Association Rule Learning and the Apriori Algorithm

Association Rule Learning (also called Association Rule Mining) is a common technique used to find associations between many variables. It is often used by grocery stores, retailers, and anyone with a large transactional databases. It's the same way that Target knows your pregnant or when you're buying an item on Amazon.com they know what else you want to buy. The same idea extends to Pandora.com knowing what song you want to listen to next. All of these incorporate, at some level, data mining concepts and association rule algorithms.

Michael Hahsler, et al. has authored and maintains two very useful R packages relating to association rule mining: the *arules* package and the *arulesViz* package. Furthermore, Hahsler has provided two very good example articles providing details on how to use these packages in Introduction to arules and Visualizing Association Rules.

Often Association Rule Learning is used to analyze the "market-basket" for retailers. Traditionally, this simply looks at whether a person has purchased an item or not and can be seen as a binary matrix.

Association rules use the R *arules* library. The *arulesViz* add additional features for graphing and plotting the rules.

```
library("arules");
library("arulesViz");
```

For testing purposes there is a convenient way to generate random data where patterns can be mined. The random data is generated in such a way where there is correlation and has correlated items.

```
patterns = random.patterns(nItems = 1000);
summary(patterns);
trans = random.transactions(nItems = 1000, nTrans = 1000, method =
"agrawal",  patterns = patterns);
image(trans);
```

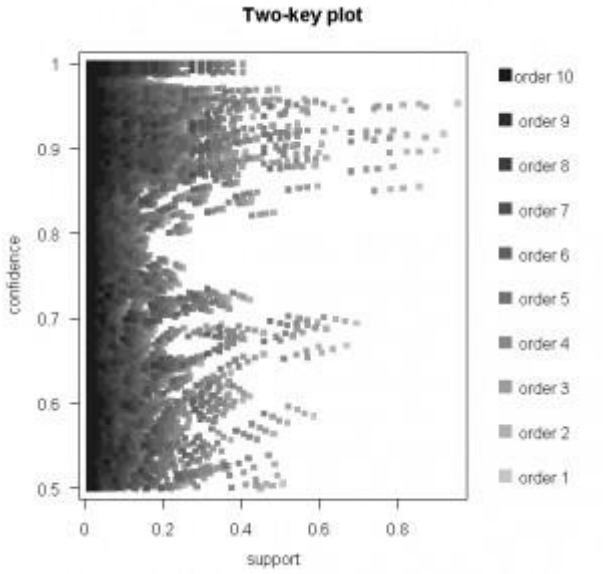
However, a transaction dataset will usually be available using the approach described in "Data Frames and Transactions". The rules can then be created using the *apriori* function on the transaction dataset.

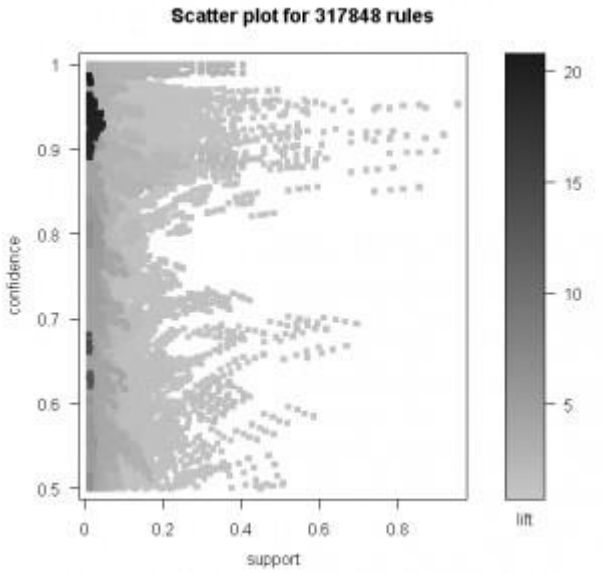
```
data("AdultUCI");
Adult = as(AdultUCI, "transactions");
```

```
rules = apriori(Adult, parameter=list(support=0.01, confidence=0.5));
rules;
```

Once the rules have been created a researcher can then review and filter the rules down to a manageable subset. This can be done in a variety of ways using both graphs and by simply inspecting the rules.

```
inspect(head(sort(rules, by="lift"),3));
plot(rules);
head(quality(rules));
plot(rules, measure=c("support","lift"), shading="confidence");
plot(rules, shading="order", control=list(main ="Two-key plot"));
```

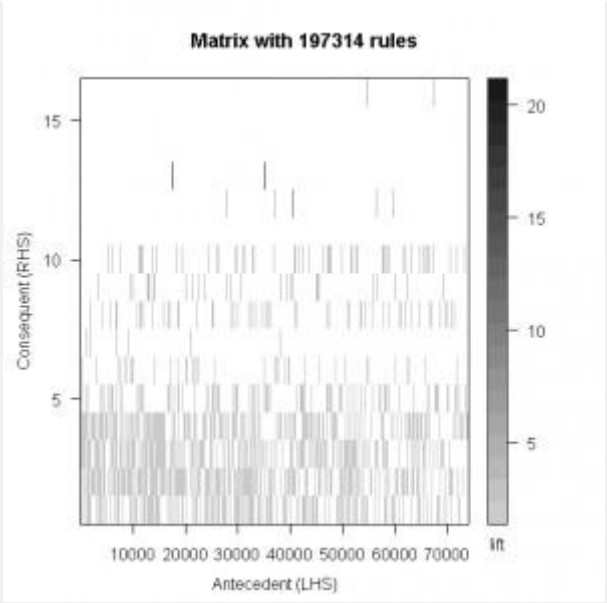




Having 317848 association rules is far too many for a human to deal with. So we're going to trim down the rules to the ones that are important.

```
sel = plot(rules, measure=c("support","lift"), shading="confidence",
interactive=TRUE);
subrules = rules[quality(rules)$confidence > 0.8];
subrules
```

Once again we can now subset the rules to get a visual. In these graphs we can see the two parts to an association rule: the antecedent (IF) and the consequent (THEN). These patterns are found by determining frequent patterns in the data and these are identified by the support and confidence. The support indicates how frequently the items appear in the dataset. The confidence indicates the number of times the IF/THEN statement on the data are true. These IF/THEN statements can be visualized by the following graph:



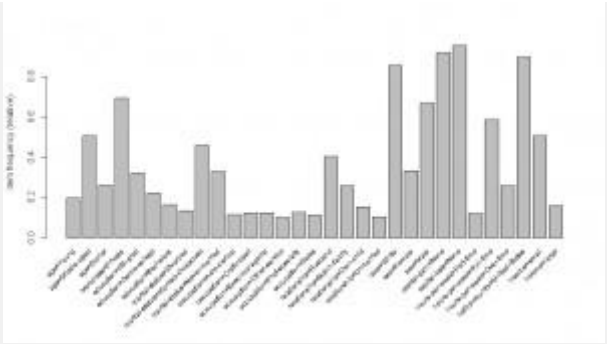
Association Rules with Consequent and Antecedent.

This code will produce many different ways to look at the graphs and can even produce 3-D graphs.

```
plot(subrules, method="matrix", measure="lift");
plot(subrules, method="matrix", measure="lift", control=list(reorder=TRUE));
plot(subrules, method="matrix3D", measure="lift");
plot(subrules, method="matrix3D", measure="lift", control = list(reorder=TRUE));
plot(subrules, method="matrix", measure=c("lift", "confidence"));
plot(subrules, method="matrix", measure=c("lift","confidence"), control =
list(reorder=TRUE));
plot(rules, method="grouped");
plot(rules, method="grouped", control=list(k=50));
sel = plot(rules, method="grouped", interactive=TRUE);
```

We can then subset the rules to the top 30 most important rules and then inspect the smaller set of rules individually to determine where there are meaningful associations.

```
subrules2 = head(sort(rules, by="lift"), 30);
plot(subrules2, method="graph");
plot(subrules2, method="graph", control=list(type="items"));
plot(subrules2, method="paracoord");
plot(subrules2, method="paracoord", control=list(reorder=TRUE));
oneRule = sample(rules, 1);
inspect(oneRule);
```



Shows the Frequent Itemsets

Here we can look at the frequent itemsets and we can use the *eclat* algorithm rather than the *apriori* algorithm.

```
itemFrequencyPlot(Adult, support = 0.1, cex.names=0.8);
fsets = eclat(trans, parameter = list(support = 0.05), control =
list(verbose=FALSE));
singleItems = fsets[size(items(fsets)) == 1];
singleSupport = quality(singleItems)$support;
names(singleSupport) = unlist(LIST(items(singleItems), decode = FALSE));
head(singleSupport, n = 5);
itemsetList = LIST(items(fsets), decode = FALSE);
allConfidence = quality(fsets)$support / sapply(itemsetList, function(x)
max(singleSupport[as.character(x)]));
quality(fsets) = cbind(quality(fsets), allConfidence);
summary(fsets);
```

Using these approaches a researcher can narrow down and determine association rules and determine what leads to frequent items. This is highly useful when working with extremely large datasets.