AN APPLICATION OF TOPIC MODELING ALGORITHMS TO TEXT ANALYTICS IN BUSINESS INTELLIGENCE

by

MAJED ALSADHAN

B.S., Kansas State University, 2011

A THESIS

submitted in partial fulfillment of the
requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2014

Approved by:

Co-Major Professor
Doina Caragea

Approved by:

Co-Major Professor
William Hsu

Copyright

Abstract

In this work, we focus on the task of clustering businesses in the state of Kansas based on the content of their websites and their business listing information. Our goal is to cluster the businesses and overcome the challenges facing current approaches such as: data noise, low number of clustered businesses, and lack of evaluation approach. We propose an LSA-based approach to analyze the businesses' data and cluster those businesses by using Bisecting K-Means algorithm. In this approach, we analyze the businesses' data by using LSA and produce businesses' representations in a reduced space. We then use the businesses' representations to cluster the businesses by applying the Bisecting K-Means algorithm. We also apply an existing LDA-based approach to cluster the businesses and compare the results with our proposed LSA-based approach at the end. In this work, we evaluate the results by using a human-expert-based evaluation procedure. At the end, we visualize the clusters produced in this work by using *Google Earth* and *Tableau*.

According to our evaluation procedure, the LDA-based approach performed slightly better then the LSA-based approach. However, with the LDA-based approach, there were some limitations which are: low number of clustered businesses, and not being able to produce a hierarchical tree for the clusters. With the LSA-based approach, we were able to cluster all the businesses and produce a hierarchical tree for the clusters.

Table of Contents

List of Figures

List of Tables

Acknowledgments

While this thesis is my own work, it benefited from the insights and direction of several people. I would like to thank them all for their help and support.

I would like to express my deep and sincere gratitude to my co-adviser, Associate Professor Dr. Doina Caragea of Department of Computing and Information Sciences, Kansas State University. Her understanding, wide knowledge, logical way of thinking, insightful and personal guidance have provided a good basis for the thesis. Her personal guidance, kind support, patience, and encouraging attitude helped constantly while working on this research and writing of my thesis.

I would also like to express my gratitude to my co-adviser, Associate Professor Dr. William Hsu of Department of Computing and Information Sciences, Kansas State University. His suggestions were very useful to me, and helped me a lot while working on this research. He always help and support me, and he is always standing right next to me when I need him. His support will never be forgotten.

I would also like to thank Dr. Mitchell Neilsen and Dr. Gurdip Singh for being members of my M.S. committee.

Very special thanks to my supervisor at AMI, Dale Wunderlich. His wide knowledge, logical way of thinking, suggestions, and motivation have helped me a lot while working on this project at AMI. His experience with economic development was very useful for answering all my questions about the field of economic development. Let me say, he is the best supervisor I have ever had.

I would also like to thank the economic development experts: Rachel Peters, Jonathan Wallace, and Dale Wunderlich who helped us evaluate the results produced in this thesis work.

Dedication

I would like to dedicate this thesis work to my wonderful parents. This work would not have been done without their support and their encouragement for me to study abroad.

Chapter 1

Introduction

In this chapter, we will first provide some motivation for the problem of clustering businesses in Section 1.1. Then, we give a high level overview of the proposed approaches in Section 1.2.

1.1 Motivation

Most people are familiar with the names of large businesses, such as Apple, Microsoft, Google, and Ford, but some people may not realize that today's famous large businesses were initially very small. They were started by either one person, or a small group of people who had a small idea that turned into a small business. That small business started getting larger and larger until it became a large business that today benefits the economy and decreases the unemployment rate.

While there always exists a chance that a small business will become a large business, small businesses in today's economy face many challenges. Running a business on your own involves hard work and making most decisions on your own. Initially, there is little time for holidays and considerable risk involved. Also, because small businesses tend to buy relatively small quantities of raw material and other supplies, they receive lower discounts than larger firms. Small businesses cannot afford to employ a range of specialists, and also

find it harder to raise finance. All those challenges might lead the business to close down, which in turn could negatively affect the economy, and increase the unemployment rate in the country. On the other hand, when a small business becomes larger, its probability of success increases, and some challenges that it faces are overcome. This in turn, benefits the economy. One of the major ways small businesses become larger and survive is through merging with other small businesses to create a larger business. Another way for a small business to survive is by getting bought by a larger company and turned into a child of that larger company, which could benefit both the large and the small businesses. However, in order for small businesses to merge with each other or get bought by larger businesses, there has to be a way for both small and large businesses to be aware of each other.

Being able to categorize businesses into different categories (clusters) based on their activities would help that awareness. The North American Industry Classification System (NAICS) provides a way to organize businesses into hierarchical clusters based on what each business produces. Specifically, the NAICS system works as follows: when a person or a group of people start a business, they have to choose one or more NAICS codes that reflects their end product and what they produce. Each NAICS code consists of digits with a minimum of 2 digits, which is a very broad industry definition, or a maximum of 6 digits, which is a very narrow industry definition.

Although the NAICS codes and NAICS clusters provide a way of organizing businesses into groups or clusters, these clusters have two major problems. This classification system does not take into consideration the fact that in today's economy, the way a business functions is sometimes more important than what it produces. Another problem with the NAICS system is the fact that the codes can be outdated and do not reflect the current status of a business. A business might change or shift its line of production at some point, while the NAICS codes are still those it picked when the business was just a start-up. Thus, we believe that there's a need for a new way to cluster businesses. A way that reflects the current status of a business and the way that a business functions.

When a customer tries to obtain information about a business, they do not look up the business's NAICS codes, but they would go to the business's website instead, or look up the business's services in the yellowpages. Therefore, it is not important for a business to update their NAICS codes, but it is important to keep their websites, and yellowpages' listing up to date, and that is what most businesses do. For that reason, what businesses say about themselves in their websites or listing description reflect their current status and the way they function in a way that NAICS codes fail to capture. Thus, we believe that, in today's economy, clustering businesses into clusters should be based on their services and websites, but not based on codes that were chosen years ago. Therefore, in this thesis work, we present a different way of clustering businesses in the state of Kansas.

1.2 High Level Overview of Proposed Approaches

One of our goals is to use topic modeling techniques to analyze the businesses' data in the state of Kansas, and cluster those businesses. We assume the self-description of businesses on the yellowpages and their websites gives an accurate view of the businesses' activities and how they function. Another goal is to try to overcome the challenges faced by Parimi [2013].

In the initial phase of the work, we cleaned and organized the businesses' data. Then, in the second phase of the work, we took two different approaches to analyze and cluster the businesses. In the first approach, we used Latent Semantic Analysis to analyze the data. On top of Latent Semantic Analysis, we used Bisection K-Means clustering algorithm to cluster the businesses into different clusters. In the second approach, we used Latent Dirichlet Allocation to analyze the cleaned data and cluster the businesses into different clusters by simply analysing the topic distributions produced by LDA. Finally, we visualized the clusters using Google Earth and Tableau.

The rest of the thesis is organized as follows: Chapter 2 gives the background on clustering and topic modeling. In Chapter 3, we present the related work to the work we present in this thesis. In Chapter 4, we formulate the problem of clustering businesses, and explain the approaches that we used in this work. Chapter 5 explains the data collecting and cleaning, the experimental setup describing the experiments that we have performed, and also the methods that we used to visualize our results. In Chapter 6, we discuss the results of our work, and explain the advantages and disadvantages of the two approaches that we used. Finally. in Chapter 7, we summarize and our work, state our conclusions, and present directions for future work.

Chapter 2

Background on Clustering with Topic Modeling

We will first define "clustering" in the field of data mining. Clustering is the process of grouping objects that are similar according to a similarity measure [Jain et al., 1999]. Objects in the same group are more similar to each other than they are to objects from other groups. The objects being clustered can be a corpus of documents or any other objects that can be compared using a similarity measure. Topic modeling is a class of statistical models for finding the underlying semantic structure of a document collection based on hierarchical analysis of the original text [Blei, 2012]. For a large corpus, topic modeling can give a great view of that corpus considering the collection as a whole, the relationship between documents, and the individual documents. There are several topic modeling techniques, but three of the most well-known are Latent Semantic Analysis [Dumais et al., 1988], Probabilistic Latent Semantic Analysis [Hofmann, 1999], and Latent Dirichlet Allocation [Blei et al., 2003]. Clustering and topic modeling are two different techniques; however, they are related to each other. Topic modeling is a viable way of giving us a representation that can be used to calculate similarity. Clustering can then use that similarity in deciding how to cluster objects into groups.

The process of clustering using topic modeling techniques is relatively simple. By representing each document as a topic distribution, topic modeling techniques reduce the feature dimensionality from the number of distinct words appearing in the corpus to the number of topics. Then, similarity between documents' topic distributions can be calculated using a similarity measure, which reflects the similarity of the documents themselves in terms of the topics they cover. Once a similarity measure is calculated, clustering algorithms can be applied to group the documents.

In this chapter, we will explain the two topic modeling techniques, and the clustering algorithm used in this work. Latent Sematic Analysis (LSA) will be explained in Section 2.1. Then we will explain Bisecting K-Means clustering algorithm in Section 2.2. Finally, Latent Dirichlet Allocation (LDA) will be explained in Section 2.3.

2.1 Latent Semantic Analysis

In natural language processing, Latent Sematic Analysis (LSA) is a technique used to analyze the relationship between a set of documents and the terms they contain [Dumais et al., 1988]. In order to do that analysis, LSA produces concepts that represent these documents and terms. Producing these concepts involves the need to perform three major steps [Dumais, 2004], which are: building a term-document matrix, a transformed term-document matrix, and performing a dimension reduction. In the next three sections, we will explain these three major steps in detail.

2.1.1 Term-Document Matrix

The first step of LSA is creating a term-document matrix that captures the number of times every term appears in every document [Dumais, 2004]. The matrix's rows represent the terms in the documents, and the columns represent the documents. The entry in row

m and column n will give the number of times the term i appeared in document j. It is important to point out that before building a term-document matrix, there is a need for pre-processing the corpus that we want to analyze. Pre-processing a corpus that contains textual content can be done by the removal of the stop words that the corpus contain, and stemming all words in our corpus [Manning et al., 2008]. Stop words are a list of extremely common words that appear in a collection of documents, but do not have a high value to any of the documents such as, "a", "in", "and","the", etc. Stemming, on the other hand, is the process of retrieving the root of a word; for example, the words "cars", "car's", and "cars' " can be stemmed to the word "car" [Porter, 1980].

To illustrate the process of pre-processing and creating a term-document matrix, we use the following example: let us assume that we have the following documents:

Document 1: I like to eat apples and bananas.

Document 2: I ate an apple and banana smoothie for breakfast.

Document 3: Cats are cute.

Document 4: I have a cat.

Document 5: Look at this cute cat eating a piece of apple!

After applying stemming and removal of stop words, we have the following collection:

Document 1: like eat apple banana

Document 2: eat apple banana smoothie breakfast

Document 3: cat cute

Document 4: cat

Document 5: look cute cat eat piece apple

We can build a term-document matrix that represents the number of times every term appears in every document. In this example, we have five documents and ten unique terms. That means that our matrix will have ten rows and five columns since the rows of the matrix represent the terms, and the columns represent the documents. Table 2.1 shows the term-document matrix corresponding to our example.

Words	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5
apple	1	1	0	0	1
banana	1	1	0	0	0
breakfast	0	1	0	0	0
cat	0	0	1	1	1
cute	0	0	1	0	1
eat	1	1	0	0	1
like	1	0	0	0	0
look	0	0	0	0	1
piece	0	0	0	0	1
smoothie	0	1	0	0	0

Table 2.1: A term-document matrix that shows the count of terms in documents

2.1.2 Transformed Term-Document Matrix

The second step in LSA is transforming the term-document matrix [Dumais, 2004]. Building a term-document matrix that represents the number of times every term appeared in every document is not enough. That is because there are many common words in English that might appear many times in a collection of documents, but that does not mean they are important to a certain document [Russell, 2013]. A very popular numerical weight that is used to reflect the importance of a word in a document is called "term frequency-inverse document frequency" (TF-IDF).

In order to calculate the term frequency (TF) for a term i in a document j, we first need to count the number of times the term i appeared in document j, which is what we have in our term-document matrix in Table 2.1. Then, we need to find the number of terms in document j, which would be the sum of the values in the column that represents document j. Dividing the term count by the sum gives us the TF for that term. In order to calculate the inverse document frequency (IDF) for a term i, we first count the number of documents, in our collection, that contain the term i, which would be the number of non-zero entries in the row that represents term i. Then, we divide the number of documents in our collection by that count. Taking the $\log_2$ of the division result gives us the inverse document frequency. Finally, multiplying the term frequency (TF) by the inverse document frequency (IDF) gives us the TF-IDF weight for term i in document j [Rajaraman and Ullman, 2012]. The following equation shows how the TF-IDF weight is calculated:

$$w_{ij} = tf_{ij} * idf_i = \left(\frac{f_{ij}}{\sum_x f_{xj}}\right) * \log_2\left(\frac{N}{df_i}\right) \tag{2.1}$$

where i is the term for which we are calculating the weight and j is a document that contains i; f_{ij} is the count of term i in document j, and $\sum_x f_{xj}$ is the total number of terms in document j; N is the number of documents in our collection and df_i is the number of documents, in our collection, that contain term i.

Transforming the term-document matrix in Table 2.1 using TF-IDF gives us a more accurate representation of the importance of every term in our collection relative to the document that contains it. We can apply Equation 2.1 to every single entry in Table 2.1 and that gives us the transformed values for our term-document matrix. Table 2.2 shows the term-document matrix after transformation.

Words	Doc 1	Doc 2	Doc 3	Doc 4	Doc 5
apple	0.12771	0.10217	0	0	0.08514
banana	0.22907	0.18326	0	0	0
breakfast	0	0.32189	0	0	0
cat	0	0	0.25541	0.51083	0.08514
cute	0	0	0.45815	0	0.15272
eat	0.12771	0.10217	0	0	0.08514
like	0.402361	0	0	0	0
look	0	0	0	0	0.26824
piece	0	0	0	0	0.26824
smoothie	0	0.32189	0	0	0

Table 2.2: The transformed term-document matrix for our original matrix

2.1.3 Dimension Reduction

The third step in LSA is using a reduced-rank Singular Value Decomposition (SVD) to perform a dimension reduction [Dumais, 2004]. SVD is a way of factoring matrices into a series of liner approximations that expose the underlying structure of the matrix [Klema and Laub, 1980]. A reduced-rank Singular Value Decomposition is the heart of Latent Semantic Analysis because it finds a reduced dimensional representation of the transformed term-document matrix that emphasizes the strongest relationships between terms and documents, and throws away the noise that the matrix may contain [Dumais, 2004]. As the name implies, singular value decomposition works on decomposing the original matrix into 3 different matrices that when multiplied together give the original matrix [Klema and Laub, 1980]. The reduced-rank singular value decomposition of an $m \times n$ matrix is a factorization of the form:

$$M \approx T_r \times S_r \times D'_r \tag{2.2}$$

where T is an $m \times r$ matrix with orthonormal columns; S is an $r \times r$ diagonal matrix, and D' is the transpose of D which is a $n \times r$ matrix with orthonormal columns.

We can apply a reduced-rank singular value decomposition to the transformed term-document matrix that was constructed in Section 2.1.2 to perform a dimension reduction. However, if we use too few dimensions that means that we lost some of the important patterns in our matrix. On the other hand, using too many dimensions means that we included some noise in our matrix. Some large corpus might require using about 300 to 500 dimensions [Bradford, 2008], but in our example, which was introduced in Section 2.1.1, we can see that there are two main concepts. Furthermore, using two dimensions will help us to identify those concepts. The results of applying SVD and a rank 2 reduction to our transformed term-document matrix is shown in Table 2.3.

$T_r =$

-0.0700	-0.2941
-0.0395	-0.4996
-0.0301	-0.4120
-0.7850	0.1203
-0.5686	0.0429
-0.0700	-0.2941
-0.0393	-0.4656
-0.1512	-0.0491
-0.1512	-0.0491
-0.0301	-0.4120

$S_r =$

0.6616	0
0	0.5707

$D'_r =$

-0.0646	-0.0618	-0.6968	-0.6061	-0.3729
-0.6604	-0.7304	0.0883	0.1077	-0.1045

Table 2.3: The results of applying a reduced-rank SVD to our matrix

Finally, to find the representation of the documents in the reduced space, we need to take the transpose of multiplying the D matrix with the S matrix $(D \times S)'$. Each row in the resulting matrix, which is shown in Table 2.4, represents one of the documents in our collection. Since we have five documents in our collection, we have five rows in our new matrix $(D \times S)'$. Since we reduced the dimensions to 2, we only have two columns. In a 2-dimensional plot, the first column represents the X-axis, and the second column represents the Y-axis. Figure 2.1 shows a graph that represents our collection in a 2-dimensional plot.

-0.0427	-0.3769
-0.0409	-0.4169
-0.4610	0.0504
-0.4010	0.0615
-0.2467	-0.0596

Table 2.4: The transpose of multiplying the D matrix with the S matrix $(D \times S)'$.

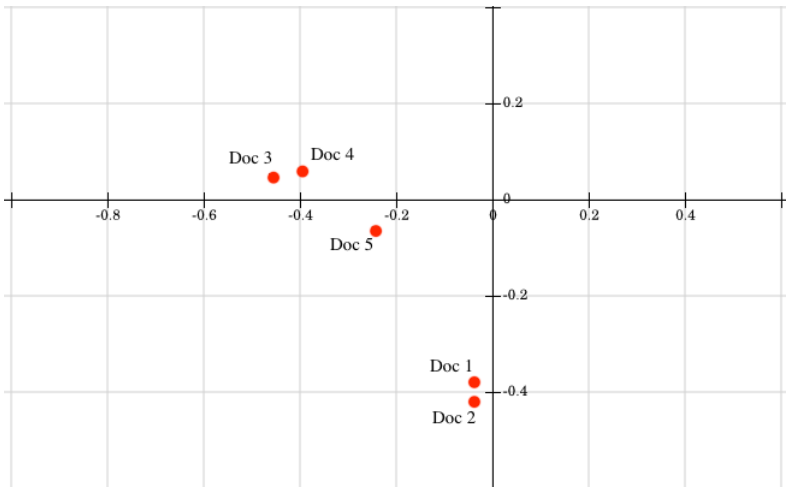


Figure 2.1: *A 2-dimenisitional plot of the documents in our collection*

In Figure 2.1, we can see that the first two documents are very close to each other and that is because they contain similar words which are food-related. Document 3 and 4 are also close to each other because both documents contain words that are related to pets. Document 5 is positioned between the other 4 documents because it contains food-related, and pets-related words.

2.2 Bisecting K-Means Clustering

There are different approaches to clustering such as connectivity-based, centroid-based, distribution-based, and density-based clustering. In this section, we focus on centroid-based clustering, specifically, Bisecting K-Means clustering because this is the type of clustering used in this thesis work. The Bisecting K-Means clustering is an extension of the, well-known technique, Basic K-Means clustering [Steinbach et al., 2000]. In the Bisecting K-Means clustering, we first split the data into two different clusters, then one of these clusters is selected and bisected further. We continue this process until the desired number of clusters is found by performing a sequence of $k-1$ repeated bisections [Steinbach et al., 2000]. Figure 2.2 shows a simple demonstration of the iterations that Bisecting K-Means clustering performs to generate 4 clusters.

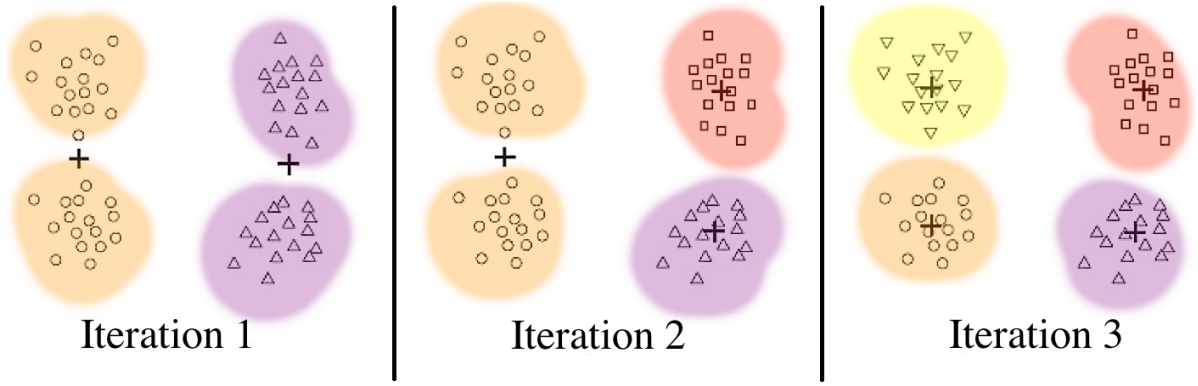


Figure 2.2: *A demonstration of the iterations performed by the Bisecting K-Means clustering.*

We can apply the Bisecting K-Means clustering algorithm to the results of the LSA analysis on the example that we introduced in Section 2.1 to cluster the documents. In that example, we generated a graph of the results obtained by taking the transpose of multiplying the D matrix with the S matrix $(D \times S)'$. In this section, we will use that transpose $(D \times S)'$ to cluster the documents in the example we introduced in Section 2.1.1 which we represent here for the convenience of the readers: let us assume that we have the following documents:

Document 1: I like to eat apples and bananas.

Document 2: I ate an apple and banana smoothie for breakfast.

Document 3: Cats are cute.

Document 4: I have a cat.

Document 5: Look at this cute cat eating a piece of apple!

We first pre-process and analyze the documents by using Latent Semantic Analysis. Then we can apply the Bisecting K-Means clustering algorithm to cluster the documents representation in the reduced space. In this example, we want to cluster the documents into three clusters. Thus, we choose $k = 3$. Figure 2.3 shows the results of clustering the documents.

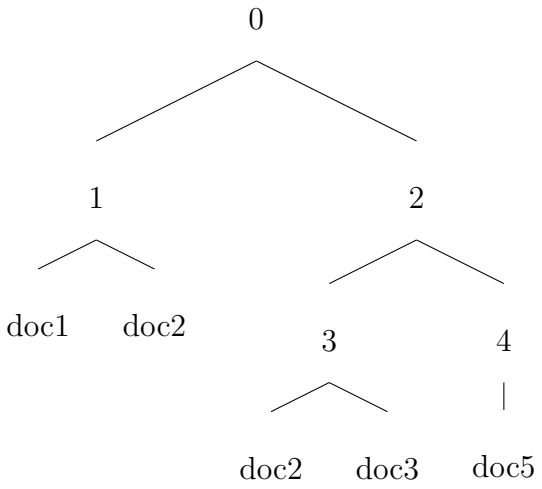


Figure 2.3: *A hierarchical tree showing the way documents are clustered*

We can see in Figure 2.3 that documents 1 and 2 were clustered into the same cluster because they contain food-related words; similarly, documents 3 and 4 were clustered into the same cluster because they contain pet-related words. Documents 3, 4, and 5 are in the same hierarchical branch because document 5 is closer to documents 3 and 4 than it is to documents 1 and 2 as Figure 2.1 shows.

2.3 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a way of automatically discovering topics that a collection of documents contains [Blei, 2012]. The idea behind LDA is that documents exhibit multiple topics. For example, a document entitled "Pets' life", which is about dogs and cats, might contain words about cats, such as "milk", "meow", and "kitten". It might also contain words about dogs, such as "puppy", "bark", and "bone". In this example, we have two different topics, which are cat-related topic, and dog-related topic [Blei, 2012]. What LDA can do in our example is to discover these two topics and represents the document as a topic distributions. In general, when we have a collection of documents, LDA can discover topics in our collection, and represent the documents as distributions of these topics. LDA can perform this discovery by using a process of learning that involves probabilities [Blei, 2012]. LDA is a generative probabilistic model of a corpus, and assumes that a document is a topic distributions, and a topic is a word distributions [Blei, 2012].

In Section 2.3.1, we will explain the learning process that LDA uses to discover topics in a collection of documents. In Section 2.3.2, we will give an example using a small collection of documents.

2.3.1 The Learning Process

The learning process of LDA can be done by using the Gibbs sampler. Casella and George [1992] explains the Gibbs sampler as "a technique for generating random variables from a (marginal) distribution indirectly, without having to calculate the density".

For simplicity in illustration, suppose you have a collection of documents, and you have chosen a fixed number of N topics to discover in your collection. Suppose that you also want to use Latent Dirichlet Allocation to learn the topic representation of each document, and the words associated to each topic.

First, we need to go through each document, and randomly assign each word in the document to one of the N topics. This random assignment would give us both topic representation of all documents, and also word distribution of all the topics. However, that representation and distribution are very poor because they are based on a random assignment, and they definitely need some improvements. Thus, for each document d in our collection, we go through each word w in d, and for each topic t, we need to compute two things:

1. The probability of the topic t giving document d, $P\left(topic\ t|document\ d\right)$, and that equals the proportion of words in document d that are currently assigned to topic t.

2. The probability of word w giving topic t, $P\left(word\ w|topic\ t\right)$, and that equals the proportion of assignments to topic t over all documents that contain word w.

Finally, we reassign word w a new topic where we choose topic t with the following probability:

$$P\left(topic\ t|document\ d\right) \times P\left(word\ w|topic\ t\right) \tag{2.3}$$

By repeating the previous computations a large number of times, we will reach a steady state where our assignments are very good. We can then use those assignments to estimate the topic mixture of each document by counting the proportion of words assigned to each topic within that document. We can also estimate the words associated to each topic by counting the proportion of words assigned to each topic overall.

2.3.2 Example

In this example, we will not demonstrate the learning process of LDA as it involves long computations that can be done programmatically. However, we will show what LDA can do for us when it comes to topic discovery and clustering. Suppose that we have the following set of documents:

Document 1: I like to eat apples and bananas.

Document 2: I ate an apple and banana smoothie for breakfast.

Document 3: Cats are cute.

Document 4: I have a cat.

Document 5: Look at this cute cat eating a piece of apple!

After applying stemming and removal of stop words, we have the following collection:

Document 1: like eat apple banana

Document 2: eat apple banana smoothie breakfast

Document 3: cat cute

Document 4: cat

Document 5: look cute cat eat piece apple

What LDA can do for us is automatically discovering the topics that these documents contain. If we asked LDA to discover 2 topics in our collection, then LDA might produce the following:

Document 1 and 2: 1 Topic A, 0 Topic B

Document 3 and 4: 1 Topic B, 0 Topic A

Document 5: 0.6 Topic B, 0.4 Topic A

Topic A: 0.3 apple, 0.3 banana, 0.3 eat, 0.1 breakfast

Topic B: 0.4 cat, 0.4 cute, 0.1 look, 0.1 piece

By looking at the word distributions of Topic A and Topic B, we can conclude that Topic A is about food, and Topic B is about pets. Then, by looking at the topic distribution for every document, we can conclude that Documents 1 and 2 Belong to (cluster) Topic A, and that Documents 3 and 4 Belong to (cluster) Topic B. Document 5 contains terms from both topics, and thus it cannot be exclusively assigned to a single (cluster) topic. By setting a threshold for the topic probabilities, we can cluster the set of documents in our collection. We can set the threshold to 0.7 and that is going to cluster only four documents in our collection because the highest topic probability for document 5 is 0.6 and that is bellow the threshold we set up. Now, it is clear that documents 1 and 2 belong to the food cluster, and documents 3 and 4 belong to the pets cluster. That is our approach of clustering by using Latent Dirichlet Allocation, and that is how it has been used in this thesis work.

However, it is important to note the ability of using clustering algorithms on top of Latent Dirichlet Allocation as we will explain in the future work section. The reason we clustered by setting a threshold for the topic probabilities is because this is how it was done in [Parimi, 2013], and we would like to compare the results of their approach with the LSA-based approach.

Chapter 3

Related Work

Parimi [2013] introduced the problem of clustering businesses in the state of Kansas based on the content of their websites. Their approach was to clean the data first, and then apply Latent Dirichlet Allocation to analyze the data. They set a threshold for the topic distributions that Latent Dirichlet Allocation found, and clustered businesses based on that threshold. However, according to Parimi [2013], their research faced some challenges that we try to overcome in this work such as data noise, low number of clustered businesses, and lack of proper evaluation technique.

In this work, we analyze the businesses' data, and aim to cluster those businesses by using two topic modeling techniques which are Latent Semantic Analysis [Dumais et al., 1988] and Latent Dirichlet Allocation [Blei et al., 2003]. When we used LSA, we followed the process described by Dumais [2004]. After we applied SVD, we reduced our dimensions to a specific dimension according to the work done in [Bradford, 2008] which investigates the best dimension reduction when using Latent Semantic Indexing.

We also used Bisecting K-Means clustering algorithm, which is described in [Steinbach et al., 2000], to cluster the businesses by using an implementation of the algorithm in a package called "Cluto" that is available from the University of Minnesota and provides an implementation of Bisecting K-Means algorithm [Karypis, 2002].

When we clustered the businesses by using LDA, we followed the approach that was used in [Parimi, 2013] which is using Latent Dirichlet Allocation to discover the different topics in our data set, and cluster the businesses by setting a threshold on the topic distributions obtained by LDA. We used a package called "MALLET" which provides an implementation of Latent Dirichlet Allocation [McCallum, 2002]

Chapter 4

Problem Discussion and Approaches

In this chapter, we first provide a formal definition of the problem addressed in Section 4.1. We then discuss our first approach for solving the problem in Section 4.2.1. Finally, we discuss our second approach in Section 4.2.2.

4.1 Problem Addressed

This thesis deals with the problem of clustering businesses in the state of Kansas into different clusters. Parimi [2013] introduced the problem of clustering businesses based on the content of their websites. According to Parimi [2013], they faced some challenges with their research. Their data set contained some noise, such as *HTML* and *XML* tags, that was caused by using an earlier version of the crawler used in this work; the crawler they used was under the development phase and was not ready to be used. Lack of proper evaluation technique was another challenge that their research faced. Finally, the number of businesses that they were able to cluster was very low. Our goal is to cluster the businesses and attempt to overcome the challenges that were faced by [Parimi, 2013].

4.2 Approaches

In this section, we discuss the approaches followed in this thesis. In the initial phase of the work, we cleaned and organized the businesses' data set, the cleaning phase will be discussed in Chapter 5. Then, in the second phase of the work, we took two different approaches to analyze and cluster the businesses. In the first approach, which we will discuss in Section 4.2.1, we used Latent Semantic Analysis [Dumais et al., 1988] to analyze the data and we used Bisecting K-Means clustering algorithm [Steinbach et al., 2000] to cluster the businesses into different clusters. In the second approach, which we will discuss in Section 4.2.2, we used Latent Dirichlet Allocation [Blei et al., 2003] to analyze the cleaned data and cluster the businesses into different clusters.

4.2.1 LSA-based Approach

Clustering and Latent Semantic Analysis are two different techniques; however, they are related to each other. Latent Semantic Analysis provides a viable way of deciding how similar documents are. Clustering can use that similarity in deciding how to cluster documents into groups. The process of clustering using Latent Semantic Analysis is relatively simple. In representing each document as a reduced concept space, LSA reduces the feature dimensionality from number of distinct words in a corpus to the number of concepts [Dumais et al., 1988]. Then, similarity between documents can be calculated using a similarity measure, which reflects the similarity of the documents themselves in terms of the topics they cover. Based on this quantified similarity measure, many clustering algorithms can be applied to group the documents.

In this approach, we used a well-known topic modeling technique which is Latent Semantic Analysis [Dumais et al., 1988] to analyze the data that we cleaned. We will discuss the process of LSA that we followed in Section 4.2.1.1. On top of Latent Semantic Analysis, we used Bisecting K-Means clustering [Steinbach et al., 2000] to cluster the businesses into

different clusters. We will discuss the clustering part of our first approach in Section 4.2.1.2 Finally, we evaluated our newly generated clusters by using an evaluation method that we will discuss in Chapter 5. Figure 4.1 shows an overview of our LSA-based approach.

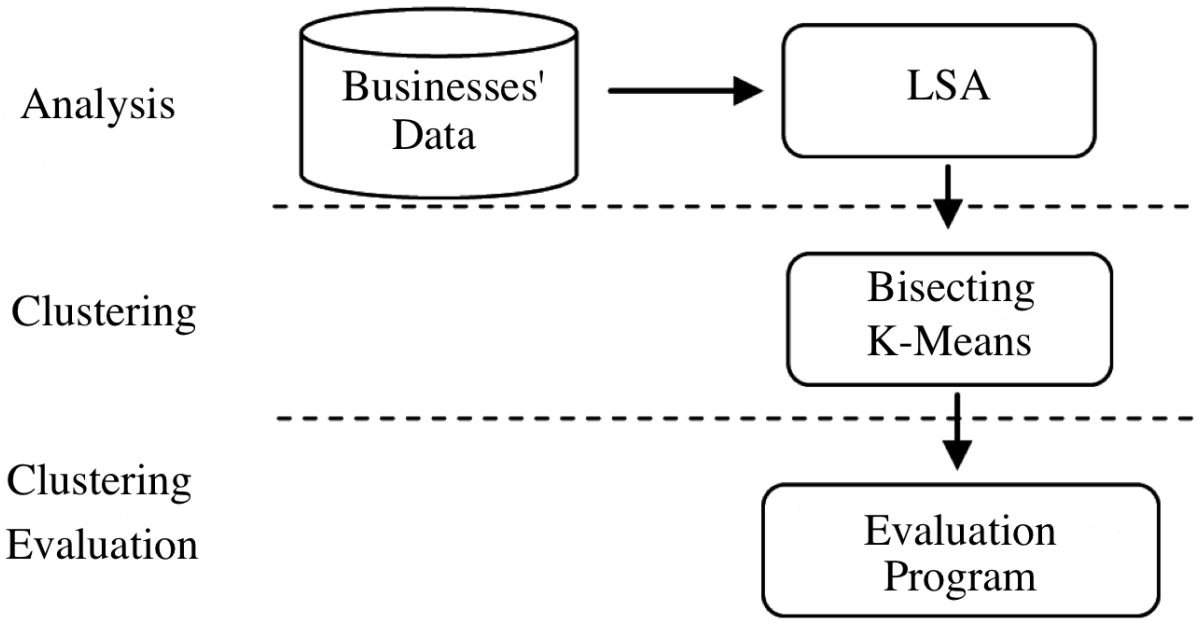


Figure 4.1: *An Overview of the LSA-based approach*

4.2.1.1 LSA

The first phase of our first approach is using Latent Sematic Analysis. As we explained in Section 2.1, In order to use LSA, we need to perform three major steps [Dumais, 2004], which are building a term-document matrix, a transformed term-document matrix, and using singular value decomposition (SVD). However, for our problem addressed in Section 4.1, we do not have a set of documents, but we have a set of businesses that have services and websites which contain textual contents. We consider those businesses to be our documents that we are trying to analyze using LSA, and their services and websites' contents to be our terms. We analyzed the data of the businesses by doing the following:

1. We created a *Python* script that reads our cleaned data set, and generates our term-document matrix which captures the counts of every term in our corpus. The $m \times n$ matrix that we generated contains m rows which represent the services and the websites' contents of the businesses, and n columns which represent the businesses that we are trying to cluster. Our $m \times n$ matrix contains 18888 rows (terms) and 4291 columns (businesses)

2. We transformed our $m \times n$ matrix using a very popular numerical weight that is used to reflect the importance of a word in a document which is called "term frequency-inverse document frequency" (TF-IDF). The transformed matrix has the same number of rows and columns as our original matrix, but it has different entries. The entries in our transformed matrix have been calculated by applying to our matrix Equation 2.1

3. We wrote a *Matlab* script that takes the transformed term-document matrix as its input and calculates the reduced-rank Singular Value Decomposition (SVD) of our transformed term-document matrix by using the *Matlab*'s implementation of SVD. The script reduces the dimensions of the SVD results to a rank r of 300 which is a proper rank for our corpus size which contains 4291 businesses [Bradford, 2008].

4. We then used the results of the reduced-rank SVD to calculate the businesses representation in the reduced space. As we discussed in Section 2.1.3 the reduced-rank singular value decomposition returns three matrices according to Equation 2.2: T is a $m \times r$ matrix, S is an $r \times r$ matrix, and D' is an $r \times n$ matrix. Calculating the businesses representation in the reduced space was straight forward and easily done by using *Matlab*. All we had to do was take the transpose of multiplying the D matrix by the S matrix which gives an $r \times n$ matrix. Calculating the businesses representation in the reduced space was the end of the analysis part of the businesses' data, and the beginning of the clustering part which we will discuss in the Section 4.2.1.2.

4.2.1.2 Bisecting K-Means Clustering

The second phase of our first approach is using Bisecting K-Means clustering to cluster the businesses. After we analyzed our data set by using Latent Sematic Analysis in Section 4.2.1.1, we obtained an $r \times n$ matrix, where r is the number of the reduced dimensions, and n is number of businesses. We then applied Bisecting K-Means clustering to our $r \times n$ matrix by using an implementation of the algorithm; specifically, a package called "Cluto" which is available from the University of Minnesota [Karypis, 2002]. *Cluto* comes with a library interface and two stand-alone programs which are the *vcluster* and *scluster* command-line programs. There are not many differences between the *vcluster* and *scluster* programs except the input type. The *vcluster* program takes as its input a matrix; on the other hand, the *scluster* takes a graph as its input. We used the *vcluster* program to cluster the businesses. The *vcluster* is an easy-to-use program, and doesn't require many modifications to our $r \times n$ matrix. All we had to add is one line at the beginning of the text file that contains our $r \times n$ matrix. The line is added to let the *vcluster* program know how many rows and columns the matrix contains. In our $r \times n$ matrix, we had 300 rows, which represent the reduced dimensions, and 4291 columns, which represents the businesses. We run the *vcluster* by using its default parameters which apply a Bisecting K-Means clustering and use cosine similarity measure. The experts believe that there are 30 unique clusters in the data set; thus, we sat $k = 30$ to cluster the businesses into 30 clusters. After running the program, we were able to cluster the businesses into 30 different clusters and produce a hierarchical tree for our clusters. Figure 4.2 shows the hierarchical tree produced for our clusters.

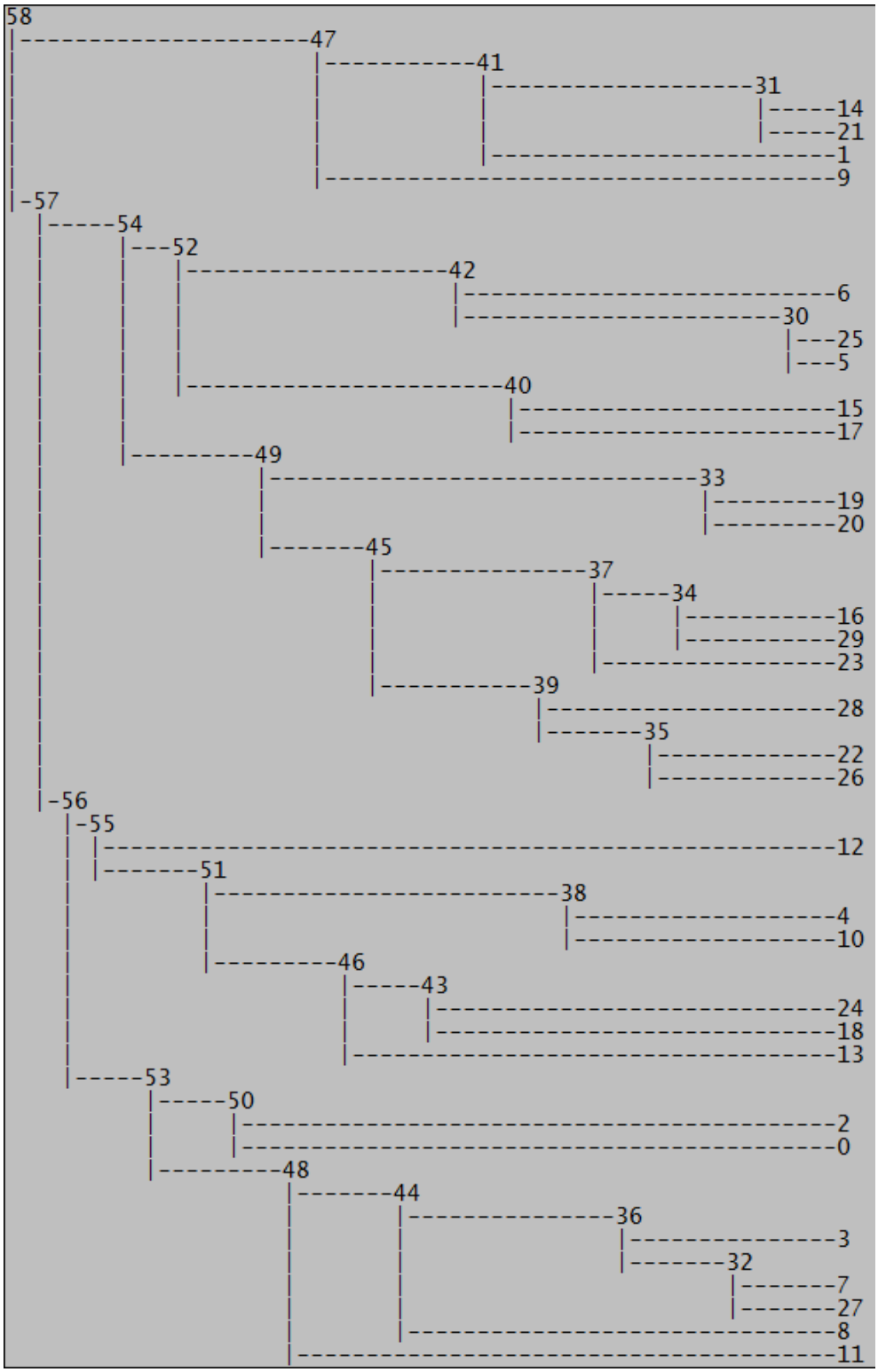


Figure 4.2: *The hierarchical tree produced for clustering the businesses*

4.2.2 LDA-based Approach

In this approach, we used a well-known topic modeling technique, which is Latent Dirichlet Allocation, to discover topics distributions in our corpus [Blei et al., 2003]. We used the topic distributions of businesses in our corpus, that Latent Dirichlet Allocation discovered, to cluster the businesses by taking a threshold of the highest topic probability for every business, which is the same approach that was followed by Parimi [2013]. In this clustering approach, any businesses, whose highest topic probability is above our threshold, have been clustered into a cluster that represents that topic. Any businesses, whose highest topic probability is bellow our threshold, have been discarded. Finally, we evaluated our newly generated clusters by using an evaluation method that we will discuss in Chapter 5. Figure 4.3 shows an overview of our LDA-based approach.

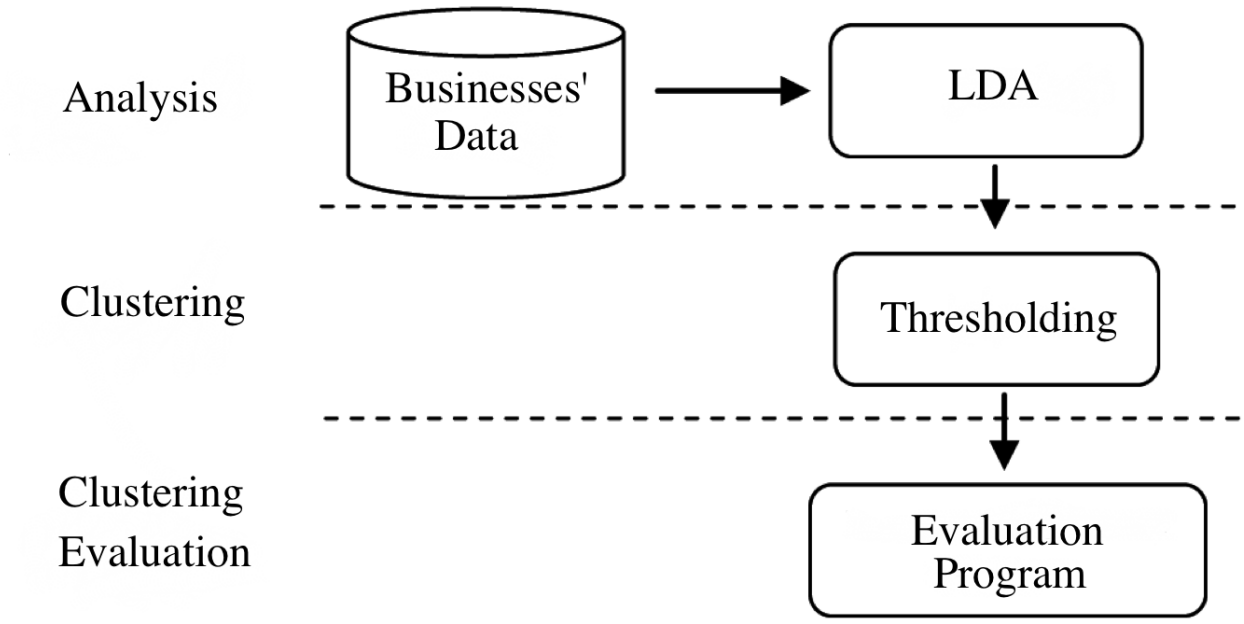


Figure 4.3: *An Overview of the LDA-based approach*

We used "MALLET" which provides an implementation of Latent Dirichlet Allocation [McCallum, 2002]. Like *Cluto*, *MALLET* also provides a library interface, and a stand-

alone program. We used the stand-alone program to apply Latent Dirichlet Allocation to our data set. We instructed the program to discover 30 topics from our data set since this is what the economic development experts think about the businesses that we are trying to cluster. The program produced different files that contained different results such as the top words of each topic and the topic composition of businesses which is very important to us since we will be using that file to cluster the businesses.

The topic composition of businesses contains a list of businesses and the topic distributions for every business. We used this list to cluster the business by writing a *JAVA* program that takes the list as its input, and creates another list which contains the businesses whose highest topic probability is higher than a specific threshold. We set our threshold to 0.7 which is almost the average of the businesses' highest topic probabilities. Any business whose highest topic probability is bellow the threshold has been discarded. Any business whose highest topic probability is over the threshold has been clustered into a cluster that represents that topic. For example, a business b might have the following topic distributions: 0.75 Topic A, 0.12 Topic B, ... etc, then business b is clustered into Cluster A because its A-topic probability, which is b's highest topic probability, is higher than the threshold. Another example, a business b might have the following topic distributions: 0.45 Topic A, 0.09 Topic B, ... etc, then business b is discarded and never included in any cluster because it's highest topic probability, which is topic A, is less than the threshold. Since we chose 30 topics to discover, then the number of clusters at the end of clustering was 30 clusters.

However, since we are discarding businesses whose highest topic probability is less than our threshold, and since our threshold is set to 0.7 which is almost the average of the businesses' highest topic probabilities, then we lost more than half of the businesses that we were trying to cluster. The list of businesses in our data set, after cleaning and organizing, contained 4291 businesses, but after clustering, had 2049 businesses.

Chapter 5

Experimental Setup

In this chapter, we discuss the experimental setup for this thesis work. We first explain the research questions that we have addressed in Section 5.1. Then, we describe how the data was collected, how the data was organized and formatted, give an overview of the data set, and explain the process that we followed to clean the data set in Section 5.2. We then explain the evaluation procedure that was used to evaluate the clusters that were produced in this work in Section 5.3. Finally, we visualize the clusters that we produced in Section 5.4.

5.1 Research Questions

In this work, we address the following research questions and the challenges faced by Parimi [2013]:

1. How can we reduce the noise in the businesses' data set?
2. Is there a way to produce a hierarchical tree when we cluster the businesses?
3. Is it possible to increase the number of clustered businesses?
4. How can we evaluate the clusters produced in this work?

We perform 2 experiments in this thesis to cluster the businesses and overcome some of the challenges faced by Parimi [2013]. The approaches of the experiments are the following:

LSA-based Approach: As described in Chapter 4, in this approach, we converted the businesses' data from words to concepts by using LSA. Then, we clustered the businesses by applying Bisecting K-Means clustering to the concepts produced by LSA. We were able to produce a hierarchical tree of our clusters. We were also able to cluster all the businesses in our collection which was one of the challenges faced by Parimi [2013].

LDA-based Approach: This approach was the same approach taken by Parimi [2013]. We discovered the different topics in the businesses' data by applying LDA. Then, we clustered the businesses by setting a threshold on the topic distributions produced by LDA. Unfortunately, we were not able to produce a hierarchical tree of our clusters, and we were not able to cluster all the businesses in our collection because of the way that the LDA-based approach works. More than half the businesses were discarded because their highest topic probability was bellow a certain threshold.

5.2 Data Collection and Preprocessing

This project is a collaborative work with the "Advanced Manufacturing Institute" (AMI), which is an engineering company located in Manhattan, Kansas. AMI provided us with the data, and hired us to find a solution to the businesses clustering problem and the challenges that we discussed in Chapter 4. In this section, we describe how the data was collected in Section 5.2.1. We then give an overview of the data set, explain how the data was organized and formatted, and explain the process that we followed to clean the data set in Section 5.2.2.

5.2.1 Data Collection

AMI purchased a list with information about businesses in the state of Kansas. The list is available commercially from "dun and bradstreet", a company located in Short Hills, New Jersey. The list contains 4317 Kansas businesses with the following information about them: their names, contact information, location information (including addresses, altitude, latitude, and longitude), employee class information, services, and most importantly, their websites. AMI also purchased a crawler called *ExpertFinder* that was developed by Dr. Tim Reichling who is a member of "C^3 networking solutions".

The crawler used in this work is an updated version of the crawler that was used in Parimi [2013]. The updated version of the crawler performs a better job at removing noise such as *HTML* and *XML* tags which was one of the challenges faced by Parimi [2013].

The crawler works by crawling a list of seeds, then extracting the words from the websites it crawled by removing *HTML* or *XML* tags. The crawler also assigns a weight of 1 - 4 to the words based on their size in the *HTML* or *XML* tags. AMI provided the *ExpertFinder* software, which is shown in Figure 5.1, with a list of the businesses' websites to crawl (while respecting the "robots.txt" rules of the websites). The crawler crawls the websites every 24 hours to ensure that the data obtained is up to date. The data that was used in this thesis work was obtained on the 21st of October, 2013. The number of websites that AMI was able to crawl was 2890, which means that there are 1427 businesses that AMI was not able to crawl because of one of the following reasons:

1. The "robots.txt" file does not allow crawling.

2. The business does not have a website.

In Section 5.2.2, we discuss how we were able to include into our data set some of the 1427 businesses that AMI was not able to crawl.

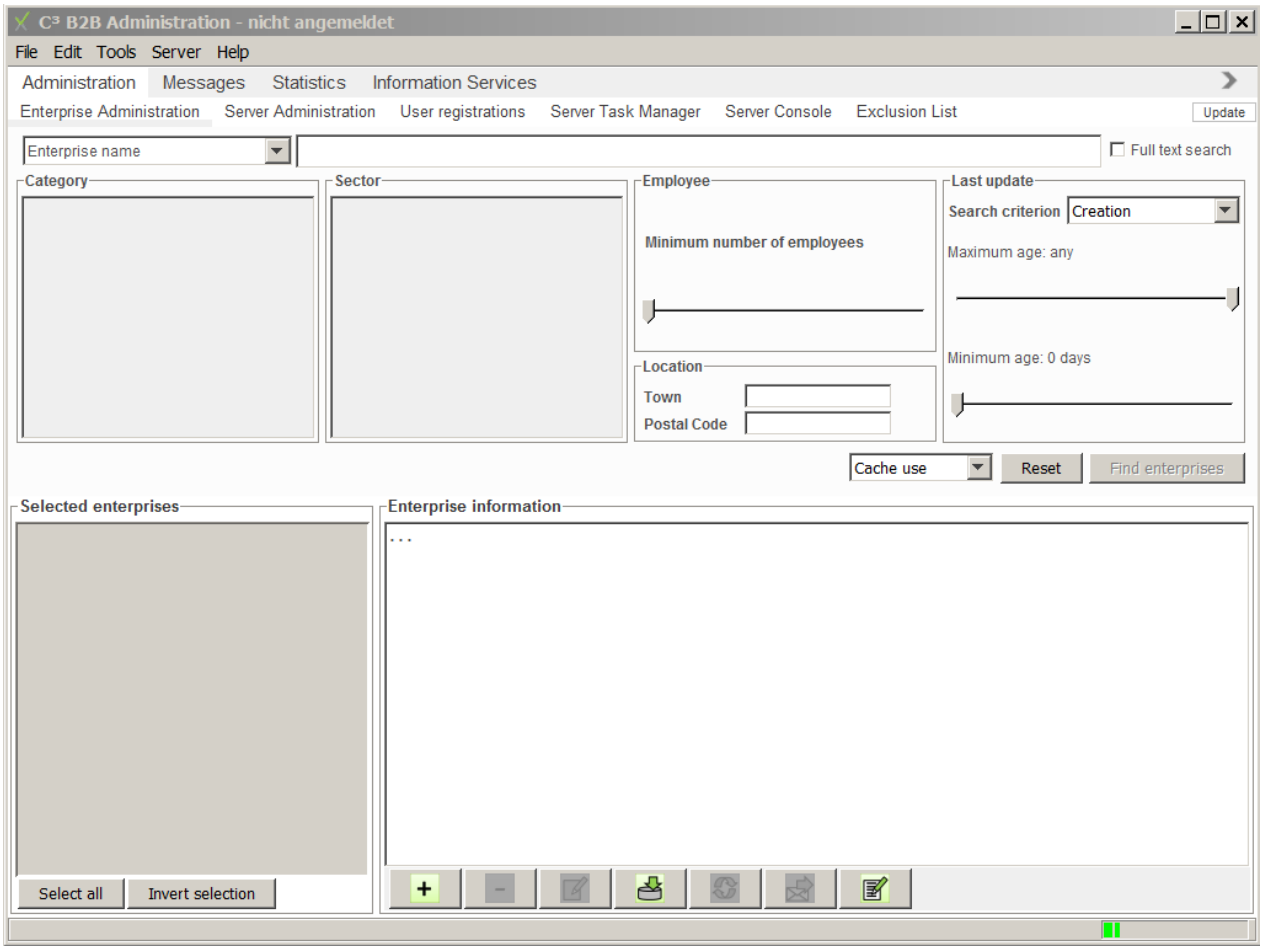


Figure 5.1: *The interface of the ExpertFinder*

5.2.2 Data Preprocessing

As we explained in Section 5.2.1, AMI bought a list with information about businesses and was able to crawl 2890 businesses of that list. The list provides some information about the businesses including their services. The services information is basically what businesses say about themselves and is used to list businesses in different places such as yellowpages. Therefore, we believe that businesses are keeping their services information up to date, and thus, we believe that the services information is an important part of the data that we have and should not be ignored. We used the bag-of-words model to represent the textual

content in our collection. In this model, the order of words and grammar is ignored. We used both the information that we obtained from the businesses' websites and also the services information that *dun and bradstreet*'s list provides. By using the services information, we were able to include 1401 businesses out of the 1427 businesses that AMI was not able to crawl because the businesses do not want their websites to be crawled or they do not have websites. Therefore, the total number of businesses in our collection has increased to 4291 businesses. We used the weights that the *ExpertFinder* assigns to the words by repeating the words in our collection the same number of times as the words' weights. For example, if a word w that appears in the website of business b has been assigned a weight of 2 by the *ExpertFinder*, then the word w is repeated 2 times in the bag-of-words of business b. We also repeated the words in the services information 10 times and added them to the bag-of-words of the related business. So our data set now includes a bag-of-words for every business of the 4291 businesses which includes the services information of the business (repeated 10 times), and the words of its website (if it was crawled) repeated 1 - 4 times based on their weights. Figure 5.2 shows our representation of the data.

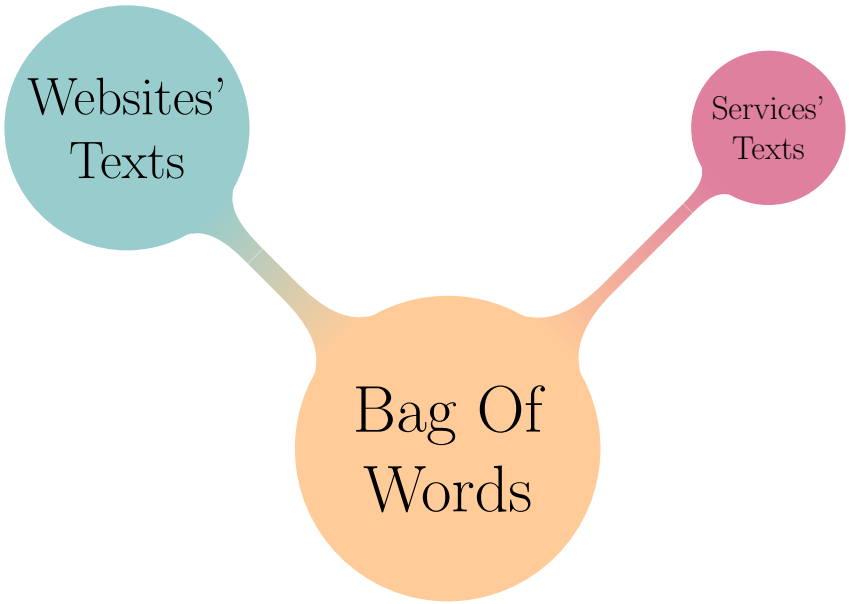


Figure 5.2: *The data set representation*

Cleaning the data that we organized was done in two phases. In the first phase of cleaning the data, we used a list of stop words that contains more than 1300 words in order to remove some unwanted words. The list of stop words consists of the common stop words in English, and some other words such as, the months of the year, the cities of Kansas, and the states of America. The reason why we removed any geographical locations (cities of Kansas) is because we believe that if two businesses are located in the same city, that does not necessary mean that they are related to each other. In other words, the locations of businesses that appear on their websites are considered to be noise in our data set, and they had to be removed from our data set. We also decided to exclude the words that contain the following: "@", "-", "#", and "*" because these words are more likely to represent e-mail addresses, phone numbers, and some notes that websites usually have at the bottom of their pages, and for that reason, we believe that they represent noise in the data set and there is a need to remove them.

In the second phase of cleaning the data, we decided to stem every word in our data set because stemming has proven to be effective [Porter, 1980]. Since we are using a bag-of-words model, then stemming will not harm our data model because grammar and the format of words are not important in this model. We used an implementation of Porter stemmer algorithm [Porter, 1980] which was provided in the "LingPipe" package [Baldwin and Carpenter, 2003]. The following represent the approach we used to clean the bag-of-words set for every business b:

1. Create a new bag-of-words set for business b.
2. For every word w in the bag-of-words of business b do the following:
   (a) If the word w is not in the stop words list, proceed to step (b).
   (b) If the word w does not contain: "@", "-", "#", or "*" :
      i. Stem the word using Porter stemmer.
      ii. Add the word to the new bag-of-words of business b.

5.3 Evaluation Procedure

To evaluate our results, we decided to use 3 human experts in the field of economic development to judge our results. We implemented a program that reads the results of either of our approaches, then randomly promotes two businesses to the judge, and displays their websites without mentioning if the two businesses have been chosen from the same cluster or different clusters. The judge then decides if the two businesses are supposed to be in the same cluster or not. There are 100 test cases that the judge has to answer with either "yes" or "no". In case the judge is not able to make a decision, then they have the option to ask the program to randomly promote another two businesses without counting a test case. When the test cases are over, the program then displays a save file dialog that allows the judge to save the results of testing. The program, whose interface is shown in Figure 5.3, works as follows:

1. Load the results of one of the two approaches.
2. For 100 times, randomly generate a number from 0 to 100.
   (a) If the number is less than 50:
      i. Randomly choose a cluster.
      ii. Randomly choose two different businesses from the cluster.
      iii. Display the two businesses, and their websites to the judge.
   (b) If the number is grater than or equal to 50:
      i. Randomly choose two different clusters.
      ii. Randomly choose two different businesses from the two clusters.
      iii. Display the two businesses, and their websites to the judge.
3. Save the results of testing.

Figure 5.3: *The interface of the program used to evaluate the results*

The program counts the number of times a judge agrees or disagrees with the results of our two approaches, then saves the results of testing in a text file. For example, if the program chooses two businesses from the same cluster and the judge answers "Yes", then that's an agreement, but if the judge answers "No", then that's a disagreement. Similarly, if the program chooses two businesses from different clusters and the judge answers "Yes", then that's a disagreement, but if the judge answers "No", then that's an agreement. We then averaged the results of the testings that were done by the 3 judges, and reported them in Chapter 6. This evaluation procedure overcomes the lack of evaluation approach faced by Parimi [2013].

5.4 Visualization

In this Section, we discuss the different methods that we used to visualize the businesses after clustering them, and show some samples of our visualizations. In section 5.4.1, we explain how we used *Google Earth* to visualize the businesses in our clusters and in Section 5.4.2, we explain how we used *Tableau* to visualize the businesses that we clustered.

5.4.1 Google Earth

Google Earth is a very well-known geographical information program that can be used to browse 3D maps, and navigate across the world. By using its satellite view option, you can look at almost any place in the world while you are sitting in your home. It is a powerful program that can be used to draw places in its maps, and that is why we chose to use it. *Google Earth* provides the ability for users to add their personal locations and buildings, and be able to use it for visualization purposes. It has the ability to read a kml-formatted file that contains information about personal locations, and displays the locations on its up-to-date maps. The kml-formatted file needs to contain information about the locations that the user is trying to view such as, altitude, latitude, and longitude. It can also take information about locations such as, an icon to display for a location, name of the location, and a description for that location. As we mentioned in Section **??**, the list of businesses that AMI purchased contains the information that we need in order to generate the kml-formatted file that needs to be used in *Google Earth*. We wrote a *JAVA* program that takes the list of businesses, and generates the kml-formatted file. The following kml code is an example of one of the businesses in our clusters:

```
<Placemark xmlns="">
    <styleUrl>#m_point-icon</styleUrl>
    <styleUrl>#m_point-icon</styleUrl>
    <name>A Z Mobile Rv Inc</name>
    <description>;a href=http://www.azmobilerv.com/</description>
    <Style>
        <IconStyle>
            <Icon><href>http://majedalsadhan.com/AMI/0.png</href></Icon>
        </IconStyle>
        <LineStyle>
            <width>6</width>
        </LineStyle>
    </Style>
    <Point>
        <extrude>1</extrude>
        <altitudeMode>relativeToGround</altitudeMode>
        <coordinates>-94.900595,38.534293,6000</coordinates>
    </Point>
</Placemark>
```

We created similar codes for every business in our clusters, and we were able to visualize the businesses by using *Google Earth*. Figure 5.4 shows an example of the visualization of the clustered businesses by using *Google Earth*.

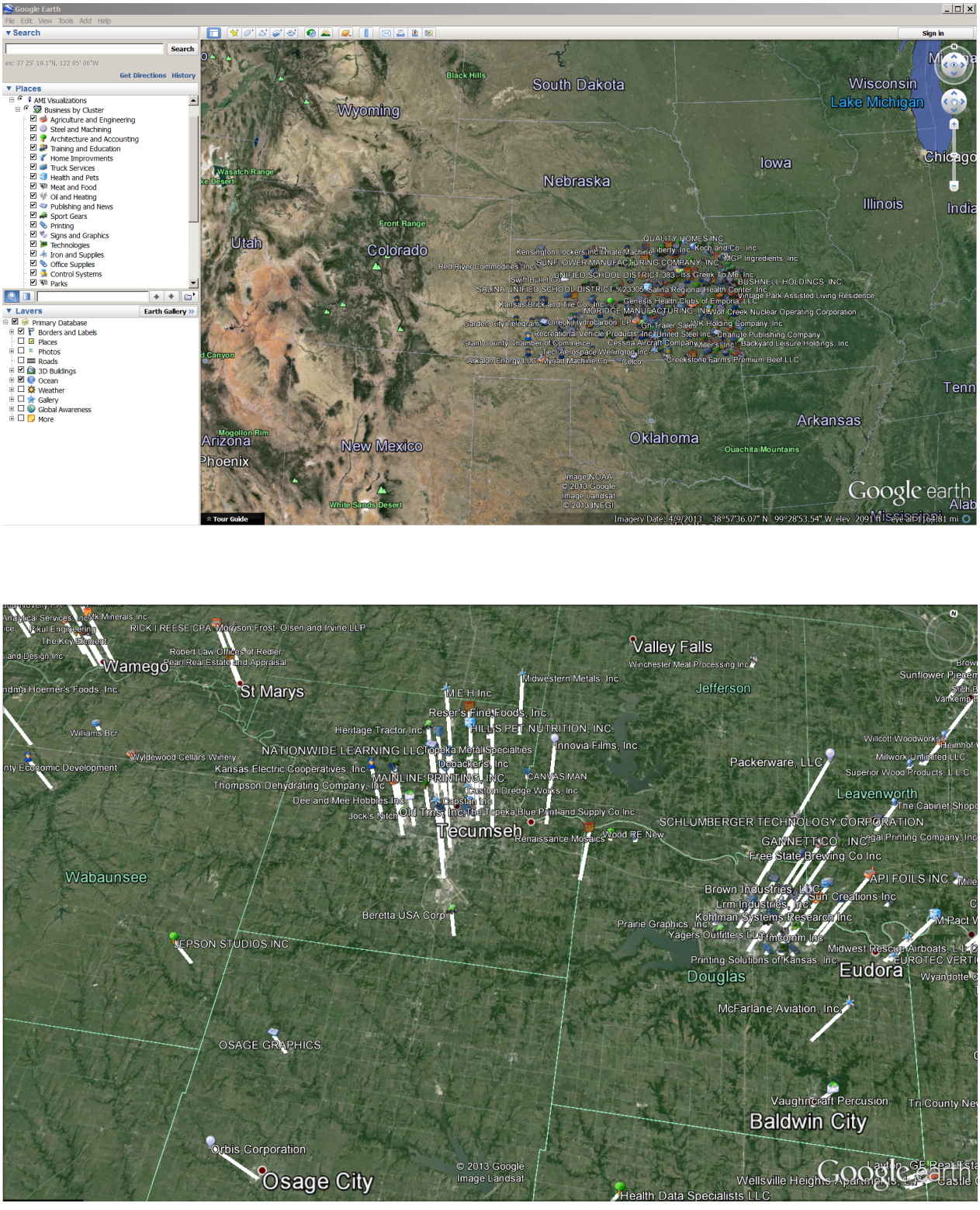


Figure 5.4: *Google Earth visualization*

5.4.2 Tableau

Tableau [1], which is shown in Figure 5.5, is one of the best and easy-to-use software for data analysis and visualization. With *Tableau* you can create interactive graphs, dashboards, maps and tables from virtually any data.

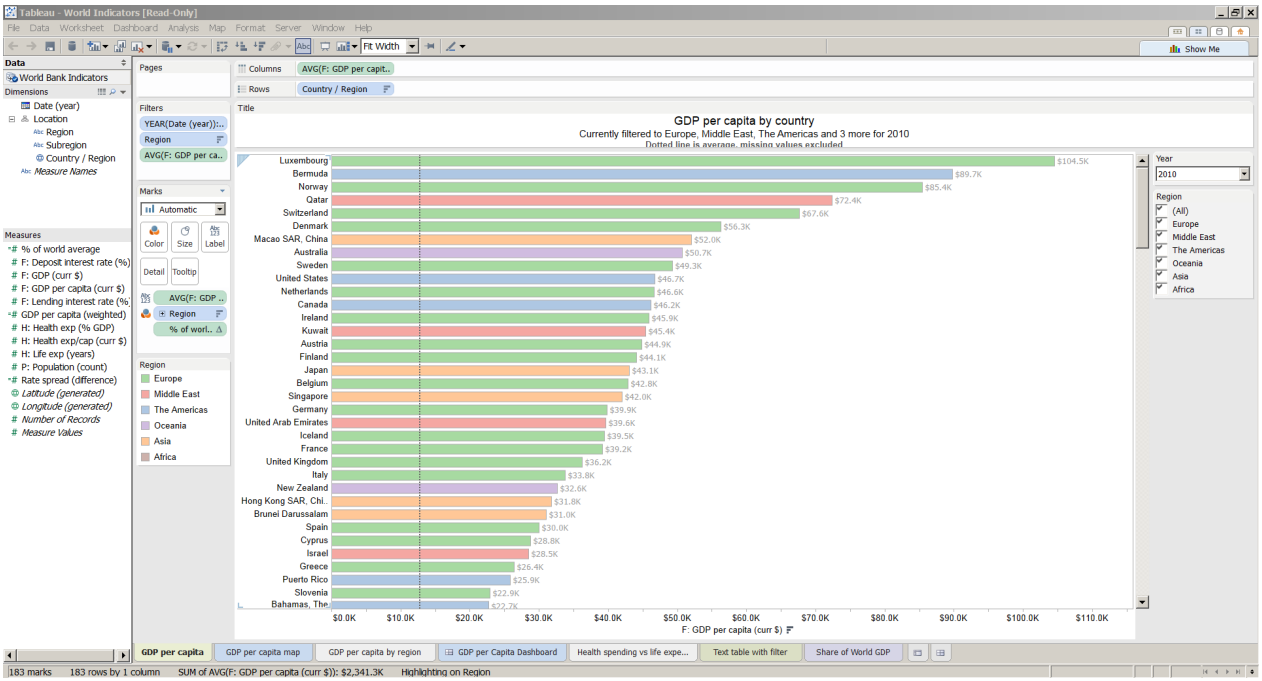


Figure 5.5: *Tableau software*

The great technology behind *Tableau* is called "VizQL" which stands for "A Visual Query Language". *VizQL* is a formal language that is designed to fill in the gap that conventional query languages, such as *SQL* and *MDX*, have between their powerful queries and their limited formatting and visualization capabilities [Hanrahan, 2006]. *VizQL* is used to describe many types of visual representations such as, tables, charts, graphs, maps, and time series; it can be used with relational databases, and it supports Hyperion Essbase, Microsoft SQL Server, Microsoft Analysis Services, MySQL, Oracle, as well as spreadsheet-based data files such as CSV and Excel files [Hanrahan, 2006].

[1] http://www.tableausoftware.com/

We used *Tableau*'s research license which allows the use of *Tableau Pro* version for research purposes. The license, which is a year long, provides to students with *.edu* email addresses a free version of *Tableau Pro*. As we mentioned before, *Tableau* supports spreadsheet-based data files such as CSV and Excel files. Therefore, we decided to write a simple *JAVA* program that converts the results of clustering into a CSV-formatted file. We included for every business that we clustered, the following information about the business: latitude, longitude, cluster, website, county, and services. We were able to provide a visualization of the businesses, the ability for the user to filter businesses by cluster, or county, and show some useful statistics as shown in Figure 5.6, which is a composition of what we did in Figure 5.7.

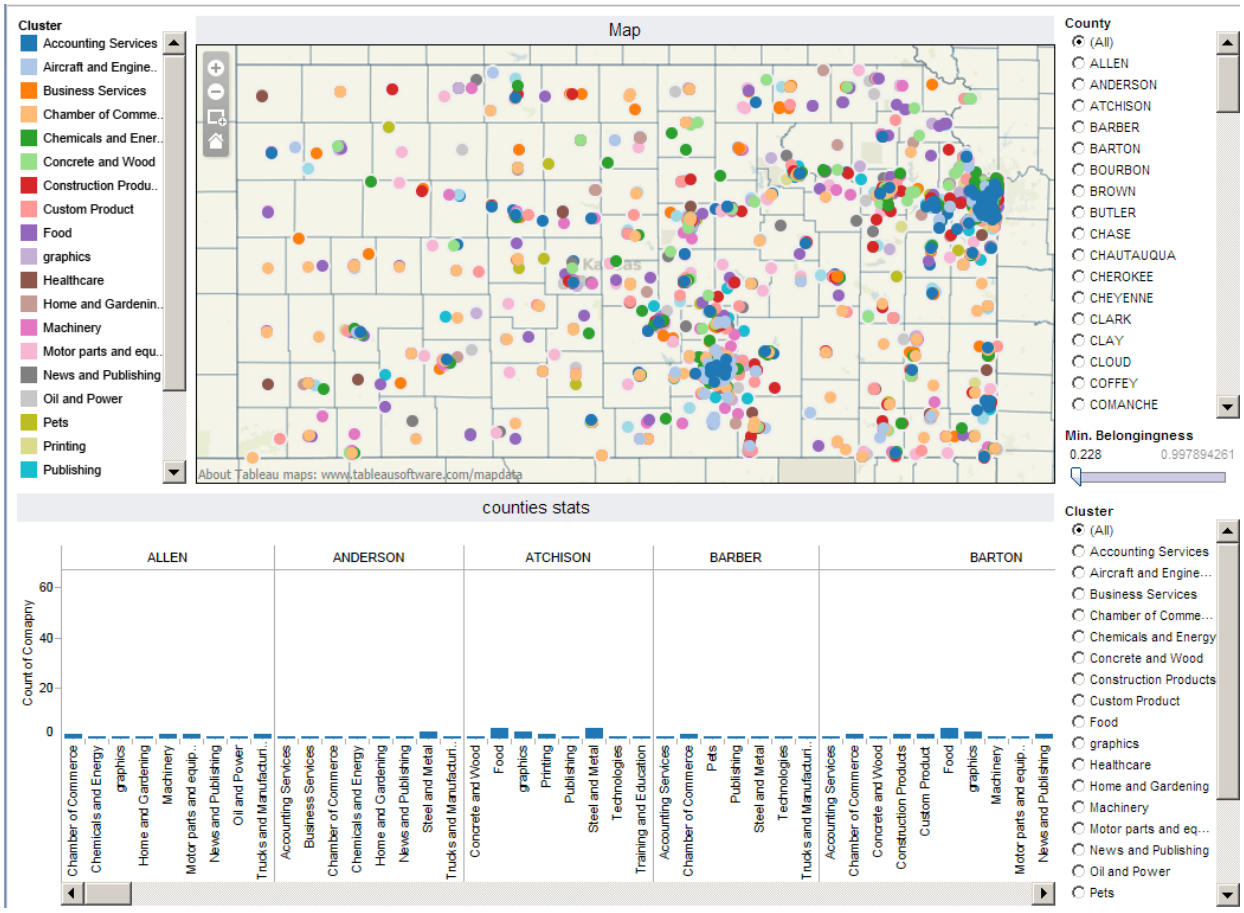
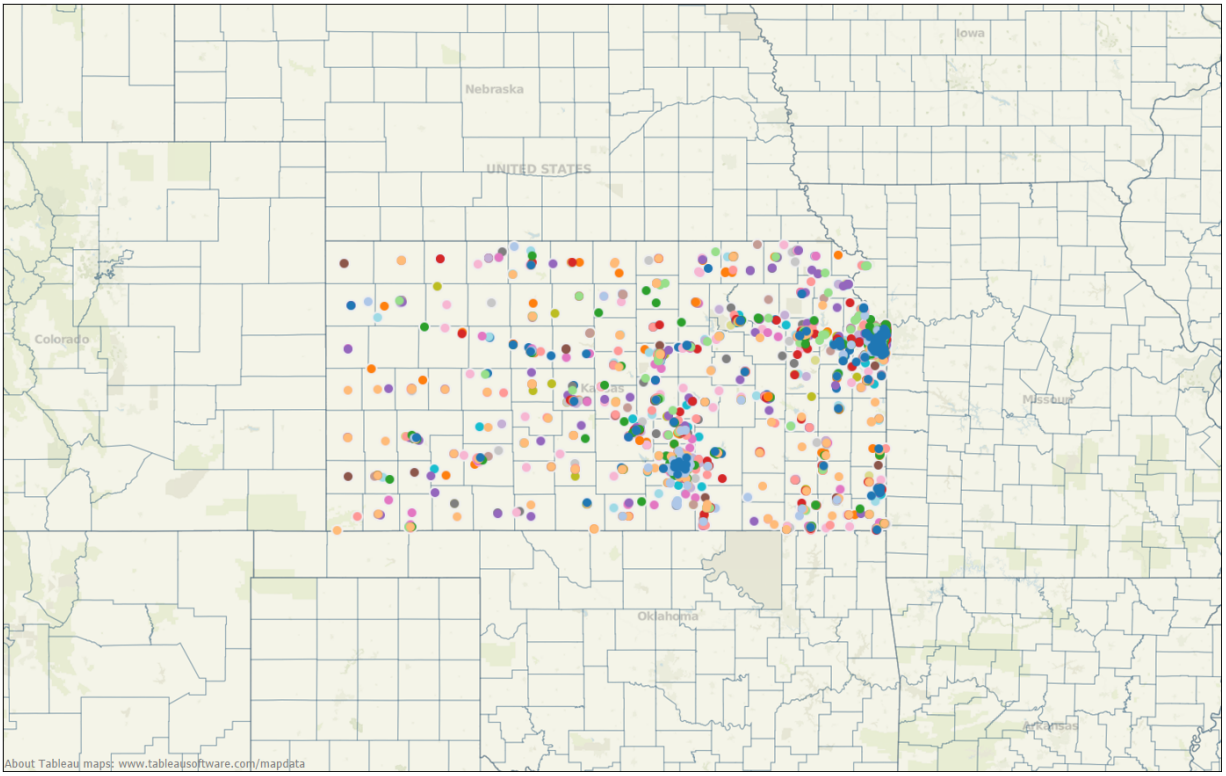


Figure 5.6: *The ability to filter businesses by cluster, or county*



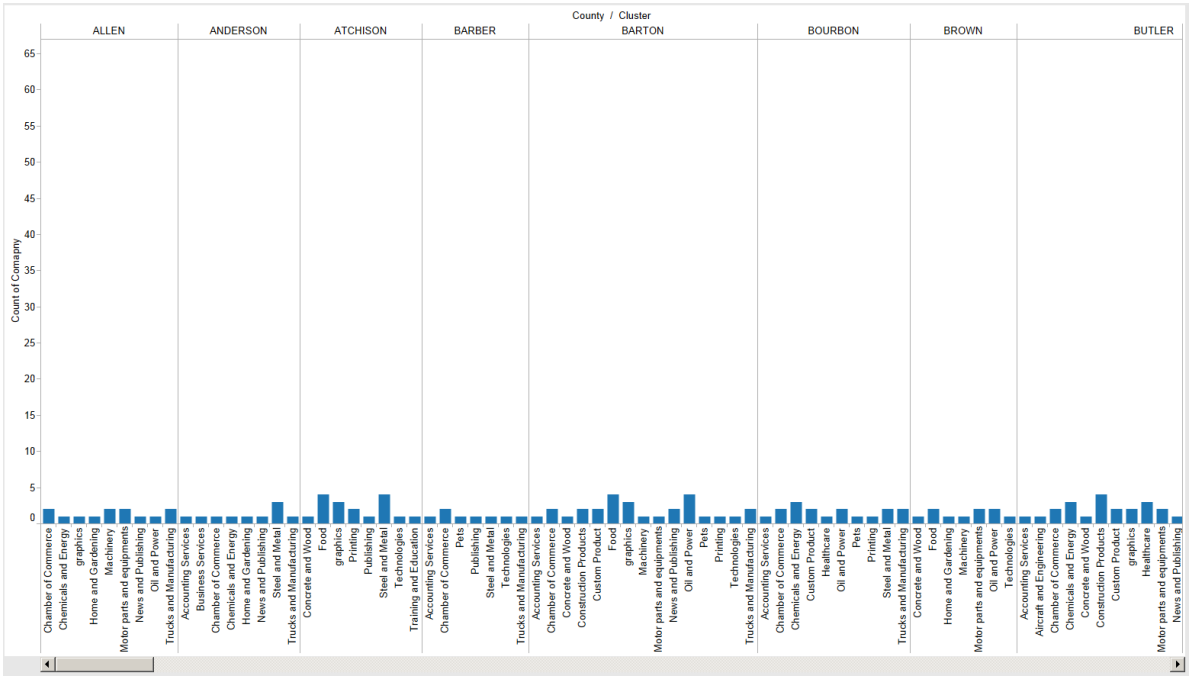


Figure 5.7: *Visualization and Statistics about the clustered businesses*

Chapter 6

Results

In this chapter, we discuss the results of the proposed two approaches. As we explained in Section 5.2, to evaluate the results, we decided to use the help of 3 human experts in the field of economic development to judge our results. To Respect the privacy of the judges, we refer to them in this Chapter as *judge 1*, *judge 2*, and *judge 3*. The chapter is organized as follows: In Section 6.1, we discuss the results that were obtained by using LSA-based approach. Section 6.2 describes the results of clustering by using the LDA-based approach. Finally, we make a comparison between the results of the two approaches in Section 6.3.

6.1 LSA Clustering

In our LSA-based approach, we were able to cluster all the 4291 businesses that we have data for into 30 unique clusters. We then gave our results to 3 experts in the field of economic development to evaluate our clusters using the evaluation program that we designed specifically for this task. *judge 1* did 100 test cases that they agreed on 69 of them. *judge 2* did 100 test cases that they agreed on 63 of them. *judge 3* did 100 test cases that they agreed on 64 of them. The average of the judges agreements with our results was 65.33%. Table 6.1 shows the results of evaluations that the judges did.

Judge	Number of Agreements	Number of Disagreements
1	69	31
2	63	37
3	64	36
Average	65.33	34.66

Table 6.1: The results of the LSA-based approach

6.2 LDA Clustering

In our LDA-based approach, we were able to cluster only 2049 businesses of the 4291 businesses that we have data for into 30 unique clusters. The reason why we were not able to cluster all the 4291 businesses is because of the way that the LDA-based approach works. More than half the businesses were discarded because their highest topic probability was bellow a certain threshold. We gave the results of our LDA-based approach to the same experts that we used their help to judge the results of the LSA-approach. *judge 1* did 100 test cases that they agreed on 71 of them. *judge 2* did 100 test cases that they agreed on 65 of them. *judge 3* did 100 test cases that they agreed on 73 of them. The average of the judges agreements with our results was 69.66%. Table 6.2 shows the results of evaluations that the judges did.

Judge	Number of Agreements	Number of Disagreements
1	71	29
2	65	35
3	73	27
Average	69.66	30.33

Table 6.2: The results of the LDA-based approach

6.3 Comparison

According to our evaluation procedure, we were able to achieve better results by using the LDA-based approach. The average difference between the LSA-based and LDA-based approaches was less than 5% as Table 6.3 shows. With the LDA-approach, we were able to obtain a list of the top words in every cluster, but we couldn't obtain a similar list with the LSA-based approach because LSA coverts words into concepts which makes it impossible to find out the top words in every cluster. However, there are some trade-offs involved when using the LDA-based approach. With the LDA-based approach, we were able to cluster only 2049 businesses out of the 4291 businesses that we have data for because of the way that the LDA-based approach works; more than half the businesses were discarded because their highest topic probability was bellow a certain threshold. On the other hand, with the LSA-based approach, we were able to cluster all the 4291 businesses because we are actually using a clustering algorithm on the top of LSA. Unlike the LDA-based approach, we were able to generate a hierarchical tree of the clusters with the LSA-based approach as Figure 6.1 shows. Table 6.4 shows a feature comparison of the two approaches.

Judge	Number of Agreements		Number of Disagreements	
	LDA	LSA	LDA	LSA
1	71	69	29	31
2	65	63	35	37
3	73	64	27	36
Average	69.66	65.33	30.33	34.66

Table 6.3: A comparison between the LSA-based and LDA-based approaches

Feature	LSA-based Approach	LDA-based Approach
Number of clustered businesses	4291	2049
Ability to produce a hierarchical tree	Yes	No
Top words in every cluster	No	Yes

Table 6.4: Feature comparison of the two approaches

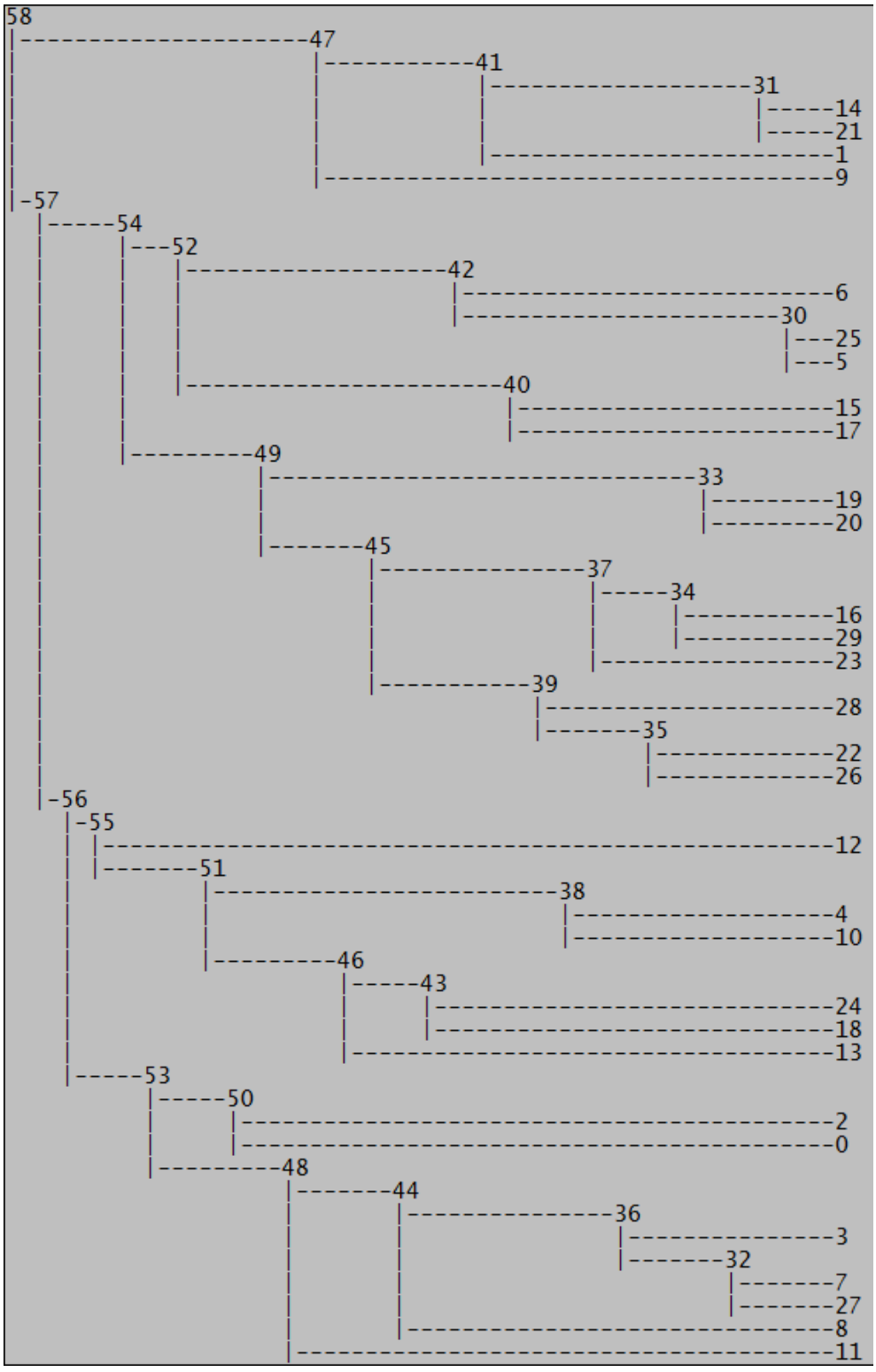


Figure 6.1: *The hierarchical tree produced for our clusters using the LSA-based approach*

Chapter 7

Conclusion and Future work

In this Chapter, we summarize our work and discuss our conclusion in Section 7.1. We then propose some improvements and new directions for this work in Section 7.2

7.1 Summarization and Conclusions

This work deals with the problem of clustering businesses into different clusters to identify and improve those businesses. One goal was to cluster the businesses in the state of Kansas. Another goal was to attempt to overcome the challenges that were faced by [Parimi, 2013] which are: data noise, low number of clustered businesses, and lake of evaluation approach.

We were able to successfully overcome all the challenges faced by [Parimi, 2013]. To overcome the data noise challenge, we used a newer version of the crawler used in [Parimi, 2013], and we were able to obtain a cleaner data set. To increase the number of clustered businesses, we decided to include the services information about businesses into our data set, and we introduced a new approach to cluster the businesses. To overcome the lake of evaluation approach, we introduced an evaluation procedure that is based on the judgment of human experts.

In the initial phase of the work, we cleaned and organized the businesses' data. In the second phase of the work, we took two different approaches to analyze and cluster the data. In the first approach, which we introduced in this work, we used Latent Semantic Analysis to analyze the data that we cleaned. On top of Latent Semantic Analysis, we used Bisection K-Means clustering algorithm to cluster the businesses into different clusters. In the second approach, we used Latent Dirichlet Allocation to analyze the cleaned data and cluster the businesses into different clusters which is the same approach followed by [Parimi, 2013]. Finally, we visualized the clusters using Google Earth and Tableau.

When we applied LSA to analyze our data, we followed the steps mentioned in [Dumais, 2004]. When we used Bisecting K-Means clustering, we used an implementation of the algorithm that was provided in *Cluto* [Karypis, 2002]. When we applied LDA, we used an implementation of the algorithm that was provided in *MALLET* [McCallum, 2002].

According to our evaluation procedure, we were able to obtain better results with the LDA-based approach. However, the improvement was not worth the trade-offs that we encountered with the LDA-based approach. In the LDA-based approach, we were not able to build a hierarchical tree of our clusters. We were also not able to cluster all the businesses by using the LDA-based approach. The LSA-based approach successfully generated a hierarchical tree of our clusters, and we were able to cluster all the businesses in our collection.

Visualizing the clusters that we created was done by using two different methods. Our first approach to visualize the clusters was done by using *Google Earth*. We had to covert our data set to a kml-formatted file, then we imported the file to *Google Earth*. In our second approach to visualize the businesses, we used *Tableau*. We organized our data into a CSV-formatted file, then we were able to visualize the clusters by importing the file into *Tableau*. Using *Tableau* was much easier than using *Google Earth*. Unlike *Google Earth*, with *Tableau*, we were able to display some statistics about the clusters, and filer the clusters by county. Our experience with *Tableau* was much better than *Google Earth*.

7.2 Future Work

As a next step into our future research, a third approach to cluster the businesses can be proposed. We believe that the LDA-based approach can be improved to provide the ability to cluster all the businesses that we have in our collection, and also generate a hierarchical tree of the clusters. We did not use any clustering algorithms on top of Latent Dirichlet Allocation, and we believe that further research can be done to achieve that. Figure 7.1 shows an overview of the suggested approach.

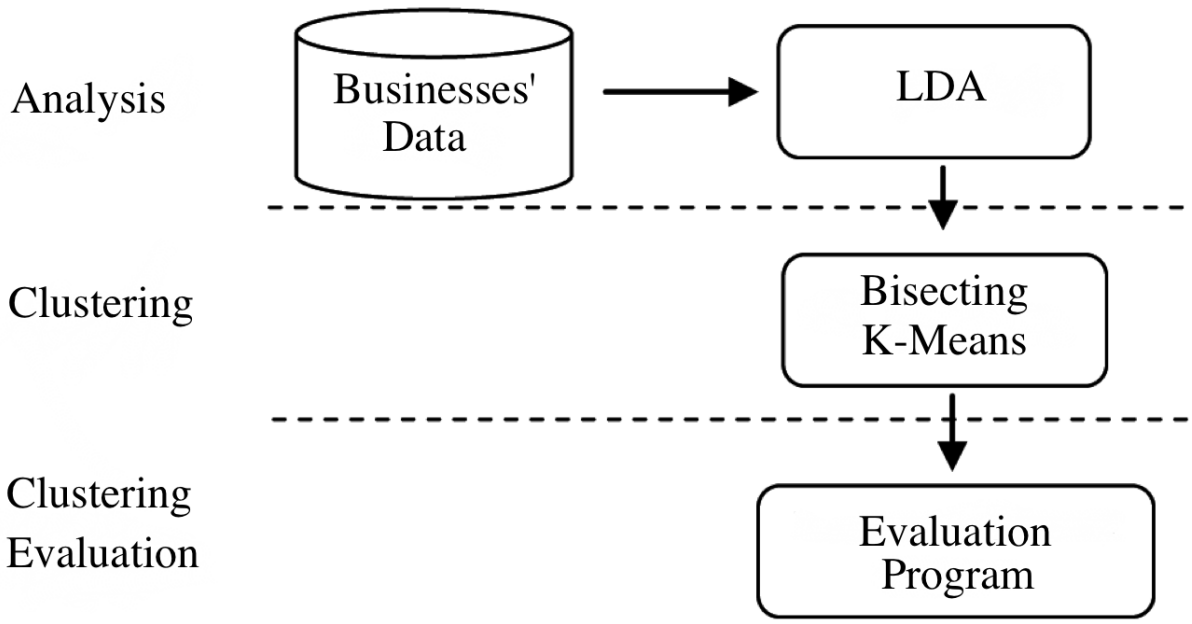


Figure 7.1: *An overview of the suggested approach*

This work can be extended to a partnership recommender system between businesses. We believe that such a system will further help businesses to identify potential partners or suppliers. Such a system can be built by using the data that we cleaned and organized, and also by using the clusters that we were able to generate in this thesis work.

Bibliography

Breck Baldwin and Bob Carpenter. Lingpipe. *Available from World Wide Web: http://alias-i. com/lingpipe*, 2003.

David M Blei. Probabilistic topic models. *Communications of the ACM*, 55(4):77–84, 2012.

David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

Roger B Bradford. An empirical study of required dimensionality for large-scale latent semantic indexing applications. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 153–162. ACM, 2008.

George Casella and Edward I George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

Susan T Dumais, George W Furnas, Thomas K Landauer, Scott Deerwester, and Richard Harshman. Using latent semantic analysis to improve access to textual informatio. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–285. ACM, 1988.

Pat Hanrahan. Vizql: a language for query, analysis and visualization. In *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 721–721. ACM, 2006.

Thomas Hofmann. Probabilistic latent semantic analysis. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 289–296. Morgan Kaufmann Publishers Inc., 1999.

Anil K Jain, M Narasimha Murty, and Patrick J Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

George Karypis. Cluto-a clustering toolkit. Technical report, DTIC Document, 2002.

Virginia Klema and Alan Laub. The singular value decomposition: Its computation and some applications. *Automatic Control, IEEE Transactions on*, 25(2):164–176, 1980.

C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. An Introduction to Information Retrieval. Cambridge University Press, 2008. ISBN 9781139472104. URL http://books.google.com/books?id=t1PoSh4uwVcC.

Andrew Kachites McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

Wunderlich D. Parimi, Caragea. Economic development through business profiling: A text analysis based approach. In *Proceedings of the 2013 IEEE/WIC/ACM International Conference on Web Intelligence (WI 2013)*, 2013.

Martin F Porter. An algorithm for suffix stripping. *Program: electronic library and information systems*, 14(3):130–137, 1980.

A. Rajaraman and J.D. Ullman. *Mining of Massive Datasets*. Mining of Massive Datasets. Cambridge University Press, 2012. ISBN 9781139505345. URL http://books.google.com/books?id=OefRhZyYOb0C.

M.A. Russell. *Mining the Social Web: Data Mining Facebook, Twitter, LinkedIn, Google+, GitHub, and More*. O'Reilly Media, 2013. ISBN 9781449368227. URL http://books.google.com/books?id=_VkrAQAAQBAJ.

Michael Steinbach, George Karypis, Vipin Kumar, et al. A comparison of document clustering techniques. In *KDD workshop on text mining*, volume 400, pages 525–526. Boston, 2000.