Building on the arules Infrastructure for Analyzing Transaction Data with R

Michael Hahsler[1] and Kurt Hornik[2]

[1] Department of Information Systems and Operations,
Wirtschaftsuniversität, A-1090 Wien, Austria
[2] Department of Statistics and Mathematics,
Wirtschaftsuniversität, A-1090 Wien, Austria

Abstract. The free and extensible statistical computing environment R with its enormous number of extension packages already provides many state-of-the-art techniques for data analysis. Support for association rule mining, a popular exploratory method which can be used, among other purposes, for uncovering cross-selling opportunities in *market baskets,* has become available recently with the R extension package **arules**. After a brief introduction to transaction data and association rules, we present the formal framework implemented in **arules** and demonstrate how clustering and association rule mining can be applied together using a market basket data set from a typical retailer. This paper shows that implementing a basic infrastructure with formal classes in R provides an extensible basis which can very efficiently be employed for developing new applications (such as clustering transactions) in addition to association rule mining.

1 Introduction

An aim of analyzing transaction data is to discover interesting patterns (e.g., association rules) in large databases containing transaction data. Transaction data can originate from various sources. For example, POS systems collect large quantities of records (i.e., transactions, *market baskets*) containing products purchased during a shopping trip. Analyzing market basket data is called *Market Basket Analysis* (Russell et al. (1997), Berry and Linoff (1997)) and is used to uncover unexploited selling opportunities. Table 1 depicts a simple example for transaction data. Formally, let $I = \{i_1, i_2, \ldots, i_n\}$ be a set of n binary attributes called *items*. Let $\mathcal{D} = \{t_1, t_2, \ldots, t_m\}$ be a set of transactions called the *database*. Each transaction in $\mathcal{D}$ has a unique transaction ID and contains a subset of the items in I.

Categorical and/or metric attributes from other data sources (e.g., in survey data) can be mapped to binary attributes and thus be treated in the same way as transaction data (Piatetsky-Shapiro (1991), Hastie et al. (2001)). Here interesting relationships between values of the attributes can be discovered.

transaction ID	items
1	milk, bread
2	bread, butter
3	beer
4	milk, bread, butter
5	bread, butter

(a)

	items			
trans. ID	milk	bread	butter	beer
1	1	1	0	0
2	0	1	1	0
3	0	0	0	1
4	1	1	1	0
5	0	1	1	0

(b)

Table 1. Example of market basket data represented as (a) shopping lists and as (b) a binary purchase incidence matrix where ones indicate that an item is contained in a transaction.

Agrawal et al. (1993) stated the problem of mining association rules from transaction data (e.g., database of market baskets) as follows:

A *rule* is defined as an implication of the form $X \Rightarrow Y$ where $X, Y \subseteq I$ and $X \cap Y = \emptyset$. The sets of items (for short *itemsets*) X and Y are called *antecedent* (left-hand-side or lhs) and *consequent* (right-hand-side or rhs) of the rule. To select interesting rules from the set of all possible rules, constraints on various measures of strength and interestingness can be used. The best-known constraints are minimum thresholds on *support* and *confidence.* The support $\text{supp}(X)$ of an itemset X is defined as the proportion of transactions in the data set which contain the itemset. The confidence of a rule is defined $\text{conf}(X \Rightarrow Y) = \text{supp}(X \cup Y)/\text{supp}(X)$. *Association rules* are required to satisfy both a minimum support and a minimum confidence constraint at the same time.

An infrastructure for mining transaction data for the free statistical computing environment R (R Development Team (2005)) is provided by the extension package **arules** (Hahsler et al. (2005, 2006)). In this paper we discuss this infrastructure and indicate how it can conveniently be enhanced (by providing functionality to compute proximities between transactions) to create application frameworks with new data analysis capabilities.

In Section 2 we give a very brief overview of the infrastructure provided by **arules** and discuss calculating similarities between transactions. In Section 3 we demonstrate how the **arules** infrastructure can be used in combination with clustering algorithms (as provided by a multitude of R extension packages) to group transactions representing similar purchasing behavior and then to discover association rules for interesting transaction groups. All necessary R code is provided in the paper.

2 Building on the arules Infrastructure

The **arules** infrastructure implements the formal framework presented by the S4 class structure (Chambers (1998)) in Figure 1. For transaction data the

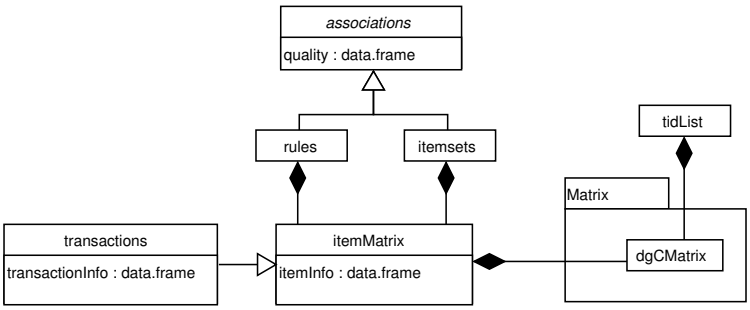

Fig. 1. Simplified UML class diagram (see Fowler (2004)) of the **arules** package.

classes transactions and tidLists (transaction ID lists, an alternative way to represent transaction data) are provided. When needed, data formats commonly used in R to represent transaction data (e.g., data frames and matrices) are automatically transformed into transactions.

For mining patterns, the popular mining algorithm implementations of Apriori and Eclat (Borgelt (2003)) are used in **arules**. Patterns are stored as itemsets and rules representing sets of itemsets or rules, respectively. Both classes directly extend a common virtual class called associations which provides a common interface. In this structure it is easy to add a new type of associations by adding a new class that extends associations.

For efficiently handling the typically very sparse transaction data, items in associations and transactions are implemented by the itemMatrix class which provides a facade for the sparse matrix implementation dgCMatrix from the R package **Matrix** (Bates and Maechler (2005)). To use sparse data (e.g. transactions or associations) for computations which need dense matrices or are implemented in packages which do not support sparse representations, transformation into dense matrices is provided. For all classes standard manipulation methods as well as special methods for analyzing the data are implemented. A full reference of the capabilities of **arules** can be found in the package documentation (Hahsler et al. (2005)).

With the availability of such a conceptual and computational infrastructure providing fundamental data structures and algorithms, powerful application frameworks can be developed by taking advantage of the vast data mining and analysis capabilities of R. Typically, this only requires providing some application-specific "glue" and customization. In what follows, we illustrate this approach for finding interesting groups of market baskets, one of the core value-adding tasks in the analysis of purchase decisions. As grouping (clustering) is based on notions of proximity (similarity or dissimilarity), the glue needed is functionality to compute proximities between transactions, which can be done by using the asymmetric Jaccard dissimilarity (Sneath (1957)) often used for binary data where only ones (corresponding to purchases in our application) carry information. Alternatively, more domain specific proximity

measures like *Affinity* (Aggarwal et al. (2002)) are possible. Extensions to proximity measures for associations are straightforward.

In a recent version of **arules**, we added the necessary extensions for clustering. In the following example, we illustrate how conveniently this now allows for combining clustering and association rule mining.

3 Example: Clustering and Mining Retail Data

We use 1 month (30 days) of real-world point-of-sale transaction data from a typical local grocery outlet for the example. The items are product categories (e.g., *popcorn*) instead of the individual brands. In the available 9835 transactions we found 169 different categories for which articles were purchased. The data set is included in package **arules** under the name `Groceries`. First, we load the package and the data set.

```
R> library("arules")
R> data("Groceries")
```

Suppose the store manager wants to promote the purchases of beef and thus is interested in associations which include the category *beef*. A direct approach would be to mine association rules on the complete data set and filter the rules which include this category. However, since the store manager knows that there are several different types of shopping behavior (e.g., small baskets at lunch time and rather large baskets on Fridays), we first cluster the transactions to find promising groups of transactions which represent similar shopping behavior. From the distance-based clustering algorithms available in R, we choose *partitioning around medoids* (PAM; Kaufman and Rousseeuw (1990)) which takes a dissimilarity matrix between the objects (transactions) and a predefined number of clusters (k) as inputs and returns cluster labels for the objects. PAM is similar to the well-known k-means algorithm, with the main differences that is uses medoids instead of centroids to represent cluster centers and that it works on arbitrary dissimilarity matrices. PAM is available in the recommended R extension package **cluster** (Maechler (2005)).

To keep the dissimilarity matrix at a manageable size, we take a sample of size 2000 from the transaction database (Note that we first set the random number generator's seed for reasons of reproducibility). For the sample, we calculate the dissimilarity matrix using the function `dissimilarity()` with the method `"Jaccard"` which implements the Jaccard dissimilarity. Both `dissimilarity()` and `sample()` are recent "glue" additions to **arules**. With this dissimilarity matrix and the number of clusters pre-set to $k = 8$ (using expert judgment by the store manager), we apply PAM to the sample.

```
R> set.seed(1234)
R> s <- sample(Groceries, 2000)
R> d <- dissimilarity(s, method = "Jaccard")
```

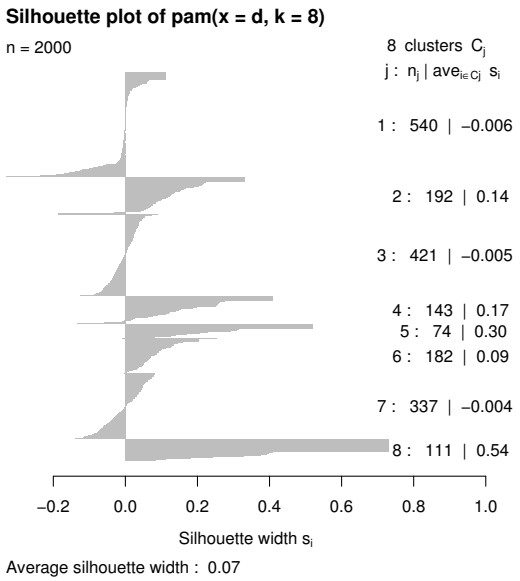

Fig. 2. Silhouette plot of the clustering.

```
R> library("cluster")
R> clustering <- pam(d, k = 8)
R> plot(clustering)
```

Visualization is employed to assess the obtained clustering. The silhouette plot (Kaufman and Rousseeuw (1990)) in Figure 2 displays the silhouette widths for each object (transaction) ordered by cluster as horizontal bars. The silhouette width is a measure of how well an object belongs to its assigned cluster. Compact clusters exclusively consist of objects with high silhouette widths. In the plot we see, that cluster 8 is by far the most compact cluster. Several other clusters have objects with negative silhouette widths which indicates dispersed clusters. However, this is typical for clustering in high-dimensional space.

To predict labels for the whole data set based on the clustered sample, we use the nearest neighbor approach. Package **arules** provides now a new "glue" `predict()` method which can be used to find the labels for all transactions in the Groceries database given the cluster medoids. With labels for all transactions, we can generate a list of transaction data sets, one for each cluster.

```
R> allLabels <- predict(s[clustering$medoids], Groceries,
+     method = "Jaccard")
R> cluster <- split(Groceries, allLabels)
```

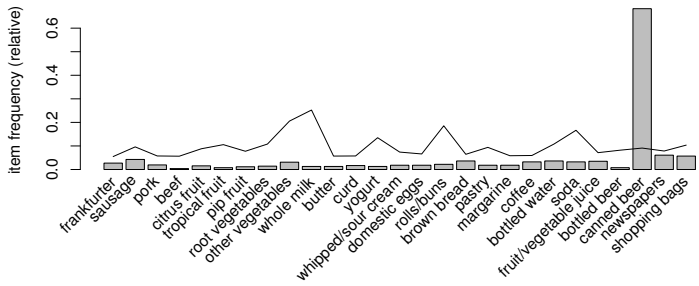

Fig. 3. Item frequencies in cluster 8.

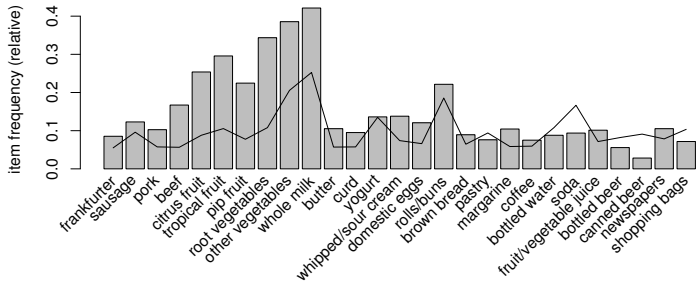

Fig. 4. Item frequencies in the cluster 3.

The transaction data set for each cluster can now be analyzed and used independently. We will demonstrate this by choosing two different clusters, clusters number 8 and 3. For visualization of the clusters, we use `itemFrequencyPlot()` from **arules** which produces a cluster profile where bars are used to represent the relative frequency of product categories in the cluster and a line is used for the relative frequency of the categories in the whole data set. Large differences between the data set and the cluster are interesting since they indicate strong cluster-specific behavior. For better visibility, we only show the categories with a support greater than 5%.

```
R> itemFrequencyPlot(cluster[[8]], population = s, support = 0.05)
R> itemFrequencyPlot(cluster[[3]], population = s, support = 0.05)
```

The cluster profile of the compact cluster 8 (Figure 3) shows a group of transactions which almost entirely consists of canned beer.

Cluster 3 consists of many transactions containing a large number of items. The cluster's profile in Figure 4 shows that almost all product categories are

on average bought more often in the transactions in this group than in the whole data set. This cluster is interesting for association rule mining since the transactions contain many items and thus represent high sales volume.

As mentioned above, we suppose that the store manager is interested in promoting beef. Because beef is not present in cluster 8, we will concentrate on cluster 3 for mining association rules. We choose relatively small values for support and confidence and, in a second step, we filter only rules which have the product category beef in the right-hand-side.

```
R> rules <- apriori(cluster[[3]], parameter = list(support = 0.005,
+     confidence = 0.2), control = list(verbose = FALSE))
R> beefRules <- subset(rules, subset = rhs %in% "beef")
```

Now the store manager can use a wide array of methods provided by **arules** to analyze the found 181 rules. As an example, we show the 3 rules with the highest confidence values.

```
R> inspect(head(SORT(beefRules, by = "confidence"), n = 3))

  lhs                    rhs     support confidence  lift
1 {tropical fruit,
    root vegetables,
    whole milk,
    rolls/buns}       => {beef} 0.006189     0.3889 2.327
2 {tropical fruit,
    other vegetables,
    whole milk,
    rolls/buns}       => {beef} 0.006631     0.3846 2.302
3 {tropical fruit,
    root vegetables,
    other vegetables,
    rolls/buns}       => {beef} 0.005747     0.3824 2.288
```

4 Conclusion

In this contribution, we showed how the formal framework implemented in the R package **arules** can be extended for transaction clustering by simply providing methods to calculate proximities between transactions and corresponding nearest neighbor classifications. Analogously, proximities between associations can be defined and used for clustering itemsets or rules (Gupta et al. (1999)). Provided that item ontologies or transaction-level covariates are available, these can be employed for interpreting and validating obtained clusterings. In addition, stability of the "interesting" groups found can be assessed using resampling methods, as e.g. made available via the R extension package **clue** (Hornik (2005, 2006)).

References

AGGARWAL, C. C., PROCOPIUC, C. M. and YU, P. S. (2002): Finding localized associations in market basket data. *Knowledge and Data Engineering, 14(1), 51–62.*

AGRAWAL, R., IMIELINSKI, T. and SWAMI, A. (1993): Mining association rules between sets of items in large databases. In: *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data.* ACM Press, 207–216.

BATES, D. and MAECHLER, M. (2005): **Matrix***: A Matrix Package for R.* R package version 0.95-5.

BERRY, M. and LINOFF, G. (1997): *Data Mining Techniques for Marketing, Sales and Customer Support.* John Wiley & Sons.

BORGELT, C. (2003): Efficient implementations of Apriori and Eclat. In: *FIMI'03: Proceedings of the IEEE ICDM Workshop on Frequent Itemset Mining Implementations.*

CHAMBERS, J. M. (1998): *Programming with Data.* Springer, New York. ISBN 0-387-98503-4.

FOWLER, M. (2004): *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* Addison-Wesley Professional, third edition.

GUPTA, G. K., STREHL, A. and GHOSH, J. (1999): Distance based clustering of association rules. In: *Proceedings of the Artificial Neural Networks in Engineering Conference, 1999, St. Louis.* ASME, volume 9, 759–764.

HAHSLER, M., GRÜN, B. and HORNIK, K. (2005): arules – A computational environment for mining association rules and frequent item sets. *Journal of Statistical Software, 14(15), 1–25.*

HAHSLER, M., GRÜN, B. and HORNIK, K. (2006): **arules***: Mining Association Rules and Frequent Itemsets.* R package version 0.2-7.

HASTIE, T., TIBSHIRANI, R. and FRIEDMAN, J. (2001): *The Elements of Statistical Learning.* Springer-Verlag.

HORNIK, K. (2005): A CLUE for CLUster Ensembles. *Journal of Statistical Software, 14(12).*

HORNIK, K. (2006): *CLUE: CLUster Ensembles.* R package version 0.3-3.

KAUFMAN, L. and ROUSSEEUW, P. (1990): *Finding Groups in Data.* Wiley-Interscience Publication.

MAECHLER, M. (2005): **cluster***: Cluster Analysis Extended Rousseeuw et al.* R package version 1.10.2.

PIATETSKY-SHAPIRO, G. (1991): Discovery, analysis, and presentation of strong rules. In: G. Piatetsky-Shapiro and W. J. Frawley (Eds.): *Knowledge Discovery in Databases*, AAAI/MIT Press, Cambridge, MA.

R DEVELOPMENT CORE TEAM (2005): *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

RUSSELL, G. J., BELL, D., BODAPATI, A., BROWN, C. L., JOENGWEN, C., GAETH, G., GUPTA, S. and MANCHANDA, P. (1997): Perspectives on multiple category choice. *Marketing Letters, 8(3), 297–305.*

SNEATH, P. H. (1957): Some thoughts on bacterial classification. *Journal of General Microbiology, 17, 184–200.*