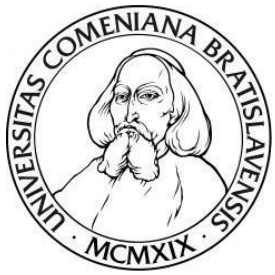
COMENIUS UNIVERSITY IN BRATISLAVA
FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS



Prediction of future events for companies by analysing articles of financial newspapers

MASTER THESIS

2018 Ján Siviček

COMENIUS UNIVERSITY IN BRATISLAVA

FACULTY OF MATHEMATICS, PHYSICS AND INFORMATICS
DEPARTMENT OF APPLIED MATHEMATICS AND STATISTICS

Prediction of future events for companies by analysing articles of financial newspapers

MASTER THESIS

Study programme:	Mathematical Economics, Finance and Modelling
Field of study:	1114 Applied Mathematics
Supervisor:	Dr. Sander Koemans

Bratislava 2018 **Ján Siviček**

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
KATEDRA APLIKOVANEJ MATEMATIKY A ŠTATISTIKY

Predikcia budúcich udalostí v spoločnostiach analýzou článkov finančných denníkov

DIPLOMOVÁ PRÁCA

Študijný program:	Ekonomicko-finančná matematika a modelovanie
Študijný odbor:	1114 Aplikovaná matematika
Vedúci práce:	Dr. Sander Koemans

Bratislava 2018 **Ján Siviček**

28712077

Comenius University in Bratislava
Faculty of Mathematics, Physics and Informatics

THESIS ASSIGNMENT

Name and Surname:	Bc. Ján Siviček
Study programme:	Mathematical Economics, Finance and Modelling (Single degree study, master II. deg., full time form)
Field of Study:	Applied Mathematics
Type of Thesis:	Diploma Thesis
Language of Thesis:	English
Secondary language:	Slovak

Title: Prediction of future events for companies by analysing articles of financial newspapers

Annotation: There are many financial newspapers (for example Het Financieele Dagblad) which write about financial situation and performance of companies, world economies; in general - about everything important what is happening in the financial world. What still remains unclear is whether are these newspapers able to indicate/foresee exceptional performance of companies? Based on what newspapers are writing about a certain company today, can one predict what will most probably happen to the company in the future? The main aim of this master thesis is, by analyzing the text of articles in the financial newspapers using text mining, to create a predictive model to predict presence of a certain event (such as scandal, fraud, financial problems, bankruptcy) for the companies in the future based on what financial newspapers write in the presence or wrote in the past. The predictive model, for instance regression model, will be created based on the sample of companies where the certain event has occured and the variables of the model shall be determined by analysing text of articles of the financial newspapers related to the companies from the sample.

Supervisor:	Dr. Sander Koemans
Department:	FMFI.KAMŠ - Department of Applied Mathematics and Statistics
Head of department:	prof. RNDr. Daniel Ševčovič, DrSc.
Assigned:	27.01.2017

Approved: 27.01.2017 prof. RNDr. Daniel Ševčovič, DrSc.
Guarantor of Study Programme

.....
Student

.....
Supervisor

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Ján Siviček
Študijný program: ekonomicko-finančná matematika a modelovanie (Jednoodborové štúdium, magisterský II. st., denná forma)
Študijný odbor: aplikovaná matematika
Typ záverečnej práce: diplomová
Jazyk záverečnej práce: anglický
Sekundárny jazyk: slovenský

Názov: Prediction of future events for companies by analysing articles of financial newspapers
Predikcia budúcich udalostí v spoločnostiach analýzou článkov finančných denníkov

Anotácia: Existuje mnoho finančných denníkov (napríklad Het Financieele Dagblad), ktoré sa zaoberajú finančnou situáciou a výkonom spoločností, svetových ekonomík; vo všeobecnosti - všetkým dôležitým vo finančnom svete. Avšak, stále ostáva nejasné, či sú tieto denníky schopné predpovedať výnimočné výkony spoločností? Na základe toho, čo denníky píšu o určitej spoločnosti dnes, je možné predpovedať čo sa s najväčšou pravdepodobnosťou stane so spoločnosťou v budúcnosti? Hlavným cieľom tejto práce je, analyzovaním textu článkov vo finančných denníkoch použitím metód text miningu, vytvoriť predikčný model na predikciu prítomnosti určitej udalosti (napríklad škandál, podvod, finančné problémy, bankrot) v spoločnosti v budúcnosti na základe toho čo finančné denníky píšu v súčasnosti alebo napísali v minulosti. Predikčný model, napríklad regresný model, bude zostrojený na základe vzorky spoločností, v ktorých sa určitá udalosť udiala a premenné modelu by mali byť určené analýzou textu článkov finančných denníkov súvisiacich so spoločnosťami zo vzorky.

Vedúci: Dr. Sander Koemans
Katedra: FMFI.KAMŠ - Katedra aplikovanej matematiky a štatistiky
Vedúci katedry: prof. RNDr. Daniel Ševčovič, DrSc.

Dátum zadania: 27.01.2017

Dátum schválenia: 27.01.2017 prof. RNDr. Daniel Ševčovič, DrSc.
garant študijného programu

.....
študent

.....
vedúci práce

Acknowledgement I would like to express my gratitude and appreciation to my supervisors Dr. Sander Koemans from Deloitte Nederland and Dr. Rikkert Hindriks from Vrije Universiteit Amsterdam for their professional guidance and valuable advices. Moreover, I would like to thank my colleagues from the Forensic team of Deloitte Nederland for creating a nice working environment. Last but not least, my big thanks go to my parents, without whom nothing of this would have been possible, for their strong support during my studies.

Poďakovanie Chcel by som vyjadriť svoju vďačnosť a uznanie mojim školiteľom Dr. Sanderovi Koemansovi z Deloitte Nederland a Dr. Rikkertovi Hindriksovi z Vrije Universiteit Amsterdam za ich profesionálne vedenie a cenné rady. Taktiež by som chcel poďakovať kolegom z forenzného teamu Deloitte Nederland za vytvorenie príjemného pracovného prostredia. V neposlednom rade moje veľké poďakovanie patrí mojim rodičom, bez ktorých by nič z tohto nebolo možné, za silnú podporu počas môjho štúdia.

Abstract

Siviček, Ján: Prediction of future events for companies by analysing articles of financial newspapers [Master Thesis], Comenius University in Bratislava, Faculty of Mathematics, Physics and Informatics, Department of Applied Mathematics and Statistics; Supervisor: Dr. Sander Koemans, Bratislava, 2018, 101 p.

In this thesis a new methodology for prediction of events at companies based on textual information was built. The methodology was built on the event of bankruptcy and articles of Dutch financial newspaper Het Financieele Dagblad were used as data source. The concept of so-called TF-IDF matrix was applied to textual data in order to represent it numerically and adjusted version of this matrix was used as a basis of predictive models. Logistic regression and several feedforward neural networks were created and their performance was approximated using 8-fold cross-validation. However, the final outcome of this thesis is not a concrete model but a model framework. This is due to TF-IDF matrix, which has to be recreated every time a new article is added to dataset. Reader will understand how data was gathered and processed in order to create TF-IDF matrix, how TF-IDF matrix was used in predictive models and how so-called 'bankruptcy score' was determined for a company based on the predictions of its articles. In addition, there were several ideas described on how the methodology can be further improved.

Keywords: text mining, TF-IDF matrix, logistic regression, neural networks, bankruptcy score

Abstrakt v štátnom jazyku

Siviček, Ján: Predikcia budúcich udalostí v spoločnostiach analýzou článkov finančných denníkov [Diplomová práca], Univerzita Komenského v Bratislave, Fakulta matematiky, fyziky a informatiky, Katedra aplikovanej matematiky a štatistiky; školiteľ: Dr. Sander Koemans, Bratislava, 2018, 101 s.

V tejto práci je vytvorená nová metodológia na predpovedanie udalostí v spoločnostiach na základe informácií z textových dát. Metodológia bola vybudovaná na udalosti bankrotu a články holandského finančného denníka Het Financieele Dagblad boli využité ako zdroj dát. Koncept takzvanej matice TF-IDF bol aplikovaný na textové dáta za účelom ich numerickej reprezentácie a upravená verzia tejto matice bola použitá ako základ predikčných modelov. Logistická regresia a niekoľko neurónových sietí boli vytvorené a ich úspešnosť bola odhadnutá pomocou 8-fold krížovej validácie. Avšak, konečným výsledkom tejto práce nie je konkrétny model, ale modelový rámec. Dôvodom je matica TF-IDF, ktorá musí byť nanovo vytvorená vždy, keď je nový článok pridaný do databázy. Čitateľ porozumie ako boli dáta zozbierané a spracované za účelom vytvorenia matice TF-IDF, ako bola matica TF-IDF použitá v predikčných modeloch a ako bolo takzvané skóre bankrotu určené pre spoločnosti na základe predpovedí pre články spoločnosti. Navyše, v práci je popísaných niekoľko spôsobov, akými môže byť vyvinutá metodológia zdokonalená.

Kľúčové slová: text mining, matica TF-IDF, logistická regresia, neurónové siete, skóre bankrotu

Contents

Introduction

The Financial Times, The Wall Street Journal, The Economist and in The Netherlands Het Financieele Dagblad - do they have anything in common? Certainly! These and many others are financial newspapers which write about everything important happening in the economies and companies from the financial point of view and have gained a solid reputation over the years amongst the masses of readers due to their reliability.

Surely, there are reasons why so many readers are reading financial newspapers. For instance, to be updated on what is happening in the financial world in general, to get all the available information to accurately estimate the changes in the stock prices and many other possible reasons.

The aim of this project is to go a little bit further and investigate whether the information published in the financial newspapers can help us to indicate or foresee the occurrence of certain events at the companies.

Certain events mentioned above are events chosen in accordance with the interests and specialization of the Forensic Team of Deloitte Nederland and are described in more detail in the Chapter 3.

The main question to be answered in this thesis stands as follows: 'Is it possible to predict certain events for certain companies based on what the financial newspapers have been writing in the past?'

In this thesis we will focus on the event of bankruptcy and build the methodology on that. Theoretical and practical core of this thesis will be formed by the combination of text mining and predictive modelling.

Techniques of text mining shall be applied on the database of financial articles to process and format textual data so that concept of TF-IDF matrix, which assigns weights of importance and specificity for each word appearing in text corpus, can be applied in order to numerically represent textual data.

Adjusted version of TF-IDF matrix is then used as basis of predictive models, specifically logistic regression model and neural network.

The prediction performance of models will be approximated using 8-fold cross-validation, models will be compared and best models will be chosen for futher analysis. By further analysis we mean predicting the event of bankruptcy for the company

based on predictions for all articles about that particular company. Two ways will be described of how so-called 'bankruptcy score' for company can be determined - using simple average and weighted average of predictions.

It is important to understand that no concrete model can be final outcome of this thesis since TF-IDF matrix has to be recreated with every new article added to the database. Nevertheless, reader shall understand how such concrete model can be constructed as well as how its performance may be measured.

In the last chapter we will describe several ideas on how the methodology built in this thesis can be improved in order to create more powerful model since it certainly makes sense to use textual data for event predictions as it hides lots of useful information in itself.

This thesis also shows the high potential of textual data for event prediction. On the other hand, lots of potential coming from text was unused and therefore by improving the methodology and using as much information as available, the results could be astonishing.

1 Background

In this chapter we will shortly introduce the company at which the thesis project is carried out as well as describe the data used in this project.

1.1 Company overview

This master thesis is, in fact, a report from an internship project carried out at the company Deloitte Nederland in Amsterdam, The Netherlands from March 2018 until August 2018.

"Deloitte" is the brand under which tens of thousands of dedicated professionals in independent firms throughout the world collaborate to provide audit & assurance, consulting, financial advisory, risk advisory, tax and related services to select clients.

These firms are members of Deloitte Touche Tohmatsu Limited, a UK private company limited by guarantee ("DTTL" hereinafter). Each DTTL member firm provides services in particular geographic areas and is subject to the laws and professional regulations of the particular country or countries in which it operates. Each DTTL member firm is structured in accordance with national laws, regulations, customary practice, and other factors, and may secure the provision of professional services in its territory through subsidiaries, affiliates, and other related entities.

As of April 2018, Deloitte has more than 263 900 professionals at member firms delivering services in audit & assurance, tax, consulting, financial advisory, risk advisory, and related services in more than 150 countries and territories. Revenues for fiscal year 2017 were US$38.8 billion [8].

Within the geographical area of The Netherlands, Deloitte Nederland is the DTTL member firm. With over 5 500 employees and 14 offices throughout the Netherlands, Deloitte is one of the largest providers of professional services in the country [9].

The internship project was carried out at the Department of Financial Advisory Services, within the Forensic Team - specifically in the sub team of Financial Crime Analytics. The Forensic Team's main focus is on fraud detection, investigation, analysis and prevention.

1.2 Data description

The data source for this project are articles of Het Financieele Dagblad - a daily Dutch newspaper writing about finance matters on the Dutch but also international market.

Het Financieele Dagblad officially exists since 1943 and is one of the main sources for the Dutch public to get a high quality financial news. Unsurprisingly, a majority of articles is written in Dutch language.

The articles were downloaded from the LexisNexis database. Company names were used as keywords in LexisNexis search engine. Due to the restrictions on download from LexisNexis, articles were downloaded in bulks of 200 articles.

The dataset used for this project contains articles dated from 1994 to nowadays. Initially, data are in txt format and in bulks of 200 articles. It is possible to retrieve content of each article as well as its headline and publication date.

1.3 Initial processing of the data

Since it is preferred to work with single articles, initial processing of bulks of 200 articles is necessary. This processing is performed using Python and we split bulks into separate articles, removed all unnecessary features of LexisNexis standardized output and stored only content, headline and publication date of each article in a separate txt file.

2 Theory overview

This chapter deals with the theoretical and literature overview essential for the topic of this thesis. We will go through the theory behind text mining and predictive modelling, specifically logistic regression and neural networks.

2.1 Text mining

Text mining, or Text Analytics, can be broadly defined as a knowledge-intensive process in which a user interacts with a document collection over time by using a suite of analysis tools [12]. It is a rather new discipline within the more general field of Data Mining and has gained an enormous popularity and grown very fast over the last years. Structured data, for instance numeric data, are known for being easily categorized as well as their pattern is usually not too difficult to recognize. Techniques have been developed to analyze this type of data and retrieve information. However, nowadays, when it often holds that the one who knows more, wins, it seems essential to go beyond structured data.

There is a lot to investigate and retrieve information from in order to make a better assumption or prediction than the competitor. Unstructured data hides an enormous amount of information in itself and at the same time goes with the challenging and difficult task how to retrieve this information. As an example, data falling into the category of unstructured data are texts in different languages, webpages and other types of multimedia, such as images or videos [22]. These data are known for being hard to process and analyze as they usually do not show any clear evidence of structure, pattern or category.

Estimates were made on how much data available worldwide is unstructured. Every estimate is slightly different from the other but what they all agree on is that a large majority of all data is unstructured - somewhere between 80 to 90 % [22]. For such portion of the data and possible information hidden inside it makes sense and is of an interest to develop techniques able to retrieve this information. We could call it a lost opportunity to let this data go away unanalyzed as there are real examples that textual data might have a certain impact on what is happening afterwards. Two examples for

all can be how financial articles affect the development of stock prices or how news articles affect the result of political elections.

Therefore, the techniques of text mining were developed and are consistently being improved in order to analyze textual data and extract the most information. Text is written according to complicated rules forming a language, more precisely numerous different languages, which humans can read, but computers can not. Thus, we can consider text mining being part of machine learning as its aim is to develop algorithms allowing computers to efficiently process and retrieve information from large amounts of textual data which humans would not have a capacity to process.

The process of creating a suitable algorithm for text analysis involves several steps which we will outline below.

2.1.1 Text mining software

There is loads of software and programming languages used for text mining. For the purpose of this project, the analysis was performed using Python and R and text mining packages available for them. In particular, we used R and its `tm package` which has many text mining methods already programmed and supports several languages, including the Dutch language [11]. Python was used during the project as well, but only for the initial data processing. The text mining package for Python is called Natural Language Toolkit ('NLTK') [24].

2.1.2 Data collection and import

The most essential task for text mining to even begin is, of course, data collection. As the data suitable for the analysis we can consider set of documents containing textual information. This data is usually unstructured, such as pdf documents or plain text documents. However, we also distinguish so-called semi-structured data. Semi-structured data is structured data that is not organised in a table but rather containing some semantic tags or metadata which make it easier to process this data comparing to the unstructured data. Some well-known examples of semi-structured data are e-mails or XML files.

After we have collected our data, we load it into a large set of documents called

text corpus. Text mining techniques and operations are usually performed on the text corpus as a whole.

2.1.3 Data pre-processing

Data pre-processing is a process of cleaning our data or transforming them into a suitable form for performing the further analysis. Textual data comes with a lot of unnecessary elements or characters when aiming for a qualitative analysis of the text.

Usually, we are not interested in having addresses of web pages - URLs, some characters (examples: @, ”, ;) in our corpus.

Moreover, each language contains a lot of so-called stopwords, which are words appearing very often but contributing very little or not at all to the meaning of the text, such as prepositions. We are often interested in removing all the stopwords and potentially some particular words of our interest as well.

When a word is a starting word of the sentence, its first letter is capital. In our text analysis, this would create two different words from one word just because of its position in the sentence. Thus, we should also remove all capital letters from our text and change them to lower case. We are often not interested in having numbers and punctuation in our analysed text so they can be removed, too.

After all these pre-processing steps, we managed to clean and reduce our text but probably also created a lot of additional and unnecessary whitespace in our text, such as triple space instead of usual single space between words. Fortunately, this can be easily fixed.

Lastly, as part of preprocessing, stemming should be performed. Stemming is a process of deleting all the prefixes or afixes the words have and keeping only the word base in order to group together the words which have the same meaning but are not written in the same way because of prefixes or afixes. For instance, after stemming, words ’going’, ’gone’, and ’go’ would be all changed to the word ’go’.

All these modifications were performed on the text corpus as a whole and using R and its commands which are already programmed within the text mining package [11, 24].

2.1.4 Data analysis

Once we have our text data pre-processed we can further analyse it in order to retrieve some prior unknown information from this data.

2.1.4.1 Tokenization

Tokenization is a process of splitting the text into semantic units, which are called tokens. The most common tokenizations are sentence tokenization, where each sentence is a token, and word tokenization, where each word forms a token [24]. Tokenization is a crucial step in the process of text analytics.

2.1.4.2 Document Term Matrix

The Document Term Matrix is a large matrix which has indices of all documents in the corpus as its rows and all unique words from the corpus as its columns. The elements of the Document Term Matrix are numbers stating how many times a certain word appears in a certain document. In order to construct the Document Term Matrix, we need to first subtract word tokens from the text.

Through the Document Term Matrix we can carry out some statistical analysis on our text data such as looking at word frequencies or term correlations. Term correlation indicates how strong certain terms are correlated, thus how often they appear together.

Document Term Matrix is known for being of a very large size due to the number of unique words in the corpus and the large number of documents in the corpus. Moreover, another characteristic of this matrix is the sparseness - i.e. having 'a lot of' zero elements. By selecting suitable criteria to remove infrequently used terms from the Document Term Matrix (for instance by removing all terms which appeared in less than 5 documents) we can significantly reduce size of this matrix [2].

2.1.4.3 Word cloud and letter cloud

A word cloud or letter cloud can be considered as a graphic interpretation of the Document Term Matrix. Word cloud, most usually in the shape of cloud, or

letter cloud, shaped as a particular letter, describes the Document Term Matrix by depicting the words with their size being proportional to their frequencies within the text corpus.

Looking at the words depicted in a word cloud or letter cloud can give us very quickly an idea about the text corpus as the bigger the word is, the more it appears in the text corpus.

On the other hand, it is difficult to make any scientific conclusions from word cloud or letter cloud.

2.1.4.4 TF-IDF

TF-IDF stands for Term frequency-inverse document frequency and is a numerical statistic used in text mining, which allows us to represent the text corpus numerically. Document Term Matrix is the basis of TF-IDF but there is a significant difference between these two matrices.

Document Term Matrix counts how many times has a certain word appeared in a certain document. However, there are some words which appear generally more often within a document as well as in multiple documents but these words do not contribute significantly to the meaning of a single document. However, TF-IDF considers and measures how specific and important a certain word is for a particular document [2]. The formula, which is used for the calculation of elements of TF-IDF matrix, rewards words which appear more often in the document but at the same time penalises words which appear in multiple documents. Mathematically we can express this as follows:

Term frequency $tf_{i,j}$ counts the number of occurrences $n_{i,j}$ of a term t_i in a document d_j. In the case of normalization, the term frequency is divided by the number of words within a document. Inverse document frequency for a term t_i is defined as:

$$idf_i = log_2\Big(\frac{D}{d_{t_i}}\Big) \tag{1}$$

where D denotes the total number of documents and where d_{t_i} is the number of documents in which the term t_i appears. Term frequency - inverse document

frequency is defined as the multiplication

$$tf_{i,j} \cdot idf_i \tag{2}$$

and is conveniently programmed within R inside the `weightTfIdf` function [26].

This matrix is a high-dimensional matrix with each row belonging to a single article and each column representing a single word from the text corpus. Elements of this matrix are weights assigned to each word and each document, which describe how important and specific this particular word is for this particular document.

For the purpose of this master thesis we shall consider the TF-IDF matrix as the output of the text analysis and use this matrix and its row vectors as inputs for the creation of predictive models.

2.1.4.5 Sentiment Analysis

Sentiment analysis, or so-called opinion mining, is a part of text mining aimed at extracting the emotion from the text and assigning a number by which we can identify the type of emotion. +1 usually indicates highly positive sentiment, 0 is neutral sentiment and -1 is a very negative sentiment.

Using R the sentiment of Dutch text can be analysed with the command `analyzeSentiment` which is a part of library `SentimentAnalysis` [25].

This command uses a special dictionary which classifies list of words into two groups: positive and negative. The dictionary used is a psychological Harvard-IV dictionary developed by Harvard University for general purposes. The R command counts how many positive and negative words our article contains and determines the sentiment of the article by the following formula:

$$\text{sentiment} = \frac{\text{number of positive words - number of negative words}}{\text{number of all words}} \in [-1, 1] \tag{3}$$

2.1.4.6 Other text mining techniques

Text mining procedure can involve more tasks and more complex theoretical concepts but those will not be described in this chapter since they were not used for this project as they fall beyond the scope of this project.

2.2 Predictive modelling

2.2.1 Logistic regression

Logistic regression is a special case of a broader class of predictive models - generalized linear models ('GLM' hereinafter). The logistic regression model is defined as follow:

$$n_i Y_i \sim Bin(n_i, p_i) \tag{4}$$

$$\eta_i = {x_i}^T \beta \tag{5}$$

$$\eta_i = g(p_i) = ln(\frac{p_i}{1 - p_i}) \tag{6}$$

for $i = 1, ..., n$. Variable η_i represents predictors, which are composed of the matrix X of explanatory variables having n rows and $(k + 1)$ columns. Explanatory variable denoted by x_{ik} represents value of predictor k in observation i. Unknown vector of coefficients $\beta = (\beta_0, \beta_1, ..., \beta_k)$ belongs to each explanatory variable and coefficient β_k indicates us the effect of the predictor x_{ik} on the response variable Y_i for all $i = 1, ..., n$. Response variables $Y_1, ..., Y_n$ form random component represented by equation (4). Systematic component is represented by predictors in equation (5) and equation (6) consists of so-called 'link function' g, which is a connection between random and systematic component [15].

Logistic regression is used in cases when the response variables Y_i are categorical. In case of univariate logistic regression the response variable is binary, thus having only values 0 or 1 ('true' or 'false' alternatively).

Outcome of the logistic regression prediction procedure is the estimate of coefficients β_i for $i = 0, ..., k$. After estimating the coefficients β by $\hat{\beta}$ and by combining equations (5) and (6) we can calculate the estimate $\hat{p_i}$ of p_i:

$$\hat{p_i} = \frac{e^{\hat{\beta}^T x_i}}{1 + e^{\hat{\beta}^T x_i}} \tag{7}$$

Unlike in the case of linear regression, estimate for p_i can take only values from 0 to 1 due to the shape of logistic function, which may remind us of the letter 'S'. Therefore, we may understand the value p_i as the conditional probability of the response variable being 1 given the predictors x_i:

$$p_i = \mathbb{P}(Y_i = 1|x_i) \tag{8}$$

2.2.1.1 Maximum Likelihood Estimation

Estimates $\hat{\beta}_i$ of β_i are calculated using Maximum Likelihood Estimation by calculating the parameters β in which the likelihood function reaches its maximum. Likelihood function is defined as [7]:

$$L(\beta|y) = \prod_{i=1}^{n} \frac{n_i!}{y_i!(n_i - y_i)!} \cdot p_i^{y_i} \cdot (1 - p_i)^{n_i - y_i} \tag{9}$$

By using the relationship in equation (7), removing factorials from the likelihood function because they are constant with respect to β, and taking a logarithm of the likelihood function, we can express the log likelihood function in the following way [7]:

$$l(\beta) = \sum_{i=1}^{n} y_i (\sum_{k=0}^{K} x_{ik}\beta_k) - n_i \cdot log(1 + e^{\sum_{k=0}^{K} x_{ik}\beta_k}) \tag{10}$$

In order to find the maximum of log likelihood function, we set partial derivatives with respect to β_k equal to 0 and solve for each β_k:

$$0 = \frac{\partial l(\beta)}{\partial \beta_k} = \sum_{i=1}^{N} y_i x_{ik} - n_i p_i x_{ik} \tag{11}$$

This leaves us with a system of $(K + 1)$ non-linear equations which we can solve with some software iteratively, for instance using the Newton method [7].

2.2.1.2 Overfitting

To avoid the possibility of our logistic regression model being overfitted, which means that the model has too many predictors and therefore does not predict well on unseen data, we followed so-called 'one in ten' rule, the rule of thumb, which suggests to have ten times more observations than predictors [3].

2.2.1.3 Implementation in R

In R software the logistic regression is implemented as a part of the command `glm` by specifying the parameter `family = "binomial"`.

We can control the length and extensiveness of the iterative model training process by specifying the maximum number of iterations and value ϵ - the convergence threshold.

The iterative method is said to be converged when the difference in log likelihood functions of two iterations is less than the set convergence threshold.

2.2.2 Neural networks

Machine learning models are another class of predictive models. Neural networks, also commonly known as artificial neural networks, are an example of a machine learning model and were used in this project. This section will outline main theoretical concepts behind the neural networks to the extent which is needed to understand their application in this project.

The origin of neural networks comes from the brain and neurobiological processes happening inside the brain.

The neural networks methodology enables us to design useful nonlinear systems accepting large numbers of inputs, with the design based solely on instances of input-output relationships (e.g., pairs $\{(x_i, t_i)\}$ of feature vector x and pattern category t) [13].

2.2.2.1 Artificial neuron

Artificial neuron is the main basic unit of artificial neural networks and shares some characteristics with a human neuron, which it originates from.

Artificial neuron takes multiple inputs and sums them using specific weights. Artificial neuron has a single output which is determined by plugging the weighted sum of inputs into the so-called activation function.

Mathematically, we can model the single output y of the artificial neuron as follow [13] :

$$y = f(\sum_{i=0}^{d} w_i \cdot x_i - \tau) \tag{12}$$

Multiple inputs are represented by the values of x_i, their corresponding weights are w_i, τ represents the firing threshold and f is the activation function.

2.2.2.2 Activation function

There are several activation functions which can be used [13]. Sign activation function is defined as:

$$f(z) = sgn(z - \tau) = \begin{cases} 1 & \text{if } z \geq \tau \\ -1 & \text{if } z < \tau \end{cases} \tag{13}$$

Another type of the activation function, unit-step function, is defined as:

$$f(z) = U(z - \tau) = \begin{cases} 1 & \text{if } z \geq \tau \\ 0 & \text{if } z < \tau \end{cases} \tag{14}$$

These two types of the activation function also correspond with the characteristic 'all or none' of the output of the human neuron. Thus, neuron is either activated or not activated.

However, there exist activation functions which create graded outputs instead of binary outputs. For instance, the logistic activation function, specified by its parameter α, produces output from the interval $(0, 1)$:

$$f(z) = \frac{1}{1 + e^{-\alpha z}} \tag{15}$$

Hyperbolic tangent activation function, also having some parameter α, creates output from the interval $(-1, 1)$:

$$tanh(\alpha z) = \frac{e^{\alpha z} - e^{-\alpha z}}{e^{\alpha z} + e^{-\alpha z}} \tag{16}$$

2.2.2.3 Feedforward neural network

Architecture, which characterizes the feedforward neural network, is the simplest type of neural networks. It is acyclic network. Thus, the information from the neuron goes only forward through the network and does not ever return back to the same neuron due to no cycles being present.

2.2.2.4 Multi-layer perceptron

Multi-layer perceptron is a special type of feedforward neural networks. Multi-layer perceptron is formed of multiple layers with each layer consisting of multiple neurons. The simplest multi-layer perceptron is formed by a layer of input neurons and a layer of output neurons.

However, it is more interesting to construct a neural network with some layers in between the input and output layer. There can be multiple layers in between and with various number of neurons in each of them. Since it is the feedforward neural network, the information stored inside the particular neuron is based only on information stored in the neurons of one previous layer of the network.

The layers between the input and output layer are called hidden layers. The reason for this name is that it is not possible to observe what exact information is stored inside the neurons of hidden layers during the network training process [13].

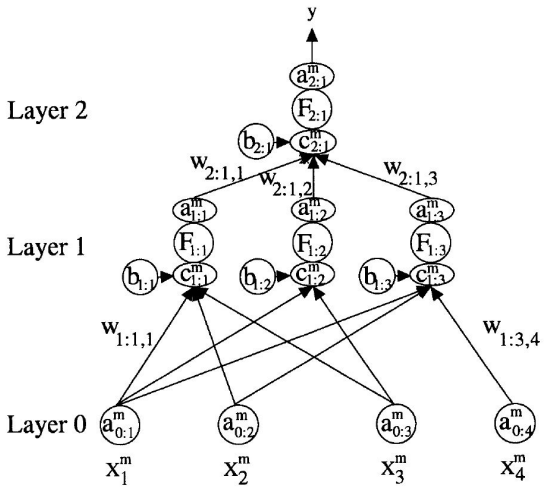


Figure 1: Example of the multi-layer perceptron [13]

2.2.2.5 Training algorithm

The algorithm uses supervised learning, which means that it learns based on the given outputs. Firstly, the algorithm assignes weights to the neurons randomly, from normal distribution. After the predictions have been made and compared with the given outputs, the weights are changed accordingly in order to minimize the error function.

Resilient backpropagation algorithm is used to find the minimum of the error function by modifiying the weights assigned to neurons and comparing the gradients of the error function until the convergence threshold is achieved [14].

Multiple other algorithms may be used as well.

2.2.2.6 Implementation in R

In R software the feedforward neural network can be trained using the package `neuralnet` [14].

We can control the length and extensiveness of the model training process by specifying the maximum number of steps performed in one repetition, maximum number of repetitions and value ϵ - the convergence threshold for the value of partial derivatives of error function.

2.2.2.7 Other types of neural networks

There are many more machine learning models as well as several other types of neural networks, for instance recurrent neural networks which are more complex because they are not acyclic. However, no detailed description of other types of neural networks or machine learning models will be made in this chapter because they were not used for this project.

2.2.3 Cross-validation

The predictive models are constructed and trained using all the data from the dataset and later used for testing on some new, previously unseen, data. However, after the construction of the model, it is desired to have some performance measures of this model so that in the future, when testing it on new data, we have some expectation regarding

the accuracy of predictions. This is where cross-validation, used to approximate the performance of predictive models, comes in handy and in the next subsections two types of cross-validation procedure will be outlined.

2.2.3.1 Leave-one-out cross-validation

In case of performing leave-one-out cross-validation, the model is trained using $(n-1)$ observations and the remaining observation forms the testing set as it looks like a new, previously unseen, observation for the model. This is repeated n times by changing the testing observation so that each observation is tested exactly once and used for training remaining $(n-1)$ times [3].

Approximation of the model performance is obtained by evaluating the accuracy of predictions from leave-one-out cross-validation.

2.2.3.2 K-fold cross-validation

This type of cross-validation differs from leave-one-out cross-validation by dividing data into k groups (folds) and using observations of one fold in the test set and observations from remaining $(k-1)$ folds in the training set. Thus, $\frac{n}{k}$ observations are tested and remaining $(n-\frac{n}{k})$ are used to train the model [3].

We can choose observations in the test set such that all possible combinations are included. However, in case of large datasets, this type of k-fold cross-validation may become computationally unfeasible as it requires many repetitions.

The test set may be also composed in a way that each observation is tested exactly once. For large datasets this is a very feasible approach which requires k repetitions.

Approximation of the model performance is again obtained by averaging the prediction accuracies of k runs of the model.

2.2.4 Confusion matrix

The event, which we will try to predict, is binary which means that it has either happened (having value of 1) or not happened (value 0). The model, which predicts such events, can be called binary classifier and its accuracy can be evaluated at some

decision threshold value τ with four possible outcomes:

- true positive ('TP') - in reality, the event has happened (value 1) and its prediction was greater or equal than τ

- true negative ('TN') - the event has not happened (value 0) and its prediction was lower than τ

- false positive ('FP') - the event has not happened (value 1) but its prediction was greater or equal than τ

- false negative ('FN') - the event has happened (value 1) but its prediction was lower than τ

These four outcomes together may be placed in a matrix of two rows and two columns, which is commonly known as confusion matrix [10].

		Prediction	
		1	0
True	1	TP	FN
	0	FP	TN

Table 1: Confusion matrix

2.2.4.1 Receiver operating characteristic curve

Receiver operating characteristic curve (hereinafter 'ROC curve') is a plot of true positive rate, plotted on the Y axis, against the false positive rate, plotted on the X axis, at different decision threshold values τ. Model, which is not any better than just random guessing, would have ROC curve represented by linear funcion $y = x$ and the best possible model is the model with ROC curve passing through the point $(0, 1)$ [10].

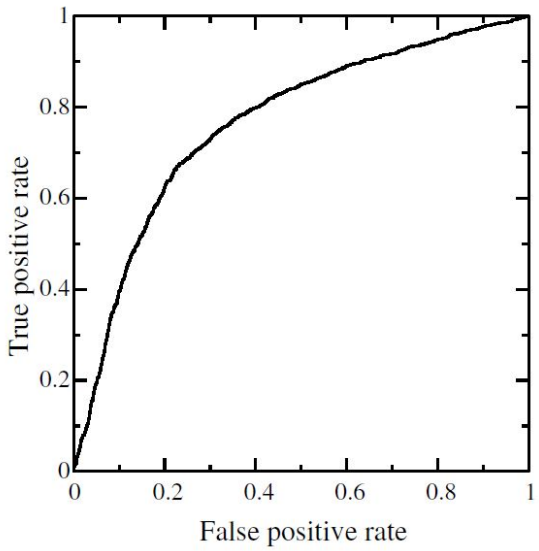


Figure 2: Example of the ROC curve [10]

2.2.4.2 Performance measures

In this section several performance measures, which originate from confusion matrix or ROC curve, will be described as they are used later for evaluation of model performance [10].

Area under the ROC curve (hereinafter 'AUC') is the number between 0 and 1 equal to the area under the ROC curve. AUC of 0.5 describes the model equivalent to random guessing and AUC of 1 describes the best possible model which has true positive rate of 1 and false positive of 0. In general, the higher the AUC value is, the better the model is.

Sensitivity, or true positive rate, is determined as percentage of correctly classified positives, mathematically:

$$\text{Sensitivity} = \frac{\text{true positive rate}}{\text{true positive rate} + \text{false negative rate}} \tag{17}$$

Specificity, or true negative rate, is determined as percentage of correctly classified negatives, mathematically:

$$\text{Specificity} = \frac{\text{true negative rate}}{\text{true negative rate} + \text{false positive rate}} \tag{18}$$

Accuracy (hereinafter 'ACC'), is determined as percentage of correctly classified, mathematically:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FN} + \mathrm{FP}} \tag{19}$$

Sensitivity, specificity and accuracy depend on chosen value of τ.

3 Events

In this chapter we will cover some events and their basic characteristics. These events can be of an interest for an individual or an entity to analyse using the approach and methodology described in this thesis. However, it falls beyond the scope of this thesis to analyse all the events and thus the purpose of this chapter is to give ideas or inspiration for further investigation in the field.

3.1 Bankruptcy

Bankruptcy is a legal proceeding involving a person or business that is unable to repay outstanding debts. The bankruptcy process begins with a petition filed by the debtor, which is most common, or on behalf of creditors, which is less common. All of the debtor's assets are measured and evaluated, and the assets may be used to repay a portion of outstanding debt.

Bankruptcy offers an individual or business a chance to start fresh by forgiving debts that simply cannot be paid, while offering creditors a chance to obtain some measure of repayment based on the individual's or business' assets available for liquidation. In theory, the ability to file for bankruptcy can benefit an overall economy by giving persons and businesses a second chance to gain access to consumer credit and by providing creditors with a measure of debt repayment. Upon the successful completion of bankruptcy proceedings, the debtor is relieved of the debt obligations incurred prior to filing for bankruptcy [16].

3.2 Fraud

Fraud is an intentionally deceptive action designed to provide the perpetrator with an unlawful gain, or to deny a right to a victim. Fraud can occur in finance, real estate, investment and insurance. It can be found in the sale of real property, such as land, personal property, such as art and collectibles, as well as intangible property, such as stocks and bonds. Types of fraud include tax fraud, credit card fraud, wire fraud, securities fraud, and bankruptcy fraud. A fraudulent activity can be carried out by one individual, multiple individuals or a business firm as a

whole.

Fraud involves the false representation of facts, whether by intentionally withholding important information or providing false statements to another party for the specific purpose of gaining something that may not have been provided without the deception.

Often, the perpetrator of fraud is aware of information that the intended victim is not, allowing the perpetrator to deceive the victim. At heart, the individual or company committing fraud is taking advantage of information asymmetry; specifically, that the resource cost of reviewing and verifying that information can be significant enough as to create a disincentive to fully invest in fraud prevention. Fraud can have a devastating impact on a business [19].

3.3 Takeover

A takeover occurs when an acquiring company makes a bid in an effort to assume control of a target company, often by purchasing a majority stake. If the takeover goes through, the acquiring company becomes responsible for all of the target company's operations, holdings and debt. When the target is a publicly traded company, the acquiring company makes an offer for all of the target's outstanding shares.

A welcome takeover, such as an acquisition or merger, generally goes smoothly because both companies consider it a positive situation. In contrast, an unwelcome or hostile takeover can be quite aggressive as one party is not participating voluntarily [20].

3.4 Financial problems

Financial distress is a condition where a company cannot meet, or has difficulty paying off, its financial obligations to its creditors, typically due to high fixed costs, illiquid assets or revenues sensitive to economic downturns. A company under financial distress can incur costs related to the situation, such as more expensive financing, opportunity costs of projects and less productive employees. Employees of a distressed firm usually have lower morale and higher stress caused by the

increased chance of bankruptcy, which would force them out of their jobs [18].

3.5 Corruption

Corruption is dishonest behavior by those in positions of power, such as managers or government officials. Corruption can include giving or accepting bribes or inappropriate gifts, double dealing, under-the-table transactions, manipulating elections, diverting funds, laundering money and defrauding investors.

There are many situations in which a person can be considered corrupt. In the financial services industry, chartered financial analysts and other financial professionals are required to adhere to a code of ethics and avoid situations that could create a conflict of interest. Penalties for being found guilty of corruption include fines, imprisonment and a damaged reputation. Engaging in corrupt behavior may have negative long-lasting effects for an organization [17].

3.6 Tax evasion

Tax evasion is an illegal practice where a person, organization or corporation intentionally avoids paying his true tax liability. Those caught evading taxes are generally subject to criminal charges and substantial penalties. To willfully fail to pay taxes is a federal offense under the Internal Revenue Service (IRS) tax code. Tax evasion applies to both the illegal nonpayment as well as the illegal underpayment of taxes [21].

4 Bankruptcy

In this chapter we will focus solely on the event of bankruptcy and develop the methodology for a prediction of this event at companies based on the information retrieved from the articles of financial newspapers.

4.1 Sample of articles

In order to create a predictive model, which can indicate the event of bankruptcy for the company based on the text of financial articles, it should be investigated whether there is a difference between the articles about companies which went bankrupt sometime after these articles ('bankrupted companies' hereinafter) and about companies which did not ('healthy companies' hereinafter) and what this difference is. Naturally, the sample of articles should be a collection of all available articles about selected bankrupted companies and selected healthy companies. The selection process for both, bankrupted and healthy companies, needs to be defined.

4.1.1 Bankrupted companies

We are interested in having all articles about bankrupted companies, which were published before these companies went bankrupt, in the sample of articles. Using the knowledge of the market we looked for bankrupted companies in publicly available sources, such as articles, figures and statistics. The aim was to look at companies which possibly could have been mentioned by Het Financieele Dagblad. Since it is a national financial newspaper, we assume that not every single small bankruptcy would be mentioned. In total, 40 bankrupted companies were selected and are listed in Table 2. After the selection of bankrupted companies, we collected the articles from the LexisNexis database. By this procedure we gathered 11 251 articles about the bankrupted companies.

No.	Company name	Bankruptcy date	No.	Company name	Bankruptcy date
1	DSB Bank	19-10-2009	21	Oilily	08-04-2009
2	Schoenenreus	24-01-2013	22	Impact Retail	31-01-2011
3	Imtech	19-08-2015	23	Hans Textiel	25-05-2011
4	Estro	05-07-2014	24	Zalco	13-12-2011
5	Macintosh	30-12-2015	25	Henk ten Hoor	05-10-2012
6	Ardenberg	13-07-2015	26	Selexyz	27-03-2012
7	Spyker	18-12-2014	27	Harense Smid	01-07-2013
8	Halfords	07-10-2014	28	Ruwaard van Putten	24-06-2013
9	Siebel	21-01-2014	29	Oad	25-09-2013
10	Polare	24-02-2014	30	Free Record Shop	24-05-2013
11	Mexx	04-12-2014	31	BAS Group	12-11-2015
12	Thermphos	21-11-2012	32	Nederlandse Munt	05-11-2015
13	Kroymans	31-03-2009	33	Renz	13-10-2015
14	Fortis Bank	02-10-2009	34	A-film	14-09-2015
15	iCentre	17-06-2013	35	Solland Solar	25-08-2015
16	Sabon	09-07-2012	36	Kuyichi	10-12-2015
17	Licom	19-10-2012	37	V&D	31-12-2015
18	Etam	21-04-2015	38	La Place	31-12-2015
19	Dico	22-12-2004	39	KPNQwest	31-05-2002
20	Meavita	09-03-2009	40	Van der Moolen	10-09-2009

Table 2: Bankrupted companies [4, 5, 23]

4.1.2 Healthy companies

For the purpose of choosing the healthy companies as a part of the sample we also benefited from the knowledge of the market. In order to have a balanced sample in terms of companies, we chose 40 companies as well. Healthy companies from our sample are listed in the Table 3. We gathered more articles about these companies since we did not have any time restriction which we had for bankrupted companies due to the date of bankruptcy. In total, we downloaded 29 972 articles about healthy companies

from the LexisNexis database.

No.	Company name	No.	Company name
1	Shell	21	Delta Lloyd
2	KPMG	22	Elseview Weekblad
3	ING	23	TNT express
4	Philips	24	Becel
5	Rabobank	25	KLM
6	Heineken	26	ASR Nederland
7	Unilever	27	Ahold Delhaize
8	Randstad	28	HunterDouglas
9	KPN	29	Vopak
10	ABN AMRO	30	Douwe Egberts
11	Aegon	31	Calve
12	ASML	32	Aalberts Industries
13	AkzoNobel	33	Refresco
14	Wolters Kluwer	34	Senseo
15	NXP	35	Tomtom
16	Albert Heijn	36	Boskalis
17	Ziggo	37	USG People
18	Gall&Gall	38	Arcadis
19	Etos	39	Tencate
20	Gemalto	40	Independer

Table 3: Healthy companies [6]

4.2 Text analysis of the articles

After we have collected all 41 223 articles from the LexisNexis database, we followed steps described in sections 1.3 and 2.1.3 in order to initially process the data and clean the text and headlines of the articles to keep only the information necessary for the analysis. Moreover, we also removed all 80 company names from the text corpus because we already used the company names in the search engine of the LexisNexis

database to select the articles for our sample. Therefore, we considered it unnecessary to have company names present in the Document Term Matrix and TF-IDF Matrix as well as we wanted to avoid the risk of our predictive model predicting the bankruptcy based on company names.

4.2.1 Document Term Matrix

Having our text processed and split into word tokens, we were able to construct Document Term Matrix as the basis for further text analysis. Since we also wanted to include the text of headlines in our analysis, we created a special version of the Document Term Matrix. Instead of having 41 223 rows in the matrix corresponding to the number of articles, our special version of the Document Term Matrix consists of twice as many rows, 82 446, because we treat headlines as new articles for the process of matrix creation. However, rows of this matrix are well ordered and content and headline of an article are linked with that article in a way that document of order i is content of the article i and document of order $(i + 41223)$ is headline of the article i.

This version of the Document Term Matrix has 82 446 rows and 201 071 columns which means that our text corpus contains 82 446 documents or articles (but we know that real number of articles is 41 223 and second half of rows corresponds to headlines) and 201 071 unique words, which have appeared either in the content of the articles or their headlines.

For comparison, the Document Term Matrix, constructed out of articles' content only, contains 41 223 rows and 199 594 columns and Document Term Matrix from headlines has 41 223 rows and 23 708 columns. However, headlines and contents of articles contain 201 071 unique words and thus it can be seen that there is a significant overlap in words appearing in both matrices. Therefore, it is not a good idea to create two matrices and merge them later because we are interested in having only unique words in columns and not interested in having the same word in two columns just because it is in content-based Document Term Matrix as well as headline-based Document Term Matrix.

Moreover, it is understandable that headlines do not contribute much to the uniqueness of words in the text corpus since headlines of articles usually contain only

few words in order to briefly outline what the article is about and most probably the content of the article consists of these words as well.

4.2.2 Word clouds and letter clouds

Document Term Matrix also serves as a basis for the creation of word clouds or letter clouds, which were theoretically described in the section 2.1.4.3. We split the collection of headlines of our text corpus into two parts. First part consisted of 11 251 headlines solely about bankrupted companies and word cloud showing the most frequent words of these headlines is depicted in Figure 3. Second part consisted of 29 972 headlines solely about healthy companies and the lettercloud of letter 'H' is depicted in Figure 4.

Figure 3: Word cloud from headlines of articles about bankrupted companies

Figure 4: Letter cloud H from headlines of articles about healthy companies

4.2.3 TF-IDF

From the Document Term Matrix we created TF-IDF Matrix, a weighted matrix, which has the same dimensions as the Document Term Matrix - 82 446 rows and 201 071 columns. In order to be able to use TF-IDF as a basis for our predictive models, we had to decrease the column dimension of the matrix since we treat rows as observations and columns as predictors in the predictive model.

However, having way more predictors than observations is not the desired state, but rather the opposite. In order to decrease the column dimension, we followed the so-called 'one in ten' rule, rule of thumb for predictive modelling, which advices to have 10 observations for one predictor in order to predict the output adequatelly. In our case, this means that the final TF-IDF should have around 4 122 columns as we have 41 223 articles in reality.

We decreased the column dimensions of our TF-IDF matrices by using R command `removeSparseTerms` and by specifying the parameter `MaxSparsity`.

Interpretation of this procedure can be easily explained on solution of the following inequality for x which represents the number of documents containing the certain term:

$$1 - \frac{x}{\text{number of all documents}} \leq \texttt{MaxSparsity} \tag{20}$$

$$1 - \frac{x}{41223} \leq 0.993$$

$$x \geq 288.56$$

In this example, by setting the parameter `MaxSparsity` to 0.993 we would remove all terms which appeared in less than 289 documents out of 41 223.

The column dimension of TF-IDF was decreased to 4 499 by setting the parameter `MaxSparsity` to 0.997, which means that we removed all terms which appeared in less than 247 documents out of 82 446 documents (articles and their headlines).

Since the real number of articles is 41 223, we also want to use this number in our predictive models as number of observations and therefore we had to decrease the row dimension of TF-IDF matrix by merging articles with their corresponding headlines. Since the rows of this matrix are well ordered, this can be done easily by summing row of order i and the row of order $(i + 41223)$.

In addition, we considered a single word appearing in the headline of an article more important and crucial for the message of that article comparing to a single word appearing in the content of the article. We implemented this assumption by adding an arbitrary constant 2 into the summation formula followingly:

$$\text{row of order } i + 2 \ \cdot \text{row of order } (i + 41223) \tag{21}$$

This procedures have left us with TF-IDF of dimensions 41 223 rows and 4 499 columns.

Therefore, we kept 2.24 % of the most-frequent words from all the unique words in the text corpus for futher analysis and construction of a predictive model.

4.2.4 Enlargement of TF-IDF

After decreasing the column size of TF-IDF Matrix, we made a minor enlargement of the matrix by adding a few new columns (new predictors) to the TF-IDF Matrix assuming that they might improve the prediction power of our models.

4.2.4.1 Sentiment analysis

After conducting the sentiment analysis on each article we obtained a sentiment score for each article - a number between -1 and 1 indicating how positive, negative or neutral the article is. The maximum value, the most positive sentiment, was 0.25 and minimum, the most negative sentiment, was -0.20 .

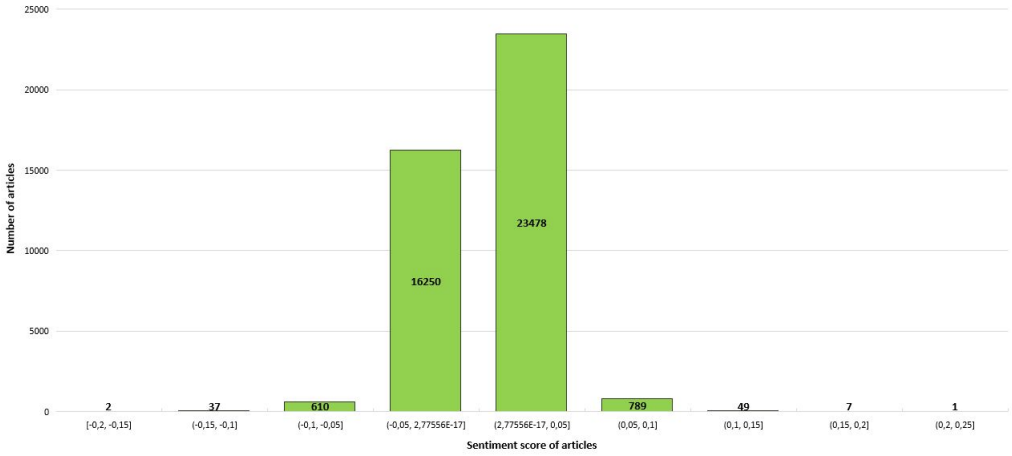


Figure 5: Histogram of sentiment scores of articles

As can be observed from the Figure 5, the majority of articles had their sentiment score very close to 0, thus being almost fully neutral. This is understandable since Het Financieele Dagblad is a factual based and high quality newspaper rather than the opposite.

Therefore, we also created additional variable which assigned a signum function to the sentiment score of the article. Signum function is defined as:

$$sgn(x) = \begin{cases} 1 & \text{if } x > 0 \\ 0 & \text{if } x = 0 \\ -1 & \text{if } x < 0 \end{cases} \tag{22}$$

Thus, with this variable we were interested only in classifying articles into 3 classes: positive, neutral and negative, but not caring about how much positive or negative they are.

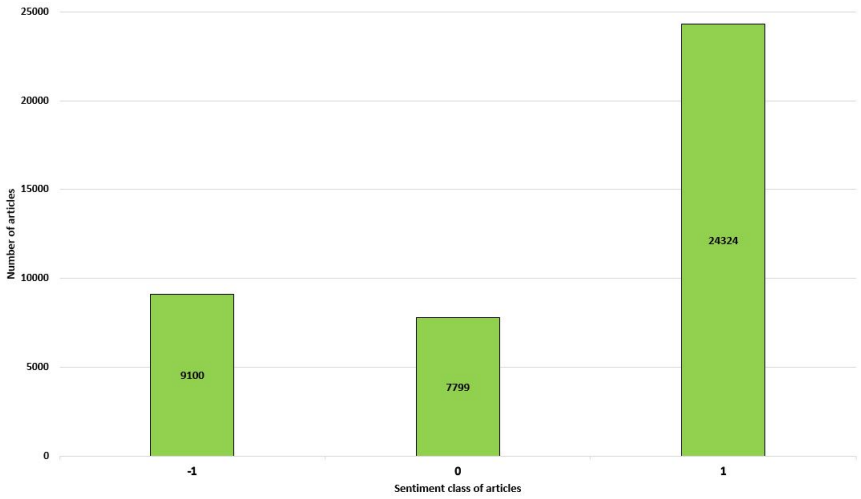


Figure 6: Histogram of sentiment classes of articles

Two new sentiment variables were then added as new columns to TF-IDF matrix and therefore our model got two new predictors.

4.2.4.2 Days before bankruptcy

During the collection process of articles from the LexisNexis database, we collected all articles about the bankrupted companies until their bankrupcy date. However, official bankruptcy date is determined by the decision of the court and it can be obvious already few days or weeks earlier that the company will most probably go bankrupt. Moreover, bankruptcy is the last stage of a quite long process following the stage in which the company asks for the suspension of payments.

We expect the newspapers informing about the bad situation of companies and their big chances to go bankrupt earlier before the actual bankruptcy happens. Since we know the date of each article and bankruptcy date of each bankrupted company in our sample, we created a new variable 'days before bankruptcy', which indicates for each article how many days before the bankruptcy of that particular company the article was published.

However, we did not use this variable as a new predictor but the purpose of creating this variable was to narrow our database of articles by excluding the articles about bankrupted companies which were published less than x days prior to the bankruptcy. The reason is that we wanted our model to detect the bankruptcy

earlier than just a few days before the decision of the court.

4.2.4.3 Date of articles

Undeniably, the time plays an important role in our analysis. In fact, articles about a single company form a sequence in time. In order to include the element of time in our analysis we looked at the frequency of articles about a single company with an assumption that there might be a pattern. For instance, we may expect that newspapers start writing more and more about the company which is in some trouble or is heading there.

Therefore, we created several new variables, specifically: 'last week', 'last month', 'last quarter', 'last half-year' and 'last year'. These numeric variables state for a certain article how many articles about the same company were published in last 7, 30, 90, 180 and 365 days, respectively. Since the amount of articles available is different for each company, we divided the number by the amount of articles available for the particular company in order to use relative values in our analysis instead of absolute.

The above mentioned 5 new variables were also added as new columns to TF-IDF matrix and will be used as predictors in the model.

4.2.5 Output of text mining

As the final output of the text analysis we shall consider large TF-IDF matrix with 41 223 rows, which represent the articles, and 4 508 columns. 4 507 columns represent predictor variables for the predictive model. 4 499 of these variables are words which appeared in the text or headlines of the articles. The additional predictors represent newly added variables for sentiment of articles and time element of articles. Last column, which is not considered as a predictor for our model, is only for a purpose of narrowing the database of articles.

4.3 Predictive modelling

In this section, it will be described how TF-IDF matrix was used to create and train the predictive models as well as which types of predictive models were created.

As an input in our model, we consider a row vector corresponding to one row of TF-IDF matrix, which represents one article from the text corpus. Our predictive models shall predict for a single article and based on the article predictions we will describe how the prediction for a company can be determined.

4.3.1 Days before bankruptcy

As was mentioned previously, our database of articles contains articles from bankrupted companies until their official bankruptcy date. However, we aimed to create a predictive model which can foresee coming bankruptcy of the company in advance. Therefore, before training the model, all articles from bankrupted companies, which were published 60 or less days prior to bankruptcy of the respective company, were removed. This narrowed the database of articles from bankrupted companies from 11 251 to 10 671.

4.3.2 Training the model

The event of bankruptcy which we aim to predict is binary: it has either happened or not happened. Therefore, we train the model on binary basis as well.

The main idea behind the training process is that the number 1, representing that the event of bankruptcy has happened, was assigned to all articles from bankrupted companies and number 0, representing that the event of bankruptcy has not happened, was assigned to all articles from healthy companies.

Thus, our model can learn using 10 671 examples what the signs of an article from bankrupted company are and on 29 972 examples it shall learn what are the signs of an article from healthy company.

Even though we assigned number 1 or 0 to each single article in our dataset, for training and testing purposes we always keep articles from the same company together and do not use them in both, training and testing set. The reason for this is that in the end our main aim is to predict the event of bankruptcy for a company based on the collection of its articles rather than for a single article.

4.3.3 Models used

We created several predictive models and compared their predictive capabilities and performance.

In order to have a class of regression models represented, we fitted a logistic regression model to our data. Main characteristics of the logistic regression model were described in the section 2.2.1.

To compare the performance of regression model to a machine learning model, we also fitted feedforward neural networks to the data, which were described in more detail in the section 2.2.2. In order to choose suitable feedforward neural network, we opted for experimental way and fitted several neural networks with different combinations of hidden layers and neurons, specifically:

- 1 hidden layer with 10 neurons
- 1 hidden layer with 50 neurons
- 1 hidden layer with 100 neurons
- 2 hidden layers with 10 neurons in the first layer and 5 neurons in the second
- 2 hidden layers with 50 neurons in the first layer and 20 neurons in the second
- 3 hidden layers with 100 neurons in the first layer, 30 neurons in the second layer and 10 neurons in the third

However, due to the size of our dataset and computational complexity, we were not able to try as many combinations as we would have wished for. After that we evaluated the accuracy of predictions to find the best performing combination of hidden layers and neurons.

The models were trained using the convergence threshold $\epsilon = 0.1$.

4.3.4 Cross-validation

Understandably, the final model shall be trained on all available data. However, cross-validation, explained in the section 2.2.3, can be used to obtain the approximation of model predictive power on unseen data.

We performed 8-fold cross-validation without combinations, which means that each observation was used for model testing exactly once. The main reason for choosing 8-fold cross-validation instead of leave-one-out cross-validation is computational feasi-

bility with respect to the size of our dataset. As stated previously, we are interested in event prediction for a company. However, we need to use predictions for articles to calculate the prediction for a company.

8-fold cross-validation in our case means that we split our dataset into 8 folds equally sized in terms of companies and unequally sized in terms of articles. Therefore, we chose articles from 10 companies as testing set and all articles from remaining 70 companies as training set. Moreover, 5 companies out of 10 tested companies are bankrupted and other 5 are healthy. The rest is done randomly where we use software to randomly order companies in order to create groups of 10 companies. Each company is used for testing exactly one time and remaining seven times it is used for training the model.

4.3.4.1 Results

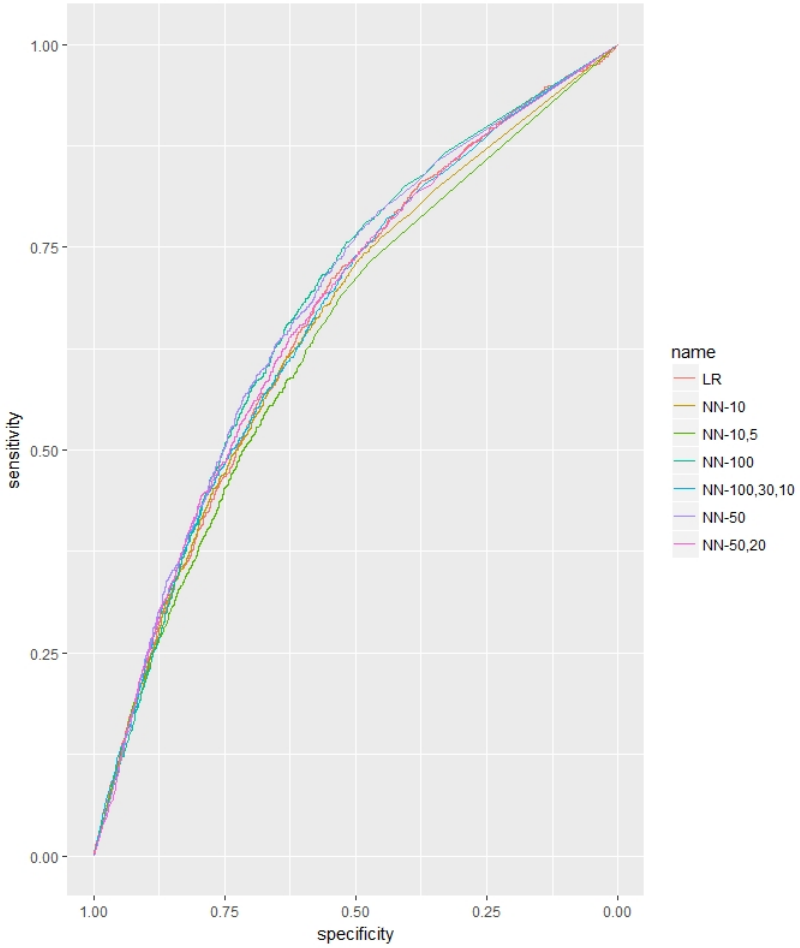


Figure 7: ROC curves for multiple trained models

We run 8-fold cross-validation for logistic regression and neural networks to have approximation of how well our models predict on unseen data.

All ROC curves lie above the linear line which means that models are better than just random guessing. However, there is no ROC curve being always above all the other ROC curves and therefore we also looked at AUC and ACC to evaluate the models. Decision threshold value of 0.5 was used to compute ACC.

	AUC	ACC
LR	0.6621	0.7099
NN – 10	0.6529	0.6744
NN – 50	0.6766	0.6880
NN – 100	0.6749	0.6842
NN – 10, 5	0.6395	0.6658
NN – 50, 20	0.6659	0.6876
NN – 100, 30, 10	0.6626	0.6839

Table 4: Computed AUC and ACC for the selected models

As can be clearly observed from the Table 4, logistic regression model has the highest ACC and neural network with 1 hidden layer and 50 neurons has the highest AUC as well as ACC amongst all fitted neural networks and therefore we choose this neural network as the best performing one. Further computations will be performed only on 2 models - logistic regression and neural network with 1 hidden layer and 50 neurons so that there is a representative from both classes of predictive models, regression models and machine learning models, for further analysis.

We used decision threshold value of 0.5 to create the confusion matrices from predictions. Therefore, we evaluated an article from bankrupted company as being predicted correctly if its prediction was greater or equal to 0.5. In case of an article from healthy company, the prediction was evaluated as correct if its value was below 0.5.

The confusion matrices of predictions are in Table 5 and 6.

		Prediction	
		1	0
True	1	3 337 (8.21 %)	7 334 (18.04 %)
	0	4 457 (10.97 %)	25 515 (62.78 %)

Table 5: Confusion matrix of predictions from logistic regression

		Prediction	
		1	0
True	1	5 304 (13.05 %)	5 367 (13.21 %)
	0	7 314 (17.99 %)	22 658 (55.75 %)

Table 6: Confusion matrix of predictions from neural network

As can be observed from Tables 5 and 6, in case of logistic regression, 70.99 % of all articles were predicted correctly and in case of neural network 68.80 % of all predictions were correct.

One may think that models are not any better than the model which assigns 0 to every article due to the imbalance in our dataset. In fact, by assigning 0 to each article, the prediction would be correct for 73.74 % articles.

Although our dataset is not balanced in terms of articles, it is perfectly balanced in terms of companies - having 40 bankrupted and 40 healthy companies. Thus, by assigning 0 to each article, the event prediction on a company level would be correct in 40 cases out of 80.

In the Section 4.3.5 we will show that, in case of prediction on a company level, our models are still better than that.

4.3.5 Company 'bankruptcy score'

If we look at companies in our sample, their articles and predictions form a sequence in time. Since a collection of articles may give us better picture of the company comparing to a randomly chosen single article, we aim to find the most suitable formula to determine the 'bankruptcy score' for a company based on the prediction scores predicted for the all articles of that particular company. Several ways of possible calculations of the 'bankruptcy score' and achieved results will be described in the next subsections.

4.3.5.1 Simple mean

The most straight-forward and easiest way of getting one final prediction from multiple predictions is taking the simple mean as a probability of the company going bankrupt.

To evaluate how good this method of calculation of the bankruptcy score is, we took the predictions from 8-fold cross-validation, applied the simple mean to them and constructed the ROC curves, computed ACC at a decision threshold value of 0.5 and constructed confusion matrices on a company level using the same decision threshold value.

Shape of ROC curves in Figure 8 gives us an indication of a good model performance, which is also supported by the calculated AUC values in Table 7.

However, low calculated values of ACC in combination with confusion matrices in Table 8 and 9 might be worrying.

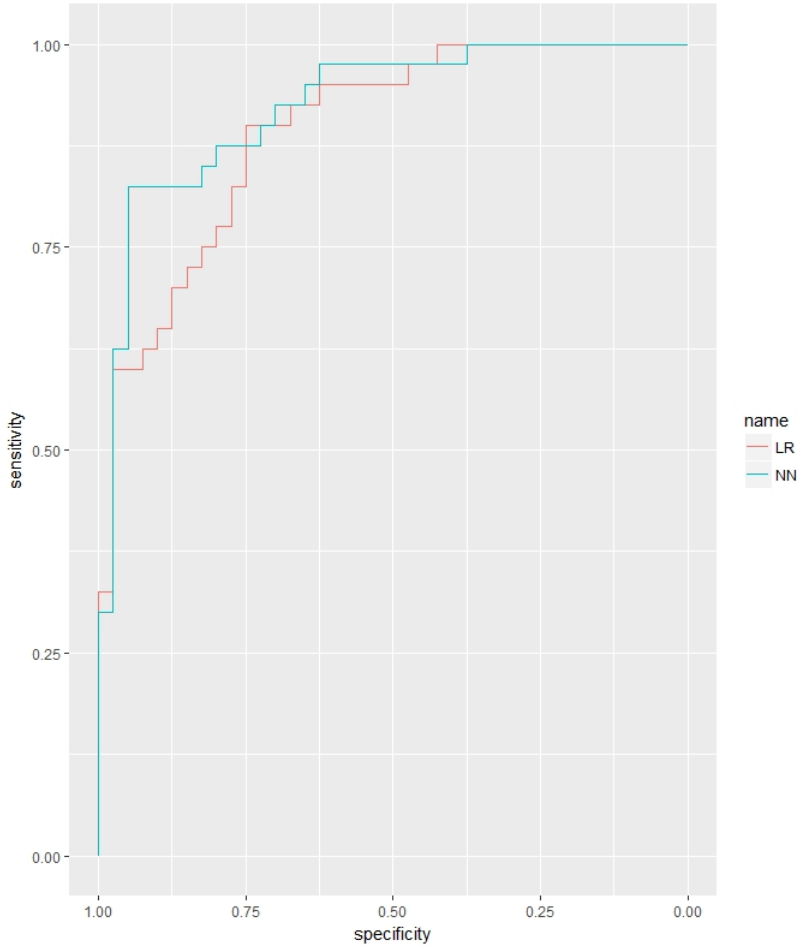


Figure 8: ROC curves for simple averages of predictions from logistic regression and neural network

	AUC	ACC	Sensitivity	Specificity
Logistic regression	0.8944	0.5875	0.175	1
Neural network	0.9244	0.675	0.375	0.975

Table 7: Computed performance measures for the selected models

		Prediction	
		1	0
True	1	7 (8.75 %)	33 (41.25 %)
	0	0 (0.00 %)	40 (50.00 %)

Table 8: Confusion matrix of simple averages of predictions from logistic regression ($\tau = 0.5$)

As can be seen from Table 8, 58.75% of companies were predicted correctly. In case of neural network, company prediction using simple mean was correct for 67.5 % of companies as shown in Table 9. Both models have very high specificity and low sensitivity.

		Prediction	
		1	0
True	1	15 (18.75 %)	25 (31.25 %)
	0	1 (1.25 %)	39 (48.75 %)

Table 9: Confusion matrix of simple averages of predictions from neural network ($\tau = 0.5$)

Since ROC curve plots true positive rate against false positive rate at various decision threshold values and AUC reached very high scores for our models, but on the other hand, ACC determined at decision threshold value of 0.5 reached rather low scores in comparison, it might seem reasonable to adjust the decision threshold value in order to improve the predictions at the company level.

In order to explain this need for adjustment, we believe that low values of ACC may be caused by the class imbalance in our dataset in terms of articles as

the model has much more examples of 0's to learn from comparing to examples of 1's. Therefore, the model may be quite 'confident' about predicting that the article comes from healthy company but rather 'careful' about predicting that article comes from bankrupted company. However, there still might be difference in predictions of articles from healthy and bankrupted companies but cannot be correctly captured using decision threshold value of 0.5 and therefore evaluating at different decision threshold value is appropriate.

To test if the above mentioned assumption is correct, we decided to follow 2 techniques of decision threshold adjustment which were described and performed in [1].

4.3.5.2 Simple mean - decision threshold adjustment

The first tested approach of decision threshold adjustment suggests that the effect of unequal class sizes might be alleviated by imposing more weight on the minority class [1]. In our case the adjusted decision threshold value turns out to be equal to $\frac{n_1}{n}$ where n_1 is number of observation belonging to minor class (articles from bankrupted companies) and n is number of all observations. Therefore, the new decision threshold value shall be:

$$\tau = \frac{10671}{40643} = 0.2626$$

		Prediction	
		1	0
True	1	38 (47.5 %)	2 (2.5 %)
	0	21 (26.25 %)	19 (23.75 %)

Table 10: Confusion matrix of simple averages of predictions from logistic regression ($\tau = 0.2626$)

		Prediction	
		1	0
True	1	40 (50 %)	0 (0 %)
	0	14 (17.5 %)	26 (32.5 %)

Table 11: Confusion matrix of simple averages of predictions from neural network ($\tau = 0.2626$)

	ACC	Sensitivity	Specificity
Logistic regression	0.7125	0.95	0.475
Neural network	0.825	1	0.65

Table 12: Computed performance measures for the selected models

Sensitivity of both models has increased a lot but on the other hand specificity has decreased significantly. ACC has increased for both models and therefore using the lower decision threshold value more companies were predicted correctly.

We will test another approach of decision threshold adjustment to see whether it brings a better trade-off between sensitivity, specificity and accuracy. The second approach uses the monotonic relationship between τ and sensitivity and between τ and specificity. It suggests to choose some required sensitivity and specificity levels and limit the desired value of decision threshold by the largest τ using which the required sensitivity level is met and by the smallest τ using which the required specificity level is met. The decision threshold value is then the value which maximizes the accuracy [1].

Since the choice of required sensitivity and specificity can be arbitrary, we decided to require each of these two values to be at least 0.75.

From Figure 9 and 10 the change in specificity and sensitivity at different decision threshold values can be observed.

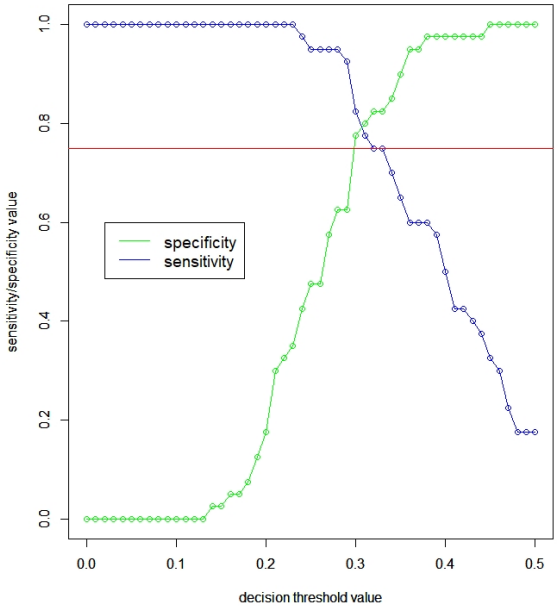


Figure 9: Sensitivity and specificity at different decision threshold values - logistic regression

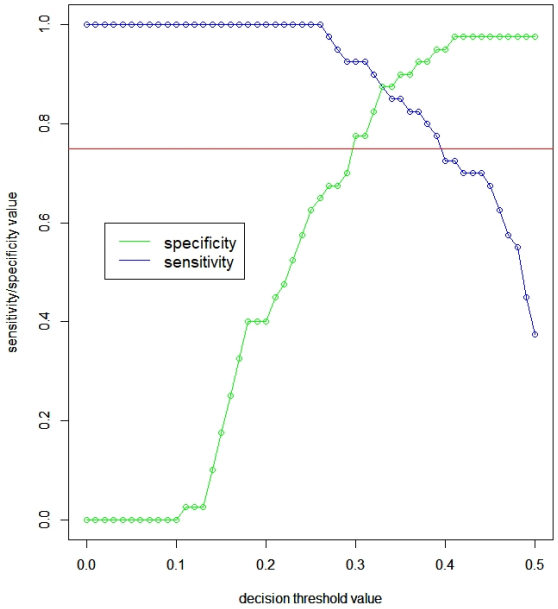


Figure 10: Sensitivity and specificity at different decision threshold values - neural network

Figures 11 and 12 depict the accuracies at different decision threshold values

and vertical red lines indicate the interval in which the decision threshold value must lie so that both, specificity and sensitivity, are greater or equal to 0.75.

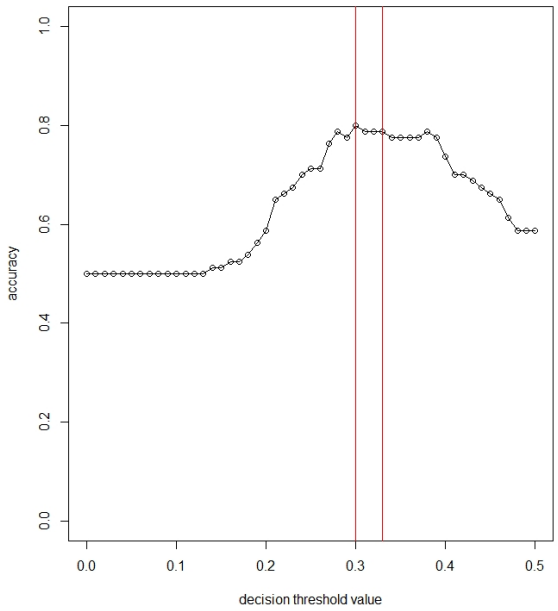


Figure 11: Accuracy at different decision threshold values - logistic regression

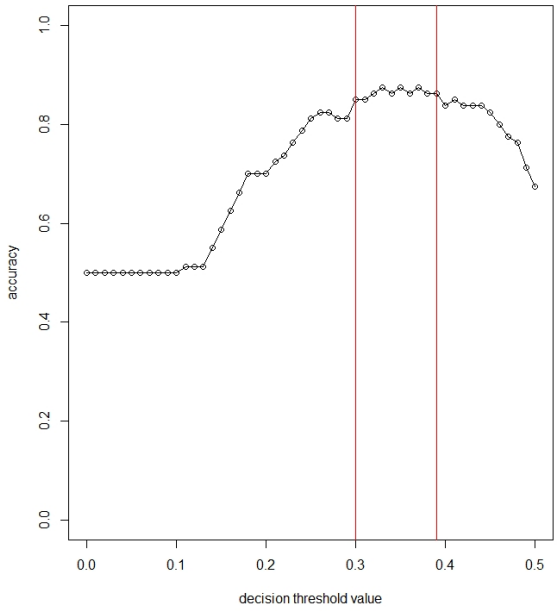


Figure 12: Accuracy at different decision threshold values - neural network

For the simple average of predictions from logistic regression, the optimal de-

cision threshold value is 0.3 and reached accuracy of predictions is 0.8. In case of simple average of predictions from neural network, the optimal decision threshold value is 0.33 and corresponding accuracy is 0.875.

	ACC	Sensitivity	Specificity
Logistic regression ($\tau = 0.3$)	0.8	0.825	0.775
Neural network ($\tau = 0.33$)	0.875	0.875	0.875

Table 13: Computed performance measures for the selected models

		Prediction	
		1	0
True	1	33 (41.25 %)	7 (8.75 %)
	0	9 (11.25 %)	31 (38.75 %)

Table 14: Confusion matrix of simple averages of predictions from logistic regression ($\tau = 0.3$)

		Prediction	
		1	0
True	1	35 (43.75 %)	5 (6.25 %)
	0	5 (6.25 %)	35 (43.75 %)

Table 15: Confusion matrix of simple averages of predictions from neural network ($\tau = 0.33$)

As can be seen from Table 13, 14 and 15, high accuracies were reached for both models while meeting the required sensitivity and specificity level. In all cases, neural network outperformed the logistic regression model.

From the conducted experiments, computed approximations of model performance, we evaluate neural network with 1 hidden layer and 50 neurons as the best performing model on a company level. If simple average of the predictions at article level is evaluated using the decision threshold value 0.33, the prediction accuracy at company level reaches its maximum - 0.875 while condition on sensitivity and specificity, to be at least 0.75, is met.

4.3.5.3 Weighted mean

It may seem reasonable to consider weighting the articles according to their date of publication in a way that the older articles would affect the bankruptcy score less than the newer ones.

In order to test if this assumption improves the predictions, we used two schemes to calculate the weighted mean instead of simple mean. In case of daily-weighted scheme, the weight 1 was assigned to the oldest article of the company and other articles were given weights of:

$$1 + \text{number of days from the oldest article of the same company} \tag{23}$$

In case of weekly-weighted scheme, the weight 1 was assigned to the oldest article of the company and other articles were given weights of:

$$1 + \text{number of weeks from the oldest article of the same company} \tag{24}$$

From Figure 13 and 14 it can be observed that for both, logistic regression and neural network, the ROC curve for simple mean lies above the ROC curves for weighted means at all data points.

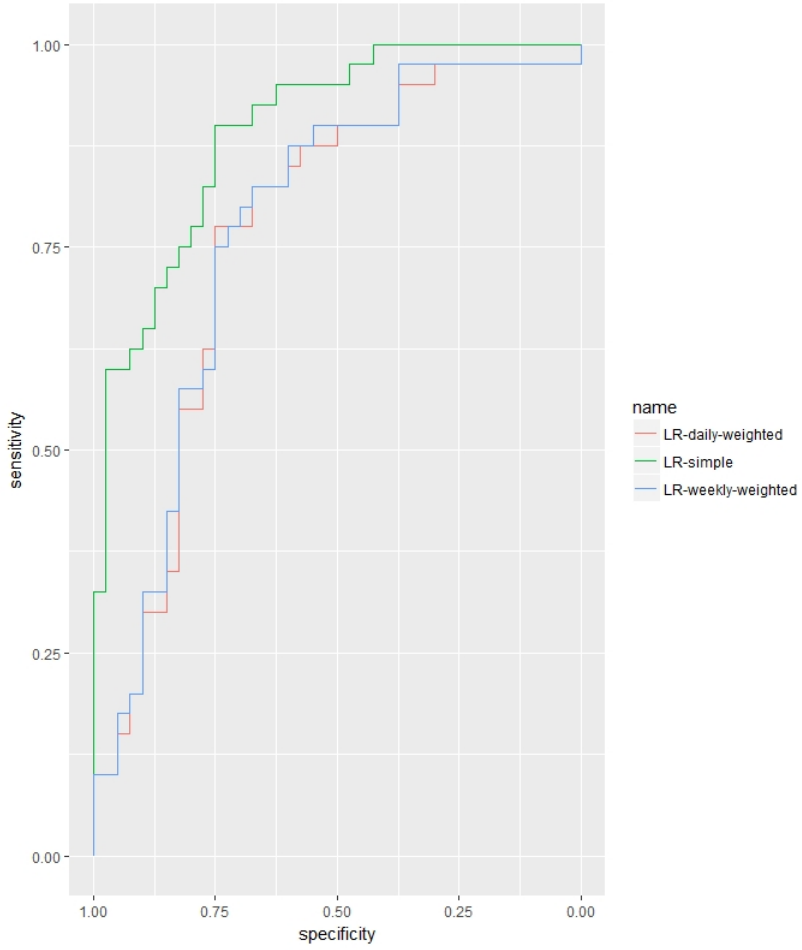


Figure 13: ROC curves for predictions from logistic regression - simple and weighted means

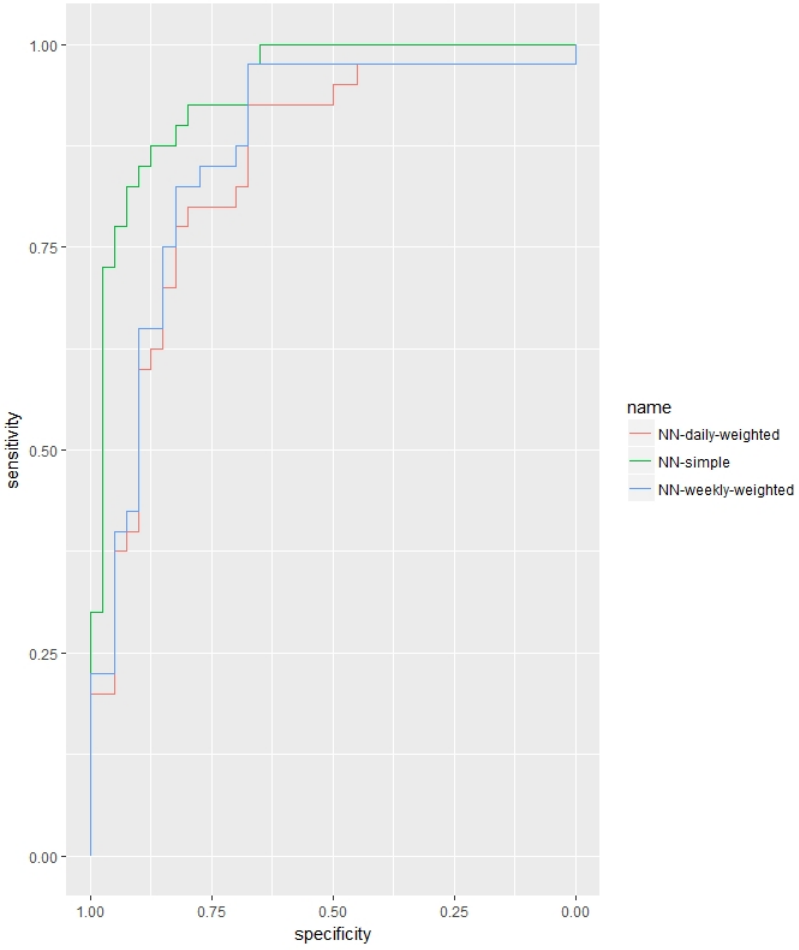


Figure 14: ROC curves for predictions from neural network - simple and weighted means

Therefore, decision threshold adjustment for weighted average will not outperform decision threshold adjustment for simple average and it will not be performed.

It can be concluded that schemes for weighted average do not bring any improvement of predictions on company level comparing to simple average.

4.4 Final model framework

It is important to realize that the outcome of this project cannot be a specific model but only a framework, described in detail in this chapter, using which the model can be constructed and some approximations of the prediction accuracies of the model.

The reason for this is hidden in the fundamental basis of this project - the construction of TF-IDF Matrix.

This matrix has to be created from all articles, also the articles of tested company. Therefore, every time the new article is added to training or testing set, TF-IDF Matrix has to be recreated so that it contains words of newly added article.

Based on the current dataset of articles and companies, model training experiments performed, computed model performance from 8-fold cross-validation, we may conclude that the final model framework should be neural network with 1 hidden layer and 50 neurons. To calculate 'bankruptcy score' of company, simple average of articles' predictions shall be computed and evaluated using decision threshold value 0.33 where bankruptcy score above 0.33 may indicate a potential risk of particular company heading into bankruptcy soon and therefore special attention should be given to such company.

However, in case of significant changes, for instance if training set is enlarged, model performance shall be approximated again as parameters of neural network or value of decision threshold may not be optimal.

5 Futher possibilities

Almost every project from real life has limitations or imperfections and this project is not an exception. This chapter consists of more detailed descriptions of several limitations, which were encountered during this project. Solving or reducing these limitations may serve as an inspiration for further improvements of the methodology in the future.

5.1 Other events

Other events, for instance the events defined in Chapter 3, can be predicted using the framework developed in Chapter 4. The crucial step is to identify sample of companies, which were affected by particular event and date when the event has occurred.

5.2 Data collection

Services of library of Vrije Universiteit Amsterdam include access to LexisNexis database but some restrictions are imposed. Maximum amount of 200 articles can be downloaded at a time, which makes the process of data collection very costly with respect to time.

Moreover, search engine of LexisNexis can return maximum of 3 000 results, which meet the search criteria and therefore some results might not be shown in case of more than 3 000 results.

Lastly, it is not possible to download all articles of selected newspaper but there have to be some search criteria to find certain articles. We used company names in the search engine. However, articles are often about more than just one company which may have caused that same article appeared multiple times in our sample. In addition, some articles might have mentioned the company in a very minor sense but were considered as articles about that particular company since LexisNexis search has returned them.

The above mentioned obstacles can be overcome by using LexisNexis database without restrictions or by using a different data collection process.

5.3 TF-IDF

TF-IDF Matrix is a powerful concept in text mining to numerically represent large set of documents. However, it has some disadvantages as well.

TF-IDF counts frequencies of words and using the special formula it assigns higher weights to words which are specific or important for a certain document. Unfortunately, TF-IDF Matrix does not take into consideration any relationship between the words or how the words form sentences together. This obstacle may be overcome by using more advanced concepts in text mining, for instance so-called 'word2vec'.

Moreover, there are much more words in text corpus than documents in text corpus. However, if we want to use TF-IDF as a basis of a predictive model, we treat row as observation. Therefore, we need less predictors than observations and column dimension has to be radically decreased. In our analysis, we used only 2.24% of all words and therefore lots of predictive potential was lost.

5.4 Computational complexity

Large size of the dataset causes that running time of the models can take very long time. However, nowadays, this obstacle can be overcome by using powerful computational machines.

The framework developed in this master thesis is computationally complex also because of TF-IDF matrix, which needs to be recreated from scratch every time the new article is added to the training set or articles of new company are added to the tested set.

5.5 Financial data

Our models are only using textual information available from articles of financial newspapers and the approximation of model's performance indicates a high prediction potential. We believe that addition of financial performance measures, or suitable combination of prediction based on textual data with prediction based on financial data, could significantly improve the model prediction power.

5.6 Vision of a 'perfect' model

Our long-term vision of model, which could be interesting for the real business world nowadays and useful for companies, is model which would automatically gather articles of financial newspapers, process them, analyse the textual information from articles and based on this information it would, using the predictive model, regularly test and check what are the odds that companies are 'running into' some events.

In case that odds reach certain decision threshold value, notification would be sent so that status of the company can be further investigated or the performance of company can be closely monitored in the future.

Conclusion

This master thesis is a summary of internship project carried at Deloitte Nederland in which we investigated whether it is possible to use articles of financial newspapers to predict the occurrence of certain events at certain companies. The topic may be considered as an ambitious one and was not widely investigated yet.

We focused on Dutch market, using the articles of Dutch financial newspaper Het Financieele Dagblad as data source and choosing only articles about Dutch companies.

Since no known and widely used concepts were developed, we created own methodology of event prediction using the concept of TF-IDF matrix enlarged by sentiment and time variables to numerically represent large set or articles.

From the perspective of events, the methodology was built for the event of bankruptcy but can be easily applied to prediction of other events, for instance fraud, corruption or financial problems.

Even though the concrete model was created, a model framework shall be considered as the output of this thesis since TF-IDF matrix has to be recreated every time new article is added to either training or testing set.

However, to have an idea what can be expected from the model, an approximation of model performance was computed using techniques of cross-validation.

Since the aim was to predict event for a company, sequence of articles from the company and their predictions were used to determine so-called 'bankruptcy score'. From the conducted experiments and computed performance approximations, it has been shown that simple average shall be used to calculate the bankruptcy score and decision threshold value shall be adjusted due to the significant imbalance in our dataset in terms of articles.

The best possible result was achieved using neural network model with 1 hidden layer and 50 neurons and evaluating the 'bankruptcy score' using adjusted decision threshold value of 0.33. The accuracy of company prediction reached value of 0.875, with 70 companies out of 80 being predicted correctly, while keeping model sensitivity and specificity above the lowest required value of 0.75.

Nevertheless, these results should be understood only as approximations and in case of significant changes in dataset it may be necessary to conduct new experiments and

calculate new approximations of model performance.

Moreover, in this thesis we also compared prediction performance of 2 classes of predictive models - regression models and machine learning. We can conclude that machine learning model, specifically neural network with 1 hidden layer and 50 neurons, outperforms regression model, specifically logistic regression model.

Unfortunately, this methodology contains multiple imperfections and we believe that solving them may lead to creation of very powerful predictive model. LexisNexis database, used as our data source, imposed restrictions on download as well as its search engine returns limited amount of results. In addition, concept of TF-IDF Matrix has a downside of not considering relationship between words or how words form sentences together. Only 2.24 % of all words were used in predictive models and therefore lots of potential from textual data has been unused. Models are also computationally very complex but this can be overcome using computationally powerful machines nowadays.

An idea also worth to consider is to combine textual data with financial data for improved event prediction.

In the end, our long-term vision of a 'perfect' model would be a model which automatically gathers and process the textual data, regularly test bankruptcy score or other 'event score' of companies and sends an alert when a certain decision threshold value was reached so that the company, which is possibly heading towards some event, can be closely monitored.

To conclude, we consider as one of the main outcomes the concrete methodology, built in exploratory and experimental way, to solve task which was not investigated very much yet.

As the one final and the most important message to take, this thesis shall serve as a proof that it certainly makes sense to use textual information to predict something as 'big' as an event and hopefully further improvements of the methodology will be conducted in order to fully make a use of valuable information hidden inside the textual data.

References

[1] Chen, J.J.; Tsai, C.; Moon, H.; Ahn, H.; Young, J.J.; Chen, C.: *The Use of Decision Threshold Adjustment in Classification for Cancer Prediction*, Division of Biometry and Risk Assessment, National Center for Toxicological Research, Arkansas, USA, 2006, available online (26.07.2018):
http://www.ams.sunysb.edu/~hahn/psfile/papthres.pdf

[2] Benoit, K.; Welbers, K.; Van Atteveldt, W.: *Text Analysis in R*, Communication Methods and Measures, Vol. 11, 2017

[3] Bishop, C.M.: *Pattern recognition and machine learning*, Springer, New York, 2006, available online (26.07.2018):
http://users.isr.ist.utl.pt/~wurmd/Livros/school/Bishop%20-%20Pattern%20Recognition%20And%20Machine%20Learning%20-%20Springer%20%202006.pdf

[4] Concultancy.nl: *De 10 grootste faillissementen in Nederlandse historie*, August 2015, available online (26.07.2018):
https://www.consultancy.nl/nieuws/10967/de-10-grootste-faillissementen-in-nederlandse-historie

[5] Concultancy.nl: *De 25 grootste faillissementen van retailketens en winkels*, March 2016, available online (26.07.2018):
https://www.consultancy.nl/nieuws/11992/de-25-grootste-faillissementen-van-retailketens-en-winkels

[6] Concultancy.nl: *De 50 meest waardevolle bedrijven — merken van Nederland*, 21 August 2017, available online (26.07.2018):
https://www.consultancy.nl/nieuws/14501/de-50-meest-waardevolle-bedrijven-merken-van-nederland

[7] Czepiel, S.A.: *Maximum Likelihood Estimation of Logistic Regression Models: Theory and Implementation*, Semantic Scholar, available online (26.07.2018):
https://www.semanticscholar.org/paper/Maximum-

Likelihood-Estimation-of-Logistic-Models-%3A-Czepiel/fd7035ab3fca912c1f77c34ff0c4435d41ebd0d3

[8] Deloitte website: *About Deloitte*, available online (26.07.2018): https://www2.deloitte.com/global/en/pages/about-deloitte/articles/about-deloitte.html

[9] Deloitte Nederland website: *Over ons*, available online (26.07.2018): https://www2.deloitte.com/nl/nl/pages/over-deloitte/articles/waar-wij-voor-staan.html

[10] Fawcett, T.: *An introduction to ROC analysis*, Science Direct, December 2005, available online (26.07.2018): http://people.inf.elte.hu/kiss/11dwhdm/roc.pdf

[11] Feinerer, I.; Hornik, K,; Meyer, D.: *Text Mining Infrastructure in R*, Journal of Statistical Software, Volume 25, Issue 5, March 2008, available online (26.07.2018): http://www.stat.wvu.edu/~jharner/courses/dsci503/docs/tm.pdf

[12] Feldman, R.; Sanger, J.: *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*, Cambridge University Press, New York, February 2007, available online (26.07.2018): https://wtlab.um.ac.ir/images/e-library/text_mining/The%20Text%20Mining%20HandBook.pdf

[13] Fine, T.L.: *Feedforward Neural Network Methodology*, Statistics for Engineering and Information Science, Springer-Verlag New York, 1999

[14] Fritsch, S.; Gunther, F.: *neuralnet: Training of Neural Networks*, The R Journal Vol. 2/1, June 2010, available online (26.07.2018): https://journal.r-project.org/archive/2010/RJ-2010-006/RJ-2010-006.pdf

[15] de Gunst, M.C.M.: *Statistical Models*, Vrije Universiteit Amsterdam, Lecture notes - course in Statistical models, January 2013

[16] Investopedia: *Bankruptcy*, available online (26.07.2018):
`https://www.investopedia.com/terms/b/bankruptcy.asp`

[17] Investopedia: *Corruption*, available online (26.07.2018):
`https://www.investopedia.com/terms/c/corruption.asp`

[18] Investopedia: *Financial distress*, available online (26.07.2018):
`https://www.investopedia.com/terms/f/financial_distress.asp`

[19] Investopedia: *Fraud*, available online (26.07.2018):
`https://www.investopedia.com/terms/f/fraud.asp`

[20] Investopedia: *Takeover*, available online (26.07.2018):
`https://www.investopedia.com/terms/t/takeover.asp`

[21] Investopedia: *Tax evasion*, available online (26.07.2018):
`https://www.investopedia.com/terms/t/taxevasion.asp`

[22] Kambies, T; Mittal, N.; Roma, P.; Sharma, S.K.: *Dark Analytics*, Tech Trends 2017: The kinetic enterprise, Deloitte University Press, 2017, available online(26.07.2018):
`https://www2.deloitte.com/content/dam/Deloitte/global/Documents/Technology/gx-tech-trends-the-kinetic-enterprise.pdf`

[23] MT.nl: *De 12 grootste faillissementen van het jaar*, December 2015, available online (26.07.2018):
`https://www.mt.nl/business/de-12-grootste-faillissementen-van-het-jaar/88810`

[24] Perkins, J.: *Python 3 Text Processing with NLTK 3 Cookbook*, Packt Publishing Ltd., August 2014, available online (26.07.2018):
`http://zempirians.com/ebooks/Jacob%20Perkins-Python%203%20Text%20Processing%20with%20NLTK%203%20Cookbook-Packt%20Publishing%20-%20ebooks%20Account%20%282014%29.pdf`

[25] RDocumentation: *SentimentAnalysis*, available online (26.07.2018):
`https://cran.r-project.org/web/packages/SentimentAnalysis/SentimentAnalysis.pdf`

[26] RDocumentation: *weightTfIdf*, available online (26.07.2018):
`https://www.rdocumentation.org/packages/tm/versions/0.7-3/topics/weightTfIdf`

Appendix

Script 1: Python script for initial data processing

```
import os
import re
import xml.etree.cElementTree as ET
import nltk
import pandas as pd
import numpy as np
from nltk import sent_tokenize, word_tokenize, pos_tag, FreqDist
import textmining
import stemming
import datetime
from langdetect import detect
from nltk.corpus import stopwords

#os.chdir('C:/Users/jasivicek/Documents/LNhealthy')
#directory = os.listdir('C:/Users/jasivicek/Documents/LNhealthy')
os.chdir('C:/Users/jasivicek/Documents/LNbankrupted')
directory = os.listdir('C:/Users/jasivicek/Documents/LNbankrupted')

for k in range(len(directory)):
    exec(f'cat_{k} = pd.DataFrame(index=range(0,200), columns=["date","
        headline","article","language","year","month","day"])')
    exec(f'cat_{k} = cat_{k}.fillna(0)')

dataf = pd.DataFrame(index=range(0,len(directory)), columns=range(0,2))
dataf = dataf.fillna(0)
dataf

dataf.dtypes
dataf = dataf.astype('object')

i=0
for file in directory:
    insert=directory[i]
```

```
    dataf.at[i,0] = open(insert,"r",encoding="utf8").read().replace('\n',
        ' ').split("DOCUMENTS")
    del dataf.at[i,0][0]
    for m in range(0, len(dataf.at[i,0])) :
        dataf.at[i,0][m] = " ".join(re.split("\s+", dataf.at[i,0][m],
            flags=re.UNICODE))
        if 'BODY' in dataf.at[i,0][m]:
            exec(f'cat_{i}.iloc[m,2] = dataf.at[i,0][m].split("BODY: ",1)
                [1]')
        if 'HEADLINE' in dataf.at[i,0][m]:
            exec(f'cat_{i}.iloc[m,1] = dataf.at[i,0][m].split("HEADLINE: 
                ",1)[1].split(" BODY: ",1)[0]')
            if 'All Rights Reserved Het Financieele Dagblad' in dataf.at[
                i,0][m]:
                exec(f'cat_{i}.iloc[m,0] = dataf.at[i,0][m].split("All 
                    Rights Reserved Het Financieele Dagblad ",1)[1].split
                    (" HEADLINE: ",1)[0]')
            else:
                exec(f'cat_{i}.iloc[m,0] = dataf.at[i,0][m].split("Het 
                    Financieele Dagblad ",1)[1].split(" HEADLINE: ",1)[0]
                    ')
        else:
            exec(f'cat_{i}.iloc[m,1] = dataf.at[i,0][m].split("All Rights
                Reserved Het Financieele Dagblad ",1)[1].split(" BODY: 
                ",1)[0]')
    i=i+1
    k=str(i)
    print("file"+k+"processed")

frames=[cat_0,cat_1]

#50 files
#frames = [cat_0, cat_1, cat_2, cat_3,cat_4,cat_5,cat_6,cat_7,cat_8,cat_9
    ,cat_10,cat_11,cat_12,cat_13,cat_14,cat_15,cat_16,cat_17,cat_18,cat_19
    ,cat_20,cat_21,cat_22,cat_23,cat_24,cat_25,cat_26,cat_27,cat_28,cat_29
    ,cat_30,cat_31,cat_32,cat_33,cat_34,cat_35,cat_36,cat_37,cat_38,cat_39
    ,cat_40,cat_41, cat_42, cat_43, cat_44, cat_45, cat_46, cat_47, cat_48
    , cat_49]
```

```
df = pd.concat(frames)
df.index = range(len(df))

df = df[df.article != 0]  #delete where is no article in the data frame
df.index = range(len(df))

df.dtypes
df = df.astype('object')

import dateparser

date = str(dateparser.parse(df.iloc[30,0]).day)+"-"+str(dateparser.parse(
    df.iloc[30,0]).month)+"-"+str(dateparser.parse(df.iloc[30,0]).year)

for m in range(len(df)):
    df = df.astype('str')
    df.iloc[m,0] = re.sub(r"Het␣Financieele␣Dagblad",'',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r'\w+dag\s?','',df.iloc[m,0]) #delete weekday
        names from the date
    df.iloc[m,0] = re.sub(r'\w+day\s?','',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"januari",'january',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"februari",'february',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"maart",'march',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"mei",'may',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"juni",'june',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"juli",'july',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"augustus",'august',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"oktober",'october',df.iloc[m,0])
    df.iloc[m,0] = re.sub(r"12:00␣AM␣GMT",'',df.iloc[m,0])
    df.iloc[m,0] = df.iloc[m,0].lower()     #remove capital letters
    df.iloc[m,2] = df.iloc[m,2].lower()
    df.iloc[m,2] = re.sub(r'http\S+', '␣', df.iloc[m,2])
    df.iloc[m,0] = "␣".join(re.split("\s+", df.iloc[m,0], flags=re.
        UNICODE)) #remove duplicate whitespace
    df.iloc[m,0] = re.sub("^\s+|\s+$", "", df.iloc[m,0], flags=re.UNICODE
        ) #remove whitespace from the beggining and the end
```

```
    df.iloc[m,1] = " ".join(re.split("\s+", df.iloc[m,1], flags=re.
        UNICODE)) #remove duplicate whitespace
    df.iloc[m,1] = re.sub("^\s+|\s+$", "", df.iloc[m,1], flags=re.UNICODE
        ) #remove whitespace from the beggining and the end
    df.iloc[m,2] = " ".join(re.split("\s+", df.iloc[m,2], flags=re.
        UNICODE)) #remove duplicate whitespace
    df.iloc[m,2] = re.sub("^\s+|\s+$", "", df.iloc[m,2], flags=re.UNICODE
        ) #remove whitespace from the beggining and the end
    df.iloc[m,3] = detect(df.iloc[m,2])
    df.iloc[m,4] = str(dateparser.parse(df.iloc[m,0]).year)
    df.iloc[m,5] = str(dateparser.parse(df.iloc[m,0]).month)
    df.iloc[m,6] = str(dateparser.parse(df.iloc[m,0]).day)
    o=str(m)
    if (m % 100==0):
        print('article '+o)

df.language.value_counts()
df = df[df.language == 'nl']
df.index = range(len(df))

nameofnewfile = "bankr_dsbbank"

#write each article into separate txt file
for m in range(len(df)):
    p=m+1
    o=str(p)
    if (m % 100==0):
        print('art '+o)
    if (p<10):
      text_file = open(nameofnewfile+"0000"+o+".txt", "w", encoding="utf
          -8")
      text_file.write(df.iloc[m,4] + "\n")
      text_file = open(nameofnewfile+"0000"+o+".txt", "a", encoding="utf
          -8")
      text_file.write(df.iloc[m,5] + "\n")
      text_file = open(nameofnewfile+"0000"+o+".txt", "a", encoding="utf
          -8")
      text_file.write(df.iloc[m,6] + "\n")
```

```
        text_file = open(nameofnewfile+"0000"+o+".txt", "a", encoding="utf
            -8")
        text_file.write(df.iloc[m,1] + "\n")
        text_file = open(nameofnewfile+"0000"+o+".txt", "a", encoding="utf
            -8")
        text_file.write(df.iloc[m,2] + "\n")
        text_file.close()
    elif (p>9 and p<100):
        text_file = open(nameofnewfile+"000"+o+".txt", "w", encoding="utf-8
            ")
        text_file.write(df.iloc[m,4] + "\n")
        text_file = open(nameofnewfile+"000"+o+".txt", "a", encoding="utf-8
            ")
        text_file.write(df.iloc[m,5] + "\n")
        text_file = open(nameofnewfile+"000"+o+".txt", "a", encoding="utf-8
            ")
        text_file.write(df.iloc[m,6] + "\n")
        text_file = open(nameofnewfile+"000"+o+".txt", "a", encoding="utf-8
            ")
        text_file.write(df.iloc[m,1] + "\n")
        text_file = open(nameofnewfile+"000"+o+".txt", "a", encoding="utf-8
            ")
        text_file.write(df.iloc[m,2] + "\n")
        text_file.close()
    elif (p>99 and p<1000):
        text_file = open(nameofnewfile+"00"+o+".txt", "w", encoding="utf-8"
            )
        text_file.write(df.iloc[m,4] + "\n")
        text_file = open(nameofnewfile+"00"+o+".txt", "a", encoding="utf-8"
            )
        text_file.write(df.iloc[m,5] + "\n")
        text_file = open(nameofnewfile+"00"+o+".txt", "a", encoding="utf-8"
            )
        text_file.write(df.iloc[m,6] + "\n")
        text_file = open(nameofnewfile+"00"+o+".txt", "a", encoding="utf-8"
            )
        text_file.write(df.iloc[m,1] + "\n")
```

```
        text_file = open(nameofnewfile+"00"+o+".txt", "a", encoding="utf-8"
            )
        text_file.write(df.iloc[m,2] + "\n")
        text_file.close()
    elif (p>999 and p<10000):
        text_file = open(nameofnewfile+"0"+o+".txt", "w", encoding="utf-8")
        text_file.write(df.iloc[m,4] + "\n")
        text_file = open(nameofnewfile+"0"+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,5] + "\n")
        text_file = open(nameofnewfile+"0"+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,6] + "\n")
        text_file = open(nameofnewfile+"0"+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,1] + "\n")
        text_file = open(nameofnewfile+"0"+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,2] + "\n")
        text_file.close()
    else:
        text_file = open(nameofnewfile+o+".txt", "w", encoding="utf-8")
        text_file.write(df.iloc[m,4] + "\n")
        text_file = open(nameofnewfile+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,5] + "\n")
        text_file = open(nameofnewfile+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,6] + "\n")
        text_file = open(nameofnewfile+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,1] + "\n")
        text_file = open(nameofnewfile+o+".txt", "a", encoding="utf-8")
        text_file.write(df.iloc[m,2] + "\n")
        text_file.close()
```

Script 2: R script for text mining and creation of TF-IDF Matrix

```
library(tm)
library(SnowballC)
library(wordcloud)
library(data.table)

##create data frame to store article data
df <- data.frame(date=character(),title=character(),content=character(),
```

```
    stringsAsFactors=FALSE)

setwd("C:/Users/jasivicek/Desktop/text mining and R practice/articles")
file.names <- list.files(path = "C:/Users/jasivicek/Desktop/text mining
    and R practice/articles")
files <- as.character(list.files(path="C:/Users/jasivicek/Desktop/text
    mining and R practice/articles"))

##load articles into the data frame
for (i in 1:length(files)) {
if (i==1) {print(Sys.time())}
df[i,]<-readLines(files[i], 3, ok = TRUE, warn = TRUE,encoding = "UTF-8",
     skipNul = FALSE)
if (i %% 1000 ==0) { print(i)}
if (i==length(files)){print(Sys.time())}
}

preparation<-data.frame(content=character(),stringsAsFactors=FALSE)
for (i in 1:length(files)) {
preparation[i,1]<-df[i,3]
preparation[(i+length(files)),1]<-df[i,2]
if (i %% 1000 ==0) { print(i)}
}

docs <- VCorpus(VectorSource(preparation$content), readerControl=list(
    reader=readPlain,language="dutch",load=FALSE))

bankruptccompanies<-c("dsb bank","schoenenreus","imtech","estro","
    macintosh","ardenberg","spyker","halfords","siebel","polare","mexx","
    thermphos","kroymans","fortis bank","icentre","sabon","licom","etam","
    dico","meavita","oilily","impact retail","selexyz","hans textiel","
    zalco","henk ten hoor","harense smid","ruwaard van putten","oad","free
    record shop","bas group","nederlandse munt","renz","a-film", "solland
    solar", "kuyichi","v&d","la place","kpnqwest","van der moolen")
healthycompanies<-c("shell","kpmg","philips","rabobank","heineken","
    unilever","aegon","akzonobel","klm","abn amro","randstad","kpn","asml"
    ,"albert heijn","delta lloyd","senseo","tomtom","boskalis","independer
    ","vopak","etos","aalberts industries","arcadis","ing groep","wolters
```

```
    kluwer","nxp","ziggo","gall","gemalto","elseviers weekblad","becel","
    ahold delhaize","tnt express","asr nederland","hunter douglas","
    refresco","usg people","tencate","douwe egberts","calve")
combinations<-c("dsb","fortis","impact","retail","harense","smid","
    solland","solar","free","record","shop","akzo","nobel","abn","amro","
    albert","heijn","delta","lloyd","aalberts","industries","wolters","
    kluwer","ahold","delhaize","tnt","express","asr","usg","people","
    hunter","douglas","douwe","egberts","ing","group","bas")
companynames<-c(bankruptccompanies, healthycompanies,combinations)

##corpus transformations
getTransformations()
docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern = "-
    ", replacement = " ")))
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern = "\
    "", replacement = " ")))
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern = "'
    ", replacement = "")))
docs <- tm_map(docs, removeWords, stopwords("dutch"))
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs, removeWords, companynames)
docs <- tm_map(docs, stemDocument, language = "dutch") #stemming
docs2 <- tm_map(docs2, stemDocument, language = "dutch")
docs <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs2, stripWhitespace)

##TFIDF
weighted<-weightTfIdf(DocumentTermMatrix(docs2), normalize = TRUE)
#smaller matrices
MaxSparsity<-0.997
weighted2<-removeSparseTerms(weighted, MaxSparsity)
weighted2 <- data.frame(as.matrix(weighted2), stringsAsFactors=FALSE)

matrix1<-weighted2[1:41223,]
```

```
matrix2<-weighted2[41224:82446,]
matrix3<-matrix1+2*matrix2
matrixdataframe<-matrix3

##create data frame to store article data
df <- data.frame(date=character(),title=character(),content=character(),
    stringsAsFactors=FALSE)

setwd("C:/Users/jasivicek/Desktop/text mining and R practice/articles")
file.names <- list.files(path = "C:/Users/jasivicek/Desktop/text mining
    and R practice/articles")
files <- as.character(list.files(path="C:/Users/jasivicek/Desktop/text
    mining and R practice/articles"))

##load articles into the data frame
for (i in 1:length(files)) {
if (i==1) {print(Sys.time())}
df[i,]<-readLines(files[i], 3, ok = TRUE, warn = TRUE, encoding = "UTF-8",
    skipNul = FALSE)
if (i %% 1000 == 0) { print(i)}
if (i==length(files)){print(Sys.time())}
}

docs <- VCorpus(VectorSource(df$content), readerControl=list(reader=
    readPlain,language="dutch",load=FALSE))
docs

bankruptccompanies<-c("dsb bank","schoenenreus","imtech","estro","
    macintosh","ardenberg","spyker","halfords","siebel","polare","mexx","
    thermphos","kroymans","fortis bank","icentre","sabon","licom","etam","
    dico","meavita","oilily","impact retail","selexyz","hans textiel","
    zalco","henk ten hoor","harense smid","ruwaard van putten","oad","free
    record shop","bas group","nederlandse munt","renz","a-film","solland
    solar","kuyichi","v&d","la place","kpnqwest","van der moolen")
healthycompanies<-c("shell","kpmg","philips","rabobank","heineken","
    unilever","aegon","akzonobel","klm","abn amro","randstad","kpn","asml
    ","albert heijn","delta lloyd","senseo","tomtom","boskalis","
    independer","vopak","etos","aalberts industries","arcadis","ing groep
```

```
    ","wolters kluwer","nxp","ziggo","gall","gemalto","elseviers weekblad
    ","becel","ahold delhaize","tnt express","asr nederland","hunter
    douglas","refresco","usg people","tencate","douwe egberts","calve")
combinations<-c("dsb","fortis","impact","retail","harense","smid","
    solland","solar","free","record","shop","akzo","nobel","abn","amro","
    albert","heijn","delta","lloyd","aalberts","industries","wolters","
    kluwer","ahold","delhaize","tnt","express","asr","usg","people","
    hunter","douglas","douwe","egberts","ing","group","bas")
companynames<-c(bankruptccompanies, healthycompanies, combinations)

docs <- tm_map(docs, content_transformer(tolower))
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern =
    "-", replacement = " ")))
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern =
    "\"", replacement = " ")))
docs <- tm_map(docs, stripWhitespace)
docs <- tm_map(docs, content_transformer(function(x) gsub(x, pattern =" '
    ", replacement = "")))
docs <- tm_map(docs, removeWords, stopwords("dutch"))
docs <- tm_map(docs, removeNumbers)
docs <- tm_map(docs, removePunctuation)
docs <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs, removeWords, companynames)
docs <- tm_map(docs, stemDocument, language = "dutch") #stemming
docs2 <- tm_map(docs2, stemDocument, language = "dutch")
docs <- tm_map(docs, stripWhitespace)
docs2 <- tm_map(docs2, stripWhitespace)


##Sentiment Analysis
library(SentimentAnalysis)
df$sent<-0
dim(df)
for(i in 1:length(files)){
df[i,4]<-analyzeSentiment(docs[[i]]$content, language = "dutch",
    aggregate = NULL, removeStopwords = FALSE, stemming = FALSE)[2]
}
df[,4]
```

```
sentimentclankov <- data.frame(as.matrix(df[,4]), stringsAsFactors=FALSE)
fwrite(sentimentclankov, file = "C:/Users/jasivicek/Desktop/text mining
    and R practice/TFIDF/sentimentclankov.csv")

##modify metadata
for (i in 1:length(files)) {
meta(docs[[i]], type="local", tag="date") <- df[i,1]
meta(docs[[i]], type="local", tag="heading") <- df[i,2]
meta(docs[[i]], type="local", tag="name") <- file.names[i]
meta(docs[[i]], type="local", tag="datetimestamp") <- NULL
meta(docs[[i]], type="local", tag="origin") <- NULL
meta(docs[[i]], type="local", tag="description") <- NULL
meta(docs[[i]], type="local", tag="author") <- NULL
meta(docs[[i]], type="local", tag="language") <- NULL
}
#getting dates
names<-data.frame(as.matrix(file.names), stringsAsFactors=FALSE)
colnames(names) <- "meno"
namesbankr<-length(names[grep("bankr_", names[,1]),])
nameshealthy<-length(names[grep("heal_", names[,1]),])

names$zbankrotovane<-NA
names$zdrave<-NA
names$unbank<-NA
names$unheal<-NA
length(names)
for (i in 1:dim(names)[1]) {
names[i,2]<-unlist(strsplit(names[i,1], split='bankr_', fixed=TRUE))[2]
names[i,3]<-unlist(strsplit(names[i,1], split='heal_', fixed=TRUE))[2]
names[i,4]<-unlist(strsplit(names[i,2], split='0', fixed=TRUE))[1]
names[i,5]<-unlist(strsplit(names[i,3], split='0', fixed=TRUE))[1]
}

testinghealthycompanies<-unique(names[,5], incomparables = FALSE)
testinghealthycompanies<- testinghealthycompanies[!is.na(
    testinghealthycompanies)]

testingbankruptcompanies<-unique(names[,4], incomparables = FALSE)
```

```
testingbankruptcompanies<- testingbankruptcompanies[!is.na(
    testingbankruptcompanies)]

menozbankrotovanych<-names[,4]
length(menozbankrotovanych)
menabankr<-menozbankrotovanych[!is.na(menozbankrotovanych)]

menozdravych<-names[,5]
length(menozdravych)
menazdrave<-menozdravych[!is.na(menozdravych)]

d<-"00001.txt"

allcompanies<-c(testinghealthycompanies, testingbankruptcompanies)

choice<-testingbankruptcompanies
if (choice[1] %in% testinghealthycompanies == TRUE) {a <- "heal_"} else {
    a <- "bankr_"}
j<-1
ac<-0
lengthtested<-0
starttested<-0
datumy <- data.frame(matrix(0, ncol = length(choice), nrow = 1100))
colnames(datumy)<-choice
lastweek<-0
lastmonth<-0
lastquarter<-0
lasthalf<-0
lastyear<-0
for (j in 1:length(choice)){
ac[j]<-choice[j]
starttested[j]<-which(names$meno == paste(a,ac[j],d,sep=""))
lengthtested[j]<-dim(names[grep(paste(a,ac[j],sep=""), names[,1]),])[1]
for (k in starttested[j]:(starttested[j]+lengthtested[j]-1)) {
datumy[(1+k-starttested[j]),j]<-as.Date(meta(docs[[k]])$date)
}
#print(sum(lengthtested[0:(j-1)]))
for (l in starttested[j]:(starttested[j]+lengthtested[j]-1)) {
```

```
lastweek[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumy[1:lengthtested[j],j], datumy[1:lengthtested[j],j]>=(datumy[(1+
    l-starttested[j]),j]-7) & datumy[1:lengthtested[j],j] < datumy[(1+l-
    starttested[j]),j] )))/(lengthtested[j])
#poslednyrok[(1+l-starttested[j]),j]<-length(subset(datumy[1:lengthtested
    [j],j], datumy[1:lengthtested[j],j]>(datumy[(1+l-starttested[j]),j
    ]-365) & datumy[1:lengthtested[j],j] < datumy[(1+l-starttested[j]),j]
    ))
lastquarter[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(
    subset(datumy[1:lengthtested[j],j], datumy[1:lengthtested[j],j]>=(
    datumy[(1+l-starttested[j]),j]-90) & datumy[1:lengthtested[j],j] < (
    datumy[(1+l-starttested[j]),j]-30) )))/(lengthtested[j])
lastmonth[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumy[1:lengthtested[j],j], datumy[1:lengthtested[j],j]>=(datumy[(1+
    l-starttested[j]),j]-30) & datumy[1:lengthtested[j],j] < (datumy[(1+l-
    starttested[j]),j]-7) )))/(lengthtested[j])
lasthalf[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumy[1:lengthtested[j],j], datumy[1:lengthtested[j],j]>=(datumy[(1+
    l-starttested[j]),j]-180) & datumy[1:lengthtested[j],j] < (datumy[(1+l
    -starttested[j]),j]-90) )))/(lengthtested[j])
lastyear[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumy[1:lengthtested[j],j], datumy[1:lengthtested[j],j]>=(datumy[(1+
    l-starttested[j]),j]-365) & datumy[1:lengthtested[j],j] < (datumy[(1+l
    -starttested[j]),j]-180) )))/(lengthtested[j])
}
}

choice<-testingbankruptcompanies
bdates<-0
cbd<-read.csv(file="C:/Users/jasivicek/Desktop/text mining and R practice
    /TFIDF/companybankruptcydates.csv", header=TRUE, sep=",")
if (choice[1] %in% testinghealthycompanies == TRUE) {a<-"heal_"} else {
    a<-"bankr_"}
j<-1
ac<-0
lengthtested<-0
starttested<-0
colnames(datumy)<-choice
```

```
for (j in 1:length(choice)){
ac[j]<-choice[j]
starttested[j]<-which(names$meno == paste(a,ac[j],d,sep=""))
lengthtested[j]<-dim(names[grep(paste(a,ac[j],sep=""),names[,1]),])[1]
for (i in 1:(dim(cbd)[1])){
if (choice[j]==cbd[i,1]){
print("YES")
bdates[(starttested[j]:(starttested[j]+lengthtested[j]-1))]<-toString(as.
    Date(cbd[i,2]))
}
}
}

daysbeforebankr<-0
for(i in 1:length(bdates)){
daysbeforebankr[i]<-as.Date(bdates[i])-as.Date(meta(docs[[i]])$date)
}

daysbeforebankr[(namesbankr+1):length(docs)]<-NA

choice<-testinghealthycompanies
if (choice[1] %in% testinghealthycompanies == TRUE) {a <- "heal_"} else {
    a <- "bankr_"}
j<-1
ac<-0
lengthtested<-0
starttested<-0
datumyH <- data.frame(matrix(0, ncol = length(choice), nrow = 1100))
colnames(datumyH)<-choice
lastweekH<-0
lastmonthH<-0
lastquarterH<-0
lasthalfH<-0
lastyearH<-0
for (j in 1:length(choice)){
ac[j]<-choice[j]
starttested[j]<-which(names$meno == paste(a,ac[j],d,sep=""))
lengthtested[j]<-dim(names[grep(paste(a,ac[j],sep=""),names[,1]),])[1]
```

```
#print(lengthtested[j])
for (k in starttested[j]:(starttested[j]+lengthtested[j]-1)) {
datumyH[(1+k-starttested[j]),j]<-as.Date(meta(docs[[k]])$date)
}
#print(sum(lengthtested[0:(j-1)]))
for (l in starttested[j]:(starttested[j]+lengthtested[j]-1)) {
lastweekH[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumyH[1:lengthtested[j],j] , datumyH[1:lengthtested[j],j]>(datumyH
    [(1+l-starttested[j]),j]-7) & datumyH[1:lengthtested[j],j] < datumyH
    [(1+l-starttested[j]),j] )))/(lengthtested[j])
#poslednyrok[(1+l-starttested[j]),j]<-length(subset(datumy[1:lengthtested
    [j],j] , datumy[1:lengthtested[j],j]>(datumy[(1+l-starttested[j]),j
    ]-365) & datumy[1:lengthtested[j],j] < datumy[(1+l-starttested[j]),j]
    ))
lastquarterH[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(
    subset(datumyH[1:lengthtested[j],j] , datumyH[1:lengthtested[j],j]>(
    datumyH[(1+l-starttested[j]),j]-90) & datumyH[1:lengthtested[j],j] < (
    datumyH[(1+l-starttested[j]),j]-30) )))/(lengthtested[j])
lastmonthH[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset
    (datumyH[1:lengthtested[j],j] , datumyH[1:lengthtested[j],j]>(datumyH
    [(1+l-starttested[j]),j]-30) & datumyH[1:lengthtested[j],j] < (datumyH
    [(1+l-starttested[j]),j]-7) )))/(lengthtested[j])
lasthalfH[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumyH[1:lengthtested[j],j] , datumyH[1:lengthtested[j],j]>(datumyH
    [(1+l-starttested[j]),j]-180) & datumyH[1:lengthtested[j],j] < (
    datumyH[(1+l-starttested[j]),j]-90) )))/(lengthtested[j])
lastyearH[1+l-starttested[j]+sum(lengthtested[0:(j-1)])]<-(length(subset(
    datumyH[1:lengthtested[j],j] , datumyH[1:lengthtested[j],j]>(datumyH
    [(1+l-starttested[j]),j]-365) & datumyH[1:lengthtested[j],j] < (
    datumyH[(1+l-starttested[j]),j]-180) )))/(lengthtested[j])
}
}

length(lastweekH)+length(lastweek)
lastweek[(namesbankr+1):(dim(names)[1])]<-lastweekH
lastmonth[(namesbankr+1):(dim(names)[1])]<-lastmonthH
lastquarter[(namesbankr+1):(dim(names)[1])]<-lastquarterH
lasthalf[(namesbankr+1):(dim(names)[1])]<-lasthalfH
```

```
lastyear[(namesbankr+1):(dim(names)[1])]<-lastyearH

#saving the article dates
articledates <- data.frame(date=character(),stringsAsFactors=FALSE)
for (i in 1:length(files)){
articledates[i,1]<-meta(docs[[i]])$date
}
articledates <- data.frame(as.matrix(articledates[,1]), stringsAsFactors=
    FALSE)
fwrite(articledates, file = "C:/Users/jasivicek/Desktop/text mining and R
     practice/TFIDF/articledates.csv")

lastweek <- data.frame(lastweek, stringsAsFactors=FALSE)
lastmonth <- data.frame(lastmonth, stringsAsFactors=FALSE)
lastquarter <- data.frame(lastquarter, stringsAsFactors=FALSE)
lasthalf <- data.frame(lasthalf, stringsAsFactors=FALSE)
lastyear <- data.frame(lastyear, stringsAsFactors=FALSE)

fwrite(lastweek, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/TFIDF/lastweek.csv")
fwrite(lastmonth, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/TFIDF/lastmonth.csv")
fwrite(lastquarter, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/TFIDF/lastquarter.csv")
fwrite(lasthalf, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/TFIDF/lasthalf.csv")
fwrite(lastyear, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/TFIDF/lastyear.csv")

##Adding new columns to TF-IDF
sentimentclankov<-read.csv(file="C:/Users/jasivicek/Desktop/text mining
    and R practice/TFIDF/sentimentclankov.csv", header=TRUE, sep=",")
sentimentclankov<-data.frame(as.matrix(sentimentclankov),
    stringsAsFactors=FALSE)
sentimentclankovrounded<-0
for(i in 1:41223){
if(sentimentclankov[i]==0){
sentimentclankovrounded[i]<-0
```

```
} else if(sentimentclankov[i]>0) {
sentimentclankovrounded[i]<-1
} else {
sentimentclankovrounded[i]<--1
}
}

menaspolocnosti<-0
menaspolocnosti[1:length(menabankr)]<-menabankr
menaspolocnosti[(length(menabankr)+1):41223]<-menazdrave
sentimentclankovrounded<- data.frame(as.matrix(sentimentclankovrounded), 
    stringsAsFactors=FALSE)
matrixdataframe$articledate<-articledates[,1]
matrixdataframe$companynames<-menaspolocnosti
matrixdataframe$daysbeforebankr<-daysbeforebankr
matrixdataframe$SENT<-sentimentclankov[,1]
matrixdataframe$SENTROUNDED<-sentimentclankovrounded[,1]
matrixdataframe$LASTWEEK<-lastweek
matrixdataframe$LASTMONTH<-lastmonth
matrixdataframe$LASTQUARTER<-lastquarter
matrixdataframe$LASTHALF<-lasthalf
matrixdataframe$LASTYEAR<-lastyear

##Assign 1 and 0 to new column 'event'
matrixdataframe$event<-0
matrixdataframe$event <- ifelse(matrixdataframe$event == "beforebankrupt"
    , 1, 0)
matrixdataframe$event <- factor(matrixdataframe$event , levels = c(0, 1))
#assign 1 to articles about bankrupted companies
matrixdataframe[1:namesbankr, length(matrixdataframe)]<-1
table(matrixdataframe$event)
write.csv(matrixdataframe, file ="C:/Users/jasivicek/Desktop/text mining
    and R practice/TFIDF/TFIDFmatrix.csv")
```

Script 3: R script for training logistic regression and neural network models

```
matrixdataframe<-read.csv(file="C:/Users/jasivicek/Desktop/text mining 
    and R practice/TFIDF/TFIDFminimini.csv", header=TRUE, sep=",")
```

```
matrixdataframe$X<-NULL
positionad<-which(colnames(matrixdataframe)=="articledate")

#random sorting
randomB<-sample.int(40, 40, replace = FALSE)
randomH<-sample.int(40, 40, replace = FALSE)
length(unique(randomB))
unique(randomB)
indexB <- data.frame(matrix(NA, ncol = 5, nrow = 8))
indexH <- data.frame(matrix(NA, ncol = 5, nrow = 8))

##LOGISTIC REGRESSION
#K-fold cross-validation
bankruptcyscore<-0
indexes<-0
logitmod<-data.frame(matrix(NA, ncol = 8, nrow = -3+length(
    matrixdataframe)))
preddf <- data.frame(matrix(NA, ncol = 8, nrow = 700))
preddfdf <- data.frame(matrix(NA, ncol = 16, nrow = 700))
ac<-data.frame(matrix(NA, ncol = 10, nrow = 8))
lengthtested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
starttested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
endtested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
for (i in 1:8){
 indexB[i,1:5]<-randomB[(5*i-4):(5*i)]
 indexH[i,1:5]<-randomH[(5*i-4):(5*i)]
 choiceB<-testingbankruptcompanies[c(indexB[i,1], indexB[i,2], indexB[i
     ,3], indexB[i,4], indexB[i,5])]
 choiceH<-testinghealthycompanies[c(indexH[i,1], indexH[i,2], indexH[i
     ,3], indexH[i,4], indexH[i,5])]
 choice<-c(choiceB,choiceH)
 print(choice)
 for (j in 1:length(choice)) {
   if (choice[j] %in% testinghealthycompanies == TRUE) {a <- "heal_"}
       else {a <- "bankr_"}
   #print(j)
   ac[i,j]<-choice[j]
   #print(ac[j])
```

```
  starttested[i,j]<-which(names$meno == paste(a,ac[i,j],d,sep=""))
  lengthtested[i,j]<-dim(names[grep(paste(a,ac[i,j],sep=""), names[,1]),
      ])[1]
  endtested[i,j]<-(starttested[i,j]+lengthtested[i,j]-1)
}
indexes = c(starttested[i,1]:(starttested[i,1]+lengthtested[i,1]-1),
    starttested[i,2]:(starttested[i,2]+lengthtested[i,2]-1), starttested[
    i,3]:(starttested[i,3]+lengthtested[i,3]-1), starttested[i,4]:(
    starttested[i,4]+lengthtested[i,4]-1), starttested[i,5]:(starttested[
    i,5]+lengthtested[i,5]-1), starttested[i,6]:(starttested[i,6]+
    lengthtested[i,6]-1), starttested[i,7]:(starttested[i,7]+lengthtested
    [i,7]-1), starttested[i,8]:(starttested[i,8]+lengthtested[i,8]-1),
    starttested[i,9]:(starttested[i,9]+lengthtested[i,9]-1), starttested[
    i,10]:(starttested[i,10]+lengthtested[i,10]-1))
#print(indexes)
print(i)
test = matrixdataframe[indexes,]
train = matrixdataframe[-indexes,]
#print(dim(train))
#print(dim(test))
#print(sum(dim(test)[1]+dim(train)[1]))
train2<-filter(train, daysbeforebankr > 60 | is.na(daysbeforebankr))
test2<-filter(test, daysbeforebankr > 60 | is.na(daysbeforebankr))
#print(dim(train2))
#print(dim(test2))
train2$daysbeforebankr<-NULL
test2$daysbeforebankr<-NULL
train3<-train2
test3<-test2
#print(dim(test3))
train4<-train3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
test4<-test3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
#print(dim(test4))
#print(colnames(train4)[(-10+length(train4)):length(train4)])
remove(train,test,train2,test2,train3)
gc()
logit <- glm(event ~ ., family = "binomial", data=train4, control = list
    (epsilon=0.1, maxit = 20, trace=TRUE))
```

```
 logitmod[,i]<-logit$coefficients
 preddf[1:(dim(test4)[1]),i] <- predict(logit, newdata = test4, type = "
     response")
 remove(logit,train4)
 preddfdf[1:(dim(test4)[1]),i+8] <- as.character(test3$articledate)
 preddfdf[1:(dim(test4)[1]),i] <- as.character(test3$companynames)
 remove(test3,test4)
}

logitmoddf<-data.frame(as.matrix(logitmod), stringsAsFactors=FALSE)
preddf<- data.frame(as.matrix(preddf), stringsAsFactors=FALSE)
preddfdf<- data.frame(as.matrix(preddfdf), stringsAsFactors=FALSE)

write.csv(logitmoddf, file = "C:/Users/jasivicek/Desktop/text mining and
    R practice/results/LR10CVeps0.1coef.csv")
write.csv(round(preddf,digits=4), file = "C:/Users/jasivicek/Desktop/text
    mining and R practice/results/LR10CVeps0.1pred.csv")
write.csv(preddfdf, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/results/LR10CVeps0.1dates.csv")

##Leave-one-out cross-validation
choice<-testinghealthycompanies
#choice<-testingbankruptcompanies
if (choice[1] %in% testinghealthycompanies == TRUE) {a <- "heal_"} else {
    a <- "bankr_"}
j<-1
bankruptcyscore<-0
ac<-0
lengthtested<-0
starttested<-0
logitmod<-data.frame(matrix(NA, ncol = length(choice), nrow = -4+length(
    matrixdataframe)))
preddf <- data.frame(matrix(NA, ncol = 2*length(choice), nrow = 700))
preddfdf <- data.frame(matrix(NA, ncol = 40, nrow = 700))
colnames(preddf) <- c(choice,choice)
colnames(logitmod) <- choice
for (j in 1:length(choice)) {
  print(j)
```

```
  ac[j]<-choice[j]
  print(ac[j])
  starttested[j]<-which(names$meno == paste(a,ac[j],d,sep=""))
  #print(starttested[j])
  lengthtested[j]<-dim(names[grep(paste(a,ac[j],sep=""), names[,1]), ])
      [1]
  #print(lengthtested[j])
  train<-matrixdataframe[c(1:(starttested[j]-1),(starttested[j]+
      lengthtested[j]):(dim(matrixdataframe)[1])),]
  test<-matrixdataframe[(starttested[j]:(starttested[j]+lengthtested[j
      ]-1)),]
  preddf[1:(dim(test)[1]),(j+length(choice))] <- as.character(test[,
      positionad])
  remove(train,test)
  train2<-matrixdataframe[c(1:(starttested[j]-1),(starttested[j]+
      lengthtested[j]):(dim(matrixdataframe)[1])),c(1:(positionad-1),(
      positionad+1):(dim(matrixdataframe)[2]))]
  test2<-matrixdataframe[(starttested[j]:(starttested[j]+lengthtested[j
      ]-1)),c(1:(positionad-1),(positionad+1):(dim(matrixdataframe)[2]))]
  train3<-filter(train2, daysbeforebankr > 60 | is.na(daysbeforebankr))
  test3<-filter(test2, daysbeforebankr > 60 | is.na(daysbeforebankr))
  remove(train2,test2)
  print(paste("test matrix without 60 days",dim(test3)[1],dim(test3)[2]))
  train3$daysbeforebankr<-NULL
  test3$daysbeforebankr<-NULL
  train4<-train3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
  test4<-test3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
  remove(train3,test3)
  print(paste("test matrix without column",dim(test4)[1],dim(test4)[2]))
  logit <- glm(event ~ ., family = "binomial", data=train4, control =
      list(epsilon=0.1, maxit = 20, trace=TRUE))
  logitmod[,j]<-logit$coefficients
  preddf[1:(dim(test4)[1]),j] <- predict(logit, newdata = test4, type = "
      response")
  #preddfdf[1:(dim(test4)[1]),i] <- as.character(test3$companynames)
  #preddfdf[1:(dim(test4)[1]),i] <- as.character(test3$articledate)
  remove(train3,test3,train4,logit)
}
```

```
logitmoddf<-data.frame(as.matrix(logitmod), stringsAsFactors=FALSE)
preddf<- data.frame(as.matrix(preddf), stringsAsFactors=FALSE)

if (choice[1] %in% testinghealthycompanies == TRUE) {
  write.csv(logitmoddf, file = "C:/Users/jasivicek/Desktop/text mining
      and R practice/results/LRLOOCVeps0.1coefHEALTHY.csv")
  write.csv(preddf, file = "C:/Users/jasivicek/Desktop/text mining and R
      practice/results/LRLOOCVeps0.1predHEALTHY.csv")
  } else {
  write.csv(logitmoddf, file = "C:/Users/jasivicek/Desktop/text mining
      and R practice/results/LRLOOCVeps0.1coefBANKRUPT.csv")
  write.csv(preddf, file = "C:/Users/jasivicek/Desktop/text mining and R
      practice/results/LRLOOCVeps0.1predBANKRUPT.csv")
}

##Neural networks
neuraldf<-matrixdataframe
neuraldf[,length(neuraldf)] <- as.numeric(as.character(neuraldf[,length(
    neuraldf)]))

feats <- c(names(neuraldf)[1:(positionad-1)],names(neuraldf)[(positionad
    +3):(-1+length(names(neuraldf)))])
f <- paste(feats,collapse=' + ')
f <- paste('event ~ ',f)
f <- as.formula(f)
f

#K-fold cross-validation
bankruptcyscore<-0
indexes<-0
logitmod<-data.frame(matrix(NA, ncol = 8, nrow = -3+length(
    matrixdataframe)))
prednn <- data.frame(matrix(NA, ncol = 8, nrow = 700))
prednndf <- data.frame(matrix(NA, ncol = 16, nrow = 700))
ac<-data.frame(matrix(NA, ncol = 10, nrow = 8))
lengthtested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
starttested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
endtested<-data.frame(matrix(NA, ncol = 10, nrow = 8))
```

```
for (i in 1:8){
  indexB[i,1:5]<-randomB[(5*i-4):(5*i)]
  indexH[i,1:5]<-randomH[(5*i-4):(5*i)]
  choiceB<-testingbankruptcompanies[c(indexB[i,1], indexB[i,2], indexB[i
      ,3], indexB[i,4], indexB[i,5])]
  choiceH<-testinghealthycompanies[c(indexH[i,1], indexH[i,2], indexH[i
      ,3], indexH[i,4], indexH[i,5])]
  choice<-c(choiceB,choiceH)
  for (j in 1:length(choice)) {
    if (choice[j] %in% testinghealthycompanies == TRUE) {a <- "heal_"}
       else {a <- "bankr_"}
    #print(j)
    ac[i,j]<-choice[j]
    #print(ac[j])
    starttested[i,j]<-which(names$meno == paste(a,ac[i,j],d,sep=""))
    lengthtested[i,j]<-dim(names[grep(paste(a,ac[i,j],sep=""), names[,1])
        , ])[1]
    endtested[i,j]<-(starttested[i,j]+lengthtested[i,j]-1)
  }
  indexes = c(starttested[i,1]:(starttested[i,1]+lengthtested[i,1]-1),
      starttested[i,2]:(starttested[i,2]+lengthtested[i,2]-1), starttested
      [i,3]:(starttested[i,3]+lengthtested[i,3]-1), starttested[i,4]:(
      starttested[i,4]+lengthtested[i,4]-1), starttested[i,5]:(starttested
      [i,5]+lengthtested[i,5]-1), starttested[i,6]:(starttested[i,6]+
      lengthtested[i,6]-1), starttested[i,7]:(starttested[i,7]+
      lengthtested[i,7]-1), starttested[i,8]:(starttested[i,8]+
      lengthtested[i,8]-1), starttested[i,9]:(starttested[i,9]+
      lengthtested[i,9]-1), starttested[i,10]:(starttested[i,10]+
      lengthtested[i,10]-1))
  #print(indexes)
  print(i)
  test = matrixdataframe[indexes,]
  train = matrixdataframe[-indexes,]
  print(dim(train))
  print(dim(test))
  train2<-filter(train, daysbeforebankr > 60 | is.na(daysbeforebankr))
  test2<-filter(test, daysbeforebankr > 60 | is.na(daysbeforebankr))
  print(dim(train2))
```

```
  print(dim(test2))
  train2$daysbeforebankr<-NULL
  test2$daysbeforebankr<-NULL
  train3<-train2
  test3<-test2
  print(dim(test3))
  train4<-train3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
  test4<-test3[,c(1:(positionad-1),(positionad+2):(dim(train3)[2]))]
  print(dim(test4))
  remove(train,test,train2,test2,train3)
  gc()
  nn <- neuralnet(f,train4,hidden=c(50),linear.output=FALSE, threshold =
      0.1,lifesign="full", lifesign.step=500)
  prednncomp <- compute(nn,test4[,1:(dim(test4)[2]-1)])
  remove(nn,train4)
  prednn[1:(dim(test4)[1]),i] <- prednncomp$net.result
  prednndf[1:(dim(test4)[1]),i+8] <- as.character(test3$articledate)
  prednndf[1:(dim(test4)[1]),i] <- as.character(test3$companynames)
  remove(test3,test4)
}
prednn<-data.frame(as.matrix(prednn), stringsAsFactors=FALSE)
prednndf<- data.frame(as.matrix(prednndf), stringsAsFactors=FALSE)

write.csv(round(prednn,digits=4), file = "C:/Users/jasivicek/Desktop/text
    mining and R practice/results/NN10CVc(50)eps0.1pred.csv")
write.csv(prednndf, file = "C:/Users/jasivicek/Desktop/text mining and R
    practice/results/NN10CVc(50)eps0.1dates.csv")

#Leave-one-out cross-validation
choice<-testinghealthycompanies
#choice<-testingbankruptcompanies
if (choice[1] %in% testinghealthycompanies == TRUE) {a <- "heal_"} else {
    a <- "bankr_"}
j<-1
bankruptcyscorenn<-0
ac<-0
lengthtested<-0
starttested<-0
```

```
prednn <- data.frame(matrix(NA, ncol = 2*length(choice), nrow = 1100))
colnames(prednn) <- c(choice,choice)
for (j in 1:length(choice)) {
  print(j)
  ac[j]<-choice[j]
  print(ac[j])
  starttested[j]<-which(names$meno == paste(a,ac[j],d,sep=""))
  lengthtested[j]<-dim(names[grep(paste(a,ac[j],sep=""), names[,1]), ])
      [1]
  train<-neuraldf[c(1:(starttested[j]-1),(starttested[j]+lengthtested[j])
      :(dim(neuraldf)[1])),]
  test<-neuraldf[(starttested[j]:(starttested[j]+lengthtested[j]-1)),]
  train<-filter(train, daysbeforebankr > 60 | is.na(daysbeforebankr))
  test<-filter(test, daysbeforebankr > 60 | is.na(daysbeforebankr))
  train$daysbeforebankr<-NULL
  test$daysbeforebankr<-NULL
  train2<-train
  test2<-test
  train2$articledate<-NULL
  test2$articledate<-NULL
  remove(train)
  print("NN is going to be trained")
  nn <- neuralnet(f,train2,hidden=c(50),linear.output=FALSE, threshold =
      0.1,lifesign="full", lifesign.step=500)
  prednncomp <- compute(nn,test2[,1:(dim(test2)[2]-1)])
  prednn[1:(dim(test2)[1]),j] <- prednncomp$net.result
  prednn[1:(dim(test2)[1]),(j+length(choice))] <- as.character(test$
      articledate)
  remove(prednncomp)
}

prednn<- data.frame(as.matrix(prednn), stringsAsFactors=FALSE)
if (choice[1] %in% testinghealthycompanies == TRUE) {
  fwrite(prednn, file = "C:/Users/jasivicek/Desktop/text mining and R
      practice/results/NNdfHEALTHY.csv")
  } else {
  fwrite(prednn, file = "C:/Users/jasivicek/Desktop/text mining and R
      practice/results/NNdfBANKRUPT.csv")
```

```
}
```

Script 4: R script for evaluation of the predictions

```
lpredictions<-read.csv(file="C:/Users/jasivicek/Desktop/text mining and R
    practice/results/mini/NN10CVc(50)eps0.1pred.csv", header=TRUE, sep=",
    ")
lpredictions2<-read.csv(file="C:/Users/jasivicek/Desktop/text mining and
    R practice/results/mini/NN10CVc(50)eps0.1dates.csv", header=TRUE, sep=
    ",")
#lpredictions<-read.csv(file="C:/Users/jasivicek/Desktop/text mining and
    R practice/results/mini/LR10CVeps0.1pred.csv", header=TRUE, sep=",")
#lpredictions2<-read.csv(file="C:/Users/jasivicek/Desktop/text mining and
     R practice/results/mini/LR10CVeps0.1dates.csv", header=TRUE, sep=",")
predictions<-data.frame(matrix(NA, ncol = 160, nrow = 700))
colnames(predictions)<-c(testingbankruptcompanies,testinghealthycompanies
    ,testingbankruptcompanies,testinghealthycompanies)
lpredictions2$X<-NULL
lpredictions$X<-NULL

for (i in 1:80){
  x<-which(lpredictions2 == colnames(predictions)[i],arr.ind = TRUE)
  y<-x
  y[,2]<-x[,2]+8
  predictions[1:length(lpredictions[x]),i]<-lpredictions[x]
  predictions[1:length(lpredictions2[y]),i+80]<-lpredictions2[y]
}

realvalues<-data.frame(matrix(NA, ncol = 80, nrow = 9000))
for(i in 1:40){
  realvalues[(1:length(predictions[!is.na(predictions[,i]),i])),i]<-1
  realvalues[(1:length(predictions[!is.na(predictions[,i+40]),i+40])),i
      +40]<-0
}

sucet<-0
x<-round(predictions[,1:80],0)
for(i in 1:80){
```

```
  dlzka<-length(which(x[,i]==0))
  sucet<-sucet+dlzka
}

##Confusion Matrix from article predictions
truepositive<-0
truenegative<-0
falsenegative<-0
falsepositive<-0
for(i in 1:80){
  for(j in 1:length(predictions[!is.na(predictions[,i]),i])){
    if(x[j,i]==1 & realvalues[j,i]==1) {
      truepositive<-truepositive+1
    }
    if(x[j,i]==0 & realvalues[j,i]==0) {
      truenegative<-truenegative+1
    }
    if(x[j,i]==0 & realvalues[j,i]==1) {
      falsenegative<-falsenegative+1
    }
    if(x[j,i]==1 & realvalues[j,i]==0) {
      falsepositive<-falsepositive+1
    }
  }
}
confmat<-data.frame(matrix(NA, ncol = 2, nrow = 2))
confmat[1,1]<-truepositive
confmat[2,2]<-truenegative
confmat[1,2]<-falsenegative
confmat[2,1]<-falsepositive
confmat
(truepositive+truenegative)/(truepositive+truenegative+falsepositive+
    falsenegative)

statistics<-data.frame(matrix(NA, ncol = 80, nrow = 9))
colnames(statistics)<-c(testingbankruptcompanies,testinghealthycompanies)
rownames(statistics)<-c("average","median","class","timediff","preddiff",
    "stdev","dailyavg","weeklyavg","event")
```

```
for (i in 1:80){
  statistics[1,i]<-mean(predictions[!is.na(predictions[,i]),i])
  statistics[2,i]<-median(predictions[!is.na(predictions[,i]),i])
  statistics[3,i]<-(length(which(predictions[!is.na(predictions[,i]),i
      ]>=0.5)))/(length(predictions[!is.na(predictions[,i]),i]))
  statistics[4,i]<-as.numeric(as.Date(max(predictions[!is.na(predictions
      [,i+80]),i+80]))-as.Date(min(predictions[!is.na(predictions[,i+80]),
      i+80])))
  statistics[5,i]<-max(predictions[!is.na(predictions[,i]),i])-min(
      predictions[!is.na(predictions[,i]),i])
  statistics[6,i]<-sd(predictions[!is.na(predictions[,i]),i])
  dff<-data.frame(predictions[!is.na(predictions[,i]),i],predictions[!is.
      na(predictions[,i+80]),i+80])
  smalldate<-min(as.Date(dff[,2]))
  for(j in 1:dim(dff)[1]){
    dff$daily[j]<-1+difftime(as.Date(dff[j,2]), smalldate, units = c("
        days"))
    dff$dailyavg[j]<-(dff$daily[j]/(sum(dff$daily)))*dff[j,1]
    dff$weekly[j]<-1+difftime(as.Date(dff[j,2]), smalldate, units = c("
        weeks"))
    dff$weeklyavg[j]<-(dff$weekly[j]/(sum(dff$weekly)))*dff[j,1]
  }
  statistics[7,i]<-sum(dff$dailyavg)
  statistics[8,i]<-sum(dff$weeklyavg)
  statistics[9,1:40]<-1
  statistics[9,41:80]<-0
}

statistics<-t(as.matrix(statistics))
statistics<-as.data.frame(statistics)

##Confusion matrix from simple averages
cutoff<-0.35
truepos<-length(which(statistics$average[1:40]>=cutoff))
falseneg<-length(which(statistics$average[1:40]<cutoff))
trueneg<-length(which(statistics$average[41:80]<cutoff))
falsepos<-length(which(statistics$average[41:80]>=cutoff))
```

```
confmat1<-data.frame(matrix(NA, ncol = 2, nrow = 2))
confmat1[1,1]<-truepos
confmat1[2,2]<-trueneg
confmat1[1,2]<-falseneg
confmat1[2,1]<-falsepos
confmat1

sensitivity<-0
##loop for sensitivity
for(i in 1:51) {
  cutoff<-0.5-((i-1)/100)
  truepos<-length(which(statistics$average[1:40]>=cutoff))
  sensitivity[i]<-truepos/40
}

plot(seq(0.5,0,-0.01),sensitivity)
x<-seq(0.5,0,-0.01)
sen<-x[which(sensitivity>=0.75)]

specificity<-0
##loop for specificity
for(i in 1:51) {
  cutoff<-0.5-((i-1)/100)
  trueneg<-length(which(statistics$average[41:80]<cutoff))
  specificity[i]<-trueneg/40
}

plot(seq(0.5,0,-0.01),specificity)
spe<-x[which(specificity>=0.75)]

plot(seq(0.5,0,-0.01),sensitivity,xlab="decision threshold value",ylab="
    sensitivity/specificity value",ylim=c(0,1),col="blue")
par(new=TRUE)
plot(seq(0.5,0,-0.01),specificity,xlab="decision threshold value",ylab="
    sensitivity/specificity value",ylim=c(0,1),col="green")
lines(seq(0.5,0,-0.01),sensitivity,col="blue")
lines(seq(0.5,0,-0.01),specificity,col="green")
```

```
abline(h=0.75,col="red")
legend(0.02, 0.6, legend=c("specificity", "sensitivity"),col=c("green", "
    blue"), lty=1, cex=1.2)

accuracy<-0
##loop for accuracy
for(i in 1:51) {
  cutoff<-(i/100)-0.01
  truepos<-length(which(statistics$average[1:40]>=cutoff))
  trueneg<-length(which(statistics$average[41:80]<cutoff))
  accuracy[i]<-(truepos+trueneg)/80
}
plot(seq(0,0.5,0.01),accuracy,ylim=c(0,1),xlab="decision threshold value"
    )
lines(seq(0,0.5,0.01),accuracy)
abline(v=0.3,col="red")
abline(v=0.39,col="red")
(truepos+trueneg)/(truepos+trueneg+falsepos+falseneg)

##ROC curves
d<-as.vector(as.matrix(realvalues))
d <- d[!is.na(d)]
lr<-as.vector(as.matrix(predictions[,1:80]))
lr <- lr[!is.na(lr)]

library(pROC)
rocobj <- roc(d, lr)
plot(rocobj,col="blue", lwd=2, main="ROC curve")
auc(rocobj)

##Comparison of types of averages
b<-as.vector(as.matrix(statistics[,9]))
b <- b[!is.na(b)]
c<-as.vector(as.matrix(statistics[,1]))
c <- c[!is.na(c)]
d<-as.vector(as.matrix(statistics[,7]))
d <- d[!is.na(d)]
e<-as.vector(as.matrix(statistics[,8]))
```

```
e <- e[!is.na(e)]

rocobj1 <- roc(b, c)
rocobj2 <- roc(b, d)
rocobj3 <- roc(b, e)

ggroc(list("LR-simple" = rocobj1, "LR-daily-weighted" = rocobj2,"LR-
    weekly-weighted"=rocobj3))
```