UNIVERSIDADE FEDERAL DO TOCANTINS
CAMPUS DE PALMAS
PROGRAMA DE PÓS-GRADUAÇÃO MESTRADO PROFISSIONAL EM MODELAGEM COMPUTACIONAL DE SISTEMAS

Rogério Nogueira de Sousa

MINERJUS: SOLUÇÃO DE APOIO À CLASSIFICAÇÃO PROCESSUAL COM USO DE INTELIGÊNCIA ARTIFICIAL

Palmas/TO
2019

ROGÉRIO NOGUEIRA DE SOUSA

MINERJUS: SOLUÇÃO DE APOIO À CLASSIFICAÇÃO PROCESSUAL COM USO DE INTELIGÊNCIA ARTIFICIAL

Dissertação apresentada ao Programa de Pós-Graduação em Modelagem Computacional de Sistemas da Universidade Federal do Tocantins (PPGMCS/UFT). Foi avaliada para obtenção do título de Mestre em Modelagem Computacional de Sistemas e aprovada em sua forma final pelo orientador e pela Banca Examinadora.

Orientador: Doutor David Nadler Prata

Palmas/TO
2019

Dados Internacionais de Catalogação na Publicação (CIP)
Sistema de Bibliotecas da Universidade Federal do Tocantins

N778m Nogueira de Sousa, Rogério.

MINERJUS: SOLUÇÃO DE APOIO À CLASSIFICAÇÃO PROCESSUAL COM USO DE INTELIGÊNCIA ARTIFICIAL. / Rogério Nogueira de Sousa. – Palmas, TO, 2019.

59 f.

Dissertação (Mestrado Acadêmico) - Universidade Federal do Tocantins – Câmpus Universitário de Palmas - Curso de Pós-Graduação (Mestrado) em Modelagem Computacional de Sistemas, 2019.

Orientador: David Nadler Prata

1. Inteligência Artificial. 2. Prestação Jurisdicional. 3. Máquina de Aprendizado. 4. Processamento de Linguagem Natural. I. Título

CDD 4

Elaborado pelo sistema de geração automática de ficha catalográfica da UFT com os dados fornecidos pelo(a) autor(a).

UNIVERSIDADE FEDERAL DO TOCANTINS
CAMPUS DE PALMAS
PROGRAMA DE PÓS-GRADUAÇÃO MESTRADO PROFISSIONAL EM MODELAGEM COMPUTACIONAL DE SISTEMAS

Palmas, 10 de setembro de 2019

Aos 10 (dez) dias do mês de setembro do ano de 2019, na sala 204, bloco III do Campus de Palmas, realizou-se a defesa final da Dissertação de Mestrado de ROGÉRIO NOGUEIRA DE SOUSA, do Mestrado Profissional de Modelagem Computacional de Sistemas, Universidade Federal do Tocantins(UFT), intitulado: MINERJUS: SOLUÇÃO DE APOIO À CLASSIFICAÇÃO PROCESSUAL COM USO DE INTELIGÊNCIA ARTIFICIAL, realizado sob a Orientação do Dr. David Nadler Prata tendo como banca avaliadora, os professores abaixo relacionados. Atribuíram a Nota Final A, (Aprovado). Nada mas tendo a constar, assinam esta Ata os professores componentes da banca.

Professor Dr. David Nadler Prata
Orientador – PPGMCS-UFT

Professor Dr. Humberto Xavier de Araújo
Avaliador interno – PPGMCS-UFT

Marcelo Lisboa Rocha
Professor Dr. Marcelo Lisboa Rocha
Avaliador interno – PPGMCS-UFT

Professora Dra. Juliana Ricarte Ferraro
Avaliadora externa – GESPOL-UFT

Professor Dr. Wagner Figueiredo Sacco
Avaliador externo – Universidade Federal do Oeste do Pará

Dedico este trabalho à minha Mãe que sempre
é a luz que me guia.

"Não há nada tão inútil quanto fazer com grande eficiência algo que não deveria ser feito"
(Peter Drucker)

AGRADECIMENTOS

Agradeço aos meus Pais, Ismael Caetano de Sousa e Maria Lúcia Nogueira de Sousa, que nunca me faltaram, sempre me incentivado e apoiado.

À minha namorada Jacqueline Rodrigues, que pacientemente me apoiou incondicionalmente.

Aos professores do Programa de Pós-graduação de Modelagem Computacional de Sistema da Universidade Federal do Tocantins, principalmente ao meu orientador Prof. Dr. David N. Prata.

À Prof. Dra. Juliana Ricarte Ferraro, por me encorajar a enfrentar e apoiar no enfrentamento deste desafio.

Aos mesmo amigos, por torcerem por mim e por momentos de tranquilidade e descontração.

RESUMO

O processo judicial eletrônico é uma realidade no Brasil, onde 70% dos casos novos em todo poder judiciário são virtuais. Fazer uso adequado desta realidade e aprimorá-la é primordial para dar vazão à demanda de aproximadamente 25 milhões de processos novos por ano. Este projeto propõe uma melhoria na celeridade e corretude da classificação dos processos eletrônicos, por meio da utilização de Inteligência Artificial. Com isso, conjectura-se auxiliar operadores do direito responsáveis pelo cadastro do documento petição inicial (criação do processo), bem como os responsáveis pela sua análise, por meio de sugestão automática e assertiva quanto ao assunto do processo, imprimindo maior agilidade de tramitação e qualidade nas informações contidas nos autos judicias brasileiros. Durante o desenvolvimento da solução foram testados diversos algoritmos de aprendizado de máquina identificando o de melhor performance, no caso o Suporte Vector Machine, com relação a acurácia e precisão, bem como o tempo resposta e treinamento, para uma eficiente classificação processual.

Palavras-chaves: Inteligência Artificial, Aprendizagem de Máquina, Prestação Jurisdicional, Processo Judicial Eletrônico.

ABSTRACT

The electronic court case is a reality in Brazil, where 70% of new cases in all judiciary are virtual. Making proper use of this reality and improving it is paramount to meet the demand for approximately 25 million new processes per year. This project proposes an improvement in the speed and correctness of the electronic process classification through the use of Artificial Intelligence. Thus, it is conjectured to assist legal operators responsible for the registration of the initial petition document (creation of the process), as well as those responsible for its analysis, by means of automatic and assertive suggestion on the subject matter of the process, providing greater agility of processing and quality to the information contained in the Brazilian court records. During the development of the solution several machine learning algorithms were tested identifying the best performing, in this case Support Vector Machine, accuracy and precision as well as the response time and training for efficient process classification.

Keywords: artificial intelligence, machine learning, jurisdictional provision, Eletronic Legal Process.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AM	Aprendizado de Máquina
API	Interface de Programação de Aplicativos
CNJ	Conselho Nacional de Justiça
E-Proc	Processo Eletrônico
IA	Inteligência Artificial
MA	Máquina de Aprendizagem
PDF	Portable Document Format
PLN	Processamento de Linguagem Natural
TI	Tecnologia da Informação
TF	Frequência do Termo
IDF	Inverso da Frequência do Termo no Documento
TJTO	Tribunal de Justiça do Tocantins
TJDF	Tribunal de Justiça do Distrito Federal
TJBA	Tribunal de Justiça da Bahia
TJRR	Tribunal de Justiça de Roraima
TPU	Tabela Processual Unificada
UFT	Universidade Federal do Tocantins

LISTA DE SÍMBOLOS

P	Precisão
R	Recall
F1	F-Score
COS	Coseno

SUMÁRIO

1. INTRODUÇÃO

Em 2016, o Poder Judiciário gastou R$ 2.248.734.431 com Tecnologia da Informação (TI) e contava com uma força de trabalho composta por 442.345 colaboradores, divididos entre magistrados, servidores e auxiliares, para fazerem frente aos 79,7 milhões de processos que estavam pendentes naquele ano na justiça brasileira (CONSELHO NACIONAL DE JUSTIÇA, 2017). No ano de 2017, o gasto com TI reduziu para R$ 2.207.995.675 e o número de processos em tramitação passaram os 80 milhões, com praticamente a mesma quantidade de colaboradores de 2016 (CONSELHO NACIONAL DE JUSTIÇA, 2016).

Diante deste cenário com números expressivos, apresenta-se uma situação preocupante de aumento de demanda judicial e escassez de recursos. As buscas por soluções cada vez mais eficientes, que possam maximizar a capacidade de trabalho dos colaboradores, bem como reduzir custos por processos, tornam-se imperiosas para a prestação jurisdicional no Brasil.

A Tecnologia da Informação é então conceituada como uma das formas de imprimir maior celeridade às atividades judiciais, com menor dispêndio de tempo dos profissionais envolvidos e, via de consequência, com maior economia de recursos (FELIPE; PERROTA, 2018). O dever de eficiência implica a exigência de que a Administração Pública incorpore os progressos tecnológicos em suas atividades (JUSTEN FILHO, 2016). A justiça brasileira tem plena consciência da importância da TI para a prestação jurisdicional, tanto que destina em torno de 25% do seu orçamento (excluindo gasto com pessoal) à informática (CONSELHO NACIONAL DE JUSTIÇA, 2017). Entre as soluções tecnológicas voltadas ao sistema de justiça, destacamos o uso de processos judiciais eletrônicos, uma vez que 70% dos novos processos judiciais são eletrônicos.

Alguns tribunais brasileiros se destacam por possuir 100% de processos eletrônicos nos dois graus de jurisdição, entre eles (CONSELHO NACIONAL DE JUSTIÇA, 2017), o Tribunal de Justiça do Tocantins (TJTO), que na vanguarda do processo judicial eletrônico, implantou o e-Proc/TJTO em 2011. Ainda em 2011, 100% dos casos novos passaram a ser virtuais. Após 4 anos, todos os processos em tramitação foram digitalizados, tonando-se em 2015, o primeiro tribunal a ter todo acervo de processos judiciais em formato digital (TJTO, 2015).

A digitalização de dados jurídicos constitui uma megatendência, transformando fluxos de trabalho e modelos de negócios. O volume de dados utilizados no aconselhamento jurídico aumentou exponencialmente (VEITH et al., 2016), gerando maior demanda por seleção, análise e interpretação de uma quantidade de dados sem precedentes. Em contrapartida, tal virtualização facilita o processo de automação, permitindo o crescimento da produtividade, e

ainda reduzindo custos; ampliando a qualidade e minimizando o tempo de inatividade dos operadores do direito.

Neste novo contexto de modernização, atualmente, está se vivenciando uma nova era de automação, na qual robôs e computadores podem não apenas executar uma série de atividades de trabalho físico de rotina de forma mais eficiente e barata que os humanos. Mas, também cada vez mais capazes de realizar atividades que incluem capacidades cognitivas (MCKINSEY GLOBAL INSTITUTE, 2017). Com os recentes desenvolvimentos em robótica, inteligência artificial e aprendizagem de máquina, as tecnologias não apenas fazem coisas que pensávamos que apenas humanos pudessem fazer, mas também podem fazê-las cada vez mais em níveis sobre-humanos de desempenho (MCKINSEY GLOBAL INSTITUTE, 2017).

A petição inicial, como o nome diz, é o primeiro ato para a formação do processo judicial (TJDFT, 2014). O processo passa a existir eletronicamente quando ocorre o cadastro do mesmo nos sistemas de processos eletrônicos. Frente a este fato, um grupo de colaboradores formados por Magistrados do Tribunal de Justiça do Tocantins apontaram que, não raramente, os cartórios judiciais efetuam a reclassificação dos processos, gerando retrabalho, ou, simplesmente, o processo classificado erroneamente passa a tramitar no sistema.

O Conselho Nacional de Justiça (CNJ), com o objetivo de melhorar a administração da justiça e a prestação jurisdicional, definiu padrões de interoperabilidade a serem utilizados no Poder Judiciário. Entre eles, a padronização das tabelas básicas de classificação processual, movimentação, fases processuais, assuntos e partes (TJRR, 2008). Logo, garantir maior confiabilidade à classificação do processo no ato do cadastro da petição inicial é vital. Não apenas para a promoção de dados estatísticos fidedignos, mas também para integrações futuras entre sistemas de informação.

A implantação de um sistema automatizado que auxilie no processo de classificação processual, com base nas informações contidas na petição inicial, tem o potencial de impactar diretamente na eficiência dos colaboradores do judiciário responsáveis pela análise preliminar da inicial. Também se pressupõe benefícios advindos desta automação para advogados que cadastram a petição inicial. Para isso, a solução aqui proposta utiliza registros contidos nas tabelas de classificações processuais geridas pelo CNJ.

Frente à demanda explicitada, este projeto tem como objetivo apresentar uma ferramenta de automação para classificação processual utilizando Aprendizagem de Máquina, que é um seguimento extremamente importante na Inteligência Artificial (JR, 2016). Técnicas de aprendizagem de máquina serão usadas para reduzir significativamente o número de documentos que hoje exigem revisão manual (JR, 2016). Pois, vislumbra-se que a

Aprendizagem de Máquina (AM) seja capaz de prever a classificação de documentos a partir de modelos oriundos de documentos anteriormente classificados corretamente (ZAKI; MEIRA, JR, 2014) em sua base de treinamento. Inicialmente, a ferramenta irá focar na predição do assunto do processo judicial a partir da extração de dados da petição inicial. Mediante o pré-processamento da petição, que consiste inicialmente, em retirar o conteúdo textual dos documentos digitais, geralmente estarão em formato PDF, que compõe a petição. Posteriormente, são aplicadas técnicas de Processamento de Linguagem Natural (PLN) que irão converter os textos em vetores de termos relevantes para a classificação pretendida, e compreensível pelo computador.

Para formação do modelo preditivo (Aprendizagem de Máquina) serão usadas petições iniciais de processos que tramitam na comarca de Augustinópolis- TO, onde o pesquisador e Magistrado titular do Juizado Especial Cível, Dr. Jefferson David Asevedo Ramos, com sua equipe, realizou o trabalho de triagem e validação dos assuntos de um grupo de processos que tramitam naquela vara especializada, atuando como supervisor dos conteúdos a serem submetidos à aprendizagem da máquina.

Os processos selecionados serão divididos em dois grupos, sendo denominados de corpus de treinamento e corpus de teste. O primeiro grupo será utilizado para ensinar a aprendizagem de máquina, padrões de dados relacionados ao conteúdo dos documentos de um determinado assunto, gerando um modelo analítico. O segundo grupo será usado para validar o processo de aprendizagem, por meio da comparação da assertividade e desempenho da solução tecnológica com o processo utilizado atualmente, que consistem em atribuir assuntos manualmente, após análise individual do processo.

Como resultados, esperamos alcançar uma considerável redução do tempo de protocolamento de petições iniciais, aumentando a assertividade em relação ao assunto da petição e mitigar o retrabalho realizado pelos colaboradores dos cartórios judiciais, promovendo assim maior agilidade de tramitação processual e confiança nos dados atribuídos aos processos judiciais.

1.1 Problema de pesquisa

Diante da situação em que se encontra a justiça brasileira, frente à demanda de judicialização, com uma entrância que gira em torno de 25 milhões de processos novos por ano, e sabendo que 70% destes encontram-se em formato digital em 2017, os órgãos do sistema de justiça brasileiro buscam soluções tecnológicas para apoio a prestação jurisdicional, visando

imprimir maior agilidade na tramitação processual, mitigar o retrabalho e promover maior qualidade à prestação jurisdicional.

Não raramente, os cartórios judiciais efetuam ações de saneamentos de processos quanto a sua classificação inicial, a exemplo do Tribunal de Justiça da Bahia, que analisou 404,3 mil processos e identificou que 56% apresentavam erros no cadastro da petição inicial, sendo que dos analisados, 176.161 apresentam falhas na classificação com relação ao assunto, representado 78% dos erros encontrados.

Sendo assim, a pesquisa se propõe a responder o seguinte questionamento: Como reduzir as falhas de classificação processual com relação ao assunto da petição inicial, usando técnicas de inteligência artificial?

1.2 Objetivos

Esta dissertação apresenta o desenvolvimento de uma ferramenta que por meio de técnicas de processamento de linguagem natural e aprendizagem de máquinas, se propõe a apoiar os operadores do direito que efetuam a autuação de processos judicias, sugerindo um assunto ao processo. O assunto sugerido deve estar elencado nas tabelas processuais unificadas do Conselho Nacional Justiça.

1.2.1 Objetivo Geral

Classificar os assuntos dos processos judiciais no ato do cadastro da petição inicial, visando garantir maior corretude e celeridade na tramitação do processo eletrônico.

1.2.2 Objetivos Específicos

1. Automatizar a classificação processual, por meio de aprendizagem de máquina;
2. Extrair informações textuais das petições iniciais;
3. Utilizar técnicas de Processamento de Linguagens Natural (PLN), para melhorar o desempenho da Aprendizagem de Máquina;
4. Avaliar a acurácia e precisão do modelo de predição gerado;

1.3 Estrutura Da Dissertação

A estrutura do trabalho proposto está distribuída da seguinte forma:

O Capítulo 1 traz a Introdução que contextualiza e apresenta o problema a ser enfrentado por esse trabalho e define os objetivos no mesmo.

O Capítulo 2 apresenta a fundamentação teórica, abordando os conceitos jurídicos de petição inicial e a padronização da classificação processual e conceitos tecnológicos com a apresentação de técnicas de processamento de linguagem natural e aprendizagem de máquina.

O Capítulo 3 descreve a metodologia usada para desenvolvimento da solução e formação do corpus de treinamento e teste.

O Capítulo 4 apresenta a avaliação da performance do modelo preditor gerado neste trabalho.

No Capítulo 5, são apresentadas as principais conclusões do trabalho, e o direcionamento para possíveis trabalhos futuros.

2. FUNDAMENTAÇÃO TEÓRICA

2.1 Petição Inicial

Os órgãos jurisdicionais são, por sua própria índole, inertes (CINTRA; DINAMARCO; GRINOVER, 2010), quer dizer, só agem quando provocados. O processo é o método de pleitear algo em juízo através de uma relação jurídica vinculativa de direto público (JÚNIOR, 2015), os processos de conhecimento são iniciados pelo autor por meio da petição inicial. Conforme o artigo 319 da Lei Nº 13.105/15(BRASIL, 2015) trata-se de um documento escrito que deverá ser composto minimamente de:

> I - o juízo a que é dirigida;
> II - os nomes, os prenomes, o estado civil, a existência de união estável, a profissão, o número de inscrição no Cadastro de Pessoas Físicas ou no Cadastro Nacional da Pessoa Jurídica, o endereço eletrônico, o domicílio e a residência do autor e do réu;
> III - o fato e os fundamentos jurídicos do pedido;
> IV - o pedido com as suas especificações;
> V - o valor da causa;
> VI - as provas com que o autor pretende demonstrar a verdade dos fatos alegados;
> VII - a opção do autor pela realização ou não de audiência de conciliação ou de mediação (BRASIL, 2015).

O Novo Código de Processo Civil no seu artigo 206 roga que cabe ao escrivão ou chefe da secretaria autuar à petição inicial, cadastrando dados quanto ao juízo, natureza do processo e outras informações. Porém, quando a autuação ocorre de forma automática, por meio do uso de processo eletrônico, não há necessidade de intervenção do cartório ou da secretaria judicial, uma vez que os advogados públicos ou privados realizam o cadastramento e distribuição diretamente em formato digital, conforme autorizado no 10º artigo da Lei 11.419 de 19 de dezembro de 2006.

Nesse novo contexto, os operadores do sistema de justiça, internos ou externos ao poder judiciário, invariavelmente, são responsáveis pelas informações contidas nos processos judiciais. Com o crescimento progressivo dos processos eletrônicos, os procedimentos de autuação deixam cada vez mais de ser incumbência exclusiva dos servidores dos cartórios judiciais, passando a ser também de responsabilidade de membros externos ao judiciário, como advogados, defensores, procuradores e promotores, que fornecem os dados elementares ao nascedouro do processo.

Os sistemas de processo eletrônico como PJe do CNJ, e-Proc do Tribunal Regional Federal da 4ª região (TRF4), e-SAJ do Tribunal de Justiça de São Paulo (TJSP) e outros

adotaram como padrão o formato Portable Document Format (PDF), para documentos digitais. Desenvolvido pela empresa Adobe em 1993, o PDF foi criado para exibir e compartilhar documentos. A International Organization for Standardization (ISO) passou a manter o formato, transformando em padrão aberto, assim assegurando seu acesso a longo prazo.

A classificação do assunto no sistema de processo eletrônico, ocorre no ato do cadastramento do documento digital em formato PDF que contem a petição inicial. Na ocasião, deve ser analisado o objeto e os pedidos contidos na inicial para então informar o(s) tema(as) do(s) processo, o primeiro assunto informado dever ser o assunto correspondente ao tema principal da lide. Cabe notabilizar que os assuntos cadastrados serão via de regra inalteráveis até o término do processo.

2.2 Tabelas Processuais Unificadas

O artigo 196 da Lei nº 13.015/2015, conferiu ao Conselho Nacional de Justiça (CNJ) o poder de regular a prática e a comunicação de atos processuais e promover a compatibilidade dos sistemas, disciplinando a incorporação de novos avanços em tecnologia da informação e editando as normas necessárias para essa finalidade.

Com a intenção de padronizar a nomenclatura com relação a classificação dos processos judiciais em todo o território brasileiro, o CNJ por meio da Comissão de Padronização e Uniformização Taxonômica e Terminológica, editou a Resolução n. 46 em 18 de dezembro de 2007, que institui as Tabelas Processuais Unificadas (TPU), padronizando taxonomicamente as classes, assuntos e movimentações processuais. Visando promover interoperabilidade entre sistemas de informações que operam com processos eletrônicos, por meio da uniformidade no tratamento dos metadados que representam a informação contida nos atos judiciais, assim facilitando a geração de estatísticas mais precisas e possibilitando o aproveitamento de informações processuais entre os diferentes graus de jurisdição, mesmo se tratando de sistemas de informação diferentes.

As estatísticas coletadas são fundamentais, não apenas para o processo em si, mas também para o planejamento estratégico do poder judiciário, podendo ser utilizadas para identificação de gargalos em cada fase processual e promoção de ações com a intenção de mitigá-los de forma precisa. As informações extraídas das estatísticas também são preciosas para a sociedade, como por exemplo a identificação de assuntos mais recorrentes em processos judicias para formulação de políticas públicas com intuito de evitar novos conflitos judiciais.

2.3 Processamento de Linguagem Natural

O Processamento de linguagem Natural (PLN), conhecido também na academia como linguística computacional, vem crescendo rapidamente, pois suas teorias e métodos estão sendo aplicados em uma gama de novas tecnologias (BIRD; KLEIN; LOPER, 2009). Essa área de estudo objetiva fornecer ferramentas para que um sistema computacional seja capaz de lidar com linguagens naturais em diversos níveis, como morfológico, sintático e semântico (COPPIN, 2017). Para construção de rotinas que implementam métodos de PLN, utilizamos a biblioteca NLTK (Natural Language Toolkit)(BIRD; KLEIN; LOPER, 2009), inicialmente projetada para o ensino, na atualidade é adotado pelo mercado devido a sua usabilidade e abrangência (PERKINS, 2010).

Para análise linguísticas de textos, em formato digital, escritos em linguagens naturais é necessário processar o texto buscando definir e identificar claramente o que são caracteres, palavras e sentenças em qualquer documento.

2.3.1 Processamento do texto

O método de quebrar um texto em pequenos pedaços é conhecido como tokenização (PERKINS, 2010), esses pedaços são denominados tokens, sendo esses termos individuais detectados no processamento do texto . Os textos podem ser tokenizados em sentenças, isto é gerada uma lista de sentenças, a partir do texto original, onde cada sentença é considerada um token. Tokenização de sentenças consiste em dividir as frases em palavras individuais. A simples tarefa de criação de lista de palavras advindas de uma sentença é uma parte essencial para todo o processamento de texto (PERKINS, 2010).

Algumas palavras comuns ocorrem com muita frequência em qualquer idioma, mas apresentam baixa relevância para expressar o significado da frase, essas são chamadas de stopwords (LANE; HOWARD; HAPKE, 2017). Geralmente artigos, conjunções, preposições, interjeições, verbos auxiliares e palavras muito repetidas na linguagem natural que compõe esse grupo. Tais palavras são retiradas dos textos após a tokenização visando reduzir o esforço computacional, quando se quer extrair informações de um texto. Cabe salientar que em alguns casos como processamento de textos curtos, a retirada dos stopwords pode levar à perda de informações relevantes para o significado do texto.

O vocabulário representa o conjunto de palavras (tokens) que será usado no processamento do texto. Logo o tamanho do vocabulário implica diretamente na complexidade

computacional e na memória requerida para o devido processamento. O uso de técnicas que reduzam o vocabulário é imprescindível para o ganho de performance bem como pode proferir maior generalidade ao processamento.

Tais técnicas buscam transformar diversas palavras com significados semelhantes em uma só. Uma dessas técnicas é converter todas as letras do texto para minúsculas. Por ser muito comum palavras iniciadas com letra maiúscula, ter o mesmo significado da mesma com a letra inicial em minúsculo. Mas em alguns casos o significado muda, por exemplo as palavras 'gentil' e 'Gentil', a primeira é usada como um adjetivo e a segunda como substantivo, no caso nome próprio. Assim o uso da técnica de conversão do texto em minúscula deve ser avaliado de acordo com propósito do processamento, não sendo recomendado quando se almeja detectar no texto entidades nomeadas, como nomes próprios.

Para dar mais generalidade aos termos, é efetuado um processamento em cada termo, onde cadeias morfologicamente complexas são identificadas, decomposta em radical e afixos, sendo descartados os afixos e o termo passa a ser apenas o radical, processo conhecido como stemming (LANE; HOWARD; HAPKE, 2017).

Quando se adota a técnica de stemming para formação do token, removendo o sufixo e prefixo, temos um termo mais genérico, por exemplo a palavras 'livro', 'livrinho', 'livros' e 'livrecos', todas possuem significados semelhantes ou próximos e em comum a cadeia de caracteres 'livr', sendo está o elemento base para o significado. Logo pode se substituir as quatro palavras pelo radical 'livr' que não há perda considerável de significado. Mesmo que 'livr' não seja uma palavra existente, não importa por que o objetivo é casar as palavras em consultas e em documentos e não as mostras ao usuário (COPPIN, 2017).

Algumas técnicas de processamento de linguagem natural, diferentemente das citadas, buscam enriquecer o vocabulário. Como a identificação de séries de palavras que apresentam um único significado, formando um único token, quando se apresentam juntas, por exemplo a sequência de palavras "matéria prima". Essa representação simplificada da linguagem escrita é chamada de Ngram. O 'N' é o número de unidades ou tokens, que compõe a representação, tipicamente são caracteres ou palavras delimitadas por espaço, (BANERJEE; PEDERSEN2, 2003). Uma forma de validar se um Ngram carrega significado é checar a sua frequência de ocorrência em vários documentos, logo aqueles que são mais raros tendem a não possuir correlação (LANE; HOWARD; HAPKE, 2017).

2.3.2 Representação do texto

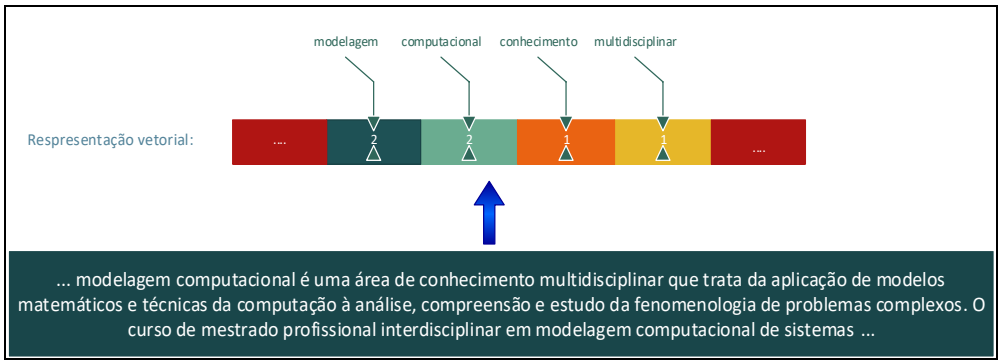
A ação mais importante no Processamento de Linguagens Naturais (PLN) é a conversão do texto, em uma representação legível aos algoritmos computacionais, possibilitando a extração de informação contidas nos textos, por meio de ferramentas tecnológicas(FARIA, 2018).

Uma forma de representar o texto é por meio do Modelo de Espaço Vetorial *(Vector Space Model* - VSM), proposto por (SALTON; WONG; YANG, 1975), consiste em representar um documento por meio de um vetor formado por um ou mais índices de tokens (termos), quer dizer cada token é representado por uma posição no vetor (índice), e o conteúdo da posição é preenchido com pesos, ou somente 0 e 1, onde zero significa que o termo não está presente no documento. A dimensionalidade do vetor de representação do documento se dá pela quantidade de termos, logo o tamanho dos vetores é a mesma da quantidade de palavras contidas no vocabulário.

Esse modelo de representação também conhecido como "saco de palavras" (*Bag of Words* - BOW), não leva em consideração a posição em que os termos ocorrem, não trazendo informações de relacionamento semântico entre palavras. Pois é formada uma matriz, onde cada linha representa um documento e as colunas representam os tokens.

Uma forma de atribuir pesos às palavras é contando quantidade de ocorrência (frequência) de uma palavra em um documento (Term Frequency - TF), de forma simplificada quer dizer quanto maior a frequência de uma palavra mais relevante ela é para o significado do texto.

Figura 1: Representação vetorial com frequência do termo



Essa abordagem simples de contagem de palavras pode atribuir relevância a palavras muito comuns que não contribuem para o significado do texto, que naturalmente possuem uma frequência muito alta, embora a retirada das stopwords mitigue esse problema.

Uma outra abordagem é usar um fator que reduza o grau de relevância (peso) de um

token com base na análise da frequência do termo em uma coleção de documentos (SALTON; BUCKLEY, 1988), fator esse conhecido como inverso da frequência do documento (Inverse Document Frequence - IDF). Esse fator varia inversamente de acordo o número de documentos *n* que contém o termo e relação ao total de documentos da coleção (JONES, 1972).

$$idf_{\mathrm{i}} = \log \frac{N}{n} \tag{1}$$

Neste contexto, se obtém relevância do termo (tf-idf), por meio do produto da frequência do termo (TF) e o fator IDF, assim os termos mais relevantes serão aqueles com maior ocorrência em um documento e menor ocorrência no restante da coleção de documentos N.

$$tfidf_{\mathrm{i}} = tf_i * \log \frac{N}{n} \tag{1.1}$$

Representação do texto usando o modelo de espaço vetorial são adequadas para classificação de texto, pois representam matematicamente os documentos em vetores, onde os sistemas computacionais são capazes aferir similaridade entre documentos por meio do ângulo formado entre pares de vetores.

Possibilitao uso de medidas de similaridades como cosseno, ou são acessadas diretamente por diversos algoritmos de aprendizado de máquinas, que fazem uso de medidas de similaridades próprias(TURNEY; PANTEL, 2010) .

2.3.3 Representação distribuída

Foi apresentado anteriormente uma forma de representação dos textos considerando as frequências das palavras presentes nos documentos para composição de vetores. Tais vetores desconsideram a ordem das palavras.

A reapresentação distribuída gera um vetor de representação de uma palavra com base no contexto em que ela é inserida, palavras que ocorrem em contextos similares tendem a possuir significados próximos(HARRIS, 1954),

Neste sentido em 2013 foi desenvolvido o algoritmo word2vec (MIKOLOV et al., 2013) que cria representações vetoriais distribuídas chamadas word embedding, capazes de representar palavras, considerando as relações sintáticas e semânticas de cada palavra em relação ao vocabulário e independentemente do idioma dos textos, dando mais flexibilidade ao processamento dos dados(AGUIAR, 2016).

Tais vetores são gerados usando redes neurais que fazem uso de uma camada oculta

do algoritmo backpropagation para atualizar os pesos dessa camada, que dizer gera um vetor por meio de aprendizagem de máquina capaz de capturar propriedades linguísticas indiretamente.

O algoritmo word2vec implementa dois modelos de representação vetorial. Um gera o vetor com as dimensões pré-definidas, considerando o contexto, buscando informar qual seria a palavra faltante, este é conhecido com Continuous Baf-of-Word (Cbow) (AGUIAR, 2016) enquanto o Continuous Skip-Gram por meio de uma palavra, busca informar qual seria o contexto (MIKOLOV et al., 2013).

As representações textuais usando word embededding apresentam uma maior gama de informações quando comparadas com representações que fazem uso de contagem de frequência de palavras, sendo verdade até mesmo em comparação com modelos que utilizam parâmetros de compensação para efeitos de frequência (SCHNABEL et al., 2015).

2.4 Aprendizagem de Máquina

Desde que a inteligência artificial alcançou reconhecimento pela primeira vez como uma disciplina em meados da década de 1950, o aprendizado de máquina tem sido uma área central de pesquisa (QUINLAN, 1986).

O aprendizado e a inteligência estão intimamente ligados, porque um sistema capaz de aprender, ele pode ser chamado de inteligente e vice-versa. Aprendizagem de máquina é um campo central da Inteligência Artificial (IA), que se preocupa com os aspectos computacionais de aprendizagem (SEN; WEISS, 1999). Algoritmos de aprendizado de máquinas possuem como principal característica a capacidade de aprender e se auto programar, podendo adquirir diversas funcionalidades, entre elas a capacidade de generalizar a partir de um conjunto de dados de treinamentos para que possam classificar dados, que não foram submetidos ao algoritmo anteriormente (COPPIN, 2017).

A classificação automática é a tarefa de atribuir um rótulo correto a uma entrada, por exemplo, um sistema pode ler um e-mail e rotular se é um spam ou não, ou analisar um comentário em uma rede social e classificá-lo como positivo ou negativo.

Aprendizagem de Máquina (AM) utilizam funções que são capazes de reconhecer padrões a partir de exemplos, formados por um conjunto de dados devidamente classificados, conhecido como conjunto ou corpus de treinamento. Ao reconhecer tais padrões, a máquina adquiri a capacidade de generalizar, quer dizer que pode rotular indutivamente, informando qual classe pertence uma nova entrada de dados, a partir do conhecimento assimilado dos

exemplos que lhe foram apresentados anteriormente.

Para o reconhecimento de padrões, AM se utiliza de teoria estatística para formulação de modelos matemáticos, capazes de inferir a partir de uma amostra (ALPAYDIN, 2010).

AMs são aplicadas na solução de diversos problemas, como detecção de padrões em imagens, previsão do tempo, reconhecimento de voz, robótica e sistema de conversação. Podem realizar uma abordagem preditiva ou para fazer uma previsão do futuro ou descritiva para obter conhecimento de dados ou até mesmo em ambos.

O desempenho de um algoritmo de aprendizado que atua como classificador, depende da inter-relação entre tamanhos do conjunto de treinamento, dimensionalidade (número de atributos) e complexidade do algoritmo (JAIN; DUIN; MAO, 2000). Deve se evitar o desbalanceamento da base de treinamento, que ocorre quando o número de instâncias negativas supera em muito as instâncias positivas, pois gera uma considerável queda de desempenho do modelo gerado (AKBANI; KWEK; JAPKOWICZ, 2004).

Outro problema recorrente em alguns algoritmos de aprendizado é a superadaptação, também conhecido como *overfitting*, ocorre quando o modelo se adapta aos dados de treinamento de tal forma que perca a capacidade de generalizar. Esse problema se apresenta geralmente quando há ruído nos dados de treinamento ou quando eles não representam adequadamente o espaço de dados possíveis (COPPIN, 2017).

2.4.1 Aprendizagem supervisionada

É a forma de aprendizagem utilizada por AMs, capazes de gerar modelos matemáticos que classificam novas observações, a partir de um conjunto de pares de entrada e saída, previamente correlacionados, D=$\{(X_i, y_i)\}_{i=1}^{N}$. Onde D *é* conhecido como conjunto de treinamento, e N é quantidade de exemplares que serão usados para o treinamento (MURPHY, 2012).

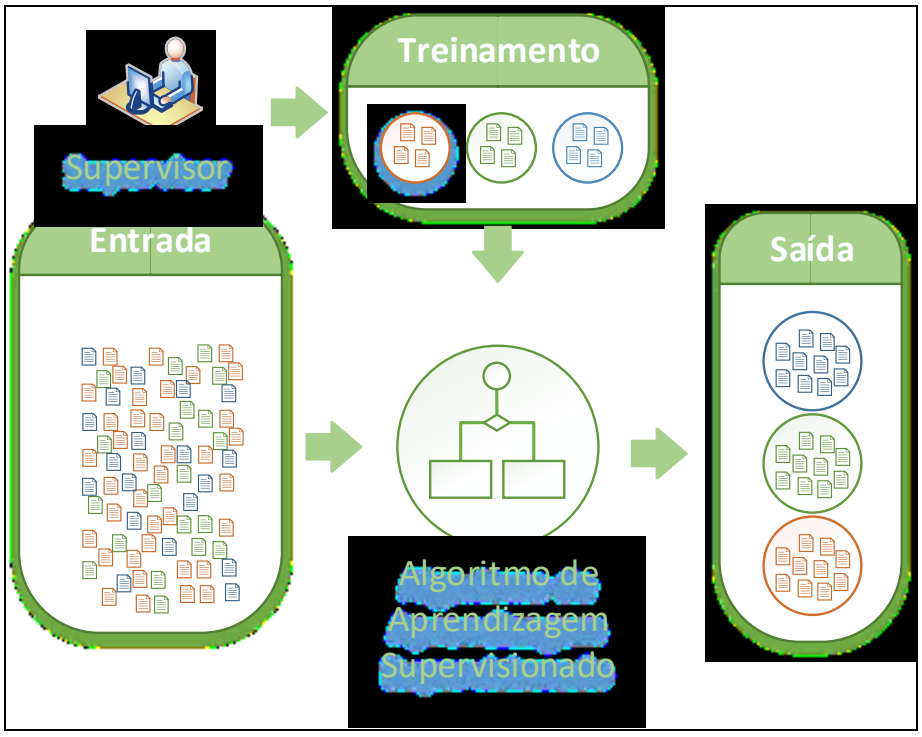
Em um exemplo prático, O X_i pode ser vetores de palavras advindos de documentos, sendo que cada documento é representado por um vetor, e o y_i é o vetor que contém lista de assunto dos documentos. Neste caso, as palavras são atributos (features) dos documentos e os assuntos são os alvos (target), sendo que o tamanho do conjunto de treinamento N é quantidade de documentos devidamente rotulados com relação ao assunto.

Em princípio a saída y_i, pode ser de qualquer tipo, mas na maioria dos casos são variáveis categóricas ou nominais de um conjunto finito de, $y_i \in \{1, \dots, C\}$ (como bom ou ruim), ou podem ser valores escalares (como custo de produção de um produto) (MURPHY, 2012).

Nessa abordagem, existe um elemento externo atuando como um supervisor, com conhecimento prévio que relaciona um conjunto de entradas a um conjunto de saídas, essas informações relacionadas servem de exemplos para o treinamento da máquina de aprendizagem (BISHOP, 2006).

Nesse contexto, onde se tem variáveis de entrada e se conhece as possíveis saídas, as AMs que fazem uso desta forma de aprendizagem, são aplicáveis para problemas de classificação (categorização dos dados de entrada) e regressão (conversão de um conjunto de valores de entrada em um valor de saída).

Figura 2: Aprendizado supervisionado.



2.4.2 Naive Bayes

Thomas Bayes foi um matemático que viveu entre 1702 e 1761, que colaborou com a criação e emprestou seu nome ao teorema de Bayes, este subjacente ao raciocínio probabilístico, sendo um teorema amplamente utilizado para soluções de problemas nos quais não há certezas (COPPIN, 2017).

A estatística bayesiana, incorpora um ciclo de aplicação de conhecimentos prévios, teóricos e empíricos para formular hipóteses, classificando-as com base em dados observados e atualiza estimativas de probabilidade e hipóteses anteriores usando dados observados, tendo em seu núcleo o teorema de Bayes. O teorema descreve as probabilidades dos resultados de

eventos relacionados (dependentes) usando o conceito de probabilidade condicional (BZDOK; KRZYWINSKI; ALTMAN, 2018).

Quando os resultados dos eventos mapeiam naturalmente as probabilidades condicionais, o teorema de Bayes fornece um método intuitivo de raciocínio e computação conveniente (BZDOK; KRZYWINSKI; ALTMAN, 2018). Sendo expresso matematicamente na seguinte equação:

$$P(\mathrm{A}|\mathrm{B}) = \frac{P(\mathrm{B}|\mathrm{A})P(\mathrm{A})}{P(\mathrm{B})} \quad (2)$$

Naive Bayes é um algoritmo de aprendizagem supervisionada que faz uso do teorema de Bayes, também conhecido como classificador ingênuo, por considerar que cada atributo (*feature*) contribui de forma independente para a classificação, sendo essa a implementação mais simples de uma rede bayesiana (KUMAR; SAHOO, 2012).

Cada exemplo usado para o treinamento do algoritmo pode impactar diretamente na probabilidade de uma hipótese está correta, enquanto outros algoritmos de aprendizagem eliminam completamente a hipótese caso identifique inconsistência entre a hipótese e o exemplo, proporcionando maior flexibilidade ao Naive Bayes (MITCHELL, 1997).

Embora seja um algoritmo que possua uma boa eficiência computacional e robustez (SOUSA, 2013), possui algumas limitações, fazendo que tradicionalmente não seja foco de pesquisas, mas tem sido usado diversas vezes em comparativos com algoritmos mais sofisticados (DOMINGOS; PAZZANI, 1997).

2.4.3 K- Nearest Neighbor

O algoritmo de aprendizado K-Nearest Neighbor (KNN), busca classificar uma informação lhe atribuindo a classificação dos vizinhos mais próximos, que já foram previamente classificados (COVER; HART, 1967).

Diferente dos outros algoritmos, que tentam generalizar a partir do conjunto de treinamento, este é um método de aprendizado baseado em instância ou baseado em memória. Isto é, ao receber uma nova entrada o algoritmo a pesquisa no conjunto de treinamento, coleta as informações de classificação dos vizinhos mais próximos e determina a classificação da nova entrada (COPPIN, 2017).

O conjunto de treinamento é formado de entradas dispostas em vetores de *n* dimensões, onde n é o número de atributos que descrevem uma entrada. Assim, quando ocorre uma nova

entrada o algoritmo calcula a distância entre cada vetor do conjunto de treinamento a o novo vetor, sendo a métrica mais comum a distância euclidiana, embora outras métricas possam ser usadas (MURPHY, 2012). Por exemplo, em um caso onde as entradas possuem apenas duas dimensões $E_1 = [x_{1,} y_{1,}]$, $E_2 = [x_{2,} y_{2,}]$, a distância euclidiana entre as duas entradas é dada por:

$$Distância = \sqrt{(x_1 - x_{2,})^2 - (y_1 - y_{2,})^2} \quad (3)$$

Ao obter os *K* vizinhos mais próximos da entrada a ser classificada, o algoritmo lhe atribui a classificação mais recorrente entre os vizinhos selecionados. Onde o valor de K é um parâmetro proferido pelo usuário, com o objetivo de melhorar a assertividade do algoritmo.

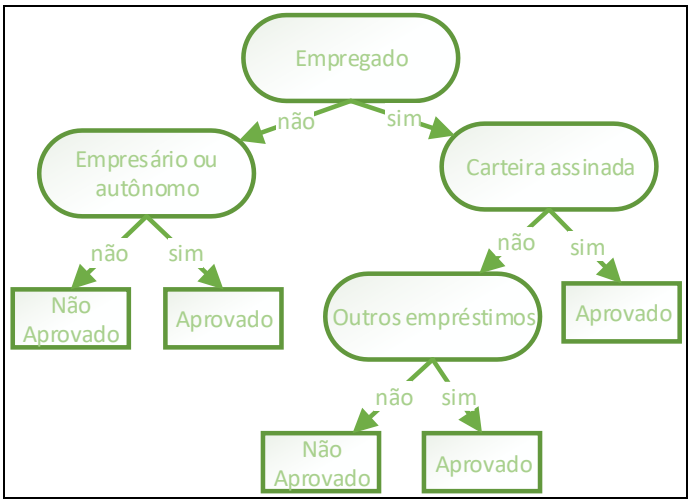
O KNN pode ser usado em problemas de classificação e regressão, esse algoritmo apresenta um bom desempenho com dados de entrada com ruídos (COPPIN, 2017). Além de ser simples e funcionar bem quando a métrica é boa, desde que o conjunto de treinamento seja suficiente. Mas, não funciona bem quando o problema apresenta uma alta dimensionalidade (MURPHY, 2012).

2.4.4 Árvore de Decisão

As máquinas de aprendizagem supervisionada que implementam árvores de decisão, fazem uso do conjunto de treinamento para construir recursivamente de cima para baixo (QUINLAN, 1986), uma estrutura em formato de árvore, onde a parte superior é formada pelos atributos que mais impactam na classificação (COPPIN, 2017). Cada nó da árvore representa uma divisão baseada no conteúdo do atributo, e as classes alvos (targets) ficam contidas em nós folhas.

Tal estrutura é capaz de classificar correntemente novas entrada de dados, por meio de conjuntos de regras do tipo se-então, sendo um dos algoritmos de inferência mais populares (MITCHELL, 1997).

Figura 3: Exemplo de árvore de decisão



Na Figura 3, é mostrada uma árvore de decisão para aprovação de crédito. Nesse caso de uma árvore mais simplificada, é notório que humanos possam interpretar sem grandes dificuldades, podendo ser bastante útil em certas circunstâncias (COPPIN, 2017).

2.4.5 Surpport Vector Machine

O algoritmo Support Vector Machine (SVM) foi desenvolvido à luz da Teoria da Aprendizagem Estatística, que insere o princípio de Minimização do Risco Estrutural. Busca minimizar erros por meio do desenvolvimento de limites teóricos para a generalização de algoritmos de aprendizagem (VAPNIK, 2000).

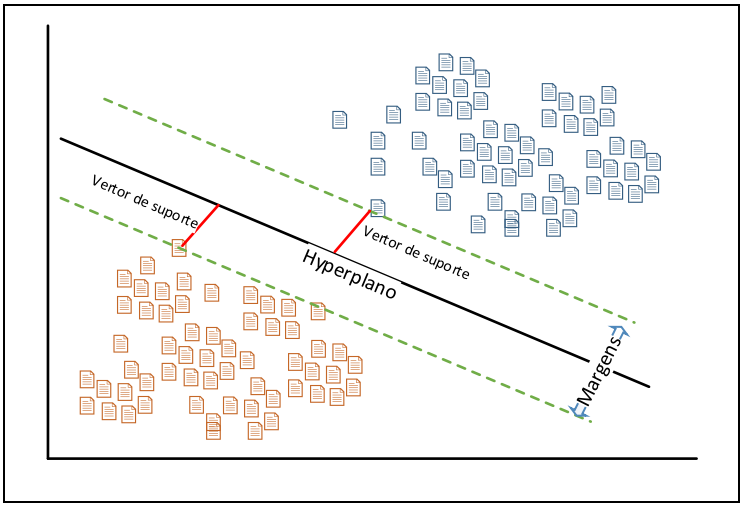
Durante a formulação do modelo preditor, o SVM busca criar divisões no espaço Euclidiano de características, de acordo com o número de alvos (classes). Essa segmentação se dá por meio de hiperplanos de separação ótima, que divide as classes maximizando a separação entre elas. Assim, o SVM passa a ser um problema de programação quadrática, cuja função objetiva convexa sempre pode ser maximizada eficientemente sob as restrições dadas (CRISTIANINI; SCHOLKOPF, 2002).

A Figura 4, exemplifica um espaço linearmente separável e bidimensional, onde um conjunto de dados previamente classificado está disposto. Os pontos mais próximos da linha central (hiperplano de separação) são os vetores de suporte, que são usados para estabelecer as margens de separação máxima.

Devido a modularidade do SVM, a detecção da margem máxima pode ser independente do uso de uma função kernel. Tornando esse tipo de algoritmo de aprendizagem capaz de trabalhar com dados de entradas não lineares, pois o kernel mapeia implicitamente os dados de entrada em um espaço de características de alta dimensionalidade e gerar um

hiperplano no novo espaço para separar os dados mapeados.

Figura 4: Vetores de suporte



O SVM busca adquirir maior flexibilidade ao algoritmo, são inseridas algumas variáveis de relaxamento, que suavizam as restrições impostas no hiperplano ótimo. Assim, o SVM busca encontrar um hiperplano com margem máxima que também minimize as variáveis de folga (ZAKI; MEIRA, JR, 2014).

Os algoritmos *Support Vector Machine* foram originalmente desenhados para classificações binárias, mas podem ser estendidas para regressão e classificações multi-classes.

Ao fazer uso dos vetores de suporte para predição, este tipo de algoritmo tende a mitigar problemas de saturação, que ocorre quando o modelo preditivo (sobre novas entradas) perde a capacidade de generalização por estar super ajustado ao conjunto de treinamento; problema este conhecido como *overffiting*. Outra vantagem do SVM é a ausências de mínimos locais, pois o problema de otimização de margem máxima possui apenas uma solução que pode ser encontrada de forma eficiente (CRISTIANINI; SCHOLKOPF, 2002).

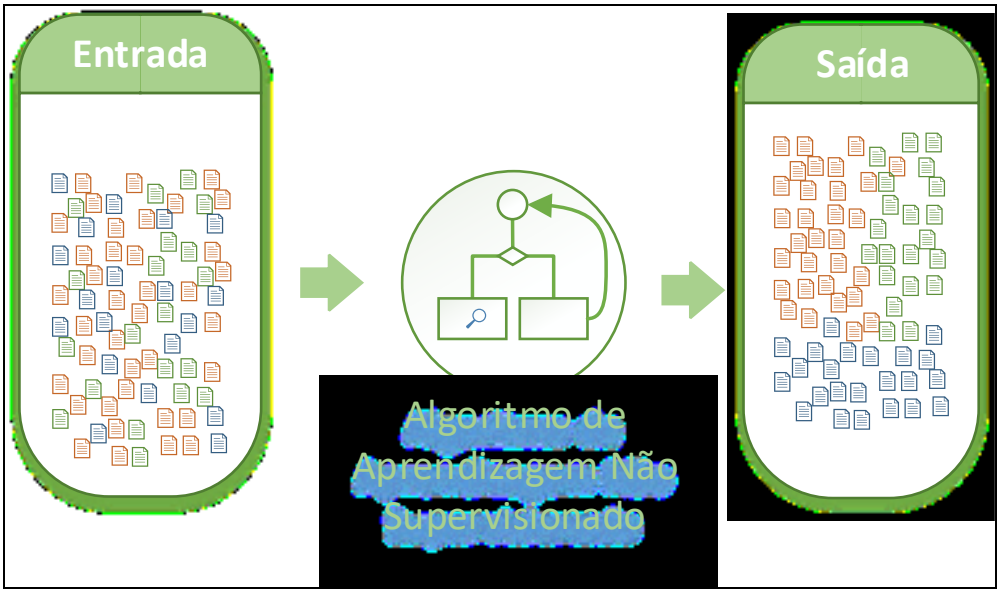
O SVM tem-se mostrado como método muito popular e poderoso, possuindo diversas aplicações como classificação de imagens e diagnósticos de câncer (SUN; CRAIG; ZHANG, 2017). Apesar das vantagens do SVM, os processamentos de treinamento e predição, em casos de problemas complexos, podem ser bastante custosos computacionalmente. Porém, existem implementações do SVM que fazem uso de recursos computacionais, como processadores gráficos e unidade de processamentos (CPU) multi-core, que torna a aplicação das SVMs mais eficientes (WEN et al., 2018).

2.4.6 Aprendizagem não supervisionada

A abordagem de aprendizado descritivo, também conhecida aprendizagem não supervisionada, é aquela onde o algoritmo faz uso somente dos dados de entrada, $D=\{(X_i)\}_{i=1}^{N}$, para buscar aprender por meio da descoberta de padrões. Ao contrário da aprendizagem supervisionada, não se tem acesso a priori das possíveis saídas. Pode-se, desta forma, obter respostas não pensadas. Esse modelo também é chamado de "descoberta de conhecimento" (MURPHY, 2012).

Outra peculiaridade da aprendizagem não supervisionada, advém do fato que o vetor de entrada é formado por múltiplos atributos. Assim, a aprendizagem de máquina deve necessariamente, implementar modelos de probabilidade multivariados, ao contrário dos supervisionados, que buscam prever uma única variável em sua maioria" (MURPHY, 2012).

Figura 5: Aprendizado não supervisionado



2.4.7 K-means

O K-means consiste em um algoritmo de aprendizagem não supervisionada interativo, com baixa complexidade computacional (LUO; LI; CHUNG, 2009), onde o número de clusters (agrupamentos) é atribuído de forma arbitrária por meio de uma constante. Cada cluster se forma em torno de um centroide, que é reposicionado a cada interação visando se tornar o ponto mais central do cluster.

As sentenças são rotuladas com base na sua relação com os centroides. Ou seja, se o elemento é mais próximo de um centroide em relação aos demais, ela passa a pertencer ao

cluster do centroide mais próximo, e recebe o rótulo deste cluster. Isto porque os elementos de um cluster tendem a serem similares e diferentes dos não pertencentes ao grupo. O K-means itera até que não haja mais movimentações de elementos entre os cluster, ou até que o número máximo de iterações tenha sido atingido (JAIN, 1988).

Essa técnica foi utilizada em uma tentativa de detectar cabeçalhos de documentos, por meio do agrupamento de frases de um texto, mas foi retirada do projeto por não está no escopo do projeto, que é um problema de classificação supervisionada.

3. METODOLOGIA

Com a finalidade de apresentar as estruturas que compõe a solução, esta seção detalha a arquitetura e os métodos utilizados para o desenvolvimento de uma ferramenta que se propõe apoiar os operadores do Direito no ato de classificação processual. Nas próximas subseções, serão expostas como foram utilizadas as técnicas de inteligência artificial como as de processamento de linguagem natural e as de aprendizagem de máquina.

A revisão literária realizada anteriormente, embasou o desenvolvimento de um sistema capaz de extrair informações de documentos em formato PDF, processar o texto oriundo dos documentos, submetê-los a um algoritmo de aprendizagem de máquina, previamente treinado, capaz de sugerir o assunto da informação contida no documento.

O problema descrito neste trabalho é de classificação de texto. E, o modelo de aprendizado abordado foi o supervisionado. Logo, por meio de supervisores (pessoas com conhecimento prévio), será formado um conjunto de dados devidamente rotulados, a serem aplicados como exemplos para a aprendizagem de máquina aplicada na solução.

3.1 Corpus de Treinamento

Esse conjunto de dados previamente classificados é conhecido como corpus de treinamento, formado por documentos (petições iniciais) criteriosamente classificados em relação ao assunto. Essa massa de dados será responsável por apresentar à aprendizagem de máquina os contextos e os assuntos, que serão usados para classificação das petições iniciais a ele submetidas.

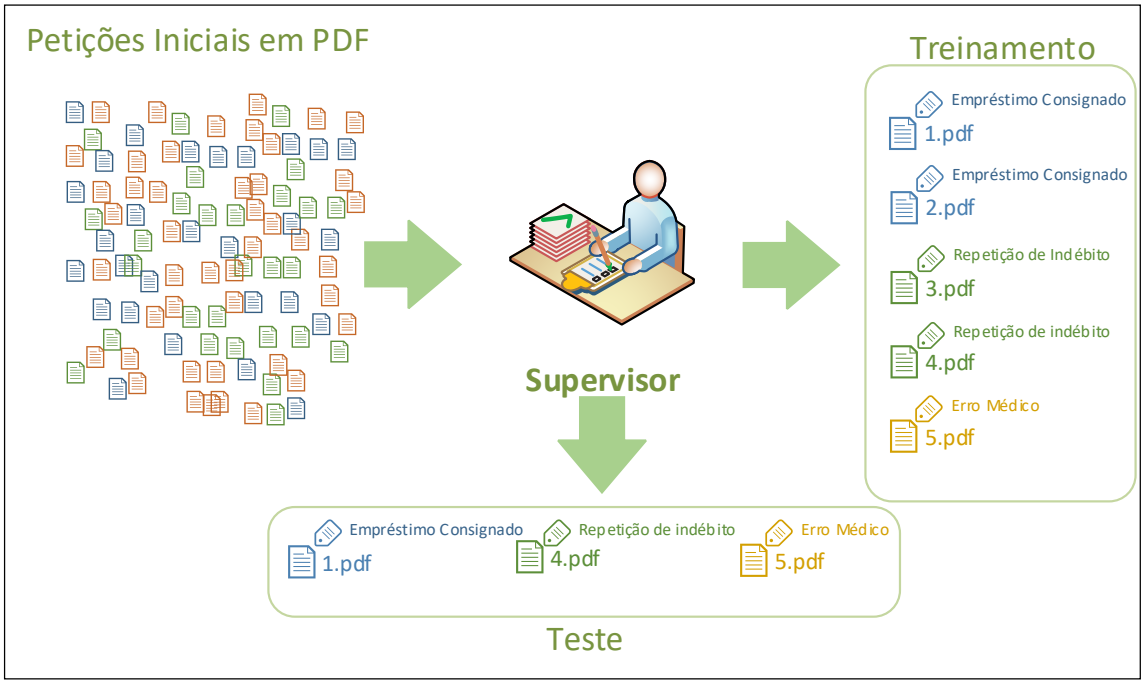
A seleção dos processos judiciais e suas respectivas petições iniciais deverão passar por uma análise rigorosa, um a um. E, quando necessário, corrigido o assunto que lhe fora atribuído no ato do cadastro inicial no sistema de processo eletrônico.

Esta ação foi realizada por uma equipe formada por Magistrados e analistas judiciários do Tribunal de Justiça do Tocantins com experiência em prestação jurisdicional. Busca-se minimizar possíveis falhas de interpretação quanto ao assunto atribuído, que posteriormente seriam repassadas equivocadamente para a aprendizagem de máquina, refletindo negativamente na capacidade de predição da ferramenta. A equipe citada capitaneada pelo Magistrado Dr. Jefferson David Asevedo Ramos, atuou como supervisora da Aprendizagem de Máquina (Figura 6).

A legislação interna do TJTO, coloca que deve-se usar exclusivamente arquivos no formato PDF (Portable Document Format) para textos (TJTO, 2011). Logo, todos os documentos utilizados estarão em PDF. Assim, é imperioso para esta solução, que o sistema seja capaz de extrair textos deste formato de documentos. Todos os documentos deverão passar por um extrator de textos, possibilitando a manipulação do conteúdo pelo computador.

Os processos de onde foram retiradas as petições iniciais com base no assunto, estão no sistema eletrônico de processos judiciais do Tribunal de Justiça do Tocantins (E-Proc), tais processos tramitam no juizado especial cível, da comarca de Augustinópolis, e estão em fase de conhecimento.

Figura 6: Formação do corpus de treinamento e teste



3.2 Arquitetura da solução

A arquitetura da solução é inicialmente dividida em duas, sendo que a primeira é a parte que roda no servidor, implementando as regras que compõe o sistema. E, a segunda, é uma interface web que roda no cliente e é parte responsável pela ligação da solução com os usuários.

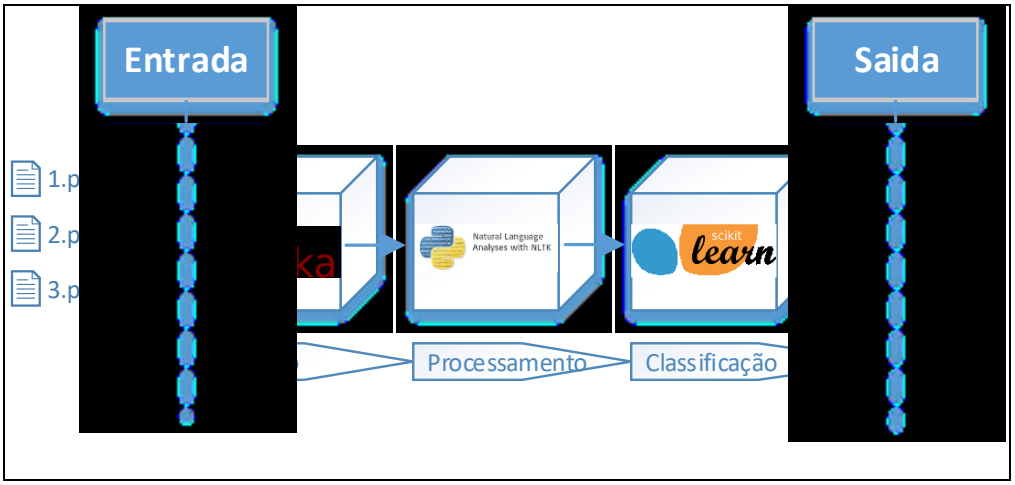
A linguagem utilizada no desenvolvimento das ferramentas, que rodam no servidor e compõem a solução, é a Python, na versão 3.7. Por ser uma linguagem de alto nível, orientada a objetos, Python pode ser utilizado em diversas plataformas pelo fato de ser interpretada (PYTHON.ORG, 2019). Esta linguagem tem se mostrado uma boa escolha pela velocidade de

desenvolvimento e manutenção e vem se estabelecendo como uma das linguagens mais populares da computação científica (PEDREGOSA et al., 2011). O Python ainda conta com uma comunidade que se orgulha da sua cultura de desenvolvimento, com tradição de APIs bem definidas e capilaridade de métodos de aprendizagem de máquina (SZYMAŃSKI; KAJDANOWICZ, 2019).

A interligação entra ambas partes se dá por meio serviços web. Este sistema que roda servidor por meio de API *(Application Programming Interface*), permite que diferentes aplicativos em ambientes distintos possam requisitar e prover informações independentemente da plataforma ou linguagem em que foram escritos, garantindo assim um baixo acoplamento entre a interface do usuário e sistema que processa as informações. A estratégia adotada para que a solução pode ser integrada em outros sistemas, como por exemplo em sistemas de processos eletrônicos.

Além da API, o sistema que roda no servidor (figura 2) também é responsável por: extração dos textos contidos nos arquivos PDFs; e, processamento e classificação do texto, sendo assim a parte mais complexa deste trabalho.

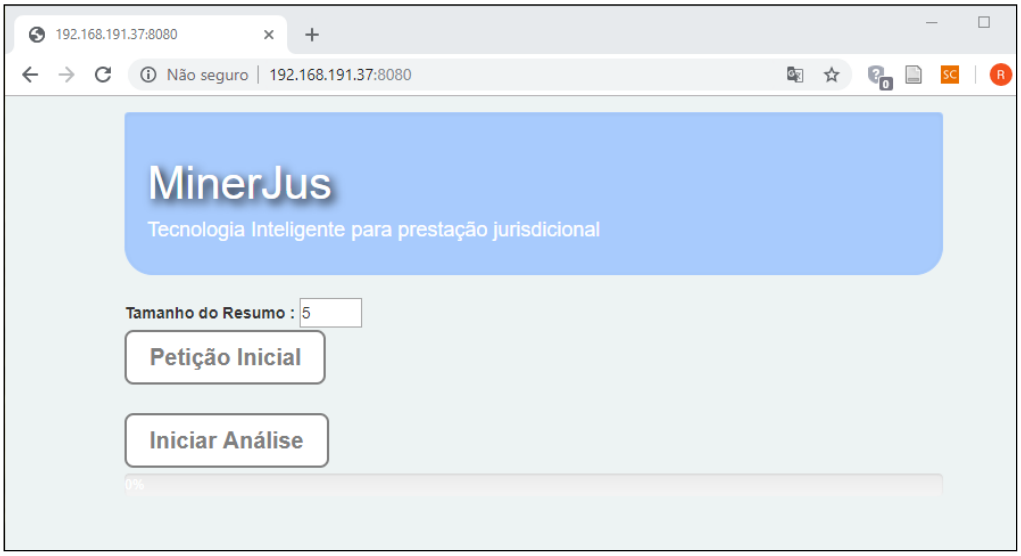
Figura 7: Processo de classificação do texto



3.2.1 Interface WEB

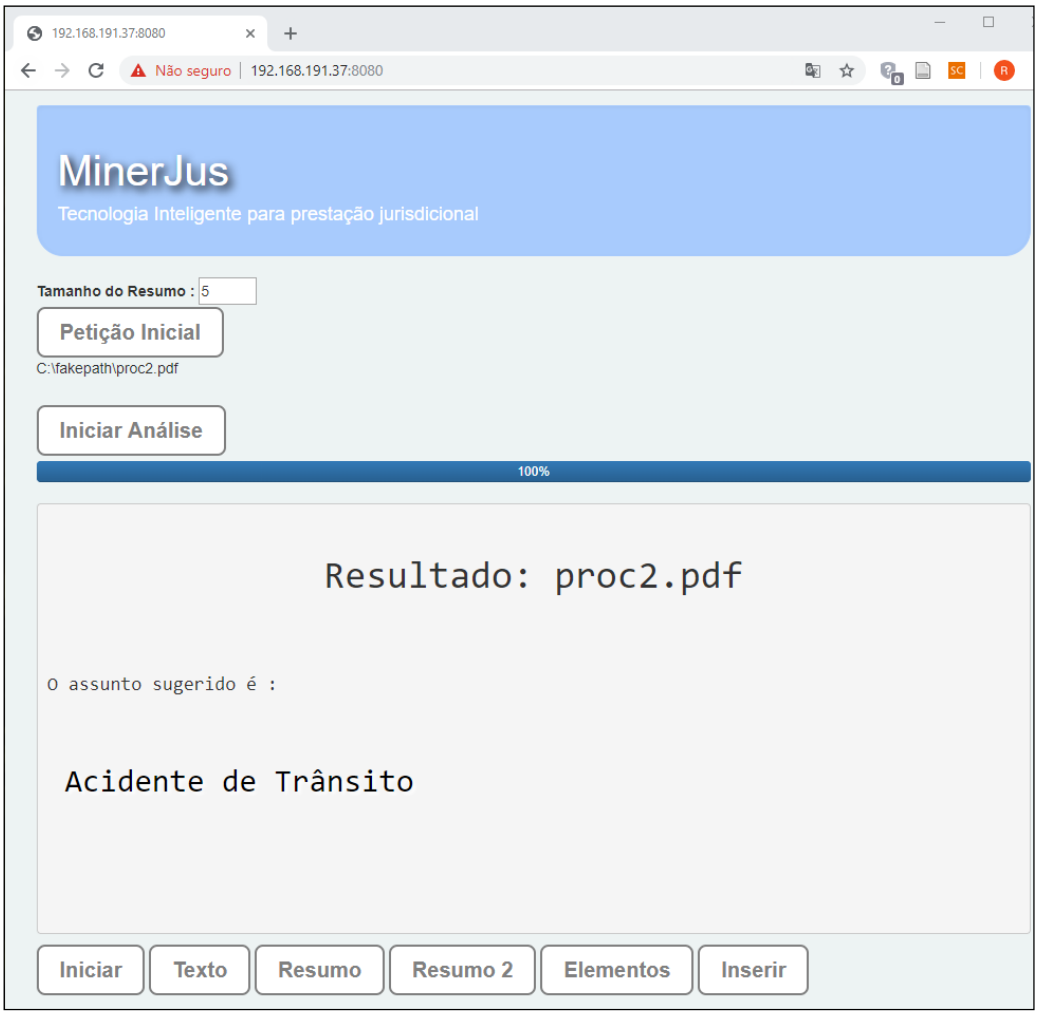
A solução MinerJus foi concebida como um serviço web, que pode ser acoplado a qualquer sistema de processo eletrônico. Mas, para demonstrar sua aplicabilidade foi desenvolvido uma interface WEB capaz de fazer uploads de uma petição inicial (figura 8).

Figura 8: Tela inicial do MinerJus



Ao clicar no botão “Iniciar Análise” (figura 9), a interface envia a petição inicial em formato PDF ao servidor, que inicia a sequência de extração, processamento e classificação de texto, encaminhando o resultado ao usuário pode meio da interface web (figura 10).

Figura 9: Tela com a sugestão de assunto



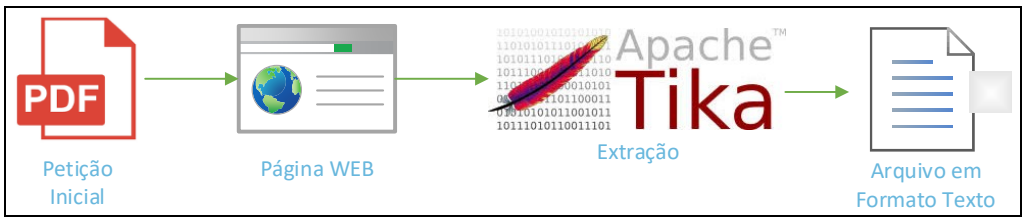
A interface foi desenvolvida em JavaScript e HTML5 com a capacidade de se comunicar com o servidor via API, como mencionado na descrição da arquitetura da solução. Assim, a interface pode ser substituída por qualquer aplicação com capacidade de se conectar às APIs da solução, promovendo, desta forma, um baixo acoplamento, garantindo que a solução possa ser usada em outros ambientes.

3.2.2 Extração do texto

Como dito anteriormente, petições iniciais se encontram em formato PDF. Para que o sistema possa classificar esses documentos quanto ao seu assunto, é necessário processar o texto ali contido. Para processar o texto, primeiramente é preciso extraí-los dos PDFs. Para tal, adotamos um serviço de extração de metadados e textos, denominado Apache Tika. Escrito em Java e mantida pela Apache Software Foundation, o Tika é um software livre capaz de analisar diferentes tipos de arquivos e devolver a informação extraída em texto puro.

Neste projeto, utilizamos a versão 1.19 do Apache Tika, que, operando como um serviço, pode ser acessado pela porta 9998. Assim, quando ocorre o *upload* de uma petição inicial, a solução encaminha o arquivo em formato PDF para o serviço e obtém como resposta o conteúdo do arquivo em formato texto, como na figura 10 :

Figura 10: Processo de extração de texto



3.2.3 Processamento do texto

Alguns arquivos escritos em Python podem conter definições, funções e classes, que são inicializados quando invocados dentro de uma aplicação. Geralmente são abstrações de objetos, denominados de módulos. Para o processamento do texto, o principal módulo usado foi o NLTK (*Natural Language Toolkit*), na versão 3.4, construído para trabalhar com linguagem humana, por meio de técnicas de Processamento de Linguagem (BIRD; KLEIN; LOPER, 2009).

Ao receber os arquivos em formato texto, que deverão ser classificados, a solução submete a um conjunto de técnicas de PLN para processá-los previamente com intuito de abstraí-los para facilitar a compreensão computacional.

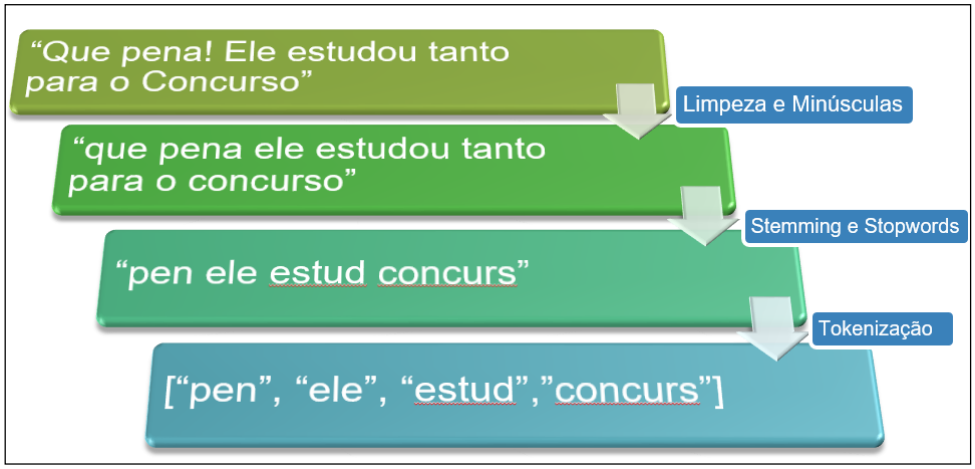
Visando reduzir a complexidade dos textos, são retirados os caracteres especiais das palavras contidas nos mesmos. As acentuações são retiradas para evitar que erros de grafia impactem na interpretação das palavras a serem transformadas em token, além de desconsiderar os números e pontuações.

Os termos remanescentes serão convertidos para letras minúsculas e posteriormente submetidos à uma lista de supressão, conhecida como stopwords. A lista de stopwords é formada por termos comuns da língua que usualmente não apresentam relevância para a classificação do texto. Geralmente, artigos, conjunções, preposições, interjeições, verbos auxiliares e palavras muito repetidas na linguagem natural, compõe a lista de stopwords. Os termos do texto que coincidem com a lista de stopwords são retirados do texto, sem prejuízos ao seu valor semântico, uma vez que as stopwords são necessárias à linguagem natural pelo seu valor sintático.

Para dar mais generalidade aos termos, é efetuado um processamento em cada termo, onde cadeias morfologicamente complexas são identificadas, decompostas em radicais e afixos. Os afixos são descartados e o termo passa a ser apenas o radical, processo conhecido como stemming.

O texto agora é segmentado em tokens, concluindo uma sequência de processamento (Figura 11), gerando uma lista com todos os termos contidos no texto. Tal lista, é um vetor de tokens, que representa o texto. Assim, o corpus passa a ser representado em uma matriz de vetores de tokens.

Figura 11: Sequência de processamento do texto



Buscando representar matematicamente os vetores de tokens, submetemos os mesmos a um algoritmo que atribui a medida estatística TF-IDF, gerada a partir da contagem de ocorrências de um determinado termo dentro de um texto. Normatizado pela quantidade de ocorrências do termo em todo o corpus, gerando, assim, um fator matemático de relevância para cada token contido no texto.

Para realizar as funções de contagem e transformação dos tokens foi utilizado o módulo Scikit-learn, na versão 0.21, sendo um conjunto de ferramentas utilizadas para implementação de aprendizagem de máquina, supervisionado e não supervisionado, com uma interface fácil de usar e estritamente integrado à linguagem Python (PEDREGOSA et al., 2011). Neste caso, o módulo foi utilizado para transformação de atributos do texto com base no TF-IDF, assim as listas de tokens passam a ser representadas por vetores numéricos.

3.2.4 Classificação do texto

De acordo com o teorema “não existe almoço grátis” (WOLPERT; MACREADY, 1997), não há algoritmo melhor que outro, e sim algoritmo com performance melhor do que outro em um conjunto de dados especifico. Logo, para encontrar o algoritmo a ser usado na solução realizamos testes nos seguintes algoritmos, pois os mesmo são implementados por meio da biblioteca Kklearn do Python : Árvore de Decisão, Naiver Bayes, Suport Vector Machine (SVM), K-Nearest Neighbors (KNN).

Foi usado novamente o módulo Scikit-learn, para formulação dos modelos preditores testados, que foram treinados inicialmente com 897 exemplos de petições iniciais, devidamente rotulados pela equipe supervisora, com relação ao assunto.

4. RESULTADOS E ANÁLISE

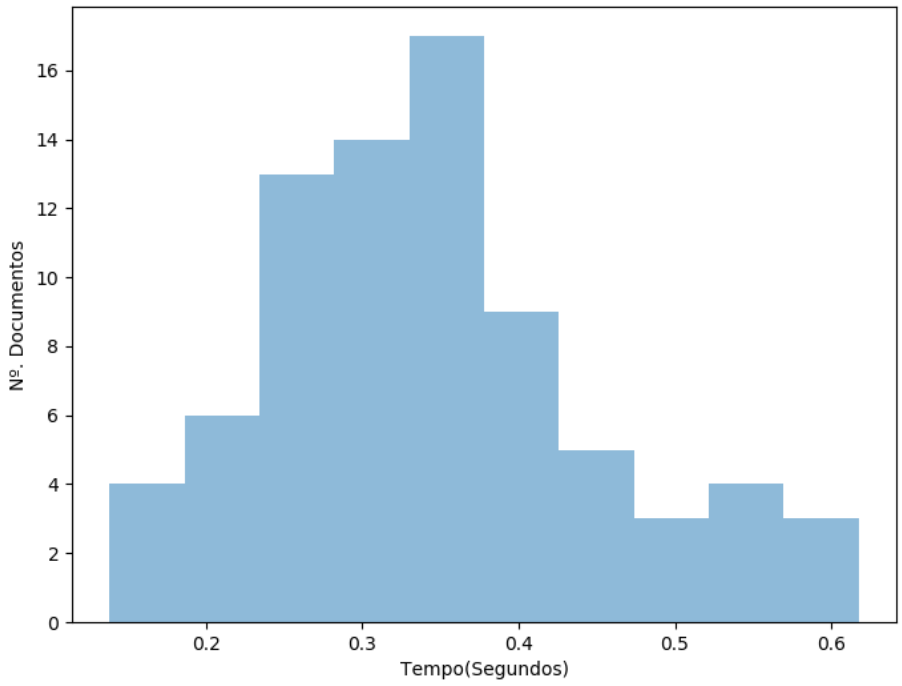
Para avaliação da solução usamos um corpus de teste composto por dados devidamente rotulados, que não foram previamente submetidos ao modelo preditor. Esses dados ajudaram a aferir o desempenho, a precisão e a acurácia do modelo. Ao analisar desta forma pode-se certificar da não existência de sobreajustamento.

Os testes foram realizados em um PC rodando Windows 10 de 64 bits, com um processador Intel Core i5-3570 com velocidade de 3.4 GHz e 24 GB de memória RAM.

4.1 Desempenho

Para aferir o desempenho da solução, foram extraídas aleatoriamente as petições inicias de 78 processos judiciais reais que tramitam no juizado especial cível da comarca de Augustinópolis, formando uma base de testes e validações. Esses processos possuem seus dados dispostos no processo eletrônico e-Proc do Tribunal de Justiça do Tocantins configurados como não sigilosos.

Tais documentos foram submetidos um a um à solução, sendo checado o tempo de resposta, compreendido entre a submissão da petição inicial em formato PDF e a resposta com a classificação predita (sugestão). Com intuito de conhecer o tempo resposta da solução, os tempos de resposta foram dispostos no histograma (figura 12).



O tempo de resposta está diretamente ligado ao número de páginas do documento, apresentando neste caso um tempo médio de 0,041 segundos gastos por página.

Em torno de 90% das petições iniciais analisadas apresentaram tempo de resposta abaixo dos 0,5 segundos. O tempo de resposta médio da solução diante de todos os documentos que compõe o corpus de treinamento foi de 0,34 segundos por documento, sendo o tempo máximo encontrado de 0,61 segundos.

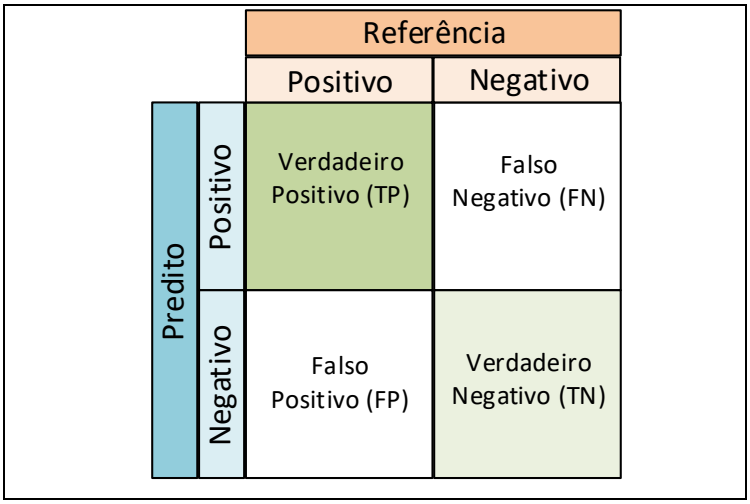
4.2 Acurácia e Precisão

A acurácia, é a mais simples métrica que pode ser usada para avaliar um classificador, mede a porcentagem dos dados classificados corretamente (BIRD; KLEIN; LOPER, 2009), calculado por meio da somatória das classificações corretas divido pela quantidade de documentos do corpus de testes (ZAKI; MEIRA, JR, 2014).

De acordo com o tipo de problema, outras métricas podem ser usadas. Por exemplo, em problemas que apresentam uma base de teste muito desbalanceada, como a tarefa de classificar se um documento é relevante ou não no ato de uma busca, logicamente a quantidade de documentos irrelevantes se aproxima de 100% (BIRD; KLEIN; LOPER, 2009). Assim, um modelo que aferir que nenhum documento tem relevância, apresentará uma acurácia perto de 100%.

A matriz de confusão (Figura 7) é uma técnica usada para de avalições com base nos acertos e erros de predições, trata-se de uma matriz *NxN*, onde *N* é número de classes contidas no conjunto de teste, cada célula[*i,j*], indica quantas classes *i* foram preditas enquanto o certo seria *j*. Logo a diagonal desta matriz informa quantos acertos ocorreram e o que não está na diagonal são os erros (BIRD; KLEIN; LOPER, 2009). Em outras palavras, as linhas representam as classes informadas pelo classificador, e as colunas são classes de referência, advindas do conjunto do teste.

Figura 13: Matriz de confusão



Verdadeiros positivos (TP) é a quantidade de classificações positivas corretas.

Verdadeiros negativos (VN) é o total de classificações negativas corretas.

Falsos positivos (FP) é a porção de classificação positivas incorretas.

Falsos negativos (FN) é formado pela quantidade de classificações negativas incorretas.

Quando se quer saber qual a proporção de classificações positivas que estão corretas, usamos a métrica precisão, denotado por *P* e dada (4):

$$P = \frac{TP}{(TP+FP)} \tag{4}$$

Outra métrica é a cobertura (recall) que afere a proporção entre as possíveis classificação positivas e quantas foram classificadas corretamente, denotado por *R* e calculada por (5):

$$R = \frac{TP}{(TP+FN)} \tag{5}$$

Uma forma de combinar a precisão e o recall para prover um índice único é usando a média harmônica entre a precisão e o recall (INDURKHYA; DAMERAU, 2010). Esse índice é conhecido como *F1 score* (3).

$$F1 = \frac{2 \cdot P \cdot R}{(P+R)} \tag{6}$$

Para escolher o algoritmo de aprendizagem que irá compor a solução, selecionamos aleatoriamente 78 petições iniciais (pdf) de processos que tramitam no juizado especial cível da comarca de Augustinópolis – TO, do Tribunal de Justiça do Tocantins, e submetemos aos algoritmos de aprendizagem com intuito de predizer os assuntos. A tabela1 apresenta a precisão,

acurácia e o tempo de treinamento de cada algoritmo avaliado:

Tabela 1: Avaliação dos algoritmos de aprendizagem supervisionada

Classificador	**Precisão**	**Acurácia**	**Tempo de Treinamento(S)**
Árvore de Decisão	57,89%	79,48%	16,42s
Naiver Bayes	53,78%	57,69%	36,02s
Suport Vector Machine	**72,72%**	**93,58%**	167,81s
K-Nearest Neighbors	51%	82,05%	**2,11s**

Como observado na Tabela 1, o algoritmo que mostrou melhores índices de precisão e acurácia foi o Support Vector Machine SVM, mas com um tempo de treinamento maior que todos os outros.

Na Tabela 2, se apresenta de forma detalhada a precisão, recall e F1-Score por assuntos referentes às petições iniciais.

Tabela 2: Análise detalhada por assunto

Assunto	Precisão	Recall	F1-scores
Acidente de Trânsito	1.00	1.00	1.00
Assinatura Básica Mensal	1.00	1.00	1.00
Cancelamento de Voo	1.00	1.00	1.00
Cartão de Crédito	1.00	1.00	1.00
Cheque	1.00	1.00	1.00
Compromisso	0.00	0.00	0.00
Dever de informação	1.00	1.00	1.00
Direito de imagem	0.00	0.00	0.00
Direito de vizinhança	1.00	1.00	1.00
Empreitada	0.00	0.00	0.00
Empréstimo Conseguinado	1.00	1.00	1.00
Fornecimento de energia elétrica	1.00	1.00	1.00
Fornecimento de água	1.00	1.00	1.00
Inclusão indevida cadastro de inadimplentes	1.00	0.95	0.97
Locação de imóveis	0.00	0.00	0.00
Nota Promissória	0.00	0.00	0.00
Obrigação de fazer/não fazer	1.00	1.00	1.00
Oferta e publicidade	1.00	1.00	1.00
Perdas e Danos	1.00	1.00	1.00
Rescisão de contrato e devolução do dinheiro	1.00	1.00	1.00
Substituição e produto	1.00	1.00	1.00
Tarifas	1.00	1.00	1.00

Alguns assuntos como “Compromisso” e “Direito de imagem“, embora possuam

exemplos dentro da base de treinamento, apresentaram o resultado 0, apontado para a possibilidade que os exemplos não são adequados ou a quantidade não é suficiente, sendo necessário uma avalição pontual de cada assunto, visando aprimorar a base de treinamento.

Diante dos testes, podemos dizer que a solução apresentou uma acurácia de 93,58%, quando testado com um conjunto de documentos desconhecido pela máquina de aprendizagem e uma precisão de 72,72%, nesse mesmo conjunto de dados.

Após os testes (tabela 1), para implantação de uma aprendizagem de máquina capaz de predizer o assunto de uma petição inicial, foi utilizado o algoritmo que faz uso da abordagem de aprendizado supervisionada SVM (Suporte Vector Machine), pois além de se mostrar superior aos demais algoritmos testados, este oferece vantagens, que o torna atraente para classificação de textos, onde na sua implementação são usadas técnicas de classificação linear e discriminatória. Assim, o SVM é capaz de lidar com problemas de alta dimensionalidade e dados esparsos (JOACHIMS, 1998). Sendo estas características importantes para se trabalhar com a classificação de grandes coleções de documentos (D'ORAZIO et al., 2014). O tempo de aprendizagem do SVM se apresentou consideravelmente mais elevado que os demais, devido a sua maior complexidade de cálculos que geram hiperplanos separadores, refletindo diretamente na melhor capacidade de generalização.

5. CONCLUSÃO

Esse trabalho apresenta o desenvolvimento de uma solução que visa garantir maior confiabilidade à classificação dos processos judiciais no ato do cadastro da petição inicial com relação ao assunto, por meio do uso de técnicas de processamento de linguagem natural para extração de informações contidas em documentos em formato pdf. Foi construída uma máquina de aprendizagem capaz de predizer automaticamente qual o assunto do processo.

Atualmente não existe nenhuma ferramenta que auxilie os operadores do Direito na tarefa de classificação do processo, o que ocasiona várias inconsistências nos dados de classificação dos processos, a exemplo do Tribunal de Justiça da Bahia (TJBA), que analisou 404,3 mil processos e identificou que 56% apresentavam erros no cadastro da petição inicial, sendo que dos analisados, 176.598 apresentam falhas na classificação com relação ao assunto, representado 78% dos erros encontrados.

Ao obter uma acurácia de 93,58% e um tempo médio de resposta menor que meio segundo, embora o tempo de treinamento seja elevado, a solução comprova sua viabilidade e capacidade de apoiar os operadores do direito. No caso do TJBA a classificação realizada por humanos sem apoio tecnológico chega a uma taxa de acerto de 56% com relação ao assunto, e os treinamentos podem ser realizados em horários em que a infraestrutura esteja mais ociosa, podendo ser deslocado mais poder de processamento para tal tarefa sem prejuízo para os usuários do sistema.

Desta forma, a solução MinerJus por meio da implementação de técnicas inteligência artificial, tem a capacidade de mitigar problemas no ato da classificação processual, bem como imprimir maior agilidade na tramitação processual ao sugerir automaticamente o assunto do processo judicial.

5.1 Contribuições

Esta dissertação apresentou uma solução desenvolvida totalmente utilizando tecnologias de código aberto, garantindo flexibilidade, interoperabilidade, confiabilidade, e baixo custo de desenvolvimento e manutenção. Devidamente efetiva, capaz de promover agilidade na tramitação processual, mitigando o retrabalho, e melhorando a qualidade das informações contidas nos processos judiciais, proporcionando, assim, uma contribuição significativa para a prestação jurisdicional.

5.2 Perspectivas

Uma característica intrínseca das aprendizagens de máquina é a capacidade de aprender. Assim sendo, para melhorar os índices de acurácia e precisão atuais do software deve-se “ensiná-la” continuamente. No caso especifico da precisão, é primordial melhorar os exemplos e/ou ampliar a base de treinamento priorizando aqueles assuntos que apresentaram baixa precisão.

Além de melhorar sua assertividade a solução pode ser adaptada para solução de outros problemas, como a predição da classe processual com base nos dados devidamente extraídos da petição inicial a ser classificada.

Diante de um cenário com maior confiabilidade dos dados de classificação do processo, outras soluções podem fazer uso de tais informações. Como, por exemplo, uma ferramenta capaz de classificar o processo quando da sua tramitação processual, com capacidade de ler os atributos do processo e informar se um processo está em fase de execução ou conhecimento.

Outra possibilidade a ser explorada é a construção de soluções capazes de apoiar os magistrados em suas decisões, como a elaboração de um banco de sentenças inteligente, que sugere o texto da decisão ao magistrado, considerando as informações extraídas do processo até aquele momento. E, o histórico de decisões exaradas anteriormente naquele contexto, promovendo agilidade processual e uniformidade nas decisões.

5.3 Publicações

Durante o período do curso de mestrado os seguintes trabalhos foram apresentados:

5.3.1 Publicados

- NOGUEIRA, R.; ARAÚJO, H.; PRATA, D. Robot Chow: Automatic Animal Feeding with Intelligent Interface to Monitor Pets. **International Journal of Advanced Engineering Research and Science**, v. 6, p. 262–267, 1 jan. 2019.

- NOGUEIRA, R.; FERREIRA, L.;RAMOS,J; PRATA, D. Minerjus: solution to the processual classification with use of artificial intelligence. **International Journal of Development Research**, v.09, p. 28642-28646, jul. 2019.

5.3.2 Aceito para publicação

- PRATA, D ; ROCHA, M; FERREIRA, L; NOGUEIRA, R. Geospatial Dimension in Association Rule Mining: The Case Study of the Amazon Charcoal Tree. **5th International Conference, LOD 2019**, set. 2019.

REFERÊNCIAS

AGUIAR, E. M. de. Aplicação do Word2vec e do Gradiente descendente dstocástico em tradução automática. 30 maio 2016. Disponível em: <http://bibliotecadigital.fgv.br/dspace/handle/10438/16798>. Acesso em: 9 abr. 2019.

AKBANI, R.; KWEK, S.; JAPKOWICZ, N. Applying Support Vector Machines to Imbalanced Datasets. (J.-F. Boulicaut et al., Eds.) In: Machine Learning: ECML 2004, **Anais**...Springer Berlin Heidelberg, 2004.

ALPAYDIN, E. **Introduction to machine learning**. 2nd ed ed. Cambridge, Mass: MIT Press, 2010.

BANERJEE, S.; PEDERSEN2, T. The Design, Implementation, and Use of the Ngram Statistics Package. **In International Conference on Intelligent Text Processing and Computational Linguistics**, p. 370–381, 2003.

BIRD, S.; KLEIN, E.; LOPER, E. **Natural Language Processing with Python**. 1. ed. Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472: Julie Steele, 2009.

BISHOP, C. M. **Pattern recognition and machine learning**. New York: Springer, 2006.

BRASIL. LEI N º 13.105 DE 16 DE MARÇO DE 2015. **Código Processual Civil**, 2015. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13105.htm>. Acesso em: 19 maio. 2019.

BZDOK, D.; KRZYWINSKI, M.; ALTMAN, N. Points of Significance: Machine Learning: Supervised Methods. **Nature Methods**, v. 15, p. 5–6, 3 jan. 2018.

CINTRA, A. C. de A.; DINAMARCO, C. R.; GRINOVER, A. P. **Teoria Geral do Processo**. 26ª ed. São Paulo: Malheiros, 2010.

COPPIN, B. **Inteligência Artificial**. Rio de Janeiro: LTC, 2017.

COVER, T.; HART, P. Nearest neighbor pattern classification. **IEEE Transactions on Information Theory**, v. 13, n. 1, p. 21–27, jan. 1967.

CRISTIANINI, N.; SCHOLKOPF, B. Support Vector Machines and Kernel Methods: The New Generation of Learning Machines. **AI Magazine**, v. 23, n. 3, p. 31–31, 15 set. 2002.

DOMINGOS, P.; PAZZANI, M. On the Optimality of the Simple Bayesian Classifier under Zero-One Loss. **Machine Learning**, v. 29, n. 2, p. 103–130, 1 nov. 1997.

FARIA, E. L. de. REDES NEURAIS CONVOLUCIONAIS E MÁQUINAS DE APRENDIZADO EXTREMO APLICADAS AO MERCADO FINANCEIRO BRASILEIRO. **Programa de Pós-graduação em Engenharia Civil, COPPE, da Universidade Federal do Rio de Janeiro**, set. 2018.

HARRIS, Z. S. Distributional structure. **Word**, v. 10, p. 146–162, 1954.

INDURKHYA, N.; DAMERAU, F. J. **Handbook of Natural Language Processing**. 2nd. ed. [s.l.] Chapman & Hall/CRC, 2010.

JAIN, A. K. **Algorithms for clustering data**. [s.l.] Englewood Cliffs, N.J.: Prentice Hall, 1988.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, v. 22, n. 1, p. 4–37, 2000.

JONES, K. S. A statistical interpretation of term specificity and its application in retrieval. **Journal of Documentation**, v. 28, p. 11–21, 1972.

JÚNIOR, H. T. **Curso de Direito Processual Civil**. 56ª Edição ed. Rio de Janeiro: Editora Forense, 2015. v. Volume I

KUMAR, Y.; SAHOO, G. Analysis of Parametric & Non Parametric Classifiers for Classification Technique Using WEKA. **International Journal of Information Technology and Computer Science**, v. 4, n. 7, p. 43–49, 1 jul. 2012.

LANE, H.; HOWARD, C.; HAPKE, H. M. **Natural Language Processing in Action**. 3. ed. [s.l.] Manning Publications Co., 2017.

LUO, C.; LI, Y.; CHUNG, S. M. Text document clustering based on neighbors. **Data & Knowledge Engineering**, Including Special Section: Conference on Privacy in Statistical Databases (PSD 2008) – Six selected and extended papers on Database Privacy. v. 68, n. 11, p. 1271–1288, 1 nov. 2009.

MIKOLOV, T. et al. Efficient Estimation of Word Representations in Vector Space. **arXiv:1301.3781 [cs]**, 16 jan. 2013. Disponível em: <http://arxiv.org/abs/1301.3781>. Acesso em: 10 abr. 2019.

MITCHELL, T. M. **Machine Learning**. New York: McGraw-Hill, 1997.

MURPHY, K. P. **Machine learning: a probabilistic perspective**. Cambridge, MA: MIT Press, 2012.

PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, v. 12, n. Oct, p. 2825–2830, 2011.

PERKINS, J. **Python Text Processing with NLTK 2.0 Cookbook**. 32 Lincoln Road Olton Birmingham, B27 6PA, UK.: Packt Publishing Ltd., 2010.

PYTHON.ORG. **The Python Tutorial - Documentação do Python 3.7.1**. Disponível em: <https://docs.python.org/3/tutorial/index.html>. Acesso em: 18 nov. 2018.

QUINLAN, J. R. Induction of Decision Trees. **Machine Learning**, v. 1, n. 1, p. 81–106, mar. 1986.

SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. **Information Processing & Management**, v. 24, n. 5, p. 513–523, 1 jan. 1988.

SALTON, G.; WONG, A.; YANG, C. S. A Vector Space Model for Automatic Indexing. **Communications of the ACM**, v. 18, n. 11, p. 613–620, 1 nov. 1975.

SEN, I.; WEISS, G. **Multiagent Systems**. [s.l: s.n.]

SOUSA, R. T. Avaliação de classificadores na classificação de radiografias de tórax para o diagnóstico de pneumonia infantil. p. 63, 2013.

SUN, H.; CRAIG, B. A.; ZHANG, L. Angle-based Multicategory Distance-weighted SVM. **Journal of Machine Learning Research**, v. 18, n. 85, p. 1–21, 2017.

SZYMAŃSKI, P.; KAJDANOWICZ, T. scikit-multilearn: A Python library for Multi-Label Classification. **Journal of Machine Learning Research**, v. 20, n. 6, p. 1–22, 2019.

TJTO. **INSTRUÇÃO NORMATIVA Nº 5, DE 24 DE OUTUBRO DE 2011.** Disponível em: <http://wwa.tjto.jus.br/elegis/Home/Imprimir/423>. Acesso em: 9 nov. 2018.

TURNEY, P. D.; PANTEL, P. From Frequency to Meaning: Vector Space Models of Semantics. **Journal of Artificial Intelligence Research**, v. 37, p. 141–188, 27 fev. 2010.

VAPNIK, V. **The Nature of Statistical Learning Theory**. 2. ed. New York: Springer-Verlag, 2000.

WEN, Z. et al. ThunderSVM: A Fast SVM Library on GPUs and CPUs. **Journal of Machine Learning Research**, v. 19, n. 21, p. 1–5, 2018.

WOLPERT, D. H.; MACREADY, W. G. No free lunch theorems for optimization. **IEEE Trans. Evolutionary Computation**, v. 1, p. 67–82, 1997.

ZAKI, M. J.; MEIRA, JR, W. **Data Mining and Analysis: Fundamental Concepts and Algorithms**. 1. ed. [s.l.] Cambridge University Press, 2014.

APÊNDICE A – Lista de processos usados no corpus de teste

Tabela abaixo dispõe todos os processos usados no corpus de teste com número do processo, número de páginas da petição inicial, tamanho do arquivo da petição inicial e tempo compreendido entre a submissão do documento, a ferramenta e a resposta ao usuário quanto a classificação sugerida pela solução MinerJus.

Número de processo	**Nº de Paginas**	**Tamanho(KB)**	**Tempo(s)**
5000911-98.2012.827.2710	9	122,06	0.320
0001200-43.2017.827.2710	5	113,82	0.1917
0003947-63.2017.827.2710	9	473,80	0.3486
0002588-15.2016.827.2710	8	499,36	0.3484
0001199-63.2014.827.2710	5	126,53	0.2167
0003342-54.2016.827.2710	6	136,82	0.2715
0005228-54.2017.827.2710	17	441,02	0.4338
0006204-61.2017.827.2710	9	448,23	0.3198
0001493-13.2017.827.2710	17	769,18	0.5512
0002347-07.2017.827.2710	17	750,01	0.5876
0001396-76.2018.827.2710	11	428,75	0.3684
0002708-92.2015.827.2710	6	582,06	0.2691

0000926-45.2018.827.2710	8	762,88	0.3955
0001372-48.2018.827.2710	11	420,06	0.3494
0002239-75.2017.827.2710	11	247,31	0.3091
0000284-72.2018.827.2710	19	430,38	0.5021
0002098-90.2016.827.2710	14	265,55	0.3086
0001246-95.2018.827.2710	11	420,30	0.3711
0002538-86.2016.827.2710	4	139,60	0.1726
0002313-03.2015.827.2710	5	260,47	0.2704
0005770-72.2017.827.2710	8	654,39	0.3449
0001614-07.2018.827.2710	13	123,30	0.4435
0001441-80.2018.827.2710	10	714,60	0.3663
0006230-59.2017.827.2710	12	258,18	0.3067
0000350-23.2016.827.2710	4	172,83	0.4184
0004937-88.2016.827.2710	8	400,31	0.2696
0001536-81.2016.827.2710	5	326,81	0.2089

0001443-84.2017.827.2710	19	767,54	0.5906
0001040-81.2018.827.2710	18	381,41	0.4537
0001742-95.2016.827.2710	8	345,13	0.3665
0005312-89.2016.827.2710	2	109,63	0.1596
0005495-26.2017.827.2710	14	237,45	0.3878
0005226-84.2017.827.2710	17	441,92	0.4461
0006533-73.2017.827.2710	9	563,99	0.3373
0001567-04.2016.827.2710	7	135,03	0.2580
0004889-32.2016.827.2710	10	497,77	0.3729
0006861-03.2017.827.2710	10	459,11	0.3164
0006748-49.2017.827.2710	12	232,04	0.4109
0002589-97.2016.827.2710	8	499,15	0.3635
0001113-87.2017.827.2710	5	426,29	0.3150
0003132-03.2016.827.2710	7	195,29	0.2355
0002898-84.2017.827.2710	4	89,81	0.2155

0002739-15.2015.827.2710	3	508,45	0.2010
0002457-40.2016.827.2710	9	400,13	0.3467
0004936-06.2016.827.2710	8	400,32	0.2469
0002709-09.2017.827.2710	10	447,35	0.2972
0006532-88.2017.827.2710	9	564,01	0.3643
0004713-19.2017.827.2710	8	524,58	0.3808
0003397-05.2016.827.2710	6	594,26	0.2717
0001442-02.2017.827.2710	17	750,57	0.5645
0000001-49.2018.827.2710	11	656,46	0.3165
0002696-10.2017.827.2710	7	246,07	0.2368
0003569-44.2016.827.2710	6	133,98	0.2170
0006113-68.2017.827.2710	8	425,35	0.35304
0002072-58.2017.827.2710	14	236,67	0.3150
0002300-67.2016.827.2710	12	884,53	0.4851
0003079-56.2015.827.2710	6	576,23	0.2603

0006550-12.2017.827.2710	10	615,88	0.3947
0003786-53.2017.827.2710	8	182,64	0.2722
0000316-77.2018.827.2710	1	325,32	0.1621
0002384-34.2017.827.2710	17	768,17	0.61762
0004218-72.2017.827.2710	17	435,74	0.4163
0002610-73.2016.827.2710	7	205,71	0.2493
0006531-06.2017.827.2710	8	563,78	0.3048
0000401-63.2018.827.2710	17	441,00	0.4766
0006423-74.2017.827.2710	16	486,49	0.3715
0003034-18.2016.827.2710	17	393,15	0.5633
0006644-57.2017.827.2710	1	339,98	0.1382
0007073-24.2017.827.2710	6	343,38	0.2860
0000555-18.2017.827.2710	18	306,62	0.4118
0006534-58.2017.827.2710	9	564,25	0.31686
0002113-59.2016.827.2710	9	257,03	0.4114

0001377-70.2018.827.2710	11	419,94	0.3600
0002587-30.2016.827.2710	8	499,69	0.33975
0002715-50.2016.827.2710	17	389,90	0.5298
0001385-47.2018.827.2710	11	428,27	0.4364
0001582-07.2015.827.2710	9	161,23	0.2462
0003332-10.2016.827.2710	7	596,77	0.2853

Os processos podem ser consultados no link:
https://consultaeproc.tjto.jus.br/eprocV2_prod_1grau/externo_controlador.php?acao=processo_consulta_publica