Visualizing Political Sentiment Data Using Machine Learning Techniques

by

Liz Nabasajj Joseph

10075268

Submitted in partial fulfillment of the requirements for the degree
Bachelor of Science (Honours) (Computer Science)
in the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria, Pretoria

April 2016

Publication data:

Liz Nabasajj Joseph. Visualizing Political Sentiment Data Using Machine Learning Techniques. Honours Mini-dissertation, University of Pretoria, Department of Computer Science, Pretoria, South Africa, April 2016.

Electronic, hyperlinked versions of this thesis are available online, as Adobe PDF files, at:

http://cirg.cs.up.ac.za/

Visualizing Political Sentiment Data Using Machine Learning Techniques

by

Liz Nabasajj Joseph
E-mail: lizjoseph@tuks.co.za

Abstract

Social media has gained popularity in the past few years and has made it into the homes of millions of users. Social media users have taken to these multiple platforms to share their sentiments regarding their lives, daily habits and political opinions. It has also made a big contribution in terms of data on the web and is now used in research and businesses to develop marketing strategies. Big data is difficult to analyze and visualize as it is massive in size, contains noise and duplicates. It is a challenge to extract useful information from it. The interest in data mining has led to the creation of analysis techniques and the use of machine learning for interpreting texts and displaying results. This study investigates the use of self-organizing maps for the classification and visualization of sentiment data using the tool SOM-PAK.

Keywords: self-organizing maps, artificial intelligence, data visualization, sentiment analysis, data mining

Supervisors : Dr. J. Mwaura
Mr. W. van Heerden
Department : Department of Computer Science
Degree : Bachelor of Science (Honours)

"Not everything that can be counted counts, and not everything that counts can be counted."

Albert Einstein, Physicist

"If you torture the data long enough, it will confess... "

Ronald Coase, Economist

Contents

List of Figures

List of Graphs

List of Algorithms

List of Tables

Chapter 1

Introduction

The stance of South African politics has been in dispute following the end of the apartheid regime in the early 1990's. After the national elections in 1994, a political party that racially represented the majority of South Africans came into power and has stayed elected over the past 22 years. Recent uproars in parliament and presidential corruption has left South Africans taking to social media to express their true sentiments with respect to the current party and its competitors despite how they vote. It would be of great interest to capture and analyze their opinions.

From history, computer science was mainly used for storage and data processing purposes. In recent years, a demand for computational analysis has introduced a discipline titled data science, which is a domain that allows experts to build systems that hold and realize the vast amounts of data used to make key decisions [1]. Within the discipline of data science is a data analysis field called data mining [3] which entails the gathering of information from a reliable source. This is followed by data analytics and interpretation, the summation of that data into knowledge and the representation of that information either in a visual format for example a graph or in a worded format. This process can be aided with the use of Artificial Intelligence (AI)[REFERENCE].

1.1 Motivation

There is a gap in the research of self-organizing maps (SOM) introduced in 1982 by Teuvo Kohonen[2] for sentiment analysis[REFERENCE] in the field of politics in Africa and the visualization of this data. SOMs have been used in data clustering, analysis and they exhibit excellent visualization abilities[REFERENCE].The SOM has seldom been used with the intent to analyze polarity sentiments from social media data.The intent of this work is to classify the given data and to visually depict and interpret the associations and relationships formed by the clusters.

1.2 Objectives

The objectives of this report are as follows:

- Perform a literature survey on existing machine learning data mining techniques.
- Perform a general case study for the SOM algorithm in political sentiment analysis.
- Gather tweets from Twitter and store them into a database.
- Pre-process the data by removing stop words, symbols and links.
- Normalize the clean textual data and place it into vectors.
- Initialize the reference vectors of the map.
- Train the map using the SOM algorithm.
- Use the trained map to visualize data.
- Characterize the nature of the data.

1.3 Contributions

Big data is a term coined to describe the massive sets of data containing the worlds information[5]. The representation of data has been the matter in question among various

big data organizations. The numerous questions that arise are " How do we derive meaningful information from big data?", " What is the best way to visually represent big data?". Among various classification techniques such as the Naive Bayes[4], is the SOM. The proposed paper will demonstrate the unsupervised SOMs ability to address the above questions.

1.4 Thesis Outline

The remaining chapters will read as follows:

- **Chapter 2** provides an overview of the related work for this topic.
- **Chapter 3** provides informative background knowledge of the main topics.
- **Chapter 4** gives an in-depth introduction to self organizing maps.
- **Chapter 5** discusses visualization using self-organizing maps.
- **Chapter 6** lists the methods used to perform the experiment.
- **Chapter 7** discusses the results obtain from the conducted experiments.
- **Chapter 9** gives an overall summary of the work.

List of appendices:

- **Appendix A** describes ....

Chapter 2

Background

This chapter focuses on related works.

Section 2.1.1 gives a brief background on Big Data. Section 2.1.2 introduces artificial intelligence and neural networks, Section 2.1.3 describes self-organizing maps, Section 2.1.4 discusses sentiment analysis and the methods of opinion mining.

2.1 Background

2.1.1 Big Data

[Add information]

2.1.2 Artificial Neural Networks

Artificial Neural Networks are a group of algorithms that represent the operations of neurons in the brain. [Add information]

2.1.3 Self-organizing maps

The Self-Organizing Map is a remarkable visualization tool that uses an algorithm to train and scale a map to represent high-dimensional data in a manner that is simple to understand. It is widely used in applications and published in various articles. [ADD

REFERENCE] The generates maps cluster similar groupings of input as and the relationships formed. This knowledge is applied in industries such as Mining to solve multiple problems. The SOM is topology-preserving, so similar inputs are presented near each other in the output on the map. This can be investigated further to determine the similarity between the clusters.

2.1.4 Sentiment Analysis

Sentiment analysis, a process of categorizing words in given text with an emotion [5], is useful in elds such as business. Businesses are able to mine data relating to their services and products with the intention of improving customer service or user experience [4]. It has practical applications in other sectors such as lm and theater such that the classification of Twitter data (tweets) can suggest an improvement of performance for actors [6].

Data mining applies machine learning[14] algorithms to perform data mining tasks, however, the structure of the majority of these algorithms involve an additional processing step for the removal of symbolic rules to understand their underlying operations [7]. Symbolic rules are tokens within the extracted text for e.g.

2.2 Summary

This chapter introduces the main topics this work covers. It introduces the main topics as well as various related work in these fields. The next chapter covers the main topic which is self-organizing maps.

Chapter 3

Related Work

This chapter focuses on related works.

Section 3.1 is about Tree Kernels. Section 3.2 is about Classification Techniques. Section 3.3 is about Extracting Rules. Section 3.4 is about a case study on algorithms for sentiment analysis.Section 3.5 is about detecting small clusters. Section 3.6 concludes up the chapter.

3.1 Tree Kernels

Prior work in sentiment analysis investigates the use of a tree kernel to replace the process of feature manipulation. It was concluded that the tree surpassed its set goals. [9]. Coupled with the use of polarity scoring they drafted a tree that represented the tweets and calculated the likeness between the trees using a Partial Tree kernel. In conclusion the combination of word polarity and Parts-of-Speech (POS) tags achieves a better result for analyzing sentiments [9], and this very knowledge assists us in confirming the success of the use of word polarity and POS tags.

3.2 Classification techniques

A look at whether sentiment analysis would be accurately represented using positive and negative categorization was researched by Bo Pang who concluded that well-known ma-

chine learning techniques(naive Bayes, maximum entropy classification(MEC), support vector machines(SVM)) were better suited for topic-based categorization than sentiment classification [6]. This work assists the proposed study in narrowing down the list of suitable classification techniques. In contrast to Bo's work, this paper will be using a Kohonen self-organizing feature map to assist in the clustering and ranking of the data [8].

3.3 Extracting rules

Notable work with the SOM has been conducted within an article that pro-vides its own process of extracting rules from Self-Organizing networks and has applied it to cancer data [8], which has a similar analysis task. However, the proposed paper is more intent on the observation of the clusters formed as per our output derived from the SOM. The work presented by Sharma and Baig [2] similarly uses data sets from twitter by extracting tweets with earmarked keywords from Twitter however it differs in the algorithm used. They use the 10 TF*IDF(Term Frequency-Inverse Document Frequency) algorithm and perform three steps to clean their data .These steps are removing stop words, rating each word and a final step they title " Sentiment Variation Tracking" [2]. This study will in addition to the TF*IDF algorithm, be taking into account the appearance of positive and negative words in the tweets.

3.4 Case study on algorithms for sentiment analysis

A case study performed by Titus, Alapatt and Rao [13] will make a significant impact to this study. They use movie reviews as their subject. It differs from other sentiment analysis studies in that they use the minimum cuts algorithm[16] to support the naive Bayes algorithm. Their study resulted in the knowledge that the best approach to sentiment analysis involved the incorporation of each algorithm based on their strengths. These results will be used to compare the performance results of the SOM on each processing step e.g. feature extraction.

3.5 Detecting small clusters

Joao and Lobo's [15] research on identifying and visualizing clusters is closely related to the attributes of the SOM. It is known that often times, during SOM analysis, a small number of samples are discovered. Moreover, our study is interested in the detection of outliers within the map, and their work goes on to use a procedure that looks at the neurons either individually or collectively. One of the challenges they encountered was that the detection prolonged computation time. However, they were able to conclude that data pre-processing is of greater importance and if performed well, can produce better visual results and clusters. Similarly this work makes use of the SOM, and has gathered that the standard of the SOM needs to be assessed in future work.

3.6 Summary

This chapter introduces the main topics this work covers. It introduces the main topics as well as various related work in these fields. The next chapter covers the main topic which is self-organizing maps.

Chapter 4

Artificial Neural Networks

4.1 Introduction

Artificial Neural Network(ANN) is a branch of Artificial intelligence(AI) that is modelled after the neuron and synapsis structure of the biological brain [1]. The model of a neural network comprises of a vast amount of interconnected elements ,called neurones [1]. These work together to process input information I, and solve a given problem. ANNs have distinguished themselves within the field of AI, as being efficient in solving problems related to classifications of input data through the process of repetitive learning [1].

Figure 4.1 depicts a typical ANN, where the circles shows the neutrons of the network. The neutrons on the left represents the input neurons $\mathrm{I_n}$, the last neurons depicts the output neutrons $\mathrm{O_n}$,and the ones in the middle represent the processing neurons $\mathrm{P_n}$ of the network.

4.1.1 ANN learning methods

ANN can use one of three learning techniques to accomplish their task e.g. classifying if an image is a human or a tree. These techniques are referred to as Supervised, Unsupervised and Semis-Supervised learning respectively [2]. These techniques are discussed below.

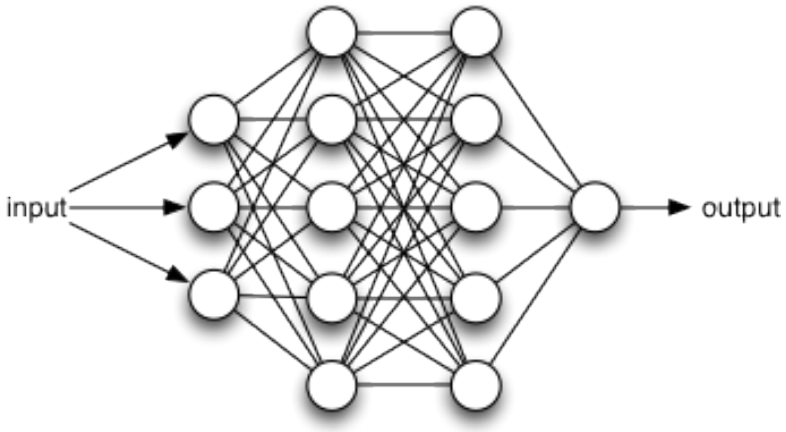

Figure 4.1: An image of the a neural network

Supervised learning

This is the most popular type of the learning techniques within the field of AI [2]. This learning techinque, trains the mapping function of the neural network to produce a given, controlled output, from the input of the neural network [2]. Thus this learning technique attempts to get output O_n from input I_n , using the mapping function F_m, if the mapping function produces the incorrect output O_n, the mapping function F_m is adjusted. This process is repeated until F_m, can produce all the $o \subset O_n$ for the given I_n.

Unsupervised learning

This type of learning differs from Supervised learning, mostly because when employing it, the output is not provided [2]. The objective of this type of learning it to model the distribution structure of the input data [2], *e.g. given a bucket of balls with different wieghts, how far will each fall go if the bucket is tipped over, and what is the correlation between distance (d) and weight (w)*, thus the mapping function Fm, is not controlled as it is in Supervised Learning.

Semis-Supervised Learning

This learning technique is a combination of both Supervised and Unsupervised learning [2]. With this technique, the inputs of only a certain portian of the input dataset is provided, and therefore F_m, only adjusts for data in I_n that have a corresponding output in O_n.

Chapter 5

Self Organizing Maps

5.1 Introduction

Self Organising Maps(SOM) are a type of Artificial Neural Network, strictly speaking, a SOM is mostly used as an unsupervised learning ANN algorithm. A SOM attempts to build a approximate model of the given input dimensions I . The model is approximated into a discrete output space called a *Map structure*. The Map structure comprises of small mapping units that are represented by neutrons [2].

Since the Map structure is just an approximation of the input dimensions, the neurons forming it have two important properties, namely

1. The output neutron map has a lower dimension than the input dimensions [2].
2. The neurons forming the Map structure, are less than those forming the input dimensions [2].

THereafter the SOM forms a model by reducing the input-dimension for the Map structure.The Map structure has the following two properties

1. The Map structure approximates the *density function* of the given input dimensions of the SOM. This is done by the neurons, that are used to analyse and cluster inputs that are similar to each other. Therefore, neutrons are more likely to model areas that are more dense within the input dimensions than areas that arent.

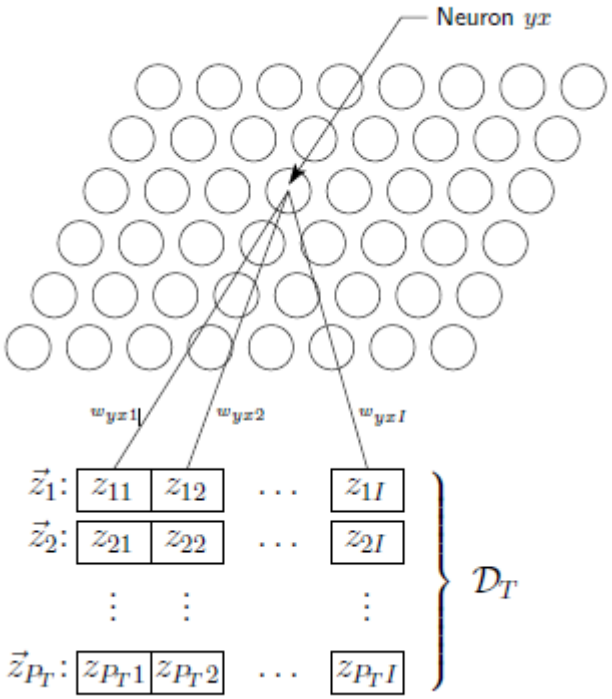

Figure 5.1: An image of the SOM Architecture

2. The SOM maintains the topological structure of the input space, thus if two input dimensions, I_A and I_B are similar to each other, their Map spaces M_A and M_B, should also be similar to each other.

5.2 SOM Architecture

SOMs are represented as a $N\ x\ M$ grid where the N represents the row in the map and the M represents the columns, refers to Figure 4.2. The neurons are distributed across the SOM as shown in Figure 4.2. Each one of the neutrons are connected to a weight vector that represents the centroid of a cluster that is associated with the neuron.

$$weightvector = w_{\mathrm{yx}} \tag{5.1}$$

where y and x are positive integers, y >= 0 and x >= 0

The weight vector is a tuple, where the y property represents the respective vertical indexes and the x property represents the horizontal indexes. Refer to Fig 4.2, therefore

$$\begin{aligned} w_1 &= [w_{11}, w_{12}, w_{13}, ., w_{1\mathrm{y}}] \\ w_2 &= [w_{21}, w_{22}, w_{23}, ., w_{2\mathrm{y}}] \\ w_3 &= [w_{31}, w_{32}, w_3 3, ., w_{3\mathrm{y}}] \end{aligned} \tag{5.2}$$

$$.$$

$$.$$

$$.w_{\mathrm{n}} = [w_{\mathrm{x}1}, w_{\mathrm{x}2}, w_{\mathrm{x}3}, ., w_{\mathrm{xn}}]$$

A SOM architecture comprises of a number of neurons arranges in certain structure, such as the neuron arrangement in Figure 4.2. These neurons represent the Map structure that was discussed in section 3.1. The neurons are trained to obtain the model, and the training data set $\mathrm{D_T}$

$$D_{\mathrm{T}} \tag{5.3}$$

$\mathrm{D_T}$ comprises of a training vector $\mathrm{z_s}$,

$$z_{\mathrm{s}} \tag{5.4}$$

Thus

$$D_{\mathrm{T}} = [z_{\mathrm{s}}] \tag{5.5}$$

The training vector $\mathrm{z_s}$ holds a set of input parameter values that can be represented by $\mathrm{z_s}$, therefore

$$z_{\mathrm{s}} = [z_{\mathrm{s}1}, z_{\mathrm{s}2}, z_{\mathrm{s}3}, .z_{\mathrm{sn}}] \tag{5.6}$$

where n is some random positive integer Thus

$$D_{\mathrm{T}} = [z_{\mathrm{s}1}, z_{\mathrm{s}2}, z_{\mathrm{s}3}, .z_{\mathrm{sn}}] \tag{5.7}$$

The respective training values for each of the input parameters in the SOM, is represented by a tuple $\mathrm{z_{sw}}$, where the *s* property represents the respective vertical indexes

and the w property, represents the horizontal indexes across the SOM neuron grid representation. Therefore

$$\begin{aligned} z_1 &= [z_{11}, z_{12}, z_{13}, ., z_{1\mathrm{y}}] \\ z_2 &= [z_{21}, z_{22}, z_{23}, ., z_{2\mathrm{y}}] \\ z_3 &= [z_{31}, z_{32}, z_{33}, ., z_{3\mathrm{y}}] \end{aligned} \tag{5.8}$$

$$.$$

$$.$$

$$.z_{\mathrm{n}} = [z_{\mathrm{x}1}, z_{\mathrm{x}2}, z_{\mathrm{x}3}, ., z_{\mathrm{xn}}]$$

5.3 SOM Training

The SOM builds the input approximated model by training and continuously adjusting the weights of the neutrons until they have an arbitrary accurate model for the given input data. The is accomplished through an iterative process referred to as *training* [2]. Through training, the model of the SOM starts to take form and becomes more accurate with each iteration of the training process.
The training process comprises of a series of training iterations. During an iteration step, the weights of the neutrons are adjust to build the SOM model. The initial iteration is iteration *0*, and the iterations gradually increase by a factor of 1 until the training process is complete. During a given training, the training data can be processed a arbitrary number of times, these are referred to as *epochs*.

5.3.1 Training Fairness

After each epoch, the training data is randomly organised as an attempt to create fairness amongst the neutron weight adjustments. Thus producing an un-biased evaluation and model representations.

Training phases

The training phases can be grouped into three main categories, namely the initialisation phase, weight adjustment phase and the evaluation of the stopping condition phase.

These phases are better described in the chapter below.

Initialisation Training phase (s num)

Before the training process of the SOM algorithm can commence; its training parameters must first be initialized. Some of the parameters that need to be initialized before training can start are, the number of rows Y and the number of columns X, these two properties are important as they affect the structure of the SOM map [2]; the parts of the weight vector must be initialled. There are various techniques and approaches that one can take to initialise these values, some of these approaches include:

1. One approach is to assign random numbers to the weights of the SOM before training can commence, and build the SOM mode, buy adjusting the random weights during each training iterative step [5]. Though this approach is effective, it is however inefficient as the initial weight are completely random and are not derived from the training data at all.

2. One is to employ a biased approach toward the initialisation step. The weights of the SOM neutrons are randomly selected from the training data set, thus the training will be biased towards those selected training data values [4].

These are many more initialisation strategies one can employ, and they have a large influence on the training of the SOM, varying in complexities, such as the simple randomised initialisation discussed in point (1), to more complex and elaborate initialisation strategies like the Hypercubic initialization strategy developed by Su et al [6].

Weight Adjustment Phase

In order for our SOM to build our model, we must adjust the initial weights during the training process to best fit our model. the adjustment of the weights comprises of two main steps, namely determining the best matching unit for the weight and update the weight vectors.

best matching unit There are many methods that can be employed in order to determine the best possible matching unit for the SOM. These include the Euclidean distance [3],

Nearest Neighbour Algorithm [7] and K-nearest Neighbour[7]. This work will focus on the Euclidean distance approach of determining the best matching unit for the SOM training process.

Euclidean Distance The Euclidean distance approach aims to find the smallest distance between a collection of vectors v, and it uses that distance as the Best Matching Unit (BMU). The distance between two vectors q1 and q2 is calculated using the formula: where q1 and q2, represent two vectors in the SOM. The BMU is a representation of the neutron with the smallest weight vector amongst the neurons. This is calculated using the following formula. Where z, represents the training vector and wba represents the weight vector.

ting Weight Vector Once the BMU is found, all the neutrons in the SOM are adjusted, relatives to the BMU. The adjustment magnitude is the greatest at the BMU neuron, and the magnitudes gradually decreases as it moves away from the BMU, until the adjustment magnitude becomes relatively negligible. The magnitude by which a relative neuron is adjusted by is determined by the following formula [3], where wyx denotes an arbitrary weight vector that is located row y and column x of the map at a given training iteration t.

Stopping Criteria

The training of the SOM is not a infinite process, and must thus come to a halt when certain halting conditions are reached. The ideal stopping condition is the natural stopping condition that occurs when SOM stabilises, this means that when the SOM converges to an arbitrary point. [3] This means that when the model generated through training, can accurately be mapped to the training input data I. Though this is the ideal condition under which the SOM should terminate, there are other conditions that could affect the SOM training, these conditions are discussed below [3].

s has been reached Setting a maximum iteration value is used to prevent the SOM from training to infinity, if the SOM never converges, and thus cannot terminate normally[3].

Training Parameters

In section (s num), we discussed the significance of the SOM parameters and the effect they have on the learning ability of the SOM. In this section, we are going to further discuss more of these parameters.

Map Dimensions This parameter is refers to the size of the SOM [4]. The map dimensions parameter has a direct influence on the earning ability and model produced by the SOM, small SOM could lead to inaccurate models whilst large models can have a lot of redundant neutrons and tend to have a high time complexity [3]. It is thus important to choose an appropriately sized SOM for your training.

Learning Rate The neighbourhood function used in this work is a Gaussian Kernel, with a learning rate r(t). The r(r) signifies the SOMs learning at any given time, thus if the there are large adjustments that must be made for a given iteration, will result in a large learning rate, and low weight adjustments will result in a smaller learning rate, i.e. the weight adjustment magnitude is directly proportional to the learning rate. Please refer to

ghbourhood Radius With smooth Gaussian kernels like the one used in this paper, the kernel width has a very important significance, that is, it has a influence on the radius of the neighborhood, and thus has an influence on the neutrons around the BMU with regards to the update weight vectors.[3] The relationship between the Kernel width and the Neighbourhood Radius is as follows. The higher the Neighbourhood Radius, the wider the Kernel width, and the lower the Neighbourhood Radius, the narrower the Kernel width, please refer to

Summary

This chapter has introduced the SOM algorithms and its important properties and artefacts. These are important, not only in the process of building the model of the SOM, but also for its interpretational value for a given problem. The next chapters will discuss the application of a SOM for visualization within the context of sentiment data analysis and studies.

Chapter 6

Self-organizing map visualization

Chapter 7

Methodology

Apply pre-brief here

7.1 Knowledge Mining

Data collection from a public Twitter repository on-line. Data Pre-processing Perform data cleaning on the given data. Store data into database for further review.

7.2 Knowledge Retrieval

To retrieve the specified tweets, the programming language python was used along with the TwitterAPI and other libraries to stream live tweets between the span of two days. These tweets were limited to specific parameters such as location, South Africa, Track, which is the keywords needed to be present in each individual tweet, along with Language, English.

7.3 Security and Privacy

These tweets would need to be cleaned in such a manner that the remaining data is both useful and cannot be person-descriptive. Usernames were removed from the tweets.

7.4 Algorithms

Algorithms - The SOM was chosen for its ability to visualize data that is di cult to interpret as it constructs a map of the data from the input and conserves the topological properties. The algorithm proceeds iteratively. On each training step a data sample x from the input space is selected. The learning process is competitive, meaning that we determine a winning unit c on the map whose weight vector mc is most similar to the input sample x. The weight vector mc of the best matching unit is modified to match the sample x even closer. As an extension to standard competitive learning, the nodes surrounding the best matching unit are adapted as well. Their weight vectors mi are also .moved towards. the sample x. Nodes closer to the best matching unit will be more strongly adjusted than nodes further away. At the beginning of the learning process, the best matching unit will be modified very strongly and the neighbourhood is fairly large. Towards the end, only very slight modifications will take place and the neighbourhood includes little more than the BMU itself. This corresponds to .rough ordering. at the beginning of the training phase and .fine tuning. near the end

7.5 Tools

SOM PAK or Viscovery are the preferred tools for visualizing data. SOM PAK program package includes necessary programs for the use of the SOM algorithm and the visualization of its data [10]. The use of Python, Panda, Anaconda and SQLite will assist in extracting the data from the csv les into a data le and store it into the database. The on-line tool DataPreparator for data pre-processing will be used [11].

7.6 Data Visualization

Data visualization will take place in MATLAB as well as Python. Visualization will be in the form of graphs. In many practical situations, vast sets of unknown multi-dimensional data are present. Clustering is an approach to identify natural groupings of similar entries in such sets of unclassified data - often without any a priori knowledge as to what that similarity may involve. The principal idea is to partition the dataset into meaningful

sub-classes, called clusters. Especially if good visualization support is available clustering can provide a helpful first impression of the way the data is distributed. It is therefore often undertaken as an exploratory exercise before doing further data mining.

7.6.1 Topology preserving

Since not only the winning node is tuned towards the input pattern but also the neighbouring nodes, it is probable that similar input patterns in future training cycles will find their best matching weight vector at nearby nodes on the map. In the run of the learning process, this leads to a spatial arrangement of the input patterns, thus inherently clustering the data. The more similar two patterns are, the closer their best matching units are likely to be on the final map. It is often said, that the Self-Organizing Map folds like an elastic net onto the cloud formed by the input data. [Simula (1999); Vesanto and Alhoniemi (2000)]. It is important to state that the Self-Organizing Map algorithm is not a clustering algorithm. It is intended primarily as a tool in reducing the dimensionality of the data and for information visualisation. Of course, this includes the visualisation of groups of similar items. But the Self-Organizing Map is not a tool that will produce an explicit partitioning of a dataset into a precise number of groups. This also explains why the concept of a cluster is not well defined for the Self-Organizing Map. The maps do not show sharp cluster borders and there is no obvious centroid. Of course, one can theoretically think of each node on the map as a cluster centroid. The cluster corresponding to each node could then be said to include all dataset items mapping to this node. But this is not a sound approach in the practical application of the SOM. It tempts the user to use small maps of only k nodes, expecting that this will produce k clusters in the same way kmeans does. The results with such small maps, however, are very poor. The heart of the algorithm is the neighbourhood function and the concept of adjusting not only the best matching unit but also its surrounding units. This will create neighbourhoods of similar nodes but only if the space on the map is sufficiently large to allow this.

7.6.2 The choice of parameters

The maps produced with the SOM algorithm are very much influenced by our choice of parameters. This includes: the map width and height, the number of iterations, the size the initial radius and the initial value of the learning rate. There are no strict guidelines for choosing any of these parameters. A process of .trial and error. Is necessary to determine a set of values that are suitable for the dataset at hand. As a rule of thumb, it has been suggested [Kohonen et al (1996)] to use rectangular (but non-quadratic) maps, say, of size 15 by 10 and to use an initial radius equal to the height of the map. For the value of the initial learning rate factor a value of 0.05 has been suggested. In the following diagrams, we have assumed: Total number of iterations: 10000 Initial Radius: 10 Initial Learning Rate: 0.05

7.6.3 Visualizing Self-Organizing Maps

Extracting the visual information provided by the Self Organizing Map is a central concept of this paper. The choice of visualization technique, however, is far from straightforward. Visualizing a SOM is challenging because the input data is usually of a high dimensionality. By projecting the input space to a two-dimensional grid we can express the similarity of two samples as the distance between them. But while simplicity is gained by reducing dimensionality, information is effectively lost when the data item is simply represented by a dot. The mere position on the map cannot sufficiently embody the complexity of an n-dimensional vector. The problem of visualising multivariate data is, of course, not a new one. Information Representation is a mature area of research and numerous approaches of displaying multidimensional multivariate data have been proposed. [Wong and Bergeron (1997)] The following paragraphs briefly review a number of these methods.

7.7 Text feature extraction

In information retrieval or text mining, the term frequency inverse document frequency (also called tf-idf), is a well know method to evaluate how important is a word in a

document. tf-idf are is a very interesting way to convert the textual representation of information into a Vector Space Model (VSM), or into sparse features, well discuss more about it later, but first, lets try to understand what is tf-idf and the VSM.

7.8 Normalization

Add reference to min-max normalization

7.9 Summary

Add summary

Chapter 8

Experimentation Results

8.1 Experiment

This section analyses the techniques and statistical methods applied in carrying out the study investigation contained in this research paper. Subsequently, the presentation of the results, including the discussions, follows.

Chapter 8.2 analyses the data set which was the foundation of the investigations, and 8.4 introduces and attempts to explain the experimental results. Lastly, section 8.5 gives an overview of the chapter.

8.1.1 Experimental Data Sets

This section provides details on the experimental data sets that were used as a platform for the analysis of the procedures explored in this chapter.

Political Tweets Data Set

This section examines the Political Tweetsdata set. Background information on the data set is given in Section 8.2.1.1, while Section 8.2.1.2 describes the data preparation that was performed on the Iris plants data set before the experimental analysis.

ta Set Background

1. The Iris plants data set was produced in 1936 by Sir Ronald Aylmer Fisher, for his work on taxonomic problems [67]. This data set is considered to be a very simple learning domain. Despite its simplicity, the set is a very common standard for pattern recognition and data mining applications, and was therefore included in the analysis.

2. The data set consists of tuples that represent individual examples of Iris plants. The descriptive attributes of the data set consist of four attributes, namely username etc,Each attribute value is continuous,and denotes a measurement that is taken in centimeters. No attribute values are missing for any of the examples making up the data set. Table 8.1 shows the most important characteristics related to the descriptive attributes of the data set.

3. Each example is classified by means of a single nominal attribute. A classification is either IrisSetosa, IrisVersicolor or IrisVirginica. The distribution of data set examples between these three classes is shown within Table 8.2.

4. The IrisSetosa class is known to be linearly separable from the other classes, and IrisVersicolor and IrisVirginica are not linearly separable from one another. High correlations to the example classification exist for petallength and petalwidth, with a moderate correlation for sepallength, and no correlation for sepalwidth.

Chapter 9

Conclusions

Provide an introduction, stating that this chapter summarises the conclusions of your work, and consider future directions that related research could take. Again, reference Sections 9.1 and 9.2.

9.1 Summary of Conclusions

Summarise your conclusions here. You should consider each of the objectives you listed in the introduction, and explain how each was met, providing a discussion on what your specific findings were for each. Also mention what novel contributions (these might include a new taxonomy, model, algorithm, or empirical results not previously published) your work has introduced to the field while the various objectives were being addressed.

9.2 Future Work

Enumerate the future work that you could foresee developing from the work you have done here. Mention areas you could not focus on, or possible extensions to your work. It is a good idea to be thorough, since you increase your chances of being referenced by other researchers who follow up on your work, even if you do not do so yourself. You may consider writing this as a bulleted list, if you mention many aspects.

Bibliography

[1] Eric Gaussier and Longbing Cao. Conference report on 2015 ieee international conference on data science and advanced analytics (dsaa'2015)[conference reports]. *IEEE Computational Intelligence Magazine*, 11(1):13–14, 2016.

[2] Patrick Kouontchou, Amaury Lendasse, Yoan Miche, Alejandro Modesto, Peter Sarlin, and Bertrand Maillet. A r-som analysis of the link between financial market conditions and a systemic risk index based on ica-factors of systemic risk measures. In *2016 49th Hawaii International Conference on System Sciences (HICSS)*, pages 1759–1770. IEEE, 2016.

[3] Deren Li, Shuliang Wang, and Deyi Li. *Spatial Data Mining: Theory and Application*. Springer, 2016.

[4] Nikhil George Titus, Tinto Anto Alapatt, and Niranjan Rao. A case study on the different algorithms used for sentiment analysis. *International Journal of Computer Applications*, 138(12), 2016.

[5] Zongben Xu and Yong Shi. Exploring big data analysis: Fundamental scientific problems. *Annals of Data Science*, 2(4):363–372, 2015.

Appendix A

The First Appendix

Appendices follow exactly the same structure as chapters. They are used to describe aspects that are not central to the thesis (for example, an algorithm that you benchmark against, but do not focus on in the main text, or a discussion on the sample datasets you test your algorithm on).

A.1 Summary

As always, provide a summary at the end.

Appendix B

The Second Appendix

Appendices follow exactly the same structure as chapters. They are used to describe aspects that are not central to the thesis (for example, an algorithm that you benchmark against, but do not focus on in the main text, or a discussion on the sample datasets you test your algorithm on).

B.1 Summary

As always, provide a summary at the end.

Appendix C

Acronyms

Appendix D

Symbols

Provide a brief introduction, in which you explain that all the symbols used throughout the thesis are defined in this appendix, under the chapter in which they first appear. Also mention that re-definitions of symbols are introduced in the chapters in which they occur (but try very hard to avoid this). Do not repeat symbol definitions in later chapters. Try to keep definitions as short as possible (try not to wrap onto the next line). You may play with the length called `namewidth` (defined in `thesis.tex`) to make sure that all the symbols fit properly throughout this list. Provide the chapter name in each section, as appropriate. Leave out chapters with no symbol definitions. Also note that you may provide a reference to an equation defining a symbol if you choose to (although this limits the space you have to work with) — decide if you want to do this, or not, and stick to either one or the other:

D.1 Chapter 2: The First Chapter

$\mathcal{A}$	Some symbol that we use
m	Another symbol we use
t_i	The i^{th} something

D.2 Chapter 3: The Second Chapter

β	Yet another symbol	
$\eta(t)$	And another one	[Eq. (??), pg. ??]

Appendix E

Derived Publications

Explain that the following list includes a list of the publications derived from this thesis. You may list already accepted publications, as well as ones that are currently under review. Make sure that the format is the same as that produced by BibTeX (we will hopefully release an automated way of generating this list some time in the future), and that the references are all correct.

- First reference.
- Second reference.
- Third reference.