UNIVERSIDADE DE CAXIAS DO SUL
CENTRO DE CIÊNCIAS EXATAS E DA TECNOLOGIA
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

FERNANDO ANDRIOLO ABEL

ANÁLISE TEXTUAL AUTOMÁTICA: APREENSIBILIDADE E QUALIDADE DA INFORMAÇÃO NA ÁREA DA SAÚDE

CAXIAS DO SUL - RS
2016

FERNANDO ANDRIOLO ABEL

ANÁLISE TEXTUAL AUTOMÁTICA: APREENSIBILIDADE E QUALIDADE DA INFORMAÇÃO NA ÁREA DA SAÚDE

Trabalho de Conclusão de Curso para obtenção do título de Bacharel em Ciência da Computação pela Universidade de Caxias do Sul, orientado pela Profa. Carine Geltrudes Webber.

Caxias do Sul - RS

2016

RESUMO

O acesso facilitado de pessoas e pacientes a fontes de informação sobre saúde ampliou a necessidade de que tais informações disponíveis sejam revisadas e analisadas, principalmente em contextos onde o paciente deve participar da decisão do seu tratamento. A análise textual é caracterizada pela apreensibilidade e a qualidade do conteúdo. A apreensibilidade de um texto trata da facilidade de compreensão. Existem diversas fórmulas utilizadas para avaliá-la (*Flesch Reading Ease*, *SMOG Index*, etc.). A qualidade textual, por sua vez, é abordada através de estudos realizados sobre Mineração de Textos. A Mineração de Textos caracteriza-se como um processo que contém diversas técnicas a fim de organizar, descobrir e extrair informações em bases de dados textuais de forma ágil e automática. O objetivo principal deste trabalho foi propor a concepção de uma ferramenta *web* para avaliar textos em português a partir da fórmula de apreensibilidade *Fernández-Huerta* e técnicas de classificação (*J48, Bayes Net, Naïve Bayes, Support Vector Machines, K-Nearest Neighbors e Multilayer Perceptron*). Para atingir o objetivo proposto, coletou-se uma amostra de dados textual composta por textos de *sites* sobre saúde na internet. O conjunto de dados foi dividido entre dados para treinamento e dados para testes. A fim de proceder com as análises, implementou-se uma ferramenta de plataforma *web* para apoiar tanto a análise de apreensibilidade quanto de qualidade da informação. Foram realizados testes com o software. Os resultados obtidos foram comparados com classificações realizadas por especialistas humanos. Identificou-se que o algoritmo *Naïve Bayes* apresentou os melhores resultados na classificação dos dados (89% de acerto). Como conclusão, considera-se que os resultados são promissores e evidenciam a viabilidade de uso de técnicas de aprendizado automático no tratamento de textos da área da saúde.

Palavras-chave: Avaliação textual. Apreensibilidade. Mineração de Textos. Aprendizado de Máquina. Weka.

ABSTRACT

The access of people and patients to health-related information on the Internet boosted the need for a review of the available information, especially when the patient needs to make decisions regarding his own treatment. A textual analysis consists of a readability and quality analysis of its content. Readability is the ease of understanding or comprehension of a text. There are several formulas used to measure it (Flesch Reading Ease, SMOG Index, etc.). Textual quality, on the other hand, can be evaluated through Text Mining studies. Text Mining is described as a process containing several techniques to organize, discover and extract information on text databases quickly and automatically. The mainly purpose of this paper is the development of a web tool capable of evaluating Portuguese texts using the Fernández-Huerta readability formula and classification techniques (J48, Bayes Net, Naïve Bayes, Support Vector Machines, K-Nearest Neighbors and Multilayer Perceptron). To reach the proposed goal, it was collected a text database composed of health-related online texts. The database collected was split into a training and test set. In order to proceed with analysis, a web tool was developed to support the readability and quality analysis. Tests were performed with the software. The results were compare with the texts classifications from human specialists. The Naïve Bayes technique showed the best results on data classification (89% of correct instances). In conclusion, the results are promising and evidence the viability of use of machine learning techniques on health-related texts.

Keywords: Textual analysis. Readability. Text Mining. Learning Machine. Weka.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE SIGLAS

Sigla	Significado
.NET	*DotNET*
API	*Application Programming Interface*
ARI	*Automated Readability Index*
DLL	*Dynamic-link Library*
FORCAST	Nome dos criadores da fórmula: Ford, Caylor, Sticht.
GNU	*General Public License*
JAR	*Java Archive*
KNN	*K-Nearest Neighbor*
MD	Mineração de Dados
MT	Mineração de Textos
MVC	*Model View Controller*
RI	Recuperação de Informações
ROUND	ROUND(X) - Fórmula utilizada para arredondar um valor X.
SMOG	*Simple Measure of Gobbledygook*
SQL	*Structured Query Language*
SVM	*Support Vector Machine*
URL	*Uniform Resource Locator*
US	United States
VP	Verdadeiro Positivo
VN	Verdadeiro Negativo
Weka	*Waikato Environment for Knowledge Analysis*
XML	*eXtensible Markup Language*

SUMÁRIO

1 INTRODUÇÃO

A tecnologia da informação vem auxiliando a medicina tanto para o seu desenvolvimento científico, quanto para a divulgação das informações sobre saúde. Com a expansão da Internet, ela se tornou uma das principais fontes de divulgação de informações, atingindo um amplo número de pessoas. No entanto, segundo profissionais da área da saúde, nem toda fonte de informação é capaz de abordar um tema de forma clara e correta.

O acesso facilitado de pessoas e pacientes a fontes de informação sobre saúde ampliou a necessidade de que tais informações disponíveis sejam revisadas e analisadas, principalmente em contextos onde o paciente deve participar da decisão do seu tratamento. Levando em consideração que, atualmente, a análise dessas informações disponíveis em Português na Internet necessita ser realizada manualmente por especialistas da área da saúde, é imprescindível que exista um instrumento que auxilie nessa tarefa de maneira automática, utilizando a tecnologia para tal.

1.1 PROBLEMA E QUESTÃO DE PESQUISA

Uma informação textual pode ser avaliada a partir de sua apreensibilidade e a qualidade do seu conteúdo. A apreensibilidade, ou *readability*, de uma informação textual é definida por Klare (1963) como sendo a facilidade do entendimento ou compreensão devido ao estilo da escrita. Desde então, diversas técnicas foram criadas a fim de avaliar a apreensibilidade de um texto, sendo em sua grande maioria, desenvolvidas e aplicadas exaustivamente na língua Inglesa. A avaliação da apreensibilidade representa uma análise quantitativa da informação, pois criam-se resultados mais objetivos, exatos e rígidos.

A aplicabilidade desses métodos para a língua Portuguesa deve ser investigada, tendo em vista que as técnicas mencionadas são baseadas na contagem de palavras, sílabas e frases de um determinado texto. Devido à origem greco-latina da língua Portuguesa, as suas palavras contêm, em média,

um número maior de sílabas do que palavras da língua Inglesa (MARTINS, NUNES, *et al.*, 1996).

Contudo, se for realizada apenas uma avaliação quantitativa sobre as informações textuais, não é possível avaliar a qualidade acerca do assunto abordado. Portanto, as técnicas de Mineração de Textos podem ser utilizadas para extrair padrões ou tendências de grandes volumes de textos, a fim de aproveitar ao máximo o conteúdo do texto (ARANHA e PASSOS, 2006).

A Mineração de Textos, pela visão de Barion e Lago (2008), surgiu da necessidade de descobrir, de forma automática, informações (padrões e anomalias) em dados não estruturados, sendo a maioria em formato textual. Já Feldman e Sanger (2007) a definem como um processo de conhecimento intensivo em que um usuário interage com uma coleção de documentos ao longo do tempo por meio de um conjunto de ferramentas de análise.

Tendo em mente o problema de pesquisa apresentado nesse tópico, tem-se a seguinte questão de pesquisa:

“De que forma uma ferramenta pode contribuir para a análise da apreensibilidade e qualidade de informações textuais na língua Portuguesa? ”

1.2 OBJETIVOS DO TRABALHO

O objetivo principal deste trabalho consiste na proposta, modelagem e desenvolvimento de um protótipo de ferramenta na plataforma *web* que permita avaliar informações textuais acerca de sua apreensibilidade e qualidade de forma automática.

De forma a atingir o objetivo principal desse trabalho, os seguintes objetivos específicos foram definidos:

a) Conhecer técnicas de avaliação da apreensibilidade (*readability*).
b) Conhecer técnicas de Mineração de Textos.
c) Desenvolver uma ferramenta na plataforma *web* que aplique as técnicas selecionadas.
d) Aplicar as técnicas sobre uma amostra de textos dos profissionais da área da saúde.
e) Obter os resultados produzidos pela ferramenta desenvolvida.

f) Comparar os resultados obtidos no estudo com outros disponíveis na literatura.
g) Avaliar os testes e avaliações da ferramenta sobre a amostra de textos.

1.3 ESTRUTURA DO TRABALHO

O presente trabalho está estruturado da seguinte forma: no Capítulo 2 são apresentadas informações sobre a avaliação textual, assim como a avaliação quantitativa e qualitativa, e os métodos disponíveis para a avaliação de cada um destes. No Capítulo 3 são apresentadas as características da ferramenta desenvolvida, tal como sua arquitetura, modelagem e resultados da mesma. No Capítulo 4 são apresentadas as considerações finais do trabalho.

2 AVALIAÇÃO TEXTUAL

A avaliação textual é um processo que consiste em extrair informações relevantes de dados em formato textual a fim de identificar características e vocabulário presente. Tais aspectos podem contribuir para a inferência da qualidade e facilidade de compreensão do texto. De forma a permitir o entendimento do conceito de avaliação textual, o presente capítulo está organizado em duas seções, que tratam de abordagens quantitativas e qualitativas de análise textual comumente empregadas no desenvolvimento de softwares.

2.1 AVALIAÇÃO QUANTITATIVA

Uma avaliação quantitativa consiste em uma avaliação que resulta em resultados numéricos como contagens ou medidas. Para realizarmos essa avaliação, utiliza-se do conceito de apreensibilidade. A apreensibilidade, ou *readability*, de uma informação textual é definida por Klare (1963) como sendo a facilidade do entendimento ou compreensão devido ao estilo da escrita.

A apreensibilidade foi definida por McLaughlin (1968, p. 188) como o grau ao qual uma dada população acha um certo texto compreensível e atrativo. Portanto, de outra forma pode-se dizer que quanto maior a apreensibilidade de um texto, mais claro ou entendível ele é para uma determinada população.

Os estudos sobre avaliação quantitativa de textos iniciaram-se por volta de 1920, quando educadores elaboraram uma maneira de utilizar características do vocabulário e do tamanho de frases para inferir o nível de dificuldade de um texto (DUBAY, 2004). Desde então, diversas fórmulas foram criadas a fim de avaliar a apreensibilidade textual, e segundo Dubay, pela década de 80, já havia no mínimo 200 fórmulas diferentes para avaliação textual, e mais de mil estudos publicados atestando a validade teórica dos modelos. Tais fórmulas são equações derivadas de uma análise de regressão (MCLAUGHLIN, 1969), que possibilitam encontrar uma relação razoável entre as variáveis de entrada (o texto) e saída (o resultado esperado). Uma vez que essas fórmulas, em sua grande maioria, foram desenvolvidas e aplicadas exaustivamente na língua

Inglesa, a aplicação destas para a língua Portuguesa será investigada ao final desse capítulo.

2.1.1 *Fry Graph Readability Formula*

Edward Fry criou um dos mais populares testes de avaliação da apreensibilidade utilizando um gráfico. A técnica conhecida como *Fry Graph Readability Formula* determina o grau de escolaridade de ensino nos Estados Unidos necessário para compreensão de uma amostra textual, baseado em um simples algoritmo. Primeiramente criada em 1963, o gráfico somente indicava o grau de escolaridade de alunos do ensino médio. Posteriormente, em 1969 e 1977, o gráfico foi ampliado para indicar também ensino médio e universitário respectivamente (FRY, 1963-1977 apud DUBAY, 2004, p. 44-45).

Para calcular, os seguintes passos devem ser realizados:

a) Seleciona-se amostras de 100 palavras do texto (Números não são considerados na contagem).
b) Encontra-se o número médio de frases das amostras de 100 palavras. (Eixo Y do gráfico)
c) Encontra-se o número médio de sílabas das amostras de 100 palavras. (Eixo X do gráfico).
d) Identifica-se o ponto onde as duas coordenadas se encontrem.
e) A área, ao qual o ponto foi adicionado, indica o grau de escolaridade aproximado necessário para a leitura e devida compreensão da amostra textual.
f) As áreas pertencentes aos níveis 13, 14 e 15, na Figura 1, indicam texto de nível universitário.

Para exemplificar, vamos calcular o resultado da técnica de Fry utilizando um trecho do trabalho de Dubay (2004):

> Some experts consider the number of morphemes for each 100 words to be a major contributor to semantic (meaning) difficulty and the number of Yngve word depths (branches) in each sentence to be a major contributor to syntactic (sentence) difficulty. One study (Coleman 1971) showed that Flesch's index of syllables for each 100 words correlates .95 with morpheme counts. Another study (Bormuth 1966)

> found that the number of words in each sentence correlates .86 with counts of Yngve word depths. Measuring the average number of syllables per word and the number of words in each sentence is a much easier method and almost as accurate as measuring morphemes and word depths.
> (DUBAY, 2004, p. 19)

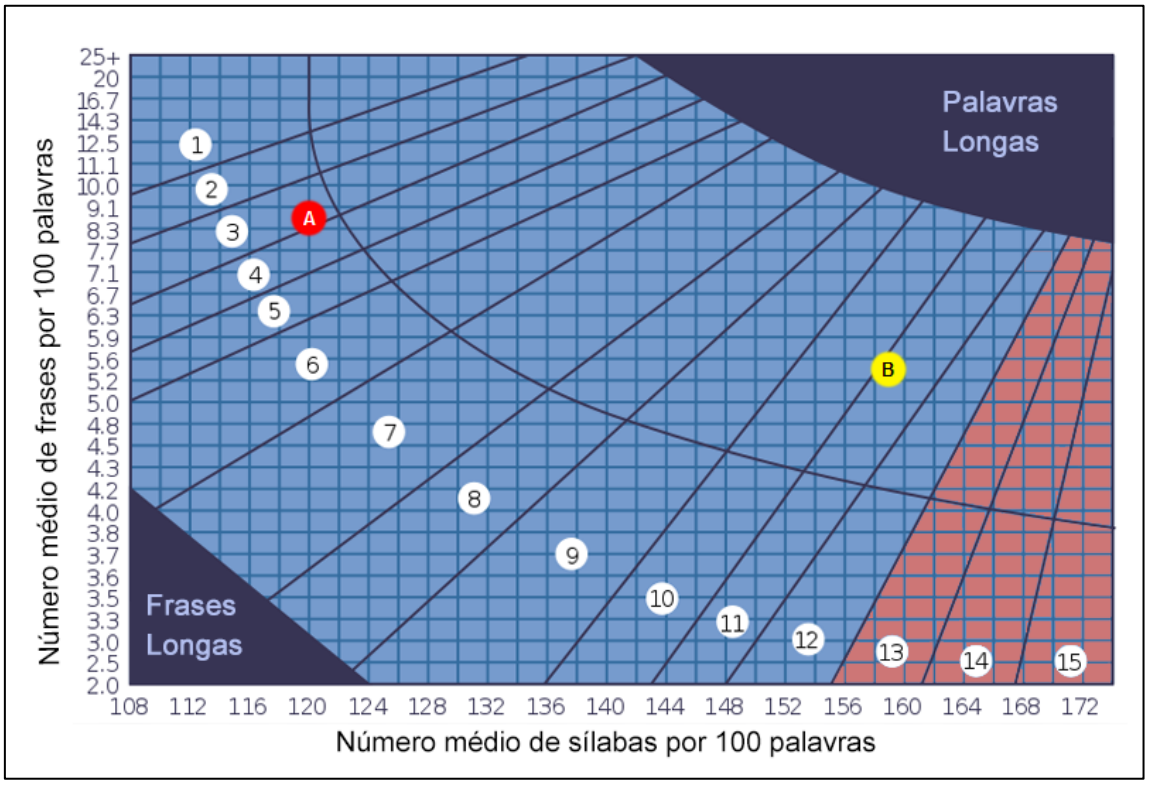
O número médio de sílabas (eixo X) por 100 palavras encontrado foi de 158, e o número médio de frases (eixo Y) por 100 palavras foi de 5,4. Analisando a Figura 1, pode-se perceber que a intersecção entre as duas coordenadas resultou em um ponto pertencente à zona de número 11, indicado pelo ponto “B”, classificando-se em um grau de escolaridade do texto em torno do 11º ano nos Estados Unidos.

No entanto, se utilizarmos um trecho do livro *The Little Prince* de Antoine de Saint-Exupéry:

> All men have stars, but they are not the same things for different people. For some, who are travelers, the stars are guides. For others they are no more than little lights in the sky. For others, who are scholars, they are problems... But all these stars are silent. You-You alone will have stars as no one else has them. In one of the stars I shall be living. In one of them I shall be laughing. And so it will be as if all the stars will be laughing when you look at the sky at night..You, only you, will have stars that can laugh! And when your sorrow is comforted (time soothes all sorrows) you will be content that you have known me... You will always be my friend. You will want to laugh with me. And you will sometimes open your window, so, for that pleasure... It will be as if, in place of the stars, I had given you a great number of little bells that knew how to laugh.
> (SAINT-EXUPÉRY, WOODS e HARCOURT, 1943, p. 59)

Para esse exemplo, o número médio de sílabas (eixo X) por 100 palavras foi calculado em 120, enquanto o número médio de frases (eixo Y) por 100 palavras encontrado foi de 8.5. Analisando novamente a Figura 1, pode-se notar pelo ponto “A”, que o texto, retirado de um livro infantil, foi classificado como pertencente à zona de número 3, indicando um grau de escolaridade referente ao 3º ano de escolaridade nos Estados Unidos.

Figura 1 - Fry Graph para estimativa de nível de escolaridade



Fonte: https://en.wikipedia.org/wiki/File:Fry_Graph_SVG.svg, modificado pelo autor.

A técnica de *Fry Graph Readability Formula* possui algumas limitações, quando utilizado textos com palavras ou frases excessivamente longas. Dessa forma, os resultados pertencentes às zonas “Palavras Longas” ou “Frases Longas”, conforme Figura 1, são inconclusivos.

Por exemplo, uma amostra de texto formada por um número médio de frases por 100 palavras igual à 3.0 frases e um número médio de sílabas por 100 palavras de 119 sílabas é pertencente à zona de “Frases Longas”, gerando um resultado inconclusivo sobre o grau de escolaridade necessário para a devida compreensão.

O mesmo resultado inconclusivo é obtido para uma amostra de texto formada por um número médio de frases por 100 palavras igual à 20 frases e um número médio de sílabas por 100 palavras de 164 sílabas, pertencente à zona de “Palavras Longas”.

2.1.2 *Gunning-Fog Index*

Em torno de 1930, educadores começaram a notar que estudantes do ensino médio possuíam dificuldades na leitura. Robert Gunning, por sua vez, percebeu que muito desse problema de leitura era, na verdade, um problema de escrita. Notou que jornais e livros eram escritos com uma complexidade desnecessária. Surgiu daí o termo *fog* (tradução para névoa ou neblina) de sua fórmula *Gunning-Fog Index*, também conhecida por *Fog Index*, publicada em 1952 (GUNNING, 1952 apud DUBAY, 2004, p. 24).

A fórmula de *Gunning-Fog Index* é construída com o seguinte método:

a) Seleciona-se um trecho do texto com cerca de 100 palavras.
b) Conta-se o número de palavras e o número de frases.
c) Conta-se o número de palavras complexas, ou seja, palavras que contenham três ou mais sílabas (polissílabas).
d) Aplica-se a seguinte fórmula:

$$Score = (ASL + PHW)\ x\ 0{,}4$$

Onde:

$$ASL = \frac{Número\ de\ Palavras}{Número\ de\ Frases}$$

$$PHW = \left(\frac{Número\ de\ Polissílabas}{Número\ de\ Palavras}\right)\ x\ 100$$

Com o resultado (*score*), pode-se indicar o grau de escolaridade necessário para o entendimento do texto utilizando a Tabela 1. Textos indicados para uma ampla audiência, pela fórmula de Gunning, situam-se em um resultado menor que 12. Textos indicados para um entendimento universal geralmente necessitam um resultado menor que 8.

Tabela 1 - Relação entre grau de escolaridade e resultados do Gunning-Fox Index

Resultado do Gunning-Fog Index	Grau de Escolaridade Estimado
6	Sexto ano
7	Sétimo ano
8	Oitavo ano
9	Ensino Médio, 1º ano
10	Ensino Médio, 2º ano
11	Ensino Médio, 3º ano
12	Ensino Médio, 4º ano
13	Universidade, 1º ano
14	Universidade, 2º ano
15	Universidade, 3º ano
16	Universidade, 4º ano
17+	Graduação Concluída

Fonte: Gunning (1952 apud DUBAY, 2004, p. 24)

2.1.3 *Flesch Reading Ease*

O mais importante estudo sobre apreensibilidade textual surgiu de Rudolf Flesch, e garantiu um grande avanço nas pesquisas sobre avaliação quantitativa textual. Em sua dissertação de doutorado, Flesch (1943) publicou a sua primeira fórmula para avaliar a apreensibilidade de um texto, popularizando-se entre os interessados de diversos meios da comunicação.

Com o objetivo de tornar mais fácil a utilização da fórmula, Flesch (1948) publicou uma revisão da fórmula, que passou a contar com apenas duas variáveis, sendo elas o número de sílabas do texto e o número de frases para cada amostra de 100 palavras. Os resultados da fórmula são indicados numa escala de 0 (difícil) a 100 (fácil), onde um resultado maior significa uma maior apreensibilidade do texto. A fórmula pode trazer resultados negativos, sendo esses considerados como 0 na escala.

A fórmula de *Flesch Reading Ease* foi estabelecida como:

$$Score = 206{,}835 - (1{,}015\ x\ ASL) - (84{,}6\ x\ ASW)$$

Onde:

$$ASL = \frac{Número\ total\ de\ palavras}{Número\ total\ de\ frases}$$

$$ASW = \frac{Número\ total\ de\ sílabas}{Número\ total\ de\ palavras}$$

A Tabela 2 descreve como o resultado da fórmula (*score*) pode ser interpretado em termos de apreensibilidade (FLESCH, 1949, p. 149).

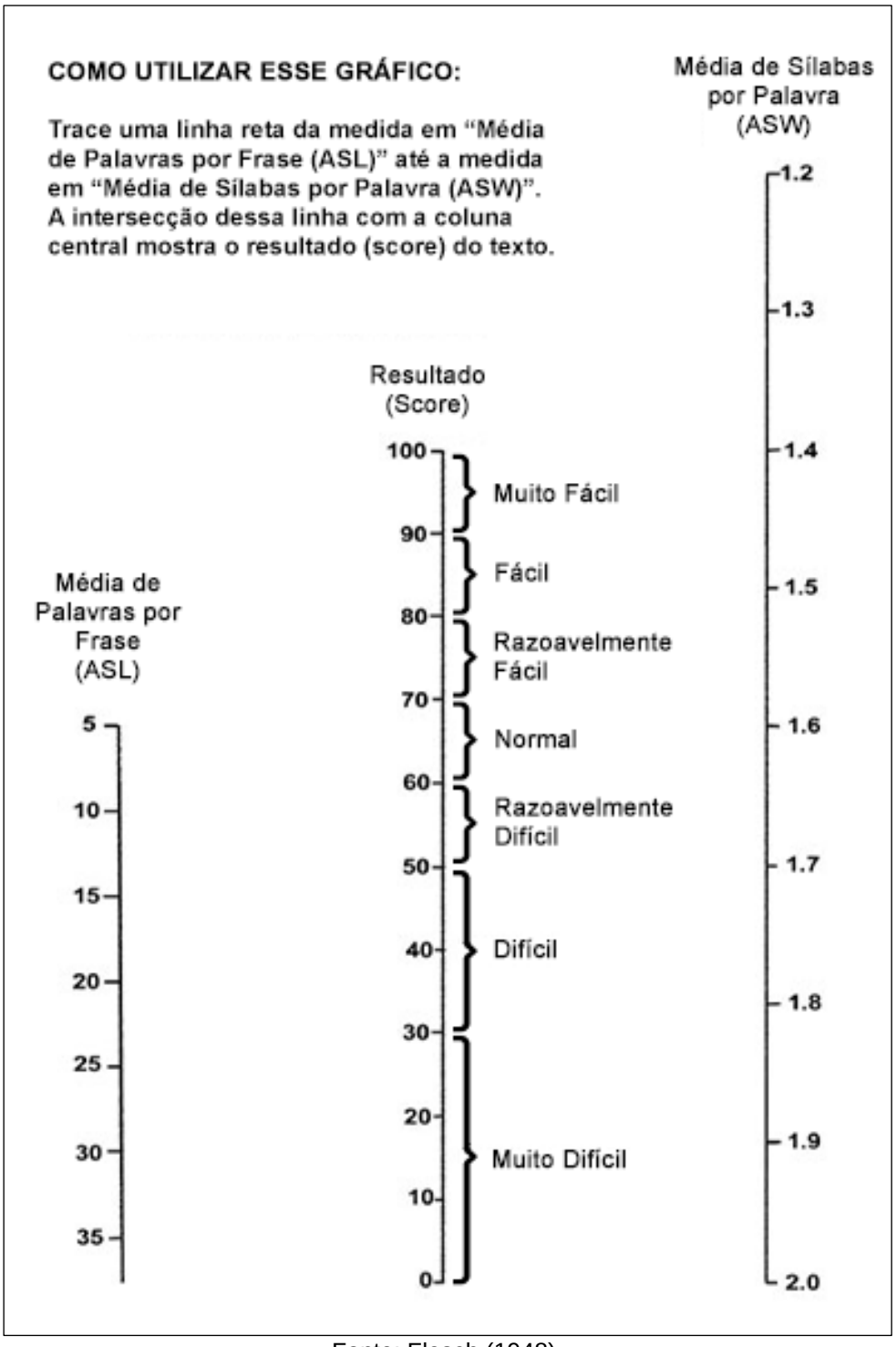
Tabela 2 - Relação entre o grau de escolaridade e resultados do Flesch Reading Ease

Resultado (*score*)	**Nível de Apreensibilidade**	**Grau de Escolaridade Estimado**	**Percentual Estimado de Adultos nos EUA (1949)**
90 a 100	Muito fácil	5º ano	93%
80 a 89	Fácil	6º ano	91%
70 a 79	Razoavelmente fácil	7º ano	88%
60 a 69	Normal	8º e 9º anos	83%
50 a 59	Razoavelmente difícil	10º a 12º ano	54%
30 a 49	Difícil	13º a 16º ano	33%
0 a 29	Muito difícil	Universitário	4,5%

Fonte: Flesch (1949, p. 149)

Rudolf Flesch também realizou a criação de um gráfico, ilustrado na Figura 2, utilizado para facilitar a busca pelo resultado de sua fórmula *Flesch Reading Ease*. Para isso, é necessário o conhecimento das duas variáveis utilizadas na fórmula, *ASL* e *ASW*, e aplicar as instruções descritas.

Figura 2 - Avaliação da Flesch Reading Ease de um texto pelo gráfico



Fonte: Flesch (1948).

A fórmula *Flesch Reading Ease*, dentre as formulas levantadas nesse capítulo, foi a única em que foram encontradas adaptações à outras línguas. Ocorreram adaptações para as línguas Espanhola (fórmula de *Fernández-*

Huerta), Francesa (fórmula de *Kandel & Moles*) e Holandesa (fórmula de *Douma*).

A fórmula de *Fernández-Huerta* apresenta os resultados no mesmo formato da fórmula *Flesch Reading Ease*, e classifica também o grau de escolaridade do texto (HUERTA, 1959 apud BARBOZA e NUNES, 2007). No entanto, a sua fórmula possui algumas mudanças quanto às suas variáveis, sendo estruturada como:

$$Score = 206{,}84 - (0{,}60\ x\ ASW) - (1{,}02\ x\ ASL)$$

Onde:

$$ASW = \left(100 * \frac{Número\ total\ de\ sílabas}{Número\ total\ de\ palavras} \right)$$

$$ASL = \left(100 * \frac{Número\ total\ de\ frases}{Número\ total\ de\ palavras} \right)$$

O resultado (*score*) pode então ser aplicado à Tabela 2, para nos indicar a apreensibilidade e o grau de escolaridade.

2.1.4 *Automated Readability Index*

O método *Automated Readability Index* (ARI) foi criado por Senter e Smith (1967) para a força aérea dos Estados Unidos como uma alternativa à métodos que utilizavam a contagem de sílabas por palavra para medir a apreensibilidade. Segundo os autores, membros de uma classe de 65 alunos foram instruídos a contar as sílabas de um texto. Esse texto continha 169 sílabas, no entanto o resultado da contagem das sílabas apresentou uma grande variância nos valores, e consequentemente, uma apreensibilidade não confiável.

O método utiliza da contagem de caracteres por palavra e número de palavras por sentença, e tem como resultado aproximado o grau de escolaridade, assim como Flesch (SENTER e SMITH, 1967).

A fórmula *Automated Readability Index* é indicada por:

$$Score = 4{,}71\left(\frac{Número\ de\ Caracteres}{Número\ de\ Palavras}\right) + 0{,}5\left(\frac{Número\ de\ Palavras}{Número\ de\ Frases}\right) - 21{,}43$$

O resultado (*score*) da fórmula deve ter o seu valor sempre arredondado para cima (SENTER e SMITH, 1967), e então com base na Tabela 3, pode-se identificar o grau de escolaridade e a idade aproximada do texto.

Tabela 3 - Relação entre grau de escolaridade e resultados da Automated Readability Index

Resultado (*score*)	**Idade Aproximada**	**Grau de Escolaridade Estimado**
1	5 a 6 anos	Jardim de Infância
2	6 a 7 anos	1º ano
3	6 a 8 anos	2º ano
4	8 a 9 anos	3º ano
5	9 a 10 anos	4º ano
6	10 a 11 anos	5º ano
7	11 a 12 anos	6º ano
8	12 a 13 anos	7º ano
9	13 a 14 anos	8º ano
10	14 a 15 anos	9º ano
11	15 a 16 anos	10º ano
12	16 a 17 anos	11º ano
13	17 a 18 anos	12º ano
14 ou mais	18 a 22 anos	Universidade

Fonte: Senter e Smith (1967, p. 10)

2.1.5 *Flesch Kincaid Grade Level*

A partir de um estudo comissionado pela Marinha dos Estados Unidos, Kincaid et al. (1975) recalcularam antigas fórmulas de apreensibilidade, tais como *ARI*, *Flesch* e *Fog Index*, com o objetivo de convertê-las em fórmulas de medição do grau de escolaridade e, então, aplicá-las aos manuais de treinamento da Marinha.

A fórmula resultante da conversão da fórmula de *Flesch Reading Ease* é atualmente conhecida como *Flesch-Kincaid formula*:

$$Score = (0{,}39\ x\ ASL) + (11{,}8\ x\ ASW) - 15{,}59$$

Onde:

$$ASL = \frac{Número\ total\ de\ palavras}{Número\ total\ de\ frases}$$

$$ASW = \frac{Número\ total\ de\ sílabas}{Número\ total\ de\ palavras}$$

O resultado (*score*) da fórmula representa aproximadamente o grau de escolaridade necessário para o entendimento do texto, considerando se tratar de níveis de escolaridade dos Estados Unidos. A apresentação do resultado dessa forma facilita o cálculo para professores, pais, bibliotecários e interessados em avaliar a apreensibilidade de livros e textos para os estudantes.

2.1.6 *The New Dale-Chall Readability Formula*

Inspirados pela *Flesch Reading Ease Formula* de Rudolph Flesch, Edgar Gale e Jeane Chall criaram a fórmula *The Dale-Chall Readability* com o objetivo de aperfeiçoar a fórmula de Flesch. Diferente de outras fórmulas, a fórmula de *Dale-Chall* leva em consideração para o cálculo uma lista de 763 palavras consideradas fáceis e familiares à 80% de estudantes do quarto ano dos Estados Unidos (DALE e CHALL, 1948). Essa lista de palavras foi posteriormente revisada em 1995, e aperfeiçoada para uma lista de 3000 palavras, modificando o nome da fórmula para *The New Dale-Chall Readability Formula* (CHALL e DALE, 1995).

Em vista disso, pode-se dizer que a fórmula de Dale e Chall é baseada no princípio que, um texto escrito com palavras familiares torna-se mais apreensível à leitura e consequente entendimento.

A fórmula de Dale e Chall aplica-se da seguinte maneira:

$$Score\ = (0{,}1579\ *\ PPD) + (0{,}0496\ *\ AVL) +\ 3{,}6365$$

Onde:

$$PPD = \frac{\begin{matrix} Número\ de\ palavras \\ fora\ da\ lista \end{matrix}}{Número\ total\ de\ palavras}$$

$$AVL = \frac{Número\ total\ de\ palavras}{Número\ total\ de\ frases}$$

Com o resultado (*score*) representando o grau de escolaridade necessário para o entendimento do texto, podemos utilizar a Tabela 4 para nos indicar o grau de escolaridade necessário para o entendimento do texto.

Tabela 4 - Resultados da fórmula New Dale-Chall com as idades e graus de escolaridade

Resultado (score)	**Grau de Escolaridade**	**Nível de Ensino**	**Idade Aproximada**
4,9 ou menos	1º, 2º, 3º e 4º anos	Nível Primário	5 a 10 anos
5,0 até 5,9	5º e 6º anos		10 a 12 anos
6,0 até 6,9	7º e 8º anos	Nível Secundário	12 a 14 anos
7,0 até 7,9	9º e 10º anos		14 a 16 anos
8,0 até 8,9	11º e 12º anos		16 a 18 anos
9,0 até 9,9	13º, 14º e 15º	Nível Universitário	18 a 22 anos
10 ou mais	16º ano ou superior		22 anos ou mais

Fonte: Dale e Chall (1948, p. 18)

2.1.7 *Coleman-Liau Index*

Desenvolvido por Coleman e Liau (1975) para determinar um grau de escolaridade aproximado para o entendimento de um texto. Tinham como objetivo fornecer uma facilidade à *U.S. Office of Education* que permitisse a mensuração da apreensibilidade de todos os livros do sistema público de ensino, a fim de melhorar a compreensão dos alunos. Assim como Senter e Smith (criadores da fórmula ARI), Coleman e Liau também entendiam que avaliações

computadorizadas tem maior facilidade e precisão na contagem de caracteres por palavra, do que sílabas por palavra.

A fórmula de *Coleman-Liau Index* se dá, então, pela contagem média de caracteres e frases para cada 100 palavras:

$$Score = (0{,}0588\ x\ L) - (0{,}296\ x\ S) - 15{,}8$$

Onde:

$$L = \left(\frac{Número\ total\ de\ caracteres}{Número\ total\ de\ palavras}\right) * 100$$

$$S = \left(\frac{Número\ total\ de\ frases}{Número\ total\ de\ palavras}\right) * 100$$

O resultado (*score*) da fórmula aproxima o grau de escolaridade requerido para que um leitor compreenda o texto. Por exemplo, um resultado de 10.6 significaria que o texto é apropriado para um estudante do 10º ou 11º ano. Para realizarmos essa comparação, pode-se utilizar a mesma Tabela 3, utilizada para a fórmula ARI.

2.1.8 *SMOG Grading*

A fórmula de *SMOG Grading* (“*Simple Measure of Gobbledygook*”) é descrita pelo seu criador McLaughlin (1969) como um método de medida da apreensibilidade textual, que apesar de ser “comicamente” simples, é de fato mais rápida e válida do que outras técnicas até então criadas. Criada para substituir a fórmula de *Gunning-Fog Index* por apresentar maior facilidade e precisão nos resultados.

A fórmula de *SMOG Grading* pode ser calculada conforme a seguir:

a) Extrai-se trinta frases do texto, sendo dez frases consecutivas do início do texto, dez frases consecutivas do meio do texto e outras dez consecutivas do final do texto.

b) Conta-se o número de palavras no texto extraído que contenham três ou mais sílabas, denominando-as polissílabas.
c) Aplica-se a seguinte fórmula:

$$Score = 3 + ROUND\ (\sqrt{Número\ total\ de\ polissílabas}\ )$$

O resultado (*score*) determina o número estimado de anos necessários de estudo para entender completamente um texto, e é considerada apropriada para avaliação do grau de escolaridade entre 4º ano até nível universitário nos Estados Unidos. Dessa forma, McLaughlin exemplifica os resultados de sua fórmula conforme demonstrado na Tabela 5.

Tabela 5 - Resultados da fórmula SMOG Grading

Resultado (*score*)	**Grau de Escolaridade Estimado**
4	4º ano
5	5º ano
6	6º ano
7	7º ano
8	8º ano
9	9º ano
10	10º ano
11	11º ano
12	12º ano
13	Nível Universitário
14	
15	
16	
17	Nível Universitário Completo
18	
19+	Qualificação Profissional Superior

Fonte: McLaughlin (1969)

2.1.9 *FORCAST formula*

Os homens que ingressavam nas Forças Armadas Americanas mostravam variados níveis de habilidades de leitura. Dessa forma, era essencial que houvesse uma forma de avaliar a apreensibilidade do material de trabalho do exército, para que os homens pudessem ser destinados à serviços onde fossem melhor aproveitados (CAYLOR, STICHT, *et al.*, 1973).

A fórmula *FORCAST* foi desenvolvida por um estudo comissionado pelas Forças Armadas Americanas, com o intuito de medir a apreensibilidade de todo material de trabalho do exército. Devido ao fato desse material possuir, em geral, uma linguagem muito técnica e ser destinado especificamente à adultos, a maioria das fórmulas já existentes até esse momento eram inapropriadas, por terem sido aplicadas e desenvolvidas para o ensino público infantil (CAYLOR, STICHT, *et al.*, 1973).

Por possuir a peculiaridade de não utilizar tamanho de frases do texto para a sua medição, a fórmula de *FORCAST* é a favorita para medir apreensibilidade de textos sem frases completas como curtos enunciados, textos na Web e formulários (DUBAY, 2004, p. 48).

A fórmula de *FORCAST* utiliza somente uma variável que representa o número de palavras monossilábicas de uma amostra de 150 palavras:

$$Score = 20 - \left(\frac{\begin{matrix} Número\ de\ palavras\ monossilábicas \\ de\ uma\ amostra\ de\ 150\ palavras \end{matrix}}{10} \right)$$

O resultado (*score*) da fórmula representa aproximadamente o nível de escolaridade necessário para o entendimento do texto. A fórmula apenas retorna níveis de escolaridade entre o 5º ano e o 12º ano do ensino dos Estados Unidos.

2.1.10 Aplicabilidade das fórmulas à língua Portuguesa

Conforme já mencionado, as fórmulas de apreensibilidade apresentadas, em sua grande maioria, foram desenvolvidas e validadas originalmente para a língua Inglesa. Devido às diferenças entre as línguas Inglesa e Portuguesa, nesse tópico é investigado a aplicabilidade das fórmulas à língua Portuguesa.

Martins et al. (1996) aplicaram e validaram as fórmulas de *Flesch Reading Ease*, *Coleman-Liau Index*, *Flesch Kincaid Grade Level* e *ARI* em variados trechos de livros empregados nas escolas, sendo eles de ensino fundamental, ensino médio e ensino superior. Os resultados mostraram-se semelhantes, e por isso optou-se pela utilização da fórmula de *Flesch Reading Ease*.

Nos testes do estudo, aplicou-se a fórmula em livros na língua Inglesa, e nesses mesmos livros traduzidos para a língua Portuguesa. O resultado dessa comparação direta mostrou que os textos em Português apresentaram, em média, escores com 42 pontos a menos que os textos em Inglês. Em síntese, utilizou-se da fórmula de *Flesch Reading Ease* padrão, e então somou-se 42 pontos aos resultados para avaliar a apreensibilidade na língua Portuguesa (MARTINS, NUNES, *et al.*, 1996).

Goldim (2006), por sua vez, validou as fórmulas de *Flesch Reading Ease* e *Flesch Kincaid Grade Level* aplicando-as em publicações de jornais selecionadas aleatoriamente, mas garantindo que pudesse conter desde as reportagens mais simples, como as de esportes, até as mais elaboradas, como os editoriais e crônicas. Em complemento a esses testes, um fragmento de William Shakespeare foi submetido às fórmulas em sua forma original (inglês) e em sua forma traduzida (português), e os resultados de ambos se mantiveram constantes.

Finalmente, Barboza e Nunes (2007) desenvolveram um estudo em que era necessário a utilização de uma fórmula de apreensibilidade para a língua Portuguesa, e para o mesmo, selecionou-se pela adaptação da fórmula *Flesch Reading Ease* para a língua Espanhola, índice de *Fernández-Huerta*.

2.2 AVALIAÇÃO QUALITATIVA

Uma avaliação quantitativa consiste em uma avaliação cujo objetivo é a mensuração da qualidade e validade de um texto. Ou seja, avaliar um texto acerca das suas informações, podendo estas mostrarem uma relevância maior ou menor ao conteúdo abordado. E para a realização dessa avaliação, utilizaremos dos conceitos da Mineração de Textos e do Processamento de Linguagem Natural.

2.2.1 Mineração de Textos

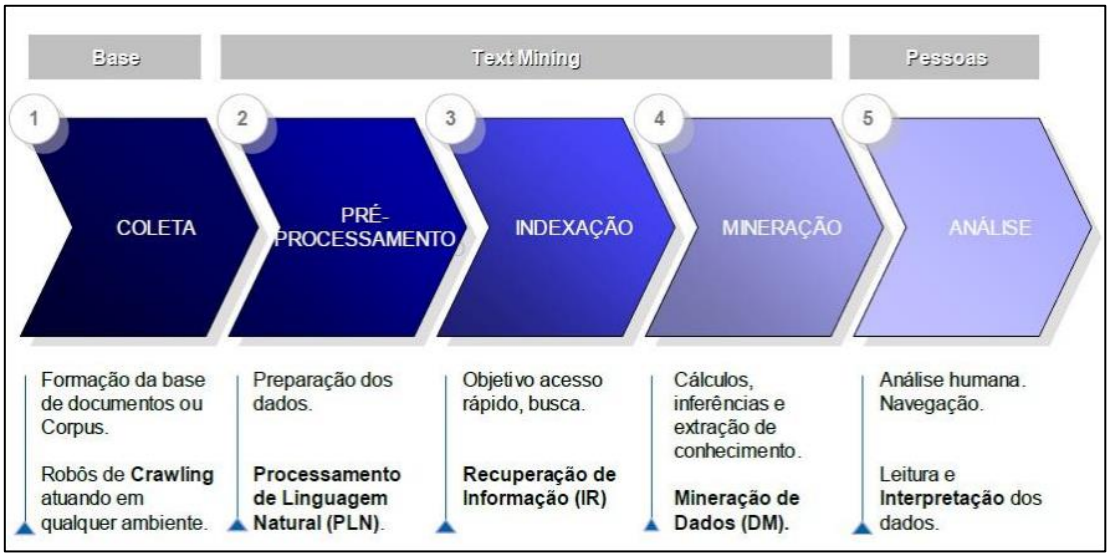
A Mineração de Textos (MT) surgiu da necessidade de extrair conhecimento não conhecido previamente, a partir de fontes textuais úteis para tomar decisões cruciais de negócio (ZANASI, 1997 apud ARANHA e PASSOS, 2006, p.2). Inspirado pela mineração de dados (MD), que procura descobrir padrões emergentes de bancos de dados estruturados, a MT pretende extrair informações úteis de dados não estruturados ou semiestruturados (ARANHA e PASSOS, 2006, p. 1).

Segundo Silva (2010), os dados estruturados são dados que estão armazenados utilizando um modelo de armazenamento previamente conhecido, enquanto que dados não estruturados são livres de qualquer formato ou padrão de armazenamento, exatamente como os textos. Os dados semiestruturados são dados textuais livres formatados a alguma estrutura, como por exemplo, os arquivos XML.

A área de MT é considerada multidisciplinar, tendo em vista que as definições e técnicas acerca de si variam conforme o campo de atuação dos seus autores. Portanto, é possível notar que as técnicas mais expressivas da MT têm origem nas áreas da Estatística, Redes Neurais, Aprendizado de Máquina e Banco de Dados (CAMILO e SILVA, 2009).

De acordo com Aranha (2007, p. 19), o processo de MT divide-se em cinco etapas e constitui-se conforme o diagrama ilustrado na Figura 3.

Figura 3 - Etapas do processo de Mineração de Textos



Fonte: Aranha (2007).

2.2.2 Coleta

A etapa de coleta dos dados tem como função fundamental a formação de uma base de dados textual (ARANHA, 2007). De acordo com Aranha e Passos (2006), a base de dados, também conhecida como *corpus*, é composta por e-mails, textos livres obtidos por resultados de pesquisas, documentos, páginas da Web e até campos textuais em bancos de dados. De forma sucinta, todos os dados podem ter origem de basicamente três ambientes distintos: um diretório de pastas do disco rígido, tabelas de um banco de dados e a Internet (ARANHA, 2007).

Segundo Aranha (2007, p. 42), independente do ambiente utilizado para a coleta, a obtenção dos dados textos é responsabilidade de robôs denominados *crawler*, que realizam uma varredura de forma sistemática, autônoma e exploratória das informações da rede.

Geralmente, os repositórios de dados possuem milhares de registros. Nesse contexto, é inviável que todos os registros sejam utilizados para a construção de um modelo de Mineração de Dados. Portanto, os dados são divididos, conforme Camilo e Silva (2009, p. 7), em três conjuntos:

1. Conjunto de Treinamento (*Training Set*): conjunto de registros submetidos ao modelo desenvolvido;

2. Conjunto de Testes (*Test Set*): conjunto de registros usados para testar o modelo construído;
3. Conjunto de Validação (*Validation Set*): conjunto de registros usados para validar o modelo construído;

A principal vantagem da divisão dos dados em conjuntos menores, é tornar o modelo não dependente de um conjunto de dados específico, evitando que ao ser submetido a outros conjuntos de dados, apresente valores insatisfatórios (CAMILO e SILVA, 2009).

2.2.3 Pré-Processamento

Na etapa de pré-processamento, visa-se organizar os textos coletados, que nesse momento ainda são dados não estruturados, e transformá-los em uma representação estruturada adequada para ser submetida à próxima etapa de Indexação ou Mineração (ARANHA, 2007). Além de organizar os dados, a etapa de preparação tem outras funções que vão desde a limpeza dos dados, responsável pela remoção de erros ortográficos e inconsistências que influenciariam negativamente no resultado, até funções que reduzem a dimensionalidade do problema (CAMILO e SILVA, 2009). Portanto, nesse capítulo são descritas algumas técnicas sobre os dados, que normalmente são aplicadas a fim de preparar os dados para a etapa de mineração.

2.2.3.1 *Tokenization*

Tem como objetivo a identificação de todas as palavras (termos) do documento. Além disso, nessa técnica ainda podem ser contemplados a remoção de hifens, caracteres inválidos, pontuação e acentos (SILVA, 2010), e também, possíveis erros de ortografia das palavras podem ser eliminados com a utilização de um dicionário de palavras (ARANHA, 2007).

2.2.3.2 Remoção de *stopwords*

Consiste na eliminação de termos da língua Portuguesa que não são utilizados na análise textual. Esses termos são compostos por artigos, preposições, conjunções, números, símbolos, entre outros e são denominados *stopwords*. As *stopwords* são palavras comuns, que se repetem diversas vezes num texto e não acrescentam informações relevantes para a extração do conhecimento (SILVA, 2010).

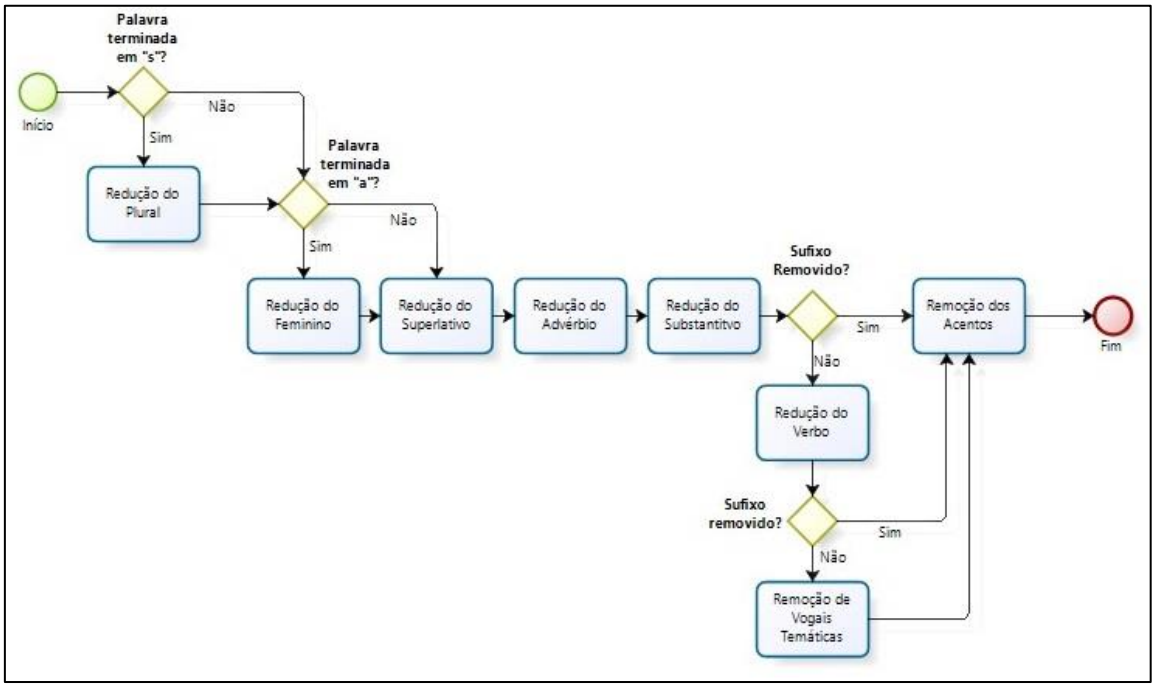
2.2.3.3 *Stemming*

A técnica conhecida como *stemming* tem como principal objetivo reduzir o número de palavras únicas (SOLKA, 2008, p. 96). Em outras palavras, *stemming* é o processo de remover todos os prefixos, sufixos, plurais, gerúndios e gêneros, permitindo que sejam agrupadas as palavras com significados semelhantes pelo seu radical (*stem*, no sentido de “raíz”).

A grande vantagem dessa técnica se concentra em reduzir o número de termos indexados, tendo em vista que todos os termos similares são mapeados como uma única entrada. Assim, evita-se que um termo relevante para o conhecimento não seja levado em consideração para a análise, simplesmente por ter sido escrito de diferentes maneiras (SILVA, 2010). Por exemplo, as palavras “médico”, “medicina”, “medicinal” poderão ser agrupadas pelo seu radical “medic”.

Um algoritmo de *stemming* para a língua Portuguesa é apresentado por Orengo e Huyck (2001), e possui 8 etapas, destacadas em azul pela Figura 4. As etapas do algoritmo de Orengo e Huyck não são detalhadas nesse trabalho.

Figura 4 - Algoritmo de Stemming para a língua Portuguesa



Fonte: Orengo e Huyck (2001).

2.2.3.4 Seleção dos termos mais relevantes

Devido ao efeito negativo que termos irrelevantes podem causar aos resultados de algoritmos de Aprendizado de Máquina, é comum preceder o aprendizado com uma etapa de seleção de atributos, que procura eliminar todos, senão os termos mais relevantes (WITTEN e FRANK, 2005).

A melhor maneira para realizar a seleção dos termos relevantes ainda é a manual, baseado no profundo conhecimento do problema e dos termos mais significativos. No entanto, existem abordagens automáticas, utilizando os algoritmos de Aprendizado de Máquina, que são posteriormente aplicados na etapa de Mineração (WITTEN e FRANK, 2005).

Pode-se por exemplo, aplicar um algoritmo de árvore de decisão no conjunto completo dos dados, e então selecionar os termos que apareceram na árvore, considerando que todos os termos não presentes na mesma são considerados irrelevantes. Apesar de parecer de grande utilidade, selecionar os termos utilizando o algoritmo de árvore de decisão não representa um grande efeito se, posteriormente, aplicarmos novamente o mesmo algoritmo para a mineração. Contudo, se aplicado um algoritmo diferente do aplicado na seleção

dos termos, teremos um efeito positivo no resultado final (WITTEN e FRANK, 2005).

2.2.4 Indexação

A indexação de documentos é uma área da Recuperação de Informações (RI) a qual vem sofrendo uma grande evolução desde o início da Internet, devido ao grande volume de informações dispostas de forma desestruturada no meio. Tem como principal objetivo catalogar toda essa informação pelo seu conteúdo de forma automática, visando uma recuperação dessa informação de maneira rápida e precisa.

As técnicas de Mineração de Textos promovem análises mais extensas do conteúdo dos documentos, pois identificam fatos, relações e padrões de forma mais similar à percepção por um humano lendo o mesmo documento. Na Mineração de Textos, as técnicas de indexação são usadas para diferentes propósitos como categorizar um documento, identificar o significado ou auxiliar a leitura de grandes volumes de textos (ARANHA, 2007).

Sem a indexação, localizar as informações sobre uma palavra-chave em um banco de dados textual, onde seus dados não estão distribuídos em uma estrutura organizada, seria necessária uma busca exaustiva caractere a caractere até que a sequência de caracteres desejada à palavra chave fosse encontrada (ARANHA, 2007). Logo, para nossa conveniência, a indexação tem como principal função atribuir palavras (termos) mais relevantes à um documento, o que torna a busca muito mais eficiente do que uma busca linear.

2.2.5 Mineração

A etapa de mineração é responsável pela escolha dos algoritmos de Aprendizado de Máquina apropriados para os dados. A decisão do algoritmo mais apropriado para o problema em questão não é óbvia, pois não existe um algoritmo ideal para todas as aplicações, tendo em vista que seguem estratégias diferentes e mostram resultados diferentes sobre diferentes tipos de textos.

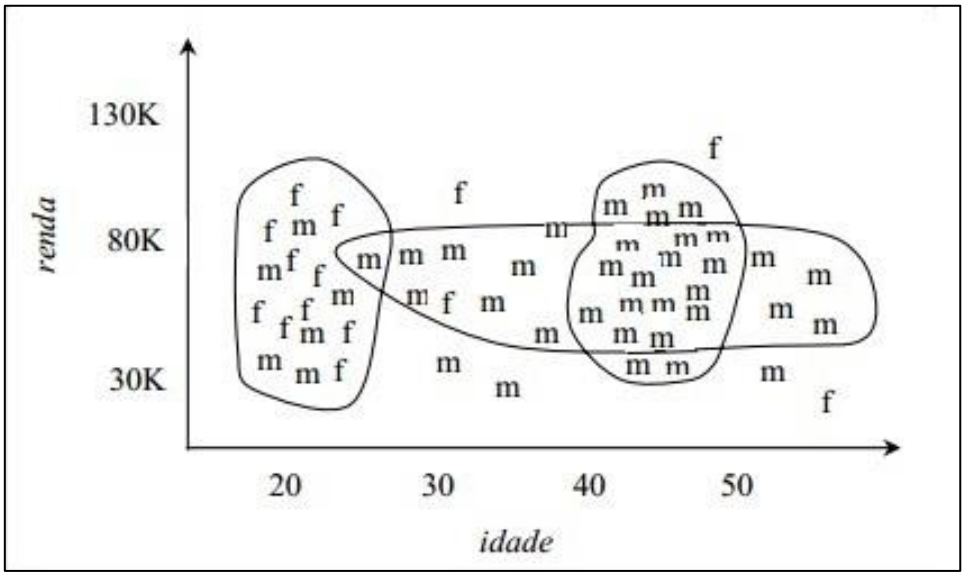
Os métodos de mineração de dados não são necessariamente utilizados de forma exclusiva, podendo ser aplicado primeiramente um método a base de dados, para melhor prepará-los para a aplicação e, então, aplicar outro no conjunto de dados resultante (GARCIA, 2003, p. 21). Portanto, durante o processo de mineração, diversas técnicas devem ser testadas e combinadas para que os seus resultados possam ser comparados e a mais apropriada seja utilizada.

As técnicas de classificação (*classification)* têm como objetivo classificar elementos de um conjunto de dados em categorias. A classificação dos dados é baseada em características que os elementos tenham em comum. Essas técnicas podem ser utilizadas, por exemplo, por bancos para classificar os seus clientes como “arriscado” e “seguro”, e assim facilitar a tomada de decisão sobre a aprovação de um empréstimo para os mesmos. As características que seriam avaliadas, nesse caso, seriam a “idade” e a “renda” do cliente.

Diferente das técnicas de classificação, as quais os dados são classificados em categorias previamente definidas, as técnicas de agrupamento (c*lustering*) são utilizadas para a identificação das categorias em determinado grupo de dados. Isto é, aglomerados de dados, também conhecidos como *clusters*, são formados pelo agrupamento de dados que possuem alta similaridade entre si, mas tem maior distinção se comparado aos dados de outros aglomerados (HAN, KAMBER e PEI, 2012, p. 23). A partir de então, é possível verificar os atributos com que cada aglomerado deriva.

Em um exemplo hipotético, apresentado por Garcia (2003) , é aplicado um modelo de agrupamento em uma base de dados de compradores de carros esportes, em que são conhecidos sexo, idade e renda de cada comprador. Pode-se perceber pela Figura 5, que a população foi subdividida em três aglomerados: compradores jovens, compradores do sexo masculino com faixa etária entre os 40 e 50 anos e compradores de várias faixas etárias com renda média-alta.

Figura 5 - Exemplo de método de agrupamento ou *clustering*



Fonte: Garcia (2003).

As técnicas de associação (*association*) buscam estabelecer relacionamentos entre um conjunto de dados, a fim de encontrar afinidades entre eles (GARCIA, 2003). São comumente utilizadas em sites de vendas online, com base na informação coletada das compras finalizadas, pode-se extrair uma regra de associação mostrando, por exemplo, que quando um cliente compra um computador, em 50% dos casos ele também adquire um *software*. Após a aplicação de regras de associação a um conjunto de dados, o resultado obtido é uma lista de padrões que indicam afinidades entre itens (GARCIA, 2003).

As técnicas de análise de regressão (*regression*) são métodos estatísticos que são comumente usados para predições numéricas (HAN, KAMBER e PEI, 2012, p. 19). Portanto, regressão refere-se à descoberta de padrões em dados já existentes para estimar a probabilidade de eventos voltarem a acontecer (GARCIA, 2003, p. 21). O uso de métodos de regressão pode servir, por exemplo, para estimar a receita das vendas de um determinado item em vendas futuras, baseando-se nos dados de suas vendas anteriores.

Nas seções a seguir, são descritas as técnicas que se aplicam com sucesso em dados textuais, foco deste trabalho, segundo Witten e Frank (2005).

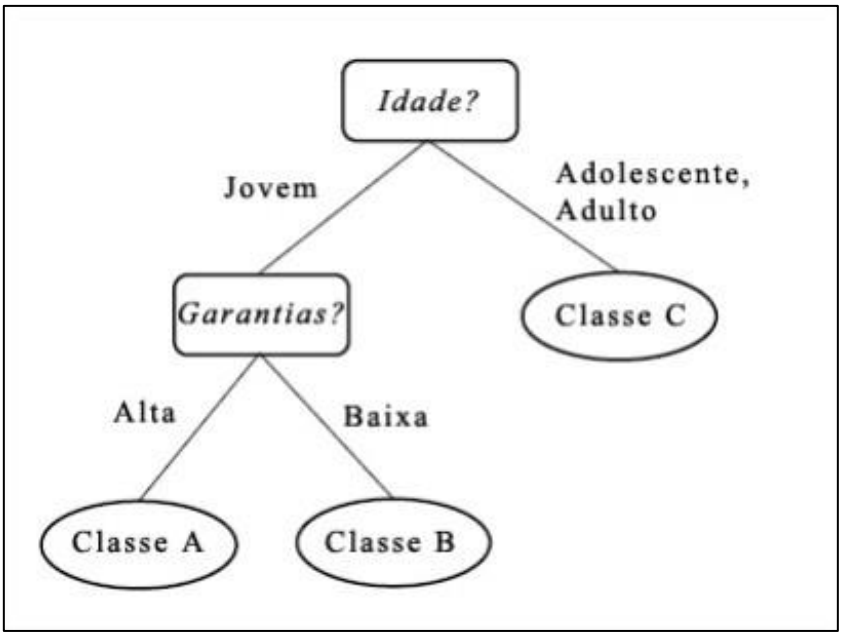
2.2.5.1 Árvores de Decisão (*J48*)

Uma árvore de decisão é um fluxograma em forma de árvore, onde cada nodo (não folha) indica um teste sobre um atributo, cada ramo (ou ligação entre os nodos) representa uma saída do teste do nodo superior, e cada nodo folha representa uma categoria a que um dado pertence (HAN, KAMBER e PEI, 2012, p. 105).

Quando um classificador de árvore de decisão é utilizado para seleção dos atributos mais relevantes, todos os atributos que não aparecem na árvore construída são considerados irrelevantes para o problema, enquanto que os atributos que aparecem são considerados os mais relevantes.

Um dos algoritmos mais utilizados de árvore de decisão é o algoritmo J48, sendo considerado o que apresenta o melhor resultado na montagem de árvores de decisão, a partir de um conjunto de dados de treinamento. Utiliza uma abordagem de dividir-para-conquistar, onde um problema complexo é decomposto em subproblemas mais simples, e aplica recursivamente a mesma estratégia a cada subproblema, dividindo o espaço definido pelos atributos em subespaços, associando-se a eles uma classe (WITTEN e FRANK, 2005).

Figura 6 - Exemplo de árvore de decisão



Fonte: Han, Kamber e Pei (2012).

2.2.5.2 Classificador Bayesiano (*Bayes Net*) e Bayesiano Ingênuo (*Naïve Bayes*)

Os classificadores Bayesianos são técnicas estatísticas de probabilidade condicional, baseadas no Teorema de Thomas Bayes, o qual afirma que as probabilidades devem ser revistas quando conhecemos algo mais sobre os eventos (TRIOLA, 1999).

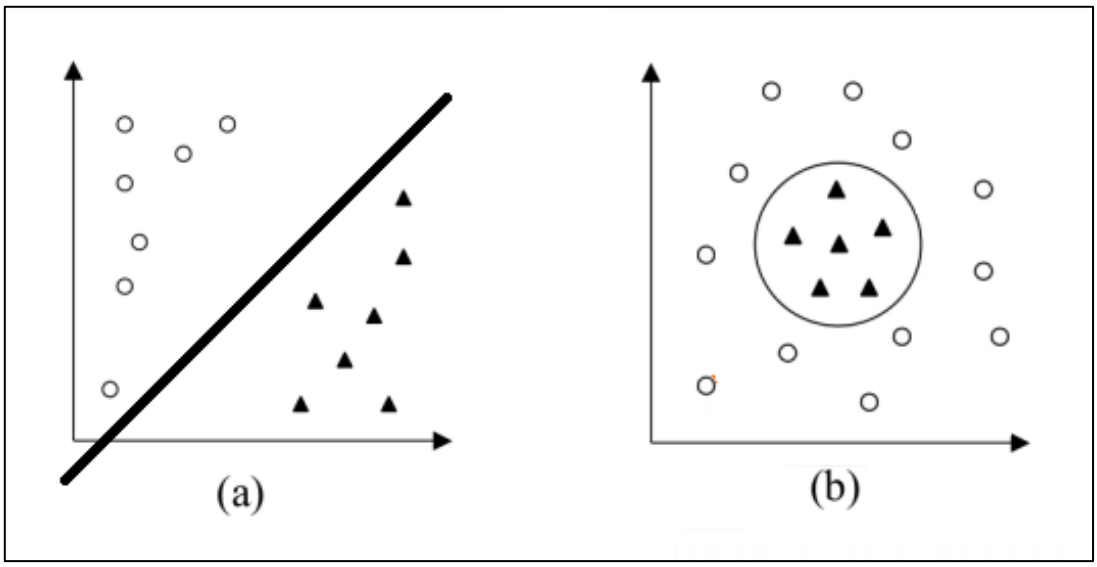
O classificador Bayesiano, também conhecido como *Bayes Net*, assume que existem dependências entre os eventos, dessa forma procura classificar um determinado atributo a uma determinada classe, levando em consideração a probabilidade de outro atributo pertencer também à essa classe.

O classificador bayesiano ingênuo (*Naïve Bayes*), por sua vez, assume que não existem dependências entre os atributos, e por isso é particularmente recomendado quando a dimensionalidade dos dados é muito grande, devido à sua simplicidade.

2.2.5.3 Máquina de Vetores de Suporte (*Support Vector Machines - SVM*)

A técnica SVM tem a capacidade de resolver problemas de classificação, e se caracteriza por apresentar uma boa capacidade de generalização (LORENA e CARVALHO, 2007). É considerado um classificador linear binário não probabilístico, portanto, é capaz de dividir as instâncias de um conjunto de dados em duas classes distintas. A partir de adaptações do algoritmo, atualmente também é possível resolver problemas não-lineares. Na Figura 7 é representado o algoritmo SVM aplicado em um conjunto de dados linear e não linear.

Figura 7 - Aplicação do algoritmo SVM em: (a) conjunto de dados linear; (b) conjunto de dados não linear



Fonte: Próprio autor.

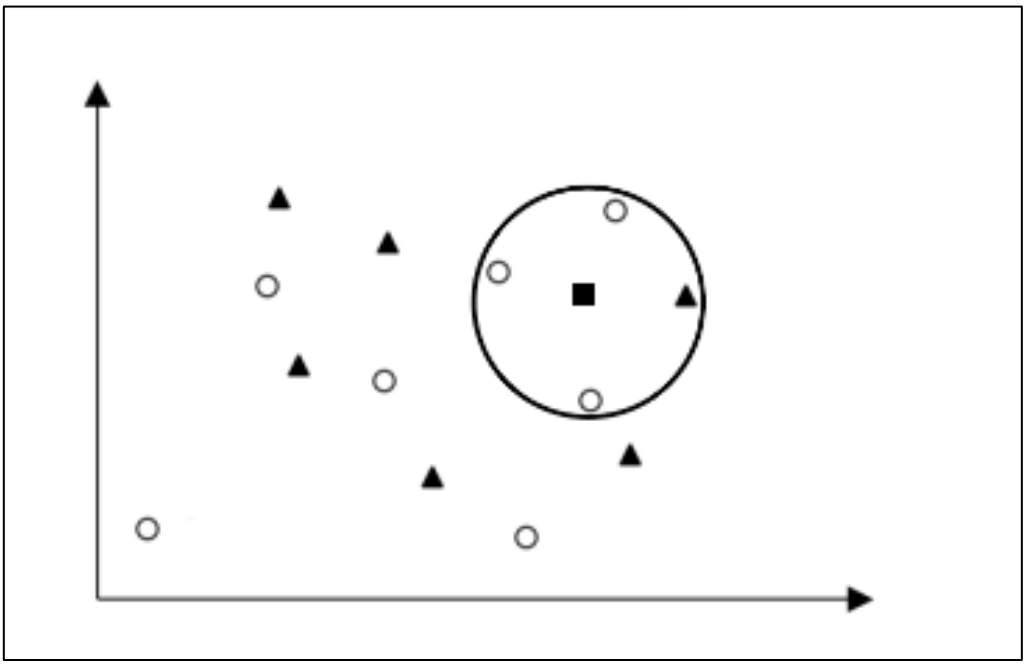
2.2.5.4 K-Vizinhos mais próximos (*K-Nearest Neighbor - KNN*)

O algoritmo KNN é um método simples que utiliza a estimação da densidade, para classificação de seus dados. O mesmo assume que todas as instâncias de dados correspondem a pontos em um espaço n-dimensional. Logo, as suas distâncias podem ser medidas através da distância Euclidiana (PETERSON, 2009).

Ao classificar uma nova instância, os K vizinhos mais próximos à posição da nova instância são contabilizados de acordo com sua classe, e dessa forma, a classe com maior frequência no menor espaço é herdada à nova instância (PETERSON, 2009). O parâmetro K utilizado pelo KNN indica o número de vizinhos que são usados pelo algoritmo durante a fase de teste. O parâmetro K pode que o algoritmo adquira um resultado preciso, no entanto, é necessária uma análise de cada problema para atribuir-se o valor ideal ao parâmetro K.

Na Figura 8, por exemplo, considerando um conjunto de dados já classificados durante a fase de treinamento (círculos e triângulos), uma nova instância representada pelo quadrado seria classificada como círculo por se tratar da classe de maior frequência entre os seus K vizinhos mais próximos.

Figura 8 - Representação do algoritmo KNN

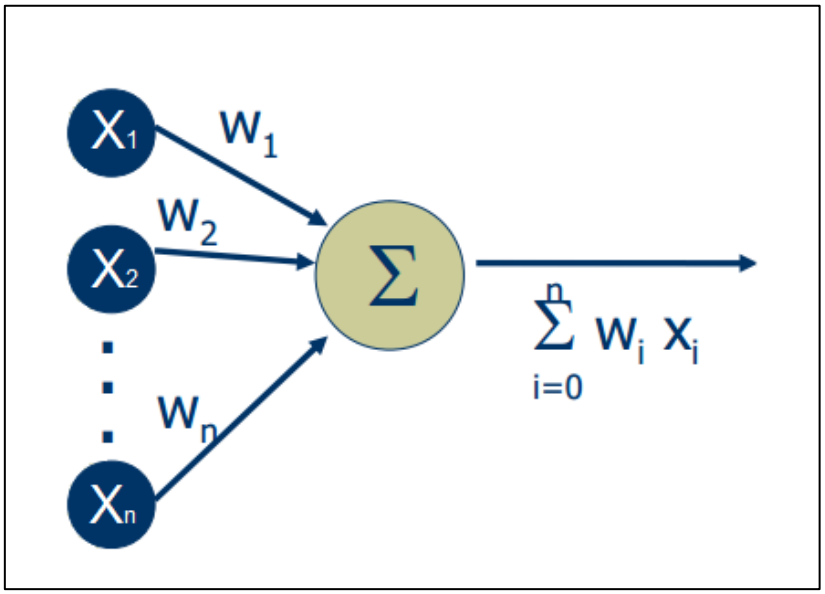


Fonte: Próprio autor.

2.2.5.5 Rede Neural (*Multilayer Perceptron*)

As redes neurais artificiais são modelos computacionais que simulam um sistema nervoso humano. A rede neural é formada por neurônios artificiais interligados por conexões, cada conexão entre neurônios conexões de entrada que são compostas por um valor de entrada (X) e um peso (W), onde a saída é o somatório das multiplicações de todos os valores de entrada pelos seus pesos (BISHOP, 1995). O neurônio artificial e suas conexões, são representados na Figura 9.

Figura 9 - Representação de um Neurônio Artificial e suas conexões de entrada e saída



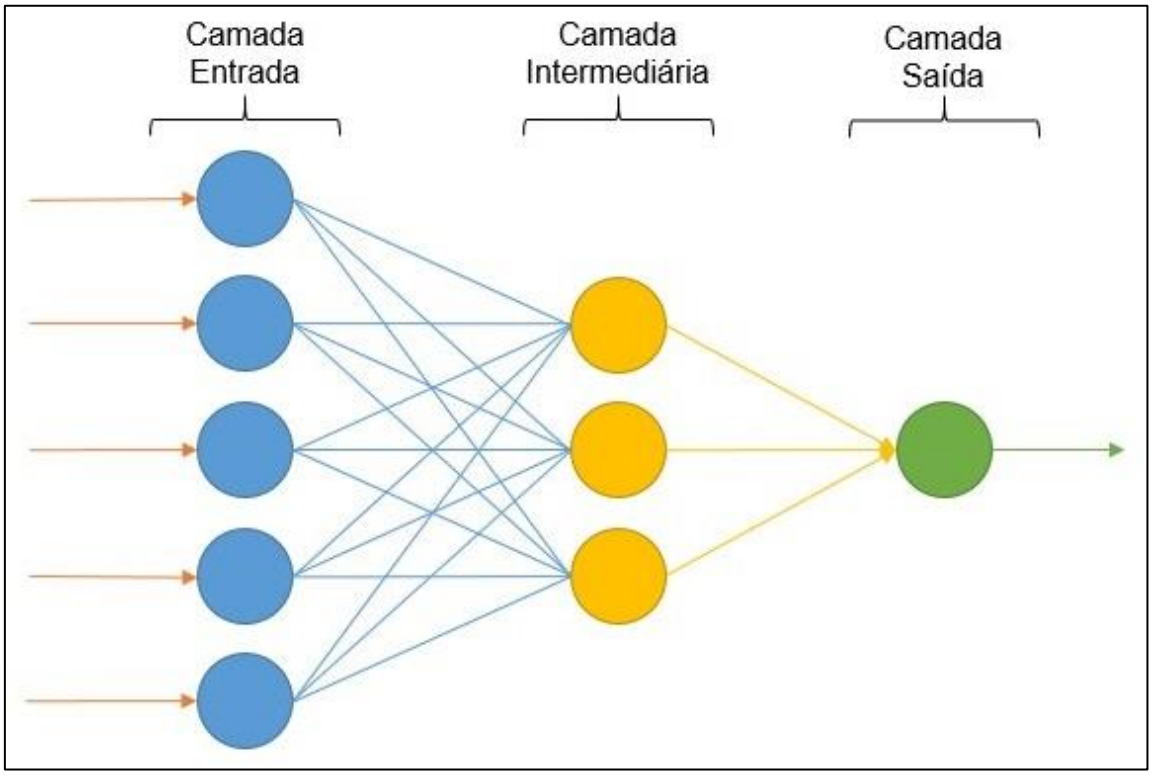
Fonte: Próprio autor.

O algoritmo *Multilayer Perceptron* é um algoritmo que divide a rede neural em multicamadas, ou seja, os sinais de entrada são propagados camada a camada, através das conexões dos neurônios (BISHOP, 1995). As camadas, normalmente, são:

a) Camada de Entrada: recebem os valores do mundo externo como estímulo;
b) Camadas Intermediárias: realizam os processamentos internos à rede;
c) Camadas de Saída: recebem o resultado final e o apresentam ao mundo exterior;

As camadas de entrada e saída são sempre uma apenas, no entanto, a quantidade de camadas intermediárias pode variar de acordo com a complexidade de cada problema. As conexões entre os neurônios de diferentes camadas são representadas na Figura 10.

Figura 10 - Representação de uma Rede Neural ou *Multilayer Perceptron*



Fonte: Próprio autor.

2.2.6 Análise

Finalmente, após a etapa de mineração, as próximas etapas correspondem ao processo de analisar e interpretar os resultados, permitindo que o classificador possa ser avaliado. Nessa etapa, os resultados obtidos pelo classificador devem ser comparados aos resultados esperados, realizando a avaliação de acordo com métricas definidas pelo avaliador. Também é imprescindível que especialistas no domínio do conhecimento verifiquem a compatibilidade dos resultados com o conhecimento disponível no domínio (ARANHA, 2007).

2.3 CONSIDERAÇÕES FINAIS

Nesse capítulo foram selecionadas e analisadas diversas fórmulas de avaliação da apreensibilidade, que foram desenvolvidas para diferentes aplicabilidades. Todas as fórmulas selecionadas são métodos estatísticos aplicados sobre as características do texto, como a contagem de palavras, sílabas, caracteres e frases, o que as tornam passíveis de serem computáveis, ou seja, transcritas para um algoritmo.

A maioria das fórmulas foram originalmente desenvolvidas para língua Inglesa, e algumas adaptações para as línguas Espanhola, Holandesa e Francesa. Apesar desse fato, existem estudos que demonstram a coerência dos resultados da aplicação de algumas das fórmulas para a língua Portuguesa.

Na Tabela 6, é possível visualizar o resumo das fórmulas explicitadas neste estudo, bem como suas áreas de aplicação e aplicabilidade para a língua portuguesa.

Tabela 6 - Fórmulas de apreensibilidade selecionadas

Ferramenta de Análise	**Ano**	**Língua Desenvolvida**	**Aplicabilidade para a língua Portuguesa?**	**Área (s) de Aplicação**
The New Dale-Chall Readability Formula	1948	Inglês	Não	Medicina e Jornalismo
Flesch Reading Ease	1948	Inglês	Sim	Medicina e Jornalismo
Gunning-Fog Index	1952	Inglês	Não	Medicina e Jornalismo
Fernández-Huerta	1959	Espanhol	Sim	Nenhuma específica.
Fry Graph Readability Formula	1963	Inglês	Não	Medicina e Jornalismo
Automated Readability Index	1967	Inglês	Sim	Documentos técnicos e força militar.
SMOG Grading	1969	Inglês	Não	Jornalismo
FORCAST formula	1973	Inglês	Não	Medicina, documentos técnicos, força militar e em textos não lineares.
Flesch Kincaid Grade Level	1975	Inglês	Sim	Documentos técnicos, força militar e em textos não lineares.
Coleman-Liau Index	1975	Inglês	Sim	Jornalismo

Fonte: Próprio autor.

Da lista de fórmulas avaliadas, há evidências de que as fórmulas *Flesch Reading Ease* e *Fernández-Huerta* são as mais indicadas para os fins desse trabalho, pelos seguintes motivos:

a) A fórmula de *Flesch Reading Ease*, não modificada, possui a vantagem de ser uma das mais usadas para avaliações da apreensibilidade, tanto para textos na língua Inglesa quanto para a Portuguesa. Além disso, devido ao fato de ser bastante utilizada, a mesma apresenta um maior número de estudos disponíveis para consulta, se comparado a outras fórmulas.
b) A fórmula de *Fernández-Huerta* possui a vantagem de ser uma fórmula desenvolvida para a língua Espanhola, um idioma originado do Latim assim como o Português. No entanto, apesar de existirem variadas aplicações da fórmula tanto para a língua Espanhola quanto Portuguesa, ainda existem algumas discussões sobre a confiabilidade da mesma (LAW, 2011).

As interpretações sobre os resultados de ambas as fórmulas são indicadas na Tabela 7, que demonstra a dificuldade de leitura e escolaridade necessária aproximada.

Tabela 7 - Interpretação dos valores obtidos com *Flesch Reading Ease* e *Fernández-Huerta* de acordo com os graus de escolaridade do Brasil

Índice da Fórmula	**Dificuldade de leitura**	**Escolaridade Aproximada**
90 a 100	Muito fácil	5ª Série
80 a 89	Fácil	6ª Série
70 a 79	Razoavelmente fácil	7ª Série
60 a 69	Padrão	8ª e 9ª Série
50 a 59	Razoavelmente difícil	Início do Ensino médio
30 a 49	Difícil	Ensino Médio e Superior
0 a 29	Muito difícil	Nível Superior

Fonte: Flesch (1949, p. 149)

Nesse capítulo, também foram apresentados conceitos da Mineração de Textos e os algoritmos de aprendizado de máquina que são aplicados com

sucesso em conjuntos de dados textuais. Essas técnicas são avaliadas acerca de sua efetividade para o problema a ser resolvido por esse trabalho. Na Tabela 8, são listadas as técnicas e algoritmos de aprendizado de máquina abordados nesse capítulo.

Tabela 8 - Técnicas e algoritmos de aprendizado de máquina utilizados na Mineração de Textos

Técnicas	**Algoritmo Mais Utilizado**
Árvores de Decisão	*J48*
Classificador Bayesiano	*Bayes Net*
Classificador Bayesiano Ingênuo	*Naïve Bayes*
Máquinas de Vetores de Suporte	*Support Vector Machines - SVM*
K-Vizinhos mais próximos	*K-Nearest Neighbor - KNN*
Rede Neural	*Multilayer Perceptron*

Fonte: Próprio autor.

As técnicas da Tabela 8 são indicadas na literatura como sendo as mais apropriadas para a aplicação em problemas de Mineração de Textos (WITTEN e FRANK, 2005). E por esta razão, serão testadas no escopo deste trabalho, essencialmente como técnicas de classificação.

3 IMPLEMENTAÇÃO E TESTES DA APLICAÇÃO *MINNING*

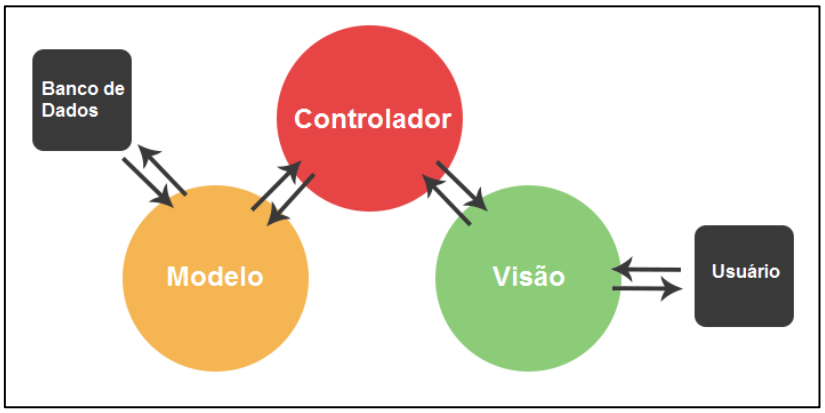
Devido à inexistência de uma aplicação que permita avaliar informações textuais acerca de sua apreensibilidade e qualidade de forma automática na língua Portuguesa e com foco para a área da saúde, esse trabalho tem por objetivo propor a implementação de uma ferramenta que atinja esse objetivo, chamado neste trabalho por *Minning*.

A seguir são descritas características da aplicação desenvolvida, as tecnologias utilizadas, arquitetura, modelagem, interface gráfica e detalhes das funcionalidades.

3.1 ARQUITETURA DA APLICAÇÃO

Para a arquitetura, optou-se pela utilização do padrão de arquitetura MVC, ou *Model View Controller*, por ser o mais utilizado atualmente para a criação de aplicações na plataforma *web*. O MVC, segundo Burbeck (1992), é estruturado em três camadas que possuem interações entre si. O modelo (*model*) é responsável pelo negócio, persistência e dados da aplicação. A visão (*view*) é responsável pela interação com o usuário, tanto como o recebimento de instruções como a representação dos dados. O controlador (*controller*) é responsável por receber as instruções da visão e enviar para o modelo, assim como receber comandos do modelo a fim de enviar para a visão representar.

Figura 11 - Padrão de Arquitetura *MVC*



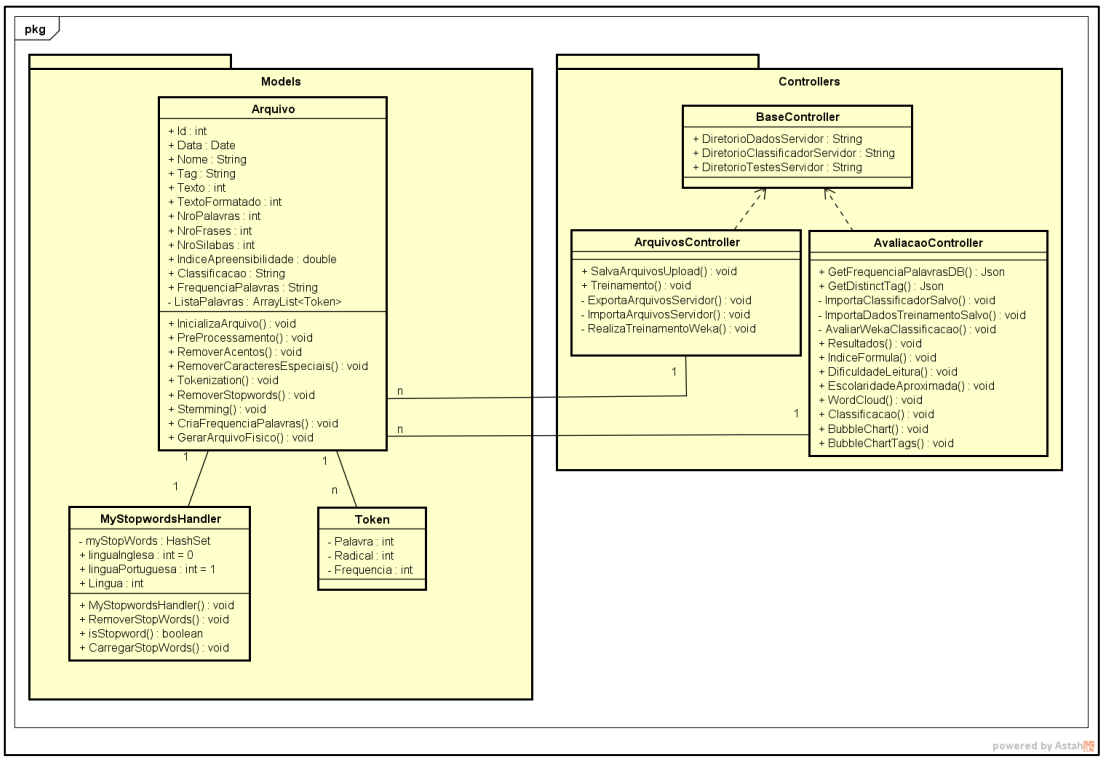
Fonte: http://lappis.unb.br/redmine/attachments/download/3717/mvc_diagram.png, modificado pelo autor.

Segundo a documentação da Microsoft (2016), uma das principais vantagens do padrão de arquitetura MVC, é que por ser organizado em camadas, a interface visual é desacoplada da lógica da aplicação. Isso garante benefícios como separação dos conceitos e reaproveitamento de código. Em outras palavras, múltiplas visões podem ser desenvolvidas utilizando a mesma lógica da aplicação, tendo em vista que o acesso ao modelo acontece unicamente através do controlador. Da mesma forma, múltiplas aplicações podem utilizar uma mesma visão para representação dos dados.

3.2 MODELO DE CLASSES

A ferramenta a ser desenvolvida é dividida em duas etapas: a etapa de treinamento, responsável pela leitura e armazenagem dos arquivos textos e o treinamento do avaliador através dos algoritmos de aprendizagem de máquina; e a etapa de Avaliação, responsável pela classificação do arquivo de acordo com o treinamento realizado. A modelagem das classes da nossa aplicação é igualmente dividida, conforme a Figura 12.

Figura 12 - Diagrama de classes da aplicação



Fonte: Próprio autor.

No modelo de classes, são exibidas as camadas de *Controller* e *Model*. A *Model* “Arquivo” é responsável pelo armazenamento de todos os arquivos de treinamento adicionados à aplicação, contendo atributos como texto, texto formatado, número de palavras, frases, sílabas, resultado do índice de apreensibilidade e da classificação, além de métodos responsáveis pelo pré-processamento do texto como *Tokenization* e *Stemming*. Todo “Arquivo” contém uma instância da classe *MyStopwordsHandler*, responsável pela remoção das *stopwords*, e uma lista da classe *Token*, responsável pelo armazenamento de todas as palavras do texto.

Na camada *Controller*, existem as duas principais classes do sistema. A *Controller* ArquivosController é responsável por gerenciar os arquivos de treinamento no banco de dados, e possui métodos como SalvaArquivosUpload, ExportaArquivosServidor, ImportaArquivosServidor e RealizaTreinamentoWeka. Já a *Controller* AvaliacaoController é responsável pela classificação e avaliação de novos textos, bem como a apresentação dos resultados gráficos para o usuário, e possui alguns métodos como ImportaClassificadorSalvo e ImportaDadosTreinamentoSalvo para realizar a classificação do texto, métodos como IndiceFormula, DificuldadeLeitura, WordCloud e BubbleChart responsáveis pela geração e representação dos resultados.

Além disso, as duas *controllers* mencionadas estendem da classe BaseController que contém atributos com diretórios locais do servidor, que são usados por ambas as *controllers*.

3.3 TECNOLOGIAS E FERRAMENTAS UTILIZADAS

Para o desenvolvimento da aplicação, foram utilizadas as ferramentas: o framework ASP.NET, o banco de dados *SQL Server*, a implementação *Weka*, a biblioteca *IKVM.NET* e as bibliotecas de visualização de dados *D3.JS* e *jqPlot*.

3.3.1 ASP.NET

A aplicação foi desenvolvida utilizando o framework *.NET* da Microsoft, mais especificamente pela plataforma *ASP.NET* que oferece suporte ao desenvolvimento de aplicações web. Essa plataforma foi escolhida por diversos motivos: apresentar vasta documentação, estar bem estruturada e ser robusta, possuir recursos que atendem os requisitos da aplicação a ser desenvolvida e pela afinidade do desenvolvedor com as ferramentas citadas.

3.3.2 SQL Server

Para a persistência dos dados será utilizado o *SQL Server* da Microsoft, devido à sua forte aderência com as plataformas de desenvolvimento ASP.NET, garantindo maior facilidade e produtividade quando comparado ao uso de outros bancos de dados. Em especial ao fato do framework *.NET* possuir bibliotecas que automatizam a criação do modelo de dados em bancos *SQL Server*, baseando-se nas classes de dados, como por exemplo a *Entity Framework*.

3.3.3 Weka

O *Weka*, ou *Waikato Environment for Knowledge Analysis*, é uma coleção de algoritmos de aprendizado de máquina para a tarefa de mineração de dados. É um *software* de domínio público, licença GNU (*General Public License*), desenvolvido em *Java* por pesquisadores do Departamento de Computação da Universidade de Waikato da Nova Zelândia (BOUCKAERT, FRANK, *et al.*, 2016).

O Weka possui ferramentas para as técnicas de pré-processamento, classificação, regressão, agrupamento, associação e visualização. E além disso, os seus algoritmos podem ser aplicados diretamente a um conjunto de dados, ou serem utilizados diretamente do seu código, independentemente da plataforma utilizada (BOUCKAERT, FRANK, *et al.*, 2016).

Tendo isso em mente, utilizou-se do *software Weka* diretamente de nossa aplicação, pelos seguintes motivos e benefícios:

a) Todas as técnicas que se aplicam aos dados textuais já estão implementadas;
b) Recursos para testes disponíveis;
c) Ferramentas simples de visualização de resultados.

3.3.4 IKVM.NET

Para ser possível a utilização de qualquer código *Java* diretamente de um código .NET, torna-se necessário a conversão de arquivos de extensão *JAR* para a extensão *DLL*. A *IKVM.NET* é uma ferramenta *open source* que possibilita a interoperabilidade entre plataformas *Java* e .NET.

Além da conversão dos arquivos *JAR* em *DLL*, é necessário referenciar a biblioteca IKVM ao projeto, para que o projeto desenvolvido em C# consiga interpretar e compilar objetos de bibliotecas *Java*.

A biblioteca IKVM teve grande utilidade no desenvolvimento da ferramenta, pois garantiu a integração perfeita entre o *Weka* e o nosso projeto.

3.3.5 D3.JS

D3.JS é uma biblioteca *JavaScript open source* para manipulação de documentos baseados em dados. Desenvolvida pela *D3 Data Driven Documents.* A biblioteca possibilitou a geração de visualizações e representações de dados em diversos formatos visuais (BOSTOCK, 2013).

3.3.6 jqPlot

A biblioteca *jqPlot* é uma versátil biblioteca *jQuery open source* utilizada para geração de gráficos (LEONELLO e PRITCHARD). Essa biblioteca garante a fácil geração de gráficos mais simples, quando comparados as visualizações geradas pela biblioteca *D3.JS*.

3.4 DESENVOLVIMENTO

Esta seção descreve a implementação da aplicação, apresentando detalhes sobre o desenvolvimento do software para avaliação textual. Nela são demonstradas amostras de códigos mais relevantes para o funcionamento e a integração com bibliotecas desenvolvidas por terceiros.

A seção ESTÁ organizada em três partes: avaliação da apreensibilidade, avaliação da qualidade e visualizações dos resultados das avaliações.

3.4.1 Avaliação da Apreensibilidade

A avaliação da apreensibilidade de uma amostra textual independe que seja realizado o treinamento de um classificador, diferente da avaliação da qualidade. Dessa forma, para calcularmos a apreensibilidade textual, uma fórmula de apreensibilidade teve de ser selecionada dentre as que foram investigadas nesse trabalho.

Em um primeiro momento, a fórmula *Flesch Reading Ease* se mostrou mais vantajosa. Isto por ela ser a fórmula mais disseminada entre os estudos de áreas como medicina e jornalismo, enquanto que, a fórmula de *Fernández-Huerta* não demonstrou confiabilidade de acordo com a bibliografia encontrada (LAW, 2011).

Durante o desenvolvimento e testes do software, os resultados da fórmula *Flesch Reading Ease* quando aplicados a textos na língua Portuguesa mostraram um valor muito inferior ao esperado. Em outras palavras, os resultados obtidos, quando interpretados através da Tabela 7, tendiam a indicar uma maior dificuldade de leitura e entendimento do que a constatada.

Para identificarmos o motivo de tais divergências, algumas amostras de textos foram transcritas para a língua Inglesa, e então, a comparação dos números de palavras, frases e sílabas de cada texto foi realizada. Conforme observado, a principal diferença entre as contagens acontecia devido ao número de sílabas do texto em cada língua, o qual mostrou-se 40% maior nos textos da língua Portuguesa quando comparado aos textos na língua Inglesa.

Tendo isso em vista, tornou-se necessário a utilização da fórmula de *Fernández-Huerta*, cujos termos da fórmula foram balanceados para melhores resultados na língua Espanhola.

A fórmula selecionada para a nossa aplicação necessita da contagem das seguintes variáveis para a sua implementação (HUERTA, 1959 apud BARBOZA e NUNES, 2007):

a) A contagem de palavras
b) A contagem de frases
c) A contagem de sílabas

A contagem de palavras é realizada através da separação dos termos por espaços em brancos, quebras de linhas e pontuação. Para exemplificar o algoritmo de contagem de palavras, é demonstrado a amostra de código em Algoritmo 1.

Algoritmo 1 - Algoritmo de contagem de palavras de um texto

```
// Function desenvolvida na linguagem C#
public int ContagemPalavras(string texto)
{
   int cont = 0;

   // Desconsidera espaços antes e depois dos textos
   texto = texto.Trim();

   // Loop sobre todos os caracteres do texto
   for (int i = 1; i < texto.Length; i++)
   {
      // Se o caracter da posição anterior é um espaço em branco ou quebra
de linha
      if (char.IsWhiteSpace(texto[i - 1]) == true)
      {
         // Se o caracter da posição atual é uma letra ou número
         if (char.IsLetterOrDigit(texto[i]) == true)
         {
            cont++;
         }
      }
   }
   // Conta a primeira palavra
   if (texto.Length > 2)
      cont++;
   // Retorna o número de palavras
   return cont;
}
```

Fonte: Próprio autor.

A contagem de frases assume que uma frase sempre terminará com ponto final, ponto de interrogação ou ponto de exclamação. Portanto, a contagem das frases é realizada pela separação do texto pelas suas possíveis terminações, conforme demonstrado em Algoritmo 2.

Algoritmo 2 - Algoritmo de contagem de frases de um texto

```
// Function desenvolvida na linguagem C#
public int ContagemFrases(string texto)
{
   // Caracteres que indicam terminação de frases
   char[] separatingChars = { '.', '?', '!', '\0' };

   // Divide o texto em frases
   string[] frases = texto.Split(separatingChars,
StringSplitOptions.RemoveEmptyEntries);

   // Retorna o número de frases
   return frases.Length;
}
```

Fonte: Próprio autor.

A contagem de sílabas não apresenta uma maneira universal de ser realizada, tendo em vista que cada língua tem suas próprias particularidades gramaticais. Dessa forma, aderiu-se ao uso de uma implementação *open source*, criada por Oliveira (2007), no formato de biblioteca *Java*, e disponibilizada pelo mesmo em https://code.google.com/archive/p/silabaspt.

Para ser possível a utilização de uma *Java API* em um código C#, torna-se necessário converter o arquivo de extensão *JAR* para a extensão *DLL*, através de outra ferramenta *open source* conhecida por IKVM (www.ikvm.net). Dessa forma, o arquivo *DLL* pode ser adicionado como uma referência ao projeto C#, resultando no código demonstrado em Algoritmo 3.

Algoritmo 3 - Algoritmo de contagem de sílabas de um texto, utilizando a biblioteca Java API de Oliveira (2007)

```
// Function desenvolvida na linguagem C#
public int ContagemSilabas(string texto)
{
   // Utilização da JAVA API de Oliveira (2007)
   var s = silabas.SilabasPT.separa(texto);

   // Retorna o número de sílabas do texto
   return s.size();
}
```

Fonte: Próprio autor.

Após a obtenção das variáveis necessárias, deve-se então, transcrever a fórmula de *Fernández-Huerta* para um algoritmo, conforme Algoritmo 4. O algoritmo desenvolvido se aplica a textos na língua Portuguesa, devido ao fato de utilizar a contagem de sílabas em português.

Algoritmo 4 - Algoritmo de *Fernández-Huerta* para textos na língua Portuguesa

```
// Function desenvolvida na linguagem C#
public static double FernandezHuerta(string texto)
{
   // Realiza a contagem de palavras, frases e sílabas do texto
   var nroPalavras = ContagemPalavras(texto);
   var nroFrases   = ContagemFrases(texto);
   var nroSilabas  = ContagemSilabas(texto);

   // Variável Contagem de Palavras / Variável Contagem de Frases
   var ASL = (float) nroPalavras / nroFrases;

   // Variável Contagem de Sílabas / Variável Contagem de Palavras
   var ASW = (float) nroSilabas / nroPalavras;

   // Fórmula de Fernandez-Huerta
   var resultado = 206.84 - (1.02 * ASL) - (60 * ASW);

   // Limita o resultado a uma escala de 0 a 100
   resultado = Math.Max(resultado, 0);
   resultado = Math.Min(resultado, 100);

   return Math.Round(resultado);
}
```

Fonte: Próprio autor.

3.4.2 Avaliação da Qualidade

A avaliação da qualidade de um texto pode ser inferida através de um classificador devidamente treinado, utilizando as técnicas de mineração de textos investigadas nesse trabalho. Para facilitar a utilização dessas técnicas, o *software Weka* foi integrado ao nosso projeto, graças a sua versão sem interface visual gráfica, que possibilita a utilização de suas classes integradas a qualquer código (BOUCKAERT, FRANK, *et al.*, 2016).

O treinamento do classificador figura a etapa de treinamento da nossa aplicação, dessa foram, nesse capítulo serão descritos os detalhes do desenvolvimento da avaliação da qualidade, e validação das técnicas de mineração de textos.

3.4.2.1 Coleta de amostras textuais para o treinamento

A etapa de treinamento é responsável pela aprendizagem do sistema sobre um conjunto de dados de treinamento, permitindo que o modelo de

aprendizagem criado seja utilizado com o objetivo de avaliar novas entradas de dados.

O treinamento do classificador depende, então, essencialmente dos dados de treinamento, que correspondem a amostras textuais da área da saúde (para o nosso caso). Para o treinamento e testes do classificador deste trabalho, uma coleta de amostras textuais foi realizada pelos especialistas da área da saúde.

Entre as 68 amostras coletadas, estão contidos textos sobre “dor nas costas”, “coluna vertebral”, “hérnia de disco” e “cirurgias de coluna” encontrados na internet. As amostras foram previamente analisadas e pré-classificadas entre três classificações: “Negativos”, “Regulares” e “Positivos”, contando com o apoio do DISCERN, que corresponde a perguntas-chave para avaliação dos critérios de qualidade de um texto para as informações sobre saúde.

Dentre as 68 amostras, 39 amostras foram pré-classificadas como “Negativas”, 22 amostras como “Regulares” e 7 amostras como “Positivas”.

A pré-classificação das amostras textuais nos permite avaliar a porcentagem de acerto do classificador construído, bem como avaliar qual a técnica de classificação mais indicada para o nosso problema.

3.4.2.2 Pré-Processamento

O pré-processamento das amostras de textos é um processo crítico para a mineração de textos, pois garante que o texto seja devidamente estruturado antes de ser submetido a um classificador.

Dessa forma, no momento que as amostras textuais de treinamento são submetidas a nossa aplicação, passam por algumas técnicas de pré-processamento. São elas:

a) O texto é convertido para minúsculo, ou *lowerCase*.

b) Os caracteres numéricos e especiais são removidos, como pontuação e outros mais específicos.

c) Os caracteres com acentuação são substituídos pelo respectivo caractere sem acentuação, por exemplo “á” é substituído por “a”.

d) *Tokenization*, em que o texto é transformado em uma lista com todas as palavras. As palavras repetidas são adicionadas quantas vezes forem presentes no texto, pois indicam a frequência das mesmas no texto.

e) Remoção de *stopwords*, conforme Apêndice A.

f) *Stemming*, em que as palavras são convertidas para os seus devidos radicais. Essa técnica foi obtida através da utilização de um pacote não oficial do *Weka,* PTStemmer. Essa é uma biblioteca de *stemmer* para a língua portuguesa, desenvolvida por Pedro Oliveira (WEKA, 2016).

Apesar do *Weka* oferecer implementações da grande maioria dessas técnicas, todas essas técnicas foram aplicadas previamente à integração do *Weka*. Uma amostra de código é demonstrada em Algoritmo 5.

Algoritmo 5 - Método responsável pelo Pré-Processamento do texto

```
// Método desenvolvida na linguagem C#
private void PreProcessamento()
{
   //Converte o texto para minúsculas
   TextoFormatado = Texto.ToLower();

   //Remove os caracteres especiais e números
   TextoFormatado = RemoverCaracteresEspeciais(TextoFormatado);

   //Substitui os caracteres com acentuação pelo respectivo caractere sem
acentuação
   TextoFormatado = RemoverAcentos(TextoFormatado);

   //Tokenization, transforma o Texto em uma Lista de Palavras
   _listaPalavras = Tokenization(TextoFormatado);

   //Remove as Stopwords da Lista de Palavras
   RemoverStopwords(MyStopwordsHandler.linguaPortuguesa, _listaPalavras);

   //Converte as palavras nos seus radicais (Stemming de PTStemmer)
   Stemming(_listaPalavras);

   //Recompõe o texto a partir da lista de palavras contendo os radicais
   TextoFormatado = string.Empty;
   foreach (var palavra in _listaPalavras)
      TextoFormatado = TextoFormatado + " " + palavra.Radical;

}
```

Fonte: Próprio autor.

Antes que o treinamento possa ser realizado, são necessários outros dois pré-processamentos. Neste trabalho eles tiveram de ser executados diretamente no *Weka* por meio da utilização dos seus filtros: *StringToWordVector* e *InfoGainAttributoEval*.

O filtro *StringToWordVector* é responsável pela técnica de *tokenization*, pois o mesmo converte o texto em atributos e realiza a criação da classe do treinamento. A classe do treinamento corresponde às possíveis classificações que os textos foram previamente classificados pelo analista (Negativas, Regulares e Positivas).

O filtro *InfoGainAttributeEval* é responsável pela técnica de seleção dos termos mais relevantes (*AttributeSelection*), e é também essencial para reduzir o número de atributos, reduzindo assim a dimensionalidade do problema e eliminando palavras irrelevantes para o processo de classificação. Os textos utilizados para o treinamento eram compostos por aproximadamente 2000 atributos que foram reduzidas para somente os 100 mais relevantes.

O filtro *InfoGainAttributeEval* foi escolhido dentre outros filtros capazes de realizar a seleção dos termos mais relevantes, pois os seus resultados mostraram uma maior porcentagem de acertos comparado aos outros, como por exemplo o filtro *CfsSubsetEval*.

3.4.2.3 Treinamento

Uma vez que os textos foram todos pré-processados, então o treinamento do classificador pode ser realizado.

Em um primeiro momento, algumas técnicas de classificação foram investigadas, quanto à sua aplicação à mineração de textos. Alguns testes de classificação foram realizados com todas as técnicas, utilizando a técnica de *Cross-Validation* do *Weka*, que separa o conjunto de dados em dados de treinamento e teste, para que aquela com a maior porcentagem de acertos no treinamento fosse a escolhida. Na Tabela 9, são listadas as técnicas testadas e os respectivos percentuais de acertos.

Tabela 9 - Porcentagem de acertos de técnicas testadas

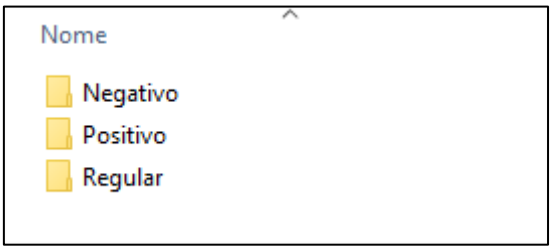
Técnica de Mineração de Textos	**Porcentagem de Acertos (%)**
Bayes Net	61,00%
Naïve Bayes	89,71%
Support Vector Machines - SVM	85,29%
J48	64,70%
Multilayer Perceptron	72,05%
K-Nearest Neighbor - KNN	69,11%

Fonte: Próprio autor.

Conforme indicado pela Tabela 9, a técnica com o melhor resultado foi a técnica de *Naïve Bayes*, com uma porcentagem de acertos de 89,71%. Portanto, essa foi a escolhida para o treinamento da aplicação deste trabalho.

Para que o treinamento com os arquivos salvos no banco de dados fosse realizado, tornou-se necessário exportar os arquivos em um diretório temporário do servidor da aplicação, para que pudesse ser novamente importada para a aplicação utilizando a classe *TextDirectoryLoader* do *Weka*. Essa classe é responsável pela leitura de arquivos textos em diretórios físicos, e ainda permite a criação automática da classe do treinamento, de acordo com as pastas aos quais os arquivos foram separados. A Figura 13 demonstra a pré-classificação dos arquivos textos conforme suas pastas.

Figura 13 - Divisão das pastas para criação da classe de treinamento automática



Fonte: Próprio autor.

Assim que os arquivos são importados para a aplicação, os filtros *StringToWordVector* e *InfoGainAttributeEval* são incluídos em um novo filtro de nome *MultiFilter*. Esse filtro permite acoplar outros inúmeros filtros para que possam ser aplicados sequencialmente ao conjunto de dados.

Também se tornou necessário a utilização do classificador *FilteredClassifier* que, semelhante ao *MultiFilter*, permite acoplar o classificador escolhido e os filtros. Dessa forma, ao construir o classificador, os filtros são previamente aplicados ao conjunto de dados. No Algoritmo 6, é demonstrado a amostra de código que realiza o treinamento do classificador.

Algoritmo 6 - Amostra de código do Treinamento do Classificador.

```
// Function desenvolvida na linguagem C#
private void RealizaTreinamentoWeka()
{
   // Importa os arquivos do servidor, criando uma classe Instances
   Instances dadosTreinamento = ImportaArquivosServidor();

   // Adiciona os filtros ao MultiFilter
   weka.filters.Filter[] filters = new weka.filters.Filter[2];
   filters[0] = new StringToWordVector();
   filters[1] = new InfoGainAttributeEval();

   weka.filters.MultiFilter filter = new weka.filters.MultiFilter();
   filter.setInputFormat(dadosTreinamento);
   filter.setFilters(filters);

   // Cria o FilteredClassifier e adiciona o MultiFilter e NaiveBayes
   FilteredClassifier classifier = new FilteredClassifier();
   classifier.setFilter(filter);
   classifier.setClassifier(new NaiveBayes());
   classifier.buildClassifier(dadosTreinamento);

   // Salva o Classificador (model) como /Classificador/Classificador.model
   SerializationHelper.write(string.Format("{0}Classificador.model",
DiretorioClassificadorServidor), classifier);

   // Salva os dados de treinamento como /Classificador/DadosTreinamento.arff
   Utilidades.SalvarArff(dadosTreinamento, DiretorioClassificadorServidor,
"DadosTreinamento.arff");
}
```

Fonte: Próprio autor.

3.4.2.4 Classificando novas instâncias

A classificação das novas instâncias foi o ponto de maior dificuldade deste trabalho, pois para que novas instâncias pudessem ser classificadas, é necessário que a mesma possua o mesmo número de atributos dos dados de treinamento. Caso contrário, a nova instância seria classificada de forma incorreta, por não utilizar o mesmo dicionário de palavras do treinamento.

Dessa forma, o classificador construído e os dados de treinamento tiveram de ser salvos no diretório do servidor, para que fossem utilizados na classificação das novas instâncias, conforme demonstrado em Algoritmo 7.

Algoritmo 7 - Classificação de novas instâncias utilizando o classificador treinado.

```
// Function desenvolvida na linguagem C#
private Arquivo AvaliarWekaClassificacao(Arquivo _arquivoAvaliado)
{
   // Importa o Classificador que foi salvo pelo treinamento
   FilteredClassifier classifier = (FilteredClassifier)ImportaClassificadorSalvo();

   // Importa os Dados de Treinamento salvo pelo treinamento
   Instances dadosTreinamento = ImportaDadosTreinamentoSalvo();
   dadosTreinamento.setClassIndex(dadosTreinamento.numAttributes()-1);

   int numAttributes = dadosTreinamento.numAttributes();

   // Cria a instância de teste
   Instance instance = new DenseInstance(numAttributes);
   instance.setDataset(dadosTreinamento);

   // Inicializa todos os atributos como valor zero.
   for (int i = 0; i < numAttributes; i++)
      instance.setValue(i, 0);

   // Insere o texto a ser avaliado no primeiro atributo
   for (int i = 0; i < numAttributes-1; i++)
   {
      instance.setValue(i, _arquivoAvaliado.TextoFormatado);
   }

   // Indica que a Classe é o resultado sendo procurado
   instance.setClassMissing();

   // Classifica a instância de teste
   var resultado = "";
   try
   {
      // Realiza a classificação da instância, retornando o resultado previsto
      var predicao = classifier.classifyInstance(instance);
      instance.setClassValue(predicao);

      // Realiza a tradução do resultado numérico na Classificação esperada
      resultado = dadosTreinamento.classAttribute().value((int) predicao);

   } catch (Exception)
   {
      throw (new ClassificationException("O texto não pode ser classificado
quanto à sua qualidade."));
   }

   // Atribui o resultado ao arquivo avaliado
   _arquivoAvaliado.Classificacao = resultado;

   return _arquivoAvaliado;
}
```

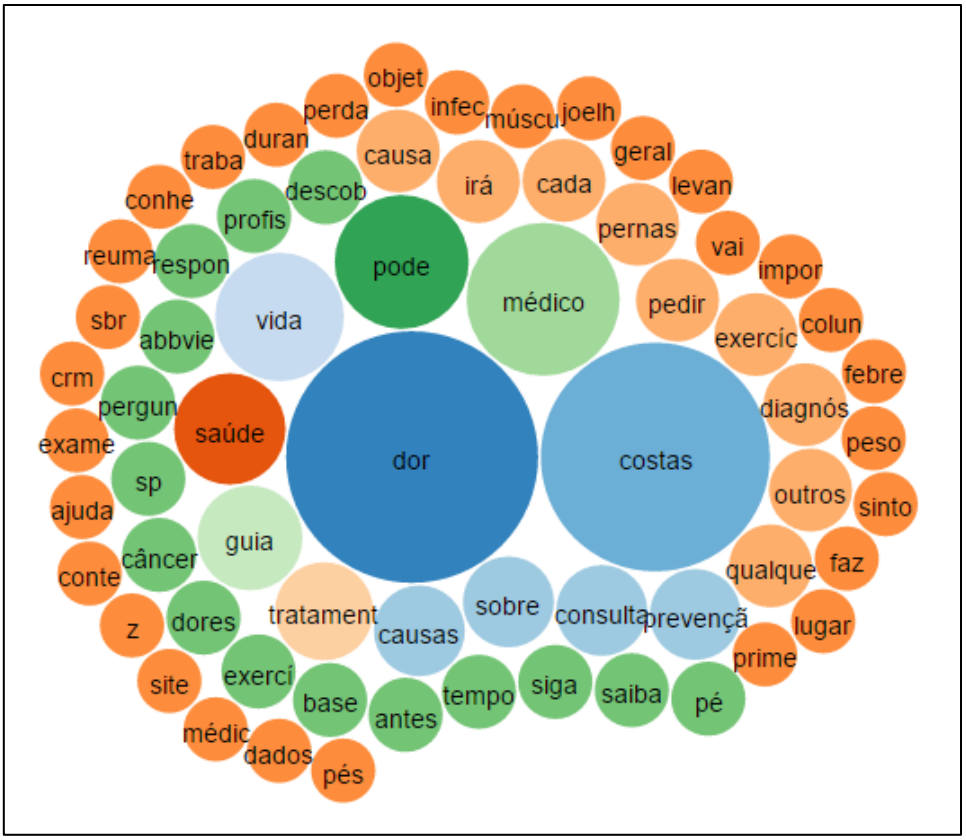
Fonte: Próprio autor.

3.4.3 Visualização dos Resultados

Para a visualização e apresentação dos resultados da avaliação textual, utilizou-se das bibliotecas *javascript D3.JS* e *jqPlot* para a geração dos componentes gráficos para as visualizações.

A biblioteca *D3.JS* foi anteriormente utilizada por Lovato (2015) no seu trabalho de conclusão de curso, onde implementou-se a visualização dos resultados para o software *Undermine Text Miner* (CRISTOFOLI, 2012). Na ocasião, selecionou-se os formatos de visualização mais aplicados à textos. E então, implementou-se um servidor *web* que utiliza a biblioteca *D3.JS* para ser integrado ao *Undermine Text Miner* desenvolvido em plataforma *desktop*. A implementação permitia a geração de diversos formatos de visualização, como por exemplo *BarChart*, *BubbleChart*, *CollapsibleTree*, *Rotating Cluster*, *TreeMap* e *Word Cloud*.

Figura 14 - Gráfico de Bolhas ou *Bubble Chart*



Fonte: Próprio autor.

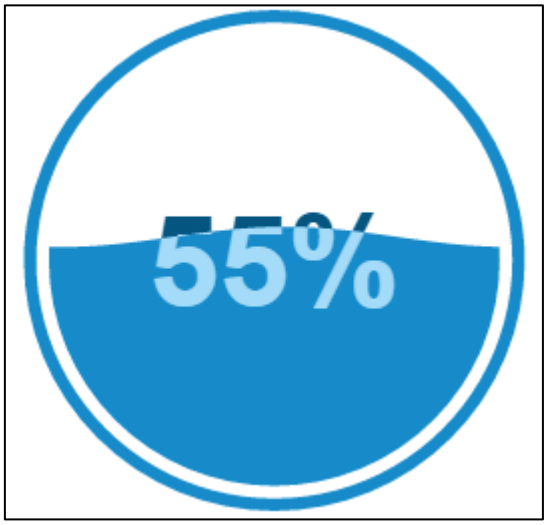
Figura 15 - Nuvem de Palavras ou *Word Cloud*

Fonte: Próprio autor.

Dos formatos de visualização selecionados e utilizados por Lovato, apenas os formatos *BubbleChart* (Figura 14Figura 15) e *WordCloud* (Figura 15) são capazes de representar a relevância de cada palavra dentro de um conjunto de textos, portanto esses formatos foram utilizados em nossa aplicação.

Além destes, foram incluídos em nossa aplicação alguns componentes gráficos conhecidos como *gauges*, que traduzidos para a língua Portuguesa, correspondem a medidores ou contadores. A biblioteca *D3.JS* possui um componente do formato *gauge*, conhecido como *Liquid Fill Gauge*, e é representado na Figura 16.

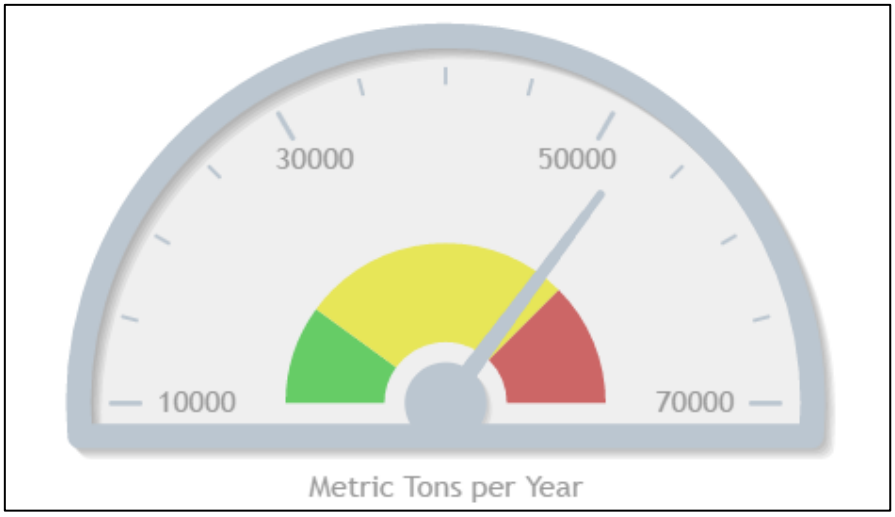
Figura 16 - Componente gráfico *Liquid Fill Gauge* da biblioteca *D3.JS*



Fonte: http://bl.ocks.org/brattonc/5e5ce9beee483220e2f6

Outro componente gráfico do tipo *gauge*, o *Meter Gauge* da biblioteca *jqPlot,* foi incluído à nossa aplicação. O mesmo é uma representação de um velocímetro, e muito útil para dividir os resultados por faixas de valores, conforme representado em Figura 17.

Figura 17 - Componente gráfico *Meter Gauge* da biblioteca *jqPlot*



Fonte: http://www.jqplot.com/examples/meterGauge.php

3.4.4 Hospedagem

A aplicação foi hospedada por meio do site *Somee.com*, que possui um plano gratuito de hospedagem de aplicações *.Net*. O plano gratuito conta com 150MB de armazenamento e suporta as plataformas *ASP.NET*, além de disponibilizar 15MB para armazenamento de dados através do SQL Server. A hospedagem foi realizada por meio de um acesso FTP que o site oferece.

O link para acesso ao site é http://minning.somee.com/.

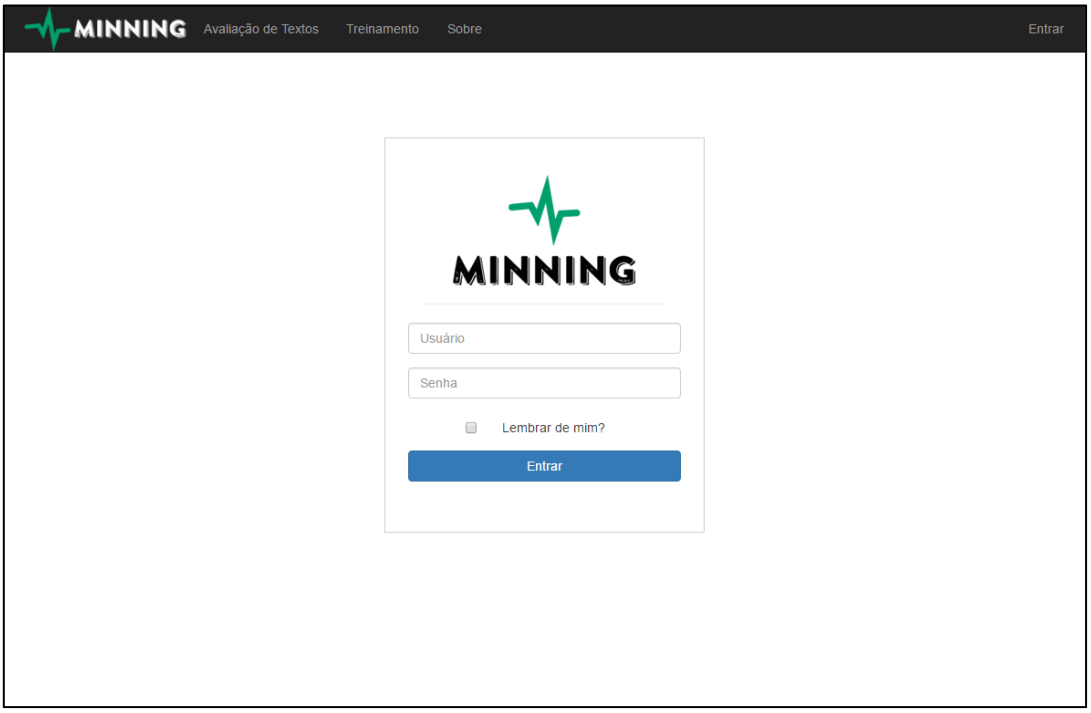
3.5 INTERFACE

Esta seção apresenta a interface visual da aplicação implementada. A aplicação é separada em etapa de treinamento da etapa de avaliação de textos e apresentação dos resultados.

3.5.1 Etapa de Treinamento

A etapa de treinamento da aplicação somente pode ser acessada e gerenciada por usuários identificados através do acesso ao site. Dessa forma, foi implementado uma página para realizar o acesso a um usuário (*login*). Assim, um usuário administrador pode conectar-se e realizar o treinamento do avaliador, conforme exibida em Figura 18.

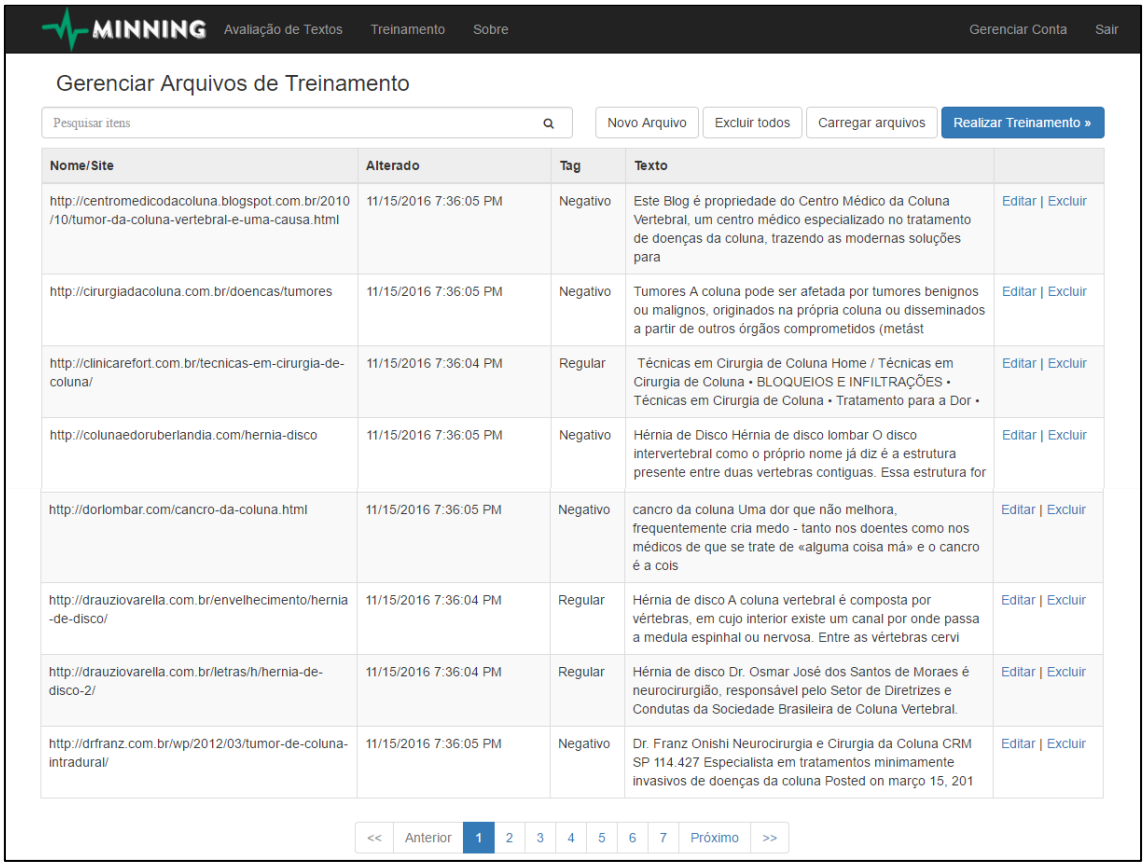
Figura 18 - Página de acesso ao site



Fonte: Próprio autor.

A página “Gerenciar Arquivos de Treinamento” é acessada através do clique no botão “Treinamento” no menu superior. Nessa página, é possível incluir um novo arquivo de treinamento manualmente, através do botão “Novo Arquivo”. Também é possível realizar o *upload* de vários arquivos, através do botão “Carregar arquivos”, além de editar ou excluir os arquivos, e realizar o treinamento, conforme mostrado na Figura 19.

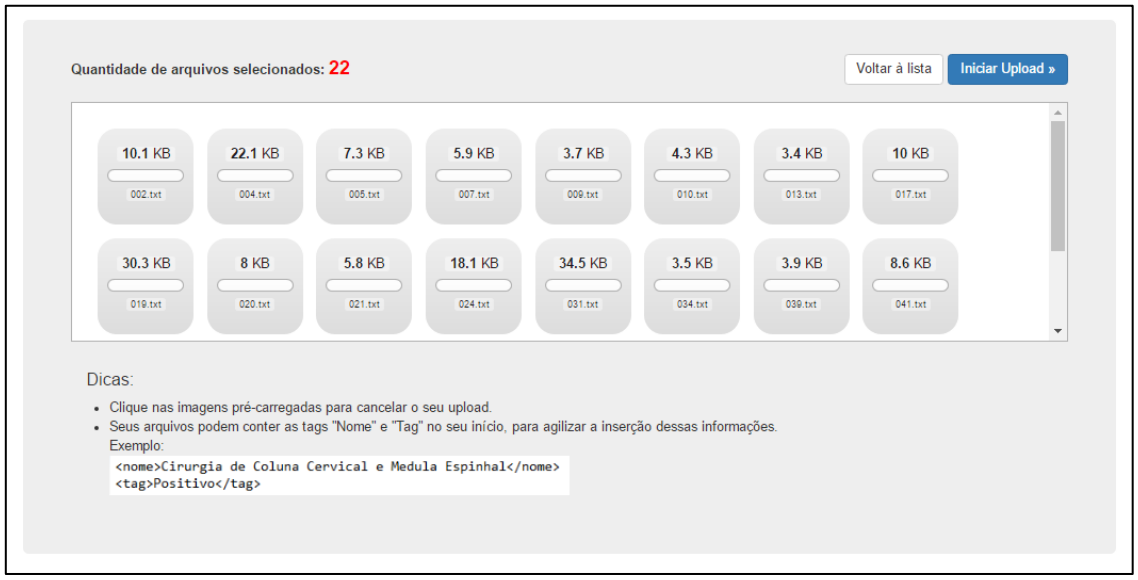
Figura 19 - Página de gerenciamento dos arquivos de treinamento



Fonte: Próprio autor.

Geralmente, um treinamento possui uma grande quantidade de arquivos a serem adicionados, dessa forma, é imprescindível que a funcionalidade do *upload* de vários arquivos seja utilizada. Ao clicar em “Carregar arquivos”, a página é carregada, conforme Figura 20.

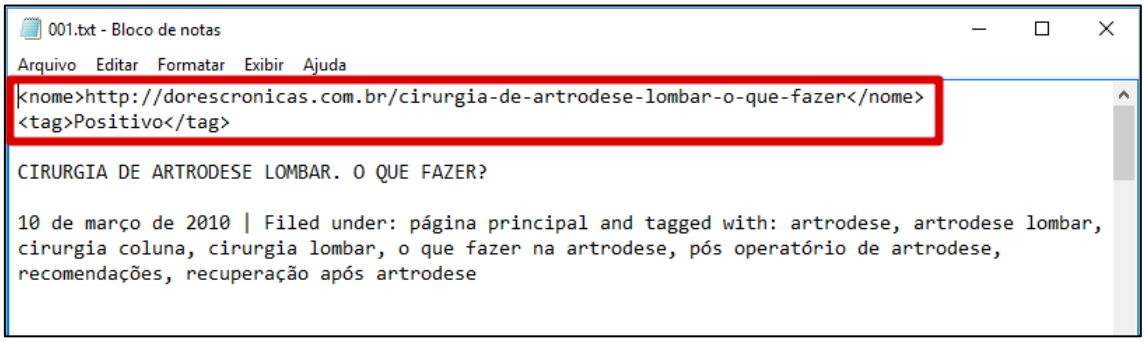
Figura 20 - Upload de arquivos de treinamento



Fonte: Próprio autor.

Para agilizar o processo de informar as classificações (*tags*) de vários arquivos, é possível que dentro de cada arquivo sejam incluídos dois atributos: nome e *tag* do arquivo. Essas informações serão importadas para a base de dados da aplicação, facilitando a inclusão de múltiplos arquivos. Um exemplo de arquivo com os atributos informados é representado em Figura 21.

Figura 21 - Arquivo de treinamento com os atributos de "nome" e "tag"



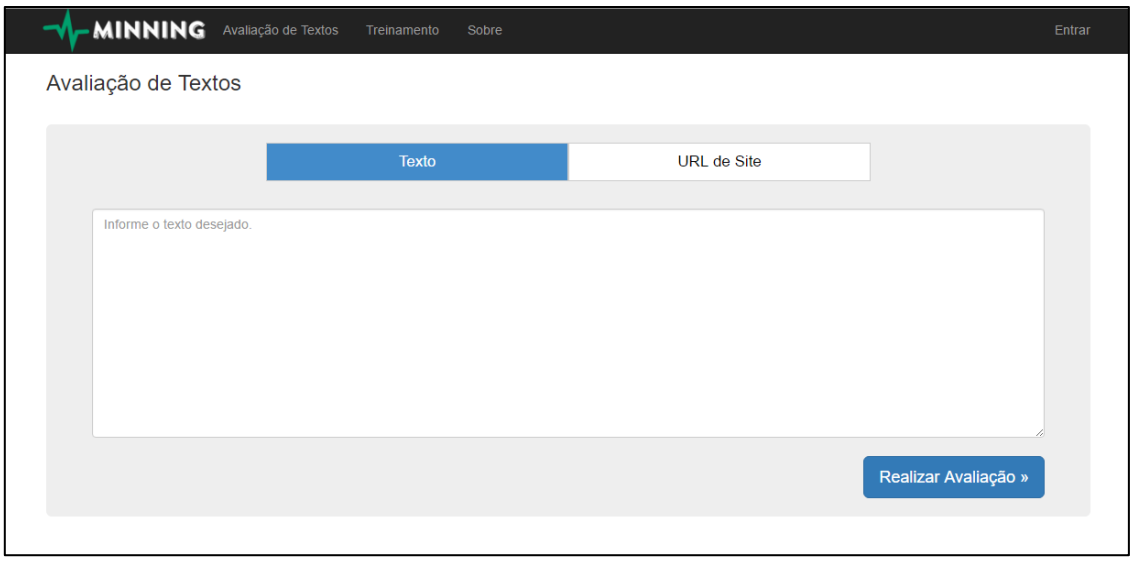
Fonte: Próprio autor.

3.5.2 Etapa de Avaliação de Textos

A etapa de avaliação dos textos é a página inicial da aplicação, e pode ser realizada por qualquer usuário, sem a necessidade de identificação. Nessa página, acessada também pela opção “Avaliação de Textos” no menu, são

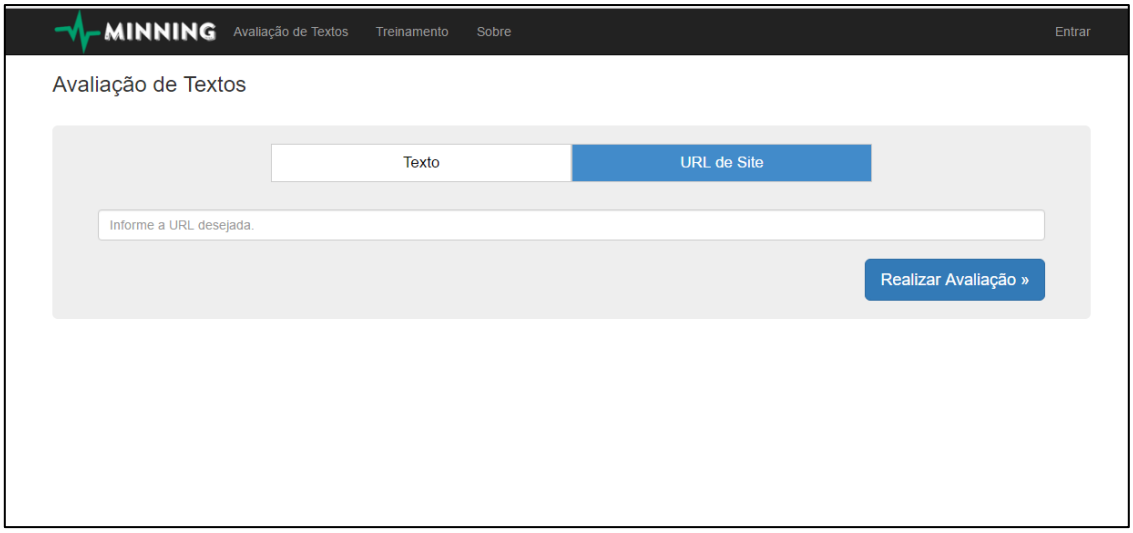
exibidas para o usuário as opções de informar um texto ou a URL de um site. As opções são separadas por abas, conforme mostrado na Figura 22 e Figura 23.

Figura 22 - Interface de entrada dos dados em formato texto para avaliação



Fonte: Próprio autor.

Figura 23 - Interface de entrada dos dados em formato URL para avaliação



Fonte: Próprio autor.

Para que a etapa de avaliação de textos possa ser utilizada por qualquer usuário, é necessário que o treinamento tenha sido realizado por um usuário administrador do sistema. Quando a informação textual é submetida ao avaliador, através do botão “Realizar Avaliação”, a apresentação dos resultados da avaliação textual é exibida, conforme o próximo tópico.

3.5.3 Visualização dos Resultados

A visualização dos resultados é dividida por abas: resultados da apreensibilidade, resultados da qualidade, e detalhes sobre o texto avaliado. A página de visualização dos resultados foi construída em um formato *dashboard*, ou painel de indicadores.

3.5.3.1 Apreensibilidade

Na aba de apresentação de resultados da avaliação da apreensibilidade, são utilizados alguns componentes gráficos que tornam o resultado mais claro e compreensível, conhecidos como *gauges*, que traduzidos para a língua Portuguesa, correspondem a medidores ou contadores.

A apreensibilidade foi calculada a partir da fórmula de *Fernández-Huerta*, que tem como resultado um número na escala de 0 a 100. O resultado da tal fórmula é apresentado através de um componente da biblioteca *D3.JS*, chamado *Liquid Fill Gauge*. Em complemento a isso, utilizou-se uma escala de cores que tornasse nítido a interpretação do resultado, conforme pode ser observado nas Figura 24, Figura 25, Figura 26, Figura 27 e Figura 28.

Figura 24 - *Liquid Fill Gauge* para resultados entre 90 e 100 da fórmula de *Fernández-Huerta*

Fonte: Próprio autor.

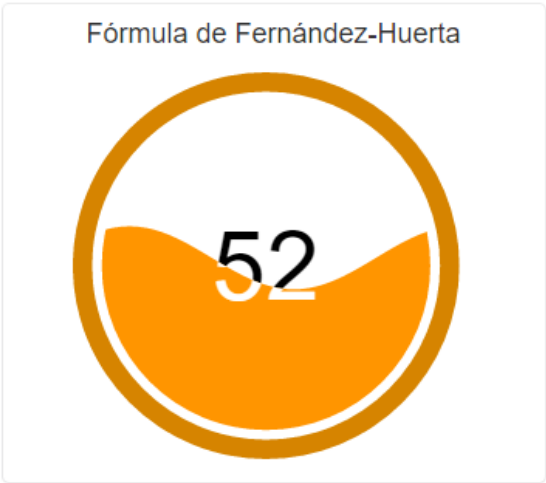
Figura 25 - *Liquid Fill Gauge* para resultados entre 70 e 90 da fórmula de *Fernández-Huerta*

Fonte: Próprio autor.

Figura 26 - *Liquid Fill Gauge* para resultados entre 60 e 70 da fórmula de *Fernández-Huerta*

Fonte: Próprio autor.

Figura 27 - *Liquid Fill Gauge* para resultados entre 30 e 60 da fórmula de *Fernández-Huerta*



Fonte: Próprio autor.

Figura 28 - *Liquid Fill Gauge* para resultados entre 0 e 30 da fórmula de *Fernández-Huerta*

Fonte: Próprio autor.

Baseado no resultado da Fórmula de *Fernández-Huerta*, algumas interpretações sobre o texto podem ser indicadas, quando aplicadas à Tabela 7, são elas: a dificuldade de leitura do texto e a escolaridade necessária aproximada para o pleno entendimento do texto.

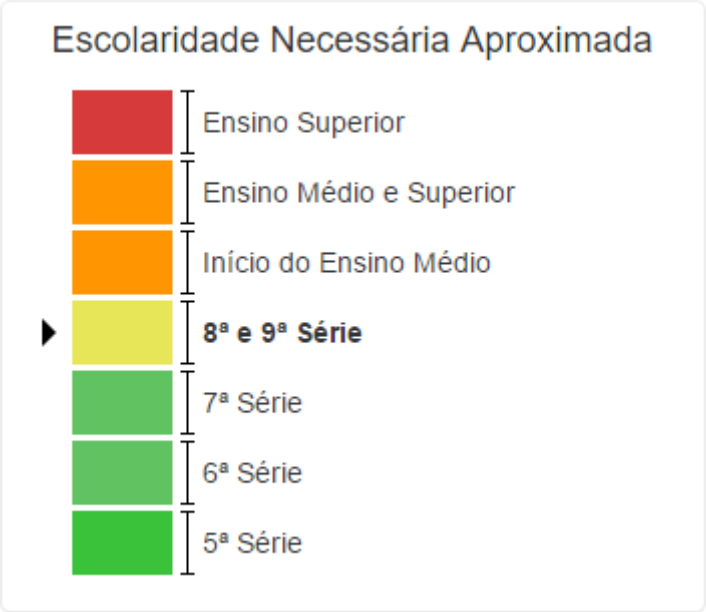
Para representação da dificuldade de leitura, utilizou-se o componente *Meter Gauge* da biblioteca *jqPlot*, conforme exibido na Figura 29.

Figura 29 - *Meter Gauge* utilizada para medição da dificuldade de leitura do texto

Fonte: Próprio autor.

Para representação da escolaridade necessária aproximada para o entendimento do texto, utilizou-se o conceito de uma simples escala, conforme exibida na Figura 30.

Figura 30 - Escala utilizada para medição da escolaridade mínima necessária aproximada para o entendimento completo do texto

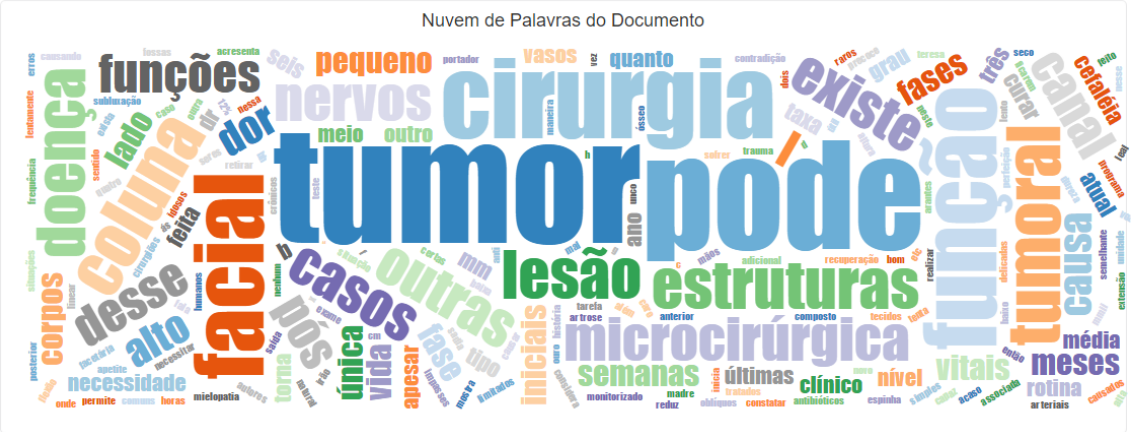


Fonte: Próprio autor.

Nota-se que a mesma escala de cores foi mantida para facilitar o reconhecimento dos conceitos pelo usuário. Isso torna o sistema intuitivo e de fácil memorização.

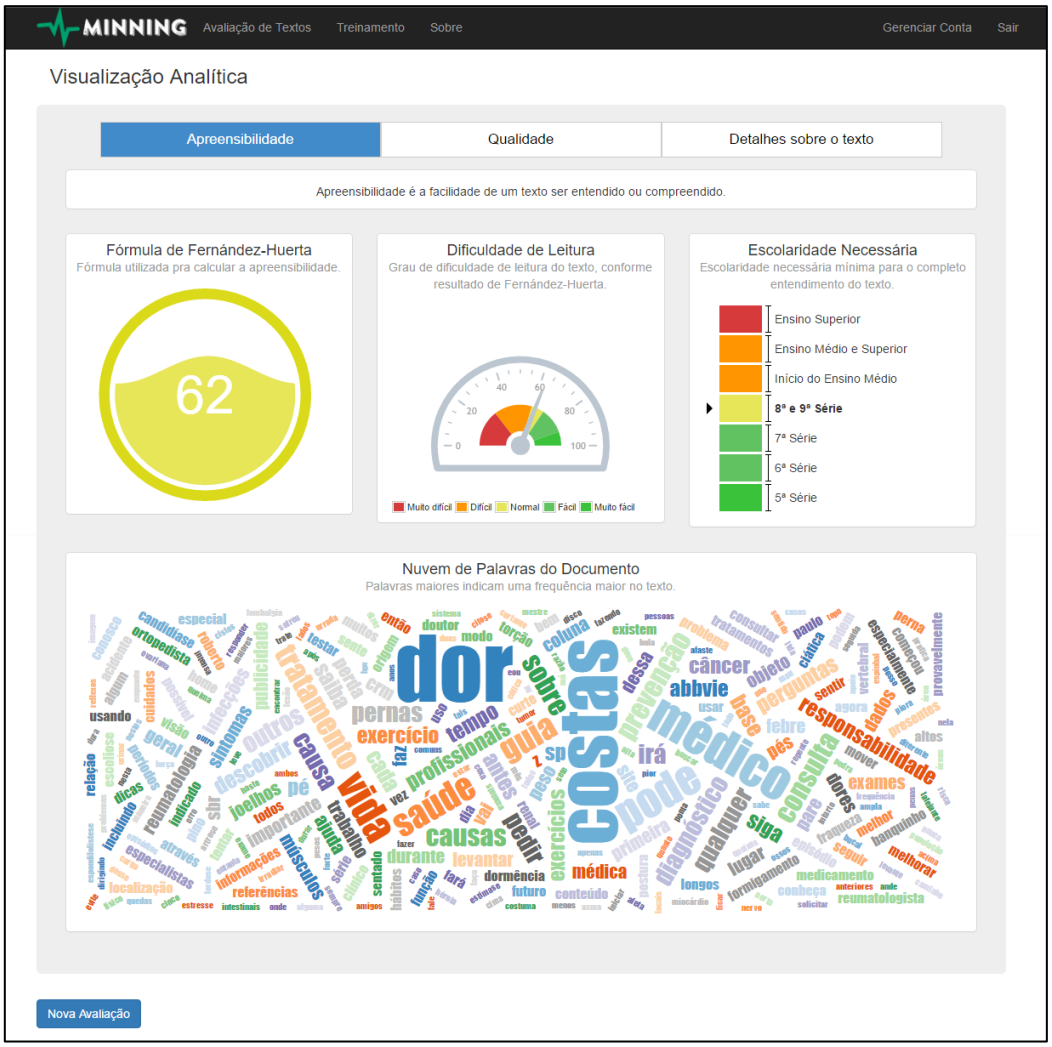
Por final, uma nuvem das palavras mais frequentes do documento avaliado foi criada através do componente *WordCloud* da biblioteca *D3.JS*. Nesse gráfico, exibido em Figura 31, as maiores palavras representam as de maior frequência no texto, enquanto palavras menores representam uma frequência de no mínimo uma ocorrência. As *stopwords* do texto foram removidas para evitar a exibição de palavras que não são tão relevantes para o tema. A página de resultados da avaliação da apreensibilidade é exibida por completo na Figura 32.

Figura 31 - Nuvem de palavras mais frequentes do texto avaliado (*D3.JS*)



Fonte: Próprio autor.

Figura 32 - Painel de indicadores (*dashboard*) dos resultados da avaliação da apreensibilidade



Fonte: Próprio autor.

3.5.3.2 Qualidade

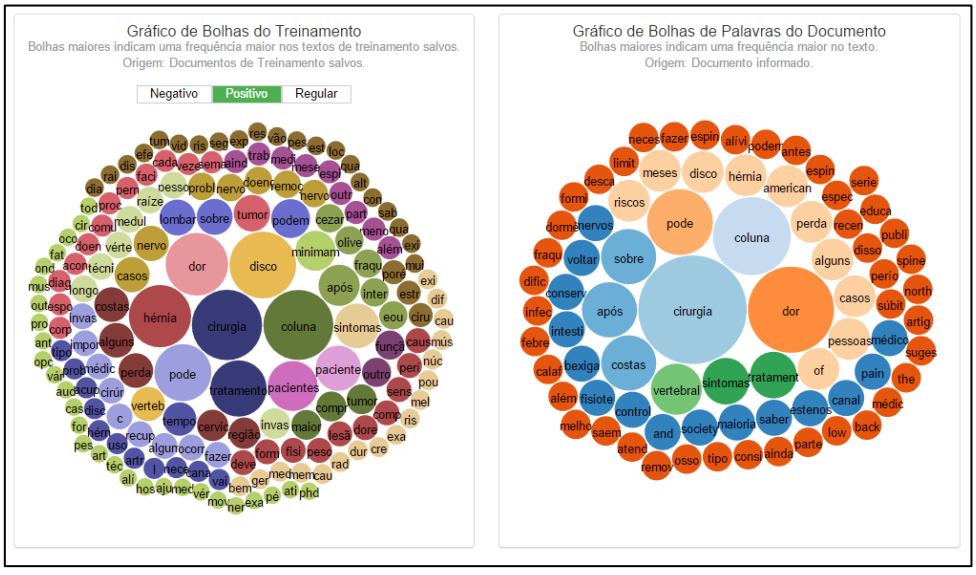
Na aba de apresentação de resultados da avaliação da qualidade, o resultado do classificador é exibido em formato literal, conforme Figura 33, e é mostrado de acordo com as possíveis *tags* presentes nos arquivos de treinamento.

Figura 33 - Resultado da Avaliação da Qualidade

Fonte: Próprio autor.

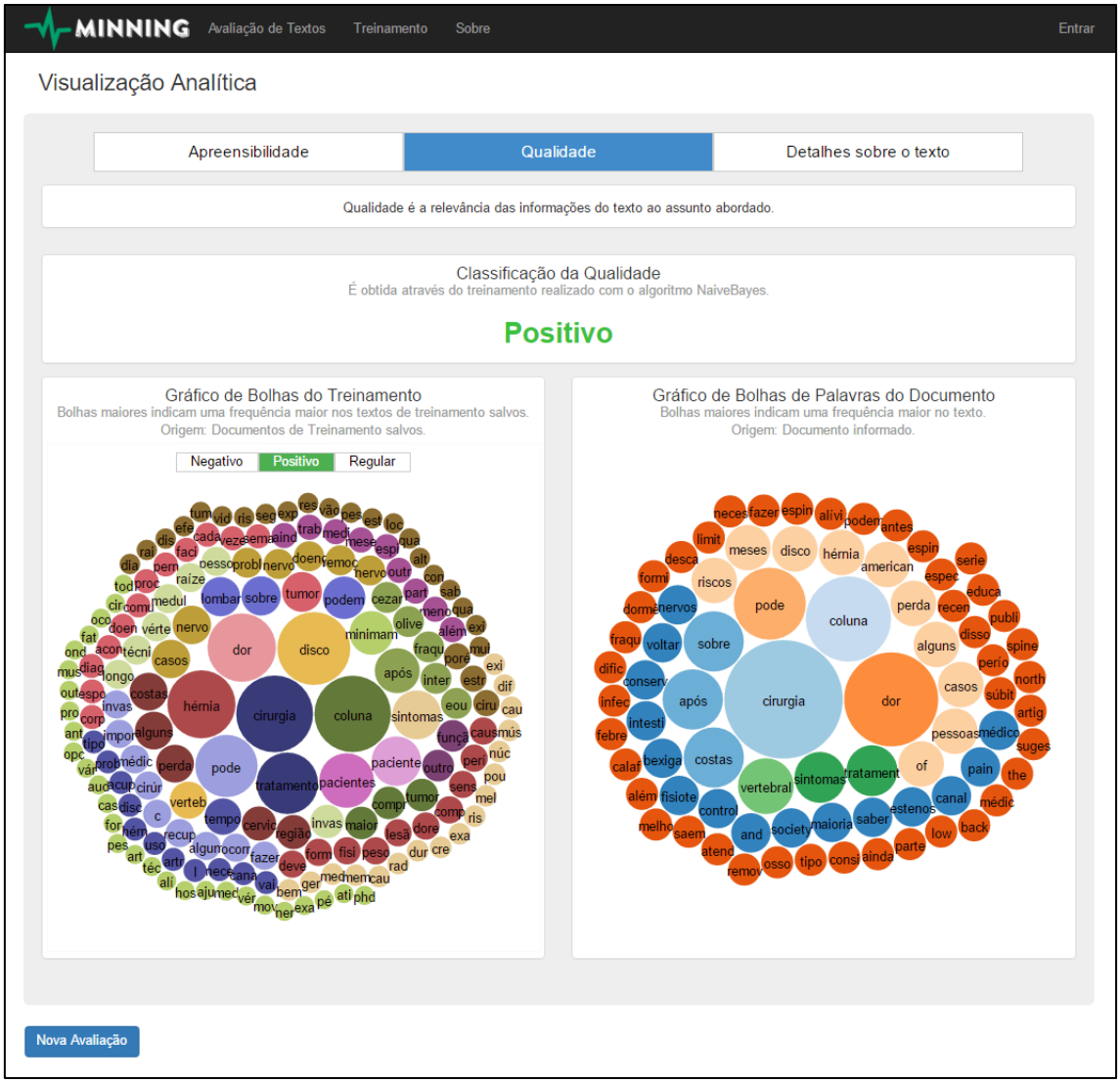
Além disso, também foi utilizado o componente gráfico *BubbleChart* da biblioteca *D3.JS*, para exibir dois gráficos de bolhas para o usuário: o primeiro com as palavras mais frequentes no texto submetido para avaliação, e o segundo com as palavras mais frequentes dos arquivos do treinamento, separados por *tag*. Dessa maneira, o usuário pode comparar as duas visualizações e identificar as palavras em comum com outros exemplos, conforme exibido em Figura 34. A página de resultados da avaliação da qualidade é exibida por completo na Figura 35.

Figura 34 - Gráficos de Bolhas (*BubbleChart*) dos arquivos de treinamento e do documento



Fonte: Próprio autor.

Figura 35 - Painel de indicadores (*dashboard*) dos resultados da avaliação da qualidade



Fonte: Próprio autor.

3.5.3.3 Detalhes sobre o texto

Na aba de detalhes sobre o texto, são exibidas as informações do texto submetido. Ao informar a URL de um site, por exemplo, é exibido a URL inserida e o respectivo texto extraído da mesma.

3.6 RESULTADOS EM RELAÇÃO À QUALIDADE

A avaliação dos resultados da ferramenta foi realizada comparando-se os resultados da ferramenta com as análises de especialistas humanos. Os dados textuais foram inicialmente avaliados pelos especialistas da área da saúde, possibilitando que os resultados da ferramenta fossem comparados com tais avaliações.

A comparação dos resultados da ferramenta e dos especialistas humanos atingiu um percentual de acerto de 89,7%, ou seja, dos 68 textos coletados, o número de 61 textos foi corretamente classificado pela ferramenta. A comparação dos resultados da ferramenta e dos especialistas humanos é demonstrada na Tabela 10, sendo que as linhas marcadas correspondem aos textos classificados incorretamente pela ferramenta.

Tabela 10 - Comparativo entre resultados da ferramenta e de especialistas humanos

# Texto	Resultado humano	Resultado da ferramenta	# Texto	Resultado humano	Resultado da ferramenta
1	Negativo	Negativo	35	Regular	Regular
2	Negativo	Negativo	36	Negativo	Negativo
3	Negativo	Negativo	37	Negativo	Negativo
4	Negativo	Negativo	38	Negativo	Negativo
5	Positivo	Positivo	39	Negativo	Negativo
6	Regular	Regular	40	Positivo	Positivo
7	Regular	Regular	41	Regular	Regular
8	Negativo	Negativo	42	Regular	Negativo
9	Negativo	Negativo	43	Negativo	Negativo
10	Negativo	Negativo	44	Negativo	Negativo
11	Negativo	Negativo	45	Negativo	Regular
12	Positivo	Positivo	46	Negativo	Negativo
13	Regular	Regular	47	Regular	Regular
14	Regular	Regular	48	Regular	Regular
15	Negativo	Negativo	49	Regular	Negativo
16	Negativo	Negativo	50	Negativo	Negativo
17	Negativo	Negativo	51	Negativo	Positivo
18	Negativo	Negativo	52	Negativo	Negativo
19	Positivo	Positivo	53	Negativo	Negativo
20	Regular	Regular	54	Regular	Regular
21	Regular	Regular	55	Regular	Regular
22	Negativo	Negativo	56	Regular	Regular
23	Negativo	Negativo	57	Negativo	Negativo
24	Negativo	Negativo	58	Negativo	Negativo
25	Negativo	Negativo	59	Negativo	Negativo
26	Positivo	Negativo	60	Negativo	Negativo
27	Regular	Regular	61	Regular	Regular
28	Regular	Regular	62	Regular	Regular
29	Negativo	Negativo	63	Negativo	Negativo
30	Negativo	Negativo	64	Negativo	Negativo
31	Negativo	Negativo	65	Negativo	Negativo
32	Negativo	Negativo	66	Positivo	Regular
33	Positivo	Positivo	67	Regular	Negativo
34	Regular	Regular	68	Regular	Regular

Fonte: Próprio autor.

Baseado no comparativo dos resultados exibido na Tabela 10, pode-se realizar a apresentação da matriz de confusão. Uma matriz de confusão é capaz de explicitar a diferença entre o resultado esperado e o resultado obtido pela ferramenta. Por exemplo, em um treinamento realizado com 100% de acertos, a matriz de confusão é somente preenchida com valores na sua diagonal.

A Tabela 11 representa a matriz de confusão obtida, a partir dos resultados do treinamento realizado. Em uma matriz de confusão, um bom modelo apresenta altos valores na sua diagonal principal.

Tabela 11 - Matriz de confusão do treinamento da ferramenta

Resultado obtido ►	**Positivo**	**Regular**	**Negativo**
Positivo	5	1	1
Regular	0	19	3
Negativo	1	1	37

Fonte: Próprio autor.

Na diagonal principal, são encontrados os elementos que foram corretamente classificados, e estes elementos são conhecidos por Verdadeiros Positivos (VP) e Verdadeiros Negativos (VN). A partir da soma dos elementos da diagonal principal, a taxa de acerto do modelo pode ser encontrada, através da fórmula:

$$Taxa\ de\ Acerto = \frac{VP + VR + VN}{Total} = \frac{5 + 19 + 37}{68} = 0{,}89\ ou\ 89\%$$

Além disso, algumas medidas para avaliação de modelos podem ser utilizadas para medir a precisão por classe, como por exemplo a Sensitividade ou Especificidade. A sensitividade das classes do nosso modelo é representada pela Tabela 12, e pode ser encontrada por meio da divisão da quantidade de valores corretos pela quantidade total da classe.

Tabela 12 - Sensitividade ou Especificidade por Classe

Classe	Quantidade de Acertos	Quantidade total da classe	Sensitividade ou Especificidade
Positivo	5	7	71,24%
Regular	19	22	86,36%
Negativo	37	39	94,87%

Fonte: Próprio autor.

Analisando a Tabela 12, torna-se evidente que o modelo criado é preciso na classificação de resultados negativos e regulares, com um resultado de 94,87% e 86,36% de sensitividade, respectivamente. Esses números provavelmente se dão, devido a quantidade de dados das classes Negativa e Regular que foram utilizados no treinamento serem muito maiores do que comparados à classe Positiva.

Em razão do modelo criado ser utilizado para classificação de dados da área da saúde, dados originalmente negativos que não forem classificados como negativos podem representar um risco a saúde de um paciente, que busca um tratamento ou diagnóstico para si. Dessa forma, apesar da sensitividade de resultados positivos ser de apenas 71,24%, uma sensitividade de aproximadamente 95% para resultados negativos representa um ótimo resultado para a ferramenta, pois torna-se capaz de identificar os sites de baixa confiabilidade.

A partir da análise desses resultados, pode-se então concluir que:

a) A ferramenta é capaz de fornecer avaliações precisas da qualidade dos textos, vide resultados das classes Negativa e Regular.
b) É necessário a inclusão de mais arquivos de treinamento que exemplifiquem um resultado Positivo, para que assim, a sensitividade dessa classe possa aumentar e, consequentemente, tornar a ferramenta ainda mais precisa.

4 CONCLUSÃO

Este capítulo visa a apresentação de uma síntese das atividades desenvolvidas durante o desenvolvimento deste trabalho, descrevendo, também, as contribuições deste trabalho e ideias para trabalhos futuros.

4.1 SÍNTESE

O acesso facilitado de pessoas e pacientes a fontes de informação sobre saúde ampliou a necessidade de que tais informações disponíveis sejam revisadas e analisadas. Levando em consideração que, atualmente, a análise dessas informações disponíveis em Português na Internet necessita ser realizada manualmente por especialistas da área da saúde, é imprescindível que exista um instrumento que auxilie nessa tarefa de maneira automática, utilizando a tecnologia para tal.

O principal objetivo deste trabalho foi a concepção de uma ferramenta *web* para análise dessas produções textuais disponíveis na língua Portuguesa, a partir da avaliação da apreensibilidade e qualidade do conteúdo textual.

Para que esse objetivo fosse atingido, foram apresentados estudos sobre técnicas de avaliação da apreensibilidade utilizadas na área da saúde. Também foram abordados estudos sobre os conceitos e técnicas da Mineração de Textos, visando a avaliação da qualidade.

Em seguida, uma equipe de especialistas da área da saúde disponibilizou um conjunto de amostras textuais analisadas, permitindo que os resultados das técnicas abordadas fossem comparados com as análises dos especialistas. Dessa maneira, foram selecionadas a fórmula de avaliação da apreensibilidade e a técnica de Mineração de Textos ideais para o propósito da funcionalidade proposta pela ferramenta.

Com base no conhecimento adquirido, tornou-se possível o desenvolvimento da aplicação capaz de analisar as produções textuais da área da saúde. A partir do treinamento da ferramenta com base no conjunto de dados de treinamento disponibilizados, novas produções textuais podem ser processadas e analisadas. Por meio da análise de novas produções textuais, o

sistema pode inferir resultados sobre a apreensibilidade e qualidade das mesmas, e apresentar ao usuário diversos formatos de visualizações gráficas capazes de aumentar a cognição humana sobre os resultados.

Com o sistema desenvolvido, considera-se que os resultados são promissores e evidenciam a viabilidade de uso de técnicas de aprendizado automático no tratamento de textos da área da saúde.

4.2 CONTRIBUIÇÕES DO TRABALHO

A área da ciência da computação é uma área em constante progresso, e há muito tempo, a sua evolução segue beneficiando outras áreas do conhecimento, como a da saúde.

Ao início desse trabalho, a expectativa era de que a ferramenta desenvolvida permitisse oferecer apoio à área da saúde, que careciam de instrumentos que possibilitassem avaliar textos e artigos escritos na língua Portuguesa acerca de sua apreensibilidade e qualidade.

Atualmente, é concludente que a sua realização somente foi possível dado a expansão do conhecimento, realizada por meio de pesquisas por assuntos que ultrapassam os estudos presentes no currículo do curso de graduação. E como herança, o presente trabalho deixou importantes contribuições, em especial as áreas da Ciência da Computação e Medicina.

A área da Ciência da Computação foi favorecida de um maior conhecimento acerca dos estudos da Inteligência Artificial e Mineração de Textos, permitindo que os conceitos de apreensibilidade e Mineração de Textos abordados nesse trabalho, possam também servir de fonte de referência a futuros projetos e estudos relacionados.

Enquanto a área da Medicina, foi enriquecida com um instrumento que permite que melhorar a percepção sobre o conteúdo disponível na Internet sobre saúde, especialmente caso pacientes tenham a necessidade de avaliar a confiabilidade a respeito de um diagnóstico e respectivo tratamento.

4.3 TRABALHOS FUTUROS

O presente trabalho realizou a concepção de um sistema *web* para análise de textos da área da saúde escritos em língua Portuguesa, utilizando uma fórmula de apreensibilidade e um algoritmo de Mineração de Textos.

Como sugestão para trabalhos futuros, é proposto o estudo de novos algoritmos de Mineração de Textos com o objetivo de encontrar resultados cada vez mais satisfatórios para a avaliação da qualidade do conteúdo.

A precisão do treinamento pode ser aprimorada, por meio da identificação dos motivos de resultados falsos positivos terem sido obtidos.

A apresentação dos resultados pode ser enriquecida com novos componentes gráficos interativos que são desenvolvidos ao passar do tempo, visando aprimorar o entendimento cognitivo das informações.

Também seria interessante a evolução da aplicação em um sistema multiusuários, acessível e compartilhado, que fosse composto por uma grande base de dados, reunindo informações relevantes e confiáveis sobre doenças, tratamentos e diagnósticos.

REFERÊNCIAS BIBLIOGRÁFICAS

ARANHA, C. **Uma abordagem de pré-processamento automático para mineração de textos em português: sob o enfoque da inteligência computacional**. Departamento de Engenharia Elétrica, PUC-Rio. Rio de Janeiro. 2007.

ARANHA, C.; PASSOS, E. **A Tecnologia de Mineração de Textos**. PUC-Rio. Rio de Janeiro. 2006.

ASP.NET MVC Overview. **Microsoft**, 2016. Disponivel em: <https://msdn.microsoft.com/en-us/library/dd381412(v=vs.108).aspx>. Acesso em: 29 Mai. 2016.

BARBOZA, E. M. F.; NUNES, E. M. D. A. **A inteligibilidade dos websites governamentais brasileiros e o acesso para usuários com baixo nível de escolaridade**. [S.l.]. 2007.

BARION, E. C. N.; LAGO, D. Mineração de Textos. **Revista de Ciências Exatas e Tecnologia, São Paulo, v. 3, n. 3**, p. 123-140, 2008.

BISHOP, C. M. **Neural Networks for Pattern Recognition**. [S.l.]: Oxford University Press, 1995.

BOSTOCK, M. D3 Data-Driven Documents. **D3 Data-Driven Documents**, 2013. Disponivel em: <http://d3js.org>. Acesso em: 19 Jun. 2016.

BOUCKAERT, R. R. et al. **Weka Manual for Version 3-8-0**. University of Waikato. Hamilton, New Zealand. 2016.

BURBECK, S. Applications programming in smalltalk-80 (tm): How to use model-view-controller (mvc). **Smalltalk-80 v2**, n. 5, 1992.

CAMILO, C. O.; SILVA, J. C. D. **Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas**. Universidade Federal de Goiás. [S.l.]. 2009.

CAYLOR, J. S. et al. **Methodologies for Determining Reading Requirements of Military Occupational Specialties**. Human Resources Research Organization. Alexandria, VA. 1973.

CHALL, J. S.; DALE, E. **Readability revisited:** The new Dale-Chall readability formula. [S.l.]: Brookline Books, 1995.

COLEMAN, M.; LIAU, T. L. A computer readability formula designed for machine scoring. **Journal of Applied Psychology**, v. 60, n. 2, p. 283, 1975.

CRISTOFOLI, L. G. **Mineração de Textos baseada em Algoritmos Imunológicos**. Universidade de Caxias do Sul. Caxias do Sul. 2012.

DALE, E.; CHALL, J. S. A Formula for Predicting Readability. **Educational Research Bulletin**, 27, n. 1, 1948.

DUBAY, W. H. The Principles of Readability. **Impact Information**, Costa Mesa, Cafifornia, 2004.

FELDMAN, R.; SANGER, J. The Text Mining Handbook Advanced Approaches in Analyzing Unstructured Data. **New York: Cambridge University Press**, 2007.

FLESCH, R. Marks of readable style; a study in adult education. **Teachers College Contributions to Education**, 1943.

FLESCH, R. A new readability yardstick. **Journal of applied psychology**, v. 32, n. 3, 1948.

FLESCH, R. **The Art of Readable Writing**. New York: The Haddon Craftsmen, 1949. Disponivel em: <https://dc135.files.wordpress.com/2012/11/flesch-the-art-of-readable-writing.pdf>. Acesso em: 13 abr. 2016.

GARCIA, S. C. **O Uso das Árvores de Decisão na Descoberta de Conhecimento na Área da Saúde**. Universidade Federal do Rio Grande do Sul. Porto Alegre, p. 21. 2003.

GOLDIM, J. R. Consentimento e Informação: A importação da qualidade do texto utilizado. **Rev HCPA**, v. 26, n. 3, p. 177, 2006. Disponivel em: <https://www.ufrgs.br/bioetica/cilegib.pdf>. Acesso em: 28 Abr. 2016.

HAN, J.; KAMBER, M.; PEI, J. **Data Mining:** Concepts and Techniques. 3. ed. [S.l.]: Elsevier, 2012.

KINCAID, J. P. et al. **Derivation of new readability formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy enlisted**. Naval Technical Training Command Millington TN Research Branch. [S.l.]. 1975.

KLARE, G. R. **The measurement of readability**, Ames: Iowa State University Press, 1963.

LAW, G. Error in the Fernandez Huerta Readability Formula. **The Linguist List (International Linguistics Community Online)**, 2011. Disponivel em: <http://linguistlist.org/LL/posttolinguist.cfm>. Acesso em: 29 Abr. 2016.

LEONELLO, C.; PRITCHARD, P. jqPlot Chars and Graphs for jQuery. **jqPlot**. Disponivel em: <http://www.jqplot.com/info.php>. Acesso em: 2016 Out. 16.

LORENA, A. C.; CARVALHO, A. C. D. Uma introdução às support vector machines. **Revista de Informática Teórica e Aplicada**, v. 14, n. 2, p. 43-67, 2007. Disponivel em:

<http://www.seer.ufrgs.br/rita/article/download/rita_v14_n2_p43-67/3543>. Acesso em: 31 Mai. 2016.

LOVATO, G. **Aplicação da Mineração de Textos na Análise de Produções Textuais**. Universidade de Caxias do Sul. Caxias do Sul. 2015.

MARTINS, T. B. F. et al. Readability fórmulas applied to textbooks in Brazilian Portuguese. **Nota do ICMS nº 28. São Carlos**, 1996.

MCLAUGHLIN, G. H. Proposals for British readability measures. **Third international reading symposium eds. Brown and Downing. London: Cassell.**, 1968.

MCLAUGHLIN, G. H. SMOG grading: A new readability formula. **Journal of Reading**, 1969.

OLIVEIRA, H. R. G. **Geração de texto com base em ritmo**. Universidade de Coimbra. Coimbra, p. 45. 2007.

ORENGO, V. M.; HUYCK, C. **A stemming algorithm for the portuguese**. Eighth International Symposium on String Processing (Spire 2001). [S.l.]: [s.n.]. 2001. p. 186.

PETERSON, L.. K-nearest neighbor. **Scholarpedia**, 2009. Disponivel em: <http://scholarpedia.org/article/K-nearest_neighbor>. Acesso em: 27 Mai. 2016.
SAINT-EXUPÉRY, A. D.; WOODS, K.; HARCOURT, B. &. W. **The little prince**. New York: Harcourt, Brace & World, 1943. 59 p. Disponivel em: <http://download.bioon.com.cn/upload/201111/21084046_8501.pdf>. Acesso em: 06 abril 2016.

SENTER, R. J.; SMITH, E. A. **Automated readability index**. University of Cincinnati. Cincinnati. 1967.

SILVA, E. R. C. **Técnicas de Data e Text Mining para anotação de um arquivo digital**. Universidade de Aveiro. Aveiro, Portugal. 2010.

SOLKA, J. L. Text data mining: theory and methods. **Statistics Surveys**, v. 2, p. 94-112, 2008. Disponivel em: <http://projecteuclid.org/download/pdfview_1/euclid.ssu/1216238228>. Acesso em: 16 Mai 2016.

TRIOLA, M. F. **Introdução à Estatística**. 7. ed. [S.l.]: LTC, 1999.

WEKA. Weka - Stemmers. **Weka - Stemmers**, 2016. Disponivel em: <https://weka.wikispaces.com/Stemmers>. Acesso em: 15 Nov 2016.

WITTEN, I. H.; FRANK, E. **Data Mining:** Practical Machine Learning Tools and Techniques. 2. ed. [S.l.]: Elsevier, 2005.

APÊNDICE A

LISTA DE STOPWORDS

de	tem	quem	essas	aqueles	havemos	eram	tínhamos
a	à	nas	esses	aquelas	hão	fui	tinham
o	ser	me	pelas	isto	houve	foi	tive
que	seu	esse	este	aquilo	houvemos	fomos	teve
e	sua	eles	fosse	estou	houveram	foram	tivemos
do	ou	estão	dele	está	houvera	fora	tiveram
da	quando	você	tu	estamos	houvéramos	fôramos	tivera
em	muito	tinha	te	estão	haja	seja	tivéramos
um	há	foram	vocês	estive	hajamos	sejamos	tenha
para	nos	essa	vos	esteve	hajam	sejam	tenhamos
é	já	num	lhes	estivemos	houvesse	fosse	tenham
com	está	nem	meus	estiveram	houvéssemos	fôssemos	tivesse
não	eu	suas	minhas	estava	houvessem	fossem	tivéssemos
uma	também	meu	teu	estávamos	houver	for	tivessem
os	só	às	tua	estavam	houvermos	formos	tiver
no	pelo	minha	teus	estivera	houverem	forem	tivermos
se	pela	têm	tuas	estivéramos	houverei	serei	tiverem
na	até	numa	nosso	esteja	houverá	será	terei
por	isso	pelos	nossa	estejamos	houveremos	seremos	terá
mais	ela	elas	nossos	estejam	houverão	serão	teremos
as	entre	havia	nossas	estivesse	houveria	seria	terão
dos	era	seja	dela	estivéssemos	houveríamos	seríamos	teria
como	depois	qual	delas	estivessem	houveriam	seriam	teríamos
mas	sem	será	esta	estiver	sou	tenho	teriam
foi	mesmo	nós	estes	estivermos	somos	tem	tínhamos
ao	aos	tenho	estas	estiverem	são	temos	tinham
ele	ter	lhe	aquele	hei	era	têm	tive
das	seus	deles	aquela	há	éramos	tinha	teve

APÊNDICE B

GUIA DO USUÁRIO

Neste apêndice são descritas as instruções de uso e guia de uso para o usuário. Também são descritos como devem ser disponibilizados os textos para o treinamento da ferramenta e para a avaliação.

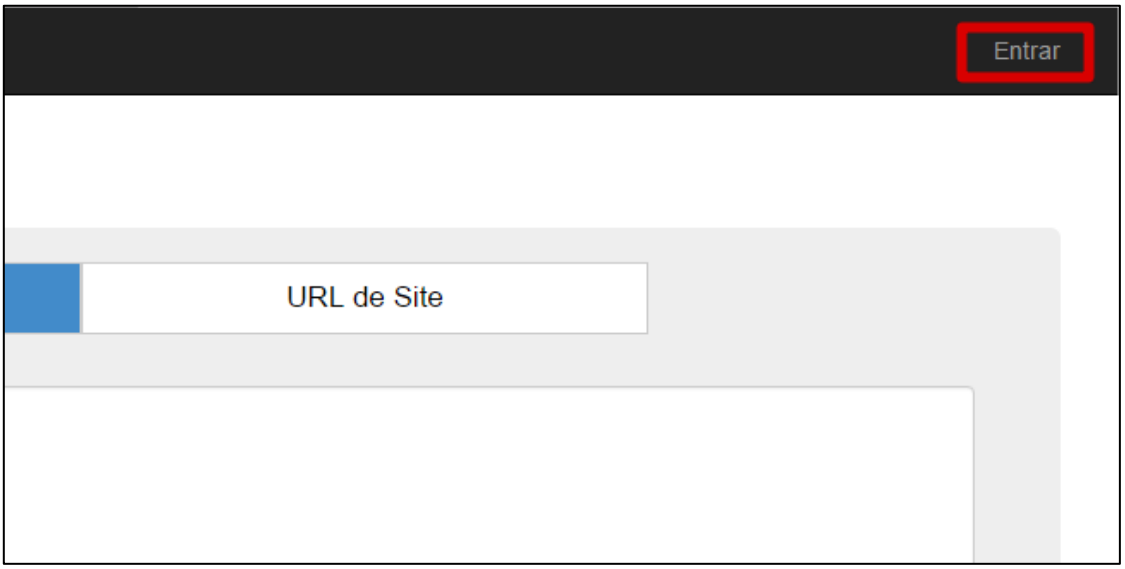
REALIZAR TREINAMENTO

A etapa de treinamento somente pode ser manuseada por usuários administradores da aplicação. A seguir, é descrito o passo-a-passo para realização do treinamento da aplicação.

Etapa 1 - Acesso ao sistema e alteração de senha

Através de qualquer página da aplicação, é possível realizar o acesso em um usuário. A aplicação possui apenas um usuário chamado “*admin*” que possui uma senha padrão “*admin*”, que foram configurados para o primeiro uso. Recomenda-se, que a senha seja alterada após o primeiro acesso ao sistema. Para acessar a tela de acesso ao sistema, é necessário clicar em “Entrar” no canto direito superior no menu, conforme exibido na Figura 36.

Figura 36 - Menu "Entrar" para realizar acesso ao sistema



Fonte: Próprio autor.

Na página de acesso ao sistema, deve-se informar o usuário e senha conhecidos, e confirmar o acesso, conforme Figura 37.

Figura 37 - Usuário e senha informados na página de acesso ao sistema

Fonte: Próprio autor.

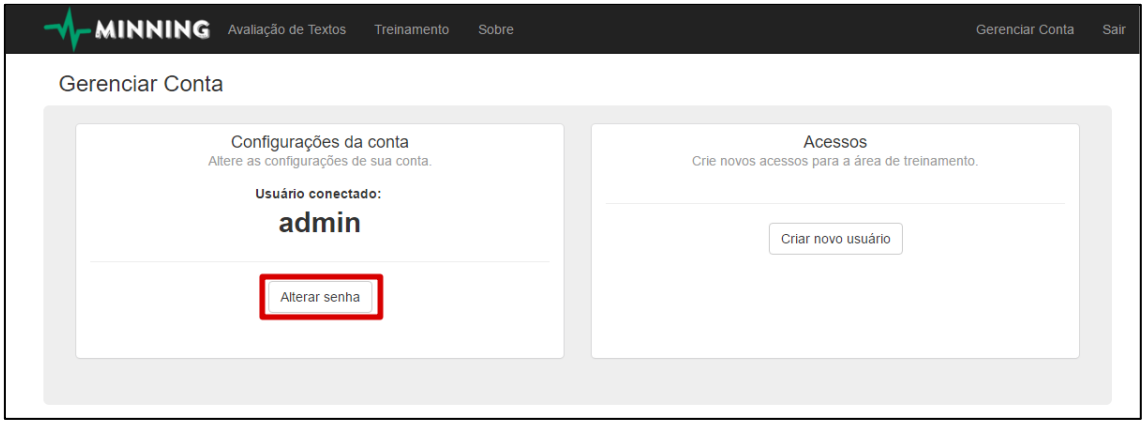
Uma vez que o acesso é realizado, no canto direito superior são exibidas as opções “Gerenciar Conta” e “Sair”. Para realizar a troca da senha, é necessário clicar em “Gerenciar Conta”, e para finalizar o acesso ao sistema é necessário clicar em “Sair”. As opções são demonstradas em Figura 38.

Figura 38 - Opção para gerenciar conta e finalizar acesso ao sistema

Fonte: Próprio autor.

Em “Gerenciar Conta”, é possível realizar a alteração da senha, conforme exibido em Figura 39. Também é possível, realizar a criação de outros usuários para acesso.

Figura 39 - Alteração de senha ao gerenciar conta



Fonte: Próprio autor.

Etapa 2 - Adicionando arquivos de treinamento

Através de qualquer página, é possível clicar em “Treinamento” no menu superior do site. Nessa página, é possível incluir um único arquivo de treinamento de forma manual, através do botão “Novo Arquivo”. Também é possível realizar o *upload* de vários arquivos, através do botão “Carregar arquivos”.

Etapa 3 - Realizando o treinamento

O treinamento é realizado através do clique no botão “Realizar Treinamento” exibido em azul no canto superior direito da tabela de arquivos. Ao final do treinamento, a página de treinamento concluído é exibida com algumas informações sobre o treinamento, conforme a Figura 40Figura 40.

Figura 40 - Treinamento concluído

Fonte: Próprio autor.

AVALIAR UM NOVO TEXTO

A avaliação de um novo texto apresenta os resultados acerca de sua apreensibilidade e qualidade. Enquanto que os resultados acerca da qualidade do texto possuem como pré-requisito a realização da etapa de treinamento da aplicação, os resultados acerca da apreensibilidade independem da mesma.

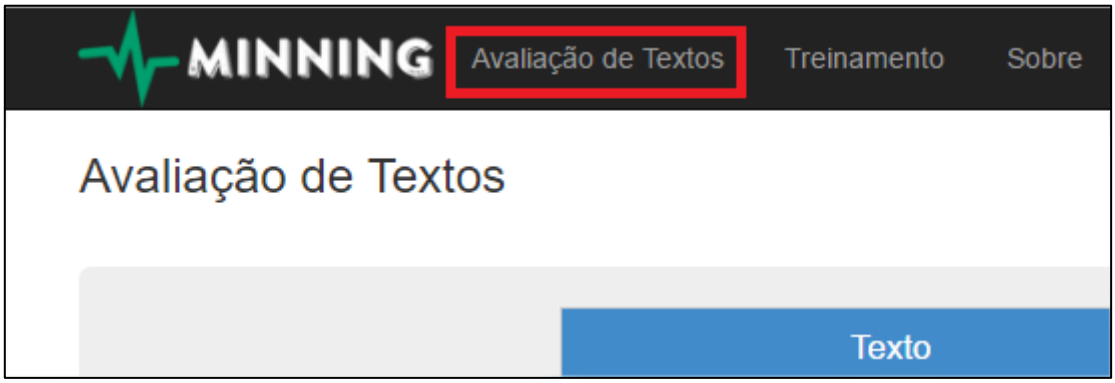
Essa funcionalidade não necessita que o usuário se identifique através do seu *login*, sendo disponível para qualquer pessoa com acesso à internet.

A seguir, é descrito o passo-a-passo para avaliar uma nova amostra textual através da aplicação.

Etapa 1 - Inserindo um texto ou *URL*

A página inicial da aplicação é responsável pela avaliação de textos, e também pode ser acessada através da opção “Avaliação de Textos” no menu superior da aplicação, conforme Figura 41.

Figura 41 - Opção para acessar a avaliação de textos no menu

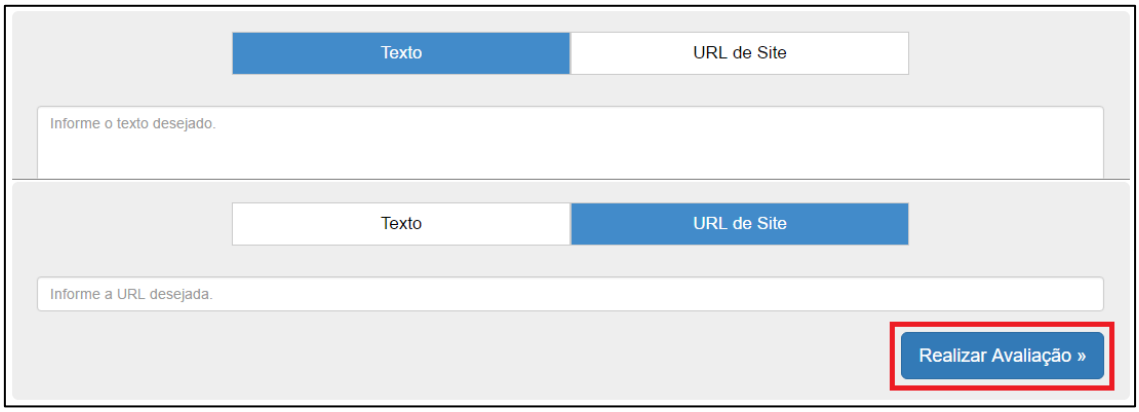


Fonte: Próprio autor.

Na página de avaliação de textos, são exibidas duas opções para o usuário, através das abas “Texto” e “*URL* de site”. A primeira das opções permite que um texto seja informado manualmente, já a segunda permite que a *URL* de um site seja informada.

Após informar o texto manual ou a *URL* de um site, para que o texto seja submetido à avaliação, o botão “Realizar Avaliação” deve ser clicado, conforme destacado na Figura 42.

Figura 42 - Opções para submeter um texto ou *URL* de um site para serem avaliados



Fonte: Próprio autor.