INSTITUTO ENSINAR BRASIL
FACULDADES DOCTUM DE CARATINGA

ITHALO HENRIQUE DE SOUSA LEAL

O USO DE APRENDIZAGEM DE MÁQUINA PARA IDENTIFICAÇÃO E CLASSIFICAÇÃO DE *FAKE NEWS* NO TWITTER REFERENTES A ELEIÇÃO PRESIDENCIAL DE 2018

CARATINGA
2018

INSTITUTO ENSINAR BRASIL
FACULDADES DOCTUM DE CARATINGA

ITHALO HENRIQUE DE SOUSA LEAL

O USO DE APRENDIZAGEM DE MÁQUINA PARA IDENTIFICAÇÃO E CLASSIFICAÇÃO DE *FAKE NEWS* NO TWITTER REFERENTES A ELEIÇÃO PRESIDENCIAL DE 2018

Trabalho de Conclusão de Curso apresentado ao Curso de Ciência da Computação das Faculdades Doctum de Caratinga, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação.

Área de Concentração: Mineração de dados, textos e aprendizagem de máquina.

Orientador (a): MSc. Glauber Luiz da Silva Costa

CARATINGA
2018

rede de ensino DOCTUM	FACULDADES DOCTUM DE CARATINGA	FORMULÁRIO 9
	TRABALHO DE CONCLUSÃO DE CURSO	
	TERMO DE APROVAÇÃO	

TERMO DE APROVAÇÃO

O Trabalho de Conclusão de Curso intitulado: O USO DE APRENDIZAGEM DE MÁQUINA PARA IDENTIFICAÇÃO E CLASSIFICAÇÃO DE FAKE NEWS NO TWITTER REFERENTES A ELEIÇÃO PRESIDENCIAL DE 2018, elaborado pelo(s) aluno(s) ITHALO HENRIQUE DE SOUSA LEAL foi aprovado por todos os membros da Banca Examinadora e aceito pelo curso de CIÊNCIA DA COMPUTAÇÃO das FACULDADES DOCTUM DE CARATINGA, como requisito parcial da obtenção do título de

BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Caratinga 12/12/2018

GLAUBER COSTA
Prof. Orientador

FABRICIA PIRES
Prof. Avaliador 1

HUDSON SILVA
Prof. Examinador 2

AGRADECIMENTOS

Agradeço primeiramente a Deus por cada conquista em minha vida, aos meus país e a meu irmão por sempre me apoiarem em todos os momentos.

Agradeço a todos os professores do curso de Ciência da Computação pelo conhecimento compartilhado em especial ao meu orientador Glauber Luiz da Silva Costa.

Agradeço também a todos os amigos que formei durante o decorrer do curso.

"Eu acredito que ás vezes são as pessoas que ninguém espera nada que fazem as coisas que ninguém consegue imaginar." (Alan Turing)

RESUMO

Tendo em vista a popularização das mídias sociais, o acesso à informação tornou-se rápido e prático, onde muitas vezes essas informações compartilhadas não condizem com a verdade. O objetivo desse trabalho foi o estudo e implementação de técnicas para classificação e identificação de *fake news* referentes à eleição presidencial de 2018. Para implementação foi necessário o estudo de técnicas de aprendizagem de máquina, mineração de dados e textos. Para aplicação das técnicas, foi efetuada a coleta de dados na rede social (Twitter), com objetivo de obter mensagens e notícias referentes aos candidatos a eleição presidencial. Logo após, foi realizado um pré-processamento na base de dados coletada, para redução de ruídos. O treinamento e teste dos algoritmos apresentaram ótimos resultados, tendo uma acurácia acima dos 90% em três dos quatro utilizados. Foi possível observar que os algoritmos apresentaram diferença estatística significativa. O projeto mostrou-se viável para criação de ferramentas para combate a *fake news* utilizando o modelo criado.

Palavras-chave: Mineração de Dados. Aprendizagem de Máquinas. Mineração de Textos.

ABSTRACT

With the popularization of social media, access to information has become fast and practical, where often this shared information does not match the truth. The objective of this work was the study and implementation of techniques for classification and identification of fake news regarding the presidential election of 2018. For implementation it was necessary the study of machine learning techniques, data mining and texts. For the application of the techniques, data was collected on the social network (Twitter), with the objective of obtaining messages and news regarding candidates for presidential elections. Subsequently, a pre-processing was performed in the collected database, for noise reduction. The training and test of the algorithms presented excellent results, having an accuracy above 90% in three of the four used. It was possible to observe that the algorithms presented significant statistical difference. The project proved feasible for creating tools to combat fake news using the created model.

***Keywords**: Data Mining. Machine Learning. Text Mining.*

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
GPL	*General Public License*
KDD	*Knowledge Discovery in Databases*
MIT	Massachusetts Institute of Technology
WEKA	*Waikato Environment for Knowledge Analysis*
GUI	*Graphical User Interface*
ARFF	*Attribute-Relation File Format*
EUA	Estados Unidos da América
RNA	Rede Neural Artificial
GHz	*Gigahertz*
GB	*Gigabyte*
IDE	*Integreted Develpment Environment*
DDR	*Double Data Rate*
HD	*Hard Disk*
BD	Banco de Dados
kNN	*K-Nearest Neighbors*
IBGE	Instituto Brasileiro de Geografia e Estatística
SBT	Sistema Brasileiro de Televisão
IBK	*Instance-based nearest neighbor*
CD	*Critical Distance*
STF	Supremo Tribunal Federal
TSE	Tribunal Superior Eleitoral

SUMÁRIO

INTRODUÇÃO

Atualmente com o grande volume de dados, que são gerados todos os dias ao redor do mundo, surgiu a necessidade de exploração desses dados, com isso o mercado na área de mineração de dados vem crescendo a cada ano. Grandes quantidades de informações que antigamente eram armazenadas de formas manuais em papéis, começaram a ser informatizada.

A sociedade tem procurado maneiras práticas e rápidas para se comunicar e realizar suas tarefas cotidianas, com isso o número de internautas a cada ano tem crescido. Um levantamento realizado pelo IBGE (2016), aponta que no Brasil existem mais 100 (cem) milhões de internautas. Ainda segundo o estudo, 94,2% das pessoas, utilizaram a *internet* para troca de mensagens.

As mídias sociais permitem aos usuários se interagirem, contudo, muitas informações que são disseminadas nas mídias sociais não são verdadeiras. Uma pesquisa realizada pela Reuters Institute (2018) mostra que 66% dos entrevistados no Brasil usam as mídias sociais como fonte de notícias.

O objetivo desse trabalho foi o desenvolvimento de um sistema protótipo capaz de classificar e Identificar *fake news* referentes à eleição presidencial no Brasil do ano de 2018, para atingir o objetivo foi necessário o estudo de técnicas de aprendizagem de máquina, mineração de dados e texto. A coleta de dados foi realizada através do Twitter, uma das dez maiores redes sociais do mundo (WE ARE SOCIAL LTD, 2018).

O tema foi escolhido, devido à eleição presidencial que ocorreu neste ano de 2018, e as *fake news* foram um dos assuntos mais tratados durante toda a eleição, sendo uma das principais preocupações, haja vista que a eleição é um processo democrático fundamental para uma sociedade. Segundo o ministro do STF Luiz Fux, poderiam ser anuladas as candidaturas que foram baseadas em *fake news* (TSE, 2018).

Este estudo pode influenciar e servir como um modelo para outros profissionais da área para desenvolvimento de ferramentas que ajudem ao combate de notícias falsas ou qualquer outro tema que envolva aprendizagem de máquina.

A presente monografia foi dividida em quatro capítulos essenciais para melhor entendimento, sendo eles:

O primeiro capítulo denominado Referencial Teórico, contém todo o embasamento teórico necessário para o desenvolvimento desse trabalho.

O segundo capítulo denominado Metodologia, contém todas as etapas realizadas para alcançar os objetivos traçados.

O terceiro capítulo denominado Análise dos Resultados, traz os resultados obtidos com os algoritmos implementados para classificação.

Por fim o quarto capítulo denominado Conclusão, é apresentado às considerações finais do trabalho.

1 **REFERENCIAL TEÓRICO**

Este capítulo apresenta um embasamento teórico para entendimento e compreensão desse trabalho, onde serão apresentados os conceitos e técnicas utilizadas para o desenvolvimento como *fake news*, processos do KDD, tipos de aprendizagem de máquina, biblioteca WEKA e a rede social Twitter.

1.1 *FAKE NEWS*

O termo *fake news* traduzido para o português significa notícias falsas. Esse termo tem sido muito falado atualmente, inclusive a imprensa tem feito várias matérias para relatar sobre o tema.

O termo ganhou destaque na imprensa internacional no ano de 2016, quando ocorreu a eleição presidencial dos EUA, sendo eleito Donald Trump. Empresas de tecnologia fizeram uma análise e descobriram vários conteúdos na internet que continham *fake news*, sendo a maioria para denegrir Hillary Clinton um dos candidatos a presidente daquele ano (BATISTA, 2018).

Esse fato chamou tanto atenção que o Departamento de Justiça Americano acusou formalmente treze cidadãos e três entidades russas, afirmando que elas teriam espalhado *fake news*, influenciando no resultado das eleições (BATISTA, 2018).

> De acordo com as alegações na acusação, doze dos réus individuais trabalharam em vários momentos para a Internet Research Agency LLC, uma empresa russa sediada em São Petersburgo, na Rússia. O outro réu individual, Yevgeniy Viktorovich Prigozhin, financiou a conspiração por meio de empresas conhecidas como Concord Management and Consulting LLC, Concord Catering e muitas subsidiárias e afiliadas (U.S. Department of Justice, 2018, com adaptações).

A criação de *fake news* tem por finalidade denegrir uma pessoa ou entidade. Geralmente são criadas matérias absurdas para atrair acesso. Segundo um estudo realizado Aral, Roy e Vosoughi (2018) notícias falsas se espalham de forma muito mais rápida no Twitter, sendo 70% mais propensas a serem compartilhadas do que as notícias verdadeiras.

No ano de 2014 uma *fake news* trouxe um fim trágico, uma mulher foi linchada até a morte no Brasil, após boatos que circularam na internet acusando-a de sequestrar crianças para rituais de magia negra (D'AGOSTINO, 2017).

No Brasil grandes portais de notícias, criaram setores para análise de veracidade das informações durante o processo eleitoral, com objetivo de reduzir o número de boatos que tem se espalhado na internet. Segundo o jornal Valor Econômico (2018) do Grupo O Globo, foram verificadas 959 notícias durante o período eleitoral e destas 200 eram falsas.

> No fim de semana do 2º turno, a equipe do Fato ou *Fake* se uniu a outras cinco agências de checagens de notícias no Brasil para checar as mensagens de conteúdo suspeito circulando pela web. O objetivo: juntar forças para ganhar mais agilidade e aumentar o alcance das checagens. A parceria reuniu o Fato ou *Fake*, Projeto Comprova, Lupa, Aos Fatos, Boatos.org e E-Farsas. O resultado: com o compartilhamento de conteúdo, apenas o Fato ou *Fake* publicou 18 boatos nos 2 dias (Valor Econômico, 2018).

O Projeto Comprova (2018), reuniu vários jornalistas de diferentes veículos de comunicação durante o período eleitoral, como SBT, BandNews FM, Gazeta Online, Folha de São Paulo e vários outros, segundo o projeto foram 12 semanas verificando a veracidade das informações compartilhadas por fontes não oficiais nas mídias sociais.

> No nosso radar estavam conteúdos duvidosos cuja capacidade de causar danos às campanhas eleitorais era evidente. Uma parte das verificações foi arquivada por elas não serem conclusivas. Nosso compromisso e nossa meta era o erro zero. Mesmo com todos esses recortes, publicamos 146 histórias para desmentir ou confirmar informações que viralizaram. Desse total, 92% se mostraram falsas, enganosas ou descontextualizadas. Apenas 9 histórias eram verdadeiras (Projeto Comprova, 2018).

Ainda segundo o Projeto Comprova (2018), o bom sinal foram o número de pessoas que entraram em contato com o projeto enviando sugestões de conteúdo a serem analisados, sendo que somente via WhatsApp foram mais de 67 mil.

1.2 KDD

Atualmente com a popularização da internet e das mídias sociais, o volume de dados vem crescendo de forma acentuada: "A cada dia, 2.5 exabytes de dados são

produzidos." (AGEAC, 2017). O relatório divulgado pela We Are Social LTD (2018), aponta que cerca de 3,4 bilhões de usuários utilizaram as mídias sociais em setembro de 2018, um aumento de 10% comparado com setembro de 2017.

O volume de dados que são gerados todos os dias, geralmente são apenas dados e não conhecimento, diante desse problema surge o processo chamado Descoberta de Conhecimento em Bancos de Dados (*Knowledge Discovery in Databases - KDD*).

> KDD é um processo, de várias etapas, não trivial, interativo e iterativo, para identificação de padrões compreensíveis, válidos, novos e potencialmente úteis a partir de grandes conjuntos de dados (FAYYAD et al, 1996, com adaptações).

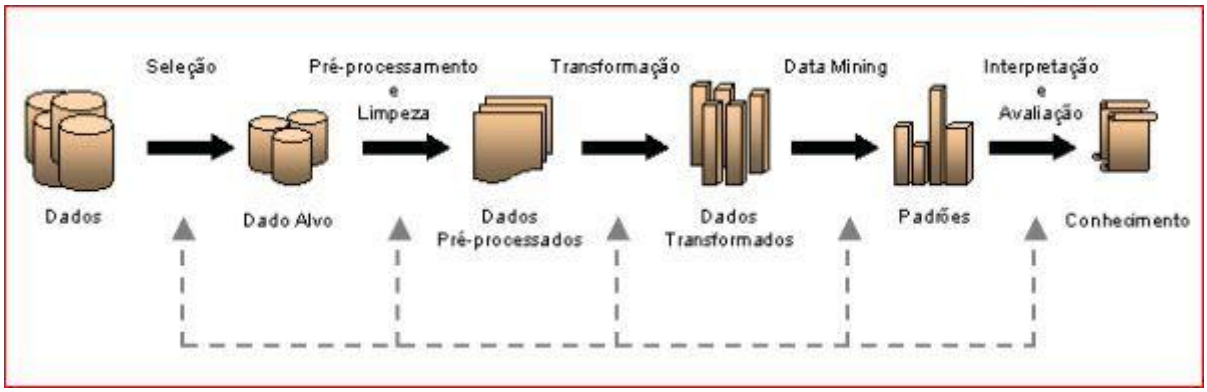
O KDD tem como objetivo a descoberta de conhecimento válido e relevante de forma automatizada em uma base de dados. A Figura 1 a seguir, apresenta as principais diferenças entre os dados, informação e conhecimento:

Figura 1 - Pirâmide do Conhecimento

Fonte: Martins (2018)

Segundo Prass (2012), o KDD é um processo que possui duas características relevantes: é interativo e iterativo. Interativo, pois o usuário pode intervir e controlar o curso das atividades. Iterativo, por ser uma sequência finita de operações sendo que o resultado de cada sequência é dependente dos resultados das que a precedem.

Figura 2 - Ciclo do processo de KDD



Fonte: Prass (2012, p. 2)

A Figura 2, apresenta o ciclo do processo do KDD, onde inicialmente é submetido um conjunto de dados, logo após é realizado o pré-processamento, sendo eliminados os registros irrelevantes, posteriormente os dados são armazenados e formatados adequadamente para que os algoritmos de aprendizado de máquina sejam aplicados na etapa seguinte. Após todas essas etapas o conhecimento obtido deve ser interpretado e avaliado, para verificar se o objetivo traçado foi alcançado (PRASS, 2012).

1.2.1 **Mineração de Dados**

A mineração de dados, ou no inglês *data mining*, tem sido utilizada em diversas áreas, como nas áreas educacionais, financeira e medicina, por sua capacidade em extrair informação que antes de sua aplicação não era possível. Segundo Júnior (2018), a mineração de dados é fase que geralmente possui mais ênfase na literatura, nesta fase é realizada a identificação de padrões em grandes bases de dados.

> A mineração de dados está relacionada à criação de modelos, um modelo é simplesmente um algoritmo ou conjunto de regras que conecta uma coleção de entradas para uma meta ou resultado específico (BERRY; LINOFF, 2004, p. 8).

De acordo com Camilo e Silva (2009), as tarefas mais comuns dentro da mineração de dados são:

- Classificação
- Estimação

- Predição
- Agrupamento
- Associação

As tarefas que foram supracitadas, se dividem em dois métodos de aprendizado, sendo eles o supervisionado e não supervisionado, os dois métodos são abordados na seção de aprendizagem de máquina desse trabalho.

São consideradas como aprendizagem supervisionada as tarefas de agrupamento e associação. Já as tarefas de classificação, estimação e predição estão associadas ao método de aprendizagem supervisionado (CAMILO; SILVA, 2009).

1.2.1.1 Classificação

Classificação é uma tarefa, em que as características de um determinado objeto são analisadas e atribuídos a um conjunto predefinido de classes. Os objetos a serem classificados são representados por registros em uma tabela de banco de dados ou em um arquivo, e o ato de classificação consiste em rotular os registros em uma determinada classe. (BERRY; LINOFF, 2004, p. 9)

Alguns exemplos de classificação são:

- Classificação de risco de crédito;
- Reconhecimento facial;
- Reconhecimento de fraudes em sistemas;
- Detecção de sintomas de doenças;

De acordo com Soares (2005), o processo de classificação é utilizado para encontrar relacionamento entre os atributos preditivos e objetivo, tendo assim um conhecimento que possa ser utilizado para antever a classe de um registro que ainda não foi classificado.

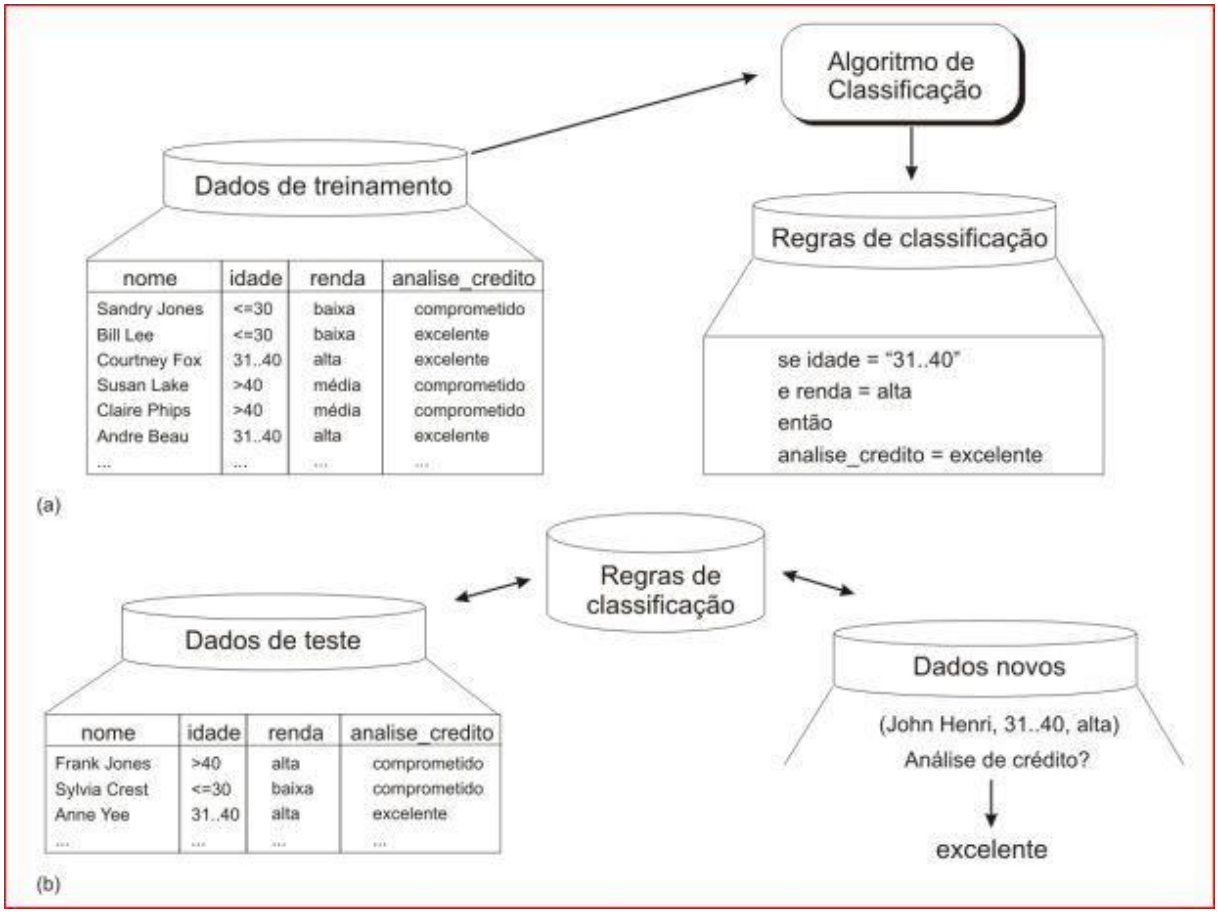
Segundo Petermann (2006), o processo de classificação possui duas etapas, a primeira etapa é o aprendizado ou treinamento, que é executada com uma base de

dados já classificada, já na segunda etapa é utilizada a generalização, no qual se utilizam registros que não foram utilizados no treinamento.

Basicamente o que é feito é o mapeamento das entradas e saídas para posteriormente utilizá-las na tomada de decisão em novos registros, ainda não classificados.

Ainda segundo Petermann (2006), o primeiro passo de classificação é fundamental pois através dele é feita a construção de um modelo de dados pré-definidos, através da análise de tuplas de um banco de dados, no qual cada uma dessas tuplas é pertencente de uma classe..

Figura 3 - Processo de classificação



Fonte: Petermann (2006, p. 76)

A Figura 3 apresenta as duas etapas do processo de classificação, a primeira etapa é submetida uma base de treinamento a um algoritmo de classificação e esse algoritmo gera um modelo. Na segunda etapa é submetido a base de teste ou novos registros não classificados pelo algoritmo.

1.2.1.2 Estimação

A estimativa trabalha com dados numéricos. Dado alguns valores de entrada, a estimativa atribui um valor para um dado desconhecido, baseado em atributos variáveis como renda, altura ou saldo do cartão de crédito. Na prática, a estimação é frequentemente usada para executar uma tarefa de classificação (BERRY; LINOFF, 2004, p. 9).

1.2.1.3 Predição

A predição é semelhante às tarefas de classificação ou estimativa, sendo que a principal diferença é que os registros são classificados visando descobrir um valor futuro para um atributo (CAMILO; SILVA, 2009).

Segundo Baker, Isotani e Carvalho (2011), a predição requer uma certa quantidade de dados de boa qualidade, classificados manualmente para uma previsão correta. Alguns exemplos de predição são:

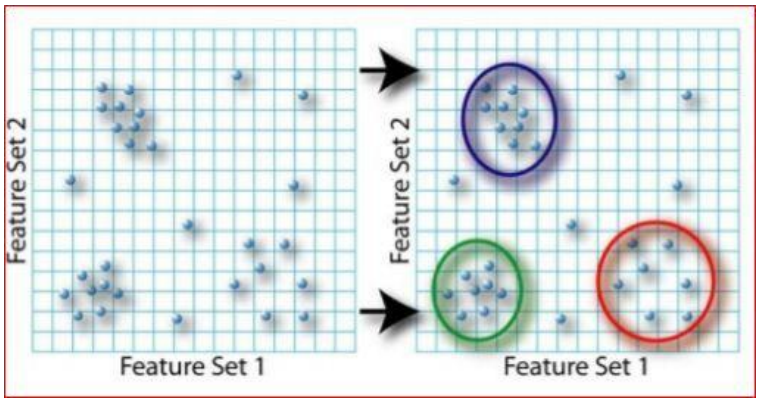
- Previsão de evasão escolar;
- Previsão da bolsa de valores;
- Previsão do campeão de um campeonato;

1.2.1.4 Agrupamento

O principal objetivo do agrupamento é associar dois ou mais objetos em determinados grupos. Estes grupos inicialmente não são conhecidos, sendo estes identificados automaticamente através da manipulação das características dos dados (BAKER; ISOTANI; CARVALHO, 2011).

O agrupamento de afinidades pode ser usado para identificar oportunidades de vendas cruzadas e projetar pacotes atraentes ou agrupamentos de produtos e serviços (BERRY; LINOFF, 2004, p. 11).

Figura 4 - Exemplo de agrupamento



Fonte: Hugo (2017)

A Figura 4, apresenta um simples exemplo de agrupamento, no qual os pontos mais próximos foram agrupados com o propósito de extrair uma informação.

1.2.1.5 Associação

Segundo Brusso, Navaux e Geyer (2000), através da mineração de dados é possível extrair padrões utilizando regras de associação. A principal utilização de associação tem sido em comércios para análise de itens adquiridos em uma mesma compra. Basicamente associação representam a probabilidade de que um item exista em um conjunto, ou transação, dado que outro também esteja presente.

Figura 5 - Exemplo de associação

```
Regra 1: SE idade = jovem AND estudante = não ENTÂO compra = não
Regra 2: SE idade = jovem AND estudante = sim ENTÂO compra = sim
Regra 3: SE idade = média ENTÂO compra computadores = sim
Regra 4: SE idade = adulta AND avaliação de crédito = excelente ENTÂO compra computadores = sim
Regra 4: SE idade = adulta AND avaliação de crédito = ruim ENTÂO compra computadores = não
```

Fonte: O autor (2018)

A Figura 5 apresenta um exemplo de associação, no qual são analisados os atributos que estão associados em uma determinada situação.

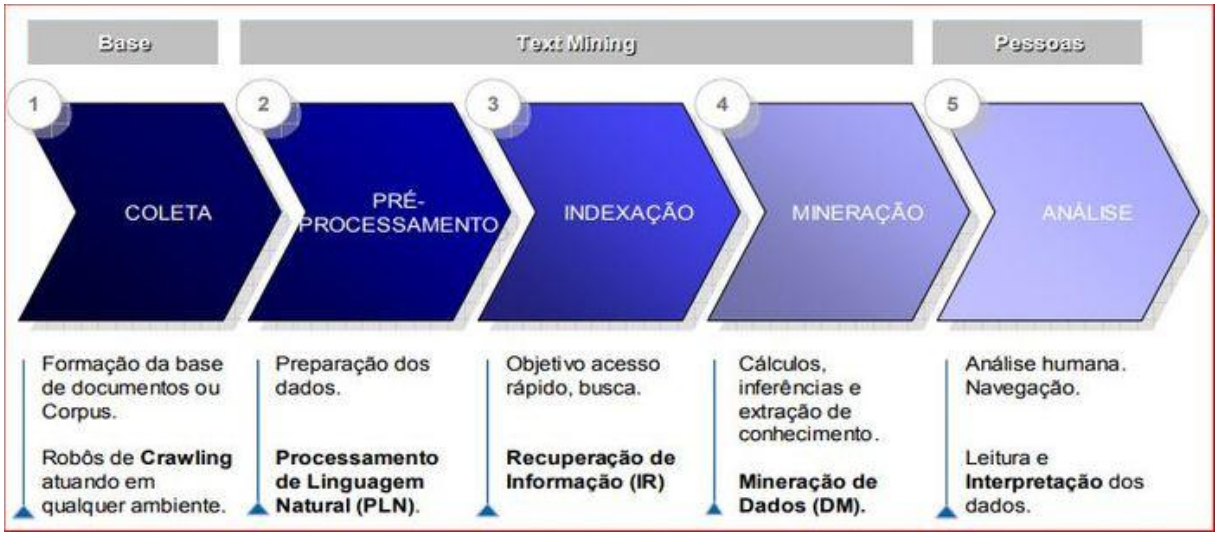
1.2.2 **Mineração de textos**

A mineração de texto pode ser considerada, uma busca de informações relevantes a partir de um grande volume de textos, escrito na linguagem natural. Tanto na mineração de dados quanto na mineração de textos são aplicados algoritmos de aprendizagem de máquinas, visando o conhecimento (VIANA, 2014).

Segundo Brito (2016), a principal diferença entre mineração de dados para a mineração de textos, é que em mineração de dados a informação a ser extraída não está implícita, mas estão estruturadas dentro de uma base de dados. Já na mineração de textos, a informação é explicita no texto, mas não é possível o processamento automático da informação, haja vista que a informação não é estruturada.

A mineração de textos pode ser dividida em diversas etapas, mas neste trabalho utiliza-se a mesma abordagem que Aranha (2007) utilizou em sua tese. O autor seguiu as seguintes etapas em seu projeto: Coleta de Dados, Pré-Processamento, Indexação, Mineração de Dados e Análise dos Resultados.

Figura 6 - Etapas Mineração de Textos



Fonte: Aranha (2007, p. 19)

A Figura 6 apresenta as etapas da mineração de textos utilizada nesse trabalho, a seguir é descrito cada umas dessas etapas:

- Coleta de dados: É realizada a extração de textos a ser trabalhada, os textos podem serem extraídos de livros, redes sociais e diversos outros meios, a coleta de dados pode ser uma etapa muito trabalhosa e custosa (ARANHA, 2007).

- Pré-processamento: Tem como objetivo transformar os dados não estruturados para representação atributo-valor, melhorando a qualidade dos dados e organizando os mesmos (ARANHA, 2007).
- Indexação: Os dados são catalogados para seus devidos grupos, facilitando a busca e agilizando o acesso a informação (ARANHA, 2007).
- Mineração: Ocorre a mineração de dados em si, utilizando técnicas de cálculos e inferências, afim de extrair a informação dos dados coletados (ARANHA, 2007).
- Análise: É realizado a interferência humana para analisar os resultados de acordo com entendimento humano (ARANHA, 2007).

Um simples exemplo de utilização da mineração de textos é a filtragem de *spam*, utilizada pelos provedores de e-mail, onde é realizado uma análise no corpo do e-mail de modo a obter os padrões de uma mensagem afim de classifica-la como *spam* ou não *spam*.

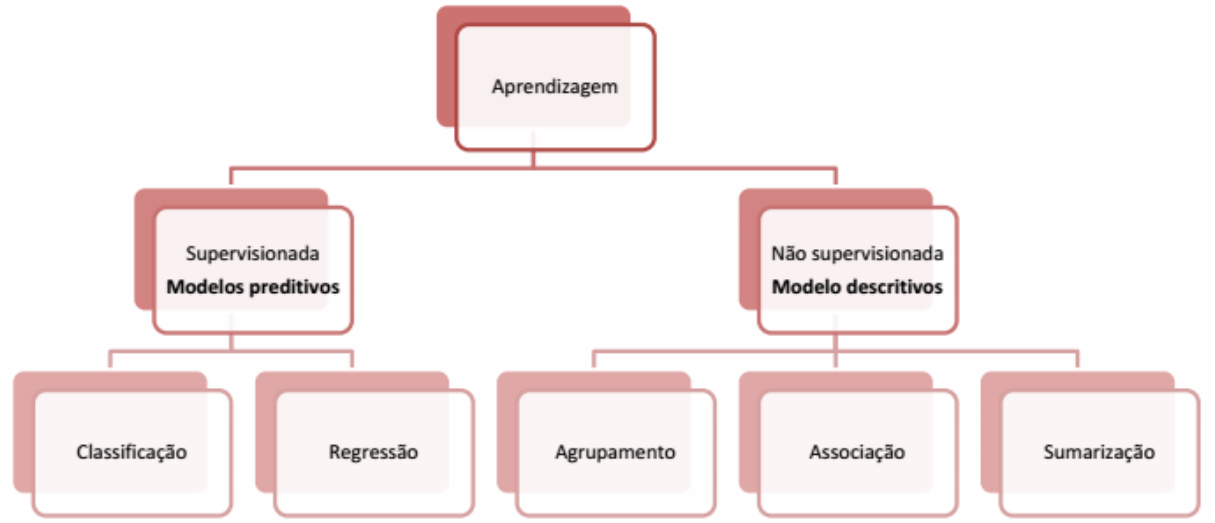
1.3 APRENDIZAGEM DE MÁQUINAS

De acordo com Goldschmidt (2010) o aprendizado de máquina é uma área da inteligência artificial voltada para o desenvolvimento de algoritmos e técnicas computacionais que possibilitam que os computadores sejam capazes de aprenderem.

Ainda segundo Goldschmidt (2010) o aprendizado só é possível através dos sistemas de aprendizado. Programas de computador são capazes de tomar decisões baseados em um banco de dados com dados reais que já ocorreram, ou seja, são tomadas decisões através de padrões.

Segundo Pereira (2013), padrões são conjuntos de características que determinam um objeto, sendo que essas características são determinadas buscando a separação das classes a fim simplificar as tarefas de um classificador. Existem duas categorias da aprendizagem de máquinas sendo elas a: supervisionada e não supervisionada.

Figura 7 - Os tipos de aprendizado de máquina



Fonte: Vilar (2017)

Algoritmos supervisionados, são quando o ser humano controla a entrada e saída e avalia a precisão das previsões durante o treinamento. Após o treinamento o algoritmo aplica o que aprendeu para novos registros desconhecidos. Já os algoritmos não supervisionados, não é necessário serem treinados com dados de resultados desejados (SCHMITT, 2013).

É utilizada uma técnica iterativa chamada aprendizagem profunda (*deep learning*). Esses algoritmos são utilizados para realizar tarefas mais complexas (SCHMITT, 2013). Nas seções a seguir são abordados os conceitos sobre os algoritmos utilizados nesse projeto.

1.3.1 **Teorema de Bayes e Classificador *Naive Bayes***

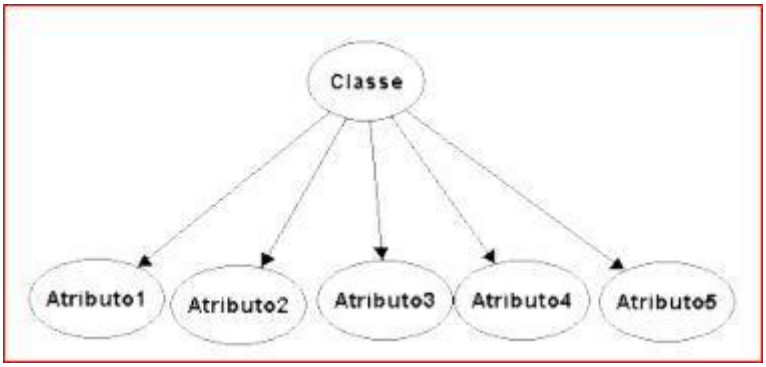
Segundo Ara-Souza (2010), as técnicas Bayesianas além de serem muito utilizadas em estatísticas são também utilizadas em outras áreas, sendo uma delas a Inteligência Artificial também conhecida como Inteligência Artificial Probabilística. O modelo probabilístico trabalha com conhecimento prévio de uma base histórica. Ao submeter um novo registro, ele utiliza essa base histórica para cálculo da probabilidade.

Segundo Brito (2016), o algoritmo Naive Bayes é um gerador probabilístico, sendo muito utilizado em aprendizagem de máquina, devido a sua simples

compreensão além de possuir uma fácil implementação, quando traduzido o termo *Naive* significa ingênuo, fazendo referência ao fato de o modelo assumir que existe uma independência condicional dos atributos.

O classificador Naive Bayes foi desenvolvido utilizando o Teorema de *Bayes*, tendo como criador desse teorema Thomas Bayes no século XVIII. O classificador tem como princípio que seus atributos são independentes, dado a variável classe, a Figura 8 a seguir mostra a estrutura de uma classe com 5 atributos.

Figura 8 - Estrutura de uma classe



Fonte: Brito (2016)

O Teorema de *Bayes* trata sobre probabilidade condicional, ou seja, verifica a probabilidade de um evento acontecer, dado outro evento. Para melhor entendimento, vejamos um exemplo: Dado uma base de dados com históricos financeiros de vários clientes, qual a probabilidade de um novo cliente seja um bom ou mau pagador.

De acordo com Alteryx Inc (2018) o classificador *Naive Bayes* funciona bem, mesmo com pequenos conjuntos de dados para treinamento, tendo essa vantagem, devido fato que o classificador é parametrizado pela média e variância de cada atributo independente de outros.

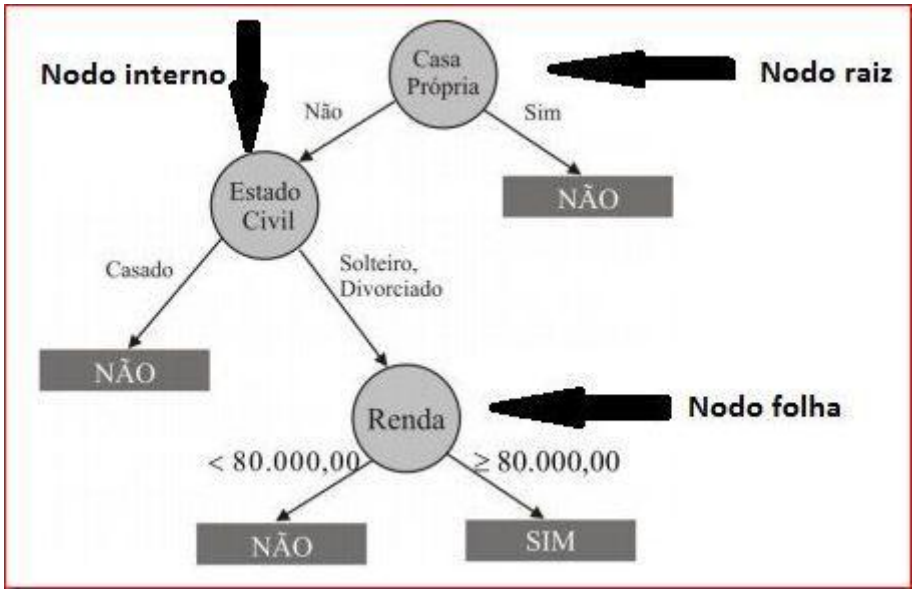
1.3.2 **Aprendizagem por Árvores de Decisão**

Segundo Crepaldi et al. (2010), as árvores de decisão são simples representação do conhecimento, sendo uma ótima opção para construção de um classificador, existindo inúmeros modelos de árvores de decisões, podendo ser aplicado em vários casos.

Segundo Librelotto e Mozzaquatro (2013), o princípio da árvore de decisão é dividir para conquistar, no qual um problema complexo é divido em subproblemas tornando mais simples para resolução. Os dados são separados utilizando as características em comum, até chegar a um ponto que seja indivisível para representação da classe.

A árvore de decisão parte de um nodo raiz que é aquele que não possui nenhuma aresta de entrada, podendo haver zero ou mais arestas de saídas, já os nodos internos possuem obrigatoriamente uma aresta de entrada e duas ou mais de saída e por fim os nodos folhas da árvore são os pontos indivisíveis, sendo eles a representação da classe (HOSOKAWA, 2011).

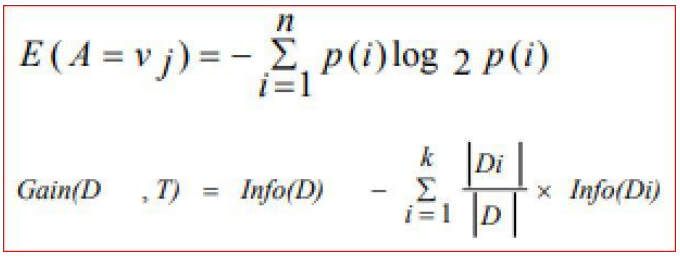
Figura 9 - Exemplo de uma árvore de decisão para classificação de clientes devedores



Fonte: Hosokawa (2011)

Basicamente para construção de uma árvore de decisão é feito o cálculo de entropia e de ganho, sendo que primeiramente é escolhido o atributo de nodo raiz, e logo após é realizado o cálculo de forma recursiva até chegar ao nodo folha, a cada nodo é escolhido o atributo que possui maior relevância, podendo um atributo ser descartado caso o seu ganho de informação seja irrelevante.

Figura 10 - Fórmula entropia e ganho



$$E(A = v_j) = -\sum_{i=1}^{n} p(i) \log_2 p(i)$$

$$Gain(D, T) = Info(D) - \sum_{i=1}^{k} \frac{|Di|}{|D|} \times Info(Di)$$

Fonte: Padilha, Almeida e Alves (2003)

Segundo Takatuzi (2016), a maioria dos algoritmos de aprendizagem de máquina, não possui a capacidade de incluir novos dados ao conjunto que já foi treinado durante toda sua execução, vários algoritmos buscam métodos eficientes para o ganho da informação, sem que haja o crescimento da árvore, visto que alguns casos uma árvore pode crescer de forma acentuada não trazendo nenhuma melhora na capacidade de predição e classificação.

1.3.3 **Aprendizagem por Regras**

Métodos de aprendizagem baseados em regras são classificadores pertencentes a sistemas de inteligência artificial, porém o uso deste método ainda é modesto (TAKATUZI; STANGE, 2018).

> Uma visão geral dos métodos baseados em regras para representação do conhecimento e aprendizagem se concentra em classes de regras do tipo "Se..então". O conjunto de regras deste tipo é uma das formas mais expressivas e legíveis para representar hipóteses em aprendizagem de máquina. (TAKATUZI; STANGE, 2018).

De acordo com Prati (2006), os métodos mais utilizados em aprendizagem baseados em regras, são: ordem zero, baseada em atributos e baseada em lógica de primeira ordem. Um algoritmo para indução de regras de classificação tem um conjunto de atributos de entrada, sendo que um atributo pode assumir um conjunto finito de valores.

Lógica de ordem zero, o item pode ser descrito por conjunções, disjunções e negações de constantes booleanas que representam atributos individuais, tendo um baixo poder descritivo. A lógica de atributos é equivalente a ordem zero, mas emprega

uma notação mais flexível e poderosa, pois os atributos são tratados como variáveis podendo assumir diversos valores (MONARD; BARANAUSKAS, 2003).

Já a lógica de primeira ordem possui o maior poder entre elas, permitindo descrever e raciocinar sobre objetos e predicados que especificam propriedades de objetos ou relacionamentos entre objetos do domínio D (MONARD; BARANAUSKAS, 2003).

Figura 11 - Exemplo de aprendizagem por regras

```
if paciente se sente bem = sim then
  classe = saudável
end if
if paciente se sente bem = não and paciente tem dor = não
and temperatura do paciente ≤ 37 then
  classe = saudável
end if
if paciente se sente bem = não and paciente tem dor = não
and temperatura do paciente > 37 then
  classe = doente
end if
if paciente se sente bem = não and paciente tem dor = sim then
  classe = doente
end if
```

Fonte: Monard e Baranauskas (2003)

A Figura 11 apresenta um exemplo de aprendizagem por regras onde em cada "*IF*" é verificado se a condição é verdadeira de modo a classificar o registro em uma determinada classe.

1.3.4 **Aprendizagem por Redes Neurais Artificiais**

Uma Rede Neural Artificial (RNA) possui várias camadas para o processamento da informação, em cada camada possui vários nós, sendo que cada nó, simula um "neurônio", todas as camadas são interligadas, cada "neurônio" possui um peso de acordo com a informação armazenada, o peso de um "neurônio" é recalculado a cada interação (FALQUETO, 2007).

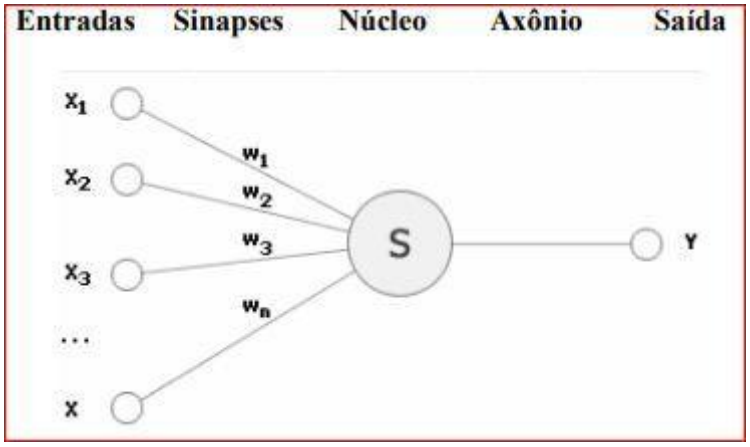
A RNA é um sistema físico que podem adquirir, armazenar e utilizar conhecimentos experimentais, podendo alcançar uma performance satisfatória,

devido ao grande número de conexões entre os "neurônios" da rede (FALQUETO, 2007).

De acordo com Gonçalves e Tostes (2004), o primeiro modelo de RNA, foi proposto em 1943 por McCulloch e Pitts, no qual a ideia era simular a realidade biológica. McCulloch considerou o funcionamento de um neurônio sendo um circuito binário, onde as entradas dos neurônios são binárias e são combinadas por uma soma ponderada, utilizando a seguinte fórmula: S = ∑ wi .xi

A Figura 12 a seguir, é apresentando o funcionamento de um neurônio artificial, sendo que o mesmo funciona de forma semelhante ao natural, cada sinapses liga as entradas do neurônio ao núcleo; o núcleo do neurônio processa os sinais de entrada e possui axônios que o conecta com os neurônios da próxima camada. Cada sinapse tem um peso, determinando o peso daquela informação na rede (GONÇALVES; TOSTES, 2004).

Figura 12 - Neurônio



Fonte: Gonçalves e Tostes (2004)

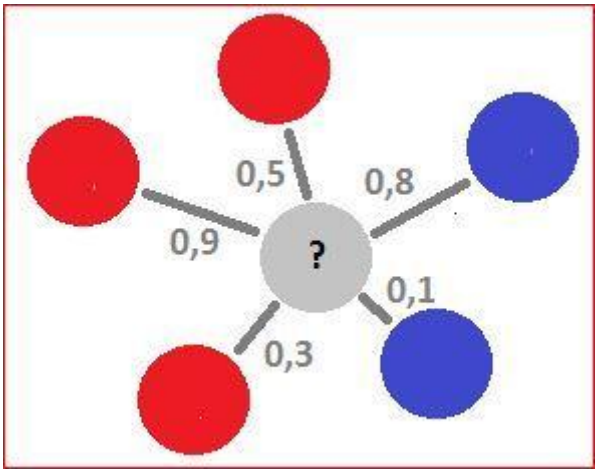
Segundo Aranha (2007), as Redes Neurais Artificiais começaram a ganhar bastante destaque na década de 90, segundo o autor Redes Neurais Artificiais é um modelo bastante complexo, e seu uso é mais restrito para problemas mais complexos, onde possui grande volume de dados.

1.3.5 **Aprendizagem Baseada em Instâncias**

Aprendizagem baseado em instâncias ou kNN, é um algoritmo do tipo *lazy* (preguiçoso). A principal ideia do algoritmo é procurar os registros classificados mais próximos e com base nessa informação ele classifica o novo registro. O número de vizinhos mais próximos a serem comparados é definido no valor "K" (FERRERO, 2009).

Os algoritmos do tipo kNN necessitam de pouco processamento na etapa de treinamento, porém, o processamento necessário para classificar um novo registro é alto, onde no pior dos casos será necessário o novo registro, ser comparado com todos registros do conjunto (FERRERO, 2009).

Figura 13 - Exemplo kNN



Fonte: O autor (2018)

A Figura 13 mostra um simples exemplo do funcionamento dos algoritmos kNN, para poder definir a cor que a bolinha cinza obterá, é preciso analisar os vizinhos mais próximos, se o valor de K = 1, a mesma será azul, se o valor de K = 2, há um empate, podendo a cor ser vermelha ou azul, se o valor de K = 3 a bolinha será vermelha, pois haverá duas bolinhas vermelhas e uma azul e assim sucessivamente.

Figura 14 - Fórmula da distância euclidiana.

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

Fonte: Bricce et al. (2016)

Segundo Bricce et al. (2016) pode-se utilizar diversas fórmulas para cálculo de distância, mas o cálculo distância euclidiana é o mais simples principalmente em casos bidimensionais.

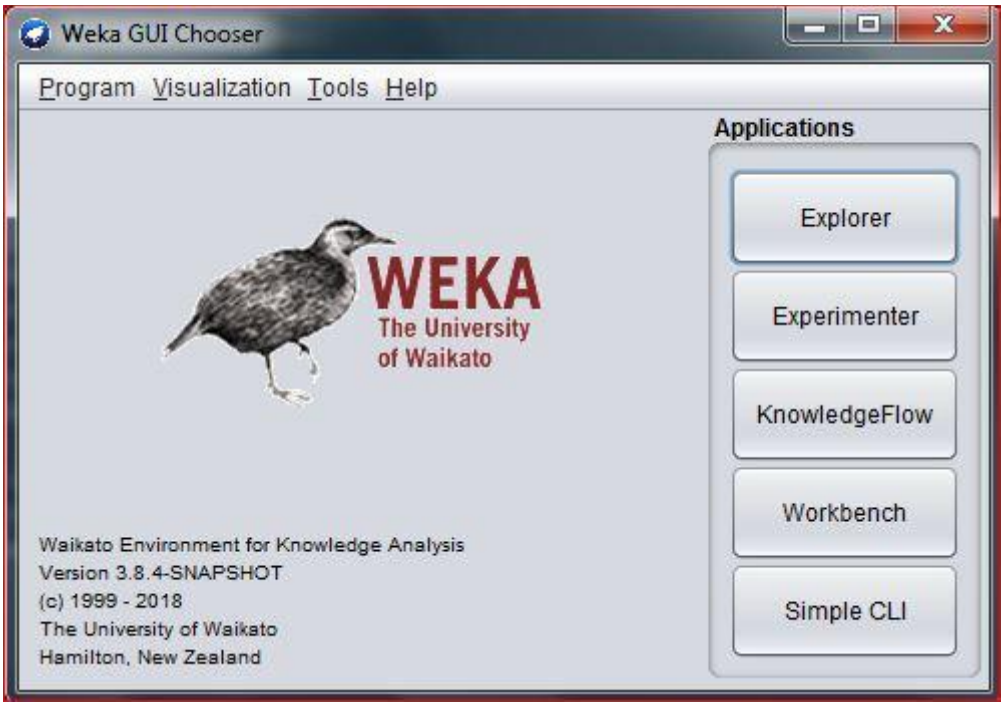
- D = Distância euclidiana;
- X e Y = Valores de entrada a serem calculados;
- ∑ = Somatória.

1.4 WEKA

WEKA é uma coleção de algoritmos de aprendizado de máquina para tarefa de mineração de dados, contendo inúmeras ferramentas para preparação, classificação, regressão e agrupamento de dados, a ferramenta começou a ser escrita em 1993 na Universidade de Waikato, Nova Zelândia.

Seu nome foi inspirado de aves encontradas apenas nas ilhas da Nova Zelândia. WEKA é um *software* de código aberto emitido sob a Licença Pública Geral GNU (HALL et al., 2009).

Figura 15 - Interface WEKA



Fonte: O autor (2018)

O software foi escrito na linguagem Java, além de possuir uma API para integração, foi desenvolvido um pacote GUI para interação, podendo ser manipulado de forma visual conforme pode-se observar na Figura 15. O WEKA trabalha com uma extensão própria (ARFF) para manipulação dos dados, através desse arquivo é possível definir os tipos de dados que serão carregados (ABERNETHY, 2010).

1.5 TWITTER

França e Oliveira (2014) define o Twitter como um microblog onde permitem pessoas e instituições a divulgar qualquer tipo de informação ou opinião de forma quase instantânea, para todos que estão conectados a sua rede. Cada publicação é limitada um número de caracteres, obrigando os usuários a expressar sua opinião ou dar uma informação de forma rápida e objetiva.

O Twitter foi criado em Março de 2006, e seus fundadores foram Jack Dorsey, Evan Williams, Biz Stone e Noah Glass, a rede social foi lançada de forma oficial nos EUA em Julho de 2006, sua principal ideia era uma espécie de "SMS da internet", o Twitter ganhou extensa notabilidade e popularidade por todo mundo, contando atualmente com 284 milhões de usuários registrados (Wikipédia, 2018).

No próximo capítulo é abordado os métodos empregados para a realização desse trabalho, sendo o mesmo dividido nas seguintes seções: objetivo do estudo, ambiente de trabalho, coleta e preparação dos dados e implementação dos algoritmos.

.

2 **METODOLOGIA**

Para realização do trabalho foi utilizado a metodologia de pesquisa exploratória, visando o desenvolvimento de um sistema capaz de realizar a classificação de mensagens envolvendo os candidatos a eleição presidencial. A pesquisa exploratória é uma ótima alternativa quando deseja identificar ou compreender uma determinada situação, podendo utilizar abordagens qualitativas e quantitativas (BRITO, 2016).

Foram utilizadas técnicas de aprendizagem de máquina supervisionada, mineração de textos e dados. Os algoritmos utilizados para classificação: Naive Bayes, Redes Neurais Artificiais, Árvores de Decisão, Aprendizagem por Regras e Aprendizagem Baseada em Instâncias, foram escolhidos com base em pesquisas em literaturas de diversos autores, como por exemplo: Brito (2016) que utilizou em sua dissertação o algoritmo Naive Bayes para detecção automática de sentimentos em comentários nas mídias sociais.

A biblioteca WEKA foi utilizada no projeto haja vista que a mesma possui uma coleção de algoritmos, incluindo os citados anteriormente, prontos para serem implementados, além de ser uma das mais utilizadas no mundo e possuir uma fácil integração com a linguagem de programação Java, que foi utilizada para desenvolvimento do projeto.

Para a coleta das mensagens do Twitter, foi utilizada a biblioteca Twitter4J que conecta a API oficial disponibilizada pelo Twitter. Todas as mensagens capturadas foram armazenadas em um banco de dados próprio, devido que a conta registrado no Twitter pelo Autor deste trabalho é *Standard* tendo atualmente uma limitação de captura dos últimos 7 dias além de um limite de 180 requisições a cada 15 minutos.

Algumas etapas descritas abaixo, foram indispensáveis para o desenvolvimento do presente trabalho em busca de atingir o objetivo final:

- Preparação do ambiente para desenvolvimento do sistema.
- Modelagem de um banco de dados para armazenamento dos dados extraídos através da API do Twitter.
- Coleta de dados
- Escolha dos algoritmos de classificação que foram utilizados.

- Implementação dos algoritmos de aprendizagem de máquina, definidos na etapa anterior.
- Avaliação e Validação dos resultados

2.1 OBJETIVO DE ESTUDO

Este trabalho utilizou a metodologia triangular, onde consiste na combinação de duas ou mais abordagens metodológicas, o primeiro método utilizado foi o qualitativo no que diz respeito a classificação das mensagens, o segundo método utilizado foi o quantitativo, para análise dos resultados, sendo os resultados representados em gráficos e tabelas.

Esta pesquisa se enquadra na natureza aplicada, onde o principal intuito foi criação de um sistema protótipo de classificação e identificação de *fake news*, podendo ser utilizado para outros temas no futuro.

O tema foi escolhido considerando que as eleições ocorreram neste ano de 2018, e as *fake news* sobre os candidatos eram motivos de preocupação para todos os envolvidos. Apesar das eleições que ocorreram neste ano de 2018 eram para: Presidente, Senador, Deputado estadual, Deputado federal e Governador, por se tratar de um processo custoso, o tema foi delimitado somente aos candidatos a eleição presidencial.

2.2 AMBIENTE DE TRABALHO

Para realização desse trabalho foram utilizados como ferramentas para suporte, além do embasamento teórico, um PC com processador Intel Core i3-2100 3.10 GHz, HD de 500 GB, memória de 4 GB DDR2 e sistema operacional Windows 7 *Ultimate*.

Para implementação do projeto foi utilizado IDE *Intellij* versão 8.0.1 e a linguagem de programação Java versão 8, além das bibliotecas Twitter4j versão 4.0.8 para consumo da API do Twitter e a biblioteca WEKA versão 3.8.4, sendo que a

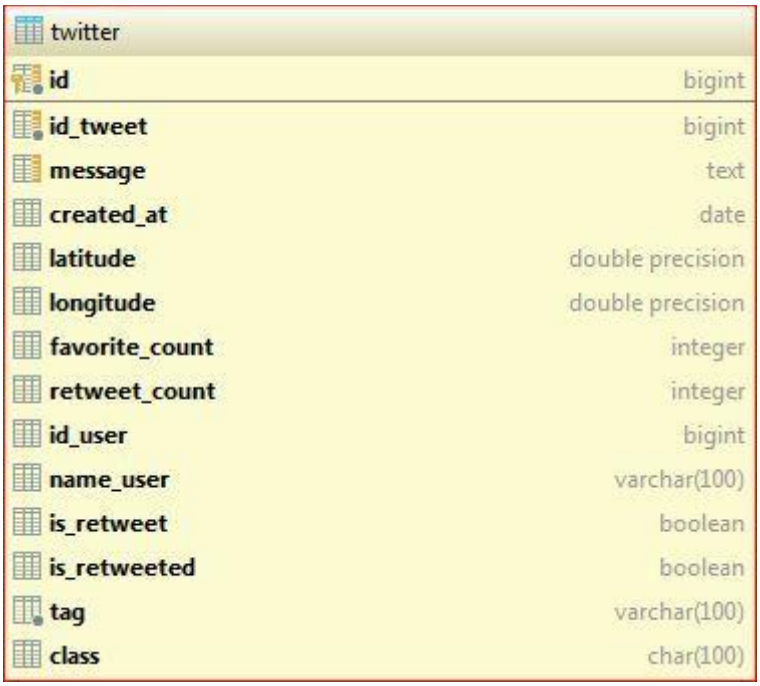
mesma possui inúmeros algoritmos para classificação. O banco de dados escolhido para armazenamento dos *tweets* foi o PostgreSQL versão 9.3.

2.3 BASE DE DADOS PARA COLETA DOS TWEETS

Para realização da coleta, foi necessária a modelagem de uma base de dados para receber os dados coletados. Essa base de dados própria foi necessária devido às limitações impostas pelo Twitter citadas anteriormente. A estrutura da tabela, foi criada de acordo com o retorno da API de um *tweet,* além de alguns atributos necessários para controle na coleta e mineração de dados. Foram desconsiderados alguns campos da API que não teriam utilidades para o objetivo proposto neste trabalho.

O Twitter fornece duas categorias de API's para coleta de dados, sendo elas a de pesquisa histórica e a de *streaming* em tempo real. Neste trabalho, foi utilizado a API de pesquisa histórica, considerando que seria capturado *tweets* recentes e não recentes.

Figura 16 - Estrutura da Tabela



Fonte: O autor (2018)

A Figura 16 apresenta a estrutura da tabela TWITTER. Os atributos da tabela são retratados a seguir no Quadro 1.

Quadro 1 – Colunas da tabela TWITTER

Atributo	**Descrição**
id_tweet	Código sequencial único para cada *tweet* enviado
message	Mensagem publicada
created_at	Data e hora que o *tweet* foi enviado
latitude	Latitude do local que o usuário do Twitter estava na hora de enviar o *tweet*
longitude	Longitude do local que o usuário do Twitter estava na hora de enviar o *tweet*
favorite_count	Quantidade de vezes que outros usuários marcaram que gostou do *tweet*
retweet_count	Número de vezes que foi retweetado
Id_user	Código sequencial único para cada usuário cadastrado
name_user	O nome do usuário, não sendo necessariamente o nome da pessoa
is_retweet	Campo booleano se foi um *retweet*
is_retweeted	Campo booleano se foi retweetado
tag	Campo para armazenar a chave utilizada para achar o *tweet*
class	Classe do tweet (fato, *fake*, opinião)

Fonte: O autor (2018)

No Quadro 1 acima é descrito cada atributo da tabela TWITTER, a qual irá armazenar todos os dados capturados.

2.4 COLETA E ARMAZENAMENTO DE DADOS

A etapa de coleta de dados é fundamental para um sistema de aprendizagem de máquinas, quanto mais rica a informação coletada, maior hipótese de atingir o objetivo proposto.

O primeiro passo para coleta de dados, foi a definição das palavras-chave a serem buscadas, sendo que as mesmas, foram definidas de acordo com os nomes, sobrenomes e números dos partidos dos candidatos cadastrados no site do TSE[1].

Com base nessas informações foram definidas as seguintes palavras-chave: "jair messias bolsonaro", "fernando haddad", "vera lucia pereira da silva salgado", "alvaro dias", "cabo daciolo", "ciro gomes", "jose maria eymael", "geraldo alckmin", "guilherme boulos", "henrique meirelles", "joão amoêdo", "joão goulart filho", "marina silva", "bolsonaro", "haddad", "daciolo", "ciro", "alckmin", "boulos", "meirelles", "amoêdo", "joão goulart", "marina", "bolsonaro17", "haddad13", "ciro12", "daciolo51", "alvaro19", "eymael27", "alckmin45", "boulos50", "meirelles15", "amoedo30", "joaogoulart54", "marina18", "vera16".

Figura 17 - *ArrayList* com palavras-chave

```
ArrayList<String> tags = new ArrayList();
Collections.addAll(tags, "jair messias bolsonaro", "fernando haddad",
        "vera lucia pereira da silva salgado", "alvaro dias", "cabo daciolo",
        "ciro gomes", "jose maria eymael", "geraldo alckmin", "guilherme boulos",
        "henrique meirelles", "joão amoêdo", "joão goulart filho", "marina silva",
        "bolsonaro", "haddad", "daciolo", "ciro", "alckmin", "boulos", "meirelles",
        "amoêdo", "joão goulart", "marina", "bolsonaro17", "haddad13", "ciro12",
        "daciolo51", "alvaro19", "eymael27", "alckmin45", "boulos50", "meirelles15",
        "amoedo30", "joaogoulart54", "marina18", "vera16");
```

Fonte: O autor (2018)

Após definidas as palavras-chave, iniciou-se o processo de conexão com API do Twitter, para conexão com a mesma foi utilizado a biblioteca Twitter4J, a biblioteca possui fácil integração com sistemas desenvolvidos em Java para utilização dos serviços do Twitter (Twitter4J, 2018).

[1]http://divulgacandcontas.tse.jus.br/divulga/#/estados/2018 /2022802018/BR/candidatos

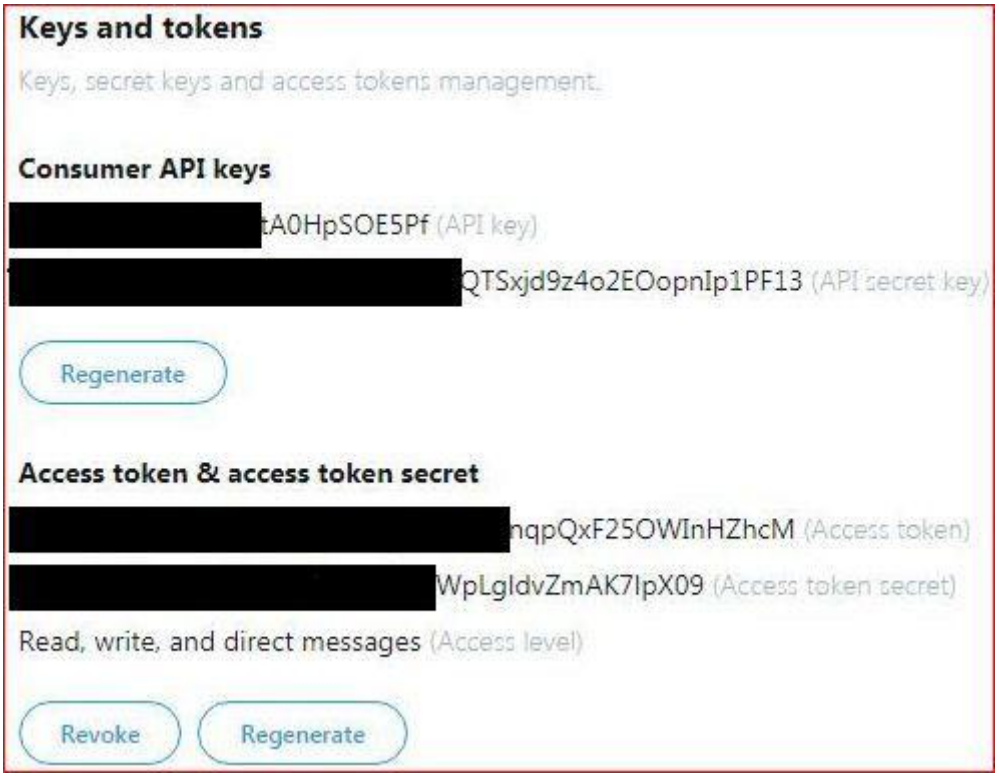
Figura 18 - Conexão com API do Twitter utilizando Twitter4J

```
ConfigurationBuilder cb = new ConfigurationBuilder();
cb.setDebugEnabled(true)
        .setOAuthConsumerKey("[redacted]A0HpSOE5Pf")
        .setOAuthConsumerSecret("[redacted]z4o1EOopnIp0PF13")
        .setOAuthAccessToken("[redacted]nqpQxF15OWInHZhcM")
        .setOAuthAccessTokenSecret("[redacted]b5WpLgldvZmAK72pX09")
        .setTweetModeExtended(true);
TwitterFactory tf = new TwitterFactory(cb.build());
Twitter twitter = tf.getInstance();
```

Fonte: O autor (2018)

Por motivos de segurança, o Twitter exige uma chave de acesso para que possa ser feito a conexão com sua API, essa chave foi obtida através de seu site na área do desenvolvedor [2], conforme é apresentado na Figura 19 a seguir.

Figura 19 - *Tokens* conexão com API



Fonte: O autor (2018)

A API do Twitter oferece vários parâmetros que podem ser utilizados para busca além da palavra-chave, neste trabalho foram utilizados os seguintes filtros:

[2]Twitter *Developer* - Disponível em <https://developer.twitter.com>

Quadro 2 - Parâmetros de busca

Filtro	Descrição
since	Retorna *tweets* criados a partir da data especificada. A data deve ser formatada como AAAA-MM-DD.
until	Retorna *tweets* criados antes da data especificada. A data deve ser formatada como AAAA-MM-DD.
since_id	Retorna resultados com um ID maior que o especificado.
query	Palavra-chave a ser buscada
count	O número de tweets a serem retornados por página, até o máximo de 100
locale	Idioma a ser buscado

Fonte: Adaptado de Twitter (2018)

A partir dos parâmetros e palavras-chave definidos iniciou-se a coleta de dados, sendo que a mesma foi realizada no período do dia 26/09/2018 a 27/10/2018, ao todo foram extraídos 1.780.821 (um milhão e setecentos e oitenta mil e oitocentos e vinte e um) *tweets*.

Tabela 1 - Quantidade de registros dia a dia

Data	**Quantidade**
26/09/2018	659
27/09/2018	3.098
28/09/2018	8.226
29/09/2018	87.268
30/09/2018	204.804
01/10/2018	435.564
02/10/2018	27.269
03/10/2018	147.486
04/10/2018	128.830
05/10/2018	334.494
06/10/2018	7.806
07/10/2018	249.726
08/10/2018	21.955
09/10/2018	23.734

(continuação)

Data	**Quantidade**
11/10/2018	79.410
18/10/2018	11.574
22/10/2018	431
27/10/2018	8487
Total	**1.780.821**

Fonte: O autor (2018)

Desse total foi verificado que 498.346 (quatrocentos e noventa e oito mil e trezentos e quarenta e seis) registros, não são oriundos de compartilhamento (*retweets*). O código-fonte completo utilizado nesse projeto pode ser encontrado no GitHub[3].

2.5 GERAÇÃO DO ARQUIVO .ARFF

O arquivo .ARFF é um tipo texto ASCII que descreve a lista de instâncias que compartilham um conjunto de atributos. O arquivo está dividido em duas seções diferentes, sendo a primeira seção um cabeçalho que contém o nome da relação, a lista de atributos e os tipos dos atributos. A segunda seção é basicamente os dados em si (University of Waikato, 2018).

[3] https://github.com/ithaloleal/ProjetoFakeNews/

Figura 20 - Exemplo arquivo .ARFF

```
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature numeric
@attribute humidity numeric
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes
sunny,72,95,FALSE,no
sunny,69,70,FALSE,yes
rainy,75,80,FALSE,yes
sunny,75,70,TRUE,yes
overcast,72,90,TRUE,yes
overcast,81,75,FALSE,yes
rainy,71,91,TRUE,no
```

Fonte: University of Waikato (2018)

A Figura 20 mostra um simples exemplo da estrutura do arquivo. Os quatro primeiros atributos (*outlook*, *temperature*, *humidity*, *windy*) são os atributos previsores e o último é o atributo classe, sendo que o mesmo é classificado com base nos outros quatro atributos.

Figura 21 - Código-fonte para geração do arquivo .ARFF

```
public class GeraArquivoArff {
    public static File geraArquivo() {
        File arquivo = new File(System.getProperty("user.dir") + "\\TccIthaloLeal\\src\\fakenewsTCC.arff");
        try {
            Connection con = ColetaAPI.conexaoBD();

            PreparedStatement cmd1 = con.prepareStatement("select message, class from twitter where class is not null");
            ResultSet result = cmd1.executeQuery();

            // Cabeçalho do arquivo Weka
            String exportacao = "@relation fakenewsTCC\n\n";
            exportacao += "@attribute message string\n";
            exportacao += "@attribute class {opiniao, fato, fake}\n\n";
            exportacao += "@data\n";

            while (result.next()) {
                exportacao += "'" + Normalizer.normalize(result.getString("message").replaceAll("\n", " ").
                        replaceAll("'", ""), Normalizer.Form.NFD).replaceAll("[^\\p{ASCII}]", "") + "'," +
                        result.getString("class") + "\n";
            }

            FileOutputStream f = new FileOutputStream(arquivo);
            f.write(exportacao.getBytes());
            f.close();

        } catch (SQLException | IOException e) {
            e.printStackTrace();
        }

        return arquivo;
    }
}
```

Fonte: O autor (2018)

A Figura 21 apresenta o código desenvolvido para geração do arquivo .ARFF, utilizando a base de dados coletada.

2.6 PRÉ-PROCESSAMENTO DOS DADOS

A etapa de pré-processamento dos dados, podem ser aplicadas várias técnicas para melhora da qualidade dos dados a serem minerados. O WEKA fornece vários filtros para pré-processamento, neste trabalho foram utilizados os filtros: *StringToWordVector* e *Reorder*.

2.6.1 ***StringToWordVector***

O principal objetivo do filtro *StringToWordVector* é a conversão dos atributos do tipo *string* para atributos numéricos que representam informações de ocorrência de palavras do texto (University of Waikato, 2018). O filtro fornece em suas propriedades outros algoritmos, para auxílio na conversão dos atributos, sendo eles: *stemmer* e *stopwords*.

- ***Stemmer*:** Os algoritmos de *stemming* basicamente efetuam a redução das palavras até o seu radical, eliminando a variação de uma palavra e reduzindo o tempo de processamento. Exemplo: canta**ram** → canta.
- ***Stopwords*:** Os *stopwords* são palavras, pronomes, artigos, pontuações que geralmente são muito usadas no texto, mas não trazem nenhum valor com sua análise. Sendo útil apenas para entendimento da sentença, a eliminação desses tokens em grandes quantidades de dados diminui o custo e melhora o desempenho. Exemplo: "O Twitter é uma rede social." → "[O] [Twitter] [é] [uma] [rede] [social][.]".

Para realização desse trabalho, foi utilizado uma lista de *stopwords* contendo artigos, preposições, conjunções e sinais de pontuação e esta pode ser encontrada

no ANEXO A. Segundo Wives e Loh (1998), esses termos por estarem na maioria das *strings* e se repetirem de forma contínua, possuem baixo grau de discriminação.

Outras propriedades que foram utilizadas neste filtro são: *lowerCaseTokens* que converte todos os *tokens* para minúsculos, pois, termos maiúsculos e minúsculos não influenciam no objetivo proposto desse trabalho, e a propriedade *attributeIndices* onde define quais índices serão convertidos em *tokens*.

2.6.2 ***Reorder***

Este filtro reordena os atributos, sendo muito útil quando se utiliza filtros como o *StringToWordVector*, pois, a ordem é alterada, e muitos classificadores necessitam que o atributo classe esteja na última posição, além disso, é possível remover ou duplicar um atributo.

2.7 IMPLEMENTAÇÃO DOS ALGORITMOS CLASSIFICADORES

A biblioteca WEKA fornece uma fácil integração de seus algoritmos com projetos desenvolvidos em Java, neste trabalho foram implementados os algoritmos:

- *NaiveBayes* (Naive Bayes)
- *MultilayerPerceptron* (Redes Neurais Artificiais)
- J48 (Árvores de Decisão)
- JRip (Aprendizagem por Regras)
- iBK (Aprendizagem Baseada em Instâncias)

Os algoritmos foram escolhidos com base na literatura, sendo esses os mais utilizados atualmente. A Figura 22 mostra a implementação do algoritmo *NaiveBayes*.

Figura 22 - Implementação do algoritmo *NaiveBayes*

```
File file = GeraArquivoArff.geraArquivo();
ConverterUtils.DataSource ds = new ConverterUtils.DataSource(new FileInputStream(file));
Instances instancias = ds.getDataSet();
// Seta o atributo classe
instancias.setClassIndex(instancias.numAttributes() - 1);

NaiveBayes nb = new NaiveBayes();
StringToWordVector stringToWordVector = new StringToWordVector();
stringToWordVector.setInputFormat(instancias); // seta a base de dados (arquivo .ARFF)
// Carrega a lista de stopwords
WordsFromFile wordsFromFile = new WordsFromFile();
wordsFromFile.setStopwords(new File(System.getProperty("user.dir") + "\\TccIthaloLeal\\src\\stopwords.txt"));
// Converte tudo para minúsculo
stringToWordVector.setLowerCaseTokens(true);
// Define o atributo a ser aplicado o filtro
stringToWordVector.setAttributeIndices("first");
// Inverte a ordem dos atributos
Reorder reorder = new Reorder();
reorder.setAttributeIndices("last-first");
reorder.setInputFormat(instancias);
// Aplica os filtros
MultiFilter mf = new MultiFilter();
Filter[] filters = new Filter[2];
filters[0] = stringToWordVector;
filters[1] = reorder;
// Seta os filtros
mf.setFilters(filters);
// Efetua a classificação
FilteredClassifier fc = new FilteredClassifier();
fc.setFilter(mf);
fc.setClassifier(nb);
fc.buildClassifier(instancias);
```

Fonte: O autor (2018)

2.8 TREINAMENTOS DOS ALGORITMOS

Para treinamento dos algoritmos, foram classificados 5.000 (cinco mil) registros de forma manual, a escolha desses registros foi baseada em número de vezes que foram compartilhados (retweetados), considerando o estudo feito por (ARAL et al, 2018), onde apresenta que as *fake news* são 70% mais propensos a serem compartilhadas.

Os *tweets* foram classificados nas seguintes classes: fato, *fake* ou opinião. Para classificação foi analisado o conteúdo do texto, para ajuda na análise foram utilizados alguns portais que analisam boatos da *internet*. Ex: projetocomprova.com.br e g1.globo.com/fato-ou-fake/.

O registro foi classificado como fato, quando o mesmo citava um candidato, e o fato realmente tinha acontecido, conforme pode ser visto na Figura 23, vários órgãos de imprensa noticiaram esse ato[4].

Quadro 3 - Exemplo notícia classificada como fato

Mensagem	Classe
Marina Silva participa de ato contra Bolsonaro https://bit.ly/2QiYcRo #Eleições2018	fato

Fonte: O autor (2018)

O registro foi classificado como *fake*, quando o mesmo citava um candidato, e o fato não tinha ocorrido ou o fato ocorrido foi diferente da mensagem, conforme pode ser visto na Figura 24. Apesar das manifestações realmente tivessem acontecido, o título real da notícia era: Manifestações contra Jair Bolsonaro ocorrem em 66 cidades pelo mundo[5].

Quadro 4 - Exemplo notícia classificada como *fake*

Mensagem	Classe
Manifestações contra @jairbolsonaro ocorrem em 66 países https://t.co/rDt6pjg7HP https://t.co/gWPoQxbzwJ	*fake*

Fonte: O autor (2018)

Por fim os registros classificados como opinião, foram registros que poderiam ou não, citar nome de um candidato, mas o usuário estava expressando sua própria opinião do assunto, um exemplo pode ser visto na Figura 25.

Quadro 5 - Exemplo notícia classificada como opinião

Mensagem	Classe
Ato da Virada! No Mercado de São Brás! Vem pra democracia! #Haddad13 https://t.co/TMKmXQQ6At	opinião

Fonte: O autor (2018)

Além dos registros classificados manualmente, foi utilizado um corpus público e gratuito de 3.600 (três mil e seiscentos) notícias falsas e 3.600 (três mil e seiscentos)

[4]https://politica.estadao.com.br/noticias/eleicoes,ato-contra-bolsonaro-une-adversarios-em-sao-paulo,70002525476

[5]https://oglobo.globo.com/brasil/manifestacoes-contra-jair-bolsonaro-ocorrem-em-66-cidades-pelo-mundo-23113418

verdadeiras, já classificadas, disponibilizado no GitHub[6] por (MONTEIRO et al., 2018).

Foi utilizado esse corpus, de modo a obter um número maior de registros para treinamento e teste, além de balançar o número de registros para cada classe. Após classificados os registros manualmente foram notados um desbalanceamento entre as classes, o que pode afetar o desempenho do algoritmo.

Segundo Marquezan (2009) a palavra corpus vem do latim e seu significado quer dizer corpo, esse termo foi consagrado no campo de Direito Romano para designar ideia de conjunto. No contexto acadêmico corpus é um conjunto de dados sobre um determinado tema.

O treinamento de um algoritmo basicamente é submetê-lo registros que já possuem uma classe definida, com esses registros são criados os modelos para classificação de novos registros ainda não classificados.

O processo mais custoso na aprendizagem de máquina é a geração e teste desse modelo. Cada algoritmo gera um modelo de acordo com sua finalidade, exceto os algoritmos de aprendizagem baseada em instâncias que apenas armazena os exemplos de treinamento.

- *NaiveBayes*: O modelo é uma tabela probabilística de cada registro.
- *MultilayerPerceptron*: É gerado um modelo contendo todos os pesos encontrados nos registros.
- J48: O modelo gerado é uma árvore de decisão com todos os registros.
- JRip: Gera um modelo com todas as regras encontradas.

O treinamento do algoritmo é realizado juntamente com o teste, através do teste é possível avaliar o desempenho dos algoritmos, existem várias técnicas para efetuar o teste, o mais utilizado atualmente pela comunidade científica é o *K-fold Cross Validation*, a análise de desempenho dos algoritmos é abordada na próxima seção de resultados.

Durante o processo de treinamento dos algoritmos notou-se uma grande lentidão no algoritmo *MultilayerPerceptron*, sendo que o algoritmo executou por mais de 12 horas e não obteve nenhum resultado durante esse período, optou-se então

[6]https://github.com/roneysco/Fake.br-Corpus

pelo descarte do mesmo, devido à limitação de *hardware* que o autor desse trabalho obtinha e ao tempo que seria necessário para obtenção dos resultados.

O próximo capitulo é apresentado os resultados obtidos através dos testes de validação cruzada, Friedman e Nemenyi.

3 ANÁLISE DOS RESULTADOS

Este capítulo tem por finalidade apresentar os resultados obtidos através dos algoritmos utilizados para classificação. Esta análise além de avaliar o desempenho dos algoritmos, avaliou se a existência de diferença estatística significativa entre os mesmos.

Inicialmente, foram gerados os modelos dos algoritmos e posteriormente foram submetidos todos os registros não classificados de forma manual, de modo a obter o número e percentual do total de registros classificados em cada classe.

Tabela 2 - Quantidade de registros classificados em cada classe

	NaiveBayes	**J48**	**JRip**	**IBK**
opinião	1.342.164	1.390.367	1.696.925	1.348.893
fato	245.825	214.722	48.698	255.551
fake	192.832	175.732	35.198	176.377

Fonte: O autor (2018)

A Tabela 2 apresenta a distribuição de registros classificados em cada classe, onde é possível observar que o número de registros classificados na classe *fake* através do algoritmo *NaiveBayes* é superior aos demais.

A seguir na Tabela 3 é apresentado o percentual de registros classificados em cada classe. Para cálculo do percentual, utilizou-se a seguinte fórmula: **P = (qc / qt) × 100**, onde:

- P → Percentual a ser obtido
- qc → Quantidade de registros da classe
- qt → Quantidade total de registros

Tabela 3 - Percentual de registros classificados em cada classe

	NaiveBayes	**J48**	**JRip**	**IBK**
opinião	75,36771%	78,07449%	95,28891%	75,74557%
fato	13,80403%	12,05747%	2,73458%	14,35018%
fake	10,82826%	9,86803%	1,97650%	9,90425%

Fonte: O autor (2018)

Os valores apresentados nas Tabelas 2 e 3, retratam apenas os números e percentuais de registros classificados em cada classe. Para avaliação de desempenho que o algoritmo obteve em classificar toda a base de dados, seria necessária a verificação manual de registro a registro, para conferência dos erros e acertos, o que em uma base de dados grande torna-se praticamente inviável.

Para avaliação de desempenho dos algoritmos, foi utilizado a técnica *K-fold Cross Validation*. Com objetivo de uma avaliação mais precisa foram realizados 10 (dez) testes, sempre variando parâmetro *Random seed*.

Figura 23 - Código para realização dos testes utilizando *K-fold Cross Validation*

```
int vezes = 5; // numero de vezes que vai ser realizado o teste
ArrayList<Thread> threads = new ArrayList<>();
for (int i = 1; i <= vezes; i++) {
    int finalI = i;
    threads.add(
            // Foi criada a Thread para poder executar mais de um teste ao mesmo tempo, reduzindo o tempo de espera
            new Thread(() -> {
                try {
                    System.out.println("====== Inicio - Seed: " + finalI + " - " + tmp.getClassifier().toString() + " ======");
                    Evaluation eval = new Evaluation(instances);
                    // A cada teste o Random seed soma + 1
                    eval.crossValidateModel(tmp, instances, 10, new Debug.Random(finalI));
                    System.out.println("====== " + tmp.getClassifier().toString() + " ======");
                    System.out.println("Seed " + finalI + ":" + String.valueOf(eval.pctCorrect()).replace('.', ','));
                    Optional<Thread> first = threads.stream().filter(e -> e.getState() == Thread.State.NEW).findFirst();
                    first.get().start();
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }));
}
// Inicia o teste, executa 3 testes por vez
threads.get(0).start();
threads.get(1).start();
threads.get(2).start();
```

Fonte: O autor (2018)

Random seed significa semente aleatória, sua função é determinar quais partes do conjunto de dados serão utilizadas para treinamento e teste (SILVA et al, 2018). Neste trabalho foi utilizado a variação do *Random seed* de 1 a 10, com a finalidade de determinar que o algoritmo utiliza-se partes diferentes do conjunto.

3.1 *K-FOLD CROSS VALIDATION*

K-fold Cross Validatation é basicamente um processo de reamostragem. Esse método é utilizado para avaliar um modelo com um número de amostras limitadas. Esse procedimento possui um parâmetro "K" o qual determina o número de grupos

para o, qual uma determinada amostra de dados deve ser dividida (BROWNLEE, 2018).

A Tabela 4, apresenta os resultados obtidos nos testes variando o *seed*. Ao todo foram utilizados 12.200 (doze mil e duzentos) registros para treinamento e teste, conforme já abordado na seção de Treinamento dos Algoritmos.

Tabela 4 - Testes percentual de acertos alterando o *seed*

	NAIVEBAYES	**J48**	**JRIP**	**IBK**
***Seed* 1**	91,20491%	91,59836%	91,68032%	73,71311%
***Seed* 2**	91,20491%	91,72131%	91,67213%	73,72131%
***Seed* 3**	91,21311%	91,68032%	91,89344%	73,93442%
***Seed* 4**	91,22950%	91,56557%	91,93442%	73,58196%
***Seed* 5**	91,21311%	91,37704%	91,86065%	73,86065%
***Seed* 6**	91,19672%	91,34426%	91,87704%	74,06557%
***Seed* 7**	91,21311%	91,63934%	91,71311%	73,85245%
***Seed* 8**	91,21311%	91,45901%	91,68852%	73,50000%
***Seed* 9**	91,22131%	91,23770%	91,89344%	73,89344%
***Seed* 10**	91,21311%	91,85245%	91,80327%	73,96721%
MÉDIA:	**91,21229%**	**91,54754%**	**91,80163%**	**73,80901%**

Fonte: O autor (2018)

Conforme pode ser visto na Tabela 4, foram destacados de azul o melhor resultado de cada algoritmo e laranja o pior resultado. Os algoritmos *NaiveBayes*, *J48* e *JRip* obtiveram acurácia superior a 91% o que pode ser considerado um excelente resultado considerando o quanto a língua portuguesa é complexa.

O algoritmo IBK teve seu melhor resultado 74,06%, comparando-se a sua média com os demais algoritmos é possível notar-se que o mesmo é inferior aos demais aproximadamente 17,71%. Como os algoritmos lidam com três classes, se considerarmos a classificação através de uma escolha aleatória a chance de erro e acerto são 33,33% e 66,67% respectivamente.

3.2 MATRIZ DE CONFUSÃO

Segundo Meneghetti e Kux (2014), a avaliação quantitativa de classificações realizadas pelos algoritmos é na maioria das vezes a partir de uma matriz de confusão. As tabelas abaixo, apresentam as matrizes de confusão de todos os algoritmos utilizados nesse trabalho.

Foram geradas duas matrizes para cada algoritmo, contendo o melhor e pior desempenho. Os valores destacados na diagonal são a quantidade de acertos e o restante são os erros do classificador.

Nesse trabalho foi realizado o cálculo de precisão utilizando a seguinte fórmula:

P = VP / VP + FP

- VP → Verdadeiro Positivo (número de acertos da classe)
- FP → Falso Positivo (número de vezes que a classe foi utilizada de maneira errada)

3.2.1 **Algoritmo *NaiveBayes***

Tabela 5 - Melhor desempenho algoritmo *NaiveBayes* com *seed* 4

	opinião	**fato**	***fake***
opinião	4.159	0	29
fato	458	3.512	90
fake	416	77	3.459

Fonte: O autor (2018)

Conforme pode ser analisado na Tabela 5, o algoritmo *NaiveBayes* com *seed* valor 4, obteve 11.130 (onze mil e cento e trinta) acertos, contra 1.070 (um mil e setenta) erros. Abaixo é destacado a precisão individual de cada classe:

- Opinião: 82,63%
- Fato: 97,85%
- *Fake*: 96,67%

Tabela 6 - Pior desempenho algoritmo *NaiveBayes* com *seed* 6

	opinião	**fato**	***fake***
opinião	4.158	0	30
fato	458	3.511	91
fake	415	80	3.457

Fonte: O autor (2018)

A Tabela 6 apresenta o pior desempenho do algoritmo *NaiveBayes,* comparando com a Tabela 5, nota-se que reduziu o número de acertos em todas as classes. A precisão ficou da seguinte forma:

- Opinião: 82,64%
- Fato: 97,77%
- *Fake: 96,61%*

3.2.2 **Algoritmo J48**

Tabela 7 - Melhor desempenho algoritmo J48 com *seed* 10

	opinião	**fato**	***fake***
opinião	3.869	150	169
fato	238	3.754	68
fake	295	74	3.583

Fonte: O autor (2018)

A Tabela 7 acima, apresenta o melhor desempenho do algoritmo J48, obtendo 11.206 (onze mil e duzentos e seis) acertos, contra 994 (novecentos e noventa e quatro) erros. Comparando-se ao melhor resultado *NaiveBayes* nota-se uma leve melhora, acertando 76 registros a mais.

Apesar de a classe opinião ter obtido menor número de acerto se comparado ao melhor resultado do algoritmo *NaiveBayes*, sua precisão aumentou, devido à classe ter sido menos classificada de maneira incorreta. Esse cenário inverte quando se comparado as classes: fato e *fake.* Conforme pode ser visto abaixo:

- Opinião: 87,89%
- Fato: 94,36%

- *Fake*: 93,79%

Tabela 8 - Pior desempenho algoritmo J48 com *seed* 9

	opinião	**fato**	***fake***
opinião	3.845	170	173
fato	272	3.724	64
fake	321	69	3.562

Fonte: O autor (2018)

O pior desempenho do algoritmo J48 também obteve melhor precisão na classe opinião e piores resultados nas classes: fato e *fake*, se comparado com o pior desempenho do *NaiveBayes*.

- Opinião: 86,63%
- Fato: 93,96%
- *Fake*: 93,76%

3.2.3 **Algoritmo JRip**

Tabela 9 - Melhor desempenho algoritmo JRip com *seed* 4

	opinião	**fato**	***fake***
opinião	4.090	25	73
fato	419	3.586	55
fake	384	28	3.540

Fonte: O autor (2018)

O algoritmo JRip obteve melhor acurácia entre todos os algoritmos utilizados, apesar de sua precisão não ser a melhor entre eles, o algoritmo obteve 11.216 (onze mil e duzentos e dezesseis) acertos e 984 (novecentos e oitenta e quatro) erros.

Analisando a quantidade de acertos no melhor desempenho dos algoritmos, *NaiveBayes* e J48, o JRip obteve um valor superior de dez e oitenta e seis registros respectivamente. Sua precisão foi superior nas classes: fato e *fake*, se comparado

com J48 e superior em todas as classes se comparado com *NaiveBayes.* Abaixo é mostrado a precisão obtida:

- Opinião: 83,58%
- Fato: 98,54%
- *Fake*: 96,51%

Tabela 10 - Pior desempenho algoritmo JRip com *seed* 2

	opinião	**fato**	***fake***
opinião	4.089	32	67
fato	437	3.569	54
fake	393	33	3.526

Fonte: O autor (2018)

O pior desempenho do JRip, obteve menor precisão que o *NaiveBayes* na classe *fake* se comparado ao pior resultado. O Restante manteve-se no mesmo padrão que seu melhor resultado.

- Opinião: 83,58%
- Fato: 98,54%
- *Fake*: 96,41%

3.2.4 **Algoritmo IBK**

Tabela 11 - Melhor desempenho algoritmo IBK com *seed* 6

	opinião	**fato**	***fake***
opinião	3.981	126	81
fato	545	2.689	826
fake	1.518	68	2.366

Fonte: O autor (2018)

O algoritmo IBK teve a pior acurácia entre os demais, apesar de sua precisão na classe: fato, estar bem próxima aos outros.

- Opinião: 65,86%

- Fato: 93,27%
- *Fake*: 72,28%

Tabela 12 - Pior desempenho algoritmo IBK com *seed* 8

	opinião	**fato**	***fake***
opinião	3.994	120	74
fato	545	2.660	855
fake	1.567	72	2.313

Fonte: O autor (2018)

Abaixo é descrito a precisão do algoritmo IBK, em seu pior desempenho, tendo resultados bem próximo a seu melhor resultado:

- Opinião: 65,41%
- Fato: 93,26%
- *Fake*: 71,34%

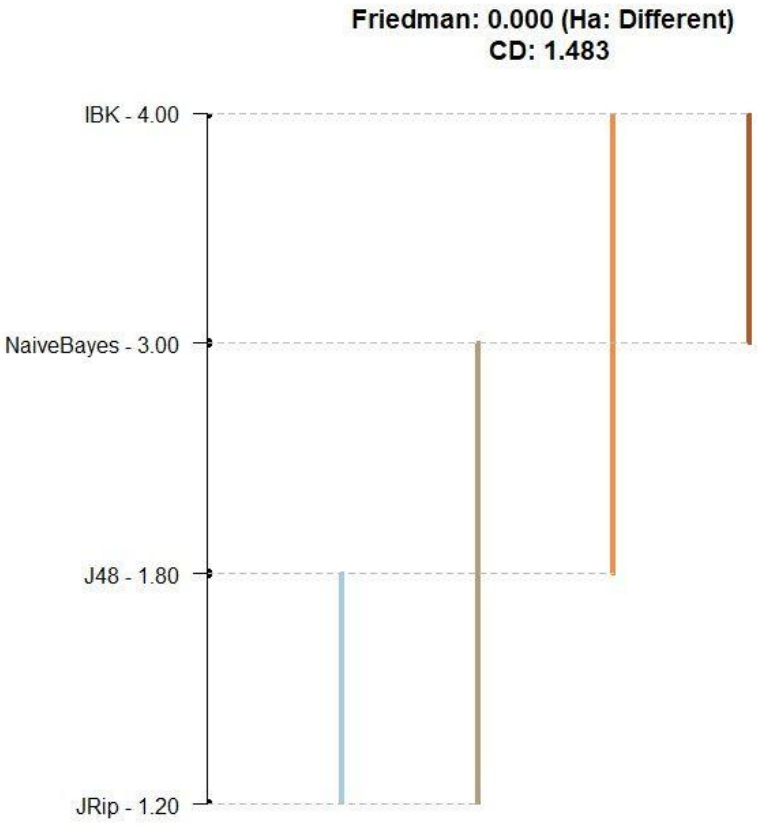
Analisando a precisão dos algoritmos é possível notar que o algoritmo *NaiveBayes* teve a melhor performance quanto a classe *fake*, já na classe fato o algoritmo que se destacou foi o JRip tendo uma precisão de 98,54% e por fim na classe opinião o algoritmo J48 foi superior aos demais.

3.3 TESTE DE FRIEDMAN E NEMENYI

O teste de Friedman e Nemenyi abaixo, foi realizado com finalidade de afirmar se um algoritmo possui diferença estatística significativa entre os demais algoritmos, utilizando a base de resultados variando o valor do *seed,* conforme foi exibido na Tabela 4.

O Gráfico 1 a seguir, apresenta a distância crítica (CD) igual a 1.483, para analisar se possui ou não diferença estatística significativa, é realizado a subtração entre os dois algoritmos a ser comparados, caso o valor seja superior a distância crítica pode-se afirmar a diferença.

Gráfico 1 - Teste de Friedman e Nemenyi



Fonte: O autor (2018)

Através desses testes chega-se à conclusão que o algoritmo JRip possui diferença estatística significante dos algoritmos *NaiveBayes* e IBK. Já o algoritmo J48 possui diferença estatística significante apenas do IBK.

CONCLUSÃO

Através do desenvolvimento desse trabalho, foi possível validar a utilização de aprendizagem de máquina para a identificação e classificação das *fake news*, haja visto que a classe *fake* obteve ótima precisão, sendo superior a 90% em três dos quatro algoritmos utilizados. Analisando o algoritmo na totalidade, ainda sim obtemos uma acurácia acima dos 90% em três algoritmos.

Efetuando um levantamento quantitativo dos registros classificados como *fake news,* foi possível notar-se que os três candidatos mais citados foram Cabo Daciolo sendo citado em 68.330 (sessenta e oito mil e trezentos e trinta) registros, seguido do candidato Fernando Haddad com 57.145 (cinquenta e sete mil e cento e quarenta e cinco) e por fim o candidato Henrique Meirelles, sendo citado em 47.795 (quarenta e sete mil e setecentos e noventa e cinco) registros.

Por se tratar de um levantamento quantitativo não é possível obter uma conclusão quanto a algum tipo de influência positiva ou negativa sobre os candidatos. Com os algoritmos implementados nesse trabalho é possível efetuar uma análise mais profunda nesses registros. Sendo necessário apenas o treinamento dos algoritmos para análise a ser realizada.

Algumas dificuldades foram encontradas durante o desenvolvimento, como a limitação de *hardware* do autor, dificuldades na classificação dos registros de forma manual, conteúdos de qualidade sobre aprendizagem de máquina e coleta de dados limitada.

Foi fundamental para os resultados obtidos nesse trabalho, a utilização do corpus disponibilizado por Monteiro et al. (2018), pois, os registros classificados manualmente estavam desbalanceados para a classe opinião afetando o desempenho dos algoritmos nas classes *fake* e fato.

Aprendizagem de máquina tem sido utilizado para resolução de diversos problemas da sociedade, e foi possível notar durante o desenvolvimento do trabalho, que no Brasil essa técnica ainda é pouco explorada se comparada com outros países como os Estados Unidos.

As *fake news* apesar de serem um grande problema a ser combatido, nota-se pouco material cientifico para resolução de forma automatizada. Pode-se concluir que o objetivo principal do trabalho em classificação e identificação de *fake news* obteve

sucesso. Os resultados obtidos foram além da expectativa do autor, haja visto que classificação de textos em português é uma atividade complexa.

TRABALHOS FUTUROS

- Classificar mais registros de forma manual e efetuar novo treinamento e teste dos algoritmos para os reavaliar.
- Analisar a propagação de uma *fake news* identificada.
- Analisar uma influência nos resultados da eleição.
- Desenvolvimento de aplicativos utilizando o modelo criado para ajuda aos usuários na detecção de *fake news*.
- Testar outros algoritmos.
- Utilização de outras bases de dados para coleta de registros.

REFERÊNCIAS

ABERNETHY, Michael. **Mineração de dados com WEKA, Parte 1**: Introdução e regressão. 2010. Disponível em: <https://www.ibm.com/developerworks/br/opensource/library/os-weka1/index.html>. Acesso em: 21 out. 2018.

AGEAC. TENDÊNCIAS TECNOLÓGICAS PARA 2018. **Associação Gaúcha de Empresas de Automação Comercial**. 2017. Disponível em: <http://www.ageac.inf.br/noticias/36,0,/tendencias-tecnologicas-para-2018.html>. Acesso em: 15 nov. 2018.

ALTERYX INC. A ferramenta Classificador Naive Bayes. **Alteryx**. 2018. Disponível em: <https://help.alteryx.com/current/pt-br/Naive_Bayes.htm>. Acesso em: 15 nov. 2018.

ARA-SOUZA, Anderson Luiz. REDES BAYESIANAS: UMA INTRODUÇÃO APLICADA A CREDIT SCORING. In: SIMPÓSIO NACIONAL DE PROBABILIDADE E ESTATÍSTICA, 19. 2010, São Carlos, 2010. Disponível em: <http://www.ime.unicamp.br/sinape/sites/default/files/Anderson%20L.%20Souza%20-%20Redes%20Bayesianas-%20vSINAPE%20final_0.pdf>. Acesso em: 15 out. 2018.

ARAL, Sinan; ROY, Deb; VOSOUGHI, Soroush. *Study: On Twitter, false news travels faster than true stories: Research project finds humans, not bots, are primarily responsible for spread of misleading information.* **MIT News**. 2018. Disponível em: <http://news.mit.edu/2018/study-twitter-false-news-travels-faster-true-stories-0308>. Acesso em: 15 nov. 2018.

ARANHA, Christian Nunes. **Uma Abordagem de Pré-processamento Automático para Mineração de Textos em Português: Sob o Enfoque da Inteligência Computacional**. Rio de Janeiro, 2007 Tese (Engenharia Elétrica) - PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO, 2007.

BAKER, Ryan Shaun Joazeiro; ISOTANI, Seiji; CARVALHO, Adriana Maria Joazeiro Baker. Mineração de Dados Educacionais: Oportunidades para o Brasil.**Revista Brasileira de Informática na Educação**, v. 19, n. 2, 24 ago 2011. Disponível em: <http://br-ie.org/pub/index.php/rbie/article/view/1301/1172>. Acesso em: 18 nov. 2018.

BATISTA, Rafael. Fake News: A divulgação de notícias falsas, conhecidas como fake news, pode interferir negativamente em vários setores da sociedade, como política, saúde e segurança. **Mundo Educação**, 2018. Disponível em:

<https://mundoeducacao.bol.uol.com.br/curiosidades/fake-news.htm>. Acesso em: 15 nov. 2018.

BERRY, Michael J. A.; LINOFF, Gordon S. ***Data Mining Techniques****: For Marketing, Sales, and Customer Relationship Management.* 2. ed. Wilwy Publishing, Inc., 2004.

BRICCE, Bruno Roberto et al. APLICAÇÃO DO ALGORITMO KNN PARA CONTROLE DE MOVIMENTOS DE NPCS EM UM AMBIENTE DINÂMICO (JOGO). **REGRAD - Revista Eletrônica de Graduação do UNIVEM - ISSN 1984-7866**, v. 9, n. 1, p. 18-32, aug 2016. Disponível em: <http://revista.univem.edu.br/REGRAD/article/view/1317>. Acesso em: 15 nov. 2018.

BRITO, Edeleon Marcelo Nunes. **Mineração de Textos: Detecção automática de sentimentos em comentários nas mídias sociais**. Belo Horizonte, 2016 Dissertação (Sistemas de Informação e Gestão do Conhecimento) - UNIVERSIDADE FUMEC, 2016.

BROWNLEE, Jason. A Gentle Introduction to k-fold Cross-Validation. ***Machine Learning Mastery****.* 2018. Disponível em: <https://machinelearningmastery.com/k-fold-cross-validation/>. Acesso em: 6 nov. 2018.

BRUSSO, Marcos J.; NAVAUX, Philippe O. A.; GEYER, CLÁUDIO F. R. **Um Modelo para a Descoberta de Regras de Associação Aplicado à Mineração do Uso da Web**. 2000 Trabalho de Disciplina (Geociências) - UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL.

CAMILO, Cássio Oliveira; SILVA, João Carlos. **Mineração de Dados: Conceitos, Tarefas, Métodos e Ferramentas**. Goiás, 2009 Trabalho de Disciplina (Ciência da Computação) - UNIVERSIDADE FEDERAL DE GOIÁS. Disponível em: <http://www.portal.inf.ufg.br/sites/default/files/uploads/relatorios-tecnicos/RT-INF_001-09.pdf>. Acesso em: 15 nov. 2018.

CREPALDI, Paola Guarisso et al. UM ESTUDO SOBRE A ÁRVORE DE DECISÃO E SUA IMPORTÂNCIA NA HABILIDADE DE APRENDIZADO. **Faculdade Inesul**. 2010. Disponível em: <https://www.inesul.edu.br/revista/arquivos/arq-idvol_15_1320100263.pdf>. Acesso em: 16 out. 2018.

D'AGOSTINO, Rosanne. Três anos depois, linchamento de Fabiane após boato na web pode ajudar a endurecer lei. **G1**. 2017. Disponível em: <https://g1.globo.com/e-ou-nao-e/noticia/tres-anos-depois-linchamento-de-fabiane-apos-boato-na-web-pode-ajudar-a-endurecer-lei.ghtml>. Acesso em: 15 nov. 2018.

FALQUETO, Daniel. **REDE NEURAL ARTIFICIAL PARA RECONHECIMENTO DE HORÁRIOS DE ARME/DESARME NO SISTEMA SIGMA**. São José, SC, 2007 TCC (Ciência da Computação) - UNIVERSIDADE DO VALE DO ITAJAÍ, 2007. Disponível em: <http://siaibib01.univali.br/pdf/Daniel%20Falqueto.pdf>. Acesso em: 21 out. 2018.

FAYYAD, Usama; PIATETSKY-SHAPIRO, Gregory; SMYTH, Padhraic. ***From Data Mining to Knowlegde discovery****: American Association for Artificial Intelligence.* 1996.

FERRERO, Carlos Andres. **Algoritimo kNN para previsão de dados temporais: funões de previsão e critérios de seleção de vizinhos próximos aplicados a variáveis ambientais em limnologia**. São Carlos, 2009 Dissertação (Ciências de Computação e Matemática Computacional) - UNIVERSIDADE DE SÃO PAULO, 2009.

FRANÇA, Tiago C.; OLIVEIRA, Jonice. Análise de Sentimento de Tweets Relacionados aos Protestos que ocorreram no Brasil entre junho e agosto de 2013. In: III BRAZILIAN WORKSHOP ON SOCIAL NETWORK ANALYSIS AND MINING, 3. 2014. 2014.

GOLDSCHMIDT, Ronaldo Ribeiro. **Uma introdução à Inteligência Computacional**: Fundamentos, Ferramentas e Aplicações. 1. ed. Rio de Janeiro: Instituto Superior de Tecnologia do Rio de Janeiro, 2010.

GONÇALVES, Cristiano; TOSTES, Lelia de Mello. **O Uso de Redes Neurais Artificiais no Diagnóstico de Doenças Reumatológicas.** Florianópolis, 2004 TCC (Sistema de Informação) - UNIVERSIDADE FEDERAL DE SANTA CATARINA, 2004. Disponível em: <https://repositorio.ufsc.br/bitstream/handle/123456789/184264/TCC.pdf?sequence=-1>. Acesso em: 21 out. 2018.

HALL, Mark et al. **The WEKA Data Mining Software: An Update**. SIGKDD Explorations, v. 11, 2009. Disponível em: <http://www.kdd.org/exploration_files/p2V11n1.pdf>. Acesso em: 17 out. 2018.

HOSOKAWA, Eric Ossamu. **Técnica de Árvore de Decisão em Mineração de Dados**. São Paulo, 2011 Monografia (Processamento de dados) - FACULDADE DE TECNOLOGIA DE SÃO PAULO, 2011. Disponível em: <http://www.fatecsp.br/dti/tcc/tcc0003.pdf>. Acesso em: 16 out. 2018.

HUGO, Kewerson. **Utilizando Técnicas de extração de conhecimento no apoio a tomada de decisão no ambiente educacional: Um estudo de caso sobre a influência da escolaridade dos pais dos candidatos no desempenho do enem.**. 2017. Disponível em: <https://www.slideshare.net/kewerson93/processo-de-

descoberta-de-conhecimento-em-base-de-dados-extrao-de-informaes-acerca-da-realidade-dos-estudantes-brasileiros>. Acesso em: 4 out. 2018.

IBGE. PNAD Contínua TIC 2016: 94,2% das pessoas que utilizaram a Internet o fizeram para trocar mensagens. **Instituto Brasileiro de Geografia e Estatística**. 2016. Disponível em: <https://www.ibge.gov.br/estatisticas-novoportal/sociais/trabalho/17270-pnad-continua.html?edicao=19937&t=sobre>. Acesso em: 15 nov. 2018.

LIBRELOTTO, Solange Rubert; MOZZAQUATRO, Patricia Mariotto. ANÁLISE DOS ALGORITMOS DE MINERAÇÃO J48 E APRIORI APLICADOS NA DETECÇÃO DE INDICADORES DA QUALIDADE DE VIDA E SAÚDE. **Revista Interdisciplinar de Ensino, Pesquisa e Extensão**, v. 1, n. 1, p. 26-37, 2013. Disponível em: < http://www.revistaeletronica.unicruz.edu.br/index.php/eletronica/article/view/26-37/pdf>. Acesso em: 15 nov. 2018.

JÚNIOR, Renato Gomes Borges. **Aprendizado de Máquina para Análise de Recaída para Depressão em Pacientes com Transtorno Bipolar**. Goiânia, 2018 Tese (Ciência da Computação) - UNIVERSIDADE FEDERAL DE GOIÁS. Disponível em: <https://repositorio.bc.ufg.br/tede/bitstream/tede/9024/5/Disserta%C3%A7%C3%A3o%20-%20Renato%20Gomes%20Borges%20J%C3%BAnior%20-%202018.pdf>. Acesso em: 1 nov. 2018.

MARQUEZAN, Reinoldo. A constituição do corpus de pesquisa. **Educação Especial**. Santa Maria, v. 22, n. 33, p. 97-110, 2009. Disponível em:<https://periodicos.ufsm.br/educacaoespecial/article/view/172/102>. Acesso em:25 nov. 2018.

MARTINS, Dálton . **Gestão da Informação e do Conhecimento**. 2018 Trabalho de Disciplina (Gestão da Informação) - UNIVERSIDADE FEDERAL DE GOIÁS. Disponível em: <https://l3p.fic.ufg.br/up/771/o/Aula_01_-_Gest%C3%A3o_da_Informa%C3%A7%C3%A3o_e_do_Conhecimento.pdf?1443994379>. Acesso em: 15 nov. 2018.

MENEGHETTI, Graziela Thaís; KUX, Hermann Johann Heinrich. MAPEAMENTO DA COBERTURA DA TERRA DO MUNICÍPIO DE RAPOSA (MA) UTILIZANDO IMAGENS WORLDVIEW-II, O APLICATIVO INTERIMAGE E MINERAÇÃO DE DADOS. **Revista Brasileira de Cartografia**, n. 66/2, 2014. Disponível em: < http://www.lsie.unb.br/rbc/index.php/rbc/article/view/547/684>. Acesso em: 14 dez. 2018.

MONARD, Maria Carolina; BARANAUSKAS, José Augusto . Indução de Regras e Árvores de Decisão. In: MONARD, Maria Carolina; BARANAUSKAS, José Augusto.

Indução de Regras e Árvores de Decisão. 2003. cap. 5. Disponível em: <http://dcm.ffclrp.usp.br/~augusto/publications/2003-sistemas-inteligentes-cap5.pdf>. Acesso em: 17 out. 2018.

MONTEIRO, R.A. et al. *Contributions to the Study of Fake News in Portuguese: New Corpus and Automatic Detection Results.* In: COMPUTATIONAL Processing of the Portuguese Language. Springer International Publishing, 2018, p. 324-334.

PADILHA, Thereza Patrícia Pereira; ALMEIDA, Leandro Maciel; ALVES, João Bosco da Mota. Modelagem do Desempenho do Aprendizado de Grupos de Alunos Utilizando Data Mining. **XIV Simpósio Brasileiro de Informática na Educação**, 2003. Disponível em: <http://www.br-ie.org/pub/index.php/sbie/article/viewFile/263/249>. Acesso em: 1 nov. 2018.

PEREIRA, Luciano de Santana. **METODOLOGIA PARA AVALIAR TECNICAS ´ DE REDUC¸ AO DE PROT ˜ OTIPOS: ´ PROTOTIPOS GERADOS ´ VERSUS PROTÓTIPOS SELECIONADOS**. Recife, 2013 Dissertação (Pós-graduação em Ciência da Computação) - UNIVERSIDADE FEDERAL DE PERNAMBUCO, 2013. Disponível em: <https://repositorio.ufpe.br/bitstream/123456789/12402/1/Disserta%C3%A7ao%20Luciano%20Pereira.pdf>. Acesso em: 15 nov. 2018.

PETERMANN, Rafael Jordan. **Modelo de mineração de dados para classificação de clientes em telecomunicações**. Porto Alegre, 2006 Dissertação (Energias Renováveis) - PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL, 2006. Disponível em: <http://tede2.pucrs.br/tede2/bitstream/tede/3044/1/388093.pdf>. Acesso em: 10 out. 2018.

PRASS, Fernando Sarturi. **KDD - UMA VISAL GERAL DO PROCESSO**. 2012. Disponível em: <http://fp2.com.br/blog/wp-content/uploads/2012/07/KDD_Uma_visao_geral_do_processo.pdf>. Acesso em: 8 out. 2018.

PRATI, Ronaldo Cristiano. **Novas abordagens em aprendizado de máquina para a geração de regras, classes desbalanceadas e ordenação de casos**. São Carlos, 2006 Tese () - UNIVERSIDADE DE SÃO PAULO, 2006. Disponível em: <http://www.teses.usp.br/teses/disponiveis/55/55134/tde-01092006-155445/publico/RonaldoPratiTese.pdf>. Acesso em: 17 out. 2018.

PROJETO COMPROVA. Até logo. **Comprova**. 2018. Disponível em: <https://projetocomprova.com.br/>. Acesso em: 15 nov. 2018.

REUTERS INSTITUTE. Reuters Institute Digital News Report 2018. **Digital News Report**. 2018. Disponível em: <http://www.digitalnewsreport.org/survey/2018/brazil-2018/>. Acesso em: 15 nov. 2018.

SCHMITT, Vinícius Fernandes. **UMA ANÁLISE COMPARATIVA DE TÉCNICAS DE APRENDIZAGEM DE MÁQUINA PARA PREVER A POPULARIDADE DE POSTAGENS NO FACEBOOK**. Ponto Alegre, 2013 TCC (Ciência da Computação) - UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL, 2013.

SILVA, Rodrigo R.; MACIEL, Thales V.; LAMPERT, Vinícius do N.. Previsão de Indicadores de Qualidade de Carcaças na Pecuária de Corte Através de Aplicações de Mineração de Dados. **Congresso Argentino de Agro Informática**, 2018. Disponível em: <http://47jaiio.sadio.org.ar/sites/default/files/CAI-37.pdf>. Acesso em: 15 nov. 2018.

SOARES, Silviane. **Aplicação de técnicas de mineração de dados na gestão de sistemas de energia elétrica**. Porto Alegre, 2005 Dissertação (Engenharia Elétrica) - PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO GRANDE DO SUL.

TAKATUZI, Fabio Kenji Oshiro; STANGE, Renata Luiza. **Aprendizagem de Regras de Decisão Utilizando Técnicas Adaptativas: Experimentos Preliminares**. 2018. Disponível em: <http://lta.poli.usp.br/lta/publicacoes/artigos/2018/takatuzi-e-stange-2018-aprendizagem-de-regras-de-decisao-utilizando-tecnicas-adaptativas-experimentos-preliminares/at_download/file>. Acesso em: 17 out. 2018.

TAKATUZI, Fabio Kenji Oshiro. **Algoritmo Incremental para Aprendizagem de Árvores de Decisão Adaptativas**. Guarapuava, 2016 TCC (Tecnologia em Sistemas para Internet) - UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ, 2016. Disponível em: <http://tcc.tsi.gp.utfpr.edu.br/attachments/approvals/45/GP_COINT_2016_1_FABIO_KENJI_OSHIRO_TAKATUZI_PROPOSTA.pdf?1463752293>. Acesso em: 15 nov. 2018.

TSE. Ministro Luiz Fux afirma que Justiça Eleitoral faz combate efetivo às *fake news*. **Tribunal Superior Eleitoral**. 2018. Disponível em: <http://www.tse.jus.br/imprensa/noticias-tse/2018/Agosto/ministro-luiz-fux-afirma-que-justica-eleitoral-faz-combate-efetivo-as-fake-news>. Acesso em: 15 nov. 2018.

TWITTER. *Search Tweets*. **Twitter *Developer***. 2018. Disponível em: <https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets.html>. Acesso em: 15 nov. 2018.

TWITTER4J. **Twitter4J**. 2018. Disponível em: <http://twitter4j.org/en/index.html>. Acesso em: 1 nov. 2018.

U.S. *DEPARTMENT OF JUSTICE. Grand Jury Indicts Thirteen Russian Individuals and Three Russian Companies for Scheme to Interfere in the United States Political System.* ***The United States Departament Justice***. 2018. Disponível em: <https://www.justice.gov/opa/pr/grand-jury-indicts-thirteen-russian-individuals-and-three-russian-companies-scheme-interfere>. Acesso em: 15 nov. 2018.

UNIVERSITY OF WAIKATO. Weka 3: Software de Mineração de Dados em Java. **University of Waikato**. 2018. Disponível em: <https://www.cs.waikato.ac.nz/ml/weka/>. Acesso em: 17 jul. 2018.

VALOR ECONÔMICO. FATO ou FAKE: quase mil checagens publicadas na eleição. Veja balanço. **Valor Econômico**. 2018. Disponível em: <https://www.valor.com.br/fatooufake/5955433/fato-ou-fake-quase-mil-checagens-publicadas-na-eleicao-veja-balanco>. Acesso em: 15 nov. 2018.

VIANA, Zarathon Lopes. **MINERAÇÃO DE TEXTOS: ANÁLISE DE SENTIMENTO UTILIZANDO TWEETS REFERENTES ÀS ELEIÇÕES PRESIDENCIAIS 2014**. Quixadá, 2014 TCC (Sistema de Informação) - UNIVERSIDADE FEDERAL DO CEARÁ, 2014.

VILAR, João. Fundamentos de Data Science – Machine Learning (Parte 1). **JOAO VILAR |** ***TECHNOLOGY***. 2017. Disponível em: <https://jvilar.wordpress.com/2017/01/29/fundamentos-de-data-science-machine-learning-parte-1/>. Acesso em: 27 set. 2018.

WE ARE SOCIAL LTD. Digital In 2018. **We Are Social**. 2018. Disponível em: <https://wearesocial.com/blog/2018/10/the-state-of-the-internet-in-q4-2018>. Acesso em: 15 nov. 2018.

WIKIPÉDIA. Twitter. **Wikipédia**. 2018. Disponível em: <https://pt.wikipedia.org/wiki/Twitter>. Acesso em: 21 out. 2018.

WIVES, Leandro Krug; LOH, Stanley. Recuperação de Informações usando a Expansão Semântica e a Lógica Difusa. In: CONGRESSO INTERNACIONAL EM INGENIERIA INFORMÁTICA, Buenos Aires, 1998. Disponível em: <http://livrozilla.com/doc/817821/recupera%C3%A7%C3%A3o-de-informa%C3%A7%C3%B5es-usando-a-expans%C3%A3o-sem%C3%A2ntica-e-a>. Acesso em: 15 nov. 2018.

ANEXO A — LISTA DE *STOPWORDS*

a, agora, ainda, alguém, algum, alguma, algumas, alguns, ampla, amplas, amplo, amplos, ante, antes, ao, aos, após, aquela, aquelas, aquele, aqueles, aquilo, as, até, através, cada, coisa, coisas, com, como, contra, contudo, da, daquele, daqueles, das, de, dela, delas, dele, deles, depois, dessa, dessas, desse, desses, desta, destas, deste, deste, destes, deve, devem, devendo, dever, deverá, deverão, deveria, deveriam, devia, deviam, disse, disso, disto, dito, diz, dizem, do, dos, e, é, ela, elas, ele, eles, em, enquanto, entre, era, essa, essas, esse, esses, esta, está, estamos, estão, estas, estava, estavam, estávamos, este, estes, estou, eu, fazendo, fazer, feita, feitas, feito, feitos, foi, for, foram, fosse, fossem, grande, grandes, há, isso, isto, já, la, lá, lhe, lhes, lo, mas, me, mesma, mesmas, mesmo, mesmos, meu, meus, minha, minhas, muita, muitas, muito, muitos, na, não, nas, nem, nenhum, nessa, nessas, nesta, nestas, ninguém, no, nos, nós, nossa, nossas, nosso, nossos, num, numa, nunca, o, os, ou, outra, outras, outro, outros, para, pela, pelas, pelo, pelos, pequena, pequenas, pequeno, pequenos, per, perante, pode, pude, podendo, poder, poderia, poderiam, podia, podiam, pois, por, porém, porque, posso, pouca, poucas, pouco, poucos, primeiro, primeiros, própria, próprias, próprio, próprios, quais, qual, quando, quanto, quantos, que, quem, são, se, seja, sejam, sem, sempre, sendo, será, serão, seu, seus, si, sido, só, sob, sobre, sua, suas, talvez, também, tampouco, te, tem, tendo, tenha, ter, teu, teus, ti, tido, tinha, tinham, toda, todas, todavia, todo, todos, tu, tua, tuas, tudo, última, últimas, último, últimos, um, uma, umas, uns, vendo, ver, vez, vindo, vir, vos, vós, ;, ., +, -, *, /, \, !, ?, :