Using Latent Semantic Indexing for Data Mining

A Thesis

Presented for the

Master of Science Degree

The University of Tennessee, Knoxville

Jingqian Jiang

December 1997

Acknowledgments

I thank my advisor, Dr. Michael Berry, for his encouragement, support, and guidance throughout this project. I also thank Dr. June Donato and Dr. Nancy Grady for their support and the opportunity to work on this project and Dr. Bradley Vander Zanden for serving on my thesis committee. I thank Dr. George Ostrouchov for his advice on the statistical aspect of the project.

This research has been supported by the Visual and Information Sciences Group, Oak Ridge National Laboratory.

Abstract

Data Mining is the application of algorithms for extracting valuable information from large databases in order to make important business decisions. This study explores a new technique for data mining – Latent Semantic Indexing (LSI). LSI is an efficient information retrieval method for textual documents. By determining the singular value decomposition (SVD) of a large sparse term-by-document matrix, LSI constructs an approximate vector space model which represents important associative relationships between terms and documents that are not evident in individual documents. This thesis explores the applicability of the LSI model to numerical databases, especially consumer product data. By properly chosing attributes of data records as terms or documents, a term-by-document *incidence* matrix is built and then a distribution-based indexing scheme is employed to construct a *correlated distribution* matrix. Hence a similar LSI vector space model can be generated to detect useful or hidden patterns in the databases. The extracted information can then be validated using statistical hypotheses testing or resampling. LSI is an automatic yet intelligent indexing method, its application to numerical data introduces a promising way to discover knowledge in important commercial application areas such as retail and consumer banking.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation and Goals

Large amounts of data have been collected in daily operations of organizations due to inexpensive storage and high computing power, but many companies have been unable to extract useful information from the data and utilize the information to benefit their business. Data mining is the application of algorithms for extracting valid and useful information from large databases in order to make critical business decisions. The fact that data is being accumulated at a faster rate than it can be analyzed creates a significant demand for efficient data mining systems [DF94, FPSSU96]. Techniques used for data mining include decision trees and rule induction [Qui92], nonlinear regression and classification [Fri89, BFOC84], genetic algorithms [MCM86], and neural networks [CT94, GBD92]. This study explores a new technique, *Latent Semantic Indexing* (LSI), for data mining.

LSI is an efficient information retrieval technique which has been commonly used for textual documents [BDO95, DDF$^+$90]. Traditional lexical-matching methods try to match words of queries with words of documents, which may fail to retrieve related documents or may return unrelated documents to users. This kind of failure to retrieve relevant documents or the retrieval of irrelevant documents is called the *word-matching* problem. LSI overcomes the *word-matching* problem by using statistically derived conceptual indices instead of individual words [BDO95]. Using the singular value decomposition (SVD) [GL89] of a large sparse term-by-document matrix, LSI constructs a conceptual vector space in which each term or document is represented as a vector in the space. The clustering of the term or document vectors reveals the underlying semantic structure of association between terms and documents in the data.

This thesis explores the applicability of the LSI vector-space model to numerical databases. By properly chosing attributes of data records as terms or documents, a term-by-document frequency matrix is built, and then a distribution-based indexing scheme is employed to construct a correlated distribution matrix which reflects relationships between attributes of data records. Hence, the LSI-like vector space model is generated so that the clustering of attributes in the space can be analyzed for the detection of useful or hidden patterns. The extracted information can then be validated using statistical hypotheses testing or resampling.

Applications for data mining extract information from the data to make important business decisions, predict business trends, and develop new products. A common application is to analyze customer purchases to discover patterns among existing customer preferences and then use those patterns for forecasting sales and optimizing marketing strategies. Other uses of data mining methods include the analysis and selection of stocks, fraud detection and prevention, and scientific applications in Astronomy and Molecular Biology [FPSSU96]. Data mining systems have been effectively implemented in a number of sectors: chemical/pharmaceutical industry, retail and consumer banking, and financial transactions [DF94]. This thesis focuses upon applications of the LSI model to consumer product data.

1.2 Overview

The following chapters outline the development and application of LSI to numerical databases. Chapter 2 is an overview of basic concepts of data mining and reviews the background of the LSI vector space model. Chapter 3 illustrates how terms and documents are chosen to construct the LSI model for numerical databases and demonstrates the application of LSI to a consumer product database. A summary of this thesis and a discussion of future work are provided in Chapter 4.

Chapter 2

Overview of Data Mining

2.1 Introduction

Advances in data collection and data storage make it possible for organizations to generate large volumes of data during their daily operations. For example, electronic data-gathering devices and remote-sensing devices generate an enormous amount of data. Due to inexpensive storage, corporations no longer need to be careful about the accumulation of data [DF94]. This explosive collection of data creates significant need to analyze the data for underlying or hidden yet crucial information.

Data mining is the application of algorithms for extracting valid, useful, previously unknown and ultimately comprehensible information from large databases

[FPSSU96]. The extracted information can be used to form a prediction or classification model, identify relations between database records, or provide a summary of database information. The retrieved information is not necessarily a faithful copy of information stored in the database, rather it is information that can be inferred from the database [FPSSU96].

There are a variety of computationally intensive data mining techniques based on decision trees, rule induction and neural networks. Only now with the availability of large amounts of computing power is it possible to mine large databases with intelligent and automatic tools. For this reason alone, data mining has become one of the most rapidly growing fields of research.

2.2 Primary Tasks of Data Mining

In practice, two primary goals of data mining are prediction and description. Prediction uses known fields in the database to predict unknown values of other fields of interest. Description determines human-interpretable patterns which describe the data. The following primary data mining tasks are used to achieve the goals of prediction and description [FPSSU96]:

- *Classification* maps (groups) a data item into one of several predefined classes. For example, a bank may want to group loan applicants into two classes: offer loan or do not offer loan.

- *Regression* is learning a function which maps a data item to a real-valued prediction variable. Sample regression applications might include predicting consumer demand for a new product based on advertising expenditure, or estimating the probability that a patient will die given the results of a set of diagnostic tests.

- *Clustering* identifies a finite set of categories to describe the data. One example of clustering applications is discovering homogeneous sub-populations for consumers in marketing databases.

- *Summarization* involves methods for finding a compact description for a subset of data. Used primarily for exploratory analysis, summarization might involve a simple tabulation of means and standard deviations for all fields.

2.3 Inductive Learning

One interpretation of data mining is that data mining is the process of learning useful knowledge from the databases. From an inductive learning or machine learning point of view, where a model is created to infer knowledge from databases, there are two basic ways of performing data mining: *supervised learning* and *unsupervised learning* [HS95].

In supervised learning, an external *teacher* defines classes and provides the data mining system with examples of each class. The system has to discover common properties in the examples for each class. Since patterns are generalized from known cases, this method is commonly known as *learning from examples*. Common properties form a class description which can be used to predict the class of previously unseen objects.

In unsupervised learning, there is no external teacher and no pre-classified objects, the data mining system has to discover the classes itself based on common properties of objects. So, patterns are found through observation. This technique is also known as *learning from observation and discovery*. The result of an unsupervised learning is a set of class descriptions.

Based on the type of data and the underlying information to be derived, there are different techniques for data mining as described in Chapter 1. For example, the decision tree technique for data mining is discussed next in Section 2.4, and the LSI model is introduced later in Section 2.5.

2.4 Decision Trees and Rule Induction

Decision trees and rule Induction are suitable for supervised learning. Assume that the dataset of interest is a collection of data records in which each record has the same structure, consisting of a number of attribute/value pairs. The attribute to be modeled or predicted is usually chosen as the class attribute. There are two sets of data required for a decision tree: training data and testing data. Using the training data set, the decision tree method creates a tree structure which serves as a classifier for the database [Qui92]. The result of the classification can then be verified on the testing data.

2.4.1 Tree Structure

A decision tree is a tree structure in which

- a *leaf* indicates a class,
- an *interior node* specifies a test on a single non-class attribute, and
- an *arc* corresponds to a possible value of the tested attribute on the node.

A data record can then be classified by following a path from the root of the tree to the appreciate leaf.

2.4.2 Algorithm

The algorithm to build a decision tree is based on a *divide and conquer* approach. If the training set contains one or more cases which all belong to a single class, then the tree is a leaf identifying this class. If the training set contains a mixture of classes, then the training set is partitioned into subsets of cases based on some splitting criterion. Applying the same algorithm to the subsets recursively will eventually terminate the process and construct a decision tree [Qui92].

There are many splitting criteria such as the *information gain* criterion (see Appendix E) which use information entropy to determine on which attribute to split during the partitioning process. One can choose the attribute with greatest information gain among the attributes not yet considered in the path from the root to split on at each node [Qui92].

2.4.3 Production Rules

Since a decision tree may become quite complex, with long and very uneven tree paths from the root to leaves, procedures to *prune* the tree are needed to simplify the tree structure. Production rules, which are *if-then* like statements, can be generated from the decision tree by following a path down the tree. Tree pruning can simplify the production rules by replacing an entire subtree with a leaf node. One replacement criterion is that if a decision rule establishes that the expected error rate in the subtree is greater than in the single leaf, the subtree is replaced by a leaf [Qui92].

2.4.4 A Simple Example

A sample decision tree based on the training dataset provided in Appendix A is given in Figure 2.1. Each data record has four fields: income level, education, household size and the class attribute indicates whether a coupon is used for purchasing a cereal product.

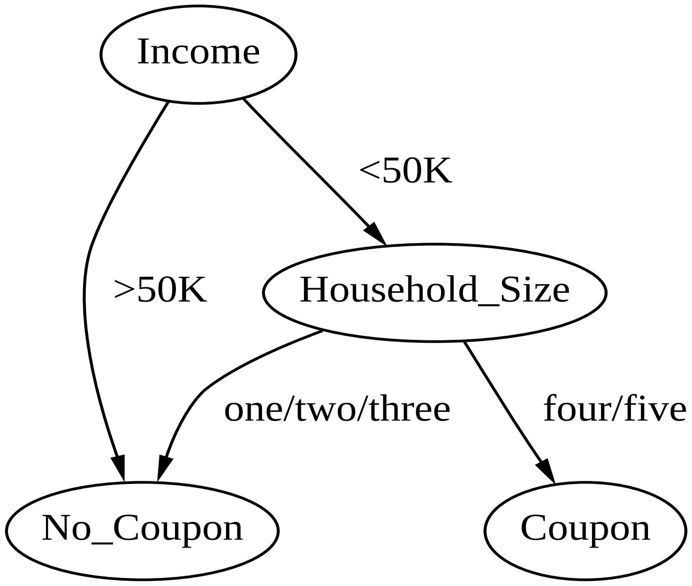


Figure 2.1: A Sample Decision Tree

2.5 Latent Semantic Indexing

The word-matching problem mentioned in Section 1.1 is due to fact that multiple words may have the same meaning (*synonymy*) and many words have more than one meaning (*polysemy*). For example, a text collection contains documents on house ownership and web home pages with some documents using the word *house* only, some documents using the word *home* only, and some documents using both words. For a query on *home ownership*, traditional lexical-matching methods fail to retrieve documents using the word *house* only, which are obviously related to the query. For the same query on *home ownership*, lexical-matching methods will also retrieve irrelevant documents about web home pages. LSI overcomes the word-matching problem by retrieving information based on the meanings of queried words and documents. Relevant documents can be retrieved even if they do not share common words with users' queries [BDO95].

By choosing a set of frequently used words in a database as terms to index documents, a term-by-document matrix can be constructed which defines the relationship between terms and documents. Since LSI assumes that this relationship is partially obscured by variability in word choice, a singular value decomposition (SVD) of the sparse term-by-document matrix is used to approximate term-document associations by using only the k-largest singular values and corresponding singular vectors. The k-dimensional conceptual vector space is constructed from the singular vectors (scaled by singular values), so that each term or document can be represented as a point in the space. This reduced vector space captures the implicit high-order structure in the association of terms and documents and eliminates much of the *noise* due to the variability of word usage, such as *synonymy* and *polysemy*. Finally, a user's query is projected into this vector space and documents related to the query are returned to the user even though they may not share terms with the user's query. LSI is a completely automatic yet intelligent method which improves the retrieval of information from databases.

2.5.1 Term-By-Document Matrix

For text document retrieval, a set of words are chosen as the terms, which are used as indices of the document database. Each entry of the term-by-document matrix represents the occurrences of each word in a document, i.e,

$$A = (a_{ij}),$$

where a_{ij} is the frequency in which term i occurs in document j. Since every word does not normally appear in each document, the matrix A is usually sparse [BDO95].

Local and Global Weightings

In practice, a local and global weighting scheme can be applied to matrix A in order to improve retrieval performance [Dum91] . Specifically, the raw frequency

a_{ij} can be transformed to

$$a_{ij} = L(i,j) * G(i),$$

where $L(i,j)$ is the local weighting for term i in document j, and $G(i)$ is the global weighting for term i. Some common local weightings include:

- Term Frequency: $L(i,j) = tf_{ij}$,
- Log: $L(i,j) = \log_2(tf_{ij} + 1)$,

where tf_{ij} = the frequency of term i in document j. Possible global weightings include:

- Normal: $G(i) = 1/\sqrt{(\sum_j tf_{ij}^2)}$,
- GfIdf: $G(i) = gf_i/df_i$,
- Idf: $G(i) = \log_2(n_d/df_i) + 1$,
- Entropy: $G(i) = 1 - \sum_j \frac{p_{ij}\log_2(p_{ij})}{\log_2(n_d)}, \qquad p_{ij} = tf_{ij}/gf_i$,

where

gf_i is the global frequency of term i in the document collection,

df_i is number of documents containing term i, and

n_d is number of documents in the collection.

While global weightings indicate the overall importance of terms in the document collection, local weightings are used to stress the importance of terms within a particular document [Dum91].

2.5.2 Semantic Vector Space Model

Once a term-by-document matrix is constructed, LSI requires the singular value decomposition (SVD) of this matrix to construct a semantic vector space which can be used to represent conceptual term-document associations.

Singular Value Decomposition

Given an $m \times n$ matrix A, where $m \geq n$ and $rank(A) = r$, the singular value decomposition of A [GL89] is defined as

$$A = U \Sigma V^T, \tag{2.1}$$

where $U^T U = V^T V = I_n$ and $\Sigma = diag(\sigma_1, ..., \sigma_n)$, $\sigma_i > 0$ for $1 \leq i \leq r$, $\sigma_j = 0$ for $j \geq r+1$. The σ_i's are nonnegative square roots of the eigenvalues of $A^T A$ and AA^T, which are defined as the singular values of A. The associated eigenvectors of AA^T are columns of U which are referred to as the *left* singular vectors; the associated eigenvectors of $A^T A$ are columns of V, which are called the *right* singular vectors [GL89].

Equation (2.1) reveals a breakdown of the original relationships into linearly independent factors. These factors represent extracted common meaning components of many different words and documents [DDF+90].

Reduced Vector Space

By using the first k factors or k-largest singular values along with their corresponding left and right singular vectors, respectively, an approximation (A_k) to the original matrix A is given by

$$A_k = U_k \Sigma_k V_k^T, \tag{2.2}$$

where U_k and V_k are comprised of the first k columns of the matrices U and V (see Equation 2.1), respectively, and $\Sigma_k = diag(\sigma_1, ..., \sigma_k)$. The A_k is the best rank-k approximation to A in a least squares sense, i.e.,

$$\min_{rank(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}, \tag{2.3}$$

where $\|X\|_2 = \max_{v \in R^n} \{\|Xv\|_2 \,/\, \|v\|_2\}$ [GR71].

By reducing the dimensionality of the original matrix A, the matrix A_k captures most of the important underlying structure in the association of terms and documents while ignoring *noise* due to word choice. It is important that the derived k-dimensional factor space does not reconstruct the original matrix completely in order to avoid the word-matching problem (e.g., *polysemy* and *synonymy*) associated with lexical-matching information retrieval methods [BDO95].

2.5.3 Interpretation of Vector Space Model

In the reduced vector space model defined by A_k, each term or document is represented by a vector of weights indicating its strength of association with some underlying concepts. The position of term (document) vectors reflects the correlations in their usage across documents (terms). Similar terms and documents are typically positioned near each other in the space [Dum91].

Geometric Representation

Using a geometric interpretation of the SVD, the locations of the terms and objects in k-space are given by row vectors of U_k and V_k. Since $A_k A_k^T$ reflects the similarities between terms as represented by

$$A_k A_k^T = U_k \Sigma_k^2 U_k^T,$$

the rows of $U_k\Sigma_k$ can be used as term vectors. Similarly, $A_k^TA_k$ represents relationships between documents as specified by

$$A_k^TA_k = V_k\Sigma_k^2V_k^T,$$

so that rows of $V_k\Sigma_k$ can be used as document vectors [DDF+90].

The cosine or Euclidean distance between term or document vectors in the space corresponds to their estimated similarity. Retrieval proceeds by locating a query vector in the space, and all documents and terms are compared to the query and ranked by their similarity to the query. The terms and documents whose vectors yield cosine or Euclidean distance values above some threshold are returned to the user. In the reduced space, the similarity of documents is determined by the major underlying patterns of term usage, so documents that share no words with a user's query may still be returned to the user [BDO95].

Query Matching and Relevance Feedback

A query composed of several terms may be represented as a *pseudo-document*, i.e., as a weighted sum of its component term vectors in k-space. For example, the vector $\hat{q}$ defined by

$$\hat{q} = q^TU_k\Sigma_k^{-1},$$

is a representation of the query q whose non-zero elements contain the frequency counts of the terms that appear in the query and have been properly weighted if needed. To improve retrieval performance, *relevance feedback* [SL90] can be used. This process involves altering the initial query based on user feedback or judgement of which documents are more relevant to the initial request.

2.5.4 Advantages of LSI

Compared to other concept-based approaches for information retrieval, LSI has several advantages [DDF$^+$90, Dum91]. First, LSI uses a high-dimensional space to better represent a wide range of semantic relations. Second, both terms and documents are explicitly represented in the same space. Therefore, new terms can be placed at the centroid of the documents in which they appear; similarly, new documents can be placed at the centroid of their constituent terms. Third, LSI retrieves documents from query terms directly, without interpretation of the underlying dimensions as is the case with many factor-analytic applications. It only assumes that these factors represent one or more semantic relationships in the document collection. Finally, LSI is able to represent and manipulate large datasets [DDF$^+$90, Let96].

Information retrieval performance is typically measured by *precision* and *recall*. Precision is the ratio of the number of relevant documents to the total number of documents retrieved by the system. Recall is the proportion of all relevant documents in the collection that are retrieved for a query. LSI has been demonstrated to improve retrieval precisions over different recall levels by an average of 20% over five standard document collections [Dum91]. In addition, LSI is an automatic method and very easy to use. With relevance feedback using the first relevant document from an initial query, LSI can improve retrieval precision for different recalls by 33% on average [Dum91].

2.6 Related Fields of Data Mining

Data mining refers to a class of methods that can be used in Knowledge Discovery in Databases (KDD) [FPSSU96]. KDD is the overall process of finding and interpreting patterns from data, typically involving the repeated application of specific data mining methods and interpretation of the patterns generated by these algorithms.

Data mining is certainly related to Statistics, particularly in exploratory data analysis. For example, data mining systems often use particular statistical procedures to preprocess data. But they are also different. Statistics techniques tend to assume some form for a model and then find appropriate values for the model's parameters, whereas data mining tends to generate patterns from data.

The fields of machine learning and pattern recognition also develop theories and algorithms for extracting patterns and models from data. Such algorithms tend to provide efficient data mining methods. Another related area is data warehousing, which refers to the recently popular MIS (Management Information Systems) trend for collecting and cleaning transactional data for on-line retrieval [FPSSU96].

Chapter 3

LSI for Data Mining

3.1 Introduction

Although many data mining systems are derived from machine learning and neural networks, information retrieval techniques based on conceptual searching algorithms are also evolving. The conceptual vector space model used by LSI tries to cluster similar objects in the vector space so that objects related to a given query (but perhaps not containing the exact same terminology) can be retrieved. The success of LSI for textual documents inspires its application to numerical databases. As with textual objects, the attributes of numerical data records can be represented in a vector space, and the clustering of these attributes in the space can be analyzed to reveal patterns from databases.

In order to apply LSI, a relationship matrix must be constructed from the given database. A semantic vector space model can then be generated by computing the truncated singular value decomposition [GL89] of the constructed matrix. This LSI-based approach is an unsupervised learning technique and the process is fully automatic and intelligent. The following sections illustrate how LSI can be applied to consumer product data.

3.2 Distribution-by-Behavior Matrix

For textual documents, it is natural to select words used in the documents as terms or referents, therefore a term-by-document matrix, which represents usage patterns between terms and documents, can be constructed as described in Section 2.5.1. For numerical databases, it is not clear how *terms* should be selected from documents in order to build a matrix which reflects important relationships embedded in the dataset(s).

3.2.1 Incidence Matrix

Consumer product databases may contain consumer demographic dataset(s) and corresponding purchase behavior dataset(s). Here a dataset refers to a collection of data records, where each data record consists of a number of attribute variables. Rather than using all the attributes of each data record, several important attributes which characterize the record can be selected. For the LSI application to consumer product databases, the documents can be selected as instances of purchase behavior and the terms are instances of demographic variables. Once terms and documents are determined, a term-by-document *incidence* matrix $A = (a_{ij})$ can be defined, where a_{ij} is the co-occurrence frequency of a demographic variable category represented by term i and a characteristic of purchase behavior represented by document j. By defining the incidence matrix A this way, the purchase behavior characteristics are indexed by demographic variables, so that association patterns between consumers and their purchasing behaviors can be encoded.

The incidence matrix A can be constructed in an efficient way. Each row description of the incidence matrix A is the combination of a demographic variable and its categorical value. Only the (variable, value) pairs which exist in the dataset are selected as rows, since there is no entry in the matrix for those (variable, value) pairs not existing in the data. One way to assign the rows is to make two passes over a demographic dataset, the first pass could determine which rows actually exist in the data, and the second pass would build the matrix. But for large databases, making two passes is computationally expensive, hence the incidence matrix should be created by assigning rows during one pass through the data. A variable initialized to 1 can be used to keep track of the next available row number.

Each time a new (variable, value) pair is encountered, it is assigned to the next available row number. The row number for any particular demographic variable associated with the database can then be easily found. A sample incidence matrix is provided in Table 3.1.

Table 3.1: A Sample Incidence Matrix

Demographic Instance	Purchase Behavior		
	p1	p2	p3
Age = 30	10	7	5
Income = 40K	1	3	4
Age = 50	2	6	11
Income =50K	3	10	15

3.2.2 Correlated Distribution Matrix

LSI assumes that the variability of word (or term) choice partially obscures the underlying semantic structure of the documents. The incidence matrix defined in Section 3.2.1 for categorical databases, however, does not necessarily satisfy this assumption since the instances of demographic variables do not typically vary like word choice in textual documents. This study proposes to use distributions of demographic variables as *terms* so that the variability of term-to-document association can be measured by correlation coefficients between distributions. The documents in this case, are chosen (similar to construction of the incidence matrix) to define a distribution-by-behavior matrix, or simply a *correlated distribution* matrix (CDM). Each entry d_{ij} of a CDM $D = (d_{ij})$ is either 0 or 1 indicating whether a specific characteristic of purchasing behavior (represented by a column of D) has a particular demographic distribution.

3.2.3 Building the Correlated Distribution Matrix

A correlated distribution matrix can be constructed from an incidence matrix using correlation coefficients between distributions of demographic variables across the variables' categories. Consider two instances of purchasing behavior represented by columns j, j' of an incidence matrix $A = (a_{ij})$, and let I be the set of all possible row indices in the matrix A, which correspond to a demographic variable V. Define n to be the number of elements in I. The correlation coefficient between distributions of this particular demographic variable V for purchasing behaviors represented by columns j, j' of A is

$$r(j, j') = \sum_{i \in I} \frac{(a_{ij} - \overline{a_j})(a_{ij'} - \overline{a_{j'}})}{\sqrt{(\sum_{i \in I}(a_{ij} - \overline{a_j})^2)(\sum_{i \in I}(a_{ij'} - \overline{a_{j'}})^2)}}, \tag{3.1}$$

where $\overline{a_j} = \sum_{i \in I} a_{ij}/n$, and $\overline{a_{j'}} = \sum_{i \in I} a_{ij'}/n$.

A correlation coefficient $r \in [-1, 1]$ is a measure of the linear relationship between two random variables. In the LSI application to consumer product data, if $r \geq 0.7$, a strong linear correlation between two variables is assumed. The distributions of demographic variables that are unique (i.e., not strongly correlated with other distributions) are chosen to index purchasing behaviors, a CDM can then be constructed to reflect the relationships between consumers and purchasing behaviors.

The algorithm to build a correlated distribution matrix from an incidence matrix is described as follows. Let the instances of purchasing behavior represented by columns of the incidence matrix be denoted by $(B_1, B_2, ..., B_n)$, and let the demographic variables be represented by $(V_1, V_2, ..., V_k)$ for $k, n > 0$. Then, for each demographic variable V_i and each instance of purchasing behavior B_j, there exists a distribution of the demographic variable V_i, denoted as (V_i, B_j) for $i = 1, ..., k$, and $j = 1, ..., n$. Different behaviors have different demographic distributions. Let the CDM be $D = (d_{ij})$, whose columns are represented by the purchasing behaviors $(B_1, B_2, ..., B_n)$ and whose rows are generated during the construction. Let h be an integer initialized to 1, which represents the next available row number for the CDM D. Consider the purchasing behavior B_1 first, for each demographic variable V_i, $i = 1, ..., k$, the distribution (V_i, B_1) is assigned the next row number h, denoted as $R(v_i, b_1) = h$, and h is then incremented by 1. Thus, $d_{i1} = 1$ for $i = 1, ..., k$. Now, consider each purchasing behavior B_j for $j = 2, ..., n$, with each demographic variable V_i, $i = 1, ..., k$. The correlation coefficient $r(j, j')$ between the distribution (V_i, B_j) and previously existing distributions $(V_i, B_{j'})$ for $j = 1, ..., n$ and $j' = 1, ..., j$ can be computed using Equation (3.1). If the coefficient $r(j, j') \geq 0.7$ and the $R(V_i, B_{j'})$ has already been defined, then $d_{lj'} = 1$, where $l = R(V_i, B_{j'})$. Otherwise the distribution (V_i, B_j) is assigned to the current row number h, $d_{hj} = 1$, and h is then incremented by 1. A pseudo-code for the above algorithm to build the correlated distribution matrix is provided below:

```
Begin
        h = 1;
        for (i =1,...,k; j =1,...,n) {
            R(V_i, B_j) = 0;
            d_ij = 0;
        }
        for (i = 1,...,k) {
            R(V_i, B_1) = h;
            d_i1 = 1;
            h = h + 1;
        }
        for (j = 2,...,n) {
            for (i = 1,..., k) {
                for (j' = 1,...,j) {
                    Compute r(j, j') according to Equation (3.1);
                    if (r(j, j') ≥ 0.7 and R(V_i, B_j') ≠ 0) {
                        l = R(V_i, B_j');
                        d_lj = 1;
                        break;
                    } else {
                        R(V_i, B_j) = h;
                        d_hj = 1;
                        h = h + 1;
                    }
                }
            }
        }
End
```

The correlated distribution matrix is built after one pass through all possible purchasing behaviors. The CDM is a sparse matrix, since for each instance of purchasing behavior and a demographic variable, there exists only one distribution from the data. A sample correlated distribution matrix is shown in Table 3.2, in which each column has only five 1's corresponding to the distributions of five demographic variables (HHsize, Income, MHage, Fhemp, MHemp, see Appendix B).

Table 3.2: A Sample Correlated Distribution Matrix

	Purchasing Behavior									
Distribution	p1	p2	p3	p4	p5	p6	p7	p8	p9	p10
HHsize:p1	1	0	1	0	1	0	1	0	0	1
Income:p1	1	1	0	1	0	1	0	1	0	1
MHage:p1	1	0	0	1	0	0	1	0	0	0
FHemp:p1	1	1	1	0	0	1	0	0	1	0
MHemp:p1	1	0	1	1	0	0	1	0	0	1
HHsize:p2	0	1	0	0	0	1	0	0	1	0
MHage:p2	0	1	0	0	0	1	0	0	0	0
MHemp:p2	0	1	0	0	1	1	0	1	1	0
Income:p3	0	0	1	0	1	0	1	0	1	0
MHage:p3	0	0	1	0	1	0	0	0	1	0
FHemp:p4	0	0	0	1	1	0	0	1	0	1
HHsize:p4	0	0	0	1	0	0	0	0	0	0
FHemp:p7	0	0	0	0	0	0	1	0	0	0
HHsize:p8	0	0	0	0	0	0	0	1	0	0
MHage:p8	0	0	0	0	0	0	0	1	0	1

3.3 Semantic Vector Space Model

Once the correlated distribution matrix $D = (d_{ij})$ is constructed, a single-vector Lanczos algorithm can be used to compute the singular value decomposition (SVD) of the correlated distribution matrix [BDO+93]. The LSI vector space model D_k can then be constructed using the k-largest singular values and corresponding singular vectors of the matrix D (see Section 2.5.2). Each distribution or instance of purchasing behavior can then be represented as a vector in the space. The similarity between or within distributions and possible purchasing behaviors can be measured by Euclidean distance or the cosine of distribution or purchasing behavior vectors.

3.3.1 The Query Engine

After the semantic vector space is constructed, a user's query can be easily projected into the k-dimensional vector space. Let $(D_1, D_2, ..., D_m)$ be the distribution vectors (row vectors of $U_k\Sigma_k$) in the semantic subspace, where m is the total number of distributions used in the correlated distribution matrix D, and let the purchasing behavior vectors (row vectors of $V_k\Sigma_k$) be $(C_1, C_2, ..., C_n)$, where n is the total number of possible purchasing behaviors in the matrix D. A user's query consists of a number of distributions, which can be represented as a *pseudo-purchasing-behavior* vector. For example, if a query contains distributions

$(D_{i_1}, ..., D_{i_l})$ for $i_1, ..., i_l \in \{1, ..., m\}$, then the pseudo-purchasing-behavior vector, $\hat{q} = q^T U_k \Sigma_k^{-1}$, is at the centroid of queried distribution vectors $(D_{i_1}, ..., D_{i_l})$, where q is a vector of length m with non-zeros corresponding to the distributions $(D_{i_1}, ..., D_{i_l})$. If relevance feedback [SL90] is used, the projected query vector in the space is represented by

$$\hat{q} = q^T U_k \Sigma_k^{-1} + \sum_{p=1}^{h} C_{j_p}, \tag{3.2}$$

where $1 \leq h \leq n$ and $C_{j_1}, ..., C_{j_h}$ are selected purchasing behavior vectors.

The query vector $\hat{q}$ in Equation (3.2) can be used as a *pseudo* purchasing behavior vector for making comparisons between or within distribution vectors and purchasing behavior vectors. The cosines between the query vector $\hat{q}$ and purchasing behavior or distribution vectors can be computed using the dot products $(\hat{q}, C_j)$ or $(\hat{q}, D_i)$ for $j = 1, ..., n$, or $i = 1, ..., m$. By sorting the cosines between the query vector $\hat{q}$ and purchasing behavior vectors in descending order, the closest purchasing behavior vectors can be displayed and analyzed to find common properties among all possible purchasing behaviors. In addition, the ordered list of distribution vectors in descending cosine order can be used to explore relationships between distributions and purchasing behaviors.

A command-line interface can be implemented for this query matching process, where a user can specify the dimensionality of the semantic vector space (the number of factors) as well as a list of distributions which comprise the query.

3.3.2 Validation

The patterns generated from the LSI model can be validated using statistical concepts and methods. One way to evaluate a relationship between two variables is statistical hypothesis testing, such as chi-square independence testing [HC78]. If a pattern shows that two variables are dependent, chi-square testing can be used to evaluate this result with some degree of certainty.

For categorical data, a *contingency* dataset can be used to express the relationship that exists between two variables [Jam91]. Let V_1 and V_2 be two categorical variables so that V_1 has h forms denoted by $\{A_1, A_2, ..., A_h\}$, and V_2 has k forms denoted by $\{B_1, B_2, ...B_k\}$. A *contingency* dataset is a rectangular table, where rows correspond to the forms of V_1, and the columns correspond to forms of V_2. Each table entry is the number of observations, denoted by n_{ij}, that possess the forms A_i and B_j simultaneously.

Two variables are independent from each other if the following relation is satisfied:

$$\frac{n_{ij}}{n} = \frac{n_i}{n} \cdot \frac{n_j}{n}, \qquad \forall i \in I = \{1, 2, ..., h\} \text{ and } \forall j \in J = \{1, ..., k\}$$

where

$$n_{ij} = \text{crossed frequency of the two forms } i \text{ and } j,$$

$$n_i = \sum_{j \in J} n_{ij} = \text{frequency of form } i,$$

$$n_j = \sum_{i \in I} n_{ij} = \text{frequency of form } j, \text{ and}$$

$$n = \sum_{i \in I, j \in J} n_{ij}.$$

Since the theoretical crossed frequency, assuming independence of i and j, is $n'_{ij} = n_i n_j / n^2$, the deviations between the observed and theoretical values $(n_{ij} - n'_{ij})$ characterize the disagreement between the observations and the hypothesis of independence [Jam91]. All of the deviations can be computed by

$$\chi^2 = \sum_{i \in I, j \in J} \frac{(n_{ij} - n'_{ij})^2}{n'_{ij}}. \tag{3.3}$$

Assuming that n'_{ij} is not too small, χ^2 is distributed according to Pearson's law with $(k-1)(h-1)$ degrees of freedom [HC78]. The chi-square testing proceeds by computing χ^2 and $\nu = (k-1)(h-1)$, then reading from the chi-square distribution table the value corresponding to the ν and the probability level p. If χ^2 is greater than the table look-up value, the hypothesis of independence is rejected with p certainty.

Another way to validate a result is to assess its variability on datasets using resampling. Several samples are generated from the original dataset, then analysis can be applied to each sampled data subset and results from each sampling are compared with each other to evaluate the variability of results. A more complicated way to validate a result is cross validation, in which a dataset is divided into several equal-sized blocks and analysis is applied on all the datasets with one block omited, then results from each such dataset are compared to each other to assess the variability of conclusions [EG83]. Both resampling and cross-validation approaches are computationally expensive. The chi-square testing and resampling methods are used in the LSI application to consumer product data.

3.4 LSI Application to Nielsen Consumer Data

The LSI model as applied to the electronic consumer data obtained from the A. C. Nielsen company is now presented. The ultimate goal of this study is to demonstrate the applicability of the LSI model to find the relationships between consumers and purchasing behaviors, which can be used for target marketing.

3.4.1 Nielsen Datasets

Two kinds of categorical data are available for this study: one is purchase data for orange juice, cereal, yogurt consumer items and the other is consumer demographic data. Each data record in a dataset is comprised of a sequence of digits.

The purchase datasets for orange juice, cereal, yogurt items contain 459,979 purchase records respectively, and the demographic dataset has $28,863$ data records, where each record represents a single consumer or household. The demographic variables and corresponding numerical encodings are listed in Appendix B, and a sample purchase data record showing selected record fields is provided in Appendix C. Since the purchase datasets are quite large (459,979 data records), randomly sampled subsets are used as sources to build correlated distribution matrices for the LSI model.

3.4.2 Distributions and Purchasing Behaviors

The instances of a 5-tuple defined by

(*Brand Name, Deal, Size, Type, Flavor*)

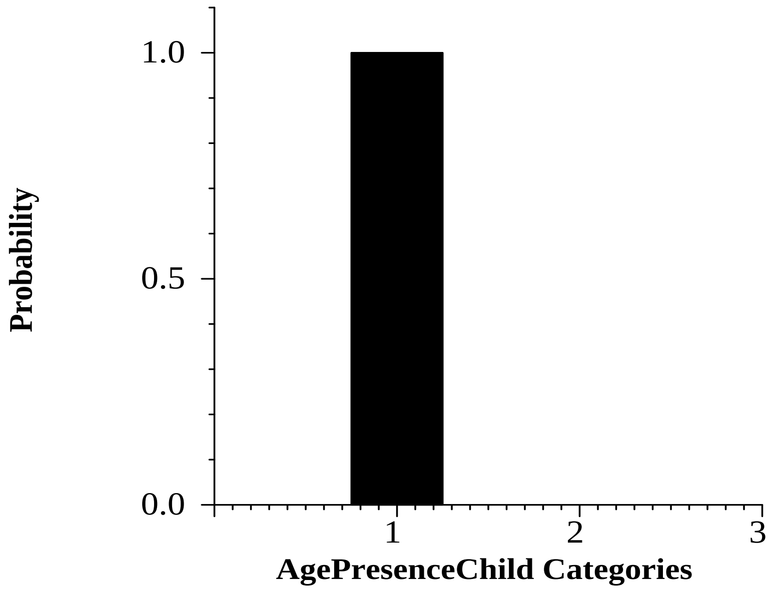
are selected to represent purchase behaviors, where *brand name* is the brand name of a consumer product, *Deal* indicates the type of purchase such as using a coupon or a store sale, *Size* is the size of a product in ounces, *Type* indicates either a single or multiple purchase is made, and *Flavor* indicates the flavor of products (which is only applicable to the yogurt products). To better interpret the clustering of purchasing behaviors, an artificial coding for brand name and deal attribute categorical values is used (see Appendix D). A sample instance of purchasing behavior is shown in Table 3.3. Here, *OJ_TR* is the orange juice brand *Tropicana*, the deal *MC* refers to the use of manufacturer coupons or double manufacturer coupons, the size of the product is 64 ounces, the type of the product is single, and there is no specified flavor for this product.

Table 3.3: A Sample Instance of Purchasing Behavior

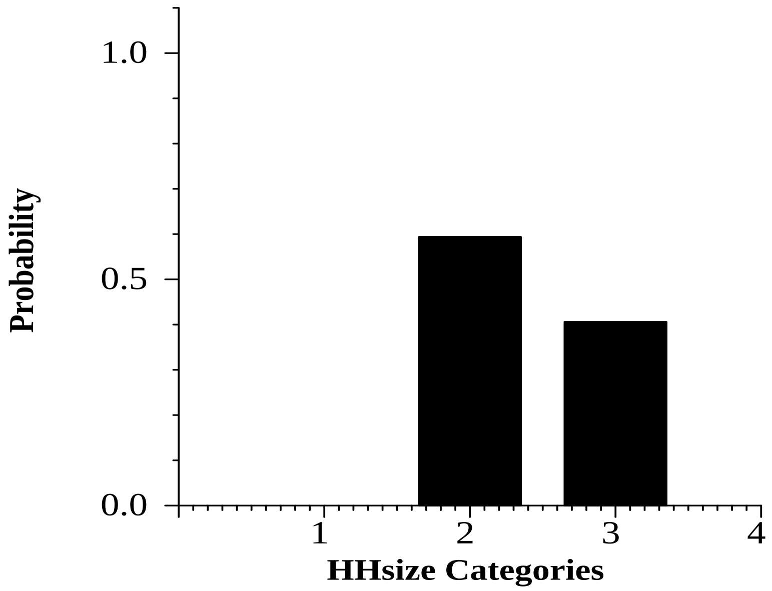
Brand Name	Deal	Size	Type	Flavor
OJ_TR	MC	64	Single	N/A

The distributions of demographic variables for purchasing behaviors define the rows for the correlated distribution matrix. Two-dimensional plottings are used to visualize distributions, where x-axis represents the categories for a demographic variable and y-axis represents the probability of a categorical value occurs among some consumer population. For example, the distribution of variable household size *HHsize* is shown in Figure 3.1 (see Section 3.4.3). Since each purchasing behavior corresponds to a column of the correlated distribution matrix, a distribution name can be denoted as *V:H* where *V* is a demographic variable and *H* is a column of the CDM which represents some purchasing behavior.

Once distributions and purchasing behaviors are determined, a correlated distribution matrix can be constructed using the algorithms presented in Section 3.2.3. A semantic vector space model is then built for interactive query matching.



(a) AgePresenceChild Query Distribution



(b) HHsize Query Distribution

Figure 3.1: Two Query Distributions for the Orange Juice Dataset

Euclidean Distance versus Cosine

Either Euclidean distances or cosines can be employed to measure the similarity of purchasing behaviors. The cosine, of course, computes the *angle* between a query and purchasing behaviors or distributions, which deemphasizes the lengths of vectors while Euclidean distance also takes the lengths of vectors into account. In some cases, the cosine value is a more reliable measure of semantic similarity of objects than the Euclidean distance between objects in the semantic subspace [FBY92]. The cosine calculation is used throughout the LSI application to the Nielsen consumer datasets.

3.4.3 Orange Juice Dataset

A random sample of $65,000$ data records is extracted from the orange juice purchase dataset, a 154×65 incidence matrix is then constructed from the sample data using the *log-entropy* weighting scheme. A 276×65 correlated distribution matrix is then created using the 154×65 incidence matrix. Query distributions such as those in Figure 3.1 can then be matched in a 4-dimensional semantic space. The distribution in Figure 3.1(a) is a distribution of the variable *AgePresenceChild* (which represents childrens' age), category 1 of the variable *AgePresenceChild* indicates a household with children under 6 years old. The distribution of household size (*HHsize*) is in Figure 3.1(b), where categorical values 2 and 3 of the variable

HHsize indicates a household with 2 and 3 family members, respectively. The top 10 returned purchasing behaviors to the query distributions (in Figure 3.1) in descending *closeness* order are provided in Table 3.4.

Most purchasing behaviors in Table 3.4 indicate a TR (Trade-Only) behavior which refers to the purchase of a sale item or using store coupons. Combining this purchase behavior with the query distributions from Figure 3.1, one may conclude that *two-member or three-member households with children under six years old are very likely to buy orange juice on sale or use store coupons.*

Table 3.4: Returned Purchasing Behaviors for Orange Juice Dataset (factor = 4)

Purchase Behavior	Cosine	Brand Name	Deal	Size	Type	Flavor
4	0.837	OJ_TR	MC	64	Single	N/A
22	0.758	OJ_CB	TR	32	Single	N/A
39	0.725	OJ_MM	TR	16	Single	N/A
9	0.609	OJ_TR	TR	32	Single	N/A
20	0.586	OJ_CB	TR	128	Single	N/A
57	0.582	OJ_FG	TC	64	Single	N/A
49	0.534	OJ_FN	TR	64	Single	N/A
41	0.486	OJ_MM	TC	96	Single	N/A
53	0.430	OJ_FG	ND	96	Single	N/A
55	0.430	OJ_FG	TR	64	Single	N/A

Hypothesis testing for the independence of variable (*HHsize)* and the returned purchasing behaviors in Table 3.4 can be applied to validate the above conclusion about the relationship between household size and purchasing behaviors. Based on the original weighted 154×65 incidence matrix, a contingency table whose rows are instance values of *HHsize* and columns are returned purchasing behaviors in Table 3.4, is generated in Table 3.5. The chi-square testing is then carried out with the degree of freedom $(7-1)(6-1) = 30$. If 85% probability level is used, the chi-square table look-up value is 38.57. Since the observed chi-square value $\chi^2 = 39.35$ is greater than the expected 38.57, the above conclusion about the relationship of household size and orange juice purchasing behaviors can be made at 85% certainty.

Table 3.5: Contingency Table for the Orange Juice Dataset

HHsize	Purchase Behavior							
	4	22	39	9	20	57	41	Total
2	16.146	3.871	0	3.871	10.741	0	4.310	38.939
3	13.641	1.673	1.673	2.651	11.015	0	0	30.653
4	12.515	2.651	4.695	0	10.144	1.672	0	31.677
5	10.392	1.653	0	0	7.146	0	0	19.191
6	7.379	0	0	0	6.300	0	0	13.679
7	2.510	0	0	0	3.156	0	0	5.657
Total	62.574	9.848	6.368	6.522	48.502	1.672	4.310	139.796

With the purchasing behaviors in Table 3.4 as columns and the top 10 returned distributions to the query in Figure 3.1 as rows, part of the correlated distribution matrix is shown in Table 3.6. Consider the row of query distribution *HHsize:8* in Table 3.6. The purchasing behaviors presented in Table 3.4 are clustered together even though they do not all share the queried distribution *HHsize:8*, which demonstrates the advantage of using LSI to find the underlying patterns.

Dimensionality

Different choices for the dimensionality of the reduced vector space model may produce different clustering patterns. Too few dimensions could over-generalized patterns or lose important information. Sufficient dimensionality is needed to capture the underlying structure in a correlated distribution matrix. Because the CDM may not truthfully represent the relationship between distributions and purchasing behaviors, it is also important not to reconstruct the original CDM completely [BDO95].

Table 3.6: Part of Correlated Distribution Matrix for the Orange Juice Dataset

Distribution	Purchase Behavior									
	4	22	39	9	49	20	57	41	53	42
HHsize:5	0	0	0	0	0	0	0	0	0	0
HHsize:8	0	0	1	1	0	0	0	1	0	0
HHsize:24	0	0	0	0	0	0	0	0	0	0
AgePresenceChild:24	0	0	0	0	0	0	0	0	0	0
KitchenApplicances:26	0	0	1	0	0	0	1	0	0	0
MHage:32	0	0	0	0	1	0	0	0	0	1
MHoccupation:37	0	0	0	0	0	0	0	1	0	0
HHsize:37	0	0	0	0	0	0	0	0	0	0
NumDogs:49	0	0	0	0	0	0	0	0	0	0
MHage:53	0	0	0	0	0	0	0	0	0	0

Reconsider the query distributions in Figure 3.1 using a 10-dimensional vector space as opposed to a 4-dimensional space (see Section 3.4.3), the purchasing behaviors retrieved from the LSI model are listed in descending *closeness* order in Table 3.7.

Table 3.7: Returned Purchasing Behaviors for Orange Juice Dataset (factor = 10)

Purchase Behavior	Cosine	Brand Name	Deal	Size	Type	Flavor
22	0.744	OJ_CB	TR	32	Single	N/A
4	0.588	OJ_TR	MC	64	Single	N/A
49	0.555	OJ_FN	TR	64	Single	N/A
20	0.485	OJ_CB	TR	128	Single	N/A
55	0.431	OJ_FG	TR	64	Single	N/A
57	0.424	OJ_FG	TC	64	Single	N/A
29	0.362	OJ_MM	ND	32	Single	N/A
41	0.340	OJ_MM	TC	96	Single	N/A
42	0.302	OJ_FN	ND	64	Single	N/A
53	0.299	OJ_FG	ND	96	Single	N/A

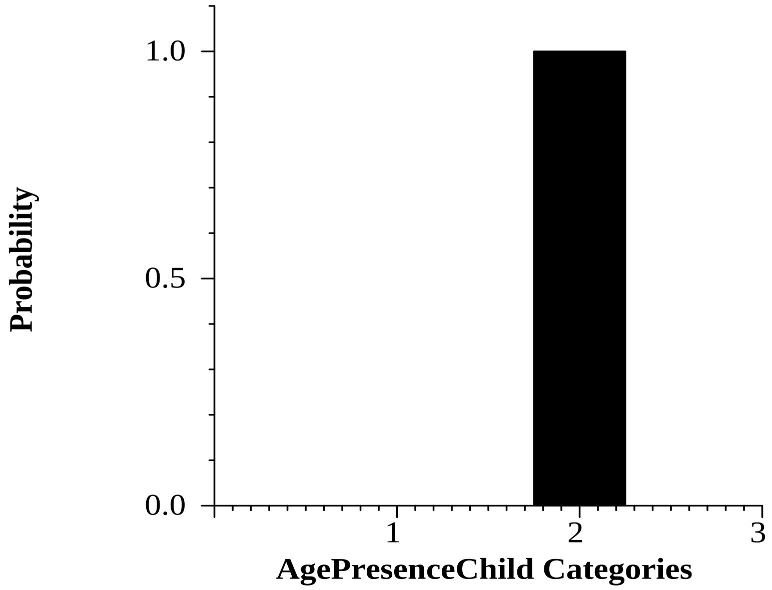
The purchasing behaviors shown in Table 3.7 reveal the similar clustering pattern from Table 3.4 except that the purchasing behaviors in Table 3.7 are not as close to the query as those shown in Table 3.4 in the cosine sense. One explanation for this result is that the added dimensions sometimes account for the randomness and noise associated with the variability of distributions across different purchasing behaviors.

Resampling can be used to assess variability of the returned purchasing behaviors shown in Table 3.7. Hence, another randomly sampled orange juice dataset with 65,000 records is generated, from which a 254×66 correlated distribution matrix is constructed.

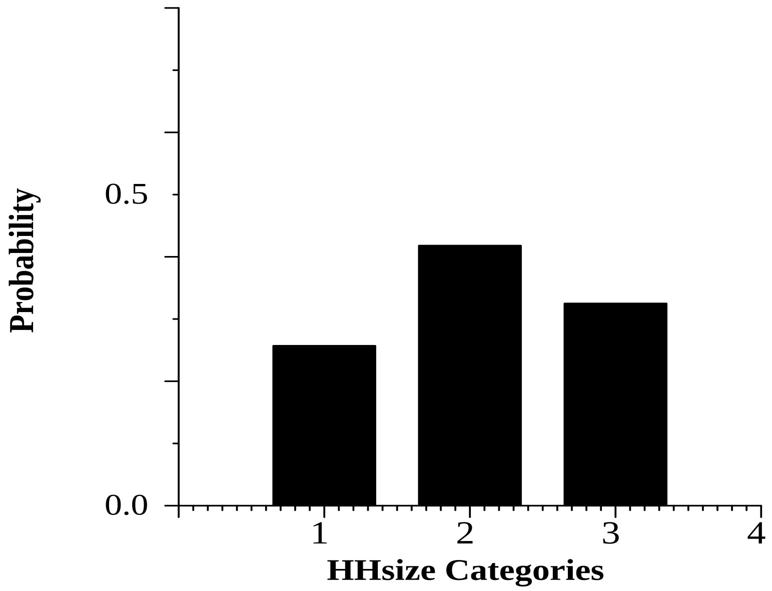
The query distributions in Figure 3.2, which are similar to the query distributions in Figure 3.1, represent two-member or three-member families with children. Projecting this query distributions into a 10-dimensional vector space, the LSI vector space model returns the purchasing behaviors shown in Table 3.8. These purchasing behaviors demonstrate a similar pattern, namely that mid-size families with children are likely to buy orange juice on store sale or using store coupons.

Table 3.8: Returned Purchasing Behaviors for Resampling

Purchase Behavior	Cosine	Brand Name	Deal	Size	Type	Flavor
38	0.479	OJ_MM	TR	128	Single	N/A
24	0.454	OJ_CB	TR	32	Single	N/A
41	0.448	OJ_MM	TR	16	Single	N/A
13	0.431	OJ_TR	TC	32	Single	N/A
52	0.428	OJ_FN	TR	96	Single	N/A
44	0.384	OJ_MM	TC	16	Single	N/A
48	0.329	OJ_FN	MC	64	Single	N/A
51	0.314	OJ_FN	TR	64	Single	N/A
66	0.310	OJ_DD	TC	96	Single	N/A
1	0.307	OJ_TR	ND	96	Single	N/A



(a) AgePresenceChild Query Distribution



(b) HHsize Query Distribution

Figure 3.2: Two Query Distributions for Resampling

3.4.4 Cereal Dataset

A random sample containing 15,000 data records is extracted from the cereal purchase data, from which a 430×166 correlated distribution matrix is built and a 2-dimensional semantic vector space is then constructed. The distribution of *AgePresenceChild* in Figure 3.3 is used as a query, which represents households with children of different ages. The categorical values for the variable *AgePresenceChild* is listed in Table 3.9.

Table 3.9: AgePresenceChild Variable Coding

Category	Value
1	under 6 only
2	6/12 only
3	13/17 only
4	under 6 & 6/12
5	under 6 & 13/17
6	6/12 & 13/17
7	under 6 & 6/12 & 13/17
9	no children under 18

The query distribution in Figure 3.3 is then searched in the 2-dimensional vector space, and the LSI query engine returns the top 10 closest distributions and purchasing behaviors shown in Tables 3.10 and 3.11. The distribution names in Table 3.10 are in the form of *V:H* as described in Section 3.4.2. For example, the distribution *HHsize:13* is shown in Figure 3.4, where the categorical values 1

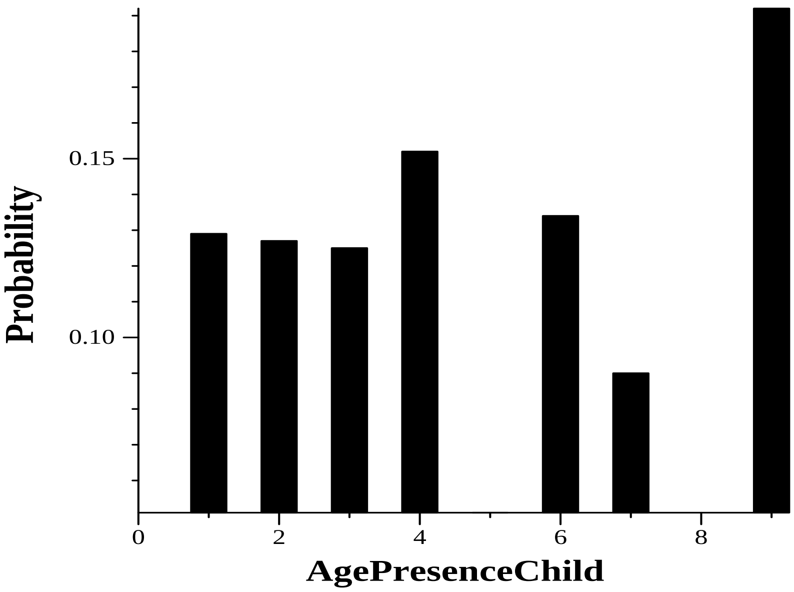


Figure 3.3: Query Distribution of AgePresenceChild for the Cereal Dataset

Table 3.10: Returned Distributions for the Cereal Dataset

Distribution Name
AgePresenceChild:0
NumCats:0
PetOwnership:10
HHsize:13
NumCats:89
NumDogs:4
AgePresenceChild:112
Race:9
Race:34
FHage:85

Table 3.11: Returned Purchasing Behaviors for the Cereal Dataset

Purchase Behavior	Cosine	Brand Name	Deal	Size	Type	Flavor
39	0.998	CE_KCF	TR	20_25	Single	N/A
95	0.997	CE_PGN	MC	20_25	Single	N/A
139	0.993	CE_GMG	TR	18_19	Single	N/A
10	0.978	CE_CB	TR	9_12	Single	N/A
56	0.977	CE_KFF	TR	15	Single	N/A
80	0.976	CE_KRB	TR	20_25	Single	N/A
78	0.958	CE_KRB	MC	20_25	Single	N/A
129	0.946	CE_PRB	TC	20_25	Single	N/A
124	0.940	CE_PRB	MC	15	Single	N/A
25	0.939	CE_GMC	TR	9_12	Single	N/A

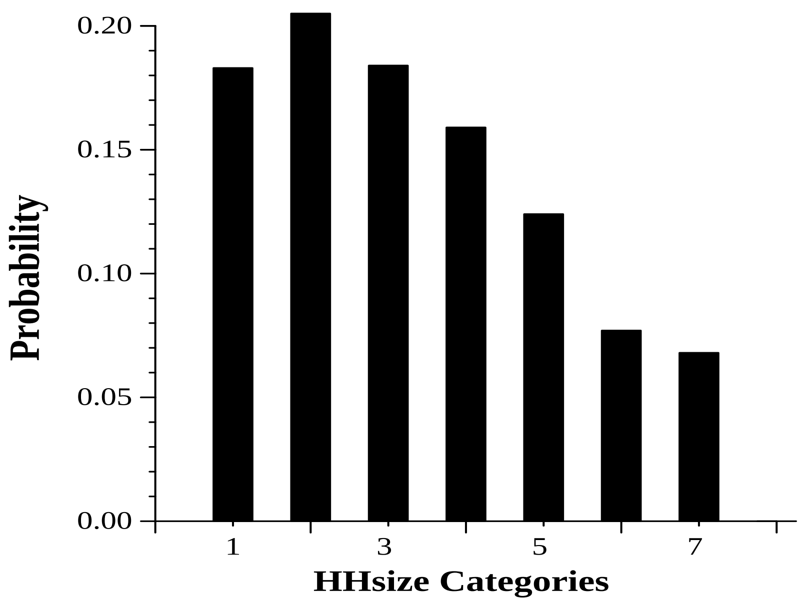


Figure 3.4: Distribution of HHsize for the Cereal Dataset

to 7 represent the number of family members in a household. The distribution *NumCats:0* is shown in Figure 3.5, which represents the number of cats a household owns. The detailed coding for all demographic variables are listed in Appendix B.

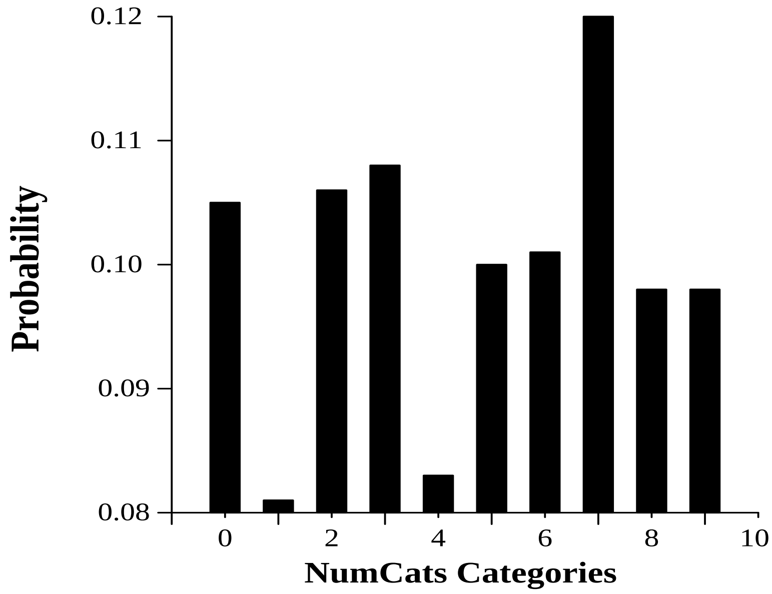


Figure 3.5: Distribution of NumCats for the Cereal Dataset

If we consider the distributions shown in Figures 3.4 and 3.5 in addition to the query distribution in Figure 3.3, the conclusion that *mid-size families with children and pets are very likely to buy large boxes of cereal on sale* can be made.

Chi-square testing for the independence of purchase behaviors in Table 3.11 and the variable *AgePresenceChild* is used to validate the above conclusion. The contingency table whose rows are instance values of *AgePresenceChild* and whose columns are returned purchasing behaviors shown in Table 3.11, is generated from the original weighted incidence matrix and is shown in Table 3.12. If 10% probability level is used, the chi-square value is about 42.5 with the degree of freedom 54. Since the observed chi-square value $\chi^2 = 42.704$ is greater than the expected 42.5, the above conclusion about the relationship between *AgePresenceChild* and cereal purchasing behaviors can only be made at 10% certainty. The rank-2 approximation of the original 430×166 correlated distribution matrix doesn't model enough association to reveal underlying relationships between distributions and purchasing behaviors. The above pattern concerning cereal purchasing may be over-generalized.

Table 3.12: Contingency Table for the Cereal Dataset

	Purchase Behavior										
Child	39	95	139	10	56	80	78	129	124	25	Total
1	1.83	3.65	0	0	0	4.24	4.72	2.89	1.83	1.83	21.1
3	2.96	4.83	0	1.87	3.73	5.24	3.73	2.96	4.34	2.96	32.6
4	2.94	3.7	0	0	2.94	4.3	2.94	2.94	2.94	3.7	26.4
5	0	1.65	0	0	0	0	0	0	0	0	1.65
6	2.95	3.72	0	0	4.32	5.22	4.32	2.95	2.95	1.86	28.3
7	1.68	0	0	0	0	0	0	0	0	2.66	4.34
Total	14.2	20.5	0	3.74	15.3	24.6	21.6	13.6	12.1	17.3	143

3.4.5 Orange Juice, Cereal and Yogurt Datasets

The previous examples consider orange juice and cereal purchase data separately. Additional experiments involving all three types of purchase data (orange juice, cereal, yogurt) are needed in order to study relationships between different product types.

Three randomly sampled datasets from orange juice, cereal and yogurt purchase data are generated, respectively, and combined together, from which a 949×306 correlated distribution matrix without weighting is constructed. From this CDM, a 5-dimensional semantic vector space is created and patterns are searched through query matching.

The query distribution in Figure 3.6 shows the probabilities of variable household composition (*HHcomp*) categories, which are listed in Table 3.13.

Table 3.13: Household Composition Coding

Category	Value
1	married
2	female head living with others related
3	male head living with others related
5	female living alone
6	female living with non-related
7	male living alone
8	male living with non-related

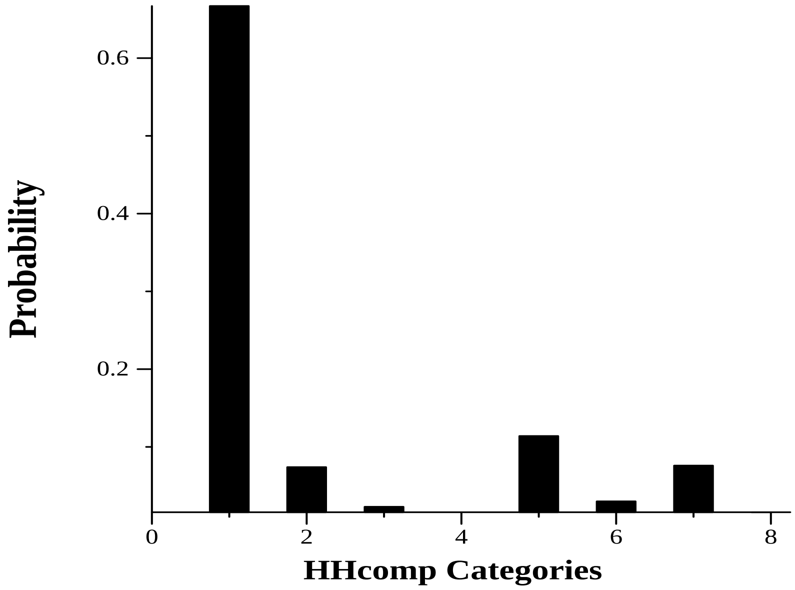


Figure 3.6: Query Distribution of HHcomp for the Mixed Datasets

The query distribution indicates that most households among some consumer population are composed of married couples. The corresponding query vector is projected into the 5-dimensional semantic space and the top 10 closest distributions and purchasing behaviors to this particular query are shown in Tables 3.14 and 3.15.

The distribution names in Table 3.14 is of form *V:H* as described in Section 3.4.2. For example, the distribution of number of dogs variable (*NumDogs*) for the purchasing behavior represented by column 74 of the CDM is named as *NumDogs:73*, which reflects household with many dogs (see Appendix B). The distribution of female head employment variable (*FHemployment*) for the purchasing

Table 3.14: Returned Distributions for the Mixed Datasets

Distribution Name
HHcomp:0
NumDogs:73
NumCats:123
NielsenCounty:182
NielsenCounty:66
NielsenCounty:283
FHemployment:222
Race:303
MHeducation:76
HHcomp:28

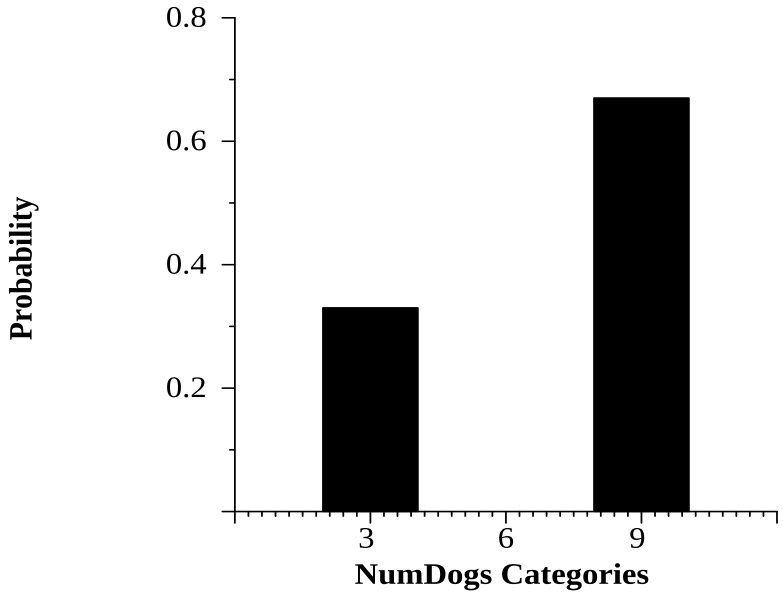
Table 3.15: Returned Purchasing Behaviors for the Mixed Datasets

Purchase Behavior	Cosine	Brand Name	Deal	Size	Type	Flavor
264	0.871	YG_CB	TR	8	Single	Strawberry/Banana
59	0.743	CE_CB	ND	15	Single	N/A
285	0.714	YG_YO	MC	3	Single	Plain
19	0.714	OJ_CB	TR	128	Single	N/A
82	0.655	CE_GMC	TR	9-12	Single	N/A
271	0.612	YG_YO	TC	6	Single	Strawberry
267	0.582	YG_CB	ND	8	Single	Lemon
146	0.582	CE_GMH	TR	12.3	Single	N/A
112	0.580	CE_KFF	MC	20	Single	N/A
198	0.577	CE_GMG	TC	18	Single	N/A

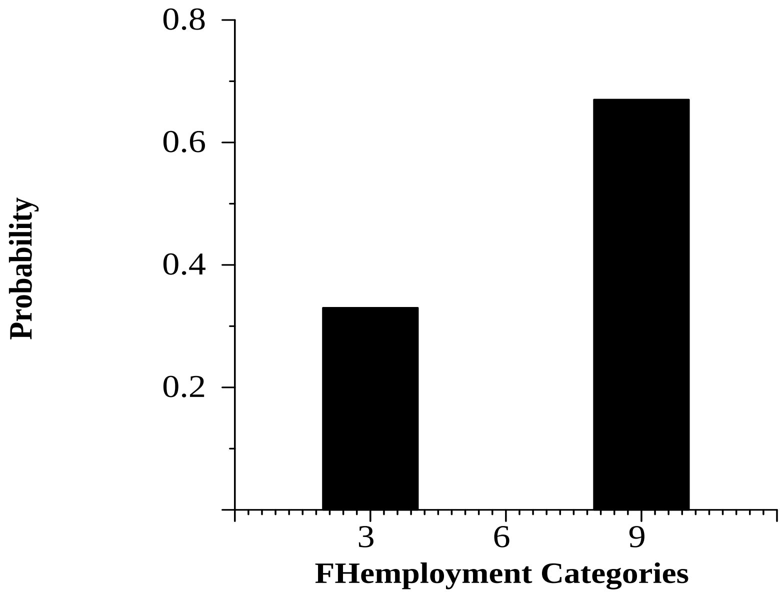
behavior represented by column 223 is labeled as *FHemployment:222*, which reflects household with female head not employed (see Appendix B). Both *NumDogs* and *FHemployment* distributions are presented in Figure 3.7.

From the returned purchasing behaviors in Table 3.15, married consumers who tend to buy yogurt *Control Brands* are likely to buy cereal and orange juice *Control Brands* also. If the returned distributions in Figure 3.7 are also taken into account, the conclusion that *married consumers having lots of dogs, especially if the female head of a family is not employed, are likely to purchase these three products as Control Brands* can be made.

A chi-square testing of the independence of variable household composition (*HHcomp*) and returned purchasing behaviors in Table 3.15 can be determined. The contingency table, whose rows are instance values of variable *HHcomp* and columns are the top 9 instances of purchasing behavior in Table 3.15, is generated in Table 3.16 from the original incidence matrix. The chi-square value is 63.2 for 85% probability level and 48 degrees of freedom. Since the observed chi-square value is 87.4, which is greater than 63.2, the pattern concerning the purchase of control brands can be made with at least 85% certainty. If the probability level 95% is considered, then the chi-square value is 67.5 for 48 degrees of freedom. Since the observed chi-square value 87.5 is greater than 67.5, the extracted pattern concerning control brands can actually be make at 95% certainty.



(a) Returned NumDogs Distribution



(b) Returned FHemployment Distribution

Figure 3.7: Two Returned Distributions for the Mixed Dataset

Table 3.16: Contingency Table for the Mixed Datasets

HHcomp	Purchase Behavior									
	264	59	285	19	82	271	267	146	112	Total
1	2	115	0	65	22	1	1	32	24	262
2	1	15	0	13	2	0	0	4	3	38
3	0	4	0	3	0	0	0	0	0	7
5	0	8	0	4	8	0	0	1	2	23
6	0	2	1	1	1	0	0	1	1	7
7	0	5	0	6	0	0	0	1	1	13
8	0	2	0	0	0	0	0	2	0	4
Total	3	151	1	92	33	1	1	41	31	354

Chapter 4

Summary and Future Work

In addition to the efficient information retrieval from textual documents, LSI can be applied to numerical databases for data mining. The LSI conceptual vector space model represents similar objects in such a way that they can be retrieved even though the objects may not share common attribute values. By projecting user queries into the vector space and analyzing the clustering of nearby attributes or categories, underlying patterns can be extracted from large databases. Further, the extracted patterns can be validated by testing the independence of data variables or resampling to assess the variability of patterns on datasets. The following future work may be conducted to further explore the application of LSI to data mining.

4.1 Loglinear Preprocessing

For consumer product datasets, the distributions of demographic variables have been chosen as terms to index the purchase behaviors. Instead of using correlation analysis, preprocessing with loglinear models [HL85] can be used to determine which variables are strongly correlated. Only those demographic variables strongly correlated to the purchase behaviors would be selected for the LSI model. In general, the loglinear model can be used to choose variables for data mining based on the dependence of all variables.

4.2 Decision Trees

Given a collection of data records with a predefined class attribute, supervised learning from this dataset can be implemented. Decision trees, in particular, can be used to classify data records.

Treating all instance values of the class attribute as *documents* and instances of all other attributes as *terms*, a term-by-document matrix $F = (f_{ij})$ can be defined for subsequent encoding by the LSI vector space model. The entries of the term-by-document matrix can be normalized in such a way that distances between row or column vectors in the space are chi-square distances [HL85].

Even though LSI does not intend to interpret the underlying dimensions of the vector space, the importance of dimension α can be evaluated by $\sigma_\alpha^2 / \sum_\alpha \sigma_\alpha^2$, where σ_α is a singular value of the term-by-document matrix and the ratio is the proportion of total variance that is decomposed in dimension α. Since the sum of weighted squared distances of row vectors (or column vectors) to the origin is equal to σ_α^2 for dimension α, one can also evaluate the relative contribution of row i to dimension α with the ratio $((f_{i+}/n)r_{i\alpha}^2)/\sigma_\alpha^2$, which can be interpreted as the proportion of variance in dimension α accounted for by row i, where $f_{i+} = \sum_j f_{ij}$ and $r_{i\alpha}$ is the coordinate of row i in dimension α [HL85]. With these measures of importance, a decision tree may be constructed from a training dataset by splitting on those important row attributes or developing a new splitting criterion. This decision tree can then be validated against test datasets.

Bibliography

Bibliography

[BDO$^+$93] M. Berry, T. Do, G. O'Brien, V.Krishna, and S. Varadhan. *SVD-PACK (Version 1.0) User's Guide*. Department of Computer Science, University of Tennessee, Knoxville, TN, April 1993.

[BDO95] M.W. Berry, S.T. Dumais, and G. O'Brien. Using Linear Algebra for Intelligent Information Retrieval. *SIAM Review*, 37(4):573–595, 1995.

[BFOC84] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J.Stone. *Classification and Regression Trees*. Wadsworth, Belmont, CA, 1984.

[CT94] B. Cheng and D.M. Titterington. Neural Networks - A Review from a Statistical Perspective. *Statistical Science*, 9(1):2–30, 1994.

[DDF$^+$90] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41:391–407, 1990.

[DF94] K.M. Decker and S. Focardi. Technology Overview: A Report on Data Mining. Technical Report CSCS TR-95-02, Centrum voor Wiskunde en Informatica, Amsterdam, The Netherlands, 1994.

[Dum91] S. T. Dumais. Improving the Retrieval of Information from External Sources. *Behavior Research Methods, Instruments & Computers*, 23:229–236, 1991.

[EG83] B. Efron and G. Gong. A Leisurely Look at the Bootstrap, the Jackknife, and Cross-Validation. *The American Statistician*, 37(1):36–48, 1983.

[FBY92] W. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms.* Prentice Hall, Englewood Cliffs, NJ, 1992.

[FPSSU96] U.M. Fayyad, G. Piatetsky-Shapiro, P. Smyth, and R. Uthurusamy. *Advances in Knowledge Discovery and Data Mining.* AAAI Press/MIT Press, Menlo Park, California, 1996.

[Fri89] J.H. Friedman. Multivariate Adaptive Regression Splines. *Annals of Statistics*, 19:1–141, 1989.

[GBD92] S. Geman, E. Bienenstock, and R. Doursat. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, 4:1–58, 1992.

[GL89] G. Golub and C. V. Loan. *Matrix Computations.* Johns Hopkins University, Baltimore, MD, Second edition, 1989.

[GR71] G. Golub and C. Reinsch. *Handbook for Automatic Computation - Linear Algebra.* Springer-Verlag, New York, NY, 1971.

[HC78] R.V. Hogg and A. T. Craig. *Introduction to Mathematical Statistics.* Macmillan Publishing Co., Inc, New York, NY, fourth edition, 1978.

[HL85] P.G.M. Heijden and J.D. Leeuw. Correspondence Analysis Used Complementary to Loglinear Analysis. *Psychometrika*, 50(4):429–447, 1985.

[HS95] M. Holsheimer and A. Siebes. Data Mining: The Search for Knowledge in Databases. Technical Report CS -R9406, Swiss Scientific Computing Center, 1995.

[Jam91] Michel Jambu. *Exploratory and Multivariate Data Analysis.* Academic Press, Inc., Harcourt Brace Jovanovich Publisher, National Centre for Telecommunications Studies, Paris, France, 1991.

[Let96] T.A. Letsche. Toward Large-Scale Information Retrieval Using Latent Semantic Indexing. Master's thesis, Department of Computer Science, University of Tennessee, 1996.

[MCM86] R.S. Michaslski, J.G. Carbonell, and T.M. Mitchell. *Machine Learning: An Artificial Intelligence Approach*. Morgan Kaufmann, Mateo, CA, 1986.

[Qui92] J Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Francisco, CA, 1992.

[SL90] G. Salton and M. Lesk. Improving Retrieval Performance by Relevance Feedback. *Journal of the American Society for Information Science*, 41(4):288–297, 1990.

Appendices

Appendix A

The Training Dataset for Sample Decision Tree

Table A.1: A Sample Training Dataset

Income	Education	Household Size	class
20	high	four	Coupon
30	college	four	Coupon
20	high	three	No Coupon
70	graduate	four	No Coupon
60	graduate	three	No Coupon
50	college	five	Coupon
70	high	four	No Coupon
30	grade	four	Coupon
60	high	four	No Coupon
30	high	five	Coupon
40	graduate	three	No Coupon
60	college	one	No Coupon
50	graduate	two	No Coupon
40	college	three	Coupon
40	college	four	No Coupon
70	graduate	two	No Coupon
30	high	two	No Coupon
40	grade	three	No Coupon
60	graduate	two	No Coupon
50	college	three	No Coupon
30	college	three	No Coupon
40	graduate	three	Coupon
30	college	two	Coupon

Appendix B

Demographic Variable Coding

Table B.1: Female/Male Head Age Variable Coding (FHage/MHage)

Code	Value
1	under 25 years
2	25-29 years
3	30-34 years
4	35-39 years
5	40-44 years
6	45-49 years
7	50-54 years
8	55-64 years
9	65+ years
0	No female/male head

Table B.2: Female/Male Head Employment Variable Coding (FHemp/MHemp)

Code	Value
0	No female head or unknown
1	under 30 hours
2	30/34 hours
3	35+ hours
9	Not employed for pay

Table B.3: Household Size Variable Coding (HHsize)

Code	Value
1	single member
2	two members
3	three members
4	four members
5	five members
6	six members
7	seven members
8	eight members

Table B.4: Income Variable Coding (Income)

Code	Value
03	under 5000
04	5000-7999
06	8000-9999
08	10000-11999
10	12000-14999
11	15000-19999
13	20000-24999
15	25000-29999
16	30000-34999
17	35000-39999
18	40000-44999
19	45000-49999
21	50000-59999
23	60000-69999
26	70000-99999
27	100000 & over

Table B.5: Kitchen Applicance Variable Coding (KitchenApplicances)

Code	Value
1	microwave only
2	dishwasher only
3	garbage disposal only
4	microwave & dishwasher
5	microwave & garbage disposal
6	dishwasher & garbage disposal
7	microwave, dishwasher, & garbage disposal
8	none

Table B.6: Male Head Education Variable Coding (MHeducation)

Code	Value
1	grade school
2	some high school
3	graduated high school
4	some college
5	graduated college
6	post college grad
0	no female/male head or unknown

Table B.7: Male Head Occupation Variable Coding (MHoccupation)

Code	Value
1	professional
2	prop, managers, officials
3	clerical
4	sales
5	craftsmen/foreman (skilled)
6	operative (semi-skilled)
7	military
8	service workers & private HH workers
9	farm owners, managers, foremen & laborers
0	students employed $<$ 30 hours
10	laborers
11	retired, unemployed

Table B.8: Numer of Dogs and Cats Coding (NumDogs/NumCats)

Code	Value
0	zero
1	one
2	two
3	three
4	four
5	five
6	six
7	seven
8	eight
9	nine

Table B.9: Pet Ownship Variable Coding (PetOwnership)

Code	Value
0	no dog or cat
1	dog only
2	cat only
3	both dog and cat

Table B.10: Race Variable Coding (Race)

Code	Value
1	white
2	black
3	oriental
4	other

Appendix C

Sample Purchase Data Record and Fields

Table C.1: Sample Purchase Data Record

Purchase Data Record
08088948940594012710401650006905000000004850000001 0200394000000006034852MLOZ0000006400000 1000000 5OOO02050E000000029000003000004400000500016126 68000000500010000053300000100003300000000000 0
08108381941794042010401500000236000000001630015114 0200496000000006071007MLOZ0000006400000 1000000 2OOO02007N000000029000003000004400000300016836 15400000500010000053300000300001800000000000 0

Table C.2: Sample Purchase Data Record Fields

Field Title	Columns
Brand Name	67-72
Deal	57-58
Size	81-87
Type	87-90
Flavor	91-96

Appendix D

Purchase Dataset Coding

Table D.1: Orange Juice Brand Name Coding

Code	Brand Name
OJ_CB	Control-Brand
OJ_DD	Donald-Duck
OJ_FG	FL-Gold
OJ_FN	FL-Natural
OJ_MM	Minute Maid
OJ_TR	Tropicana

Table D.2: Yogurt Brand Name Coding

Code	Brand Name
YG_BR	Breyers
YG_CB	Control-Brand
YG_CO	Colombo
YG_DN	Dannon
YG_LN	Light N Lively Free 70
YG_YF	Yoplait Fat Free
YG_YO	Yoplait
YG_WW	Weight Watchers Ult 90

Table D.3: Cereal Brand Name Coding

Code	Brand Name
CE_CB	Control-Brand
CE_GMC	General Mills Cheerios
CE_GMG	General Mills Total Whl Grain
CE_GMH	General Mills Hny/Nut Cheerios
CE_GML	General Mills Lucky Charms
CE_GMW	General Mills Wheaties
CE_KCF	Kelloggś Corn Flakes
CE_KCP	Kelloggś Corn Pops
CE_KCX	Kelloggś Crispix
CE_KFF	Kelloggś Frosted Flakes
CE_KFL	Kelloggś Froot Loops
CE_KFW	Kelloggś Frosted Mini-Wheats
CE_KRB	Kelloggś Raisin Bran
CE_KRK	Kelloggś Rice Krispies
CE_KSK	Kelloggś Special K
CE_NSW	Nabisco Spn Sz Shred Wheat
CE_PGN	Post Grape-Nuts
CE_PRB	Post Premium Raisin Bran

Table D.4: Deal Attribute Coding

Code	Brand Name
ND	No-Deal
MC	Mfr-Coupon
TR	Trade-Only
TC	Trade + Mfr-Coupon

Appendix E

Information Gain Splitting Criterion

The *information gain* criterion uses information entropy to determine on which attribute to split during the partitioning process.

In general, given a probability distribution $P = (p_1, p_2, ..., p_n)$, where p_i is the probability over an attribute for $i = 1$,...,n. then the entropy of P, $I(P)$, is defined by

$$I(P) = -(p_1 * \log_2 p_1 + p_2 * \log_2 p_2 + ... + p_n * \log_2 p_n).$$

If a set T of records is partitioned into disjoint classes $(C_1, C_2, ..., C_k)$ on the class attribute, then the information needed to identify the class of an element of T is $info(T) = I(P)$, where the probability distribution of classes is

$$P = (|C_1|/|T|, |C_2|/|T|, ..., |C_k|/|T|).$$

If we partition T based on a non-class attribute X into $(T_1, T_2, ..., T_n)$, then the information needed to identify the class of an element of T, $info_X(T)$, becomes the weighted average of the $info(T_i)$ defined by:

$$info_X(T) = \sum_{i=1}^{n} (|T_i|/|T|)\ info(T_i).$$

The difference in the information needed before and after partitoning by an attribute X is defined as follows:

$$Gain(X, T) = info(T) - info_X(T).$$

Using $Gain(X, T)$, one can choose the attribute with greatest gain among the attributes not yet considered in the path from the root to split on at each node [Qui92].

Vita

Jingqian Jiang was born in Wuxi, P. R. China on January 28, 1966. She graduated from Xiamen University in P.R. China with the Bachelor of Science degree in Mathematics in 1988. She came to the U.S.A. in November 1990, entered the graduate program in Mathematics at the Johns Hopkins University and received a Master of Arts degree in Mathematics in 1993. She joined the graduate program in Computer Science at the University of Tennessee in January 1996, and was awarded the Master of Science degree in Computer Science in December 1997.