Association Rules Mining

Professor Jeff Saltz
Professor Jeff Stanton

Copyright 2021: Jeffrey Saltz and Jeffrey Stanton; please do not upload.

School of Information Studies
SYRACUSE UNIVERSITY
ischool.syr.edu

Summary of Previous Learning

What you should know and be able to do at this point :

1. List major skills needed by data scientists and describe the development of a DS project with domain analysis, SMEs, data and modeling
2. Explain and use a data frame in R and various diagnostics for variables and data structures; describe and use multiple strategies for accessing external data from R; use SQL facilities from within R; automate with functions
3. Define and calculate the most common descriptive statistics; describe the effects of randomness on sampling; create and interpret a sampling distribution including defining the law of large numbers and the central limit theorem
4. Use plot and ggplot to visualize data and create maps
5. Create and interpret a multiple regression model using lm()
6. Use supervised learning with SVM and Trees

Objectives for the Week

- Basic concepts of data mining
- Use case of "**association rules mining**"
- Develop representative data mining R code

An Example of Unsupervised Learning: Association Rules Mining

School of Information Studies
SYRACUSE UNIVERSITY
ischool.syr.edu

What is Association Rules Mining?

Association rules:

- Association rules are if/then statements where the "if" refers to a set and the "then" refers to another set (often a single item)
- The rules help uncover relationships among elements or attributes that occur in sets

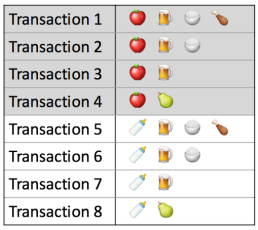
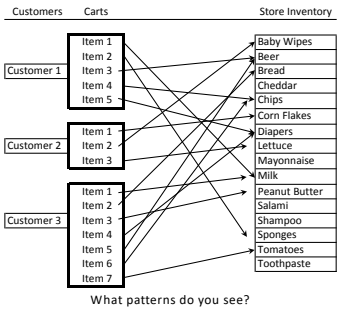


Image credit: kdnuggets

An example:

"There's a 50% chance that a customer will buy an apple. If a customer buys an apple, there's a 75% chance they will also buy a beer."

A Classic Example



What patterns do you see?

Copyright 2019, Jeff Stanton and Jeff Saltz

Sparse Matrix Representation (Incomplete)

	Wipes	Beer	Diapers	Lettuce	Flakes	Milk
Cart 1		1	1	1		1
Cart 2	1			1		1
Cart 3		1	1		1	1

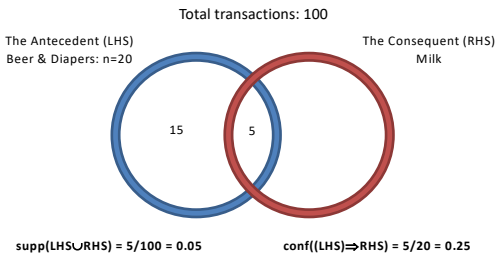
Itemset Representation

2,3,4,6
1,4,6
2,3,5,6

Key Concepts: Support & Confidence

- Two key statistical quantities:
  - **Support** is the proportion of times that the union of LHS and RHS occur versus the total number of transactions
  - **Confidence** is the probability of finding the RHS in transactions that also contain the LHS; technically, conf(LHS⇒RHS) = supp(LHS∪RHS/supp(LHS)
  - Pairings may have high support and high confidence but still not be interesting (e.g., milk and bread)
  - Pairings may have low support, but still be interesting if they have high confidence

Support and Confidence as a Venn Diagram



Copyright 2019, Jeff Stanton and Jeff Saltz

Measures of Interestingness

- **Lift** is the ratio of support for LHS:RHS together versus if they were independent:
  lift(LHS⇒RHS) = supp(LHS∪RHS)/(supp(LHS)supp(RHS))
- Lift can exceed one, you can think of it as an odds ratio
- **Conviction** is the ratio of "incorrect" predictions:
  (1- supp(RHS))/(1 – conf(LHS⇒RHS)) and can also be interpreted as an odds ratio
  - For example, if the conviction for {beer;diapers} ⇒{milk} is 1.5, then the rule occurring is 50% more than would be expected purely by chance
- There are dozens of other measures of interestingness, see:

  http://michael.hahsler.net/research/association_rules/measures.html

A Worked Example: Association Rules Mining

ischool.syr.edu
School of Information Studies
SYRACUSE UNIVERSITY

The DM Process – Data Prep

```
install.packages("arules") # Implements the apriori algorithm
library("arules")
data (Groceries) # Load data into memory
myGroc <- Groceries # Make a copy for safety
str(myGroc) # What is the structure?
```

A sparse matrix: An efficient structure that R uses to hold large, but empty matrices

```
Formal class 'transactions' [package "arules"] with 3 slots
  ..@ data       :Formal class 'ngCMatrix' [package "Matrix"] with 5 slots
  .. .. ..@ i       : int [1:43367] 13 60 69 78 14 29 98 24 15 29 ...
  .. .. ..@ p       : int [1:9836] 0 4 7 8 12 16 21 22 27 28 ...
  .. .. ..@ Dim     : int [1:2] 169 9835
  .. .. ..@ Dimnames:List of 2
  .. .. .. ..$ : NULL
  .. .. .. ..$ : NULL
  .. .. ..@ factors : list()
  ..@ itemInfo   :'data.frame':        169 obs. of  3 variables:
  .. ..$ labels: chr [1:169] "frankfurter" "sausage" "liver loaf" "ham" ...
  .. ..$ level2: Factor w/ 55 levels "baby food","bag...: 44 44 44 44 44 44 44 42 42 41 ...
  .. ..$ level1: Factor w/ 10 levels "canned food",...: 6 6 6 6 6 6 6 6 6 6 ...
  ..@ itemsetInfo:'data.frame':        0 obs. of  0 va...es
```

A list of labels for the items included in this data set

Copyright 2019, Jeff Stanton and Jeff Saltz

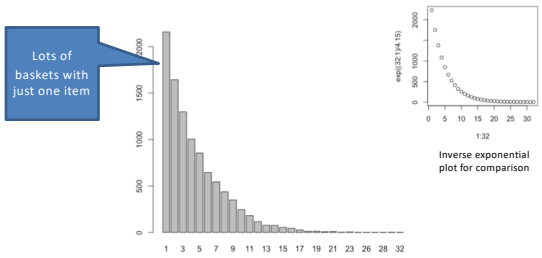
summary(myGroc)

```
transactions as itemMatrix in sparse format with
 9835 rows (elements/itemsets/transactions) and
 169 columns (items) and a density of 0.02609146

most frequent items:
      whole milk other vegetables       rolls/buns             soda           yogurt
            2513              1903             1809             1715             1372
          (Other)
            34055

element (itemset/transaction) length distribution:
sizes
   1    2    3    4    5    6    7    8    9   10   11   12   13   14   15   16   17   18   19
2159 1643 1299 1005  855  645  545  438  350  246  182  117   78   77   55   46   29   14   14
  20   21   22   23   24   26   27   28   29   32
   9   11    4    6    1    1    1    1    3    1

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.000   2.000   3.000   4.409   6.000  32.000

includes extended item information - examples:
       labels  level2           level1
1 frankfurter sausage meat and sausage
2     sausage sausage meat and sausage
3  liver loaf sausage meat and sausage
```

9835 different "baskets" of items

Higher level categories included

barplot(summary(myGroc)@lengths)



Lots of baskets with just one item

Inverse exponential plot for comparison

Exploring the Data

```
myGrocIF <- itemFrequency(myGroc)
myGrocIF <- sort(myGrocIF)
head(myGrocIF,10)
tail(myGrocIF,10)
```

Least Frequent Items

```
       baby food  sound storage medium preservation products        kitchen utensil
      0.0001016777          0.0001016777          0.0002033554          0.0004067107
             bags         frozen chicken         baby cosmetics         toilet cleaner
      0.0004067107          0.0006100661          0.0006100661          0.0007117438
```

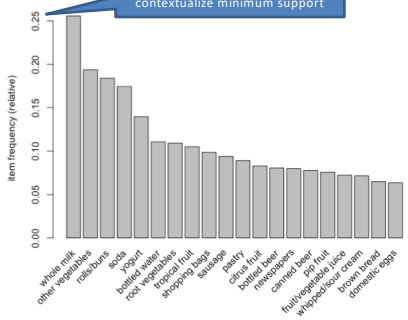
```
tropical fruit  root vegetables    bottled water          yogurt            soda
     0.1049314       0.1089985       0.1105236       0.1395018       0.1743772
  rolls/buns other vegetables      whole milk
     0.1839349       0.1934926       0.2555160
```

Most Frequent Items

Copyright 2019, Jeff Stanton and Jeff Saltz

itemFrequencyPlot(myGroc, topN=20)

Note: Relative frequencies help contextualize minimum support



Exploring the Data (cont.)

```
ct <- crossTable(myGroc, sort=TRUE)
ct[1:6, 1:6]
```

Diagnonal has raw frequency of single item

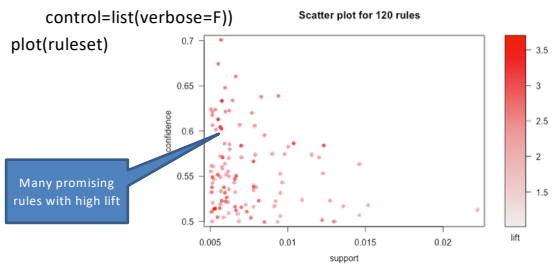
	whole milk	other vegetables	rolls/buns	soda	yogurt	bottled water
whole milk	2513	736	557	394	551	338
other vegetables	736	1903	419	322	427	244
rolls/buns	557	419	1809	377	338	238
soda	394	322	377	1715	269	285
yogurt	551	427	338	269	1372	226
bottled water	338	244	238	285	226	1087

Matrix is square and symmetric

These frequencies help start the thinking about minimum support. 736/9835 is about 0.075 and that is likely to be the ceiling for support levels in these data. We will probably have to set support much lower to get a usable group of rules.

Model Development

```
# Now run a test model to examine some rules
ruleset <- apriori(myGroc,
    parameter=list(supp=0.005, conf=0.50),
    control=list(verbose=F))
plot(ruleset)
```



Many promising rules with high lift

Copyright 2019, Jeff Stanton and Jeff Saltz

The DM Process 3 – Model Development

highLift <- ruleset[quality(ruleset)$lift > 2.249]
inspectDT(highLift)

Sorted by decr order of confidence

Interactive display – use inspect() for console output

Why is whole milk so popular as RHS?

	LHS	RHS	support	confidence	lift	count
[55]	{tropical fruit,root vegetables,yogurt}	{whole milk}	0.006	0.700	2.740	56.000
[50]	{pip fruit,root vegetables,other vegetables}	{whole milk}	0.005	0.675	2.642	54.000
[17]	{butter,whipped/sour cream}	{whole milk}	0.007	0.660	2.583	66.000
[32]	{pip fruit,whipped/sour cream}	{whole milk}	0.006	0.648	2.537	59.000
[23]	{butter,yogurt}	{whole milk}	0.009	0.639	2.500	92.000
[22]	vegetables,butter}	{whole milk}	0.008	0.638	2.496	81.000
[...]	{whole	{whole				

Showing 1 to 10 of 60 entries

Model Development

```
# Not interested in milk? Here's another run, controlling the RHS
ruleset <- apriori(myGroc,
          parameter=list(supp=0.0025, conf=0.25),
          control=list(verbose=F),
          appearance = list(default="lhs",rhs=("bottled water")))
summary(ruleset)
```

Small set of rules, but all sized 3, 4

```
set of 17 rules

rule length distribution (lhs + rhs):sizes
 3  4
14  3

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.000   3.000   3.000   3.176   3.000   4.000

summary of quality measures:
    support         confidence          lift            count
 Min.   :0.002542   Min.   :0.2500   Min.   :2.262   Min.   :25.00
 1st Qu.:0.002847   1st Qu.:0.2598   1st Qu.:2.351   1st Qu.:28.00
 Median :0.002847   Median :0.2718   Median :2.460   Median :28.00
 Mean   :0.003678   Mean   :0.2912   Mean   :2.634   Mean   :36.18
 3rd Qu.:0.004169   3rd Qu.:0.3077   3rd Qu.:2.784   3rd Qu.:41.00
 Max.   :0.007422   Max.   :0.3733   Max.   :3.378   Max.   :73.00
```

Model Development

highLift <- ruleset[quality(ruleset)$lift > 2.249]
inspectDT(highLift)

Sorted by decr order of confidence

Actionable insight: place healthy drinks, including bottled water, near the beer cooler

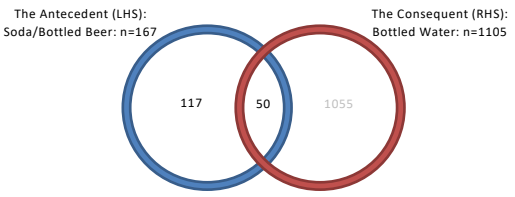
	LHS	RHS	support	confidence	lift	count
[1]	{UHT-milk,soda}	{bottled water}	0.003	0.373	3.378	28.000
[2]	{fruit/vegetable juice,bottled beer}	{bottled water}	0.003	0.362	3.278	25.000
[8]	{yogurt,rolls/buns,soda}	{bottled water}	0.003	0.353	3.193	30.000
[3]	{tropical fruit,bottled beer}	{bottled water}	0.003	0.321	2.904	26.000
[5]	{yogurt,bottled beer}	{bottled water}	0.003	0.308	2.784	28.000
[4]	{soda,bottled beer}	{bottled water}	0.005	0.299	2.709	50.000
[6]	{whole milk,bottled beer}	{bottled water}	0.006	0.299	2.701	60.000

Showing 1 to 8 of 8 entries

Copyright 2019, Jeff Stanton and Jeff Saltz

Questions

Total transactions: 10,000



supp(S;BB) = ?

conf((S;BB)⇒BW) = ?

Bonus: lift((S;BB)⇒BW) = ?

Questions

What are some **examples** of association rules mining "**in the real world**"?

What might be some **possible issues** with using this algorithm?

23

Deploying Results

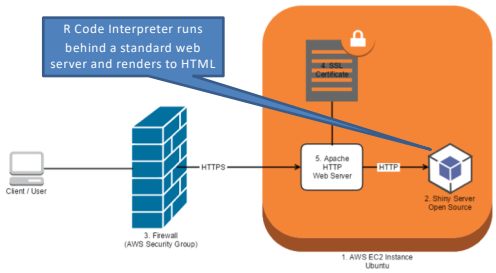
- Ingest point-of-sale data daily from each store location; aggregate into weekly transaction matrix
- Run apriori models weekly for each store, email store manager with summary of high-lift rules
- Actions taken by management might include:
  - Add "pop-up" display shelves to reinforce product pairings
  - Add in-store coupon dispensers to promote cross-aisle product combinations
  - Publish recipes on store receipts for item combinations
  - Publish visuals of finished recipes in weekly flier
  - Offer store coupons that match high-lift combinations

Copyright 2019, Jeff Stanton and Jeff Saltz

Worked Example: Building a Shiny App

School of Information Studies
SYRACUSE UNIVERSITY

What is Shiny?



Shiny Apps Have Two Pieces

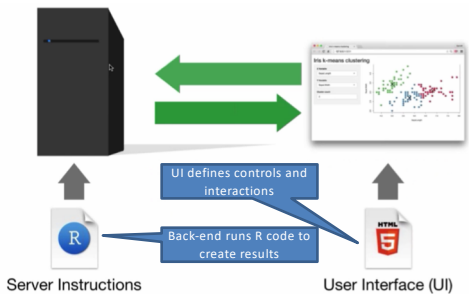
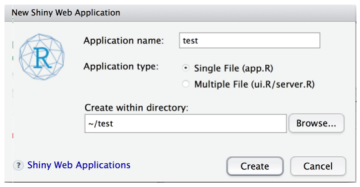


Image credit: R-Studio

Copyright 2019, Jeff Stanton and Jeff Saltz

A New Type of File

- **a 'Shiny Web App…"**
- **Not an R Script**



UI Code

```
# Define UI for application that draws a histogram
ui <- fluidPage(

  # Application title
  titlePanel("Old Faithful Geyser Data"),

  # Sidebar with a slider input for number of bins
  sidebarLayout(
    sidebarPanel(
      sliderInput("bins", "Number of bins:", min = 1,  max = 50, value = 30)
    ),

    # Show a plot of the generated distribution
    mainPanel(
      plotOutput("distPlot")
    )
  )
)
```

This translates inside the server to a mess of HTML5 code that implements the visible UI on the web page

Here's the connection to some R output; see next slide

Server Code

```
server <- function(input, output) {
  output$distPlot <- renderPlot({
    # generate bins based on input$bins from ui.R
    x    <- faithful[, 2]
    bins <- seq(min(x), max(x), length.out =  input$bins + 1)

    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')
  })
}
```

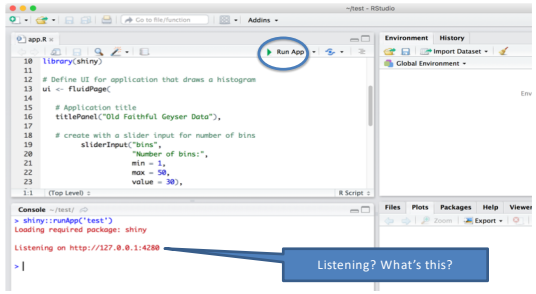
There's the other side of the connection to the UI

Plain vanilla R code: The resulting plot output gets rendered into HTML5 and delivered to the server that is providing the HTTPS response to your browser

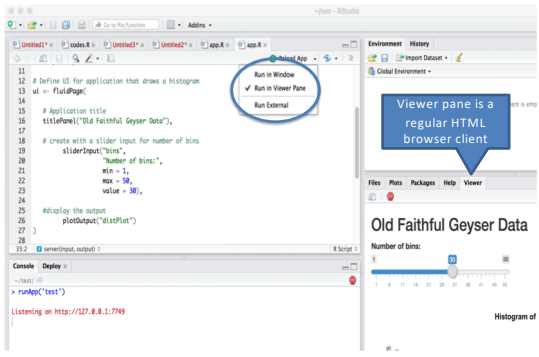
Copyright 2019, Jeff Stanton and Jeff Saltz

Running the Application

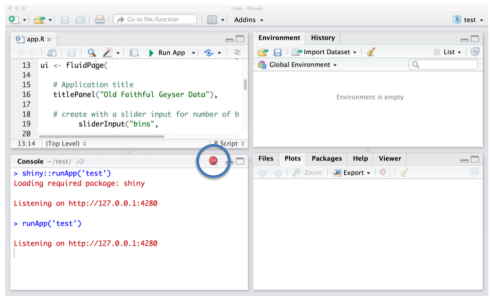
shinyApp(ui = ui, server = server)



Many Ways to Run the App



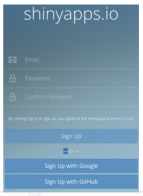
How to Stop the App

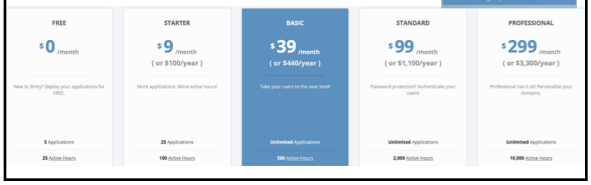


Copyright 2019, Jeff Stanton and Jeff Saltz

Shinyapps.io

- Sign up for a free account
- You will get a token and a secret
- These will be used with the rsconnect package to communicate with your account





Deploying the Application

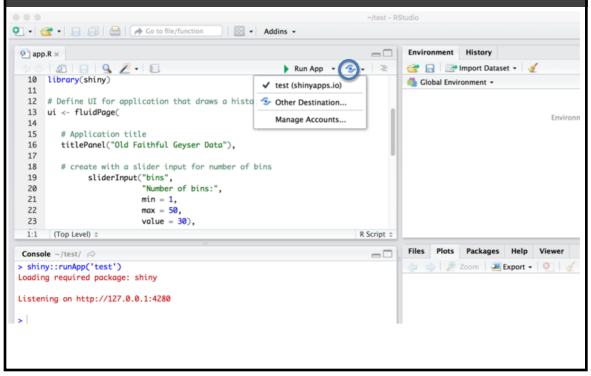
Use http://www.shinyapps.io/,

```
> install.packages("rsconnect")
> library(rsconnect)
```

```
> setAccountInfo(name='xxx',
+   token='yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy',
+   secret=zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz')
```

Need a shinyapps account

Publishing the application



Copyright 2019, Jeff Stanton and Jeff Saltz