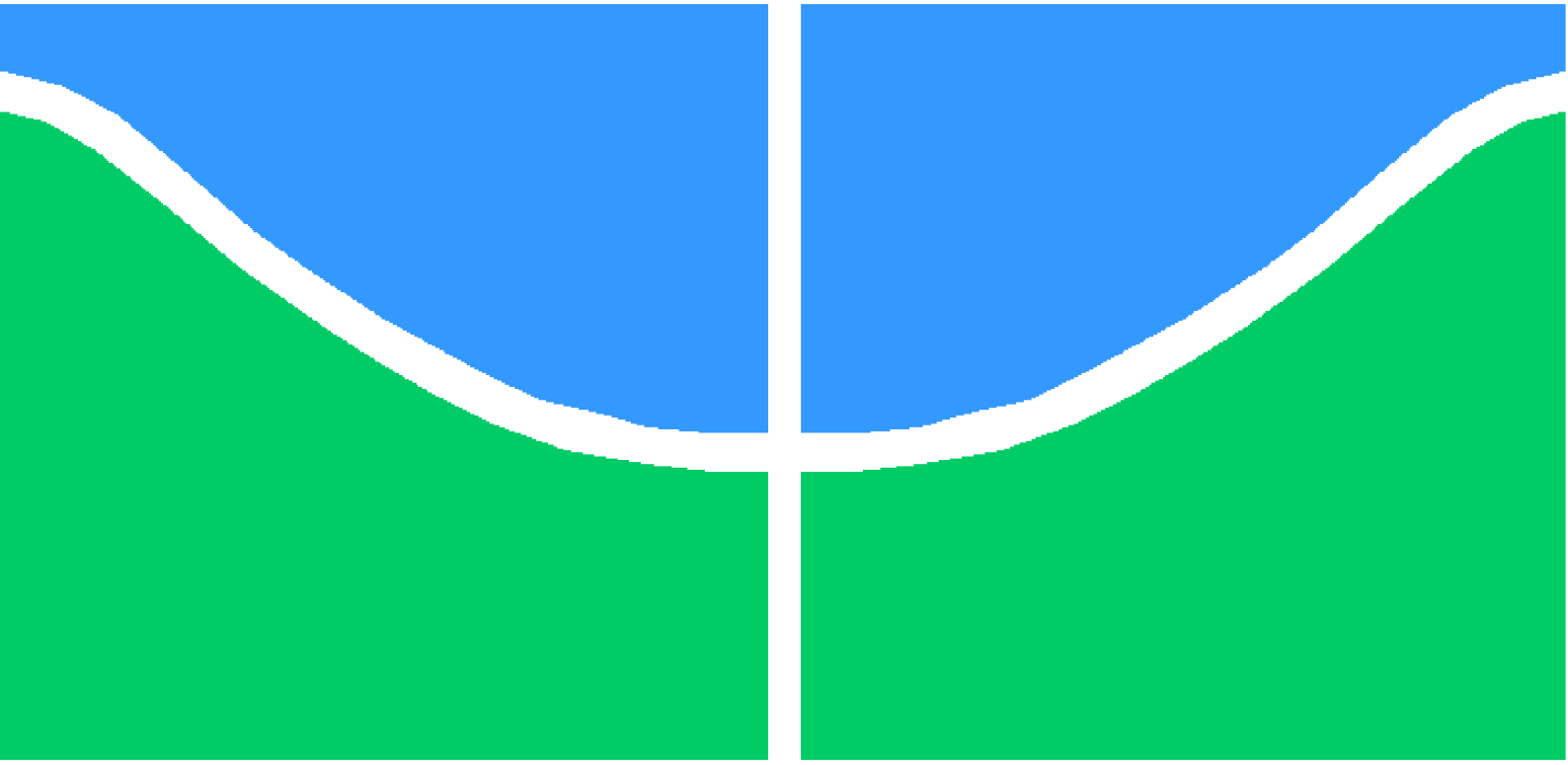

Universidade de Brasília - UnB
Faculdade UnB Gama - FGA
Engenharia de Software

Classificação de peças processuais jurídicas: inteligência artificial no direito

Autor: Marcelo Herton Pereira Ferreira
Orientador: Doutor Nilton Correia da Silva

Brasília, DF
2018

Marcelo Herton Pereira Ferreira

Classificação de peças processuais jurídicas: inteligência artificial no direito

Monografia submetida ao curso de graduação em (Engenharia de Software) da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em (Engenharia de Software).

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Doutor Nilton Correia da Silva

Brasília, DF

2018

Marcelo Herton Pereira Ferreira
Classificação de peças processuais jurídicas: inteligência artificial no direito/ Marcelo Herton Pereira Ferreira. – Brasília, DF, 2018-
68 p. : il. (algumas color.) ; 30 cm.

Orientador: Doutor Nilton Correia da Silva

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2018.

1. Aprendizado de Máquina. 2. Classificação de documentos. I. Doutor Nilton Correia da Silva. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Classificação de peças processuais jurídicas: inteligência artificial no direito

CDU 02:141:005.6

Marcelo Herton Pereira Ferreira

Classificação de peças processuais jurídicas: inteligência artificial no direito

Monografia submetida ao curso de graduação em (Engenharia de Software) da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em (Engenharia de Software).

Trabalho aprovado. Brasília, DF, 01 de junho de 2018:

Doutor Nilton Correia da Silva
Orientador

Doutor Fabricio Ataides Braz
Convidado 1

Mestre Tainá Aguiar Junquilho
Convidado 2

Brasília, DF
2018

Resumo

O Supremo Tribunal Federal tem a necessidade de separar as peças processuais jurídicas para facilitar a distribuição do processo internamente. Atualmente esta separação de volumes em peças e a classificação destas são feitas manualmente por uma equipe. A metodologia para o trabalho é a explorativa, utilizando métodos indutivos. O processo de desenvolvimento é baseado no de pesquisa e desenvolvimento de aprendizado de máquina, em que utilizou-se este dois para elaborar um processo único. No referencial teórico, descreveram-se técnicas para tratamento de texto, transformação de texto em representação computacional, modelos de aprendizado de máquina para classificação de documentos. O uso dessas técnicas proporcionaram analisar o conjunto de dados e detectar que haviam peças de tipos diferentes com o mesmo conteúdo de texto. Bem como identificar correlações entre as classes do problema. Como conclusão, para classificar as peças é necessário remover os multi-rótulos além de que é viável utilizar técnicas de aprendizado de máquina para a solução do problema devido a baixa correlação.

Palavras-chaves: Classificação de documentos. Aprendizado de máquina. Peças jurídicas.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

AI	Inteligência Artificial
ARE	Agravo em Recurso Extraordinário
BoW	*Bag of words*
CNJ	Conselho Nacional de Justiça
CNN	Rede neural convolucional
CPC	Código processual civil
CRISP-DM	*Cross-industry standard process for data mining*
ML	Aprendizado de máquina
NLP	Processamento de linguagem natural
OCR	Reconhecedor Ótico de Caracteres
PDF	*Portable Document Format*
RE	Recurso Extraordinário
RNN	Rede neural recorrente
STF	Superior Tribunal Federal
STJ	Superior Tribunal de Justiça
STM	Superior Tribunal Militar
SVM	Máquinas de suporte vetorial
TSE	Tribunal Superior Eleitoral
TST	Tribunal Superior do Trabalho

Lista de símbolos

$\odot$	Operação de Hadamard
δ	Delta
θ	Theta
$\hat{y}$	Valor predito em uma classificação
Σ	Somatório

Sumário

1 Introdução

Técnicas de Aprendizado de Máquina (do inglês *Machine Learning* ML) tem sido amplamente adotadas no mundo da computação para realizar tarefas em que há muitos dados disponíveis e existe algum tipo de relação entre eles. De forma que seja possível escrever um algoritmo para processá-lo (BRINK; RICHARDS; FETHEROLF, 2015).

Pode-se dizer que o aprendizado de máquina é uma vertente da Inteligência Artificial (do inglês *Artificial artificial intelligence* AI), a qual possui um corpo de conhecimento específico ou um conjunto de técnicas para que a máquina possa aprender com os dados. Enquanto que a AI é um campo de estudo muito mais abrangente que engloba os campos de visão computacional, processamento de linguagem natural, robótica e outros (BRINK; RICHARDS; FETHEROLF, 2015).

Outro campo dentro da AI é o Processamento de Linguagem Natural (do inglês *Natural Language Processing* NLP), o qual é extremamente importante para aplicações modernas, pois trata-se da interpretação dos constructos da linguagem humana para serem processadas por computadores (GOLDBERG, 2017). Com a digitalização dos documentos, viabilizou-se o uso de técnicas, métodos para facilitar análises e extração de informações de conteúdos (OLIVEIRA; FILHO, 2017).

Alguns dos conjuntos de técnicas para o NLP são o diálogo ou sistemas de fala, análises textuais e recuperação de informações através de perguntas e respostas (ESLICK; LIU, 2005). As técnicas, que são úteis para lidar com documentos digitais e facilitar a separação de conteúdo destes, são a classificação de documentos, busca e recuperação de informações (OLIVEIRA; FILHO, 2017).

Ainda que haja um grande esforço nas áreas específicas para as categorias de técnicas de NLP, é inerente a todas elas as dificuldades encontradas em fazer com que a máquina consiga lidar com a enorme variabilidade da linguagem natural, ambiguidade e usos complexos da língua como figuras de linguagens (GOLDBERG, 2017).

O Supremo Tribunal Federal (STF) é um órgão público da esfera de Poder Jurídico do Brasil. A ele compete realizar a guarda da Constituição (1988), no qual julga casos em que os Artigos da Constituição possam ser violados ou mal interpretados, conflitos entre a União e os Estados incluindo o Distrito Federal, ações movidas por estados estrangeiros, razões contra o Conselho Nacional de Justiça (CNJ), conflitos entre os tribunais (BRASIL, 1988).

Além disso, cabe a ele julgar em recurso ordinário crimes políticos, *habeas corpus*, mandados de segurança e, em recurso extraordinário, infrações a Constituição, incons-

titucionalidades em tratados ou em leis, invalidar atos do governo caso contrariem a Constituição (BRASIL, 1988).

O Governo brasileiro implantou a política de transparência pública, no qual todas as informações públicas devem ser disponibilizadas para que os cidadão pudessem acessar qualquer informação sobre as três esferas do poder: Judiciário, Executivo e Legislativo (BRASIL, 2011).

O sistema judiciário, assim como os outros dois poderes, teve um esforço para implantar todo o acesso a informação por meios eletrônicos, visto que a população em geral, quando quer buscar algo, utiliza os sites de busca como o Google [1] e DuckDuckGo [2] (RUSCHEL; ROVER; SCHNEIDER, 2011).

Logo então, o foco do sistema judiciário brasileiro foi no desenvolvimento dos *web site* de cada tribunal e na digitalização dos processos. O CNJ tem como uma de suas competências coordenar e auxiliar a digitalização de todos os 91 tribunais de justiça (RUSCHEL; ROVER; SCHNEIDER, 2011). Ele colocou como metas para o desenvolvimento do parque tecnológico, a de informatizar todas as unidades judiciárias, interligá-las e implantar o processo eletrônico em parcela de suas unidades judiciárias. Com estas metas, esperava-se que todos os tribunais pudessem realizar a tramitação de processos por meio digital (BRASIL, 2009). Também, que o processo fosse representado da mesma forma em diferentes sistemas, para que a população pudesse ter acesso a eles através de mecanismos de consulta online (RUSCHEL; ROVER; SCHNEIDER, 2011).

Com isso, surgiu uma grande variabilidade de sistemas. Mesmo com esforços do Conselho de Justiça para realizar a unificação com o Modelo Nacional de Interoperabilidade (BRASIL, 2009) e a adoção do sistema PJe. Muitos tribunais usam diferentes sistemas como PJe, Projudi e e-SAJ.

O Processo Judicial eletrônico (PJe) foi um dos sistemas desenvolvidos, o qual passou a ser adotado por diversos tribunais como uso obrigatório. O seu principal objetivo é manter um sistema que possibilite a transição dos processos, bem como o registro de peças processuais, além do acompanhamento de todos os acontecimentos independentemente do tribunal em que ele tramite. A proposta de sua solução é que ele fosse único para todos os tribunais (BRASIL, 2018).

Grande parte do problema envolto com a grande variabilidade destes sistemas e que dificultam sua implementação, é o fato destes sistemas ainda seguirem a lógica de processos físicos. Tal qual a classificação do tipo de peça jurídica, a montagem de volumes, a forma de peticionamento, a tramitação, os meta-dados, até o fato de que ainda existem muitos processos que são digitalizados. A Figura 1 exibe um desses processos, os quais são cópias digitais de processos físicos e que podem estar sujeitos a problemas de identificação

[1] Site :<https://google.com>
[2] Site: <https://duckduckgo.com>

do conteúdo, furos, manchas, desalinhamento da página e suas incorretas ordenações.

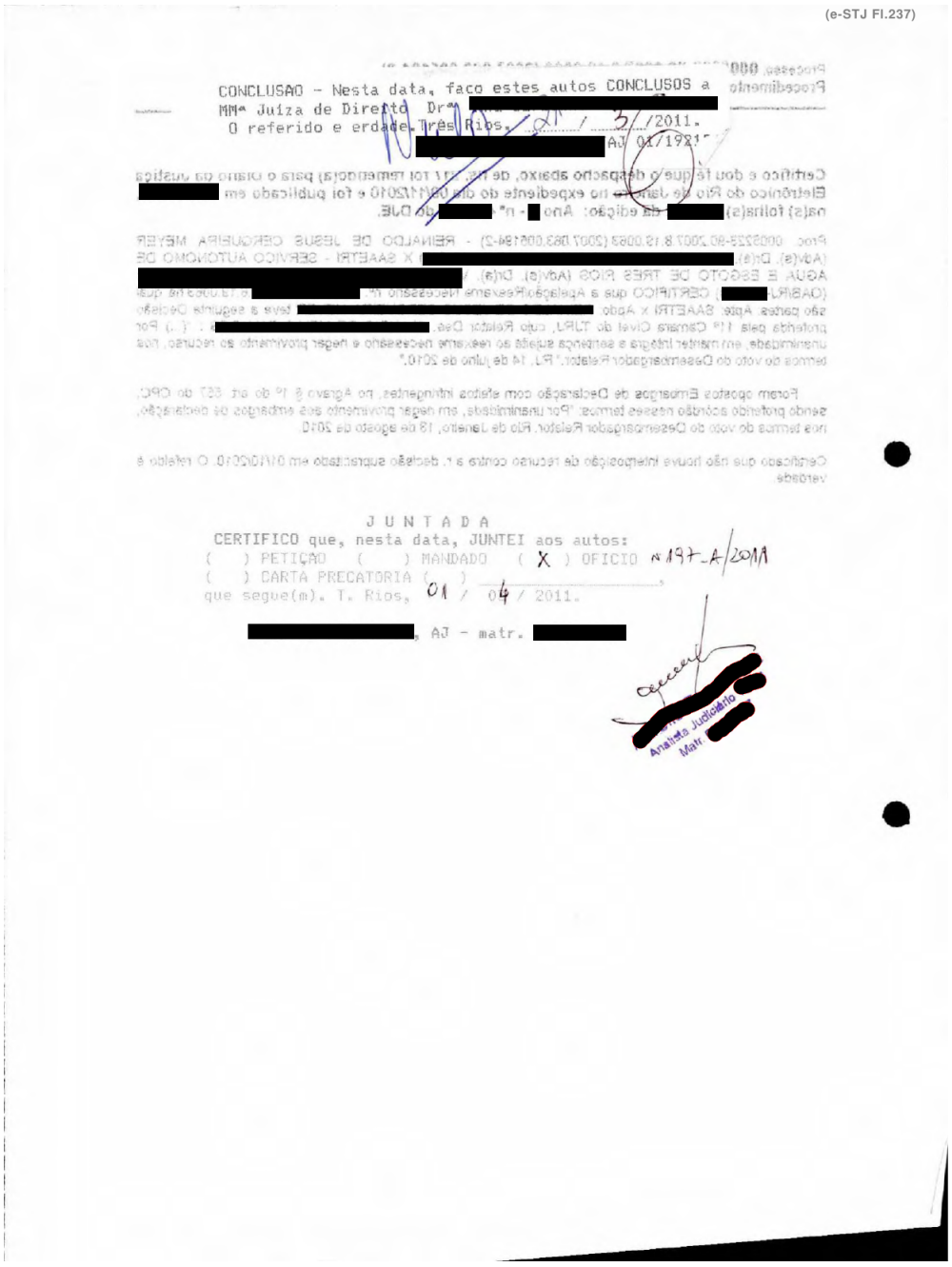

Figura 1 – Certidão de despacho digitalizada em volume

1.1 Problema e motivação

Como o STF é um órgão público central, lida com os casos de toda a Federação, advindos dos tribunais e juizados da segunda instância (BRASIL, 1988). Ele tem que lidar com as diferentes padronizações de marcadores nos volumes de processos e código das peças. Além disso, os servidores avaliam apenas as peças essenciais a sua atividade,

por conseguinte, precisam encontrar entre os diversos documentos do processo os que serão utilizados por eles.

Os próprios advogados podem fazer a classificação do tipo de peça ao submeter no sistema eletrônico do STF, desde que obedeçam o tamanho máximo de 10 MB por PDF[3] (BRASIL, 2010). Por conta disso, há documentos que estão com apenas uma categoria carregando o conteúdo de várias peças.

Desta forma, foram identificados os seguintes problemas enfrentados pelo STF ao lidar com as peças de processos:

- Processos advindos de diferentes fontes possuem códigos identificadores de peças e marcadores de volumes não padronizados, fato o qual dificulta os servidores de encontrar rapidamente as de seu interesse.
- Alguns marcadores nos volumes são errados ou incompletos, ou seja, não trazem informação suficiente para caracterizar uma peça específica.
- Peças duplicadas produzidas pela ressubmissão de um processo, por conta da adição do recurso de admissibilidade. Com esta duplicação, os servidores têm dificuldade de encontrar informações de interesse rapidamente.

Diante da problemática apresentada, definiu-se a pergunta de pesquisa: Como identificar automatizadamente o corpo de texto de diferentes tipos de peças processuais jurídicas, que sejam analisadas pelo STF para repercussão geral, em um único documento?

1.2 Hipótese

Foram desenvolvidas duas hipóteses (GIL, 2002) para este trabalho baseadas na especificação do contexto e problemática do STF.

- As peças jurídicas avaliadas pelo STF para classificar um processo numa repercussão geral são computacionalmente separáveis.
- Não existem classes de peças diferentes com o mesmo conteúdo de texto.

1.3 Objetivo

Nesta seção, serão apresentados o objetivo geral e os objetivos específicos para a delimitação deste trabalho de conclusão de curso.

[3] Formato de arquivo criado pela empresa Adobe Systems Incorporated para unificar o compartilhamento de conteúdo. Disponível em: <https://acrobat.adobe.com/br/pt/acrobat/about-adobe-pdf.html>. Acesso em: 17-06-2018

1.3.1 Objetivo geral

Classificar e indexar documentos jurídicos em categorias de peças processuais utilizando técnicas de ML e NLP.

1.3.2 Objetivos específicos

- Levantar o estado da arte de classificação de documentos;
- Realizar análise exploratória dos dados;
- Construir um dicionário de palavras para o corpo textual dos processos.
- Elaborar arquitetura de preprocessamento dos dados;
- Realizar a transformação dos textos para representação computacional;
- Avaliar modelos para classificação documentos;
- Avaliar técnicas para validação de modelos.

1.4 Organização do trabalho

Este trabalho estará organizado em cinco capítulos. O primeiro é a Introdução onde contextualiza-se a problemática e define-se os objetivos, o segundo capítulo é para descrição da metodologia científica adotada. O terceiro é a fundamentação teórica para o desenvolvimento. O quarto é onde elabora-se uma breve análise dos dados e define propostas de soluções para o problema. O quinto capítulo é onde discute-se os resultados da implementação da solução proposta e que será desenvolvido apenas na continuação desta pesquisa.

2 Metodologia

Este capítulo caracterizará o tipo de pesquisa e apresentará os aspectos metodológicos.

2.1 Caracterização da pesquisa

Este trabalho é uma pesquisa exploratória utilizando os métodos indutivo e experimental para o seu desenvolvimento. Uma pesquisa exploratória é utilizada quando deseja-se explorar as causas, os fatores influenciadores envolvidos e as fronteiras de um determinado problema. Usa-se da experimentação para controlar as variáveis envolvidas, tomar nota dos impactos delas para o problema (PRODANOV; FREITAS, 2013).

Métodos de pesquisa indutivos tem o objetivo de ampliar conhecimentos de uma determinada área, baseando-se na amostra de dados analisados. Diferentemente da dedução, em que chega-se a uma conclusão verdadeira. Este método tem o propósito gerar uma probabilidade de a conclusão ser verdadeira (PRODANOV; FREITAS, 2013).

Na indução parte-se de casos particulares para a generalização de outros semelhantes (PRODANOV; FREITAS, 2013), logo, busca-se através da experimentação e observação dos métodos de classificação de documentos, alcançar o rotulamento de documentos do STF com os métodos aqui explorados.

2.2 Processo de desenvolvimento

A seguir será apresentada o processo para o desenvolvimento. Considerando-se que é uma pesquisa e um projeto de ML, mesclou-se ambos os processos para alcançar um processo final.

2.2.1 Projeto Pesquisa

Um projeto de pesquisa tem as etapas de **planejamento** no qual formula-se os objetivos e o propósito para a pesquisa, **execução** que representa o desenvolvimento do trabalho caracterizado pela coleta de dados, **análise** e proposições das conclusões, **documentação** em que é formulado o texto resultante do trabalho e, por fim, a **publicação** da pesquisa (PRODANOV; FREITAS, 2013).

Destas 4 fases, têm-se as principais atividades definidas por PRODANOV; FREITAS (2013) que são:

- **Formular e planejar a pesquisa**: nesta atividade, faz-se a escolha pelo assunto e levantamento bibliográfico para investigação do problema. Determinação dos objetos de estudo, além de averiguar assuntos correlatos com o escolhido (PRODANOV; FREITAS, 2013).

- **Escolher assunto e delimitar tema**: realizar a delimitação do assunto proposto, especificando-o para um tema o qual facilite o desenvolvimento e aprofundamento da pesquisa (PRODANOV; FREITAS, 2013).

- **Revisar a literatura**: esta atividade tem a proposição de situar o trabalho com as pesquisas já desenvolvidas sobre o tema e identificar qual é o estado da arte. Ademais, auxilia os leitores a compreenderem sobre o assunto tratado, de forma que entendam o desenvolvimento da pesquisa (PRODANOV; FREITAS, 2013).

- **Justificar trabalho**: identificar motivos, causas, razões da relevância e contribuição do trabalho (PRODANOV; FREITAS, 2013).

- **Definir do problema de pesquisa**: trata-se da reflexão sobre um problema específico do tema. Após esta reflexão, exprime-se em uma única frase concisa a problemática do trabalho (PRODANOV; FREITAS, 2013). Define-se, também, o enfoque que o trabalho terá de forma bastante objetiva e específica (GIL, 2002).

- **Determinar os objetivos geral e específicos**: caracteriza-se por estabelecer os itens que serão estudados e executados para responder a pergunta do problema de pesquisa. É neste momento que se explicita o que será realizado no trabalho.

- **Coletar dados**: etapa realizada para obter os valores das variáveis do problema (PRODANOV; FREITAS, 2013). Na pesquisa experimental, faz-se isto manipulando certas condições para observar os efeitos no objeto de estudo. Para isto, determina-se uma amostragem diante de toda a população dos dados para a coleta (GIL, 2002).

- **Tabular e apresentar os dados**: organizar os dados, realizar cálculos, construir gráficos, elaborar tabelas e o uso de outras técnicas que facilitem o entendimento dos dados.

- **Analisar e interpretar os dados**: utiliza-se de métodos estatísticos, visualização gráfica e da literatura para entender o fenômeno em pesquisas experimentais (GIL, 2002). Esta atividade exige a capacidade analítica, descritiva e crítica do pesquisador (PRODANOV; FREITAS, 2013).

- **Concluir ou realizar considerações finais**: deixa-se claro a vinculação entre os dados e resultados obtidos, tal qual transparecer se os objetivos estabelecidos foram alcançados ou não (GIL, 2002).

- **Redigir e apresentar o trabalho**: concretizar toda a pesquisa em formato textual, de forma que fique claro todas as etapas e bibliografias consultadas. Atividade realizada em paralelo com todas as outras (PRODANOV; FREITAS, 2013).

2.2.2 Projeto de Aprendizagem de Máquina

A gerência de projetos de ML é semelhante a um projeto de Ciência de dados. Porém este é mais completo, pois envolve mais características como a de extração de dados (Mineração de dados), a implantação do sistema e a sua utilização para oferecer dados para negócios inteligentes (CHAPMAN et al., 2000).

As atividades de Ciência de dados apresentadas na gerência de projetos CRISP-DM são ilustradas na Figura 2.

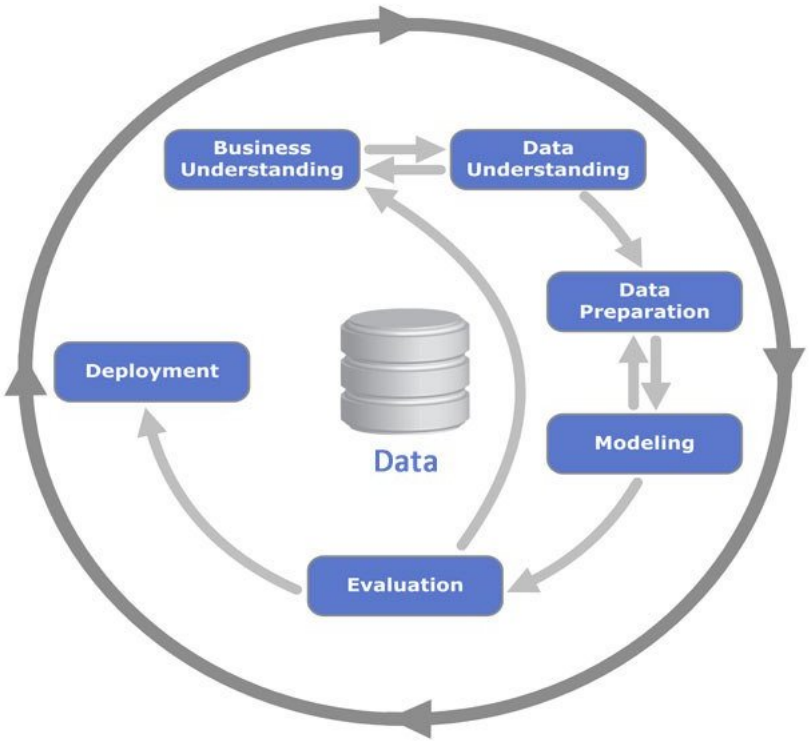

Figura 2 – Modelo de processo CRISP-DM. Fonte: (CHAPMAN et al., 2000, Página 10)

- **Entendimento do negócio**: atividade inicial que envolve o entendimento dos requisitos da organização, essa provê insumos para definir quais dados serão extraídos (CHAPMAN et al., 2000).

- **Entendimento dos dados**: explorar os dados almejando identificar suas características, problemas e possíveis informações ocultas (CHAPMAN et al., 2000).

- **Implantação**: a implantação é a ação de propor valor para a empresa ou para o consumidor através dos resultados obtidos do processo. Inclui, também, a entrega do modelo treinado (CHAPMAN et al., 2000).

A seguir será apresentada as atividades de um fluxo de trabalho para projetos de ML.

- **Obter dados históricos**: coletar dados históricos sobre o problema a ser resolvido, extrair as características dos dados, organizá-los estruturadamente para identificar dados faltantes e realizar um pre-processamento. Basicamente, trata-se de executar as atividades da engenharia de características, a qual é o conjunto de técnicas e ferramentas para transformação dos dados em um projeto de ML (BRINK; RICHARDS; FETHEROLF, 2015).
- **Construir um modelo**: construir ou utilizar um modelo que seja capaz de lidar bem com a natureza dos dados explorados na atividade de "Obter dados históricos". Este deve atender aos objetivos do projeto. Em problemas de classificação, o modelo deve classificar corretamente os rótulos de novas amostras (BRINK; RICHARDS; FETHEROLF, 2015).
- **Utilizar modelo**: com os dados resultantes da obtenção dos dados e da construção do modelo, faz-se uso do modelo para novos dados almejando a obtenção de respostas (BRINK; RICHARDS; FETHEROLF, 2015).
- **Validar modelo**: os modelos utilizados precisam passar pelo teste de lidar com novos dados. Através dos resultados obtidos destes testes, deve-se utilizar métricas adequadas a cada tipo de problema para identificar se o modelo teve bom resultado (BRINK; RICHARDS; FETHEROLF, 2015).
- **Otimizar o modelo**: após coletar o resultado das métricas, cabe ao desenvolvedor de um projeto de ML ter capacidade crítica, conhecimento do domínio e de técnicas para propor melhorias ao projeto. O foco nesta atividade é obter melhores resultados nas métricas. São três principais modos de melhorar: trocando os parâmetros do modelo, selecionando outro conjunto de características e/ou melhorando a engenharia de características (BRINK; RICHARDS; FETHEROLF, 2015).

2.2.3 Processo Final

Em projetos de Ciência de dados que incluem ML, a melhor metodologia para o desenvolvimento é o CRISP-DM (CROWSTON; SALTZ; SHAMSHURIN, 2017). Portanto, será utilizada uma junção das atividades de um projeto de pesquisa com as de projeto de ML.

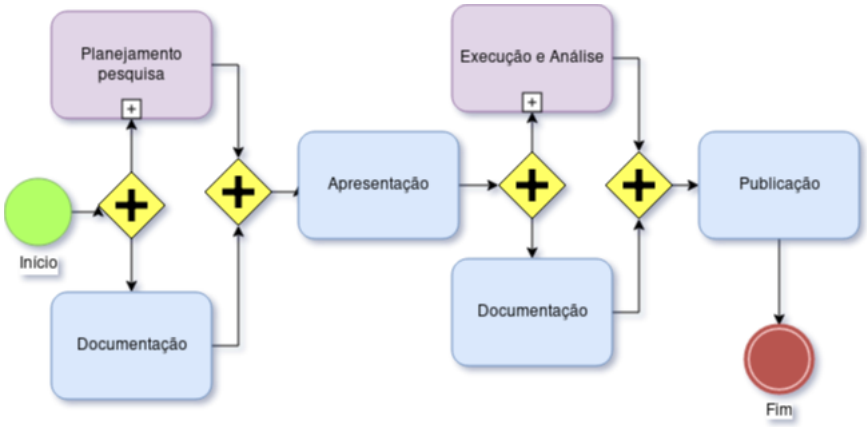

Figura 3 – Processo desenvolvimento da pesquisa. Fonte: elaboração própria

A Figura 3 trás a visão geral de como serão desenvolvidas as atividades do projeto. A tarefa de "Documentação" será executada em paralelo durante todo o projeto, exceto nas etapas de apresentação e publicação.

Das atividades da seção 2.2.1, temos que as de "Coletar dados", "Tabular e apresentar os dados", "Analisar e interpretar os dados" foram substituídas pelo fluxo de atividades de um projeto de ML. Esta substituição deve-se pela natureza do problema a ser resolvido, pois exige um fluxo diferenciado para análise dos resultados e obtenção dos dados. Essas atividades já foram definidas num projeto de ML descritas na seção 2.2.2.

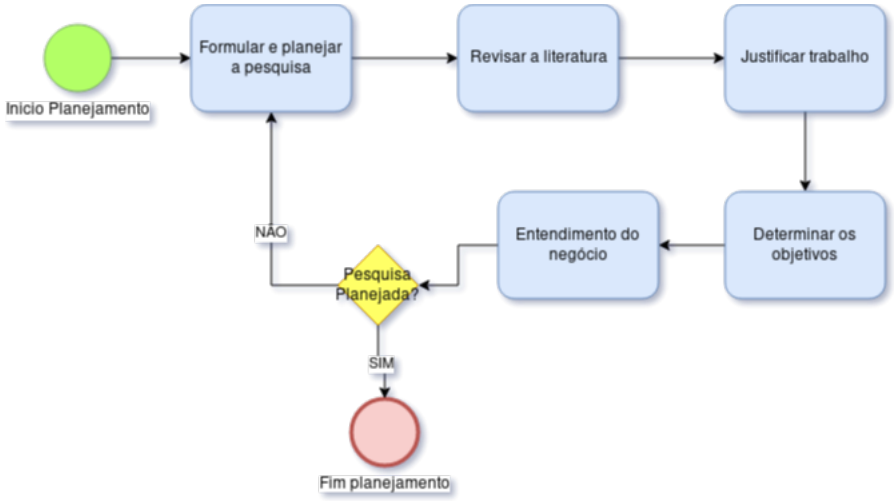

Figura 4 – Subprocesso planejamento da pesquisa. Fonte: elaboração própria

Além disso, na 4, foi adicionada a atividade "Entendimento do negócio" para melhorar a compreensão do problema do ponto de vista de um trabalho de ML, o que facilitará, inclusive, o desenvolvimento da pesquisa (CROWSTON; SALTZ; SHAMSHURIN, 2017).

Na Figura 5, foi definido o fluxo de ML semelhante ao proposto por BRINK; RICHARDS; FETHEROLF (2015). A única diferença entre eles são os propósitos de pesquisa, onde adicionou-se mais uma condição: "Os objetivos da pesquisa foram alcan-

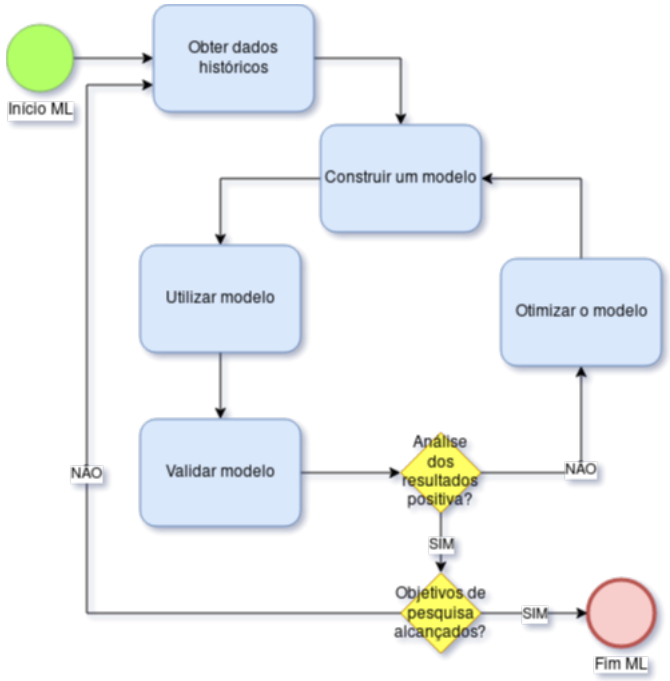

Figura 5 – Subprocesso execução e análise. Fonte: elaboração própria

çados?". Esta pergunta guiará o desenvolvimento do projeto, pois garante-se, mesmo com ótimos resultados no modelo, que a pesquisa tenha prosseguimento e seja possível alcançar os objetivos.

A atividade "Entendimento dos dados" foi removida do fluxo CRISP-DM, porque há uma outra atividade de "Obtenção dos dados" que possui equivalência. A outra de "Implantação" foi removida, pois ela é para projetos de Ciência de dados. A atividade equivalente a esta no trabalho científico é a de "Publicação dos resultados".

2.3 Cronograma

A seguir será apresentado uma visão macro do cronograma na Tabela 1.

Tabela 1 – Planejamento macro de atividades.

	Março		Abril				Maio					Junho			
Semanas Macro Tarefa	3	4	1	2	3	4	1	2	3	4	5	1	2	3	4
Formular pesquisa	X	X	X												
Revisar literatura				X	X	X									
Definir objetivos específicos							X								
Levantar necessidade STF								X							
Contextualizar NLP e ML								X							
Contextualizar ML								X	X						
Metodologia									X	X					
Referencial ML										X	X				
Referenciar entendimento do negócio										X	X				
Documentar obtenção de dados											X	X			
Análise exploratória dos dados												X	X		
Finalização do texto													X		
Defesa															X

Fonte: elaboração própria.

3 Referencial teórico

Este capítulo apresentará o referencial necessário para entendimento do conteúdo técnico da solução e contexto do problema.

3.1 Classificação de documentos

Nas áreas de NLP, existem grupos de técnicas para lidar com documentos de texto, assim como em ML. Destas, serão abordados apenas os conjuntos para a classificação de corpos de textos.

3.1.1 Técnicas de pré-processamento

As técnicas de pré-processamento utilizadas para a classificação de textos são as de transformação do texto em símbolos, remoção de palavras recorrentes, a radicalização, normalização e criação de regras específicas para cada contexto utilizando de expressões regulares (OLIVEIRA; FILHO, 2017).

3.1.1.1 Transformação em símbolos

A transformação em símbolos, trata-se de criar símbolos das palavras identificadas no texto (MANNING; RAGHAVAN; SCHÜTZE, 2008). A complexidade de como transformar a sentença numa lista de símbolos vai depender do contexto. Pode-se transformar endereços de rede, recursos de *web site* em representações únicas (MANNING; RAGHAVAN; SCHÜTZE, 2008). Na Tabela 2 é apresentado a separação da sentença em símbolos.

Tabela 2 – Transformação de palavras em símbolos.

Original	**Símbolos textuais**
juiz federal relator formar incisar iii **lei nº 11419** dezembro resolução trf região março conferência autenticidade documento disponível endereçar eletrônico **www.stf.org.br**	"juiz", "federal", "relator", "formar", "incisar", "iii", **"LEI_11419"**, "dezembro", "resolução", "trf", "região", "março", "conferência", "autenticidade", "documento", "disponível", "endereçar", "eletrônico", **"SITE"**

Fonte: elaboração própria.

3.1.1.2 Remoção de palavras recorrentes

Ocorre, em documentos de texto, a repetição de algumas palavras que não trazem valor na classificação dos documentos. Essas podem ser removidas do texto, com isto

diminui-se a quantidade de dados a serem processadas sem perder propriedades estatísticas e sintáticas do texto (MANNING; RAGHAVAN; SCHÜTZE, 2008).

Na Tabela 3 é apresentado a remoção das palavras: da, de, a, está.

Tabela 3 – Remoção de palavras recorrentes.

Original	**Palavras removidas**
juiz federal relator forma inciso iii **da** LEI_11419 **de de** dezembro resolução trf região março **a** conferência autenticidade documento **está** disponível endereço eletrônico SITE	juiz federal relator forma inciso iii LEI_11419 dezembro resolução trf região março conferência autenticidade documento disponível endereço eletrônico SITE

Fonte: elaboração própria.

3.1.1.3 Radicalização e normalização

Estas técnicas auxiliam na classificação dos textos, pois ressaltam propriedades estatísticas de uma palavra no texto. A tarefa de radicalização é remover sufixos e prefixos, verificar palavras compostas e substituí-las apenas pelo seu radical. Já a normalização, é transformar palavras que tenham diferentes formas com o mesmo significado para uma única representação (SINGH; GUPTA, 2016). A Tabela 4 apresenta um exemplo de aplicação destas técnicas.

Tabela 4 – Aplicação de radicalização e normalização.

Original	**Normalizado**	**Normalizado e Radicalizado**
juiz federal relator **forma inciso** iii da LEI_11419 de de dezembro resolução trf região março a conferência autenticidade documento **está** disponível endereço eletrônico SITE	juiz federal relator **formar incisar** iii da LEI_11419 de de **dezembro resolução** trf região março o **conferência** autenticidade documento estar disponível endereçar eletrônico SITE	juiz federal relator form incis iii da lei_11419 de de dezembr resolu trf regiã marc o conferent autent document estar dispon enderec eletrôn sit

Fonte: elaboração própria.

3.1.1.4 Expressões regulares

Quando deseja-se encontrar algum padrão específico em um texto, utiliza-se de regras de formação para capturar todos os grupos que se encaixaram. Esta técnica também auxilia a realizar substituições, remover trechos do texto. Ela diminui a complexidade de código para realizar manipulações em *strings* (GOYVAERTS; LEVITHAN, 2012). A

Tabela 5 mostra um exemplo da aplicação das expressões regulares presentes no Apêndice B.

Tabela 5 – Aplicação de expressões regulares.

Original	**Processado com Expressão Regular**
Juiz Federal Relator, na\nforma do artigo 1º , inciso III, da Lei 11.419, de 19 de dezembro de 2006 e Resolução TRF 4ª\nRegião nº 17, de 26 de março de 2010. A conferência da autenticidade do documento está\ndisponível no endereço eletrônico http://www.jfpr.jus.br/ gedpro/verifica/verifica.php,	juiz federal relator forma inciso iii da LEI_11419 de de dezembro resolução trf região março a conferência autenticidade document está disponível endereço eletrônico SITE

Fonte: elaboração própria.

3.1.2 Representações de textos

Quando faz-se o processamento de linguagem natural, utiliza-se diferentes formas de representação para o texto ao invés de *string*. As técnicas utilizadas são as de *one-hot-encoder*, *bag of words* (BoW) e *word embedding*.

3.1.2.1 One-hot-encoder

Esta técnica consiste em transformar cada palavra num único vetor, no qual a posição da palavra no dicionário de dados receberá o valor 1. Desta forma, uma sentença se tornará uma matriz, como no exemplo abaixo (BRINK; RICHARDS; FETHEROLF, 2015).

$$\begin{aligned} Frase = & \quad [0 \quad 1 \quad 0 \quad 0] \\ de = & \quad [1 \quad 0 \quad 0 \quad 0] \\ exemplo = & \quad [0 \quad 0 \quad 1 \quad 0] \\ simples = & \quad [0 \quad 0 \quad 0 \quad 1] \end{aligned}$$

3.1.2.2 Bag of words

Diferentemente do *One-hot-encoder*, o BoW transforma toda a sentença em apenas um vetor. Cada posição deste, possuirá a quantidade de vezes que um símbolo apareceu. Ela permite que possam ser removidas palavras com alta ou baixa incidência (BRINK; RICHARDS; FETHEROLF, 2015).

$$\begin{aligned} Frase\ de\ exemplo\ simples = & \quad [1 \quad 1 \quad 1 \quad 1 \quad 0] \\ Frase\ de\ frase\ repetida = & \quad [2 \quad 1 \quad 1 \quad 0 \quad 1] \end{aligned}$$

3.1.2.3 Word Embedding

O *word embedding*, diferentemente dos métodos anteriores que são representados como símbolos únicos, lida com uma representação distribuída (GOLDBERG, 2017).

Neste método, o significado de uma palavra, que foi capturada numa janela de contexto, é representado na forma de um vetor. Cada dimensão deste vetor não representa necessariamente um aspecto do mundo real, mas o conjunto delas que trás alguma significância (GOLDBERG, 2017). Entende-se as palavras como vetores que podem possuir relações em seu espaço de representação, como na Figura 6.

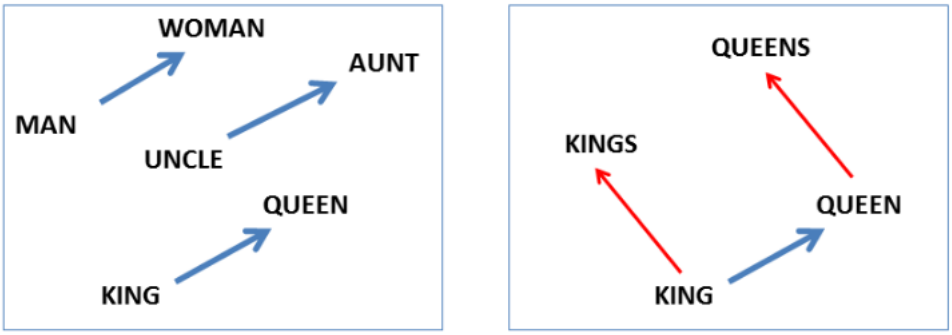

Figura 6 – Representação discreta de palavras. Fonte: (MIKOLOV; YIH; ZWEIG, 2013, Página 479).

Com os vetores das palavras Rei, Rainha, Homem e Mulher torna-se possível realizar consultas às palavras da forma: $vect('Rei') + vect('Mulher') - vect('Homem')$, no qual a resposta é Rainha (MIKOLOV; YIH; ZWEIG, 2013). Esta proposição também se confirma para outras relações como País-Capital e Adjetivos Aumentativo-Diminutivo (MIKOLOV et al., 2013).

$Palavra =$	$[$	$Sexo$	$Realeza$	$Pluralidade$	$Parentesco$	$]$
$Rei =$	$[$	1	1	-1	$0,5$	$]$
$Reis =$	$[$	1	1	1	$0,5$	$]$
$Rainha =$	$[$	-1	1	-1	$0,5$	$]$
$Rainhas =$	$[$	-1	1	1	$0,5$	$]$
$Homem =$	$[$	1	0	-1	0	$]$
$Mulher =$	$[$	-1	0	-1	0	$]$
$Tio =$	$[$	-1	0	-1	1	$]$
$Tia =$	$[$	1	0	-1	1	$]$

No exemplo acima, cada palavra é representada por um vetor, no qual, foi definido apenas quatro dimensões de exemplos. As duas principais formas de chegar a este resultado é pelo *Word2Vect* elaborado por MIKOLOV et al. (2013) e o *GloVe* definido por PENNINGTON; SOCHER; MANNING (2014).

O uso de *word embedding* para modelos de classificação, são mais estáveis (ENRÍQUEZ; TROYANO; LÓPEZ-SOLAZ, 2016), representam ganhos de acurácia para classificação de textos curtos (BUTNARU; IONESCU, 2017) (GE; MOH, 2017).

3.1.3 Técnicas de ML

As principais técnicas para aprendizado de máquina para classificação de texto são Máquinas de Vetores de Suporte (do inglês *Suporte Vector Machine* SVM), K Vizinhos Próximos, Multinomial Naive Bayes.

Para este trabalho, utilizar-se-a apenas o modelo SVM. Isto porque serve para lidar com vetores que possuem muitas características. Além disso, seu algoritmo possibilita extrair dos vetores de suporte, os quais são as dimensões mais importantes para realizar a separabilidade dos dados (HEARST et al., 1998).

3.1.3.1 Máquinas de Vetores de Suporte

Este algoritmo recebe os vetores de entrada, com uma alta dimensionalidade, e faz transformações neles utilizando seus núcleos para que seja possível aplicar métodos lineares, mesmo em problemas não-lineares (HEARST et al., 1998).

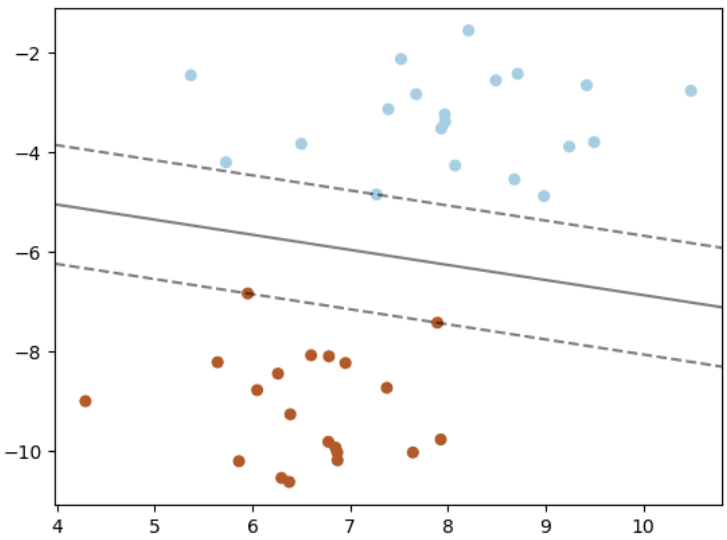

Figura 7 – Funcionamento do algoritmo SVM. Fonte: (PEDREGOSA et al., 2011, Acesso em 10 de Junho de 2018.)

A função de classificação do SVM é baseada em hiperplanos. O algoritmo busca otimiza-los de forma que ele fique ortogonal às linhas que separam as duas classes, como

é possível ver na Figura 7. Chama-se estes vetores de separação de suporte (SMOLA; SCHöLKOPF, 2004).

$$f(x) = sign(\sum_{i=1}^{t} v_i \cdot k(x, x_i) + b) \tag{3.1}$$

A Equação 3.1 é a de classificação utilizada neste algoritmo. A função k é a que representa o núcleo, na implementação de PEDREGOSA et al. (2011) existe disponível o: linear, polinomial, função de base radial e o de sigmoide.

O valor de x em 3.1 representa o vetor de entrada a ser classificado. Já o x_i representa os vetores de suporte. Os pesos v_i, são calculados através de uma solução quadrática, no qual são a combinação linear dos padrões de entrada (SMOLA; SCHöLKOPF, 2004). Os valores de b, são limiares para maximização das margens entre duas classes distintas (HEARST et al., 1998).

3.1.3.2 Redes Neurais

Um dos algoritmos desenvolvidos para realizar o aprendizado de máquina, foi baseado em um neurônio humano (GOLDBERG, 2017). O neurônio, chamado de perceptron possui sensores de retina (entrada), os sinais recebidos se propagam para uma área de associação, onde são combinados. O sinal propagado para as ligações a frente, se e apenas se, o valor dele for maior que um limitar θ (ROSENBLATT, 1958).

Como na figura 8, há pesos para os vetores de entrada, que são todos somados e, em seguida, passam pela função de ativação para gerar o resultado do perceptron (RASCHKA, 2015).

Agrupou-se os neurônios para criar uma rede, na qual ela possui diferentes camadas.

Entrada: esta camada não possui funções de ativação, sua função é propagar os dados de entrada para a camada seguinte.

Camadas ocultas: são as camadas que possuem funções de ativações. Uma camada oculta pode possuir vários neurônios.

Saída: esta é a ultima camada de uma rede neural, ela que irá retornar o resultado do processamento realizado pelas camadas ocultas.

Os neurônios com a coloração verde na figura 9, indicam que apenas estes foram ativados durante o processamento.

O fluxo comum da execução de uma rede neural é o recebimento dos dados de entrada na primeira camada, em seguida há a propagação dos valores para uma camada

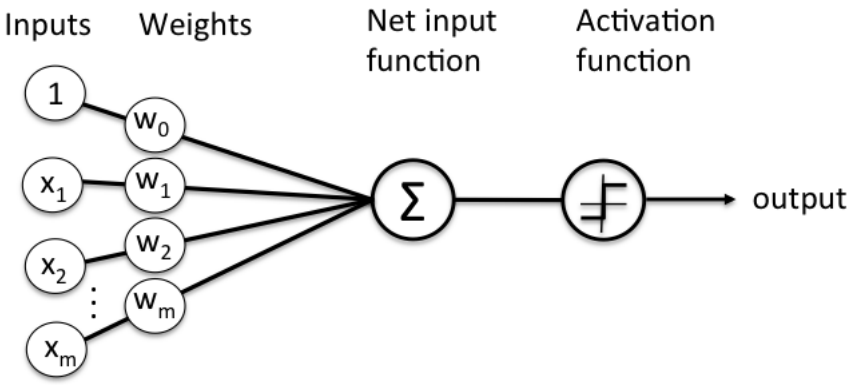

Figura 8 – Ilustração esquemática de um perceptron de Rosenblatt. Fonte: (RASCHKA, 2015).

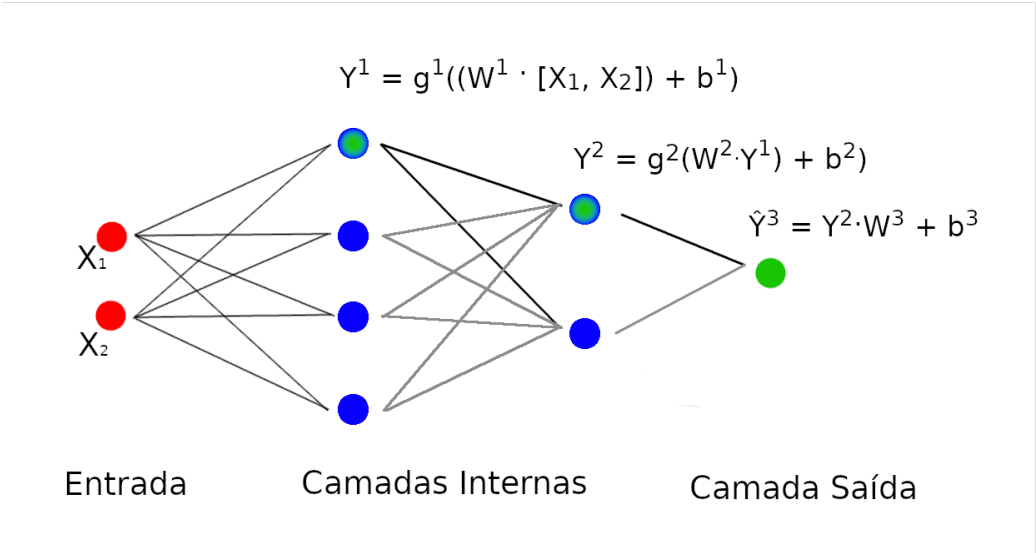

Figura 9 – Rede neural com duas camadas internas. Fonte: elaboração própria

oculta, após o cálculo de ativação, os dados são propagados novamente para a próxima camada, até que ao final, chega-se na camada de saída (GOLDBERG, 2017).

O processo de propagação (*feed-forward propagation*) é representado na Figura 9, no qual há propagação dos valores com o produto entre o vetor x e W (GOLDBERG, 2017). A representação de vetores da computação para os pesos são feitas de forma que o neurônio da camada seguinte i tem o peso associado a posição j ($W[i][j]$) (NIELSEN, 2015).

Para modelos com mais camadas do que foi ilustrado na Figura 9, pode-se generalizar a equação lá apresentada. Em cada camada i, o vetor W^i possui dimensão $d^i_{in} \times d^i_{out}$ dimensões. Os vetores de viés b^i terão $1 \times d^i_{out}$. O valor de N é igual ao número de camadas na rede. Na Equação 3.2, cada camada da rede neural poderá ter sua própria função de ativação (GOLDBERG, 2017).

$$x^i = \begin{cases} x & \textit{se for camada de entrada} \\ \hat{y}^{i-1} & \textit{caso contrário} \end{cases}$$

$$y^i = g^i(x^i W^i + b^i) \tag{3.2}$$

$$\hat{y} = y^{N-1} W^N$$

Para ajustar os valores dos pesos e vetor de viés W^i e b^i, calcula-se o quanto os neurônios estão corretos em relação ao distanciamento entre o $\hat{y}$ e y. Utiliza-se da função de erro $L(\hat{y}, y)$ (GOLDBERG, 2017), para calcular o erro na camada de saída. Este deve ser propagado para a camada anterior, com isto, todas as camadas vão ajustando recursivamente seus pesos e vetores de viés (NIELSEN, 2015). Este processo chama-se de retro-propagação (*back-propagation*) apresentado na Figura 10.

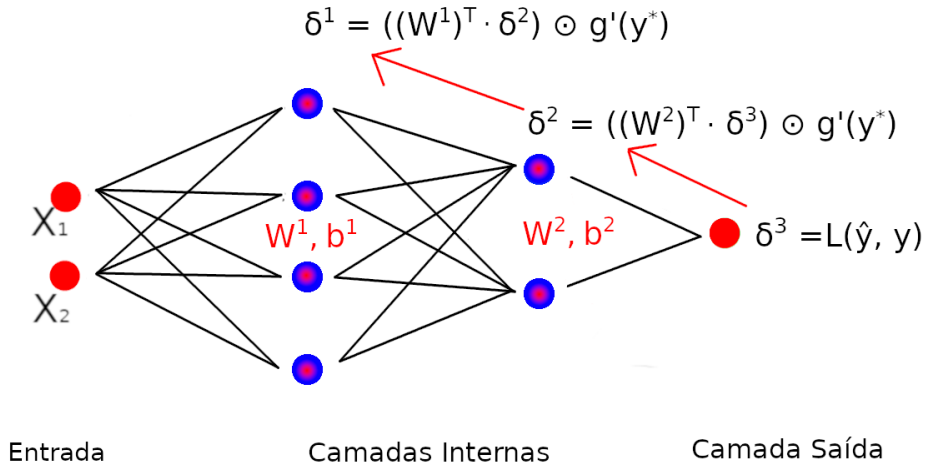

Figura 10 – Simplificação da retro-propagação em Rede Neural. Fonte: elaboração própria

Na Figura 10, o valor de g' é a derivada para a função de ativação, executada no valor de y^*, o qual é o resultado da operação de $x^i W^i + b^i$ sem aplicar a função de ativação g (NIELSEN, 2015).

A operação de $\odot$ é o produto de Hadamard, ela representa multiplicar os elementos de cada posição de um vetor a_n pelo do valor de b_n resultando em um outro vetor c_n

(NIELSEN, 2015). A Equação 3.3 ilustra essa operação entre os vetores.

$$\begin{bmatrix} 3 \\ 7 \end{bmatrix} \odot \begin{bmatrix} 4 \\ 9 \end{bmatrix} = \begin{bmatrix} 3 & * & 4 \\ 7 & * & 9 \end{bmatrix} = \begin{bmatrix} 12 \\ 63 \end{bmatrix} \tag{3.3}$$

Com a função de δ^i, faz-se a atualização dos pesos e vetores de viés. A estrutura é recursiva, de forma que, para calcular o δ^i, é necessário ter calculado o valor de δ^{i+1}. A prova matemática para a retro-propagação e a definição da função $L(\hat{y}, y)$ encontra-se em NIELSEN (2015).

A retro-propagação é utilizada juntamente com uma função de gradiente. Com esta combinação, faz-se os ajustes nos pesos das camadas, de forma que, a cada nova predição de $\hat{y}$, é feita uma retro-propagação calculando os valores mínimos para a função $L(y, \hat{y})$ (NIELSEN, 2015).

Com as técnicas apresentadas, é possível implementar uma rede neral Multicamadas de Perceptron (do inglês *Multilayer Perceptron* - MLP). Existem outros tipos de arquiteturas, como as Redes Neurais Recorrentes (RNN), as Redes Neurais Convolucionais (CNN). Elas podem ser utilizadas como camadas internas de uma MLP, bem como podem existir camadas de incorporação e de conjugação (GOLDBERG, 2017).

3.1.4 Aprendizado Profundo

O aprendizado de máquina profundo é quando utiliza-se várias camadas ocultas em uma rede neural para realizar o aprendizado. Não é restrito o uso de apenas de um tipo de camada ou função de ativação para a construção de uma rede profunda.

3.1.4.1 Redes Convolucionais

Este tipo de arquitetura, baseia-se na aplicação da operação de convolução nos dados de entrada e, em seguida, realizar junções. Uma CNN é definida como extratora de características de dados textuais por realizar as etapas de convolução e junção repetidas vezes (GOLDBERG, 2017).

O uso dessa rede não se limita apenas a extração de características, pode ser utilizada também, sozinha, para classificar textos utilizando abordagem de rede neural muito profunda com uma última camada de classificação linear (CONNEAU et al., 2017).

Considera-se uma sentença $x_{1:n} = x_1, x_2, ..., x_n$ e um filtro w que pode ser aplicado a intervalo de h palavras. Através da operação de convolução é possível gerar uma nova característica, dada pela aplicação de uma função de ativação g (KIM, 2014):

$$c_i = g(w \cdot x_{i:i+h-1} + b) \tag{3.4}$$

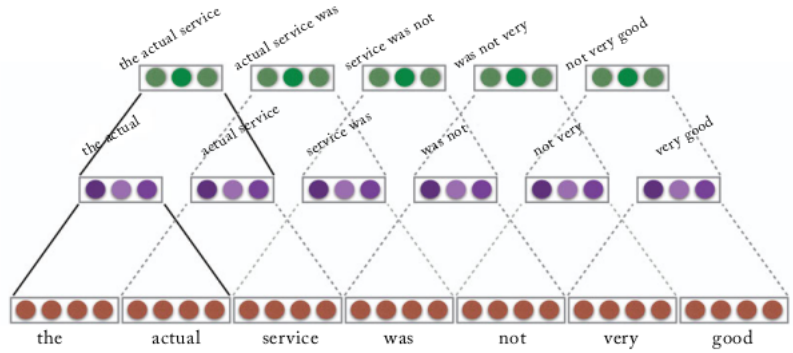

Figura 11 – Aplicação de CNN em texto. Fonte: (GOLDBERG, 2017, Página 160)

Na Figura 11, é apresentado graficamente o processo da aplicação de filtros de convolução, seguidos da operação de junção em 2 camadas. A partir da Equação 3.4, gera-se um novo conjunto de características extraídas do texto. Após isso, utiliza-se de uma camada de junção para obter o maior valor daquele novo sub conjunto (KIM, 2014).

$$c = [c_1, c_2, ..., c_{n-h+1}]$$

$$\hat{c} = \begin{cases} maior(c) \\ média(c) \\ k - maiores(c) \end{cases} \tag{3.5}$$

O resultado de $\hat{c}$, é o maior valor significativo para o filtro w aplicado sobre uma sentença entrada x. Em uma rede convolucional, tem-se vários filtros que possibilitam processar (KIM, 2014) e identificar os n-gramas mais importantes (GOLDBERG, 2017).

3.1.4.2 Redes Recorrentes

As redes neurais recorrentes possibilitam capturar características sequenciais dos dados. Sendo capaz de receber um vetor de entrada com tamanho variado e entregar um tamanho fixo na camada de saída (GOLDBERG, 2017).

Dado um vetor $x_{1:n} = x_1, x_2, ..., x_n$ de entrada na $f(x_{i:n})$, será retornado um vetor $\hat{y}_{1:m} = \hat{y}_1, \hat{y}_2, ..., \hat{y}_m$, com o valor de m fixado. Há um vetor de estados s que carrega a informação do processamento dos neurônios anteriores. Uma rede neural recorrente é recursiva, pois o neurônio i, realizará o seguinte processamento $s_i = RNN_i(x_i, s_{i-1})$ (MIKOLOV et al., 2010).

Na figura 12, as funções de R, O em cada perceptron são, respectivamente, as funções f e g da equação 3.6. O θ, são os valores de hiperparametrização da rede.

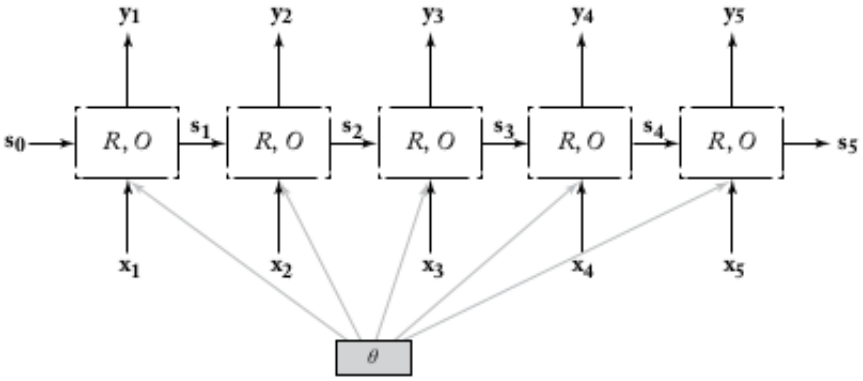

Figura 12 – Arquitetura recursiva da RNN. Fonte: (GOLDBERG, 2017, Página 166)

Para utilizar a RNN como método de classificação, deve-se aplicar uma função de ativação g. Na equação 3.6, tem-se também a função f, que é uma função de ativação responsável por juntar o vetor de estado s_i com o vetor de entrada x_i (MIKOLOV et al., 2010).

$$s_i = \begin{cases} i = 0 & f(S, x_i) \\ i \neq 0 & f(s_{i-1}, x_i) \end{cases} \qquad \hat{y}_i = g(s_i) \tag{3.6}$$

Na Equação 3.6, o valor de S representa um hiper-parâmetro para a rede neural. Este é iniciado com um vetor de zeros $[0, 0, ..., 0]$.

3.1.5 Métodos de avaliação

A seguir serão apresentados o método de validação cruzada para modelos de classificação e as métricas para determinar o desempenho.

3.1.5.1 Validação Cruzada

Para realizar a validação do modelo de classificação, há o modelo de validação cruzada. Este consiste em dividir o conjunto de dados em validação (30%) e treino (70%). A Figura 13 apresenta o processo para realizar a validação de um modelo classificador (BRINK; RICHARDS; FETHEROLF, 2015).

De acordo com BRINK; RICHARDS; FETHEROLF (2015), os passos para a validação são:

1. Realizar a divisão dos dados em conjunto de treino e teste.

2. Utilizar o conjunto de treino para a construção do modelo.

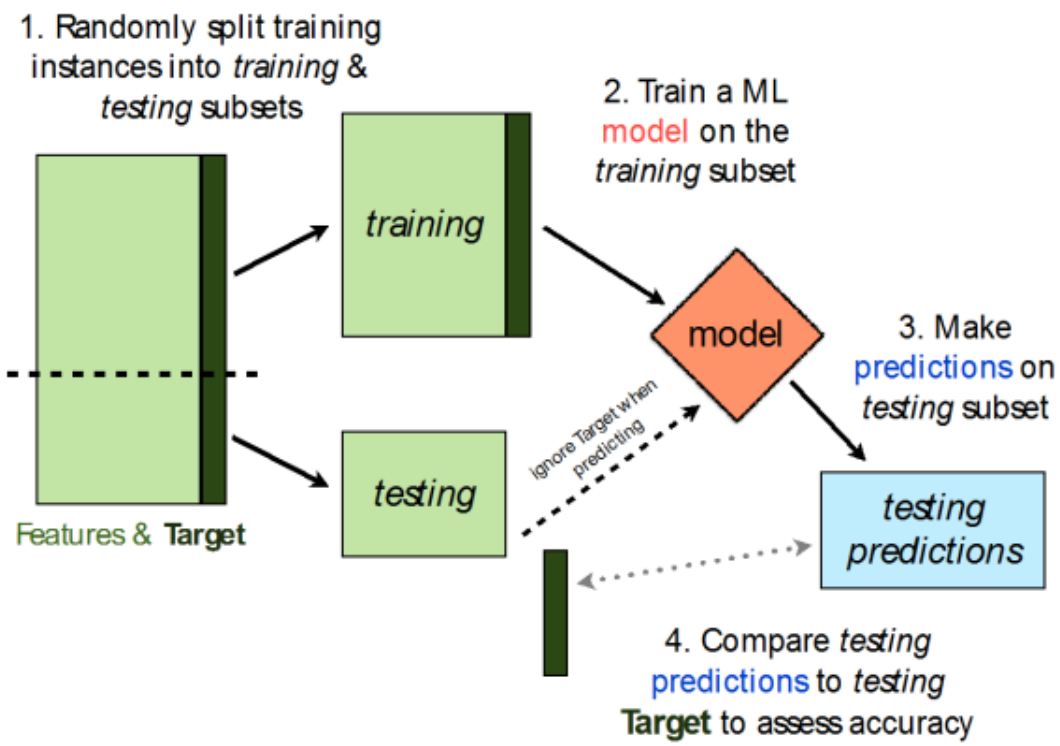

Figura 13 – Validação cruzada de modelos. Fonte: (BRINK; RICHARDS; FETHEROLF, 2015, Página 82)

3. Utilizar o modelo para predizer o conjunto de dados de teste.

4. Comparar os resultados da predição com os rótulos da base de teste e gerar as métricas.

3.1.5.2 Métricas

Em modelos de classificação, é possível determinar com exatidão o resultado esperado em categorias. Com isso, é possível determinar os Verdadeiros Positivos, Verdadeiro Negativos que são quando o modelo acerta a predição. Para quando ele erra, há o Falso Positivo e Falso Negativo (BRINK; RICHARDS; FETHEROLF, 2015).

A Tabela 6 mostra a relação de Verdadeiros/Falsos Positivos/Negativos quando há mais de uma classe.

Tabela 6 – Matriz de confusão.

Classe (N)	A	B	C	
A	VP_A	eB	eC	FP_A
B	eA	VP_B	eC	FP_B
C	eA	eB	VP_C	FP_C
	FN_A	FN_B	FN_C	$T = FP_A + FP_B + FP_C = FN_A + FN_B + FN_C$

Fonte: elaboração própria.

Os valores de eA, eB e eC representam erros na predição, enquanto que os de VP_A, VP_B e VP_C, os acertos. Ao somar-se os valores da coluna A na Tabela 6, tem-se o resultado do $VP_A + FN_A$. Isso indica o quanto o modelo está predizendo a classe A. Ao mesmo tempo que somando-se a linha A, tem-se $VP_A + FP_A$, a qual representa o total de valores de A.

Com esta tabela, é possível obter valores numéricos para as métricas de acurácia (ou eficiência), revocação (ou sensibilidade), taxa de verdadeiros negativos, taxa de falsos positivos e negativos, precisão. Além de outras métricas que utilizam estas para seus cálculos (RODRÍGUEZ; CASTAÑO; SAMBLÁS, 2016).

As métricas a seguir serão utilizadas a fim de validar a qualidade e evolução de performance dos modelos. Todas as métricas a seguir são descritas por RODRÍGUEZ; CASTAÑO; SAMBLÁS (2016):

Revocação: trata-se da taxa de acertos para a classe N. A fórmula é: $R = \frac{VP_N}{FP_N}$.

Precisão: também conhecido como 'acurácia da classe N', representa a proporção de VP para os valores preditos em N. $P = \frac{VP_N}{FN_N}$

Taxa geral de acerto: também conhecido como "acurácia do modelo", representa o quanto ele acerta de todas as classes. $G = \frac{\sum VP_N}{T}$

Chance geral de acerto: representa quais as chances do modelo predizer corretamente um novo dado. $CG = \frac{\sum (FN_N \times FP_N)}{T^2}$

A revocação e precisão são para analisar pontualmente o comportamento do modelo com cada classe. Enquanto que a taxa geral de acerto e a chance geral de acerto irão proporcionar a informação da capacidade do modelo em aprender como separar as peças. Além disso, a matriz de confusão (Tabela 6) será utilizada para verificar quais classes são mais confundidas pelo modelo.

3.2 Sistema jurídico brasileiro

A estrutura do sistema jurídico brasileiro é estabelecido pela Constituição Federal de 1988. Ela prevê quais são os órgãos que compõe o Poder Judiciário, as competências que eles possuem, onde estão localizados, o tamanho dos representantes das instâncias superiores e como se faz a investidura aos cargos (BRASIL, 1988). Além da Constituição, para um estudo mais completo, são necessárias as Lei n.º 8.457/1992, que estabelece a primeira instância da Justiça Militar (BRASIL, 1992) e a n.º 12.665/2012, a qual define as Turmas Recursais para segunda instância de Juizados Especiais (BRASIL, 2012).

Para ter um entendimento completo sobre as peças jurídicas e suas origens, é preciso ter conhecimento sobre como estão organizados os órgãos do Judiciário, pois são através deles que a população vivencia um processo (AMENDOEIRA JR, 2012).

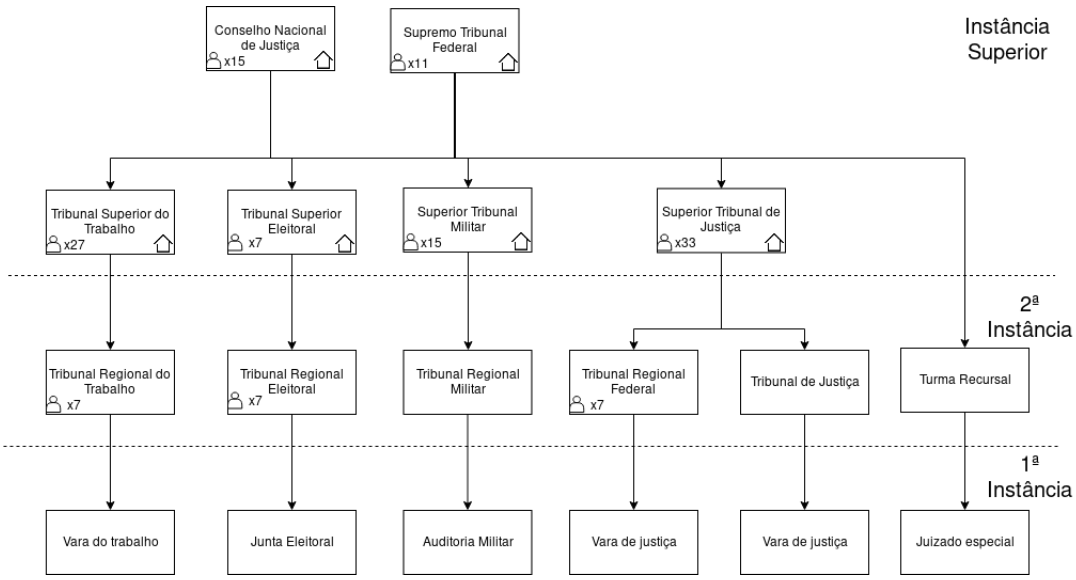

Figura 14 – Organização do sistema jurídico brasileiro. Fonte: elaboração própria

Para organizar melhor o trabalho do sistema judiciário, separou-se a justiça comum em algumas áreas para lidar com temas específicos: os assuntos trabalhistas, os eleitorais e uma especialização para lidar com as leis militares (AMENDOEIRA JR, 2012).

3.2.1 Instância superior - Tribunais Superiores

Como apresentado na Figura 14, são tribunais superiores Supremo Tribunal Federal, Supremo Tribunal de Justiça (STJ), Tribunal Superior Eleitoral (TSE), Tribunal Superior do Trabalho (TST) e Superior Tribunal Militar (STM) (BRASIL, 1988).

O STF e o STJ não se encaixam em nenhuma das justiças comum ou especializadas. Cabe ao STF a responsabilidade de julgar, em Recurso Extraordinário (RE), processos que tratam da infração a Constituição e ao STJ julgar, em Recurso Especial, a não obediência ou ilegitimidade das Leis Federais (BRASIL, 1988). Eles não são chamados de terceira instância devido ao princípio da dupla jurisdição. A correta denominação é instância superior, pois em Recurso Extraordinário, julgam as teses jurídicas e não os fatos do processo (AMENDOEIRA JR, 2012).

Os tribunais TST, TSE e STM compõe também a instância superior, mas estes julgam apenas os recursos de sua área de especialização na justiça (BRASIL, 1988).

Todos possuem sua sede na capital do Brasil e o número mínimo de representantes está estabelecido diretamente na Constituição (1988). Na Figura 14, são representados

estes valores para cada um deles. Estes Tribunais também possuem a competência para estabelecer o número de servidores nas instâncias inferiores (BRASIL, 1988).

3.2.2 2ª instância - Tribunais

Além dos Tribunais de segunda instância da justiça especial: o Tribunal Regional Eleitoral, o Tribunal Regional do Trabalho e o Tribunal Regional Militar, tem-se os Tribunais que compõe a justiça comum o Tribunal Regional Federal e os Estaduais Tribunal de Justiça (BRASIL, 1988). Há também as turmas recursais que representam a segunda instância para os juizados especiais (BRASIL, 2012).

Nestes Tribunais, julga-se os Recursos Ordinários, as provas, as evidências, diferentemente dos Tribunais de instância superior (AMENDOEIRA JR, 2012).

3.2.3 1ª instância

Para se entender a organização da justiça de primeira instância, é preciso definir como estão estruturadas a separação da jurisdição dos Tribunais.

Cada Estado e o Distrito Federal, dividem-se em comarcas (ou foros), no qual cada um destes possue uma ou mais varas especializadas para as justiças comum e do trabalho (AMENDOEIRA JR, 2012). Para a Justiça Militar, o território nacional está separado em circunscrições, onde cada uma possue sua Auditoria Militar (BRASIL, 1992). Já para a Justiça Eleitoral, existem juntas eleitorais que subdividem os Estados e o Distrito Federal em zonas distintas (BRASIL, 1988).

A seguir são apresentadas as definições:

Circunscrição é uma divisão geográfica administrativa para restringir a atuação de um tribunal (GUIMARÃES, 2012, p. 71).

Comarca é uma circunscrição sob jurisdição de juízes, na qual o território esta subdividido (GUIMARÃES, 2012, p. 75).

Vara é uma repartição judiciária com seu domínio a cargo de um juiz (GUIMARÃES, 2012, p. 259).

3.3 Código Processual Civil (CPC)

O CPC (Lei n.º 13.105/2015) é a lei que define como devem ser estruturados os processos civis. Ele é o meio pelo qual o juiz concretiza as leis, considerando os interesses entre as partes envolvidas.

O processo civil possui duas categorias de procedimentos adotados: o comum e especial. Neste trabalho tratar-se-á apenas do comum. A seguir, será apresentado para o entendimento das peças jurídicas avaliadas pelo STF, suas características em meio ao processo civil.

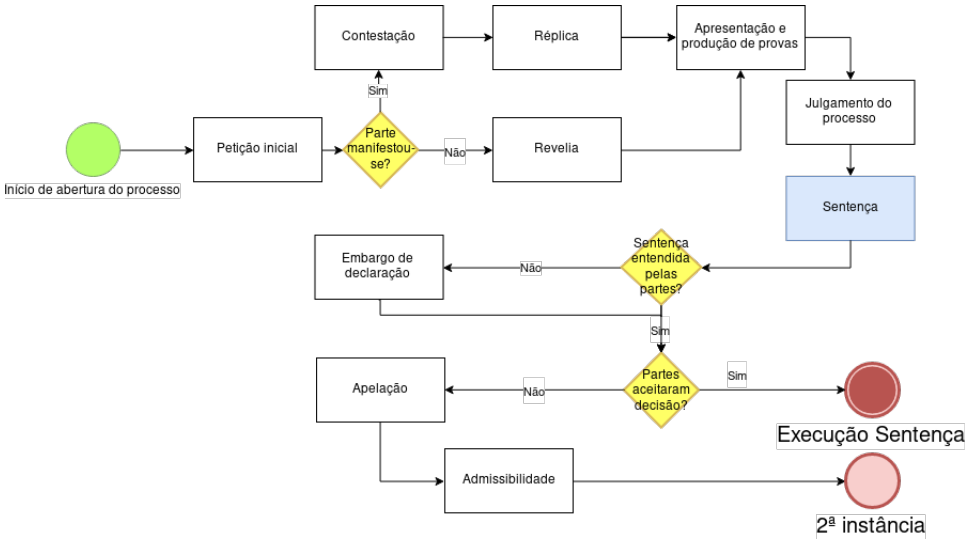

Figura 15 – Processo judiciário - 1ª Instância. Fonte: elaboração própria

3.3.1 1ª instância

O processo civil dá-se início por meio de uma peça chama de Petição Inicial. Através desta, a parte faz seus pedidos ao juiz e identifica quem é a outra parte, que por sua vez, possui o direito de se manifestar ou não (BRASIL, 2015). Chama-se esta fase de postulatória (GONÇALVES, 2016).

A etapa seguinte é a ordinatória, caracterizada pelo direito de réplica. Depois disso, inicia-se a fase instrutória caracterizada pela produção de provas (GONÇALVES, 2016).

A etapa decisória tem o ato da sentença (GONÇALVES, 2016). Deste, é gerada a peça que descreve a decisão tomada pelo juiz mediante os fatos apresentados pelas partes. Uma Sentença é inalterável pelo próprio juiz que a proferiu, poderá apenas ser esclarecida pelo recurso de Embargo de Declaração (BRASIL, 2015).

Após a sentença, as partes têm o direito garantido pelo duplo grau de jurisdição de realizar a apelação. O processo será enviado a um órgão segunda instância caso esteja em conformidade com a lei (GONÇALVES, 2016).

O Artigo n.º 489 do CPC (Lei n.º 13.105/2015) define que existe em todas as sentenças três elementos fundamentais:

> I - o relatório, que conterá os nomes das partes, a identificação do caso, com a suma do pedido e da contestação, e o registro das principais ocorrências havidas no andamento do processo;

> II - os fundamentos, em que o juiz analisará as questões de fato e de direito;
>
> III - o dispositivo, em que o juiz resolverá as questões principais que as partes lhe submeterem. (BRASIL, 2015)

3.3.2 2ª Instância

Garantido pelo princípio da dupla jurisdição, os processos que cumprem com os requisitos formais de uma apelação são direcionados ao tribunal de segunda instância competente para sua análise. Um Acórdão é o resultado do voto de três ou mais desembargadores, a decisão que mais receber votos é a final, portanto deve ser ímpar o número de juízes na sessão.

Após essa fase decisória, há o momento para as partes ou o Ministério Público elaborarem os recursos. Na Figura 16, são as atividades do fluxo, após o procedimento de primeira instância.

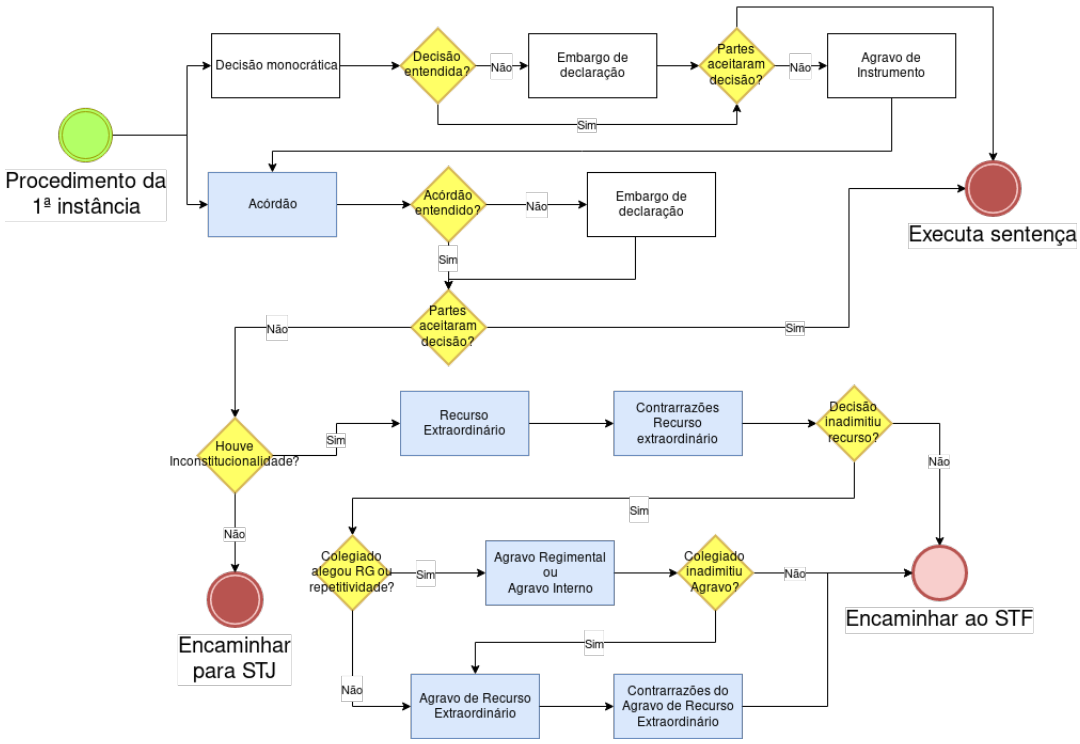

Figura 16 – Processo judiciário - 2ª Instância. Fonte: elaboração própria

O Agravo Interno ou Regimental, deve ter em seu conteúdo apenas a informação sobre a questão agravada e encaminhado diretamente ao tribunal competente.

Para Recurso Extraordinário, este deve ser apresentado ao Presidente ou Vice-Presidente do Tribunal em que foi recorrida a decisão. O RE deve conter as informações sobre as razões do pedido, a exposição do quê ocorreu associado a regra do direito e

discorrer sobre a validade da interposição do recurso. Quando houver recursos simultâneos ao STJ e STF, o processo deve, primeiramente, ser encaminhado ao STJ (BRASIL, 2015).

As características procedurais de um Agravo em Recurso Extraordinário (ARE) são iguais às de um RE. A distinção entre ambas as peças são seus conteúdos, no qual o RE será relacionado a alguma inconstitucionalidade e o ARE será um contraponto à inadmissão do recurso.

3.3.3 Instância superior - Supremo Tribunal Federal

A decisão proferida pelos ministros do STF sobre a Repercussão Geral (RG) são irrecorríveis. Caso haja RG, ou seja, ele cumpre os requisitos do § 1º do Artigo n.º 1.035 do CPC (Lei n.º 13.105/2015) "Para efeito de repercussão geral, será considerada a existência ou não de questões relevantes do ponto de vista econômico, político, social ou jurídico que ultrapassem os interesses subjetivos do processo" (BRASIL, 2015) o processo é distribuído para que os Ministros do Tribunal possam julgar a questão do RE.

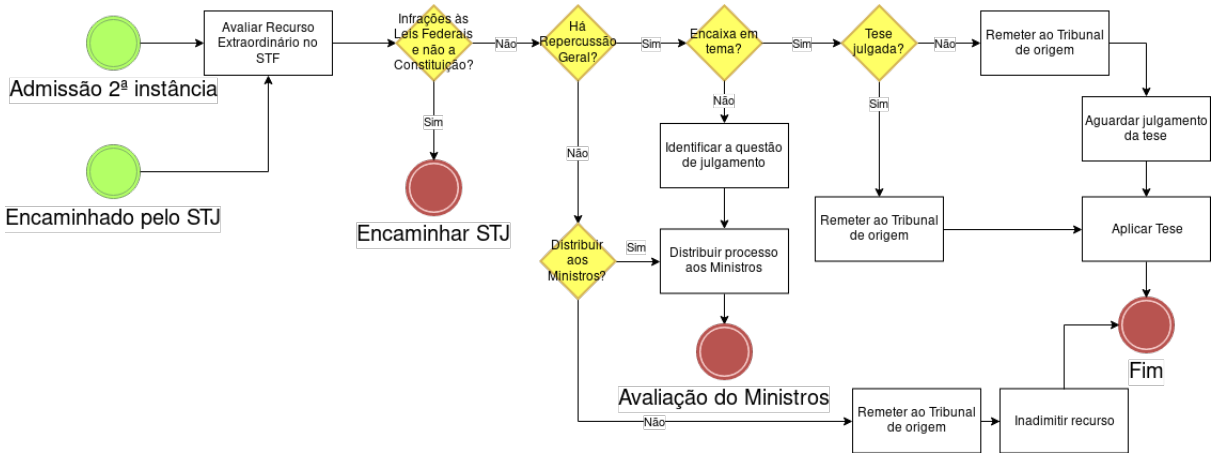

Figura 17 – Processo judiciário - Instância Superior. Fonte: elaboração própria

Na figura 17, a atividade de Avaliação do Ministro é um outro conjunto de atividades descritas no Regimento Interno do STF (BRASIL, 2016) e na Constituição Federal (1988).

Existem outras atividades desenvolvidas pelo STF que não foram apresentadas na Figura 17. Estas atividades estão relacionadas ao Regimento Interno (BRASIL, 2016) deste Tribunal e o que eles fazem com as peças. Ou seja, não há produção de novos documentos internamente os quais sejam analisados para definir se há ou não Repercussão Geral.

4 Os dados

Neste capítulo será tratada as características dos dados disponíveis para este trabalho.

4.1 Extração dos textos

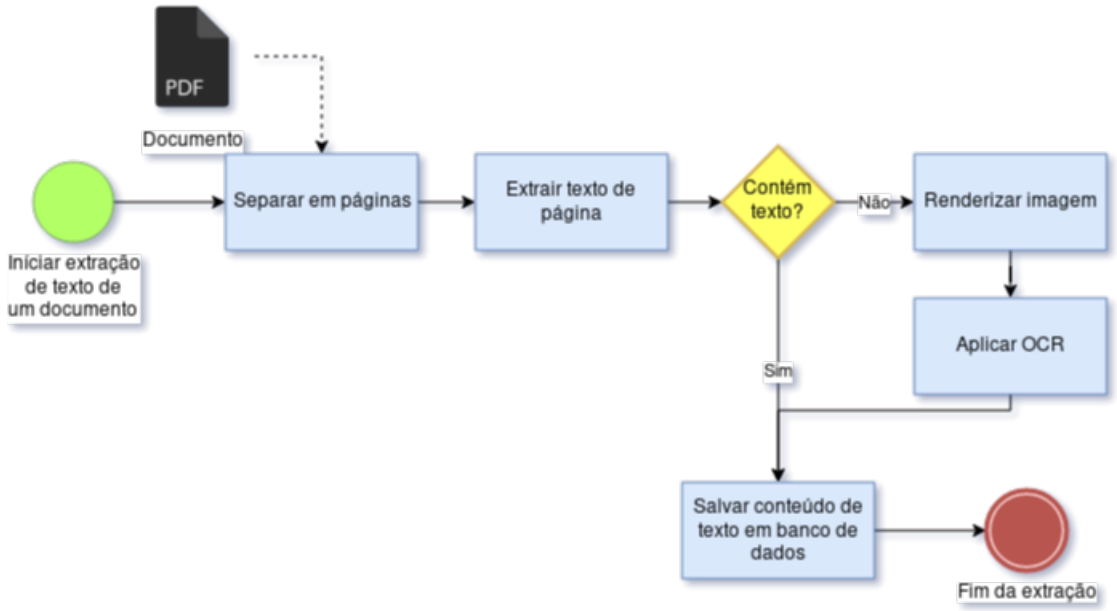

Figura 18 – Procedimento para extração de textos. São aceitos volumes ou peças separadas. O texto extraído pode é salvo em uma base de dados. Fonte: elaboração própria

O STF disponibilizou os processos e suas peças no formato PDF. Nos arquivos recebidos, haviam volumes e peças já separadas.

As etapas necessárias para realizar a extração de documentos está representada na Figura 18, onde utilizou-se da ferramenta XpdfReader[1] para extrair o texto das páginas de um PDF que estava em formato digital ou digitalizado com uma camada de texto.

Quando encontrada uma página sem texto, transformou-se esta em uma imagem para, em seguida, aplicar um reconhecedor ótico de caracteres (do inglês *Optical Character Recognition* OCR).

[1] Executou-se os comandos **pdftotext** e **pdftopng**. Disponível em: <https://www.xpdfreader.com/pdftotext-man.html>. Acesso em: 17-06-2018.

4.2 Tratamento dos textos

Utilizou-se apenas parte dos dados para realizar a análise. Fez-se a limpeza utilizando expressões regulares (GOYVAERTS; LEVITHAN, 2012) a fim de:

- Remover caracteres especiais, tais como: # @ \n 0xCE
- Remover números misturados com letras.
- Remover números
- Capturar leis, artigos e decretos.
- Remover espaços em branco
- Remover e-mail e *link*.

Em seguida, aplicou-se as técnicas de radicalização, normalização (SINGH; GUPTA, 2016) e remoção de palavras recorrentes (MANNING; RAGHAVAN; SCHÜTZE, 2008).

Além das palavras recorrentes, foram removidos nomes de pessoas e algumas palavras chaves específicas: 'elytho', 'neve', 'chenaud', 'tayrone', 'besen', 'youngéqyoung', 'sarubbi', 'balogh', 'pezarini', 'zezinho', 'intdo', 'izmailov', 'zotto', 'angelicoadvogados', 'limar', 'steiger', 'acidelma', 'vitoriar', 'tic', 'valcy', 'dadico', 'aloisia', 'itos', 'tendolo', 'rossol', 'catapani', 'cleudes', 'her', 'araçatuba', 'boeing', 'melar', 'rs', 'onaita', 'britar', 'ferreiro', 'licht', 'josé'.

Os nomes de pessoas coletados para adicionar na remoção de palavras, foram obtidos através da mineração dos dados abertos do governo[2], com o procedimento do Apêndice A.

4.3 Características dos dados

Como abordado na Seção 1.1, os rótulos das peças já separadas advindas do STF não são confiáveis. Por conta disso, especialistas da área jurídica iniciaram o rotulamento adequado a estes documentos, classificando apenas em 7 categorias e Outros, apresentados na Tabela 7.

A quantidade de documentos obtidos são apresentados na Tabela 7. Nela, percebe-se que as peças de Agravo, Sentença, Petição de Agravo, Despacho de Agravo e Recurso Extraordinário estão com uma contagem não distribuída.

[2] Arquivo da data de Janeiro/2017 dos gastos diretos. Disponível em: <http://www.portaldatransparencia.gov.br/downloads/mensal.asp?c=FavorecidosGastosDiretos>. Acesso em: 17-06-2018

Tabela 7 – Quantidade de cada peça.

Tipo	**Quantidade total**	**tamanho relativo**
Outro	2.285	× 142,81
Agravo de Recurso Extraordinário	1795	× 112,18
Acordão	1.568	× 98,00
Sentença	252	× 15,75
Despacho	89	× 5,56
Petição de Agravo	63	× 3,93
Despacho de Agravo	26	× 1,62
Recurso Extraordinário	16	× 1,00
Total	6094	-

Fonte: elaboração própria.

Após o tratamento dos dados, buscou-se por documentos com o mesmo conteúdo que tivessem classificações diferentes. Neste cenário, obteve-se 428 (7.00%) documentos com confusão. As categorias que se confundiram foram as tuplas Outros e Acórdão, Agravo de Recurso Extraordinário e Recurso Extraordinário, Despacho e Outros, Acórdão e Sentença, e por último Petição de Agravo e Outros.

Como resultado do pre-processamento, obteve-se os resultados da Tabela 8, na qual transformou-se os textos em símbolos para realizar as contagens de símbolos.

Tabela 8 – Métricas dos documentos.

Nome	**Valor**
Menor número de símbolos	113
Maior número de símbolos	7.036
Média de símbolos por documento	1.113,17
Total de símbolos	6.783.709
Tamanho do vocabulário	20.009

Fonte: elaboração própria.

Antes de treinar um modelo classificador, fez-se uma correlação entre as classes com o BoW relativo ao total de cada palavra (Equação 4.1. Ou seja, o corpo de texto de cada tipo de documento foi concatenado, logo após, a contagem de palavras em cada tipo foi dividido pelo total de ocorrências em todos os documentos. Na Equação 4.1, o i representa o tipo de documento e o j representa a palavra pertencente ao dicionário.

$$BoWrelative_{i,j} = \frac{BoW_{i,j}}{BoWt_j} \times 100 \tag{4.1}$$

A partir desse resultado, fez-se a correlação de spearman para gerar uma matriz triangular. Os valores para o coeficiente de spearman tem os intervalos entre $[-1, 1]$. Os dados observados na Figura 19 mostram que não há forte correlação entre os documentos, na qual o maior valor presente é a correlação inversa entre Outros e a peça ARE.

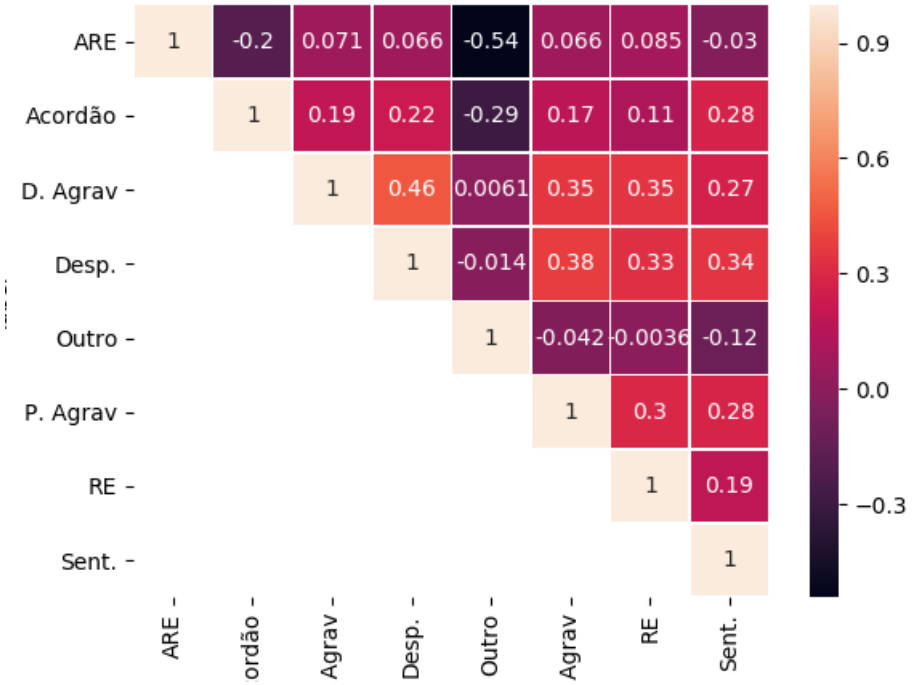

Figura 19 – Mapa de calor para correlação entre peças. Fonte: GPAM [3]

Mesmo com a duplicação de rótulos nos documentos, utilizou-se a implementação do SVM Linear (HEARST et al., 1998) para treinar com todos os dados, a fim de obter uma caracterização das palavras mais importantes. Os parâmetros utilizados foram os padrões de acordo com a implementação de PEDREGOSA et al. (2011), modificando apenas o número máximo de iterações para 50.000, pois somente com este valor que garantiu-se a convergência do modelo.

A figura 20 mostra que o modelo conseguiu separar bem as peças, e portanto, a Tabela 9 mostra as características mais importantes extraídas dos vetores de suporte (HEARST et al., 1998).

Muitas palavras que não fazem sentido para o contexto do conteúdo das peças, como nomes próprios, apareceram de forma muito frequente durante a classificação (Exemplo: 'piauí', 'ubaldino', 'glacy', 'crisanto'). Apesar disso, as peças ARE, RE, Sentença e Despacho de Agravo mostraram palavras de importância significativas. Na tabela 9, as palavras relevantes de acordo com a Seção 3.3 estão em negrito.

[3] O Grupo de Pesquisa em Aprendizado de Máquina (GPAM) está envolvido na tarefa de classificar peças para o STF. Esta imagem foi gerada para a exploração de dados realizada no projeto. Elaborado pelos membros: Davi Alves Bezerra e Davi Benevides Gusmão.

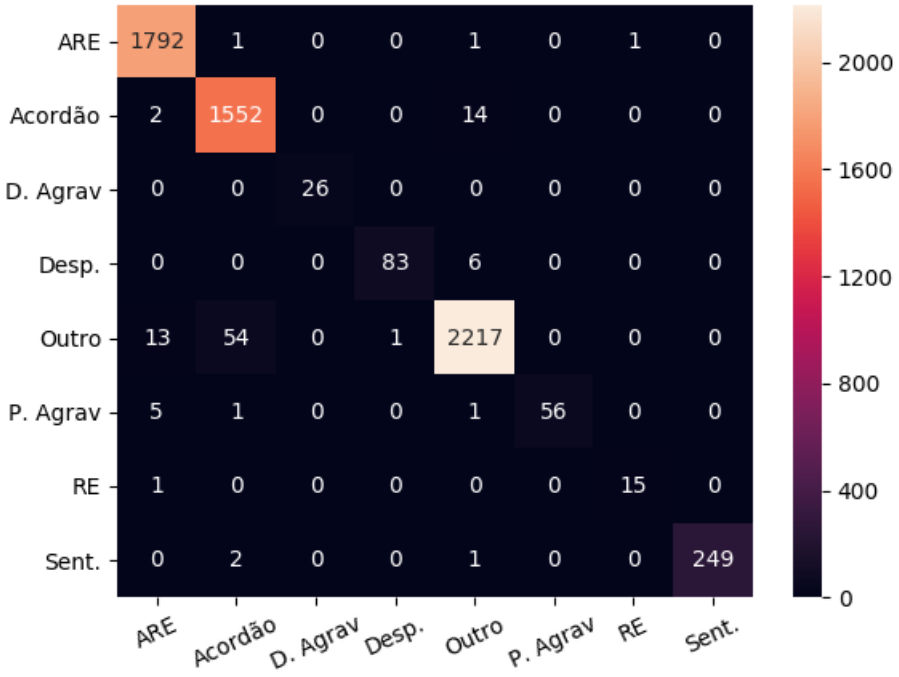

Figura 20 – Matriz de confusão para extração de características usando SVM. Fonte: elaboração própria

Tabela 9 – Características importantes extraídas dos vetores de suporte.

Pos	**Agr. RE**	**Acórdão**	**D. Agr.**	**Desp.**	**Outro**	**P. Agr.**	**RE**	**Sentença**
1	https	liar	monsani	marmelo	ubaldino	sulmar	**artigo__102**	montrazi
2	**extraordinário**	chegar	municipio	rollin	sumie	glacy	**constituição**	**inspeção**
3	quintar	borrar	dezembro	aminadabe	minhos	bradesco	iii	**decidir**
4	bandeiro	pg	município	cear	membro	**ag**	presentar	requis
5	toste	piauí	vigência	spprev	crispin	vencedor	**venia**	**dispensar**
6	muciciaria	filhar	observância	previdencia	rizar	marcar	objeto	**artigo__42**
7	unimed	**recurso**	**instrumentar**	líbero	parcial	**regimental**	alínea	invalidez
8	**federar**	carrá	blumenau	cásper	cleci	sortear	**repercussão**	**fundamentar**
9	acessar	unanimidade	prever	itau	vanin	paraiba	incisar	moléstia
10	quo	universo	termo	**recuperar**	ller	crisanto	dispositivo	expor
11	**juizado**	agostar	**lei__13256**	recuperação	evori	ferrar	**recursal**	**postular**
12	novembro	batistuzo	**regimental**	creditos	pastar	incs	respeito	**produzir**
13	diretoria	eichherr	assistir	fundação	especial	aguardar	presença	representante
14	denegatório	dezembro	recursais	pintar	**agravo**	pf	**juizados**	julgamento
15	trâmite	sangiogo	**presidência**	representação	**artigo__48**	freudenthal	cabível	rocar
16	segurar	autoprev	base	artigo_25	suprir	obrigação	taubaté	íntegro
17	eletronicamente	bitello	união	veiculos	**preâmbulo**	servidor	**razão**	capacidade
18	enviar	entretanto	alterar	mjr	**cnj**	paraíba	exa	**código**
19	**processual**	registrar	**lei__13105**	nelcis	panamericano	efone	prolatado	intuito
20	**artigo__544**	sedimentar	repúblico	quantum	despachar	**previsto**	egrégio	procurador

Fonte: Elaboração própria.

5 Conclusão

O processo elaborado para projeto de pesquisa que envolva pesquisa em Aprendizado de Máquina, mostrou-se efetivo para o planejamento da pesquisa. Executou-se todas as atividades previstas, que concretizaram-se na Introdução, Referencial Teórico e Dados.

O estudo feito para os diferentes tipos de peças, envolveu o entendimento da organização do Poder Judiciário brasileiro e o do CPC (Lei 13.105/2015). O responsável pela confecção da peça, o tipo de conteúdo que ela contém, a quem é destinada e quais são as peças necessárias para a avaliação de Repercussão Geral foram detalhadas para ter um bom entendimento do domínio do problema.

A hipótese apresentada na Seção 1.2, de que não há textos iguais para peças diferentes, mostrou-se falsa após a aplicação do tratamentos de dados exposto no Apêndice B. Portanto, para a finalização deste trabalho, é necessário remover estes multi rótulos.

As palavras chaves obtidas com o uso do SVM não se mostraram tão efetivas. Com o melhoramento do processamento de texto e mais amostras de documentos rotulados, espera-se melhorar as palavras chaves. Além disso, a correlação entre os tipos de documentos mostram já demostram sua separabilidade.

Para o trabalho futuro, espera-se melhorar a engenharia de características dos dados e aplicar os diferentes modelos para classificação.

Referências

AMENDOEIRA JR, S. *Manual de direito processual civil*: Teoria geral do processo e fase de conhecimento em primeiro grau de jurisdição. 2. ed. São Paulo: Editora Saraiva, 2012. v. 1. ISBN 978-85-02-12710-4. Citado 2 vezes nas páginas 44 e 45.

BRASIL. *Constituição da República Federativa do Brasil*. 1988. Brasília, DF. Citado 7 vezes nas páginas 17, 18, 19, 43, 44, 45 e 48.

BRASIL. *Lei Nº 8.457, de 4 de setembro de 1992*. Brasília, DF, 1992. Disponível em: <https://www.planalto.gov.br/ccivil_03/leis/l8457.htm>. Acesso em: 26-05-2018. Citado 2 vezes nas páginas 43 e 45.

BRASIL. *Lei Nº 12.665, de 13 de junho de 2012*. Brasília, DF, 2012. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2011-2014/2012/lei/l12665.htm>. Acesso em: 26-05-2018. Citado 2 vezes nas páginas 43 e 45.

BRASIL. *Lei Nº 13.105, de 16 de março de 2015. Código de Processo Civil*. 2015. Brasília, DF. Disponível em: <www.planalto.gov.br/ccivil_03/_ato2015-2018/2015/lei/l13105.htm>. Acesso em: 27-05-2018. Citado 3 vezes nas páginas 46, 47 e 48.

BRASIL. Congresso Nacional. *Lei Nº 12.527, de 18 de novembro de 2011*. Brasília, DF, 2011. Citado na página 18.

BRASIL. Conselho Nacional de Justiça. *Termo de acordo de cooperação técnica Nº 058/2009*. Brasília, DF, 2009. 1-5 p. Citado na página 18.

BRASIL. Conselho Nacional de Justiça. *Wiki PJe*. 2018. Disponível em: <http://www.pje.jus.br/wiki/index.php/P%C3%A1gina_principal>. Acesso em: 18-05-2018. Citado na página 18.

BRASIL. Supremo Tribunal Federal. *Resolução Nº 427, de 20 de abril de 2010*. Brasília, DF, 2010. 1-6 p. Citado na página 20.

BRASIL. Supremo Tribunal Federal. *Regimento Interno*. Brasília, DF, 2016. 1-166 p. Citado na página 48.

BRINK, H.; RICHARDS, J. W.; FETHEROLF, M. *Real-World Machine Learning*. Meap. [S.l.]: Manning, 2015. 2-26 p. Citado 6 vezes nas páginas 17, 26, 27, 33, 41 e 42.

BUTNARU, A. M.; IONESCU, R. T. From image to text classification: A novel approach based on clustering word embeddings. *Procedia Computer Science*, Marseille, France, v. 112, p. 1783–1792, September 2017. ISSN 1877-0509. Citado na página 35.

CHAPMAN, P. et al. *CRISP-DM 1.0*: Step-by-step data mining guide. [S.l.], 2000. Disponível em: <https://www.the-modeling-agency.com/crisp-dm.pdf>. Acesso em: 20-05-2018. Citado 2 vezes nas páginas 25 e 26.

CONNEAU, A. et al. Very deep convolutional networks for text classification. In: *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational

Linguistics, 2017. v. 1, p. 1107–1116. Disponível em: <http://aclweb.org/anthology/E17-1104>. Acesso em: 24-06-2018. Citado na página 39.

CROWSTON, K.; SALTZ, J. S.; SHAMSHURIN, I. Comparing data science project management methodologies via a controlled experiment. In: *Proceedings of the 50th Hawaii International Conference on System Sciences*. Mānoa, Hawaii: [s.n.], 2017. p. 10. ISBN 978-0-9981331-0-2. Citado 2 vezes nas páginas 26 e 27.

ENRÍQUEZ, F.; TROYANO, J. A.; LÓPEZ-SOLAZ, T. An approach to the use of word embeddings in an opinion classification task. *Expert Systems with Applications*, v. 66, p. 1–6, 2016. ISSN 0957-4174. Citado na página 35.

ESLICK, I.; LIU, H. Langutils: A natural language toolkit for common lisp. In: *Proceedings of the International Conference on Lisp*. Stanford, California: [s.n.], 2005. Citado na página 17.

GE, L.; MOH, T.-S. Improving text classification with word embedding. In: IEEE. *Big Data (Big Data), 2017 IEEE International Conference on*. Boston, MA, 2017. p. 1796–1805. Citado na página 35.

GIL, A. C. *Como elaborar projetos de pesquisa*. 4ª. ed. São Paulo, SP: Atlas S.A., 2002. 23-85 p. ISBN 85-224-3169-8. Citado 2 vezes nas páginas 20 e 24.

GOLDBERG, Y. *Neural Network Methods for Natural Language Processing*. [S.l.]: Morgan & Claypool, 2017. 1-142 p. ISBN 978-16-27052-95-5. Citado 8 vezes nas páginas 17, 34, 36, 37, 38, 39, 40 e 41.

GONÇALVES, M. V. R. *Direito processual civil esquematizado*. 6. ed. São Paulo: Saraiva, 2016. Citado na página 46.

GOYVAERTS, J.; LEVITHAN, S. *Regular Expressions Cookbook*. Second. Sebastopol, CA: O'Reilly Media, 2012. 1-19 p. ISBN 978-1-449-31943-4. Citado 2 vezes nas páginas 32 e 50.

GUIMARÃES, D. T. *Dicionário compacto jurídico*. 16. ed. São Paulo: Rideel, 2012. ISBN 978-85-339-2023-1. Citado na página 45.

HEARST, M. A. et al. Support vector machines. *IEEE Intelligent Systems and their applications*, IEEE, v. 13, n. 4, p. 18–28, 1998. Citado 3 vezes nas páginas 35, 36 e 52.

KIM, Y. Convolutional neural networks for sentence classification. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, 2014. p. 1746–1751. Citado 2 vezes nas páginas 39 e 40.

MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN 0521865719, 9780521865715. Citado 3 vezes nas páginas 31, 32 e 50.

MIKOLOV, T. et al. Recurrent neural network based language model. In: *Eleventh Annual Conference of the International Speech Communication Association*. [S.l.: s.n.], 2010. Citado 2 vezes nas páginas 40 e 41.

MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. USA: Curran Associates Inc., 2013. (NIPS'13), p. 3111–3119. Disponível em: <http://dl.acm.org/citation.cfm?id=2999792.2999959>. Acesso em: 24-06-2018. Citado na página 34.

MIKOLOV, T.; YIH, W. tau; ZWEIG, G. Linguistic regularities in continuous space word representations. In: *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. [S.l.: s.n.], 2013. p. 746–751. Citado na página 34.

NIELSEN, M. A. *Neural networks and deep learning*. Determination Press, 2015. 50-76 p. Disponível em: <http://neuralnetworksanddeeplearning.com/chap2.html>. Acesso em: 12-06-2018. Citado 3 vezes nas páginas 37, 38 e 39.

OLIVEIRA, E.; FILHO, D. B. Automatic classification of journalistic documents on the internet. *Transinformação*, scielo, v. 29, p. 245 – 255, 12 2017. ISSN 0103-3786. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0103-37862017000300245&nrm=iso>. Acesso em: 16-06-2018. Citado 2 vezes nas páginas 17 e 31.

PEDREGOSA, F. et al. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, v. 12, p. 2825–2830, 2011. Citado 3 vezes nas páginas 35, 36 e 52.

PENNINGTON, J.; SOCHER, R.; MANNING, C. D. Glove: Global vectors for word representation. In: *Empirical Methods in Natural Language Processing (EMNLP)*. [s.n.], 2014. p. 1532–1543. Disponível em: <http://www.aclweb.org/anthology/D14-1162>. Citado na página 34.

PRODANOV, C. C.; FREITAS, E. C. de. *Metodologia do trabalho cientifico*: Métodos e técnicas da pesquisa e do trabalho acadêmico. 2ª. ed. Novo Hamburgo, RS: Editora Feevale, 2013. 13-141 p. ISBN 978-85-7717-158-3. Citado 3 vezes nas páginas 23, 24 e 25.

RASCHKA, S. *Single-Layer Neural Networks and Gradient Descent*. 2015. Disponível em: <https://sebastianraschka.com/Articles/2015_singlelayer_neurons.html>. Acesso em: 14-06-2018. Citado 2 vezes nas páginas 36 e 37.

RODRÍGUEZ, L. C.; CASTAÑO, E. P.; SAMBLÁS, C. R. Quality performance metrics in multivariate classification methods for qualitative analysis. *TrAC Trends in Analytical Chemistry*, v. 80, p. 612–624, 2016. ISSN 0165-9936. Citado na página 43.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, p. 386–408, 1958. Citado na página 36.

RUSCHEL, A. J.; ROVER, A. J.; SCHNEIDER, J. Governo eletrônico: o judiciário na era do acesso. In: *CALLEJA, P.L. (Org). La Administración Electrónica como Herramienta de Inclusión Digital*. Zaragoza, Espanha: Zaragoza: Prensas Universitarias de Zaragoza, 2011, (LEFIS series; 13). p. 59 – 79. ISBN 978-84-15274-66-7. Citado na página 18.

SINGH, J.; GUPTA, V. Text stemming: Approaches, applications, and challenges. *ACM Comput. Surv.*, ACM, New York, NY, USA, v. 49, n. 3, p. 45:1–45:46, set. 2016. ISSN 0360-0300. Disponível em: <http://doi-acm-org.ez54.periodicos.capes.gov.br/10.1145/2975608>. Citado 2 vezes nas páginas 32 e 50.

SMOLA, A. J.; SCHöLKOPF, B. A tutorial on support vector regression. *Statistics and Computing*, Kluwer Academic Publishers, Hingham, MA, USA, v. 14, n. 3, p. 199–222, ago. 2004. ISSN 0960-3174. Disponível em: <https://doi.org/10.1023/B:STCO.0000035301.49549.88>. Acesso em: 24-06-2018. Citado na página 36.

Apêndices

APÊNDICE A – Coleta de nomes dos dados abertos

Primeiramente, foi necessário baixar os dados no formato CSV para a máquina local. Em seguida, executar o código python.

```
# Validator for CPF
def validate_cpf(cpf):
    cpf=str(cpf)
    if len(cpf) != 11:
        return False

    # CPF in form 00000000000 11111111111 ... 99999999999
    # are invalid
    if cpf in [11 * str(i) for i in range(10)]:
        return False

    # (10 * num1 + 9 * num2 + ... + 2 * num9) * 10 % 11
    # if rest == 10, result = 0
    zip_numbers = zip(range(10, 1, -1), cpf[:10])
    sumatory = [ x * int(i) for x, i in zip_numbers]
    result = ((sum(somatory) * 10) % 11) % 10
    if result != int(cpf[9]):
        return False

    # (11 * num1 + 10 * num2 + ... + 2 * num10) * 10 % 11
    # if rest == 10, result = 0
    zip_numbers = zip(range(11, 1, -1), cpf[:11])
    somatory = [ x * int(i) for x, i in zip_numbers]
    result = ((sum(somatory) * 10) % 11) % 10

    return result == int(cpf[10])

import pandas as pd

dataFrame = pd.read_csv('201701_GastosDiretos.csv',
        encoding='Latin-1', sep='\t')
```

```
# Create a new Column in data frame to define if the
# 'Codigo Favorecido' is a CPF
dataFrame['Codigo CPF Valido'] = dataFrame.progress_apply(
    lambda x: validate_cpf(x['Codigo Favorecido']),
    axis=1
)

# Filter data frame droping all unnecessary columns (axis = 1)
dataFrame = dataFrame[dataFrame['Codigo CPF Valido']]
dataFrame['name'] = dataFrame['Nome Favorecido']

# Generate a new data frame with names and surnames splited
names = dataFrame.name.str.cat(sep=' ').lower().split()
namesDF = pd.DataFrame(names, columns=['name'])
namesDF = namesDF.sort_values('name')

# Resize the data frame and add a new column of occurencies of the name
namesDF = namesDF.groupby('name').size().reset_index(name='counts')
namesDF = namesDF[namesDF.name.str.len() > 2]

# Persist data to be used in other codes
namesDF.to_csv('names.csv', index=False)
```

APÊNDICE B – Limpeza dos dados

O código abaixo foi utilizado para aplicar as expressões regulares, as normalizações e a transformação em símbolos. Este código foi desenvolvido pelo Grupo de Pesquisa em Aprendizado de Máquina da Universidade de Brasília e foi adicionado aqui por estar em um repositório de código privado.

```
import re
from spacy.lang import pt
import nltk
from nltk.stem.snowball import SnowballStemmer

try:
    nltk.word_tokenize('some word')
except:
    nltk.download('punkt')

try:
    nltk.corpus.stopwords.words('portuguese')
except:
    nltk.download('stopwords')
finally:
    STOP_WORDS = pt.STOP_WORDS.union(
        set(nltk.corpus.stopwords.words('portuguese'))
    )


class CorpusHandler:

    def __init__(self):
        pass

    @staticmethod
    def clean_number(document, **kwargs):
        return re.sub(r'\s\d+\s', ' ', document)

    @staticmethod
    def clean_email(document, **kwargs):
```

```
        local_part = r"[0-9a-zA-Z!#$%&'*+-/=?^_`{|}~.]+"
        domain = r"[a-zA-Z][a-zA-Z0-9-.]*[a-zA-Z]\.\w{2,4}"
        document = re.sub(r"\s{}@{}".format(
                local_part, domain),
            ' EMAIL ', document)
        return document

    @staticmethod
    def clean_site(document, **kwargs):
        # Addaption for rules of
        # https://tools.ietf.org/html/rfc3986#section-3 to
        # stf's documents
        scheme = r"[a-zA-Z][a-zA-Z0-9+-.]*://?"
        host_ip = r'\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}'
        host_name = r"[a-zA-Z]+(\.[a-zA-Z0-9-_~]+)+"
        www = r'www\.{}'.format(host_name)
        scheme_host = r'({}{}|{}{}|{}{}|{})'.format(
            scheme, host_ip, scheme, www, scheme, host_name, www
        )
        port = r"(:\d+)?"
        resource = r"(/[a-zA-Z0-9-._~!$&'/*+,;=]*)?"
        query = r"(\?[a-zA-Z0-9-._~!$&'/*+,;=]*)?"
        fragment = r"(#[a-zA-Z0-9-._~!$&'/*+,;=]*)?"
        document = re.sub(r"{}{}{}{}{}".format(
                scheme_host, port, resource, query, fragment),
            ' SITE ', document)
        return document

    @staticmethod
    def transform_token(document, **kwargs):
        word_number = r'(n.?|numero|numero|n.?|no.?)?'
        # Law number: 00/YEAR or 00
        number_law = r'([0-9]+)((\s|\.)+\d+)?'
        matchs = [('LEI', 'lei'), ('ARTIGO', r'art\.|art\w*'),
                  ('DECRETO', 'decreto'),]

        for word, regex in matchs:
            document = re.sub(r'{}\s*{}\s*{}'.format(
                    regex, word_number, number_law),
```

```
                r'{}_\2'.format(word),
                document, flags=re.I)

        return document

    @staticmethod
    def remove_small_big_words(document, **kwargs):
        # remove 2 chars
        document = re.sub(r'\s\w{0,2}\s', '␣', document)
        # remove bigger words ex.: infrainconstitucionalidade
        document = re.sub(r'\s\w{30,}\s', '␣', document)
        return document

    @staticmethod
    def remove_letter_number(document, **kwargs):
        # Remove wor00 00wo 00wor00 wo00rd and keep WORD_000
        return re.sub(r'([A-Z]+_\d+)|[^␣]*\d+[^␣]*', r'\1', document)

    @staticmethod
    def clean_document(document, **kwargs):
        # Replace 0.0 for 00
        document = re.sub(r'(\d)\.(\d)', r'\1\2', document)

        # Remove all non alphanumeric
        document = re.sub(r'\W', '␣', document)

        return document

    @staticmethod
    def clean_spaces(document, **kwargs):
        # Remove multiple spaces
        document = re.sub(r'\s+', '␣', document)
        document = document.strip()
        return document

    @staticmethod
    def clean_alphachars(document, **kwargs):
        return re.sub(r'[^␣]*[^-_\'u\'i\'o\~o\^o\'e\^e\~a'
          '\'a\`a\^aa-z0-9\{\c␣c}␣]+[^␣]*',
```

```
                ' ', document)

    @staticmethod
    def tokenize(document, **kwargs):
        """Transform string documents in array of tokens"""
        if isinstance(document, str):
            document = document.split()
        return document

    @classmethod
    def remove_stop_words(cls, document, stop_words=[],
        extra_stop_words=[], **kwargs):
        tokens = cls.tokenize(document)
        words = set(stop_words) or STOP_WORDS
        if extra_stop_words != []:
            words = words.union(set(extra_stop_words))
        document = " ".join(filter(lambda x: x not in words, tokens))
        return document

    stemmer = SnowballStemmer("portuguese")
    @classmethod
    def snowball_stemmer(cls, document, **kwargs):
        """Use nltk Snowball Stemmer to stemmize words"""
        tokens = cls.tokenize(document)
        document = ' '.join([cls.stemmer.stem(word) for word in tokens])
        return document
```