Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Department of Mathematics and Physics
Master of Data Science

Unsupervised text classification: a contractual risk detection approach

THESIS to obtain the **DEGREE** of
MASTER IN DATA SCIENCE

A thesis presented by: **Omar Antonio Villalobos Ramos**

Thesis Advisor: **Dr. Esteban Jiménez Rodríguez**

Tlaquepaque, Jalisco, November, 2020

Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Departamento de Matemáticas y Física
Maestría en Ciencia de Datos

Unsupervised text classification: a contractual risk detection approach

Tesis para obtener el **Grado** de
MAESTRÍA EN CIENCIA DE DATOS

Tesis presentada por: **Omar Antonio Villalobos Ramos**

Asesor de Tesis: **Dr. Esteban Jiménez Rodríguez**

Tlaquepaque, Jalisco, November, 2020

Instituto Tecnológico y de Estudios Superiores de Occidente

Reconocimiento de validez oficial de estudios de nivel superior según acuerdo secretarial 15018, publicado en el Diario Oficial de la Federación del 29 de noviembre de 1976.

Department of Mathematics and Physics

Master of Data Science Approval Form

Thesis Title: **Unsupervised text classification: a contractual risk detection approach**

Author: **Omar Antonio Villalobos Ramos**

Thesis Approved to complete all degree requirements for the Master of Science Degree in Data Science.

Thesis Advisor, **Dr. Esteban Jiménez Rodríguez**

Thesis Reader, **Dr. Saúl Alonso Nuño Sánchez**

Thesis Reader, **Dr. Juan Diego Sánchez Torres**

Academic Advisor, **M. Juan Carlos Martínez Alvarado**

Tlaquepaque, Jalisco, November, 2020

Dedicated to my Family.

Unsupervised text classification: a contractual risk detection approach

Abstract

Enterprise contracting process tends to be tedious when there is thousands of active contracts to manage. The aim of this work was to implement an automatic indexing and information retrieval method in order to classify the semantic structure within contract documents into two classes, risk and non-risk legal language, on the basis of terms contained in new documents further called queries. The technique implemented is term frequency as the transformation procedure for each of the documents and singular-value decomposition to represent such transformations into a set of optimized number of factors. Queries are analyzed as vectors formed from the linear combination of the terms and compared to known documents class with cosine values to determine the nature of the legal language (as risk or non-risk). The result of this work shows that the class detection is possible using the proposed methodology with high relative percentage of accuracy.

Keywords: Natural language processing, contracting, classification.

Contents

Page

List of Figures

Page

List of Tables

Page

1 Introduction

1.1 Business problem

The company, IBM, has a contract management system that provides several solutions; among them, there is one that stores original documents/contracts along with some metadata. However, these documents are often in the form of scanned files, preventing a machine to directly "read" the plain text from them. The solution to this particular problem was to use a tool to "translate" the scanned files to machine readable plain texts; this tool is called an Optical Character Recognition (OCR).

Different from number-related objects, plain texts do not have an underlying structure beyond how they were originally written. In this sense, the models to encode and preprocess plain texts must be capable of interpreting the subjacent semantic structure. Hence, at the output of the OCR system, an entity detection tool classifies short pieces of texts according to some predefined categories such as names, legal entities, place, quantity expressions, among others.

Finally, the outcome from the above process is a set of enriched phrases (pieces of text) per document.

The problem, and the matter of this document, is to identify if each of the enriched phrases is related or not to certain topics, which are based on the business input. In particular, the topic that IBM is interested in is **Contractual risk**, and the categories that are to be identified are:

- Contractual risk related to the topic based on the business input.

- No contractual risk in particular.

It is worth to mention that this classification is being carried out manually by lawyers, who are experts in the matter and manage to label 210 phrases building the data set utilize in this thesis.

1.2 State of the art and motivation

Contracting is a common activity in today's global marketplace. It is so common, that firms often struggle to manage it because of the large number, the great diversity, and significant complexity of contracts that are operated at the same time with both, local and international parties [1].

Traditionally, all the contracting issues are handled by a team of specialized lawyers, who draft, execute, and improve not only the contracts themselves but also the contracting processes and the agreements that these contracts govern. This means that the operation of contracts requires hundreds of hours of specialized manpower, which traduces in high costs and does not guarantee the absence of errors. In fact, it has been estimated that firms lose between 5% to 40% of the value on a given deal, due to errors in drafting contracts [2].

Recent technological developments in the areas of data science and artificial intelligence allow to come up with solutions that help companies to overcome the challenge of handling a nonuniform large number of contracts. For instance, study[3] how relevant features in contracts can be automatically extracted using linear classifiers such as logistic regression and support vector machines, and a deep learning approach; schemes for automatic segmentation and tagging of contracts are developed in[4]; the study conducted in [5] propose an automatic recognition algorithm of requisite and affectation parts in legal documents using and comparing several neural network schemes. However, there is a lack of research studies concerning contractual risk detection, although it has been identified as a relevant application of artificial intelligence in contracting [6].

The risk detection problem can be identified as a binary classification problem, being the positive class fragments of contracts which involve risk in some predefined sense, and being the negative class fragments of contracts that do not involve risk. Several binary classification algorithms such as Logistic Regression[7], Support Vector Machines[8], Random Forest [9], among others have been well studied and, to some extent, one can consider that binary classification a well-developed area that has solved numerous applications showing great success. One key ingredient behind this success is that the behavior in unknown domains can be accurately estimated by quantitatively learning the pattern from sufficient training examples[10]. Because of the technical difficulties mentioned before, in particular, the need for a great amount of time of specialized manpower to perform risk analyses over a set of contracts, the training data set for the risk detection problem is small. This undesired property negatively impacts the performance of the conventional binary classification algorithms because:

[1] B. Rich. How ai is changing contracts. https://hbr.org/amp/2018/02/how-ai-is-changing-contracts, February 2018

[2] KPMG. Supply chain capacity management – the key to value. https://home.kpmg/au/en/home/insights/2017/03/supply-chain-capacity-management.html, March 2017

[3] I. Androutsopoulos I. Chalkidis and A. Michos. Extracting contract elements. *International Conference on Artificial Intelligence and Law*, (2):19–28, 1 2017; and I. Chalkidis and I. Androutsopoulo. A deep learning approach to contract element extraction. *30th International Conference on Legal Knowledge and Information Systems*, (1):155–164, 1 2017

[4] J. Parapar I. Hasan and R. Blanco. In proc. of the 19th int. conf. on database and expert systems application. In *Segmentation of legislative documents using a domain-specific lexicon*, 19, pages 665–669, Turin, Italy, 6 2008; and E. L. Mencia. Artificial intelligence and law. In *Segmentation of legal documents*, 12, pages 88–97, Barcenola, Spain, 6 2009

[5] Nguyen L. Nguyen T. and Tojo S. et al. Recurrent neural network-based models for recognizing requisite and effectuation parts in legal texts. *Artif Intell Law*, (26): 169–199, 2018

[6] B. Rich. How ai is changing contracts. https://hbr.org/amp/2018/02/how-ai-is-changing-contracts, February 2018

[7] T. P. Ryan. *Modern regression methods*. 1997

[8] I. Steinwart and A. Christmann. Support vector machines. *Springer Publishing Company, Incorporated*, (1), 2008

[9] L. Breiman. Machine learning. https://doi.org/10.1023/A:1010933404324, January 2001

[10] Y. Zhang and C Ling. A strategy to apply machine learning to small datasets in materials science. https://doi.org/10.1038/s41524-018-0081-z, January 2018

(i) The algorithms tend to over fit the small training data.

(ii) Numerical optimization algorithms may not converge.

(iii) The effect of the outliers is amplified.

(iv) Among others.

1.3 Objectives

In this sense, the main objective of this thesis is to address the contractual risk detection problem using the latent semantic analysis method.

This main objective involves the following specific objectives:

1. To carry out an exploratory data analysis of the data set of contracts.
2. To train different binary classifier structures for risk detection over the data set contracts.
3. To develop a latent semantic model for risk detection.
4. To propose a metric to evaluate the considered risk detection schemes.
5. To select the best scheme according to the proposed metric.

1.4 Document organization

The rest of the thesis is organized as follows. Next Section 2 shows the pre-analysis and data exploration of the *corpus*. Section 3 exposes the mathematically preliminaries along with the proposed risk detection solution. Finally, performance classification tables and results are in Section 4, and main conclusions are discussed in Section 5.

2 Data Description and Exploratory Data Analysis

In this Section the corpus is analyzed with a information retrieval tool called Term Frequency analysis and then those results being processed with a graphics engine in order to visualize it.

2.1 Problem description

Plain text itself is not an analyzable data unit, therefore it needs to be interpreted in order to be used as an input for a statistical model. In this sense, it has been transformed such that words within the phrases were weighed regarding their frequency of appearance in the phrases and documents.

Even do the selection of the transformation method is an on-going discussion within Natural Language Processing (NLP) community of scientists, mathematicians and practitioners in general, the standard transformation method is the Term Frequency - Inverse Document Frequency (TF-IDF). This is a method that measures the relevance of a word in a document within a collection of documents; it is composed by two main calculations: *(i)* the frequency of a given word in a document, and *(ii)* the inverse document frequency is the logarithm of the number of documents in the collection by the number of documents containing that given word.

These two calculations weighs, words by words, in each of the documents of the collection providing a fair transformation of the text into vectors of weights. TF tends to give a bigger weight to high frequency words like ‘the’ since it is counting the appearances of the words within a document, but IDF intends to offset that weight by penalizing the weight of the high frequency words and giving more weight to words with low appearance through the different documents.

2.2 Data description

The data-set consists of 210 clauses from already signed commercial contracts for which an expert lawyer already had reviewed and classified each of them correctly within two classes risk and no-risk. (see table 2.1).

name	description
text	Clause contract paragraphs.
class	Binary risk classification. labels: ['no-risk', 'risk']

Table 2.1: Metadata table

2.2.1 Evaluation data

The data was splitted into two sets, *training* (70%) and *testing* (30%), both created form the original data-set with a random sample keeping a balance of the classes in each of the splits.

	risk	no-risk	total phrases
training	73	73	146
testing	32	32	64

Table 2.2: Train and test split for evaluation purposes

2.3 Text Mining

Unlike a conventional data descriptive analysis, one cannot use the same type of methods for text. Nonetheless the goal is similar. We may ask:

- What is the frequency of words' usage?
- What are the most important ones?
- How do we measure the importance?

These questions drive us to convert the unstructured dataset into a structured form in order to make the data handling easier. This conversion can be made with a Tidy structure implemented in R [1], which is a reliable and easy to implement tool. Furthermore, since it also provides a solution for TF-IDF, we will use it for the rest of the analysis. When in comes to natural language we use words with a certain distribution in such we use a set of words with more frequency than others and for contracts is not the exception. We can see this behavior in the Figure 2.1, where we can observe that a small set of words is less frequently used than other words.

The frequencies of the x-axis in Figure 2.1 show how often words are being used in this collection of documents. On the other hand, we want to see what words are these and how frequent are they used in

[1] J. Silge and D. Robinson. Text mining with r. https://www.tidytextmining.com/tfidf.html, January 2020

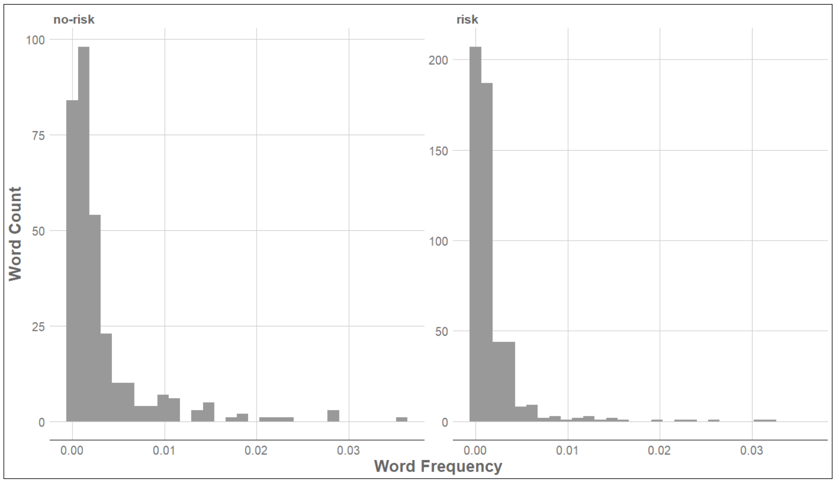

Figure 2.1: Shows the Frequency of unique words per risk class.

each document. For this goal we analyzed the documents with TF-IDF. Such analysis determines the relevance of each of the words given the appearance in each document.

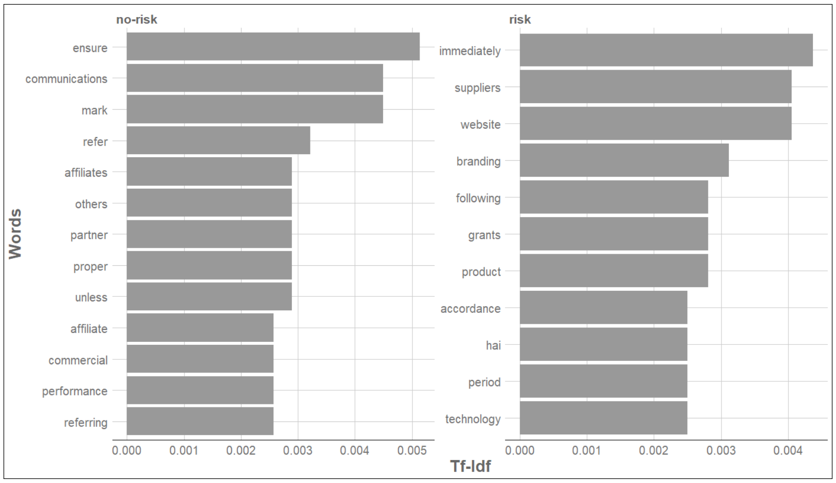

Figure 2.2: Tf-Idf calculation over contract collection.

The graph in Figure 2.2 helps to identify the most important words for each of the classes. Some of those words are company names which make sense given the pre-defined risks, also another verbs and businesses-related words such as ‘tradename’, ‘claim’ and ‘gpj’ are good indicatives of the context of the text. A similar analysis but with two adjacent words are the Bi-grams. Figure 3 shows the most important bi-grams on this data set.

Just like word importance, bi-gram also provides insight of the relevance of how words appear in this collection. Although the differences between some of the bi- grams in Figure 2.3 are not obvious

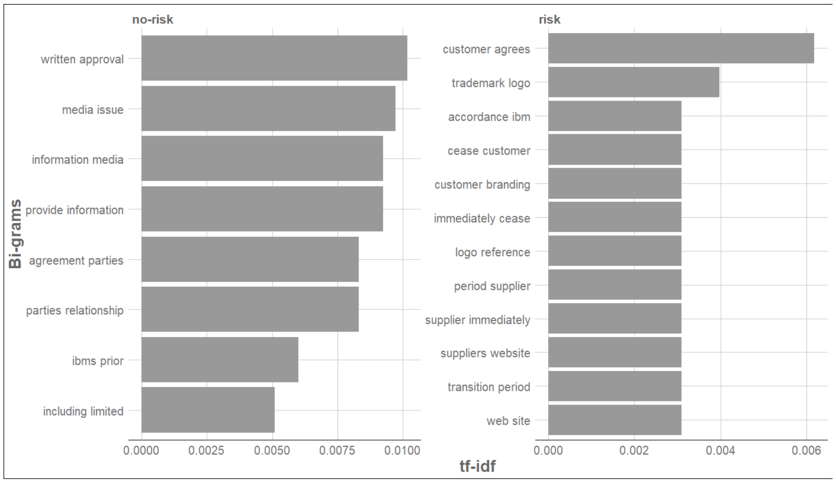

Figure 2.3: Bi-grams by TF-IDF importance.

we still can conclude that some pair of words are quite important when they come together, for instance the pairs "trademark tradename" and "copyright logo" are very good examples of it, since this collection of documents is publicity related topic then one can say this bi-grams are relevant for the semantic meaning.

2.4 Section conclusions

It can be concluded that TF-IFD is a fair method to represent our unstructured data into a vector like structure for further analysis, since it captures the relevance of the words usage along the collection of documents.

3 Latent Semantic Classifier

This Section consists in three topics, the mathematical backgrounds 3.1; definitions and theorems, the proposed methodology 3.2 to detect risk in the contracts and a Python code 3.3 example implementation.

3.1 Mathematical preliminaries

3.1.1 Basic definitions and notation

We begin defining the core objects for all the developments in this thesis.

Definition 1 (Term)**.** A **term** t is simply a word.

Definition 2 (Document)**.** A **document** d is some text, i.e., a collection of terms. It may be a phrase, a paragraph, or a complete writing.

Definition 3 (Corpus)**.** A **corpus** is a collection of documents.

Throughout this document, the following notation is prevalent:

(i) $\mathcal{D} = \{d_1, \ldots, d_n\}$ is the set of all the documents. Moreover, $n = |\mathcal{D}|$ is the number of total documents.

(ii) $\mathcal{T}_d = \left\{t_1^d, \ldots, t_{m_d}^d\right\}$ is the set of terms belonging to the document d. Moreover, $m_d = |\mathcal{T}_d|$ is the number of terms in document d.

(iii) $\mathcal{T} = \bigcup\limits_{d \in \mathcal{D}} \mathcal{T}_d = \{t_1, \ldots, t_m\}$ is the set of all the terms in all the documents. Moreover, $m = |\mathcal{T}|$ is the number of total terms. This set will be often referred to as the **vocabulary**, since it contains all the terms.

3.1.2 Term frequency - inverse document frequency

The central problem of analyzing natural language is how to measure the meaning of a given document. Often, that is achievable by assigning importance or weight to each of the words in the document.

A very powerful and widely used tool for weighting terms in different documents was proposed and discussed in Spärck et al.

(1961)[1]. This proposed weighting method balances the relation of terms in a certain collection of documents considering both:

(i) The exhaustivity of a document, defined as the number of terms it contains.

(ii) The specificity of a term, defined as the number of documents where it appears.

Measuring the exhaustivity of the documents and the specificity of the terms allow to define a weighting factor for each term in each document that balances the number of times that the term appears in the document with the frequency of the term in the whole set of documents. Formally:

Definition 4 (Term Frequency)**.** Given a document $d \in \mathcal{D}$ and a term $t \in \mathcal{T}$, we define the term frequency, denoted by $tf(t,d)$, as the number of times that the term t shows up in the document d divided by the total number of terms in the document.

Definition 5 (Inverse Document Frequency)**.** Given a document $d \in \mathcal{D}$ and a term $t \in \mathcal{T}$, we define the inverse document frequency, denoted by $idf(t,d)$, as

$$idf(t,d) = \log\left(\frac{n}{\sum_{i=1}^{n} I(t \in \mathcal{T}_{d_i})}\right),$$

where $I(\cdot)$ stands for the indicator function and n the number of documents. Hence, the denominator $\sum_{i=1}^{n} I(t \in \mathcal{T}_{d_i})$ is equal to the number of documents where the term t appears.

Definition 6 (Term Frequency - Inverse Document Frequency)**.** Given a document $d \in \mathcal{D}$ and a term $t \in \mathcal{T}$, we define the term frequency - inverse document frequency (tf-idf), denoted by $tf - idf(t,d)$, as

$$tf - idf(t,d) = tf(t,d)idf(t,d).$$

Although the TF-IDF execution can be somewhat complex, there already exist several robust implementations incorporating solutions to some practical issues. One of these implementations comes in the Python's scikit-learn library under the *Tfidf-vectorizer*[2] wrapper function, which executes the tf-idf over a corpus and returns a term-document matrix $A \in \mathbb{R}^{m \times n}$, whose entry i, j, $A_{i,j} = tf - idf(t_i, d_j)$, is the tf-idf representation of the term $t_i \in \mathcal{T}$ in the document $d_j \in \mathcal{D}$.

Remark 1. In general, the matrix $A \in \mathbb{R}^{m \times n}$ described above will be a *tall matrix*, i.e. $m > n$, since there will be more terms than documents.

[1] Spärck Karen. A statistical interpretation of term specificity and its application in retrieval. (1):11–21, 1 1972

[2] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011

3.1.3 Latent semantic indexing

In Subsection 3.1.2, we studied a methodology to represent a corpus into a matrix with real entries. This approach already gives us a mathematical representation of a corpus in the term-document matrix $A \in \mathbb{R}^{m \times n}$; naturally, one associates the rows of the matrix A to the terms, and the columns of the matrix A to the documents. However, these representations of terms and documents have two major drawbacks:

(a) In this form, these representations of terms and documents are **over-fitted** to the sample corpus they were obtained from.

(b) It is easy to realize that the mathematical representation of any of the terms is a n-dimensional vector, whereas the mathematical representation of any of the documents is a m-dimensional representation. Hence, it is not possible to establish comparisons of a term with a document, which is a desired feature, for instance, in information retrieval engine applications.

The **latent semantic indexing (LSI)**[3] helps to overcome these drawbacks, making use of the well-known singular-value decomposition (SVD) factorization.

The following theorems and remarks formalize the concepts and some results around the SVD and their relation with the latent semantic indexing.

Theorem 1 (Existence of the SVD). [4] *Let $A \in \mathbb{R}^{m \times n}$, with $m > n$, be a matrix of rank $k \in \mathbb{N}$. There exist a matrix $U \in \mathbb{R}^{m \times k}$ with orthonormal columns ($U^T U = I_k$, with $I_k \in \mathbb{R}^{k \times k}$ the identity matrix), a matrix $V \in \mathbb{R}^{n \times k}$ with orthonormal columns ($V^T V = I_k$), and a diagonal matrix $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$, such that:*

$$A = U \Sigma V^T = \sum_{i=1}^{k} \sigma_i u_i v_i^T,$$

where u_i, v_i are the i-th columns of U and V, respectively.

Remark 2 (Solution to drawback *(b)*). Using Theorem 1, we can always decompose the term-document matrix into a terms matrix $T_0 \in \mathbb{R}^{m \times k}$, a documents matrix $D_0 \in \mathbb{R}^{n \times k}$, and the singular values matrix $\Sigma_0 = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$, with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k > 0$, as

$$A = T_0 \Sigma_0 D_0^T. \tag{3.1}$$

In this setting, the rows of T_0 constitute k-dimensional representations of each term, and similarly, the rows of D_0 are k-dimensional representations of each one of the documents. Hence,

[3] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407, 1990

[4] A. Jepson and F. Flores-Mangas. The singular value decomposition. http://www.cs.toronto.edu/~jepson/csc420/notes/introSVD.pdf, 2011

this solves the drawback *(b)* mentioned before. On the other hand, these representations are still overfitting the sample corpus.

Theorem 2 (Optimal low-rank approximation)**.** [5] *Let* $A \in \mathbb{R}^{m \times n}$*, with* $m > n$*, be a matrix of rank* $k \in \mathbb{N}$*, and consider the SVD* $A = U\Sigma V^T = \sum_{i=1}^{k} \sigma_i u_i v_i^T$ *described in Theorem 1. Then, the solution to the optimization problem*

$$\begin{array}{cc} \min_{\hat{A} \in \mathbb{R}^{m \times n}} & ||A - \hat{A}||_2 \\ \text{such that} & \text{rank}(\hat{A}) \leq p \end{array}$$

is $A = U_p \Sigma_p V_p = \sum_{i=1}^{p} \sigma_i u_i v_i^T$*, where* $U_p \in \mathbb{R}^{m \times p}$ *is the matrix formed by the first* p *columns of* U*,* $V_p \in \mathbb{R}^{n \times p}$ *is the matrix formed by the first* p *columns of* V *and* $\Sigma_p = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_p) \in \mathbb{R}^{p \times p}$*.*

Remark 3 (Solution to drawback *(a)*)**.** Now, from (3.1) and applying Theorem 2 we obtain the best least squares fit and low rank approximation of the term-document matrix A, $\hat{A}$ as:

$$A \approx \hat{A} = T\Sigma D^T,$$

where the matrices $T \in \mathbb{R}^{m \times p}$, $\Sigma \in \mathbb{R}^{p \times p}$, and $D \in \mathbb{R}^{n \times p}$ are the truncated versions of the matrices T_0, Σ_0, and D_0, respectively.

This operation helps us deal with the drawback *(b)* mentioned before, since in the approximation the corpus sampling errors and unimportant details are thrown away. This, of course, heavily depends on the selection of the hyper parameter p.

Remark 4. [On the selection of p] The low rank p should be large enough to capture all the important semantic structure of our data (contracts), and small enough to avoid fitting unimportant details for information retrieval.

In an ideal setting, there is an evident difference between the magnitude of the top singular values of the term-document matrix A with the rest. In this case, the intuitive choice is to set p equal to the number of "big values". In a less ideal setting, which is the most common situation, this difference is not that evident and the selection of p is not direct. A rule of thumb is to choose p such that the sum of the top p singular values is at least x% (70%, 80%, 90%) of the total sum of all the singular values.

Now, we have a methodology to represent both, terms and documents, in a reduced p-dimensional space. This methodology is possible because of the existence of the SVD (Theorem 1) and the optimal low-rank approximation (Theorem 2). The unique piece that is left for a complete information retrieval engine is the capability of comparing any pair of terms, documents and queries. These comparisons can be carried out with the (cosine of the) angle between the vector representation of the corresponding items.

[5] S. Boyd and S. Lall. Singular value decomposition. http://ee263.stanford.edu/lectures/svd-v2.pdf, August 2015

Definition 7. Let $x, y \in \mathbb{R}^p$ be two vectors. Then, we define the angle between x and y as the angle $-\pi \leq \theta < \pi$ that satisfies

$$\cos(\theta) = \frac{x^T y}{||x||\,||y||}.$$

A well-known result of the SVD is that although the singular values matrix is unique, the other matrices may not be. However, the non-uniqueness that these matrices are subject to is very special, and it is described rigorously in the following theorem:

Theorem 3 (Uniqueness of the SVD)**.** *Let $A \in \mathbb{R}^{m \times n}$, with $m > n$, be a matrix of rank $k \in \mathbb{N}$, and consider the SVD $A = U\Sigma V^T$ described in Theorem 1:*

(a) The singular values $\sigma_1, \sigma_2, \ldots, \sigma_k$ are unique and, for distinct positive singular values, the corresponding columns of U and V are also unique up to a change of signs of both columns.

(b) For any repeated singular values, the corresponding columns of U and V are unique up to any rotation/reflection applied to both sets of columns. This is, if $\sigma_i = \sigma_{i+1}$ are two repeated singular values, then the columns $[u_i, u_{i+1}] \to [u_i, u_{i+1}]\,W$ and the columns $[v_i, v_{i+1}] \to [v_i, v_{i+1}]\,W$ may suffer rotations/reflections according to some orthogonal (rotation) matrix W.

Remark 5 (Non-uniqueness of SVD does not affect LSI)**.** Let $x^T = [x_1, \ldots, x_j, x_{j+1}, \ldots, x_p]$ and $y^T = [y_1, \ldots, y_j, y_{j+1}, \ldots, y_p]$ be the row-vector LSI representations of some terms, documents, a term and a document, or a document and a term.

On the other hand assume that the singular values j and $j+1$ are repeated, i.e. $\sigma_j = \sigma_{j+1}$. Then, by Theorem 3, the row vector representations x and y may not be unique. Let $\bar{x}^T = [x_1, \ldots, [x_j, x_{j+1}]W, \ldots, x_p]$ and $\bar{y}^T = [y_1, \ldots, [y_j, y_{j+1}]W, \ldots, y_p]$ be the alternative representations to x and y, where W is an orthogonal matrix.

First of all, note that ($WW^T = I_2$, since W is orthogonal):

$$\begin{aligned} \bar{x}^T\bar{y} &= \sum_{i=1; i\neq j,j+1}^{p} x_i y_i + [x_j, x_{j+1}]WW^T[y_j, y_{j+1}]^T \\ &= \sum_{i=1; i\neq j,j+1}^{p} x_i y_i + [x_j, x_{j+1}][y_j, y_{j+1}]^T \\ &= \sum_{i=1}^{p} x_i y_i \\ &= x^T y. \end{aligned}$$

Similarly, following the above steps with $\bar{x}$ in quality of $\bar{y}$, we would have obtained $||\bar{x}|| = ||x||$.

Hence, the non-uniqueness exhibited by the SVD factorization (see Theorem 3) does not affect the angle between the representations (see Definition 7).

3.1.4 Querying the LSI representation

We showed how to obtain a representation of both, the terms and the documents, of an original indexed corpus of documents. However, this is not the purpose by itself, but to be able to compute the comparison of a new document with the current ones.

Let $q \in \mathbb{R}^m$ be the TF-IDF representation of the new document (or query). Assuming that the LSI representation is a correct model for this document also, we have that $q = T\Sigma d_q$. Thus, the representation of the query in the latent semantic space is given by

$$d_q = \Sigma^{-1} T^T q \in \mathbb{R}^p. \tag{3.2}$$

This d_q is just like a row of the documents matrix D, and can be used for comparison with the terms or other documents.

3.2 Latent semantic classifier

In this section we describe the complete procedure to carry out the Latent semantic classifier.

1. Get the corpus of the phrases (documents) corresponding to the n contracts' pieces that have been previously tagged with risk label.

2. Build the term-document matrix $A \in \mathbb{R}^{m \times n}$ out of the corpus, via the *tf-idf vectorizer* (m is the length of the vocabulary).

3. Perform the SVD decomposition of the term-document matrix $A \in \mathbb{R}^{m \times n}$ into the terms matrix $T_0 \in \mathbb{R}^{m \times k}$, the documents matrix $D_0 \in \mathbb{R}^{n \times k}$, and the singular values matrix $\Sigma_0 \in \mathbb{R}^{k \times k}$ (k is the rank of the term-document matrix A).

4. Perform the dimension reduction (selection of $p \leq k$) according to the ideas mentioned in Remark 4. Applying p-reduction to matrices T_0, D_0 and Σ_0, obtain matrices $T \in \mathbb{R}^{m \times p}$ (consisting on the first p columns of T_0), $D \in \mathbb{R}^{n \times p}$ (consisting on the first p columns of D_0), $\Sigma \in \mathbb{R}^{p \times p}$ (consisting on the first p columns and first p rows of Σ_0).

5. For each of new contract phrase with unknown risk label (query):

   - Find the TF-IDF representation q for the query, restricted to the vocabulary used while building A.

   - Compute the representation of the query in the latent space, d_q, according to (3.2).

- Calculate the cosine (see Definition 7) between d_q and each of the columns of the matrix D (representation of the documents in the latent space).
- Find the document (column of D) with the highest cosine with respect to d_q, and assign the risk label tagged to document d to the query d_q.

3.3 *Python Example:*

The corpus on this example consist in 5 titles about human-computer interaction (labeled c) and four titles about graph theory (labeled m).

1. Corpus of the phrases.

```
corpus = [
    "Human machine interface for lab abc computer applications",
    "A survey of user opinion of computer system response time",
    "The EPS user interface management system",
    "System and human system engineering testing of EPS",
    "Relation of user perceived response time to error measurement",
    "The generation of random binary unordered trees",
    "The intersection graph of paths in trees",
    "Graph minors IV Widths of trees and well quasi ordering",
    "Graph minors A survey",
]
```

2. Build Term-Document matrix A out of the corpus.
   Python's Scikit Learn has a feature extraction set of tools useful for this text application. The *tf-idf vectorizer* is the function that will allow us to analyze the corpus.

   Hence the term-document matrix A is constructed and defined by passing the *corpus* to python's *tf-idf vectorizer* function, here a sample code of it:

~~~
```
A = tfidf_vectorizer(corpus)
```
~~~

And the view of term-document matrix A

~~~
```
print(A)
                   c1      c2    c3      c4      c5      m1   m2      m3
abc            0.3742  0.0000  0.00  0.0000  0.0000  0.0000  0.0  0.0000
and            0.0000  0.0000  0.00  0.3296  0.0000  0.0000  0.0  0.3029
applications   0.3742  0.0000  0.00  0.0000  0.0000  0.0000  0.0  0.0000
binary         0.0000  0.0000  0.00  0.0000  0.0000  0.4324  0.0  0.0000
computer       0.3161  0.3444  0.00  0.0000  0.0000  0.0000  0.0  0.0000
engineering    0.0000  0.0000  0.00  0.3902  0.0000  0.0000  0.0  0.0000
eps            0.0000  0.0000  0.42  0.3296  0.0000  0.0000  0.0  0.0000
error          0.0000  0.0000  0.00  0.0000  0.3717  0.0000  0.0  0.0000
for            0.3742  0.0000  0.00  0.0000  0.0000  0.0000  0.0  0.0000
~~~

~~~
generation    0.0000  0.0000  0.00  0.0000  0.0000  0.4324  0.0  0.0000
```
~~~

3. SVD decomposition of the matrix A.

   Python comes with a handy solution for SVD, scipy contains a function called 'svd()' that decomposes matrices using 'eigenvector decomposition analysis'.

~~~
```
from scipy.linalg import svd
T, Sigma, DT = svd(A)
```
~~~

   Here the size of matrices U, Σ, V^T, output:

~~~
```
(41, 9) = (41, 41) (41, 9) (9, 9)
```
~~~

   Sigma matrix's diagonal contains d singular values. Note that matrix Σ values are ordered.

~~~
```
print(Sigma)
[[1.3915 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 1.1785 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 1.0683 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 1.0376 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.9600 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.8697 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.8346 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.7879 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.6793]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
 ...         ...        ...        ...        ...        ...
 [0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000 0.0000]
```
~~~

4. Dimension reduction.
   In Python is easy to implement with scipy function over matrices. k was 3 in this example and it can be observe that A_{hat} is a good approximation of A considering the rank reduction.

~~~
```
k = 2
Sigma = Sigma[:k, :k]
DT = DT[:k, :]
d = A.T.dot(T[:,:k].dot(pinv(Sigma)))
```
~~~

5. Query latent space representation and cosine similarity between each document in d and d_q.

~~~
```
print(d)
           0         1
c1 -0.153240 -0.262979
c2 -0.483566 -0.203803
c3 -0.384404 -0.360126
c4 -0.382031 -0.309861
c5 -0.294104 -0.212375
m1 -0.266416  0.193938
m2 -0.336110  0.378004
m3 -0.317850  0.479913
m4 -0.278377  0.455722
```
~~~

~~~
```
Q = ['Human computer interaction']
Q = tfidf_vectorizer(Q)
q = Q.T.dot(T[:,:k].dot(pinv(Sigma)))
```
~~~

~~~
```
print(q)
          0         1
0 -0.142183 -0.172369
```
~~~

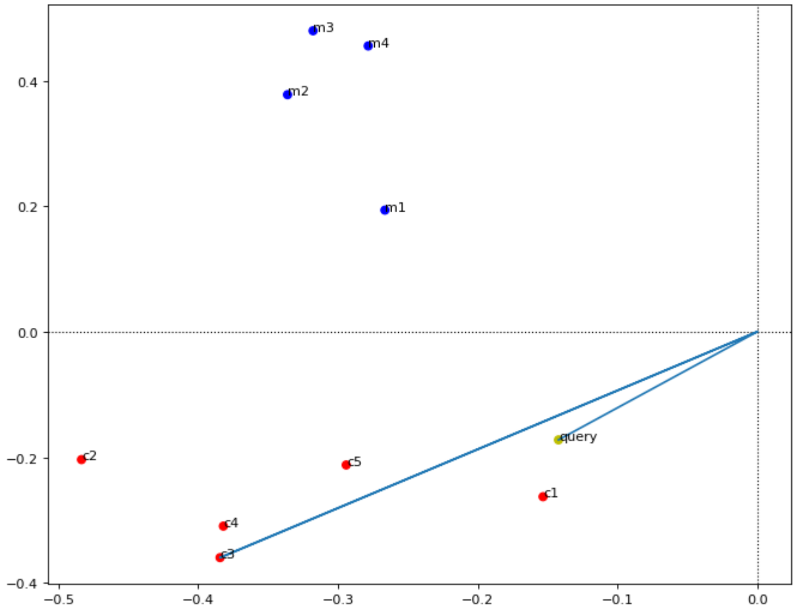

Figure 3.1: 2-D plot of document singular vectors, rows of matrix V and matrix V_q.

Figure 3.1 shows the semantic space for the sample corpus reduced to two dimensions, it can be seen that the query *Human computer interaction* is closer to those the titles with label *c* which is expected. The relationship between terms and documents is such that the rotational matrices from the SVD decomposition of the original matrix A transforms the query vector semantic space closer to the equals.

3.4 Benchmark algorithms

Accuracy for contractual risk detection is important for the business, bench-marking hence is relevant, various models where tested and compared using the same *test* data set. The information retrieval explained in (reference to cap 3.1) was the baseline for all of the bench marking models as the information retrieval method.

Models selected for bench-marking are Random Forest and Neural Network both heuristic methods briefly described in (make ref of sub chap) and both models were trained with the fixed-size vectorized texts as the *inputs* and the *objective* the vector with the risk classification.

3.4.1 Tree-based methods

This type of model partitions the data space into a set of rectangles, and then it fits a model (like a constant) in each of the rectangles. Let's consider a regression problem with a response y and inputs x_1 and x_2, it can be partitioned with parallel lines to the axes, that is, split at $x_1 = t_1$ then the region $x_1 \leq t_1$ is a split at $x_2 = t_2$ and the region $x_1 > t_3$ is a split at $x_1 = t_3$, the region $x_1 > t_3$ a split at $x_2 = t_4$. Resulting into five regions $R_1, R_2, ..., R_5$, (see Figure 3.2) such regression model predict Y with constant c_m in region R_m:

$$\hat{f}(x) = \sum_{m=1}^{5} c_m I\left\{(x_1, x_2) \in R_m\right\} \tag{3.3}$$

The problem is to choose the splitting variables and split points efficiently and for that a minimization of the sum of squares criterion $\sum(y_i - f(x_i))^2$, the best $\hat{c}_m$ is an average y_i in the region R_m:

$$\hat{c}_m = \text{ave}\left(y_i \mid x_i \in R_m\right) \tag{3.4}$$

However, finding the best partition(s) using this criterion is computationally not feasible hence we seek the splitting variable and the split point that solves:

$$\min_{j,s}\left[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2\right] \tag{3.5}$$

where R_1 and R_2 are the pair of half-planes defined as:

$$R_1(j,s) = \left\{x \mid x_j \leq s\right\} \text{ and } R_2(j,s) = \left\{x \mid x_j > s\right\} \tag{3.6}$$

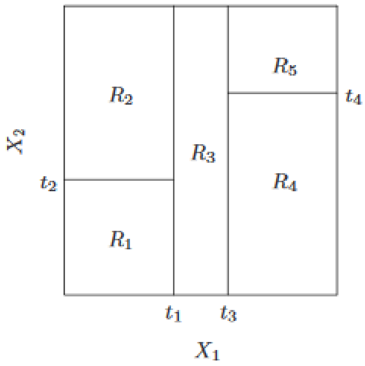

Figure 3.2: Partition of a two-dimensional feature space by **recursive binary splitting.**

Now, the intention is to fit a tree based method but the target is a classification (risk document or not), then the only changes needed in the tree algorithm is the splitting criteria. For regression was squared-error **impurity** measure. In a node m, representing a region R_m with N_m let:

$$\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I\left(y_i = k\right) \tag{3.7}$$

The proportion of class k observations in mode m. Then classify the observations in node m to class $k(m) = argmax_k \hat{p}_{mk}(m)$, the classification error $E = 1 - k(m)$ helps tree-growing however is not sufficiently.

A purity measure is the Ginni index [6] a measure of total varias across the K classes. Small values of Ginni index indicates a note that contains predominantly observations from a simple class.

$$G = \sum_{k=1}^{K} \hat{p}_{mk}\left(1 - \hat{p}_{mk}\right) \tag{3.8}$$

3.4.2 *Neural Networks*

A neural network is a two-stage regression (or classification) model, for K-class classification there are K target measurements y_k, $K = 1, ..., K$ each coded as a $0 - 1$ for the kth class.

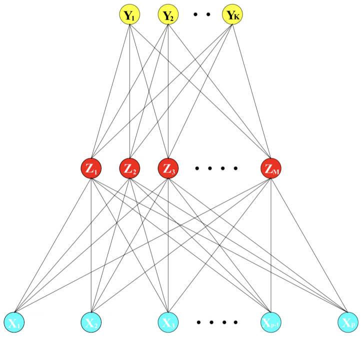

Figure 3.3: Partition of a two-dimensional feature space by **recursive binary splitting.**

Z_m are derived features created from linear combinations of the inputs and then the target y_k is modeled as a function of linear combinations of the Z_m as,

$$Z_m = \sigma\left(\alpha_{0m} + \alpha_m^T X\right), m = 1, \ldots, M \tag{3.9}$$

$$T_k = \beta_{0k} + \beta_k^T Z, k = 1, \ldots, K \tag{3.10}$$

$$f_k(X) = g_k(T), k = 1, \ldots, K \tag{3.11}$$

The activation function $\sigma(v)$ usually is the **sigmoid** where $\sigma(v) = \frac{1}{(1+e^{-v})}$. Figure 3.4 is the plot of the sigmoid function.

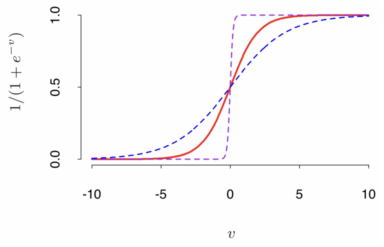

Figure 3.4: Partition of a two-dimensional feature space by **recursive binary splitting.**

Neural networks like in Figure 3.3 are often drawn with an additional bias unit feeding every unit in the hidden and output layers and captures the intercepts α_{0m} and β_{0k} in Z_m and T_k.

The output function $g_k(T)$ transforms the vector of outputs T. For K-class this function is the *softmax*:

$$g_k(T) = \frac{e^{T_k}}{\sum_{\ell=1}^{K} e^{T_\ell}} \tag{3.12}$$

[6] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. 2014. ISBN 1461471370

Fitting a Neural network we seek to estimate the unknown parameters (often called *weights* with values that fit the training data well. θ denotes all the weights which consist of,

$$\{\alpha_{0m}, \alpha_m; m = 1, 2, \ldots, M\} \quad M(p+1) \text{ weights} \tag{3.13}$$

$$\{\beta_{0k}, \beta_k; k = 1, 2, \ldots, K\} \quad K(M+1) \text{ weights.} \tag{3.14}$$

For classification we use either squared error or cross-entropy:

$$R(\theta) = -\sum_{i=1}^{N} \sum_{k=1}^{K} y_{ik} \log f_k\left(x_i\right) \tag{3.15}$$

With the softmax function and the cross-entropy error function, the neural network is exactly a linear logistic regression model in the hidden units and all parameters are estimated by maximum likelihood. The global minimizer of $R(\theta)$ is not desired instead a regularization usually by a penalty term or simply by early stopping[7].

3.5 Section conclusions

The Python example 3.3 was showed that the methodology proposed to detect risk language in the contracts, along with the mathematical backgrounds specially but not uniquely theorem 2 is feasible to implement.

[7] T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2009. ISBN 9780387848846. URL https://books.google.com.mx/books?id=eBSgoAEACAAJ

4 Results

4.1 Performance

The confusion matrix allows to visualize the performance for the proposed methodology results, the matrix displays the frequency distribution of the actual class value against the predicted class value, such table is as follows:

		Actual Class	
		Positive	Negative
Predicted Class	Positive	**True Positive (TP)**	False Negative (FN)
	Negative	False Negative (FN)	**True Negative (TN)**

This table shows the necessary information to inform the performance of a classification prediction for our risk detection problem, in our case, the models described in Chapter 3. For this purpose three metrics are relevant from this matrix, precision, recall and F-1 score.

The *precision* metric is the ratio of correct positive predictions of the total predicted positives, that is:

$$P = \frac{TP}{TP + FP}.$$

The *recall* metric, on the other hand is the true positive rate known as sensitivity.

$$R = \frac{TP}{TP + FN}.$$

Lastly, the *F-1 score* metric is the harmonic mean between the recall and the precision metrics:

$$F = \frac{P \cdot R}{P + R}.$$

The *confusion matrix* below shows the predicted values of the proposed methodology (see Section 3.2) over the *training* data and the real values.

		Actual	
		no-risk	risk
Predicted	no-risk	**26**	6
	risk	3	**29**

Table 4.1: Frequency distribution of predicted and the actual class of training data.

From Table 4.1 one can see that 26 out of 32 **no-risk** classes and 29 out of 32 **risk** classes were predicted correctly, which shows a good performance of the proposed methodology.

However, in order to have a measure to compare the performance of all of the models implemented, precision, recall and F1-score was applied. These metrics can be seen in Table 4.2 for each of the three models:

	Precision	Recall	F1-Score
Proposed methodology	0.95	0.95	0.95
Random Forest	0.86	0.86	0.86
Neural Net	0.85	0.85	0.85

Table 4.2: Frequency distribution of predicted and the actual class of training data.

4.2 *Section conclusion*

The business requires to have the lowest error possible when deciding if a contract document could potentially affect the organization negatively, so the confusion matrix properly shows the performance metrics that the business is looking for, moreover, it does provides a fair score to determine if the semantic structure defined by the factors from the SVD decomposition are good enough to classify the contracts.

5 Conclusion

5.1 Conclusion and Future Work

The method proposed in section 3.2 was successfully applied over commercial contract paragraphs to detect if the language analyzed is a potential risk for IBM or not. Alongside in section 2.3 the exploratory analysis showed that word frequencies for the contract language has a similar distribution between those risk paragraphs and non-risk, where many words occur rarely and few words occur frequently.

As proposed in benchmark algorithms (see section 3.4) two binary classifiers were implemented with the purpose to compare a heuristic approach for risk detection and to benchmark the performance of the *Latent Semantic Classifier* method. Lastly the performance metrics applied in chapter 4 and showed in table 4.2 as a percentage of accuracy are determinant to select the best methodology for the contract risk detection.

Results are encouraging, a NLP implementation requires a wide variety of skills beyond of mathematics and statistics, it depends a lot on software solution not only for pre-processing the data but also for extraction and understanding however it is proven that Information Retrieval is more than plausible. Risk detection for contracting is a continuing and changing problem that will require further research in the field of NLP however with the tools and procedures most frequently used for this type of tasks it is concluding that detecting contractual risk using a semantic information retrieval solution was successfully achieved considering the lack of data and the transformation nuisances of the text the accuracy reached by the LSI method is fair enough for the business and for the purposes of this project.

Results are promising nonetheless there is work that could improve legal language detection out of the scope of this work, first of all the size of the corpus being relative small due to the cost of manually classify the documents however an investigation for a none supervised classification of the factors derived from the SVD reduction method may prove if risk semantic detection could be done automatically thus the cost of tagging the existing contract documents should reduce.

Bibliography

S. Boyd and S. Lall. Singular value decomposition. http://ee263.stanford.edu/lectures/svd-v2.pdf, August 2015.

L. Breiman. Machine learning. https://doi.org/10.1023/A:1010933404324, January 2001.

I. Chalkidis and I. Androutsopoulo. A deep learning approach to contract element extraction. *30th International Conference on Legal Knowledge and Information Systems*, (1):155–164, 1 2017.

Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *JOURNAL OF THE AMERICAN SOCIETY FOR INFORMATION SCIENCE*, 41(6):391–407, 1990.

T. Hastie, R. Tibshirani, and J.H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer series in statistics. Springer, 2009. ISBN 9780387848846. URL https://books.google.com.mx/books?id=eBSgoAEACAAJ.

I. Androutsopoulos I. Chalkidis and A. Michos. Extracting contract elements. *International Conference on Artificial Intelligence and Law*, (2): 19–28, 1 2017.

J. Parapar I. Hasan and R. Blanco. In proc. of the 19th int. conf. on database and expert systems application. In *Segmentation of legislative documents using a domain-specific lexicon*, 19, pages 665–669, Turin, Italy, 6 2008.

Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. 2014. ISBN 1461471370.

A. Jepson and F. Flores-Mangas. The singular value decomposition. http://www.cs.toronto.edu/~jepson/csc420/notes/introSVD.pdf, 2011.

Spärck Karen. A statistical interpretation of term specificity and its application in retrieval. (1):11–21, 1 1972.

KPMG. Supply chain capacity management – the key to value. https://home.kpmg/au/en/home/insights/2017/03/supply-chain-capacity-management.html, March 2017.

E. L. Mencia. Artificial intelligence and law. In *Segmentation of legal documents*, 12, pages 88–97, Barcenola, Spain, 6 2009.

Nguyen L. Nguyen T. and Tojo S. et al. Recurrent neural network-based models for recognizing requisite and effectuation parts in legal texts. *Artif Intell Law*, (26):169–199, 2018.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

B. Rich. How ai is changing contracts. https://hbr.org/amp/2018/02/how-ai-is-changing-contracts, February 2018.

T. P. Ryan. *Modern regression methods*. 1997.

J. Silge and D. Robinson. Text mining with r. https://www.tidytextmining.com/tfidf.html, January 2020.

I. Steinwart and A. Christmann. Support vector machines. *Springer Publishing Company, Incorporated*, (1), 2008.

Y. Zhang and C Ling. A strategy to apply machine learning to small datasets in materials science. https://doi.org/10.1038/s41524-018-0081-z, January 2018.