Universidade Católica do Salvador
Bacharelado em Engenharia de Software

Matheus da Silva Amorim
Juan Ribeiro Silva Pereira

Tipificação de Ocorrências Policiais Utilizando Machine Learning

Salvador
2019

Matheus da Silva Amorim
Juan Ribeiro Silva Pereira

Tipificação de Ocorrências Policiais Utilizando Machine Learning

Trabalho de Conclusão de Curso apresentado à Universidade Católica do Salvador como parte dos requisitos necessários para a obtenção do Título de Bacharel em Engenharia de Software.
Orientador: Prof. MSc Marcelo Indio dos Reis

Universidade Católica do Salvador
Bacharelado em Engenharia de Software

Salvador
2019

Matheus da Silva Amorim
Juan Ribeiro Silva Pereira

Tipificação de Ocorrências Policiais Utilizando Machine Learning

Trabalho de Conclusão de Curso apresentado à Universidade Católica do Salvador como parte dos requisitos necessários para a obtenção do Título de Bacharel em Engenharia de Software.Orientador: Prof MSc Marcelo Indio dos Reis

Comissão Examinadora

Prof. MSc Marcelo Indio dos Reis
Universidade Católica do Salvador
Orientador

Prof. MSc Pâmella Arielle Brito de Aquino
Universidade Católica do Salvador

Prof. Especialista Osvaldo Requião Melo
Universidade Católica do Salvador

Salvador, 7 de julho de 2019

Dedicamos este trabalho a todos aqueles que, de alguma forma,
auxiliaram para a concretização desta etapa.

Agradecimentos

Agradecemos a todos os professores e colaboradores do curso, além de nossas respectivas familias e amigos.

"Train yourself to let go of everything you fear to lose."
(Yoda)

Resumo

A segurança pública é um dos principais pilares para a sociedade, influenciando diretamente na qualidade de vida do cidadão. Ultimamente esta área vem recebendo um grande foco na questão de investimento, como Ballesteros (2014) diz em seu artigo, grande parte deste valor vem sendo destinado a automatização de processos. Nesse contexto, a tecnologia entra para dar apoio em diversas atividades rotineiras, tornando-as mais eficientes por meio de uma melhor gestão de seus recursos. Com isso foi proposto neste trabalho o uso de algoritmos de aprendizado de máquina, para tipificar de forma automática ocorrências policiais. Foi utilizada uma base de dados com registros policiais e dela foram retiradas algumas amostras para a realização de validação. Para essa tarefa foram selecionados os algoritmos C4.5, CART, KNN, SVM, Rede Neural, Ripper e *Random Forest*. Os resultados obtidos indicaram que o C4.5 e o Ripper tiveram a melhor acurácia, chegando a 99% em determinados testes. Em um teste a aplicação do coeficiente de correlação de *Matthews* e do F1 *score* chegaram a 0,89 e 0,94 nessa ordem.

Palavras-Chave: 1. Aprendizado de máquina. 2. Ocorrência policial. 3. Classificação.

Abstract

Public safety is one of the main pillars for society that directly influence the citizens quality of life. Lately this area has been receiving a great focus on the investment issue, as Ballesteros (2014) says his article, much of this value has been destined to process automation. In this context, technology enters to support various routine activities, making them more efficient through better management of their resources. With that, it was suggested in this assignment the use of machine learning algorithms, to automatically typify police occurrences. It was used a base of data with police records and from it some samples were taken for validation. For this tests were selected the algorithms C4.5, CART, KNN, SVM, Rede Neural, Ripper and Random Forest. The results gotten, indicated that the C4.5 and the Ripper had the best accuracy, becoming until 99% in some tests. In a test, the application of the coefficient of correlation of Matthews and of F1 score, reached 0.89 and 0.94 in that order.

Keywords: 1. Machine learning. 2. Police report. 3. Classification.

Lista de figuras

Lista de tabelas

Lista de Siglas e Abreviaturas

AS	*Aprendizado supervisionado*
AM	*Aprendizado de máquina*
ANS	*Aprendizado não supervisionado*
ASS	*Aprendizado semi-supervisionado*
CART	*Classification and regression trees*
CLTC	*Cross-lingual text classification*
CNN	*Convolutional Neural Networks*
CPU	*Central Process Unit*
DNN	*Deep Neural Networks*
HDLTex	*Hierarchical Deep Learning for Text classification*
ID3	*iterative Dichotomiser 3*
IREP	*incremental reduced error pruning*
KDD	*Knowledge discovery in Databases*
KNN	*K Nearest neighbor*
ML	*Machine learning*
ME	*Maximum entropy*
NA	*Not available*
NB	*Naive bayes*
SVM	*Support Vector Machines*
SGD	*Stochastic gradient descent*
RAM	*Random Access Memory*
RF	*Random forest*
Ripper	*Repeated Incremental Pruning to Produce Error Reduction*
RNN	*Recurrent Neural Network*

Sumário

1 Introdução

Segurança pública é um tema recorrente, presente na vida de todo brasileiro. Hamada e Nassif (2018) dizem que os problemas de segurança pública, apesar de generalizados em todo o país, concentram-se nas grandes metrópoles, nas quais grande parte dos crimes são cometidos, ocasionando uma sensação de insegurança.

Como é dito em Souza (2018), a segurança pública é vista como um sistema responsável por adotar ações que garantam a segurança da sociedade, mas conforme o aumento na quantidade de homicídios no Brasil (cerca de 62.517 homicídios, segundo (CERQUEIRA et al., 2018)), essas ações tem-se mostrado ineficazes. Neste contexto, encontra-se grandes desafios voltados para essa área, no qual aspectos como a falta de contingente policial, a visão negativa da população e da mídia e a remuneração baixa e desproporcional ao risco dos agentes da lei colabora para o aumento desses índices.

A segurança é um direito fundamental garantido pela Constituição brasileira e deve ser garantido pelo estado. Spanhol, Lunardi e Souza (2016) dizem que a Constituição brasileira de 1988, em seu artigo 5º, apresenta o direito à segurança como um dos direitos fundamentais, de igual importância ao direito à vida, à liberdade, à igualdade e à propriedade. Mas aparentemente atividades cotidianas como transitar em vias públicas tem-se tornado grandes aventuras em determinadas regiões do Brasil. Por conta disso foram criados alguns indicadores para organizar e classificar os inúmeros tipos de crimes contra patrimônio, a vida e entre outros.

Após a coleta desses indicadores é feita uma análise pela secretaria de Estado responsável por determinada região. Com os dados de todas as regiões coletados é feito um panorama nacional com uma porcentagem de crimes violentos e letais, sendo as vítimas em grande parte negras e jovens de acordo com o parâmetro homicídio demonstrado pelo Cerqueira et al. (2016). Nele é apresentado uma estatística, onde a cada sete jovens assassinados cinco são negros, e quando essa esfera é reduzida para um âmbito regional, há um aumento significativo nas proporções chegando a 19 negros mortos para cada branco que sofre homicídio. Já em roubos e furtos os papéis se invertem, pois os negros são as pessoas que mais cometem este tipo de delito. Isso vem de uma cultura onde jovens, com baixa escolaridade, renda e moradores de periferia veem na criminalidade uma alternativa a uma vida com poucas oportunidades.

Perante os desafios anteriormente citados, o *machine learning* destaca-se através da sua característica de auto aprendizado (com base no conhecimento previamente adquirido oriundo de uma base de dados de tamanho considerável), sendo possível utilizá-lo para realizar tarefas como identificação de padrões a partir de fichas criminais, realizar a predição de crimes por meio dos modelos gerados, detectar tendências e comportamentos de

atividades suspeitas.

O uso de *machine learning* na área é útil, pois ele permite a automatização de várias tarefas que são, normalmente, realizadas por humanos. O seu potencial de uso pode ser visto a depender de como e onde ele é aplicado. Entre algumas possíveis aplicações do *machine learning* inclui a filtragem de informações úteis, detecção de invasão e reconhecimento de padrões e imagens.

1.1 Justificativa

Nota-se que, de forma geral, o investimento em tecnologia na área de segurança pública vem aumentando com o passar do tempo. Mas ainda assim não é suficiente, visto que Cezar Schirmer[1] (2017) diz que o Brasil está muito aquém em novas tecnologias relacionadas a área, pois mesmo com investimentos em câmeras e cercamentos eletrônicos, além de outras formas de apoios tecnológicos, as taxas de criminalidades continuam altas.

Segundo Oliveira e Almeida (2018) a execução dos atos somente por escrito demonstra conflito com todos os avanços tecnológicos realizados no mundo atual, documentos e processos vem sendo cada vez mais modernizados, enquanto que o inquérito policial continua sendo produzido de forma manual tornando-se ineficiente. Visto que a modernização tecnológica vem crescendo cada vez mais na sociedade contemporânea, uma solução para contornar essa crescente criminalidade é utilizar a tecnologia para dar apoio, e em alguns casos, substituir a necessidade de utilização de mão de obra em algumas atividades mais objetivas.

Logo, este trabalho pretende mostrar como o *machine learning* pode ser inserido no contexto policial a fim de ajudar na tipificação de ocorrências, visando auxiliar as atividades dos profissionais da área de segurança pública.

1.2 Problema

Boletim de Ocorrência policial é a forma de registrar e detalhar as circunstâncias de um ato, sendo ele criminal ou não, registrado através de um funcionário público. Ele impacta diretamente na segurança pública ou na manutenção da mesma. Neste contexto, este trabalho tem como problema de pesquisa: É possível realizar a tipificação automatizada de uma ocorrência policial através do uso de *machine learning*?

[1] 7º Fórum Nacional de Tecnologia e Inovação da Segurança Pública, que ocorreu em Porto Alegre.

1.3 Hipótese

Com base no problema, a seguinte hipótese foi formulada: É possível realizar a tipificação automatizada de ocorrências policiais utilizando técnicas de *machine learning*.

1.4 Objetivos

1.4.1 Objetivo Geral

Neste trabalho, apresenta-se como objetivo geral tipificar ocorrências policiais utilizando algoritmos de *machine learning*.

1.4.2 Objetivo Específicos

Os objetivos específicos definidos para este trabalho são:

- Gerar o *dataset* que será utilizado no trabalho.
- Selecionar os algoritmos de *machine learning* que serão utilizados no trabalho.
- Gerar os modelos de classificação que serão utilizados.
- Realizar a classificação.
- Avaliar os resultados obtidos.

Este trabalho está dividido da seguinte forma: a seção 2 é destinada a falar sobre a revisão de literatura necessária para a realização do projeto. A seção 3 fala da metodologia que é utilizada. A seção 4 mostra os resultados obtidos durante os experimentos e a seção 5 as considerações finais.

2 Revisão de Literatura

2.1 *Machine Learning* (Aprendizado de máquina)

O aprendizado de máquina (AM) utiliza técnicas para permitir que um dispositivo computacional adquira experiência, por meio do conhecimento que já aprendeu, para melhorar a sua performance. Ele tem como base as ideias do aprendizado indutivo, que como Mitchell (1997) diz, é o problema de induzir uma hipótese que cubra todos os exemplos positivos e não cubra nenhum dos exemplos negativos.

Segundo Mitchell (1997) ao longo do tempo em que algoritmos de *machine learning* eram criados, foi utilizado embasamento empírico, teórico ou até mesmo uma combinação de ambos. Esses sistemas têm sido desenvolvidos utilizando diferentes paradigmas de aprendizado, tais como estatístico, conexionista, baseado em instância, genético e sistemas de aprendizado simbólico entre outros.

Segundo Kotsiantis (2007), aprendizado de máquina indutivo é o processo de aprender um conjunto de regras a partir de instâncias. De maneira mais geral é um processo, no qual é criado um classificador que pode ser usado para generalizar a partir de novas instâncias.

O aprendizado indutivo é implementado por meio de algoritmos que fazem o processamento de determinado conjunto de dados. A partir disso, é possível extrair modelos de classificação que vão servir de base para dar uma classe a uma instância nova. O aprendizado indutivo pode ser dividido em três partes: aprendizado supervisionado, não supervisionado e semi-supervisionado:

- Aprendizado supervisionado (AS): Ocorre quando uma classe é atribuída a instâncias de treinamento por meio de um especialista. Ele é dividido em métodos de regressão e classificação. Os métodos de regressão tendem a encontrar a evolução de uma variável em relação às outras. Já os métodos de classificação são aqueles que procuram dar uma explicação a uma variável categórica.

- Aprendizado não supervisionado (ANS): Diferente do AS, o ANS não envolve o trabalho de um especialista e, consequentemente, as instâncias não possuem uma classe atribuída a elas. É mais usado para descobrir relacionamentos em conjuntos de dados.

- Aprendizado semi-supervisionado (ASS): Une um pouco de ambos AS e ANS, no qual o algoritmo aprende tanto de dados supervisionados como de não supervisionados. Têm como maior característica o uso de algoritmos híbridos, que tem como atributo o máximo uso de medidas de qualidade e recursos para correção de erro.

Segundo Mitchell (1997), um entendimento detalhado dos algoritmos de aprendizado de máquina pode levar também a um melhor entendimento da capacidade (e incapacidade) do aprendizado humano. A praticidade do AM e sua interdisciplinaridade em áreas como estatística, inteligência artificial, teoria da informação, dito em Mitchell (1997), proporciona sua aplicação para diversas atividades, visando uma maior experiência dos programas para que quando deparado com atividades possa tomar decisões práticas para contornar problemas presentes em áreas como inteligência articial, probabilidade e estatística, complexidade computacional, teoria da informação, psicologia, neurociências, filosofia e etc.

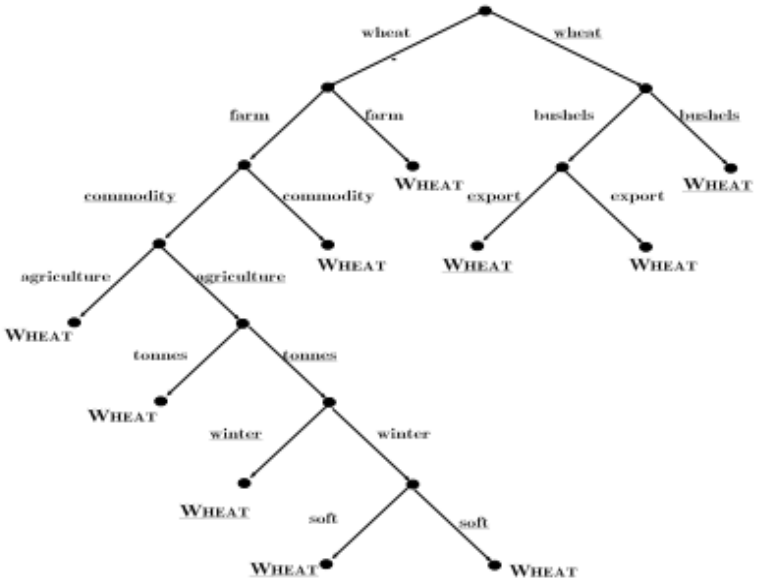
2.2 Algoritmos de classificação

Algoritmos de classificação são técnicas voltadas para o aprendizado supervisionado que, segundo Camilo e Silva (2009) são usados para prever valores de variáveis do tipo categóricas.

2.2.1 Árvores de Decisão (*Decision Trees*)

A árvore de decisão tem o seu funcionamento próximo ao de um fluxograma em formato de árvore, assim como mostra a figura 1.

Figura 1 – Exemplo de árvore de decisão.



Fonte: (SEBASTIANI, 2001).

Cada nó é a indicação de um teste feito sobre determinado valor. Kotsiantis (2007) fala que cada nó em uma árvore de decisão representa uma *feature* de uma instância a ser

classificada e cada ramificação representa um valor que o nó pode assumir. Os nós folhas representam a classe no qual a instância pertence.

A estrutura da árvore de decisão tem um funcionamento, que com algumas adaptações, pode ser transformadas em regras de classificação. Além disso, como é dito em Hall (2000), o resultado da árvore é uma representação compacta e de fácil interpretação do conceito de destino (classe). O uso dessa técnica é bastante popular devido a sua forma simples de ser aplicada e de sua grande porcentagem de assertividade, mas tem uma grande dependência de uma análise minuciosa para que possa atingir bons resultados.

2.2.1.1 CART

CART é um algoritmo de classificação elaborado por Breiman et al (1984) que se tornou uma referência para a aprendizagem de máquina. Têm como principal característica a possibilidade de verificar a relação entre os dados, ainda que não seja perceptível. Ao final do processo gera árvores do tipo binária (sim ou não). O seu processo é conduzido de forma automatizada, tendo interferência humana em momentos específicos.

Na etapa de crescimento, o critério utilizado para efetuar a divisão dos valores de um atributo em nós filhos, é a função de impuridade. Uma das funções de impuridade usadas é o Índice Gini, que é uma medida que o algoritmo usa para medir a pureza do nó. Ao ser aplicado no problema de maximização da fase de crescimento, o Gini é dado pela formula descrita na figura 2,

Figura 2 – Índice de gini aplicado no CART.

$$\Delta i(t) = -\sum_{k=1}^{K} p^2(k|t_p) + P_l \sum_{k=1}^{K} p^2(k|t_l) + P_r \sum_{k=1}^{K} p^2(k|t_r)$$

Fonte: (TIMOFEEV, 2004).

no qual *k, l, K* representa o índice da classe e p(k | t) indica a probabilidade condicional da classe k desde que esteja no nó t. Se for zero, então o nó é puro, senão quanto mais próximo está de um, mais impuro o nó é. O Gini busca pela classe maior, e então isola-a do resto dos dados.

Por meio desse processo de divisão, o algoritmo faz a árvore crescer o máximo possível até que não seja mais possível dividir os nós, ou seja, até que o índice de pureza do nó seja o mais próximo de zero (nó puro).

Após o crescimento da árvore é realizado a poda para melhorar o resultado da classificação. Ele é feito por meio da medida de custo complexidade descrita na figura 3.

$R_\alpha(T)$ é o custo da amostra de treinamento da árvore, |T| é o número de nós terminais na árvore e α é uma penalidade imposta a cada nó. Caso α seja zero, o custo de complexidade mínimo da árvore é a maior possível. A partir do nó raiz é feito o processo de

Figura 3 – Medida custo complexidade.

$$R_\alpha(T) = R(T) + \alpha(\tilde{T}) \longrightarrow \min_T$$

Fonte: (TIMOFEEV, 2004).

poda para encontrar subárvores candidatas. Se o custo destas for menor que o da árvore original, é repetido o processo de poda em um outro caminho da árvore original, até que encontre a subárvore que trará melhor resultado. A poda serve para reduzir o tamanho da árvore por meio da remoção de seções menos importantes para a classificação.

2.2.1.2 C4.5

O C4.5 é um algoritmo de classificação para a construção de árvores de decisão. Foi elaborado por Quinlan (1993 apud QUINLAN, 1996) sendo ele uma extensão do ID3, que é um outro algoritmo de árvore de decisão proposta também por Quinlan (1986). Ele permite que seja gerado um conjunto de regras para classificação a partir de sua árvore. A divisão binária não é uma obrigatoriedade no C4.5. Os resultados dos experimentos feitos no estudo de (LIM; LOH; SHIH, 2000) mostram que o C4.5 possui uma boa combinação entre taxa de erro e velocidade.

O C4.5 é um algoritmo de divisão e conquista. Segundo Quinlan (1996) para efetuar a divisão em um conjunto de casos D, o C4.5 efetua um conjunto de testes e então é escolhido um para maximizar o *splitting criterion*. Os testes realizados são:

- $A = ?$ para um atributo discreto A, com um resultado para cada valor de A.
- $A \leq$ t para um atributo contínuo A, com dois resultados, verdadeiro e falso. Para encontrar o limiar t que maximiza o critério de divisão, os casos em D são classificados a partir de seus valores do atributo A para fornecer valores distintos ordenados $v_1, v_2, ..., v_N$. Cada par de valores adjacentes sugere um limiar potencial $t = (v_i + v_i + 1)/2$ e uma partição correspondente de D. O limite que produz o melhor valor do critério de divisão é então selecionado.

O *splitting criterion* usado pelo algoritmo é o *gain ratio*, que é uma medida baseada em informação. Antes, é necessário obter a informação da incerteza residual sobre a classe à qual um caso em D pertence, como é definida na figura 4.

Figura 4 – Fórmula de info(D).

$$Info(D) = -\sum_{j=1}^{C} p(D,j) \times \log_2(p(D,j))$$

Fonte: (QUINLAN, 1996).

C denota o número de classes, *p(D, j)* a proporção de casos em D que pertencem à classe *jth*. As informações correspondentes obtidas por um teste T com k resultados é mostrado na fórmula da figura 5.

Figura 5 – Fórmula de *gain*.

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

Fonte: (QUINLAN, 1996).

Com base na informação obtida, que é muito afetada pelo número de resultados, a informação que retorna da função do *split*, descrita na figura 6, tende a aumentar com o número de resultados de um teste.

Figura 6 – Fórmula *split*.

$$Split(D,T) = -\sum_{i=1}^{k} \frac{|D_i|}{|D|} \times \log_2\left(\frac{|D_i|}{|D|}\right)$$

Fonte: (QUINLAN, 1996).

O critério *gain ratio* avalia a conveniência do teste a partir da razão entre o ganho de informação e a informação dividida. A taxa de ganho de todos os testes possíveis é determinada, e entre aqueles com pelo menos um ganho médio, a divisão com taxa máxima de ganho é selecionada.

Quinlan (1993 apud QUINLAN, 1996) em seu livro diz que a performance do C4.5 acaba sendo menor ao ser executado em domínios que possuem maior quantidade de atributos contínuos tendo preferência a atributos discretos.

Segundo Mitchell (1997), o C4.5 calcula sua estimativa pessimista calculando a precisão da regra sobre os exemplos de treinamento aos quais ela se aplica, calculando o desvio padrão nessa precisão estimada assumindo uma distribuição binomial.

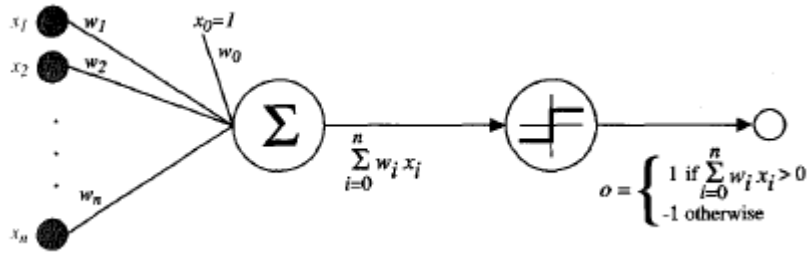
2.2.2 Rede neural (*Neural Networks*)

Rede neural é um modelo de aprendizado de máquina, que se assemelha ao cérebro humano por ser baseado em sua estrutura e funcionamento. Esse algoritmo implementa uma rede neural artificial, permitindo o aprendizado após a inserção de novos dados.

Uma rede neural pode ser representada com entradas e saídas contendo ligações, pesadas inicialmente de forma aleatória, e são intermediadas com camadas que possuem uma determinada quantidade de neurônios em cada uma delas. Os pesos são ajustados durante o aprendizado para poder classificar de forma correta o objeto.

Esta técnica tem uma necessidade de treinamento complexo, pois o período de treinamento é muito longo. A vantagem de usar a rede neural é que ela trabalha para não ter valores errados e conseguir resolver padrões que nunca foram apresentados ou treinados.

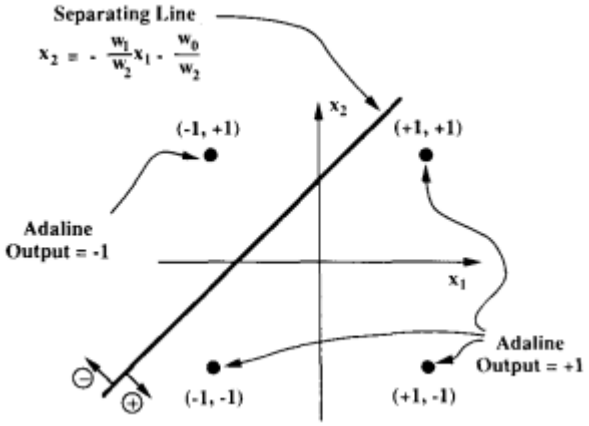
Figura 7 – Representação de um *perceptron*.



Fonte: (MITCHELL, 1997).

Como é representado na figura 7, um *perceptron* é um tipo de aplicação de rede neural e o seu funcionamento é descrito por Mitchell (1997) como uma função que a partir de um vetor de entradas de valor real calcula uma combinação linear delas. Caso o resultado for maior que o limiar, então gera o valor 1, se não gera o valor -1.

Figura 8 – Exemplo de um problema linearmente separável.



Fonte: (WIDROW; LEHR, 1990).

Kotsiantis (2007) diz que os *perceptrons* podem apenas classificar conjuntos de instâncias linearmente separáveis, como na figura 8. Caso uma linha reta ou plano puder ser desenhado para separar as instâncias de entrada em suas categorias corretas, Então elas serão linearmente separáveis e o *perceptrons* encontrará a solução. Se as instâncias não forem linearmente separáveis, o aprendizado nunca chegará a um ponto, em que todas as instâncias sejam classificadas corretamente.

2.2.3 *K Nearest neighbor*(KNN)

KNN é um método de classificação baseado em instâncias. Ele fornece o rótulo que é mais frequentemente usado entre as amostras próximas utilizando um esquema de votação. Ele define quais amostras estão mais próximas pelo cálculo de distância entre as instâncias de teste e treinamento. Assim como Mitchell (1997) diz em seu livro, o algoritmo é adaptado para aproximar as funções de destino de valor contínuo, então calcula o valor médio dos k exemplos de treinamento mais próximos, em vez de calcular seu valor mais comum.

Figura 9 – Fórmula da distância Euclidiana.

$$\text{Euclidean: } D(x,y)=\left(\sum_{i=1}^{m}\left|x_i-y_i\right|^2\right)^{1/2}$$

Fonte: (KOTSIANTIS, 2007).

As fórmulas mais comumente usadas são as distâncias euclidiana, como é mostrado na figura 9, e *manhattan*. Kotsiantis (2007) diz que o KNN localiza as k instâncias mais próximas da instância de consulta e determina sua classe identificando o rótulo de classe mais frequente.

Uma desvantagem deste algoritmo é o seu consumo computacional, já que para cada instância nova (sem classe) é aplicado a fórmula de distância em cada uma das instâncias de treinamento. Uma outra desvantagem é que o KNN tem dificuldades para classificar uma instância nova, quando o conjunto de dados possui um número grande de variáveis.

2.2.4 *Support Vector Machines*(SVM)

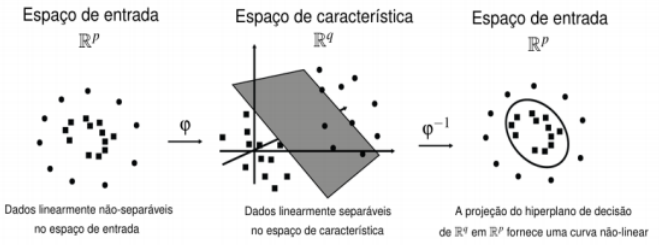
Apesar do SVM ser uma técnica relativamente nova, têm demonstrado altos níveis de assertividade trabalhando com modelos não lineares complexos, enquanto concebe modelos de simples interpretação, além da facilidade para a utilização de modelos lineares e não lineares. Segundo Hsu, Chang e Lin (2003), o objetivo deste algoritmo é produzir

um modelo a partir dos dados de treinamento, que prevê os valores alvo dos dados de teste por meio apenas de seus atributos.

O SVM é baseado no princípio de Minimização de Risco Estrutural que tem como ideia encontrar uma hipótese h para que possa garantir o menor erro verdadeiro. Noble (2006) em seu artigo diz que para entender a essência da classificação SVM não é necessário ler uma equação, basta compreender quatro conceitos básicos: a separação do hiperplano, o hiperplano de margem máxima, a margem flexível e a função do *kernel*.

A separação do hiperplano é uma linha que separa uma amostra de outra, ou seja, é basicamente uma linha reta em um espaço de alta dimensão, como mostra a figura 10. No hiperplano de margem máxima, o seu objetivo é identificar uma linha que separe uma amostra de outra neste espaço bidimensional, para isso o SVM seleciona a margem máxima que separa o hiperplano. Isso permite a maximização da capacidade de predição de uma instância nova.

Figura 10 – Exemplo de classes linearmente separáveis.



Fonte: (PADULA et al., 1999).

O *soft margin* é usado para lidar com os erros nos dados. Ele permite que alguns pontos de dados percorram a margem do hiperplano de separação sem afetar o resultado final.

Figura 11 – Funções *kernel*.

Kernel	Equation
Linear	$K(x,y) = x \cdot y$
Sigmoid	$K(x,y) = \tanh(ax.y + b)$
Polynomial	$K(x,y) = (1 + x \cdot y)^d$
KMOD	$K(x,y) = a\left[\exp\left(\frac{\gamma}{\|x-y\|^2+\sigma^2}\right) - 1\right]$
RBF	$K(x,y) = \exp(-a\|x-y\|^2)$
Exponential RBF	$K(x,y) = \exp(-a\|x-y\|)$

Fonte: (CHOU, 2010).

A função *kernel* serve para evitar um problema, quando o *soft margin* não é suficiente para separar duas amostras, com a inserção de uma dimensão extra. Uma boa classificação

depende da escolha da função *kernel*. Algumas dessas funções são mostradas na figura 11.

2.2.5 *Random Forest*

O *Random forest* é um algoritmo que utiliza classificadores do tipo árvore por meio da técnica *ensemble*. Essa técnica é formada por um conjunto de classificadores que são treinados de forma individual, e após sua finalização suas decisões são combinadas. Segundo Sirikulviriya and Sinthupinyo (2011) o uso de vários classificadores com o ensemble resulta em uma maior acurácia do que o uso de apenas um classificador dado o mesmo conjunto de dados.

Segundo Breiman (2001) há duas razões para usar o *bagging* no *random forest*. A primeira é para aumentar a precisão quando se usa *features* aleatórias e a segunda é que ele provê estimativas contínuas do erro de generalização do *ensemble* de árvores, bem como estimativas para a resistência e correlação. *Bagging* é definido por Breiman (1996) como um método para gerar várias versões de um preditor e usá-los para obter um preditor agregado.

Sirikulviriya and Sinthupinyo (2011) diz que o *Random Forests* classifica a instância simplesmente combinando todos os resultados de cada uma das árvores na floresta. Esses resultados, segundo Lopes et al. (2017) são obtidos por meio de um sistema de votação utilizando os componentes preditores presentes nas classes escolhendo a que teve o maior número de votos recebidos.

O algoritmo faz a classificação de uma instância nova, com base em um conjunto de treinamento, por meio da função de margem, descrita na figura 12.

Figura 12 – Função de margem.

$$mg(\mathbf{X}, Y) = av_k I(h_k(\mathbf{X}) = Y) - \max_{j \neq Y} av_k I(h_k(\mathbf{X}) = j)$$

Fonte: (BREIMAN, 2001).

h_k representa um conjunto de classificadores, X,Y a distribuição do vetor aleatório pelo qual o conjunto de treino é sorteado aleatoriamente e I representa a função do indicador. A função de margem serve para indicar o quanto que a média de votos para a classe correta supera os votos para as outras classes gerando maior confiança na classificação.

Segundo Breiman (2001) para um grande número de árvores, o algoritmo segue a Lei Forte dos Grandes Números que resulta no teorema, mostrado na figura 13, que diz "*As the number of trees increases, for almost surely all sequences 1,..., PE* converges to:*[1]

[1] Em tradução livre: À medida que o número de árvores aumenta, para quase com certeza todas as sequências 1, ..., *PE** converge para

Figura 13 – Fórmula teorema do Random Forest.

$$P_{\mathbf{X},Y}(P_{\Theta}(h(\mathbf{X},\Theta)=Y)-\max_{j\neq Y}P_{\Theta}(h(\mathbf{X},\Theta)=j)<0)$$

Fonte: (BREIMAN, 2001).

*PE** representa a generalização de erro que é definida pela fórmula mostrada na figura 14.

Figura 14 – Fórmula generalização de erro.

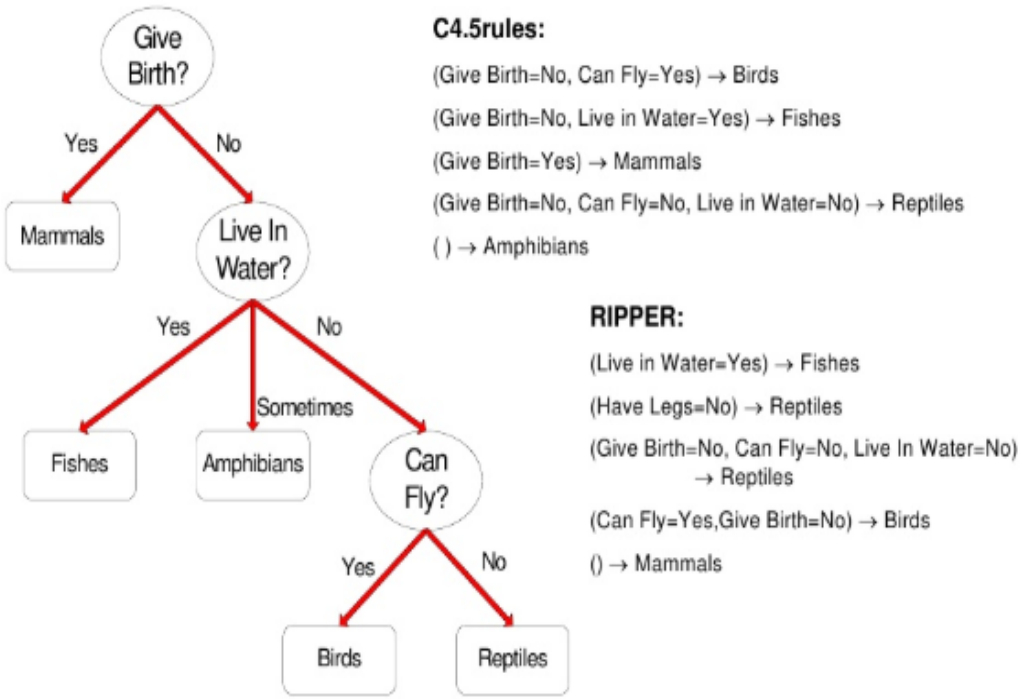
$$PE^{*}=P_{\mathbf{X},Y}(mg(\mathbf{X},Y)<0)$$

Fonte: (BREIMAN, 2001).

2.2.6 RIPPER

O algoritmo Ripper foi criado por Cohen (1995), após serem realizadas modificações a partir de uma análise do algoritmo *incremental reduced error pruning* (IREP). As regras do algoritmo são formadas a partir da repetição dos processos de crescimento e poda. A figura 15 mostra a comparação entre uma árvore de decisão(C4.5), as regras extraídas do C4.5 e as regras do Ripper.

Figura 15 – Exemplo de regras do Ripper.



Fonte: (LANZI, 2007).

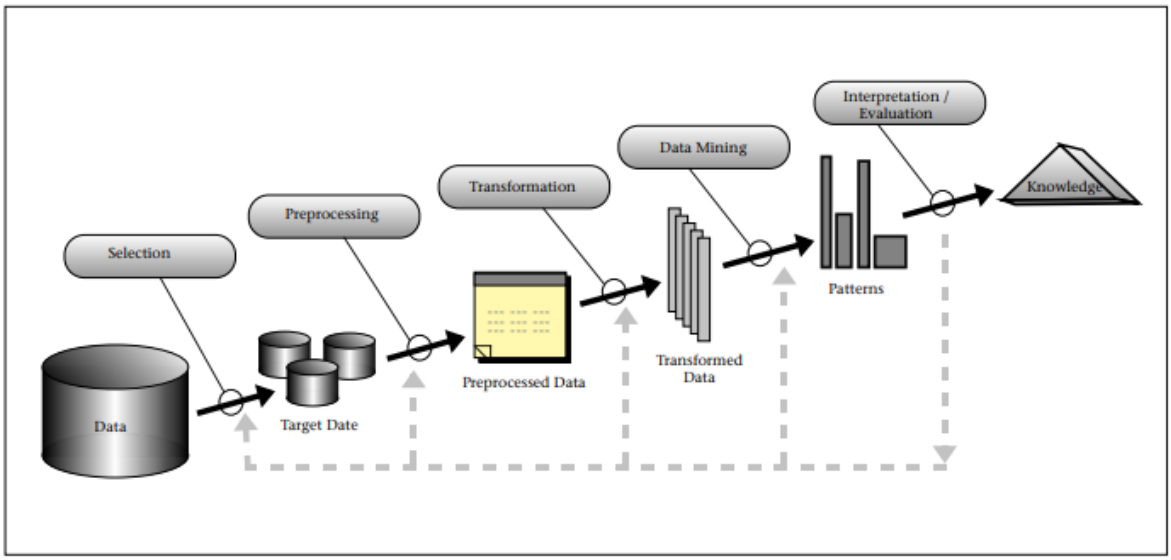
Kotsiantis (2007) diz que na fase de crescimento é necessário ajustar os dados de treinamento por meio da limitação das regras, enquanto que na fase de poda essa limitação ajuda a evitar o *overfitting*. *Overfitting* é quando um classificador ajusta-se tão bem aos dados de treinamento que acaba tornando-se ineficaz em classificar uma instância nova, diminuindo o desempenho do classificador.

Uma modificação feita foi o desenvolvimento de um novo método para a otimização de um conjunto de regra. Para isso Cohen (1995) diz que cada regra é considerada uma por vez: primeiro R_1 depois R_2 na ordem em que foram aprendidas. Para cada regra R_i, duas regras alternativas são construídas. A substituição para R_i é feita pelo crescimento, e em seguida, pela remoção da Regra R'_i, no qual a poda é guiada para a minimização do erro do conjunto de regras R_1, ..., R'_i, ..., R_k nos dados de poda.

2.3 Descoberta de conhecimento

O KDD consiste num conjunto de tarefas consecutivas, como é mostrado na figura 16, no qual a partir de uma base de dados é gerado "conhecimento". Para Fayyad, Piatetsky-Shapiro e Smyth (1996), essas tarefas são compostas por nove etapas: identificação do objetivo do KDD; seleção dos dados; pré-processamento e limpeza dos dados; transformação dos dados; especificação método de mineração de dados; análise exploratória e a seleção de modelos; mineração de Dados (*Data Mining*), interpretação e avaliação dos resultados e aplicação do conhecimento.

Figura 16 – Exemplificação das fases do processo KDD.



Fonte: (FAYYAD; PIATETSKY-SHAPIRO; SMYTH,1996).

Com o avanço das tecnologias as bases de dados tem volumes absurdos, gerando uma enormes dificuldades de interpretá-los de forma manual, constituindo assim a necessi-

dade de automatizar este processo de forma inteligente. Estas etapas tem como objetivo transformar os dados armazenados em conhecimento. As fases do KDD são:

1. Identificação do objetivo do KDD

   Segundo Fayyad; Piatetsky-Shapiro e Smyth (1996), a primeira fase é voltada para entender o domínio da aplicação e o conhecimento prévio relevante, para assim identificar o objetivo do processo do KDD por meio do ponto de vista do cliente.

2. Seleção dos dados

   Na seleção de dados são apresentados os dados que irão sofrer uma seleção a fim de obter resultados mais relevantes e úteis para todo o processo.

3. Pré-processamento e Limpeza dos dados

   A etapa de pré-processamento funciona como uma espécie de faxina para os dados, nela retira-se o que é irrelevante para o processo e seleciona os atributos importantes, o que é mais comum de se verificar são informações ausentes, erradas ou inconsistentes.

4. Transformação dos dados

   Na transformação ou formatação, os dados gerados pela fase anterior são rearrumados de forma que sejam interpretados facilmente na fase seguinte.

5. Especificação método de mineração de dados

   Fayyad, Piatetsky-Shapiro e Smyth (1996) dizem que essa fase é destinada para a combinação dos objetivos do processo de KDD, definidos na primeira fase, com um método específico de mineração de dados, como por exemplo sumarização e classificação.

6. Análise exploratória e a seleção de modelos

   A sexta é para a análise exploratória e a seleção de modelos e hipóteses, no qual é escolhido o algoritmo de mineração de dados e selecionado o método a ser utilizado para pesquisar padrões de dados.

   Esse processo inclui decidir quais modelos e parâmetros podem ser apropriados e combinar um método de mineração de dados específico com os critérios gerais do processo KDD.

7. Mineração de dados

   O termo mineração de dados tem uma certa interdisciplinaridade que varia em cada área de atuação, mas apesar de diversas definições o mesmo tem como objetivo a extração de conhecimento de forma automática, com o auxílio de diversas ferramentas para a execução de algoritmos de mineração, mas com necessidade de apoio

humano. Mesmo com esta limitação, a mineração oferece uma contribuição considerável gerando a possibilidade dos analistas darem prioridade ao trabalho mais significativo.

Fayyad, Piatetsky-Shapiro e Smyth(1996) dizem que a mineração de dados tem métodos que foram testados e experimentados. São eles:

- Descrição: É o trabalho utilizado para apresentar e explicar inclinações a padrões demonstrados pelos dados. A descrição utiliza-se de suas características para apresentar possíveis interpretações aos resultados gerados. Comumente usada com o apoio de diversas técnicas da área de análise exploratória a fim de verificar a atuação de variáveis nos resultados apresentados.
- Classificação: Tem como objetivo a identificação de cada registro e determinar a qual classe o mesmo pertence. Utilizando o modelo de aprendizagem supervisionada, ocorrerá a análise dos registros disponibilizados. Após a finalização da análise e com todos os registro classificados pode-se dizer que a máquina adquiriu um determinado "conhecimento".
- Estimação ou Regressão: A estimação detém uma certa similaridade com que é apresentado na classificação, mas com a diferença de ser usada quando o registro é apresentado em valor numérico e não categórico, possibilitando assim predição de um valor de uma determinada variável utilizando o conhecimento das demais.
- Predição: A atividade de predição tem como objetivo, por meios de análises, tentar descobrir qual será o valor futuro que o atributo poderá chegar. É possível que alguns métodos de classificação e regressão possam ser utilizados para gerar uma predição.
- Agrupamento: O agrupamento tem como objetivo a aproximação de registros que são semelhantes. Um *cluster*, que é a forma de denominação de uma coleção de registros que são similares, faz a junção dos registros equivalentes e os afasta das demais coleções. Esta tarefa não é igual a classificação, pois não se tem a necessidade de categorização prévia dos registros, ou seja, segue o modelo de aprendizagem não supervisionada. Além de não ter como objetivo a classificação, predição ou estimativa de valor de nenhum tipo de registro ou variável.
- Associação: A associação tem como objetivo verificar quais propriedades tem relação com outros atributos. Um exemplo mais comum é a cesta de compras, onde é identificado quantas vezes uma combinação de produtos é levada pelo consumidor final.

8. Interpretação e avaliação dos resultados

A última etapa do KDD consiste na interpretação das regras e na avaliação das mesmas. Após o processo de interpretação terá a possibilidade do surgimento de padrões, relacionados à informação dos dados, que poderão ser aproveitadas para pesquisas, otimização dentre outras possibilidades.

9. Aplicação do conhecimento

   A última fase é para a atuação sobre o conhecimento que foi descoberto, ou seja, a aplicação desse conhecimento, seja de forma direta ou em um sistema. Além disso, é possível fazer a resolução de conflitos entre o conhecimento recém extraído e os que já foram extraídos.

2.4 Classificação de texto

Categorização de texto é uma técnica de descoberta de conhecimento, que tem como objetivo o estudo de métodos que auxilie a manipulação de textos de forma eficiente fazendo a classificação de documentos sobre um conjunto de categorias predefinidas. Segundo Sebastiani (2001), a mineração de texto é utilizada mais para indicar tarefas, que a partir da análise de grandes quantidades de texto e da detecção de padrões, extrai algumas informações úteis.

Sebastiani (2001) diz que a categorização de texto atribui um valor booleano a cada par $\langle d_j, c_i \rangle$ D x C, no qual D é um domínio de documentos e $C = c_1, ..., c_{|C|}$ é um conjunto de categorias (ou rótulo, classe) predefinidas. Um valor de T designado para $\langle d_j, c_i \rangle$ indica uma decisão de arquivar d_j em c_i, enquanto que um valor F indica uma decisão de não arquivar d_j em c_i. Ele busca aproximar a função alvo desconhecida $\overline{\phi} : D\ x\ C \rightarrow \{T, F\}$, que descreve como os documentos devem ser classificados, por meio da função $\phi : D\ x\ C \rightarrow \{T, F\}$ chamado de classificador (ou modelo), tal que $\overline{\phi}$ e ϕ "coincidem tanto quanto possível".

Antes de classificar um texto é necessário mapear o conteúdo dele, por meio de um processo de indexação, para aplicar a um documento de treinamento, validação ou teste. A representação de um texto dependerá da forma como ele será separado em unidades significativas (semântica lexical) e das regras da linguagem natural necessárias para realizar a combinação dessas unidades (semântica composicional).

Na classificação de texto a semântica composicional é, frequentemente, desconsiderada. Desta forma, um texto é normalmente apresentado como um vetor de pesos de termo d_j= $\langle w_{1j}, \ldots,$ w | T | j$\rangle$, onde T é o conjunto de termos (às vezes chamados de *features*) que ocorrem pelo menos uma vez em pelo menos um documento e $0 \leq w_{kj} \leq 1$ mostra o quanto que o termo w_{kj} contribuiu para a semântica do documento.

Há diferentes abordagens com maneiras diferentes de entender o que é um termo e maneiras diferentes de calcular pesos de um termo. Normalmente para o primeiro caso

é utilizado a identificação de termos por palavras. Isso é chamado de *bag of words*. Um problema que a classificação de texto enfrenta é a alta dimensionalidade do espaço de termos gerado pela grande quantidade de termos que precisam ser processados.

2.5 Trabalhos Correlatos

Em relação aos trabalhos correlatos, Tripathy; Agrawal; Rath (2016) usou o modelo n-gram para realizar a classificação de textos com intuito de identificar sentimentos com base nas análises de filmes do site IMDb separando-as nas classes: positivo, negativo e neutro. O dataset utilizado, proveniente do banco de dados do IMDb (2011), é composto por 12.500 revisões de teste e de treino positivamente rotuladas. Da mesma forma, existem 12.500 revisões de testes e de treino com rótulos negativos. Entre os algoritmos que foram utilizados inclui o Naive Bayes (NB), Maximum entropy (ME), Stochastic gradient descent (SGD) e Support Vector Machine (SVM), desses só o SVM foi utilizado para a experimentação proposta no trabalho de tipificação de ocorrências policiais. Além disso, utilizaram como parâmetro de desempenho a precisão, recall, f-measure e acurácia. Para o trabalho de tipificar ocorrências policiais a única diferença na utilização da classificação de texto é o objetivo da classificação.

Já Kowsari; Brown; Heidarysafa (2017) apresentam uma nova abordagem de classificação de texto, que foi chamada de Hierarchical Deep Learning for Text classification (HDLTex), no qual emprega pilhas de arquiteturas de deep learning para lidar tanto com o aumento do número de documentos, quanto com o aumento do número de classes. Para a base da implementação, eles utilizaram algoritmos de aprendizado profundo como o DNN (Deep Neural Networks), CNN (Convolutional Neural Networks) e o RNN (Recurrent Neural Network). Nenhum algoritmo apresentado nesse trabalho foi utilizado para a tipificação de ocorrência policial. Este trabalho foi utilizado para adquirir conhecimento e experiência para a execução da classificação de texto em ocorrências policiais.

Xu; Yang (2017) propõem uma nova abordagem para a classificação de texto multilíngue utilizando o CLTC. Ele é baseado em corpus paralelo e foca na redução das correspondências domínio / distribuição. Para isso, eles realizaram experimentos em dois conjuntos de dados CLTC de referência, tratando o inglês como idioma de origem e alemão, francês, japonês e chinês como os idiomas de destino não rotulados. O algoritmo de classificação utilizado foi o CNN. Este trabalho foi estudado com o intuito de entender melhor a utilização da classificação de texto para executa-lo de forma precisa.

3 Metodologia

No desenvolvimento deste projeto foi usado o ambiente computacional estatístico R versão 3.5.1 em conjunto com a IDE Rstudio versão 1.2.1335 em uma máquina com 8 cpus e 52 GB de memória RAM. Para realizar a predição dos dados foram escolhidos os algoritmos mais indicados e mais comumente usados na classificação de texto, segundo Sebastiani (2001): C4.5, *K-nearest neighbors* (KNN), *Support Vector Machine* (SVM), Rede Neural, *Repeated Incremental Pruning to Produce Error Reduction* (Ripper), *Naive Bayes*, *iterative Dichotomiser 3* (ID3) e foram escolhidos os seguintes algoritmos a partir de experiência pessoal: *Classification and regression trees* (CART), *Random Forest*, Regressão logística. Na implementação dos algoritmos foram aplicados os seguintes pacotes:

- quanteda (versão 1.4.3): Usado para implementar a classificação de texto.
- rpart (versão 4.1-13): Usado para implementar o algoritmo CART.
- e1071 (versão 1.7-0): Usado para implementar o algoritmo SVM.
- *class* (versão 7.3-14): Usado para implementar o algoritmo KNN.
- nnet (versão 7.3-12): Usado para implementar o algoritmo Rede neural.
- randomForest (versão 4.6-14): Usado para implementar o algoritmo Random forest.
- RWeka (versão 0.4-40): Usado para implementar os algoritmos Ripper e C4.5.

O Conjunto de dados é voltado para ocorrências policiais. Ele é formado por 465.376 exemplos e 17 atributos, sendo eles nomeBairro, Sexo, dataNascimento, ocupacao, grauInstrucao, estadoCivil, religiao, cutis, orientacaoSexual, motivação, anoOcorrencia, nomeMunicipio, natureza, contadorVitima, contadorAutor, contadorComunicante e historicoFato nessa ordem, dividindo-se em categóricos e numéricos. A classe desta base é o atributo natureza que contém 381 valores distintos relacionados aos tipos de crimes ou infrações penais cometidos. Como o atributo historicoFato contém uma descrição de cada ocorrência, como é mostrado na tabela 1, foi necessário a execução de um algoritmo de classificação de texto, mas devido a limitação da máquina usada, não foi possível utilizar o conjunto de dados inteiro por causa da quantidade de variáveis geradas após a execução da classificação de texto, pois o Rstudio não conseguia alocar na memória o *dataset* resultante. Foi então necessário utilizar amostras com uma quantidade reduzida de exemplos.

Tabela 1 – Exemplo de texto do atributo historicoFato.

INFORMA O COMUNICANTE QUE, EM HORÁRIO, DATA E LOCAL JÁ CITADOS, ENCONTRAVA-SE OCUPANDO O COLETIVO DA EMPRESA BOA VIAGEM, QUE PERCORRIA A LINHA ALTO DE COUTOS / PITUBA.
ALEGA, O COMUNICANTE, QUE, AGUARDAVA, NO PONTO, O TRANSPORTE COLETIVO, QUANDO, DE REPENTE, FOI ABORDADO, POR UM ELEMENTO, ARMADO DE REVÓLVER.
ALEGA, A COMUNICANTE, QUE FOI VÍTIMA DE AGRESSÃO COM MURROS POR PARTE DE SEU EX COMPANHEIRO DE QUEM JÁ ESTA SEPARADA HÁ 08 (OITO) MESES E O MESMO NÃO ACEITA.

Fonte: Compilação do autor.

Foram utilizadas amostras com instâncias selecionadas do *dataset* original tanto de forma sequencial (as primeiras instâncias sequencialmente) como de forma aleatória. Elas tinham cerca de 1800 instâncias, com os 16 primeiros atributos e com a classificação de texto do historicoFato, que gerou um adicional de 8749 atributos para a amostra selecionada de forma sequencial e, em média, 14000 para a amostra aleatória. A quantidade de classes nas amostras variavam de 68 a 96 classes distintas e a tabela 2 mostra exemplos de algumas delas que foram selecionadas.

Tabela 2 – Algumas da classes contidas nas amostras.

FURTO QUALIFICADO	HOMICIDIO
VIOLACAO DE DOMICILIO	DIFAMACAO
FURTO DE VEICULO	INJURIA
HOMICIDIO QUALIFICADO	DANO QUALIFICADO
EXTORSAO "SEQUESTRO RELAMPAGO"	APROPRIACAO DE COISA ACHADA
ABANDONO DE INCAPAZ	USO DE DOCUMENTO FALSO
RECEPTACAO	CORRUPCAO ATIVA
ROUBO	LESAO CORPORAL CULPOSA

Fonte: Compilação do autor.

Para a validação do modelo é utilizado a técnica *Hold Out*, no qual o *dataset* é dividido numa proporção de 70% para o treinamento e 30% para testes, para que a predição não gere resultados falsos (*Overfitting*). A métrica utilizada na avaliação dos resultados foi a acurácia,pois indica a taxa de acerto da predição em relação ao valor verdadeiro.

3.1 Pré-Processamento

No pré-processamento do *dataset*, inicialmente foi realizado a classificação de texto do atributo historicoFato. Nele, após dividir os termos por palavras, foi feito o tratamento de dados com a remoção de pontuação, números, símbolos, separadores e hífens. Além disso,

foi removido também as *stopwords* (palavras de parada) relacionadas a língua portuguesa. Em seguida foi realizado o tratamento dos NAs (missing values/valores ausentes), como é demostrado na figura 17.

Figura 17 – Parte do *dataset* sem pré-processamento.

	nomeBairro	SEXO	dataNascimento	ocupacao	grauInstrucao	estadoCivil
1	BARRA	MASCULINO	1935-02-06 00:00:00.000	COMERCIARIO	3 grau incompleto	CASADO
2	PITUACU	MASCULINO	1981-11-08 00:00:00.000	VENDEDOR	1 grau completo	SOLTEIRO
3	LOBATO	MASCULINO	NA	POLICIA CIVIL	2 grau completo	SOLTEIRO
4	SAO CRISTOVAO AEROPORTO	MASCULINO	1989-03-23 00:00:00.000	IGNORADA	Alfabetizado	SOLTEIRO
5	SAO CRISTOVAO AEROPORTO	MASCULINO	1988-03-31 00:00:00.000	IGNORADA	Alfabetizado	SOLTEIRO
6	COSME DE FARIAS	MASCULINO	1981-08-01 00:00:00.000	OUTRAS	2 grau incompleto	SOLTEIRO
7	Itapua	FEMININO	1952-08-03 00:00:00.000	ARTESAO	2 grau completo	CASADO
8	Pituba	FEMININO	1989-09-15 00:00:00.000	ESTUDANTE	2 grau completo	SOLTEIRO
9	pituba	MASCULINO	1969-05-06 00:00:00.000	COMERCIANTE	3 grau incompleto	CASADO
10	AMARALINA	FEMININO	NA	NA	NA	NA
11	NA	MASCULINO	NA	NA	NA	NA
12	PITUBA	MASCULINO	NA	NA	NA	NA
13	ITAIGARA	MASCULINO	NA	NA	NA	NA
14	CAMINHO DAS ARVORES	MASCULINO	NA	NA	NA	NA
15	PITUBA	MASCULINO	NA	NA	NA	NA
16	PITUBA	MASCULINO	NA	NA	NA	NA
17	PITUBA	FEMININO	1972-10-06 00:00:00.000	ADVOGADO	3 grau completo	DESQUITADO
18	PITUBA	MASCULINO	1969-07-15 00:00:00.000	NA	3 grau completo	SEPARADO
19	Itapua	MASCULINO	1979-06-23 00:00:00.000	AUXILIAR OPERADOR	2 grau completo	SOLTEIRO
20	Pituba	MASCULINO	1975-03-27 00:00:00.000	FUNCIONARIO PUB MUNICIPAL	NA	SOLTEIRO

Fonte: Compilação do autor.

Como é mostrado na figura 18, já que os NAs estão localizados em atributos categóricos, eles foram substituídos pela moda de cada atributo a qual pertencem. Em seguida, para padronizar os dados, foi removido os acentos das palavras e elas foram colocadas em caixa baixa.

Figura 18 – Parte do *dataset* com pré-processamento.

	nomeBairro	SEXO	dataNascimento	ocupacao	grauInstrucao	estadoCivil
1	jardim cruzeiro	masculino	1986-09-20 00:00:00.000	outras	2º grau completo	solteiro
2	nesta	masculino	1992-01-13 00:00:00.000	ajudante de pedreiro	1º grau incompleto	solteiro
3	nesta	masculino	1974-10-25 00:00:00.000	ignorada	alfabetizado	solteiro
4	nesta	masculino	1989-03-30 00:00:00.000	ignorada	1º grau incompleto	solteiro
5	nesta	masculino	1993-12-19 00:00:00.000	ignorada	1º grau incompleto	solteiro
6	ondina	feminino	1964-08-16 00:00:00.000	estudante	1º grau incompleto	solteiro
7	nesta	masculino	1995-01-01 00:00:00.000	ignorada	alfabetizado	solteiro
8	caminho das arvores	feminino	1950-08-21 00:00:00.000	aposentado	3º grau completo	viuvo
9	ondina	feminino	1967-04-06 00:00:00.000	empregado domestico	1º grau incompleto	solteiro
10	jardim limoeiro	feminino	1988-02-12 00:00:00.000	outras	1º grau completo	amigado
11	jardim limoeiro	masculino	1964-08-16 00:00:00.000	vigilante	alfabetizado	solteiro
12	centro	masculino	1964-08-16 00:00:00.000	estudante	1º grau incompleto	solteiro
13	centro	masculino	1959-11-24 00:00:00.000	comerciante	1º grau incompleto	solteiro
14	boca da mata	feminino	1977-07-15 00:00:00.000	gerente de empresa	3º grau incompleto	casado
15	centro	masculino	1982-07-25 00:00:00.000	feirante	2º grau completo	solteiro
16	centro	masculino	1964-08-16 00:00:00.000	estudante	1º grau incompleto	solteiro
17	vale das pedrinhas	feminino	1968-01-05 00:00:00.000	auxiliar administrativo	2º grau completo	solteiro
18	amaralina	masculino	1962-05-28 00:00:00.000	funcionario pub estadual	pos-graduados	casado
19	aguas claras	masculino	1964-08-16 00:00:00.000	estudante	1º grau incompleto	solteiro
20	massaranduba	feminino	1964-08-16 00:00:00.000	do lar	alfabetizado	solteiro

Fonte: Compilação do autor.

Foi realizado a transformação do *dataset* para inteiramente numérico com o objetivo de utilizar em algoritmos que lidam apenas com dados numéricos como, por exemplo, o KNN.

3.2 Escolha dos algoritmos

Inicialmente, haviam sido escolhidos dez algoritmos sendo eles CART, C4.5, KNN, SVM, Rede neural, *Random Forest*, *Ripper*, ID3, Regressão logística e *Naive bayes*. Devido a classificação de texto gerar valores numéricos, referentes a frequência de cada termo que foi separado, não foi possível realizar a transformação dos dados para o tipo categórico, pois é mais relevante esses dados serem mantidos como numéricos. Consequentemente, é necessário remover os algoritmos que trabalham apenas com dados categóricos, sendo eles ID3, Regressão logística e *Naive bayes*.

4 Resultados

Nesta seção é apresentado os resultados dos testes aplicados, no qual os algoritmos de classificação foram aplicados em 10 amostras não necessariamente iguais em todos os testes, com exceção da amostra sequencial, pois foi igual nos testes 1 e 2. Devido a quantidade de classes e, consequentemente a demora de execução, o algoritmo CART foi usado apenas nos testes descritos nas subseções 4.3 e 4.8.

4.1 Teste 1

No primeiro teste é comparado as acurácias entre uma amostra com instâncias selecionadas de forma sequencial e outra de forma aleatória, sendo que em ambas foi executado a classificação de texto do atributo historicoFato.

Tabela 3 – Teste 1

ALGORITMO	Teste 1 - Amostra 1	Teste 1 - Amostra 2
C4.5	57,1164	26,2385
CART	-	-
KNN	0,3696	0,5504
SVM	21,9963	16,3302
RandomForest	22,9204	0,1834
Rede Neural	0,3696	0,1834
Ripper	58,2255	29,7247

Fonte: Compilação do autor.

Em ambas as amostras da tabela 2 o algoritmo *Ripper* foi o que obteve a melhor acurácia com 58% na amostra sequencial e 29% na amostra aleatória, seguido pelo algoritmo C4.5 com 57% e 26% respectivamente. O que obteve a pior acurácia foram os algoritmos KNN e rede neural com 0,36% na primeira amostra e na segunda amostra, além do KNN e rede neural, o *Random Forest* também obteve uma acurácia baixa com 0,18%.

4.2 Teste 2

No Segundo teste é feita a comparação das acurácias entre uma amostra com instâncias selecionadas de forma sequencial e outra de forma aleatória, sem a execução da classificação de texto em ambas as amostras.

Tabela 4 – Teste 2

ALGORITMO	Teste 2 - Amostra 1	Teste 2 - Amostra 2
C4.5	38,2624	20,7339
CART	-	-
KNN	0,3696	0,1834
SVM	26,0628	23,4862
RandomForest	19,5933	18,899
Rede Neural	1,109	3,1192
Ripper	32,1626	20,7339

Fonte: Compilação do autor.

Novamente, como é mostrado na tabela 3, os algoritmos C4.5 e *Ripper* obtiveram as melhores acurácias com 38% e 32% nessa ordem para a amostra sequencial e 20% em ambos os algoritmos para a amostra aleatória. Os algoritmos com acurácia baixa foram os mesmos do teste anterior e tiveram a mesma acurácia.

4.3 Teste 3

No terceiro teste a amostra é selecionada de forma aleatória com a execução da classificação de texto, sendo que em metade das instâncias o atributo natureza é igual a furto qualificado e a outra metade igual a roubo.

Tabela 5 – Teste 3

ALGORITMO	Teste 3
C4.5	89,4444
CART	82,9629
KNN	71,6666
SVM	46,6666
RandomForest	80,7407
Rede Neural	39,4444
Ripper	89,6296

Fonte: Compilação do autor.

No teste mostrado na tabela 4, os algoritmos com a melhor acurácia foram o C4.5 e o Ripper, ambos com 89%. Já os que obtiveram a pior acurácia foram o SVM e o Rede Neural com 46% e 39% nessa ordem.

4.4 Teste 4

No quarto teste a amostra é selecionada de forma aleatória com a execução da classificação de texto, sendo que foi mantido apenas a *bag of words* e o atributo classe(natureza).

Tabela 6 – Teste 4

ALGORITMO	Teste 4
C4.5	99,8165
CART	-
KNN	95,0458
SVM	18,899
RandomForest	0,3669
Rede Neural	0,1834
Ripper	99,8165

Fonte: Compilação do autor.

A tabela 5 mostra que os algoritmos com a maior acurácia foi o C4.5 e o *Ripper*, ambos com 99%. Os que obtiveram a pior acurácia foi o *Random Forest* e a Rede Neural com 0,36% e 0,18% respectivamente.

4.5 Teste 5

No quinto teste a amostra é selecionada de forma aleatória com a execução da classificação de texto, sendo que foram retiradas alguns atributos que são dataNascimento e anoOcorrencia.

Tabela 7 – Teste 5

ALGORITMO	Teste 5
C4.5	27,8899
CART	-
KNN	0,1834
SVM	20,3669
RandomForest	0,1834
Rede Neural	0,1834
Ripper	28,9908

Fonte: Compilação do autor.

Neste teste da tabela 6, os algoritmos com a maior acurácia foram o Ripper e o C4.5 com 28% e 27% nessa ordem. Já os que obtiveram a pior acurácia foram o KNN, *Random Forest* e Rede Neural, todos os três com 0,18%.

4.6 Teste 6

No sexto teste a amostra é selecionada de forma aleatória sem a execução da classificação de texto, sendo que foram retiradas os atributos dataNascimento e anoOcorrencia.

Tabela 8 – Teste 6

ALGORITMO	Teste 6
C4.5	22,0183
CART	-
KNN	0,3669
SVM	21,4678
RandomForest	16,6972
Rede Neural	1,4678
Ripper	20,1834

Fonte: Compilação do autor.

Como é demostrado na tabela 7, os algoritmos C4.5 e SVM obtiveram a melhor acurácia com 23% e 21% respectivamente. Os que tiveram a pior acurácia foram o Rede Neural e o KNN com 1,46% e 0,36% nessa ordem.

4.7 Teste 7

No sétimo teste a amostra é selecionada de forma aleatória sem a execução da classificação de texto e sem o atributo historicoFato.

Tabela 9 – Teste 7

ALGORITMO	Teste 7
C4.5	22,9357
CART	-
KNN	0,9174
SVM	26,2385
RandomForest	1,2844
Rede Neural	2,5688
Ripper	21,4678

Fonte: Compilação do autor.

Como é demostrado na tabela 8, obtiveram as melhores acurácias os algoritmos SVM e C4.5 com 26% e 22% respectivamente. Os que tiveram as piores acurácias foram o KNN e *Random Forest*, ambos com 0,91% e a Rede Neural com 1,18%.

4.8 Teste 8

Para o último teste foi selecionado o teste com os melhores resultados e então executamos novamente, só que aplicando o *K-fold cross validation* com k igual a 10 (cada iteração foi executado de forma separada).

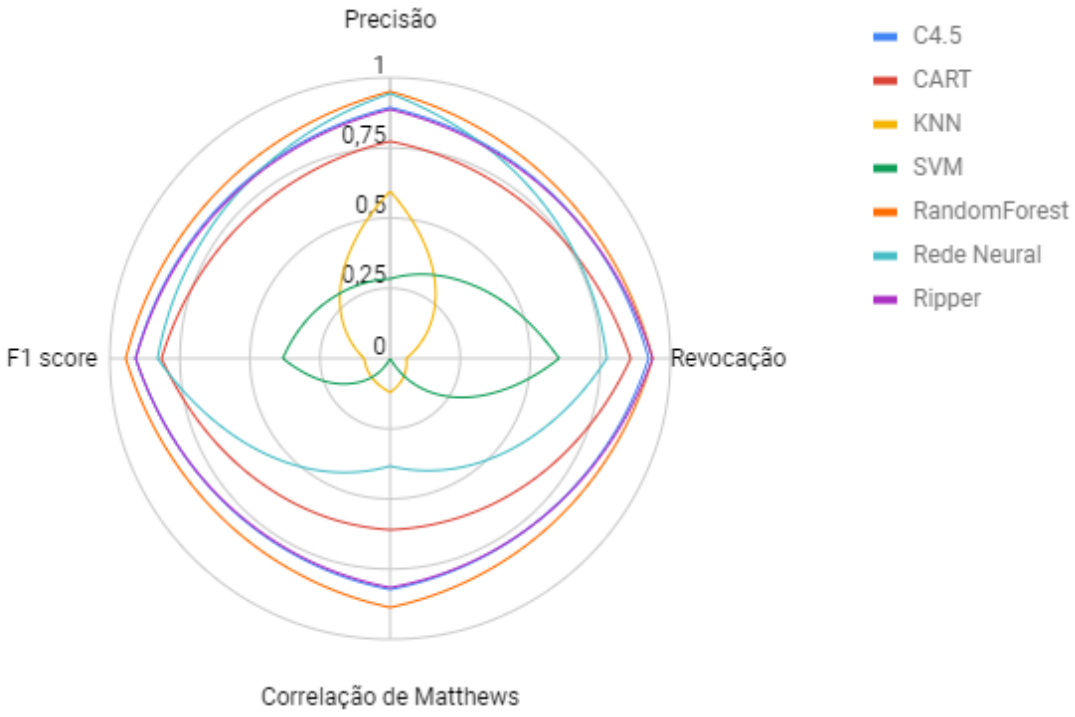
Tabela 10 – Teste 8

ALGORITMO	Teste 8
C4.5	90,9444
CART	80,2777
KNN	52,5
SVM	46,7777
RandomForest	94,5
Rede Neural	48,1666
Ripper	90,6111

Fonte: Compilação do autor.

Em relação a acurácia do teste da tabela 9, os melhores resultados ficaram com os algoritmos *Random Forest* com 94%, C4.5 e *Ripper*, ambos com 90%. As piores acurácias ficaram com SVM e rede neural com 46% e 48% nessa ordem. Foi calculado também as métricas precisão, revocação, correlação de *Matthews* e F1 *score* e a figura 19 mostra os dados obtidos em um gráfico radar.

Figura 19 – Gráfico radar



Fonte: Compilação do autor.

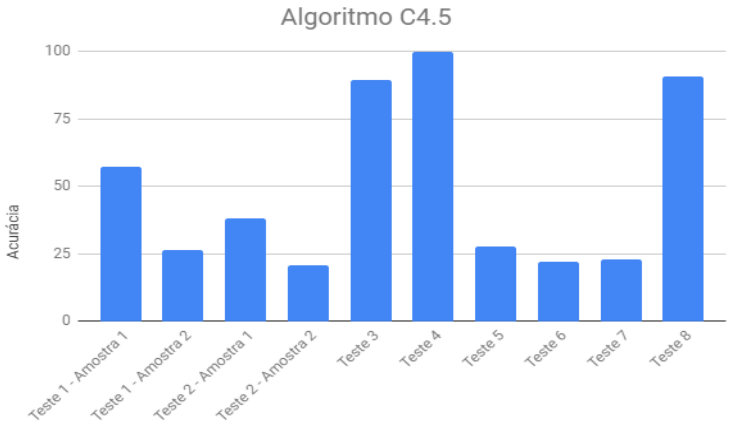
Como pode ver no gráfico da figura 19, os algoritmos com os melhores resultado, de forma geral, foram o C4.5 e o *Random Forest* atingindo valores como 0,82 e 0,89 no coeficiente de correlação de *Matthews* e 0,91 e 0,94 no F1 *score*, nessa ordem. Já o KNN e o SVM atingiram os piores resultados com 0,11 e 0 no coeficiente de correlação de *Matthews* e 0,09 e 0,38 no F1 *score*.

4.9 Resultados por algoritmo

Nesta subseção será mostrado os resultados por algoritmo dos teste em geral.

4.9.1 C4.5

Figura 20 – Gráfico C4.5.

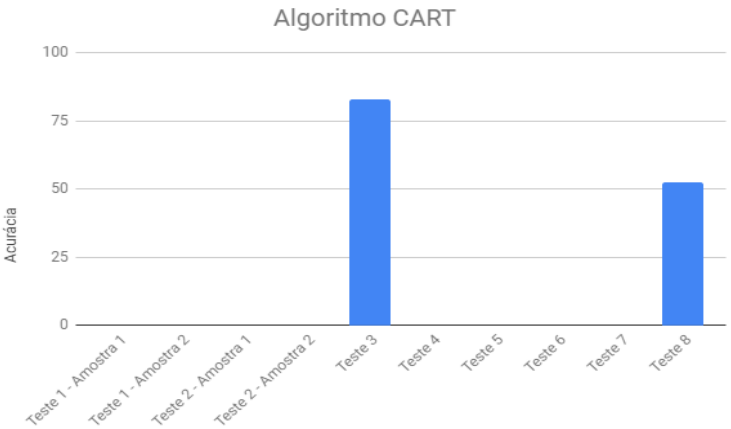


Fonte: Compilação do autor.

Ao observar o gráfico da figura 20, esse algoritmo teve um desempenho melhor nos testes três e oito, por conta da quantidade de classe (2 classes sendo elas roubo e furto qualificado), e no teste quatro, pois a remoção dos 16 primeiros atributos (com exceção da classe) pode ter afetado a acurácia de forma positiva.

4.9.2 CART

Figura 21 – Gráfico CART.

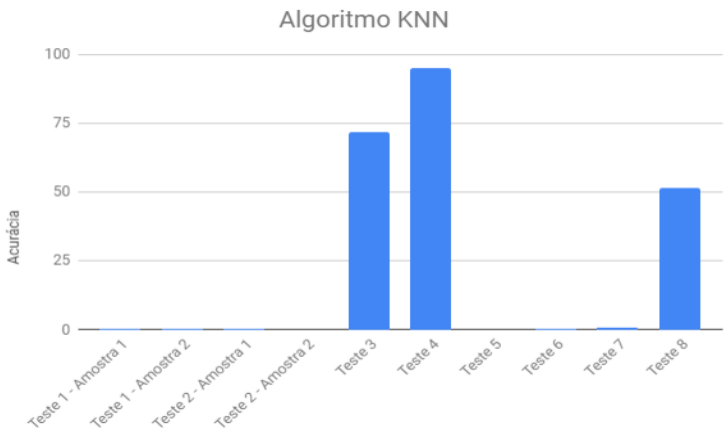


Fonte: Compilação do autor.

Com base nas amostras utilizadas, devido a quantidade de classes e, consequentemente a demora de execução, só foi possível executar o algoritmo CART nos testes três e oito, assim como mostra a figura 21.

4.9.3 KNN

Figura 22 – Gráfico KNN.

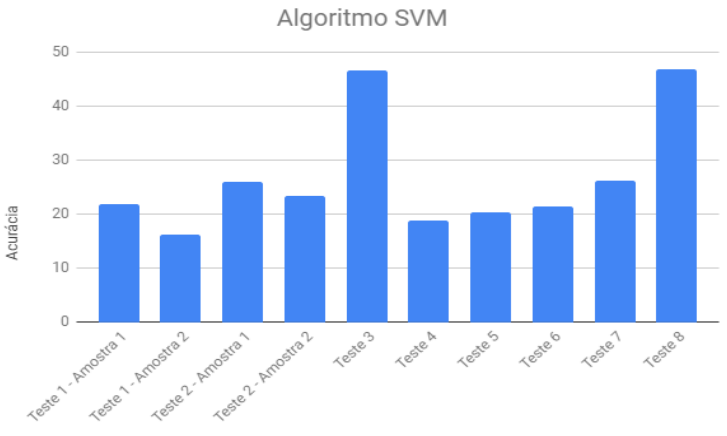


Fonte: Compilação do autor.

Ao observar o gráfico da figura 22, esse algoritmo teve um desempenho melhor nos testes três e oito, por conta da quantidade de classe (2 classes sendo elas roubo e furto qualificado), e no teste quatro, pois a remoção dos 16 primeiros atributos (com exceção da classe) pode ter afetado a acurácia de forma positiva.

4.9.4 SVM

Figura 23 – Gráfico SVM.

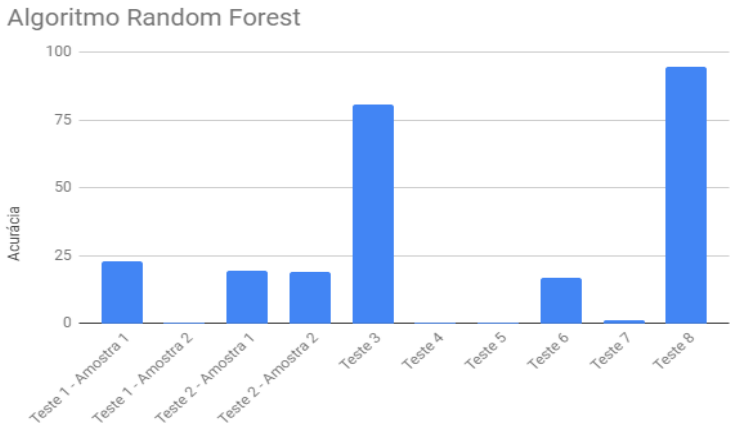


Fonte: Compilação do autor.

De forma geral, os resultados deste algoritmo, apresentado na figura 23, teve poucas variações, com exceção dos testes três e oito, no qual obteve as melhores acurácias, já que o SVM lida melhor com classificação binária, o que é o caso desses dois testes.

4.9.5 *Random Forest*

Figura 24 – Gráfico *Random Forest*.

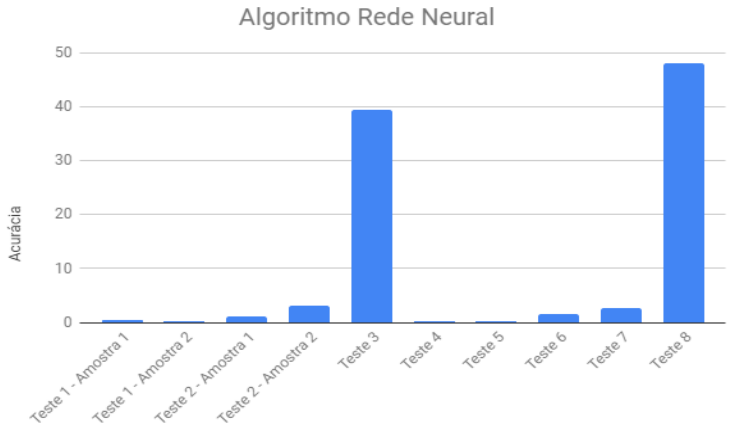


Fonte: Compilação do autor.

Este algoritmo obteve melhores resultados nos teste três e oito, assim como mostra a figura 24.

4.9.6 Rede Neural

Figura 25 – Gráfico Rede Neural.

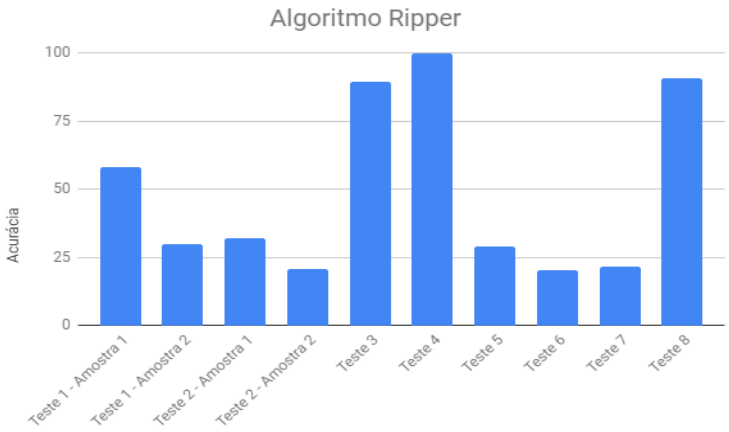


Fonte: Compilação do autor.

Como mostra a figura 25, este algoritmo obteve melhores resultados nos testes três e oito. Por causa da natureza de caixa preta que o algoritmo possui, não foi possível chegar a uma conclusão do por quê resultou em acurácias baixas.

4.9.7 Ripper

Figura 26 – Gráfico Ripper.



Fonte: Compilação do autor.

Este algoritmo obteve melhores resultados nos testes três, quatro e oito, assim como é mostrado na figura 26. Por causa do funcionamento do Ripper ser, de certa forma, similar a de uma árvore de decisão (processo de crescimento e poda, só que aplicada a regra e feita de forma repetitiva), a justificativa desta segunda também pode ser utilizada no primeiro.

4.10 Discussão

Considerando que a classificação de texto gerou, a partir das amostras, uma alta dimensionalidade, chegando a 8749 atributos para a amostra selecionada de forma sequencial e, em média, 14000 para a amostra aleatória, e apresentando dados ruidosos nesses atributos, uma possível justificativa para os resultados positivos do Ripper, segundo Cohen (1995), é que ele foi projetado para ser eficiente em situações que apresentem as características anteriormente citadas. Em relação ao C4.5, segundo Nalini; Sheela (2014), assim como um algoritmo de árvore de decisão, ele possui robustez para dados ruidosos e capacidade de aprender expressões disjuntivas, o que favorece para uma alta acurácia em uma classificação de documentos.

De forma geral, os atributos gerados pela classificação de texto, mencionados no parágrafo anterior, possuem uma importância maior para a classificação do que os dados dos 16 primeiros atributos. Além disso, é mais difícil de obter uma boa classificação quando um modelo é criado a partir de um *dataset*, no qual há poucas instâncias para uma grande quantidade de classes (em média 80 classes). Com isso, os resultados negativos podem ter sido gerados a partir da combinação de dois fatores: a importância dos atributos para a classificação e a grande quantidade de classes para poucas instâncias.

5 Considerações Finais

Uma boa segurança pública é um direito garantido pela constituição e é uma meta que vem sendo cobrada pelo brasileiro a um bom tempo. Mas, perante as dificuldades encontradas, este objetivo parece um sonho distante. Com o aumento de investimento nos setores de segurança, essa área está sendo cada vez mais automatizada para diminuir o trabalho burocrático e tornar as operações de campo mais eficazes. Contudo, ainda não é suficiente, visto que o Brasil continua muito atrasado em relação a tecnologia aplicada neste segmento.

Tendo em mente que o *machine learning* tem um grande potencial de uso, graças a sua interdisciplinaridade e o seu auto aprendizado, suas aplicações para tarefas da área de segurança pública variam desde a identificação de padrões de crimes até a detecção de comportamentos de atividades suspeitas. Dessa forma, ele auxilia de significativamente os trabalhos dos agentes, minimizando assim, erros humanos provenientes de uma rotina de trabalho perigosa e exaustiva.

Com isso, este trabalho buscou, por meio da utilização de *machine learning* e algoritmos de classificação, tipificar ocorrências policiais de forma automatizada, para que erros cometidos no registro de um ato, criminoso ou não, possam ser evitados.

Para o objetivo de gerar o *dataset* do trabalho, foram retiradas amostras, da base de dados de ocorrências policiais, com 1800 instâncias para cada teste realizado, pois não era possível alocar na memória o resultado da classificação de texto utilizando o *dataset* inteiro (465.376 instâncias) devido ao tamanho do *dataset* resultante da classificação de texto. Já que não foi possível realizar a transformação dos atributos para categórico, como foi dito na seção 3.1, para a seleção dos algoritmos de *machine learning* foi usado apenas os seguintes: CART, C4.5, KNN, SVM, Rede Neural, *Random Forest* e Ripper.

Em relação aos objetivos gerar os modelos de classificação e realizar a classificação, essas amostras foram submetidas a sete algoritmos citados anteriormente, sendo que em alguns testes foram realizados a classificação de texto e foi utilizado duas formas de validação o *Hold out* na maioria dos testes e *k-fold cross validation* no último teste.

Já no objetivo avaliar os resultados, de forma geral, o algoritmo que obteve o resultado mais satisfatório foi o C4.5 com picos, em determinadas situações, chegando a 99% de acurácia seguido pelo Ripper, que também teve picos de mesmo valor. Já os que obtiveram os resultados mais insatisfatórios foram o KNN e o Rede neural, ambos chegando a uma acurácia de 0,18%.

Para reafirmar os valores positivos do melhor teste, foi feito uma reexecução do mesmo utilizando a técnica de validação *k-fold cross validation* e com a aplicação de métricas como precisão, revocação, correlação de *Matthews* e F1 score.

Conclui-se então que a hipótese mostrou ser verdadeira, pois foi possível realizar a classificação automática obtendo bons resultados nos testes aplicados, sob contextos específicos, no qual uma amostra aleatória foi dividida em duas classes (roubo e furto qualificado), sendo que na primeira situação foi utilizado a técnica de validação *Hold Out* e na segunda o *k-fold cross validation*. Para ambas as situações foi executado a classificação de texto. Já na terceira, a amostra continha apenas os termos gerados pela classificação de texto e o atributo classe.

Como trabalhos futuros foi pensada em criação de um serviço WEB, no qual exibiria o resultado da predição da classe de uma instância de forma simplificada para facilitar a compreensão dos profissionais que a utilizariam.

Referências

BALLESTEROS, P. R. Gestão de políticas de segurança pública no brasil: problemas, impasses e desafios. *Revista Brasileira de Segurança Pública*, v. 8, n. 1, p. 6–22, 2014.

BREIMAN, L. Bagging predictors. *Machine learning*, Springer, v. 24, n. 2, p. 123–140, 1996.

BREIMAN, L. Random forests. *Machine learning*, Springer, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L. et al. *Classification and Regression Trees*. New York: Routledge, 1984. 368 p. ISBN 9781315139470.

CAMILO, C. O.; SILVA, J. C. d. Mineração de dados: Conceitos, tarefas, métodos e ferramentas. *Universidade Federal de Goiás (UFC)*, p. 1–29, 2009.

CERQUEIRA, D. et al. Atlas da violencia 2016. Instituto de Pesquisa Econômica Aplicada, 2016. Disponível em: <http://www.ipea.gov.br/portal/images/stories/PDFs/nota_tecnica/160405_nt_17_atlas_da_violencia_2016_finalizado.pdf>.

CERQUEIRA, D. et al. Atlas da violencia 2018. Instituto de Pesquisa Econômica Aplicada, 2018. Disponível em: <http://www.ipea.gov.br/portal/images/stories/PDFs/relatorio_institucional/180604_atlas_da_violencia_2018.pdf>.

CHOU, J.-S. et al. Optimizing the prediction accuracy of concrete compressive strength based on a comparison of data-mining techniques. *Journal of Computing in Civil Engineering*, American Society of Civil Engineers, v. 25, n. 3, p. 242–253, 2010.

COHEN, W. W. Fast effective rule induction. In: *Machine Learning Proceedings 1995*. [S.l.]: Elsevier, 1995. p. 115–123.

FAYYAD, U.; PIATETSKY-SHAPIRO, G.; SMYTH, P. From data mining to knowledge discovery in databases. *AI magazine*, v. 17, n. 3, p. 37–37, 1996.

HALL, M. Correlation-based feature selection of discrete and numeric class machine learning. University of Waikato, 2000. Disponível em: <https://app.uff.br/riuff/handle/1/8788>.

HAMADA, H. H.; NASSIF, L. N. Perspectivas da segurança pública no contexto de smart cities: desafios e oportunidades para as organizações policiais. *Perspectivas em Políticas Públicas*, v. 11, n. 22, p. 189–213, 2019.

HSU, C.-W.; CHANG, C.-C.; LIN, C.-J. A practical guide to support vector classification. Taipei, 2003.

KOTSIANTIS, S. B. Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, v. 160, p. 3–24, 2007.

KOWSARI, K. et al. Hdltex: Hierarchical deep learning for text classification. In: IEEE. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. [S.l.], 2017. p. 364–371.

LIM, T.-S.; LOH, W.-Y.; SHIH, Y.-S. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine learning*, Springer, v. 40, n. 3, p. 203–228, 2000.

LOPES, T. D. et al. Aplicacao do algoritmo random forest como classificador de padroes de falhas em rolamentos de motores de inducao. *Simpósio Brasileiro de Automaç~ao Inteligente*, v. 13, n. 2, p. 263–268, 2017.

MITCHELL, T. *Machine Learning*. [S.l.]: McGraw-Hill Science/Engineering/Math, 1997. 432 p. ISBN 0070428077.

NALINI, K.; SHEELA, L. J. Survey on text classification. *International Journal of Innovative Research in Advanced Engineering*, v. 1, n. 6, p. 412–417, 2014.

NOBLE, W. S. What is a support vector machine? *Nature biotechnology*, Nature Publishing Group, v. 24, n. 12, p. 15–65, 2006.

OLIVEIRA, J. C. d.; ALMEIDA, T. J. d. Inquérito policial: acompanhando a tecnologia do mundo atual. 2018. Disponível em: <http://www.ipea.gov.br/portal/images/stories/PDFs/nota_tecnica/160405_nt_17_atlas_da_violencia_2016_finalizado.pdf>.

PADULA, A. J. et al. Segurança pública e inteligência artificial: um estudo georreferenciado para o distrito federal. 2017. Disponível em: <http://www.codeplan.df.gov.br/wp-content/uploads/2018/02/TD_33_Segurança-pública-e-inteligência-artificial_2017.pdf>.

QUINLAN, J. R. Induction of decision trees. *Machine learning*, Springer, v. 1, n. 1, p. 81–106, 1986.

QUINLAN, J. R. Improved use of continuous attributes in c4. 5. *Journal of artificial intelligence research*, v. 4, p. 77–90, 1996.

SEBASTIANI, F. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, ACM, v. 34, n. 1, p. 1–47, 2002.

SIRIKULVIRIYA, N.; SINTHUPINYO, S. Integration of rules from a random forest. In: *International Conference on Information and Electronics Engineering*. [S.l.: s.n.], 2011. v. 6, p. 194–198.

SOUZA, J. R. M. d. Utilização de aprendizagem de máquina na predição de crimes. 2018. Disponível em: <https://app.uff.br/riuff/handle/1/8788>.

SPANHOL, F. J.; LUNARDI, G. M.; SOUZA, M. V. d. *Tecnologias da informação e comunicação na segurança pública e direitos humanos*. São Paulo: Blucher, 2016. 206 p. ISBN 978-85-8039-177-0.

SUYKENS, J. A.; VANDEWALLE, J. Least squares support vector machine classifiers. *Neural processing letters*, Springer, v. 9, n. 3, p. 293–300, 1999.

TIMOFEEV, R. Classification and regression trees (cart) theory and applications. *Humboldt University, Berlin*, 2004.

TRIPATHY, A.; AGRAWAL, A.; RATH, S. K. Classification of sentiment reviews using n-gram machine learning approach. *Expert Systems with Applications*, Elsevier, v. 57, p. 117–126, 2016.

WIDROW, B.; LEHR, M. A. 30 years of adaptive neural networks: perceptron, madaline, and backpropagation. *Proceedings of the IEEE*, IEEE, v. 78, n. 9, p. 1415–1442, 1990.

XU, R.; YANG, Y. Cross-lingual distillation for text classification. *arXiv preprint arXiv:1705.02073*, 2017.