Text and Predictive Analytics;
Classification of On-line Customer Opinion Surveys

by

Ahmet Yucel

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
December 12, 2011

Keywords: Singular Value Decomposition, Feature Selection, Predictive Modeling

Approved by

David Mark Carpenter, Chair, Professor of Mathematics and Statistics
Asheber Abebe, Associate Professor of Mathematics and Statistics
Peng Zeng, Associate Professor of Mathematics and Statistics

Abstract

Thanks to computers and other data storage technologies, today we can easily save and rapid access to enormous amounts of data. According to results a report published by Andrew Harbison & Pearse Ryan, the information stored in electronic platform consist of more than 80% from written text (unstructured data). Text analytics can be very useful tool in converting these large amounts of unstructured data into regular structured data and then actionable information.

Predictive models are very useful tools in text mining. The results of the models may give important information that can be used in decision making. Feature construction from pre-existing features and feature selection techniques involve creating the predictive models.

Research has shown that by applying different feature selection techniques in creating predictive models can increase the model's accuracy rate.

In this thesis we review text and predictive analytical methods, and apply them to a cast satisfaction.

Acknowledgments

First and foremost, I would like to thank God. I would like to thank Professor David Mark Carpenter for all his support, encouragement, and patience. Thank you to my parents Hasan Huseyin and Zeynep Yucel, my sister Sevda Teker and her husband Erkan Teker for their love, support, and encouragement. They have been very supportive and encouraging throughout my academic tenure here at Auburn. Last but not least I would like to say thank you to the members of Committee: Dr. Peng Zeng and Dr. Asheber Abebe for their valuable feedbacks and suggestions that helped me to improve the thesis.

Table of Contents

Abstract ..…….....…….……..……..……..……..……..…….. ii

Acknowledgements ..…….……..……..……..……...…….. iii

List of Figures ……...…….……...……..……..…….. vii

List of Tables …..……...……..…..……..……..… ix

1. INTRODUCTION..... 1

2. LITERATURE REVIEW..... 3

2.1 TEXT MINING OVERVIEW..... 3

2.1.1 TEXT CATEGORIZATION / CLASSIFICATION..... 5

2.1.1.1 TOKENIZATION 6

2.1.1.2 STOP WORD REMOVAL 8

2.1.1.3 STEMMING.....9

2.1.1.4 FEATURE EXTRACTION.....10

2.1.1.4.1 FEATURE CONSTRUCTION..... 11

2.1.1.4.2 FEATURE SELECTION..... 11

2.1.1.4.2.1 Information Gaining..... 12

2.1.1.4.2.2 Chi-square (χ^2_{max})..... 12

2.1.1.5 WEIGHTING..... 13

2.1.1.5.1 BOOLEAN RETRIEVAL..... 13

2.1.1.5.2 TERM FREQUENCY WEIGHTING..... 14

2.1.1.5.3 TERM FREQUENCY-INVERSE DOCUMENT FREQUENCY WEIGHTIN.... 14

2.1.1.6 VECTOR SPACE MODEL..... 15
2.1.1.6.1 VECTOR CREATING..... 15
2.1.1.6.1.1 Bianary Frequency Weighting..... 16
2.1.1.6.1.2 Term Frequency Weighting..... 16
2.1.1.6.1.3 Term Frequency-Inverse Document Frequency Weighting Method..... 17
2.1.1.6.2 KEYWORD SELECTION AND WEIGHTING..... 17
2.1.2 ASSOCIATION ANALYSIS..... 19
2.1.3 CLUSTERING..... 21
2.1.4 INFORMATION EXTRACTION..... 22
2.2 TEXT MINING ALGORITHMS..... 23
2.2.1 NAÏVE BAYES ALGORITH 23
2.2.2 GENERAL LINEAR MODELS (GLM) ALGORITHM..... 24
2.2.3 SUPPORT VECTOR MACHINE (SVM) ALGORITHM..... 26
2.2.4 K-MEANS ALGORITHM..... 29
2.2.5 APRIORI ALGORITHM..... 30
2.2.6 DECISION TREES..... 32
3. A TEXT ANALYTICS APPLICATION..... 34
3.1 STUDY DESIGN AND DATA DESCRIPTION..... 34
3.2 MATERIALS/TOOLS..... 34
3.3 PROCEDURE..... 35
3.4 RESULTS 53
4. CONCLUSIONS……...55
4.1 FUTURE WORK……..……55

REFERENCES……..56

APPENDIX……..61

List of Figures

Figure 1 Text mining Process..... 4

Figure 2 Text categorization/classification process.....5

Figure 3 The task of text preparation and processing..... 6

Figure 4 Breaking up the text into tokens by appointed marks..... 8

Figure 5 Matrix representation of the document vectors..... 16

Figure 6 Market Basket Analysis..... 20

Figure 7 Data Clustering..... 21

Figure 8 Hierarchical Clustering..... 22

Figure 9 The hyperplane's maximization the geometric distance..... 28

Figure 10 The hyperplanes' separation the two classes..... 28

Figure 11 Maximum-margin hyperplane and margins for an SVM..... 29

Figure 12 Steps of the K-Means Algorithm..... 30

Figure 13 Apriori Algorithm example..... 31

Figure 14 Decision Tree Layout..... 32

Figure 15 Text mining filtration window Selected words..... 35

Figure 16 Selected words..... 36

Figure 17 Scree Plot.....37

Figure 18 Scatterplot of Component -2 vs Component-1..... 38

Figure 19 Colored Scatterplot of Component -2 vs Component-1with Labels 39

Figure 20 General look of the Data Set..... 40

Figure 21 Recoding Negative_Connotations variable..... 41

Figure 22 Interaction Plot: City vs Negative_Connotations..... 42

Figure 23 Decision Tree with SVD scores..... 43

Figure 24 Decision Tree-1..... 45

Figure 25 Decision Tree-2..... 48

Figure 26 First part of the second decision tree..... 50

Figure 27 Second part of the second decision tree..... 51

Figure 28 Third part of the second decision tree..... 52

List of Tables

Table 1 Data Frequency for each city 34

Table 2 Model Evaluation of the predictive model with SVD scores..... 43

Table 3 Selected Features-1..... 44

Table 4 Model Evaluation-1..... 46

Table 5 Selected Features-2..... 47

Table 6 Model Evaluation-2..... 52

1. INTRODUCTION

Advances in storage capabilities, huge data collections, and easy access to target data left people in an immense data pool. One of the most important ways to deal with this problem is Data Mining. Data mining is the analysis process of discovering knowledge in a database. However, according to research by Merrill Lynch and Gartner, 85-90% of the data all over the world are stored in unstructured form (McKnight, 2005), and thus, data mining algorithms are not enough by themselves. At this point Text Mining plays an important role.

Text mining is the process of exploring structured data and extracting useful information from a collection of unstructured data. Text mining methods can be used in very different areas including business documents, customer reviews, web pages, e-mails and other sources.

One of the most popular text mining techniques is predictive modeling. Decision trees, neural networks and boosted trees are different types of predictive models. Predictive models are used to determine which class a set of data belongs to. For example, a technology company can apply predictive modeling algorithms to specifically target customers, and so before generating a new model.

Building an accurate predictive model is very important to get reliable results. Especially in business, inaccurate results may be very expensive. For example, if the technology company mentioned above uses inaccurate results and produce its new model according to that results, this situation may cause the company lose its market, customer confidence and money. Generally researchers think that an ensemble model approach such as decision trees will produce predictive models with higher accuracy rates. Therefore, in our experiment we are going to use a decision tree. In our experiment we investigate the premise using different feature selection methods and then build decision trees for the selected features.

The rest of this thesis is organized as follows: Chapter 2 contains the literature review and a general overview about text mining processes and descriptions about various text mining algorithms and techniques. Chapter 3 discusses the experiment design data collection, materials/tools and results. Chapter 4 presents ideas for future work.

2. LITERATURE REVIEW

2.1 TEXT MINING OVERVIEW

Thanks to computers and internet technology, today it is very easy to reach a tremendous pile of information and knowledge. Every day millions of pages of news-text are flowing into the web sites of news agencies, millions of web sites' contents are being updated and approximately 90 trillion (2010 est.) e-mails are being sent between e-mail account owners more than 2 billion (Radicati & Hoang, 2011) per year all over the world. For example, if we think that even Microsoft webmail has 256.2 million users (Graham, 2008), how we are in a pile of information can be easily seen.

In short, in the electronic platform, unstructured and continuously increasing huge piles of information such as the reader/viewer comments, academic articles/journals, e-books, blogs, e-mails, chat conversations, research documents, etc are waiting to be analyzed. According to the results of a report published by Andrew Harbison & Pearse Ryan (Harbison & Ryan, 2009), the information which can be processed digitally consists of almost 80-85% from written text (unstructured data). By using the methods of data mining, choosing this regular data through piles of unstructured dispersed data is becoming very important. Text and data mining are similar at the point that both try to obtain information from massive and unstructured sources. However, text mining is based on text sources (Chang, Healey, McHugh, Wang, Jason, 2001), (Kroeze & Bothma, 2007). Rregular structured data are extracted from unstructured data (text) and thus hidden information is discovered. This process is done with a variety of text mining techniques. (Kroeze & Bothma, 2007) However, this is a very difficult and time-consuming 'process.

Natural language processing (NLP) is a sub-discipline of computer science and linguistics. In NLP, natural language texts and/or sounds carried out on the studies in computer processing.

Therefore, modern statistical NLP algorithms require using of linguistics, computer science, and statistics (Charniak, 1984). All programming languages used around the world have specific structures, rules, and a standard filed. Natural languages cannot be explained so easily. All around the globe, there are hundreds of different official/known languages and each language has more than 100,000 words. In addition, the fact that language courses are always changing and expanding with a lot of uncertainty, and each language has its own unique grammar structure. For this reason, it is impossible a text mining software to interpret a language 100% correctly (Erol, 2009).

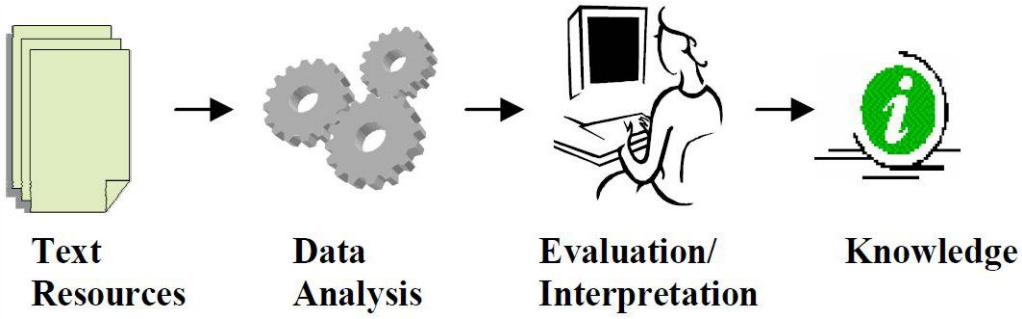


Figure-1: *Text mining Process*

Text mining Process (see Figure-1) (Stavrianou, Andritsos, & Nicoloyannis, 2007) is divided into four main categories: text classification or text categorization (TC), association analysis, clustering and information extraction (IE). The classification or TC process is to include categories or classes of objects previously known. Association analysis is used to identify the words which are often associated with each other or developing and to make the sets of documents or the contents of documents more understandable. IE techniques are used to find the useful data in the documents or statements. Cluster analysis is used to discover the underlying structure of the document sets.

2.1.1 TEXT CATEGORIZATION / CLASSIFICATION

Text categorization/classification (*TC*) is the grouping of a text into two or more classes (Mahinovs & Tiwari, 2007). The goal of TC is to classify documents (academic articles, emails, etc) into categories. For example, news articles into "local" and "global", e-mails into "spam" and "others", and customer feedbacks into "positive" and "negative" can be classified. Today, the internet is one of the biggest and most important information sources. Undoubtedly, its importance comes from being available very easily and fast. However, with the extension of internet in an uncontrollable way, reaching to aimed data becomes like hunting a special-species fish in this information ocean. However, categorization is a significant method that reduces the time to reach the information. This is one of the most important motivations for the TC. An example use for TC is to deal with span emails. However, because of a big partition of the text-data on the internet is written in natural language, the categorization of texts is very difficult in this format. So, for overcoming this problem, these texts written in natural language should be transformed into digital texts (Asyali & Yildirim, 2004). Figure-2 shows the text categorization process (Mahinovs & Tiwari, 2007).

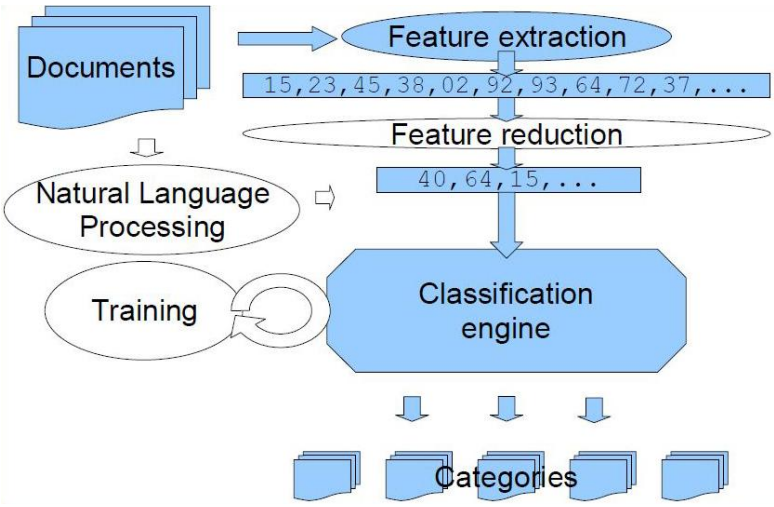


Figure-2: *Text categorization/classification process.*

In addition, of course should be a pre-process that prepares the text to be categorized. For example, specific tags like xml/html are identified as blocks of text for section searching (Oracle®, 2003), non-letter characters are replaced by spaces, single-letter words should be deleted, and all characters are converted into lower cases (Tonta, Bitirim & Sever, 2002). There are several steps before text categorization. These steps are tokenization, stop word removal, stemming, feature extraction, and vector space model (Mahinovs & Tiwari, 2007).

2.1.1.1 TOKENIZATION

Token is a simple sample of a type. In text mining, tokenization is the breaking a stream of text up into tokens, and is the very first step for preparing natural language text for before categorization. Figure-3 shows where extended tokenization is located within the text preparation and processing framework (Hassler & Fliedl, 2006).

Figure-3: *The task of text preparation and processing.*

Firstly, the texts are broken up into meaningful components. For example, texts can be broken up into chapters, sections, sentences, words, phrases, symbols, or other meaningful elements called tokens (Feldman & Sanger, 2007).

Generally words are surrounded by whitespace and may be followed by punctuations, parentheses, or quotes. So, a simple tokenization rule can be stated with the following order: Break up the character sequence at whitespace positions and cut off punctuations, parentheses, and quotes at both ends of the fragments to get the sequence of tokens (Sherpa & Choejey,

2008). However, while we follow this order we may encounter with some problems and need some more extra study.

First of all, all periods are not punctuation. They may be markers for abbreviations such as "U.K." "Mr.", "Dr.", and so on. Only periods which are sentence markers should be removed to get separate tokens (Feldman & Sanger, 2007). Also, the sentence markers "!" or "?" are generally obvious punctuations. The most difficult symbols to distinguish are semicolons ":" and ";".Distinguishing the different uses of colons and semicolons is very hard without analyzing the whole sentence. The other problem is with ordinal numbers. Ordinal numbers are written with a trailing period after the number in Turkish or other Europe Language. For example, "13rd" is written "13.". These ordinal numbers cause the same problem as abbreviations: A number which is followed by a period may either be an ordinal number, a cardinal number in sentence-final position, or an ordinal number in sentence-final position. Distinguishing is not possible without contextual information. In the above expression we said that tokens do not contain whitespace. However, there are some multi expressions that are complex prepositions like "because of", conjunctions like "so that", and adverbs like "at all", date expressions like "Jan. 16, 1986", time expressions like "2:30 am", and proper names like "AC Milan" and it is better to accept them as single tokens. The opposite job is done for the acronym words like "we'll" or "aren't". These words are separated into two words like " we will" or "are not". The last problem that might be encountered in tokenization is missing-whitespaces. Sometimes there is no blank after a punctuation mark like "hours.The" or "however,when", that should be broken up into three tokens (Hassler & Fliedl, 2006), (Ben-Hur & Weston). Here a step by step example of tokenization and typing of tokens (Hassler & Fliedl, 2006).

1- identify single-tokens
2- type single-tokens

3- split sentence end markers

4- reinterpret single-token types

5- merge and split tokens recursively

6- reinterpret all token types

• **Alphabetics T_a:** no letters capitalized (T_{a_1}), first letter capitalized (T_{a_2}), all letters capitalized (T_{a_3}), mixed cases (T_{a_4}) etc.

• **Numerics T_n:** plain numbers (T_{n_1}), numbers containing periods or colons (T_{n_2}) etc.

• **Punctuation marks T_p::** sentence ending markers (T_{p_1}), pairwise marks lick brackets and quotes (T_{p_2}), single sentence-internalmarks like commas (T_{p_3}) etc.

• **Mixtures T_m:** ending with sentence end marker (T_{m_1}), ending with hyphen (T_{m_2}), starting with hyphen (T_{m_3}), containing slashes/hyphens (T_{m_4}), containing numbers (T_{m_5}) etc."

Figure-4 shows how to break up the text into tokens by marks appointed earlier.

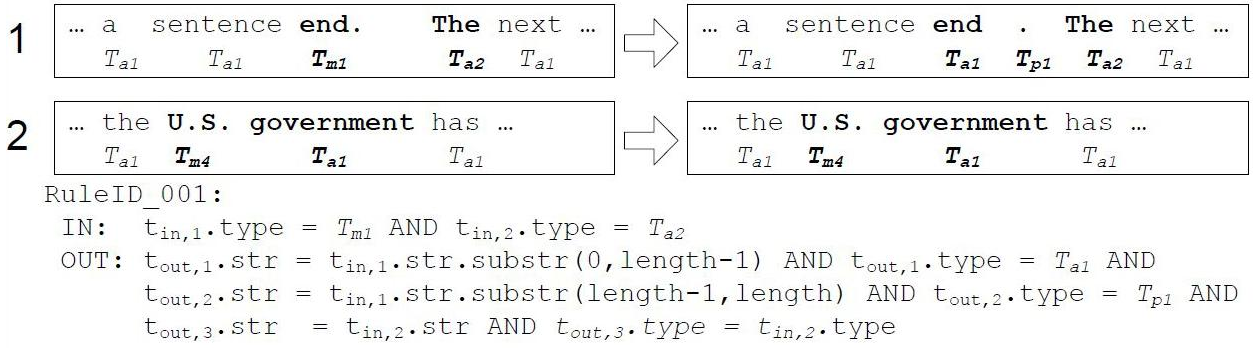


Figure-4: *Breaking up the text into tokens by appointed marks.*

2.1.1.2 STOP WORD REMOVAL

Stop-words are common words that do not have so much meaning in a retrieval system. Stop-words are a part of natural language with that a text miner will encounter. The reason that stop-words should be removed from a text is that they make the text look heavier and less important for analysts and the stop-words are not necessary for the analysis and so we do get

some data reduction by eliminating stop-words. A query done by using stop-words would have a weak ability to categorize the text because of these words return each element of the data set as a result (Adsiz, 2006). In enterprise search, all stop words, for example, common words like *a* and *the*, are removed from multiple word queries to increase search performance. There is not one master too many list of stop words which all tools use. Any group of words can be chosen as the stop words for a given purpose, depending on their importance and data reduction needs. For some search machines, these are some of the most common, short function words, such as *the, is, at, which and on*. In this case, stop words can cause problems when searching for phrases that include them, particularly in names such as '*The Who*', '*The The*', or '*Take That*'. Other search engines remove some of the most common words including *lexical words* [1], such as "*wan*t"—from query in order to improve performance (Stackoverflow, 2008).

2.1.1.3 STEMMING

Stemming is the process of reducing inflected or derived words to their stem, base or root forms. Certainly, the number of words in a text gives important information about the text. Therefore, it is quite likely that if a word is frequently repeated in a text then it should be related to the subject of the text. The root or stem with an adjunct and the simple form of a word will be counted as different words and this situation makes the word less important than it should be. However, when we consider the meaning, different forms of the word may be treated as the same. For example; stemmer, stemming, stemmed ⇢ stem (Julie, 1968). These four words will be treated as the same. The stem is the morphological root of the word is not necessarily a valid

[1] Lexical words: A Lexical item (or lexical unit, lexical entry) is a single word or chain of words that forms the basic elements of a language's lexicon (vocabulary). Examples are "cat", "traffic light", "take care of", "by-the-way", and "it's raining cats and dogs".

(A complete list of English stop-words including 429 words can be found under http://www.lextek.com/manuals/onix/stopwords1.html.)

word. Porter's Algorithm is popular stemming method (Mustafa, Akbar & Sultan, 2009).

2.1.1.4 FEATURE EXTRACTION

In the all size reduction algorithms, all words in the documents are collected into a word-list. Then, according to the results of reduction algorithms, some words in the word-list are removed and finally, only the words in the word-list are used. Feature extraction is a data reduction process. The feature extraction process results in a much smaller and richer clustering sets of words of attributes. Here, word-clusters are created by feature similarity of the words. After grouping words into clusters, these clusters are treated as features instead of individual words (Feldman & Sanger, 2007).

Feature extraction maps from a larger amount of resources into a simplifier data (Dragos & Manolescu, 1998). When dealing with a large and complex data, much text may not be meaningful for the aim of analysis. To overcome this problem we need to use classification algorithms. The goal of feature extraction is to improve the effectiveness of classification and analysis (Rustum, 2007).

In the feature extraction frame, by the specific features we can restructure the data, we can extract the important information which is crucial for the aim of analysis to improve the quality of the classifier. In addition, because of term frequency is an important information resource to distinguish the documents that have different contents, we can create feature vectors through word algorithms and statistical approaches on term frequency (Liangtu & Xiaoming, 2007).

We can operate feature extraction in two main steps; feature construction and feature selection.

2.1.1.4.1 FEATURE CONSTRUCTION

Feature construction is the process of creating a new feature that is earlier unknown (Li, 2007), or designing a modified feature from the pre-existing features associated with database.

2.1.1.4.2 FEATURE SELECTION

Feature selection is applied to eliminate insignificant and irrelevant features from texts and create subsets consisting of significant features. And so, the existing features are simplified, the dimension of the texts are reduced, in other words, the existing features are transformed into a lower dimensional space and thus the comprehensibility can be significantly improved (Feldman & Sanger, 2007). In short, feature selection is aimed at creating a more suitable and clearer data to be analyzed easily and to see important (Li, 2007) but hidden points (Kim, Street & Menczer, 2003). There are many filters to evaluate and eliminate features. In order to perform the filtering, the relevance of features such as document frequency should be calculated as a measure. Typically interclass distance, statistical dependence or information-theoretic measures can be given as examples for filters. Some of them are very strength, may remove almost 90-99% of the all features (Feldman & Sanger, 2007). Document frequency of a word is the number of appearing of the word in that document. A researcher identifies a certain point of frequency such that the words that have frequencies under that point are removed from the document. This method gets its base from the idea that words that appear less than a certain threshold do not have a decisive role or ability to identify categories (Bolat, 2003). By the way there are other useful measures of feature relevance that take into account the relations between features and categories. For example *information gaining (IG)* and *chi-square* (χ^2_{max}) (Feldman & Sanger, 2007).

2.1.1.4.2.1 Information Gaining (*IG*)

This method is about the effect of each word on the categorization. Let $c_1, c_2, \dots, c_k$ be possible categories. The IG of a word *w* is computed with the following formula

$$IG(w) = -\sum_{j=1}^{k} P(c_j)\, logP(c_j) + P(w) \sum_{j=1}^{k} P(c_j|w)\, logP(c_j|w) + P(\overline{w}) \sum_{j=1}^{k} P(c_j|\overline{w})\, logP(c_j|\overline{w}), \quad (2.1)$$

where

$P(c_j)$ is the possibility of that a document belongs to category c_j among the all categories,

$P(w)$ is the possibility of that *w* appears in a random document among the all documents,

$P(c_j|w)$ is the possibility of that *w* appears in a random document belonging to category c_j,

$P(c_j|\overline{w})$ is the possibility of that *w* does not appear in a random document belonging to category c_j.

The IG is computed for each word and the words which have IG less than a certain threshold identified by analyst are removed from the collection (Adsiz, 2006).

2.1.1.4.2.2 Chi-square ($\chi^2_{\max}$)

$\chi^2_{\max}$ measures the dependency between word *w* and category c_j.

$$\chi^2_{\max}(w, c_j) = \frac{N * (AD - CB)^2}{(A + C) * (B + D) * (A + B) * (C + D)}, \quad (2.2)$$

where

A is the number of documents in that *w* appears and belong to category c_j.

B is the number of documents in that *w* appears but do not belong to category c_j.

C is the number of documents in that *w* does not appears but belong to category c_j.

D is the number of documents in that *w* does not appears and do not belong to category c_j.

N is the total number of documents in the collection.

And the measurement method is

$$\chi^2_{\max}(w) = max_j \ \chi^2_{\max}(w, c_j)\,. \tag{2.3}$$

The $\chi^2_{\max}$ is computed for each word and the words which have $\chi^2_{\max}$ less than a certain threshold are removed from the collection (Bolat, 2003).

2.1.1.5 WEIGHTING

Weighting is a process consists of choosing terms which are important (contribute more than others) for a document and giving these terms more importance (weight) in the analysis (Wikipedia, 2011). There are several methods to apply Weighting process; *Boolean Retrieval, Term Frequency Weighting, and Term Frequency–Inverse Document Frequency* (*Tf-Idf*) *Weighting* Methods (Adsiz, 2006).

2.1.1.5.1 Boolean Retrieval

Boolean retrieval is one of the simplest and oldest retrieval methods using exact-match model. Boolean method comes from Boolean algebra where keywords are combined with connecting search operators AND, OR, and NOT. Boolean logic is a popular and simply method generally used on internet. This method is established on grouping into two classes like "0 or 1", "non-exist or exist", "white or black", "yes or no" etc (George, 2003). This method is applied to text as following

If a word occurs in text then weight of the word is 1, otherwise it is 0.

$$a_{ik} = \begin{cases} 1 & , f_{ik} > 0 \\ 0 & , otherwise \end{cases}, \tag{2.4}$$

where a_{ij} is the i[th] term of the k[th] text, and f_{ik} is the number of occurrence of the term in the text (Adsiz, 2006).

2.1.1.5.2 Term Frequency Weighting

In this method, the number of occurrences of a word in a document is considered as *weight* for that word. That is, term frequency weight is equal to the number of occurrence of the word in document (Adsiz, 2006).

$$a_{ik} = f_{ik} \ , \tag{2.5}$$

here, if we take the number of occurrence of each word as the term of the document, instead of the words, then we will have quantitative terms. This is called in the literature as the *bag of words model* (Manning , Raghavan & Schütze, 2008).

2.1.1.5.3 Term Frequency–Inverse Document Frequency (Tf-Idf) Weighting

Term Frequency–Inverse Document Frequency (Tf-Idf) Weighting is a popular weighting method. The main application area is Information Retrieval (IR). In this method documents are identified in Vector Space Model. A Tf-Idf Weighting function works as following. (Soucy & Mineau) Firstly, as we mentioned above, the *tf–idf* weights are often used to evaluate how a word is significant to a text. The *tf–idf* combines the word frequency in the document. We know that *tf* measures how frequent a word in a document, while *idf* measures infrequency. A word is considered as insignificant if it has low occurrence in a text, however, if this word has a low frequency in the whole document, then it can be very important to describe the category of the text. This is the general idea behind this *tf–idf weighting* procedure (Adsiz, 2006). The *tf–idf* combines these two kinds of frequencies as a weighting measure. We define the *inverse document frequency* of a term as follows (Manning, Raghavan & Schütze, 2008):

$$idf_i = \log \frac{N}{df_i} \ , \tag{2.6}$$

where N is the total number of documents and df_i is the number of documents in that the word occurs.

Therefore, the *tf-idf* weighting value for the i[th] term of the k[th] text given by the following formula $(X.Z)$ (Manning , Raghavan & Schütze, 2008).

$$tf - idf_{ik} = f_{ik} * \left(\log \frac{N}{df_i}\right) \quad (2.7)$$

2.1.1.6 VECTOR SPACE MODEL

The Vector Space Model (VSM) firstly introduced by *Gerard Salton*. This mathematical approach is one of the most popular information retrieval models in text mining (Salton & Singhal, 1995). Text is messy and difficult to manage. VSM provides a mathematical representation of documents. In this model, every collection of items are represented as set, or vectors of terms extracted from the text itself (Stavrianou, Andritsos & Nicoloyannis, 2007). There are some specific methods to create vectors of terms.

2.1.1.6.1 Vector Creating

$Document\ D = a\ set\ of\ weighted\ keywords$,

$Query\ Q = a\ set\ of\ non - weighted\ keywords$,

$Vector\ Space = all\ the\ keywords\ encoutered\ " < t_1, t_2, \dots, t_k > "$,

$Document\ " < a_1, a_2, \dots, a_k > "$, $a_i = weight\ of\ t_i\ in\ D$,

$Query\ Q = <b_1, b_2, \dots, b_k>$, $b_i = weight\ of\ t_i\ in\ Q$.

Matrix representation is as the following

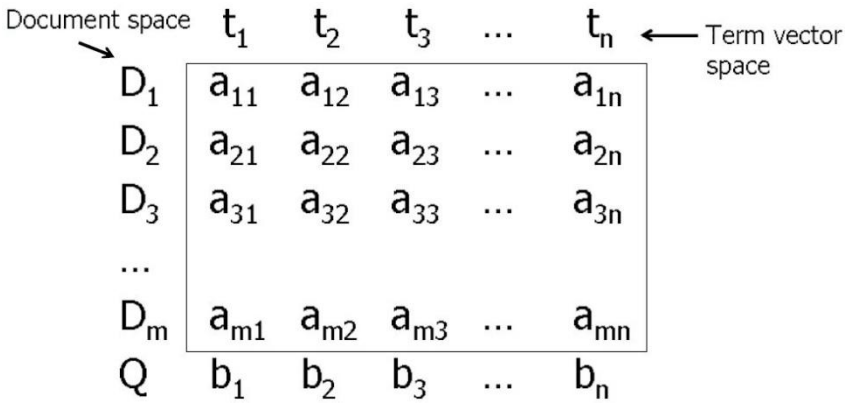


Figure-5: *Matrix representation of the document vectors.*

There are three different methods to express a text as a vector; by Binary Frequency, by Frequency and by Term Frequency–Inverse Document Frequency (Tf-Idf) Weighting Method (Pilavcılar, 2007).

2.1.1.6.1.1 Binary Frequency Weighting

This method creates an indicator vector indicating whether words in key-words-dictionary of a text are in the text or not. For example, let {inflation, the flu, the referee, medicine, fans, agricultural} be key-words-dictionary of a text and. Then we would identify the possible vectors as follows (Pilavcılar, 2007).

D1 = (0,1,0,1,0,0)
D2 = (0,0,0,1,0,0)
D3 = (1,0,0,0,0,0)
D4 = (0,0,0,0,0,1)
D5 = (0,0,1,0,0,0)
D6 = (0,0,0,1,1,0)

2.1.1.6.1.2 Term Frequency Weighting

This method is based on how many times a word is used in a text. For the same text we would identify the possible vectors as follows (Pilavcılar, 2007).

D1 = (0,2,0,1,0,0)
D2 = (0,0,0,1,0,0)
D3 = (1,0,0,0,0,0)
D4 = (0,0,0,0,0,2)
D5 = (0,0,2,0,0,0)
D6 = (0,0,0,1,2,0)

2.1.1.6.1.3 Term Frequency–Inverse Document Frequency Weighting Method

The term weight is associated with importance in a text. If a word occurs frequently in a text, then it can be said that the word is significant for the text. Since *term frequency* (*tf*) is an importance indicator, we can use this measure as the term weight (Salton & Singhal, 1995). In a similar way, the *Inverse Document Frequency* (*idf*) can be used as a weight. Sometimes marginal words in a text may be more important than a word having high frequency. For example, the term *fish* might not be very useful when we deal with a collection of articles on *Economic Sources of a Mediterranean Country,* however it might be very important when deal with a collection of articles about the fossils found in an Egypt desert. Therefore, even if a word does not occur so much, it may be considered as a decisive word (Ilhan, Duru, Karagöz & Sagir, 2008). In information retrieval and text mining the *tf–idf* weights are often used to evaluate how a word is significant to a text (Karen, 1972).

2.1.1.6.2 Keyword Selection and Weighting

The words that distinguish the characteristics of documents and used rarely in the documents are classified as a *keyword*. There are different methods to identify goodness of a keyword (Ilhan, Duru, Karagöz & Sagir, 2008). Briefly, for a c_j cluster *w* is a good keyword if it is an important word comparing to other words both in the cluster and in the whole collection (Lagus & Kaski, 1999).

If a word only belongs to a single document among the others, and gives significant information about the document, then this word can be considered as a powerful keyword. The

query processing using these keywords would be the most powerful way to the documents that these keywords belong to. Keyword selection process as follows (Ilhan, Duru, Karagöz & Sagir, 2008);

1. Pre-processing stage starts with removing certain words that cannot be keywords, from the text.
2. The process continues with the finding stems.
3. And then, the words that have the same stems are considered as the same keywords.

The number of occurrence of words in text is very important in the calculation of the weights of words. Weighting method is used to examine more sensitively the proximity of the words that are formally compared. According to weighting method procedure, every word in the texts does not have the same importance (Ilhan, Duru, Karagöz & Sagir, 2008).

$$w_i = tf_i * idf_i \tag{2.8}$$

$$idf_i = \log\left({}^{D}/_{df_i}\right) \tag{2.9}$$

The formulas (X.4) and (X.5) are used in the calculation of weight of keywords.

Here D is the total number of text and df_i is the number of documents tin that the word occurs. If this value is large then it is an indicator of that the word does not occur so much in the text, however it is more important in the calculation. So, if this value is a small then it is considered as that the number of appearance in the text is so much. If this value is zero then it is considered that the word appears in the all texts, and thus does not have a significant effect on distinguishing the texts. tf_i is the number of appearance of a word in a text. It is used in calculating the distance between the text vectors. idf_i is the discrimination index between the texts. When idf_i is multiplied by the number of times the keyword appears in the text, the total weight of the word w_i is found (Ilhan, Duru, Karagöz & Sagir, 2008).

2.1.2 ASSOCIATION ANALYSIS

Association Analysis is a classification method identifying interesting relationships hidden between large data sets. These relationships can be expressed in form of *association rules*.

Association Rule

An association rule is a method trying to find interesting rules and relationships that happens often among the item-sets in a dataset (Williams, 2010). An association rule is expressed as X → Y, where X and Y are disjoint item-sets. For example, it is quite likely that if a person is buying coffee, will also buy coffee-cream. So, thanks to this method, interesting patterns can be found in a database. The strength of an association rule can be measured in terms of its *support* and *confidence*. *Support* gives idea about if a rule can be applied to a dataset, and if it is, how often. *Confidence* measures the significance of an inference made by a rule (Ding & Sundarraj, 2006).

$$Support\ (\mathrm{X} \rightarrow \mathrm{Y}) = \frac{(\#\ of\ transactions\ in\ that\ X\ and\ Y\ appear\ together)}{total\ \#\ of\ transactions} \qquad (2.10)$$

$$Confidence\ (\mathrm{X} \rightarrow \mathrm{Y}) = \frac{(\#\ of\ transactions\ in\ that\ X\ and\ Y\ appear\ together)}{\#\ of\ transactions\ in\ that\ only\ X\ appears} \qquad (2.11)$$

The rules exceeding minimum confidence and minimum support specific threshold values chosen by a researcher, are identified as interesting rules. *Lift* is an extra but important measurement that shows the importance degree of the relationship or rule between X and Y.

$$Lift\ (\mathrm{X} \rightarrow \mathrm{Y}) = Support\ (\mathrm{X} \rightarrow \mathrm{Y}) / Confidence\ (\mathrm{X} \rightarrow \mathrm{Y}) \qquad (2.12)$$

Market Basket Analysis

A huge amount of data is collected on movements of clients shopping in supermarkets and retail sector. The most typical example of association rules is "*market basket analysis*" which is a modeling technique based on the idea that if one buy a certain group of items, then he/she is more likely to buy (or not to buy) another group of items. The discovery of this type of associations may provide important opportunity for market managers to develop more effective marketing strategies. For example, X% of customers buying sugar also buy eggs (see Figure-6) (Han & Kamber, 2001). This information can be found with association rule method.

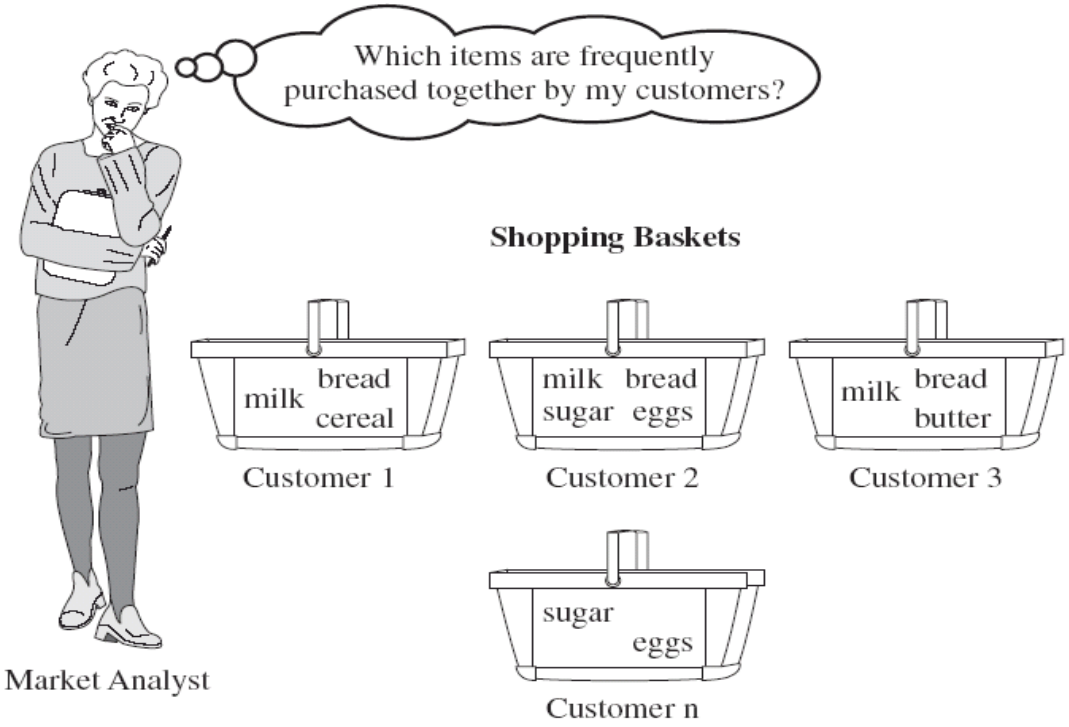


Figure-6: *Market Basket Analysis*

The most popular method used to find association rules in Association Analysis is the Apriori Algorithm. We will introduce the Apriori Algorithm later on.

2.1.3 CLUSTERING

Clustering is an important unsupervised learning method. The main idea is to cluster data points (or feature vectors, observations) into groups (Jain, Murty & Flynn, 1999) and so get a classified structure in a collection of unlabelled data, based on similarity criterion. In contrast to classification, in clustering, data points are appointed into groups whose members have similar properties in some way (Moore, 2001). (See Figure-7) (Jain, Murty & Flynn, 1999).

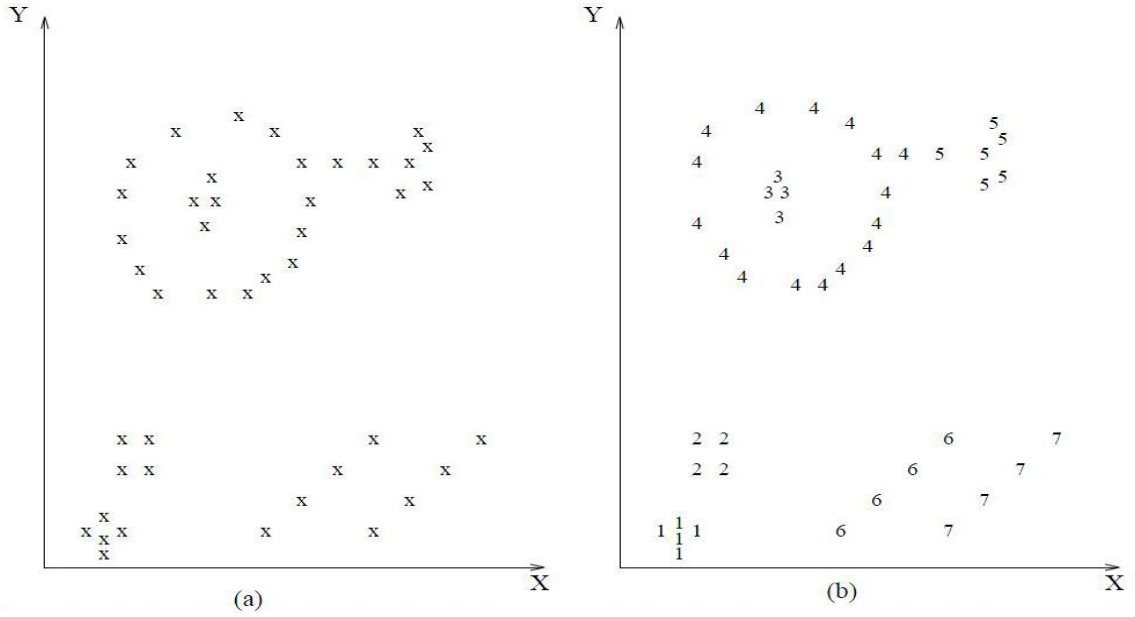


Figure-7: *Data Clustering*

The main similarity criterion is distance. The data points which are closer to each other by comparing to the other points are considered in the same cluster. This is called *distance-based clustering*. The distance between points is given by the Euclidean distance (Wittman, 2011):

$$d(x, y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}\,, \tag{2.13}$$

where x and y are any data points and x_i and y_i are the corresponding coordinates for x and y.

The Euclidean distance is a useful method, but it is not always the best distance measure. There are other distance procedures such as *City Block Distance* (2.14) and *Mahalonobis Distance* (2.15)*:*

$$d(x,y) = \sum_{i=1}^{n} |x_i - y_i|^2 \qquad (2.14)$$

$$d(x,y) = (x-y) \sum^{-1} (x-y)^T \ , \qquad (2.15)$$

where $(x-y)^T$ is the transpose matrix of $(x-y)$ and $\sum^{-1}$ is the inverse covariance matrix between x and y matrixes. Also, there is another clustering procedure called *Hierarchical Clustering*. Hierarchical clustering is a set of nested sets. The clustering works by merging 2 clusters at a time, (see Figure-8) (Wittman, 2011).

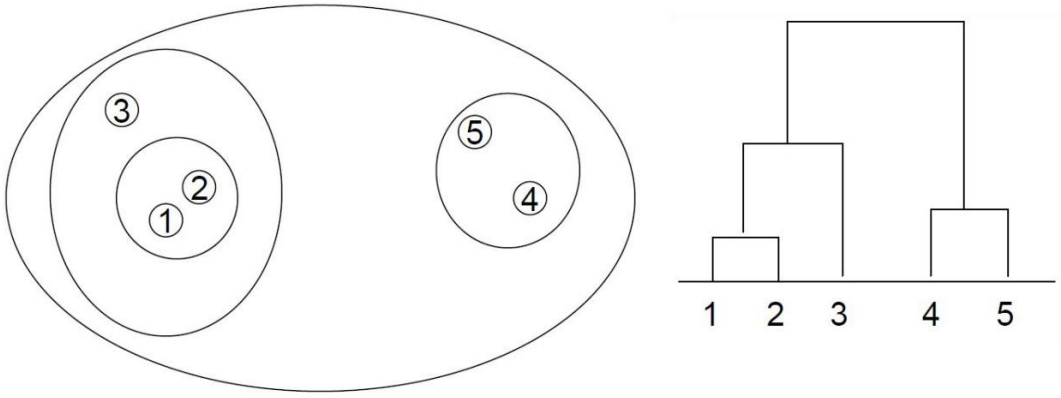


Figure-8: *Hierarchical Clustering*.

Also, we have another important clustering algorithm called K-Means Clustering. We are going to talk about this method under the "Text Mining Algorithms" title later on.

2.1.4 INFORMATION EXTRACTION

Information extraction is a process of extracting previously unknown but explicitly stated or implied useful information from text data and combining under specific feature titles (Cowie, Wilks, Dale, Moisl, Somers, 2000). In this final phase, in general, withdrawal of needed information from the text sets on that analysis and classification processes have been applied,

and saving them into the database operations take place. Knowledge acquisition is being developed in parallel with database systems for many years. In contrast to database management systems that focus on the structural data, IE is related to organization, text-based documents to extract the information. A typical IE problem depends on user input associated documents such as the key words or sample documents, etc. A typical IE system is an online document management system. Information Extraction IE mainly deals with identifying words or feature terms from within a textual file. Feature terms can be defined as those which are directly related to the domain (Mustafa, Akbar & Sultan, 2009).

2.2 TEXT MINING ALGORITHMS

In text mining there are two important part of association rules. First, large item-sets are produced then, in the second phase, rules are created. The dimensions of large item-sets are reduced by using different algorithm methods. In this chapter we are going to introduce several useful text mining algorithms.

2.2.1 NAIVE BAYES ALGORITHM

Naive Bayes is a simple classification algorithm. Naive Bayes algorithm is based on the calculation effects of each criterion to results as probability. All of the potential datasets and newly added documents are used to calculate the possibility of each newly added term to affect the categories (Adsiz, 2006). By Bayes Theorem (Han & Kamber, 2001),

$$P(c_j|d) = \frac{P(c_j)P(d|c_j)}{P(d)}, \tag{2.16}$$

where $P(c_j)$ is the prior probability of category c_j and $d = (t_1, \dots, t_M)$ is set of documents that is going to be classified.

Because of there are many features in datasets, it will be so difficult to calculate $P(d|c_j)$. Therefore, in this algorithm, features in a document are considered as independent (Adsiz, 2006).

$$P(d|c_j) = \prod_{i=1}^{M} P(t_i|c_j), \quad (2.17)$$

where t_i is the i[th] term in the document.

$P(c_j|d)$ is the probability of a document being in category c_j

$$P(c_j|d) = P(c_j) \prod_{i=1}^{M} P(t_i|c_j), \quad (2.18)$$

where $P(t_i|c_j)$ is the conditional probability of term t_i occurring in category c_j. Also, it is an indicator of how much t_i contributes to the category. In this method, we try to find the most appropriate category for the document. For this purpose, the best approach is the *maximum a posteriori* (*map*) category $c_{j_{map}}$ (Manning, Raghavan & Schütze, 2008).

$$c_{j_{map}} = argmax_{c_j} P(c_j|d) = argmax_{c_j} P(c_j) \prod_{i=1}^{M} P(t_i|c_j) \quad (2.19)$$

We do not know the exact values of parameters $P(c_j)$ and $P(t_i|c_j)$. However, by using the maximum likelihood estimate (MLE) theorem, we can make estimations about these parameters. Let $\tilde{P}(c_j)$ be the approximate calculation of $P(c_j)$, and $\tilde{P}(t_i|c_j)$ be the approximate calculation of $P(t_i|c_j)$.

$$\tilde{P}(c_j) = \frac{N_j}{N}, \quad (2.20)$$

where N is the number of all documents, and N_j is the number of documents in category c_j. And,

$$\tilde{P}(t_i|c_j) = \frac{1+N_{ij}}{M+\sum_{i=1}^{M} N_{ij}}, \quad (2.21)$$

where N_{ij} is the number of documents belonging to category c_j and including the i[th] term t_i.

2.2.2 GENERAL LINEAR MODELS (GLM) ALGORITHM

General liner Models (*GLM*) is a popular classification algorithm. In the *GLM*, a random variable Y is assumed to follow a distribution in the exponential family. In this model, the *GLM*

generalizes two components (*stochastic* and *systematic components*) and a *link* relationship function between them.

$$(y_i|x_i) \sim N(x'_i\beta, \sigma^2) \text{ with } E(y_i|x_i) = x_i'\beta \tag{2.22}$$

and

$$Var(y_i|x_i) = \sigma^2 \ , \tag{2.23}$$

here, $x_i'\beta$ (*systematic component*) is the linear combination of the predictors x'_i and β denoting vectors of predictors and the coefficients. Also, σ^2 is a *stochastic component*. Now let's take a more detailed look at each of these components and the *link* function (Park & Hastie, 2006), (Gill, 2000).

Stochastic Component

Stochastic Component identifies the response variable ($Y = (y_1, \dots, y_n)$) and assumes a probability distribution for it. When *Y* is a continuous variable, it is usually assumed that *Y* follows a Normal distribution. In fact, In GLM, we can use any distribution in Exponential Family, because this is a comprehensive class including the properties of the Normal distribution.

Systematic Component

Systematic Component identifies the predictor variables ($x' = (x_1, \dots, x_n)$). The Systematic Component consists of the linear combination of the variables called as *linear Predictor* and some linear function of them.

$$\alpha + x_1\beta_1 + \cdots + x_n\beta_n \tag{2.24}$$

The expected value of the response variable $E(Y) = \mu$ is modeled. We want to see how μ varies as a function of the levels of the predictor variables, $x_i's$.

Link Function

Link Funtion identifies the relationship (link) between the expected value of the stochastic component ($E(Y) = \mu$) and the systematic component ($\alpha + x_1\beta_1 + \cdots + x_n\beta_n$). The link function is denoted by $g(\mu)$. It is a monotone function, that is, as the systematic part gets larger, μ gets larger (or smaller). Sometimes, the relationship between the components may be non-linear. So the general model for a GLM is

$$g(\mu) = \alpha + x_1\beta_1 + \cdots + x_n\beta_n \,. \tag{2.25}$$

Some common links are *Identity Link* (ordinary regression, ANOVA, ANCOVA) with natural parameter μ, *Log Link* with natural parameter $log(\mu)$. Log Link is usually used when *Y* is nonnegative. And finally, Logit link with Natural Parameter $log(\mu/(1-\mu))$, and in addition this link is usually used when $0 \leq \mu \leq 1$.

2.2.3 SUPPORT VECTOR MACHINE (SVM) ALGORITHM

Support Vector Machines (SVM) was first introduced by Vladimir Vapnik (Vapnik, 1979) and the first paper published was (Vapnik, 1995), (Boswell, 2002). Support Vector Machine (SVM) is a popular "binary" (Boswell, 2002) classification algorithm used to analyze data and recognize patterns (Press, 2007). Because of its high accuracy, ability to deal with high-dimensional data, the SVM is a popular classification method (Ben-Hur & Weston). For a given data set of training, each of the points is classified into one of the two classes. The SVM algorithm provides a model to choose the appropriate class/category for each point (Press,2007). In this method the first step is to identify a hyperplane which separates a dataset into its two classes/categories. If the data is not linearly separable, then the kernel notion (feature space) is used to transfer the data a higher dimensional space where the data are separable (Boswell, 2002).

For given k training examples $\{x_i, y_i\}, i = 1, \ldots, k$, where each example has d inputs ($x_i \in R^d$). ($y_i \in \{-1,1\}$) are the two classes mentioned above. We define these classes as -1 and 1. Here, (w, b) is a hyperplane, vector (w), and a constant (b), expressed in the following equation

$$w\,x \;+\; b \;=\; 0\,. \tag{2.26}$$

In addition, here w vector is orthogonal to the hyperplane which separates the training data into categories. The separation is identified by the following function

$$f(x) \;=\; sign(wx \;+\; b)\;. \tag{2.27}$$

However, a given hyperplane (w, b) can be expressed by all pairs $\{\lambda w, \lambda b\}$ for $\lambda \in R^+$. So, we define *canonical hyperplane* which separates the data from the hyperplane by a distance of at least 1 as following.

$$x_i w \;+\; b \;\geq\; +1\, when\, y_i \;=\; +1 \tag{2.28}$$

$$x_i w \;+\; b \;\leq\; -1\, when\, y_i \;=\; -1, \tag{2.29}$$

or more compactly:

$$y_i\,(x_i w \;+\; b) \geq\; 1\, for\; all\; i\,. \tag{2.30}$$

For the geometric distance of the hyperplane to a data point is calculated with following formula, but first we shold normalize by vector w.

$$d\big((w,b), x_i\big) = \frac{y_i\,(x_i \mathrm{w} \;+\; \mathrm{b})}{\|w\|} \geq \frac{1}{\|w\|}\,. \tag{2.31}$$

Our purpose is a hyperplane that maximizes the geometric distance to the closest data points. (See Figure-9) (Boswell, 2002) and (See Figures-10-11) (Wikipedia, 2011).

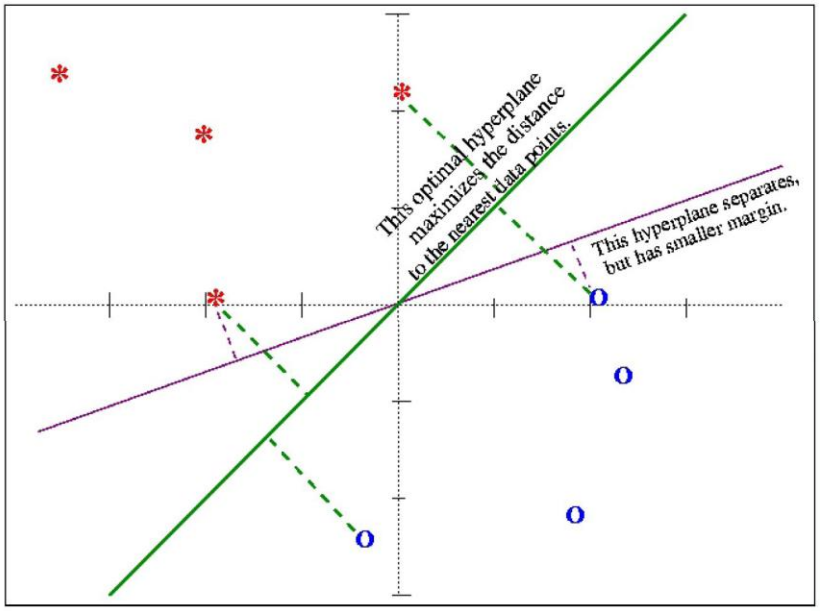


Figure-9: The green hyperplane maximizes the geometric distance to the closest data points.

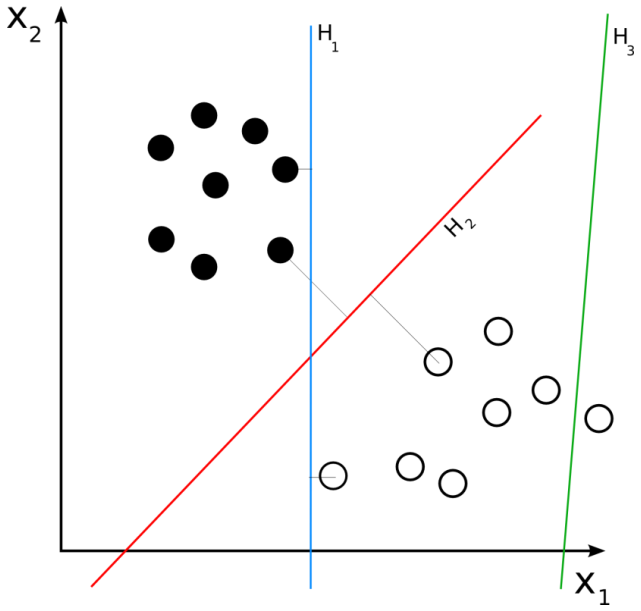


Figure-10: The green hyperplane doesn't separate the two classes. The blue one does, with a small margin and The red hyperplane with the maximum margin.

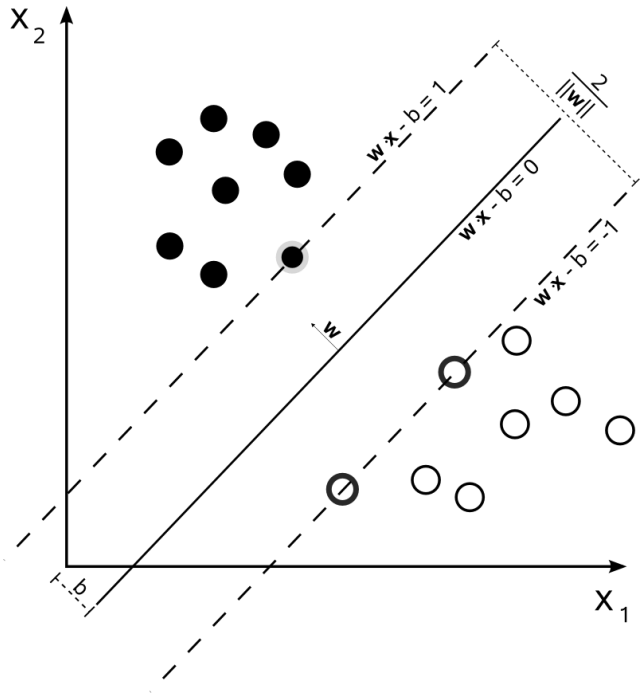


Figure-11: Maximum-margin hyperplane and margins for an SVM trained with samples from two classes. Samples on the margin are called the support vectors.

2.2.4 K-MEANS ALGORITHM

K-Means is a simple and unsupervised clustering algorithm in data mining. The general idea in this method is to separate a sum of (n) observations into k clusters. The separation is done according to means of clusters; each observation is classified into a cluster with the nearest mean (centroid) (MacQueen, 1967). In this algorithm we try to minimize a squared error function called *objective function* (Moore, 2001).

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| x_i^{(j)} - \mu_j \right\|^2, \tag{2.32}$$

where $\left\| x_i^{(j)} - \mu_j \right\|^2$ is any distance measure between a data point x_i^j and the cluster mean c_j. This function measures the distance of each data point from their respective cluster means. The algorithm is composed as following (Moore, 2001):

1. First, decide on places of k clusters and their centroids.
2. By using the above formula $(X.Z)$, assign each point into the group that has the nearest mean (centroid).
3. After the classification of all points, calculate the mean of the each cluster and decide on whether the previous centroids defined in step-1are close to the real centroids. If it is not then change the positions of the k centroids. And according to the new position, apply the second step again.
4. Repeat Steps 2 and 3 until the centroids no longer move. At the end of these repeated steps we will have stable positions for all points, and completely separated clusters. (See Figure-12) (Kumar, 2002).

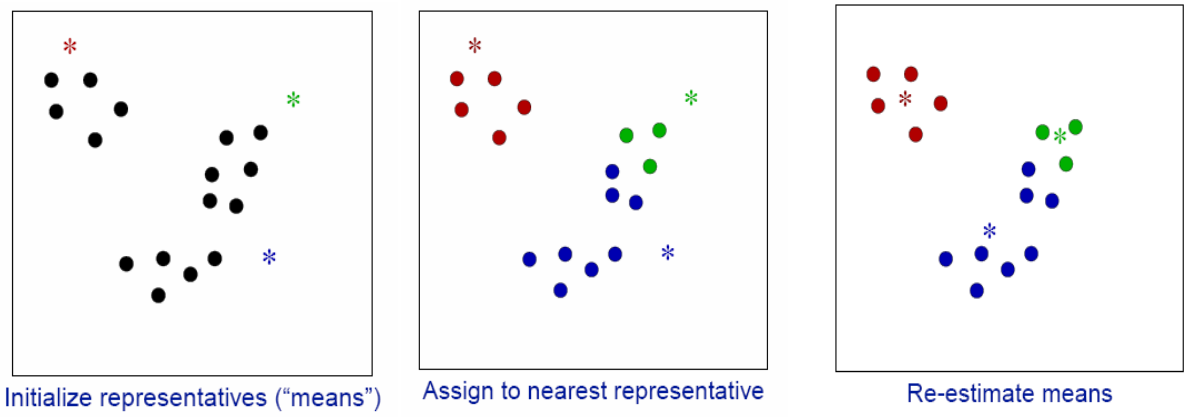


Figure-12: *Steps of the K-Means Algorithm*

Dramatic differences between the sizes, densities of clusters, empty clusters and outliers may be problem for this algorithm (Kumar, 2002).

2.2.5 APRIORI ALGORITHM

Apriori algorithm is a popular algorithm used to find the association rules. Apriori algorithm consists of repeating steps based on a criterion. The algorithm uses the minimum support value to create a set of common repetitive elements as the criterion. The minimum

support level depends on the context (Agrawal, 1994). If an item-set has a frequency larger than minimum support, then it is called as a frequent item-set. In addition, any subset if a frequent item-set is also a frequent item-set (Jin, McCallen, Breitbart, Fuhry & Wang). See the following example (see Figure-13) (Zaiane, 2001). The database of transactions consist of the sets $\{1,3,4\}, \{2,3,5\}, \{1,2,3,5\}, \{2,5\}$. Each number can be thought as a product or a term in a concrete experiment. The first step of Apriori is to count up the frequency (support) of each number (item) separately. We appointed the minimum support level as "1" for this example. Therefore, $\{4\}$ is not frequent. So, we remove $\{4\}$ from the first candidate item-set C_1. The next step is to generate C_2 that is the item-set of all 2-pairs of the frequent items. In the same way, we remove $\{1,2\}$, $\{1,5\}$ whose frequencies are "1". And then, we generate the tree of all possible sets. The third candidate item-set is $\{\{2,3,5\}\}$. Therefore, because of $\{2,3,5\}$'s support is 2, the last frequent item-set L_3 is $\{\{2,3,5\}\}$. As a note, in the last scan, we did not include $\{1,2,3\}$, $\{1,2,5\}$ and $\{1,3,5\}$ in the C_3 because in the previous scan we identified $\{1,2\}$, $\{1,5\}$ items as infrequent, and $\{1,2\} \subset \{1,2,3\}$, $\{1,2\} \subset \{1,2,5\}$, $\{1,5\} \subset \{1,2,5\}$, and $\{1,5\} \subset \{1,3,5\}$. Now we can identify a association rule between the items of the last item-set.

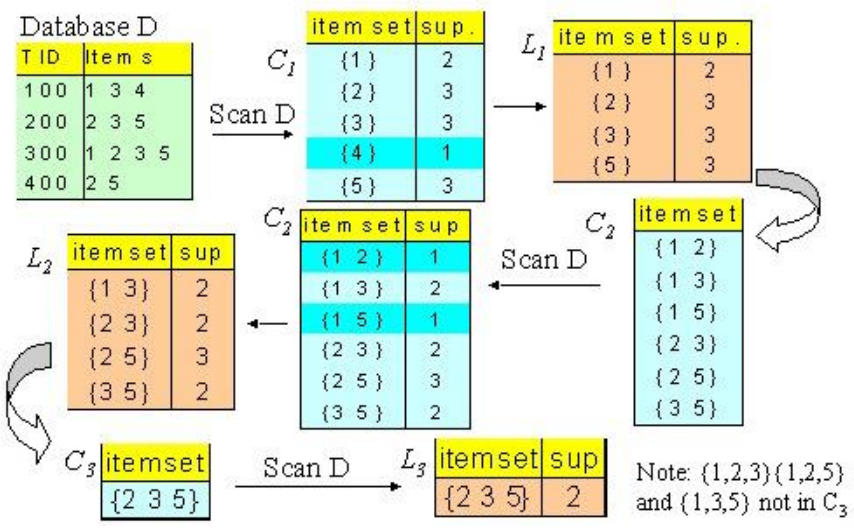


Figure-13: *Apriori Algorithm example*

where D is transactional database, C_k is candidate item-set of size k, and L_k is frequent item-set of size k.

2.2.6 DECISION TREES

Decision trees are tree-like graphs or models that help to identify strategies to reach decisions about specific goals. This is a useful method to display an algorithm and choose between several paths of subsection (Wikipedia, 2011). A decision tree consists of nodes and splits. The tree starts with a single node including all training data. The initial node split into two new nodes by using a predictor variable. And then the two nodes split into two new splits by using the second and third predictor variables, and this continues until a terminal node. A terminal node is one where no more splits are made (StatSoft, Inc., 2011).

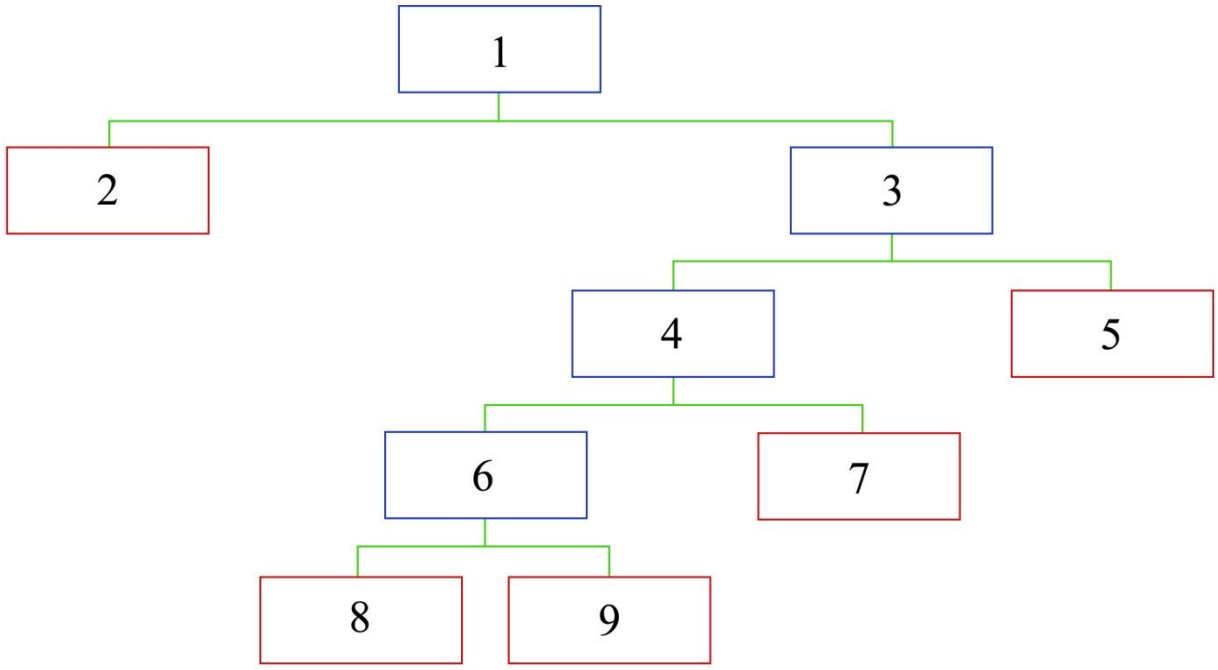


Figure-14: *Decision Tree Layout*

A decision tree may include a large number of nodes but it is not hard to evaluate and reach a decision because it is very simple and straightforward for interpretation. Generally the decision tree models give good accuracy rate, but sometimes can be minor challenges. At these times, a researcher need to find the right sized tree, and this requires time and experience (StatSoft, Inc., 2011).

3. A TEXT ANALYTICS APPLICATION

In this section, we apply text analytics, as previously described to a case study that we designed, which is related to hotel ratings. The data for this experiment was taken from *tripadvisor.com* website. This is a review sharing platform where people share their experiences about hotels, flight companies, restaurants and so on. Before scheduling holiday programs and reservations, it is a very useful website to get a preliminary impression about the holiday programs.

3.1 STUDY DESIGN AND DATA DESCRIPTION

For this experiment hotel reviews were used. We chose three popular tourist cities; *Istanbul*, *New York* and *Moscow*, and then randomly selected 20 hotels from each city and finally randomly 10 reviews for each of those hotels.

The frequency for each city is the final data set as following. (You can find other summary statistics in the appendix).

	Frequency table: City (A1- Ist_NY_Mosc)			
Category	Count	Cumulative Count	Percent	Cumulative Percent
Istanbul	202	202	34,00673	34,0067
Moscow	199	401	33,50168	67,5084
New York	193	594	32,49158	100,0000
Missing	0	594	0,00000	100,0000

Table-1: *Data frequencies for each city*

3.2 MATERIALS/TOOLS

StatSoft (*Statistica*) software was used. For all analyses reported here.

3.3 PROCEDURE

First, a usual text mining filtration was applied. One-letter words, stop-words and other words which do not match the scale given in the following table were excluded from the data. And then,

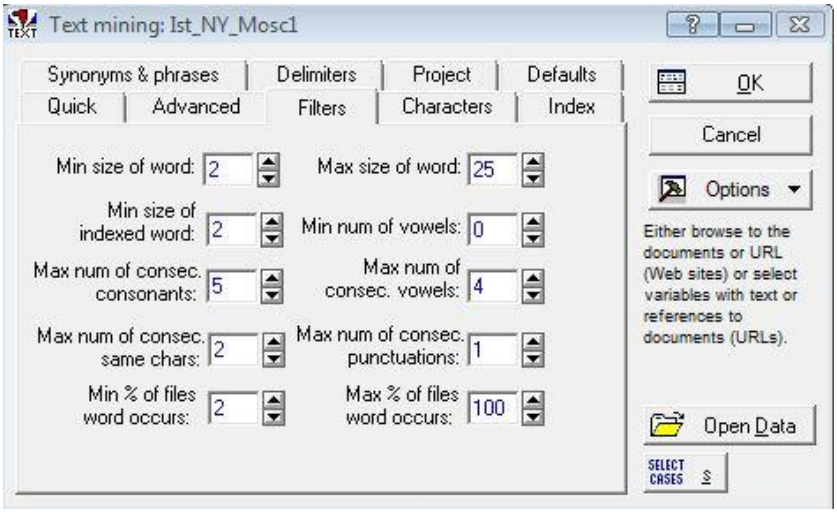


Figure-15: *Text mining filtration window*

the number of the remaining important words was counted. As shown in the following table, we have 143 selected words; an inverse-document-frequency option is selected to analyze and report document frequencies. We chose this analysis method because this is the most common and very useful method that gives both document frequencies and the general frequencies of their (selected words) occurrences (word frequencies/tf-idf).

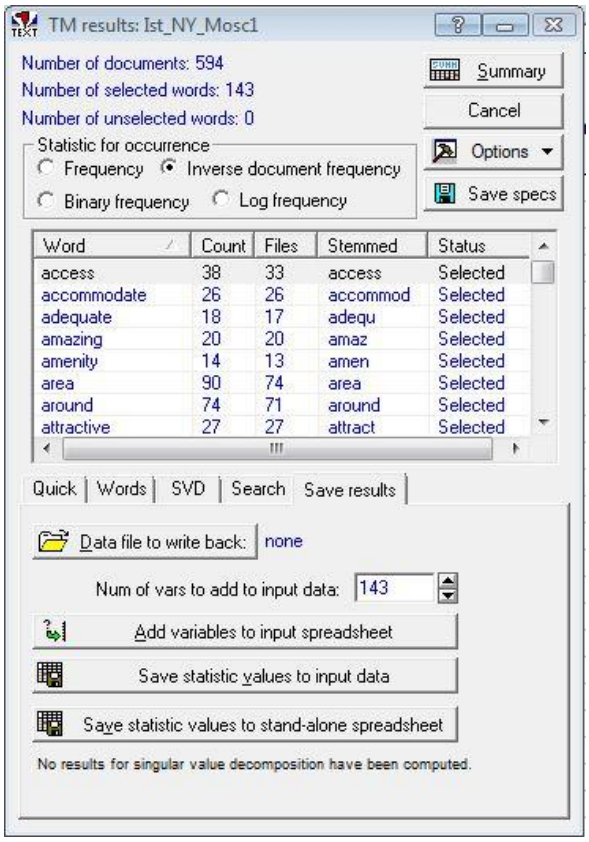
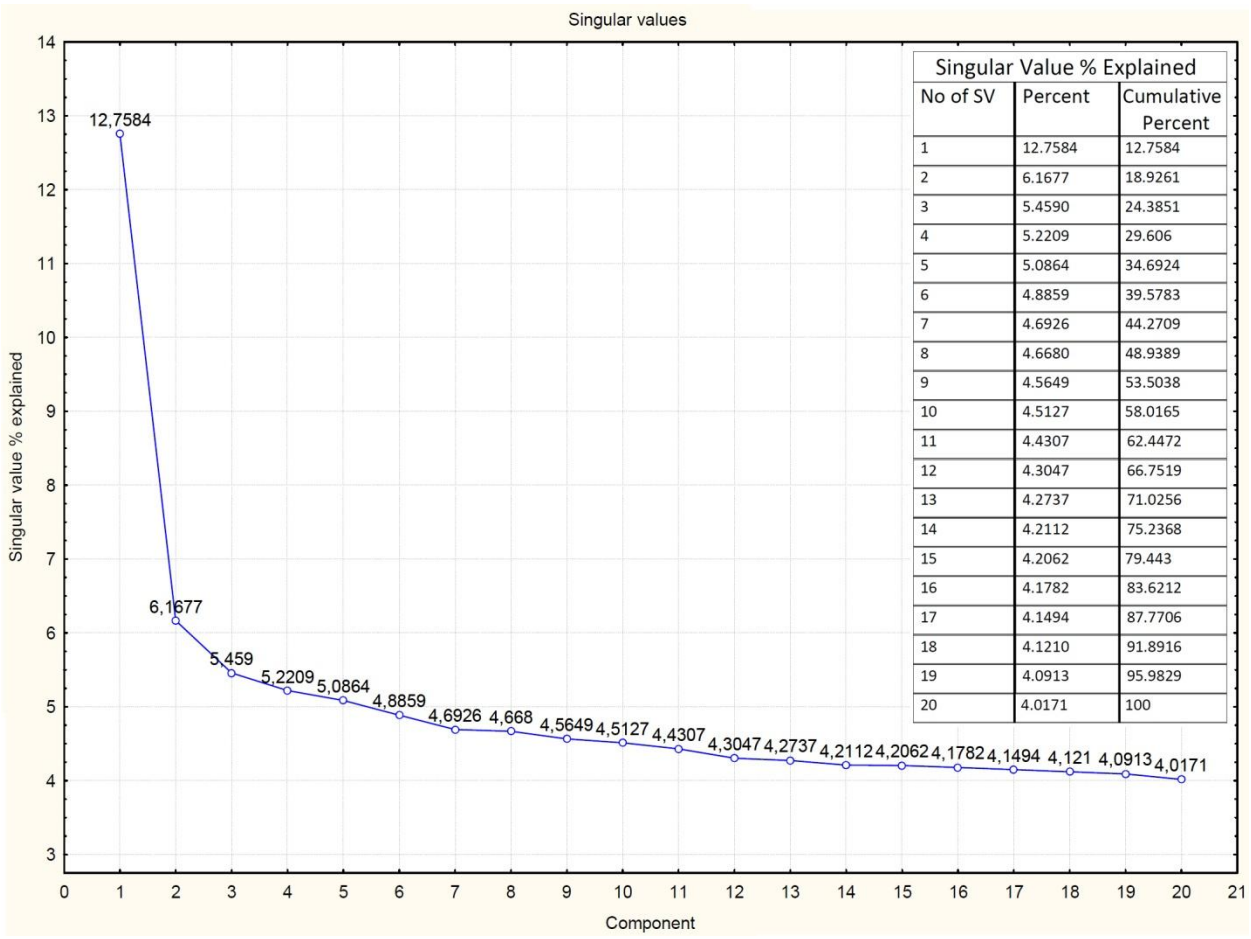


Figure-16: *Selected words*

Before creating the data set table having words as variables and their frequencies, we performed *Singular Value Decomposition* (SVD) on the document-by-words matrix, based on the selected words. The first 20 components were retrieved for analysis. Also, we created a Scree Plot to see the number of singular values that are useful for the analyses. We can tell by looking at the following scree plot that the first component explains almost 13%, the second one explains slightly more than 6% and in total these two components explain 19% of the total variance for 143 words. And then,



Singular Value % Explained

No of SV	Percent	Cumulative Percent
1	12.7584	12.7584
2	6.1677	18.9261
3	5.4590	24.3851
4	5.2209	29.606
5	5.0864	34.6924
6	4.8859	39.5783
7	4.6926	44.2709
8	4.6680	48.9389
9	4.5649	53.5038
10	4.5127	58.0165
11	4.4307	62.4472
12	4.3047	66.7519
13	4.2737	71.0256
14	4.2112	75.2368
15	4.2062	79.443
16	4.1782	83.6212
17	4.1494	87.7706
18	4.1210	91.8916
19	4.0913	95.9829
20	4.0171	100

Figure-17: *Scree Plot*

By using these two components we created the following scatter plots to visualize the relationship between the words. The points appearing close to each other are related with each other. The two plots are exactly the same. For the second one, we showed the labels for each point and appointed colors to the points that have negative or positive meaning. For example, we appointed green for the words such as *clean, close, love, good,* and purple for the words such as *bad, dirty, far, never, expense* and so on. For the words that we could not assign into any of the two groups, such as *center, Russian, breakfast*, and so on, we just left as they are, black. The nearest points for *Istanbul* are *mosque, love, neighborhood, enjoy, convenient, quiet*, and *complain*. In a similar way, the nearest points for *New York* are *comfort, clean, nice, shower,*

help, *small*, *locate* and *staff*, and for *Moscow*, *better*, *close*, *food*, *expense*, *never*, *old*, *busy*, *bar* and *serve*.

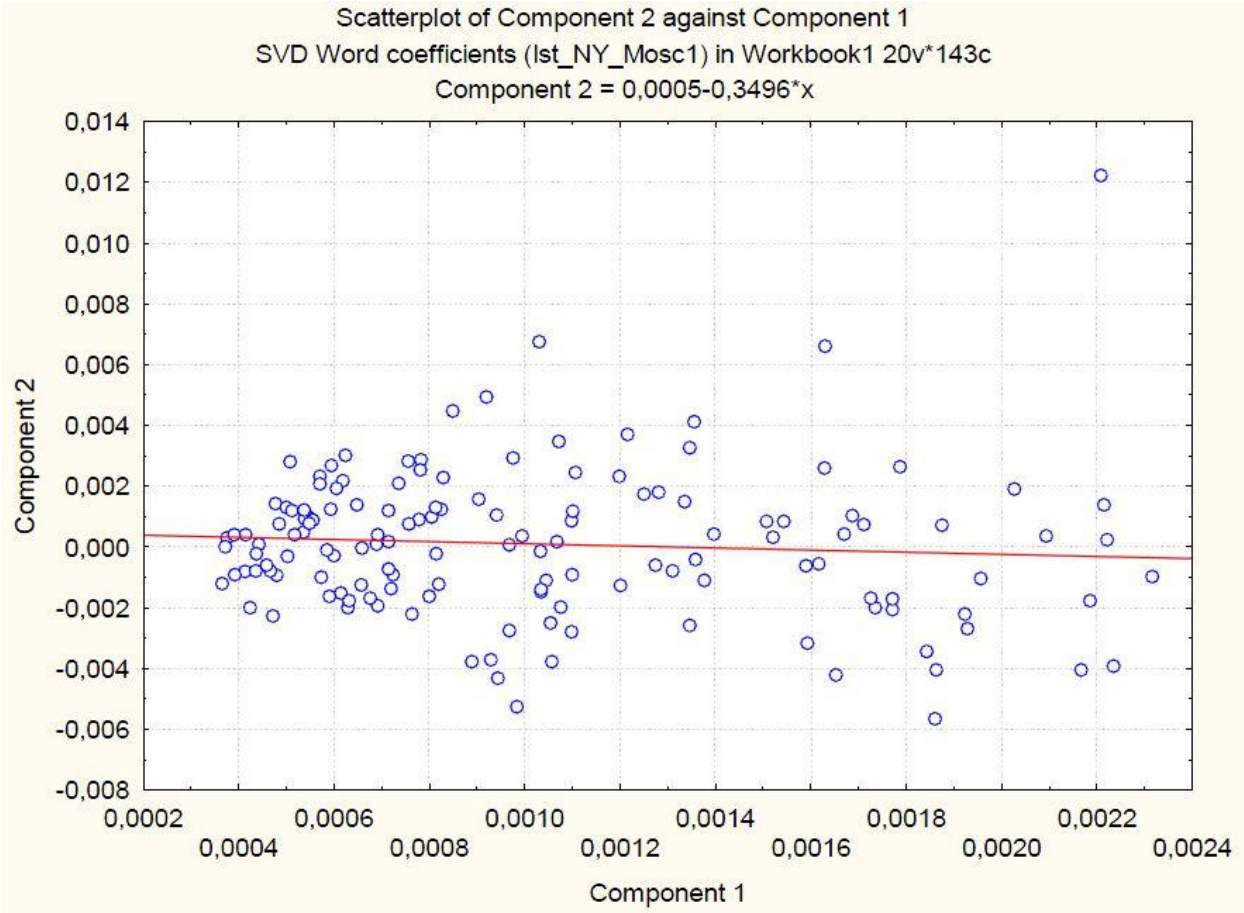


Figure-18: *Scatterplot of Component-2 against Component-1*

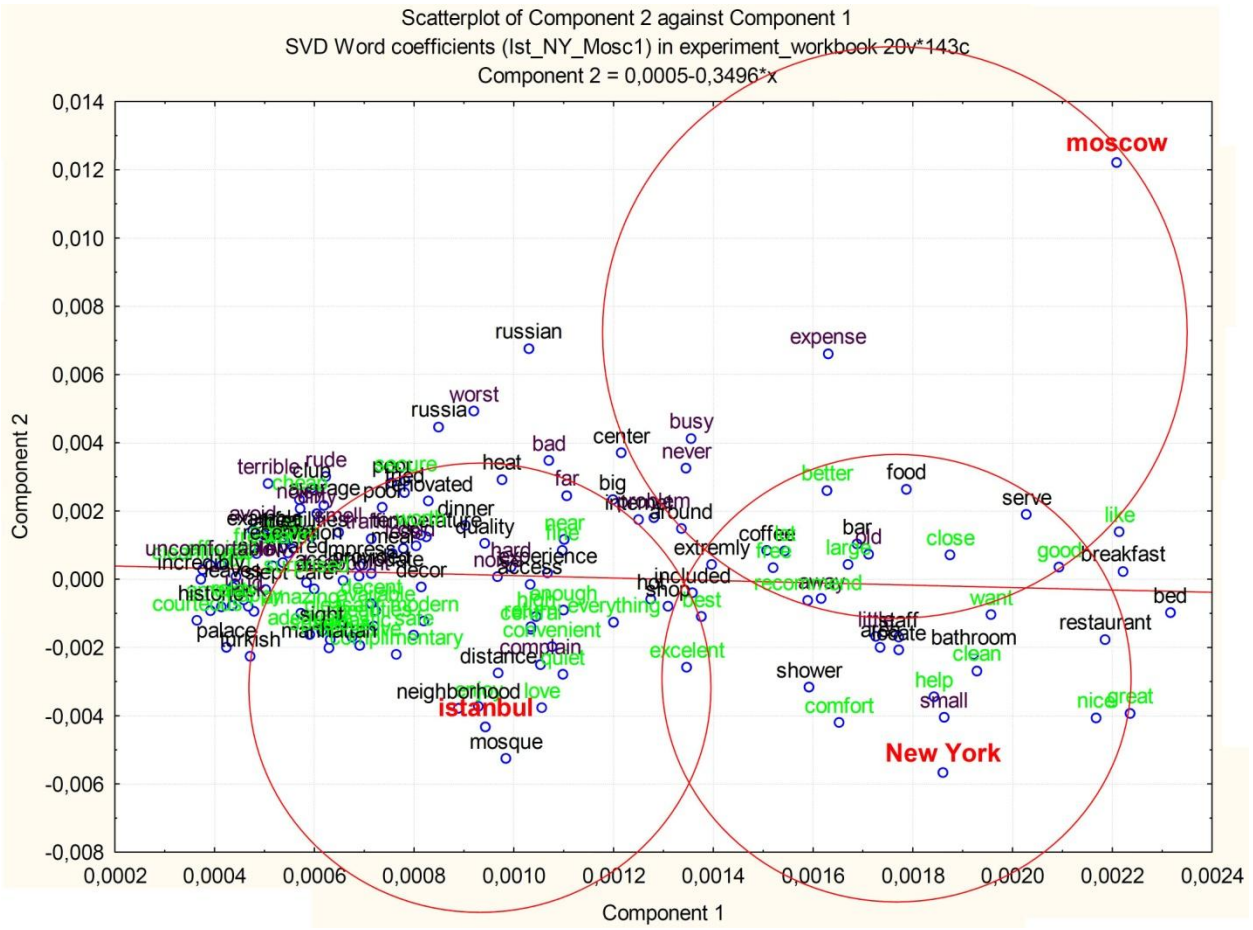
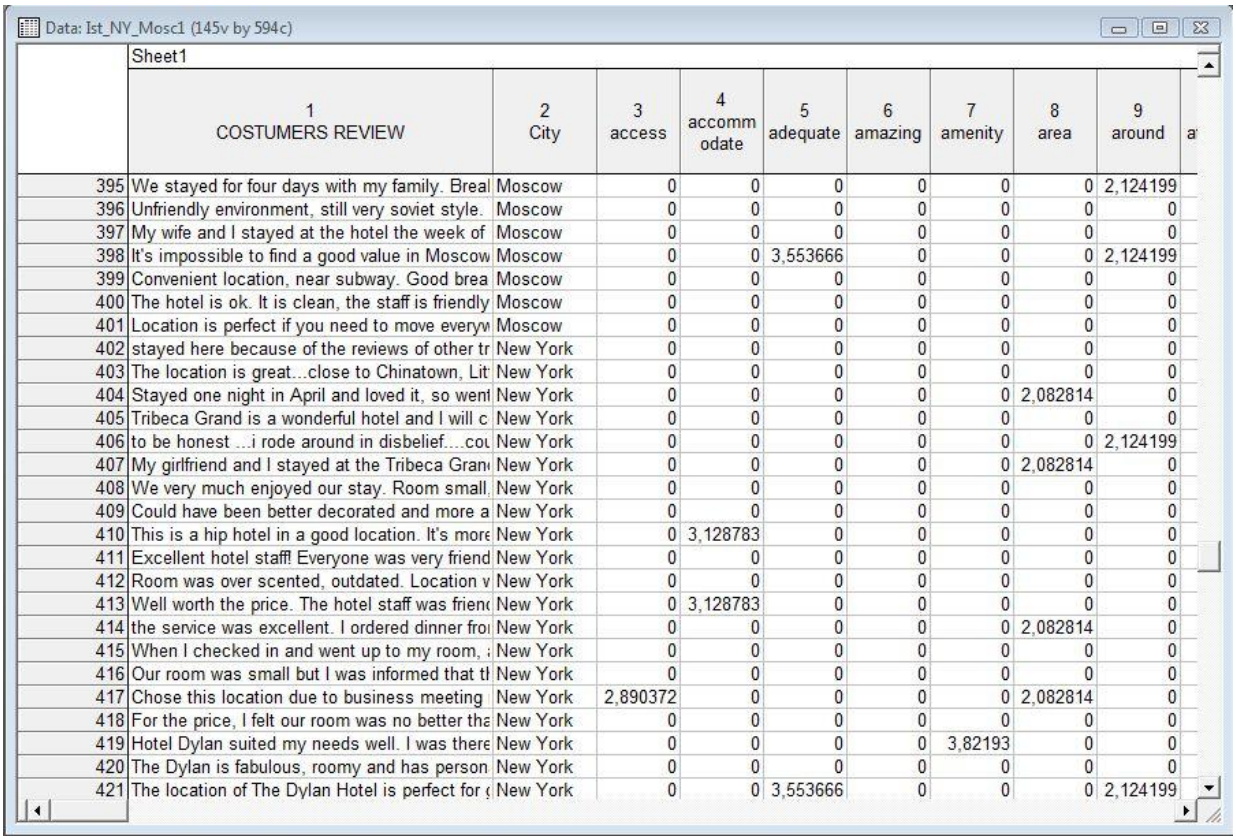


Figure-19: *Labeled and Colored Scatterplot of Component-2 against Component-1*

Now we can create our data set. The data set would resemble the following table. The first variable '*Customer Reviews*' contains customer reviews in text format, '*City*' is a categorical variable and the others are continuous variables.



Data: Ist_NY_Mosc1 (145v by 594c)

Sheet1

	1 COSTUMERS REVIEW	2 City	3 access	4 accommodate	5 adequate	6 amazing	7 amenity	8 area	9 around	a
395	We stayed for four days with my family. Breal	Moscow	0	0	0	0	0	0	2,124199	
396	Unfriendly environment, still very soviet style.	Moscow	0	0	0	0	0	0	0	
397	My wife and I stayed at the hotel the week of	Moscow	0	0	0	0	0	0	0	
398	It's impossible to find a good value in Moscow	Moscow	0	0	3,553666	0	0	0	2,124199	
399	Convenient location, near subway. Good brea	Moscow	0	0	0	0	0	0	0	
400	The hotel is ok. It is clean, the staff is friendly	Moscow	0	0	0	0	0	0	0	
401	Location is perfect if you need to move everyw	Moscow	0	0	0	0	0	0	0	
402	stayed here because of the reviews of other tr	New York	0	0	0	0	0	0	0	
403	The location is great...close to Chinatown, Lit	New York	0	0	0	0	0	0	0	
404	Stayed one night in April and loved it, so went	New York	0	0	0	0	0	2,082814	0	
405	Tribeca Grand is a wonderful hotel and I will c	New York	0	0	0	0	0	0	0	
406	to be honest ...i rode around in disbelief....cou	New York	0	0	0	0	0	0	2,124199	
407	My girlfriend and I stayed at the Tribeca Gran	New York	0	0	0	0	0	2,082814	0	
408	We very much enjoyed our stay. Room small,	New York	0	0	0	0	0	0	0	
409	Could have been better decorated and more a	New York	0	0	0	0	0	0	0	
410	This is a hip hotel in a good location. It's more	New York	0	3,128783	0	0	0	0	0	
411	Excellent hotel staff! Everyone was very friend	New York	0	0	0	0	0	0	0	
412	Room was over scented, outdated. Location v	New York	0	0	0	0	0	0	0	
413	Well worth the price. The hotel staff was frien	New York	0	3,128783	0	0	0	0	0	
414	the service was excellent. I ordered dinner fro	New York	0	0	0	0	0	2,082814	0	
415	When I checked in and went up to my room, :	New York	0	0	0	0	0	0	0	
416	Our room was small but I was informed that th	New York	0	0	0	0	0	0	0	
417	Chose this location due to business meeting	New York	2,890372	0	0	0	0	2,082814	0	
418	For the price, I felt our room was no better tha	New York	0	0	0	0	0	0	0	
419	Hotel Dylan suited my needs well. I was there	New York	0	0	0	0	3,82193	0	0	
420	The Dylan is fabulous, roomy and has person	New York	0	0	0	0	0	0	0	
421	The location of The Dylan Hotel is perfect for	New York	0	0	3,553666	0	0	0	2,124199	

Figure-20: *General look of the Data Set*

In Figure-19, in the circles that we draw around each city, there were some green, purple and black words. It was not difficult to see that the ratio of green points in the circle for Istanbul was higher than the others. Now, we can see this situation with another method called *Interaction Plot*. For this application we first created a *Negative Connotations* set;

Negative Connotations={avoid, bad, busy, crowd, dirty, disappoint, expense, far, lack, low, never, noisy, old, poor, problem, rude, terrible, uncomfortable, worst } .

The elements of this set are negative words used by customers in their reviews. Then we created a new variable for these negative connotations and recoded the variable as shown in the following table. If a document includes any of these negative words, we add '1' for each one, otherwise '0'. And finally, we created the '*City×Negative_Connotations Interactive Plot*' by using the categorical variable *city* and *Negative_Connotations.*

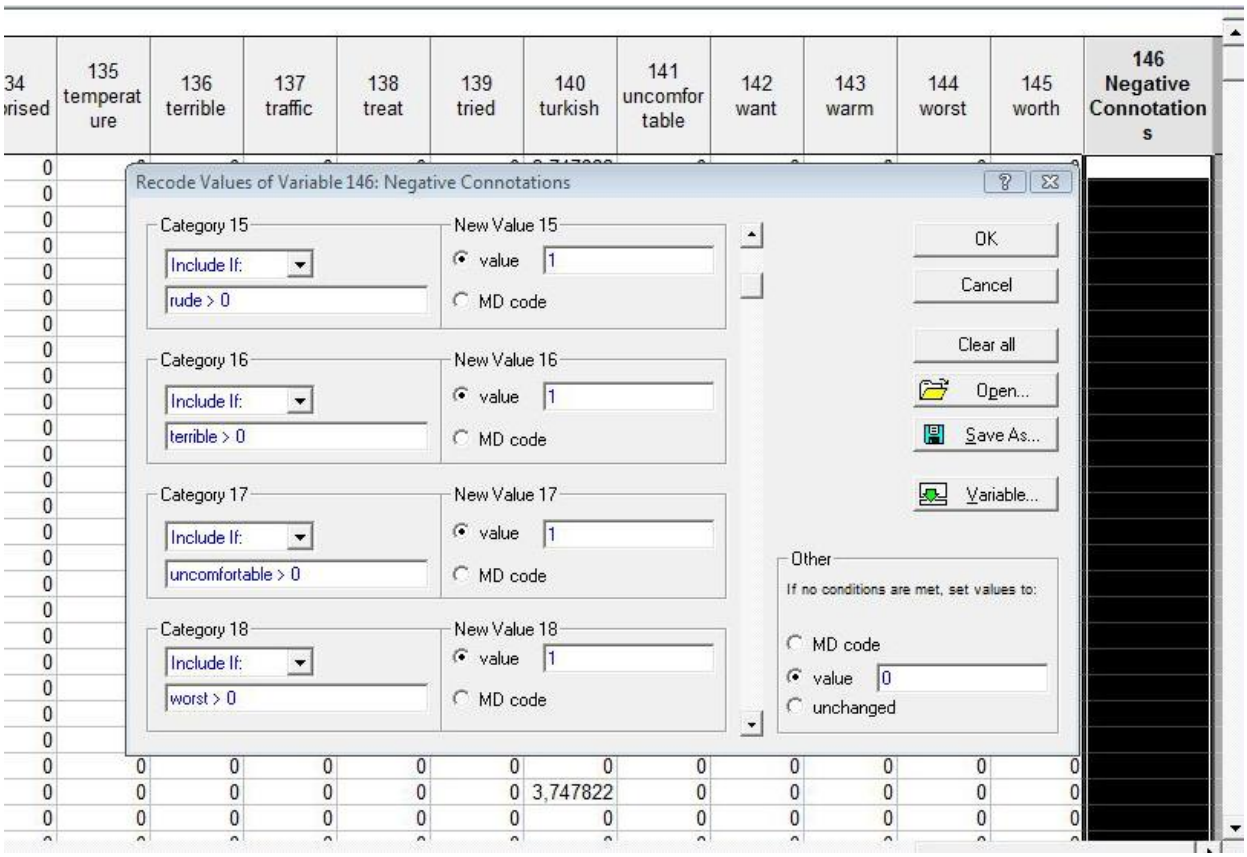


Figure-21: *Recoding Negative_Connotations variable*

From the following graph, we can tell that Moscow accumulated the greatest number of reviews containing negative connotations (more than 160), followed by New York and Istanbul. In addition, we can tell that Istanbul got the fewest number of negative reviews from the customers when compared to the other cities.

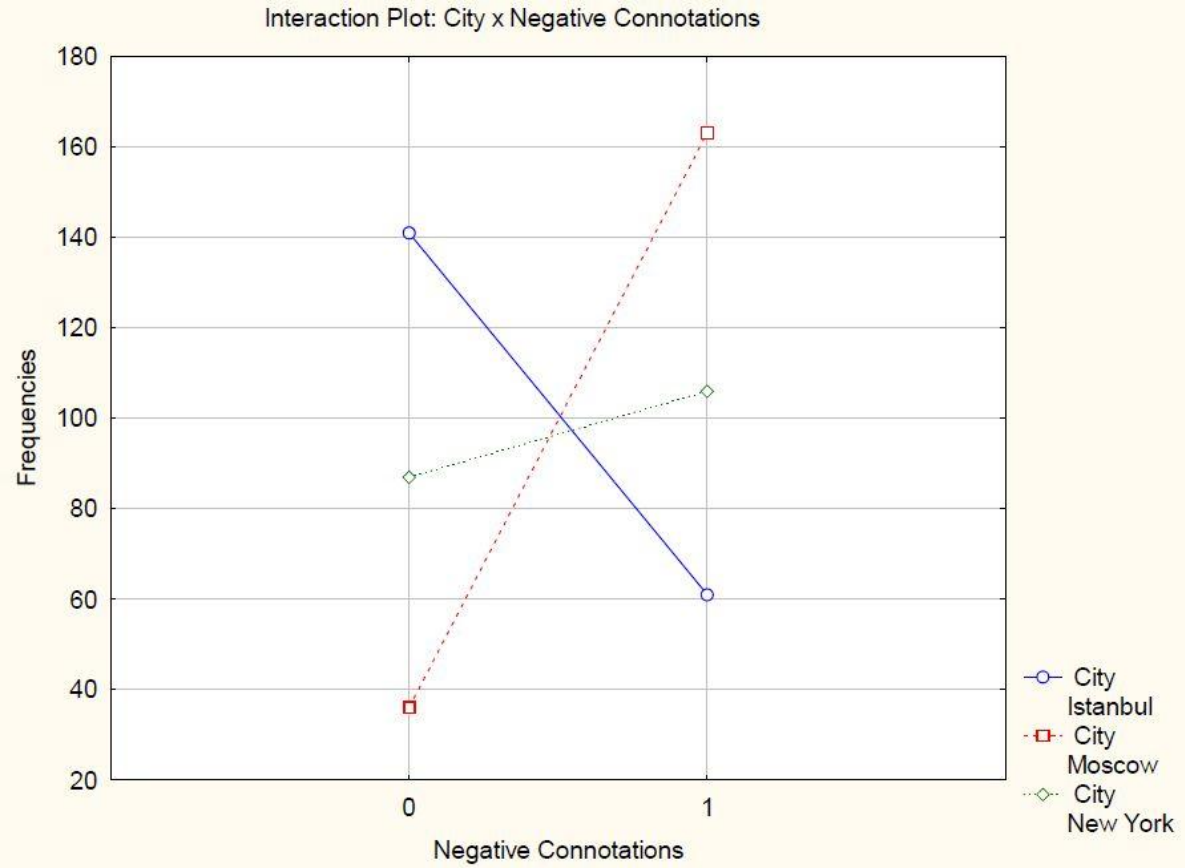


Figure-22: *Interaction Plot: City vs Negative_Connotations*

Next we build an interactive decision tree model called Classification and Regression Tree (C&RT) to classify cities and use the extracted words as predictors and so expose any possible hidden useful information. The decision tree models are quite simple and easy for interpretation. Building a tree using all the 143 variables would be very complex and useless. However, we created the following decision tree and calculated the error rate of the predictive model just to see what would be the error rate of the predictive model created by using all the selected 143 features. For this job we used the above 20 components which are giving nearly the same information about the whole dataset.

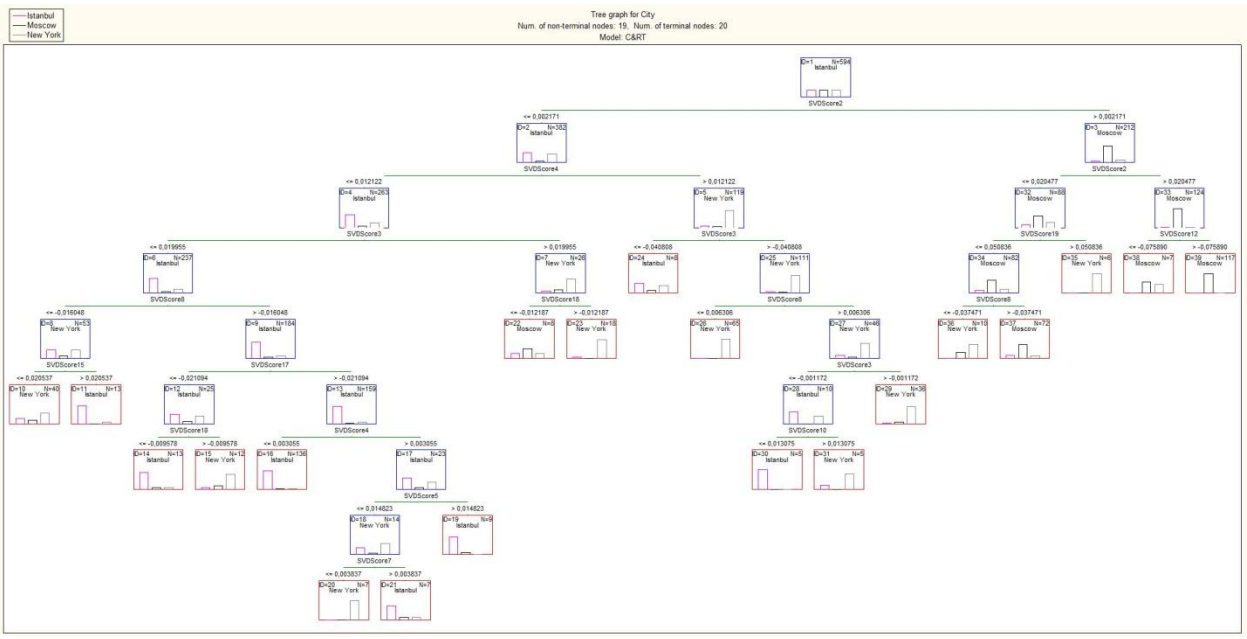


Figure-23: *Decision Tree with SVD scores*

Because of we cannot know which component gives what kind of information about the impression of reviewers, the decision tree is not useful. However, we can tell that as given in the following table, if we used all the selected features, the error rate of the decision tree would be 13.47 (or something its around).

	1	2	3
Model selected for deployment	1		
Model Evaluation Summary	ID	Name	Error rate (%)
	1	C&RT	13,47
Table	Step options		
	Date and time	31.08.2011 16:26:51	

Table-2: *Model Evaluation of the predictive model with SVD scores*

Now we would like to create a useful predictive model to reach decisions about the goal of our research. So, first we need to select some important variables. This variable selection operation is called ‘Feature Selection’ as we mentioned in the second chapter. We selected the most important 20 features. In the second step we are going to build a second decision tree and so

select another 20 features in a different format, but for the first decision tree, the features as given in the following table. They are all single words and in order of importance.

	Best predictors for categorical dependent var: City (Ist_NY_Mosc1) Important variables selection method: Feature selection	
	Chi-square	p-value
mosque	101,3374	0,000000
expense	82,6003	0,000000
russian	47,4922	0,000000
russia	43,2111	0,000000
bed	40,1920	0,000000
small	38,6244	0,000000
manhattan	38,5677	0,000000
great	36,7970	0,000000
shower	33,4934	0,000000
nice	30,2346	0,000004
locate	29,3874	0,000007
busy	28,8705	0,000001
palace	27,8241	0,000001
turkish	27,8241	0,000001
bathroom	22,7079	0,000012
comfort	22,2933	0,000014
worst	20,3363	0,000038
attractive	20,3062	0,000039
never	20,2908	0,000039
center	19,8523	0,000049

Table-3: *Selected Features-1*

In this step, we did not consider whether the selected features are giving important information about the goal of our research or not, only numerical importance was enough. Our goal is to see people's general impressions about the hotels in these cities and explore the reasons. However, for example, even though *Russian* is a numerically very important feature, it does not contribute so much to our goal.

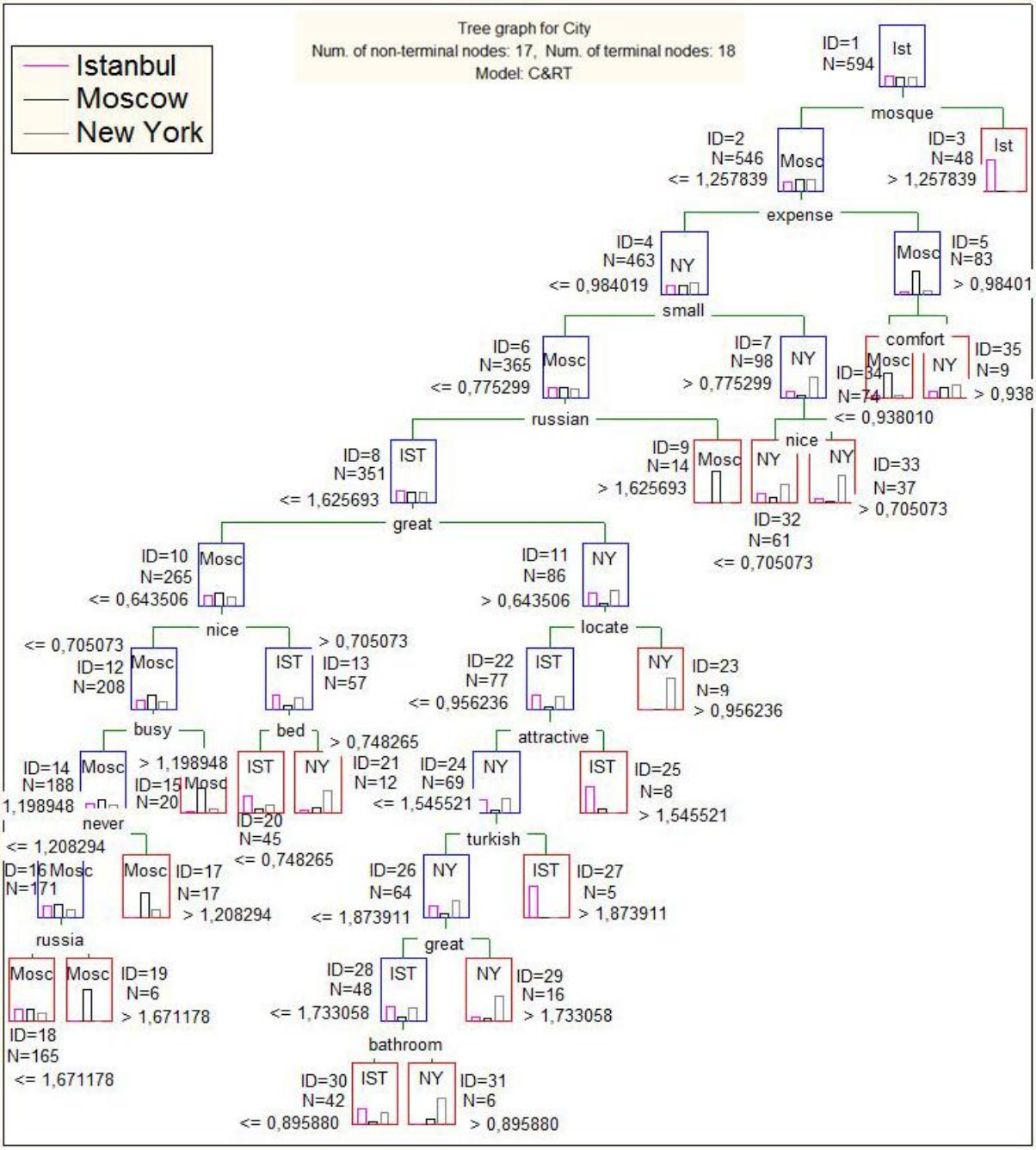


Figure-24: *Decision Tree-1*

The reason we create a decision tree is to learn how to discriminate between the hotels of different cities based on the extracted inverse document frequency of variables. The root node is divided on the inverse document frequency of the feature *mosque* creating two new nodes, one with city *Istanbul* as predominant category among 48 cases dropping into a terminal (red) node.

This means that if the importance of the feature *mosque* is higher than 1.257839, reviews are mostly about *Istanbul*, otherwise it is *Moscow*. Similarly, if the importance of the features *Attractive* and *Turkish* are high, reviewers are referring to *Istanbul*; if the importance of the features *small, comfort, nice, great, bed* and *bathroom* are high, reviewers are referring to *New York*; and if the importance of the features *expense, busy, never, Russian, Russia* are high, reviewers are referring to *Moscow*. By looking at the current situation we can tell that the main attraction of *Istanbul* city is mosques, and reviewers generally have a good impression about the hotels of the city. For *New York* City reviewers have very good impression, and they think hotels in this city are very comfortable. Although rooms, beds or bathrooms are a little bit small, it seems that this does not change the general impression so much. For Moscow the general impression is not so good; reviewers think it is a busy and expensive city.

	1	2	3
Model selected for deployment	1		
Model Evaluation Summary	ID	Name	Error rate (%)
	1	C&RT	35,19
Table	Step options		
	Date and time	09.08.2011 04:58:53	

Table-4: *Model Evaluation-1*

The predictive classification model we created above gave us an opportunity to easily make use of this textual information. The error rate of the model shown in the table is 35.19%. when error was calculated as, we know that accuracy rate is equal to $(100 - Error\ Rate)$, the model has $(100 - 35.19) = 64.81\%$ accuracy rate. It is not that bad, but needs improvement.

	Best predictors for categorical dependent var: City (Ist_NY_Mosc1) Important Variables Selection Method: Feature Selection	
	Chi-square	p-value
BetterLocation_Safe_GoodBathroom	64,23453	0,000000
CentralNeighborhood_NicePlace	62,88640	0,000000
HardCoffee_GoodSleep_GoodShower	59,03098	0,000000
Expensive_InternetAccess_Bar	57,86329	0,000000
SmallBathroom_HardCoffee_Safe	57,07258	0,000000
ExtremelyHeatTemperature_lowQualityShower	54,21591	0,000000
HistoricPlaces_Beauties_ModernHotel	50,97703	0,000000
QuietArea_ComfortConditions_EnoughQualityBathroom	50,18209	0,000000
CentralLocation_EnjoaybleShower_AdequateFacilitties	47,40924	0,000000
BadImpression_BadConditions	47,04301	0,000000
BusyCity_Bar_Club	45,10984	0,000000
WarmTemperature_Hot_Heat	37,63094	0,000009
NoisyLocation_Near_Russian	36,57177	0,000002
OldHotel_BadSmell_Pool_Temperature	36,19397	0,000016
SmellBad_BadConditions_Near	35,85149	0,000019
GoodShower_GoodConditions_GoodStaff	35,79569	0,000019
Far_NoisyArea_Clean_Club	35,22780	0,000114
FantasticCity_GoodImpression_FreeFacilities	32,98923	0,000274
GoodImpression_SafeLocation	32,53807	0,000075
FreeBreakfast_GoodStaff_GoodHotel	32,52302	0,000327

Table-5: *Selected Features-2*

In this second step we build a new decision tree and try to improve the accuracy rate that we achieved in the first model. For this purpose first we are going to change our feature selection method. As mentioned above, in the first feature selection we did not consider whether the selected features contributed to our goal or not, just the numerical importance was enough. However, in this step we will intuitively select features that are useful in quality and impression evaluation of the hotels without paying attention to the numerical importance, and construct two, three or four word phrases. In addition, while constructing the phrases we are going to use correlation between the features.

From the 143 features, we created 62 new features (phrases) and by using the feature selection option of the *StatSoft* software, we selected the most important 20 features given in the above table.

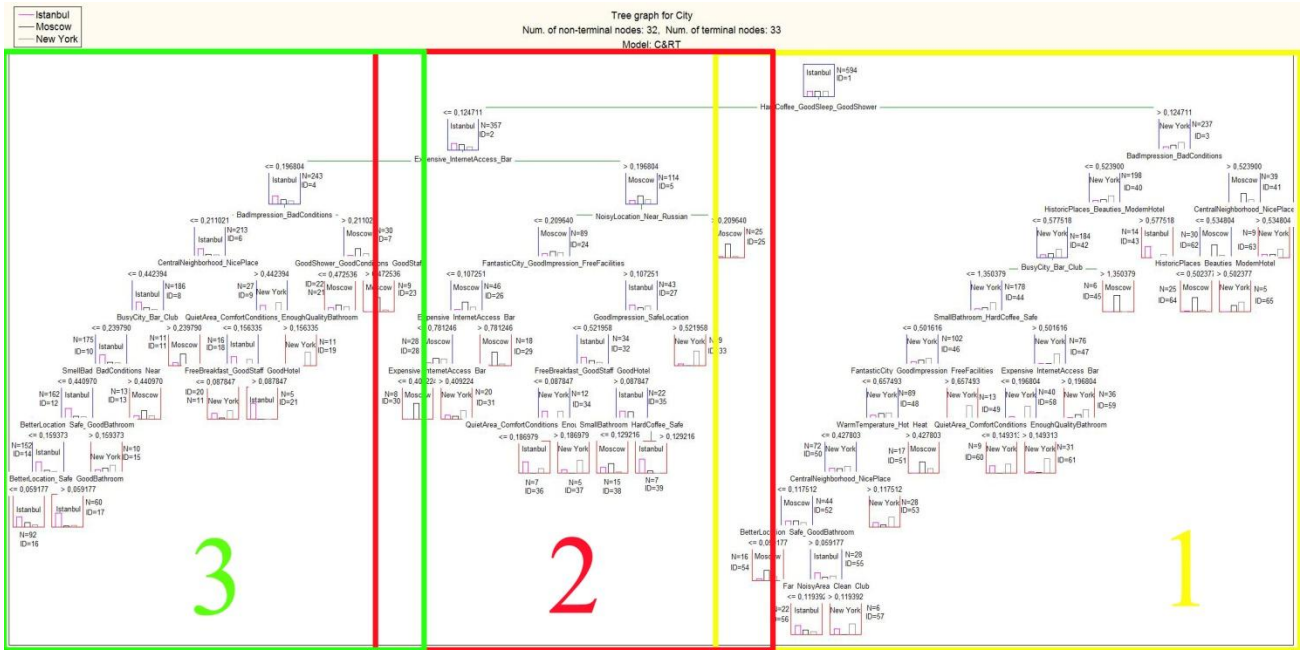


Figure-25: *Decision Tree-2*

By using the selected features we built the above decision tree. In total we have 65 nodes, 32 non-terminal and 33 terminal nodes. For the first node Istanbul is the predominant category among 594 cases. Because the tree is too big to display in a single page, we divided it into three parts.

The root node is divided on the inverse document frequency of the feature *HardCoffee_GoodSleep_GoodShower* forming two new nodes, one with city *New York* as predominant category among 237 cases. If the importance of the feature *HardCoffee_GoodSleep_GoodShower* is higher than 0.124711, reviews are mostly about *New York*, otherwise it is *Istanbul*. Similarly, if the importance of the features *HistoricPlaces_Beauties_ModernHotel, BetterLocation_Safe_GoodBathroom, FreeBreakfast_GoodStaff_GoodHotel and SmallBathroom_HardCoffee_Safe* are high, reviewers are referring to *Istanbul*, if the importance of the features *CentralLocation_NiceNeighborhood, HistoricPlaces_Beauties_ModernHotel,*

FantasticCity_GoodImpression_FreeFacilities, Expensive_InternetAccess_Bar, QuietArea_ComfortConditions_EnoughQualityBathroom, Far_NoisyArea_Clean_Club, GoodImpression_SafeLocatio and BetterLocation_Safe_GoodBathroom are high, reviewers are referring to *New York*, and if the importance of the features *BusyCity_Bar_Club, WarmTemperature_Hot_Heat, Expensive_InternetAccess_Bar, NoisyLocation_Near_Russian, GoodShower_GoodConditions_GoodStaff, SmellBad_BadConditions_Near* are high, reviewers are referring to *Moscow.*

This time the features are not individual words. We defined phrases as all words that occur sequentially and so each feature gives enough idea by itself.

The Error Rate for the second model is 29.63%, and so the accuracy rate is (100 – 29.63) = 70.37%. For the first model the accuracy rate was 64.81%. So, after changing the method of creating *feature space* the accuracy rate has been improved 5.56%. This means that the second model is more reliable than the previous one. Of course 70.37% can be improved as well. If we had a larger document set we think that the improvement would be more dramatic, but for this amount of document set, 5.56% improvement is quite good.

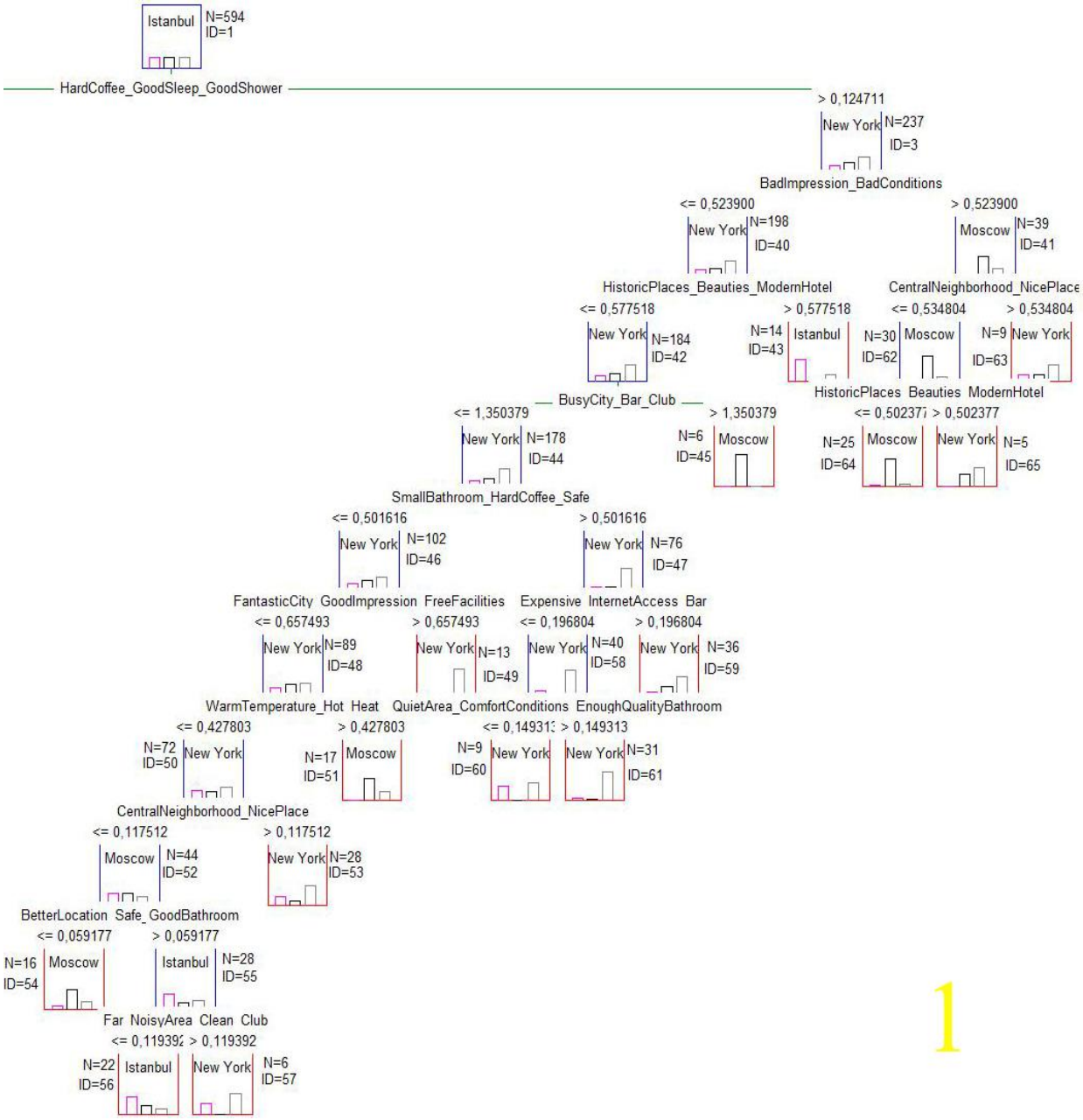


Figure-26: *First part of the second decision tree*

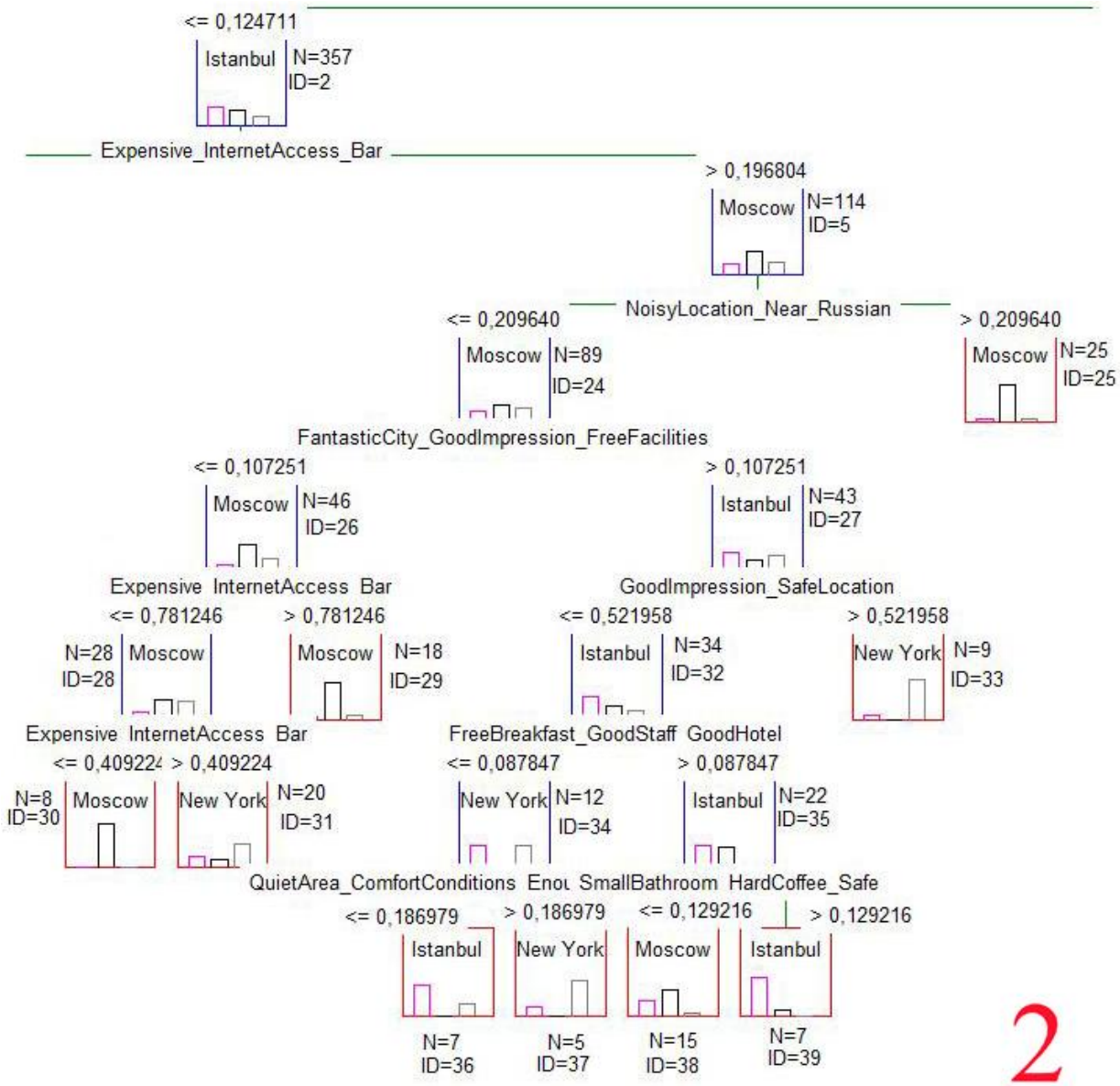


Figure-27: *Second part of the second decision tree*

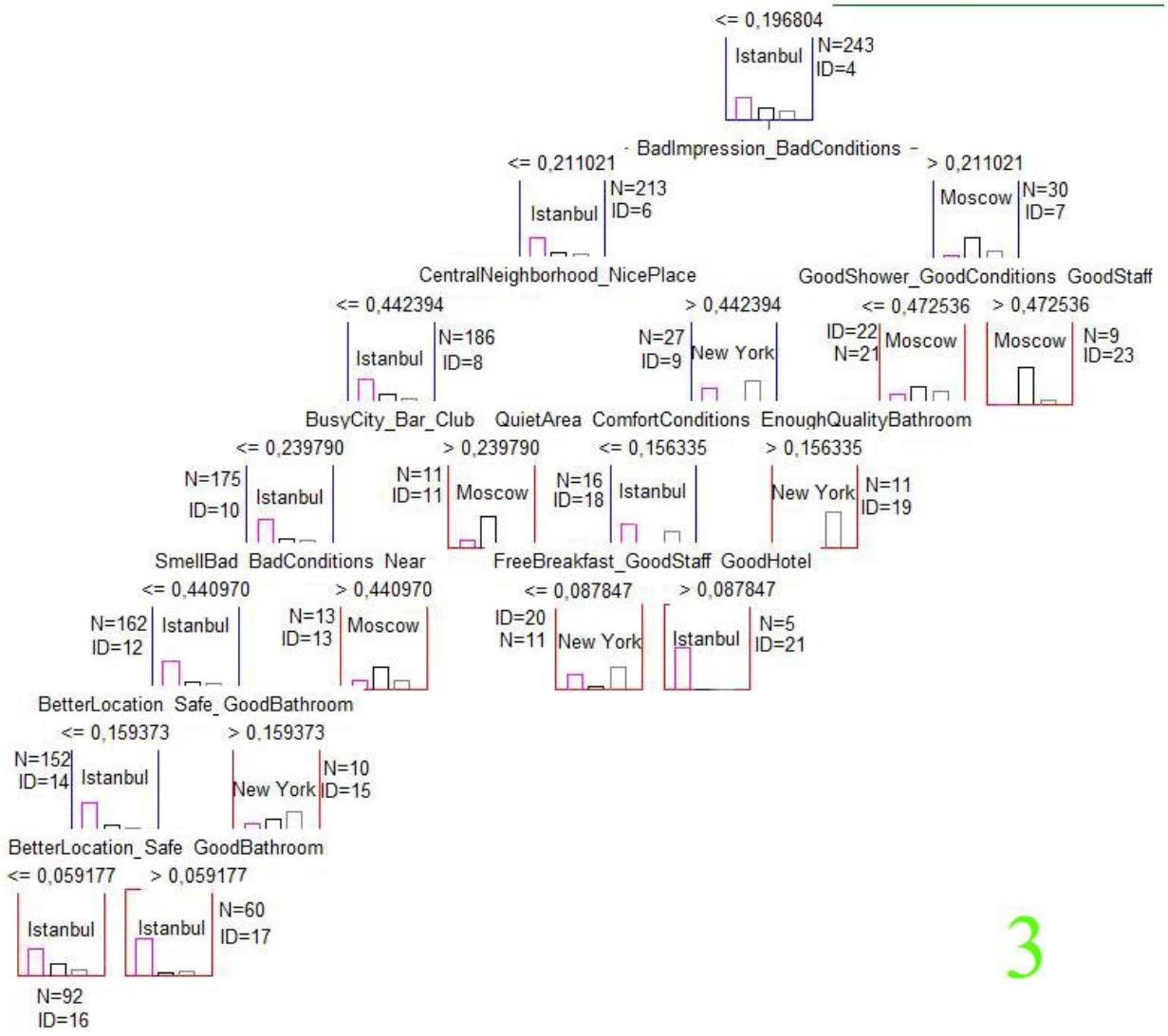


Figure-28: *Third part of the second decision tree*

	1	2	3
Model selected for deployment	1		
Model Evaluation Summary	ID	Name	Error rate (%)
	1	C&RT	29,63
Table	Step options		
	Date and time	09.08.2011 09:01:52	

Table-6: *Model Evaluation-2*

3.4 RESULTS

We created two decision tree models and used different feature spaces for each decision tree. And thus, we have seen both how to discriminate between the hotels of different cities based on the extracted inverse document frequency of variables and how to create a more accurate model by changing the method of forming a feature space. For the first model the feature space was created from individual words, and the second one was created from two-, three- or four-word phrases. The general information about the hotels for each city was very limited when a researcher looked at the first feature space. However, the information that we extracted for the second model is very satisfying. In addition, the accuracy rate of the second model is a little higher than the first model. The accuracy rate of the first model is 64.81% and the accuracy rate of the second model is 70.37%.

The information that we derived from the first model is that one of most important tourist attractions of *Istanbul* is mosques, and visitors generally have left good feedback about the hotels of the city. For *New York* City reviewers have a good impression as well, and they think hotels in this city are very comfortable even though rooms, beds or bathrooms are a little bit small. For Moscow the general impression is not so good; visitors generally left bad feedback. That is a busy and expensive city is the main problem of the city. In addition, the information that we got from the second model is that it's being a historic and fantastic city, good and safe location of hotels and having good coffee are the main advantages of *Istanbul*. Modern and comfortable hotels, nice and safe neighborhoods, being near to historic places and having good coffee are main advantages of *New York* City, and small bathrooms, expensive facilities and noise are main complaints of the visitors. Good conditions, good staffs and being near to city center are main

attractions of the hotels in Moscow, however bad smell, expensive facilities, noisy location and business of the city hotels are main complaints of the reviewers.

4. CONCLUSIONS

4.1 FUTURE WORK

This research focused on the idea of using feature selection and predictive modeling approach in text mining. We observed that using different features are changing the reliability of a predictive model. In addition, using a different feature selection technique can change the accuracy rate of the model. The dataset used in this case study was very small. We would like to do more experiments with data sets varying in size and domain. In addition, in our data set for this thesis example we had the all categories ready needed for the analysis thanks to categorical variable called '*City*'. However, in a future study we would like to extract categories ourselves from the data by using categorization methods, and thus we can have closer and wider look at the data without putting ourselves in a limited area.

REFERENCES

1- Radicati, S. & Hoang, Q. (2011). Email Statistics Report 2011-2015. The Radicati Group, Inc. A Technology Market Research Firm. From http://www.radicati.com/?p=7261

2- Graham, J. (2008). Article: "USA Today". From http://www.usatoday.com/tech/products/2008-04-15-google-gmail-webmail_N.htm

3- Harbison, A. & Ryan, P. (2009). Report: The problem of analyzing unstructured data, The limits of computers in electronic discovery.

4- Chang, G., Healey, M.J., McHugh, J.A. & Wang, Jason T.L. (2001). Book: Mining the World Wide Web, An information Search Approach.

5- Erol, U. (2009). Article: "What is Text Mining?". From http://www.metinmadenciligi.com

6- Asyali, M.F., (Computer Engineering, YTU, IST., T.R.) & Yildirim, T., (Electronic Engineering, YTU, IST., Turkey). (2004). Auto Text Categorization of News Articles.

7- Oracle®. (2003). Text Application Developer's Guide 10*g* Release 1. From http://download.oracle.com/docs/cd/B14117_01/text.101/b10729/classify.htm

8- Tonta, Y., (Hacettepe University, ANK., T.R.), Bitirim, Y., (Eastern Mediterranean University, North Cyprus, T.R.), Sever, H., (Massachusetts University, Shrewsbury, MA). (2002). Article: "Turkish Search Engine Performance Evaluation".

9- Hassler, M. & Fliedl, G. (2006). Book: Text preparation through extended tokenization.

10- Feldman R. & Sanger J. (2007). Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data, Cambridge University Press, forthcoming 2007.

11- Adsiz, A., (Ahmet Yesevi University). (2006). Dissertation: Text Mining.

12- Stackoverflow. (2008). http://blog.stackoverflow.com/2008/12/podcast-32/

13- Mustafa, A., Akbar, A. & Sultan, A. (National University of Computer and Emerging Sciences-FAST, Karachi, Pakistan). (2009). Knowledge Discovery using Text Mining: A Programmable Implementation on Information Extraction and Categorization.

14- Rustum, R.; Adeloye, A.; Simala, A. (2007) "Kohonen self-organising map (KSOM) extracted features for enhancing MLP-ANN prediction models of BOD_5", *Water Quality and Sediment Behaviour of the Future: Predictions for the 21st Century* (Proceedings of Symposium HS2005 at IUGG2007, Perugia, July 2007). IAHS Publ. 314, 181-187

15- Liangtu, S. & Xiaoming, Z. (2007). Web Text Feature Extraction with Particle Swarm Optimization.

16- Li, J., (University of Arizona). (2007). Dissertation: Feature construction, selection and consolidation for knowledge discovery.

17- Kim, Y.S., Street, W.N., & Menczer, F., (University of Iowa). (2003). Feature Selection in Data Mining.

18- Bolat, M., (Baskent University Institute of Science). (2003). Master Thesis: "Text Filtering The Fastest Descent Method Determination of the optimal query".

19- Stavrianou, A., Andritsos, P., & Nicoloyannis, N. (2007). Overview and semanticissues of text mining. ACM SIGMOD Record, 36(3):23–34

20- Salton, G. & Singhal, A., (Department of Computer Science, Cornell University). (1995). Automatic Text Browsing using Vector Space Model, May 1995.

21- Pilavcılar İ.F., (Yildiz Technical Univ. FBE, MA). (2007). Thesis : "Text Mining and Text Classification **".**

22- Ilhan, S., Duru, N., Karagöz, S., & Sagir, M., (Faculty of Engineering, Department of Computer Engineering, Kocaeli University, T.R.). (2008). Text Mining with the Question Answering System.

23- Lagus, K. and Kaski, S. (1999). Keyword selection method for characterizing text document maps. *Proceedings of ICANN99, Ninth International Conference on Artificial Neural Networks* volume 1, pages 371-376, IEE, London.

24- Kroeze, J.H. & Bothma, T.J.D., (Department of Informatics, M.C. Matthee,Department of Informatics, Department of Information Science, University of Pretoria, Pretoria). (2007). Differentiating between data-mining and text-mining terminology

25- Charniak, Eugene: *Introduction to artificial intelligence*, page 2. Addison-Wesley, 1984.

26- Mahinovs, A. & Tiwari, A., (Cranfield University). (2007). Text Classification Method Review, April 2007.

27- Sherpa, U. & Choejey, P., (Department of Information Technology, Bhutan). (2008). Dzongkha Text Normalization Algorithm.

28- Julie Beth Lovins (1968). Development of a stemming algorithm. Mechanical Translation and Computational Linguistics **11**:22–31.

29- Dragos¸ Manolescu, A., (Department of Computer Science, University of Illinois). (1998). Feature Extraction - A Pattern for Information Retrieval.

30- Spärck Jones, Karen (1972). "A statistical interpretation of term specificity and its application in retrieval".

31- Soucy, P. & Mineau, G.W., (Université Laval Québec). Beyond TFIDF Weighting for Text Categorization in the Vector Space Model.

32- Boole, George (2003) [1854]. *An Investigation of the Laws of Thought*. Prometheus Books.

33- Manning, C.D. , Raghavan, P. & Schütze, H. (2008). Introduction to Information Retrieval, Cambridge University Press.

34- Williams, G. (2010). Book: Data Mining Desktop Survival Guide by Graham Williams, 2010.
http://www.togaware.com/datamining/survivor

35- Ding, Q. & Sundarraj, G. (2006). "Association Rule Mining from XML Data", *International Conference on Data Mining*, Las Vegas, Nevada, 2006, pp. 144-150.

36- Wikipedia. (2011). Weighting. Retrieved May 16, 2011, from http://en.wikipedia.org/wiki/Weighting

37- Han, J. & Kamber, M. (2001). *Data Mining*. Morgan Kaufmann Publishers, San Francisco, CA

38- Manning, C.D. , Raghavan, P. & Schütze, H. (2008). Introduction to Information Retrieval, Cambridge University Press.
http://nlp.stanford.edu/IR-book/html/htmledition/naive-bayes-text-classification-1.html

39- Park, M.Y., (Department of Statistics) & Hastie, T., (Department of Statistics and Department of Health Research & Policy). (2006). L_1 Regularization Path Algorithm for Generalized Linear Models. Stanford University, November 12, 2006

40- Gill, J. (2000). *Generalized Linear Models*: A Unified Approach. (Sage University Paper Series on Quantitative Applications in the Social Sciences. Series No: 07-134). Thousand Oaks, CA: Sage.

41- Schulte, S. (2007). Introduction to Corpus Resources, Annotation and Access. Pompeu University, Fabra, April 16-20, 2007.

42- Boswell, D. (2002). Introduction to Support Vector Machines,Agst-6, 2002, http://dustwell.com/PastWork/IntroToSVM.pdf

43- Ben-Hur, A., (Department of Computer Science Colorado State University) & Weston, J., (NEC Labs America Princeton, NJ 08540 USA), A User's Guide to Support Vector Machines.
http://pyml.sourceforge.net/doc/howto.pdf

44- Press, WH; Teukolsky, SA; Vetterling, WT; Flannery, BP (2007). "Section 16.5. Support Vector Machines".

45- Agrawal, R. and Srikant, R.. Fast algorithms for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.

46- Jin, R., McCallen, S., Breitbart, Y., Fuhry, D. & Wang D., Department of Computer Science, Kent State University. Estimating the Number of Frequent Itemsets in a Large Database.

47- Zaiane, O.R., University of Alberta. (2001). Web Technologies and Applications, Lecture Notes, Winter 2001.
http://webdocs.cs.ualberta.ca/~zaiane/courses/cmput499/slides/Lect10

48- MacQueen, J. B. (1967). "Some Methods for classification and Analysis of Multivariate Observations". **1**. Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press. pp. 281–297

49- Moore, A. (2001). Carnegie Mellon University. K-means and Hierarchical Clustering - Tutorial Slides, November 16th 2001.
http://www-2.cs.cmu.edu/~awm/tutorials/kmeans.html

50- Kumar, V. (2002). Lecture Notes. Parallel Issues in Data Mining, VECPAR 2002.
http://www.cs.uvm.edu/~xwu/kdd/Slides/Kmeans-ICDM06.pdf

51- JAIN, A.K., *Michigan State University*, MURTY, M.N., *Indian Institute of Science* & FLYNN, P.J., *The Ohio State University* ACM Computing Surveys. (1999). Data Clustering: A Review, Vol. 31, No. 3, September 1999

52- Moore, A. (2001). Carnegie Mellon University. K-means and Hierarchical Clustering - Tutorial Slides, November 16th 2001. http://www-2.cs.cmu.edu/~awm/tutorials/kmeans.html

53- Wittman, T. (2011). Lecture Notes,20-May-2011, Department of Mathematics University of California, Los Angeles, wittman@math.ucla.edu

54- Deng,H.; Runger, G.; Tuv, E. (2011). "Bias of importance measures for multi-valued attributes and solutions". Proceedings of the 21st International Conference on Artificial Neural Networks (ICANN).

55- StatSoft, Inc. (2011). Electronic Statistics Textbook. Tulsa, OK: StatSoft. http://www.statsoft.com/textbook/

56- Cowie, J., Wilks., Y., Dale, R., Moisl, H. and Somers, H.. (2000). Information Extraction, Handbook of Natural Language Processing. New York: Marcel Dekker.

APPENDIX

Summary Statistics For The Data

City	Var	N	Mean	Std Dev	Min	Max
Istanbul	access	202	0.1915414	0.7855363	0	4.8938248
	accommodate	202	0.0929341	0.5324825	0	3.1287828
	adequate	202	0.0351848	0.3527223	0	3.5536660
	amazing	202	0.1846664	0.7714112	0	3.3911471
	amenity	202	0.0567613	0.4634428	0	3.8219300
	area	202	0.2915596	0.8368777	0	4.3710195
	around	202	0.1229633	0.5231145	0	3.5965823
	attractive	202	0.3060438	0.9255118	0	3.0910425
	available	202	0.0928895	0.6020172	0	5.5803369
	average	202	0.0326320	0.3271312	0	3.2958369
	avoid	202	0.0579501	0.4731487	0	3.9019727
	away	202	0.1171213	0.5246095	0	3.7460747
	bad	202	0.0412271	0.3366099	0	2.7759614
	bar	202	0.1564848	0.6142561	0	3.7202246
	bathroom	202	0.1994433	0.6272730	0	3.7602084
	beauties	202	0.2124148	0.7518609	0	2.8605188
	bed	202	0.1768514	0.5590562	0	3.1406367
	best	202	0.2822854	0.8135764	0	3.7460747
	better	202	0.1880388	0.6026953	0	2.1102132
	big	202	0.1191077	0.5529363	0	2.6733073
	breakfast	202	0.4749511	0.6006779	0	1.7848489
	busy	202	0.0593538	0.3734862	0	2.3978953
	cafe	202	0.0378409	0.3793491	0	3.8219300
	center	202	0.0667822	0.4202293	0	2.6979999
	central	202	0.0578444	0.4079827	0	2.9211434
	cheap	202	0	0	0	0
	clean	202	0.3255474	0.5559429	0	2.0214865
	cleanliness	202	0.0579501	0.4731487	0	3.9019727
	close	202	0.2955767	0.7726284	0	3.6168328
	club	202	0.0193167	0.2745418	0	3.9019727
	coffee	202	0.0655745	0.4126299	0	2.6492097
	cold	202	0.0292366	0.2930919	0	2.9528921
	comfort	202	0.2507551	0.6399774	0	1.8760198
	compared	202	0.0970214	0.6287958	0	5.8285582
	complain	202	0.1852787	0.6806410	0	2.6733073

City	Var	N	Mean	Std Dev	Min	Max
	complimentary	202	0.0511254	0.4174264	0	3.4424403
	convenient	202	0.2437639	0.7458401	0	4.0289218
	courteous	202	0.0772668	0.5449706	0	3.9019727
	crowd	202	0.0182120	0.2588415	0	3.6788291
	decent	202	0.0942006	0.6105146	0	5.6591021
	decor	202	0.0160960	0.2287667	0	3.2513851
	dinner	202	0.1627272	0.7714343	0	5.7417111
	dirty	202	0	0	0	0
	disappoint	202	0.0511254	0.4174264	0	3.4424403
	distance	202	0.2382155	0.7635198	0	2.6733073
	efficient	202	0.0378409	0.3793491	0	3.8219300
	enjoy	202	0.2230234	0.8090037	0	4.6549580
	enough	202	0.0967903	0.5696956	0	4.9459257
	everything	202	0.2140797	0.6660425	0	2.2760055
	example	202	0.0579501	0.4731487	0	3.9019727
	excelent	202	0.3521219	0.8553017	0	4.4578711
	expense	202	0.0846953	0.4276899	0	3.3321792
	experience	202	0.1480884	0.6300027	0	4.3314733
	extremly	202	0.1131762	0.5774341	0	4.1238896
	facilities	202	0.0378409	0.3793491	0	3.8219300
	fantastic	202	0.2077705	0.8983272	0	5.4330138
	far	202	0.0674089	0.4241727	0	2.7233177
	fine	202	0.0867666	0.4971447	0	2.9211434
	food	202	0.1544156	0.5672620	0	2.2279962
	free	202	0.2296996	0.7299770	0	3.8536122
	fresh	202	0.0378409	0.3793491	0	3.8219300
	fun	202	0.0579501	0.4731487	0	3.9019727
	good	202	0.4014526	0.6395413	0	2.6580450
	great	202	0.4791774	0.7540473	0	3.0711916
	happy	202	0.0728481	0.5138052	0	3.6788291
	hard	202	0.1182448	0.5837349	0	2.9856819
	heat	202	0.0160960	0.2287667	0	3.2513851
	help	202	0.3519983	0.6221304	0	2.2103612
	high	202	0.2026327	0.6926019	0	2.5582379
	historic	202	0.2003323	0.8368525	0	3.6788291
	hot	202	0.0934950	0.4946916	0	2.6979999
	impress	202	0.1224175	0.6043341	0	3.0910425
	included	202	0.1639226	0.7048733	0	4.5263026
	incredibly	202	0.1324431	0.7007697	0	3.8219300

City	Var	N	Mean	Std Dev	Min	Max
	internet	202	0.1493607	0.6354153	0	4.3686869
	istanbul	202	0.7642192	1.2391020	0	5.4706628
	kind	202	0.0728481	0.5138052	0	3.6788291
	lack	202	0.0371071	0.3719934	0	3.7478220
	ladie	202	0	0	0	0
	large	202	0.1131762	0.5774341	0	4.1238896
	leaves	202	0.0728481	0.5138052	0	3.6788291
	less	202	0.0459066	0.3748163	0	3.0910425
	like	202	0.2718769	0.6749498	0	2.7221672
	little	202	0.2025209	0.6635285	0	3.5729015
	locate	202	0.4076213	0.3877056	0	1.4902781
	lot	202	0.1332918	0.5317006	0	2.2437446
	loud	202	0.0189204	0.2689101	0	3.8219300
	love	202	0.2043736	0.6985523	0	2.5802168
	low	202	0.0193167	0.2745418	0	3.9019727
	manhattan	202	0	0	0	0
	meal	202	0.0357851	0.3587396	0	3.6142906
	modern	202	0.0940991	0.5391574	0	3.1680035
	moscow	202	0	0	0	0
	mosque	202	0.7041522	1.3355466	0	6.5645064
	near	202	0.1456247	0.6195218	0	4.2594136
	neighborhood	202	0.1377197	0.6393390	0	3.0910425
	never	202	0.0561455	0.4090057	0	4.0916382
	NewYork	202	0	0	0	0
	nice	202	0.4365459	0.7746338	0	2.9593488
	noise	202	0.0309780	0.3105501	0	3.1287828
	noisy	202	0.0357851	0.3587396	0	3.6142906
	ny	202	0	0	0	0
	old	202	0.2381244	0.6729919	0	3.4373775
	palace	202	0.2726104	1.0169095	0	6.3456142
	pleasant	202	0.0794264	0.4997934	0	3.2088255
	pool	202	0.0715701	0.5047914	0	3.6142906
	poor	202	0.0635411	0.4481619	0	3.2088255
	positive	202	0.0496390	0.4052904	0	3.3423569
	problem	202	0.1315963	0.5497204	0	2.4165874
	provided	202	0.1056282	0.6845768	0	6.3456142
	quality	202	0.1287789	0.5978330	0	2.8903718
	quiet	202	0.1208197	0.6751104	0	5.8256667
	recommend	202	0.3495853	0.7419970	0	2.9831670

City		Var	N	Mean	Std Dev	Min	Max
		relax	202	0.1352169	0.7154460	0	3.9019727
		renovated	202	0.0572032	0.4834627	0	5.2974897
		reservation	202	0.0912271	0.6876317	0	7.5849947
		restaurant	202	0.3430830	0.7000015	0	2.6388809
		return	202	0.1161095	0.5863235	0	4.5681109
		rude	202	0.0167879	0.2386003	0	3.3911471
		russia	202	0	0	0	0
		russian	202	0	0	0	0
		safe	202	0.1058551	0.5600897	0	3.0546748
		secure	202	0.0635411	0.4481619	0	3.2088255
		serve	202	0.2730130	0.6283295	0	2.4469988
		shop	202	0.1883712	0.6667565	0	2.5367317
		shower	202	0.0864232	0.4266424	0	2.1821867
		sight	202	0.1631602	0.7167086	0	3.2958369
		slept	202	0.0579501	0.4731487	0	3.9019727
		small	202	0.3137365	0.6882676	0	3.2541028
		smell	202	0.0306044	0.3068042	0	3.0910425
		spacious	202	0.0942006	0.6105146	0	5.6591021
		staff	202	0.3714293	0.4212431	0	1.3349744
		surprised	202	0.0335757	0.3365913	0	3.3911471
		temperature	202	0	0	0	0
		terrible	202	0.0302947	0.4305683	0	6.1195259
		traffic	202	0.0346189	0.3470490	0	3.4965076
		treat	202	0.0756818	0.5337914	0	3.8219300
		tried	202	0	0	0	0
		turkish	202	0.2983311	1.1314809	0	6.3456142
		uncomfortable	202	0.0546361	0.4460906	0	3.6788291
		want	202	0.1983728	0.6424939	0	3.9141024
		warm	202	0.0865472	0.5446016	0	3.4965076
		worst	202	0.0163160	0.2318943	0	3.2958369
		worth	202	0.0692378	0.4883412	0	3.4965076
Moscow	199	access	199	0.1466771	0.7083422	0	6.0657697
		accommodate	199	0.1257802	0.6161379	0	3.1287828
		adequate	199	0.0714305	0.4999940	0	3.5536660
		amazing	199	0.0511228	0.4142630	0	3.3911471
		amenity	199	0.0384114	0.3821833	0	3.8219300
		area	199	0.2415587	0.7239871	0	3.5265110
		around	199	0.3596531	0.8152146	0	3.5965823

City	Var	N	Mean	Std Dev	Min	Max
	attractive	199	0.0465986	0.3776022	0	3.0910425
	available	199	0.0993720	0.5650161	0	3.2958369
	average	199	0.2946670	1.0390782	0	6.9166838
	avoid	199	0.1372553	0.7206506	0	3.9019727
	away	199	0.3044822	0.8011982	0	3.7460747
	bad	199	0.3415535	1.0083456	0	5.8256667
	bar	199	0.3771595	0.9282326	0	3.7202246
	bathroom	199	0.2493440	0.6368386	0	3.0337125
	beauties	199	0.0962104	0.5619598	0	4.8432793
	bed	199	0.3885798	0.7999160	0	3.5711616
	best	199	0.2377739	0.7279679	0	3.7460747
	better	199	0.3412189	0.8588929	0	4.4285194
	big	199	0.2872972	0.8795882	0	4.5263026
	breakfast	199	0.5122900	0.6722598	0	2.7507664
	busy	199	0.4850917	1.1099175	0	5.7220840
	cafe	199	0.1152341	0.6552060	0	3.8219300
	center	199	0.3861923	1.0336169	0	5.6620557
	central	199	0.1321120	0.6085437	0	2.9211434
	cheap	199	0.2346529	0.9447148	0	6.2287992
	clean	199	0.3549303	0.5915927	0	2.0214865
	cleanliness	199	0.0784316	0.5490001	0	3.9019727
	close	199	0.3781432	0.8176088	0	3.6168328
	club	199	0.2156869	0.8939235	0	3.9019727
	coffee	199	0.2699354	0.9837676	0	6.3217942
	cold	199	0.2728279	0.9155630	0	4.9996810
	comfort	199	0.1508358	0.5114036	0	1.8760198
	compared	199	0.1729870	0.7539424	0	3.4424403
	complain	199	0.1074696	0.5264431	0	2.6733073
	complimentary	199	0.0709006	0.6150127	0	7.2243476
	convenient	199	0.1674053	0.6100773	0	2.3795461
	courteous	199	0.0196079	0.2766035	0	3.9019727
	crowd	199	0.1663792	0.7663877	0	3.6788291
	decent	199	0.1528181	0.7873773	0	7.0143112
	decor	199	0.1307089	0.6402815	0	3.2513851
	dinner	199	0.1888041	0.9316477	0	8.0922751
	dirty	199	0.1878828	0.8350074	0	5.9201019
	disappoint	199	0.0691948	0.4843447	0	3.4424403
	distance	199	0.0940359	0.4937298	0	2.6733073
	efficient	199	0.1861635	0.8756974	0	6.4710899

City	Var	N	Mean	Std Dev	Min	Max
	enjoy	199	0.0552622	0.3868203	0	2.7492932
	enough	199	0.1863241	0.7501428	0	4.9459257
	everything	199	0.1944326	0.6377850	0	2.2760055
	example	199	0.1372553	0.7206506	0	3.9019727
	excelent	199	0.2258577	0.7690199	0	3.5965823
	expense	199	0.6809219	1.0297334	0	4.6963197
	experience	199	0.2195642	0.7952716	0	4.3314733
	extremly	199	0.2702386	0.8328365	0	4.1238896
	facilities	199	0.1152341	0.6552060	0	3.8219300
	fantastic	199	0.0644990	0.4514756	0	3.2088255
	far	199	0.3171920	0.9926206	0	5.7151879
	fine	199	0.2507110	0.9080870	0	4.9459257
	food	199	0.4429609	0.9998049	0	4.6757003
	free	199	0.2481092	0.7323832	0	3.8536122
	fresh	199	0.1728511	0.7961990	0	3.8219300
	fun	199	0.0784316	0.5490001	0	3.9019727
	good	199	0.4734004	0.6852885	0	2.6580450
	great	199	0.2067616	0.5498449	0	2.7009411
	happy	199	0.0497871	0.5116741	0	6.2287992
	hard	199	0.1754373	0.7405060	0	5.0551990
	heat	199	0.2187136	0.8899195	0	5.5050735
	help	199	0.3167034	0.6202609	0	2.7396858
	high	199	0.1542656	0.6105106	0	2.5582379
	historic	199	0.0554597	0.4494063	0	3.6788291
	hot	199	0.2357197	0.8183098	0	4.5681109
	impress	199	0.1505625	0.7082332	0	5.2335898
	included	199	0.2938113	0.9773990	0	5.6102355
	incredibly	199	0.0576170	0.4668875	0	3.8219300
	internet	199	0.3354438	1.0214562	0	7.2033448
	istanbul	199	0	0	0	0
	kind	199	0.1237334	0.7227199	0	6.2287992
	lack	199	0.1129997	0.6425014	0	3.7478220
	ladie	199	0.2112625	0.8755861	0	3.8219300
	large	199	0.3837792	1.0679731	0	5.1114548
	leaves	199	0.1478926	0.7244562	0	3.6788291
	less	199	0.2485260	0.8426192	0	3.0910425
	like	199	0.4720220	0.9178816	0	3.3740561
	little	199	0.2849445	0.8247154	0	4.4285194
	locate	199	0.2930738	0.4055590	0	1.4902781

City	Var	N	Mean	Std Dev	Min	Max
	lot	199	0.3177103	0.8735867	0	4.7087500
	loud	199	0.0576170	0.4668875	0	3.8219300
	love	199	0.0648296	0.4048396	0	2.5802168
	low	199	0.1176474	0.6689280	0	3.9019727
	manhattan	199	0	0	0	0
	meal	199	0.1886386	0.8968451	0	6.1195259
	modern	199	0.1432765	0.6599706	0	3.1680035
	moscow	199	1.1324855	1.4165888	0	4.9533471
	mosque	199	0	0	0	0
	near	199	0.2956402	0.8571292	0	4.2594136
	neighborhood	199	0.0310658	0.3090964	0	3.0910425
	never	199	0.3907592	0.9835925	0	5.0714800
	NewYork	199	0	0	0	0
	nice	199	0.1974998	0.5576803	0	2.9593488
	noise	199	0.1273306	0.7057591	0	6.5661020
	noisy	199	0.1452981	0.7117469	0	3.6142906
	ny	199	0.0196485	0.2771765	0	3.9100557
	old	199	0.3667635	0.9122278	0	4.2605407
	palace	199	0	0	0	0
	pleasant	199	0.0967485	0.5500995	0	3.2088255
	pool	199	0.2249632	0.9597821	0	6.1195259
	poor	199	0.2691728	0.9253656	0	5.4330138
	positive	199	0.2015492	0.7976367	0	3.3423569
	problem	199	0.3701982	0.9479590	0	5.0714800
	provided	199	0.1072206	0.6896191	0	6.3456142
	quality	199	0.2290887	0.9004558	0	4.8938248
	quiet	199	0.1255460	0.5782989	0	2.7759614
	recommend	199	0.2547468	0.6778523	0	2.9831670
	relax	199	0.0588237	0.4766655	0	3.9019727
	renovated	199	0.2515604	0.8529073	0	3.1287828
	reservation	199	0.0908113	0.5670871	0	3.6142906
	restaurant	199	0.3684892	0.7733709	0	3.2708249
	return	199	0.1043021	0.5612844	0	4.5681109
	rude	199	0.3146735	1.1702984	0	8.0922751
	russia	199	0.4012570	1.2298280	0	7.0143112
	russian	199	0.4163882	1.1978475	0	6.8233967
	safe	199	0.1074509	0.5641641	0	3.0546748
	secure	199	0.2852976	0.9483241	0	5.4330138
	serve	199	0.5569830	0.8786222	0	4.0347538

City		Var	N	Mean	Std Dev	Min	Max
		shop	199	0.1833878	0.7141697	0	4.2950602
		shower	199	0.2116676	0.7405614	0	5.6942807
		sight	199	0.0662480	0.4637179	0	3.2958369
		slept	199	0.0588237	0.4766655	0	3.9019727
		small	199	0.1690318	0.4989582	0	2.6253896
		smell	199	0.2234605	0.8702255	0	5.2335898
		spacious	199	0.0671831	0.4702632	0	3.3423569
		staff	199	0.3331742	0.4630639	0	1.8814914
		surprised	199	0.1363275	0.6678042	0	3.3911471
		temperature	199	0.1992622	0.8410691	0	5.7417111
		terrible	199	0.2612877	1.0482253	0	8.6247613
		traffic	199	0.2352024	0.9570107	0	5.9201019
		treat	199	0.1093408	0.7032553	0	6.4710899
		tried	199	0.2608635	0.9387076	0	5.3638962
		turkish	199	0	0	0	0
		uncomfortable	199	0.1294060	0.6794384	0	3.6788291
		want	199	0.3136514	0.7695198	0	3.1578731
		warm	199	0.1527420	0.7653608	0	5.9201019
		worst	199	0.3045137	1.0218952	0	5.5803369
		worth	199	0.1473505	0.8005753	0	5.9201019
NewYork	193						
		access	193	0.1900928	0.7530222	0	4.8938248
		accommodate	193	0.1945357	0.7574885	0	3.1287828
		adequate	193	0.2153033	0.8942573	0	6.0168795
		amazing	193	0.1054243	0.5900839	0	3.3911471
		amenity	193	0.1721482	0.8490359	0	6.4710899
		area	193	0.3578435	0.8829410	0	3.5265110
		around	193	0.3047967	0.7648642	0	3.5965823
		attractive	193	0.0640631	0.4415060	0	3.0910425
		available	193	0.2184433	0.9341163	0	6.9166838
		average	193	0.0683075	0.4707576	0	3.2958369
		avoid	193	0.0404349	0.3961746	0	3.9019727
		away	193	0.3960152	0.9810558	0	3.7460747
		bad	193	0.1825683	0.7232151	0	4.7001112
		bar	193	0.3091926	0.8739939	0	3.7202246
		bathroom	193	0.5508680	0.9498899	0	3.7602084
		beauties	193	0.2132242	0.8169694	0	4.8432793
		bed	193	0.6529204	0.9343479	0	3.9051026
		best	193	0.2773895	0.8106388	0	4.6431630

City	Var	N	Mean	Std Dev	Min	Max
	better	193	0.3156704	0.8554429	0	4.4285194
	big	193	0.1950190	0.7854177	0	4.5263026
	breakfast	193	0.2804009	0.5408973	0	2.2122742
	busy	193	0.1911644	0.7030917	0	4.0599896
	cafe	193	0.1127399	0.7138894	0	6.4710899
	center	193	0.1258135	0.5703518	0	2.6979999
	central	193	0.3400292	1.0691200	0	4.9459257
	cheap	193	0.0836084	0.6838179	0	6.2287992
	clean	193	0.4835930	0.6611819	0	3.1154666
	cleanliness	193	0.1010874	0.6214682	0	3.9019727
	close	193	0.3757586	0.7506713	0	2.9180380
	club	193	0	0	0	0
	coffee	193	0.3223650	0.9612886	0	5.5596640
	cold	193	0.1835995	0.7149048	0	2.9528921
	comfort	193	0.5176794	0.9371153	0	3.9370382
	compared	193	0.0713459	0.4916976	0	3.4424403
	complain	193	0.2975946	0.9401411	0	5.6102355
	complimentary	193	0.2566008	0.9831673	0	5.8285582
	convenient	193	0.2674603	0.7761777	0	4.0289218
	courteous	193	0.1415223	0.7314092	0	3.9019727
	crowd	193	0.1085187	0.6871599	0	6.2287992
	decent	193	0.1505472	0.7424995	0	5.6591021
	decor	193	0.2543596	0.9451120	0	6.8233967
	dinner	193	0.0351414	0.3443095	0	3.3911471
	dirty	193	0.1449330	0.6987729	0	3.4965076
	disappoint	193	0.2264010	0.8968514	0	5.8285582
	distance	193	0.2216213	0.7390371	0	2.6733073
	efficient	193	0.0396055	0.3880477	0	3.8219300
	enjoy	193	0.3002775	0.9369208	0	4.6549580
	enough	193	0.2223875	0.8096011	0	4.9459257
	everything	193	0.3193417	0.8503474	0	3.8536122
	example	193	0.0404349	0.3961746	0	3.9019727
	excelent	193	0.2728634	0.7678263	0	4.4578711
	expense	193	0.1192361	0.4952443	0	3.3321792
	experience	193	0.2742901	0.8183709	0	4.3314733
	extremly	193	0.3203718	0.8681723	0	4.1238896
	facilities	193	0.0990137	0.6087197	0	3.8219300
	fantastic	193	0.1496343	0.6783393	0	3.2088255
	far	193	0.1975464	0.7082059	0	2.7233177

City	Var	N	Mean	Std Dev	Min	Max
	fine	193	0.1936091	0.8279195	0	6.1303475
	food	193	0.2018461	0.7289672	0	4.6757003
	free	193	0.2933005	0.8794416	0	4.7764530
	fresh	193	0.0533317	0.5397396	0	6.4710899
	fun	193	0.1010874	0.6214682	0	3.9019727
	good	193	0.4088517	0.6363671	0	3.1096844
	great	193	0.6351467	0.8857871	0	3.0711916
	happy	193	0.1979762	0.9263723	0	6.2287992
	hard	193	0.1701684	0.6939790	0	2.9856819
	heat	193	0.2559490	1.1768744	0	8.4842876
	help	193	0.4988024	0.7271413	0	2.7396858
	high	193	0.2385921	0.7458765	0	2.5582379
	historic	193	0.0190613	0.2648079	0	3.6788291
	hot	193	0.3127901	1.0703584	0	5.6620557
	impress	193	0.1712589	0.7490297	0	5.2335898
	included	193	0.1854180	0.7423402	0	4.5263026
	incredibly	193	0.0594082	0.4740137	0	3.8219300
	internet	193	0.1696947	0.6722182	0	4.3686869
	istanbul	193	0	0	0	0
	kind	193	0.0953065	0.5859280	0	3.6788291
	lack	193	0.1165126	0.6521479	0	3.7478220
	ladie	193	0.0396055	0.3880477	0	3.8219300
	large	193	0.2660199	0.8300307	0	4.1238896
	leaves	193	0.0571839	0.4562657	0	3.6788291
	less	193	0.1503287	0.7491197	0	5.2335898
	like	193	0.4193239	0.8289211	0	3.8365787
	little	193	0.3919970	0.9314378	0	3.5729015
	locate	193	0.4718406	0.4551123	0	1.6945685
	lot	193	0.3344992	0.8746760	0	3.7989898
	loud	193	0.1919509	0.8886468	0	6.4710899
	love	193	0.3726101	1.0342057	0	5.4148748
	low	193	0.1010874	0.6214682	0	3.9019727
	manhattan	193	0.3386566	1.0722527	0	5.9201019
	meal	193	0.1271886	0.8261604	0	7.5849947
	modern	193	0.1869006	0.8262960	0	5.3638962
	moscow	193	0.0098355	0.1366385	0	1.8982430
	mosque	193	0	0	0	0
	near	193	0.1955190	0.6752762	0	2.5156783
	neighborhood	193	0.2673535	0.9040228	0	5.2335898

City	Var	N	Mean	Std Dev	Min	Max
	never	193	0.2591026	0.7730574	0	4.0916382
	NewYork	193	0.4833246	1.2477489	0	6.4868997
	nice	193	0.6238391	0.9234314	0	3.9367873
	noise	193	0.3046390	1.0208885	0	6.5661020
	noisy	193	0.1253419	0.7207199	0	6.1195259
	ny	193	0.8592526	1.4013790	0	4.8464132
	old	193	0.3133352	0.8694671	0	5.2976039
	palace	193	0	0	0	0
	pleasant	193	0.2276628	0.8632337	0	5.4330138
	pool	193	0.0187269	0.2601623	0	3.6142906
	poor	193	0.0665042	0.4583294	0	3.2088255
	positive	193	0.1039075	0.5815941	0	3.3423569
	problem	193	0.1839755	0.6697623	0	4.0916382
	provided	193	0.1124688	0.8157934	0	7.8652253
	quality	193	0.1751168	0.7273435	0	4.8938248
	quiet	193	0.3020476	0.8666785	0	2.7759614
	recommend	193	0.3469039	0.7024432	0	1.7619065
	relax	193	0.0404349	0.3961746	0	3.9019727
	renovated	193	0.1134792	0.5864779	0	3.1287828
	reservation	193	0.1906033	1.0258523	0	7.5849947
	restaurant	193	0.3977774	0.7631498	0	3.2708249
	return	193	0.3745293	1.0033165	0	5.6620557
	rude	193	0.0702828	0.4843711	0	3.3911471
	russia	193	0	0	0	0
	russian	193	0	0	0	0
	safe	193	0.2215826	0.7943762	0	3.0546748
	secure	193	0.0614024	0.5071525	0	5.4330138
	serve	193	0.3747825	0.7367569	0	3.0329919
	shop	193	0.2497301	0.7576990	0	2.5367317
	shower	193	0.5735107	1.1629382	0	5.2073398
	sight	193	0.1366150	0.6586691	0	3.2958369
	slept	193	0.1213048	0.6789712	0	3.9019727
	small	193	0.6196661	0.8925322	0	3.7001819
	smell	193	0.2097843	0.8498959	0	6.4868997
	spacious	193	0.2198188	0.8707769	0	5.6591021
	staff	193	0.4870585	0.4787021	0	2.0574305
	surprised	193	0.1757071	0.7536035	0	3.3911471
	temperature	193	0.1703155	0.7884843	0	5.7417111
	terrible	193	0.0374538	0.3669657	0	3.6142906

City	Var	N	Mean	Std Dev	Min	Max
	traffic	193	0.0724665	0.4994202	0	3.4965076
	treat	193	0.0929372	0.6615538	0	6.4710899
	tried	193	0.1641453	0.7040150	0	3.1680035
	turkish	193	0	0	0	0
	uncomfortable	193	0.0953065	0.5859280	0	3.6788291
	want	193	0.4596468	0.8930045	0	3.9141024
	warm	193	0.1156981	0.7366464	0	5.9201019
	worst	193	0.0683075	0.4707576	0	3.2958369
	worth	193	0.1268163	0.6554064	0	3.4965076