasz=as.character(preds[,5]),
  peage=as.character(preds[,6]),
```

```
R version 3.0.2 (2013-09-25) -- "Frisbee Sailing"
Copyright (C) 2013 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from ~/.RData]

> 
```

Data	
inSeg	19985 obs. of 448 variables
rrb.survey.scored	89448 obs. of 448 variables

APappearance-class {arules} R Documentation

Class "APappearance" — Specifying the 'appearance' Argument of apriori()

Description

Specifies the restrictions on the associations mined by apriori. Note that appearance is not supported by the implementation of eclat.

Objects from the Class

If appearance restrictions are used, an appearance object will be created automatically within the apriori function using the information in the named list of the function's appearance argument. In this case, the item labels used in the list will be automatically matched against the items in the used transaction database. The list can contain the following elements:

default:

one of "both", "lhs", "rhs", "none" (the default is "both"). This element specified the default appearance for all items not explicitly mentioned in the other elements of the list.

lhs, rhs, both, none, items:

character vectors giving the labels of the items which may only appear in the corresponding place of the rules/itemsets.

Objects can also be created by calls of the form new("APappearance", ...). In this case, item IDs (column numbers of the transactions incidence matrix) have to be used instead of labels.

Slots

set:

an integer scalar indicating how many items are specified for each of lhs, rhs, items, both and none

items:

an integer vector of item IDs (column numbers)

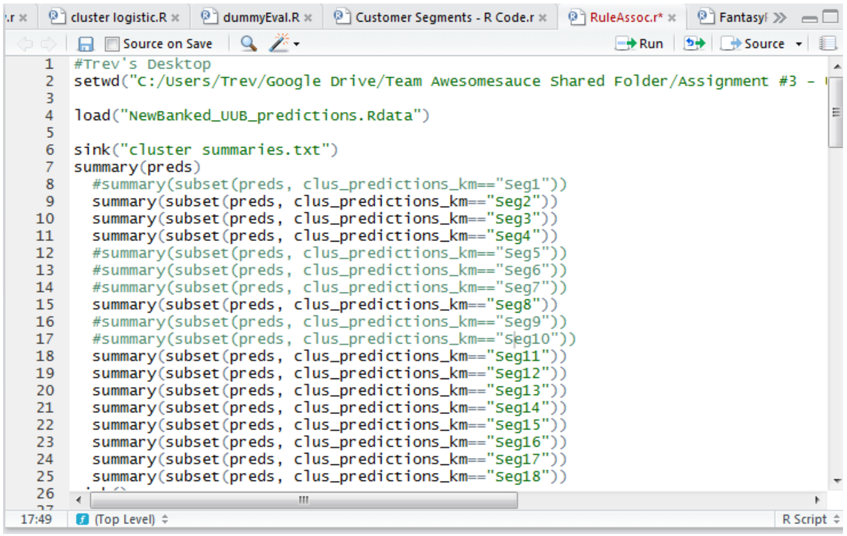
labels:

a character vector of item labels

The Code Window

The code window displays the currently selected file that is being edited. Typically R files are what are edited here, but other files can be modified here, as well. Down the road, you may hear about markdown, sweave, and other output languages. Those can all be edited in R Studio. Dataframes can also be previewed in this area. Double clicking on a data frame (and a few other object types) will provide a grid view of that data in a new tab in this same area.

The code window has several features that are quite handy. Auto completion of any



```
#Trev's Desktop
setwd("C:/Users/Trev/Google Drive/Team Awesomesauce Shared Folder/Assignment #3 -

load("NewBanked_UUB_predictions.Rdata")

sink("cluster summaries.txt")
summary(preds)
  #summary(subset(preds, clus_predictions_km=="Seg1"))
  summary(subset(preds, clus_predictions_km=="Seg2"))
  summary(subset(preds, clus_predictions_km=="Seg3"))
  summary(subset(preds, clus_predictions_km=="Seg4"))
  #summary(subset(preds, clus_predictions_km=="Seg5"))
  #summary(subset(preds, clus_predictions_km=="Seg6"))
  #summary(subset(preds, clus_predictions_km=="Seg7"))
  summary(subset(preds, clus_predictions_km=="Seg8"))
  #summary(subset(preds, clus_predictions_km=="Seg9"))
  #summary(subset(preds, clus_predictions_km=="Seg10"))
  summary(subset(preds, clus_predictions_km=="Seg11"))
  summary(subset(preds, clus_predictions_km=="Seg12"))
  summary(subset(preds, clus_predictions_km=="Seg13"))
  summary(subset(preds, clus_predictions_km=="Seg14"))
  summary(subset(preds, clus_predictions_km=="Seg15"))
  summary(subset(preds, clus_predictions_km=="Seg16"))
  summary(subset(preds, clus_predictions_km=="Seg17"))
  summary(subset(preds, clus_predictions_km=="Seg18"))
```

command, control syntax, or variable name can be initiated by pressing tab while typing the R code. This can be quite useful to speed up coding, avoid typos, or to discover a function name that might have slipped your head.

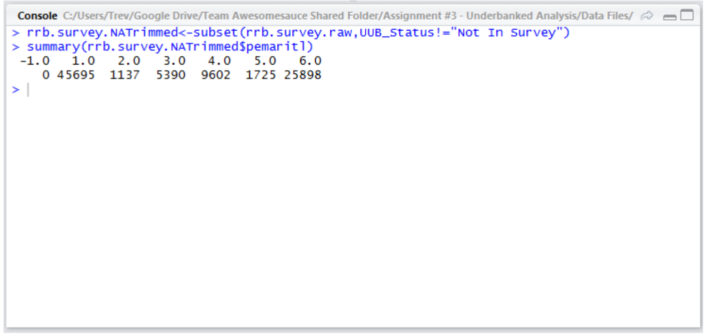
Code execution can also be performed from the code widow. The 'Run' button will execute any code that has been highlighted (click and drag). CTRL + R will also perform this action. The code will be executed with the current environment. That is, the variable values, packages, and data that is currently loaded in the environment. This can be quite an advantage when debugging, as you can highlight variable names or blocks of code in your bigger file and observe the return values prior to storing them in their own variables. Additionally, 'executing' a variable in the script with nothing else will return the value of its string function. If something has gone wrong, checking the current variable values is always a good early debugging step.

One last tip for the code window is code commenting/uncommenting. CTRL+Shift + C will toggle commenting or uncommenting an entire block of code at a time, and can be a real time saver.

There are a lot of other code editing features in and for this window. The edit menu, in particular, is full of them. If you have familiarity with source control, there are a few options that might be worth looking into, but are beyond the scope of this beginner overview.

Immediate/Console Window

The console is basically an output window for what R is doing. Commands can be typed and run here. This is exactly like executing code that is highlighted and 'run' with the run button, or CTRL + R. It will execute against the current environment and any console output will be displayed here. The history of the commands that were entered can be retrieved by using the up and down arrows. The main benefit to entering code here, as opposed to in the script itself, is for debugging. Basically, you can 'explore' without cluttering up any files. That is all personal taste, however. The console window will also show the command being executed by any other window. This can be handy to get the syntax from loading a file, library, exporting images, or any other bit of functionality that RStudio makes easier for us!

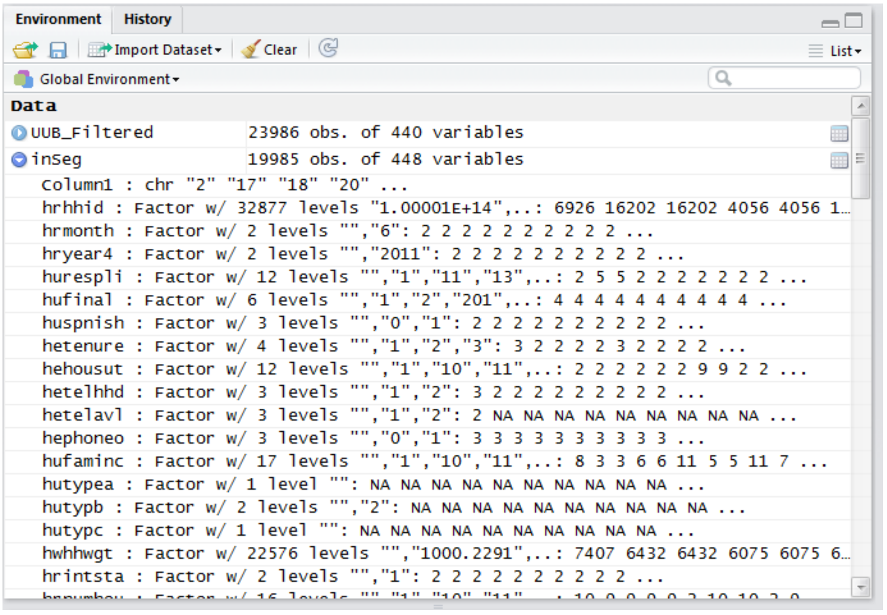

Console C:/Users/Trev/Google Drive/Team Awesomesauce Shared Folder/Assignment #3 - Underbanked Analysis/Data Files/

```
> rrb.survey.NATrimmed<-subset(rrb.survey.raw,UUB_Status!="Not In Survey")
> summary(rrb.survey.NATrimmed$pemaritl)
 -1.0   1.0   2.0   3.0   4.0   5.0   6.0 
    0 45695  1137  5390  9602  1725 25898 
> 
```

Environment Window

The environment window features an object explorer. Any object (even down to a single entity vector) will be displayed here, if you save it as a variable with the <- syntax. The object itself will be represented by the value returned by running dim() on it. If you expand the > next to the object, the objects str() value will be returned. This won't show you everything all of the time, but it can give some nice information about what is contained in objects and what nested objects might be available, as well.

The environment window also has a file import utility that can be used to launch a wizard that will help with loading various file formats into the working environment. Additionally, there is a native file format that R uses to store its environment (typically .RData). The disk and folder icons can be used to save and load to and from this format, respectively. This is really handy if you have performed a large amount of data transformations after data have been loaded, or if you have a plethora of one-off variables that you'd like to keep around. Storing the entire environment and reloading it can be much easier that reloading a file (or files) and transforming it all over again.



```
Data
UUB_Filtered          23986 obs. of 440 variables
inSeg                 19985 obs. of 448 variables
   Column1 : chr "2" "17" "18" "20" ...
   hrhhid : Factor w/ 32877 levels "1.00001E+14",..: 6926 16202 16202 4056 4056 1…
   hrmonth : Factor w/ 2 levels "","6": 2 2 2 2 2 2 2 2 2 2 ...
   hryear4 : Factor w/ 2 levels "","2011": 2 2 2 2 2 2 2 2 2 2 ...
   hurespli : Factor w/ 12 levels "","1","11","13",..: 2 5 5 2 2 2 2 2 2 2 ...
   hufinal : Factor w/ 6 levels "","1","2","201",..: 4 4 4 4 4 4 4 4 4 4 ...
   huspnish : Factor w/ 3 levels "","0","1": 2 2 2 2 2 2 2 2 2 2 ...
   hetenure : Factor w/ 4 levels "","1","2","3": 3 2 2 2 2 3 2 2 2 2 ...
   hehousut : Factor w/ 12 levels "","1","10","11",..: 2 2 2 2 2 2 9 9 2 2 ...
   hetelhhd : Factor w/ 3 levels "","1","2": 3 2 2 2 2 2 2 2 2 2 ...
   hetelavl : Factor w/ 3 levels "","1","2": 2 NA NA NA NA NA NA NA NA NA ...
   hephoneo : Factor w/ 3 levels "","0","1": 3 3 3 3 3 3 3 3 3 3 ...
   hufaminc : Factor w/ 17 levels "","1","10","11",..: 8 3 3 6 6 11 5 5 11 7 ...
   hutypea : Factor w/ 1 level "": NA NA NA NA NA NA NA NA NA NA ...
   hutypb : Factor w/ 2 levels "","2": NA NA NA NA NA NA NA NA NA NA ...
   hutypc : Factor w/ 1 level "": NA NA NA NA NA NA NA NA NA NA ...
   hwhhwgt : Factor w/ 22576 levels "","1000.2291",..: 7407 6432 6432 6075 6075 6…
   hrintsta : Factor w/ 2 levels "","1": 2 2 2 2 2 2 2 2 2 2 ...
```

Finally, the clear button will remove everything that has been loaded into the current environment. In this way, it can serve as a sort of 'reset' button on an entire analysis solution. The other tab on this window will provide the history of comments that have been executed. Those commands can be inserted into the current working file with the insert button.

File/Plots/Packages/Help/Viewer Window

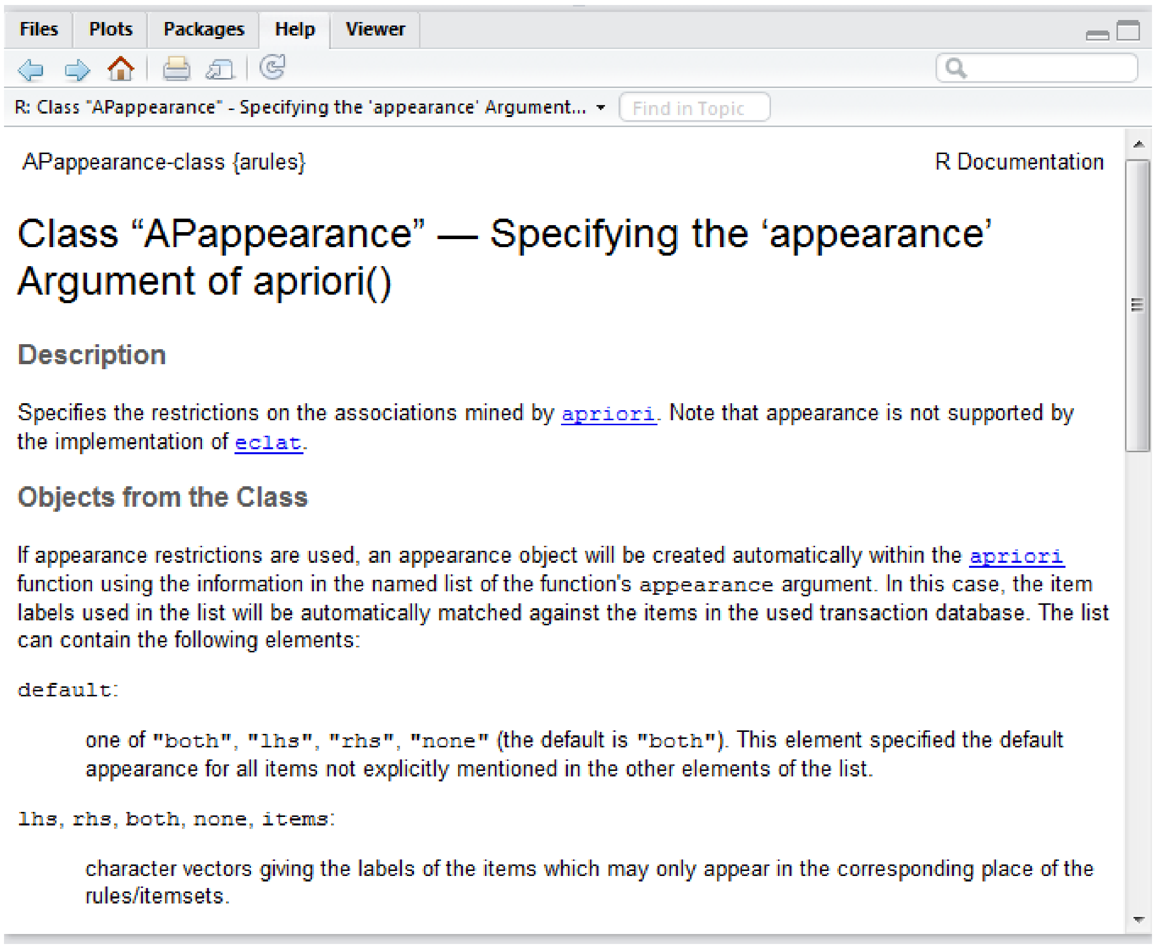
This last window is sort of a catch-all, but it is worth getting familiar with its varied functionality.

The files tab can be used to load files, much like the import or folder icon from the environment window. This differs, however, in that there is a file tree immediately available. I have used this particular window to have the system generate an error message when I point to a file that it can't load. The useful bit about this is that the error message will contain the directory it tried to load from within the error message. This can be a nice shortcut when a full file path is needed within a script to do things like load a file, in script, outside of the working directory. The 'more' dropdown can also be used to set the working directory to the directory currently selected in the file browser.

The Plots tab is the display device when R's various graphical capabilities are employed. The plot window keeps a history of those plots, which can be cleared with the 'Clear All' button. Images can be zoomed or exported into various formats from this tab, as well.

The packages tab is very handy for package management. The packages that are available can be loaded by checking the box next to them. If a package is not available to your environment, the 'Install' button will look on the internet (CRAN, specifically) for a package with the name you search for.

Lastly, the help functionality ties to the help tab here. The ? syntax will bring the appropriate help page in this window. For example ?lm will display the linear model help page, and provide the functions, usage, and capabilities of that object. If you are unsure of an object name, the ?? syntax can look for a word anywhere in help, but it can be a bit cluttered. Use it as a last resort!


Files
Plots
Packages
Help
Viewer
R: Class "APappearance" - Specifying the 'appearance' Argument...
Find in Topic

APappearance-class {arules} R Documentation

Class "APappearance" — Specifying the 'appearance' Argument of apriori()

Description

Specifies the restrictions on the associations mined by `apriori`. Note that appearance is not supported by the implementation of `eclat`.

Objects from the Class

If appearance restrictions are used, an appearance object will be created automatically within the `apriori` function using the information in the named list of the function's `appearance` argument. In this case, the item labels used in the list will be automatically matched against the items in the used transaction database. The list can contain the following elements:

`default`:

one of `"both"`, `"lhs"`, `"rhs"`, `"none"` (the default is `"both"`). This element specified the default appearance for all items not explicitly mentioned in the other elements of the list.

`lhs`, `rhs`, `both`, `none`, `items`:

character vectors giving the labels of the items which may only appear in the corresponding place of the rules/itemsets.