Michael Hahsler and Kurt Hornik and Thomas Reutterer

Implications of probabilistic data modeling for rule mining

Working Paper

http://epub.wu.ac.at/

Implications of Probabilistic Data Modeling for Rule Mining

Michael Hahsler, Kurt Hornik, Thomas Reutterer

Implications of Probabilistic Data Modeling for Rule Mining

Michael Hahsler   Kurt Hornik   Thomas Reutterer

Wirtschaftsuniversität Wien

March 2, 2005

Abstract

Mining association rules is an important technique for discovering meaningful patterns in transaction databases. In the current literature, the properties of algorithms to mine associations are discussed in great detail. In this paper we investigate properties of transaction data sets from a probabilistic point of view. We present a simple probabilistic framework for transaction data and its implementation using the R statistical computing environment. The framework can be used to simulate transaction data when no associations are present. We use such data to explore the ability to filter noise of confidence and lift, two popular interest measures used for rule mining. Based on the framework we develop the measure hyperlift and we compare this new measure to lift using simulated data and a real-world grocery database.

Keywords: data mining, transaction data model, association rules, interest measures.

1   Introduction

Mining association rules (Agrawal, Imielinski, and Swami, 1993) is an important technique for discovering meaningful patterns in transaction databases. An association rule is a rule of the form $X \Rightarrow Y$, where X and Y are two disjoint sets of items (itemsets). The rule means that if we find all items in X in a transaction it is likely that the transaction also contains the items in Y.

A typical application of mining association rules is market basket analysis where point-of-sale data is mined with the goal to discover associations between articles. These associations offer useful and actionable insights to retail managers for product assortment decisions (Brijs, Swinnen, Vanhoof, and Wets, 2004), personalized product recommendations (Lawrence, Almasi, Kotlyar, Viveros, and Duri, 2001), and from adapting promotional activities to shelving. For web-based systems (e.g., e-shops, digital libraries, search engines) associations found between articles/documents/web pages in transaction log files can even be used to automatically and continuously adapt the user interface by presenting associated items together (Lin, Alvarez, and Ruiz, 2002).

Association rules are selected from the set of all possible rules using measures of statistical significance and interestingness. *Support*, the primary measure of significance, is defined as the fraction of transactions in the database which contain all items in the rule. For association rules, a minimum support threshold is used to select the most frequent (and hopefully important) item combinations called *frequent itemsets*. The process of finding these frequent itemsets in a large database is computationally very expensive since it involves searching a lattice which grows in the worst case exponentially in the number of items. In the last decade research has centered on solving this problem and a variety of algorithms were introduced which render search feasible by exploiting various properties of the lattice (see Goethals and Zaki (2004) as a pointer to the currently fastest algorithms).

From the most chosen itemsets rules are generated using measures of interestingness. Numerous measures were suggested in the literature. For association rules, the measure *confidence* was developed (Agrawal et al., 1993). A practical problem is that often too many association rules are
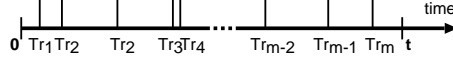
1

Figure 1: Transactions occurring over time following a Poisson process.

produced. In this case, additional interest measures, such as e.g. *lift*, can be used to further filter or rank found rules.

In this paper we will start with modeling transaction data. We will present a simple probabilistic framework for transaction data which is based on independent Bernoulli trials. Using this framework, we will present a small probabilistic data modeling exercise for transaction data and show the implications of this exercise for support, confidence and lift. Based on the findings from the framework we will develop a new interest measure called *hyperlift* and compare its performance on simulated data and a real-world grocery database.

The paper is structures as follows: The next section develops the probabilistic framework. In sections 3 and 4 we investigate the implications of the framework for the interest measures confidence and lift. In section 5 we develop the measure hyperlift. Finally, section 6 compares lift and hyperlift on real-world data.

2 A simple probabilistic framework for transaction data

A transaction database consists of a series of transactions where each transaction contains a subset of the available items. We analyze transactions which are recorded in a fixed time interval of length t. In figure 1 an example time interval is shown as an arrow with markings at the points in time when the transactions denoted by Tr_1 to Tr_m occur. We assume that transactions occur randomly following a (homogeneous) Poisson process with parameter θ. The number of transactions m in the time interval is then Poisson distributed with parameter θt:

$$P(M = m) = \frac{e^{-\theta t}(\theta t)^m}{m!} \tag{1}$$

We denote the items which occur in the database by $L = \{l_1, l_2, \ldots, l_n\}$ with n being the number of different items. For the framework we assume that all items occur independently of each other and that for each item $l_i \in L$ there exists a fixed probability p_i of being contained in a transaction. Each transaction is then the result of n independent Bernoulli trials, one for each item with success probabilities given by the vector $p = (p_1, p_2, \ldots, p_n)$. Figure 2 contains the typical representation of an example database as a binary incidence matrix with one column for each item. Each row labeled Tr_1 to Tr_m contains a transaction, where a 1 indicates presence and a 0 indicates absence of the corresponding item in the transaction. Additionally, the success probability for each item is given in an additional row labeled p and the row labeled c contains the number of transactions each items is contained in (sum of the ones per column).

Following the model, c_i, the observed number of transactions item l_i is contained in, can be interpreted as a realization of a random variable C_i. Under the condition of a fixed number of transactions m this random variable has a binomial distribution.

$$P(C_i = c_i | M = m) = \binom{m}{c_i} p_i^{c_i} (1 - p_i)^{m - c_i} \tag{2}$$

However, since for a fixed time interval the number of transactions is not fixed, the unconditional distribution gives:

	items				
	I_1	I_2	I_3	...	I_n
p	0.005	0.01	0.0003	...	0.025
Tr_1	0	1	0	...	1
Tr_2	0	1	0	...	1
Tr_3	0	1	0	...	0
Tr_4	0	0	0	...	0
.	.	.	.	.	.
.	.	.	.	.	.
.	.	.	.	.	.
Tr_{m-1}	1	0	0	...	1
Tr_m	0	0	1	...	1
c	99	201	7	...	411

(left axis label: transactions)

Figure 2: Example transaction database with success probabilities p and transaction counts per item c.

$$
\begin{aligned}
P(C_i = c_i) &= \sum_{m=c_i}^{\infty} P(C_i = c_i | M = m) \cdot P(M = m) \\
&= \sum_{m=c_i}^{\infty} \binom{m}{c_i} p_i^{c_i} (1-p_i)^{m-c_i} \frac{e^{-\theta t}(\theta t)^m}{m!} \\
&= \frac{e^{-\theta t}(p_i \theta t)^{c_i}}{c_i!} \sum_{m=c_i}^{\infty} \frac{((1-p)\theta t)^{m-c_i}}{(m-c_i)!} \\
&= \frac{e^{-p_i \theta t}(p_i \theta t)^{c_i}}{c_i!}
\end{aligned}
\tag{3}
$$

The sum term in the last but one line in equation 3 is an exponential series with the limiting sum $e^{(1-p_i)\theta t}$. With this it is easy to show that the unconditional probability distribution of each C_i has a Poisson distribution with parameter $p_i \theta t$. For short we will use $\lambda_i = p_i \theta t$ and introduce the parameter vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ of the Poisson distributions for all items. This parameter vector can be calculated from the success probability vector p and vice versa by the linear relationship $\lambda = p\theta t$ where θ is the intensity with which transactions occur and t is the length of the observed time interval.

For a given database, the values of the parameter θ and the success vectors p or alternatively λ are unknown but can be estimated from the database. The best estimate for θ from a single database is m/t. The simplest estimate for λ is to use the observed counts for each item. However, this is only a very rough estimate which especially gets unreliable for small counts. There exist more sophisticated estimation approaches. For example, DuMouchel and Pregibon (2001) use the assumption that the parameters of the count processes for items in a database are distributed according to a continuous parametric density function. This additional information can improve estimates over using just the observed counts. DuMouchel and Pregibon use the mixture of two gamma distributions as the distribution for the parameters for their analysis of international calling behavior.

For simplicity we assume for the following simulation that the parameters in λ are chosen from a single gamma distribution. The gamma distribution is a very flexible distribution which allows to fit a wide range of empirical data and the resulting Poisson-Gamma mixture model has applications in many fields including related problems in market research (Johnson, Kotz, and Kemp, 1993). We will simulate the counts c_i, for $n = 200$ different items over a $t = 30$ day period with transaction intensity $\theta = 300$ (`theta` in the code) transactions per day. For the gamma

distribution we use the parameters $k = 0.75$ and $a = 250$.

First, we choose the number of transactions m in time interval t from a Poisson distribution with parameter θt.

```
> m <- rpois(1, theta * t)
> m

[1] 8885
```

Next, we generate the parameter vector p by drawing values for the n λ_i from the gamma distribution and transform them into probabilities p_i by dividing by the database size m. For better visualization later on, we sort the items by probability so that $p_i \geq p_{i+1}$ for $i = 1, 2, \ldots, (n-1)$.

```
> p <- sort(rgamma(n, shape = k, scale = a)/m, decreasing = TRUE)
```

Now we can simulate the transactions in the database by m Bernoulli tries for each of the n items. The result is the database represented as the $m \times n$ incidence matrix Tr.

```
> Tr <- matrix(rbinom(m * n, 1, p), ncol = n, byrow = TRUE)
```

From the incidence matrix we calculate the item counts c_i as the column sums. These counts are the numbers of transactions in which the items (also called 1-itemsets since they can be treated as itemset which consists only of 1 item) appear in the database.

```
> c <- (apply(Tr, 2, sum))
```

We can directly calculate the support of each item from the transaction counts. Support for an itemset Z is defined as (Agrawal and Srikant, 1994):

$$\text{supp}(Z) = c_Z/m, \tag{4}$$

where c_Z is the count of the itemset and m is the number of transactions in the database.

```
> supp1 <- c/m
> summary(supp1)

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.0000  0.0063  0.0154  0.0243  0.0323  0.1640
```

```
> plot(supp1, type = "h", xlab = "items", ylab = "support")
```

The summary statistics of support over all 1-itemsets show that there is a significant difference between the mean and the median. This indicates a skewed count distribution. The count distribution of the simulated counts c_i is presented in the plot in figure 3. As typical for transaction data, only few items have a relatively high support while most items appear rather infrequently.

The number of items which appear in the database $0, 1, 2, \ldots$ times in a Poisson-Gamma mixture model follows a Negative Binomial distribution with the parameters determined by the parameters of the gamma mixing distribution. The Negative Binomial distribution can be used to predict the expected number of items which are more frequent than a set minimum support.

```
> h <- hist(c, breaks = c(0:max(c)), plot = FALSE)
> plot(c(0:(length(h$counts) - 1))/m, y = n - cumsum(h$counts),
+     type = "l", xlab = "minimum support", ylab = "frequent items")
> lines(x = c(0:max(c))/m, (1 - cumsum(dnbinom(c(0:max(c)), size = k,
+     prob = 1/(1 + a)))) * n, col = "red", lty = 2)
```
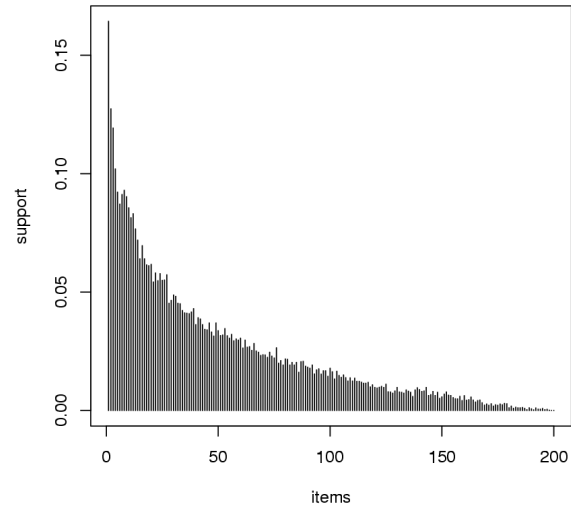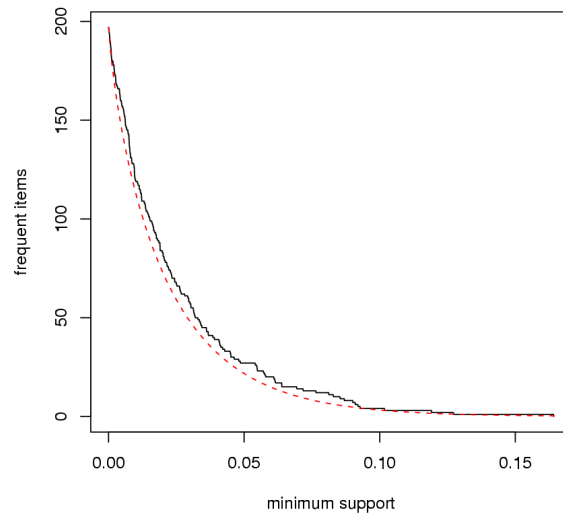
Figure 3: Simulated item support.

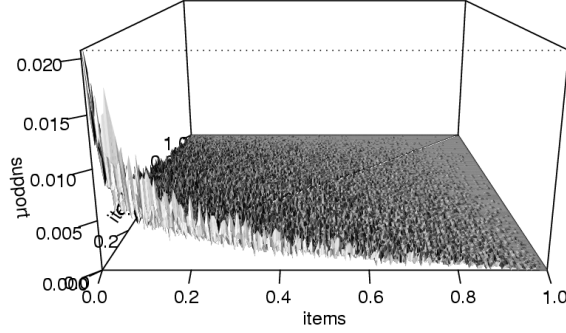Figure 4: Plot of frequent items by minimum support.

Figure 5: Simulated 2-itemset support

In figure 4 we show the number of frequent items in the database depending on the used minimum support. The dashed line represents the number of items expected using the Negative Binomial Distribution.

Next, we extend the framework to the occurrences of 2-itemsets which corresponds to co-occurrences of two items in transactions in the database. The counts for the 2-itemsets can be represented by a $n \times n$ matrix where each cell $c_{i,j}$ contains the co-occurrence count for two items $i, j \in L$. The $n \times n$ matrix is symmetric with $c_{i,j} = c_{j,i}$ and the diagonal with $i = j$ contains the counts of the individual items.

We can calculate the values $c_{i,j}$ from the database by counting the number of transactions for each item combination in the $n \times n$ co-occurrence matrix (denoted by `c2` in the code). We discard the counts in the diagonal where $i = j$ since by definition itemsets cannot contain the same item more than once. Using the counts we can calculate the support values of all 2-itemsets.

```
> c2 <- sapply(1:n, function(i) {
+      apply(Tr[, i] & Tr[, 1:n], 2, sum)
+ })
> diag(c2) <- NA
> supp2 <- c2/m
> summary(as.vector(supp2))

      Min.    1st Qu.     Median       Mean    3rd Qu.        Max.        NA's
  0.000000   0.000000   0.000225   0.000585   0.000675   0.020700 200.000000

> persp(supp2, expand = 0.5, ticktype = "detailed", border = 0,
+      shade = 1, zlab = "support", xlab = "items", ylab = "items")
```

The summary statistics for support show that for many item combinations we obtained very small support values, a property which seem typical for transaction data. We visualize the support distribution over all possible 2-itemsets with a 3D-plot in figure 5. The x and y axes represent the items and the simulated support value is represented by the z axis. The items on the x and y axes are sorted by their individual occurrence probability starting with the most frequent item. Of course, the 2-itemsets consisting of the most frequent items have the highest support (the most frequent items appear in the plot in the front and to the left because of the ordering of the items).

Since in the model all items occur following independent Poisson processes, the count in each cell of the $n \times n$ co-occurrence matrix can be modeled by n^2 random variables $C_{i,j}$ which follow hypergeometric distributions with the marginal counts $c_{i,\cdot}$ and $c_{\cdot,j}$ defining the parameters. The hypergeometric distribution arises for the so-called urn problem, where the urn contains m white balls and n black balls. The number of white balls drawn with k tries without replacement follows a
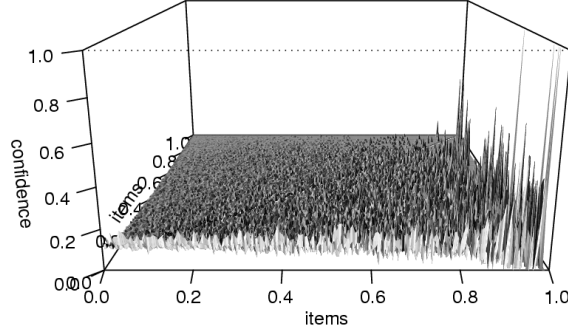
Figure 6: Simulated 2-itemset confidence

hypergeometric distribution. This model is applicable for counting co-occurrences for independent items l_i and l_j in the following way: We already know that item l_j occurs in c_j transactions, therefore, for l_j the database can be represented as an urn which contains c_j "good" transactions (white balls) and $m - c_j$ "bad" transactions (black balls). For each item $l_i \neq l_j$, we draw without replacement the c_i transactions it is contained in. The number of "good" transactions (transactions which contain item l_j) which are drawn for item l_i has a hypergeometric distribution.

The framework can be easily expanded to itemsets of arbitrary length by combining smaller itemsets with an additional item. For example, the expected counts of a 4-itemset follows a hypergeometric distribution with the parameters depending on the counts of a 3-item subset and the count of the fourth item.

3 Implications for the interest measure confidence

We used the simple framework which implies independence between all items to simulated the counts and support for individual items and 2-itemsets.

Confidence is defined by Agrawal et al. (1993) as

$$\mathrm{conf}(X \Rightarrow Y) = \frac{\mathrm{supp}(X + Y)}{\mathrm{supp}(X)}, \tag{5}$$

where X and Y are two disjoint itemsets. Often confidence is understood as the conditional probability $P(Y|X)$ (e.g., Hipp, Güntzer, and Nakhaeizadeh (2000)), where the definition above is seen as an estimate for this probability. From our 2-itemsets we can generate rules of the from $l_i \Rightarrow l_j$, where $i, j = 1, 2, \ldots, n$ and $i \neq j$. We calculate confidence for the $n(n-1)$ possible rules in the data set.

```
> conf2 <- supp2/supp1
> summary(as.vector(conf2))

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
  0.0000   0.0000   0.0116   0.0242   0.0332   1.0000 598.0000


> persp(conf2, expand = 0.5, ticktype = "detailed", border = 0,
+     shade = 1, zlab = "confidence", xlab = "items", ylab = "items")
```

The summary shows that confidence values are generally very low which reflect the fact that there are no associations in the data. However, the maximum indicates some rules with confidence of one. The plot in figure 6 shows that confidence increases with the item in the right-hand-side

7

of the rule getting more frequent and the item in the left-hand-side getting less frequent. This behavior directly follows the way confidence is calculated. But the fact that confidence clearly favors some rules makes the measure problematic when it comes to ranking rules by their interest.

4 Implications for the interest measure lift

Typically, rules mined using minimum support (and confidence) are filtered or ordered using their lift value. The measure lift (also called interest (Brin, Motwani, Ullman, and Tsur, 1997)) is defined on rules of the form $X \Rightarrow Y$ as

$$\text{lift}(X \Rightarrow Y) = \frac{P(Y \cup X)}{P(X)P(Y)} = \frac{P(Y|X)}{P(Y)}, \tag{6}$$

where X and Y are two disjoint itemsets. $P(Y \cup X)$ is the probability that all items in X and Y appear together in transactions. The product of the individual probabilities $P(X)P(Y)$ represents the expected probability for finding the items in X and Y together in transactions if they occur independently given their observed individual probabilities. To calculate lift, confidence and support values are normally used as estimates for the probabilities resulting in

$$\text{lift}(X \Rightarrow Y) = \frac{\text{conf}(X \Rightarrow Y)}{\text{supp}(Y)} \tag{7}$$

A lift value of 1 indicates that the items are co-occurring in the database as expected under independence. Values greater than one indicate that the items are associated. For marketing applications it is generally argued that $lift \gg 1$ indicates complementary products and $lift \ll 1$ indicates substitutes.

For the rules generated from 2-itemsets, lift can be directly calculated from the already computed confidence and support values.

```
> lift <- conf2/matrix(supp1, ncol = n, nrow = n, byrow = TRUE)
> summary(as.vector(lift))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  0.000   0.000   0.817   0.987   1.240 159.000 994.000

> persp(lift, expand = 0.5, ticktype = "detailed", border = 0,
+     shade = 1, zlab = "lift", xlab = "items", ylab = "items")
> length(which(lift > 2))

[1] 3424
```

Interestingly, although the data do not contain associations, many zeros and some extremely high lift values are found. For example, 3424 rules are found with a lift greater than 2. For diagnostics we visualize the lift values in figure 7. The plot shows that many zeros and the highest lift values are achieved for rare items (in the plot in the back right-hand corner). These lift values are artifacts which result for items with very low counts. The product of the probabilities for such rare items is close to zero and if such items co-occur together once by chance, an extremely high lift will result.

To counter this problem. for most applications a minimum support threshold is used to filter all itemsets and thus also rules which do not appear in the database frequently enough to be of interest. We discard all 2-itemsets which do not satisfy a minimum support of 0.1%.

```
> min_supp <- 0.001
> length(lift[supp2 >= min_supp])

[1] 7096
```
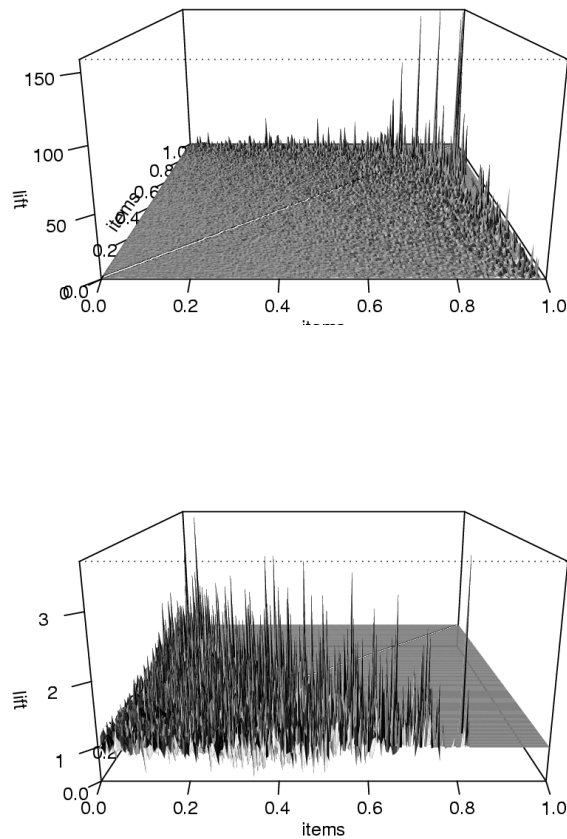
Figure 8: Lift for 2-itemsets using a minimum support of 0.5%.

```
> summary(lift[supp2 >= min_supp])

  Min. 1st Qu.  Median    Mean 3rd Qu.     Max.     NA's
 0.462   0.918   1.060   1.120   1.250    3.690  200.000

> lift[supp2 < min_supp] <- 1
> persp(lift, expand = 0.5, ticktype = "detailed", border = 0,
+     shade = 1, zlab = "lift", xlab = "items", ylab = "items")
> length(which(lift > 2))

[1] 130
```

Requiring support leaves 7096 rules. The summary statistics show that most supported rules have now a lift value around 1, the value which indicates that no associations are NA but there are still 130 rules with a lift greater than 2. The plot in figure 8 clearly shows lift's tendency to produce higher values for rules containing one or two less frequent items. This indicates that the lift measure performs poorly to filter random noise in transaction data especially if we are also interested in relatively rare items. Furthermore, if lift is used to rank discovered rules, there is a systematic tendency towards favoring rules with less frequent items.

5 Developing the measure hyperlift

In this section we will show how the knowledge that the co-occurrence counts of independent items can be seen as realizations of hypergeometric distributed random variables with known parameters can be used to better filter random noise.

The expected value of a random variable C with a hypergeometric distribution is

$$E[C] = kw/(w + b), \qquad (8)$$

where the parameter k represents the number of tries, w is the number of white balls, and b is the number of black balls. Applied to co-occurrence counts for the two items l_i and l_j in a transaction database this gives

$$E[C_{i,j}] = c_i c_j / m, \qquad (9)$$

where m is the number of transactions in the database. With using c_i/m as an estimate for $P(l_i)$ we can rewrite lift as

$$\text{lift}(l_i \Rightarrow l_j) = \frac{P(l_i + l_j)}{P(l_i)P(l_j)} = \frac{c_{i,j}}{E[C_{i,j}]} \qquad (10)$$

For items with a relatively high occurrence frequency using the expected value for lift works well. However, for relatively infrequent items, which are the majority in most transaction databases, using the ratio of the observed count to the expected value is problematic. For example, let us assume that we have the two independent items l_i and l_j, and both items have a support of 1% in the database. Then, $E[C_{i,j}]$ is:

```
> c_i <- 0.01 * m
> c_j <- 0.01 * m
> c_i * c_j/m

[1] 0.8885
```

However, if we choose randomly there is a $P[C_{i,j} > 1]$ of

```
> phyper(1, m = c_j, n = m - c_j, k = c_i, lower.tail = FALSE)

[1] 0.224
```

Therefore there is a substantial chance that we will see a lift value of $2, 3$ or even higher. Given the huge number of itemsets and rules generated by combining items (especially when also considering itemsets containing more than one item), this is very problematic. Using larger databases with more transactions reduces the problem, however, there are several drawbacks of very large databases. Some drawbacks are:

- They are more time-consuming to mine which might lead to the need to sample transactions (e.g., Mannila, Toivonen, and Verkamo (1994)). However, using a sampled database suffers again from the problem shown above.

- They are usually collected over a long period of time and thus may contain outdated information. For example, in a supermarket the articles offered may have changed.

To address the problem, we adapt lift by using instead of the expected value $E[C_{X,Y}]$, where X and Y are itemsets, a quantile denoted by δ of the hypergeometric distribution. For the quantile $Q_\delta[C_{X,Y}]$ we have $Pr(C_{X,Y} \leq Q_\delta[C_{X,Y}]) = \delta$. We call the resulting measure hyperlift, which we define as

$$\text{hyperlift}(X \Rightarrow Y) = \frac{c_{X,Y}}{Q_\delta[C_{X,Y}]}. \qquad (11)$$

In the following we will use $\delta = 0.99$ which results in hyperlift being more conservative compared to lift. The measure can be interpreted as the number of times the observed co-occurrence count
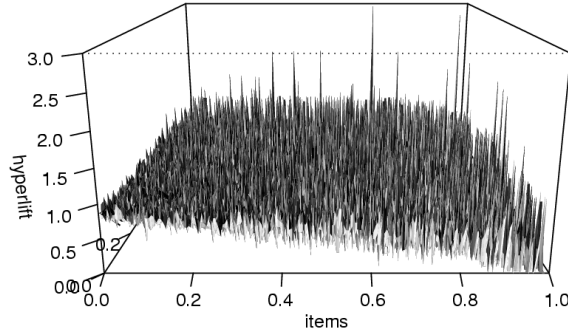
Figure 9: Hyperlift for 2-itemsets in the simulated data set.

$c_{X,Y}$ is higher than the highest count we expect 99% of the time. This means, that hyperlift for a rule with independent items will exceed 1 only in 1% of the cases, and then only slightly. Next, we calculate hyperlift with $\delta = 0.99$ for the simulated database.

```
> calc_hyperbase <- function(ci, cj) {
+     qhyper(0.99, m = cj, n = m - cj, k = ci)
+ }
> hyperlift <- c2/outer(c, c, FUN = calc_hyperbase)
> hyperlift[is.infinite(hyperlift)] <- NA
> summary(as.vector(hyperlift))

    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.     NA's
   0.000    0.000    0.333    0.345    0.571    3.000 1492.000

> persp(hyperlift, shade = 1, ticktype = "detailed", border = 0,
+     expand = 0.5, zlab = "hyperlift", xlab = "items", ylab = "items")
> length(which(hyperlift > 2))

[1] 2
```

Hyperlift (we eliminated the hyperlift values which give infinity since they result in a division by zero) produces for the simulated data set mostly values below one (only 2) rules have a lift greater than 2. In figure 9 we see that the hyperlift values are generally smaller than 1 and more evenly distributed indicating that hyperlift filters the random co-occurrences better than lift. However, hyperlift also shows a systematic dependency on the occurrence probability of items leading to smaller values for rules with less frequent items.

So far we only used an artificially generated data set which follows a model which we also used in developing the hyperlift measure. Therefore, we will test hyperlift in the next section on a real-world grocery database.

6   Comparing lift and hyperlift on a grocery database

We use 1 month of real-world point-of-sale transaction data from a local grocery outlet. To reduce the number of items we use categories (e.g., popcorn) instead of the individual brands. In the available $m = 9835$ transaction we found $n = 169$ different categories for which articles were purchased. We preprocessed the data and stored the count information in the same format as above (a vector called c for the count frequencies of individual items, and a matrix called c2 for the co-occurrence counts of two items). First, we compute support and confidence and present some descriptive statistics.
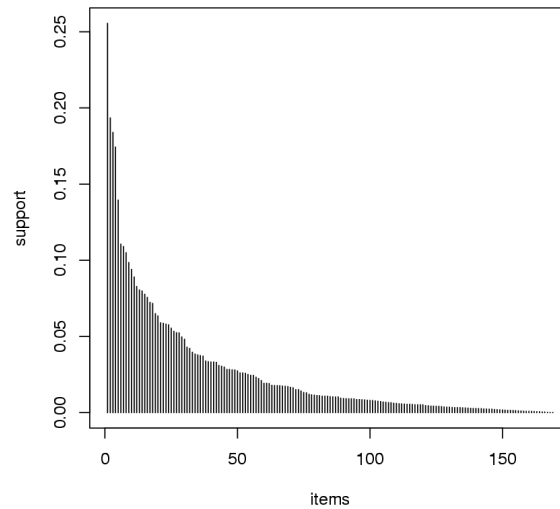
Figure 10: Item support in the grocery database.

```
> supp1 <- c/m
> summary(supp1)

    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000102 0.003860 0.010500 0.026100 0.031000 0.256000

> plot(supp1, type = "h", xlab = "items", ylab = "support")


> supp2 <- c2/m
> summary(as.vector(supp2))

      Min.    1st Qu.     Median       Mean    3rd Qu.       Max.       NA's
  0.000000   0.000000   0.000203   0.000983   0.000712   0.074800 169.000000

> persp(supp2, shade = 1, expand = 0.5, ticktype = "detailed",
+     border = 0, zlab = "support", xlab = "items", ylab = "items")

> conf2 <- supp2/supp1
> summary(as.vector(conf2))

    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  0.0000  0.0000  0.0140  0.0412  0.0516  1.0000 169.0000

> persp(conf2, shade = 1, expand = 0.5, ticktype = "detailed",
+     border = 0, zlab = "confidence", xlab = "items", ylab = "items")
```

The summary statistics of support and the plot in figure 10 show that the grocery database contains items with a similarly skewed count distribution as the simulated data set used above. Both have an almost identical mean and median. Also the distributions of the 2-itemset support and confidence in figures 11 and 12 exhibit a similar form as the simulated data. However, both measures have a large number of combinations which reach several times higher values than in the simulated data (compare figures 11 and 12 with 5 and 6 above). This indicates that the grocery database contains associated items. Next, we calculate the lift measure.
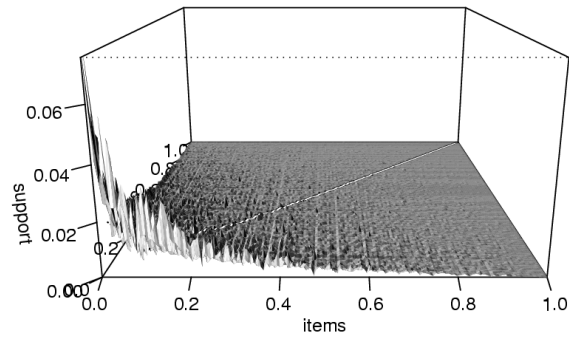
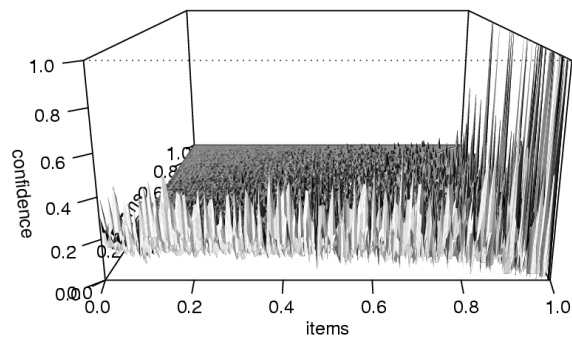Figure 11: 2-itemset support in the grocery database.

Figure 12: 2-itemset confidence in the grocery database.

13

```
> lift <- conf2/matrix(supp1, ncol = n, nrow = n, byrow = TRUE)
> summary(as.vector(lift))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.00    0.00    1.23    1.78    2.07  224.00  169.00

> persp(lift, expand = 0.5, ticktype = "detailed", border = 0,
+     shade = 1, zlab = "lift", xlab = "items", ylab = "items")
> length(which(lift > 2))

[1] 7544
```

The database contains 7544 rules with a lift greater than 2. Figure 13 shows that, as before, the rules with the least frequent items have the highest lift values. To assess the usefulness of the rules found by lift, we inspect the itemsets with the highest lift values. Since lift is a symmetric measure each line in the list actually represents the 2 rules $l_i \Rightarrow l_j$ and $l_j \Rightarrow l_i$.

```
> lift.lower <- lift
> lift.lower[upper.tri(lift.lower, diag = TRUE)] <- NA
> best <- which(lift.lower >= sort(lift.lower, decreasing = TRUE)[10],
+     arr.ind = TRUE)
> best <- data.frame(l_i = names(c[best[, 1]]), l_j = names(c[best[,
+     2]]), supp = supp2[best], lift = lift[best])
> best[order(best$lift, decreasing = TRUE), ]

                        l_i                     l_j      supp    lift
9   preservation products          cocoa drinks 0.0001017 223.52
4              baby food      finished products 0.0001017 153.67
3              baby food                  soups 0.0001017 146.79
6   preservation products        abrasive cleaner 0.0001017 140.50
10        baby cosmetics                  cream 0.0001017 126.09
2   sound storage medium frozen potato products 0.0001017 118.49
8                   bags                tidbits 0.0001017 106.90
5   preservation products                 spices 0.0001017  96.42
7          kitchen utensil                  fish 0.0001017  84.78
1              baby food                cake bar 0.0001017  75.65
```

It is easy to see that these rules are not useful since the rules only contain very rare items. In fact, the items in all rules have a co-occurrence count of only one.

To filter the rare items, we remove all itemsets which do not satisfy a minimum support of 0.1%.

```
> min_supp <- 0.001
> length(lift[supp2 >= min_supp])

[1] 6131

> summary(lift[supp2 >= min_supp])

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
  0.311   1.300   1.620   1.770   2.030  13.000  169.000

> lift.supp <- lift
> lift.supp[supp2 < min_supp] <- 1
> persp(lift.supp, expand = 0.5, ticktype = "detailed", border = 0,
+     shade = 1, zlab = "lift", xlab = "items", ylab = "items")
> length(which(lift.supp > 2))
```
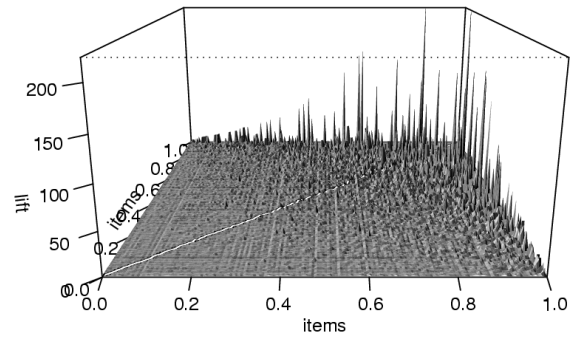
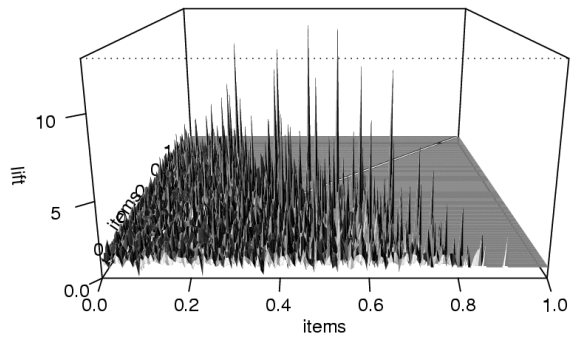Figure 13: Lift values for the grocery database.

Figure 14: Lift for 2-itemsets for items with support of 0.1% in the grocery database.

```
[1] 1586
```

The support threshold leaves 6131 rules of which 1586 have a lift higher than 2. Figure 14 shows that, as for the simulated data, the combinations including the least frequent items tend to have higher lift values. This indicates that lift values will be heavily influenced by the set minimum support value. We inspect the rules with the highest lift values.

```
> lift.lower <- lift.supp
> lift.lower[upper.tri(lift.lower, diag = TRUE)] <- NA
> best <- which(lift.lower >= sort(lift.lower, decreasing = TRUE)[20],
+     arr.ind = TRUE)
> best <- data.frame(l_i = names(c[best[, 1]]), l_j = names(c[best[,
+     2]]), supp = supp2[best], lift = lift[best])
> best[order(best$lift, decreasing = TRUE), ]

                          l_i            l_j     supp   lift
20             mayonnaise         mustard 0.001423 12.965
8   Instant food products hamburger meat 0.003050 11.421
15               softener        detergent 0.001118 10.600
16                 liquor red/blush wine 0.002135 10.025
6                   flour           sugar 0.004982  8.463
4                 popcorn     salty snack 0.002237  8.192
11        processed cheese             ham 0.003050  7.071
9                  sauces hamburger meat 0.001220  6.684
3             meat spreads    cream cheese 0.001118  6.605
14 house keeping products        detergent 0.001017  6.346
13               pet care        cat food 0.001322  6.003
2         processed cheese     white bread 0.004169  5.975
17                  flour   baking powder 0.001830  5.950
18           roll products           flour 0.001017  5.695
1            specialty fat       margarine 0.001220  5.692
19                mustard      canned fish 0.001017  5.632
7                   pasta hamburger meat 0.002745  5.487
12                   rice     hard cheese 0.001017  5.441
5           baking powder           sugar 0.003254  5.432
10  chocolate marshmallow           candy 0.001423  5.262
```

Most rules in the list look reasonable. However, in the list we can see that most rules have a support value close to the used minimum support threshold of 0.1% which results from lifts tendency to favor less frequent items. Another disadvantage is that using the support threshold we filtered many rules which might still be of interest.

We calculate hyperlift with $\delta = 0.99$ for the data set.

```
> calc_hyperbase <- function(ci, cj) {
+     qhyper(0.99, m = cj, n = m - cj, k = ci)
+ }
> hyperlift <- c2/outer(c, c, FUN = calc_hyperbase)
> hyperlift[is.infinite(hyperlift)] <- NA
> summary(as.vector(hyperlift[c2 != 0]))

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 0.0588  0.5000  0.7000  0.7530  1.0000  4.2900 183.0000

> persp(hyperlift, shade = 1, ticktype = "detailed", border = 0,
+     expand = 0.5, zlab = "hyperlift", xlab = "items", ylab = "items")
> length(which(hyperlift > 2))
```
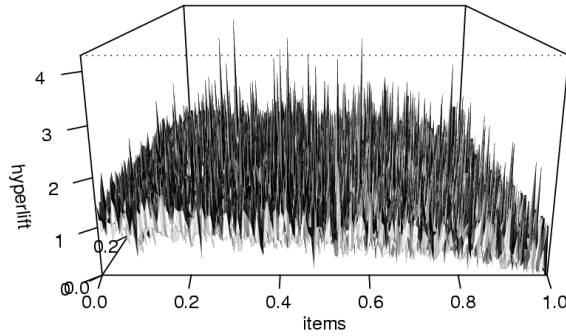
Figure 15: Hyperlift for 2-itemsets in the grocery database.

[1] 82

The summary statistics shows that most rules have a value well below 1, an indication that do not observe many co-occurrence counts which can not be explained by chance. However, the database contains 82 rules with hyperlift values greater than 2. In figure 15, there is no clear pattern visible which would indicate that a high hyperlift is more likely to occur for rules depending on the frequency of its items. We analyze the rules with the highest hyperlift values.

```
> hyperlift.lower <- hyperlift
> hyperlift.lower[upper.tri(hyperlift.lower, diag = TRUE)] <- NA
> best <- which(hyperlift.lower >= sort(hyperlift.lower, decreasing = TRUE)[20],
+     arr.ind = TRUE)
> best <- data.frame(l_i = names(c[best[, 1]]), l_j = names(c[best[,
+     2]]), supp = supp2[best], hyperlift = hyperlift[best], lift = lift[best])
> best[order(best$hyperlift, decreasing = TRUE), ]
```

	l_i	l_j	supp	hyperlift	lift
11	Instant food products	hamburger meat	0.0030503	4.286	11.421
9	flour	sugar	0.0049822	4.083	8.463
15	liquor	red/blush wine	0.0021352	3.500	10.025
17	cooking chocolate	baking powder	0.0007117	3.500	15.826
18	mayonnaise	mustard	0.0014235	3.500	12.965
6	processed cheese	white bread	0.0041688	3.154	5.975
7	popcorn	salty snack	0.0022369	3.143	8.192
13	processed cheese	ham	0.0030503	3.000	7.071
3	liquor	bottled beer	0.0046772	2.875	5.241
14	softener	detergent	0.0011185	2.750	10.600
8	baking powder	sugar	0.0032537	2.667	5.432
5	ham	white bread	0.0050839	2.632	4.640
2	herbs	root vegetables	0.0070158	2.556	3.956
4	berries	whipped/sour cream	0.0090493	2.543	3.797
16	specialty vegetables	pickled vegetables	0.0005084	2.500	16.435
19	cleaner	female sanitary products	0.0005084	2.500	16.392
20	abrasive cleaner	cleaner	0.0005084	2.500	28.100
10	pasta	hamburger meat	0.0027453	2.455	5.487
12	sauces	hamburger meat	0.0012201	2.400	6.684
1	beef	root vegetables	0.0173869	2.342	3.040

All presented rules make sense intuitively. From the experiment with simulated data we know that hyperlift somewhat favors rules with more frequent items. However, the resulting list shows that the set of rules with highest hyperlift contains rules with support varying from very rare to relatively frequent. If we compare hyperlift and lift for the selected rules, we see that all rules selected by hyperlift also have a relatively high lift. However, the ranking of rules between the two measures is different and hyperlift is also able to deal with very infrequent rules.

7 Conclusion

In this contribution we developed a simple probabilistic framework for transaction data. Using this framework to simulate a transaction database which does not contain any associations and calculated co-occurrence counts, support, confidence and lift. The distributions of these measures show that the simulated data is very close to real-world grocery data.

We showed how the interest measures are systematic influenced by the frequencies of items in the corresponding itemsets or rules. In particular, we discovered that the measure lift performs poorly to filter random noise and always produces the highest values for the rules containing the least frequent items.

Based on the presented framework, we developed a new interest measure called hyperlift which is better able to filter random noise in the simulated database and can also deal with very infrequent items.

The presented framework and the measure hyperlift provide many possibilities for further research. Some directions are:

- The framework does not yet include dependencies between items. The explicit modeling of dependencies would enable us to simulate transaction data sets with properties close to real data and with known associations. Such a framework would provide an ideal test bed to evaluate and compare different approaches and interest measures.

- The measure hyperlift only identifies complementary items, i.e., items which are associated. Often finding substitutes is also important. Hyperlift needs to be adapted for this use.

- The experiment with simulated data showed that there exists a small but systematic influence of the occurrence frequency of items on the hyperlift measure. This makes comparing hyperlift between rules with items of very different frequency inaccurate. A possible solution would be to measure interest on the p-value scale instead.

References

R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, pages 207–216, Washington D.C., May 1993.

Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *Proc. 20th Int. Conf. Very Large Data Bases, VLDB*, pages 487–499, Santiago, Chile, Sept 1994.

Tom Brijs, Gilbert Swinnen, Koen Vanhoof, and Geert Wets. Building an association rules framework to improve product assortment decisions. *Data Mining and Knowledge Discovery*, 8(1): 7–23, 2004.

Sergey Brin, Rajeev Motwani, Jeffrey D. Ullman, and Shalom Tsur. Dynamic itemset counting and implication rules for market basket data. In *SIGMOD 1997, Proceedings ACM SIGMOD International Conference on Management of Data*, pages 255–264, Tucson, Arizona, USA, May 1997.

William DuMouchel and Daryl Pregibon. Empirical bayes screening for multi-item associations. In F. Provost and R. Srikant, editors, *Proceedings of the ACM SIGKDD Intentional Conference on Knowledge Discovery in Databases & Data Mining (KDD01)*, pages 67–76. ACM Press, 2001.

Bart Goethals and Mohammed J. Zaki. Advances in frequent itemset mining implementations: Report on FIMI'03. *SIGKDD Explorations*, 6(1):109–117, 2004.

Jochen Hipp, Ulrich Güntzer, and Gholamreza Nakhaeizadeh. Algorithms for association rule mining - a general survey and comparison. *SIGKDD Explorations*, 2(2):1–58, 2000.

Norman L. Johnson, Samuel Kotz, and Adrienne W. Kemp. *Univariate Discrete Distributions*. John Wiley & Sons, New York, 2nd edition, 1993.

Richard D. Lawrence, George S. Almasi, Vladimir Kotlyar, Marisa S. Viveros, and Sastry Duri. Personalization of supermarket product recommendations. *Data Mining and Knowledge Discovery*, 5(1/2):11–32, 2001.

Weiyang Lin, Sergio A. Alvarez, and Carolina Ruiz. Efficient adaptive-support association rule mining for recommender systems. *Data Mining and Knowledge Discovery*, 6:83–105, 2002.

Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. Efficient algorithms for discovering association rules. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *AAAI Workshop on Knowledge Discovery in Databases (KDD-94)*, pages 181–192, Seattle, Washington, 1994. AAAI Press.