Value-based Customer Grouping from Large Retail Data-sets

Alexander Strehl and Joydeep Ghosh

The University of Texas at Austin, Austin, TX 78712-1084, USA

ABSTRACT

In this paper, we propose OPOSSUM, a novel similarity-based clustering algorithm using constrained, weighted graph-partitioning. Instead of binary presence or absence of products in a market-basket, we use an extended "revenue per product" measure to better account for management objectives. Typically the number of clusters desired in a database marketing application is only in the teens or less. OPOSSUM proceeds top-down, which is more efficient and takes a small number of steps to attain the desired number of clusters as compared to bottom-up agglomerative clustering approaches. OPOSSUM delivers clusters that are balanced in terms of either customers (samples) or revenue (value). To facilitate data exploration and validation of results we introduce CLUSION, a visualization toolkit for high-dimensional clustering problems. To enable closed loop deployment of the algorithm, OPOSSUM has no user-specified parameters. Thresholding heuristics are avoided and the optimal number of clusters is automatically determined by a search for maximum performance. Results are presented on a real retail industry data-set of several thousand customers and products, to demonstrate the power of the proposed technique.

Keywords: Data mining, clustering, graph partitioning, retail marketing

1. INTRODUCTION

1.1. Overview

Knowledge discovery in databases[1] often requires clustering the data into a number of distinct groups in an effective and efficient manner. Good clusters show high similarity within a group and low similarity between two different groups. Automatically generated document clusters, for example, can provide a structure for organizing large bodies of text for efficient browsing and searching. Customer buying behavior groups are useful for presenting summary information to middle-level management for making marketing decisions.

Current enterprise resource management systems accurately record all customers' purchase history, that can then be mined for patterns to obtain a competitive advantage. A typical market-basket database may involve millions of customers and several thousand product-lines. Most customers only buy a small subset of products. For each product a customer could potentially buy, a feature (attribute) is recorded in the data-set. A feature usually corresponds to some property (e.g., quantity, price, profit) of the goods actually purchased. If we try to consider each customer as a multi-dimensional data point and then try to cluster customers based on their buying behavior, the problem differs from classic clustering scenarios in several ways:

- *High dimensionality.* The number of features is very high and may even exceed the number of samples. So one has to face with the curse of dimensionality.
- *Sparsity.* Most features are zero for most samples, i.e. the customer-product matrix is sparse. This strongly affects the behavior of similarity measures and the computational complexity.
- *Discrete feature levels.* Often, features are neither nominal, nor continuous. However, most clustering algorithms are tailored for either continuous or nominal, but not for positive ordinal attribute values.
- *Feature normalization.* The range and scale of each feature implicitly determines this feature's importance for clustering. Hence, they have to be aligned with the objective. Since the number of features is very high, normalization becomes difficult. For example, high-volume products would be considered more important than low-volume products because their feature values have a higher variance. However, customers may be distinguished easier in the low-volume than in the high-volume product domain.

Further author information: (Send correspondence to Joydeep Ghosh)
Alexander Strehl: E-mail: strehl@ece.utexas.edu
Joydeep Ghosh: E-mail: ghosh@ece.utexas.edu

- *Strong non-Gaussian distribution of feature values.* The resulting skewedness may cause simple maximum-likelihood estimators (such as means) to perform worse than a trivial constant estimate (such as zero) in terms of mean absolute error.
- *Significant outliers.* Outliers such as a few, big corporate customers that appear in otherwise small retail customer data, may totally offset results. Filtering these outliers may not be easy, nor desirable since they could be very important (e.g., major revenue contributors).

Due to these issues, traditional clustering techniques work poorly on real-life market-basket data. We claim that OPOSSUM, a graph-partitioning approach using value-based features, is well suited for the market-basket clustering domain, based on a series of experiments.

1.2. Related Work

A large body of work on clustering exists. Some classic approaches include partitional methods such as k-means[2], hierarchical agglomerative clustering[3], unsupervised Bayes[4], and competitive neural networks[5]. Recently, the needs of large scale and high-dimensional data mining, as described in the previous section, have lead to a variety of new clustering algorithms. CURE (Cluster Using REprentatives)[6] represents clusters by a fixed number of well-scattered points that have been shrunk towards the center by a user-defined fraction. ROCK (RObust Clustering using linKs)[7] is an agglomerative hierarchical clustering technique based on links that are obtained from thresholded similarities. Karypis et al.[8] proposed Chameleon, a split and merge approach using graph partitioning. Unfortunately, the performance of many algorithms is only demonstrated on illustrative 2–dimensional examples and lack an objective base for comparison.

1.3. Notation

Let n be the number of sample points in the data and d the number of features of each sample point x_i with $i \in \{1, \ldots, n\}$. The input data can be represented by an n by d matrix Ξ with the i-th row representing the sample x_i^T. In retail data mining this representation is also called the customer-product-matrix. The clustering problem can be described as finding a n-dimensional vector κ of labels κ_i for each point d-dimensional point x_i. The number of desired clusters is k. The entries κ_i in the label vector κ range over all integer numbers from 1 to k. In general the labels are treated as nominals with no inherent order. In some cases, such as self-organizing feature maps (SOFMs) or top-down recursive graph-bisection, the labeling may contain extra ordering information. However, we do not make use of this since it is not a property common to all clustering techniques. We define a clustering technique as a mapping function Φ from the input data Ξ to the cluster labels κ, given the number of desired clusters k:

$$\kappa = \Phi(\Xi, k) \tag{1}$$

X_i denotes all points in the i-th cluster ($i \in \{1, \ldots, k\}$). The number of points in X_i is c_i. The next section describes the appropriate choice of features for clustering retail customers.

2. DOMAIN SPECIFIC, NON-NEGATIVE FEATURES

While most of the well-known techniques[3,4] have been for numerical features, certain recent approaches assume categorical data[7]. In general, continuously distributed features are advantageous since they capture noisy behavior better for a small number of samples. For example, in market-basket data analysis a feature represents the absence (0) or presence (1) of a particular product in the current basket. However, this treats a buyer of a single corn-flakes box the same as one who buys one hundred such boxes. In OPOSSUM, we extend the common Boolean notation to *non-negative real-valued features*. The feature $x_{i,j}$ represents the volume of product p_j in a given basket (or sample) x_i. In a first improvement to the initial Boolean product feature, we could use product quantity to measure feature volume. Instead, we propose the usage of monetary value (the product of price and quantity) to quantify feature volume. This yields an almost continuous distribution of features since the prices across various products cover the entire range of possible feature values fairly densely. More importantly, the price represents a normalization across all feature dimensions. This normalization is highly desirable because it better aligns relevance for clustering with management objectives.

An important step in pre-processing is feature selection. In our framework, very common and very uncommon products are removed before clustering. For example, products that are very rarely purchased and have a low

dollar value are removed, i.e. the corresponding attribute column is deleted. The decision to remove a product from consideration is made if a minimum average revenue contribution per customer is not exceeded. After feature selection, the customers whose market-basket is now empty have to be removed (typically less than 1%), since some similarity measures are not defined for zero-vectors.

3. SIMILARITY MEASURES

As the foundation for grouping decisions, an appropriate problem specific similarity measure has to be defined. A similarity measures $s \in [0,1]$ captures how related two data-points x_1 and x_2 are. It can be bijectively mapped to an distance measure $d \in [0,\infty]$ with $d = 1/s - 1$. We experimented with three similarity measures, which are described in this section.

3.1. Metric Distances

The Minkowski distance, also known as the L_p norm, is the standard metric for geometrical problems. Equation 2 suggests a definition of similarity based on Minkowski distance. For $p = 1$ ($p = 2$) we obtain the Manhattan (Euclidean) based similarity normalized to the $[0,1]$ interval.

$$s_{Minkowski}(x_1, x_2) = \frac{1}{1 + \|x_1 - x_2\|_p} = \frac{1}{1 + \left(\sum_{i=1}^{d} |x_{1,i} - x_{2,i}|^p\right)^{1/p}} \tag{2}$$

3.2. Cosine Measure

Similarity can also be defined by the angle or cosine of the angle between two vectors. This allows points with equal composition, but different scale to be treated identically. For example, in retail one might consider a customer buying 1 loaf of bread and 2 gallons of milk equivalent to a customer who buys 2 loafs of bread of 4 gallons of milk. It is widely used in text classification because two documents with equal word composition but different lengths can be considered identical. The cosine measure is given in equation 3 and captures a length invariant understanding of similarity.

$$s_{cosine}(x_1, x_2) = \frac{x_1^T x_2}{\|x_1\|_2 \cdot \|x_2\|_2} = \frac{x_1^T x_2}{\sqrt{x_1^T x_1 x_2^T x_2}} \tag{3}$$

3.3. Jaccard Similarity

The Jaccard coefficient[2] measures the ratio of the number of commonly active features of x_1 and x_2 to the number active in x_1 or x_2. Equation 4 gives a definition in vector notation of this measure which is often used in retail market-basket applications[7].

$$s_{Jaccard}(x_1, x_2) = \frac{x_1^T x_2}{x_1^T x_1 + x_2^T x_2 - x_1^T x_2} \tag{4}$$

For binary features, the Jaccard coefficient measures the ratio of the intersection of the product sets to the union of the product sets. When used with real-valued features, the (extended) Jaccard coefficient captures a length-dependent measure of similarity.

4. CLUSION: CLUSTER VISUALIZATION

Often, the most powerful tool for clustering is the human eye. Since it is limited to processing 2 dimensions (or 2.5 at most), the large number of high-dimensional data-points have to be converted to a perceptually more suitable format. This section illustrates how CLUSION, our CLUSter visualizatION toolkit, allows us to employ the human vision system to explore the relationships in the data, guide the clustering process, and verify the quality of the results.

Given a similarity measure, the clustering problem can be moved from the original feature space (n by d data matrix Ξ) to similarity space (n by n similarity matrix S). We visualize the symmetric similarity matrix S with entries $s_{i,j} = s(x_i, x_j)$ as a quadratic gray-level image where a black (white) pixel corresponds to minimal (maximal) similarity of 0 (1). The pixel at row i and column j corresponds to the similarity between the points x_i and x_j. Table 1 gives a toy data-set that we use to illustrate the clustering process using our approach. Figure 1 depicts

Quantity	Milk ($ 2)	Cereal ($ 3)	Bike ($ 200)	TV ($ 300)	total in $
Andrea (#1)	1	1	-	-	$ 5
Bill (#2)	1	100	-	-	$ 302
Chase (#3)	2	2	-	-	$ 10
Diana (#4)	1	1	1	-	$ 205
Eric (#5)	1	1	-	1	$ 305
Fritz (#6)	-	-	-	1	$ 300
Gina (#7)	2	2	2	-	$ 410
total in $	$ 16	$ 321	$ 600	$ 600	$ 1537

Table 1. Illustrative retail data-set consisting of seven points (customers) in four dimensions (products).

visualizations of the resulting similarities from quantity (top row) and value features (bottom row). For example if one only considers quantity, Diana and Eric are very similar (see also upper row of figure 1). However, if one looks at which products take their share of wallet, they are very distinct, since Diana bought a bike and Eric a TV. Consequently it is desirable to weight similarity by relevance which is successfully achieved by using monetary value. On weighting the features by the corresponding monetary value, Eric becomes more similar to Fritz than to Diana even though Eric and Fritz have less items in common (but spend most of their purchasing power on a TV).

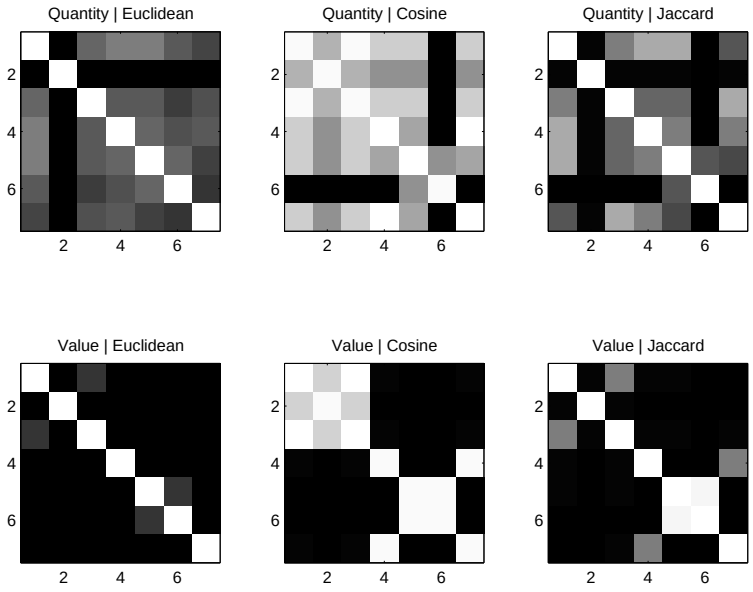


Figure 1. Similarity matrices using different features and measures for the illustrative example. The pixel brightness is proportional to the similarity between the two data-points given by the row and column indices.

Figure 1 visualizes the similarity matrix for two different feature types and three similarity measures. Using the quantity feature (top row) no good segmentations are possible. Most customers are related approximately equally. For the Euclidean case, Bill does not fit in and all others seem to form one cluster. In fact Fritz is almost as similar to Diana as to Eric despite Fritz and Eric having no items in common. In cosine of quantity space, customers 1 to 5 seem be forming a cluster. This fails to separate the high volume customer Bill, and the high value customers Diana and Eric. For quantity features and extended Jaccard similarity Bill can be separated but still not Diana and Eric.

The results with value-based features (bottom row of figure 1) are less cluttered and more intuitive. Euclidean similarity, however, is also inappropriate in this domain. For example, Diana and Gina are not considered similar even though Gina bought the same products, just twice as many. In cosine space, Bill is not clearly distinguished from the other milk and cereal buyers Andrea and Chase. The Jaccard measure does intuitively reflects the similarity pairs (Eric, Fritz), (Diana, Gina), as well as (Andrea, Chase).

Visualizing similarity space can help to quickly get a feel for the clusters in the data. Even for a large number of points, a sense for the intrinsic number of clusters k in a data-set can be gained. Many strong relations suggest few and large natural clusters, while few relations (a sparse matrix) indicate more and small clusters. OPOSSUM uses this insight to automatically pick the optimal number of clusters k.

We developed the CLUSION cluster visualization toolkit to verify cluster quality and fine-tune clusterings. A very expressive display of a given partitioning can be obtained by rearranging the columns and rows of the similarity matrix such that points with the same cluster label are contiguous and next to each other. The similarity *within* cluster i is thus represented by a quadratic region with side length c_i around the main diagonal of the matrix. The off-diagonal rectangular areas visualize the relationships *between* clusters. The brightness distribution in the rectangular areas yields insight towards the quality of the clustering and possible improvements. In order to make these regions apparent, thin horizontal and vertical lines are used to show the divisions into the rectangular regions. Examples for CLUSION are given in the following sections in figures 2 and 3.

5. OPOSSUM: OPTIMAL PARTITIONING OF SPARSE SIMILARITIES USING METIS

5.1. Desired Cluster Properties

In clustering, performance can be measured by an overall objective function, such as sum-of-squared-error or minimum variance. However, these standard criterions are less meaningful in the retail data mining applications since they are based on metric spaces.

An important issue with many clustering frameworks is the lack of constraints on the sizes of clusters. Often the distribution of sizes depends on initialization, is succeptible to high variability, and tends not to be uniform.

Suppose 100 customers are split into 2 groups of sizes 5 and 95, respectively. It appears that there is little to be gained by this clustering. However, if the small group consists of the big spending customers and the large one represents low dollar value customers, the result would be very useful. In other words, we want to constrain the properties of each cluster to assure that clusters are of importance to the user. OPOSSUM can deliver equal sized (balanced) clusters using either of the following two criteria:

- *Sample Balanced:* Each cluster should contain roughly the same number of samples, n/k. This allows retail marketers to obtain a customer segmentation with equally sized customer groups.
- *Value Balanced:* Each cluster should contain roughly the same amount of feature values. In this case a cluster represents a k-th fraction of the total feature value $\sum_{i,j} x_{i,j}$. If we use extended revenue per product (quantity $\times$ price) as features then each cluster represents a roughly equal contribution to total revenue.

We formulate the desired balancing properties by assigning each sample (customer) a weight and constraining the sum of weights in each cluster. For sample balanced clustering we assign each sample x_i the same weight $w_i = 1$. To obtain value balancing properties a sample x_i's weight is set to $w_i = \sum_j x_{i,j}$. The desired balancing properties have many application driven advantages. However, since natural clusters may not be equal sized, over-clustering (using a larger k) and merging may be helpful.

Figure 2 shows CLUSION's clustering visualizations and the balance ratio for various k on the illustrative retail data. The top row shows how sample balanced clusters are formed while preserving similarity relationships. Since clusters always have an integer number of elements, perfect balancing is not possible. Value balanced clustering results are shown in the bottom. At $k = 3$, for example, in value balanced clustering the high volume cereal buyer Bill is successfully singled out as a group by himself, rather than forming a cluster with Diana and Gina as was the case for sample balanced clustering. In the lower row, when comparing $k = 3$ to $k = 4$, Eric and Fritz are split into singletons instead of splitting the cluster with four samples into the two good pairs (Andrea, Chase) and (Eric, Gina), because the very low ($5) revenue contribution of (Andrea, Chase) gets penalized since it would greatly violate the balancing constraints.

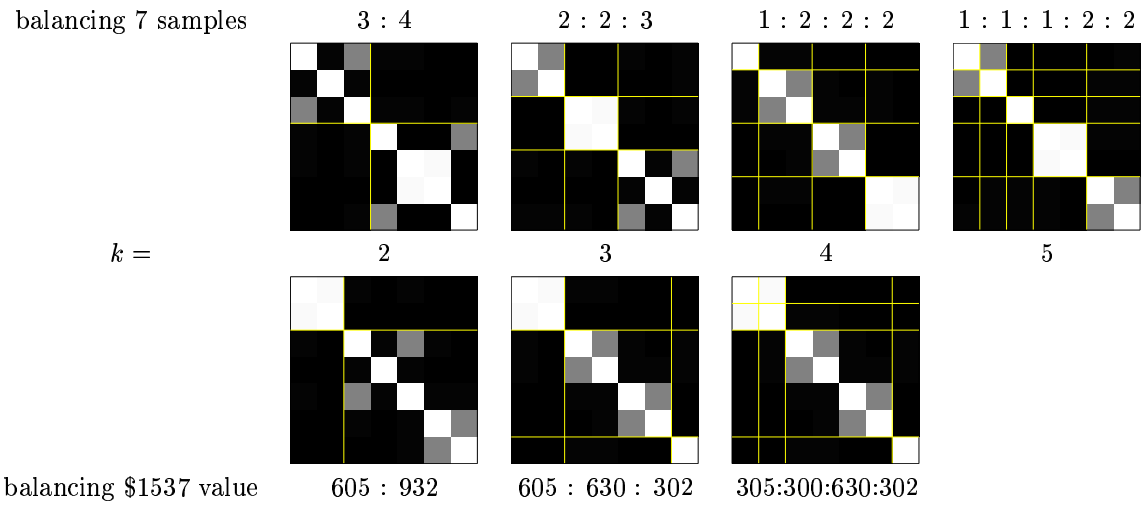


Figure 2. Clustering results for illustrative retail data and value-based Jaccard similarity. Rows and colums are permuted by cluster-labels using CLUSION. Maximum cluster quality is reached at $k = 4$ for sample balanced (top) and at $k = 3$ for value balanced (bottom) clustering.

5.2. Single-constraint Single-objective Weighted Graph Partitioning

We map the problem of clustering to partitioning a weighted graph with a minimum number of edge cuts while maintaining a balancing constraint. Graphs are a well-understood abstraction and a large body of work exists on partitioning them. The samples can be viewed as a set of vertices V. A pair of samples x_1 and x_2 (or vertices v_1 and v_2) is connected with an edge $e_{1,2}$ of positive weight $s(x_1, x_2)$. The cardinality of the set of edges $|E|$ equals the number of non-zero similarities between all data points. A set of edges, whose removal partitions a graph $G = (V, E)$ into k pair-wise disjoint sub-graphs $G_i = (V_i, E_i)$, is called an edge separator Υ. Our objective is to find such a separator that minimizes the sum of weights of cut edges as given by equation 5.

$$\min_{\Upsilon} \sum_{e_{i,j} \in \Upsilon} s(x_i, x_j) \quad , \quad \Upsilon = (E \setminus (E_1 \cup E_2 \cup \ldots \cup E_k)) \tag{5}$$

Without loss of generality, we can assume that the vertex weights are normalized to sum up to 1: $\sum_{i \in \{1,\ldots,n\}} w_i = 1$ While striving for the minimum cut objective, the constraint given by equation 6 has to be fulfilled:

$$k \cdot \max_j \sum_{\kappa_i = j} w_i \leq l \tag{6}$$

The left hand side quantifies the load balance of the partitioning κ. The load balance is dominated by the worst cluster. A value of 1 indicates perfect balance. Of course, in many cases the constraint can not be fulfilled exactly (e.g., sample balanced partitioning with n odd and k even). However they can be fulfilled within a certain narrow tolerance. We chose the maximum tolerated load imbalance $l \geq 1$ to be 1.05, or 5%, for our retail experiments.

Finding an optimal partitioning is an NP-hard problem[9]. However, there are very fast, heuristic algorithms for this widely studied problem[10,11,12]. The basic approach to dealing with graph partitioning or minimum-cut problems is to construct an initial partition of the vertices either randomly or according to some problem-specific strategy. Then the algorithm sweeps through the vertices, deciding whether the size of the cut would increase or decrease if we moved this vertex v over to another partition. The decision to move v can be made in time proportional to its degree by simply counting whether more of v's neighbors are on the same partition as v or not. Of course, the desirable side for v will change if many of its neighbors switch, so multiple passes are likely to be needed before the process converges to a local optimum.

After experimentation with several techniques, we decided to use the Metis multi-level multi-constraint graph partitioning package because it is very fast and scales well. A detailed description of the algorithms and heuristics used in Metis is beyond the scope of this paper and can be found in Karypis et al.[13]

5.3. Optimal Clustering

This subsection describes how we find a desirable clustering, with *high overall cluster quality* Γ and a *small number of clusters* k. First, let us define quality. Intra- and inter-cluster similarity capture the average similarity within one (equation 7) and between two (equation 8) clusters, respectively.

$$\text{intra}(\Xi, \kappa, i) = \frac{1}{(c_i - 1) \cdot c_i} \sum_{\kappa_a = \kappa_b = i, a > b} s(x_a, x_b) \tag{7}$$

Equation 7 does not define intra-cluster similarity for empty and singleton clusters. We extend the definition such that singletons ($c_i = 1$) have an intra-cluster similarity of 1. Inter-cluster similarity between the non-empty clusters i and j, with $i \neq j$ is given by equation 8.

$$\text{inter}(\Xi, \kappa, i, j) = \frac{1}{c_i \cdot c_j} \sum_{\kappa_a = i, \kappa_b = j} s(x_a, x_b) \tag{8}$$

Our objective is to maximize intra-cluster similarity and minimize inter-class similarity. Hence, we define the quality of a given clustering κ as the ratio of the total similarities within clusters to the total similarities amongst clusters. Total similarities are obtained by the summation of individual cluster similarities weighted by their corresponding cluster-size c_i. Intra-similarity is weighted by $c_i - 1$ since self-similarity (which is always 1) should not be taken into account.

$$\frac{\sum_{i=1}^{k} \frac{c_i - 1}{n - k} \cdot \text{intra}(\Xi, \kappa, i)}{\sum_{i=1}^{k} \sum_{j=i+1}^{k} \frac{c_i}{n} \cdot \text{inter}(\Xi, \kappa, i, j)} \tag{9}$$

This ratio ranges from 0 to ∞. For our visualization tool CLUSION, the ratio captures the factor by which the average on-diagonal region is brighter than the average off-diagonal region. Values greater than 1 indicate traditional clustering and values smaller than 1 result for inverse clustering (dissimilar samples are grouped together). Finally, for $1 < k < n$, we define our *quality* measure $\Gamma \in [0, 1]$ ($\Gamma < 0$ in case of pathological/inverse clustering) as follows:

$$\Gamma(\Xi, \kappa) = 1 - \frac{(n - k) \cdot \sum_{i=1}^{k} \sum_{j=i+1}^{k} c_i \cdot \text{inter}(\Xi, \kappa, i, j)}{n \cdot \sum_{i=1}^{k} (c_i - 1) \cdot \text{intra}(\Xi, \kappa, i)} \tag{10}$$

$\Gamma = 0$ indicates that samples within the same cluster are on average not more similar than samples from different clusters. On the contrary, $\Gamma = 1$ describes a clustering where every pair of samples from different clusters has the similarity of 0 and at least one sample pair from the same cluster has a non-zero similarity. Clearly, this perfect quality can only be achieved if the data-set and similarity measure co-operate. Of course, this is almost never the case and, hence, what quality level can be considered good is highly dependent on the data and on the similarity measure. On the other hand, clusterings on different data-sets can be compared easily. Also, our definition of quality does not take the "amount of balance" into account. Balancing is observed fairly strictly by the constraints in the graph-partitioning. Hence, it does not vary with k and can be neglected.

To achieve a high quality Γ as well as a low k, the target function $\Theta \in [0, 1]$ is the product of the quality Γ and a penalty term. Let $n \geq 4$ and $2 \leq k \leq \lfloor n/2 \rfloor$, then there exists at least one clustering with no singleton clusters. Equation 11 defines performance Θ such that performance is penalized linearly with k.

$$\Theta(k) = \left(1 - \frac{2k - 4}{n - 3}\right) \cdot \Gamma(k) \tag{11}$$

For large n, we search for the optimal k in the entire window from $2 \leq k \leq 100$. In many cases, however, a forward search starting at $k = 2$ and stopping at the first down-tick of performance while increasing k is sufficient.

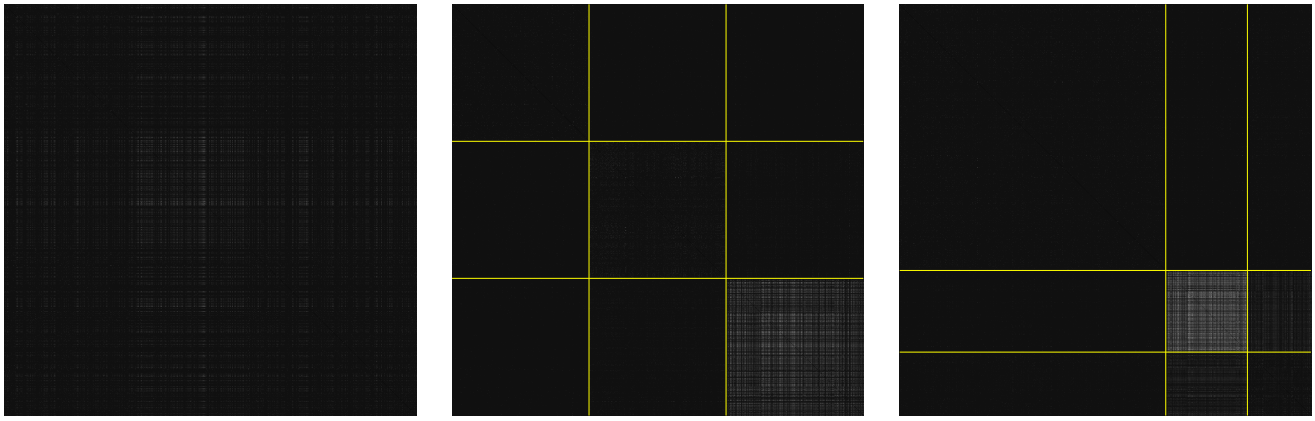
6. EXPERIMENTAL RESULTS ON RETAIL DATA

We experimented with real retail transactions from a drugstore. We randomly selected 2500 customers. The total number of transactions (cash register scans) for these customers is 33814 over a time interval of three months. We rolled up the product hierarchy once to obtain 1236 different products purchased. 473 of these products accounted for less than \$25 each in toto and were dropped. The remaining $d = 763$ features and $n = 2488$ customers (12 customers had empty baskets after removing the irrelevant products) were clustered using our OPOSSUM approach. Results for this example are obtained with a 550 MHz Pentium II PC with 512 MB RAM in under 10 seconds when similarity is precomputed. Figure 3 shows the similarity matrix (74.6% sparse) visualization before (a), after sample balanced (b), and after value balanced clustering (c). OPOSSUM and CLUSION clearly extract a highly related customer group accounting for approximately 20% of the customers and 30% of the revenue.

Figure 4(a) shows the behavior of the performance for increasing k. The optimal k for value balanced clustering Optimal performance is found at $k = 3$ for sample balanced clustering (upper graph, circles) and at $k = 14$ for value balanced clustering (lower graph, crosses). In figure 4(b) the data points of the three clusters 12, 13 and 14 is projected onto a two-dimensional plane. Similar to CViz[14], the plane is defined by the centroids of the three clusters. In this extremely low-dimensional projection, the three selected clusters can be reasonably separated using just a linear discriminant function. However, adding another cluster into the projection would not maintain the separability, as the positions of the other centroids suggest.

Clusters are described in terms of their highest revenue products and their most unusual revenue drivers. Revenue lift is the ratio of the average spending on a product in a particular cluster to the average spending in the entire data-set. For example, one cluster seems to contain parents with interests in home improvement. Their spendings are mostly on Christmas related gifts, especially for kids. On average, every customer in that group spends \$10.74 on christmas giftware. The most unusual purchases of this group include home related items like lawn seed and plumbing. While spending on lawn seed is only \$0.18 per customer, this is 15 times more than the average customer.

However, to protect the confidentiality of the data, we can not give any detailed profiles of the clusters. OPOSSUM identifies customer groups with equal buying behavior. Marketing can use the customer groups to design and deploy personalized promotional strategies to each group. The clusters are balanced by revenue value and hence provide insight into the contributions and profiles of each customer group.



2488 customers
\$148772.61 revenue
(a)

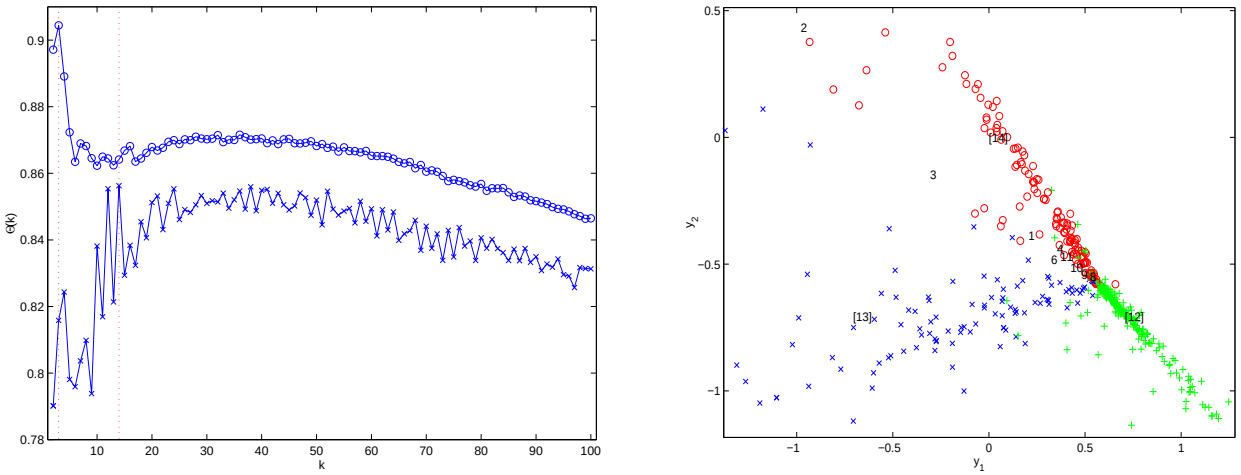
829 : 829 : 830
\$11046.62 : \$42660.14 : \$95065.85
(b)

1610 : 494 : 384
\$49282.98 : \$49907.81 : \$49581.82
(c)

Figure 3. Results of drugstore data clustering by OPOSSUM. Similarity visualizations using CLUSION (a) before, (b) after sample balanced, and (c) after value balanced clustering, with $k = 3$. The images are increased in brightness for better printing.

7. FUTURE WORK

OPOSSUM can be extended in a variety of directions. For example, meta-data such as product hierarchies and department information can be used to improve clustering performance and speed up processing. The selected features



(a) (b)

Figure 4. (a): Behavior of performance Γ for various k on the drugstore data. Circles and crosses correspond to sample and value balanced clustering, respectively. The optimal k is found at 3 (14) for sample (value) balanced clustering, and is marked with a dotted vertical line. (b): 2–dimensional projection of 2488 763–dimensional data-points on the plane defined by the three cluster centroids (shown as bracketed numbers). Cluster membership of points is indicated by their color and shape.

are of critical importance. More feature selection and feature transformation methods could be included[15,16]. We are also working on an on-line version of OPOSSUM that incrementally updates clusters as new data points become available. Moreover, methods of scaling using parallel approaches are being investigated. The computation of the $n \cdot (n-1)/2$ (worst case) similarity measures, discussed in section 3 can be distributed with a low communication overhead due to the properties of the dot-product ($x_1^T x_2 = x_{1,\{1,\ldots,i\}}^T x_{2,\{1,\ldots,i\}} + x_{1,\{i+1,\ldots,d\}}^T x_{2,\{i+1,\ldots,d\}}$).

8. CONCLUSIONS

In this paper, we present OPOSSUM, an algorithm to perform Optimal Partitioning Of Sparse Similarities Using Metis. Instead of binary features, we use extended revenue per product, to better model management objectives. We compared various similarity measures and found the extended Jaccard coefficient to be best suited for high-dimensional retail data mining. We use a constrained, top-down, weighted graph partitioning approach to cluster the data. OPOSSUM delivers clusters that are balanced in terms of either customers (samples) or revenue (value). This results in clusters that are most suitable for further analysis by marketing. Also, we suggest a performance measure that eliminates the only user defined parameter k to allow for a closed-loop deployment of the system. We demonstrated that OPOSSUM is well suited for clustering customers from high-dimensional market-baskets on a real drugstore data-set with over 30000 transactions. Also, to facilitate data exploration and validation of results we introduce CLUSION, a CLUSter visualizatION toolkit for high-dimensional clustering problems.

ACKNOWLEDGMENTS

We want to express our gratitude to Mark Davis, Net Perceptions (`http://www.netperceptions.com`) and KD1 (`http://www.kd1.com`) for providing the retail data sets. We also thank all LANS (`http://lans.ece.utexas.edu`) members, especially Shaleish Kumar and Gunjan Gupta, for their valuable comments and suggestions. This research was supported in part by the NSF under Grant ECS-9000353, and by a gift from KD1.

REFERENCES

1. M.-S. Chen, J. Han, and P. S. Yu, "Data mining: An overview from a database perspective," *IEEE Transactions on Knowledge and Data Engineering* **8**, pp. 866–883, December 1996.
2. A. K. Jain and R. C. Dubes, *Algorithms for Clustering Data*, Prentice Hall, New Jersey, 1988.

3. J. A. Hartigan, *Clustering Algorithms*, Wiley, 1975.
4. R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley, New York, 1973.
5. K. Mehrotra, C. Mohan, and S. Ranka, *Elements of Artificial Neural Networks*, MIT Press, Cambridge, Massachusetts, 1997.
6. S. Guha, R. Rastogi, and K. Shim, "Cure: An efficient clustering algorithm for large databases," in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 73–84, (New York), 1998.
7. S. Guha, R. Rastogi, and K. Shim, "Rock: a robust clustering algorithm for categorical attributes," in *Proceedings of the 15th International Conference on Data Engineering*, 1999.
8. G. Karypis, E.-H. Han, and V. Kumar, "Chameleon: Hierarchical clustering using dynamic modeling," *IEEE Computer* **32**, pp. 68–75, August 1999.
9. T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*, MIT Press, 1990.
10. R. J. Lipton and R. E. Tarjan, "A separator theorem for planar graphs," *SIAM Journal on Applied Mathematics* **36**, pp. 177–189, 1979.
11. G. L. Miller, S.-H. Teng, and S. A. Vavasis, "A unified geometric approach to graph separators," in *Proceedings of 31st Annual Symposium on Foundations of Computer Science*, pp. 538–547, 1991.
12. A. Pothen, H. Simon, and K.-P. Liou, "Partitioning sparse matrices with eigenvectors of graphs," 1990.
13. G. Karypis and V. Kumar, "Multilevel k-way partitioning scheme for irregular graphs," *Journal of Parallel and Distributed Computing* **48**(1), pp. 96–129, 1998.
14. I. Dhillon, D. Modha, and W. Spangler, "Visualizing class structure of multidimensional data," in *Proceedings of the 30th Symposium on the Interface: Computing Science and Statistics, Minneapolis, MN, May 13–16 1998*, S. Weisberg, ed., 1998.
15. Y. Yang and J. O. Pedersen, "A comparative study on feature selection in text categorization," in *Proceedings of the 14th International Conference on Machine Learning*, pp. 412–420, Morgan Kaufmann, 1997.
16. D. D. Lewis, "Feature selection and feature extraction for text categorization," in *Proceedings of Speech and Natural Language Workshop*, pp. 212–217, Morgan Kaufmann, (San Mateo, California), February 1992.