EISTI
École Internationale des Sciences du Traitement de l'Information

www.eisti.fr

MULTIDIMENSIONAL DATA ANALYSIS FOR INTERNAL AUDIT

NLP METHODS COMPARISON FOR CLASSIFICATION AND CLUSTERING OF FINDINGS IN INTERNAL AUDIT REPORTS

GUSTAVO FLEURY SOARES
INDURAJ P. RAMAMURTHY

Oriented by:
PhD. Rachid Chelouah

Cergy / FR
August, 2020

EISTI

École Internationale des Sciences du Traitement de l'Information

www.eisti.fr

Abstract

Classify or group (clustering) the audit text documents (unstructured data) can allow use the additional information to improve the existing structured data, creating better knowledge support for the Audit Process. The Natural Language Processing is applied with classical and deep Machine Learning algorithms to process the information of Audit Reports.

Keywords: *Internal Audit*, *NLP, Classification, Clustering, Text Mining*

Summary

Figure List

1 Introduction

As defined by the [1], **Internal auditing** "*is an independent, objective assurance and consulting activity designed to add value and improve an organization's operations. It helps an organization accomplish its objectives by bringing a systematic, disciplined approach to evaluate and improve the effectiveness of risk management, control, and governance processes.*"

To achieve these objectives, the internal audit work seeks to collect all available information (internally or externally, structured or not), to analyze it and to provide useful insights. In Most organizations ranging from small to large, the internal auditing is a time consuming task mostly performed manually by analyzing numerous reports and documents with the aid of some office tools. Considering the volume of data created nowadays, it is crucial to use the data tools efficiently and use the artificial intelligence to scrutinize the high volume of data to find probable problems or frauds. Data science is widely used in the areas of finance and accounting, and it is possible to identify various related articles and research. The audit area, however, is late in its use compared to other lines of research [2]. Therefore, research in this field could be useful.

One of the first steps of an audit is planning, where, according to [1], a risk-based plan should be established to determine the priorities of the internal audit activity. The AICPA-2018 stated that the audited entity should be aware of including complex and unusual transactions, including emerging or controversial areas. As exemplified by [2], analysis of logs of information systems and non-traditional information, such as photos, videos, GPS location and Audits Reports, adds information to understand the process to be audited.

Risk-based plan development and risk management can be used classification or anomaly detection methods to identify possible unexpected relationships between entities, such as nepotism or legal incompatibility of activity [3]. Most of this methods use the structured data available.

Add value of unstructured data, combined with the analysis of structured data, could give us most efficient Risk-based plans.

One of the most time consuming process in auditing is read and analyze all unstructured data, like past audit reports or news published by the media. Even in simple audits is necessary analyze these documents through manual verification by comparing several related files. There is strong need to process the documents as whole and automate the analyze. To make this analysis feasible, you need to look for most adequate data science methods.

1.1 Structured Data

One example of structured data used by the internal audit is the accountability data. The figure 1 presents a time series of the total monthly payments of the Brazilian government between 2015 and 2018. For the year 2018, the range for the forecast value is presented based on data from 2015 to 2017.

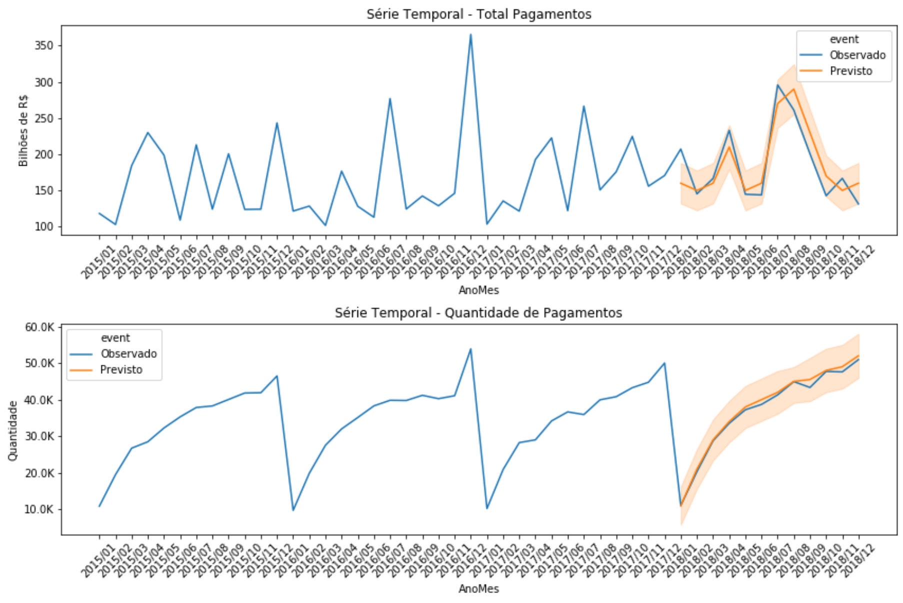


Figure 1. Structured Data – Time Series Graph

The same data set of these expenses, are presented other columns with another details, related like shown in figure 2.

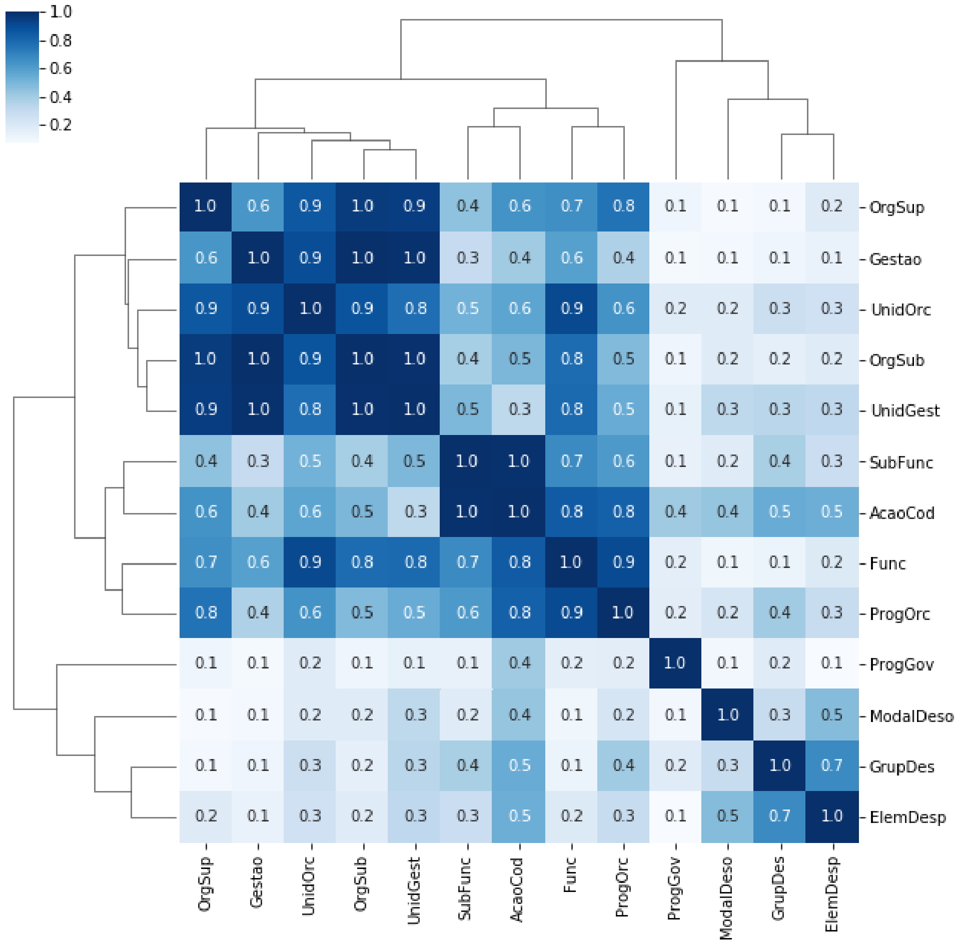
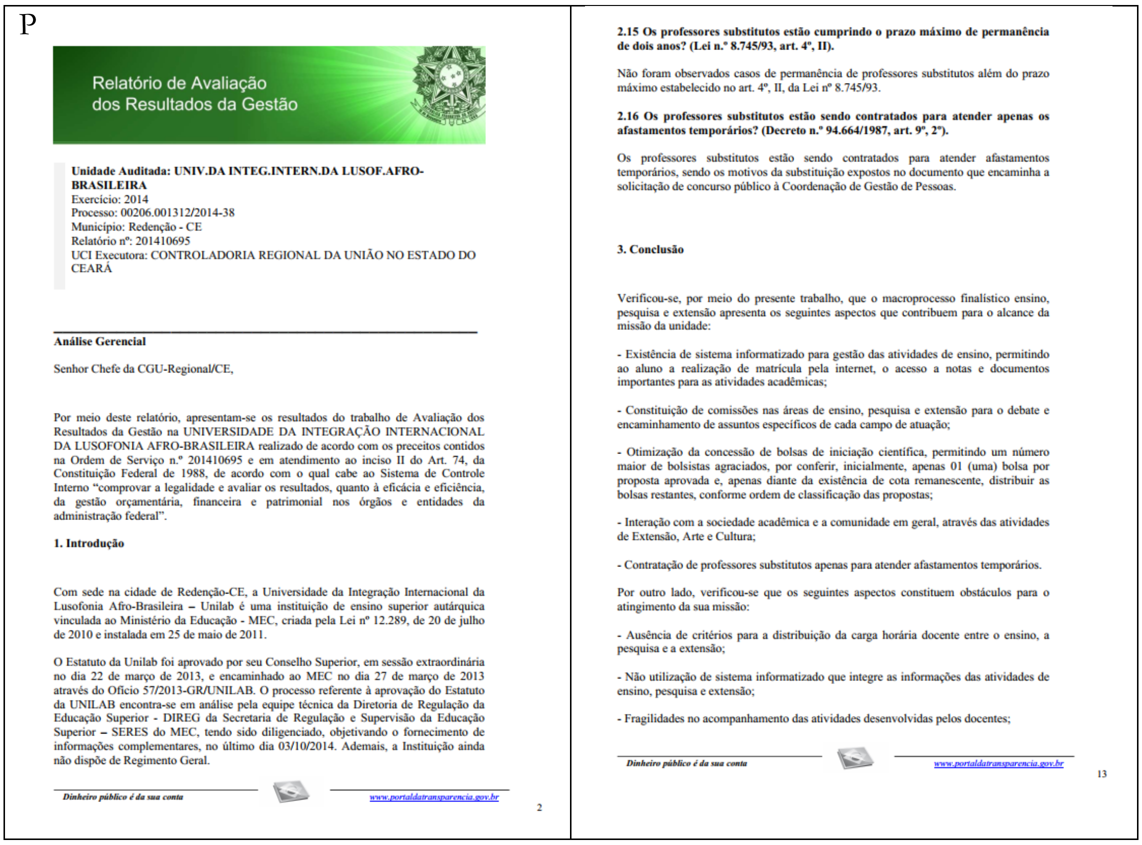


Figure 2. Structured Data – Heatmap and Dendrogram

1.2 Unstructured Data

The Internal Audits Reports of the Brazil Government are public and available in the internet in PDF format. The same way, there are a lot of information about the several sub-entities of the government that could add value.



P

Relatório de Avaliação
dos Resultados da Gestão

Unidade Auditada: UNIV.DA INTEG.INTERN.DA LUSOF.AFRO-BRASILEIRA
Exercício: 2014
Processo: 00206.001312/2014-38
Município: Redenção - CE
Relatório nº: 201410695
UCI Executora: CONTROLADORIA REGIONAL DA UNIÃO NO ESTADO DO CEARÁ

Análise Gerencial

Senhor Chefe da CGU-Regional/CE,

Por meio deste relatório, apresentam-se os resultados do trabalho de Avaliação dos Resultados da Gestão na UNIVERSIDADE DA INTEGRAÇÃO INTERNACIONAL DA LUSOFONIA AFRO-BRASILEIRA realizado de acordo com os preceitos contidos na Ordem de Serviço n.º 201410695 e em atendimento ao inciso II do Art. 74, da Constituição Federal de 1988, de acordo com o qual cabe ao Sistema de Controle Interno "comprovar a legalidade e avaliar os resultados, quanto à eficácia e eficiência, da gestão orçamentária, financeira e patrimonial nos órgãos e entidades da administração federal".

1. Introdução

Com sede na cidade de Redenção-CE, a Universidade da Integração Internacional da Lusofonia Afro-Brasileira – Unilab é uma instituição de ensino superior autárquica vinculada ao Ministério da Educação - MEC, criada pela Lei nº 12.289, de 20 de julho de 2010 e instalada em 25 de maio de 2011.

O Estatuto da Unilab foi aprovado por seu Conselho Superior, em sessão extraordinária no dia 22 de março de 2013, e encaminhado ao MEC no dia 27 de março de 2013 através do Ofício 57/2013-GR/UNILAB. O processo referente à aprovação do Estatuto da UNILAB encontra-se em análise pela equipe técnica da Diretoria de Regulação da Educação Superior - DIREG da Secretaria de Regulação e Supervisão da Educação Superior – SERES do MEC, tendo sido diligenciado, objetivando o fornecimento de informações complementares, no último dia 03/10/2014. Ademais, a Instituição ainda não dispõe de Regimento Geral.

Dinheiro público é da sua conta www.portaldatransparencia.gov.br

2

2.15 Os professores substitutos estão cumprindo o prazo máximo de permanência de dois anos? (Lei n.º 8.745/93, art. 4º, II).

Não foram observados casos de permanência de professores substitutos além do prazo máximo estabelecido no art. 4º, II, da Lei nº 8.745/93.

2.16 Os professores substitutos estão sendo contratados para atender apenas os afastamentos temporários? (Decreto n.º 94.664/1987, art. 9º, 2º).

Os professores substitutos estão sendo contratados para atender afastamentos temporários, sendo os motivos da substituição expostos no documento que encaminha a solicitação de concurso público à Coordenação de Gestão de Pessoas.

3. Conclusão

Verificou-se, por meio do presente trabalho, que o macroprocesso finalístico ensino, pesquisa e extensão apresenta os seguintes aspectos que contribuem para o alcance da missão da unidade:

- Existência de sistema informatizado para gestão das atividades de ensino, permitindo ao aluno a realização de matrícula pela internet, o acesso a notas e documentos importantes para as atividades acadêmicas;

- Constituição de comissões nas áreas de ensino, pesquisa e extensão para o debate e encaminhamento de assuntos específicos de cada campo de atuação;

- Otimização da concessão de bolsas de iniciação científica, permitindo um número maior de bolsistas agraciados, por conferir, inicialmente, apenas 01 (uma) bolsa por proposta aprovada e, apenas diante da existência de cota remanescente, distribuir as bolsas restantes, conforme ordem de classificação das propostas;

- Interação com a sociedade acadêmica e a comunidade em geral, através das atividades de Extensão, Arte e Cultura;

- Contratação de professores substitutos apenas para atender afastamentos temporários.

Por outro lado, verificou-se que os seguintes aspectos constituem obstáculos para o atingimento da sua missão:

- Ausência de critérios para a distribuição da carga horária docente entre o ensino, a pesquisa e a extensão;

- Não utilização de sistema informatizado que integre as informações das atividades de ensino, pesquisa e extensão;

- Fragilidades no acompanhamento das atividades desenvolvidas pelos docentes;

Dinheiro público é da sua conta www.portaldatransparencia.gov.br

13

Figure 3. Unstructured Data – Audit Report Example

Another examples of unstructured source of data are the social networks, the news published by the press, the inside emails of the unity, documentation and manuals of the internal processes.

The Natural Language Processing (NLP) is the segment of Data Science focused on communication between humans and computers. As presented by [4], NLP attempts to address the inherent problem that while human communications could be imprecise, computers require unambiguous and precise messages. The auditing domain normally use textual documents intended to communicate.

1.3 Methodology

For the bibliographic review, we verified the relevant literature combining the terms presents on table 1, in the singular and plural, in the languages English, French and Portuguese.

Table 1. Search Terms List

Related to "Audit"	Related to "NLP"
Audit, Internal Audit, Fraud Detection, Accounting	Natural Language Processing – NLP; Data Mining; text analysis; text mining.

Source: Created by the authors.

The sources of search were Google Scholar[1], Research Gate[2] e Science Direct[3]. We filtered the recent papers (between 2009 and 2019) and related to the topic, after the analysis of the abstract and text body. In addition, we used some academic and technical books related to the topics.

For test the methods, we used a dataset of the internal audit reports from the Brazil Government. The following chapters explain the methods of extraction and transformation from PDF to text string, clustering and classification process, including for each features extraction and data mining techniques.

1.4 NLP in Audit Reports

The Data Science works with structured (tables) and unstructured data (report texts, images, audios) and includes process like cleaning, transformation and final data analysis. For this paper we made a bibliographic review for the process of extract, cleaning, classification and clustering of audits reports (unstructured data).

In the case of fraud detection in financial annual reports by text mining, the financial annual reports are subject to rules and regulation, so all the financial annual reports follow a definitive structure led by IFRS and US GAAP. In this field of fraud deterrence using financial annual reports has been a very dominating research from the year 2000 where many big Companies were exposing themselves as bankrupt leading to huge loss of public, stack holder's money and also to big inflations. These events saw a direct effect on the research in the field of NLP-text mining. These researches were mostly based on either supervised learning algorithms or statistical analysis such as Altman's z score. The supervised learning generally used documents of fraudulent companies identified in the past by SEC (security and exchange commission) to train the machine learning model with different train to test ratios. In this area of research, many scholars have proposed many ideas all comprising of extracting certain number of financial ratios, linguistic cues from the train set to classify the test set more accurately.

1.5 Document Structure

This document is divided following the process of text analysis, starting with textual information extraction, audit findings classification and clustering. The subjects are linked together and each contains it owns introduction, literature review, methods, results and discussion. The literature review has some common items that are referenced between them.

[1] https://scholar.google.fr/
[2] https://www.researchgate.net/
[3] https://www.sciencedirect.com/

2 Extraction of Textual Information from Audit Reports

We show some solutions for scrap the information from Audit Reports available in Internet. The approach was test some tools and methods to automatic retrieve the PDF files from the audit reports, transform they in pure text and after collect the relevant information.

2.1 Literature Review

In financial area, some companies and researchers are trying to standardize financial reports, like the International Financial Reporting Standards (IFRS) and eXtensible Business Reporting Language. The objective of this standardization is facilitate the exchange of information and allow automatize the analysis [5].

For Audit Reports, exists some best practices like Generally Accepted Auditing Standards (GAAS) generally define by IFRS and report guidelines from The Institute of Internal Auditors (IIA). Unlike the financial reports, the audit reports, by its characteristics, do not have a standard format.

Despite the research on extract information from audit reports, there are a several studies in extract specific information from documents. The advance of NLP in the last years allow use this unstructured data to improve the quality of models or create new applications. Some methods applied to text mining are clustering, classification, association rule, information visualization (word cloud) [6].

The process of extract text information has been studied by the academia for several years and it is the base for several basic services available in internet, like page ranking. They can be classified in semantic understanding of text, name entity recognition and location information in larger documents [5].

Considering the source text formatted in Portable Document Format (PDF), [7] present an method based on the match patterns to layout of scientific papers (title, authors, abstract, introduction, results, methods, discussion) to extract the information. This method are very similar to the applied in this project.

2.2 Methods and Results

Like we have in published in Internet the CGU's Audit Reports[4], our approach was download them locate the information need in each report and create a table with this information. For locate this information, we create a model defining the type and the limits of each interest data. The figure above resume the steps and options tested.

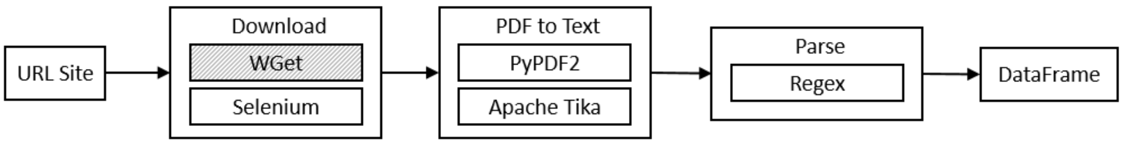


Figure 1. Automatic Extraction Steps

[4] https://auditoria.cgu.gov.br/

For download data, we used the framework for automation Selenium[5], with a headless web browser PhantomJS[6]. Like the CGU Audit Reports webpage use javascript, it was not possible to use the simple **wget** to retrieve a list of links do download the documents. We divided the process in 2 steps: download the table of links and, after using the table list like reference download the reports.

We decided to use the Reports with the 3 lines of action: "*Auditoria de Acompanhamento da Gestão*"-AAG, "*Auditoria Anual de Contas*"-AAC and "*Avaliação dos Resultados da Gestão*"-ARG.

The next step was parse the PDF files to retrieve the information need. For transform PDF for pure text format, there is several different libraries and tools. The best result in ours tests for performance and quality was with the **Tika Parser**[7] from Apache Foundation (*1c-ImportPDF_to_DF.ipynb*). The Reports are in text PDF, so it was not necessary do the OCR process.

The information needed to classify the findings of audit reports is show in table below. It was necessary parse the text to retrieve this information.

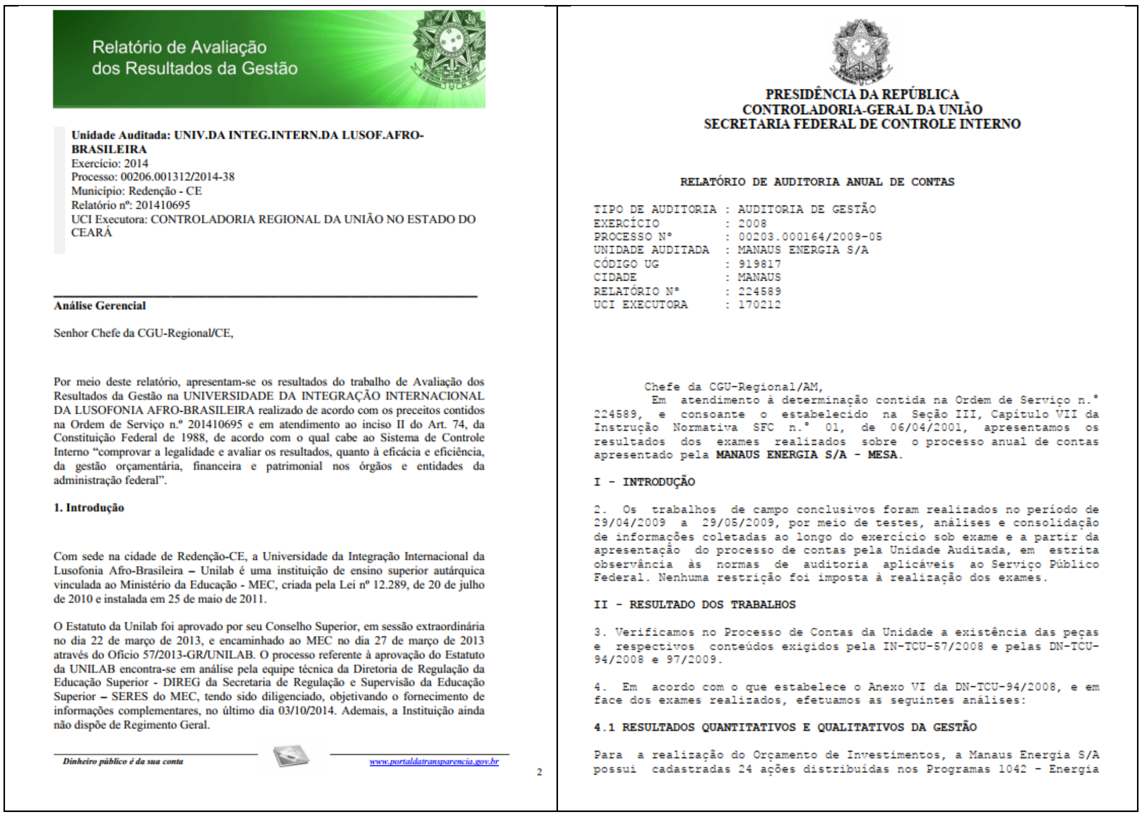
Table 2. Structured data collect form PDF and example of data.

OrgSub Cod	Report Number	Report Type	Date Begin	Date End	Const Number	Const TYPE	Const Title	Const Text
26271	1249807	Avaliação dos Resultados da Gestão	2014/8	2015/12	3.1.2.1	constatacao	002 desembolso de recursos sem a devida compr...	, a equipe responsavel pelo acompanhamento da ...
26271	1249807	Avaliação dos Resultados da Gestão	2014/8	2015/12	3.1.2.2	informacao	003 avaliacao da prestacao de contas referent...	s apontados quanto ao nao cumprimento integral...

The code of audited entity (**OrgSub Cod**), the period of analysis (**Date Begin, Date End**) are the information necessary for link to the time series of expenses, the second part of this project. The type (**Const Type**), title (**Const Title**) and text (**Const Text**) of the findings is the data used to NLP process.

Each type of report has it on format and distribution of the information, like shown in the next figure. And the format could change inside of each type.

[5] https://selenium.dev/
[6] https://phantomjs.org/
[7] https://tika.apache.org/0.7/parser.html


Relatório de Avaliação
dos Resultados da Gestão

Unidade Auditada: UNIV.DA INTEG.INTERN.DA LUSOF.AFRO-BRASILEIRA
Exercício: 2014
Processo: 00206.001312/2014-38
Município: Redenção - CE
Relatório nº: 201410695
UCI Executora: CONTROLADORIA REGIONAL DA UNIÃO NO ESTADO DO CEARÁ

Análise Gerencial

Senhor Chefe da CGU-Regional/CE,

Por meio deste relatório, apresentam-se os resultados do trabalho de Avaliação dos Resultados da Gestão na UNIVERSIDADE DA INTEGRAÇÃO INTERNACIONAL DA LUSOFONIA AFRO-BRASILEIRA realizado de acordo com os preceitos contidos na Ordem de Serviço n.º 201410695 e em atendimento ao inciso II do Art. 74, da Constituição Federal de 1988, de acordo com o qual cabe ao Sistema de Controle Interno “comprovar a legalidade e avaliar os resultados, quanto à eficácia e eficiência, da gestão orçamentária, financeira e patrimonial nos órgãos e entidades da administração federal”.

1. Introdução

Com sede na cidade de Redenção-CE, a Universidade da Integração Internacional da Lusofonia Afro-Brasileira – Unilab é uma instituição de ensino superior autárquica vinculada ao Ministério da Educação - MEC, criada pela Lei nº 12.289, de 20 de julho de 2010 e instalada em 25 de maio de 2011.

O Estatuto da Unilab foi aprovado por seu Conselho Superior, em sessão extraordinária no dia 22 de março de 2013, e encaminhado ao MEC no dia 27 de março de 2013 através do Ofício 57/2013-GR/UNILAB. O processo referente à aprovação do Estatuto da UNILAB encontra-se em análise pela equipe técnica da Diretoria de Regulação da Educação Superior - DIREG da Secretaria de Regulação e Supervisão da Educação Superior – SERES do MEC, tendo sido diligenciado, objetivando o fornecimento de informações complementares, no último dia 03/10/2014. Ademais, a Instituição ainda não dispõe de Regimento Geral.

Dinheiro público é da sua conta www.portaldatransparencia.gov.br 2

PRESIDÊNCIA DA REPÚBLICA
CONTROLADORIA-GERAL DA UNIÃO
SECRETARIA FEDERAL DE CONTROLE INTERNO

RELATÓRIO DE AUDITORIA ANUAL DE CONTAS

```
TIPO DE AUDITORIA : AUDITORIA DE GESTÃO
EXERCÍCIO         : 2008
PROCESSO N°       : 00203.000164/2009-05
UNIDADE AUDITADA  : MANAUS ENERGIA S/A
CÓDIGO UG         : 919817
CIDADE            : MANAUS
RELATÓRIO N°      : 224589
UCI EXECUTORA     : 170212
```

Chefe da CGU-Regional/AM,
Em atendimento à determinação contida na Ordem de Serviço n.° 224589, e consoante o estabelecido na Seção III, Capítulo VII da Instrução Normativa SFC n.° 01, de 06/04/2001, apresentamos os resultados dos exames realizados sobre o processo anual de contas apresentado pela **MANAUS ENERGIA S/A - MESA**.

I - INTRODUÇÃO

2. Os trabalhos de campo conclusivos foram realizados no período de 29/04/2009 a 29/05/2009, por meio de testes, análises e consolidação de informações coletadas ao longo do exercício sob exame e a partir da apresentação do processo de contas pela Unidade Auditada, em estrita observância às normas de auditoria aplicáveis ao Serviço Público Federal. Nenhuma restrição foi imposta à realização dos exames.

II - RESULTADO DOS TRABALHOS

3. Verificamos no Processo de Contas da Unidade a existência das peças e respectivos conteúdos exigidos pela IN-TCU-57/2008 e pelas DN-TCU-94/2008 e 97/2009.

4. Em acordo com o que estabelece o Anexo VI da DN-TCU-94/2008, e em face dos exames realizados, efetuamos as seguintes análises:

4.1 RESULTADOS QUANTITATIVOS E QUALITATIVOS DA GESTÃO

Para a realização do Orçamento de Investimentos, a Manaus Energia S/A possui cadastradas 24 ações distribuídas nos Programas 1042 - Energia

Figure 4. Examples of distinct Audit Reports.

To recognize the attributes of table 2 we create a main pattern for each type of report. The main tool to detect this patterns was REGEX[8]. (Files of source code: *1d-DF_to_Columns_AAG_ARG.ipynb* and *1d-DF_to_Columns_AAC.ipynb*)

Considering the reports are write and edit by humans, the main pattern cannot be applied correctly to all reports. We create several variations of the main pattern, until the number of corrects parses are acceptable. The next table show the results of the parse.

Table 3. Number of Reports Parsed

Type Report	Nb Files	Nb Parsed	%
AAC	4019	3834	95%
AAG	847	756	89%
ARG	1058	981	93%
TOTAL:	5924	5571	94%

Each audit report contains one or more findings. These findings are initially classified like “*Informação*” (information) and “*constatação*” (finding). And it has the “*conclusão*” (conclusion). Some reports don’t bring this classification (Not Classified). These attributes are parsed and the quantity numbers are shown in the next table. The numbers of parsed findings are 89099. (*1e-Group_AAG_ARG_AAC.ipynb*)

[8] https://docs.python.org/fr/2.7/howto/regex.html

Table 4. Number of Reports Parsed

Type Report	conclusao	constatacao	informacao	Not Classified	TOTAL
AAC	3762	18469	17003	32426	71660
AAG	731	2021	2197	299	5248
ARG	855	2774	1300	7262	12191
TOTAL:	5348	23264	20500	39987	**89099**

2.3 Discussion and Conclusion

The process of data extraction from documents are largely discussed, but very dependent on the objective of the project. Like observed in this project, it is very dependent on the format of document, been very hard to automate for different type of documents.

This dependence in the characteristics of document, and that are not standard cause an increase of the algorithm logic to match the desired information. This increase the complexity of the code, the time to develop the solution and possibility of errors. These are the reasons for the extract and cleaning be responsible for almost 80% of time spent in Data Science projects [8].

3 Classification

There are several methods for Text Classification, with several works comparing the results of different methods of feature extraction and Machine Learn-ML. The most used feature extraction are Bag-of-Words, TF-IDF, Word2Vector [9] [10] and several pre-trained models: fastText, GloVe, ULMFit, ELMo, BERT and others.

For ML methods are presented the classical ones like Naïve Bayes, Logistic Regression, SVM and Random Forest. But the best results presented for the recent researches are using Neural Networks, in special Convolutional and LSTM [11] [12] [13].

The objective of this chapter is compare the result of the methods to classify the findings of audit reports. The classification is binary in high or medium/low risk.

The text classification starts with data preparation. It includes:

- **Remove Stop Words**: for deliver the important features to the ML algorithm, the connection words, the numbers and some special characters are removed.
- **Sentences**: the text of audit findings could have several paragraphs. And the ML algorithms has a limit for the number of features to consider each time. To consider more information of the text, they are divided in sentences. After some tests, the metrics results are better with NO Sentences, so we decide no use this approach.
- **Tokenization**: each segment is divided in words;
- **Balance Labels**: in the dataset, the percentage of high risk findings is only approximately 1%. This structure could make the ML algorithm does not consider correctly the label with low frequency. For solve, we used some methods discussed in specific topic.

The methods of stemming and lemmatization are initially tested, but considering the corpus is in Portuguese, the existing libraries did not present a good result. The lemmatization has been the common procedure to preprocess the textual data, but lemmatization in our case for the Portuguese documentation is not possible because of the non-availability of lemmatization library. Besides that, some references show, when you have a big corpus and use pre-trained models [14] [15], these methods do not add accuracy to result.

The following figure summarize the steps for text classification and the Features Extraction methods and Supervised Machine Learning methods compared. Each of these methods are discussed in details above.

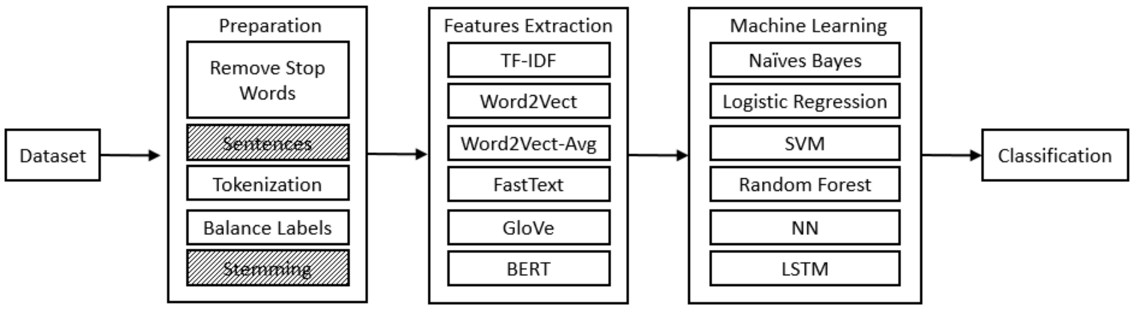


Figure 5. Text Classification steps and methods tested.

3.1 Balance Labels

Like in fraud detection datasets, the labels of audit reports dataset are very imbalanced. The percentage of high risk findings is only approximately 3%, like shown in figure below. When we use unbalanced datasets to train the standard Machine Learn algorithms, the more frequent parameter influences more in the result and in the end we could have a not useful model that only gives one result [16].

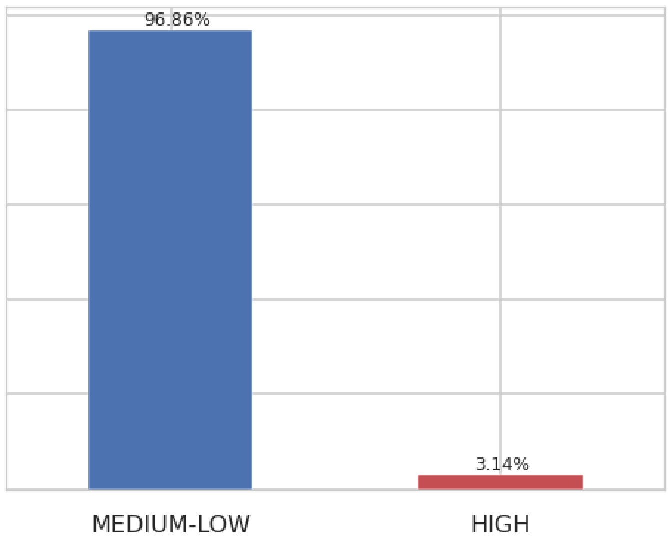


Figure 6. Very imbalanced labels - CGU Reports from 2015 to 2019.
Source: Created by authors.

The main strategies to work with imbalanced dataset are:

- **Undersampling** or sub-sampling the high frequency label. When the dataset is big, this method is adopted to decrease the computer requirements. Although, if the dataset has few occurrences, this method does not give good results.
- **Oversampling** randomly the low frequency label. Consist in create the balanced train dataset selection copies randomly repeatedly until labels has the same frequency. This method present some robust results [17]. One famous method is **Synthetic Minority Oversampling Technique (SMOTE**) and consist in oversample a dataset using classification methods. Some studies present better results (ROC curve) for this method [18].
- **Hybrid methods** combine the oversampling and undersampling. Some studies get better results mixing SMOTE with post-processing fit [19].

The following figure exemplify the results of each method:

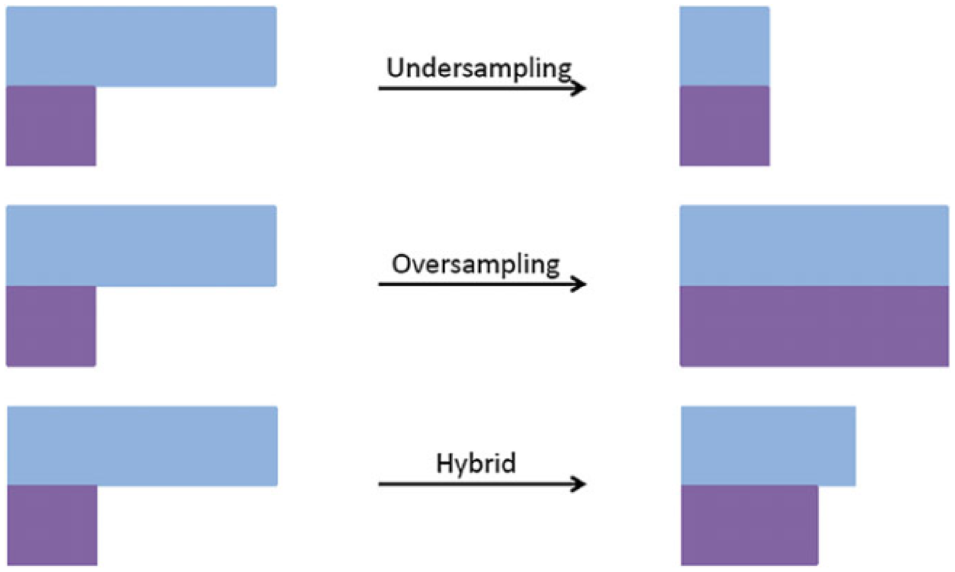


Figure 7. Resampling techniques.
Source: [20].

Another solution for treat imbalanced dataset is, instead of modifying the training data, change the machine learn algorithm. There are several different approach for change the classical classifier methods, like IFROWANN, OverBagging, UnderBagging, SMOTEBagging [20].

In this project, we decide to use Oversampling the training data using randomly copy of minority label, considering we don't have a very big dataset.

3.2 Features Extraction

Features extraction consists in transform the list of words from text data into a numerical matrix (or vector) that can be usable by the machine learn classifier. This method could create a very high dimensionally matrix, that is very difficult to analysis and process [21]. Reducing these dimensions is possible improve the accuracy, speed up the training and reduce overfitting risk.

The recent researches consider traditional Context-Free Representations and pre-trained models. The first ones consider that the word's meaning is stable, but in real text, this not the case. So, for try solve this problem, the pre-trained models use the large amounts to unlabeled data to create the representations. The current state-of-art models, like fastText, GloVe and BERT, use very large datasets and Deep Learning to create the models. The table below exemplifies the most common methods of features extraction [22] [23] [24]:

Table 5. Different type of Feature Extraction

Context-Free Representations	**Pre-trained Models**
Bag of Words (BoW)	fastText
TF-IDF	GloVe
Word2Vec	BERT

Source: Created by authors.

The **Bag of Words** is the most common and simple method. It consists in a matrix that indicates the presence of the word in document, doesn't caring about the number of times the word occurs in a document. The problem is the size of matrix created for documents with several different words.

3.2.1 TF-IDF

The Term Frequency-Inverse Document Frequency – TF-IDF is a numerical statistic with try to reflect the importance of the term/word (*w_j*) in a document (*d_i*). The Term Frequency could be the count of the words in the document. The other possibilities are binary (have or not the word) or the logarithmic frequency.

$$TF(w_j, d_i) = count(w_j, d_i)$$

The Inverse Document Frequency – IDF indicate if the word is rare or common (or the level of information that the word provides). It is a measure of the number of documents (|D|) over the number of documents that the word appears:

$$IDF(w_j) = \frac{|D|}{|\{d_i,\ w_j \in d_i\}|}$$

The result vector of TF-IDF is the size of the vocabulary and for some cases it will be very big and difficult for a classification machine learn method treat. So for reduce the dimensionally we applied methods like PCA (Principal Component Analysis) or t-SNE (t-distributed Stochastic Neighbor Embedding). In our tests, t-SNE gives the better results, so we will use it for the results presented.

3.2.2 Word2Vector

The Word2Vector is a pre-trained model to produce word embeddings, created using simple neural networks for reconstruct linguistic context of words. Word2vec is not a deep neural network, but just a way to convert text into a numerical form that computers algorithms can treat.

In bag of words and TF-IDF, there is no notion of similarity between words, as these are represented as indices in a vocabulary. Existing techniques supports modest dimensionality of word vectors between 50-100, but word2vec is tested with even higher dimension, between 300 and 600. [25] provides a way to train on more than half billion words generating higher dimensional vectors. The purpose of Word2vec is to group the vectors of similar words together in vector space.

Word2vec focuses on increasing the quality of word representation while using the neural network language model with linear projection layer and nonlinear hidden layers. Distributed representation of words learned by neural network preserves linear regularities among words and is found to be better than Latent Semantic Analysis (LSA) and Latent Dirichlet Allocation (LDA) [25]. Huffman tree based hierarchical softmax has lesser complexity, each vocabulary is represented as Huffman binary tree. More than neural network language model, the Recurrent neural net language model is found to be efficient because of its ability to learn complex patterns. It is also shown that by using hierarchical softmax the complexity is reduced. For faster and efficient distributed representation of word, CBOW (continuous bag of words) and SKIP (continuous skip gram model) were developed.

For skip-gram, the input is the target word, while the outputs are the words surrounding the target words. For instance, in the sentence "*I have share in apple's stock*", the input would be "*apple*", whereas the output is "*I*", "*have*", "*share*", "*in*" and "*stock*", assuming the window size is 6. With bigger context window size, resulting vectors get better quality, but requires higher computational complexity. Words closer to the target word carry more weight than those far from target word in the context of window. At the end of the output layer, a hierarchical softmax activation function is applied so that each element of the output vector denotes the probability of words that could appear in context.

In CBOW, we want to know which word is most likely to appear in it. CBOW uses neural network language model without nonlinear hidden layer and the projection layer is shared for all words, thus all word vectors get averaged. The graph below exemplifies the network structure.

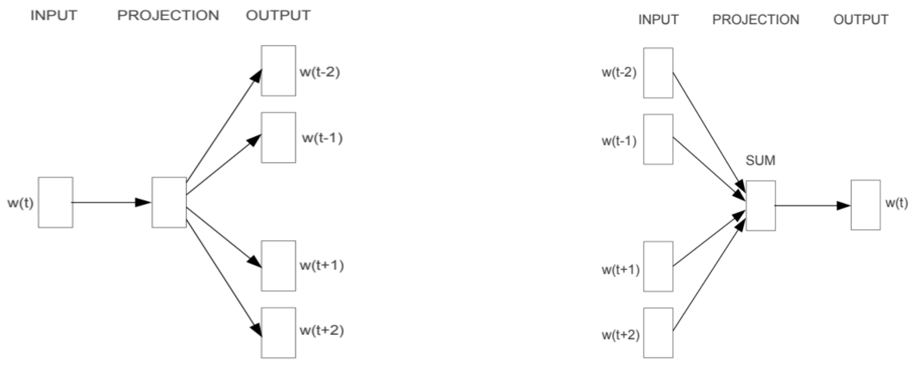


Figure 8. Skip-gram (predicts surround based on current word) and CBOW (predicts current word base on context).
Source: [25].

The word2vec word representations are quite amazing. As an example say, we want to find the word "*smallest*" which is similar to "*small*" like we have "*biggest*" similar to "*big*", when we do vector calculations such that Vector("*biggest*")-Vector("*big*")+Vector("*small*") then the resultant vector will be the actual/closest approximation of the vector that represents "*small*".

The word2vec with weighted average is a variation to try to improve the model. It is possible to use the TF-IDF parameter to weight the importance of the word in the context [26].

3.2.3 FastText

FastText was developed by Facebook Research and it is one of the popular techniques these days. It is an open-source, free, lightweight library that allows users to learn text representations and text classifiers for efficient learning of word representations and text classification. Its support for both supervised and unsupervised learning.

Unlike Word2Vector, the FastText is provided as a black box to the end user. However some reference papers [27] [28] [29] fairly depicts the working principle of FastText. Milokov [25] proposes to preserve the morphology of word by using a skip gram model, which each word is represented by a bag of character n-grams. Each character n-gram has a vector representation, so the sum of vector representations in turn depict the word. This method is fast and allows us to compute word representations of words not present even in training corpus.

Some popular methods use feed forward neural networks for generating word embedding by taking into account the context of appearance of words [30]. But they fail to take into account the morphological information embedded within the word. The method presented in [30] is the extension of Mikolov's word2vec SkipGram model. So the vector representation of each word not only considers the morphology of the word but also the context of the word, which can be adjusted by adjusting the window size.

FastText is faster than any word embedding. It is possible to train fastText on more than one billion words in less than 10 minutes using standard multicore CPU and classify half a million sentences into 312k classes in less than a minute [28].

While dealing with multi-class classification problems, the Hierarchical softmax function is used to reduce the computational complexity. Hashing trick is used for faster mapping of n-grams as n-gram can preserve information about local word ordering considering local context which is not possible in bag of words. But in general scenarios, multinomial logistic regression is used, where the sentence/document vector corresponds to the features that represent the words.

To make the fastText get trained faster and make final model consume less memory, the feature pruning was introduced, quantization, hashing and retraining features to the existing fastText for the text classification model uses less kilobytes [29].

3.2.4 GloVe

Global Vector for word representation (GloVe) is an unsupervised learning algorithm for obtaining vector representations for words. Developed by Stanford [31], it takes a corpus of text and intuitively transforms them into a matrix which is formulated by taking into account the context in which the word appears. This matrix is a co-occurrence matrix which is formed as below:

- For each word, the word co-occurrence statistics is collected and put in the form of co-occurrence matrix X. Each element *Xij* of the matrix represents the frequency of word *i* appearing in context of word *j*. *Xi* is the sum of number of times any word appears in the context of word *i*. *Pij* denotes the probability of word with index j appearing in context of the word with index *i* [31]:

$$P_{ij} = P(j|i) = \frac{X_{ij}}{X_i}$$

- Depending on the distance of the context word from target word, the weights are assigned.

Since the context of the word is considered for generating the feature matrix, this matrix preserves the meaningful substructures and word similarities with vector distance. Also while training the model in mini-batches the non-zero elements in the co-occurrence matrix alone are considered unlike word2vec which considers the sparse matrix as all. GloVe takes in the goodness of global matrix factorization method (LSA) and local context window method (skip-gram).

To keep the word meaning both syntactic and semantic relationships, GloVe proposes to use co-occurrence probability to preserve the inter relationship and meaning between words. But to capture the analogy relationships and linear relationships in vector space, the word vectors must be in such a way that the objective function *J* is minimized.

$$J = \sum_{i,j=1}^{V} f(X_{ij})\,(w_i^T w_j + b_i + b_j - logX_{ij})^2$$

Where w_i and w_j are the dot product of the wordvectors of context word and target word, X_{ij} is the cooccurence probablility. The b_i and b_j are the bias terms. This signifies that the objective function tries to minimize the difference between the dot product of word vector and the cooccurence probaility.

GloVe offers 4 different pre-trained word vectors each trained on different context such as Wikipedia data, twitter data and common crawl.

3.2.5 BERT

Now the Google Search is powered by its own home grown architecture BERT [32]. The Bidirectional Encoder Representations from Transformers – BERT like the Glove and word2vec considers the context word from both the left and the right sides of each target word. But BERT does not process the text inputs sequentially like the directional models, rather reads in the whole input text as a sequence at once. BERT uses the attention model in the language modelling.

BERT implementation is done in two stages namely pre-training stage and fine tuning. In pre-training stage, the 'Masked Language Model' (MLM) and 'Next Sentence Prediction' (NSP) are used [32]. In the MLM stage, some tokens that are inputted and randomly masked. The classifier is pre-trained with the unmasked tokens and is used to predict the masked tokens. NSP is used to understand the relationship between two sentences, hence to make the model understand the sentence relationships. The model is trained in such a way that sentence A and B are fed where B in 50% time is the actual subsequent sentence while rest of the time it is some random sentence picked randomly from the corpus.

The second stage which is fine tuning is more task specific. The pre-trained model is took and given the task specific inputs so that the pre-trained model is fine tuned to adapt to specific task.

BERT is a bidirectional deep neural network model based on Transformer architecture. The Transformer is based in attention, and replace the recurrent layers commonly used in encoder-decoder implementations [33]. The process of Transformers treats in parallel and requires less computing resources compared to recurrent NN solutions.

3.3 Supervised Machine Learning Methods

The most popular supervised machine learning algorithms for text classification could be classified in classical methods (Naïve Bayes, Logistic Regression, SVM, Random Forest, etc) and deep learning methods (Convolutional Neural Network, Long-Short Term Memory, etc).

The DL is type of machine learn algorithms based in Artificial Neural Networks (NN) with the main architectures such as Deep Neural Networks (DNN), Recurrent Neural Networks (RNN) and Convolutional Neural Networks (CNN). The main idea of DL is use several layers of NN in each one focus in a very specific simple abstract characteristic/task and train it in large scale with a big dataset.

The deep learning simplifies the process of features engineering but requires a more data and computer processing. Normally classical ML could give better performance with small datasets. In another hand, DL improves more with more data. These characteristics takes the DL uses more computer power to be trained. The figure below exemplifies this differences.

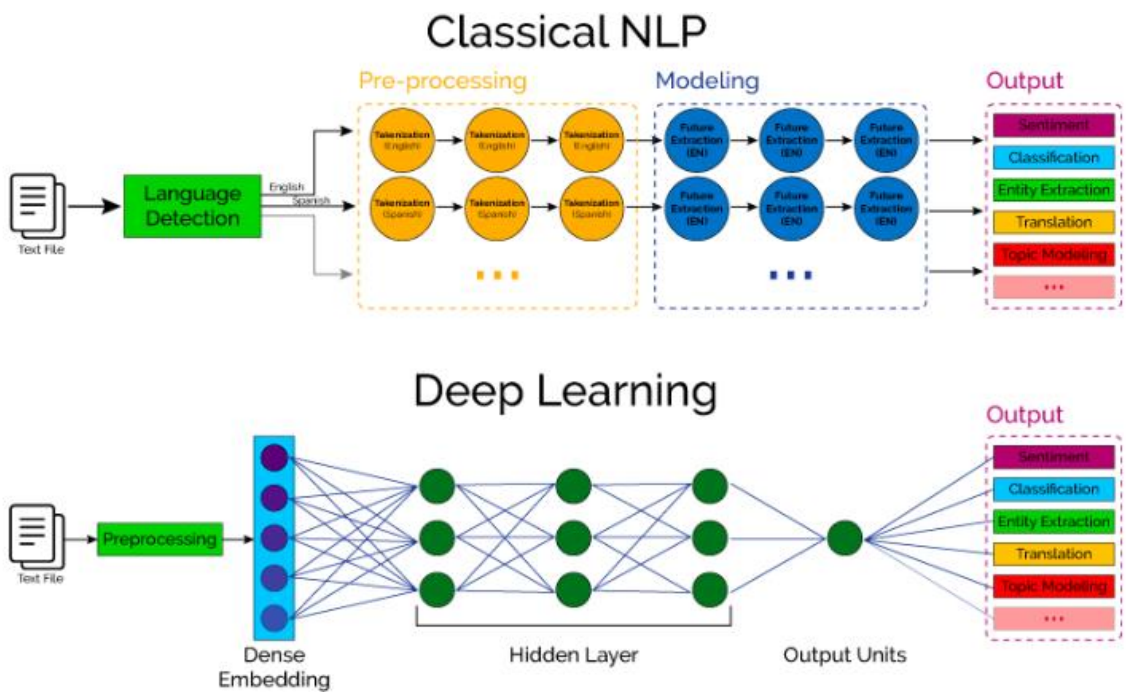


Figure 9. Classical Machine Learn compared Deep Learning.
Source: Adapted from . [34]

In the next topics we will detail the algorithms, both classical and deep learn, we used in this project.

3.3.1 Naïve Bayes

The Naïves Bayes (NB) classification algorithm uses the Bayes Theorem, which calculate the probability of an event given the probability of other event occurred. The following equation details Bayes Theorem, for calculate the probability of class "*c*" given a document "*d*".

$$P(c|d) = \frac{P(d|c) * P(c)}{\mathrm{P(d)}}$$

To create a NB classification model we need to calculate the probabilities [35]. Probability of class 'c' is the number of document in 'c' over the total of documents.

$$P(c) = \frac{|d_c|}{|d|}$$

The probability of a document given a class is described below. "N" is the number of words belonging to the document.

$$P(d|c) = |d_c|! \prod \frac{P(w|c)^N}{N!}$$

The probability of a document is based in the probability of all class and document given the classes:

$$P(d) = \sum P(c)P(d|c)$$

Like present in [36], NB method performs well in problems linear separable.

3.3.2 Logistic Regression

The Logistic Regression use similar approach of Linear Regression using a Logit Function or Sigmoid Function. This method gives better results for problems nonlinear separable.

$$P(x) = \frac{1}{1 + e^{x}}$$

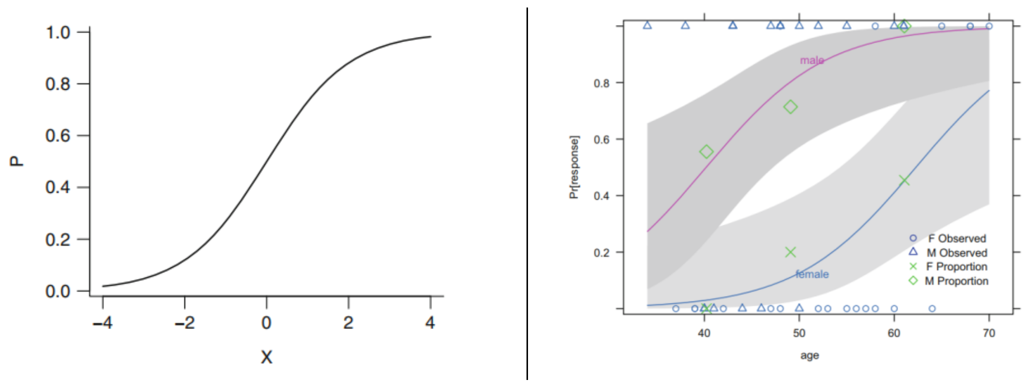


Figure 10. Logistic Function and example of model curves.
Source: Adopted from [37].

3.3.3 SVM

The Support Vector Machines (SVMs) is applied to classify linear and nonlinear data. It transforms the data in high dimension using nonlinear mapping and searches the linear optimal hyperplane in this new dimension.

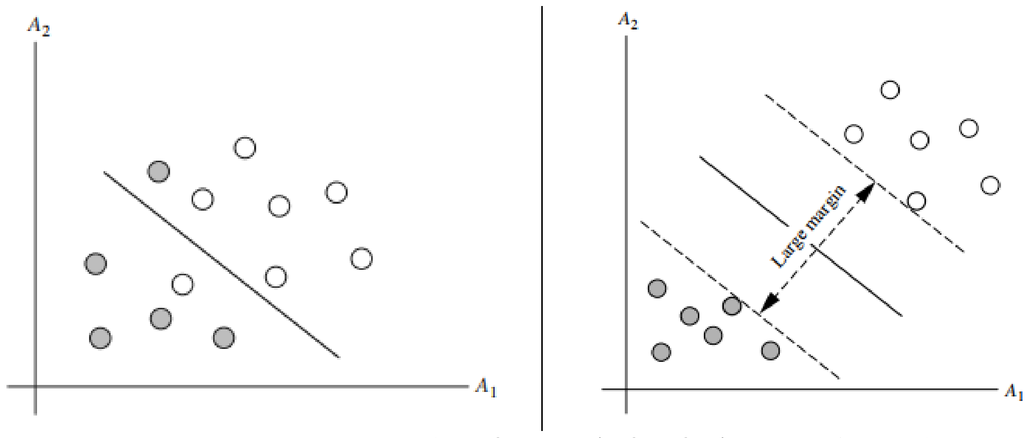


Figure 11. SVM – Transform and after find optimal hyperplane.
Source: Adopted from [35].

Some possible transformations functions are:

Polynomial kernel of degree h: K(Xi, Xj) = (Xi * Xj + 1)^{h}

Gaussian radial kernel: K(Xi, Xj) = $e^{-|Xi-Xj|^2/2\sigma^2}$

Sigmoid kernel: K(Xi, Xj) = tanh(kXi * Xj - δ)

The SVM algorithm is very popular and presents good results [35] [5] [12], normally it is less susceptible to overfitting, but could be very slow compared with other classical ML. Considering this strong characteristics, SVM is normally one of the methods used like the baseline to compare with new solutions.

3.3.4 Random Forest

The Random Forest (RF) is an ensemble method for improve the performance of classification model. Others popular ensemble methods are Bagging, Boosting and Adaboost.

The RF uses a "forest" of decisions trees generated using random selection of attributes (bootstrap). Each decision tree is considered and the end take the majority of the votes for the classification. The following figure exemplifies the RF working method.

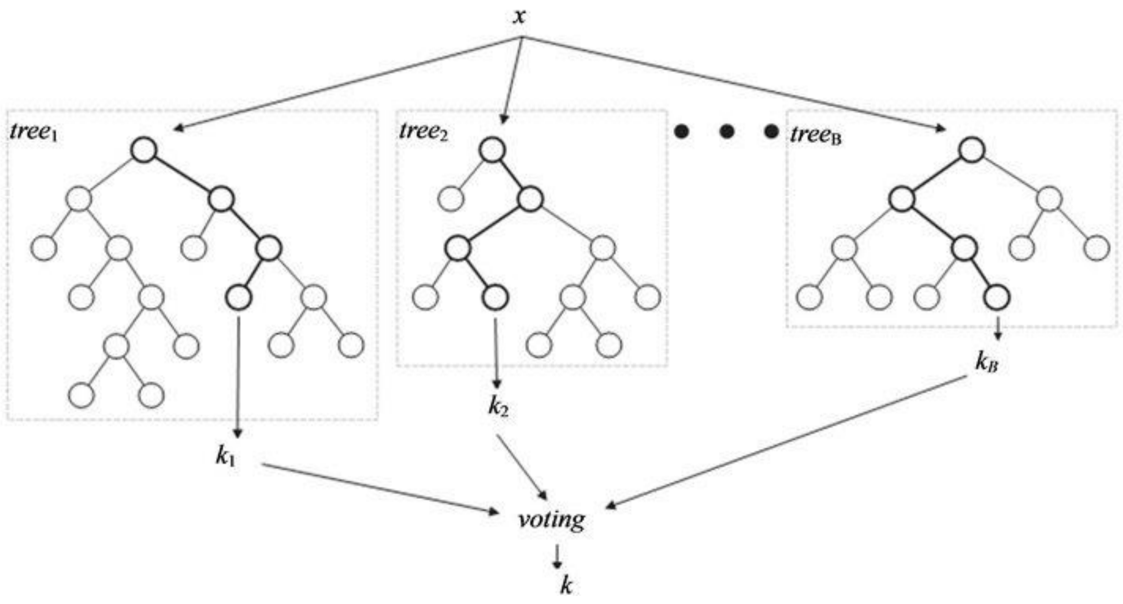


Figure 12. Random Forest.
Source: [38].

3.3.5 Neural Network

A Neural Network (NN) is a circuit of percetrons. A perceptron is an artificial neuron that receives some binary inputs (x_1 x_2, … , x_n), apply some weights (w_1, w_2,…, w_n) and produce single output [39]. Using the Backpropagation algorithm is possible to adjust the values of weights for the labels present in training dataset (supervised learning). Changing the weights values and adding more neurons and layers, it is possible to create NN for solve some problems. The follow figures exemplify these concepts.

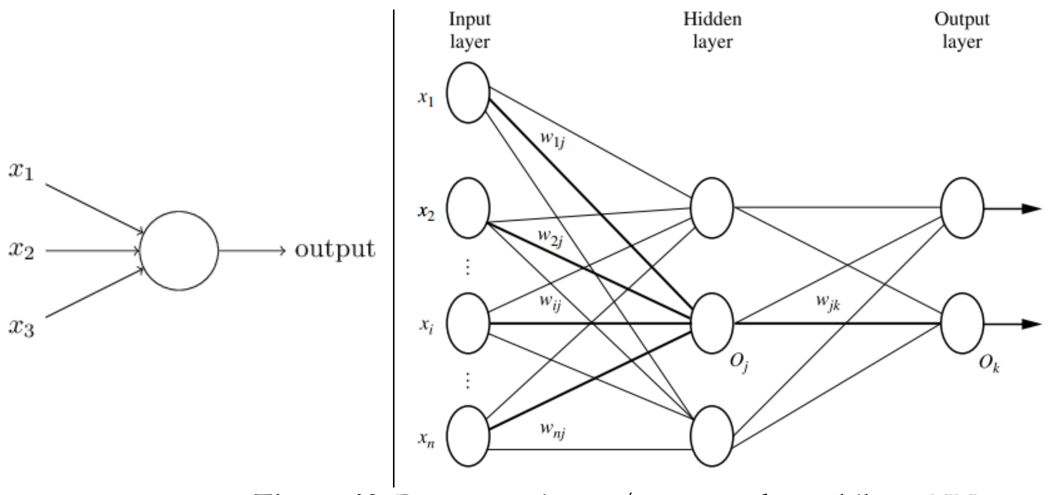


Figure 13. Perceptron inputs/output and a multilayer NN.
Source: Adopted from [39] and [35].

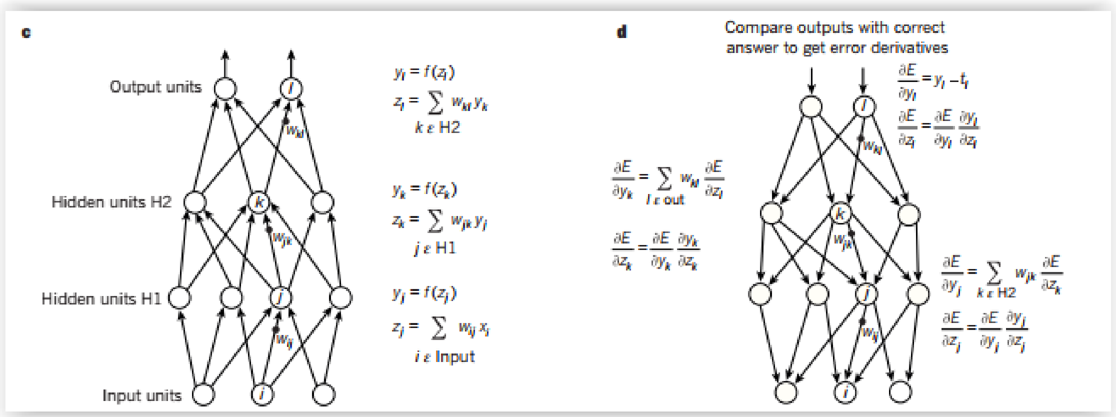

c
Output units
Hidden units H2
Hidden units H1
Input units
$y_l = f(z_l)$
$z_l = \sum_{k \in H2} w_{kl} y_k$
$y_k = f(z_k)$
$z_k = \sum_{j \in H1} w_{jk} y_j$
$y_j = f(z_j)$
$z_j = \sum_{i \in Input} w_{ij} x_i$
d
Compare outputs with correct answer to get error derivatives
$\frac{\partial E}{\partial y_l} = y_l - t_l$
$\frac{\partial E}{\partial z_l} = \frac{\partial E}{\partial y_l}\frac{\partial y_l}{\partial z_l}$
$\frac{\partial E}{\partial y_k} = \sum_{l \in out} w_{kl} \frac{\partial E}{\partial z_l}$
$\frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k}\frac{\partial y_k}{\partial z_k}$
$\frac{\partial E}{\partial y_j} = \sum_{k \in H2} w_{jk} \frac{\partial E}{\partial z_k}$
$\frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial z_j}$

Figure 14. NN Backpropagation.
Source: [40]

3.3.6 Convolutional Neural Network

The Convolutional Neural Network (CNN or ConvNet) is one type of deep learning Neural Network (NN). The CNN use 4 main ideas: local connections, shared weights, pooling and many layers [40]. The first stages there are some convolutional and pooling layers. The convolution operation consists in apply some filters to verify some features. And pooling layers select some outliers, like max or min value. The CNN are largely applied in image recognition and in NLP.

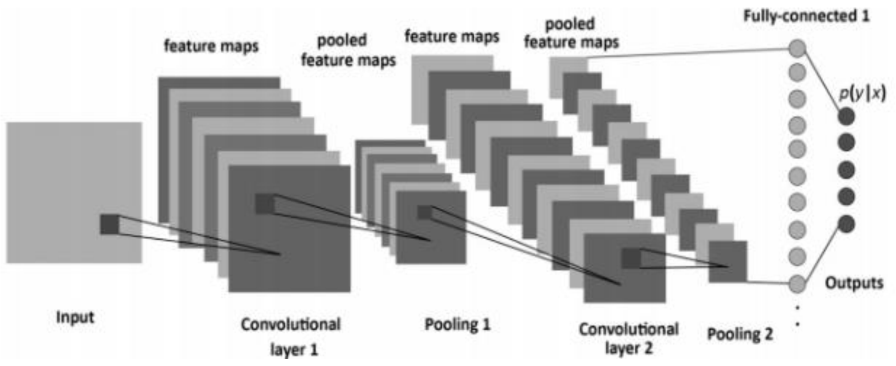


Figure 15. NN Backpropagation.
Source: Adapted from [41]

3.3.7 LSTM

The Long Short-Term Memory (LSTM) is a Recurrent Neural Network (RNN) emerged for solution of problems with sequential data. A RNN consists direct connections between nodes in a temporal sequence. They are used in several domains like handwriting recognition, speech modeling and language modeling [42].

The LSTM consists in a recurrent network with the main blocks input and output and three gates: input, output and forget. The block input receives the signal from last recurrent block. The input gate is the new value from data, and the output gate is a filtered version of the block output. The forget gate apply a function to "forget" some elements. All this gates could be influenced by internal state, called "peephole connections" [43] [42]. The following figure exemplifies LSTM network.

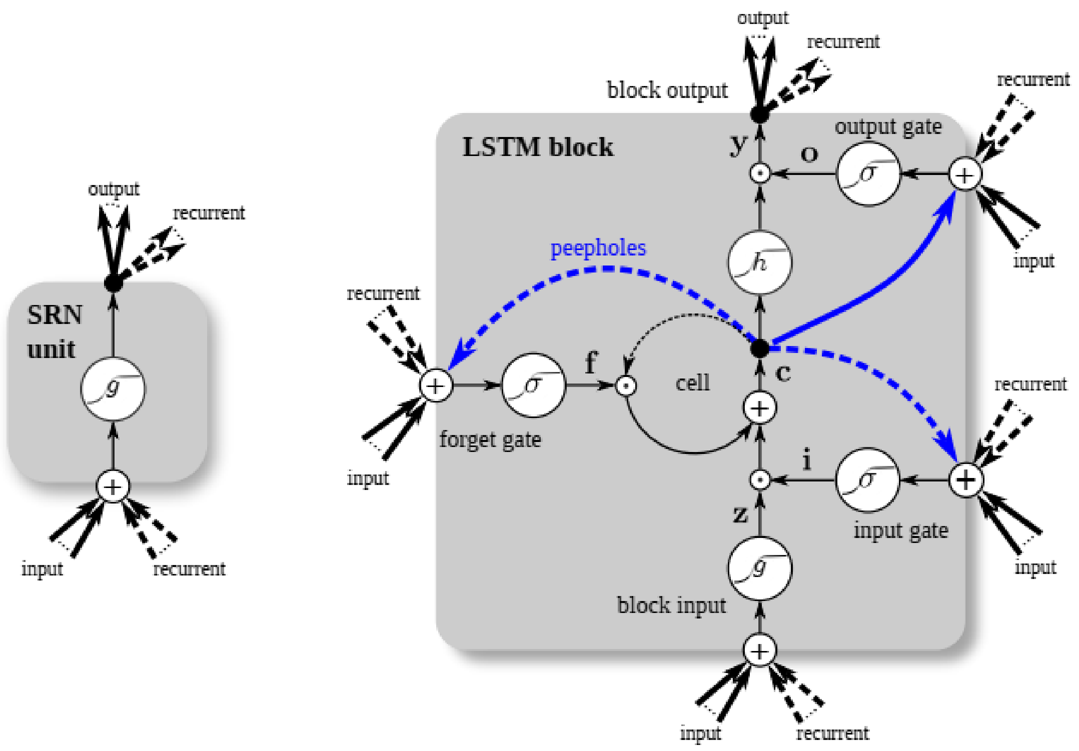


Figure 16. Simple Recurrent Network (SRN) and a detailed LSTM block
Source: Adapted from [42]

3.4 Metrics

The metrics are used to evaluate the quality of the model. We used four performance measures generally used by the researches: accuracy, precision, recall and F1 score. All these measures are based on the confusion matrix, that indicate the number of predictions correct or not. It is divided in True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). The measures are explained using this notation.

Table 5. Confusion Matrix

		True Category	
		High Risk	Low Risk
Assigned Category	High Risk	*TP*	*FP*
	Low Risk	*FN*	*TN*

Table 6. Measures of model performance

$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$
$Precision = \frac{TP}{TP + FN}$
$Recall = \frac{TN}{FN + TN}$
$F1\ Score = 2 * \frac{Precision * Recall}{Precision +\ Recall}$

3.5 Methods

Like presented in figure 5, we prepared the text, removing the stop words, transform in tokens and balance the labels. After, we combine some techniques of features extraction with machine learn algorithms and compare the results. The dataset is the CGU Reports published in internet from 2016 until 2019.

To remove stop words, we used the function '*stopwords*' from NLTK library, combined with specific characteristics of the data set. To simplify the feature extraction, the word accentuation was transformed to non-accentuated form. We also removed all numbers, months, special characters. The following table exemplify the result.

Table 7. Example of result of '*remove stop words*'

Original text example:
'*a unidade nao apresentou a totalidade da documentacao mencionada no art. 13, iii da instrucao normativa tcu 63*'
After '*remove stop words*':
'*unidade nao apresentou totalidade documentacao mencionada art . instrucao normativa tcu*'

For create the ML model, we divided randomly the dataset in 2/3 for the train and 1/3 for test. Considering that the machine learning algorithms present better results when the labels are not so unbalanced, we used oversampling technique to balance the data. The oversampling was applied just in the train dataset. The figure exemplifies the oversampling results.

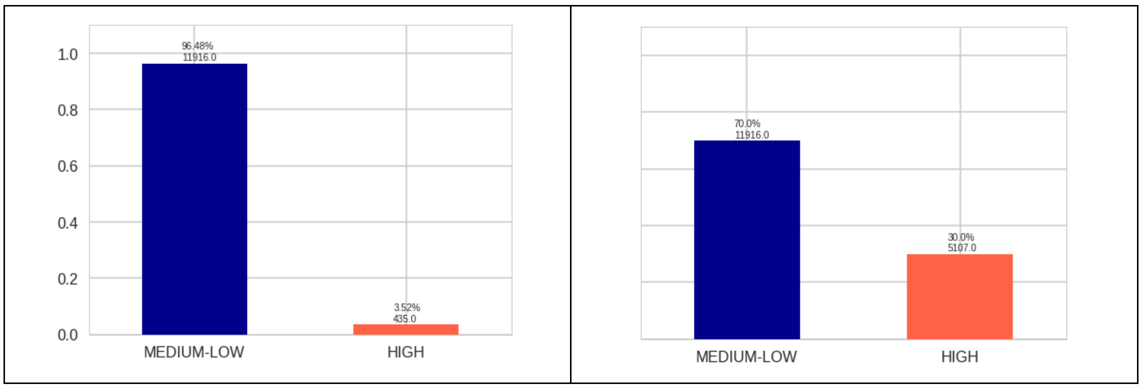


Figure 17. Original imbalanced and oversampled to 30% - CGU Reports - 2016 to2019.
Source: Created by authors.

About sentences and stemming, after some tests we verify that the metrics get worse when we use them. So the results present not considered both methods. Some possible reasons for this results are discussed in next topic.

All used feature extraction and machine learn algorithms are based on public python libraries or pre-trained models. The library to prepared the date in NLTK and Regex. The following tables list the libraries used. All the codes are available in github/gustavofleury[9].

Table 8. Features Extraction python libraries or pre-trained models.

Method	Library	Pre-trained Model
TF-IDF	NLTK	-
Word2Vector (Avg)	Gensim	Created using dataset.
FastText	FastText	Portuguese Wiki 300 dim. [44]
GloVe	-	Portuguese Glove 300 dim. [45]
BERT	Bert	Multilingual Wiki Base [46]

- Methods do not use specific library or pre-trained model.

Table 9. Machine Learning python libraries.

Method	Library
Naïve Bayes	Scikit-Learn (sklearn) [47]
Logistic Regression	
SVM	
nRandom Forest	
NN	
CNN	Keras [48]
LSTM	

The Features Extraction methods create vectors to represent the words. The TF-IDF create variable multidimensional matrices that was reshaped using t-SNE. The t-SNE presented best results than PCA. The model Word2Vector was created with dimension 300., The FastText and Glove used pre-trained models with dimension size of 300 too. Dimension bigger than 300 does not give a relevant improvement in the result, but requires a lot more of computer processing [49].

For classical ML algorithms, the input of the algorithm is the normalized result of the sum of the word vectors. So, independent of the number of words in the text, the final vector will have the same number of dimensions The final result is independent of the word order.

For the Deep Learning (DL) algorithms, we used the embedding layer before feed the neural network. In this approach, the position of the word influences in the result. For fastText and GloVe, instead to use a basic NN, we used a CNN to compare the results with the LSTM algorithms.

The implementation of LSTM we used a bidirectional approach with 20% of recurrent dropout. This approach present better results for text classification [50]. For the CNN, we used 5 convolutional layers with dropout.

The BERT approach is different from another Features Extraction like GloVe and fastText, and the deep learning algorithm is inside of its implementation. Like presented, after apply the Transformer, it feed a bidirectional deep neural network.

[9] https://github.com/gustavofleury

3.6 Results

For compare the performance of feature extractions and ML methods, we used the metrics Accuracy, Recall, Precision and F1-Score. Like our label in test set is very imbalanced, the F1-Score is an important parameter to verify the performance.

Table 10. Metrics Results for composition Extract Features and ML algorithms.

		Naïve Bayes				Logistic Regression				SVM			
Extract Features	DS*	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF	Train	0.51	0.51	0.52	0.51	0.51	0.51	0.54	0.53	0.61	0.58	0.70	0.67
	Test	0.48	0.03	0.55	0.06	0.51	0.03	0.49	0.06	0.44	0.03	0.55	0.06
Word2Vector	Train	0.66	0.68	0.59	0.63	0.94	0.91	0.97	0.94	0.94	0.91	0.98	0.95
	Test	0.70	0.06	0.61	0.11	0.88	0.13	0.48	0.21	0.88	0.13	0.51	0.21
Word2Vector-Avg	Train	0.66	0.68	0.58	0.63	0.93	0.90	0.96	0.93	0.93	0.90	0.90	0.93
	Test	0.70	0.06	0.60	0.11	0.87	0.13	0.51	0.20	0.86	0.12	0.52	0.19

		Random Forest				NN				LSTM			
Extract Features	DS*	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF	Train	1.00	1.00	1.00	1.00	0.99	0.99	1.00	0.99	1.00	0.99	1.00	1.00
	Test	0.78	0.04	0.21	0.06	0.80	0.04	0.21	0.06	0.80	0.04	0.21	0.06
Word2Vector	Train	1.00	1.00	1.00	1.00	0.99	0.98	1.00	0.99	0.86	0.90	0.81	0.85
	Test	0.97	0.52	0.11	0.18	0.94	0.23	0.33	0.27	0.88	0.06	0.20	0.09
Word2Vector-Avg	Train	0.99	0.97	1.00	0.99	0.99	0.97	1.00	0.99	0.86	0.90	0.80	0.85
	Test	0.96	0.28	0.14	0.18	0.93	0.19	0.35	0.24	0.88	0.05	0.17	0.08

		Naïve Bayes				Logistic Regression				SVM			
Extract Features	DS*	A	P	R	F	A	P	R	F	A	P	R	F
FastText	Train	0.71	0.93	0.05	0.10	0.80	0.81	0.45	0.57	0.82	0.85	0.47	0.61
	Test	0.97	0.60	0.02	0.30	0.93	0.16	0.30	0.21	0.94	0.19	0.32	0.24
GloVe	Train	0.70	0.94	0.01	0.03	0.80	0.79	0.47	0.59	0.82	0.81	0.52	0.63
	Test	0.97	0.32	0.03	0.06	0.92	0.02	0.32	0.21	0.92	0.15	0.30	0.20

		Random Forest				CNN				LSTM			
Extract Features	DS*	A	P	R	F	A	P	R	F	A	P	R	F
FastText	Train	1.00	1.00	1.00	1.00	0.97	0.97	1.00	0.98	0.70	0.70	1.00	0.82
	Test	0.97	0.52	0.12	0.20	0.96	0.96	1.00	0.98	0.96	0.96	1.00	0.98
Glove	Train	1.00	1.00	1.00	1.00	0.96	1.00	0.95	0.97	0.99	0.99	0.99	0.99
	Test	0.97	0.55	0.12	0.20	0.89	0.97	0.91	0.94	0.96	0.97	0.99	0.98
BERT**	Train	-	-	-	-	-	-	-	-	0.70	0.70	1.00	0.82
	Test	-	-	-	-	-	-	-	-	0.96	0.96	1.00	0.98

*DS - Dataset Accuracy(A), Precision(P), Recall(R) and F1-Score(F)

**BERT implementation uses his own bidirecional Recurrent Neural Network.

For the classical ML the best training metrics are Random Forest (RF), with NN with results very close. The result of RF (accuracy of 100%) is better than Naïve Bayes (NB) (accuracy of 51%). We got few improvements from NB to Logistic Regression and SVM. From the classical methods RF and NN give significant better results.

Although, in general, when we verify the metrics for the test subset using classical ML, the metrics are not good. Like for TF-IDF with RF we get an accuracy of 0.78 in test but the F1 Score of just 0.06. The explanation of this high accuracy is that the subset is very unbalanced, with more '*Low*' (simplification for '*Low-Medium*') than '*High*' risk labels. The ML classify most of test subset with '*Low*' label, the accuracy will present a good value, but in reality it is not a good result.

The following confusion matrix exemplifies this result. From the total of 6084, the predictions for '*Low*' are 5888 (97%) and just 196 for '*High*'. For this type of unbalanced data, metrics like Precision, Recall and F1-Score are more appropriate than Accuracy.

Table 11. TF-IDF with Random Forest – Confusion Matrix and Metrics

		True Category	
		Low Risk	High Risk
Assigned Category	Low Risk	*TP =4714*	*FP = 1174*
	High Risk	*FN = 145*	*TN = 51*

The big difference presented by the metrics for the train and test datasets exposes the overfitting of classical ML algorithms. When the Random Forest present a "perfect" classification for TF-IDF (accuracy, precision, recall and F1-Score all 100%), their application in test dataset does not present similar good metrics. To prevent this overfitting, could be done modifications in the parameters of the algorithm or test a different approach.

Independent of use basic extract features (TF-IDF, word2vec) or more complex extract features (fastText, GloVe), the method of create a summed final vector for feed the classical ML methods gets a little better precision (for test dataset RF-TF-IDF: 4% and RF-fastText: 52%), but the F1-Score not improved (for test dataset RF-TF-IDF: 6% and RF-fastText: 10%).

Big improvements are observed in using the pre-trained extract features with Deep Learning algorithms (CNN and LSTM). They got very strong training parameters and very similar metrics in test subsets. The results for fastText/Glove with CNN/LSTM and BERT are very similar. The best results, but not statistic significant, are BERT.

About the computational consummation, the DNN methods requires a lot more computer power. For comparison, to train the models, the classical ML requires just few minutes. In another hand, BERT training got more than one hour, using GPU. The use of GPU or TPU for processing the DNN improves significantly the speed of processing.

3.7 Discussion and Conclusion

The results show that is possible to use machine learn algorithms to detect the probably risk level of an audit finding. This model could be used to classify outside reports automatically, to support ongoing audit or the selection of object in annual planning.

The deep learning models proves to be more suitable than the classical ML for this type of text classification. These results are corroborated by similar researches [51] [52]. The classical ML approaches used are not able to create a good model. In other hand, the DL models give impressive metrics results. The difference between the DL models are not statistic relevant. So, in real applications, the decision with each method use, depends more in the requirements of time of response and resources available.

We used a dataset from 2016 to 2019, with approximately 17 thousand occurrences, that is a relative small dataset for train a DL model. We decided to use this filtered dataset because with more occurrences the computational resources available are crashing with memory limitations. Probably with more data the model could get better generalization.

In this project, we considered all the text of the audit finding together. We tried to divide the audit finding in sentences, given same label of the finding to each sentence. The initial compared results with sentences and no sentences, shows that the model with sentences decrease the performance. Additional research can understand the reason for this result.

For the unbalanced labels we worked with oversampling, getting reasonably results. More study about the impact of other types of solve this types of label balance problem could be interesting. The use of data augmentation for text, like synonym insertion, evolutionary algorithms [53] could give some improvements in the quality, mostly for less intensive ML methods.

Other methods that ensemble existing solutions or add new network layers could improve the results. Combine the result of the most prominent methods (LSTM, CNN, BERT) could increase the quality of the result. An important aspect to observe is verify if the improve of quality justify the time and computation cost to work with all models.

Conneau and LeCunn [13] present an study of use of Very Deep CNN, comparing the deep CNN uses for computer vision to an application to NLP. Some improvements in the LSTM and CNN hyper parameters and number of layers could give improvements in the model.

4 Clustering

Clustering is the process to group the data objects into multiple clusters (groups) considering that the elements of a cluster are similar, but dissimilar to elements of other clusters. The difference to classification is that the class label of each data object is unknown. This is the reason it is classified as **unsupervised learning**. So it is necessary find some parameter of similarity to make the clusters.

The clustering methods could be classified in [35]:

- **Partition methods**: normally are distance-base that use iterative relocation to create *k* clusters each has one or more elements. The elements belong only to one cluster. The main algorithms are *K-means* and K-*medoids*.
- **Hierarchical methods**: consist in a hierarchical decomposition, that could be agglomerative (bottom-up) or divisive (top-down) and are distance-based or density-based. Examples of algorithms are Balanced Iterative Reducing and Clustering using Hierarchies (*BIRCH*), *Chameleon* and Probabilistic Hierarchical Clustering.
- **Density-based methods**: different of distance-based, consider the number of objects (density) to group the elements. This method could include some outliers and take a shape different of spherical (distance-based characteristic). The main algorithms are Density-Based Clustering based on Connected Regions with High Density (*DBSCAN*) and Ordering Points to Identify the Clustering Structure (*OPTICS*).
- **Grid-based methods**: create a finite number of cell in a grid structure to apply in the object space. Compared, the other methods are data-driven and Grid-based are space-driven. Examples of algorithms are Statistical Information Grid (*STING*) and Apriori-like subspace Clustering (*CLIQUE*).

For text clustering the most used algorithms are *k-means*, Gaussian *EM*, Group Average Agglomerative (*GAAC*) and *DBSCAN* [54].

The following figure shows the Features Extraction methods and Machine Learn methods used for clustering. The Data Preparation and Features Extraction are explained in the Classification chapter. In the next topics we present the methods for dimensionality reduction and clustering.

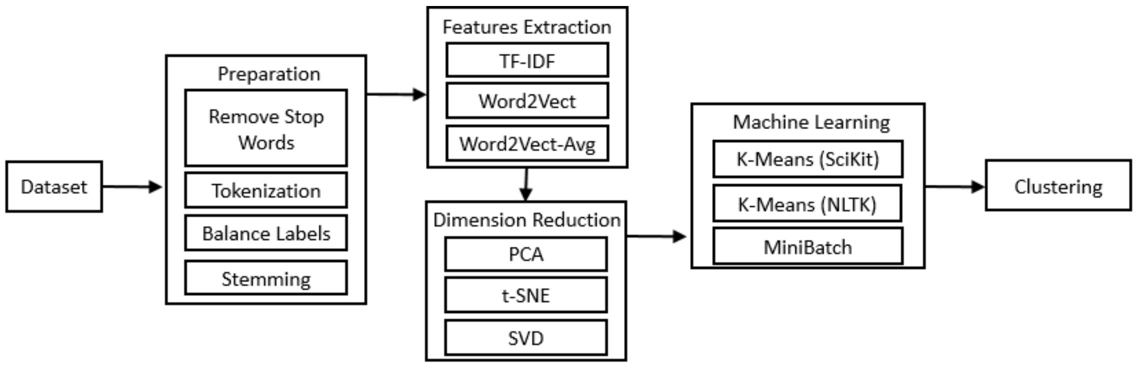


Figure 18. Text Clustering steps and methods tested.
Source: Created by authors.

4.1 Dimensionality Reduction

The objective of dimensionality reduction is obtaining a set of principal variables that represent the original dataset in a smaller volume. The application of ML algorithms will be more efficient, taking almost the same result. In this section we discuss the main used methods: PCA, SVD and t-SNE.

4.1.1 PCA

The Principal Components Analysis (PCA) combines the essence of variables and create a smaller set of attributes [35]. PCA starts with the normalization of data, to the large domains not dominate the smaller domains. After it is calculated *k* orthogonal vectors, smaller than the *n* original number of dimensions, that allow drop some attribute without lose much information.

The following figure explain visually the reduction. Applying the PCA in the original dataset, the new axis *pc2* has small variation and the problem could be treat just *pc1*.

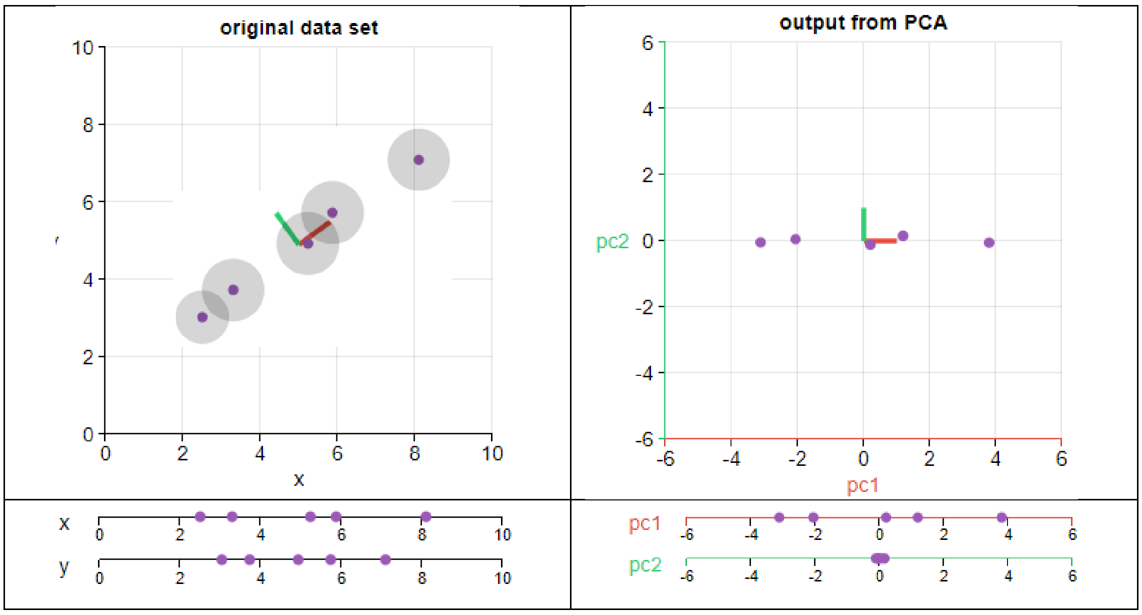


Figure 19. PCA visual example.
Source: Adapted from [55].

4.1.2 SVD

The Singular Value Decomposition (SVD) is a matrix factorization method that generalizes the square matrix calculation of eigenvalues and eigenvectors for matrixes of size *m* x *n*. The PCA and SVD are very similar, and SVD is more general. Like present in [56], SVD provides the same basis vectors and data transformation as PCA for datasets in which the mean of each attribute is zero.

The original data D is decomposed in 3 matrixes that contains the importance of each component, like visually explained in the following figure.

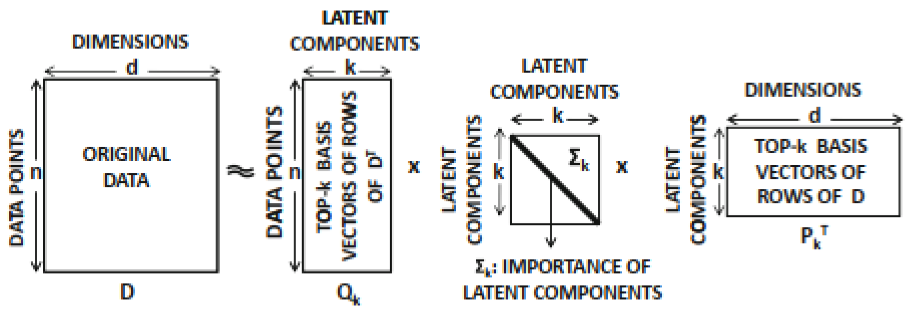


Figure 20. SVD visual example.
Source: [56].

4.1.3 t-SNE

The t-Distributed Stochastic Neighbor Embedding (t-SNE) was proposed by [57] in 2008 to create an away to visualize high-dimensional data by given each data point a location in a two or three-dimensional map. The first step of method is convert the Euclidean distances of data points into conditional probabilities that represents similarities. After, recreate a low dimensional space based on Student t-distribution. To optimize this distribution is done applying Gradient Descent.

Differently of PCA, t-SNE is not a linear projection, using local relationships of points to create the low-dimensional map. Compared with PCA, t-SNE could capture non-linear dependencies.

4.2 Unsupervised Machine Learning Methods

4.2.1 K-Means

The K-means aims to partition "*n*" observations into "*k*" clusters in which each observation belongs to the cluster. Clustering is done by separating samples in *n* groups of equal variance, minimizing a criterion known as the inertia or within-cluster sum-of-squares. Unlike Density based clustering algorithm, here the number of clusters in which observations has to be clustered has to be given by the user, the choice of number of cluster is not random, it needs domain experience to choose the cluster numbers.

The K-means method divides a set of *n* samples X into *k* disjoint clusters C, each described by the mean μ_j of the samples in the cluster. The means are commonly called the cluster "centroids". The K-means algorithm tries to choose centroids that minimize the inertia, or within-cluster sum-of-squares criterion:

$$\sum_{i=0}^{n} \min_{\mu_j \in C}(||x_i - \mu_j||^2)$$

The steps of the K-means are as follows, once the user gives the number of clusters in which data needs to be partitioned, the algorithm initially assigns *k* records randomly in space to the initially chosen random cluster Centre location. Then for each record, the nearest cluster center is found by using distance functions such as Euclidean Distance (but there are other distance functions as well). Cluster centers are updated frequently each time new cluster center is found. Hence the new centroid is the mean of all points assigned to that cluster. Until the cluster centroids stops changing or alternative if the mean square error does not become low further, the algorithm keeps running to find the nearest cluster Centre from the data points. The Mean Square Between (MSB) represents the between cluster variation and Mean Square Error (MSE), represents within cluster variation, hence the pseudo F Statistics here is the ration of MSB and MSE. This statistic will be higher if MSB is higher. Hence any clustering partitions that results in higher Pseudo F statistics and Higher MSB and very less MSE is considered as the best clustering results.

In higher dimensional spaces, running the K-means can be quite computationally challenging especially because of the curse of dimensionality. Hence, implementing any dimensionality reduction techniques prior to performing clustering will improve the speed and efficiency of algorithm.

4.2.2 MiniBatch

To reduce the computation time of traditional K-means, the MiniBatch K-means was formulated, which inputs subsets of the input data randomly sampled in each training iterations. These mini-batches drastically reduce the amount of computation required to converge to a local solution. '*B*' samples are randomly chosen from '*n*' observations and are randomly assigned to cluster. The cluster centroid is calculated from each observation in the *B* samples by using any of the distance function. And, the cluster centroid is updated by taking the average of the samples falling into the cluster. This step continues until all the *n* observations are considered and the termination criterion as in K-means is reached.

4.3 Methods

Like present in figure 18, we prepared the text, removing the stop words, transform in tokens, balance the labels and Stemming. After, we combine some techniques of features extraction (TF-IDF, Word2Vec and Word2Vec-AVG) with dimensionality reduction and clustering algorithms and compare the results. The dataset is the CGU Reports published in internet in 2019.

The initial part of preparing the dataset is similar that presented in item 3.5. One new step add is stemming that consist in simplify the words to a common base, reducing the inflectional forms. We test both stemmed and non-stemmed forms to compare the results.

All used feature extraction and machine learn algorithms are based on public python libraries or pre-trained models. The library to prepared the date in NLTK and Regex. The following tables list the libraries used. All the codes are available in github/gustavofleury.

Table 12. Features Extraction python libraries or pre-trained models.

Method	Library
TF-IDF	NLTK
Word2Vector (Avg)	Gensim

Table 13. Clustering Machine Learning python libraries.

Method	Library
K-means	Scikit-Learn (sklearn) [47]
MiniBatch	
K-means	NLTK

The K-means clustering is supported by two well-known libraries namely "Scikit-learn" and "NLTK", but the performance metrics of using these two libraries interchangeably has not been exploited. Also the results of unsupervised algorithm greatly vary with the number of words considered together (N-gram), the choice of words being lemmatized or not, stemmed or not, the choice of word embedding's (TF-IDF, Word2Vec and Word2Vec-AVG) and the type of dimensionality reduction is applied for converting the multidimensional vector space of the textual analytics into favorable vector space for easier computation without losing much of information.

While enormous observation of the order 17,000 samples as we considered for text classification are considered for K-means, the environments goes out of insufficient memory while performing K-means, hence for exploitation, we considered only the 2019 dataset with 181 observations.

We ran K-means and MiniBatch on stemmed and non-stemmed version of corpus considering different n-Grams of size 1 to 3, and testing the performance metrics in the case of TF-IDF and word2vec individually subjected to different reduction technique.

4.4 Results

For compare the performance of feature extractions, dimensionality reduction and clustering ML methods, we used the metrics Accuracy, Recall, Precision and F1-Score. For compute this metrics we used the binary Risk Label. The clustering was done for 2 groups. The best accuracy results for each Extract Feature and ML (each line) is bold.

Table 14. Metrics Results for composition Clustering – TF-IDF with N-Grams and Dimension Reduction.

t-SNE

		1-Gram				2-Gram				3-Gram			
Extract Features	ML	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF Stemmed	K-means (sci-L)	0.65	0.13	0.20	0.16	**0.68**	0.23	0.40	0.29	0.32	0.13	0.57	0.22
	K-means (NLTK)	0.35	0.17	0.77	0.28	0.42	0.20	0.80	0.31	**0.55**	0.04	0.07	0.05
	MiniBach	0.38	0.20	0.90	0.33	0.38	0.18	0.80	0.30	**0.69**	0.25	0.43	0.32
TF-IDF Non-Stemmed	K-means (sci-L)	0.70	0.23	0.33	0.27	**0.64**	0.17	0.30	0.22	0.36	0.16	0.67	0.26
	K-means (NLTK)	0.67	0.19	0.30	0.23	0.40	0.19	0.77	0.30	**0.65**	0.17	0.30	0.22
	MiniBach	0.30	0.15	0.70	0.25	**0.65**	0.19	0.33	0.24	0.35	0.16	0.70	0.26

PCA

		1-Gram				2-Gram				3-Gram			
Extract Features	ML	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF Stemmed	K-means (sci-L)	0.69	0.11	0.13	0.12	**0.70**	0.12	0.13	0.13	0.39	0.19	0.80	0.30
	K-means (NLTK)	0.33	0.15	0.67	0.25	**0.59**	0.15	0.30	0.20	0.39	0.19	0.80	0.30
	MiniBach	0.31	0.18	0.87	0.30	**0.69**	0.14	0.17	0.15	0.61	0.11	0.20	0.14
TF-IDF Non-Stemmed	K-means (sci-L)	0.28	0.16	0.77	0.26	**0.69**	0.16	0.20	0.18	0.62	0.12	0.20	0.15
	K-means (NLTK)	0.30	0.16	0.77	0.27	0.36	0.18	0.80	0.29	**0.38**	0.18	0.80	0.30
	MiniBach	0.31	0.17	0.83	0.29	**0.70**	0.18	0.23	0.20	0.66	0.14	0.20	0.16

SVD

		1-Gram				2-Gram				3-Gram			
Extract Features	ML	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF Stemmed	K-means (sci-L)	0.83	0.00	0.00	0.00	0.83	0.00	0.00	0.00	0.83	0.00	0.00	0.00
	K-means (NLTK)	**0.60**	0.15	0.30	0.20	0.46	0.15	0.50	0.23	0.55	0.16	0.40	0.23
	MiniBach	**0.19**	0.17	1.00	0.29	0.16	0.16	0.97	0.28	0.19	0.17	1.00	0.29
TF-IDF Non-Stemmed	K-means (sci-L)	**0.84**	1.00	0.03	0.06	0.83	0.00	0.00	0.00	0.83	0.00	0.00	0.00
	K-means (NLTK)	0.57	0.13	0.27	0.17	0.51	0.14	0.37	0.20	**0.59**	0.18	0.43	0.26
	MiniBach	0.16	0.16	0.97	0.28	0.16	0.16	0.97	0.28	0.16	0.16	0.97	0.28

		N-Gram											
		t-SNE				PCA				SVD			
Extract Features	ML	A	P	R	F	A	P	R	F	A	P	R	F
TF-IDF Stemmed	K-means (sci-L)	0.35	0.16	0.67	0.25	**0.69**	0.16	0.20	0.18	0.83	0.00	0.00	0.00
	K-means (NLTK)	0.33	0.13	0.57	0.22	0.33	0.16	0.73	0.27	0.56	0.19	0.50	0.28
	MiniBach	**0.67**	0.22	0.40	0.29	0.65	0.10	0.13	0.11	**0.73**	0.12	0.10	0.11
TF-IDF Non-Stemmed	K-means (sci-L)	**0.66**	0.19	0.33	0.24	**0.67**	0.15	0.20	0.17	0.83	0.00	0.00	0.00
	K-means (NLTK)	0.40	0.19	0.77	0.30	0.41	0.18	0.70	0.28	**0.62**	0.12	0.20	0.15
	MiniBach	**0.66**	0.18	0.30	0.22	**0.64**	0.17	0.30	0.21	0.16	0.16	0.97	0.28

Accuracy(A), Precision(P), Recall(R) and F1-Score(F)

Table 15. Clustering – Word2Vector and Word2Vector-Average.

		Stemmed				Non-Stemmed			
Extract Features	DS*	A	P	R	F	A	P	R	F
Word2Vector	K-means (sci-L)	0.41	0.11	0.37	0.17	0.43	0.11	0.37	0.17
	K-means (NLTK)	**0.59**	0.23	0.63	0.34	0.43	0.11	0.37	0.17
	MiniBach	0.52	0.21	0.70	0.33	**0.57**	0.22	0.63	0.33
Word2Vector-Avg	K-means (sci-L)	0.41	0.11	0.37	0.17	**0.43**	0.11	0.37	0.17
	K-means (NLTK)	**0.59**	0.23	0.63	0.34	0.43	0.11	0.37	0.17
	MiniBach	**0.64**	0.25	0.57	0.34	0.60	0.24	0.63	0.35

Accuracy(A), Precision(P), Recall(R) and F1-Score(F)

4.5 Discussion and Conclusion

In some case we got a good accuracy but the F1-Score is very low. The explanation of this high accuracy is that the subset is very unbalanced, with more '*Low*' (simplification for '*Low-Medium*') than '*High'* risk labels. The clustering algorithm has high variation for the metrics results, showing that some good results are just random.

The testing with stemmed and non-stemmed, it is surprising that the PCA, SVD and t-SNE have almost the better accuracy in the case of non-stemmed version of corpus in both libraries. So, the process of stemming the words not improve the quality of clustering.

For K-means python libraries, normally are reference two implementations: NLTK and Scikit-Learn. Impressively, although use the same seed, the results are quite different. It happens because of the initial centroids are chosen different in the distinct libraries.

Also the 2-Grams results are better compared to 1-Gram or 3-Grams in the case of non-stemmed version. In contrast to the discussed results applying Word2Vec and Word2Vec-AVG results are better in the unigram mode with non-stemmed version of the corpus. This difference could because the word2vec algorithm produce vectors to represent the words with better characteristics, without be necessary consider the combination of 2 words.

Despite the 2-Grams give better results, when we add 3-Grams and N-Grams parameters the metrics are worst. So, the additional composition of words not always improve the result.

Another approaches for clustering, in future researches, like apply text summarization techniques or Latent Semantic Analysis (LSA) before the clustering could improve the results of clustering.

Finally, in contrast with classification method, none of clustering methods applied give a strong result. Some presented a good Accuracy, and Precision and Recall alternate the results. Like F1-Score is the result of combination of precision and recall, the result for this metric are always very low. And, considering this very imbalanced dataset, the metric F1-Score are relevant.

5 Bibliographies References

[1] IIA, *International Standards for the Professional Practive of Internal Auditing,* 2012.

[2] A. Gepp, M. K. Linnenluecke, T. J. O'Neill and T. Smith, *Big Data Techniques in Auditing Research and Pratice: Current Trends and Future Opportunities,* 2018.

[3] C. Brandas, M. Muntean and O. Didraga, *Intelligent Decision Support in Auditing: Big Data and Machine Learning Approach,* 2018.

[4] I. Fisher, M. Hughes and M. Garnsey, "State University of New York, USA," *Natural Language Processing in Accounting, Auditing and Finance: A Synthesis of the Literature with a Roadmap for Future Research,* 2016.

[5] M. Fissete, "Text Mining to Detect Indication of Fraud in Annual Reports Worldwide".

[6] S. Salloum, K. Shaalan and M. Al-Emran, "Using Text Mining Techniques for Extracting Information from Research Articles," 2018.

[7] C. Ramakrishnan, A. Patnia and G. Burns, "Layout-aware text extraction from full-text PDF of Scientif articles," 2012.

[8] G. Press, "Cleaning Big Data: Most Time-Consuming, Least Enjoyable Data Science Task, Survey Says," 2016. [Online]. Available: https://www.forbes.com/sites/gilpress/2016/03/23/data-preparation-most-time-consuming-least-enjoyable-data-science-task-survey-says/#38c6dbf96f63. [Accessed April 2020].

[9] W. Zhang, X. Tang and T. Yoshida, "A comparative study of TFIDF, LSI and multi-words for text classification," 2011.

[10] J. Lilleberg, Y. Zhu and Y. Zhang, "Support Vector Machines and Word2vec for Text Classification with Semantic Features," 2015.

[11] j. Howard and S. Ruder, "Universal Language Model Fine-tuning for Text Classification," 2018.

[12] S. Lai, L. Xu and J. Zhao, "Recurrent Convolutional Neural Networks for Text Classification," 2016.

[13] A. Conneau, H. Shwenk and Y. Cun, "Very Deep Convolutional Networks for Text Classification.," 2017.

[14] M. Munikar, S. Shakya and A. Shrestha, "Fine-grained Sentiment Classification using BERT," 2019.

[15] M. Polignamo, P. Basile and M. Gemmis, "ALBERTO: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets," 2019.

[16] A. Anand, G. Fogel, G. Pugalenthi and P. Suganthan, "An approach for classification of highly imbalanced data using weighting and undersampling," November 2010.

[17] C. Ling and C. Li, "Data Mining for Direct Marketing: Problems and Solutions," 1998.

[18] N. Chawla and K. Bowyer, "SMOTE: Synthetic Minority Over-sampling Technique," 2002.

[19] F. Vanbutsele, *The Impact of Big Data on Financial Statement Auditing,* 2018.

[20] S. VLuymans, "Learning from Imbalanced Data," 2018.

[21] R. Waykile and A. Thakare, "A Review Of Feature Extraction Methods For Text Classification," April 2018.

[22] H. Shahmohammadi, M. Mansoorizadeh and M. Dezfoulian, "An Extensive Comparison of Feature Extraction Methods for Paraphrase Detection," October 2018.

[23] N. Joselnon and R. Hallen, "Emotion Classification with Natural," 2019.

[24] M. Naili, A. Chaibi and H. Ghezala, "Comparative study of word wmbedding methods in topic segmentation," September 2017.

[25] T. Mikolov, k. Chen, G. Corrado and J. Dean, "Efficient Estimation of Word Representations in Vector Space," 2013.

[26] K. Djaballah, K. Boukhalfa and O. Boussaid, "Sentiment Analysis of Twitter Messages using Word2vec by Weighted Average," 2019.

[27] P. Bojanowski, E. Grave, A. Joulin and T. Mikolov, "Enriching Word Vectors with Subword Information," 2017.

[28] A. Joulin, E. Grave, P. Bojanowski and T. Mikolov, "Bag of Tricks for Efficient Text Classification," 2016.

[29] A. Joulin, E. Grave, P. Bojanowski, M. Douze, H. Jégou and T. Mikolov, "FASTTEXT.ZIP COMPRESSING TEXT CLASSIFICATION MODELS," 2016.

[30] Y. Goldberg and O. Levy, "word2vec Explained: Deriving Mikolov et al.'s Negative-Sampling Word-Embedding Method," 2014.

[31] J. Pennington, R. Socher and C. Manning, "GloVe: Global Vectors for Word Representation," 2016.

[32] J. Devlin, M.-W. Chang, K. Lee and K. Toutanove, "BERT: Pre-training of Deep Bidirectional Transformers for Language Undestanting," 2018.

[33] A. Vaswani, N. Shazeer, N. Parmar and J. Uszkoreit, "Attention is All You Need," 2017.

[34] N. Suchaud, "Difference betwenn Classical ML and Deep Learning," March 2018. [Online]. [Accessed April 2020].

[35] J. Han, M. Kamber and J. Pei, Data Mining Concepts and Techniques, 3 ed., Elsevier, 2012.

[36] A. Bakliwal and P. Arora, "Towards Enhanced Opinion Classification using NLP Techniques," 2010.

[37] F. E. Harrell, Regression Modeling Strategies, with applications, New York: Springer, 2015.

[38] H. Mathlete, "Introduction to Machine Learn with Random Forest," 2016. [Online]. Available: http://www.hallwaymathlete.com/2016/05/introduction-to-machine-learning-with.html.

[39] M. Nielsen, Neural Networkd and Deep Learning, Lambda, 2019.

[40] Y. LeCun, Y. Bengio and G. Hinton, "Deep Learning," *Nature,* May 2015.

[41] M. Stewart, "Simple Introductiokn do Convolutional Neural Networks," 27 Feb 2019. [Online]. Available: https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac. [Accessed 03 April 2020].

[42] K. Greff, R. Srivastava, J. Stuenebrink and J. Shmidhber, "LSTM: A Search Space Odyssey," 2017.

[43] C. Olah, "Understanding LSTM Networks.," August 2015. [Online].

[44] Facebook Open Source, "Wiki Word Vectors - fastText," 2016. [Online]. Available: https://fasttext.cc/docs/en/pretrained-vectors.html. [Accessed 2020].

[45] NILC - Núcleo Interinstitucional de Linguística Computacional, "Repositório de Word Embeddings do NILC," 2017. [Online]. Available: http://nilc.icmc.usp.br/nilc/index.php/repositorio-de-word-embeddings-do-nilc. [Accessed 2020].

[46] Google Research, "BERT (Bidirectional Encoder Representations from Transformers)," Oct 2018. [Online]. Available: https://github.com/tensorflow/models/tree/master/official/nlp/bert. [Accessed April 2020].

[47] Scikit-Learn, "Scikit-Learn: Machine Learn in Python.," [Online]. Available: https://scikit-learn.org/stable/. [Accessed April 2020].

[48] Keras.io, "Keras: The Python Deep Learning Library," [Online]. Available: https://keras.io/. [Accessed April 2020].

[49] Z. Yin and Y. Shen, "On the Dimensionality of Word Embedding," 2017.

[50] P. Zhou, Z. Qi, S. Zheng, J. Xu, H. Bao and B. Xu, "Text Classification Improved by Integrating Bidirectional LSTM with Two-dimensional Max Pooling," 2016.

[51] T. Young, D. Hazarika, S. Poria and E. Cambria, "Recent trends ins Deep Learning based Natural Language Proessing," 2018.

[52] W. Yin, K. Kann, M. Yu and H. Schütze, "Comparative Study of CNN and RNN for Natural Language Processing," 2017.

[53] J. Wei and K. Zou, "EDA: Easy Data Augmentation Techniques for Boosting Performance on Text Classification Tasks," 2019.

[54] NLTK Project, "Natural Language Toolkit," Mar 2020. [Online]. Available: https://www.nltk.org/. [Accessed April 2020].

[55] V. Powell, "Principal Component Analysis," 2015. [Online]. Available: https://setosa.io/ev/principal-component-analysis/. [Accessed April 2020].

[56] C. C. Aggarwal, Data Mining, The Textbook, 1a ed., Springer, 2015.

[57] G. Hinton and L. Maaten, "Visualizing Data using t-SNE," 2008.

[58] D. J. Kelleher and B. Tierney, Data Science (Essential Knowledge Series), New York: The MIT Press, 2018.

[59] V. Dhar, *Data Science and Prediction,* May 2012.

[60] S. Sayad, "An Introduction to Data Science," 2019. [Online].

[61] W. d. O. Bussab and P. A. Morettin, Estatística Básica, 6a ed., São Paulo: Saraiva, 2010, pp. 1-5.

[62] J. Tukey, Exploratory Data Analysis, London: Addison-Wesley, 1977.

[63] C.-h. Chen, W. Hardle and A. Unwin, Handbook of Data Visualization, Berlin, Germany: Springer, 2008.

[64] J. Bertin, Semiology of Graphics: Diagrams, Networks, Maps, Redlands, CA, USA: Esri Press, 2010.

[65] H. Kieran, Data Visualization: A Practical Introduction, Princeton University Press, 2018.

[66] H. Chen, R. Chiang and V. Storey, *Business Intelligence and Analytics: From Big Data to Big Impact,* 2012.

[67] Cetax, "Data Science, Big Data, Data Analytics," 2019. [Online]. Available: https://www.cetax.com.br/blog/data-science-vs-big-data-vs-data-analytics/.

[68] K. Yoon, *Big Data as Audit Evidence: Utilizing Wheather Indicators,* 2016.

[69] Q. Liu, *The Application of Exploratory Data Analysis in Auditing,* 2014.

[70] M. Alles and G. L. Gray, *Incorporating Big Data in audits: identifying inhibitors and a research agenda to address those inhibitors,* July 2016.

[71] D. Appelbaum, A. Kogan and M. A. Vasarhelyi, *Big Fata and Analytics in the Moderns Audit Engagement: Research Needs,* 2017.

[72] J. M. Bland and D. Altman, *Statistics Notes: Transforming Data,* 1996.

[73] E. G. M. Hui, Learn R for Applied Statistics, Data Visualization, Berkeley, CA, USA: Apress, 2018, pp. 129-172.

[74] SIGKDD Curriculum Committe, *Data Mining Curriculum: A proposal,* 2006.

[75] R. Nascimento, P. J. Santos, J. F. Santiago, B. C. Araújo, F. B. Lima and A. M. Maciel, *Mineração de Dados na Identificação de Empresas Irregulares Quanto ao Pagamento de Impostos,* 2018.

[76] P. Hajek and R. Henriques, *Mining corporate annual reports for intelligent detection of financial statement fraud – A comparative study of machine learning methods,* 2017.

[77] R. Bauder and T. Khoshfogtaar, *The Detection of Medicare Fraud Using Machine Learning Methods with Excluded Provider Labels,* 2017.

[78] O. S. Yee, S. Sagadevan and N. Malim, *Credit Card Fraud Detection Using Machine Learning As Data Mining Technique,* 2018.

[79] P. Domingos, *A Few Useful Things to Know about Machine Learning,* 2012.

[80] T. Sun and L. Sales, *Predicting Public Procurement Irregularity: An Application of Neural Networks,* 2018.

[81] P. E. Byrnes, *Developing Automated Applications for Clustering and Outlier Detection: Data Mining Implications for Auditing Practice,* 2015.

[82] R. Carvalho, L. Sales, H. Rocha and G. Mendes, *Using Bayesian Networks to Identify and Prevent Split Purchases in Brazil,* 2014.

[83] L. Sales and R. Carvalho, *Measuring the Risk of Public Contracts Using Bayesian Classifiers,* 2016.

[84] M. Bach, Z. Krstic, S. Seljan and L. Turulja, *Text Mining for Big Data Analysis in Financial Sector: A Literature Review,* 2019.

[85] D. Appelbaum, *Securing Big Data Provenance for Auditors: The Big Data Provenance Black Box ,* 2015.

[86] A. A. Shmais and R. Hani, *Data Mining for Fraud Detection.,* 2010.

[87] Association of Certified Fraud Examiners, "Global study onf occupational Fraud and Abuse," 2018.

[88] M. Montesdeoca, A. Medina and F. Santana, *Research Topics in Accounting Fraud in 21st Century: A State of Art.,* January 2019.

[89] J. M. Martinez, *O método científico na investigação de fraudes e irregularidades,* 2014.

[90] B. Kemper, *Principles of Exploratory Data Analysis in Problem Solving: What Can We Learn from a Well-Known Case?,* 2009.

[91] J. Dai, *Three Essays on Audit Technology: Audit 4.0, Blockchain, and Audit App,* 2017.

[92] J. Zheng, *Data Visualization in Business Intelligence,* 2017.

[93] C. P. Sanchez, P. L. Monelos and M. R. Lopez, *Does external auditing provide insights to detecting and evaluating financial distress? A comparative analysis of econometric models and artificial intelligence,* 2012.

[94] M. C. Aniceto, *Estudo Comparativo entre Técnicas de Aprendizado de Máquina para Estimação de Risco de Crédito,* 2016.

[95] J. C. Pacheco Jr., *Modelos para Deteção de Fraudes Utilizando Técnicas de Aprendizado de Máquinas,* 2019.

[96] Z. Rezaee, A. Dorestani and S. Aliabadi, *Application of Time Series Analyses in Big Data: Practical, Research, and Education Implications,* 2017.

[97] D. Chan and A. Kogan, *Chan, Data Analytics: Introduction to Using Analytics in Auditing, 2016,* 2016.

[98] IIA, "Data Analytics," 2017.

[99] J. Leite and A. Silva, *Computing prediction intervals with CAATTs,* 2018.

[100] M. Massaro, J. Dumay and J. Guthrie, *On the shoulders of giants: undertaking a structured literature review in accounting,* 2016.

[101] M. Gaber and E. Lusk, *Analytical Procedures Phase of PCAOB Audits: A Note of Caution in Selection The Forecasting Model,* 2017.

[102] Z. Rezaee, A. Dorestani and S. Aliabadi, "Universtiy of Memphis, USA," *Application of Time Series Analyses in Forensic Accounting,* 2018.

[103] M. Cao, R. Chychyla and T. Stewart, *Big Data Analytics in Financial Statement Audits,* 2015.

[104] J. Leskovec, A. Rajaraman and J. Ullman, Mining of Massive Datasets, Stanford University, California, USA, 2014.

[105] T. Sun and M. Vasarhelyi, "Rutgers, New Jersey, USA.," *Embracing Textual Data Analytics in Auditing with Deep Learning,* 2018.

[106] G. Boskou, E. Kirkos and C. Spathis, *Assessing Internal Audit with Text Mining,* 2018.

[107] AICPA, "Association of International Certified Professional Accountants," *Understanding the Entity and Its Environment and Assessing the Risks of Material Misstatement,* December 2018.

[108] Gartner, "Big Data," 2001. [Online]. Available: https://www.gartner.com/it-glossary/big-data/. [Accessed 2019].

[109] U. Fayyad, G. Shapiro and P. Smyth, Knowledge Discovery and Data Mining: Towards a Unifying Framework, University of California, Irvine, USA, 1996.

[110] S. Li, "Time Series Analysis and Forecasting with Python," Jul 2018. [Online]. Available: https://www.kdnuggets.com/2018/09/end-to-end-project-time-series-analysis-forecasting-python.html. [Accessed May 2019].

[111] L. Deng and Y. Liu, Deep Learning in Natural Language Processing, Seatle,USA: Springer, 2018.

[112] S. Bird, E. Klein and L. E., Natural Language Processing with Python, Sebastopol: O'Reilly, 2009.