Using machine learning and Bayesian Networks to objectively analyze Central Bank statements and market sentiment

Flavia Grignani

Supervisor: Prof. M. Gasparini
Prof R. Fontana

Ingegneria Matematica
Politecnico di Torino

This dissertation is submitted for the degree of
Ingegneria Matematica,
Statistica e ottimizzazione su dati e reti.

September 2018

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Flavia Grignani
September 2018

Abstract

The minutes of the meetings of the Swedish Central Bank drive the other banks' economic choices. Nowadays minutes are read and analyzed manually, and a lot of time is wasted because of that. In finance, the reaction to each event has to be as quick as possible, so it needs to be performed in an efficient way. The goal of this project is to automate the text comprehension process using machine learning algorithms. The problem can be divided into two main tasks, to summarize the speech of each board member and to find his sentiment and intention. To retrieve the summary an unsupervised approach is used, based on the Text Rank and the Latent Semantic Analysis algorithms, combined with information about the most discussed topics in Board members meeting. The sentiment behind this kind of economical text cannot be identified as positive or negative, as in most of the literature, but it can be classified as hawkish or dovish. Several classic supervised classifiers have been used, like SVM, Logistic regression and Naive Bayes, based on the past analysis of the minutes performed by SEB bank. Furthermore, an innovative classifier is built, using a Dynamic Bayesian Networks. The accuracy of the results achieved is at the level of the state-of-the-art of this field.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

Skandinaviska Enskilda Banken AB (SEB) is a Swedish financial institution with headquarters in Stockholm. In the investment division of the Bank, a lot of information is analyzed daily and reports are created for clients and internal teams. Riksbank, the Central Bank of Sweden, following the examples of European Central Bank (ECD), Federal Reserve (FED) and several others central banks, has adopted a more transparent approach in the last 15 years. [21]

Every two months Riksbank publishes the minutes of the meeting attended by its Board Members, and that paper contains lot of useful information for the Swedish banks. Since 2007, the SEB bank analyzes the position of each Board Member in the meetings. The examination of the minutes is shared on SEB's Website few hours later the minutes' publication.

The aim of this project is to automatically create a report with the most important information of the minutes. In SEB's reports, the position of the Board Members is presented to the reader by a summary of the main concepts debated. Furthermore, the Board Members are ranked from the most Dovish to the most Hawkish. The users of the automatic report are the SEB research team members, then we tried to create a report as similar as possible to the one that is already written manually by them. The problem is faced with the help of the text analysis. After a review of the techniques to examine texts, two main methods have been chosen: Text Summarization and Sentiment Analysis.

In order to synthesize the opinion and analyze the topics of the speech of each board member, different algorithms have been applied. A formula is calculated to give a relevance score to each sentence and then extract the most important. In this phase the human help is fundamental, since the summarization algorithm provided to the Research team of SEB is based on some topics that are given ex-ante by the user. In chapter 4 our approach to Text Summarization is explained .

Sentiment Analysis techniques are frequently used to extract positive or negative feedback. We expanded this idea to give an Hawkish/Dovish score with the help of the most recent machine learning algorithms. In Chapter 5 the strategies to create this sentiment score are reported .

The summarization and sentiment analysis algorithms cannot be used directly on the raw text but some preliminary preprocessing is required. Several operations are performed to organize the collection of documents, like dividing the texts into sections, encoding the words with vectors, discarding the useless words. In Chapter 3 the different issues that have been faced to pass from the original PDF text to a more structured form of data are explained.

1.1 Economical grounds

1.1.1 Monetary policy

The Central bank of every country is responsible for its monetary policy, in Sweden this is a task of the Riksbank. The objective for monetary policy is "to maintain price stability, that is keep the inflation close to the target of 2% per year.[1] Broadly speaking, there are two types of monetary policy, expansionary and contractionary. The first one is adopted with the intention to encourage economic growth and expand the money supply; instead the second seeks to obtain the opposite result. The two policies are a reaction to different types of economic situation that a country faces.

Central banks have three main tools to guide the inflation: set the Repo Rate, buy or sell treasury bonds through open-market operations, and establish reserve requirements [5]. Repo Rate is used as benchmark by the others banks to set them lending rates, with a low lending rate people are induced to borrow more money and the economy is stimulated.
Central bank can buy a large amount of treasury bonds to decrease the market liquidity, the prices will rise and the yields will drop. Clients will shift to other assets with higher returns and the economy will expand, thanks to the diversification of the investments.
Furthermore, Riksbank can decide with percentage of costumers deposit banks must keep, the level they reserves affects the short-term interest rate banks pay to borrow and lend money from and to each other. This influence also the interest rates banks charge consumers for borrow money.

Decisions about the monetary policy are taken by the Executive Board, composed of six members who, as politicians, can have different opinions about the best monetary policy. Monetary policy effects are not easy to measure and there is not a unique way to decide when

[1]https://www.riksbank.se/en-gb/monetary-policy/

is time to change the trend. Since the 2008 financial crisis, the monetary policy of almost every country is expansionary to stimulate the economy.

In Sweden, the current repo rate is below zero and the Riksbank has a heavy presence in bond markets. The statements of the Board members said that the strategy will remain constant until the end of the year.[2]

The sentiment of the central bank monetary policy can be described as **hawkish** or **dovish**.

The two sentiments correspond to a contractionary monetary policy and a expansionary monetary policy respectively. An hawkish person is thinking that the economy is going well, and there is a risk of a too rapid inflation growth. To keep it under control, usually, the interest rates are raised. A dovish person has the opposites feeling about the economy situation, and he is afraid of a too slow inflation growth. It is important to understand that the border between the two definitions is not defined, and sometime is not easy understand in which position is a given strategy. [3]

[2]https://www.ft.com/content/8998e16c-b15c-355c-8512-32d4b6b4b2f8

[3]//www.tradingheroes.com/hawkish-and-dovish/

Chapter 2

Dataset

In this section the available data are described . All the original data are in PDF format and they have to be elaborated to be usable in the algorithms. In the following chapter the initial and the final version of the data are described, respectively, the PDF, and the structured datasets used as inputs for the program.

The first kind document analyzed is called Minutes. They are the statement, emitted by Riksbank, of the Board Members meeting. They are written in English and organized with an almost immutable structure. The text is transformed from PDF to HTML format to enriched the text with HTML tags. Subsequently, the latter are used to extract features and create a first dataset, called Minutes Dataset.

The second kind of document is called SEB Report, it is released by SEB few hours later the publication of the Minutes. It summarizes the key declarations, of each Board member, during the meeting.

It is a more structured table with six row and three columns. Rows are one for each Board Member, and for each of them is reported: name-surname, summary of their argumentations, and hawkish/dovish rank.

2.1 Minutes

Minutes of Riskbank meeting are analyzed to guide the economical decision related to monetary policy. They are public and available for everyone since 2013. They come out every two-three months and they have almost the same structure every time. They are divided into four sections, but for the purpose of this analysis, it is possible to focus only on one of them, the Board members discussion part, where their opinions and thoughts for future changes in monetary policy are debated.

In the most recent papers, board member's speeches are slightly summarized and grouped in a single and continued section. This operation simplifies the search for each board member's text and helps the human or automatic reader to go easily through the paper. Unfortunately, in the minutes published before 2015, the structure was not so well defined and more sections of the same board member were allowed, probably to maintain the idea of a dialogue. Each board member's speech is usually from three to four pages long and it addresses several topics. Some of them are recurrent, such as "Inflation" or "Repo rate", others could be specific in a meeting, such as "Oil" or "FED".

The names of the members of the Executive Board are listed at the beginning of the first section because they can change over the years. The Board members are appointed for a period of 5/6 years according to a continuous program. They are six and all of them are usually present at every meeting, also because at least half of the members must be present for decisions to be made. Six years is a significant amount of time and this gives stability to the analysis performed.

2.1.1 Summarization dataset

All the minutes available on Riskbank website[1] have been analyzed to study patterns and extract information in the future ones. The collection that has been obtained is made by nineteen Minutes.

First of all, for each minutes, the Board members speeches are extracted. As explained in the introduction, not all the text of the minutes is useful for the analysis, but just the section that in the most recent minutes is called "The economic situation and monetary policy". In that section each Board member section is preceded by the Board member name in bold letters. This recurrent feature is used to locate each Board member section. To perform this operation the minutes are converted from PDF format to HTML format, to keep track of what is written in bold letter. The process is better explained in section 3.1.

To obtain a more structured data, we find in the text, the parts related to a specific topic. Topics are indicated by keywords and a paragraph is considered related to an argument if it contains the corresponding keyword. By combining all the paragraphs of a specific author on a topic, we get all the text of a member of the board members on a given topic. We decided to search for keywords in paragraphs, rather than sentences, in order to include even sentences that do not contain keywords but are close to some of those that contain them We

[1]www.riksbank.se

Minutes	Board Member	Topic	Sentences
2016-02	Cecilia Skingsley	Inflation	307, 308, 309, 345, 346, 347, 348, 349
2016-02	Cecilia Skingsley	Krona	307, 308, 309, 322, 323, 324, 325, 340, 341, 342
2017-10	Per Jansson	ECB	180, 181, 182, 223, 224, 225, 226, 227, 228

Table 2.1 Three rows of Minutes Dataset.

hypothesized that it is better to include unrelated phrases rather than to exclude some really important phrases just because they do not contain the given word. It is particularly useful for the summarization task. All these operations are made by combining HTML tag and Python Library named NLTK, see section 3.1. The keywords (Topics) are selected by a dictionary[2] created with the help of the SEB's research team.

For the Topic Dictionary, four main topics were considered, which they always debate in every meeting, no matter in what year they are. They are

- Inflation
- Repo rate
- Rate
- Swedish Krona (or just Krona)

These Topics are suggested by the Research team of SEB. The paragraphs are labeled based on these four main topics. One paragraph could contain more than one topic or none.

In text mining, there is not a shared language to identify the different documents and part of text; however, in section 3.2 is reported the most used. Transferring the notation on our data, this is how the data are organized:

- Collection C is the set of Minutes that are available since 2015, so d_i is the i-th Minutes, $C = \{d_1, d_2, \dots, d_q\}$ with $q = 19$;
- Document d_i is a single Minutes, it is composed of Part of text pt, $d_i = \{pt_1, pt_2, \dots, pt_k\}$ with $k = 6$ the number of Board Members;
- Part of text pt_i is the section referred to a Board Member. It could be divided by topics $pt_i = \{top_1, \dots, top_p\}$, or by paragraphs $pt_i = \{pg_1, pg_2, \dots, pg_{n_i}\}$, or by sentences $pt_i = \{s_1, s_2, \dots, s_{m_i}\}$, or by words $pt_i = \{w_1, w_2, \dots, w_{t_i}\}$, it depends by the goal of the analysis applied;

[2]In this context, dictionary/lexicon are used interchangeably to indicate a collection of words with some information available.

Specifically, in this project we have:

- each Minutes d in the collection of Minutes C is identified by the date, such as $2016-02$;
- each Board member section s from a specific Minutes d is identified by the name of the Board Member;
- each topic top is identified by the name of the topic, "Krona" for instance.

Finally, for each Minutes, for each Board Member and for each Topic, the corresponding paragraphs are saved by the identification number of the sentences that compose it. The text of the equivalent sentence is retrieved by the dictionary of sentences.

The summarization dataset looks like the table above (2.1). For each minutes, Board member and topic we see the corresponding paragraphs[3].

2.2 SEB Reports

Some hours later than a Monetary Policy minutes is published, the SEB bank provides a schematic summary of it, that is uploaded to its website[4]. For each member of the board, they provide a brief summary, consisting of a few lines of revised text taken from the board member's speech. The board members are also classified from the most dovish to the most hawkish, based on what they said during the meeting. Each meeting is considered independently, the score is assigned without taking into account the past behavior of the board member. When the economic position of a member is not well defined the Board member ends in the middle of the rank and he/she is considered as neutral. When two or more of them express the same idea, they are considered at the same level.

It is important to observe that the information regarding the position of the Board Member and the summary provided are two different and independent information. This means that the sentences that are collected from the original Minutes to create the summary are not the most relevant to understand if the Board member is Hawkish or Dovish. For this reason the summarization and the sentiment analysis problems are never considered together in this work. [5].

[3]The paragraphs are indicated by the corresponding sentences. In the example in the table 2.1 the first paragraph is composed by the sentences number 307,308 and 309, the second one by the sentences number 345,346 and 347 and so on.

[4]www.seb.se

[5]The SEB research team suggests us to do not base us sentiment analysis over the output of the summarization. The two processes are separated because the most relevant sentences used in the summary are not necessary related with the economical position of the Board member.

2.2.1 Sentient Analysis dataset

The summaries are used to evaluate the performances of the summarization algorithms. The rank, given by the SEB analysts, is used to build a train set for the sentiment analysis algorithms. In our work, the two highest board member in the ranking are considered as *dovish* (labeled as 1) and the lowest two are considered as *hawkish* (labeled as 0). The third and the fourth are classified as *neutral*. In the event of a tie between three members, all of them are considered as *neutral*.

The entire text of each board member is labeled using the score given by the SEB report. As said before, Board members speak for a long time and not all of the sentences in the speech are meaningful, to assign the Hawkish/Dovish score. In this work, we let the algorithm to pick the most relevant sentences to build the Hawkish/dovish score, without the risk to *a priori* delete something important . The obtained dataset is composed of 107 instances.

Chapter 3

Text preprocessing

As we have already introduced, a written text can not be used as an input of the final summarization or sentiment analysis algorithms. The text is an complex format and without a pre-processing work, a computer would not be able to extract any interesting information from it. The techniques that we used in our project are described in the following chapter.

3.1 Text Cleaning and Data Preparation

Text data are a very unstructured source of information for IT purpose. For a human being it is usually straightforward to get the right meaning of a text, unfortunately, it is not the same for a computer. The text are available, the first issue to be solved in every Text Mining problem is the text cleaning and data preparation.

In this project, it is possible to divide this issue in two sub-problems:

- Technical text cleaning, named Data Preparation;
- Purpose-based text cleaning, named text cleaning.

The first identifies the problem of transforming the text to be ready to be analyzed by algorithms. The inputs of the Data Preparation are the Riksbank Minutes and the SEB Reports in a PDF format. The output is a structured dataset as it is described in section 2.1.

The second technique called Purpose-based Text Cleaning, is applied to transform and clean the text contained in the Minutes and the Report dataset. In the following section several techniques available for this intent are reported and it is analyzed when it is advantageous to use them.

3.2 Notations

There are several algorithms available to extract the most relevant sentences from a text. Some common concepts and the hypothesis behind this type of methods have to be defined to introduce the algorithms.

There are some important definitions that it is important to repeat in order to have a common notation in the following explanation:

- Collection C, it is the set of documents d, $C = \{d_1, d_2, \ldots, d_q\}$;
- Document d_i, it is a single unit of a Collection and it is composed of Part of Text pt, $d_i = \{pt_1, pt_2, \ldots, pt_{n_i}\}$;
- Part of Text pt_i, it is a single unit of a Document d, it can be a sentence $pt = s$ or a paragraph $pt = pg$, that, in turn, could be composed by sentences $pg_i = \{s_1, s_2, \ldots, s_{p_i}\}$.
- Sentences s are composed of words w, so that $s_i = \{w_1, ..., w_{m_i}\}$.

In particular, a general word w_i in a sentence s is not uniquely identified.The tokenization part is essential, that means to assign the right token from a collection of tokens (or labels) to a word in a sentence. The collection of tokens are generally named dictionaries or lexicons, and there are collection of words where the specific meaning is defined for each shade of meaning of the word, moreover some more information could be collected in the dictionary.

There are many dictionaries available on-line for the purpose of tokenization and Part-of-Speech Tagging (POST).[1] The latter is used to infer the right meaning of a token usually based on the context of the word that surround the word tokenized. [2]

A good reference for this purpose are the book "Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition" from Prof. Daniel Jurafsky[20] and "Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from Your Data" from Dipanjan Sarkar. [36]

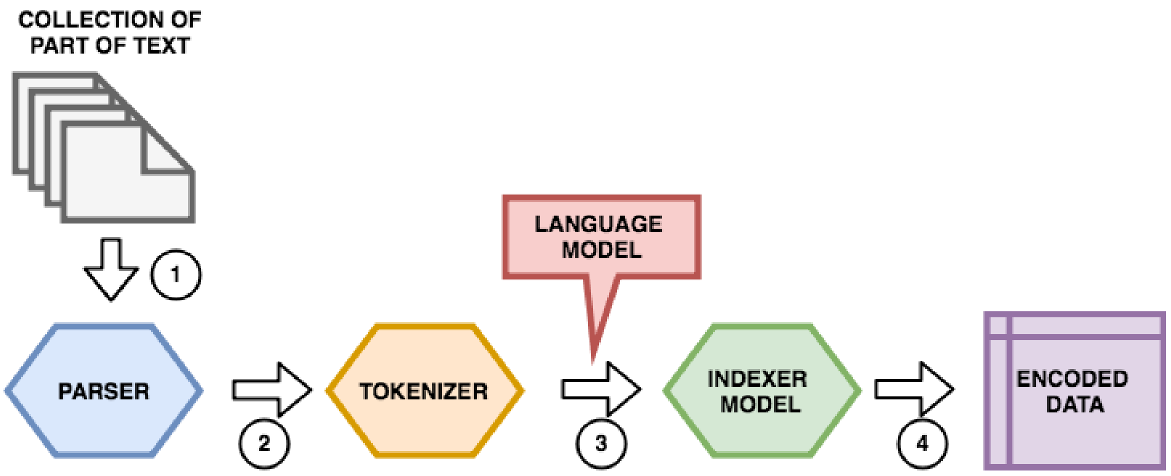


Fig. 3.1 Parsing: add a description

3.2.1 From PDF to Data encoded

In diagram 3.1the flow for the indexing of collection of documents is explained . Text indexing is used to encode texts and then apply computational techniques to retrieve information.

1. Documents Parsing: documents were in pdf format, a transformation to HTML format is done. Then with Regular Expression, the information requested are extracted in the format described in the Dataset section 2. The Python library PDFMINER.Six is used in the code for this passage.

2. Tokenization: once the text is available in a .txt format the tokenization is applied.

3. Language Model: some language based heuristics are applied to the tokens to enrich the structure of the data. Some of the techniques available and used in the project are listed and briefly introduced: Stop-words, Lemmatization, Stemming. With a Stop-words list, it is possible to exclude from the dictionary entirely the commonest words that usually are not so discriminative in the model, such as: "the", "a", "or"... Stemming and Lemmatization allow to remove the prefixes, suffixes from a word and change it to its "base form". It is straightforward that is a good way both the techniques above are a good way to reduce the numbers of unique terms in the collection. For a

[1]Some of the most used Knowledge-based dictionaries available are: WordNet, SentiWordNet, Affect-Net,GoogleNGrams, MicrosoftNGrams, NELL, FrameNet, ConceptNet, VerbNet, FreeBase, DBPedia, Probase, SGECKA, Per language resources, e.g. Cornetto.

[2]Spacy https://spacy.io/usage/linguistic-features is a powerful Python Library used to extract linguistic features like part-of-speech tags, dependency labels and named entities, customizing the tokenizer and working with the rule-based matcher.

more accurate survey on these techniques a good paper is "Preprocessing Techniques for Text Mining" from Gurusamy Vairaprakash and Kannan Subbu.[13]

4. Indexer Model: the next section introduce the most famous "Bag-of-words" Model to encode texts.

3.3 Vector Space Model

Most of the techniques in Machine Learning have a shortcoming, it is not possible to give directly unstructured information to the algorithms. In fact, features are used to have a common pattern in the informations available. With numerical data, it is quite easy to organize the data in some matrix. However, with text data it is not that trivial and some hypothesis are necessary to move from text to numbers.

To this case, there is a mandatory step after the preprocessing and the cleaning of the text: the feature extraction. It is useful to give a more structured dataset to the algorithms applied. In this section, one of the most used feature extraction model for text data is analyzed. It is called Bag-of-Words model or Vector Space Model. The first name also identifies the hypothesis behind this model: terms are considered as in a bag, the order by which the terms appear in the sentences is not relevant, only their frequencies are considered.

In other words, it is a simplistic representation of a Collection $C = \{pt_1, ..., pt_n\}$, where each Part of text pt_i[3] is modeled as a vector. A dictionary DIC is used to create a space for the terms te considered, thus the vector v_{pt_i} that represents a part of text pt_i is a point in the space of the dictionary, $v_{pt_i} \in R^D$ with $D = dim(DIC)$.

The final result of the Bag-of-Words model is usually represented in a matrix M, where the rows are the terms in the Collection C and the columns are the Part of text pt . In other words, the matrix $M \in R^{Dxn}$ has D rows and n columns, with this in mind, rows of M give information about the presence of terms in a specific Part of text pt indicated by the column.

The Bag-of-words hypothesis is valid only if words are thought to be independent of the position where appear in the text. Therefore, it is a strong supposition and a lot of available knowledge is lost. However, this technique is widely used because of its simplicity. From an unstructured data collection, we arrive at a matrix of numbers, where a lot of mathematical tools and theories are available to be applied on it.

As an example, let's consider the following collection of sentences C = {"To be or not to be, that's the question.", "To be or not to be. That's not really a question.", "Development is about transforming the lives of people, not just transforming economies."}. After the Text

[3] The Part of texts could be documents $C = \{d_1, ..., d_n\}$ or sentence $C = \{s_1, ..., s_n\}$or other type of sections

Cleaning step, the collection is represented as $\hat{C}$ = {"to be or not to be that is the question", "to be or not to be that is not really a question", "development is about transforming the lives of people not just transforming economies"}.

Starting from this cleaning collection $\hat{C} = \{s_1, s_2, s_3\}$, the first step of the Bag-of-Words model is the creation of a dictionary:

$$\begin{aligned} D = \{&"to", \quad "be", \quad "or", \quad "not", \quad "that", \\ &"is", \quad "the", \quad "question", \quad "really", \quad "a", \quad "development", \\ &"about", \quad "transforming", \quad "lives", \quad "of", \\ &"people", \quad "just", \quad "economies"\} \end{aligned} \tag{3.1}$$

To get the final representation of the Collection, the measure the relevance and the discriminative power of a term has to be chosen. There are several scores as explained in section 3.3.1, in this example the frequency in the single sentence is counted. The final matrix is:

$$\mathbf{M} = \begin{array}{l|ccc|} & s_1 & s_2 & s_3 \\ \text{"to"} & 2 & 2 & 0 \\ \text{"be"} & 2 & 2 & 0 \\ \text{"or"} & 1 & 1 & 0 \\ \text{"not"} & 1 & 1 & 0 \\ \text{"that"} & 1 & 1 & 0 \\ \text{"is"} & 1 & 1 & 1 \\ \text{"the"} & 1 & 0 & 1 \\ \text{"question"} & 1 & 1 & 0 \\ \text{"really"} & 0 & 1 & 0 \\ \text{"a"} & 0 & 1 & 0 \\ \text{"development"} & 0 & 0 & 1 \\ \text{"about"} & 0 & 0 & 1 \\ \text{"transforming"} & 0 & 0 & 2 \\ \text{"lives"} & 0 & 0 & 1 \\ \text{"of"} & 0 & 0 & 1 \\ \text{"people"} & 0 & 0 & 1 \\ \text{"just"} & 0 & 0 & 1 \\ \text{"economies"} & 0 & 0 & 1 \end{array}$$

From the matrix M, it is possible to get relevant information for both sentences and terms:

- $s_3 = \{0,0,0,0,0,1,1,1,0,0,0,1,1,2,1,1,1,1,1\}$ is the final representation of the third sentence;
- "the" $= \{1,0,1\}$ is the final representation for the term "the".

There are several remarks that it is important to do on this model. The first main disadvantage is that, because of the "Bag-of-words" simplification, some important knowledge contained in the text is lost. At a sentence-level, it is lost the semantic relation between the terms in the sentence. Moreover, the logical and grammar structure of the sentence is not maintained, if it is not for Part-of-Speech (PoS) Tagging, explained in the section 3.1, where some of these structure could be taken into account. At a Part of Text level, a paragraph for example, the logical and semantic information between sentences can not be considered by this model.

From a computational and memory perspective, the Bag-of-Words model has also some disadvantages. It is easy to notice that for a collection of books, for instance, it easy to have a Dictionary D with many terms and not all the terms are in common between books, consequently the final matrix M is sparse. Some cleaning techniques can improve the efficiency of this model, for instance, as explained above, with Language Models is possible to reduce the sparsity of the matrix M due to the reduction of the terms in the used dictionary D.

3.3.1 Scoring Words

The elements of the matrix M encode the information for each part of text and terms. In fact, $m_{i,j} = M(i,j)$ is the knowledge saved for the terms $tr_i \in D$ in the Part of text $s_j \in C$.[4] Having said that, the type of information saved depends on the application of the project.

Two main information are considered for each term, based on two levels of analysis: sentence-level and collection-level. In general, terms considered in a sentence give information about the sentence, for instance if the term "inflation" is recurrent in a sentence (or paragraph) if quite likely that the sentence is about inflation. On the other hand, a term viewed in the collection of sentences can give information about its role in the whole text considered. For instance, Stop-words are considered useless because they are too frequent in the collection to be discriminative.

To summarize, different types of weighted schemes are available to choose the right score for the matrix M. In the following some of them are presented:

[4]In this explanation, the part of text are represented by sentences, to simplify the explanation.

- Boolean Model: it is a binary representation $m_{i,j}$ of the term in the sentence. $m_{i,j} = 1$ the term $tr_i \in D$ is present in the sentence $s_j \in C$ and $m_{i,j} = 0$ the term $tr_i \in D$ is not present in the sentence $s_j \in C$.

- Term Frequency Model: the score used to weight the presence of a term is proportional to the frequency. The length of the sentence considered can influence this type of weighting scheme, some adjustments are often made. A common measure used is, where $m_{i,j} = \mathrm{tf}_{i,j}$ and $f_{i,j}$ is the frequency of term w_i in sentence s_j:

$$\mathrm{tf}_{i,j} = \frac{f_{i,j}}{\sum_{tr_k \in s_j} f_{k,j}} \tag{3.2}$$

- Inverse Document Frequency: an inverse document frequency factor is incorporated in the formula to distinguish between relevant and non-relevant terms. To this hand, a common score to take care of the frequency of terms at the collection level is:

$$\mathrm{idf}_i = log(\frac{N}{\mathrm{df}_i}) \quad \forall te_i \in D \tag{3.3}$$

  where $N \in \mathbb{N}$ is the number of sentences (or better Part of text) in the collection C, df_i is a measure of the informativeness of term te_i in the collection, for instance:

$$\mathrm{df}_i = s_j \in C | t_i \in s_j \tag{3.4}$$

  Notice that: the highest is the df_i measure, the less discriminative is term te_i

- Tf.idf Model: it puts together the term-frequency formula with the Inverse Document Frequency one. Finally, the resulting formula is:

$$\text{tf-idf}_{i,j} = \mathrm{tf}_{i,j} * \mathrm{idf}_{i,j} \tag{3.5}$$

A good reference used to compare the results of with the different types of score is "Introduction to Information Retrieval" by Christopher D. Manning.

Chapter 4

Summarization

The idea of automatic text summarizations is straightforward: create a short text that contains most of the meanings and ideas of the original text with some automatic tool. Most of the techniques available nowadays are based on machine learning algorithms. The needs of such automatic ways of summarization derives from the overload of information that in the last decades has been grown. From News articles to social media posts, the data available is a gold mine to be unclosed.

A good definition of automatic text summarization is the following: "Text summarization is the process of distilling the most important information from a source (or sources) to produce an abridged version for a particular user (or users) and task (or tasks)."[25] In other words, three main subjects were involved in our work:

- sources: minutes of the Riksbank and the reports of the research team of SEB Bank;
- users: the research team of SEB and other teams of the trading floor;
- tasks or purpose: discover the key sentence for each board member part of minutes and highlight the most relevant topics discussed.

Most of the data to be analyzed are unstructured, the knowledge in text is not organized in a traditional database. Furthermore, the techniques available are not standardized and most of them have been created in the last few years.

In one of the most cited book on the topic "Automatic text summarizations",[41] the authors provide some reasons to understand the importance of this new field:

- Time: few seconds are needed to summarized big amount of texts.
- Retrieval: if the purpose is to analyze a big amount of texts, such as in information retrieval, more data are available and better organized.

- Objectivity: the bias is standardized in each summary analyzed, and the subjective opinions of the creator are less involved.
- Economical: once the algorithm has been created, humans are involved in few task, furthermore, most of them are open source.

Two main techniques are available today: abstractive-based and extractive-based. The latter is used in this work to take the most “relevant” sentences in the original text and then put them together to create a final summary. Meanwhile, in abstractive-based summarization a new text is created based on the informations retrieved from the original text. Most of this type of technique has been developed in the last few years thanks to great progress in Deep Learning, in particular with Recurrent Neural Networks (RNN). [16]

In the next part on this section, two algorithms used in our approach: Latent Semantic Analysis (LSA) and Textrank.are explained In the end, some insights are given about the difficult task of summary evaluation.

4.0.1 Extractive-based Summarization Algorithms

Extractive-based techniques to create a summary are currently the most used summarization techniques, due to a simplification hypothesis behind: there is no need to create new part of text, the resulting summary is a subset of the original text. [2] In this way, it is not necessary to involve Natural Language Generation Techniques to rewrite the main contents, instead the summary is composed by portions (sentences, paragraphs) of the original text selected by some criteria.

Following this logic, the question is how to choose a principle to select the portions of text to create the final summary. There are many models and in the following section our approach is explained, but first a more abstract formulation of the summarization model is given.

4.0.2 Mathematical formulation of the summarization problem

There are two main points/criteria to be chosen in this model:

- How to select a "relevance" score for each sentence;
- How to extract the sentences by the score and its information.

The original text T could be represented as a collection of portions of text. In this model, sentences are the smallest part of text selected for the summary: $T = \{s_1, \dots, s_i, \dots, s_n\}$.

The summarization algorithm, chosen the function *SS*, associates a score to each sentence:

$$\begin{aligned} SS\colon T \rightarrow \mathbb{R} \\ \forall s_i \in T, \quad SS(s_i) = score_i \in \mathbb{R} \end{aligned} \tag{4.1}$$

The scores are collected in a vector $SS = \{score_1, \dots score_n\}$. It is possible to think about this first phase as a features extraction step. From unstructured data, such as a sentence, a score is obtained, that is a structured feature that describes original data.

The second step of the summarization algorithm starts from the vector *SS* and, by the extraction criteria *EC*, the summary S is obtain :

$$\begin{aligned} EC\colon SS \cup I \rightarrow S \\ where \quad SS, I \in \mathbb{R}^n \quad and \quad S \subset T, \quad |T| = k. \end{aligned} \tag{4.2}$$

where *k* could be chosen using some heuristics and the external information are represented by *I*.

4.0.3 Latent Semantic Analysis

As it is explained in the paragraph/section 4.0.2, the first important question to be solved in Extraction-based text summarization algorithms is to score each sentence by a "relevance" mark from the original text. It is not possible to have a general definition for "relevance", because it depends on the purpose of the task. From here some issues arise because of different contexts and subjects involved. For instance, the concept of relevance used in as heuristic could be valid in Biology, but it could be different from economics context.

In this section, Latent Semantic Analysis (LSA) is explained, it is one of the most used extractive-based summarization algorithm .

LSA is used in NLP(Natural Language Processing) context under the assumption of Distributional Semantics Theory, a sub-field of Linguistic. The main idea is that items that have a similar distribution in the collection have similar meanings. Items are intended as different pieces of the collection, like terms, sentences and paragraphs.[15] A convenient example of distribution of linguistic items is the Vector Space Model. It is used to obtain a distribution of the words in the collection *C* and a distribution of the Part of text desired over the documents in *C*.

The LSA Algorithm is frequently applied in the context of Information Retrieval with the name Latent Semantic Indexing (LSI). The idea is to use the matrix representation for terms and documents used in the Vector Space Model, and create a low-rank representation of that

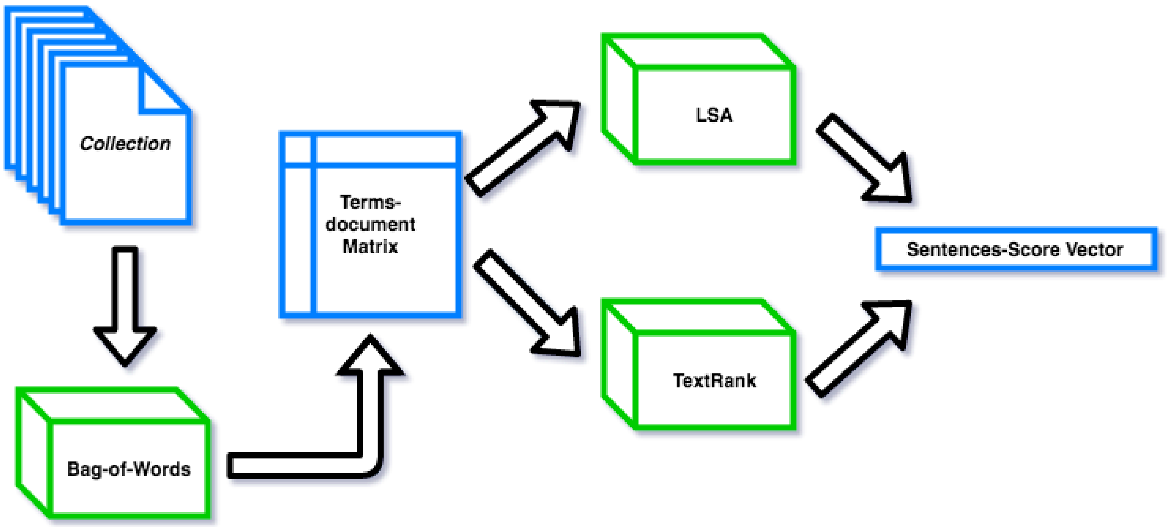


Fig. 4.1 Summary: sentences-score by LSA and TextRank.

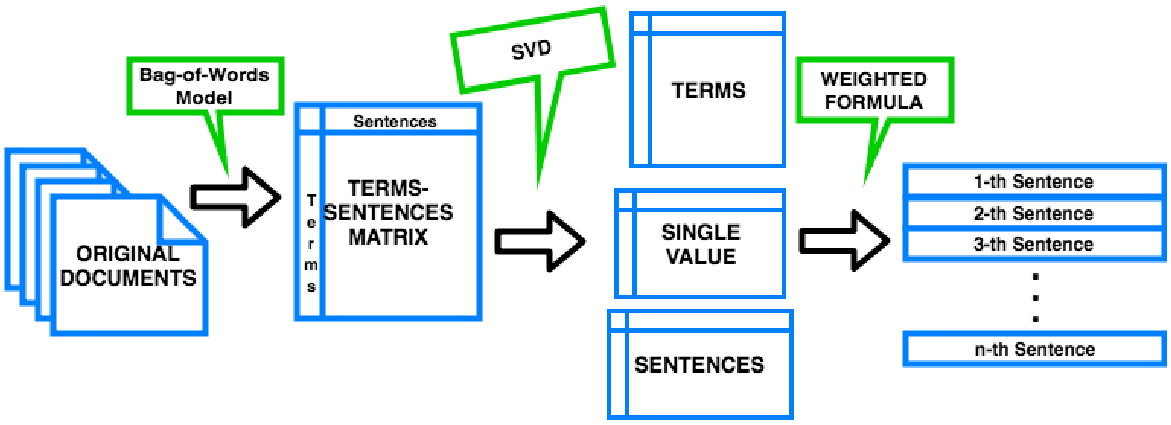


Fig. 4.2 Latent Semantic Analysis:

matrix with a Singular Value Decomposition. After this passage, it is possible to use the new space with less dimensions to get information on the sentences or terms. Moreover, thanks to the sentences-score values, it is possible to "rank" sentences and terms by "relevance".

In the diagram 4.2, LSA could be divided in two steps:

1. Low-rank representation with Single Value Decomposition (SVD);

2. Information extraction from the new space

The terms-docs matrix $M \in \mathbb{R}^{Dxn}$, produced by Bag-of-Words model, section 3.3, is the input for LSA. Without losing generality, suppose that the terms-docs matrix is scored by tf-idf, section 3.3.1, and the dimensions chosen for the subspace is p. Truncated Singular

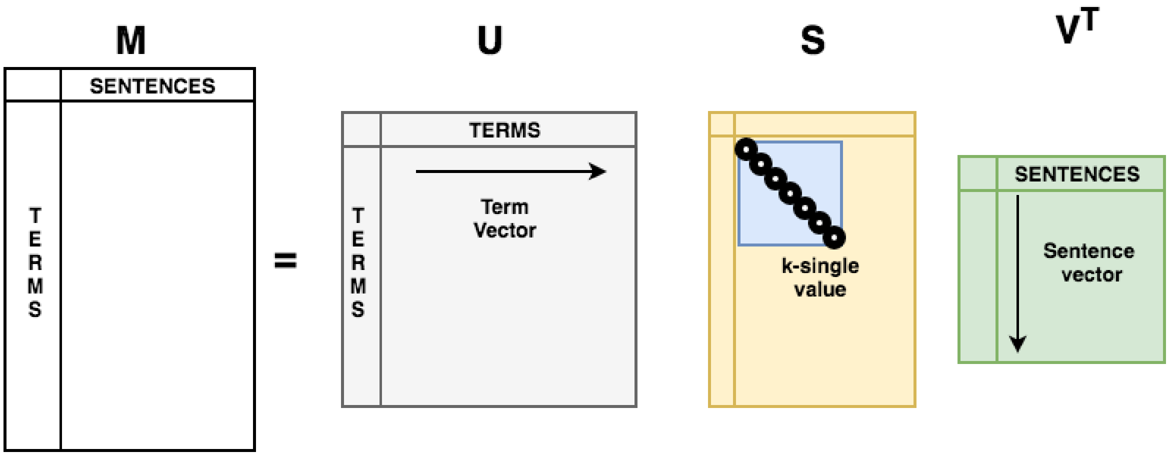


Fig. 4.3 Single Value Decomposition.

Value Decomposition is used to create a low rank representation for sentences and terms. To better understand how LSA works, let's summarize the SVD idea.

In the Figure 4.3, is represented the SVD of the original Terms-docs matrix $M = \{s_1, s_2, \dots s_n\} = \{te_1, \dots, te_D\}^T \in \mathbb{R}^{Dxn}$. In LSA approach, the columns of M are the sentences $s_i \in \mathbb{R}^D$, thus the i-th column contains the information for the i-th sentence and at the j-th place M_{ij} is the $tf.idf$ score of the j-th terms in the sentence $s_i \in C$. In general, it is always the case that $D \gg n$. Furthermore, because of the number of different and new terms for each sentence, M is always sparse.

The Single Value Decomposition, explained in the appendix A, is applied to M and three matrices are obtained, as showed in figure 4.3. The matrix $S \in \mathbb{R}^{Dxn}$ is a diagonal matrix with the single value on the diagonal. The matrix $U \in \mathrm{R}^{DxD}$ contains information for the terms, meanwhile matrix $V \in \mathrm{R}^{nxn}$ has information for sentences. For the summarization purpose, only matrix $V \in$ and matrix D are considered.

"If a word combination pattern is salient and recurring in document, this pattern will be captured and represented by one of the singular vectors."[3]. This results is fundamental to be sure that all the information and pattern in the original matrix $M \in \mathrm{R}^{Dxn}$ are conserved in the low-rank space. Moreover, from a semantic point of view, "the SVD derives the latent semantic structure from the document represented by matrix M.".[38] The original set of sentences is represented in a subspace of $\mathbb{R}^{Dxn}$, such that $p \leq rk(M) = k$, with linearly-independent base vectors. For instances, in our original corpus, set of documents (or set of paragraph or set of sentences), there are 5046 different terms (4012 after text cleaning), the cardinality of the corpus (divided in section, one for each Board member speech) is 46, the terms-docs matrix $M \in \mathrm{R}^{4012x46}$. If $rk(M) = 100$, it is possible to choose $p \leq 100$ such that: A_p, produced by SVD, is the best (see theorem 2 in appendix A) representation in the p-dimensional subspace of the original space generated by M. This approach is called truncated SVD.

As explained above, the patterns of the original documents are represented and conserved in the singular value dimensions. It is important that the subspace do not reconstruct exactly the original terms-docs matrix, in this way truncated SVD removes part of the noise that was in M. Because each of the p-dimensions of the new subspace are a representation of a salient pattern in the document, it is possible to hypothesize that each pattern is a representation of a salient topic in the text. So, if the number of topics a priori is known , therefore p has to be chosen as the cardinality of topics. In the end, with the singular values $\sigma_1 \geq \dots \geq \sigma_p$, Truncated SVD gives a measure of relevance of the topics, because the magnitude of the singular values is a measure of the importance of the pattern in the documents, see appendix A.

In the second part of LDA, a score for each part of text is extracted from the new low-rank space generated by SVD , in our example for each sentence. As introduced in the paper "Generic Text Summarization Using Relevance Measure and Latent Semantic Analysis" by Y. Gong and X. Liu[11], singular value matrix S and right singular vector matrix V^T are used for this purpose. In the new space generated, each sentence s_i is represented by the column vector $v_i = (v_{i1}, \dots, v_{ip}, \dots, v_D)$ from V^T.

To create the score for each sentence based on the SVD decomposition made before, Matrix V^T and S are considered, for each sentence S_i, $i = 1, \dots, n$ is computed a measure of "relevance" with the formula:

$$ss_i = \sum_{k=1}^{n} \sigma_k v_{ki} \tag{4.3}$$

In the end, for each sentence is available a score $R = \{ss_1, \dots, ss_n\}$, sorting the values is obtained a rank of sentences from the most to the less relevant. If the number of sentences required for the summary are l, then the first l sentences are extracted from R in R^l. By a sort operation on R^l based on the original order in the text, it is obtained the final summary.

4.0.4 TextRank

The second summarization algorithm applied is TextRank. It is also an extractive-based algorithm that computes a score of "relevance" for each sentence from the original text, therefore the final summary is a selection of the $k \in \mathbb{N}$ most relevant sentences.

As the name suggests, the core of TextRank algorithm is based on the PageRank algorithm[30], the quite famous algorithm at the base of the original search engine Google to rank website. PageRank algorithm is applied on the World Wide Web dataset to understand which are the most important websites. To summarize the idea of PageRank, the score for

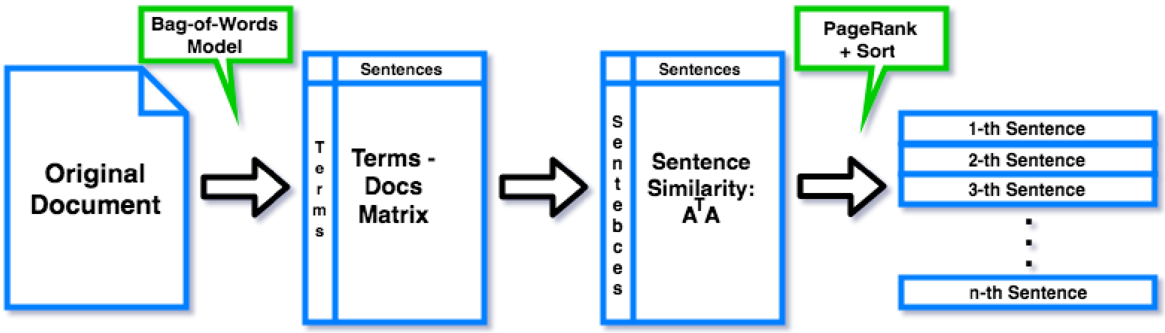


Fig. 4.4 TextRank: algorithm's flow.

each website is computed by the number and quality of links that websites share, therefore the assumption is that the most important websites are likely to receive links and the relevance of the other websites is significant.

The World Wide Web is represented as a directed weighted graph $G = \{V, E, f\}$, where V is the set of nodes that represents websites, $\forall\ v_i \in V$ there are some in-edges $IN_i(E)$, $e_{ji} \in E$ a generic edge from v_j to v_i, and out-edges $OUT_i(E)$, $e_{ij} \in E$ a generic edge from v_i to v_j. The function $f\colon E \to \mathbb{R}$ assigns a weight for each edge, such that:

$$f(e_{ij}) = a_{ij} \in \mathbb{R}, \qquad \forall e_{ij} \in E \tag{4.4}$$

The weighted scheme for the PageRank is just binary, $f(e_{ij}) = 1$ is the is a connection between node i and node j, otherwise $f(e_{ij}) = 0$. The formula of the PageRank to get the rank of each website is the following:

$$PR(v_i) = \frac{(1-\lambda)}{N} + \lambda \sum_{v_j \in IN_i(E)} \frac{PR(v_j)}{|OUT_j(E)|} \qquad \forall v_i \in V \tag{4.5}$$

where N is the total number of websites and λ is a damping factor.

In the figure 4.4, the main steps of the TextRank algorithms are reported. Starting from the terms-docs matrix A, Bag-of-Words model (see section 3.3) is applied to create a space representation of the original document with the purpose of comparing distances between the sentences (Part of Text) of the original document.

In TextRank algorithms a weighted Undirected graph is adopted, where nodes are Sentences (or more in general Part of text), and edges are weighted by a similarity score between the two sentences connected. In other words, the similarity between two sentences is a distance measure, usually the Cosine similarity. Furthermore, because the distance is symmetric between two sentences the graph is Undirected.

The next step is to compute the similarity between each sentence. It is the same to say that a product between the transpose of the terms-docs matrix is calculated M^T and M, with a normalization for each element. So, the resulting matrix $B \in \mathbb{R}^{nxn}$ is a square matrix with the dimension of the number of sentences n, so that:

$$\forall s_i, s_j \in V, \quad B_{ij} = \begin{cases} \text{sim}(s_i, s_j) = \text{sim}(s_j, s_i) \in (0,1], & if \quad i \neq j \quad \& \quad \text{sim}(s_i, s_j) > \varepsilon; \\ 0 \quad if \quad i = j \quad or \quad \text{sim}(s_i, s_j) < \varepsilon. \end{cases} \tag{4.6}$$

The similarity measure used in this application is the Cosine distance:

$$w_{ij} = \text{sim}(s_i, s_j) = \frac{A_i^T A_j}{\|A_i^T\| \|A_j\|} = \frac{\sum_{k=1}^{m} A_{ik}^T A_{jk}}{\sqrt{\sum_{k=1}^{m} (A_{ik}^T)^2} \sqrt{\sum_{k=1}^{m} (A_{jk})^2}} \tag{4.7}$$

TextRank, as PageRank, uses a graph to represent the information between the sentences, therefore the final similarity matrix B is the weighted scheme of the graph G described above. Note that the Graph G is not completed-connected, in fact, there are no self-loop and some connections are missing.

The last step of the algorithm, as represented in the right part of figure 4.4, could be divided in three main steps:

1. Modified PageRank algorithm;
2. Sort operation;
3. Selection of the first k sentences.

The formula of the Modified PageRank for the graph G is:

$$TR(s_i) = \frac{(1-\lambda)}{N} + \lambda \sum_{s_j \in IN_i(E)} \frac{w_{ji} TR(s_j)}{\left(\sum_{s_k \in OUT_j(E)} w_{ik} \right)} \qquad \forall s_i \in V. \tag{4.8}$$

There are some differences between 4.8 and 4.5. In fact, TextRank graph is weighted. The similarity between two sentences w_{ij} is used both to weight the influence of the TextRank score on another sentence $TR(s_j)$ and to normalize this influence by the total weight of the out-edges of s_j.

The output of the algorithm is a score for each sentence that represents the importance of the sentence in the document. Then, it is applied a sort operation on the sentences based on the weight $TR(s_i) \in (0,1)$. In the end, the first k sentences are selected.

4.0.5 Summary Evaluation

One of the main issue of summarization algorithms is the evaluation part. The problem arises because there is not a truth summary to compare the algorithm's summary output. In other words, even if an approved version of the summary is used as a label of the original text (to have as a standard truth to compare with the output) there is not a standard way to have a quality of the summary obtained.

In this project, thanks to the database of the reports released by the Research team of SEB Bank 2, there are this trusted summary made by humans. Indeed, for each Minutes and for each board member, the summary produced by the staff of the SEB Bank's Researchers is available . In this way, it is possible to test the algorithms on the historical data, where there are both the summary, the automatic and the human based, in order to get some insights on the quality of the algorithm.

Two main quantitative techniques have been used to "grade" the quality of both Textrank and LSA, with:

- text similarity: both the automatic summary produced by the algorithm and the human summary are translated in a vector space by the Vector Space Model with the dictionary of terms from the original text (see section 3.3), then the two vectors obtained are compared with the cosine similarity and a score is obtained.
- Recall-oriented: the two texts are compared by counting the number of overlapping units, such as n-grams or word sequence. [23]

Furthermore a third way is the evaluation made by humans, as a qualitative evaluation to compare the two summary. In our approach, this was in part possible due to the collaboration with the Research team of SEB Bank. The latter is the best way to evaluate the summary, even if there are two main drawbacks. First, it is time-consuming and economically expensive. Furthermore, it is also subjective to the person involved in the evaluation, so the truth is relative to the evaluator.

4.0.6 Weighted Formula with topics

The final algorithm used in our program is a topic-based summarization algorithm, with topics based on human choice. The idea is to use the Textrank Algorithm on the topic-based dataset explained in the section .

It means that for each board member *BM* and for each topic t_{BM} of the board member, the Textrank algorithm is applied on that portion of text. Therefore, for each board member and for each topic the algorithm gives the rank of the sentences from the most important to the least important, as it is explained above. Then we obtain:

$$\begin{cases} S(t_j, BM_i) = \{S(t_j, BM_i)^1, ..., S(t_j, BM_i)^{M(t_j, BM_i)}\} \\ \forall j = 1, ..6 \text{ and } \forall i = 1, .., T \end{cases} \tag{4.9}$$

where $M(t_j, BM_i)$ is the number of sentences about the topic t_j in the board member BM_i section.
An example can be useful to understand better how the output is composed:

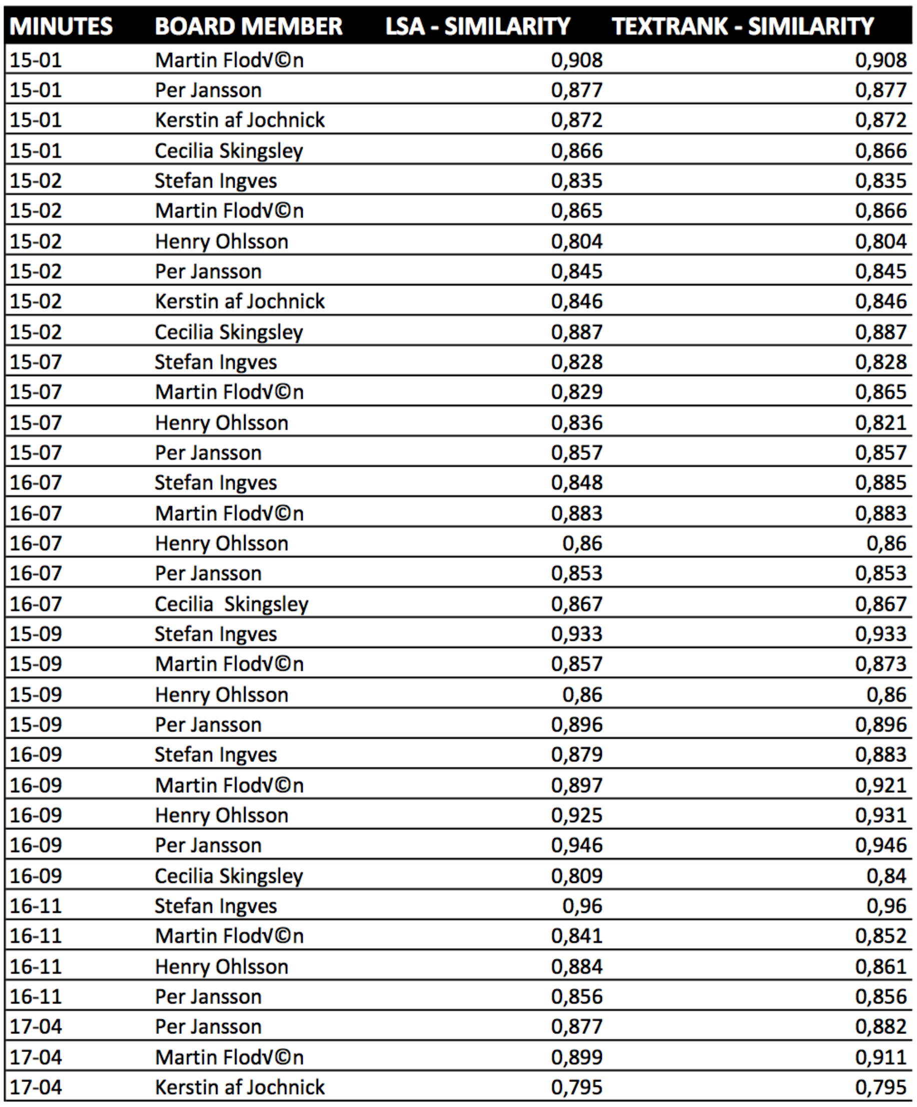
$$S(\text{rate}, \text{Cecilia}) = \{S(\text{rate}, \text{Cecilia})^1, ..., S(\text{rate}, \text{Cecilia})^{10}\} \tag{4.10}$$

It is the rank of the sentences about the topic *rate* said by *Cecilia Skingsley*. To obtain a unique summary for each board member the ranks of different topics have to be combined in a proper way. To do that, the first sentence in the rank for each topic is picked. If two or more of the selected sentences are the same the algorithm takes the second one in the rank. The algorithm continues to check if two or more sentences are repeated and if it is so it continues to goes lower in the rank until the summary is composed of all different sentences. Finally, the set of the topic-based most important sentences selected S are sorted on the order of the original text to get the final summary.

A possible improvement of this approach could be to base the choice of the topics used to create the summary on a Topic model.

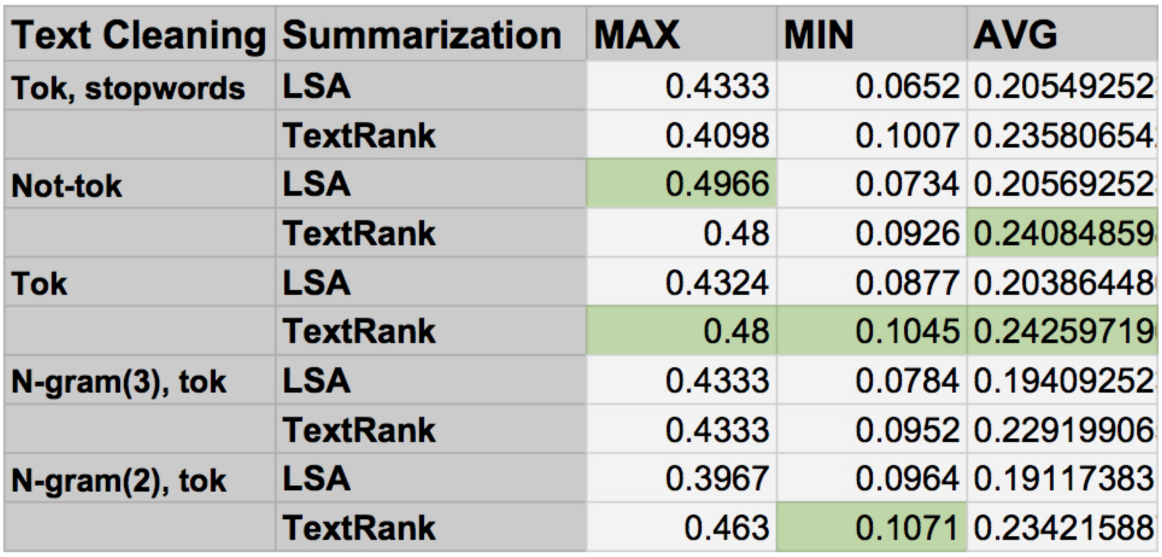
The summarization algorithms have been applied on the two main datasets described in section (2): board member-level and minutes-level. Furthermore, the two main algorithms Textrank and Latent Semantic Analysis have been evaluated with three techniques, the first two are the text similarity and the Recall-oriented evaluation, the latter is a human based evaluation performed by the members of the Research Team of SEB Bank.

The less discriminative evaluation has been the Text similarity. In fact, as showed in picture 4.5, the score is computed both for TextRank and LSA from 0 to 1. Two observations have to be done. The two scores for a certain board member in a minutes are in many



MINUTES	BOARD MEMBER	LSA - SIMILARITY	TEXTRANK - SIMILARITY
15-01	Martin FlodV©n	0,908	0,908
15-01	Per Jansson	0,877	0,877
15-01	Kerstin af Jochnick	0,872	0,872
15-01	Cecilia Skingsley	0,866	0,866
15-02	Stefan Ingves	0,835	0,835
15-02	Martin FlodV©n	0,865	0,866
15-02	Henry Ohlsson	0,804	0,804
15-02	Per Jansson	0,845	0,845
15-02	Kerstin af Jochnick	0,846	0,846
15-02	Cecilia Skingsley	0,887	0,887
15-07	Stefan Ingves	0,828	0,828
15-07	Martin FlodV©n	0,829	0,865
15-07	Henry Ohlsson	0,836	0,821
15-07	Per Jansson	0,857	0,857
16-07	Stefan Ingves	0,848	0,885
16-07	Martin FlodV©n	0,883	0,883
16-07	Henry Ohlsson	0,86	0,86
16-07	Per Jansson	0,853	0,853
16-07	Cecilia Skingsley	0,867	0,867
15-09	Stefan Ingves	0,933	0,933
15-09	Martin FlodV©n	0,857	0,873
15-09	Henry Ohlsson	0,86	0,86
15-09	Per Jansson	0,896	0,896
16-09	Stefan Ingves	0,879	0,883
16-09	Martin FlodV©n	0,897	0,921
16-09	Henry Ohlsson	0,925	0,931
16-09	Per Jansson	0,946	0,946
16-09	Cecilia Skingsley	0,809	0,84
16-11	Stefan Ingves	0,96	0,96
16-11	Martin FlodV©n	0,841	0,852
16-11	Henry Ohlsson	0,884	0,861
16-11	Per Jansson	0,856	0,856
17-04	Per Jansson	0,877	0,882
17-04	Martin FlodV©n	0,899	0,911
17-04	Kerstin af Jochnick	0,795	0,795

Fig. 4.5 Text-similarity evaluation score by LSA and TextRank.



Text Cleaning	Summarization	MAX	MIN	AVG
Tok, stopwords	LSA	0.4333	0.0652	0.20549252
	TextRank	0.4098	0.1007	0.23580654
Not-tok	LSA	0.4966	0.0734	0.20569252
	TextRank	0.48	0.0926	0.24084859
Tok	LSA	0.4324	0.0877	0.20386448
	TextRank	0.48	0.1045	0.24259719
N-gram(3), tok	LSA	0.4333	0.0784	0.19409252
	TextRank	0.4333	0.0952	0.22919906
N-gram(2), tok	LSA	0.3967	0.0964	0.19117383
	TextRank	0.463	0.1071	0.23421588

Fig. 4.6 Recall-oriented evaluation by rouge for LSA and TextRank.

cases the same number, this is because the two algorithms perform quite similar on both the datasets. In other words, the two extractive-based summarization algorithms retrieve quite often the same sentences. The second observation regards the difference between the scores when they are not equal. In fact, when LSA retrieve at list one sentence different from Textrank, the gap is a few cents. For this reason this type of evaluation has been used only to understand when the two algorithms give the exact same set of sentences.

Recall-oriented sentiment analysis is based on the software Rouge [23]. The results are represented in the picture (4.6) for both the algorithms. Furthermore, the rows of the tables represents different types of text cleaning. The score performed with Rouge has been applied to all the summary created by the algorithms for each board member. The algorithm compare the summary created by the algorithm with the summary performed by the research team of SEB Bank, contained in the dataset SEB Reports described in section (2.2). In other words, the algorithm in Rouge compare the historical summary created by humans with the output of the algorithms for the summarization. The measure computed by the algorithm is the F1-score of the terms, ngrams in the two summary.

The results in picture (4.6) are the maximum (MAX), minimum (MIN) and average (AVG) of the score performed by rouge of each minutes and for each board member.

Finally, the Human-based evaluation has been conducted with the help of the research team of SEB Bank. This type of evaluation gives a lot of information of the quality of the summary. Furthermore, the accuracy of the algorithm on the capacity to retrieve the key concepts stated by the board member is easily performed by this type of evaluation.

Human-based evaluation for summarization algorithms has been the best way to really understand the quality of the summary, the only drawback is the time. In fact, many summaries have been read to understand which one was the best techniques.

Chapter 5

Sentiment Analysis

5.1 Introduction

The information overload issue described in the introduction has caused two main problem: the difficulty in understanding and the complexity of decision making.[42] It is crucial in our society to be able to collect information, but even more to be able to understand and actively use them.

Since the beginning of Web 2.0, a lot of opinionated text has recorded in digital format from social media and media communication sources, just to give some example: reviews on e-commerce websites, chats for interpersonal communications, comments on blogs and social networks.

The natural consequence of the availability of this new sources of information is the growing number of researches in both Linguistic and Machine Learning, to develop techniques able to discover the opinion contained in texts. The umbrella name to call all this researches is Sentiment Analysis.[1]

In the last few years, social media analysis has increased popularity, and sentiment analysis is a core part of it, with the purpose of: "to extract from the social media content is what people talk about and what their opinions are.". Moreover, the idea of studying the opinion holders is gaining popularity, social analysis and customer profiling being the aims.

With the data available in our project, it is straightforward to apply sentiment analysis techniques to better understand the opinion of the Board Members of Riksbank. As explained in the economical introduction 1.1, one of the two goals of this project is to identify the Hawkish-Dovish position for each Board member in a Minutes.

[1]Notice that the field of Sentiment Analysis is also called: opinion mining, opinion analysis, opinion extraction, sentiment mining, subjectivity analysis, affect analysis, emotion analysis, and review mining.

After a review of the main techniques used in sentiment analysis[24], we have chosen to apply *Supervised algorithms*. They are considered as a mature and successful solution in traditional topical classification and they have been adopted and investigated for opinion detection with satisfactory.[12]
Supervised learning is defined as a machine learning task of learning a function that maps an input to an output based on example input-output pairs[35]. It infers a function from labeled training data consisting of a set of training examples.[28]
A Supervised Classifier is used, a special case of a supervised algorithm, that leans from the dataset composed of the past minutes, labeled by the SEB's research team, as is explained in section 2.2.

In the following paragraphs, initially, an in-depth overview is given of opinion mining, then our approach is explained and finally the results are evaluated.

5.2 Opinion and Sentiment

"Sentiment analysis, also called opinion mining, is the field of study that analyzes people's opinions, sentiments, appraisals, attitudes, and emotions toward entities and their attributes expressed in written text.".[43] The definition given by Bing Liu describes and highlights the main aspects of sentiment analysis. It is important to describe the difference between sentiment and opinion, and how they are related.

In his work, Bing Liu has explored this difference, two sentence are given as an example: *"I am concerned about the current state of the economy"* and *"I think the economy is not doing well"*. By the first sentence, the speaker expresses his/her feeling and implies a negative opinion about the economical situation, instead the second sentence reveals the concrete view point of the author and express his/her opinion on it. The difference it is not so remarkable, however it is possible to classify the first sentence as a sentiment expression and the second as an opinion.

To summarized, "sentiment" is used "to mean the underlying positive or negative feeling implied by opinion"[43] and "opinion to mean the whole concept of sentiment, evaluation, appraisal, or attitude and associated information, such as the opinion target and the person who holds the opinion"[43].

In review mining, where sentiment analysis is applied to reviews of products ore subjects, the problem of opinion mining is usually divided into sub-problems. An *entity* is identified first, that is the subject/product under investigation, and the attributes/characteristics of the entity are designated as *aspects*. Bing Liu, as state-of-the-art in Aspect-Based Sentiment Analysis, gives a comprehensive definition of opinion from a theoretical point of view in terms of quintuple:

$$\text{(Entity, Aspect, Sentiment, Holder, Time)} \tag{5.1}$$

where:

- Entity: it is the target entity/object of the analysis;
- Aspect: it is the aspect/feature of the entity analyzed;
- Sentiment: it is score given to identify the sentiment of the opinion;
- Holder: it is the opinion holder;
- Time: it is period of validity for the sentiment on the opinion.

Furthermore, the sentiment analysis goal, given a document d, is to extract all the quintuples from d.

5.3 Our Approach

To apply this idea on our data, it is important to understand where the information, to complete each quintuple, is hidden. Recall that in this project the goal of sentiment analysis is to uncover the polarity of the Board Members concerning monetary policy decision in the last minutes released by Riksbank.

Concerning the Aspect-Based model 5.1, it is straightforward to identify the *Time t* with the date of the minutes under analysis. Therefore, since there are 30 Minutes available: $t = 1, \dots, 30$. The *Opinion Holder h* are the Board Members, so $h = \{$Stefan Ingves, Kerstin af Jochnick, Martin Flodén, Per Jansson, Henry Ohlsson, Cecilia Skingsley$\}$.

About the *Entity* and *Aspects*, a more complex analysis has to be done. The *Entity* could be identified, for all the opinion holder h and independently from the time t, as the "Monetary policy decisions". The *aspects* that describe the entity can be various, in our work we decided to tackle the problem by different approaches:

- Entity-based: the model for the opinion description is simplified as:

  $$\text{(Entity, Sentiment, Holder, Time)} \tag{5.2}$$

  Therefore, the hypothesis is that there are no aspects that particularly describe the monetary decision of a Board Member. For each h, all the part of text dedicated to him/her in the minutes are used to analyzed the *sentiment*.

- Fixed-aspects: the aspects are independent of time, therefore are the same for each minutes. These aspects are the Topics suggested from the Research Team of SEB, and are: $a = \{\text{Inflation, repo rate, rate, Krona}\}$, see section 4.0.6.
- Dynamic-Aspects: the aspects could change over time and depends on the topics debated in the Board Members meeting.

To clarify the interpretation for the *Entity* and *Aspects*, a distinction on the different levels of analysis should be done. In general, it is possible to use as input in the analysis different part of text: the document itself or, paragraphs or sentences.

In the Supervised approach, described in section 5.4, all the sentences are used as a unique text, therefore the analysis is at a '*document-level*'. Specifically, the text is divided in six section, one for each Board member, then in this case is more correct talking about a 'Board member-level' (BM-level).

The *Sentiment* is interpreted as a binary class to label each Board Member h, $s = \{\text{Dovish, Hawkish}\}$. Therefore, the Sentiment analysis issue is translated in a classification problem. In Supervised approach, the results can be translate in a score $\{0-1\}$. The score gives the possibility to compare the results, for instance if a Board Member has a score of 0.1 and another one has 0.6, it could be inferred that the latter is more dovish. Therefore, a rank can be created.

5.4 Supervised algorithm

In Supervised sentiment analysis there are three main step to focus on[31], highlighted in green in figure 5.1:

- Tokenization: this step is analyzed in section 3.1;
- Feature Extraction: features are extracted from the test-set, with some criteria further discussed, and used to build the Design matrix for both the test-set and the training-set.
- Classification Model: the Training-set Design matrix is used to train the classification model. In the next sections are explained the classifiers used: SVM, Logistic regression and a Bayesian Network with a particular focus on the Naïve-Bayes model.

5.4.1 Feature extraction

Text data are very unstructured, therefore to extract features from the text is not straightforward and also to find a criterion to select the best features is challenging. The idea of feature extraction is to learn patterns in data, to give a more organized shape to the information available. Usually, the feature extraction and selection algorithms use vectors of numbers, but with text data, there is also the problem of how to encode the words to obtain vectors. Therefore, feature extraction and selection are two fundamental phases of the supervised approach: the first to get the right information available to be analyzed and the latter to optimize information given to the algorithms. The models used and described in this section follow the idea of the Bag-of-Words model to encode words, described in section 3.3 and are presented in order of complexity.

Three weighted schemes are used for the Bag-of-Words model:

- Frequency-based: is used only the frequency of a term in a document and the features are the first k most important words;
- Tf-idf: the collection of document is used to dump the frequency of the terms, see section 3.3.1;

As shown in figure 5.1, the same techniques used to extract the features in the train-set must be used for the test-set and then in the prediction phase with the new document.

5.4.2 Classification Models

After the feature extraction, the design matrix is ready to be used to train the classifier. There are many types of classification algorithms used with text data, Support Vector Machines, Logistic Regression and Bayesian Network are the state-of-the-art of these techniques, in the next sections theory behind these models is explained.

5.4.3 Support vector machine

The support vector machine classifier has been developed in the 1990's and it is still widely used nowadays.[18] It is based on the intuition to divide the two classes of observations using a hyperplane.

The model has evolved, has been generalized and now more advanced functions can be used to separate the two or more classes, as polynomial functions and radial basis functions.

To present the mathematical model, the linear binary classifier is described. The algorithm seeks for the hyperplane that maximizes the distance between itself and the observations of the two classes. Just the closest observations to the predicted hyperplane are involved in the definition of the hyperplane itself, and they are called *support vectors*, the other observations are independent of the hyperplane position. This total dependence, between the classification

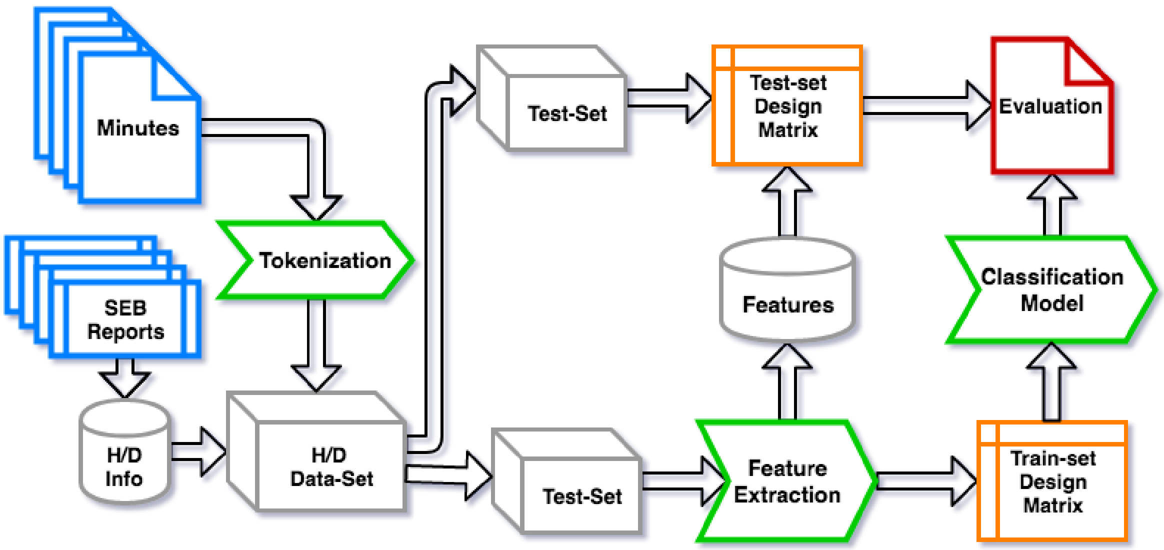


Fig. 5.1 Sentiment Analysis: Supervised algos.

hyperplane and a few observations, would make the result too sensitive to small changes in the support vectors. For this reason, some misclassification errors are tolerated to reach a greater robustness of the algorithm. The mathematical formulation of the problem is the following:

$$\max_{\beta_0,\beta_1,\ldots,\beta_p,e_1,\ldots,e_n} M \tag{5.3}$$

$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1, \tag{5.4}$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - e_i), \tag{5.5}$$

$$e_i > 0, \quad \sum_{i=1}^{n} e_i \leq C, \tag{5.6}$$

The optimization problem has is optimum when the distance between the hyperplane and the support vectors, called *margin*, is maximized.
If $\beta_0, \beta_1, \ldots, \beta_p$ are the coefficients of the maximal margin hyperplane, then the maximal margin classifier classifies the observation $x_i = (x_{i1}, \ldots x_{ip})$ based on the sign of $f(x_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}$.
We are describing a binary classifier then y_i is a binary variable that indicate the predicted class of the observation x_i and it can assumes the two values $\{-1, 1\}$. The constrain (5.5) guarantees that each observation is on the correct side of the hyperplane, with some exception given by the presence of the slack variables $e_1, \ldots, e_n$. The constrain 5.6 gives a upper-bound

C for the number of allowed misclassifications.

Usually, data are not linearly separated, furthermore, the most used version of this classifier use a more complex shape to divide the data into classes. To describe it, the notion of *kernel* function has to be introduced. It could be seen as a generalization of the inner product. The most used are:

$$\begin{aligned}\text{Linear kernel:} K(x_i, x_{i'}) &= \sum_{j=1}^{p} x_{ij} x_{i'j};\\ \text{Polynomial kernel:} K(x_i, x_{i'}) &= (1 + \sum_{j=1}^{p} x_{ij} x_{i'j})^d;\\ \text{Radial kernel:} K(x_i, x_{i'}) &= exp(-\gamma \sum_{j=1}^{p} (x_{ij} x_{i'j})^2), \quad \gamma > 0.\end{aligned} \tag{5.7}$$

It can be shown that the solution of the classification algorithm depends just on the inner products of the observations. The boundary function has the form:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i). \tag{5.8}$$

In our work, the observations are the parts of text and the features are the words, encoded with the Bag-of-Words model. We have chosen to use the *radial kernel* in the algorithm, that better follows the shape of the observations in the vector space.

5.4.4 Logistic Regression

Logistic Regression model can be used as a classifier even if the final output is not a class, but the probability that the target belongs to the class. As the name suggests, the model is linked with the Linear regression, and the latter can be used to derive his formulation.
The Linear Regression model is used to predict a probability that a variable belongs to a class and is express by the following

$$Pr(Y = s|X) = \beta_0 + \beta\, \mathbf{X}. \tag{5.9}$$

Where X are the features of the model, β_0 and β are coefficients that have to be estimated and Y is the target variable. In our work we face a binary classification problem, then, as we specified before, the possible values of s are encoded with $0, 1$.
The main problem of the model stands out, there are no constraints on the values of the

Confusion matrix	**classified hawkish**	**classified dovish**
hawkish	TH	FD
dovish	FH	TD

Table 5.1 Confusion Matrix.

probability, that should assume values between 0 and 1. To solve this problem the *logistic function* is applied to the right part of the equation

$$P(X) = \frac{e^{(\beta_0+\beta X)}}{1+e^{(\beta_0+\beta X)}} \tag{5.10}$$

With a bit of manipulation the result is

$$\frac{p(X)}{1-p(X)} = e^{(\beta_0+\beta X)}. \tag{5.11}$$

The quantity $p(X)/(1-p(X))$ is called the *odds* [17], and it can assume any value odds between 0 and ∞.
Applying the logistic function to both sides of the equation we obtain

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta X. \tag{5.12}$$

In our work, X are the words encoded with the Bag-of-Word model and p defines the probability that a text belongs to the Dovish class. The parameters β are estimated by the algorithm using the trainingset.

5.4.5 Evaluation

The classical way to study and compare the robustness of a model are the *F1 score* and the *accuracy* score. The latter is defined as 'the accuracy of a measurement system is the degree of closeness of measurements of a quantity to that quantity's true value'[9]. Or in other words, the probability of having a correct classification using the given classifier, express in the following way

$$\text{accuracy} = P(\text{hawkish}|\text{class.hawkish}) + P(\text{dovish}|\text{class. dovish}) \tag{5.13}$$

When the classifier is binary, as in this case, the accuracy can be estimated using the concepts of true/false negative/positive. In particular, in this work, the two classes positive and negative

are replaced by Hawkish and Dovish:
True Dovish (TD) = the number of cases correctly identified as Dovish
False Dovish (FD) = the number of cases incorrectly identified as Dovish
True Hawkish (TH) = the number of cases correctly identified as Hawkish
False Hawkish (FH) = the number of cases incorrectly identified as Hawkish
Using that notation the accuracy is

$$\widehat{accuracy} = \frac{TD+TH}{TD+TH+FD+FH} \tag{5.14}$$

Both the estimated and the theoretical accuracy reach the best value at 1 and worst at 0. The accuracy score is useful to have a general idea about the goodness of the model but it could mislead, especially with an unbalanced dataset.
The F1 score can help in this situation, because it is not bias, even if a class if much bigger then the other one.
To define it, we have to define two others score, the *precision* and the *sensitivity*. Using the same notation as before, we have

$$\text{sensitivity} = P(\text{dovish}|\text{ class. dovish}) \tag{5.15}$$

and

$$\text{sensitivity} = P(\text{hawkish}|\text{ class. hawkish}) \tag{5.16}$$

Also in this case the two scores can be estimated using the observations:

$$\widehat{precision} = \frac{TD}{TD+FD} \tag{5.17}$$

$$\widehat{sensitivity} = \frac{TD}{TD+FH} \tag{5.18}$$

The sampling precision indicates the 'True Dovish' over the total number of occurrences classified as dovish, instead the sampling sensitivity indicates the 'True Dovish' over the total number of occurrences that are actually dovish.
The F1 score is the harmonic average of the precision and sensitivity

$$\text{F1} = 2 * \frac{sensitivity * precision}{sensitivity + precision} \tag{5.19}$$

The relative contribution of precision and recall to the F1 score are equal. As the accuracy, the F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. In our project the entire dataset is balanced because, in each meeting, two Board members are

classified as Hawkish and two as Dovish. Nevertheless we used this score to confirm the accuracy value.
The dataset used in this work contains around one hundred occurrences and it is considered a small dataset. For this reason, a simplistic division of the dataset in training-set and test-set can lead to biased evaluations of the models. To avoid it, in this chapter it is always used a cross-validation approach.

5.4.6 Results

As we explained before, in our model the features are the words that appear in the text. It is possible to apply different types of weight to consider them, the Python library *scikit-learn* gives the possibility to use the frequency weight (Bag-of-words) and the Tf-idf weight. The first type of weight is rougher, it does not consider that a word that appears in all the documents is less important than a word that characterizes a specific document. Both using the SVM and the Logistic regression, the classifier performs better with the Tf-idf weight.
The two classifiers perform really well, they both reach an accuracy and an F1 score level higher than 90%, using the Td-idf score. Probably during the training phase, the algorithm connects each name of the Board members with his opinion, given by the label of the text. Often the Board members maintain the same economical opinion, then also the same label, over the years and this helps the model to predict every time the right class.
The results obtained with the Logistic Regression model are slightly better than the ones obtained with the SVM model, furthermore, the output of the first one is the probability to belong to the *dovish* class and it can be used to rank the Board members.
The SEB's research team suggests that ranking the Board members, from the most Dovish to the most Hawkish, is more useful than classifying them in just two classes. Except for a couple of them, that are more radical about them opinion, the others cannot be considered as Hawkish or Dovish, but something in between. A proper evaluation of the rank is more complicated than the one of a binary classification but some qualitative observations can be done. The more extremes scenarios are always detected, there are more problems in the central positions, the third and the fourth position sometimes are swapped, but it is typical of a classification problem.
The evaluation of the results has been done using the cross-validation technique, because of the reduced dimension of the dataset.
The results of the two models are summarized in the tables 5.2 and 5.3.

SVM	BoW	Tf-idf
Accuracy	0.863	0.922
F1	0.857	0.9191

Table 5.2 Confusion Matrix for Support Vector Machine Algorithm in Supervised-Entity Model.

Log regression	BoW	Tf-idf
Accuracy	0.841	0.95
F1	0.833	0.9487

Table 5.3 Confusion Matrix for Logistic Regression in Supervised-Entity Model.

5.5 Bayesian Network

5.5.1 Introduction

The Bayesian networks can be used as a classifier. In this work, they are shown to be an innovative tool to perform sentiment analysis. In this application the features of the model $A_1,...A_n$ represent the frequency of the given words in the text and they can assume the integers values $a_1,...a_n$. The Bayesian Network (B) encodes a distribution $P_B(A_k,...,A_n,C)$ that describes the data. The resulting model can be used in a way that given the set of values $a_1,...,a_n$, the classifier based on B returns the label c that maximizes the posterior probability $P_B(c|a_1,...,a_n)$.

This approach is justified by the asymptotic correctness of the Bayesian learning procedure. Given a large data set, the learned network will be a close approximation for the probability distribution governing the domain (assuming that instances are sampled independently from a fixed distribution)[10].

One of the most effective classifiers, in the sense that its predictive performance is competitive with state-of-the-art classifiers, is the so-called Naive Bayesian classifier described, for example, by Duda and Hart (1973)[6] and by Langley et al. (1992)[22].[10]. It is a special case of Bayesian network with the unrealistic assumption of conditional independence between the variables.

As always in this chapter, the goal is to classify each Board member in one of the two classes, Hawkish and Dovish,, using the frequency of the words as features $A_1,...A_N$of the network,. In the following paragraph we explain the theory behind the Bayesian networks and three models based on them. We will try to find a trade-off between the complexity of the model and number of features, for a Bayesian network classifier and the Naive Bayes classifier.

5.5.2 Bayesian statistics

Bayesian statistic is a statistics theory where the evidence about the true state of the world is expressed in terms of degrees of belief[8] or Bayesian probability. Bayesian probability is an interpretation of the concept of probability, in which, instead of frequency or propensity of some phenomenon, probability is interpreted as reasonable expectation or represents a state of knowledge[19]. Thus, we have a combination between prior assumption about the model and the evidence of the observations. In other words, probability is an orderly opinion; and inference of data is nothing else than the revision of such opinion in the light of relevant new information.[7].
The formulation of statistical models using Bayesian statistics has the identifying feature of requiring the specification of prior distributions for any unknown parameters. Indeed, parameters of prior distributions may themselves have prior distributions, leading to Bayesian hierarchical modeling, or may be interrelated, leading to Bayesian networks.

5.5.3 DAG

A Bayesian network can be described using a graphical scheme, a DAG, that is a special case of *graphical model.*
A graphical model is a tool used to illustrate in a visual way and work with conditional independence between variables in a given problem. The conditional independence notion between two variables could be explained by the absence of a direct impact on each other's value. Using a notation common in this field, given three random variables (X,Y,Z), the conditional independence of X from Z given Y is true if $P(X|Z,Y) = P(X|Y)$. A shared notation for conditional independence is $X \perp Z|Y$.

In Graph Theory, a graph G is usually indicated with a set of nodes V and a set of edges E, so $G = \{V,E\}$. In Graphical Models theory, nodes are random variables and edges represent causal relationships between variables. If the edge is directional the direction is from the cause variable to the effect variable, otherwise, if there is only a correlation between two variables, the edge is undirected. DAGs (*Directed Acyclic Graphs*) are graphical model with specific characteristics. Bayesian networks are represented graphically using DAGS, graphical models with no undirected edges and no cycles, as the one shown in figure 5.2. Each vertex in the graph represents a random variable, and edges represent direct correlations between variables. Specifically, each variable is independent of its non-descendants in the graph given the state of its parents. Using this tool the joint probability of the network is easily computed and the number of evaluated dependences is minimum.

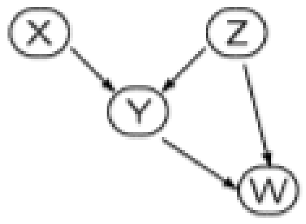


Fig. 5.2 DAG

Formally, a Bayesian network that is composed by a set of random variables **V**, is a pair $B = (G, \Theta)$. G is the directed acyclic graph that describes the network and Θ represents the set of parameters that quantify it. For example, Θ defines the probability that the nodes of the graph $X = (X_1, ...X_n)$ assume the value $x = (x_1, ..., x_n)$.
A Bayesian network defines a unique joint probability distribution over **V**, according to the connection in the DAG given by:

$$P_B(X_1, ...X_n) = \prod_{i=1}^{n} P_B(X_i|\Pi_{X_i}) \tag{5.20}$$

where Π_{X_i} are the set of parents of X_i in G.

5.5.4 Bayesian inference

The basis for Bayesian inference is derived from Bayes' theorem. It is express by the following formula

$$Pr(A|B) = \frac{Pr(B|A)Pr(A)}{Pr(B)}$$

and, if we consider **y** as the observations and Θ the set of parameters of the model, it can be reformulated as

$$p(\Theta|\mathbf{y}) = \frac{p(\mathbf{y}|\Theta)p(\Theta)}{p(\mathbf{y})}.$$

Here $p(\Theta|\mathbf{y})$ is the joint *posterior* distribution, $p(\Theta)$ is the *prior* distribution of the parameters and $p(\mathbf{y}|\Theta)$ is the likelihood of **y** under the model.
The denominator

$$p(\mathbf{y}) = \int p(\mathbf{y}|\Theta)p(\Theta)d\Theta \tag{5.21}$$

is called "prior predictive distribution" of y. This value is the same for each class, for each value of Θ, because it does not depend on it. For this reason, it can be considered as a generic

constant **c**.
The final result obtained is a proportional relation that links the assumptions made on the prior distributions and the values of the observations

$$p(\Theta|\mathbf{y}) \propto p(\mathbf{y}|\Theta)p(\Theta).$$

To properly understand the Bayesian inference updating, it is useful to take a closer look at the definition of *prior* and *posterior* distribution.

Prior and posterior distribution

Prior distribution is used by the analyst to quantify the uncertainty about a given parameter before taking the data into account.
To define a Bayesian network, a specific distribution has to be set for each parameter of the net (Θ). To decide which is the best choice for each parameter is not a straightforward task and it can strongly influence the ability of prediction of the model.

There are different kinds of prior distributions, depending on whether or not the author has information about them.
The less informative prior distribution that can be used for a real parameter $\Theta \in \mathbb{R}$ a uniform distribution, defined from minus infinity and plus infinity:

$$\theta \sim \mathbf{U}(-\infty, \infty).$$

This prior distribution allows the posterior to be effected only by the observations. Technically, $p(\theta)$ is considered as an improper prior distribution[14], because it does not have a finite integral

$$\int p(\theta)d\theta = \infty.$$

Because of this, a good practice is to avoid to use it in this formulation. Usually, this prior distribution is used with a wide domain instead of an infinite one. All the parameters of the prior distributions, like the extremes of the domain, are called *hyper-parameters*, to distinguish them from the parameters of the model.

When information about the model is available the prior distribution can be *informative*. In this case, the author has to guide the model in the right direction, to prevent deviation given by a wrong set of data. Especially if the dataset contains just a few features, a wrong

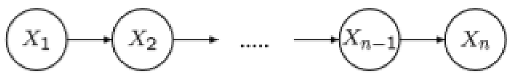


Fig. 5.3 Linear DAG

observation can produce a significant bias.
On the other hand, the usage of a too specific informative prior distribution takes away from the data the predictive power. An example where a predictive prior can be used is when the *present model* form is similar to the *previous model* form, so the present model is an updated version based on more current data. In this case, the posterior distributions used in the previous model can be used as prior distributions of the new model.
The posterior distribution instead expresses uncertainty about parameter set Θ after taking both the prior distribution and the data into account [37]. It is the final result of the *Bayesian updating*, where the evidence of the recording data are incorporated in the model.

$$p(\Theta|y) \sim p(y|\Theta)p(\Theta)$$

5.5.5 Conjugate distribution

One problem in the implementation of Bayesian approaches is analytical tractability. To predict the value of the *future* data x_{new} the two following probability has to be evaluated

$$p(x_{new}|x) = \int p(x_{new}|\Theta)p(\Theta|x)d\Theta \tag{5.22}$$

where we assume $X_{new} \perp X|\Theta$.
Furthermore the normalization of the posterior distribution involves computing the following integral

$$p(x) = \int p(x|\Theta)p(\Theta)d\Theta \tag{5.23}$$

Generally, these integrals cannot be evaluated analytically, except under special conditions on the involved distributions.
When the prior distribution $p(\theta)$ and the posterior distribution belong to the same distribution family, the prior is called *conjugate prior*[34] for the likelihood function. A conjugate prior distribution gives a closed-form expression to the posterior, and in this case, the numerical integration is not necessary.
Some of the several distributions that have this property are indicated in the diagram is 5.4.

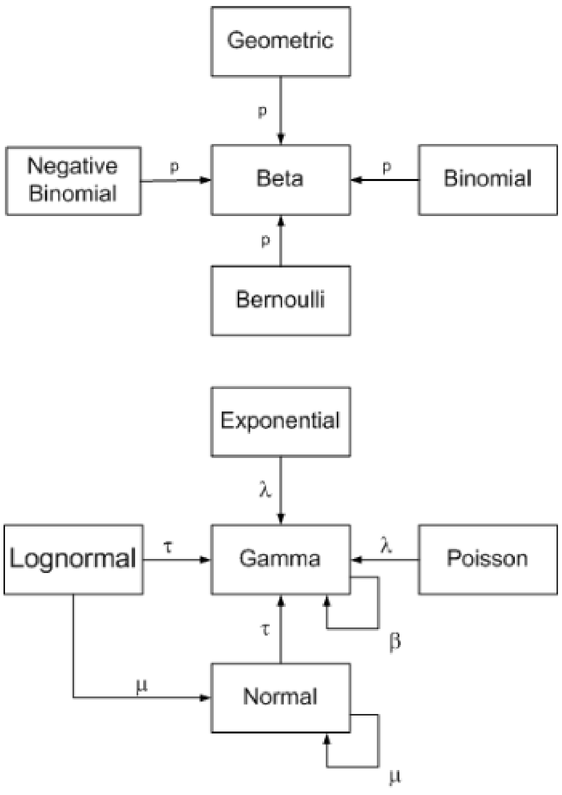


Fig. 5.4 Conjugate distribution [Cook]

5.5.6 Algorithms for Bayesian updating

When is not possible to calculate exactly the posterior distribution, approximated methods of calculation have to be introduced. The most common mathematical tools used for inference with Bayesian networks are the *Markov chains Monte Carlo* (MCMC), a class of algorithms for sampling from a probability distribution.
In Bayesian networks, the *Gibbs sampling* is widely used. It is a Markov chain Monte Carlo algorithm for obtaining a sequence of observations which are approximated from a specified multivariate probability distribution, when direct sampling is difficult. The Gibbs sampling basic version is a special case of the Metropolis-Hastings algorithm, it is used when the joint distribution is unknown but the conditional distribution of each variable is known, given the others .It is often well- adapted to sampling the posterior distribution in a Bayesian network. Given a multivariate distribution $p(x_1,...,x_n)$, suppose that we want to obtain an MCMC sample $X = (x_1,...,x_n)$. The implementation of the algorithm follows the following steps:

1 Set an initial value of $X^{(i)}$.

2 To obtain $X^{(i+1)} = (x_1^{(i+1)}, x_2^{(i+1)}, ..., x_n^{(i+1)})$, for each component $x_j^{(i+1)}$ sample according to the distribution specified by $p(x_j^{(i+1)} | x_1^{(i+1)}, ..., x_{j-1}^{(i+1)}, x_{j+1}^{(i)}, ..., x_n^{(i)})$.

3 Loop over i

In the process is common to ignore the first batch of the sample, to avoid to consider the so-called *burn-in-period.*
The relation of conditional distribution and joint distribution is the following

$$p(x_j|x_1,...,x_{j-1},x_{j+1},...,x_n) = \frac{p(x_1,...,x_n)}{p(x_1,...,x_{j-1},x_{j+1},...,x_n)} \tag{5.24}$$

The conditional distribution result is "proportional" to the joint distribution in the sense that the denominator is not a function of x_j, it forms part of the normalization constant for the distribution. In practice, to determine the nature of the conditional distribution of a factor x_j, it is easiest to factor the joint distribution according to the individual conditional distributions defined by the DAG over the variables, ignore all factors that are not functions of x_j.
In this work, the chosen distributions required the use of an approximated evaluation of the posterior distribution. The distributions have been chosen manually, matching the distributions properties end the data characteristics.
A tool to perform this kind of calculations is OpenBUGS (the open source variant of Win-BUGS) that works integrate with R.
BUGS is a software package for performing Bayesian inference Using Gibbs Sampling. The

user specifies a statistical model, of (almost) arbitrary complexity, by simply stating the relationships between related variables. The software includes an ' expert system ', which determines an appropriate MCMC (Markov chain Monte Carlo) scheme (based on the Gibbs sampler) for analyzing the specified model.[40]

5.5.7 Semantic approach

One of the big problems of the Bayesian network is the complexity. In a network with n vertices, there are up to $n(n-1)/2$ edges and each represents a correlation between two variables, then the number of features in the models has to remain limited. Working with a text, the features are the single words that appear in the document, and they can be thousands. A close selection is necessary to be able to construct an effective model, the choice can be performed with an automated tool or with the human intervention.
This field is called Bayesian network structure learning, and its aim is to learn the structure of the DAG. There are several algorithms that help to construct the network that better fit with the data and they can be included in these three approaches: *constraint-based, score-based* and *hybrid*.
Constraint-based algorithms are based on the seminal work of Pearl on maps and its application to causal graphical models. For example his Inductive Causation (IC) algorithm[33] provides a framework for learning the DAG structure of Bayesian Network using conditional independence tests. Score-based learning algorithms represent the application of heuristic optimization techniques to the problem of learning the structure of a Bayesian network. To each candidate Bayesian network is assigned a network score reflecting its goodness of fit, which the algorithm then attempts to maximize. Hybrid learning algorithms combine constraint-based and score-based algorithms to offset the respective weaknesses and produce reliable network structures in a wide variety of situations.
These algorithms do not explore all the possibilities, to reduce the computational complexity, that grows polynomially with the size of the network. For this reason, the optimal solution is not guaranteed.

In this work, we decided to do not use an automatic tool to construct the network but to perform a manual selection of the features. The decision has been driven by the dimension of the set of the features, none of the algorithms described before can be applied to construct a network with thousands of nodes.

We selected manually, with the help of SEB research team, the most meaningful words, the ones that better qualify the two main classes of the model, the Hawkish class and the Dovish class.

A part of them corresponds to the most meaningful topics faced by the Board members in every meeting. Depending on how the Board member introduces this topic they can be classified as hawkish or dovish.

5.5.8 Dynamic Bayesian Sentiment Analysis

In this work, we build a Bayesian network that tries to capture the relation between the frequency of certain words in the text and the sentiment behind it. In the model, we use the intuition that the text sentiment influences the words used in the speech by the board member. The network is trained using the past information from the previous minutes and the analysis performed by the SEB research team.

The two considered sentiments are called Hawkish and Dovish, as in the preceding chapters, and they are described by a binary variable. We decided to monitor four meaningful words that can detect the intention of the text. The words are selected after an accurate reading of the minutes and the advice of the SEB research team. The analyzed words[2] are *inflation*, *repo rate*, *krona* and *expectation* and they are considered conditional independent of each other.
The frequency of the given words is described with a Binomial distribution.

$$w_{i,j,t} \sim \text{Binomial}(q_{i,j,t}, l_{j,t})$$

In this model we consider the frequency of these words in every board member speech present in the datasets described in the Chapter 2. The parameters of the Binomial distribution are $q_{i,j,t}$ and $l_{i,j,t}$. Where $q_{i,j,t}$ is the probability that the word i appears in the part of the document referred to what j said at the time t.
$l_{j,t}$ is the length of the part of the document referred to what j said at the time t.

Each word i has is own probability distribution with its own parameters. We chose the Binomial distribution to let the length of the text influence the variable. The intuition is that a long intervention is connected with an higher frequency of the words and probably with a stronger feeling, in one or an other direction. The four monitored words are assumed to be independent even if they have a slightly negative correlation[3]. The approximation is acceptable because the number of analyzed words (4) is much lower than the total number of the words in a board member speech (~ 5000). Furthermore the conditional independence

[2]Three of them are the topics that are considered relevant in the text, by the SEB's research team.

[3] The negative correlation could have been captured using a Multinomial distribution with 5 classes, one for each considered word and one for all the other words.

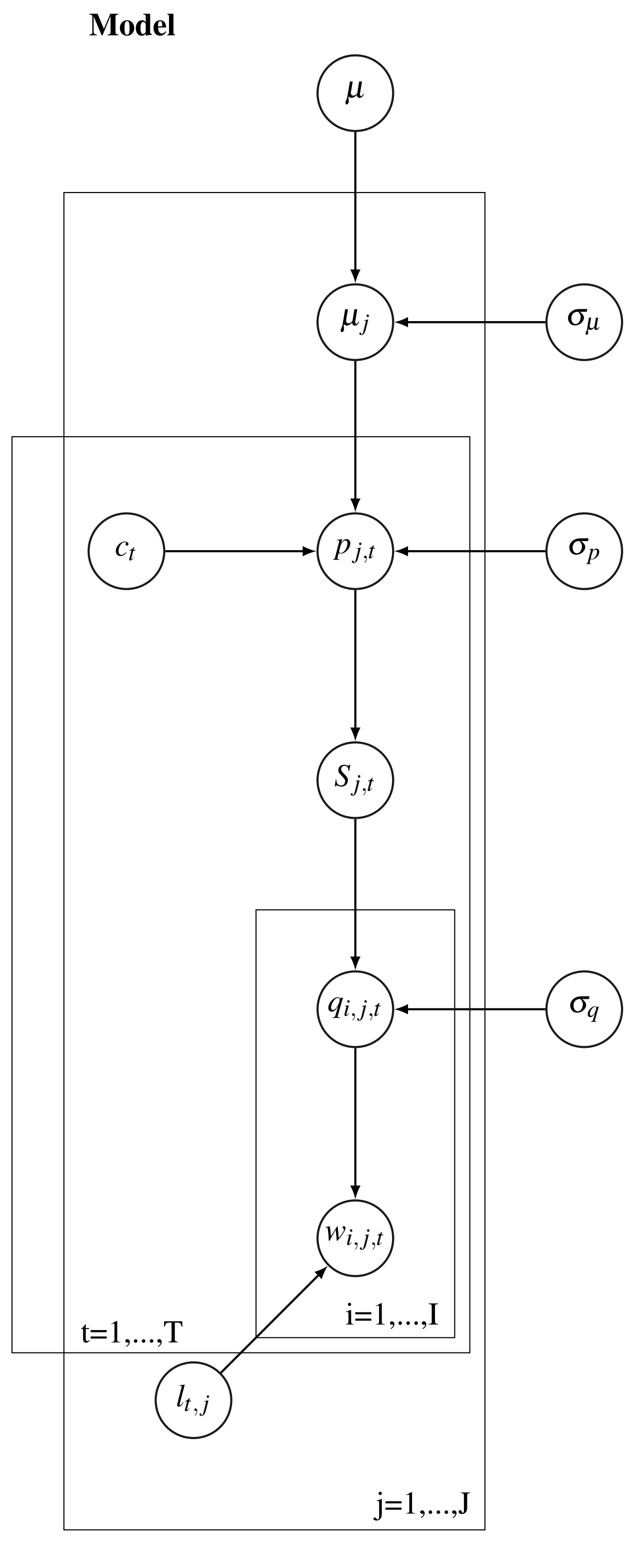

Model
μ
μ_j
σ_μ
c_t
$p_{j,t}$
σ_p
$S_{j,t}$
$q_{i,j,t}$
σ_q
$w_{i,j,t}$
i=1,...,I
t=1,...,T
$l_{t,j}$
j=1,...,J

Fig. 5.5 Model

hypothesis is realistic, since they are nouns that characterize different economic aspects, that are not related to each other.

The sentiment of the text is encoded with a Bernoulli distribution where 1 is associated with the Dovish class and 0 is associated with the Hawkish class.

$$S_{j,t} \sim \text{Bernoulli}(p_{j,t})$$

The variables $p_{j,t}$ represent the probability that the part of the document referred to what j said at the time t belongs to the Dovish class.
The complete structure of the model is shown in the Figure 5.5 and described in the following.

$$\begin{aligned}
\text{logit}(p_{j,t}) &\sim \text{Normal}(A^p + \mu_j + c_t, \sigma_p^2) \\
S_{j,t} &\sim \text{Bernoulli}(p_{j,t}) \\
\text{logit}(q_{i,t,j}) &\sim \text{Normal}(A_i^q + \delta_i(S_{t,j} = 1), \sigma_q^2) \\
w_{i,t,j} &\sim \text{Binomial}(q_{i,t,j}, l_{j,t}) \\
\forall i &= 1, ..., I \\
\forall t &= 1, ..., T \\
\forall j &= 1, ..., J
\end{aligned}$$

The variable p is between 0 and 1 is thus defined through the logit transformation[4] with a Normal distribution. A high negative value of the Normal mean brings p close to zero (high probability of the hawkish class) while a high positive mean brings p close to one (high probability of the dovish class). The mean is a sum of different variables, used to take in to account the dynamic influence of the time and the board member ideology. It is composed by

$$\begin{aligned}
A^p &\sim \text{Normal}(0, \sigma_{A^p}^2) \\
\mu_j &\sim \text{Normal}(0, \sigma_\mu^2) \\
c_t &\sim \text{Normal}(0, \sigma_c^2)
\end{aligned}$$

A_p is a ground average that defines the mean of $\text{logit}(p_{j,t})$ and does not depend on the time and on the board member.
μ_j express the economic feeling of the board member j, while c_t indicates the natural bias given by the particular economic circumstances at time t. Thanks to the presence of A_p the means of both μ_j and c_t can be zero. A negative μ_j indicates a Hawkish tendency in the board member j while a negative c_t is linked with an economic crisis at time t, where everyone

[4] $\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$

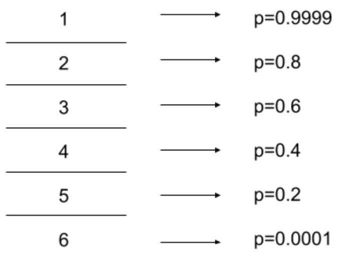


Fig. 5.6 Scheme of the conversion from the rank to p

is driven to be more dovish. All the parameters have an uninformative prior distribution because they have to be estimated using the data without any bias given by the author.

The variable $q_{i,j,t}$ is the probability that a word in the part of text referred to j at time t is the word i. Also the variable $q_{i,j,k}$ is defined between 0 and 1, therefore its probability distribution is indicated through the logit transformation. In this case, the mean is influenced by the binary variable $S_{j,t}$ distributed as a Bernoulli and by the following variables

$$A_i^q \sim \text{Normal}(0, \sigma^2_{A^q})$$
$$\delta_i \sim \text{Normal}(0, \sigma^2_{\mu})$$

The variable δ_i quantifies how much the final feeling of the text influences the use of the word i. δ_i is the differential average effect produced when the binary variable S is equal to 1. A_i^q is the ground average of the word i and does not depend on time and author of the text. Also these variables have an uninformative prior distribution.

The variable $w_{i,j,k}$ describes the frequency of the words as explained before.

Some of the variables of the model are *observable*, so they can be fed as data in the program, whereas others are the *parameters*, objects of interest that have to be estimated. The observable data are used to train the model, that is able to learn the link between the frequency of the words and the sentiment of the text. In this work the observable variables are all the $w_{i,j,t}$, all the $l_{j,t}$ and all the $p_{j,t}$ for $t \leq T-1$, while the $p_{j,T}$ have to be estimated. $w_{i,j,t}$ and $l_{j,t}$ are extracted from the Riskbank minutes, while $p_{j,t}$ come from the evaluation provided by the SEB.

The SEB research team provides a rank of the board members for every meeting, from the most dovish to the most hawkish. In this model we converted the rank in values for $p_{j,t}$, following the scheme in figure 5.6. The values of p are obtained using a linear transformation on the rank positions. The two values of $p = 1$ and $p = 0$ are approximated with $p = 0.9999$ and $p = 0.0001$ respectively, to have p belong to the logit function codomain. The first board member in the rank has a probability close to 1 to belong to the dovish class. On the contrary,

	p	**pred rank**	**rank**
Per Jansson	0.984	1	1
Stefan Ingves	0.8849	2	2
Kerstin af Jochnick	0.4651	3	4
Cecilia Skingsley	0.7063	4	3
Martin Flodén	0.1606	5	5
Henry Ohlsson	0.09362	6	6

Table 5.4 Comparison of the predicted rank with the real one

the sixth board member has a probability close to 0 to belong to the dovish class and then a probability close to 1 to belong to the hawkish class.
The dataset used as train-set of the Bayesian Network is composed by 18 minutes, then $T = 18$. The board member are 6 then $J = 6$ and the considered words are 4 the $I = 4$

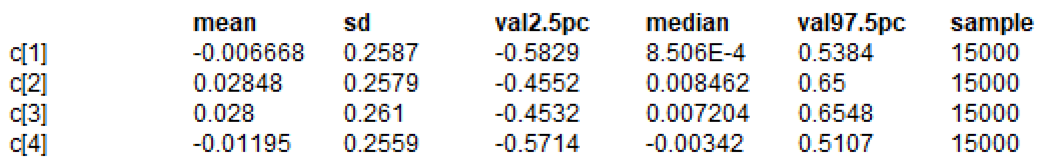
Results

The values of p_T are estimated with the software Openbugs using the code illustrated in AppendixB.1. As explained before, all the values of p_t for $t < T$ are part of the train-set and they are used to predict the values of p_T that is considered as the test-set. In our work, the train-set is composed of 17 items while the test-set is composed of one element.

The predicted p_T are used to build the same kind of rank that the SEB provides. The board members have to be sorted from the most dovish to the most dovish, then from the one with the highest p to the one with the lowest one. The results are shown in the table 5.4 Evaluate the model is not straightforward because a rank cannot be defined as wrong or right, without considering different levels of errors. In literature, several scored are proposed to evaluate the goodness of a rank[1] but they use the intuition that it is more important to correctly predict highly relevant items than marginally relevant ones. In this work, both the most Hawkish and the most Dovish Board members have to be correctly classified.
For this reason, we defined a simple score, that measure the error level

$$\text{error-level} = \frac{\sum_i |\text{pred rank}_i - \text{rank}_i|}{K} \tag{5.25}$$

where K is a normalization factor to obtain a score between $[0-1]$, here K=18. The perfect rank has an error-lever equal to 0 and the completely inverted rank has an error-level equal to 1.
For the proposed table the error-level is $0.11\overline{1}$. The rank is almost correct, except for the



	mean	sd	val2.5pc	median	val97.5pc	sample
c[1]	-0.006668	0.2587	-0.5829	8.506E-4	0.5384	15000
c[2]	0.02848	0.2579	-0.4552	0.008462	0.65	15000
c[3]	0.028	0.261	-0.4532	0.007204	0.6548	15000
c[4]	-0.01195	0.2559	-0.5714	-0.00342	0.5107	15000

Fig. 5.7 Estimated c

switch in the intermediate positions. In the classification algorithms, the errors are more common for the instances that have not a well-defined position. The board members ranked as third and fourth have probably not expressed a very strong opinion in the meeting, this makes them more difficult to classify.

To produce a more significant evaluation the error-level has to be evaluated for more than one instances. Because of the small dataset, we decided to use the Leave-one-out cross-validation (LOOCV)[32]. This technique is performed extracting from the dataset one instance at the time and using it as test-set while the remaining part is the training set. At the end of the process, the mean is evaluated. The obtained error-level is $0,2778$. Looking at the dataset we see that in the last 2 years the ranking position of the board member used to be almost constant, while in 2015 and 2016 there were more variations in the rank. The change in the rank leads to difficulties to find the right link between the words frequency and the final sentiment.

All the variables that define the means of the two Normal distributions are estimated with Openbugs and give information on the robustness of the model. The variables c_t take into account the movements of the economy that can influence the board members. A small part[5] of the values obtained in the simulation are shown in the figure 5.7. All the means of c_t are very close to zero while the standard deviation is at least one order of magnitude greater than the mean. These values of the mean and the standard deviation bring the value of c to be non statistically significant.

The reason of this result comes from the tool used to obtain the observed values of p. The SEB research team provide a rank and not a score, this means that is the position of the board members is not absolute but it depends on the overall performance of the meeting. For example, the real position of the first board member in the rank can be more or less dovish according to the years. The only thing that we know from the rank is that he has been more dovish than the others participant of the meeting.

To take in consideration the board members ideology we used the variable μ_j. The estimated values are shown in the figure 5.5, sorted following the predicted rank. As we expected, the obtained values of μ_j follow the intuition expressed before. The most dovish and the most

[5] The complete table of the result is in the Appendix B.1

Board member	**mean**	**sd**	**rank**
Per Jansson	8.747	1.41	1
Stefan Ingves	2.352	1.376	2
Kerstin af Jochnick	-0.03572	1.401	3
Cecilia Skingsley	-2.118	1.379	4
Martin Flodén	-1.335	1.377	5
Henry Ohlsson	-6.591	1.398	6

Table 5.5 μ values

Board member	**mean**	**sd**
expect	0.5319	0.1916
inflat	-1.355	0.2081
krona	0.5091	0.1861
repo rate	-1.16	0.1915

Table 5.6 δ values

hawkish board member have the highest and the lowest μ_j values respectively. Except for the Cecilia Skingsley μ_j, the others values follow the rank order indeed the error in the Cecilia Skingsley μ_j reflects also a classification error of the Bayesian network.

The estimated variables $delta_i$ represent the link between the probability that a word in the text is a specific word i and the sentiment class to which the text belongs. It can be translated in the relevance of the word i in the prediction model. If the absolute value of $delta_i$ is high than the word i is significantly influential on the final sentiment of the text. The values provided by Openbugs are in the table 5.6. The two words inflation and repo rate appear to be the most important in the model and they both drive the mean of the Normal distribution of logit$(q_{i,j,t})$ to be more negative. A negative mean of those variables brings to a lower probability $q_{i,j,t}$ and this means a lower frequency of the word i in the j's text at time t. In other words, a strong presence of the words inflation and repo rate indicates the tendency of a Hawkish text.

The variables $delta_i$ are relevant in the model because the order of magnitude of them absolute values is the same of the absolute values of A_i^q. The estimated value of all the A_i^q is are approximately -7. A.q[1] -7.617 0.1355 -7.886 -7.615 -7.353 15000 A.q[2] -7.039 0.1295 -7.297 -7.039 -6.789 15000 A.q[3] -7.005 0.1279 -7.252 -7.005 -6.756 15000 A.q[4] -6.557 0.1208 -6.785 -6.557 -6.325 15000

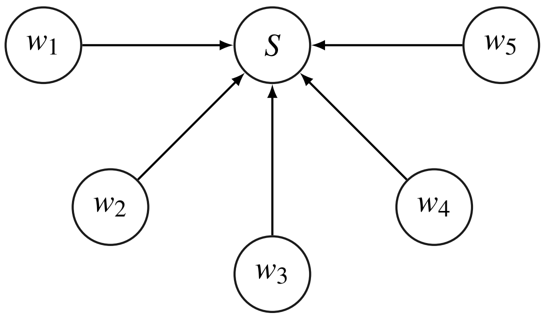


Fig. 5.8 Graphical model with three nodes and conditional independence between nodes X and Z given node Y.

5.5.9 Naïve Bayes

Naive Bayes classifier is a special case of Bayesian network, build on the so-called *naive Bayes assumption*: all the attributes of the examples are independent of each other, given the context of the class. This hypothesis is unrealistic in most of the cases but nevertheless, the model performs surprisingly well.
The DAG that describes a generic Naive Bayesian classifier is shown in the figure 5.8. In this context, the shape of the graph is the same but the number of nodes is in the order of magnitude of the thousand.

Different type of assumption can be done about the distribution involved in the net, and in this way, it is possible to build a different type of classifiers. The most common are the

- Gaussian Naive Bayes
- Multinomial Naive Bayes
- Bernoulli Naive Bayes.

In the *Gaussian* Naive Bayes algorithm for classification, the likelihood of the features is assumed to be Gaussian:

$$f(X = x_i|C = c) = \frac{1}{\sqrt{2\pi\sigma_c^2}} \exp\left(-\frac{(x_i - \mu_c)^2}{2\sigma_c^2}\right) \tag{5.26}$$

The Naive Bayes algorithm for *multinomially* distributed data is one of the two classic naive Bayes variants used in text classification. The distribution is parametrized by vectors $\theta_c = (\theta_{c,1}, \ldots, \theta_{c,n})$ for each class c, where n is the number of features and $\theta_{c,i}$ is the probability $P(x_i|c)$ of feature i appearing in a sample belonging to class c.

The naive Bayes *Bernoulli* classification uses data that are distributed according to multivariate Bernoulli distributions; i.e., there may be multiple features but each one is assumed to be a binary-valued (Bernoulli, boolean) variable.
The decision rule for Bernoulli naive Bayes is based on

$$P(x_i|c) = P(i|c)x_i + (1 - P(i|c))(1 - c_i) \tag{5.27}$$

which differs from multinomial Naive Bayes' rule in that it explicitly penalizes the non-occurrence of a feature i that is an indicator for class c, where the multinomial variant would simply ignore a non-occurring feature.
[27]

It can be shown that the number of misclassification is minimized, on average, "assigning to each observation the most likely class given its predictor values".[18] In other words, the predicted class is the one that maximized the following probability:

$$\max_{c \in C} P(C = c|X = x_0) \tag{5.28}$$

Where C are the possible class outcomes and $X = (x_1, ..., x_n)$ is a vector that represents the n features of the model. Using the Bayes theorem and the Naïve assumption is possible to write:

$$P(C|X) = \frac{P(C)P(X|C)}{P(X)} = \frac{P(C)P(x_1, x_2, ..., x_n|C)}{P(x_1, x_2, ..., x_n)}. \tag{5.29}$$

$$P(x_1, x_2, ..., x_n|C) = \prod_{i=1}^{n} P(x_i|C). \tag{5.30}$$

and the probability becomes:

$$P(C|x_1, x_2, ..., x_n) = \frac{P(C)\prod_{i=1}^{n} P(x_i|C)}{P(x_1, x_2, ..., x_n)}. \tag{5.31}$$

since the denominator does not depend on the predicted class C, then it can be ignored. In the end, the function to maximize to get the prediction of the class is:

$$c = argmax_{j=1,..K} P(C_j) \prod_{i=1}^{n} P(x_i|C_j). \tag{5.32}$$

Our Approach

To apply the Naive Bayes classifier, the same procedure applied for the SVM classifier and the logistic regression has been followed, a text cleaning and a feature extraction are

performed and then the three models are tested.
The algorithms are applied at a document level, then the speech of each board member is considered in full. This choice is given by the labels of the training-set, that are provided for the entire text and not at a sentence level. The Bag-of-words model is used to encode the text, as it is explained in section3.3, and the two weight schema are tested

- Frequency-based: is used only the frequency of a term in a document and the features are the first *k* most important words;
- Tf.idf: the collection of document is used to dump the frequency of the terms, see section 3.3.1;

The Python library *sklearn* allows to test easily the three more common types of Naive Bayes classifiers cited before.[6]
The Gaussian Naive Bayes classifier it is not meant for the text analyze but it has been tried for the sake of completeness. In the algorithm the parameters σ_c and μ_c are estimated using maximum likelihood.
The two more used classifiers in text mining are the Multivariate Bernoulli model and the Multinomial model.
In the Multinomial model, the numbers of occurrences of the words are stored in the vector that defines the document. In the Python library the parameters $\theta_c = (\theta_{c,1}, \dots, \theta_{c,n})$ [7] are estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:

$$\hat{\theta}_{c,i} = \frac{N_{c,i} + \alpha}{N_c + \alpha n} \tag{5.33}$$

where $N_{c,i} = \sum_{x \in T} x_i$ is the number of times feature i appears in a sample of class c in the training set T, and $N_c = \sum_{i=1}^{|T|} N_{c,i}$is the total count of all features for class c.
The smoothing priors $\alpha \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting $\alpha = 1$ is called Laplace smoothing, while $\alpha < 1$ is called Lidstone smoothing.
In the Bernoulli model, a document is represented by a binary vector that in each component indicates the presence or the absence of each word. In this model, the number of times a word occurs is not considered.

[6] $http://scikit-learn.org/stable/modules/naive_bayes.html$

[7] where n is the number of features and $\theta_{c,i}$ is the probability $P(x_i|c)$ of feature i appearing in a sample belonging to class c.

Naive Bayes	Gaussian BOW	Gausian Tf-idf
Accuracy	0.735	0.82167
F1	0.69321	0.81071

Table 5.7 Naive Bayes Gaussian

Naive Bayes	Multinomial BOW	Multinomial Tf-idf
Accuracy	0.88333	0.0.84667
F1	0.86476	0.82762

Table 5.8 Naive Bayes Multinomial

Results

The results of the Naive Bayes algorithm are in general at the level of the state-of-art. As expected, the Multinomial and the Bernoulli models outperform the Gaussian model, because it is not meant for text classification.
Following the literature, the Multinomial model usually outperforms the Bernoulli model with large vocabulary size [26]. In this work, the results show the opposite situation. The Binomial model with the Tf-idf weights reach the accuracy of 93% against the 88% of the Multinomial model with the frequency weights.
As explained before, all the result scores are obtained using the cross-validation technique. A dataset of about one hundred occurrences is considered too small to be split in training-set and test-set without the risk of having a strong bias.

5.5.10 Conclusion

The Naive Bayes algorithm outperforms the Dynamic Bayesian Network models. The main reason is the larger number of predictive features used in the Naive Bayes algorithm. The simplicity of the DAG structure of the Naive Bayes and the involved distributions reduce the computational complexity of the updating process. Thanks to these characteristics a feature

Naive Bayes	Binomial BOW	Binomial Tf-idf
Accuracy	0.86333	0.93833
F1	0.85714	0.93619

Table 5.9 Naive Bayes Binomial

selection is not necessary and all the word of the collection can be included in the analysis[8]. The limited number of nodes in the Bayesian model penalizes its predictive power.
The Naive Bayes model uses more than two thousand features but the Bayesian model uses just four words. Increasing the complexity of the Bayesian network the model could improve its quality, more meaningful words can be added, including adjectives and verbs. The operation can be guided by an analysis of the δ_i, to prevent the usage of useless words.

[8]We tried to use a feature selection to lighten the algorithm but the performance level has decreased without a real gain in terms of efficiency.

References

[1] Ackerman, B. and Chen, Y. (2011). Evaluating rank accuracy based on incomplete pairwise preferences.

[2] Allahyari, M., Pouriyeh, S., Asse, M., Safaei, S., Trippe, E. D., Gutierrez, J. B., and Kochut, K. (2017). Text summarization techniques: A brief survey. *In Proceedings of arXiv*, (1):9 pages.

[3] Berry, M. W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review*.

[Cook] Cook, J. D. Conjugate prior relationships.

[5] CVE-2008-1368 (2008). Monetary policy. [online] https://www.investopedia.com/terms/m/monetarypolicy.asp.

[6] Duta and Hart (1973). Pattern classification and scene analysis.

[7] Edwards, W., Lindman, H., Savage, and J., L. (1963). Bayesian statistical inference for psychological researc.

[8] Finetti, B. D. (2017). Theory of probability: A critical introductory treatment.

[9] for Guides in Metrology), J. C. (2008). Sentiment analysis techniques in recent works.

[10] Friedman, N., Geiger, D., and Goldszmidt, M. (1997). Bayesian network classifiers.

[11] Gong, Y. and Liu, X. (2001). Generic text summarization using relevance measure and latent semantic analysis. pages 19–25.

[12] Guan, Z. L. X. D. Y. and Yang, J. (2013). Reserved self-training: A semisupervised sentiment classification method for chinese microblogs.

[13] Gurusamy, V. and Kannan, S. (2014). Preprocessing techniques for text mining.

[14] H, J. (1961). Theory of probability.

[15] Harris and Zellig (1954). Distributional structure. 10:146–162.

[16] Hasselqvist, J., Helmertz, N., and Kågebäck, M. (2017). Query-based abstractive summarization using neural networks. *CoRR*, abs/1712.06100.

[17] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An introduction to statistical learning gareth james daniela witten trevor hastie robert tibshirani with applications in r.

[18] James, G., Witten, D., Hastie, T., and Tibshirani, R. (2014). An introduction to statistical learning—with applications in r. 82.

[19] Jaynes and E.T. (1986). Bayesian methods: General background.

[20] Jurafsky, D. and Martin, J. H. (2000). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.

[21] Kahveci, E. and Odabaş, A. (2016). Central banks' communication strategy and content analysis of monetary policy statements: The case of fed, ecb and cbrt. *Procedia - Social and Behavioral Sciences*, 235:618 – 629. 12th International Strategic Management Conference, ISMC 2016, 28-30 October 2016, Antalya, Turkey.

[22] Langley, P., Iba, W., and Thompson, K. (1992). An analysis of bayesian classifiers.

[23] Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries.

[24] Madhoushi, Z., Hamdan, A. R., and Zainudin, S. (2015). International vocabulary of metrology — basic and general concepts and associated terms (vim).

[25] Mani, I. (1999). *Advances in Automatic Text Summarization*. MIT Press, Cambridge, MA, USA.

[26] McCallum, A. and Nigam, K. (1998). A comparison of event models for naive bayes text classification.

[27] Metsis, V., Androutsopoulos, I., and Paliouras, G. (2006). Spam filtering with naive bayes – which naive bayes?

[28] Mohri, M., Rostamizadeh, A., and Talwalkar, A. (2012). Foundations of machine learning.

[29] Monegato, G. (1998). Fondamenti di calcolo numerico.

[30] Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. (1999-66). Previous number = SIDL-WP-1999-0120.

[31] Pang, B., Lee, L., and Vaithyanathan, S. (2002). Thumbs up? sentiment classification using machine learning techniques. *CoRR*, cs.CL/0205070.

[32] Payam Refailzadeh, Lei Tang, H. L. (2008). Cross validation.

[33] Pearl, J. and Verma, T. (1990). Equivalence and synthesis of causal models.

[34] Raiffa, H., Schlaifer, R., and Gelman, A. (1961). R2openbugs: A package for running openbugs from r.

[35] Russell, S. J. and Norvig, P. (2010). Artificial intelligence: A modern approach.

[36] Sarkar, D. (2016). *Text Analytics with Python: A Practical Real-World Approach to Gaining Actionable Insights from Your Data*. Springer Science+Business Media New York, Bangalore, Karnataka India, 1st edition.

[37] Statisticat, L. (2015). Bayesian inference.

[38] Steinberger, J. (2004). Using latent semantic analysis in text summarization and summary evaluation. *Proceedings of ISIM'04*, pages 93–100.

[39] Sturtz, S. and and, U. L. (2005). Applied statistical decision theory.

[40] Thomas, N. (2009). Frontpage openbugs.

[41] Torres-Moreno, J. (2014). *Automatic Text Summarization*.

[42] Yang, C. C., Chen, H., and Hong, K. (2003). Visualization of large category map for internet browsing. *Decision Support Systems*, 35(1):89 – 102. Web Retrieval and Mining.

[43] Zhao, J., Liu, K., and Xu, L. (2016). Sentiment analysis: Mining opinions, sentiments, and emotions bing liu (university of illinois at chicago) cambridge university press, 2015 isbn 9781107017894. 42:1–4.

Appendix A

Singular Value Decomposition[29]

Theorem 1. *Given $A \in \mathbb{R}^{mxn}$ exist two orthogonal matrix $U = \{u_1, \dots u_m\} \in \mathbb{R}^{mxm}$ and $V = \{v_1, \dots v_n\} \in \mathbb{R}^{nxn}$, such that:*

$$U^T AV = S, \qquad A = USV^T$$

where $S \in \mathbb{R}^{mxn}$ is diagonal, so that:

$$S_{i,j} = \begin{cases} 0, & i \neq j \\ \sigma_i, & i = j \end{cases}$$

with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_p \geq 0$, $p = min\{m,n\}$.

There are many different algorithms to compute U and V, they are no unique. The matrix S has the singular value of A, that are the square root of the eigenvalues of $A^T A$. The columns of U and V are respectively the left singular vector and the right singular vector, so that $\forall i = 1, \dots p$:

$$Av_i = \sigma_i u_i$$

$$Au_i = \sigma_i v_i. \quad \text{(A.1)}$$

The geometrical meaning of the singular values of A is related with the hyper-ellipsoid and represents

$$E = \{y : y = Ax, \|x\|_2 = 1\}$$

To better understand the importance of SVD as a low rank approximation, or mapping in a lower dimensional space of the original space, the following theorem is stated:

Theorem 2. *If $\exists r \in \mathbb{N}$, such that:*

$$\sigma_1 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} > \cdots > \sigma_p 0, \qquad p < r$$

then:

1. *$rk(A) = r$;*
2. *$\{u_1, \ldots, u_r\}$ is a base for $R(A)$;*
3. *$\{v_{r+1}, \ldots, v_n\}$ is a base for $N(A)$;*
4. $$A = \sum_{i=1}^{r} \sigma_i u_i v_i^T = U_r S_r V_r^T;$$
5. *$\|A\|_2 = \sigma_1$*

where:

$$U_r = (u_1, \ldots, u_r), \quad V_r = (v_i, \ldots, v_r);$$

and $N(A)$ is the core of the transformation and $R(A)$ the image space.

The next theorem it is useful to understand the relation between the space associated with the original matrix and the space of the low-rank approximation.

Theorem 3. *Given a Singular Value Decomposition of $A \in \mathbb{R}^{mxn}$, with $rk(A) = r$. If, it is fixed $k < r$, it is defined:*

$$A_k = \sum_{i=1}^{k} \sigma_i u_i v_i^T$$

and

$$\beta = \{B \in \mathbb{R}^{mxn} : rk(B) = k\}$$

then the following results are true:

$$min_{B \in \beta} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}$$

The last theorem gives a measure for the distance in 2-norm between the original matrix A and β the set of matrix with rank k. Furthermore, A_k is the best approximation for A with rank k.

Appendix B

Openbugs

B.1 Openbugs code

Here is provided the code used in R/Openbugs to build both the model based on the Bayesian network.
The R packages *R2OpenBUGS* and *coda* allow to run Openbugs from R. R2OpenBUGS automatically writes the data and scripts in a format readable by OpenBUGS for processing in batch mode. After the OpenBUGS process has finished, it is possible either to read the resulting data into R by the package itself, which gives a compact graphical summary of inference and convergence diagnostics, or to use the facilities of the coda package for further analyses of the output. [39]

The code is the following

```
model <- function() {
  for (j in 1:J){
    for (t in 1:Time){
      for(i in 1:I){
        mu.logq[j,i,t] <- A.q[i] + s[j,t]*delta[i]
         logit.q[j,i,t] ~ dnorm(mu.logq[j,i,t],tau.q)
        q[j,i,t]<- exp(logit.q[j,i,t])/(1+exp(logit.q[j,i,t]))
        w[j,i,t] ~ dbin(q[j,i,t],l[j,t])
      }
      mu.logp[j,t] <- mu.j[j]+c[t]+A.p
       logit.p[j,t]  ~ dnorm(mu.logp[j,t],tau.p)
      p[j,t]<- exp(logit.p[j,t])/(1+exp(logit.p[j,t]))
      s[j,t] ~ dbern(p[j,t])
    }
    mu.j[j] ~ dnorm(mu,tau.mu)

  A.p~ dnorm(0,sigma.A)
  mu   ~  dgamma(0.001,0.001)
  tau.mu ~ dgamma(0.001,0.001)
  tau.q  ~ dgamma(0.001,0.001)
  tau.p  ~ dgamma(0.001,0.001)
  tau.c  ~ dgamma(0.001,0.001)
  tau.v  ~ dgamma(0.001,0.001)
  tau.d  ~ dgamma(0.001,0.001)
  tau.A  ~ dgamma(0.001,0.001)
  for (t in 1:Time){
    c[t] ~ dnorm(0,tau.c)
  }
  for (i in 1:I){
    delta[i] ~ dnorm(0,tau.d)
    A.q[i] ~ dnorm(0,tau.v)
  }
}

model.file <- file.path(tempdir(),"model.txt")
write.model(model, model.file)

# Data

rank <- structure(.Data=c(4.5,5.9999,4.5,1.0001,2,3,
                                                         ...
                                    4,5.9999,3,5,1.0001,2,
                                    NA,NA,NA,NA,NA,NA), .Dim=c(6,18))

p=1/5*(-1*rank+6)
logit.p=log(p/(1-p))
l<- structure(.Data=c(7251,8153,6153,5554,7304,10567,
                                                ...
```

```
                                5744,6948,5335,9384,9774,3261),  .Dim=c(6,18))
w <- structure(.Data=c(9,12,0,0,
                                 ...
                                16,25,2,7),.Dim=c(6,4,18))
J <- 6    #Board Member
I <- 4        #Word
Time <- 18    #Time

data <- list("I","J","Time","l","w","logit.p")
params <- c("c","mu.j","p","delta")
inits <- function() { list(mu=0.001,tau.mu=0.001,tau.p=0.001,tau.v=0.00
                                                 ...
                                    delta=c(0,0,0,0),A.q=c(0,0,0,0),A.p=0)}

out <- bugs(data, inits, params, model.file, n.iter=10000, debug=TRUE)

p.new<-unlist(out$mean["p"], use.names=FALSE)
p.new
```

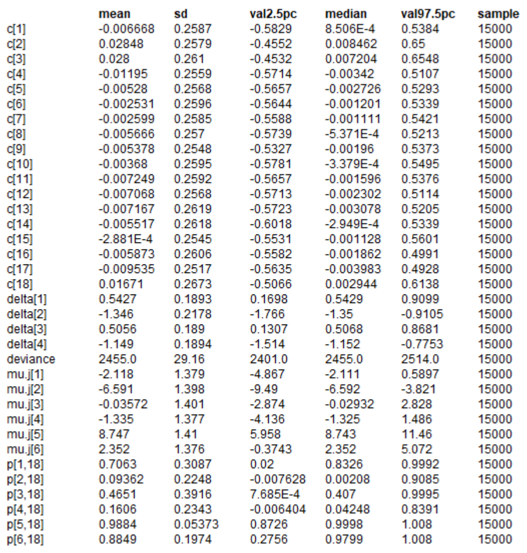
At the beginning the model is build using the Openbugs language. The program requires the precision instead of the variance then all the σ^2 are substituted with τ. The uninformative priors required by the model are obtained with the Gamma distribution

$$\tau \sim \text{Gamma}(\alpha,\beta) \tag{B.1}$$

with $\alpha = 0.001$ and $\beta = 0.001$. In this way the mean of the distribution is one and the variance is 1000. The values of the observable variables are included in the model with the list *data* and the objects of inference are set with the variable *params*.

B.2 OpenBugs Output

All the OpenBugs outputs are shown here



	mean	sd	val2.5pc	median	val97.5pc	sample
c[1]	-0.006668	0.2587	-0.5829	8.506E-4	0.5384	15000
c[2]	0.02848	0.2579	-0.4552	0.008462	0.65	15000
c[3]	0.028	0.261	-0.4532	0.007204	0.6548	15000
c[4]	-0.01195	0.2559	-0.5714	-0.00342	0.5107	15000
c[5]	-0.00528	0.2568	-0.5657	-0.002726	0.5293	15000
c[6]	-0.002531	0.2596	-0.5644	-0.001201	0.5339	15000
c[7]	-0.002599	0.2585	-0.5588	-0.001111	0.5421	15000
c[8]	-0.005666	0.257	-0.5739	-5.371E-4	0.5213	15000
c[9]	-0.005378	0.2548	-0.5327	-0.00196	0.5373	15000
c[10]	-0.00368	0.2595	-0.5781	-3.379E-4	0.5495	15000
c[11]	-0.007249	0.2592	-0.5657	-0.001596	0.5376	15000
c[12]	-0.007068	0.2568	-0.5713	-0.002302	0.5114	15000
c[13]	-0.007167	0.2619	-0.5723	-0.003078	0.5205	15000
c[14]	-0.005517	0.2618	-0.6018	-2.949E-4	0.5339	15000
c[15]	-2.881E-4	0.2545	-0.5531	-0.001128	0.5601	15000
c[16]	-0.005873	0.2606	-0.5582	-0.001862	0.4991	15000
c[17]	-0.009535	0.2517	-0.5635	-0.003983	0.4928	15000
c[18]	0.01671	0.2673	-0.5066	0.002944	0.6138	15000
delta[1]	0.5427	0.1893	0.1698	0.5429	0.9099	15000
delta[2]	-1.346	0.2178	-1.766	-1.35	-0.9105	15000
delta[3]	0.5056	0.189	0.1307	0.5068	0.8681	15000
delta[4]	-1.149	0.1894	-1.514	-1.152	-0.7753	15000
deviance	2455.0	29.16	2401.0	2455.0	2514.0	15000
mu.j[1]	-2.118	1.379	-4.867	-2.111	0.5897	15000
mu.j[2]	-6.591	1.398	-9.49	-6.592	-3.821	15000
mu.j[3]	-0.03572	1.401	-2.874	-0.02932	2.828	15000
mu.j[4]	-1.335	1.377	-4.136	-1.325	1.486	15000
mu.j[5]	8.747	1.41	5.958	8.743	11.46	15000
mu.j[6]	2.352	1.376	-0.3743	2.352	5.072	15000
p[1,18]	0.7063	0.3087	0.02	0.8326	0.9992	15000
p[2,18]	0.09362	0.2248	-0.007628	0.00208	0.9085	15000
p[3,18]	0.4651	0.3916	7.685E-4	0.407	0.9995	15000
p[4,18]	0.1606	0.2343	-0.006404	0.04248	0.8391	15000
p[5,18]	0.9884	0.05373	0.8726	0.9998	1.008	15000
p[6,18]	0.8849	0.1974	0.2756	0.9799	1.008	15000

Fig. B.1 Estimated parameters

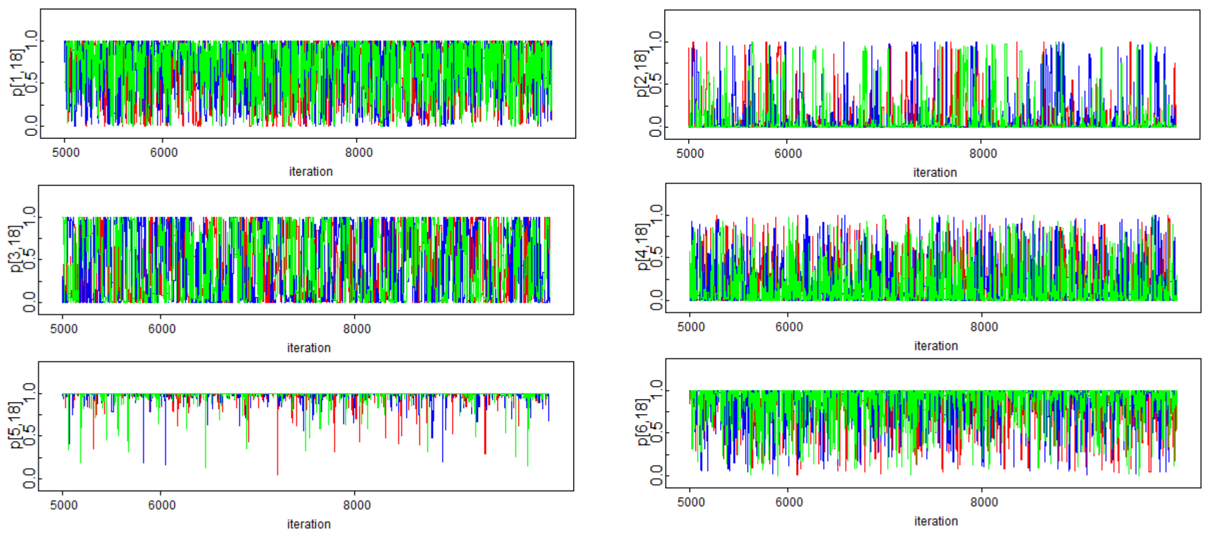


Fig. B.2 Simulation of multiple chains of $p_{j,t}$