Chapter 9

Association Rules

This chapter presents examples of association rule mining with R. It starts with basic concepts of association rules, and then demonstrates association rules mining with R. After that, it presents examples of pruning redundant rules and interpreting and visualizing association rules. The chapter concludes with discussions and recommended readings.

9.1 Basics of Association Rules

Association rules are rules presenting association or correlation between itemsets. An association rule is in the form of $A \Rightarrow B$, where A and B are two disjoint itemsets, referred to respectively as the `lhs` (left-hand side) and `rhs` (right-hand side) of the rule. The three most widely-used measures for selecting interesting rules are *support*, *confidence* and *lift*. *Support* is the percentage of cases in the data that contains both A and B, *confidence* is the percentage of cases containing A that also contain B, and *lift* is the ratio of confidence to the percentage of cases containing B. The formulae to calculate them are:

$$\begin{aligned}
\text{support}(A \Rightarrow B) &= P(A \cup B) && (9.1)\\
\text{confidence}(A \Rightarrow B) &= P(B|A) && (9.2)\\
&= \frac{P(A \cup B)}{P(A)} && (9.3)\\
\text{lift}(A \Rightarrow B) &= \frac{\text{confidence}(A \Rightarrow B)}{P(B)} && (9.4)\\
&= \frac{P(A \cup B)}{P(A)P(B)} && (9.5)
\end{aligned}$$

where $P(A)$ is the percentage (or probability) of cases containing A.

In addition to support, confidence and lift, there are many other interestingness measures, such as chi-square, conviction, gini and leverage. An introduction to over 20 measures can be found in Tan et al.'s work [Tan et al., 2002].

9.2 The Titanic Dataset

The `Titanic` dataset in the *datasets* package is a 4-dimensional table with summarized information on the fate of passengers on the Titanic according to social class, sex, age and survival. To make it suitable for association rule mining, we reconstruct the raw data as `titanic.raw`, where each row represents a person. The reconstructed raw data can also be downloaded as file "titanic.raw.rdata" at `http://www.rdatamining.com/data`.

```
> str(Titanic)

 table [1:4, 1:2, 1:2, 1:2] 0 0 35 0 0 0 17 0 118 154 ...
 - attr(*, "dimnames")=List of 4
  ..$ Class   : chr [1:4] "1st" "2nd" "3rd" "Crew"
  ..$ Sex     : chr [1:2] "Male" "Female"
  ..$ Age     : chr [1:2] "Child" "Adult"
  ..$ Survived: chr [1:2] "No" "Yes"

> df <- as.data.frame(Titanic)
> head(df)

  Class    Sex   Age Survived Freq
1   1st   Male Child       No    0
2   2nd   Male Child       No    0
3   3rd   Male Child       No   35
4  Crew   Male Child       No    0
5   1st Female Child       No    0
6   2nd Female Child       No    0

> titanic.raw <- NULL
> for(i in 1:4) {
+    titanic.raw <- cbind(titanic.raw, rep(as.character(df[,i]), df$Freq))
+ }
> titanic.raw <- as.data.frame(titanic.raw)
> names(titanic.raw) <- names(df)[1:4]
> dim(titanic.raw)

[1] 2201    4

> str(titanic.raw)

'data.frame':        2201 obs. of  4 variables:
 $ Class   : Factor w/ 4 levels "1st","2nd","3rd",..: 3 3 3 3 3 3 3 3 3 3 ...
 $ Sex     : Factor w/ 2 levels "Female","Male": 2 2 2 2 2 2 2 2 2 2 ...
 $ Age     : Factor w/ 2 levels "Adult","Child": 2 2 2 2 2 2 2 2 2 2 ...
 $ Survived: Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...

> head(titanic.raw)

  Class  Sex   Age Survived
1   3rd Male Child       No
2   3rd Male Child       No
3   3rd Male Child       No
4   3rd Male Child       No
5   3rd Male Child       No
6   3rd Male Child       No

> summary(titanic.raw)

  Class         Sex          Age       Survived
 1st :325   Female: 470   Adult:2092   No :1490
 2nd :285   Male  :1731   Child: 109   Yes: 711
 3rd :706
 Crew:885
```

Now we have a dataset where each row stands for a person, and it can be used for association rule mining.

The raw Titanic dataset can also be downloaded from `http://www.cs.toronto.edu/~delve/data/titanic/desc.html`. The data is file "Dataset.data" in the compressed archive "titanic.tar.gz". It can be read into R with the code below.

```
> # have a look at the 1st 5 lines
> readLines("./data/Dataset.data", n=5)

[1] "1st  adult male   yes" "1st  adult male   yes" "1st  adult male   yes"
[4] "1st  adult male   yes" "1st  adult male   yes"

> # read it into R
> titanic <- read.table("./data/Dataset.data", header=F)
> names(titanic) <- c("Class", "Sex", "Age", "Survived")
```

9.3 Association Rule Mining

A classic algorithm for association rule mining is APRIORI [Agrawal and Srikant, 1994]. It is a level-wise, breadth-first algorithm which counts transactions to find frequent itemsets and then derive association rules from them. An implementation of it is function `apriori()` in package *arules* [Hahsler et al., 2011]. Another algorithm for association rule mining is the ECLAT algorithm [Zaki, 2000], which finds frequent itemsets with equivalence classes, depth-first search and set intersection instead of counting. It is implemented as function `eclat()` in the same package.

Below we demonstrate association rule mining with `apriori()`. With the function, the default settings are: 1) `supp=0.1`, which is the minimum support of rules; 2) `conf=0.8`, which is the minimum confidence of rules; and 3) `maxlen=10`, which is the maximum length of rules.

```
> library(arules)
> # find association rules with default settings
> rules.all <- apriori(titanic.raw)

parameter specification:
 confidence minval smax arem  aval originalSupport support minlen maxlen target
        0.8    0.1    1 none FALSE            TRUE     0.1      1     10  rules
   ext
 FALSE

algorithmic control:
 filter tree heap memopt load sort verbose
    0.1 TRUE TRUE  FALSE TRUE    2    TRUE

apriori - find association rules with the apriori algorithm
version 4.21 (2004.05.09)        (c) 1996-2004   Christian Borgelt
set item appearances ...[0 item(s)] done [0.00s].
set transactions ...[10 item(s), 2201 transaction(s)] done [0.00s].
sorting and recoding items ... [9 item(s)] done [0.00s].
creating transaction tree ... done [0.00s].
checking subsets of size 1 2 3 4 done [0.00s].
writing ... [27 rule(s)] done [0.00s].
creating S4 object  ... done [0.00s].

> rules.all

set of 27 rules
```

```
> inspect(rules.all)

   lhs               rhs              support confidence      lift
1  {}             => {Age=Adult}   0.9504771  0.9504771 1.0000000
2  {Class=2nd}    => {Age=Adult}   0.1185825  0.9157895 0.9635051
3  {Class=1st}    => {Age=Adult}   0.1449341  0.9815385 1.0326798
4  {Sex=Female}   => {Age=Adult}   0.1930940  0.9042553 0.9513700
5  {Class=3rd}    => {Age=Adult}   0.2848705  0.8881020 0.9343750
6  {Survived=Yes} => {Age=Adult}   0.2971377  0.9198312 0.9677574
7  {Class=Crew}   => {Sex=Male}    0.3916402  0.9740113 1.2384742
8  {Class=Crew}   => {Age=Adult}   0.4020900  1.0000000 1.0521033
9  {Survived=No}  => {Sex=Male}    0.6197183  0.9154362 1.1639949
10 {Survived=No}  => {Age=Adult}   0.6533394  0.9651007 1.0153856
11 {Sex=Male}     => {Age=Adult}   0.7573830  0.9630272 1.0132040
12 {Sex=Female,
    Survived=Yes} => {Age=Adult}   0.1435711  0.9186047 0.9664669
13 {Class=3rd,
    Sex=Male}     => {Survived=No} 0.1917310  0.8274510 1.2222950
14 {Class=3rd,
    Survived=No}  => {Age=Adult}   0.2162653  0.9015152 0.9484870
15 {Class=3rd,
    Sex=Male}     => {Age=Adult}   0.2099046  0.9058824 0.9530818
16 {Sex=Male,
    Survived=Yes} => {Age=Adult}   0.1535666  0.9209809 0.9689670
17 {Class=Crew,
    Survived=No}  => {Sex=Male}    0.3044071  0.9955423 1.2658514
18 {Class=Crew,
    Survived=No}  => {Age=Adult}   0.3057701  1.0000000 1.0521033
19 {Class=Crew,
    Sex=Male}     => {Age=Adult}   0.3916402  1.0000000 1.0521033
20 {Class=Crew,
    Age=Adult}    => {Sex=Male}    0.3916402  0.9740113 1.2384742
21 {Sex=Male,
    Survived=No}  => {Age=Adult}   0.6038164  0.9743402 1.0251065
22 {Age=Adult,
    Survived=No}  => {Sex=Male}    0.6038164  0.9242003 1.1751385
23 {Class=3rd,
    Sex=Male,
    Survived=No}  => {Age=Adult}   0.1758292  0.9170616 0.9648435
24 {Class=3rd,
    Age=Adult,
    Survived=No}  => {Sex=Male}    0.1758292  0.8130252 1.0337773
25 {Class=3rd,
    Sex=Male,
    Age=Adult}    => {Survived=No} 0.1758292  0.8376623 1.2373791
26 {Class=Crew,
    Sex=Male,
    Survived=No}  => {Age=Adult}   0.3044071  1.0000000 1.0521033
27 {Class=Crew,
    Age=Adult,
    Survived=No}  => {Sex=Male}    0.3044071  0.9955423 1.2658514
```

As a common phenomenon for association rule mining, many rules generated above are uninteresting. Suppose that we are interested in only rules with `rhs` indicating survival, so we set `rhs=c("Survived=No", "Survived=Yes")` in `appearance` to make sure that only "Survived=No" and "Survived=Yes" will appear in the `rhs` of rules. All other items can appear in the `lhs`, as set with `default="lhs"`. In the above result `rules.all`, we can also see that the left-hand side (`lhs`) of the first rule is empty. To exclude such rules, we set `minlen` to 2 in the code below. Moreover, the details of progress are suppressed with `verbose=F`. After association rule mining, rules are sorted by lift to make high-lift rules appear first.

```
> # rules with rhs containing "Survived" only
> rules <- apriori(titanic.raw, control = list(verbose=F),
+                  parameter = list(minlen=2, supp=0.005, conf=0.8),
+                  appearance = list(rhs=c("Survived=No", "Survived=Yes"),
+                                    default="lhs"))
> quality(rules) <- round(quality(rules), digits=3)
> rules.sorted <- sort(rules, by="lift")

> inspect(rules.sorted)

   lhs             rhs            support confidence  lift
1  {Class=2nd,
    Age=Child}  => {Survived=Yes}   0.011      1.000 3.096
2  {Class=2nd,
    Sex=Female,
    Age=Child}  => {Survived=Yes}   0.006      1.000 3.096
3  {Class=1st,
    Sex=Female} => {Survived=Yes}   0.064      0.972 3.010
4  {Class=1st,
    Sex=Female,
    Age=Adult}  => {Survived=Yes}   0.064      0.972 3.010
5  {Class=2nd,
    Sex=Female} => {Survived=Yes}   0.042      0.877 2.716
6  {Class=Crew,
    Sex=Female} => {Survived=Yes}   0.009      0.870 2.692
7  {Class=Crew,
    Sex=Female,
    Age=Adult}  => {Survived=Yes}   0.009      0.870 2.692
8  {Class=2nd,
    Sex=Female,
    Age=Adult}  => {Survived=Yes}   0.036      0.860 2.663
9  {Class=2nd,
    Sex=Male,
    Age=Adult}  => {Survived=No}    0.070      0.917 1.354
10 {Class=2nd,
    Sex=Male}   => {Survived=No}    0.070      0.860 1.271
11 {Class=3rd,
    Sex=Male,
    Age=Adult}  => {Survived=No}    0.176      0.838 1.237
12 {Class=3rd,
    Sex=Male}   => {Survived=No}    0.192      0.827 1.222
```

When other settings are unchanged, with a lower minimum support, more rules will be produced, and the associations between itemsets shown in the rules will be more likely to be by chance. In the above code, the minimum support is set to 0.005, so each rule is supported at least by 12 (=`ceiling(0.005 * 2201)`) cases, which is acceptable for a population of 2201.

Support, confidence and lift are three common measures for selecting interesting association rules. Besides them, there are many other interestingness measures, such as chi-square, conviction, gini and leverage [Tan et al., 2002]. More than twenty measures can be calculated with function `interestMeasure()` in the *arules* package.

9.4 Removing Redundancy

Some rules generated in the previous section (see `rules.sorted`, page 89) provide little or no extra information when some other rules are in the result. For example, the above rule 2 provides no extra knowledge in addition to rule 1, since rules 1 tells us that all 2nd-class children survived. Generally speaking, when a rule (such as rule 2) is a super rule of another rule (such as rule 1) and the former has the same or a lower lift, the former rule (rule 2) is considered to be redundant. Other redundant rules in the above result are rules 4, 7 and 8, compared respectively with rules 3, 6 and 5.

Below we prune redundant rules. Note that the rules have already been sorted descendingly by lift.

```
> # find redundant rules
> subset.matrix <- is.subset(rules.sorted, rules.sorted)
> subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA
> redundant <- colSums(subset.matrix, na.rm=T) >= 1
> which(redundant)

[1] 2 4 7 8

> # remove redundant rules
> rules.pruned <- rules.sorted[!redundant]
> inspect(rules.pruned)

  lhs             rhs            support confidence  lift
1 {Class=2nd,
   Age=Child}  => {Survived=Yes}   0.011      1.000 3.096
2 {Class=1st,
   Sex=Female} => {Survived=Yes}   0.064      0.972 3.010
3 {Class=2nd,
   Sex=Female} => {Survived=Yes}   0.042      0.877 2.716
4 {Class=Crew,
   Sex=Female} => {Survived=Yes}   0.009      0.870 2.692
5 {Class=2nd,
   Sex=Male,
   Age=Adult}  => {Survived=No}    0.070      0.917 1.354
6 {Class=2nd,
   Sex=Male}   => {Survived=No}    0.070      0.860 1.271
7 {Class=3rd,
   Sex=Male,
   Age=Adult}  => {Survived=No}    0.176      0.838 1.237
8 {Class=3rd,
   Sex=Male}   => {Survived=No}    0.192      0.827 1.222
```

In the code above, function `is.subset(r1, r2)` checks whether `r1` is a subset of `r2` (i.e., whether r2 is a superset of `r1`). Function `lower.tri()` returns a logical matrix with `TURE` in lower triangle. From the above results, we can see that rules 2, 4, 7 and 8 (before redundancy removal) are successfully pruned.

9.5 Interpreting Rules

While it is easy to find high-lift rules from data, it is not an easy job to understand the identified rules. It is not uncommon that the association rules are misinterpreted to find their business meanings. For instance, in the above rule list `rules.pruned`, the first rule `"{Class=2nd, Age=Child} => {Survived=Yes}"` has a confidence of one and a lift of three and there are no rules on children of the 1st or 3rd classes. Therefore, it might be interpreted by users as *children of the 2nd class had a higher survival rate than other children*. This is wrong! The rule states only that all children of class 2 survived, but provides no information at all to compare the survival rates of different classes. To investigate the above issue, we run the code below to find rules whose `rhs` is `"Survived=Yes"` and `lhs` contains `"Class=1st"`, `"Class=2nd"`, `"Class=3rd"`, `"Age=Child"` and `"Age=Adult"` only, and which contains no other items (`default="none"`). We use lower thresholds for both support and confidence than before to find all rules for children of different classes.

```
> rules <- apriori(titanic.raw,
+                  parameter = list(minlen=3, supp=0.002, conf=0.2),
+                  appearance = list(rhs=c("Survived=Yes"),
+                                    lhs=c("Class=1st", "Class=2nd", "Class=3rd",
+                                          "Age=Child", "Age=Adult"),
+                                    default="none"),
+                  control = list(verbose=F))
> rules.sorted <- sort(rules, by="confidence")
> inspect(rules.sorted)
```

```
  lhs            rhs                support confidence      lift
1 {Class=2nd,
   Age=Child} => {Survived=Yes} 0.010904134  1.0000000 3.0956399
2 {Class=1st,
   Age=Child} => {Survived=Yes} 0.002726034  1.0000000 3.0956399
3 {Class=1st,
   Age=Adult} => {Survived=Yes} 0.089504771  0.6175549 1.9117275
4 {Class=2nd,
   Age=Adult} => {Survived=Yes} 0.042707860  0.3601533 1.1149048
5 {Class=3rd,
   Age=Child} => {Survived=Yes} 0.012267151  0.3417722 1.0580035
6 {Class=3rd,
   Age=Adult} => {Survived=Yes} 0.068605179  0.2408293 0.7455209
```

In the above result, the first two rules show that children of the 1st class are of the same survival rate as children of the 2nd class and that all of them survived. The rule of 1st-class children didn't appear before, simply because of its support was below the threshold specified in Section 9.3. Rule 5 presents a sad fact that children of class 3 had a low survival rate of 34%, which is comparable with that of 2nd-class adults (see rule 4) and much lower than 1st-class adults (see rule 3).

9.6 Visualizing Association Rules

Next we show some ways to visualize association rules, including scatter plot, balloon plot, graph and parallel coordinates plot. More examples on visualizing association rules can be found in the vignettes of package *arulesViz* [Hahsler and Chelluboina, 2012] on CRAN at `http://cran.r-project.org/web/packages/arulesViz/vignettes/arulesViz.pdf`.

```
> library(arulesViz)
> plot(rules.all)
```

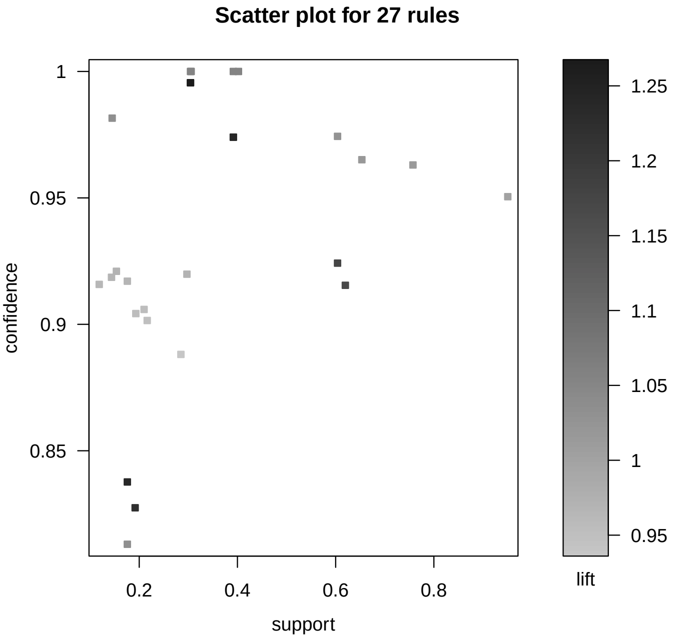

Figure 9.1: A Scatter Plot of Association Rules

```
> plot(rules.all, method="grouped")
```

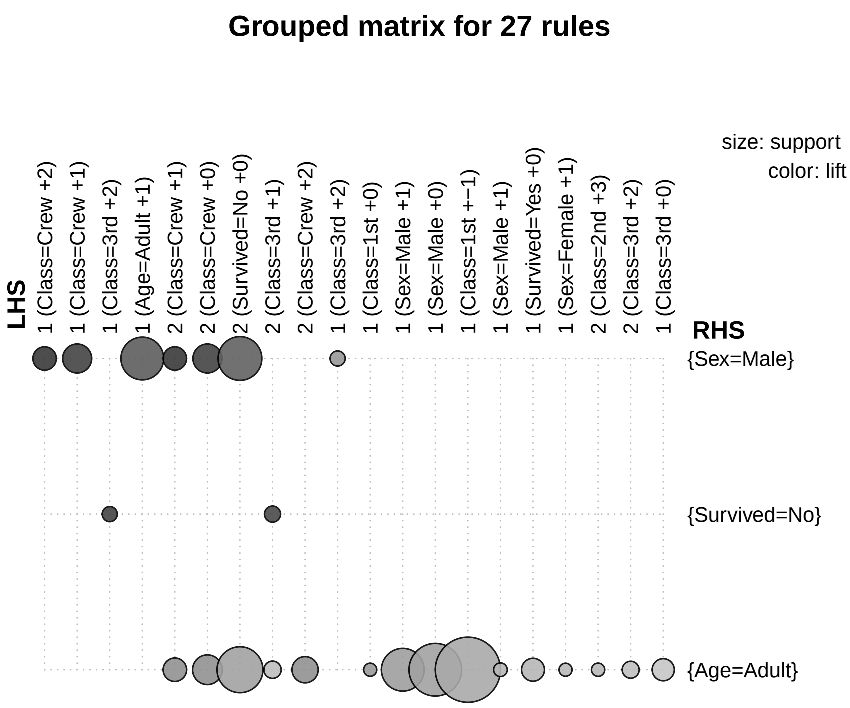

Figure 9.2: A Balloon Plot of Association Rules

```
> plot(rules.all, method="graph")
```

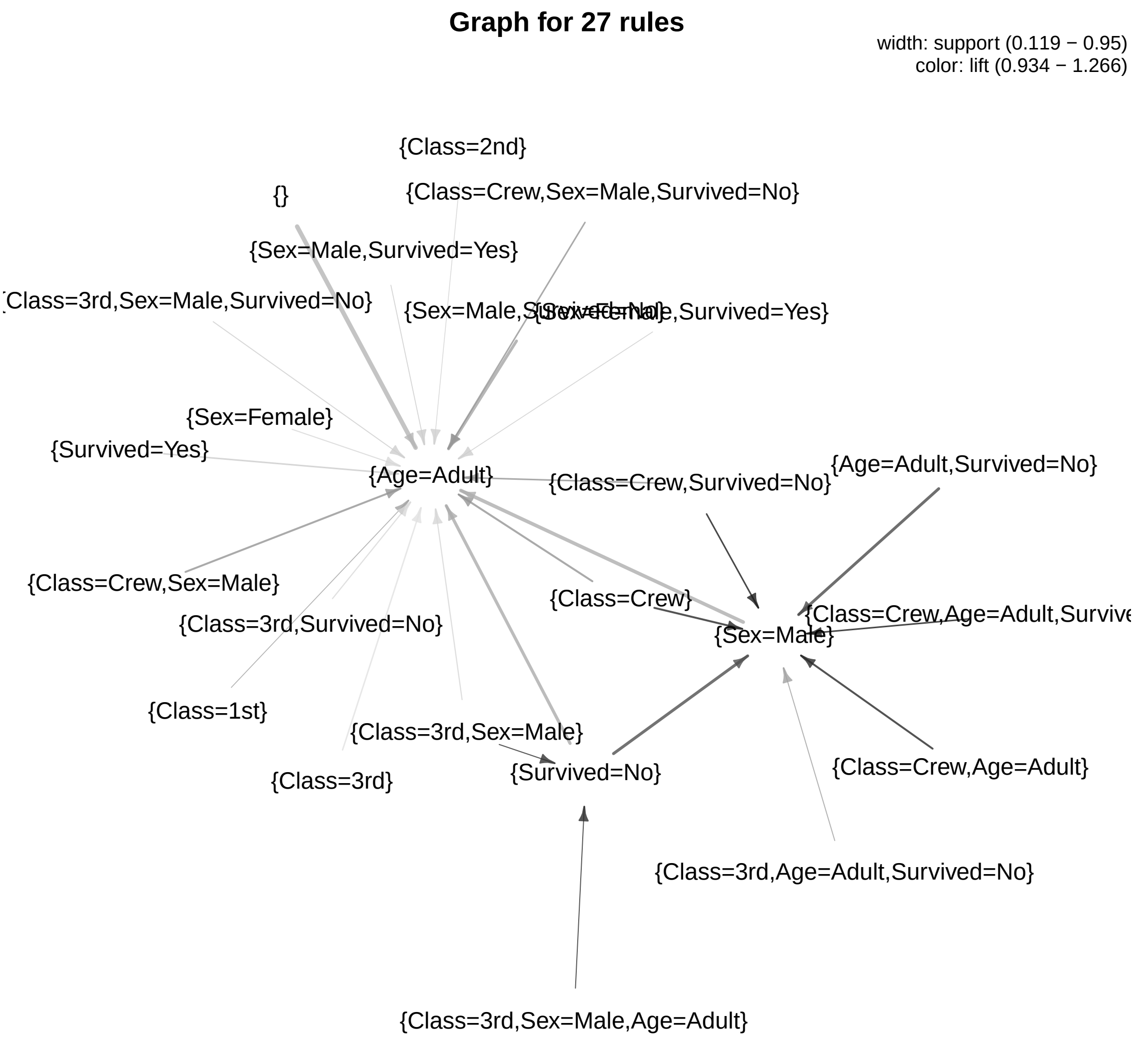

Figure 9.3: A Graph of Association Rules

```
> plot(rules.all, method="graph", control=list(type="items"))
```

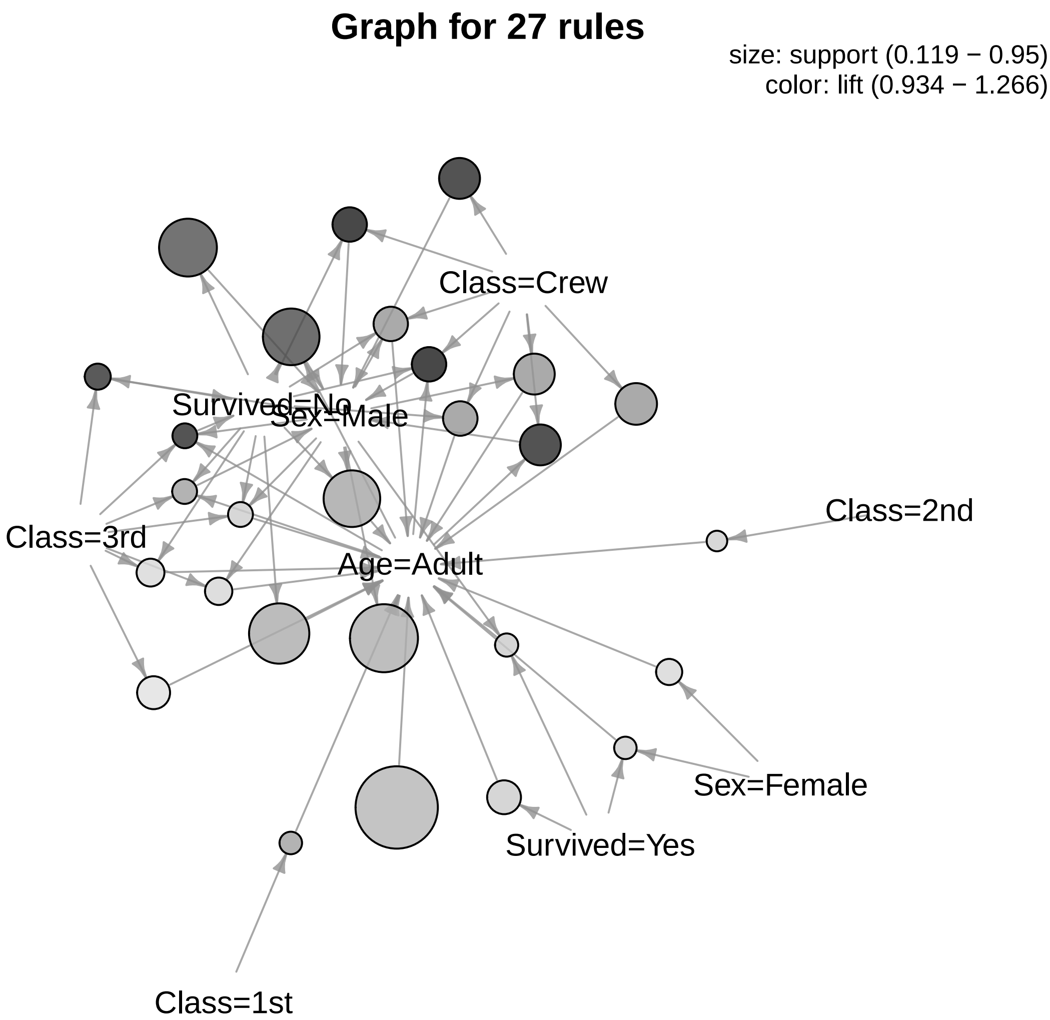

Figure 9.4: A Graph of Items

```
> plot(rules.all, method="paracoord", control=list(reorder=TRUE))
```

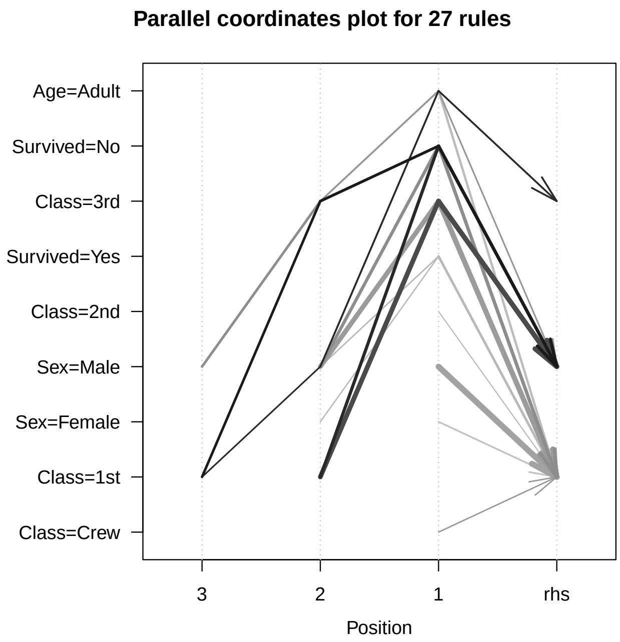

Figure 9.5: A Parallel Coordinates Plot of Association Rules

9.7 Discussions and Further Readings

In this chapter, we have demonstrated association rule mining with package *arules* [Hahsler et al., 2011]. More examples on that package can be found in Hahsler et al.'s work [Hahsler et al., 2005]. Two other packages related to association rules are *arulesSequences* and *arulesNBMiner*. Package *arulesSequences* provides functions for mining sequential patterns [Buchta et al., 2012]. Package *arulesNBMiner* implements an algorithm for mining negative binomial (NB) frequent itemsets and NB-precise rules [Hahsler, 2012].

More techniques on post mining of association rules, such as selecting interesting association rules, visualization of association rules and using association rules for classification, can be found in Zhao et al's work [Zhao et al., 2009b].