Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uma comparação entre métodos e classificadores em documentos jurídicos de atividades processuais repetitivas na PGDF

Raul Carvalho de Souza

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Orientador
Prof. Dr. Thiago de Paulo Faleiros

Brasília
2021

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

CARVALHO DE SOUZA, RAUL
Cc Uma comparação entre métodos e classificadores em documentos jurídicos de atividades processuais repetitivas na PGDF / RAUL CARVALHO DE SOUZA; orientador THIAGO DE PAULO FALEIROS. -- Brasília, 2021.
108 p.

Dissertação (Mestrado - Mestrado Profissional em Computação Aplicada) -- Universidade de Brasília, 2021.

1. Algoritmos combinados. 2. Classificação de documentos jurídicos. 3. Aprendizagem de máquina. 4. Inteligência Artificial. 5. Computação Aplicada. I. DE PAULO FALEIROS, THIAGO, orient. II. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Uma comparação entre métodos e classificadores em documentos jurídicos de atividades processuais repetitivas na PGDF

Raul Carvalho de Souza

Dissertação apresentada como requisito parcial para conclusão do
Mestrado Profissional em Computação Aplicada

Prof. Dr. Thiago de Paulo Faleiros (Orientador)
CIC/UnB

Dr. Jorge Carlos Valverde Rebaza Prof. Dr. Marcelo Ladeira

Prof. Dr. Marcelo Ladeira
Coordenador do Programa de Pós-graduação em Computação Aplicada

Brasília, 28 de julho de 2021

Dedicatória

À Deus e a minha família!

Agradecimentos

Agradeço a Deus, meus pais João Pereira e Maria, querida mamãe. Agradeço a minha esposa Brenda e filha Maria Valentina. Agradeço especialmente ao meu orientador prof. Thiago de Paulo Faleiros. Eu não teria conseguido sem sua orientação. Agradeço aos professores Leonardo e Blanca Lazarte, que me apoiaram na vida acadêmica e até na vida pessoal. Obrigado aos professores Jorge Fernandes e Marcelo Ladeira por me ajudarem a chegar até aqui nos estudos. Muito agradecido! Ao professor Donald Pianto pela orientação parcial. Uma pena que meu tema anterior não foi aprovado pelo Comitê de Ética da UnB. Isso me deixou muito desmotivado, mas com a ajuda de todos, principalmente dos meus colegas de linha de pesquisa no curso, fui adiante. Felizmente com a ajuda de todos que me rodeiam superei essa angústia. Certamente a responsabilidade é minha, só eu sou responsável pelo que penso, falo e faço. Agradeço aos professores Jorge Carlos Valverde Rebaza e Guilherme Novaes Ramos pelos comentário e sugestões na qualificação, foram bem importante para a evolução do trabalho. Agradeço aos meus parceiros de trabalho Arthur, Douglas, Paulo e Ricardo, que me apoiaram nas ausências para cumprir requisitos do curso. Agradeço a minha chefia imediata Riane, sem ela eu não teria conseguido mudar de tema e conseguir concluir. Agradeço especialmente aos colegas Augusto, Grazielly e Lorenza que me ajudaram no entendimento do negócio, fazendo o papel de especialistas. Agradeço ao sr. Rafael, que na época trabalhava na empresa contratada do sistema de gestão de documentos jurídicos da Procuradoria Geral do Distrito Federal (PGDF). Agradeço a sra. Vanessa Rocha e Orlando do TJDFT, que me ajudaram nas integrações com o Modelo Nacional de Interoperabilidade (MNI) do Tribunal de Justiça do Distrito Federal e dos Territórios (TJDFT). Obrigado a todos!

A dor passou,
vagou ao concluir.
Foi ali e já volta.
Como vaga!
Re volta!

A razão,
conclusão absoluta,
tem só um Amo.
Se Amo, não há engano.

Sentimento
Questão resoluta.
Coisa de momento, afã, alegria ou tormento.
Machuca.

"Atenta para a obra de Deus:
Quem poderá endireitar o que ele fez torto?"
– Eclesiastes 7: 13.

"O direito é a coação universal que protege a liberdade de todos."
- IMMANUEL KANT

"Nem tudo o que é torto é errado.
Veja as pernas do Garrincha e as árvores do cerrado"
- Nicolas Behr

"A verdade absoluta não pertence aos mortais"
- João Pereira de Souza(meu querido pai!)

Resumo

Os processos judiciais repetitivos, aqueles que exigem muitas ações manuais e menos ações intelectuais humanas, são uma realidade maçante nos trabalhos da Procuradoria Geral do Distrito Federal (PGDF). Geralmente esses processos repetitivos são causas conhecidas que o Governo do Distrito Federal (GDF) faz parte em muitas ações, corriqueiramente peticionadas de uma só vez. Essa repetitividade causa muito trabalho manual, desperdício de recursos e alta contenção desses processos para a identificação e resposta por parte da PGDF. Por isso, há necessidade de otimizar recursos e acelerar o andamento dos processos, bem como as análises relacionadas a eles, para tentar mudar essa situação de contenção. Assim, este trabalho busca a assistência do computador em tarefas rotineiras de PGDF, que atualmente gastam muito dinheiro em recursos humanos para agrupar e classificar esses processos repetitivos. A expectativa da organização com os avanços deste trabalho é de se evoluir as automatizações para acelerar os processos e economizar recursos, já que o computador deve realizar as tarefas de classificação em uma velocidade maior, em maior volume do que o processo atual e com o uso de menos pessoas. A principal motivação deste trabalho é encontrar um método eficiente de classificação de documentos jurídicos. Para este fim, além de experimentos com diversos algoritmos de aprendizagem de máquina para classificação em diversos dados da PGDF, iniciando pelos precatórios, é feita uma comparação entre os melhores modelos que classificaram documentos jurídicos na PGDF. O esforço utilizado neste trabalho é para encontrar o melhor método dentro do contexto do PGDF para a classificação de documentos jurídicos. O trabalho conseguiu acelerar o processo de classificação.

Palavras-chave: Algoritmos combinados, Classificação de documentos jurídicos, Aprendizagem de máquina

Abstract

Repetitive lawsuits, those that require a lot of manual work and less human intellectual activity, are a massive reality in the Federal District Attorney General (PGDF) day by day. Generally, these repetitive lawsuits are known litigation issues in which the PGDF is part, routinely petitioned at once. This repetitiveness lawsuits causes a lot of manual work, waste of resources and high contention of these judicial processes, principally, for the identification and response by PGDF lawyers. Therefore, there is a need to optimize resources and speed up these judicial processes, as well as the analysis related to them, in order to try to change this contention situation. Thus, this work seeks computer assistance in routine tasks in PGDF, which currently spend a lot of money on human resources to group and classify these repetitive judicial processes without due velocity and precision. The expectation is to speed up the process of work and save resources, since the computer must perform the classification tasks at a higher speed, in greater volume than the current process and using fewer people. The main motivation of this work would be to find an efficient method of classifying legal documents. To this end, in addition to experiments with several machine learning algorithms for classification, a comparison will be made among these classifiers with the application of multi-class classification problem in legal documents from PGDF's business. The effort used in this work is to find the best one method within the context of the PGDF to classify legal documents.

Keywords: Ensemble Learning, Legal documents classification, Machine learning

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

BERT Bidirectional Encoder Representations from Transformers.

BLOB Binary Large OBject.

CNJ Conselho Nacional de Justiça.

CRISP-DM Cross Industry Standard Process for Data Mining.

CSV Comma Separated Values.

DGCEC Deep Genetic Cascade Ensembles of Classifiers.

DIPROJ Diretoria de Protocolo Judicial.

GDF Governo do Distrito Federal.

GUI Graphical User Interface.

IA Inteligência Artificial.

KNN K-Nearest Neighbors.

LDA Latent Dirichlet Allocation.

LSTM Long Short-Term Memory.

MNI Modelo Nacional de Interoperabilidade.

OCR Optical Character Recognition.

OVA One-versus-All.

PDF Portable Document Format.

PGCONS Procuradoria Geral do Consultivo.

PGCONT Procuradoria Geral do Contencioso.

PGDF Procuradoria Geral do Distrito Federal.

PGFAZ Procuradoria Geral da Fazenda Distrital.

PJe Processo Judicial Eletrônico.

PLN Processamento de Linguagem Natural.

PMI Pointwise Mutual Information.

RPV Requisição de Pequeno Valor.

RSL Revisão Sistemática de Literatura.

RTF Rich Text Format.

RUS Random Under-Sampling.

SEGER Secretaria Geral.

SGBD Sistema de Gerenciamento de Banco de Dados.

SMOTE Synthetic Minority Oversampling Technique.

SMOTEENN SMOTE and Edited Nearest Neighbors.

SVM Support Vector Machine.

TF-IDF Term Frequency-Inverse Document Frequency.

TJDFT Tribunal de Justiça do Distrito Federal e dos Territórios.

Capítulo 1

Introdução

Atualmente na Procuradoria Geral do Distrito Federal (PGDF) existe uma necessidade latente de se otimizar o uso de recursos humanos na atividade de classificação de processos judiciais. Isso para automatizar tarefas, diminuir as atividades repetitivas humanas e aumentar a quantidade de dados para análises estratégicas. Essa necessidade pode ser observada no trabalho[1] realizado por especialistas de negócio e técnicos da PGDF, que elaboraram projeto no sentido de estudar e implementar ferramenta tecnológica neste tema do aprendizado de máquina.

Sabendo desse interesse da organização de classificação de processos judiciais com o auxílio do computador, este trabalho procura classificar documentos jurídicos utilizando aprendizado de máquina. Para isso, experimentou-se várias técnicas de pré-processamento em bases de dados desbalanceadas e comparou classificadores computacionais utilizando dados da PGDF.

A classificação de processos judiciais é a atividade que denomina um conjunto de documentos de uma determinada classe. Por exemplo, pode-se atribuir a classe de nome precatório a um determinado processo judicial, sendo ele um conjunto de documentos logicamente interligados para se comprovar um direito de receber valores, neste caso específico. Classificar um processo é uma atividade dispendiosa. Uma pessoa pode levar, em média, cerca de 20 (vinte) a 30 (trinta) minutos para classificar um processo judicial de natureza simples e conhecida.

Pensando nisso, a automatização das classificações em si já se mostra vantajosa, pois, explicada de um modo simples, retiraria as pessoas dessa atividade morosa e sem muitos requisitos intelectuais para alocar em outras atividades mais intelecto-produtivas. A classificação automatizada com o auxílio do computador pode ser mais rápida, de velocidade igual ou mais lenta que a realizada por pessoas. Mesmo no caso e que o computador classifique um processo na mesma velocidade que uma pessoa ou, pior, mais lento que uma

[1] http://www.pg.df.gov.br/inteligenciaartificial/ - acessado em 01/06/2021.

pessoa ter-se-ia a vantagem de utilizar aquela pessoa para atividades mais complexas que não poderiam ser automatizadas. Por outro lado, se o computador puder ser mais rápido que as pessoas na classificação as vantagens aumentam ainda mais.

Um exemplo de consequência interessante da aplicação da aprendizagem de máquina, que se mostra vantajoso para alguns especialistas de negócio da PGDF, seria a melhoraria dos controles quanto às ações judiciais repetitivas. Sendo elas, aquelas causas que são idênticas em conteúdo porém só mudam os nomes das partes. Reforçando a importância da aplicação da aprendizagem de máquina, as pessoas que estariam classificando esses processos da PGDF poderiam ser deslocadas para analisar e promover estratégias de controle mais eficientes para essas ações judiciais. Hoje, mesmo com o potencial analítico de sua força de trabalho especializada, a PGDF não tem condições de controlar a grande quantidade de processos contingenciados para análise.

Ou seja, os especialistas de negócio do órgão, observando as dificuldades em classificar e agrupar os processos apontam para as barreiras em se realizar investigações mais completas por parte dos analistas da PGDF, pois ficam alocados em tarefas repetitivas. É nesse sentido que o numeroso acúmulo de ações em curso tem alocado grande quantidade de recursos humanos e esses recursos têm sido insuficientes para essas tarefas repetitivas. Essa situação tem deixado os analistas da PGDF dedicados a essas atividades repetitivas e numerosas, que privam estes profissionais de realizar outras tarefas mais complexas e valorosas para o referido órgão.

Em outras palavras, esses especialistas de negócio e técnicos da PGDF entendem que uma ferramenta que classifique os processos, gere petições iniciais ou execute outras atividades repetitivas com uso da tecnologia de aprendizagem de máquina garantiria maior eficiência para a organização[1]. Desse modo, há um consenso entre as áreas técnica e de negócio sobre a importância de se desenvolver estudos e avanços dentro desse tema do aprendizado de máquina para melhor organização ou aceleração dos trabalhos.

Sendo assim, sem o correspondente incremento da força de trabalho e da estrutura operacional da PGDF, diante da volumosa atividade laboral repetitiva, surge a necessidade de ferramentas computacionais mais eficientes para auxiliar os servidores em suas tarefas analíticas. Por isso, este trabalho procura apresentar um mecanismo que classifique os documentos jurídicos com o auxílio do computador no contencioso, que é a área da PGDF que atua diretamente no judiciário no polo passivo e ativo.

1.1 Objetivo

O objetivo principal desta pesquisa de mestrado é comparar a eficiência das técnicas de aprendizado de máquina na classificação de documentos jurídicos na PGDF. Para os

objetivos secundários têm-se:

- Verificar a eficiência do cascateamento de modelos binários perante outros métodos de classificação multiclasse.
- Comparar as técnicas de redes neurais em *deep learnig* com outros classificadores.
- Verificar melhorias promovidas pelo uso das técnicas de amostragem em dados desbalanceados.
- Realizar análises descritivas com os dados encontrados de modo a entender os ganhos com a classificação automática.

1.1.1 Problema, pergunta e hipóteses de pesquisa

O problema desta pesquisa é a dificuldade de se classificar manualmente documentos jurídicos dentro da PGDF, que causa ineficiência do uso de recursos humanos. Além disso, dentre outros problemas secundários, causa a falta de dados estratégicos e consequentemente uma carência de alguns controles na PGDF.

Diante da problemática apresenta-se a pergunta de pesquisa principal: Qual das técnicas de aprendizagem de máquina na classificação de documentos jurídicos na PGDF tem melhor desempenho?

Algumas perguntas secundárias são:

- Os documentos jurídicos de processos repetitivos são melhor classificados automaticamente com o uso do cascateamento de classificadores binários ou com um classificador multiclasse?
- As redes neurais em *deep learnig* são superiores aos outros classificadores ?
- Seria possível quantificar a distribuição de documentos de ações repetitivas no contencioso utilizando o agrupamento de documentos jurídicos, por exemplo os de precatório por assunto principal? Quais classes são predominantes?
- Qual a técnica de amostragem se comporta melhor nas classificações automáticas ?

Algumas hipóteses de pesquisa foram inicialmente pensadas. Dentre elas, que a classificação dos documentos jurídicos repetitivos é mais eficiente utilizando algum documento específico do processo judicial, uma petição inicial ou uma sentença, ao invés de utilizar todos os documentos deste processo.

Outra hipótese seria de se utilizar algoritmos de classificação de textos em cascata para buscar resultados melhores ao invés de se utilizar apenas um estágio de classificação para múltiplas classes. Outra hipótese seria de que os classificadores que implementam redes

neurais em *deep learning* são superiores aos demais verificados no estudo. Comparando o desempenho do modelos via teste de hipótese estatístico.

1.2 Justificativa

Como dito, a PGDF não tem mecanismos suficientes para controlar com precisão todos seus processos devido ao alto volume desses processos e a falta de ferramentas, que a estimula a fomentar estudos e implementações tecnológicas para ajudar neste problema.

Segundo o Tribunal de Justiça do Distrito Federal e dos Territórios (TJDFT) [2], as cobranças administrativas e judiciais do Distrito Federal atualmente chegam a mais de R$ 30 (trinta) bilhões e a PGDF não tem conseguido dar a vazão pretendida a toda a demanda de processos judiciais que gostaria de processar. Outro dado interessante do TJDFT seria de que o Distrito Federal possui mais de 30 (trinta) mil precatórios a serem pagos[3]. Segundo o Conselho Nacional de Justiça (CNJ)[4], existe hoje uma alta taxa de congestionamento dos processos e com tempos de duração perante o Judiciário na média de 4 anos e 10 meses em 2019, o que demanda a alocação constante e de alta monta de servidores públicos para atuar em processos com baixa resolutividade e alto custo ao erário.

Dentre os trabalhos científicos tentando classificar documentos jurídicos no Brasil e no mundo geralmente as justificativas são similares, pois o volume de processos e o potencial analítico ficam descompassados. Dessas várias formas e métodos atualmente em estudo para classificar documentos jurídicos, destacam-se os trabalhos de Andrade [1] e Rocha [2], que estudaram métodos de classificação de textos com computação aplicada no Governo brasileiro.

Portanto, perante ao cenário de contingência de classificação de processos repetitivos, a vantagem de ter o auxílio do computador na tarefa e que o assunto é estudado no campo da computação aplicada dentro do Governo brasileiro, justifica-se que há convergência entre os objetivos deste trabalho científico e as incumbências institucionais do contencioso da PGDF em atuar de modo mais produtivo, analítico e econômico na representação judicial do Distrito Federal, pois visam, ambos, avanços no conhecimento que promova classificação automática de documentos jurídicos na PGDF.

[2] `https://www.tjdft.jus.br/institucional/imprensa/noticias/2020/novembro/nova-vara-de-execucao-fiscal-e-destaque-no-dftv`, acessado em Novembro de 2020

[3] `https://sapre.tjdft.jus.br/sapre/public/lista_externa.xhtml`, acessado em novembro de 2020

[4] `https://www.cnj.jus.br/wp-content/uploads/conteudo/arquivo/2019/08/8ee6903750bb4361b5d0d1932ec6632e.pdf`, acessado em Novembro de 2020

1.2.1 Organização deste trabalho

Os demais capítulos deste trabalho estão divididos em 7(sete) capítulos. O Capítulo 2 com a fundamentação teórica e contextual problemática com referências clássicas no tema e um aprofundamento sobre o contexto do problema. O Capítulo 3 com uma revisão de literatura para analisar o estado da arte no tema e apresentar estudos similares balizadores. O Capítulo 4 trata do desenvolvimento do estudo, com uma descrição da metodologia e processos de trabalho. O Capítulo 5 com os resultados e discussão diante dos dados de precatório. O Capítulo 6 com os resultados e discussão diante dos dados de ações repetitivas. O Capítulo 7 com as conclusões e trabalhos futuros.

Capítulo 2

Fundamentação

Neste capítulo estão definidos o que é aprendizagem de máquina e os algoritmos utilizados nos experimentos deste trabalho, iniciando pelos supervisionados para seguir apresentando brevemente os não supervisionados e por reforço. Discorre sobre métricas de avaliação de desempenho em algoritmos de aprendizagem de máquina e a comparação de desempenho entre modelos com estatística. O capítulo finaliza com as discussões sobre o processo de mineração de textos Cross Industry Standard Process for Data Mining (CRISP-DM) e traz a fase do CRISP-DM de entendimento do negócio.

Inicia-se a apresentação dos conhecimentos básicos necessários recordando de um fato histórico, principalmente com a intenção de motivar e aguçar o pensamento sobre a importância do acaso e os avanços científicos. O fato histórico que talvez tenha gerado o início dos estudos oficiais do fenômeno de uma possível máquina pensante. Assim como apresenta Dermot [3], em 1949, foi quando o noticiário *The Times* [1] publicou uma matéria que afirmava que a Universidade de Manchester havia desenvolvido um cérebro mecânico e envolveu o nome de Alan Turing, que em seguida em uma entrevista não refutou a ideia de que uma máquina teoricamente poderia pensar.

Segundo Dermot [3], esse evento histórico não intencional, notícia e entrevista, criou uma confusa impressão de se existir um cérebro artificial na época. O autor [3] apresenta que as consequências imediatas foram de grande discussão dentro da academia e na publicação do famoso artigo de Alan Turing para a revista Mind em 1950. Alan Turing [4], com termos menos matemáticos, discute e apresenta suas ideias sobre uma possível inteligência nas máquinas. Apresenta no artigo [4] o jogo de imitação, apelidado de teste de Turing, debatendo as possibilidades de uma máquina pensante, o que culminou no desenvolvimento de todo um campo científico.

Interessante notar que de uma situação fora do controle de Turing se credita a ele a luz dada ao tema Inteligência Artificial (IA). Apesar do primeiro trabalho reconhecido

[1] `https://www.thetimes.co.uk/`

no tema foi em 1943, desenvolvido por Warren McCulloch e Walter Pitts [5] [6]. Além desses nomes, John McCarthy também merece destaque, pois ao mudar-se para Stanford e depois para Dartmouth College promoveu a conferência histórica no Dartmouth College sobre inteligência artificial.

Muito tempo se passou após os acontecimentos que envolveram Warren McCulloch, Walter Pitts, Turing e McCarthy. O fenômeno do computador pensante ganhou corpo e vários campos de estudo foram criados dentro dessa chamada IA. Mas fora do campo de histórico, como afirmam Russel e Norving [6], a IA continua basicamente estudando os agentes inteligentes que recebem percepções do ambiente e executam ações cada vez mais precisas na medida do aprendizado.

Para Vargas et al. [**?**] a IA é um tipo de inteligência presente em máquinas que até então só se poderia perceber em seres humanos. Os referidos autores [**?**] ilustram um modelo que divide a inteligência artificial em 7 (sete) áreas, que se assemelham às capacidades necessária da IA para Russel e Norving [6]:

1. Processamento de Linguagem Natural (PLN)
2. Robótica
3. Modelagem cognitiva
4. Aprendizado de máquina
5. Sistema especialistas
6. Problemas resolvidos por heurísticas
7. Representação do conhecimento

Esses campos da inteligência artificial apresentados por Vargas et al. [**?**] podem se cruzar e serem estudados em conjunto. Como é o caso do aprendizado de máquina e PLN. Muitos dos algoritmos de aprendizado de máquina são usados atualmente no PLN.

Para Dhall et al. [**?**] os algoritmos de aprendizagem de máquinas surgiram de áreas como aprendizado computacional e reconhecimento de padrões. Para esses algoritmos tem-se basicamente um modelo matemático ou estatístico programado para processar entradas e retornar previsões ou descrição. Sendo que, se os dados são rotulados sugere-se utilizar algoritmos supervisionados e no caso contrário se pode utilizar algoritmos de aprendizagem de máquina não supervisionados.

Para Mitchel [7] o campo do aprendizado de máquina está preocupado em como construir programas de computador que melhoram com a experiência. Ou seja, um programa de computador é dito de aprendizagem de máquina quando por meio de uma experiência E em relação a alguma tarefa T um desempenho P na tarefa T melhora na medida que

se incrementa E. Desse modo, neste tipo de programa de computador o desempenho é diretamente proporcional a experiência.

Por sua vez, Bishop [8] também formaliza sua explicação sobre o aprendizado de máquina. Para ele o aprendizado de máquina pode ser expresso como uma função $y(x)$ que recebe os valores em x como entrada e gera uma saída y, codificada como um vetor-alvo t. A precisão da função $y(x)$ é determinada durante a fase de treinamento, também conhecida como aprendizagem. Assim, por exemplo, por meio de exemplos o algoritmo ajusta a função $y(x)$ para poder generalizar novas categorizações de x no espaço t. Vai de encontro com que Mitchel [7] ensina, que o aprendizado vai dos casos específicos no treinamento para o caso geral, daí onde surge o nome aprendizado indutivo.

Bishop [8] complementa que na prática os dados originais para entrada x devem ser pré-processados de modo ao algoritmo poder reconhecer os padrões com maior facilidade, etapa tipicamente chamada de extração de *features*.

$$TF(t,d) = \frac{\text{Total de palavras presentes}}{\text{Total de palavras do documento}} \tag{2.1}$$

$$IDF(t) = log\frac{\text{Total de documentos}}{\text{Frequencia do documento}} \tag{2.2}$$

$$\text{TF-IDF} = TF * IDF \tag{2.3}$$

Uma extração de *features* muito utilizada é a *Term Frequency-Inverse Document Frequency (TF-IDF)*, calculada pelas Equações 2.1, 2.2 e 2.3. Nessa extração de *feature*, a frequência do termo, Equação 2.1, significa a frequência bruta de um termo em um documento. Por sua vez, o termo relativo à frequência inversa do documento, Equação 2.2, é uma medida para determinar se o termo é comum ou raro em todos os documentos[9].

Portanto, o TF-IDF, Equação 2.3, para Christian et. al. [9] é um dado estatístico que reflete a importância da palavra em uma coleção de documentos - *corpus*. Ou seja, o valor TF-IDF aumenta proporcionalmente ao número de vezes que uma palavra aparece em um documento, mas é compensado pela frequência da palavra no corpus, o que ajuda a controlar o fato de que algumas palavras são mais comuns do que outras.

Ainda sobre o aprendizado de máquina, Russel e Norving [6] definem aprendizagem como uma busca através de um espaço de hipóteses. Remetendo as ideias de Mitchel [7] que afirma que o melhor aprendizado seria aquele cuja desempenho P na medida da experiência E alcança o objetivo do aprendizado, ou seja, encontra uma hipótese $h = y(x)$ ótima. Por exemplo, o desempenho P melhora na fase de treinamento, onde um conjunto de dados de treinamento com N exemplos é fornecido ao algoritmo e a rotina de treinamento aprimora os coeficientes ou pesos da função $y(x)$, chegando a uma aproximação de $h = y(x)$ a uma função verdade f, como afirma Bishop [8].

Portanto, pode-se entender o conceito de aprendizado como um problema de busca [7], pois ao executar o treinamento está ocorrendo uma busca diante de um grande espaço de hipóteses. Isso é feito no objetivo de se encontrar a hipótese que melhor se encaixe no problema em foco. Para isso, técnicas de diminuição do erro quadrático com gradiente descendente podem ser usadas para se certificar o avanço na medida do aprendizado no treinamento.

Sendo assim, para avaliar e escolher a melhor hipótese pode se realizar a validação dos modelos com um conjunto de dados de validação e outro conjunto de dados de teste. Obviamente que há caso que não se tem o vetor-alvo, situação que remete mais aos problemas de aprendizado não supervisionado. Muitas das vezes a avaliação do desempenho do algoritmo ocorre por um ser humano, mediante ao seu conhecimento de negócio.

Pode-se concluir que a ideia principal do aprendizado é aprender uma hipótese que melhor se ajuste aos dados futuros [6]. Para Russel e Norving [6] o melhor ajuste aos dados futuros está na proporção de erros que o algoritmo comete - a proporção de vezes que $h \neq y(x)$ para o exemplo (x, y). Ou sejam, para medir a precisão de uma hipótese é testado um conjunto de dados diferente daquele de treinamento onde se verifica o desempenho P com menor proporção de erros.

Nessa busca por essa hipótese ótima existe a preocupação dos dados estarem sobreajustados ao modelo - *overfitting*, quando o modelo está muito ajustado para um conjunto de dados de treinamentos e prejudica a generalização. Por isso, é comum uma fase de validação que antecede o teste dos erros. Por exemplo, a validação cruzada com k-repetições (*k-fold cross-validation*) é muito usada. Para Russel e Norving [6] a validação cruzada com k-repetições seria a repetição de treinamento e validação k vezes, onde de $1/k$ dos dados diferentes em cada rodada é usado para a validação. Desse modo, como afirma Bishop [8], a validação cruzada permite que todos os dados disponíveis sejam usados na medição da performance.

Contudo, o aprendizado de máquina é uma subárea da Inteligência Artificial que busca a melhor hipótese que se ajusta a dados futuros, que seja capaz de generalizar o aprendizado realizado na fase de treinamento. Para isso, o modelo proposto na fase de treinamento deve passar por validações e testes, de modo a minimizar a proporção de erros entre a resposta da hipótese proposta e o vetor-alvo. O aprendizado de máquina pode ser dividido em aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço.

2.1 Algoritmos de aprendizagem supervisionados

Seguindo na ideia de um treinamento para o aprendizado e teste está muito ligada ao aprendizado supervisionado. Russel e Norving [6] definem que dado um conjunto de treinamento de N pares de exemplos de entrada e saída

$$(x_1, y_1), (x_2, y_2), ...(x_n, y_n)$$

onde cada y_j foi gerado por uma função desconhecida $y = f(x)$, descobrir uma função h que se aproxime da função verdadeira f com um vetor y igual a um vetor-alvo t é o objetivo do aprendizado supervisionado. Como reforça Bishop [8], aplicações que os dados de treinamento são um vetor de entrada cuja previsões já tenham um vetor-alvo pré-estabelecidos e são problemas de aprendizado supervisionado.

Em outras palavras, os algoritmos de aprendizagem de máquina supervisionado necessitam de dados rotulados. Para Sen et al. [10] esses algoritmos podem ser divididos em dois grupos de problemas, os de regressão e os de classificação. Para este trabalho, um classificador utiliza os dados rotulados para realizar um treinamento do modelo que a partir daí pode apontar qual classe um determinado dado novo pertence.

Inicialmente apresentam-se os algoritmos *K-Nearest Neighbors (KNN)* e *Nearest centroid classifier*. Este tipo de algoritmo utiliza as medidas de similaridade dos vizinhos mais próximos. O resultado é obtido a partir de maioria simples do número k vizinhos mais próximos. Ou seja, existe uma probabilidade de um determinado ponto estar em uma vizinhança e o algoritmo decide aquela vizinhança mais provável convergindo em suas iterações[10].

2.1.1 *Random Forest*

Alguns algoritmos de classificação, por exemplo, têm em seu passo a passo funções lógicas que seguem fluxos distintos a partir de critérios pré-estabelecidos onde os dados vão sendo processados e classificados em uma estrutura de árvore. As árvores de decisão e suas variantes como o *Random Forest*, aqui em destaque, são um exemplo de algoritmo que se valem dessa estrutura em árvore para decidir em qual classe determinada observação se encontra [10].

Para Shah et. al. [11] *Random Forest* é um algoritmo que cria um grande número de árvores de decisão juntas para se fazer uma classificação. Para os autores essas árvores de decisão agem como pilares da classificação, cujos nós e suas quantidades são parte do pré-processamento do algoritmo. Os pontos de dados são as características ou *features* da informação a ser classificada. Esses pontos de dados são o que diferencia uma observação

da outra. Em uma representação tabular das observação temos as features, geralmente, representadas pelas colunas e as observações, informação sobre um determinado ente, como as linhas. Segue algoritmo:

Algorithm 1: Random Forest [11]

Result: Classificação por árvore de decisão

Etapa 1: nos dados de treinamento, escolha K pontos de dados *aleatórios*;

Etapa 2: construir uma árvore de decisão com esses K pontos de dados - *features*;

Etapa 3: Antes de repetir as etapas 1 e 2 e criar mais arvores aleatórios, escolha o número de nós - NTree - da próxima árvore que deseja construir;

Etapa 4: Repita 1, 2 e 3 até o número de árvores ser suficiente;

Etapa 5: Preveja o valor de y com cada uma das árvores e gere uma votação para cada previsão em foco;

2.1.2 SVM

Os algoritmos *Support Vector Machine (SVM)* são muito utilizados em programas de classificação. Pode-se entender sua representação como pontos nas dimensões do espaço para cada observação e os vetoras como fronteiras lineares para a separaão dos dados em classes. Baseado em segmentação por hiperplano os algoritmos SVM dividem os dados em classes específicas [10].

Segundo Lorena e Carvalho [12] os Suport Vector Machine são utilizados para a obtenção de fronteiras lineares para a separação de dados pertencentes a duas classes. Como pode ser visto na Figura 2.1 [13] os pontos x_1 e x_2 são os limiares entres as classes . A equação linear $wx + b = 0$ representa a linha que separa os dois hiperplanos H_1 e H_2.

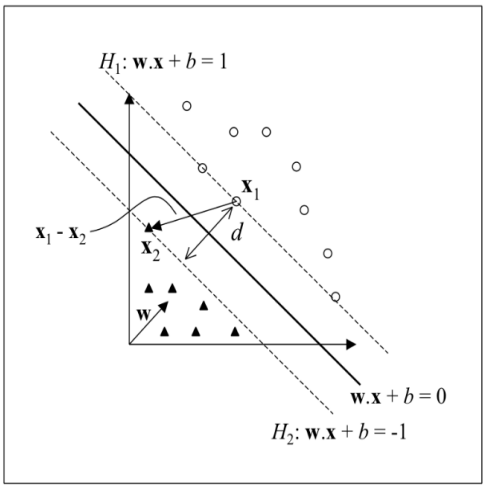

Figura 2.1: Hiperplanos no Suporte Vector Machine

2.1.3 Naive Bayes

Para Salsburg [14] o Teorema de Bayes, Equação 2.4, se define por uma probabilidade anterior implicando em uma probabilidade posterior a partir de seus dados.

$$p(A|B) = \frac{p(B|A)p(A)}{p(B)} \tag{2.4}$$

O algoritmo Naive Bayes (NB) exemplifica como o pensamento estatístico é poderoso para criar modelos de previsão. Neste algoritmo, probabilidades são dadas para cada caso específico e, então, baseado nas observações se pode concluir quais as maiores probabilidades para a classificação [10]. Estando com a ideia de um classificador ser uma função $y(x)$, onde dada uma observação x se atribui uma classe y, pode-se pensar o Naive Bayes como o algoritmo que dado um x procura-se a descoberta da função y para entregar um $p(y|x)$ máximo, como demonstra a Equação 2.5 [15].

$$y(x) = \underset{y}{\operatorname{argmax}}\, p(y|x) \tag{2.5}$$

O que diferencia o Naive Bayes de outras abordagens bayesianas é que para esse algoritmo todas as *features* são condicionalmente independentes entre si. Como demonstra a equação 2.6, onde D é o vetor de *features* e c é a classe atribuída a *feature* x_i da observação rotulada de x.

$$p(x|y = c) = \prod_{i=1}^{D} p(x_i|y = c) \tag{2.6}$$

2.1.4 *Logistic Regretion e Gradient Boosting*

Outro algoritmo utilizado é o *Logistic Regretion* - LR. O algoritmo *Logistic Regretion* usa uma função logística para fazer uma regressão. Assim, induzir a partir dos dados um modelo matemático. A função sigmoid ou curva S é associada para ajustar os modelos aos dados de treinamento. O modelo LR reconhece um vetor contendo variáveis rotuladas e avalia os coeficientes para cada variável de entrada, prevendo a classe de texto com o modelo gerado desse treinamento [11].

Segundo Natekin e Knoll [16] o *Gradient Boosting* - GB - é a união das duas técnicas de boosting e *gradiente descente*. Para os autores o boosting é uma técnica de *ensemble learning* que agrega modelos sequencialmente para melhorar desempenho reajustando pesos para dados de modo a reduzir o viés do algoritmo. Por sua vez, a técnica de gradiente descendente é uma técnica que auxilia na escolha dos parâmetros do modelo por meio de uma minimização da função de perda - *loss funtion*. Sendo a função perda a função que calcula o erro no treinamento, por exemplo a diferença quadrática entre o estimado e o

valor esperado. Seguem algoritmo de Friedman [17].

Algorithm 2: Algoritmo Gradient Boosting de Friedman

Entre os dados (x,y);
Escolha o número de interações M;
Escolha a função perda (*loss function*);
Escolha o algoritmo de aprendizagem;
for *t =1 até M* **do**
    calcule o gradiente descendente;
    ajuste a nova hipótese para o aprendizado;
    encontre o melhor gradiente descendente;
    atualize a função e estimação
end

Esta subseção tratou LR juntamente GB devido às implementações de GB. Neste trabalho, se utiliza LR para a função de perda - *loss function*.

2.1.5 Redes Neurais Artificiais

Outro grupo de algoritmos de aprendizagem de máquina são as redes neurais artificiais, doravante chamadas apenas de redes neurais. Aquelas que tentam imitar o comportamento de uma rede de neurônio naturais, por exemplo as redes de perceptrons. Essa abordagem de redes neurais tenta imitar o cérebro humano com grafos, mais especificamente os neurônios artificiais em rede, com uma função de ativação e suas interconexões por dendritos. O estado da arte das redes neurais são as redes de aprendizagem profunda (*deep learning*) [**?**].

Para Goodfellow et. al.[18] o *deep learning* permite que os computadores aprendam com a experiência e entendam o mundo em termos de uma hierarquia de conceitos, com cada conceito definido por meio de sua relação com conceitos mais simples. Reunindo conhecimento com a experiência, essa abordagem evita a necessidade de operadores humanos especificarem formalmente todo o conhecimento de que o computador precisa. Se desenharmos um grafo mostrando como esses conceitos são construídos em cima uns dos outros, um grafo profundo, com muitas camadas temos o aprendizado profundo. Por esse motivo, se chama essa abordagem de aprendizado profundo - *deep learning*.

A Figura 2.2 [18] ilustra um modelo de aprendizagem profunda. Entender o que é cada pixel em uma entrada de dados é complexo para o computador . O aprendizado profundo resolve essa dificuldade dividindo o mapeamento complicado desejado em uma série de mapeamentos simples aninhados, cada um descrito por uma camada diferente do modelo. A entrada é apresentada na primeira camada e, em seguida, uma série de

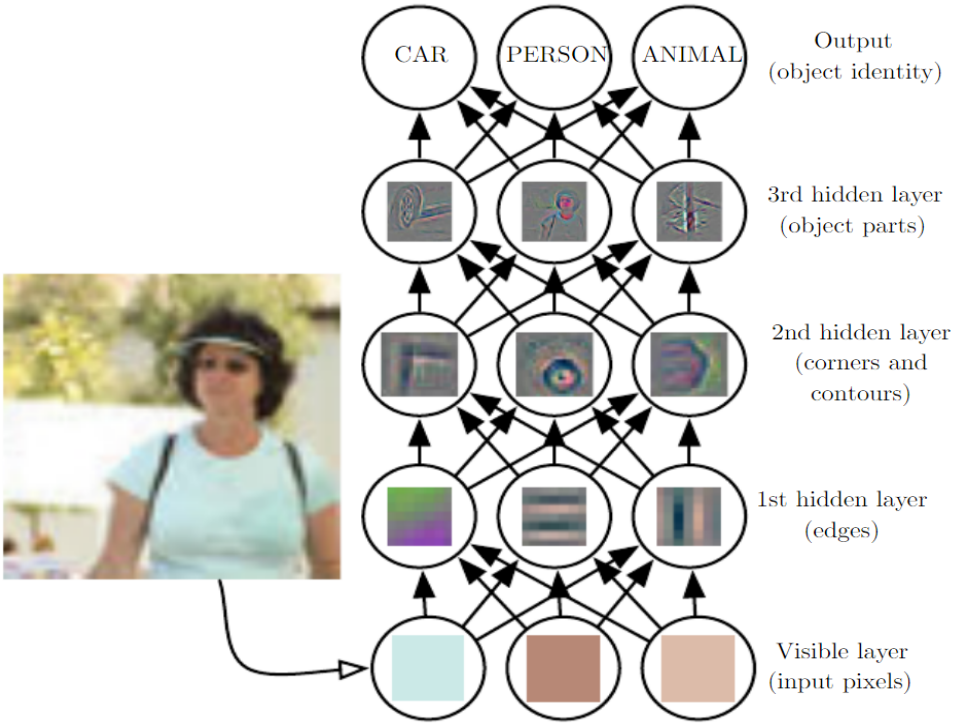

Figura 2.2: Representação do *deep learning*

camadas ocultas extrai recursos cada vez mais abstratos da imagem. Essas camadas são chamadas de "ocultas" porque seus valores não são fornecidos nos dados; em vez disso, o modelo determina, por meio das *features* na fase de treinamento, quais recursos são úteis para explicar as relações nos dados observados [18].

Bansal et al. [19] apresentam que o *deep learning* está penetrando em todos os domínios possíveis do aprendizado de máquina e o domínio jurídico também está recebendo os benefícios desse tipo de técnica. Segundo os autores, um advogado precisa passar horas e horas em busca de material relevante e preparação de argumentos com relevantes precedentes. Com essas técnicas o advogado humano pode trabalhar com mais rapidez e produzir muito mais dados em sua atividade fim.

Portanto, quando se tem uma rede com muitas camadas ocultas e cada uma dessas camadas tem uma função dentro do aprendizado, simplificando o problema, tem-se o aprendizado profundo. Na próxima seção se têm os algoritmos que não necessitam de rótulos para o aprendizado, são os algoritmos de aprendizagem não supervisionados.

2.2 Algoritmos de aprendizagem não supervisionados

No caso dos algoritmos não supervisionados, o objetivo do aprendizado é descobrir grupos onde dados similares se concentram, também chamados de *clustering* [8]. Russel e Norving [6] definem o aprendizado não supervisionado como aquele que o agente aprende apenas com os padrões da entrega fornecida, pois embora não é fornecido nenhum vetor alvo para comparação no treinamento.

Um dos algoritmos de aprendizagem não supervisionados utilizados neste trabalho foi o Latent Dirichlet Allocation (LDA). Para Blei et al. [20] o LDA é um modelo probabilístico não supervisionado que trabalha em coleções de dados discretos, como *corpus* de texto. O LDA é um modelo bayesiano no qual cada item de uma coleção, cada documento de um *corpus*, é modelado como uma mistura finita sobre um conjunto subjacente de tópicos. Cada tópico é, por sua vez, modelado como uma mistura infinita sobre um conjunto subjacente de probabilidades de tópico. Ou seja, a distribuição das probabilidades das palavras por tópico correspondem ao que o define. No contexto da modelagem de texto, as probabilidades do tópico fornecem uma representação explícita de um documento e os tópicos dominantes podem ser interpretados como grupos de documentos.

O agrupamento com o uso do *K-means*, que é outro algoritmo de aprendizagem de máquina não supervisionado, cria grupos automaticamente quando os dados possuem características semelhantes. O algoritmo é denominado *K-means* porque cria K grupos distintos. [**?**]. Para Ghosal et al. [21] o algoritmo *K-means* procura organizar os grupos de dados baseado em *centroids*, ou seja pontos centrais que as determinadas características semelhantes convergem.

2.3 Aprendizagem por reforço

O aprendizado por reforço é aquele que em geral não necessita de exemplos para sua consecução. O agente que aprende por reforço recebe um estímulo cada vez que consegue o objetivo do aprendizado. Geralmente, a experiência, no aprendizado por reforço, vem da interação com o ambiente.

Para Bishop [8] a aprendizagem por reforço está restrita aos problemas onde as ações a serem tomadas em determinada situação por um agente são recompensadas na medida de seu desempenho. Para isso, não são fornecidos dados exemplos para a fase de aprendizado. No treinamento por reforço, em contraste com o aprendizado supervisionado, o agente descobre a melhor hipótese por meio de um processo de tentativa e erro ao interagir com o ambiente.

Russel e Norving [6] definem a tarefa da aprendizagem por reforço na medida que se usam recompensas observadas para aprender uma política ótima (ou quase ótima) para o ambiente. Neste caso não se supõe nenhum conhecimento anterior do modelo ou função de recompensa. A ideia é que se conheça o estado final ao qual o agente percebe que está agindo de maneira otimizada. Por exemplo, quando em um jogo de xadrez o agente percebe que se chegou a um xeque-mate, muita das vezes recebendo um estímulo do ambiente ou de outros agentes.

Portanto, o aprendizado por reforço está muito ligado a afirmação de Russel e Norving, que a Inteligência Artificial continua sendo basicamente a problemática dos agentes inteligentes que recebem percepções do ambiente e executam ações cada vez mais precisas na medida do aprendizado. Pois, por meio da recompensa ou reforço o agente vai aprimorando seu aprendizado e generalizado suas ações de um modo cada vez mais preciso.

2.4 Métricas de desempenho

A aplicação pura dos algoritmos de aprendizagem de máquina carece de algumas verificações, pois o comportamento adequado e os resultados confiáveis do algoritmo necessitam de métricas para medir seu desempenho. Esta seção vai apresentar as métricas que foram utilizadas nos experimentos deste trabalho, tanto nos algoritmos de aprendizagem de máquina supervisionados como para os não supervisionados.

Existem muitas métricas de desempenho para os algoritmos de aprendizagem de máquina e não faz parte deste trabalho e nem se mostrou viável criar uma lista extensiva e exaustiva. A seguir apresenta-se algumas métricas utilizadas no trabalho. São elas: *coherence* com *Pointwise Mutual Information (PMI)*, precisão, revocação, F_1-*score*, acurácia e matriz de confusão.

2.4.1 Métricas de desempenho para algoritmos supervisionados

Quando se fala na métrica precisão *(precision)* se trata da razão entre a quantidade de reais acertos, os verdadeiros positivos, *True Positive - TP*, diante da soma dos verdadeiros positivos mais os falsos positivos, *False Positive - FP*. Ou seja, a métrica de precisão apresenta a predição de acertos verdadeiros realizados diante do total de acertos do modelo, Equação 2.7 [22, 23].

$$\text{Precisão} = \frac{TP}{TP + FP} \tag{2.7}$$

Por sua vez, a métrica revocação *(recall)* se trata da razão entre a quantidade de reais acertos, os verdadeiros positivos, *True Positive - TP*, diante da soma do verdadeiros

positivos mais os falsos negativos, *Fale Negative - FN*. Ou seja, a métrica de revocação apresenta a proporção de acertos verdadeiros realizados diante do que ele teoricamente deveria ter acertado, Equação 2.8 [22, 23].

$$\text{Revocação} = \frac{TP}{TP + FN} \tag{2.8}$$

Evoluindo, tem-se F_1-*score* que é a média harmônica utilizando precisão e revocação. Pode-se perceber que F_1-*score* se concentra mais na classe positiva e, portanto, as características negativas são desvalorizadas em comparação com características positivas [24]. Importante mencionar que o maior valor possível de uma pontuação F_1-*score* é 1, indicando precisão e recuperação perfeitas, e o menor valor possível é 0. A Equação 2.9 é apresentada para formalizar a medida.

$$F_1\text{-Score} = \frac{2 \times precisão \times revocação}{precisão + revocação} \tag{2.9}$$

A acurácia pode ser considerada a mais simples de todas essas métricas apresentadas. A Equação 2.10 formaliza o cálculo simples da acurácia de um modelo [24]. Também pode-se encontrar variações de cálculos de acurácia, que já contemplam características negativas, como por exemplo em [25], formalizado na Equação 2.11.

$$ACC = TP - FP \tag{2.10}$$

$$ACC = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.11}$$

A matriz de confusão é uma representação gráfica das possibilidades de classificação em formato de tabela. Com a matriz de confusão é possível visualizar o desempenho dos modelos e algoritmos implementados. Pode-se derivar dela as métricas citadas anteriormente [26].

A Tabela 2.1 ilustra como pode ser organizada uma matriz de confusão, com basicamente quatro possibilidades: *True Negativa - TN*, *True Positive - TP*, *False Negative - FN* e *False Positive - FP*. Respectivamente, os valores verdadeiro e falso negativo são os erros do modelo e os valores falso e verdadeiro positivo são os reais acertos, ou seja se esses últimos são parte ou não de uma classe, por exemplo. Adiante, no decorrer deste trabalho, matrizes de confusão bem maiores serão apresentadas e poderão ser analisadas.

Tabela 2.1: Matriz de confusão exemplo

confusion matrix:		
Classe em análise	Classificação	
	Positivo	Negativo
Positivo	TP	FP
Negativo	FN	TN

Portanto, a pura e simples aplicação dos algoritmos de aprendizagem de máquina pode induzir a interpretação de resultados errados. As métricas de desempenho auxiliam a verificar se o modelo criado na execução do algoritmo foi adequado para a classificação. Importante registrar que o modelo é a união do algoritmo, seus parâmetros proveniente dos dados usados no treinamento [27].

Ter em mãos as métricas de desempenho não retiram a importância de se observar visualizações e comparar resultados por meio de modelos gráficos, que também são boas referências de desempenho. Nem mesmo retira a importância de se utilizar testes de hipótese estatísticos para comparar classificadores, pelo contrário auxilia e reforça.

2.4.2 Métricas de desempenho para algoritmos não supervisionados

A métrica *coherence*, muito usada em análise de tópicos, é um achado que demonstra o quão bons são os tópicos e o quanto as características dessas coleções de documentos se tornam mais acessíveis à modelagem de tópicos. Além disso, demonstra o potencial da modelagem de tópicos para o entendimento humano [28].

Newman [28] apresenta várias formas de se calcular a métrica *coherence*, destaca, por terem melhor desempenho em seu estudo, *Term Co-occurrence* e *PMI*, que é uma medida da independência estatística para observar a proximidade do score de duas palavras. Para formalizar, segue a Equação 2.12 da métrica *coherence* com PMI, onde tem-se o logaritmo da razão entre a probabilidade conjunta p() de w_i e w_j com as probabilidades independentes p() de w_i e w_j.

$$PMI(w_i, w_j) = log \frac{p(w_i, w_j)}{p(w_i)p(w_j)} \tag{2.12}$$

2.5 Comparação de desempenho de algoritmo com o uso de estatística

Supostamente um algoritmo que tem as melhores métricas de desempenho pode ser considerado melhor que outro. No entanto, não funciona bem assim. Tendo dois classificadores treinados nos mesmos dados e o teste dando precisão de previsão de 98% e 96% para um e para outro. Assim, pode-se afirmar que o primeiro classificador é significativamente melhor do que o segundo? A resposta é não. Para esse tipo de afirmação por exemplo, pode-se utilizar o teste de McNemar [29]. Este teste vai se valer de um teste de hipótese, onde a hipótese nula é de que os modelos são significativamente similares, utilizando a estatística de chi-quadrado:

$$\chi^2 = \frac{(|b-c|-1)^2}{(b+c)} \tag{2.13}$$

Onde **b** é igual a quantidade daqueles testes que foram previstos corretamente para o modelo 1 e incorretamente para o modelo 2 e **c** é exatamente o inverso, para aqueles testes que foram previstos corretamente para o modelo 2 e incorretamente para o modelo 1, é necessário realizar uma correção pois está se usando uma distribuição contínua para aproximar uma função discreta.

Sobretudo, como afirma Bagui [30], não existe um melhor classificador. O autor defende que existe o melhor classificador para cada problema e que cada algoritmo pode ter um desempenho diferente dependendo do problema, consequentemente dos dados. O autor continua afirmando que geralmente os estudos comparativos levam em consideração base de dados de problemas reais e que quatro fatores de variação são considerados importantes nessas comparações: i) a escolha dos dados de treino; ii) a escolha dos dados de teste; iii) a aleatoriedade dos dados de treino e iv) a aleatoriedade dos erros de classificação.

Segundo Dietterich [31], a escolha do conjunto de treinamento é determinante para alguns modelos de classificadores quanto a sua instabilidade, porque pequenas mudanças no conjunto de treinamento podem causar mudanças substanciais no classificador. Essa instabilidade nos classificadores promove uma adaptação do modelo aos exemplos de treinamento para que sejam corretamente classificados. A instabilidade de tais classificadores é de certa forma a recompensa por essa versatilidade.

Ao se ter diferentes conjuntos de teste se pode criar modelos de forma diferente de modo a se ter a mesma precisão em toda a população [31]. Ou seja, é perigoso tirar conclusões de um único experimento de teste, especialmente quando o tamanho da base de dados é pequeno. Sendo assim a escolha dos dados de treinamento, validação e teste é fundamental. Importante recordar da validação cruzada, que aproveita todo o conjunto de dados para essas fases de aprendizado.

Além disso, a preocupação com a aleatoriedade do algoritmo de treinamento é válida pois, pode ser na inicialização dos parâmetros do classificador que podem estar ajustes importantes [31]. Desse modo, é preciso se preocupar com classificadores treinados que podem ter diferentes resultados para o mesmo conjunto de treinamento e até mesmo para a mesma inicialização dos parâmetros. Todo o conjunto de dados e parâmetros deve ser considerado ao se pensar em determinados algoritmos com alta aleatoriedade interna.

A aleatoriedade dos erros de classificação considera a possibilidade de se ter objetos com rótulos incorretos nos dados de previstos[31]. Aqui, principalmente, se observam os erros do tipo 1(um), ou também conhecidos como falsos positivos. Por exemplo, o algoritmo classifica um exame para uma pessoa que tem câncer como se a pessoa estivesse

normal, sem a doença. Os erros do tipo 1(um) são mais preocupantes que os erros do tipo 2(dois), falsos negativos pois, no exemplo anterior se uma pessoa não tem câncer e é direcionada para uma fila de tratamento, muito provavelmente o dano seria menor que alguém já doente ser direcionada para ir para casa.

Segundo Dietterich [31], ainda existem problemas a serem solucionados quando se usa apenas o teste de McNemar pois, remanescem problemas quanto à escolha dos dados de treino, a escolha dos dados de validação e a aleatoriedade dos dados de treino. O autor sugere um procedimento de teste que consiste em repetir um procedimento de validação cruzada dupla cinco vezes(5 X 2-Fold-cv) para minimizar esses problemas. Com esse teste de 5 X 2-Fold-cv os problemas da escolha dos dados de treino, a escolha dos dados de validação e a aleatoriedade dos dados de treino são minimizados, pois se aumenta a variabilidade nas fases de treinamento e validação, tornando a comparação mais verossímil porque aumenta a aleatoriedade das amostras.

Em cada uma das 5 iterações, ajusta-se A e B à divisão de treinamento e avalia-se seu desempenho (p_A e p_B) na divisão de teste. Em seguida, um *shift* é feito nos conjuntos de treinamento e teste (o conjunto de treinamento torna-se o conjunto de teste e vice-versa) calculando-se o desempenho novamente, o que resulta em 2(duas) medidas de diferença de desempenho[31]:

$$p^{(1)} = p_A^{(1)} - p_B^{(1)} \tag{2.14}$$

$$p^{(2)} = p_A^{(2)} - p_B^{(2)} \tag{2.15}$$

Em seguida, estima-se a média e a variância das diferenças:

$$\overline{p} = \frac{p^{(1)} + p^{(2)}}{2} \tag{2.16}$$

$$s^2 = (p^{(1)} - \overline{p})^2 + (p^{(2)} - \overline{p})^2. \tag{2.17}$$

A variância da diferença é calculada para as 5 iterações e então se faz um teste estatístico t de *student*:

$$t = \frac{p_1^{(1)}}{\sqrt{(1/5)\sum_{i=1}^{5} s_i^2}}, \tag{2.18}$$

Onde $p^{(1)}$ é o p_1 da primeira iteração. A estatística t de *student*, supondo que segue aproximadamente a distribuição t com 5 graus de liberdade, está sob a hipótese nula de que os modelos A e B têm desempenho igual. Usando a estatística t, o p-valor pode ser

calculado e comparado com um nível de significância previamente escolhido, por exemplo, $\alpha = 0,05$. Se o p-valor for menor que α , rejeita-se a hipótese nula e aceita-se que existe uma diferença significativa nos dois modelos.

Os testes mencionados anteriormente são para comparação de classificadores em pares, testes não paramétricos. Segundo Bagui [30] para comparar $L > 2$ classificadores nos mesmos dados de teste, o teste Q de Cochran pode ser usado. O teste Q de Cochran é proposto para medir se existem diferenças significativas nos L classificadores. O teste Q de Cochran pode ser considerado uma versão generalizada do teste de McNemar que pode ser aplicado para avaliar vários classificadores. Deve-se testar a hipótese para nenhuma diferença entre as precisões de classificação (proporções iguais):

$$p_i : H_0 = p_1 = p_2 = \cdots = p_L.$$

Seja $D_1, \ldots, D_L$ um conjunto de classificadores que foram testados no mesmo conjunto de dados. Se os classificadores L não tiverem um desempenho diferente, a seguinte estatística Q é distribuída aproximadamente como "chi-quadrado" com $L - 1$ graus de liberdade:

$$Q_C = (L-1)\frac{L\sum_{i=1}^{L} G_i^2 - T^2}{LT - \sum_{j=1}^{N_{ts}} (L_j)^2}. \tag{2.19}$$

Onde, G_i é o número de objetos de N_{ts} corretamente classificados por $D_i = 1, \ldots L$; L_j é o número de classificadores de L que classificou corretamente o objeto $\mathbf{Z}_{ts} = \{\mathbf{z}_1, ...\mathbf{z}_{N_{ts}}\}$, onde $\mathbf{Z}_{ts} = \{\mathbf{z}_1, ...\mathbf{z}_{N_{ts}}\}$ é o conjunto de dados de teste no qual os classificadores são testados; e T é o número total de votos corretos entre os L classificadores [30]:

$$T = \sum_{i=1}^{L} G_i = \sum_{j=1}^{N_{ts}} L_j. \tag{2.20}$$

Portanto, não existe um melhor classificador, o que existe seria o melhor classificador para cada problema. Cada algoritmo terá um desempenho diferenciado dependendo do problema e de seus dados. É preciso ter em mente que existem quatro fatores de variações a serem considerados importantes nessas comparações: i) a escolha dos dados de treino; ii) a escolha dos dados de teste; iii) a inter aleatoriedade dos dados de treino e iv) a aleatoriedade dos erros de classificação. Desse modo, se faz necessário alguns testes estatísticos para comparar desempenho de algoritmos de classificação com aprendizagem de máquina, principalmente comparando o desempenho referente aos erros do tipo 1.

2.6 A Mineração de Texto

Outra fundamentação importante a ser definida aqui seria a mineração de textos. Pois, pode-se entender que a realização de classificações e agrupamento com o auxílio do computador é muito mais que a pura aplicação dos algoritmos em dados brutos e seus testes de desempenho e comparação.

Existe uma necessidade de aplicação de um processo que parte do entendimento do negócio para posterior entendimento dos dados, que certamente estão intrinsecamente associados. Em seguida, avança para uma etapa de modelagem, que muitas vezes passa por experimentações e seleção de algoritmos que melhor se comportam com aquele fenômeno a ser modelado. Essa seleção é validada em uma etapa chamada avaliação, que testa a performance e efetividade do modelo. Por fim, tendo um bom resultado na avaliação o método pode ser implementado e disponibilizado para o usuário.

Esse processo pode ser entendido como um processo de mineração de textos. Rocha [2], por sua vez, aplica em seu trabalho a mineração de textos com o modelo de referência Cross Industry Standard Process for Data Mining (CRISP-DM).

Pode-se observar no trabalho de Andrade [1] que a mineração de texto extrapola o processamento da linguagem natural e se vale de métodos analíticos para descobrir padrões e conhecimento em diferentes indústrias. Ela define a mineração de textos como um processo de descoberta de conhecimento que utiliza técnicas de análise e extração de dados a partir de textos, frases ou palavras. Esse processo tem o objetivo de extrair padrões não triviais ou conhecimento a partir de documentos em textos não estruturados.

No capítulo de Desenvolvimento o CRISP-DM será detalhado como um processo cíclico e incremental. Destaca-se o entendimento do negócio, que se encontra a seguir, para um primeiro entendimento do contexto do problema de pesquisa. Vai se utilizar este processo para a mineração de texto no decorrer deste trabalho de um modo cíclico e incremental. Sobretudo, resumidamente, pode-se entender o CRISP-DM em 6 (seis) fases, que ocorre em ciclos iterativos:

1. Iniciando pelo entendimento do negócio;
2. Seguindo para o entendimento dos dados;
3. Preparando os dados para processamento;
4. Modelagem computacional;
5. Avaliação dos resultados;
6. Implementação.

2.7 Entendimento do Negócio

Nesta seção será apresentado como a PGDF é organizada e onde, dentro dela, esta pesquisa está inserida. Será definido o que é o precatório e como ele é tratado dentro da PGDF. A fase de entendimento de negócio não necessariamente é feita uma só vez. Ela depende da avaliação dos resultados, se forem insatisfatórios pode ser necessário maior entendimento do negócio e dos dados. Sendo assim, aqui se tem o entendimento inicial do negócio. Mais adiante na fase de resultados, se terá outros ciclos de entendimento de negócio e dos dados para a construção do modelo.

Portanto, aqui tem-se um entendimento inicial da unidade de análise da pesquisa de onde as informações foram extraídas e será apresentada os sistemas de informática, que são as fontes de informação da pesquisa. O fenômeno estudado é o da classificação de textos jurídicos manualmente.

Entrevistas e análises documentais foram realizadas para o entendimento do negócio e do contexto da unidade de análise. A observação não participante foi utilizada para melhor entender o fenômeno em estudo . As entrevistas desestruturadas foram feitas na medida que se foi identificada alguma possibilidade de contribuição por parte de algum dos membros da organização, especialistas, e seguiu-se o método expiratório de Yin [32].

A Procuradoria Geral do Distrito Federal (PGDF) é, nos termos do art. 110º da Lei Orgânica Distrital [2], o órgão central do sistema jurídico do Distrito Federal. Exerce, privativamente, a competência de orientação jurídica da Administração direta e indireta, além de representar o ente político judicial e extrajudicialmente, entre outras competências descritas no art. 4º da Lei Complementar nº 395, de 31 de julho de 2001.

O Decreto nº 36.236 de 1º de janeiro de 2015, estabelece que a PGDF é órgão especializado da Administração Direta do Governo do Distrito Federal (GDF). Nessa qualidade, esta Casa de Advocacia Pública deve estar alinhada aos interesses do Estado, especialmente no tocante à defesa do GDF, em ambiente tecnológico compatível com a eficiência pretendida pelo Estado e pela sociedade e exigida pelos órgãos do Poder Judiciário nos níveis federal e distrital.

A Figura 2.3 apresenta o organograma da PGDF. Atualmente a Procuradoria Geral do Distrito Federal (PGDF) está organizada em quatro grandes áreas de negócio: i) a Procuradoria Geral do Contencioso (PGCONT); ii) a Procuradoria Geral da Fazenda Distrital (PGFAZ); iii) a Secretaria Geral (SEGER) e iv) a Procuradoria Geral do Consultivo (PGCONS), como pode ser visto na Figura 2.3. Além dessas áreas existem as unidades superiores: o Gabinete da Procuradora Geral, a Corregedoria, o Conselho Superior da PGDF e o Conselho do Fundo Pró-Jurídico.

[2] `http://www.fazenda.df.gov.br/aplicacoes/legislacao/legislacao/TelaSaidaDocumento.cfm?txtNumero=0&txtAno=0&txtTipo=290&txtParte=.`

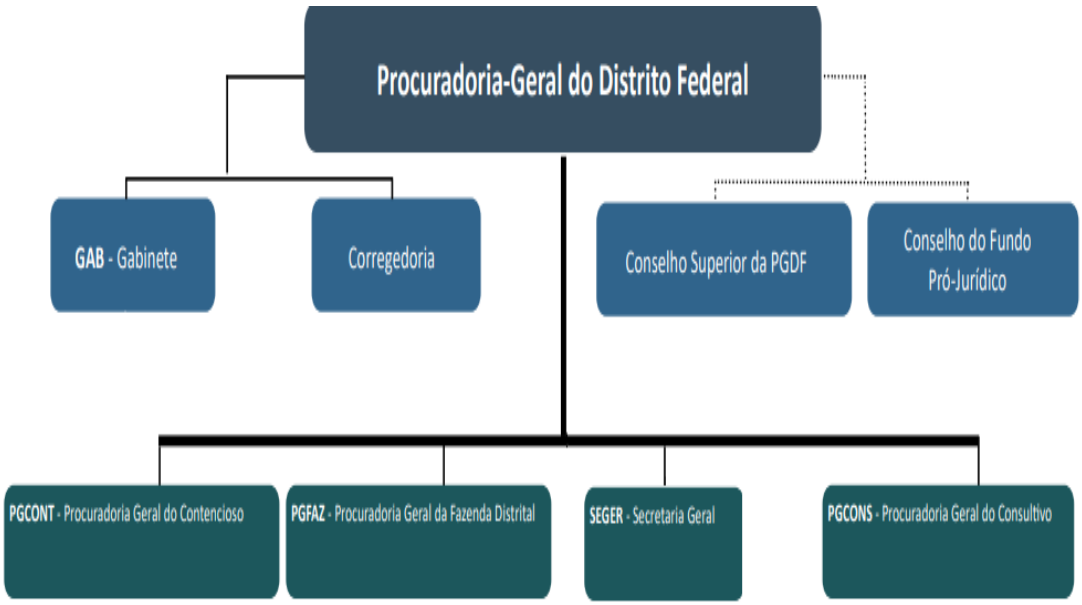

Figura 2.3: Organograma da PGDF

A PGFAZ além de diversas obrigações que constam no Regimento Interno da PGDF – Decreto nº 22.789/2002 é a área responsável pelas cobranças judiciais do Distrito Federal e algumas cobranças Administrativas. Além disso, ela é a responsável por receber as sentenças judiciais para o pagamento de precatórios.

Porém, é na SEGER, mais especificamente na Diretoria de Protocolo Judicial (DIPROJ) que os precatórios são analisados e tratados. A PGDF atualmente despende alta monta com a alocação de recursos humanos lendo processos para classificá-los de acordo com a tabelas processuais unificadas, de acordo com a Resolução Nº 46 de 18 de dezembro de 2007 do Conselho Nacional de Justiça (CNJ). Como já foi dito, a classificação com o auxílio do computador muito provavelmente trará economias ao Estado.

2.7.1 Precatórios

O precatório, um dos temas centrais deste estudo, é uma espécie de requisição de pagamento de determinado valor que a Fazenda Distrital deve pagar, após transitado em julgado, com valores acima de 60 salários mínimos por beneficiário[3]. Essa Requisição de Pagamento é realizada por meio de uma sentença judicial. O Juiz emite uma sentença declarando o dever de pagar para o Estado, gerando assim um precatório. Além do precatório, existe outro meio de requisição de pagamento que seria a Requisição de Pequeno

[3] `https://www.tjdft.jus.br/consultas/precatorios/perguntas-frequentes`, acessado em novembro de 2020

Valor (RPV). Como o nome já indica, a RPV é utilizada quando o valor é menor que o estipulado para o pagamento em precatório [4].

O processo de classificação de precatório se dá, sumariamente, após o processo tramitar em julgado. Após o Juiz emitir uma sentença judicial uma intimação é captada pelo sistema de informação da PGDF via Modelo Nacional de Interoperabilidade (MNI). Em seguida, um processo de triagem manual é feito na DIPROJ, onde um indivíduo é responsável por identificar dentro outros tipos de processo se aquela intimação sentenciada é do tipo precatório. O processo então é encaminhado para a DIPROJ e lá se dá uma nova triagem manual e análise processual, que dentre outros tipos de processos está o de precatório. Esse processo então é encaminhado para uma fila de pagamento.

2.7.2 Sistemas

O sistema que capta as intimações, onde estão todos os processos repetitivos, inclusive de precatório, é o sistema de gestão de documentos eletrônicos judiciais da PGDF. Parte de sua estrutura de dados e como ele interage com o Processo Judicial Eletrônico (PJe)[5], permite a integração ou interoperabilidade com o TJDFT, por meio do Modelo Nacional de Interoperabilidade (MNI). Será brevemente apresentado o sistema de documentos judiciais do TJDFT, denominado PJe.

Atualmente, a PGDF conta com um sistema próprio de gestão de documentos eletrônicos judiciais. O sistema é um software desenvolvido na arquitetura cliente-servidor na linguagem Delphi [6]. Esse sistema foi implantando desde 2015 e sua base de dados, que está com aproximadamente 5 (cinco) Terabytes de dados armazenados, foi implantada no Sistema de Gerenciamento de Banco de Dados (SGBD) SQL Server [7]. A Figura 2.4 exemplifica como é a tela do sistema. Neste sistema existe a possibilidade de se consultar o processo judicial completo por meio da integração com o MNI. Isso significa que esta integração permite que se faça, teoricamente, todo o trâmite processual e consultar ao PJe por meio do sistema da PGDF.

O Modelo Nacional de Interoperabilidade (MNI) foi criado por meio do termo de cooperação técnica n. 58/2009 e visa estabelecer os padrões para intercâmbio de informações de processos judiciais. Ele é implantado na arquitetura de Web Service com dinâmica de comunicações que deve seguir pelo envio de remessas processuais e a sua respectiva baixa, contemplando os fluxos judiciais principais e alternativos.

[4] `https://www.tjdft.jus.br/consultas/precatorios/perguntas-frequentes`, acessado em novembro de 2020

[5] `http://www.pje.jus.br/wiki/index.php`

[6] `https://www.embarcadero.com/products/delphi?aldSet=en-GB`

[7] `https://www.microsoft.com/pt-br/sql-server/`

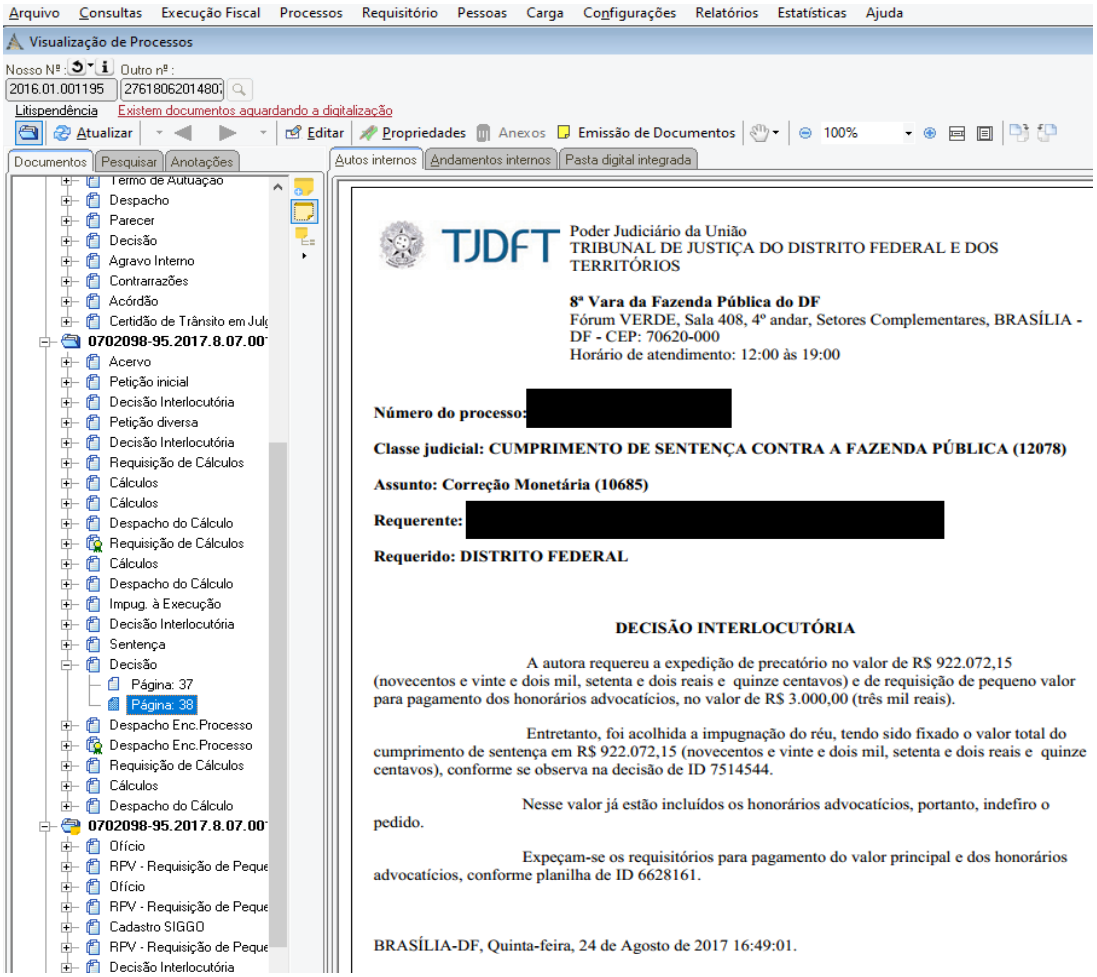

Figura 2.4: Tela do sistema com processo de precatório

A dinâmica do MNI tem três etapa: i) remeter processo, ii) consultar processo e iii) confirmar o recebimento do processo na instância superior. Como não se trata de um sistema com uma Graphical User Interface (GUI) é preciso utilizar um programa como o SOAPUI [8] para ilustrar como é o serviço. A Figura 2.5 apresenta a tela do sistema SOAPUI acessando o MNI. Portanto, utilizando o MNI pode-se consultar e remeter processos.

Voltando ao sistema de gestão de documentos eletrônicos judiciais da PGDF, é importante observar a base de dados de onde os dados foram coletados. O modelo de dados em si é muito extenso e não seria produtivo discutir e apresentá-lo por completo aqui. Por isso, a Figura 2.6 apresenta apenas as tabelas que armazenam os documentos a serem coletados. São tabelas que armazenam os documentos digitalizados no formato *Binary Large OBject (BLOB)*. Muitos dos armazenamentos foram feitos com algoritmos de compactação de dados. Então, após extrair os dados é necessário descompactar e identificar o formato do arquivo armazenado.

[8] https://www.soapui.org/

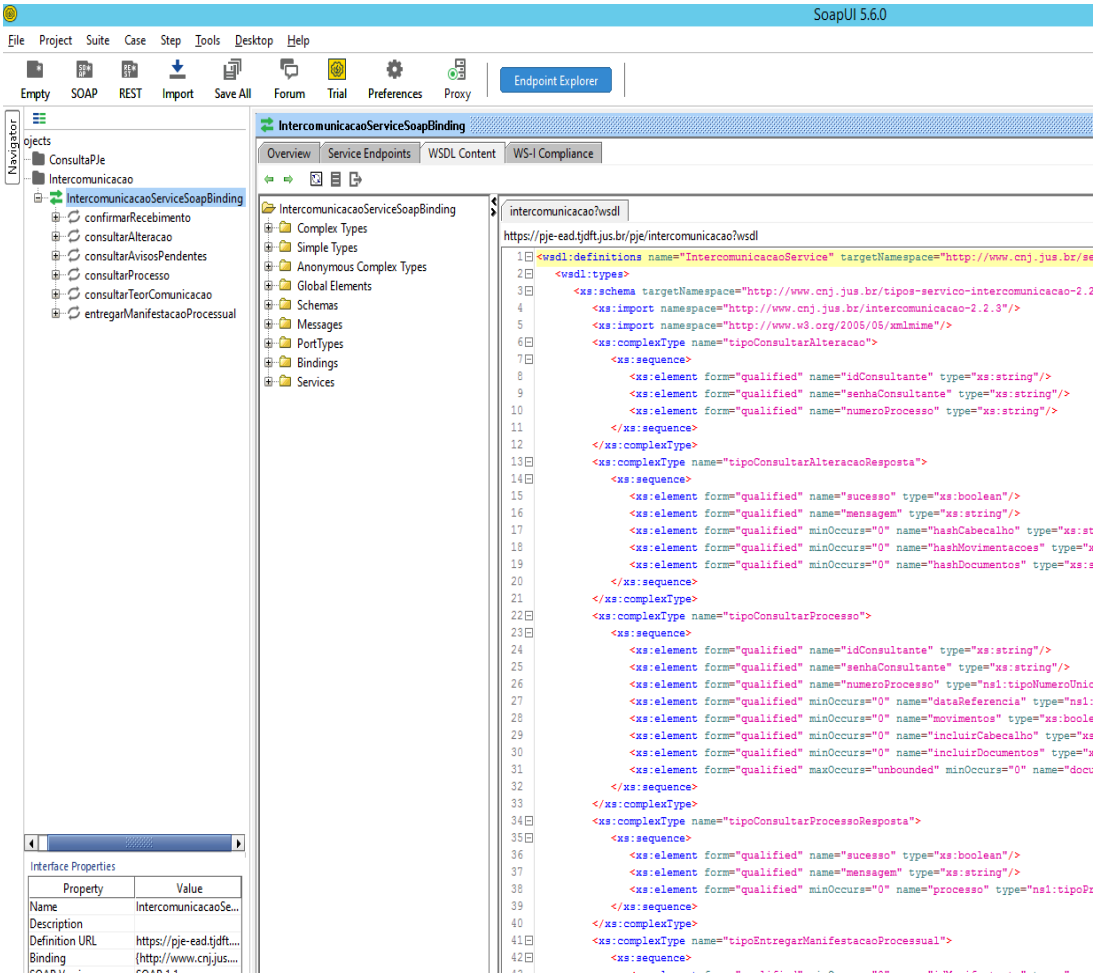

Figura 2.5: Tela do SOAPUI demonstrando um aceso ao MNI

As primeiras coletas de dados foram feitas por um *script* que conectava no banco de dados do sistema de gestão de documentos jurídicos próprio da Procuradoria Geral do Distrito Federal (PGDF) e extraía os arquivos que eram armazenados diretamente no banco de dados como Binary Large OBject (BLOB). Esses ***BLOB***, compactados e gravados em colunas de tabelas de bancos de dados destinadas ao armazenamento de documentos do sistema, estavam em formatos de arquivos *Rich Text Format (RTF)* e *Portable Document Format (PDF)* após a descompactação.

Então, após a conexão com o banco de dados e a devida consulta com os filtros específicos os arquivos dos processos de precatório foram retirados, descompactados e gravados em pastas no sistema de arquivos do servidor. Essas pastas foram denominadas com o número do processo dentro do sistema de gestão da documentos jurídicos da Procuradoria Geral do Distrito Federal (PGDF).

Desse modo, esta pesquisa está inserida dentro da PGDF, mais especificamente na DIPROJ, que são as unidades de análise da pesquisa em discussão. Nessas unidades de análise se buscou os processos de precatório. Para isso, os sistemas de informática, próprio

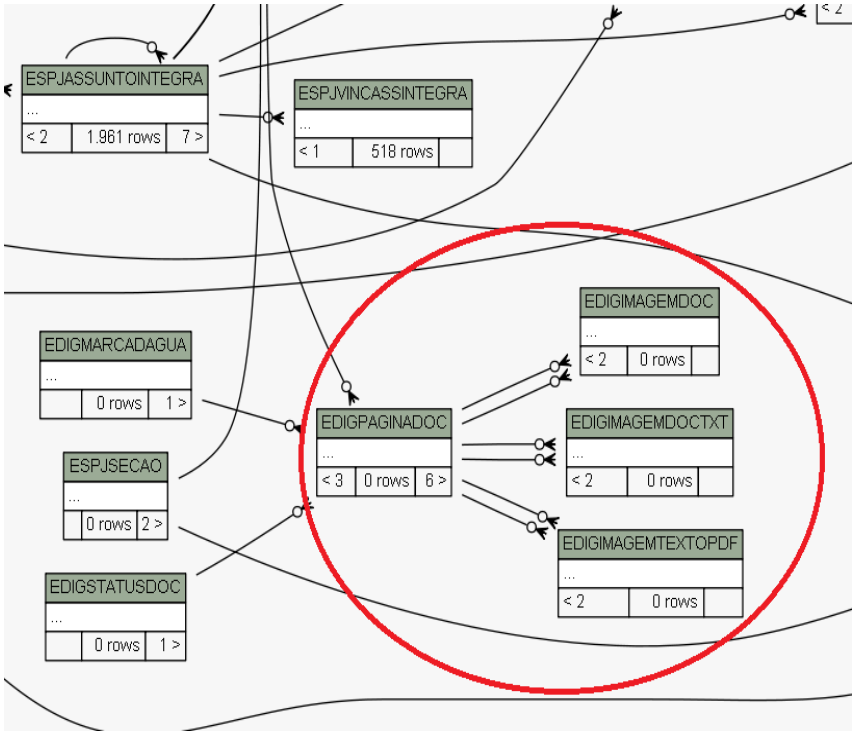

Figura 2.6: Tabelas que armazenam os documentos digitalizados - *Binary Large OBject (BLOB)*

sistema da PGDF e o PJe foram estudados. De posse dos acessos aos sistemas iniciou-se um trabalho de coleta e entendimento dos dados, que será apresentado na próximo capítulo.

Capítulo 3

Revisão de literatura

Neste capítulo está a Revisão Sistemática da Literatura executada para fundamentar os achados, discussão e conclusões da pesquisa. Segue apresentando a articulação dos autores sobre a amostragem em dados desbalanceados. Partindo para uma explanação sobre a classificação em cascata ou *ensemble learning* em bases desbalanceadas. Finaliza com um apanhado de estudos sobre classificação em textos jurídicos.

3.1 Revisão sistemática da literatura

Dresch et al. [33] definem a Revisão Sistemática de Literatura (RSL) como estudos secundários utilizados para mapear, encontrar, avaliar criticamente, consolidar e agregar os resultados de estudos primários relevantes acerca de uma questão ou tópico de pesquisa específico. Bem como, identificar lacunas nos campos de pesquisa a serem preenchidas, o que leva a RSL a resultados em formato de relatório ou síntese. Sendo que, as sínteses muitas das vezes são representadas por tabelas, algumas codificadas.

Na Figura 3.1, adaptado de Dresch et al. [33], têm-se o modelo de RSL utilizado no trabalho. Como pode ser visto nesta figura, inicialmente se define a questão de revisão. Com base na questão de revisão se determina qual a estratégia de busca será utilizada, por exemplo quais tipos de fonte de consulta e termos de busca serão empregados. Após as buscas se verifica a elegibilidade de cada texto encontrado baseado em critérios de seleção previamente estabelecidos e se registra a codificação dos estudos eleitos, por exemplo o grau de importância de cada um dos estudos selecionados. Esse processo culmina em uma etapa de análise com maior qualidade para os trabalhos eleitos baseado na codificação prévia, ou seja uma leitura mais detalhada é feita naqueles estudos considerados importantes ou de determinado tema etc. Por fim, uma síntese dos achados da revisão é apresentada.

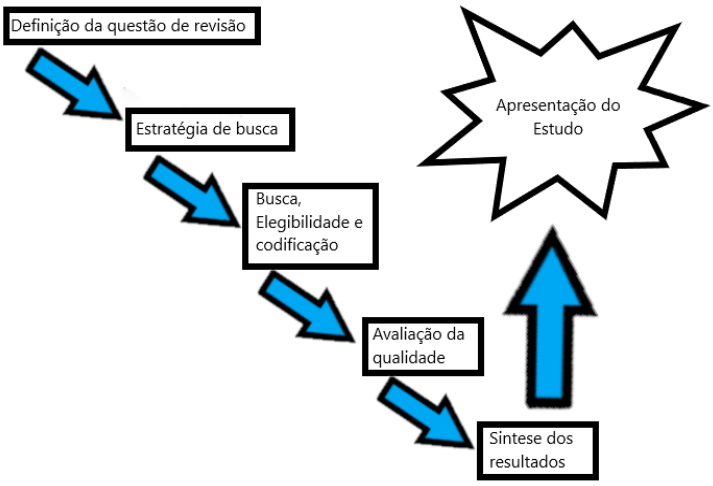

Figura 3.1: Modelo que representa as etapas de uma Revisão Sistemática de Literatura (RSL)

Tabela 3.1: Revisão sistemática para classificação de documentos jurídicos, etapa 1

Revisão Sistemática de Literatura				
Questão 1 de revisão : Qual o estado da arte na classificação em multi-classes?				
Fontes de busca:	1ª(primeira) consulta		2ª(segunda) consulta	
	Scopus (Elsevier)	Web of Science	Scopus (Elsevier)	Web of Science
Termos de busca:	legal AND documents AND classification		(AND machine AND learning)	
Quantidade de artigos:	659	319	177	40
Total	978		217	

Desse modo, uma RSL foi feita, como pode-se observar na Tabela 3.1. A primeira etapa da RSL é a definição da questão de revisão. No caso da Tabela 3.1 é a questão 1(um) que norteará os Termos de busca, as Fontes de busca e os critérios de seleção para um período de 5(cinco) anos. Neste caso, foram utilizadas as bases de dados Scopus (Elsevier) e Web of Science para responder a pergunta: Qual o estado da arte na classificação em multi-classes com o auxílio de aprendizagem de máquina ? Muitos dos artigos estavam disponíveis com mais facilidade por meio do Google Scholar, mas não se utilizou este para as contabilizações na síntese da RSL. Os totais nesta tabela decrementam na medida da execução das consultas porque os filtros são aplicados sequencialmente, um sobre os outros.

Como pode ser observado na Tabela 3.2, os filtros foram sendo feitos para a 3ª (terceira) e a 4ª(quarta) consulta. Os totais continuam decrementando. Mas no caso da 4ª (quarta)

Tabela 3.2: Revisão sistemática para classificação de documentos jurídicos, etapa 2

Revisão Sistemática de Literatura				
Questão 1 de revisão : Qual o estado da arte na classificação em multi-classes?				
Fontes de busca:	3ª(terceira) consulta		4ª(quarta) consulta	
	Scopus (Elsevier)	Web of Science	Scopus (Elsevier)	Web of Science
Termos de busca:	(AND Computer Science)		(AND (LIMIT-TO (PUBYEAR , 2021) OR LIMIT-TO (PUBYEAR , 2020) OR LIMIT-TO (PUBYEAR , 2019) OR LIMIT-TO (PUBYEAR , 2018) OR LIMIT-TO (PUBYEAR , 2017)) AND (LIMIT-TO (PUBSTAGE , "final"))	
Quantidade de artigos:	152	30	103	0
Total	182		103	

Tabela 3.3: Revisão sistemática para classificação de documentos jurídicos, etapa 3

Revisão Sistemática de Literatura		
Questão 1 de revisão : Qual o estado da arte na classificação em multi-classes com o auxílio de aprendizagem de maquina ?		
Fontes de busca:	5ª(quinta) consulta	
	Scopus (Elsevier)	Web of Science
Termos de busca:	((legal OR law OR judicial) AND (documents OR text) AND classification AND (machine AND learning))	
Quantidade de artigos:	129	140
Total	269	

consulta para a base de periódicos *Web of Science* o resultado foi zero. Desse modo, a verificação de elegibilidade foi feita nos artigos da 3ª (terceira) filtragem para a referida base.

A 5ª (quinta) busca foi aplicada livremente, ou seja, não foi um filtro sobre qualquer das buscas anteriores. Dresch et al. [33] sugerem que na fase de elegibilidade e codificação, a técnica de *screening* deve ser aplicada. Esta técnica é um processo que exige uma leitura inspecional para cada um dos estudos selecionados nas buscas. No caso da questão 1 (um) foram objeto do *screening* os 30(trinta) artigos do Web of Science na 3ª (terceira) busca e os 269(duzentos e sessenta e nove) artigos da 5ª (quinta) busca.

Nesta primeira fase foram verificados autores, títulos, resumos e DOI para identificar e codificar o método e a importância de cada texto eleito. Os estudos considerados

importantes irão para a próxima fase, que é uma leitura com maior qualidade, de inteiro teor, ou seja uma leitura analítica.

Para os achados da primeira questão aproximadoramente 30% dos trabalhos estão focados nos métodos *deep learning*, SVM, K-Nearest Neighbors (KNN), *Random Forest* ou *Logistic Regretion*. Sendo que a predominância dos estudos está focada em métodos que aplicam o *deep learning*. No total foram 28(vinte e oito) estudo considerados importantes para serem analisados com maior cuidado e nível de detalhes, inclusive aplicando técnicas de fichamento.

Por sua vez, a questão 2: Qual o estado da arte na classificação em cascata ou *ensamble learning* em textos ? teve resultados de busca inicial mais numerosos. Como pode ser observado nas Tabelas 3.4 e 3.5, os filtros ainda são aplicados sequencialmente. Isso implicou que ao final, após aplicação dos critérios de seleção e filtros, restaram 43(quarenta e três) artigos para serem avaliados com o *screening* e codificados. Neste caso, curiosamente o uso dos métodos que não se valem de *deep learning* foram a maioria. Restaram 9(nove) artigos considerados importantes para serem analisados com minúcia.

Tabela 3.4: Revisão sistemática para classificação em cascata ou *ensemble learning* em documentos jurídicos, etapa 1

Revisão Sistemática de Literatura				
Questão 2 de revisão : Qual o estado da arte na classificação em cascata ou ensamble learning em textos ?				
Fontes de busca:	1ª(primeira) consulta		2ª(segunda) consulta	
	Scopus (Elsevier)	Web of Science	Scopus (Elsevier)	Web of Science
Termos de busca:	TITLE-ABS-KEY ((cascade OR ensemble) AND text AND classification)		(AND Computer Science)	
Quantidade de artigos:	1129	388	936	306
Total	1517		1242	

Para encerrar a RSL, uma 3ª (terceira) questão de revisão: Qual o estado da arte na classificação com o auxílio do computador em bases desbalanceadas ? foi levantada. Neste caso, foram 74(setenta e quatro) trabalhos submetidos à técnica de *screening*, que resultou em 17(dezessete) artigos para serem analisados com maior nível de detalhes. Sendo o achado considerado mais relevante seria que a técnica em nível de dados mais referenciada, dentro do espectro selecionado, foi a Synthetic Minority Oversampling Technique (SMOTE).

As demais seções deste capítulo se concentram na revisão dos métodos de amostragem em dados desbalanceados, na classificação em cascata ou *ensemble learning* em dados desbalanceados e em trabalhos recentes quanto à classificação de textos jurídicos. A ideia

Tabela 3.5: Revisão sistemática para classificação em cascata ou *ensemble learning* em documentos jurídicos, etapa 2

Revisão Sistemática de Literatura				
Questão 2 de revisão : Qual o estado da arte na classificação em cascata ou ensamble learning em texto ?				
Fontes de busca:	3ª(terceira) consulta		4ª(quarta) consulta	
	Scopus (Elsevier)	Web of Science	Scopus (Elsevier)	Web of Science
Termos de busca:	last 5 years		TITLE-ABS-KEY ((cascade OR ensemble) AND text AND classification AND (legal OR judicial OR law))	
Quantidade de artigos:	540	306	21	22
Total	846		43	

Tabela 3.6: Revisão sistemática para classificação em dados desbalanceados

Revisão Sistemática de Literatura		
Questão 3 de revisão : Qual o estado da arte na classificação com o auxílio do computador em bases desbalanceadas ?		
Fontes de busca:	Scopus (Elsevier)	Web of Science
Termos de busca:	TITLE-ABS-KEY ((classification AND text AND imbalance AND (ensemble OR cascade OR hierarch))) AND (LIMIT-TO (SUBJAREA , "COMP")) AND (LIMIT-TO (DOCTYPE , "ar"))	
Quantidade de artigos:	47	27
Total	74	

é trazer estudos mais recentes possíveis que embasaram as decisões tomadas na fase de experimentação da pesquisa. Além disso, o capítulo entrelaça os pensamentos e afirmações publicadas no tema recentemente para criar um arcabouço de conhecimento que promova uma discussão mais sólida no trabalho e as conclusões da pesquisa.

3.2 Amostragem em dados desbalanceados

Nesta seção encontram-se as explicações sobre o problema muito encontrado na realidade da ciência dos dados, que é o problema das bases de dados para aprendizado de máquina com desbalanceamento em múltiplas classes. Aquelas bases de dados que tem uma grande quantidade de dados concentrada em uma ou poucas classes e uma concentração minoritária em determinadas classes.

Rout, Mishra e Mallik [34] definem que o problema de desequilíbrio significa que as instâncias de uma das classes (classe majoritária) são muito mais do que a outra classe (classe minoritária). Esses autores afirmam que a proporção entre as classes majoritárias e minoritárias podem ser na ordem de 100:1, 1000:1 ou 10000:1 e, em suma, as instâncias da classe majoritária superam grandemente a quantidade de instâncias da classe minoritária.

Para Fernández et al. [35] o problema de classes desbalanceadas está relacionado à aplicação da classificação no mundo real. No campo do problema de classificação, o cenário de desbalanceamento sobre os conjuntos de dados aparece quando o número de exemplos que representam as diferentes classes são muito diferentes. Para esses autores não há um consenso na comunidade de científica sobre o limite e a configuração para determinar se um conjunto de dados está desbalanceado. Esses autores consideram um conjunto de dados desbalanceado quando pelo menos uma das classes tem uma distribuição de exemplos abaixo de 40% do número de instâncias que pertencem à classe majoritária [35].

Como afirma Wang e Yao [36] o problema de desbalanceamento quando se tem múltiplas classes pode ser solucionado no nível dos dados, no nível do algoritmo e com métodos *ensemble*, cascateados ou encadeados. Segundo Russel e Noving [6] alguns trabalhos recentes na IA sugerem que, para muitos problemas, faz mais sentido se preocupar com os dados e ser menos exigente sobre qual algoritmo aplicar. Por isso e por uma questão de foco na solução do problema, este trabalho se concentra na solução do problema de desbalanceamento no nível dos dados.

Haixiang et al. [37] apresentam que as técnicas de pré-processamento geralmente são realizadas antes de construir o modelo de aprendizado para obter melhores dados de entrada. As técnicas de re-amostragem são usadas para reequilibrar o espaço de amostra a fim de aliviar o efeito da distribuição distorcida de classes no processo de aprendizagem. Os métodos de re-amostragem para esses autores [37] são mais versáteis porque são independentes do classificador selecionado.

Para Haixiang et al. [37] as técnicas de re-amostragem se enquadram em três grupos para equilibrar a distribuição da classe:

- Métodos de sobre-amostragem - *Oversampling*: eliminando os danos da distribuição distorcida, criando novas amostras de classes minoritárias. Por exemplo, o método *Synthetic Minority Oversampling Technique (SMOTE)*, amplamente usado, cria as amostras minoritárias sintéticas e/ou duplica aleatoriamente as amostras minoritárias.

- Métodos de subamostragem - *Undersampling*: eliminam os danos da distribuição enviesada, descartando as amostras intrínsecas na maioria. O método mais simples, porém eficaz, é o *Random Under-Sampling (RUS)*, que envolve a eliminação aleatória dos exemplos das classes maioritárias.

- Métodos híbridos: são uma combinação do método de sobre-amostragem e do método de subamostragem.

Na visão de Fernández et al. [35] o *Undersampling* quando se utiliza da técnica de *RUS* se vale de um método não heurístico, que visa equilibrar a distribuição de classes por meio da simples eliminação aleatória de exemplos de classes majoritárias. A principal desvantagem do *RUS* para estes autores seria que este método pode descartar dados potencialmente úteis que podem ser importantes para o processo de indução .

No que se refere ao *Oversampling*, Fernández et al. [35] afirmam que o método *SMOTE* se vale das classes minoritária. Este método cria uma super-amostragem, tomando cada amostra de classe minoritária e introduzindo exemplos sintéticos ao longo dos segmentos de linha que unem aos vizinhos mais próximos de classe minoritária k. Dependendo da quantidade de sobre-amostragem necessária, os vizinhos dos k-vizinhos mais próximos são escolhidos aleatoriamente. A desvantagem do método está exatamente no risco da fuga dos exemplos reais ao se criar dados sintéticos.

Resumindo, o *Undersampling* cria um subconjunto do conjunto de dados original, eliminando alguns dos exemplos da classe majoritária e o seu oposto, o *Oversampling*, cria um superconjunto do conjunto de dados original replicando alguns dos exemplos da classe minoritária ou criando novos exemplos sintéticos a partir das instâncias originais da classe minoritária. Esses métodos podem ser combinados em métodos híbridos, eliminando alguns dos exemplos antes ou depois da re-amostragem, a fim de reduzir o sobre-ajuste [35]. Afirma Gao [38] que no nível dos dados o reequilíbrio da distribuição de classe pode ser feito de dois modos: com técnicas de sub-amostradas para as classes majoritárias ou amostradas sintéticas com sobre-amostragem das classes minoritárias, ou a combinação de ambos.

Segundo Hasib et al. [39], quando a distribuição de classe em um conjunto de dados não é uniforme, os dados são chamados de desbalanceados. No estudo desses autores o problema de desbalanceamento de dados impede o desempenho do algoritmo de classificação. Ou seja, dependendo no nível de desbalanceamento a eficiência dos modelos preditivos são significativamente afetados. Nesses casos, há apenas um limitado número de instâncias representadas em pelo menos uma conhecida classes minoritárias e o restante do conjunto de dados consiste de outras classes majoritárias. Isso significa que as classes minoritárias fornecem especificidade mínima, por outro lado o modelo vai oferecer grande precisão na classe da maioria.

Na visão de Hasib et al. [39] o desbalanceamento de dados é uma preocupação comum na aplicação de aprendizado de máquina no mundo real e tem sido tema de atenção dos pesquisadores. Para esses autores o desempenho em um conjunto de dados desequilibrado

não é o mesmo esperado em um conjunto de dados balanceado. Por isso, tem-se que reprocessar os dados para fazer uma melhor precisão dos resultados.

Portanto, quando se trabalha com dados do mundo real tem-se que ter em mente que os dados estarem balanceados pode ser natural. Utilizando subamostragem ou sobre-amostragem pode-se minimizar esse desbalanceamento natural. Porém é preciso ter em mente que quando os dados são distorcidos por natureza, é realmente muito desafiador trabalhar com essas classes.

3.3 A classificação em cascata ou *ensemble learning* em bases desbalanceadas

Além das técnicas de pré-processamento como as de amostragem em bases desbalanceadas, um bom método que aumenta a precisão de classificadores seria o cascateamento de classificadores. Para Gama e Brazdil [40], os métodos que mesclam classificadores melhoram a generalização da classificação, pois acoplam classificadores livremente, os empilhando por exemplo. Segundo esses autores o acoplamento de classificadores é uma abordagem de dividir para conquistar, aplicando a generalização de cada classificados em estruturas para a melhoria do desempenho.

Para Gao et al. [38], normalmente, o problema de classificação binária envolve duas classes, e no problema de classificação multi-classe, o número de classes é maior que dois. O problema de classificação multi-classe é mais complicado do que o problema de classificação binária. O cascateamento de classificadores pode também ser chamado de *ensemble learning*. Nesta abordagem, como apresentam Gao et al. [38], uma boa estratégia sobre o problema de multi-classes seria dividir o problema em problemas menores com a binarização da classificação.

Ou seja, ao invés de se tratar as multiplas classes de uma vez só, as classificações são feitas de duas em duas classes. As duas formas mais comuns de binarização são um-contra-um (on-versus-one - OVO) e um-contra-todos (one-versus-all - OVA). A abordagem OVO divide um problema multi-classe com m classes em $m(m+1)/2$ sub-problemas binários, onde cada classificador no OVO discrimina um par de classes c_i, c_j. A abordagem OVA divide um problema multi-classe com m classes em m subproblemas binários, e cada classificador trata uma das classes como a classe positiva e todas as outras como negativas. Comparado com o esquema OVO, quando o conjunto de dados contém mais classes, a abordagem OVA implanta menos recursos ou usa menos classificadores.

Interessante são as ideias de Fernández et. al [35] que propõem que os problemas de desbalanceamento com múltiplas classes podem ser lidados com base na combinação de abordagens binárias OVO e OVA. O cascateamento em uma abordagem OVA, por

exemplo apresentado na Figura 3.2, tem a ideia básica da Generalização em Cascata e é utilizado para sequenciar um conjunto de classificadores.

Na Figura 3.2, adaptado de Gama [40], as classificações vão ocorrendo em estágios empilhados e onde a classe negativa - Outras - vai sendo passada para os outros estágios na tentativa de haver apenas uma classe alvo - positiva - a ser classificada por estágio. A intensão deste modelo é incrementar o desempenho total da classificação automática e minimizar a classificação manual.

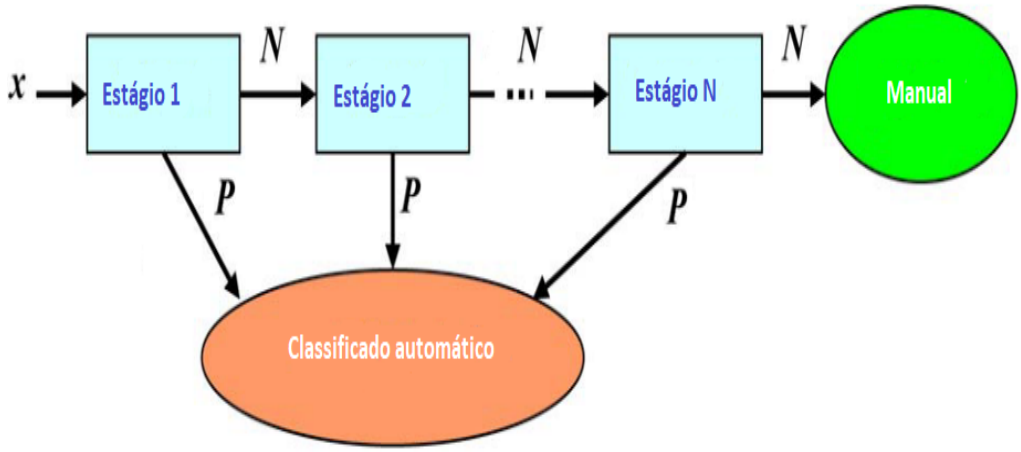

Figura 3.2: Empilhamento de algoritmos para cascateamento

.

Para medir o desempenho do modelo com os algoritmos em cascata pode ser feita uma média simples entre as métricas de desempenho de cada estágio da cascata. Ou seja, em cada etapa do cascateamento deve-se calcular uma das métricas apresentadas anteriormente, por exemplo acurácia [41], e ao final do processo se computa a média daquelas métricas de cada etapa. Importante notar que a métrica utilizada deve ser a mesma para todos os estágios e que cada estágio tem o mesmo peso ou importância na classificação, já que a comparação deve seguir um mesmo parâmetro. Um-contra-todos (OVA) é um dos métodos de decomposição convencionais pelos quais vários classificadores binários são usados para resolver tarefas de classificação multi-classes.

Para Gao et al. [38], no OVA cada classificador considera apenas uma determinada classe como positiva e todas as outras classes negativas, o que simplifica o problema e pode, em alguns casos, resolver problemas de desbalanceamento de classes pois, se existe um desbalanço que se equilibra ao unir todas as classes positivas o problema de desbalanceamento tende a ser minimizado.

No entanto, quando não há desequilíbrio ao utilizar o OVA na base entre as classes, o número de todas as outras classes pode ser maior do que o de classe alvo (positiva). Assim, isso causará o desequilíbrio no número de amostra, o que afetará o efeito de classificação de cada binário classificador. Sendo assim o autor sugere o uso de métodos de amostragem no pré-processamento para equilibrar o problema antes de aplicar o OVA.

Por sua vez, Rifkin e Klautau [42] ensinam que um dos esquemas de classificação multi-classe mais simples seria se construir sobre um método binário N classificadores binários diferentes, cada um treinado para distinguir os exemplos em uma única classe dos exemplos em todas as classes restantes. Quando se deseja classificar um novo exemplo, os N classificadores são executados e o classificador que produz o maior valor (mais positivo) é escolhido.

Rifkin e Klautau [42] defendem que um simples esquema um-contra-todos (OVA) é tão preciso quanto qualquer outra abordagem, assumindo que os classificadores binários são classificadores regularizados bem ajustados. A questão crucial para esses autores seria que provavelmente combinar classificadores simples pode alcançar resultados tão fortes e relevantes quanto usar classificadores sofisticados.

Como exemplo tem-se o estudo de Wang e Patrick [43] que propuseram um cascateamento de *Conditional Random Fields*, *Support Vector Machine (SVM)* e *Maximum Entropy* para reconhecimento de entidades nomeadas. A ideia dos referidos autores era a realização de reclassificações naqueles dados mal classificados com o cascateamento. O resultado foi um incremento nas medidas de F_1-*score* quando do uso de algoritmos em cascata, chegando a 83,26%.

No trabalho de Halgrim et al.[44] foi implementado um sistema híbrido composto por duas partes. A primeira parte detecta, com o uso de cascateamento de classificadores estatísticos, entidades nomeadas relacionadas a medicamentos em textos clínicos. O trabalho utilizou um pipeline com *Maximum Entropy* e heurísticas, combinado com técnicas de tagging. A medida de avaliação foi o F_1-*score* e os resultados se mostraram acima de 88%.

Em um estudo sobre identificação automática de linguagem, Kosmajac e Keselj [25], procuram desenvolver um sistema que identifique automaticamente a linguagem eslava, principalmente perante linguagens muito parecidas do leste europeu. Este trabalho combinou vários algoritmos, ao todo foram 17 (dezessete). A abordagem em cascata foi feita em dois estágios em uma arquitetura do tipo árvore de decisão. Os melhores classificadores nesta arquitetura foram a Regressão Logística e SVM. A métrica de performance utilizada foi a acurácia e os resultados ultrapassaram 90%.

Por sua vez, Calvo e Gambino [41] focaram em usar o cascateamento de classificadores combinados com características léxicas textuais para identificar sentimentos. O cascate-

amento foi feito com variações do *Naive Bayes*. A arquitetura do modelo foi construída com dois estágios de *Naive Bayes*. O primeiro com um classificador *Multinomial Naive Bayes* e o segundo estágio com uma árvore de classificação com *Naive Bayes*. A métrica utilizada para medir a performance foi a acurácia. O uso do cascateamente levou a uma acurácia de aproximadamente 65,6%.

No estudo de Plawiak et al. [45], uma abordagem de Deep Genetic Cascade Ensembles of Classifiers (DGCEC) baseada em um conjunto de cascateamento profundo de classificadores SVM. Neste estudo estes classificadores são aplicados aos dados de instituições financeiras australianas. A arquitetura geral da abordagem proposta consiste em *deep learning*. Neste modelo alcançou-se a maior precisão de predição com 97,39%.

Por fim, mais recentemente, Nikolic et al.[46] utilizaram classificadores em cascata em um conjunto de classificadores de SVM organizados em uma estrutura para realizar classificação binária. Os autores defendem que a vantagem dessa abordagem é a modularidade, ou seja, os classificadores binários podem ser treinados independentemente e otimizados para sua tarefa de classificação específica. A medida de desempenho adotada foi a F_1-*score* e o melhor desempenho foi em torno de 89%.

Portanto, combinar classificadores pode ser tão bom quanto aplicar métodos sofisticados que podem ser mais dispendiosos na fase de treinamento. O cascateamento ou *ensemble learning* pode se valer de uma abordagem OVA para diminuir a complexidade do problema e dividir as classificações em problemas mais simples para conquistar uma classificação tão boa quanto aquelas utilizando métodos mais sofisticados.

3.4 Classificação de textos jurídicos

O uso do processamento de documentos jurídicos com o auxílio do computador está sendo trabalhado em diversos estudos, como pode-se observar nesta revisão de literatura de Surden [47]. O autor realiza um apanhado dos estudos no campo do aprendizado de máquina aplicado no meio jurídico.

Surden [47] discute os estudos de aprendizagem de máquina como modelos de previsão para o meio jurídico, por exemplo a previsão de sentenças. Também, apresenta a aplicação do aprendizado de máquina ajudando na análise de padrões não descobertos no tema, como o padrão de decisões de um determinado magistrado. Além disso, outra aplicação destacada pelo autor, seria a classificação e o agrupamento de documentos jurídicos.

Para os documentos das cortes brasileiras, a aprendizagem de máquina tem sido aplicada em vários outros estudos. Araujo et al. [48], por exemplo, disponibilizam uma base de dados completa, de onde foi desenvolvido o trabalho acadêmico denominado VICTOR, sobre o processamento de linguagem natural em documentos jurídicos com *Support Vec-*

tor Machine (SVM), Naive Bayes (NB) e Extreme Gradient Boosting (XGBoost), redes convolucionais e recorrentes, chegando a um resultado em F_1-*score* de 88,87%.

Em outro estudo realizado no Brasil, Braz et al. [49] realizaram um trabalho de classificação de documentos jurídicos utilizando redes neurais artificiais convolucionais, *Bidirectional Long Short-Term Memory network (Bi-LSTM)*, chegando a um F_1-*score* de 84%. Braz et al. [50], em outro estudo, seguem a mesma linha do trabalho anterior, com base de dados similar, aplicando *Convolutional Neural Network (CNN)* para classificação dos documentos jurídicos, registra resultados de F_1-*score* em 90,35%.

Saindo do campo da classificação dos documentos, pode-se observar o trabalho de Araújo et al. [51] que utiliza a técnica de reconhecimento de entidades nomeadas em documentos jurídicos para melhorar o desempenho da recuperação da informação e processo de tomada de decisão, encontram resultados de F_1-*score* na ordem de 88,82%.

Por sua vez, fora do Brasil, Waltl et al. [52] aplicaram no judiciário alemão algoritmos de aprendizagem de máquina para classificação de documentos jurídicos. Tradicionalmente, segundo Waltl et al. [52], o algoritmo *Support Vector Machine (SVM)* é muito usado para a classificação de documentos por ter um bom desempenho. Dentre os métodos empregados pelos autores que apresentaram excelente desempenho as redes neurais e *Support Vector Machine (SVM)* se destacam com F_1-*score* atingindo 90%.

Um outro trabalho interessante no campo foi o de Catania et al. [53], que utilizaram os algoritmos NB e C4.5 para classificar documentos jurídicos nos tribunais italianos, chegando a 90,55% de acurácia.

Destaca-se o trabalho de Rocha [2], que aplicou algoritmos de classificação em um conjunto de 241 mil documentos do tipo Recursos Ordinários da Justiça do Trabalho por assunto principal. Foram comparados os algoritmos *Multinomial Naïve Bayes, Multi-Layer Perceptron, Random Forest e SVM*, chegando a uma micro precisão máxima de 75,21% com o Random Forest. Isso, após fazer uma análise considerando o acerto dos modelos visando não apenas o assunto principal, mas avaliando se o modelo acertou qualquer um dos assuntos existentes no processo.

Por sua vez, Undavia, Meyers e Ortega [54] trabalham a comparação de classificadores na suprema corte norte-americana. Utilizam agrupamento de documentos jurídicos com o algoritmo *Latent Dirichlet Allocation (LDA)*, se valem das *Convolutional Neural Networks (CNN)* e outros algoritmos em seu trabalho para alcançar uma acurácia de 72,4%.

São muitos os campos que o *deep learning* está sendo útil no mundo jurídico. Como afirmam Bansal, Sharma e Singh [19], ele - *deep learning* - está penetrando em todos os domínios possíveis do aprendizado de máquina e o domínio jurídico também está recebendo os benefícios desse tipo de técnica. Por exemplo: Elnaggar et al. [55] utilizam *deep learning* para sumarização de textos jurídicos na Alemanha. Os dados se tratavam

de corpus de texto do parlamento europeu (Europa) e do Joint Research Centre - Acquis Communautaire (JRC-Acquis). Landthaler et al.[56] trabalharam *Word Embeddings* para buscas em textos jurídicos. Os dados foram da *General Data Protection Regulation (GPDR)*. Por sua vez, John et al. [57] implementaram um sistema para perguntas e resposta com *deep learning* no tema jurídico.

Kowsrihawat et al. [58] realizam previsões de sentenças judiciais com *deep learning* na suprema corte da Tailândia. Os melhores resultados foram para as sentenças que consideravam o reu culpado, chegando a 74,38% de precisão com a métrica F_1-*score*.

Outro trabalho que procurou prever sentenças judiciais foi de Li et al. [59]. Os autores procuraram implementar *deep learnig*, mais especificamente uma rede *Long Short-Term Memory (LSTM)*, em aproximadamente 25.000(vinte e cinco mil) julgamentos. O melhor resultado foi uma precisão de 95,5% dos casos criminais.

Polo, Ciochetti e Bertolo [60] desenvolveram um classificador com *deep learning* em dados jurídicos brasileiros. O objetivo do classificador foi classificar os textos jurídicos em três classes: Arquivado, Ativo ou Suspenso. Os dados foram extraídos dos Tribunais de Justiça de São Paulo e Rio de Janeiro, com 6449(seis mil quatrocentos e quarenta e nove) processos. O resultado do trabalho dos autores chegou a 93% precisão em F_1-*score*.

Wei et al. [61] fazem classificações de texto jurídicos com o uso de *deep learning*. Neste estudo foram feitas comparações entre o desempenho de *Support Vector Machine (SVM)* e *deep learning*. O modelo com o algoritmo SVM chegou a um *score* de 91,89% e o *deep learning* chegou a um F_1-*score* de 92,89%.

Portanto os métodos com *deep learning* foram os que chegaram a melhores resultados. Porém, alguns métodos que se valeram de modelo mais tradicionais e do *ensemble learning* também tiveram resultados bem interessante. A questão agora é fazer alguns testes para verificar quais modelos melhor se adaptam ao problema deste estudo. O próximo capítulo trata do desenvolvimento do trabalho, como os trabalhos foram desenvolvidos no sentido da metodologia e processos de trabalho, além de apresentar questões técnicas primordiais dos modelos escolhidos e do entendimento dos dados.

Capítulo 4

Desenvolvimento

Neste capítulo temos definições da metodologia científica empregada. Uma definição do *Cross Industry Standard Process for Data Mining (CRISP-DM)* é apresentada. Segue explicando a configuração dos experimentos com seus algoritmos e parâmetros. Parte para uma iniciação do entendimento dos dados e conclui com explicações sobre o pré-processamento dos dados ou a preparação dos dados para a modelagem de aprendizagem de máquina.

4.1 Metodologias e processos

No que se refere à metodologia científica que foi empregada na pesquisa, ela é quantitativa, pois é fundamentalmente computacional [62]. O método foi um misto do indutivo e dedutivo experimental, pois iniciou-se de modo indutivo explorando o problema, se passou para um estudo bibliográfico mais profundo, elaboração hipóteses e buscando a confirmação ou negação dessas hipóteses, tudo apoiado com o uso do computador e computação como ferramenta para se testar e explorar a melhor técnica de aprendizagem de máquina em documentos jurídicos. A pesquisa é aplicada, pois trata da aplicação de técnicas computacionais em um problema de classificação de documentação jurídica [63].

Este trabalho utiliza as técnicas e métodos científicos apresentados em Dresch et al. [33], que desenvolvem um modelo em etapas para alcançar o objetivo de pesquisa. O método dos autores é desenvolvido ciclicamente em um processo de refino do conhecimento da pesquisa sobre viabilidade e aprofundamento no tema.

Como complemento, desenvolveu-se neste estudo uma revisão de literatura, refinando os termos utilizados na busca pelo estado da arte no Portal de Periódicos Capes do Governo do Brasil, utilizando as bases de pesquisa Web of Science, Scopus e Google Scholar. Isso, se valendo das técnicas de Mariano e Santos, [64] e Lopez [65], onde as

perguntas de pesquisas, objetivos e definição do estado da arte no tema foram evoluídos gradualmente na medida que o estudo foi sendo realizado.

Para este estudo, como já mencionado, o CRISP-DM foi o processo de *Data Mining (DM)* escolhido, também muito evidente estão os demais passos desse processo nos próximos capítulos. Azevedo e Santos [66] definem CRISP-DM como um acrônimo para Cross Industry Standard Process for Data Mining (CRISP-DM), que é um processo, modelo ou metodologia dividido em seis fases, como demonstra a Figura 4.1: *Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation e Deployment.*

O CRISP-DM se trata de um processo cíclico que passa pelo entendimento do negócio e dos dados, seguindo para a preparação dos dados para a modelagem da mineração propriamente. Para a evolução constante do modelo, existe uma dependência da avaliação dos resultados - *Evaluation.* O processo pode retornar para o início, incrementando o *Business Understanding* - entendimento do negócio - e seguir novamente para entendimento e preparação dos dados. Após avalições sequenciais dos resultados e chegando-se a resultados satisfatórios os modelos segue para sua implementação em produção[67]. O mais interessante da Figura 4.1 [67] é que os dados estão no centro do processo.

Figura 4.1: Modelo que demonstra o Cross Industry Standard Process for Data Mining (CRISP-DM)

Portanto, existe um processo de mineração de dados a ser realizado para a classificações e agrupamento de dados com o auxílio de computadores. Não se trata da pura aplicação dos algoritmos nos dados nos dados coletados no negócio. A mineração de dados necessita de um processo para que os textos sejam preparados para a real aplicação dos algoritmos,

levando em consideração o problema do negócio e que haja uma verificação constante e cíclica de seus resultados prévios, até se chegar a um resultado final aceitável.

4.2 Configuração experimental - algoritmos e parâmetros

Observando os aspectos metodológicos de um ponto de vista mais técnico é importante registrar que as implementações foram feitas em duas linguagens de programação, a linguagem Python [1] e Java[2]. Os algoritmos de aprendizagem de máquina foram implementados em Python e praticamente todos foram importados da biblioteca scikit-learn [3], keras[4] e mallet [5]. O Mallet - que é uma biblioteca implementada em Java mas que possui uma interface implementada em Python - foi articulada com a biblioteca Gensim [6].

Os algoritmos em *deep learning* foram implementados com o uso da biblioteca keras, que possibilita a utilização da plataforma TensorFlow[7] - outra biblioteca implementada em Java - no Python. As visualizações utilizaram matplot [8] mas também se valeram da biblioteca LDAvis [9][68].

Com dificuldades no processamento dos dados encontrados no sistema da PGDF, foram coletadas as petições iniciais e recorreu-se ao sistema de informação do Tribunal de Justiça do Distrito Federal e dos Territórios (TJDFT). A coleta das petições iniciais no sistema de informação do TJDFT foi feita com um programa de computador na linguagem JAVA, que acessava o serviço Modelo Nacional de Interoperabilidade (MNI). Desse modo, foram coletadas 3943 (três mil nove centos e quarenta e três) processos e 165119 (cento e sessenta e cinco mil cento e dezenove) documentos.

Com relação aos recursos computacionais, para fins de reprodução dos experimentos, apresentamos que utilizaram-se 3(três) máquinas virtuais e um *notebook* para a execução dos modelos. As máquinas que processaram os modelos foram configuradas com 20(vinte) e 10(dez) vCPUs e 300 GB de memória RAM, executando no sistema operacional Linux. A máquina auxiliar, que executou as interfaces com o banco de dados, interação com o Jupyter *notebook* [10] e execução dos programas Java, tinha o sistema operacional Windows

[1] `https://www.python.org/`
[2] `https://www.java.com/en/`
[3] `https://scikit-learn.org/stable/`
[4] `https://keras.io/`
[5] `http://mallet.cs.umass.edu/`
[6] `https://radimrehurek.com/gensim/`
[7] `https://www.tensorflow.org/federated`
[8] `https://matplotlib.org/`
[9] `https://github.com/bmabey/pyLDAvis`
[10] `https://jupyter.org/`

Server. Esta última máquina virtual tinha configurada 2(duas) vCPUs e 8(oito) GB de memória. O *notebook* foi utilizado como suporte para administração dos códigos fonte e manipulação de imagens dos resultados, tinha um Intel core I7 com 20GB de memória RAM.

A seguir estão apresentados os algoritmos utilizados nos experimentos. Será apresentado como foi feita a escolha destes algoritmos e como eles foram parametrizados. Foram utilizados os algoritmos *Latent Dirichlet Allocation (LDA) e K-means*, além disso 15 modelos construídos com o uso do *scikit-learn* e o *deep learning* com a biblioteca keras.

Os algoritmos de aprendizagem não supervisionados foram *K-means* e o LDA. O LDA com as configurações de *alpha e beta* otimizadas por um algoritmo que compara a melhor coerência entre tópicos em uma iteração com intervalo que inicia em 0.01 até 1. Tendo o LDA sido combinado com K-means, utilizando as configurações padrão do *scikit-learn*.

Os algoritmos de aprendizagem de máquina supervisionado foram *Ridge Classifier* [69], *Perceptron* [70], *Passive Aggressive Classifier* [71], *K-Neighbors Classifier* [72] , *Random Forest Classifier* [73], *Linear SVC* [11] [74], *SGD Classifier* [12] [74], *Nearest Centroid* [75], *Multnomial*, *Bernoulli* e *Complement Naive Bayes* [13] [76].

Foram utilizados 15 (quinze) modelos em implementações diferentes com a biblioteca *scikit-learn*. A implementação foi uma adaptação de um exemplo fornecido na própria biblioteca do *scikit-learn* para classificação de documentos com a abordagem *bag-of-words* [14]. Não foram feitos experimentos variando os parâmetros dos algoritmos inicialmente, por serem estimativas iniciais da qualidade da classificação e pela falta de tempo.

O primeiro algoritmo implementado foi o *Ridge Classifier* com os parâmetros padrões da implementação do *scikit-learn*, excetuando $1e^{-2}$ para a precisão ou tolerância da regressão e o uso de "sag" para utilizar o *Stochastic Average Gradient Descent*. Em seguida houve uma implementação do algoritmo *Perceptron* com 50 interações ou épocas. O próximo foi uma implementação de *Passive Aggressive*, também com 50 interações ou épocas e os demais parâmetros padrões do *scikit-learn*. Ainda em um mesmo grupo de execução tem-se o algoritmo *Random Forest*, que não trouxe alterações nos parâmetros padrões da biblioteca e o *K-Neighbors* configurado para classificar os documentos de precatórios com uma vizinhança de 10 vizinhos.

Em um segundo grupo de execuções têm-se *Linear SVC* e *SGD Classifier* executados com penalidades L2 e L1, totalizando mais 4 (quatro) modelos. A penalidade "L2" é o padrão usado no SVC. O "L1" leva a vetores esparsos. O *Linear SVC* teve uma mudança

[11] `https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html`

[12] `https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html`

[13] `https://scikit-learn.org/stable/modules/naive_bayes.html`

[14] `https://scikit-learn.org/stable/auto_examples/text/plot_document_classification_20newsgroups.html#sphx-glr-auto-examples-text-plot-document-classification-20newsgroups-py`

da configuração padrão no critério de parada com $1e^{-3}$. Por sua vez, o *SGD Classifier* continua com 50 interações e foi optado por um alpha=.0001. Houve uma implementação do *SGD Classifier* com os mesmos parâmetros anteriores, mas com elasticnet de penalidade, que combina L2 e L1.

A implementação de *Nearest Centroid* não teve mudanças nos parâmetros e configurações padrões da proveniente pelo *scikit-learn*. Os próximos outros 2 (dois) modelos foram variações do *Naive Bayes*. O *Multinomial* e *Bernoulli Naive Bayes* tiveram um alpha de 0.01 em ambos. Seguindo para um *alpha de 0.1* para o Complement *Naive Bayes*. Por fim, foi utilizada uma implementação Pipeline para selecionar features de uma execução do *Linear SVC* com penalidade L1 e classificação com outra execução do *Linear SVC*, agora com penalidade L2.

Em seguida, uma implementação de *deep learning* foi feita com o uso da biblioteca keras e tersorflow. A camadas de entrada da rede foi criada automaticamente, a partir das *features* do texto. As camadas ocultas foram todas do tipo densa. Apenas a última camada foi definida manualmente de acordo com as saídas dos experimentos para definir a classificação binária ou multi-classe.

As funções de ativação da rede neural em *deep learning* foram sigmoid e relu para multi-classe e binária, respectivamente. Na última camada foi usada softmax para ativação em multi-classe e sigmoid em classificação binária. Para a última camada foram usadas binary_crossentropy e categorical_crossentropy na função de perda(loss). No processamento foram utilizadas 30 épocas para treinamento, validação e teste nas classificações binárias e na classificação multi-classe foram usadas 3 épocas no 10-Fold cross-valitation.

Na busca por aplicar o estado da arte em classificação de textos, buscou-se modelos pré-treinados em *deep learning*. A biblioteca *Simple Transformers*[15] foi utilizada, sendo ela focada no Processamento de Linguagem Natural (PNL) com modelos pré-treinados. Ela foi projetada para simplificar o uso de modelos pré-treinados de *Transformer*[16]. Modelos pré-treinado como BERT[17] e XLNET[18] podem ser encontrados nesta biblioteca.

Para superar o problema de desbalanceamento dos dados em múltiplas classes seguiu-se a abordagem *one-versus-all* (OVA) [37]. As implementações foram feitas combinando as funções GridSearchCV e OneVsRestClassifier do *scikit-learn*. A função GridSearchCV realiza um ajuste dos parâmetros por força bruta nos algoritmos de classificação, testando e selecionando os melhores parâmetros para os classificadores. O problema foi decomposto em uma abordagem dividir para conquistar. Desse modo, os modelos aplicados anteriormente com os parâmetros padrões da biblioteca *scikit-learn* foram otimizados.

[15] `https://simpletransformers.ai/`
[16] `https://github.com/huggingface/transformers`
[17] `https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html`
[18] `https://cloud.google.com/tpu/docs/tutorials/xlnet-2.x?hl=pt-br`

Para a amostragem nas bases de dados desbalanceadas utilizou-se a biblioteca *imbalanced*[19] do *scikit-learn*. As funções utilizadas foram *RandomUnderSampler*[20], *SMOTE*[21] e *SMOTEENN*[22]. Os parâmetros aplicados para a execução das funções foram os padrões da biblioteca, com $random\_state = 42$.

Os dados foram divididos em grupos de treinamento e teste em um formato *holdout*[2], com 70% para treinamento e 30% para testes. Esses dados foram escolhidos aleatoriamente com $random\_state = 42$ se valendo da função *train_test_split*[23] do scikit-learn. No ato de se utilizar a função GridSearchCV fez-se a validação cruzada. Assim, os dados para treinamento e teste também passaram pela técnica de *10-Fold cross-validation*.

Portanto, no quesito metodológico a pesquisa é do tipo aplicada computacional em documentos jurídicos com o apoio do aprendizado de máquina e com um processo de mineração de dados, *CRISP-DM*, seguindo os moldes metodológicos de Rocha [2] e Andrade [1], na tentativa de se aplicar o estado da arte em classificação de textos no problema de negócio da PGDF.

4.3 Entendimento dos Dados

Esta seção trata do processo executado para o entendimento dos dados. Após uma extração dos dados e limpeza esses dados foram analisados. A análise descritiva foi o primeiro passo, observando padrões e a organização desses dados. O que promove subsídios para um pré-processamento adequado e até mesmo um entendimento do negócio por meio desses dados.

4.3.1 Análise descritiva

Esta seção demonstra como se fazer uma análise descritiva dos dados, o que já cumpre parte dos objetivos do trabalho. Apresenta como os assuntos estão registradas e distribuídas. Possui discussão de como estão os dados em relação aos assuntos, parte da ontologia do CNJ e usadas no PJe. Por fim, traz exemplos de um aprofundamento dessa relação e distribuição dos assuntos em função dos dados.

[19] `https://imbalanced-learn.org/stable/`

[20] `https://imbalanced-learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html`

[21] `https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html`

[22] `https://imbalanced-learn.org/dev/references/generated/imblearn.combine.SMOTEENN.html`

[23] `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`

Inicia-se o entendimento dos dados por uma análise descritiva, realizando uma verificação de quantos processos de precatório a PGDF possuía. A necessidade inicial da PGDF para este trabalho seria apenas utilizar os dados de precatório. Para isso, os dados analisados foram extraídos do sistema de informação de gerenciamento de documentos próprio da PGDF. Esse sistema também contêm os documentos tramitados na justiça. Foram extraídos 8.609 (oito mil seiscentos e nove) processos, contendo 412.268 (quatrocentos e doze mil duzentos e sessenta e oito) documentos, em agosto de 2020.

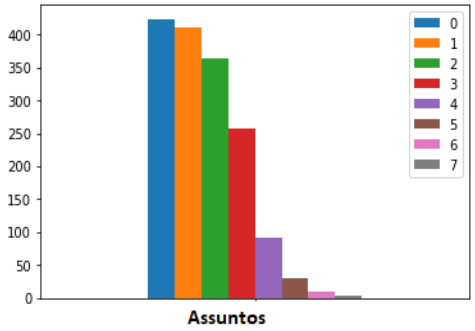

Figura 4.2: Quantidade acumulada nos níveis de classes

Realizando uma verificação das atuais assuntos utilizadas nesses processos coletados constam-se que são 8 (oito) níveis de classes e 424 (quatrocentos e vinte e quatro) classes. A Figura 4.2 ilustra a quantidade de utilizações de cada nível de assuntos. Pode-se observar que as primeiras classes são plenamente utilizadas nas classificações dos processos e as demais vão sendo utilizadas na medida da especificidade. Então, nem sempre há classificações com todos os níveis de assuntos. Por exemplo: todas as 424 (quatrocentos e vinte e quatro) classes tem o nível 0(zero) de classificação, porém o nível 7(sete) só possui 4 (quatro) assuntos. Mas todos os 8.609 (oito mil seiscentos e nove) processos são classificados.

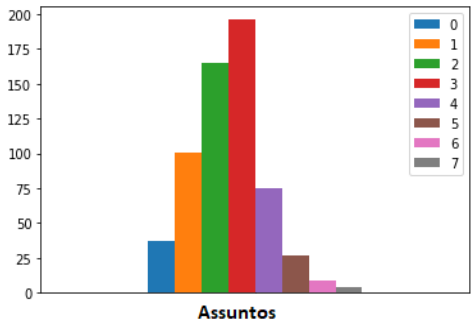

Figura 4.3: Distribuição das múltiplas classes nos níveis

Ainda sobre as classes encontradas, pode-se extrair da Figura 4.3 o quantitativo de classes para cada nível. O gráfico inicia com 37 (trinta e sete) classes para o primeiro nível, indo ao ápice de 196 (cento e noventa e seis) classes no nível 3 (três) e descendo a 4 (quatro) classes no nível 7 (sete), oitavo assunto. A Tabela 4.1 apresenta cada quantitativo de assuntos por nível de classificação.

Tabela 4.1: Quantidade de classes em cada nível

Níveis	0	1	2	3	4	5	6	7
Quantidade	37	101	165	196	75	27	9	4

Esses assuntos coletadas são combinadas no banco de dados no formato de texto separadas por hífen (-), como por exemplo : **"Direito Administrativo e Outras Matérias do Direito Público - Responsabilidade da Administração - Indenização por Dano Moral"** ou **"Direito Tributário - Impostos - ITBI - Imposto de Transmissão Intervivos de Bens Móveis e Imóveis Direito Administrativo e Outras Matérias do Direito Público - Servidor Público Civil - Sistema Remuneratório e Benefícios - Gratificações Estaduais Específica"**.

Nesses exemplos, o primeiro apresenta 3(três) níveis de assuntos e o segundo com 7(sete) níveis de assunto. Ou seja, os assuntos são separadas por hífen (-) no banco de dados, registradas no formato de texto.

Como mencionado anteriormente, têm-se 424 (quatrocentos e vinte e quatro) classes\ assuntos para todos os processos. Importante lembrar que para este estudo utilizaram-se apenas 8.609 (oito mil seiscentos e nove) processos, contendo 412.268 (quatrocentos e doze mil duzentos e sessenta e oito) documentos. Desse modo, se a distribuição dos processos por classe fosse uniforme se teriam quase 20 (vinte) processos por classe.

Porém, encontrou-se uma distribuição desbalanceda para as classes, por exemplo, para o nível 0 tem-se a distribuição apresentando na Figura 4.4. Pode-se treinar os algoritmos para algumas classes, mas, para outras não pois existem poucos exemplos. Pode-se ter que descartar aquelas classes com muitos poucos processos.

Importante lembrar que essa classificação vem da ontologia do CNJ por meio da Resolução nº 46 de 18 de dezembro de 2007 e é utilizada em milhares e milhares de processos, gerando maiores quantidades de processos por classes em outros contextos.

A Tabela 4.2 demonstra em números a disparidade entre assuntos Direito Administrativo e Outras Matérias do Direito Público e as demais classes, com 6724 (seis mil setecentos e vinte e quatro) processos nestes assuntos diante das demais 36 (trinta e seis) possibilidades para o primeiro nível de classificação.

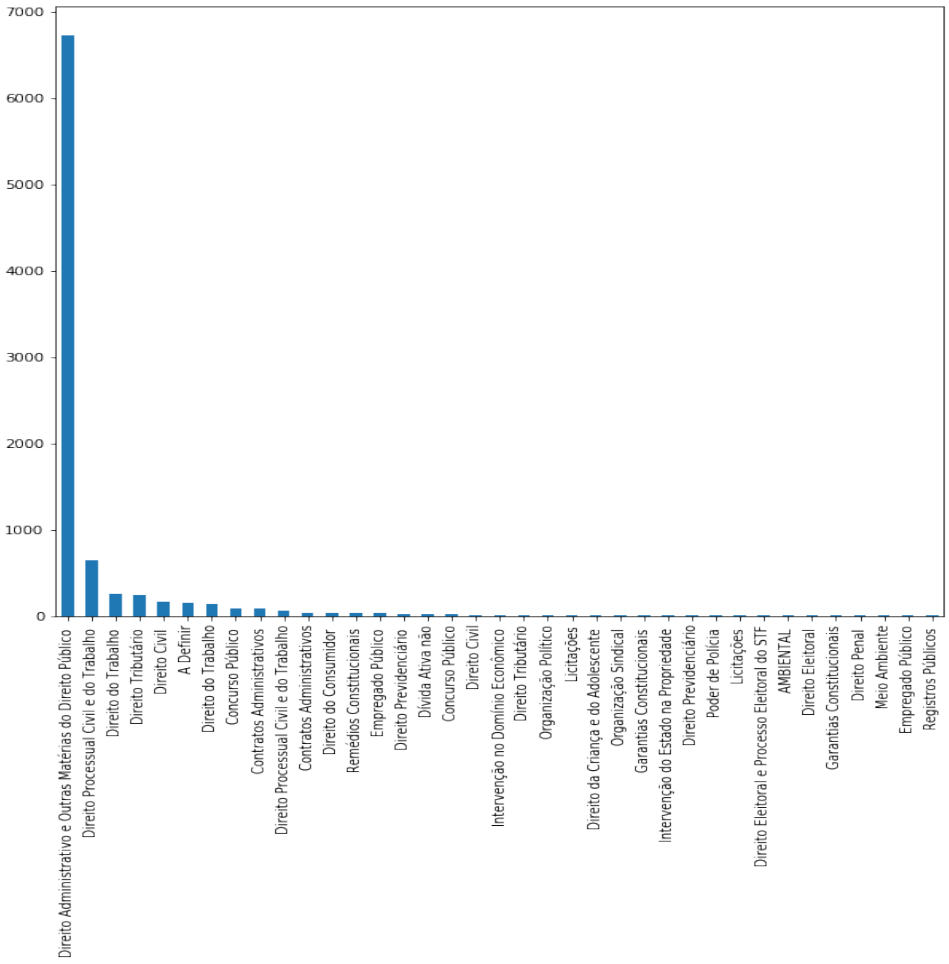

Figura 4.4: Quantidades de classificações para o primeiro nível de classes

Além disso, esses assuntos têm seus subassuntos, como pode ser visto na Tabela 4.3. Nesta Tabela, por exemplo, os subassuntos de Direito Administrativo e Outras Matérias do Direito Público não tem último nível - oitavo.

Nessa linha de observação e análise descritiva dos dados, para melhor entender como eles estão organizados e distribuídos, pode-se observar a Figura 4.5, Figuras 4.6 e 4.7 que demonstra a discrepância entre os subassuntos do nível 1 (um) - Direito Administrativo e Outras Matérias do Direito Público. Mesmo em níveis de classificação mais profundos tem-se desbalanceamento.

Portanto, para o trabalho de agrupamento e classificação automática, pode-se perceber que há um grande desbalanceamento nas classificações dos processos coletados. Outro ponto que merece destaque é a grande quantidade de classes e níveis, que não são usados exclusivamente para precatórios no judiciário, mas que acabam diluindo muito os dados entre muitos assuntos e subassuntos. Na próxima subseção é possível demonstrar como foi feito para preparar os dados para a execução dos modelos.

Tabela 4.2: Quantidade de classificações para o primeiro nível de classes

Classe/Assunto	**Quantidade de processos**
Direito Administrativo e Outras Matérias do Direito Público	6724
Direito Processual Civil e do Trabalho	640
Direito do Trabalho	258
Direito Tributário	246
Direito Civil	164
A Definir	153
Direito do Trabalho	137
Concurso Público	88
Contratos Administrativos	81
Direito Processual Civil e do Trabalho	55
Contratos Administrativos	39
Direito do Consumidor	38
Remédios Constitucionais	29
Empregado Público	28
Direito Previdenciário	25
Dívida Ativa não	15
Concurso Público	15
Direito Civil	9
Intervenção no Domínio Econômico	8
Direito Tributário	8
Organização Político	6
Licitações	4
Direito da Criança e do Adolescente	4
Organização Sindical	3
Garantias Constitucionais	3
Intervenção do Estado na Propriedade	2
Direito Previdenciário	2
Poder de Polícia	2
Licitações	2
Direito Eleitoral e Processo Eleitoral do STF	2
Ambiental	1
Direito Eleitoral	1
Garantias Constitucionais	1
Direito Penal	1
Meio Ambiente	1
Empregado Público	1
Registros Públicos	1

4.4 Pré-processamento dos Dados

Esta seção apresenta, de modo geral, como foi a etapa de pré-processamento. Inicia explicando desde a coleta do dado na fonte de informação, passando pelas transformações nesse dado e no armazenamento em um formato que será consumido pelos modelos de agrupamento e classificação com auxílio do computador.

Coletados os documentos de cada processo e incluídos em suas respectivas pastas, um script foi executado para converter os arquivos RTF e PDF em texto pleno por meio de *Optical Character Recognition (OCR)*. Nesse processo 176 (cento e setenta e seis) documentos não foram convertidos para texto pleno devido a erros no processo de transformação.

Após todo o processo de OCR outro script foi criado para remover pontuação, remover acentos, transformar letras maiúsculas para minúsculo, remover números, retirar espaços em branco desnecessários, retirar quebras de linhas em excesso e fazer o stemming. O

Tabela 4.3: Quantidade de classes no nível 0, primeiro nível

Níveis	0	1	2	3	4	5	6	7
Quantidade	1	9	36	117	31	12	2	0

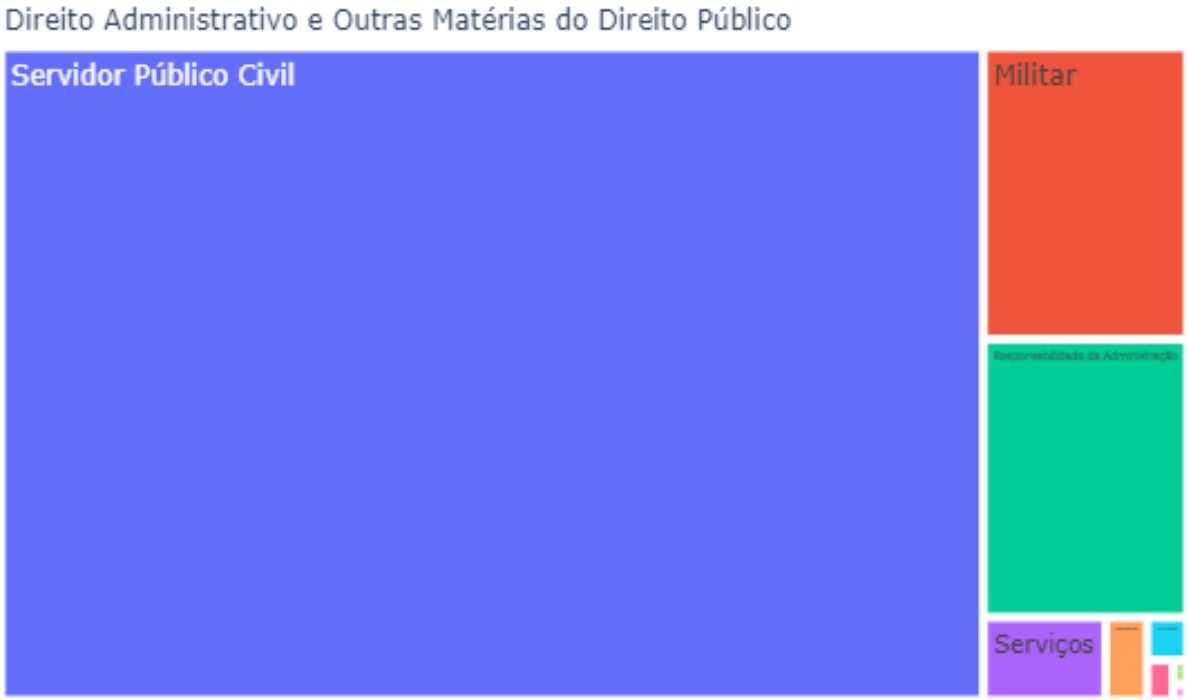

Figura 4.5: Distribuição de classificações para o segundo nível

processo de pré-processamento foi finalizado gravando-se um arquivo *Comma Separated Values (CSV)*, que passa a ser o ponto de partida de nosso corpus.

O pré-processamento dos dados inicia-se considerando como os dados estão armazenados na fonte de informação. Pois, dependendo do formato do dado alguns algoritmos de decompactação dos dados e OCR serão necessários. Ou seja, se o formato estiver em imagem, PDF etc. é necessário executar o OCR. Após a obtenção do texto pleno algumas transformações ainda se fazem necessárias para preparar o texto para ser processado pelos modelos. Por exemplo, transformar os rótulos em vetores alvo passando os textos para números. Outro exemplo é a transformação dos dados a ser classificado em vetores de probabilidade de palavras etc. A próxima seção mostrará como estes dados foram processados e discute alguns resultados.

Nos próximos capítulos têm-se os resultados e discussão a partir do retorno e achados dos experimentos. Nestes capítulos estão os complementos para o entendimento dos dados e modelagem. A etapa de avaliação dos modelos trabalhados, também, se encontra nestes próximos capítulos.

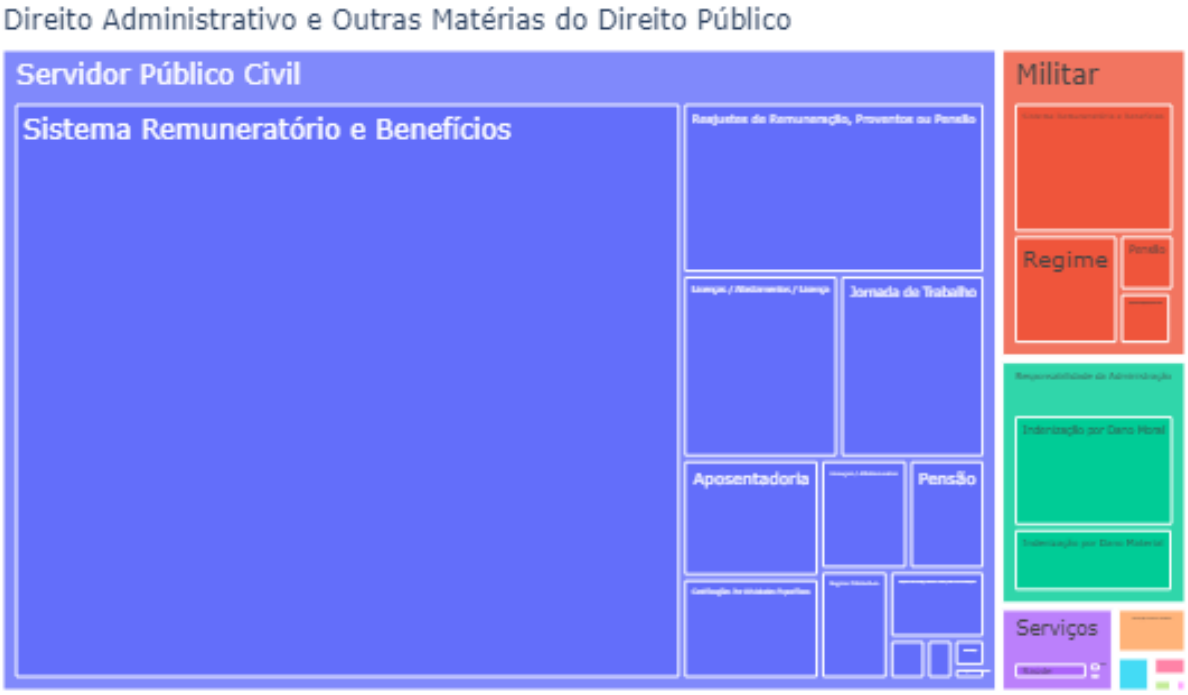

Figura 4.6: Distribuição de classificações para o terceiro nível

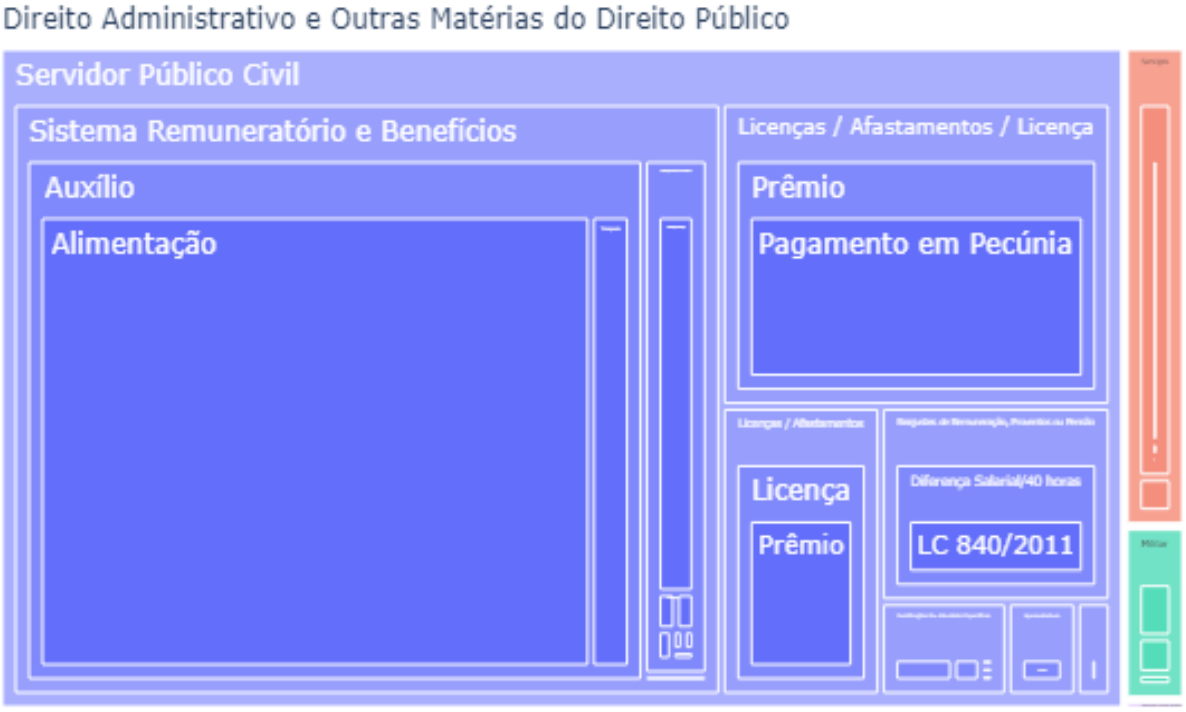

Figura 4.7: Distribuição de classificações para o quarto nível

O entendimento do negócio do *Cross Industry Standard Process for Data Mining (CRISP-DM)*, por uma questão de organização do texto e melhor entendimento do pro-

blema, foi iniciada na fundamentação do trabalho. Por sua vez, o entendimento dos dados foi iniciado neste capítulo. Porém, adiante podem ser encontradas as outras rodadas de entendimento do negócio, dos dados, bem como, a modelagem e avaliação dos resultados. Ou seja, todas as demais etapas do processo do CRISP-DM estão a seguir e serão discutidas mediante o alcance dos resultados.

Capítulo 5

Experimentos com precatório

Neste capítulo são explanados os experimentos feitos com dados de precatório puramente. Demonstra as modelagens iniciais da pesquisa e a evolução do processo diante do entendimento do problema. Apresenta as razões e caminhos para as modificações na base de dados. Destaca as dificuldades no processo de modelagem e as evoluções com a seleção de novas técnicas e algoritmos. Apresenta resultados e discussão quanto à implementação dos algoritmos de aprendizagem de máquina já introduzidos, a classificação com binarização, realce para o uso das redes neurais artificias com *deep learning*, o uso da técnica One-versus-All (OVA) e os algoritmos não supervisionados.

5.1 Modelagens iniciais

Esta seção tem o foco em apresentar as primeiras execuções de algoritmos de aprendizagem de máquina. Inicia apresentando as tentativas de classificação dos documentos jurídicos com algoritmos supervisionados. Foram feitas várias tentativas com 15 (quinze) modelos e não houve sucesso. Os resultados para o desempenho dos modelos foram inferiores ao esperado. O trabalho segue nas próximas seções em uma busca exploratória da preparação dos dados e da seleção do modelo ótimo.

5.1.1 Algoritmos supervisionados

Os trabalhos com os modelos supervisionados foram feitos, inicialmente, com a aplicação os 15 (quinze) modelos em todos os documentos coletados e todas as classes coletadas. Porém, o primeiro resultado interessante foi com a segmentação das classes - cada nível de classe foi separado.

Como pode ser visto na Figura 5.1 teve-se um bom resultado aparentemente, levando em consideração os resultados de desempenho esperados pela PGDF, com acertos na faixa

de 85% (oitenta e cinco) por cento[1]. Importante registrar que esta expectativa de acertos foi coletada em entrevistas dentro da unidade de análise e outros registros feitos em outros trabalhos[2] com objetivos similares a este. Neste caso, utilizando apenas o primeiro nível de classificação, teve-se resultados interessantes com acurácia maior que 80%.

Entretanto, esses resultados não podem ser verificados unicamente por essa métricas de desempenho pois, pode-se perceber quando visto os resultados pela Tabela 5.1 que os melhores resultados entre os 15 (quinze) classificadores na maioria dos casos não acertou nada em determinadas classes.

Portanto, para melhorar o desempenho, limitou-se a classificação entre aquelas classes que tinham acima de 50 processos. A Figura 5.2 apresenta resultados aparentemente similares aos anteriores. Todavia, quando se observa as Tabelas 5.2 e 5.3 vê-se que a limitação de número de documentos para maiores que 50 (cinquenta) processos retornou melhores resultados de acurácia para algumas classes. Ou seja, os resultados em média se mantiveram acima de 80% de acurácia, porém alguns resultados individuais melhoraram. Ainda assim, os resultados não estavam aceitáveis, pois há um desbalanço na qualidade do desempenho.

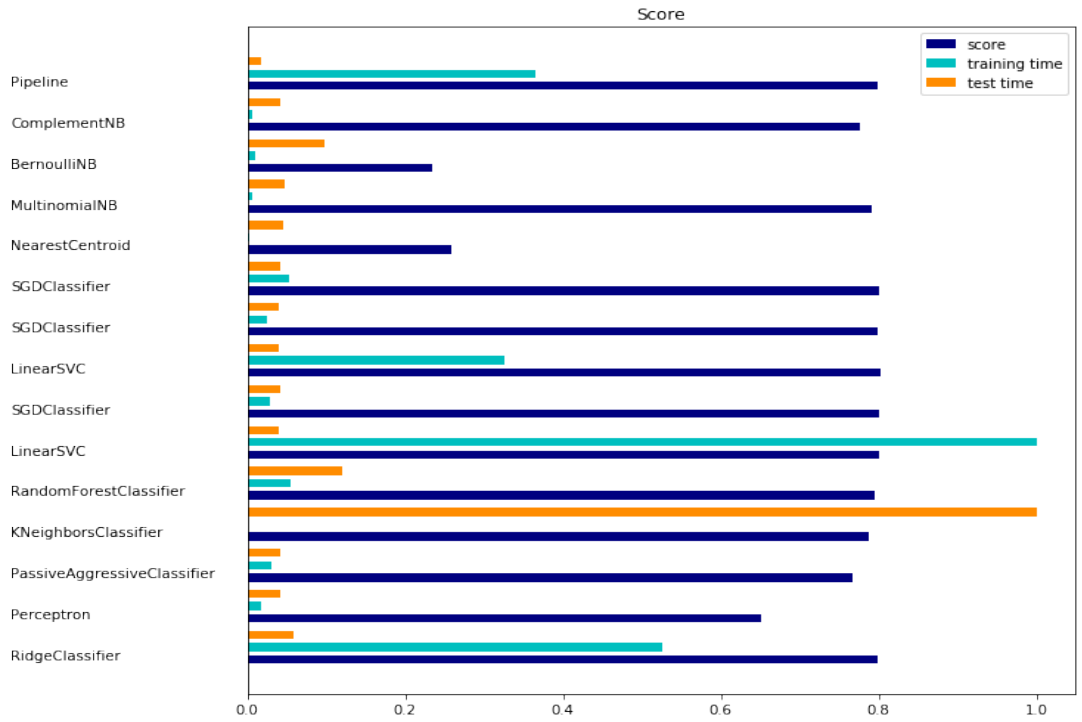

Figura 5.1: Resultado da execução dos 15 modelos de classificação com todos os documentos coletados do sistema de informação da PGDF, porém com um nível de classe.

[1] `http://www.pg.df.gov.br/wp-conteudo/uploads/2020/06/SEI_GDF-40263442-Nota-T%C3%A9cnica.pdf`

[2] `http://www.pg.df.gov.br/inteligenciaartificial/`

Tabela 5.1: Métricas de desempenho do Linear SVC para o primeiro resultado da execução dos 15 modelos de classificação com todos os documentos coletados do sistema de informação da PGDF.

Linear SVC			
Classe/Assunto	**precision**	**recall**	**f1-score**
A Definir	0.72	0.92	0.81
AMBIENTAL	0.00	0.00	0.00
Concurso Público	0.00	0.00	0.00
Contratos Administrativos	0.00	0.00	0.00
Direito Administrativo e Outras Matérias do Direito Público	0.80	0.99	0.89
Direito Civil	0.00	0.00	0.00
Direito Eleitoral	0.00	0.00	0.00
Direito Eleitoral e Processo Eleitoral do STF	0.00	0.00	0.00
Direito Previdenciário	0.00	0.00	0.00
Direito Processual Civil e do Trabalho	0.50	0.04	0.07
Direito Tributário	0.67	0.03	0.05
Direito do Consumidor	0.00	0.00	0.00
Direito do Trabalho	0.90	0.52	0.66
Dívida Ativa não	0.00	0.00	0.00
Empregado Público	0.00	0.00	0.00
Garantias Constitucionais	0.00	0.00	0.00
Intervenção do Estado na Propriedade	0.00	0.00	0.00
Intervenção no Domínio Econômico	0.00	0.00	0.00
Licitações	0.00	0.00	0.00
Meio Ambiente	0.00	0.00	0.00
Organização Político	0.00	0.00	0.00
Organização Sindical	0.00	0.00	0.00
Remédios Constitucionais	0.00	0.00	0.00
accuracy: 0.802			

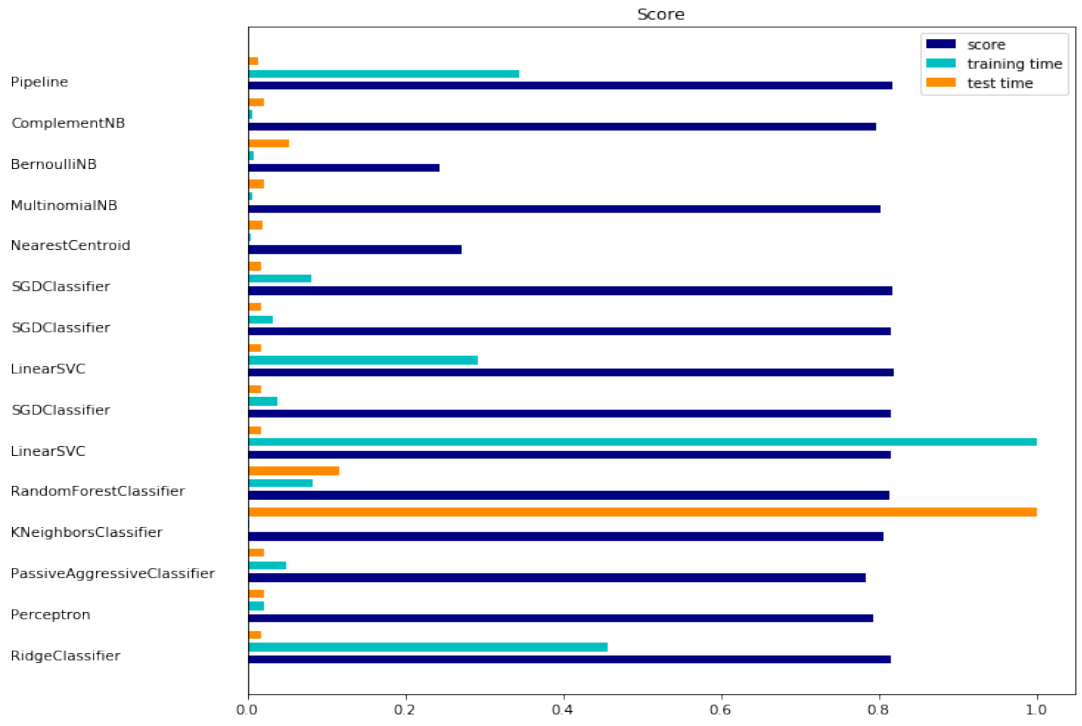

Figura 5.2: 15 modelos de classificação com todos os documentos coletados do sistema de informação da PGDF, porém com um nível de classe e mais de 50 processos por classe.

Mesmo com a melhoria no processo de classificação, limitando o número de processos e assuntos, o resultado ainda não estava adequado para uma implementação significativa

Tabela 5.2: Modelo Perceptron com dados segmentado em 50 processos por classe no mínimo

Perceptron			
Classe/Assunto	**precision**	**recall**	**f1-score**
Concurso Público	0.50	0.60	0.55
Contratos Administrativos	0.27	0.27	0.27
Direito Administrativo e Outras Matérias do Direito Público	0.91	0.94	0.93
Direito Civil	0.08	0.06	0.07
Direito Processual Civil e do Trabalho	0.19	0.10	0.13
Direito Tributário	0.74	0.54	0.62
ireito do Trabalho	0.00	0.00	0.00
accuracy: 0.868			

Tabela 5.3: Matriz de confusão do modelo Perceptron com dados segmentado em 50 processos por classe

confusion matrix:						
9	0	6	0	0	0	0
1	3	7	0	0	0	0
8	8	975	11	16	7	8
0	0	15	1	1	0	0
0	0	36	1	4	0	0
0	0	17	0	0	20	0
0	0	12	0	0	0	0

para a PGDF, que espera assertividade e equilíbrio.

A situação de algums assuntos bem classificadas e outras não tão bem classificadas não é tão eficiente. Houve uma melhoria comparando com os experimentos anteriores, mas não se mostra aceitável devido ao desbalanceamento, pois o F_1-*score* macro médio não passou de 10% nesses experimentos realizados até aqui. Importante lembrar que as assuntos no sistema de informação da Procuradoria Geral do Distrito Federal (PGDF) tem 8 (oito) níveis de classificação e passou-se a utilizar apenas o primeiro nível de classificação.

5.2 O processamento com redução do uso de documentos por processo - entendimento do negócio e preparação dos dados adicionais

Em conversa com os especialistas no negócio ficou evidente que alguns documentos do processo são apenas formalidades e registros do tramite processual. Com os resultados ainda inaceitáveis seguiu-se com uma exploração e experimentação utilizando apenas um documento significativo por processo. Verificou-se que a petição inicial e sentença são documentos que contém elementos mais significativos para identificar qual classe/assunto o processo está, pois contém maiores elementos relacionadas a matéria processual ou assunto do processo.

Para realizar a coleta de apenas um documento do processo no sistema de informação da PGDF exigiria a identificação de um documento dentro daqueles retirados do sistema no banco de dados. Quando as coletas iniciais foram feitas os documentos não foram

identificados, sendo que os processos foram coletados completamente. Seguiu-se para coletar as petições iniciais do sistema de informação da PGDF, contudo diante da uma pequena quantidade de petições iniciais identificadas no sistema se buscou outra fonte de informação, o MNI do TJDFT.

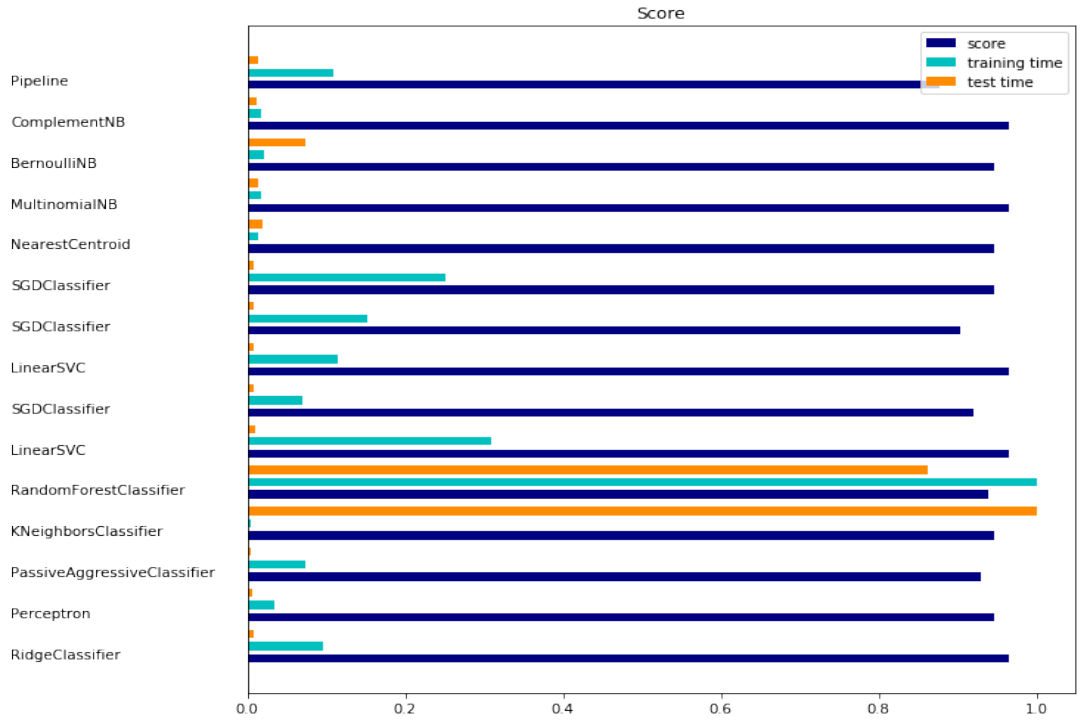

Figura 5.3: Classificação de apenas petições iniciais

Tabela 5.4: Classificação binária com Stochastic Gradient Descendent

Stochastic Gradient Descendent				
Classe/Assunto	**precision**	**recall**	**f1-score**	**support**
Direito Administrativo e Outras Matérias do Direito Público	0.9	0.99	0.96	1044
outras	0.87	0.34	0.49	137
accuracy: 0.918				

Tabela 5.5: Matriz de confusão do Stochastic Gradient Descendent da Tabela 5.4

confusion matrix:	
1037	7
90	47

Após todo o processo de coleta dos documentos e pré-processamento repetido nessa nova base, ou seja mais outro script foi criado para remover pontuação, remover acentos, transformar letras maiúsculas para minúsculo, remover números, retirar espaços em branco desnecessários, retirar quebras de linhas em excesso e fazer o stemming. O processo de pré-processamento foi finalizado gravando um arquivo csv, que passa a ser um segundo *corpus* a ser processado.

Os resultados, veja Figura 5.3, com a nova massa de dados apenas diminuiu o tempo de treinamento. Mas, todo o problema na classificação permaneceu. Sendo assim, passou-se a dividir as classes entre a mais significativa e outras para tentar melhorar o desempenho no quesito acertos classificatórios. Quando se observa as Tabelas 5.4 e 5.5 é possível apreender uma melhoria. Porém, ainda classificando de modo insatisfatório a classe denominada Outras se observada a métrica F_1-*score*. A próxima seção explica a evolução dessa classificação binária.

5.3 Outras modelagens

5.3.1 O uso da classificação binária

Nesse âmbito, seguiu-se com experimentos dividindo os dados em duas classes, uma predominante e uma classe Outras, que agrupa as demais classes. Ou seja, novas classificações binárias foram realizadas, colocando em uma classe Outras aquelas classes que tiveram pior resultado em experimentos anteriores. O processo foi feito repetidas vezes como pode ser observado a seguir.

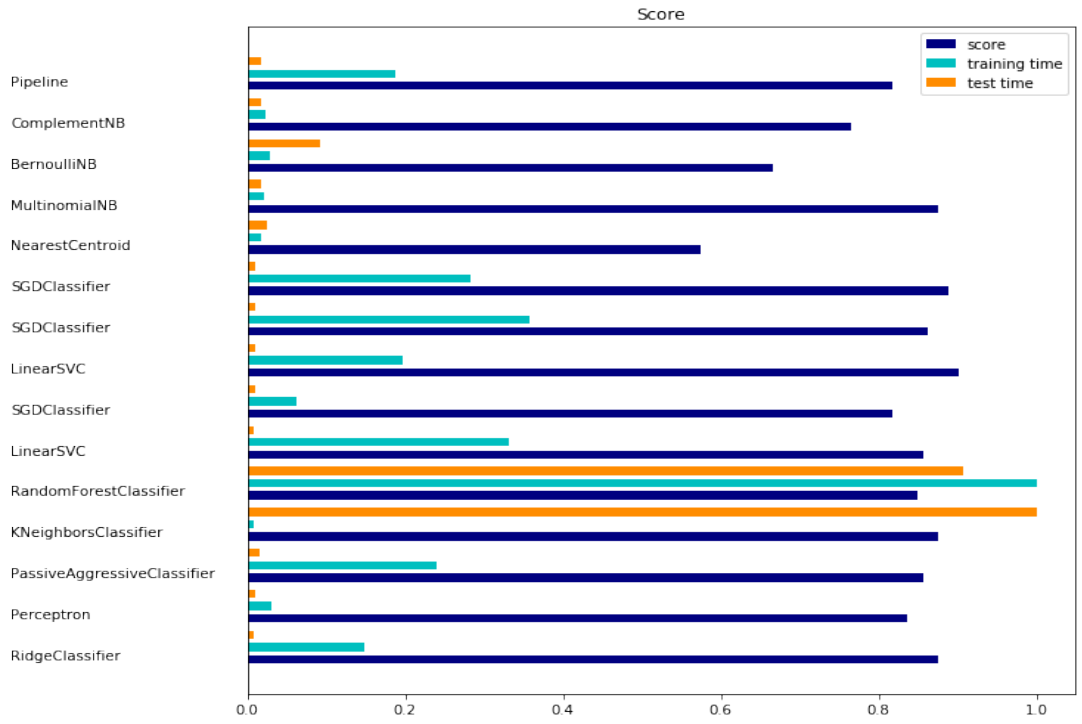

Figura 5.4: Classificação de Direito Tributário e Outras

Como dito, uma segunda classificação binária foi realizada com a classe Direito Tributário e a classe Outras, já que a primeira está na seção anterior com a classe Direito Administrativo Outras Matérias do Direito Público, e o resultado desta segunda pode ser

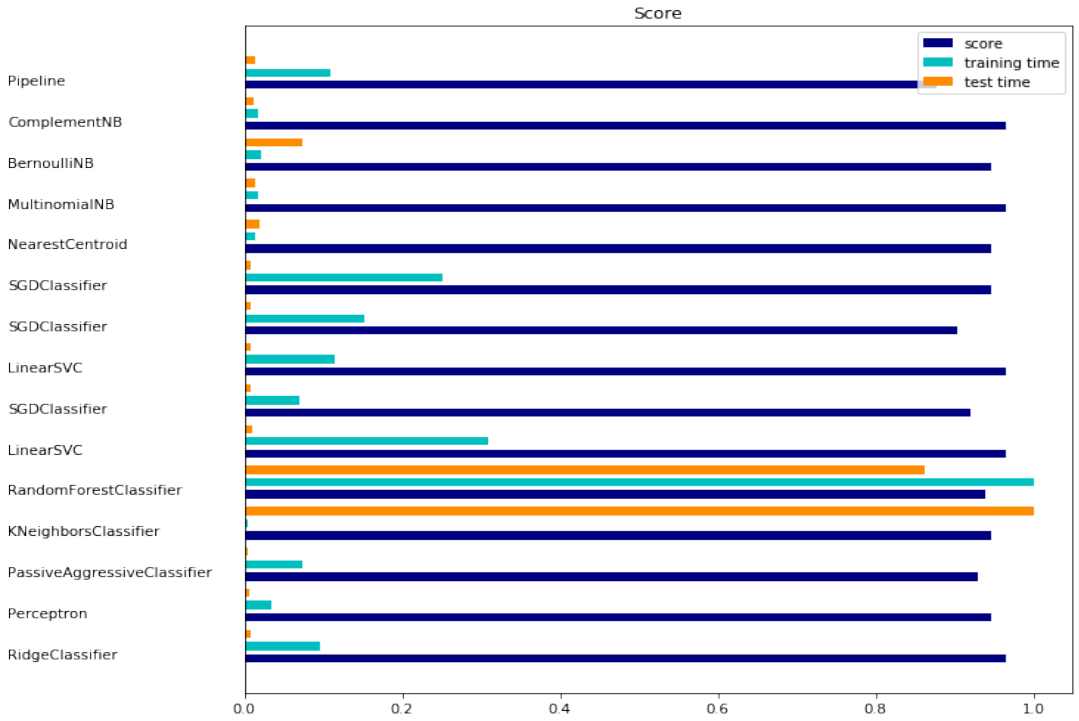

Figura 5.5: Classificação de Concurso e Outras

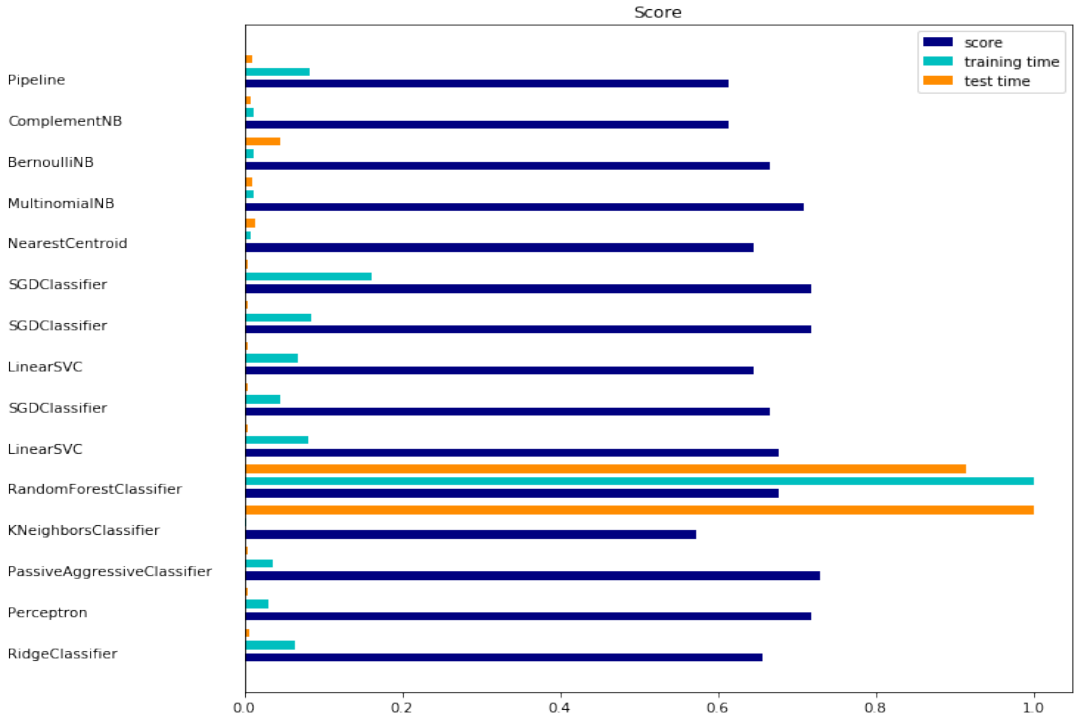

Figura 5.6: Classificação de Direito Processual Civil e do Trabalho e Outras

observado na Figura 5.4. O interessante é que as métricas retornaram melhores valores, como pode ser confirmado nas Tabelas 5.6 e 5.7 pois, a classificação binária retornou um

F_1-*score* maior que 78% para as duas classes.

Essa etapa não se mostrou tão eficiente como a classificação binária anterior, não obstante apresenta um equilíbrio de classificação aceitável para o momento. Em seguida, a classe\assunto Outras foi submetida a mais uma execução de classificação com o auxílio do computador.

Evoluindo, uma terceira classificação binária foi realizada, com a classe Concurso e a classe Outras, e o resultado pode ser observado na Figura 5.5. As Tabelas 5.8 e 5.9 apresentam um resultado satisfatória para F_1-*score*, inclusive melhor que a classificação binária anterior. Com isso, uma quarta classificação binária com a classe Direito Processual Civil e do Trabalho e a classe Outras, final, foi realizada e o resultado pode ser observado na Figura 5.6. O resultado continua se mostrando interessante se observar as Tabelas 5.10 e 5.11.

O resultado até aqui apresenta um equilíbrio de classificação aceitável para o momento. A Tabela 5.12 apresenta a média de acurácia das classificações binárias. Importante notar que o F_1-*score* macro médio acompanhou a acurácia nesses experimentos, variando para mais e para menos da acurácia em 5% a 10% dependendo do algoritmo.

Tabela 5.6: Linear Support Vector Classification nos assuntos Classificação de Direito Tributário e outros.

Linear Support Vector Classification				
assuntos	**precision**	**recall**	**f1-score**	**support**
Direito Tributário	1.00	0.63	0.78	41
Outras	0.88	1.00	0.94	112
accuracy: 0.843				

Tabela 5.7: Matriz de confusão do Linear Support Vector Classification da Tabela 5.6

confusion matrix:	
26	15
0	112

Tabela 5.8: Linear Support Vector Classification nos assuntos Classificação de Concurso Público e outros.

Linear Support Vector Classification				
assuntos	**precision**	**recall**	**f1-score**	**support**
Concurso Público	0.92	0.79	0.85	14
Outras	0.97	0.99	0.98	100
accuracy: 0.843				

Tabela 5.9: Matriz de confusão do Linear Support Vector Classification da Tabela 5.8

confusion matrix:	
11	1
3	99

Tabela 5.10: Modelo Perceptron no assunto Direito Processual Civil e do Trabalho e Outras

Perceptron				
Classe/Assunto	**precision**	**recall**	**f1-score**	**support**
Direito Processual Civil e do Trabalho	0.62	0.82	0.71	40
Outras	0.84	0.64	0.73	56
accuracy: 0.719				

Tabela 5.11: Matriz de confusão do Perceptron da Tabela 5.10

confusion matrix:	
33	7
20	36

Tabela 5.12: Média de acurácia com os melhores algoritmos e suas respectivas classes, a serem executados em cascata.

Algoritmo	**assuntos**	**Score**
Stochastic Gradient Descendent	Direito Administrativo e Outras Matérias do Direito Público	0.96
SVC	Direito Tributário	0.78
SVC	Concurso Público	0.85
Perceptron	Direito Processual Civil e do Trabalho	0.719
Média de acurária: 0.825		

5.3.2 O uso de *deep learning*

Ainda na intenção de experimentar e identificar o melhor modelo para classificar os dados disponíveis buscou-se executar modelos de *deep learning*, que tem sido muito utilizado em classificação de documentos jurídicos, segundo a revisão de literatura realizada.

A Tabela 5.13 apresenta o resultado de uma classificação multi classe com as redes neurais em *deep learnig*. Pode ser observado que o resultado foi a quem do esperado e o modelo se concentrou em classificar apenas a classe predominante Direito Administrativo e Outras Matérias do Direito Público e para as demais classes não houve assertividade nenhuma. Adiante estão as tentativas com a binarização dos dados e classificação.

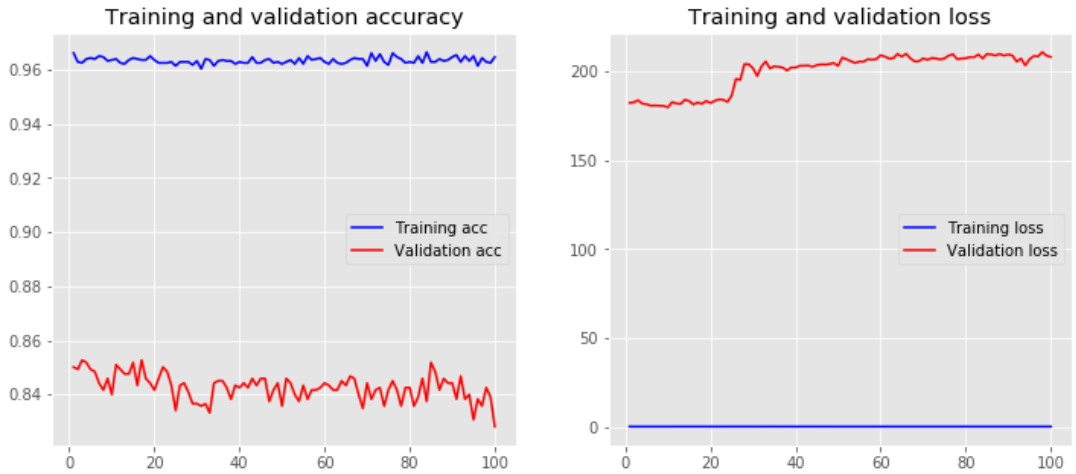

Figura 5.7: Validação e perda com *deep learning* de Direito administrativo e Outras matérias do direito público e Outras.

Tabela 5.13: Score *deep learning* com todas as classes

	precision	recall	f1-score
0	0.00	0.00	0.00
1	0.00	0.00	0.00
2	0.87	0.99	0.93
3	0.00	0.00	0.00
4	0.00	0.00	0.00
5	0.00	0.00	0.00
6	0.00	0.00	0.00
7	0.00	0.00	0.00
8	0.00	0.00	0.00
9	0.00	0.00	0.00
...	0.00	0.00	0.00
accuracy			0.86
macro avg	0.09	0.10	0.09
weighted avg	0.76	0.86	0.81

Tabela 5.14: Score *deep learning* de Direito administrativo e Outras matérias do direito público e Outras.

deep learning	
Training Accuracy:	0.9644
Testing Accuracy:	0.8281

Tabela 5.15: Score *deep learning* de Direito tributário e Outras.

deep learning	
Training Accuracy:	0.9719
Testing Accuracy:	0.8235

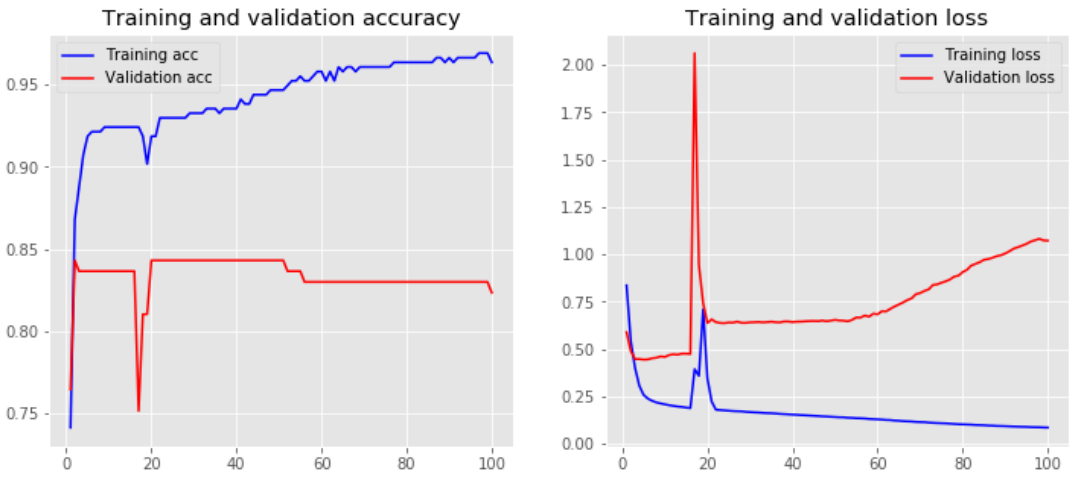

Figura 5.8: Validação e perda com *deep learning* de Direito tributário e Outras.

A Tabela 5.14 apresenta a acurácia obtida com o uso do modelo de *deep learning* treinado, validado e testado de modo binário. As classes utilizadas foram Direito admi-

Tabela 5.16: Score *deep learning* de Concurso e Outras.

deep learning	
Training Accuracy:	0.9848
Testing Accuracy:	0.9211

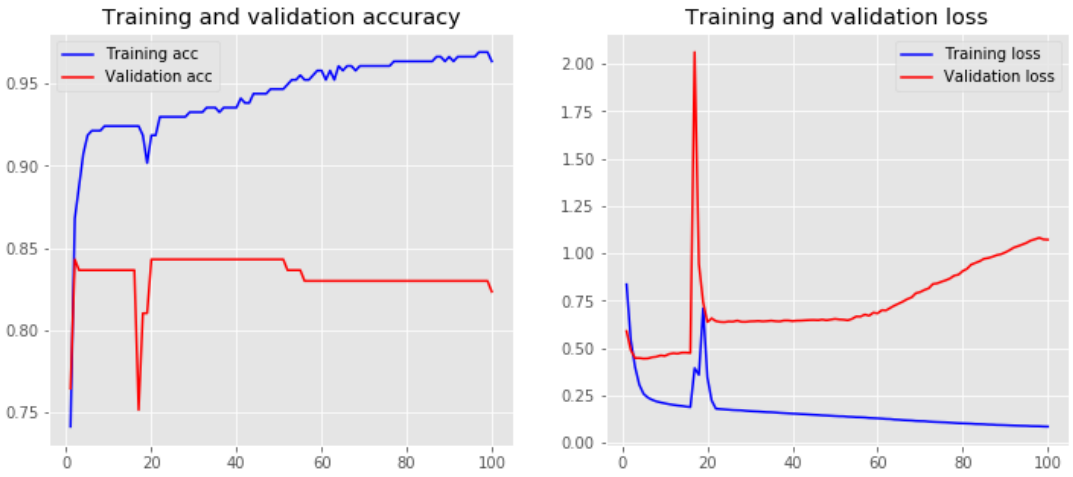

Figura 5.9: Validação e perda com *deep learning* de Concurso e Outras.

Tabela 5.17: Score *deep learning* de Direito Processual Civil e do Trabalho e Outras.

deep learning	
Training Accuracy:	0.9554
Testing Accuracy:	0.6667

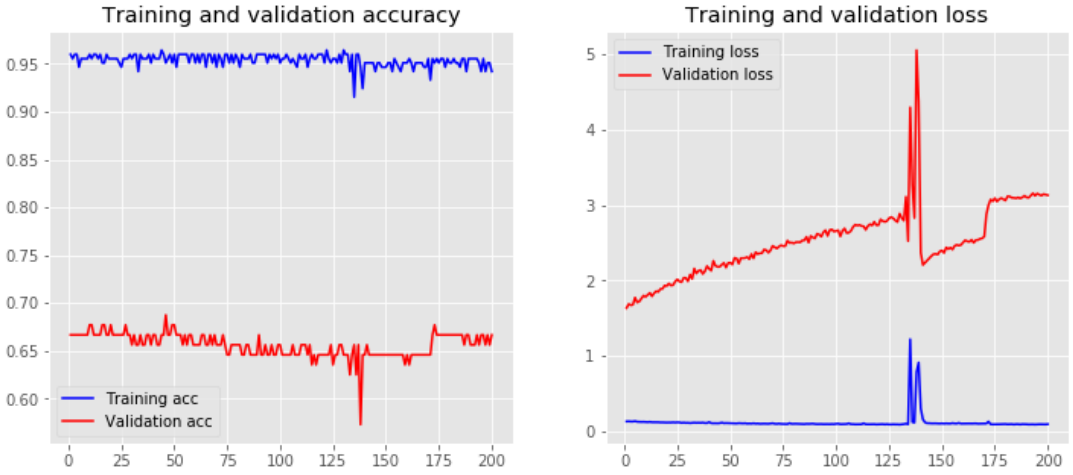

Figura 5.10: Validação e perda com *deep learning* de Direito Processual Civil e do Trabalho e Outras.

nistrativo e Outras matérias do direito público e a classe Outras. A Figura 5.7 apresenta o desempenho do treinamento, validação e teste no tempo e em função das épocas da rede neural.

Por sua vez, em uma segunda execução para a classe Direito tributário e a classe Outras teve os resultados expressos na Tabela 5.15. Esta tabela apresenta mais uma vez

Tabela 5.18: Média de acurácia com *deep learning* e suas respectivas classes, a serem executados em cascata.

***deep learning* assuntos**	**Score**
Direito Administrativo e Outras Matérias do Direito Público	0.8281
Direito Tributário	0.8235
Concurso Público	0.9211
Direito Processual Civil e do Trabalho	0.6667
Média de acurária: 0.80985	

a acurácia obtida com o uso do modelo de *deep learning* treinado, validado e testado de modo binário. A Figura 5.8, por sua vez, apresenta o desempenho do treinamento, validação e teste no tempo e em função das épocas da rede neural.

Na terceira execução do modelo para a classe Concurso e a classe Outras tem-se os resultados da Tabela 5.16 com acurácia, da mesma forma, obtida após o uso do modelo de *deep learning* treinado, validado e testado de modo binário. As classes Concurso Público e a classe Outras foram utilizadas. A Figura 5.9, por último, apresenta o desempenho do treinamento, validação e teste no tempo e em função das épocas da rede neural.

Nesse processo repetitivo, seguiu-se para a ultima execução com as classes Direito Processual Civil e do Trabalho e a classe Outras. A Tabela 5.17 apresenta a acurácia obtida com o uso do modelo de *deep learning* treinado, validado e testado de modo binário. A Figura 5.10 apresenta o desempenho do treinamento, validação e teste no tempo e em função das épocas da rede neural.

Diante desse melhor desempenho com a binarização as etapas seguintes irão realizar uma classificação em cascata ou *ensemble learning* utilizando os algoritmos apresentados anteriormente. Para isso um programa de computador único deve ser desenvolvido com os modelos treinados e validados. A expectativa é de se construir um modelo de classificação com o auxílio do computador que obtenha um score médio de no mínimo 82,5%, assim como demonstra a Tabela 5.12. Outros experimentos com *deep learning* serão apresentados no próximo capítulo.

5.3.3 O uso do One-versus-All (OVA)

Com a expectativa de utilização da binarização com a técnica One-versus-All (OVA) para incremento do desempenho seguiu-se para os novos experimentos. Também foi pensado na utilização de técnicas de amostragem em bases desbalanceadas. Aquelas 5(cinco) classes predominantes, exemplo Tabela 5.18, foram separadas da base de dados e as demais foram agregadas em uma classe chamada **Outras**. Portanto, se tem as seguintes classes: Direito Administrativo e Outras Matérias do Direito Público, Direito Tributário, Concurso Público, Direito Processual Civil e do Trabalho e Outras.

Inicialmente se mostra vantajoso analisar os resultados dos métodos de amostragem para bases de dados desbalanceados pois, trata-se de uma fase de pré-processamento e na ordem lógica de execução é a fase que precede o carregamento dos dados na memória do computador para execução dos modelos.

Observando a Figura 5.11 pode-se perceber um desbalanceamento na distribuição dos dados. O eixo horizontal (X) nesta figura representa cada uma das classes. Por motivos de exigências nas implementações as classes foram convertidas em números aqui, sendo: 0 = Direito Administrativo e Outras Matérias do Direito Público, 1 = Direito Tributário, 2 = Direito Processual Civil e do Trabalho, 3 = Direito Tributário e 4 = Outras.

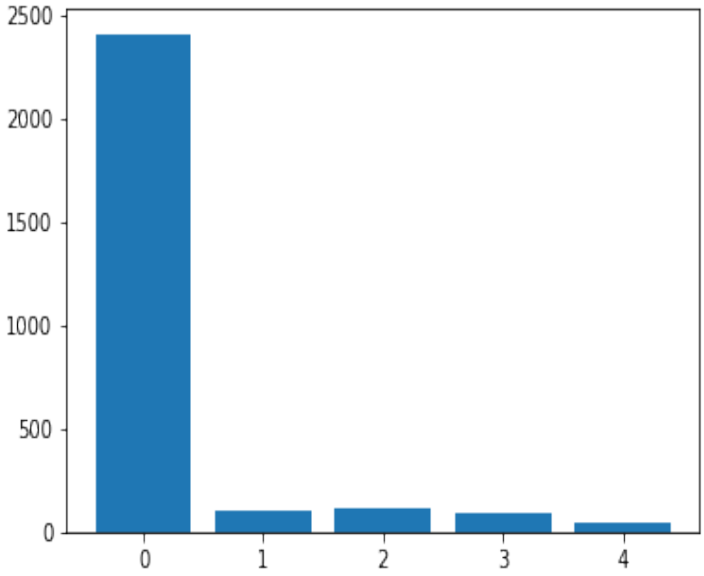

Figura 5.11: Dados desbalanceados

A primeira técnica de amostragem em bases desbalanceadas aplicada nos dados de petições iniciais foi o Synthetic Minority Oversampling Technique (SMOTE). Pode-se observar na Figura 5.12 que a transformação na base levou a um crescimento daquelas classes minoritárias até aproximadamente 2500(dois mil e quinhentas) observações. Isso foi feito com amostras sintéticas adicionadas a base de dados.

A segunda técnica implementada em bases desbalanceadas aplicada nos dados de petições iniciais foi o Random Under-Sampling (RUS). Pode-se observar na Figura 5.13 que a transformação na base levou a um decréscimo daquelas classes majoritárias até aproximadamente 40(quarenta) observações.

A terceira técnica utilizada em bases desbalanceadas aplicada nos dados de petições iniciais de precatório foi o SMOTE and Edited Nearest Neighbors (SMOTEENN). Pode-se observar na Figura 5.14 que a transformação na base levou a um decréscimo daquelas classes majoritárias até aproximadamente 1500(um mil e quinhentas) observações e um

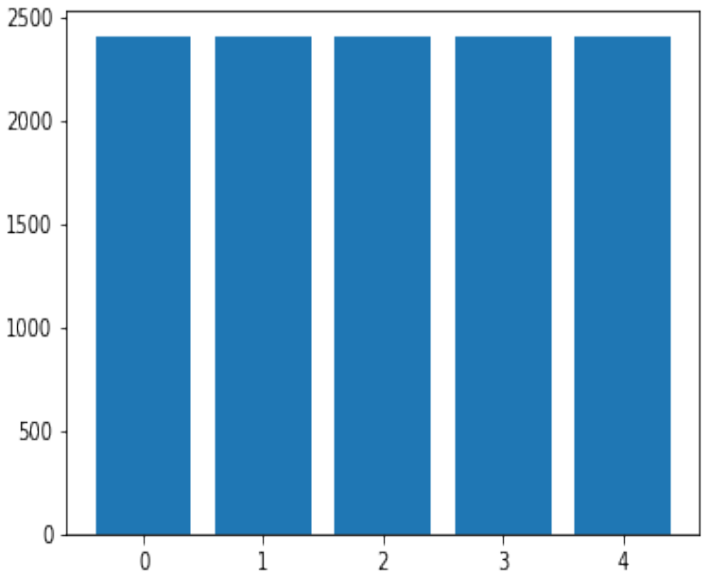

Figura 5.12: Aplicação da técnica Synthetic Minority Oversampling Technique (SMOTE)

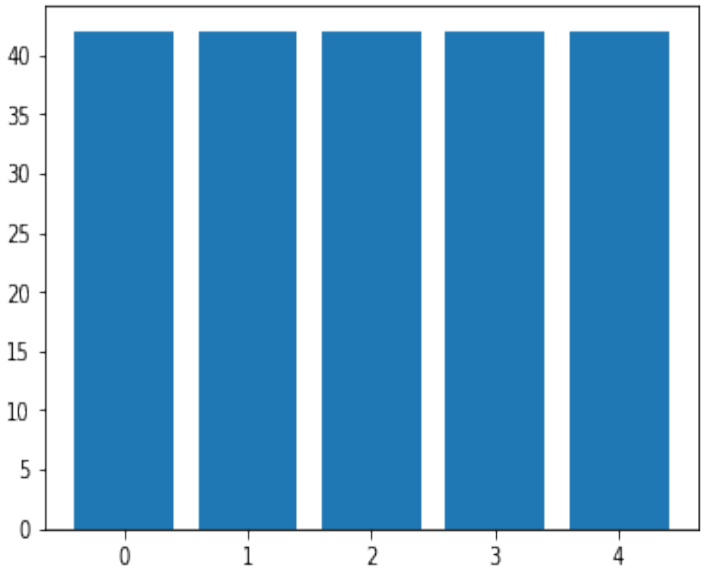

Figura 5.13: Aplicação da técnica Random Under-Sampling (RUS)

acréscimo com dados sintéticos nas demais classes, caracterizando a aplicação de um método híbrido.

Na posse dos dados balanceados uma segunda bateria de experimentos foi realizada. Assim, os resultados estão na Tabela 5.19 Os experimentos iniciais, capítulo anterior, não se levou em consideração a combinação de técnicas ou algoritmos e não se usou o K-Fold *cross-validation*. Nesta segunda bateria de experimentos foram combinadas várias técnicas e algoritmos para verificação de hipóteses de pesquisa.

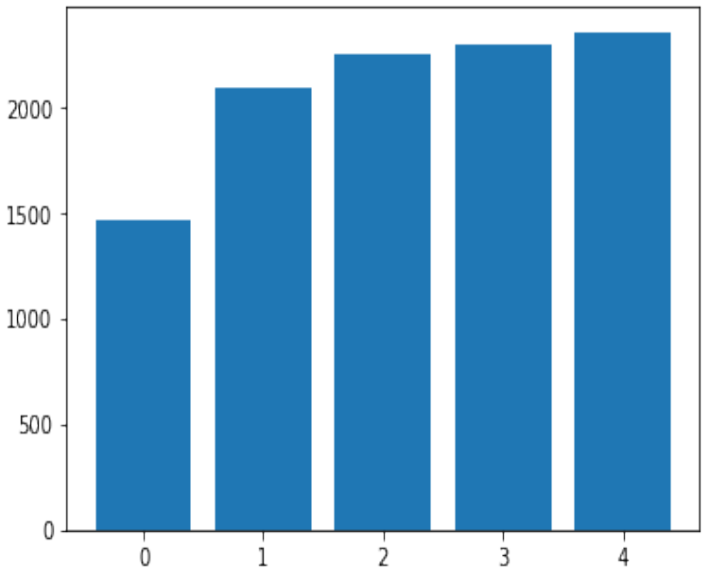

Figura 5.14: Aplicação da técnica SMOTE and Edited Nearest Neighbors (SMOTEENN)

Tabela 5.19: Resultados do desempenho dos modelos baseado em F_1-Score macro médio

Modelos combinando *GridSearchCV, OneVsRestClassifier* e *10-Fold cross-validation*						
Inbalance X Classifier	Logistic Regression	SVC	K Neighbors Classifier	Decision Tree Classifier	Random Forest Classifier	Gradient Boosting Classifier
SMOTE	0.28	0.40	0.30	0.38	0.37	0.40
RUS	0.28	0.40	0.30	0.36	0.36	0.40
SMOTEENN	0.28	0.40	0.30	0.35	0.37	0.40

Tabela 5.20: Resultados do desempenho do *deep learning* em F_1-Score macro médio

Deep learning(tensorFlow e keras)	
Desbalanceada	0.19
SMOTEENN	0.44
RUS	0.39
SMOTE	0.28

Outro experimento que merece registro foi a aplicação de redes neurais com *deep learning* nesses mesmos dados. Como pode ser visto na Tabela 5.20 o modelo de *deep learning* sem binarização, ou seja com múltiplas classes, não apresentou resultados expressivamente superiores aos modelos anteriores.

Como pode ser visto na Tabela 5.19 o desempenho dos algoritmos não foi como o esperado. Importante observar que a técnica de cascateamento ou ensemble foi aplicada, no formato One-versus-All (OVA), porém, sem sucesso. Além disso, foi utilizado uma técnica de busca por força bruta de parâmetros dos algoritmos com *GridSearchCV* do *Scikit*

Learn e o *10-Fold cross-validation*. A última técnica não foi aplicada nos modelos com binarizações isoladas do capítulo anterior, o que nos leva a pensar que estava ocorrendo um fenômeno de sobre-ajuste - *overfiting*.

Com isso, a precisão não passou de 45% de F_1-*score* macro médio, quando da aplicação da combinação de várias técnicas. Outros experimentos foram feitos, mas não houve melhoria no desempenho significativo. Por isso, não se considerou fazer registros. Restava uma comparação estatística. Mas, como o resultado foi muito a quem não se considerou válido maiores implementações. Levando em consideração que a meta seria 82,5% restou se realizar novas entrevistas com os especialistas de negócio. Chegando ao entendimento que o problema estava mais inerente aos dados. Porém, as ações repetitivas, como essas de precatório, teriam dados melhores. Maiores explicações e evoluções estarão no próximo capítulo.

5.3.4 Algoritmos não supervisionados

Para finalizar a apresentação dos trabalhos com os dados de precatórios, esta seção apresenta o esforço dado em se verificar um agrupamento melhor para os dados coletados até o momento. A intenção seria verificar a distribuição teoricamente ideal das classes diante dos dados coletados. Pois, vimos que as classificações não têm um bom desempenho se os rótulos não estão adequadamente feitos para um treinamento satisfatório.

Mais adiante se poderá observar que há muita sobre-posição de classes e que esse método pode ser usado com o auxílio de especialistas para reclassificar os dados. Vistos que, como será discutido no próximo capítulo, descobriu-se que as rotulagens não estavam adequadas e que a classe Direito Administrativo e Outras Matérias de Direito Público era usada como uma classe genérica, onde todos os processos de precatórios estavam sendo direcionados.

Pensando nessa distribuição teórica, iniciou-se utilizando o algoritmo Latent Dirichlet Allocation (LDA) em uma investigação do número de clusters ótimo. Com a métrica de coerência dos tópicos verificou-se o número mais adequado para a quantidade de grupos/clusters. A Figura 5.15, a seguir, demonstra que o número de coerência máxima para os tópicos seria 8 (oito). Importante recordar que foram utilizadas três implementações de Latent Dirichlet Allocation (LDA) na linguagem Python, assim como já discutido no capítulo metodologia.

Destarte, executou-se o algoritmo Latent Dirichlet Allocation (LDA) com 8 (oito) clusters e o resultado de sua vizualização demostra que existem muitas intersecções de palavras. A Figura 5.16, vizualização do Latent Dirichlet Allocation (LDA) com 8 (oito) clusters, permite observar que se for seguida uma linha de raciocínio de eliminar as inter-

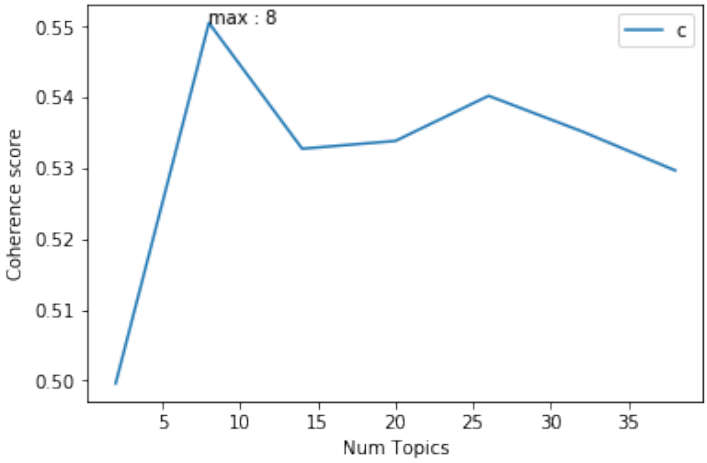

Figura 5.15: Número máximo de coerência para definir a clusterização ótima.

secções tem-se aproximadamente 7 (sete) clusters. Diante disso, executou-se do mesmo modelo com 7 (sete) clusters, o que pode ser observado na Figura 5.17.

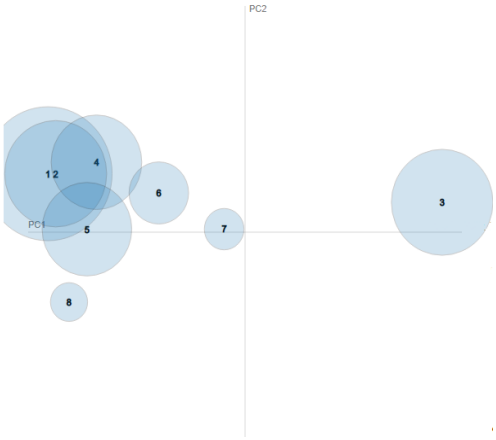

Figura 5.16: Número máximo de coerência para definir a clusterização ótima, 8 (oito) *clusters*.

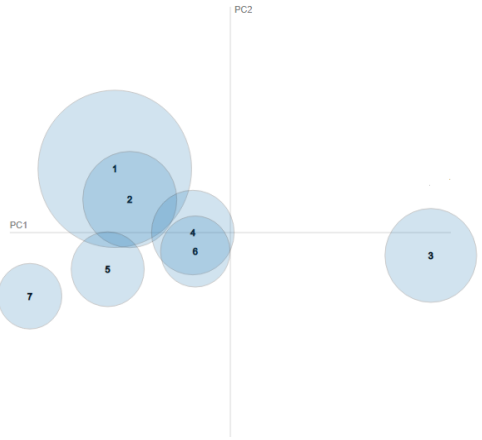

Figura 5.17: Redução do número de *clusters* para 7 (sete).

Ainda na tentativa de analisar a melhor distribuição dos documentos em tópicos dominantes executou-se o modelo LDA para 6 (seis) *clusters*, após observar que 7 (sete) *clusters* ainda apresentavam intersecções. A Figura 5.18 apresenta a distribuição dos documentos nos 6 (seis) *clusters*. Nesta visualização as distribuições possuem menos intersecção, mas ainda pode-se verificar sobreposições. Por fim, foi feito a mesma execução com 5 (cinco) *clusters*, que mesmo não sendo conclusiva já não traz a sobreposição de tópicos, 5.19.

Além disso, a exemplo de Vu Bui et. al. [77] utiliza-se os algoritmos Latent Dirichlet Allocation (LDA) combinado com o Kmeans. Assim sendo, os resultados da execução da combinação do algoritmo Kmeans e Latent Dirichlet Allocation (LDA) por meio das vizualizações, na Figura 5.21, verificou-se que o cluster 8º(oitavo) não pode ser percebido

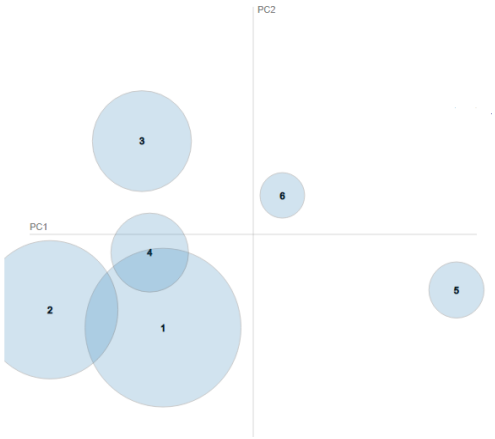

Figura 5.18: Redução do número de *clusters* para 6 (seis).

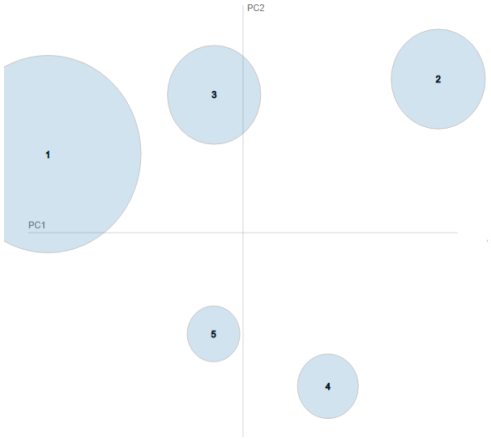

Figura 5.19: Redução do número de *clusters* para 5 (cinco).

com facilidade. Diferente da nitidez da vizualização 5.20 que já separa melhor os grupos de tópicos. Interessante notar que pode-se observar que se aumentarmos o número de clusters a sobreposição ou confusão visual aumenta, como consta nas Figuras 5.22 e 5.23

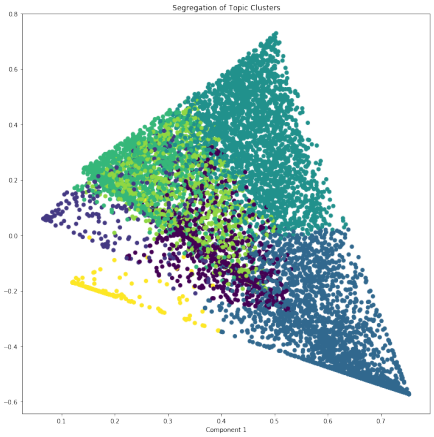

Figura 5.20: Aplicação do LDA utilizando 7 (sete) clusters.

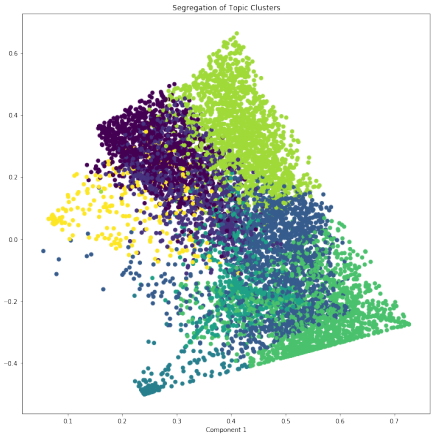

Figura 5.21: Aplicação do LDA utilizando 8 (oito) clusters.

Nessa lógica, analisando as visualizações pode-se observar que 6 (seis) ou 5 (cinco) clusters estão expostos com mais clareza, nas Figuras 5.24 e 5.25. Elas demonstram uma divisão entre as cores dos *clusters* com maior nitidez comparadas as visualizações anteriormente apresentadas. Importante lembrar que quando utiliza-se 5 (cinco) *clusters* tem-se menos intersecções, Figura 5.19.

Esse primeiro esforço implementando modelos de aprendizagem de máquina não supervisionados não gerou achados tão conclusivos, pois necessitaria de mais tempo com os especialistas para poder reclassificar grande parte da base de dados. Vai-se utilizar esses achados nas discussões e trabalhos futuros.

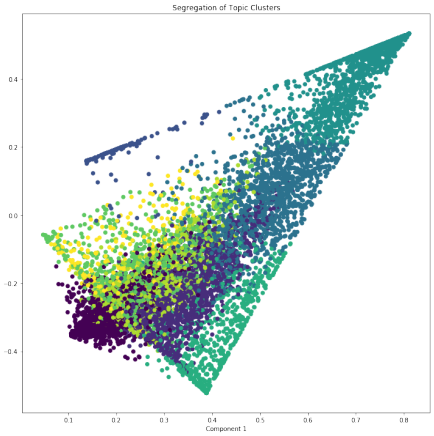

Figura 5.22: Aplicação do LDA utilizando 9 (nove) clusters.

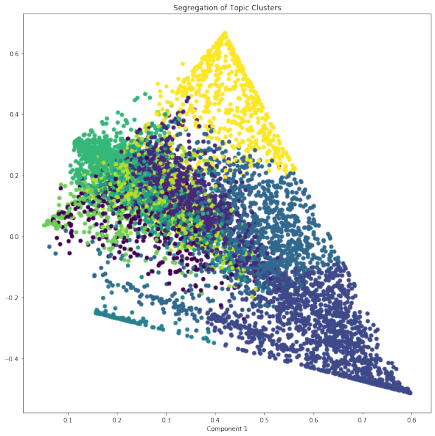

Figura 5.23: Aplicação do LDA utilizando 10 (dez) clusters.

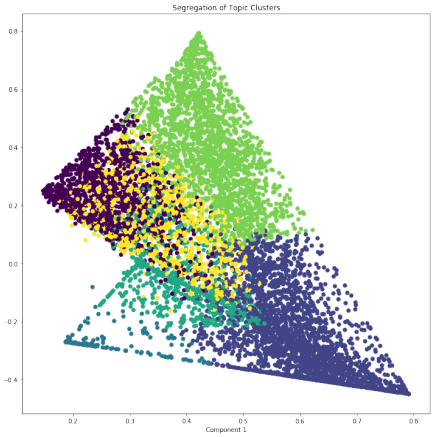

Figura 5.24: Aplicação do LDA utilizando 6 (seis) clusters.

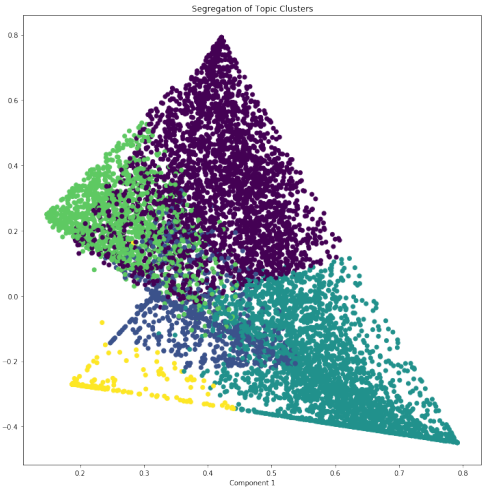

Figura 5.25: Aplicação do LDA utilizando 5 (cinco) clusters.

Capítulo 6

Experimentos com ações repetitivas

Neste capítulo são apresentados os experimentos finais. Um retorno final ao entendimento do negócio foi necessário e um novo entendimento dos dados se fez indispensável. Além disso, apresenta a modelagem e resultados finais. Por fim, compara algoritmos com o uso de estatística.

6.1 Retorno aos especialistas de negócio - entendimento do negócio e preparação dos dados adicionais

Quando se chegou na aplicação de mais de 40(quarenta) *scripts* diferentes sem melhorar o desempenho dos modelos. Esses *scripts* com a aplicação de algoritmos diferentes, técnicas diversas de *ensemble learning* e amostragem em bases desbalanceadas. Levou-se em consideração que o problema estaria nos dados e por isso um novo retorno aos especialistas de negócio de PGDF foi necessário. Nessas novas entrevistas a ideia seria melhorar o entendimento do negócio e o entendimento dos dados, fazendo-se mais uma rodada do Cross Industry Standard Process for Data Mining (CRISP-DM) para evolução dos trabalhos.

Neste momento de retorno às entrevistas, soube-se de uma informação nova. O setor que rotulou esses dados foi iniciado pouco antes do início da pandemia de COVID-19 de 2020. Ou seja, cerca de 10% desses dados foram realmente conferidos com uma técnica de revisão dupla em equipes, sendo uma equipe em certos momentos realizando as classificações e a outra revisando. Essas funções de classificação e revisão vão sendo alternadas e daí se encaminha o processo judicial para a procuradoria especializada certificar.

Uma questão interessante é que não se tinha certeza de que os dados anteriores trabalhados neste estudo estivessem bem rotulados. Na verdade o depoimento foi de que a

classe Direito Administrativo e Outras Matérias do Direito Público era realmente uma classe genérica.

Importante lembrar dos agrupamentos que foram feitos com algoritmos não supervisionados, pois a realidade de agrupamento quando se observa as categorizações feitas é diferente da apresentada com a *cluterização*. Praticamente todos os dados estavam sendo colocados manualmente em uma só classes. Neste ponto vê-se que faz sentido realizar agrupamentos com algoritmos não supervisionados para posterior ajuste por parte dos especialista, em uma re-rotulagem assistida pelo computador. Questão que ficou para ser solucionada em trabalhos futuros.

Portanto, seguiu-se para investigar dados mais confiáveis no ponto de vista de negócio e técnico para poder ter mais garantias e desempenho satisfatório na execução dos modelos atendendo as expectativas da casa. Isso, exigiu mais tempo para que os especialistas de negócio pudessem entender o que seria um dado mais confiável.

Lembrado que Russel e Noving [6] afirmam, que alguns trabalhos recentes na IA sugerem que, para muitos problemas, faz mais sentido se preocupar com os dados e ser menos exigente sobre qual algoritmo aplicar. Assim, iniciou-se uma busca por uma quantidade de dados de causas repetitivas que tivessem a garantia de rótulos revisados e/ou certificados por um outro setor.

Tabela 6.1: Assuntos repetitivos que tem revisão e certificação por outras áreas não sendo a triagem

ASSUNTOS REVISADOS E CERTIFICADOS				
Palavra(s)-chave	Assunto	Código	Quantidade	Especializada para encaminhar
GATE / GAEE	Gratificações Por Atividades Específicas - Gratificação de Atividade de Ensino Especial	1.29.5.10	21896	PROPES
Medicamento	Saúde - Fornecimento de Medicamentos	1.28.3.5	3765	PROSAUDE
Creche	Ensino Fundamental e Médio - Creche	1.28.2.1.2	4753	PROCAD

Foi encaminhado um estudo com 11(onze) possibilidades de assuntos repetitivos que tivessem maior garantia de rótulos revisados e certificados. A Tabela 6.1 apresenta os 3(três) assuntos escolhidos arbitrariamente dentre os 11(onze). Neste caso os códigos dos processos foram retirados do sistema de informação e gestão de documentos da PGDF. As palavras chave da busca são a primeira coluna da Tabela 6.1. Nestes casos, duas equipes trabalham na classificação e revisão. Quando o processo chega na procuradoria

especializada, é certificado se realmente é daquele assunto encaminhado. Desse modo, acreditou-se que se teria melhores resultados.

Tabela 6.2: Dados adicionados por meio de consolidação

CONSOLIDADOS				
Palavra(s)-chave	Assunto	Código	Quantidade	Especializada para encaminhar
Precatório	Precatório	1265	3934	PROPREC
Execução Fiscal	Execução Fiscal	1116	9604	PGFAZ

Tabela 6.3: Assuntos repetitivos que tem revisão e certificação por outras áreas não sendo apenas na triagem

NOVA BASE DE DADOS		
Palavra(s)-chave	Quantidade de petições iniciais coletadas	Especializada para encaminhar
GATE / GAEE	9976	PROPES
Medicamento	2991	PROSAUDE
Creche	2731	PROCAD
Precatório	3934	PROPREC
Execução Fiscal	9604	PGFAZ

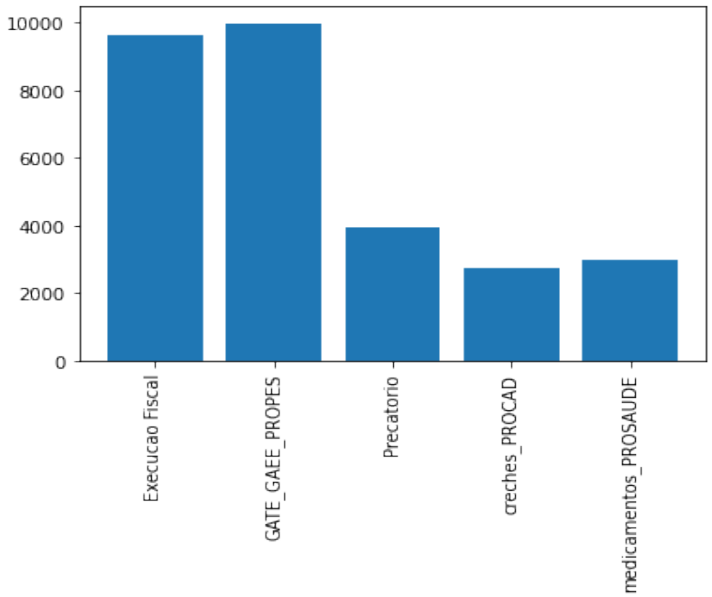

Figura 6.1: Nova base, ainda desbalanceada

Adicionalmente aos dados coletados, foi consolidada toda a base dos experimentos anteriores de precatório. Ademais, tendo em vista o projeto de inteligência artificial em

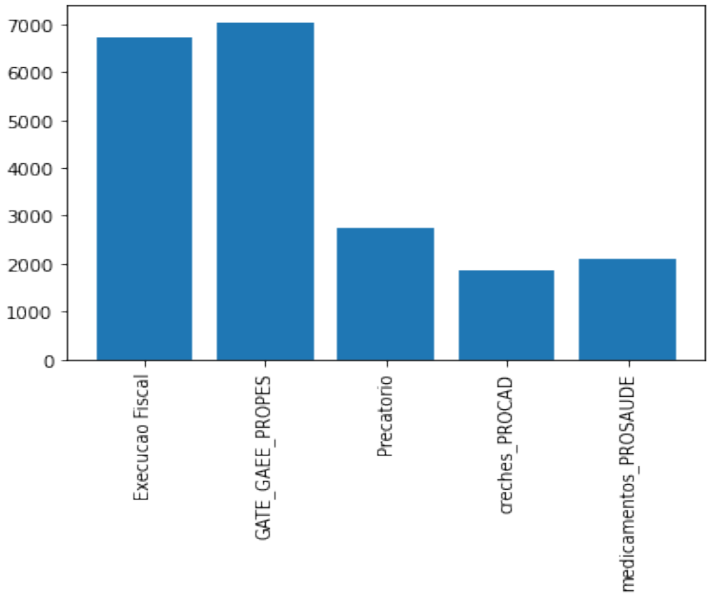

Figura 6.2: Nova base, dados de treinamento

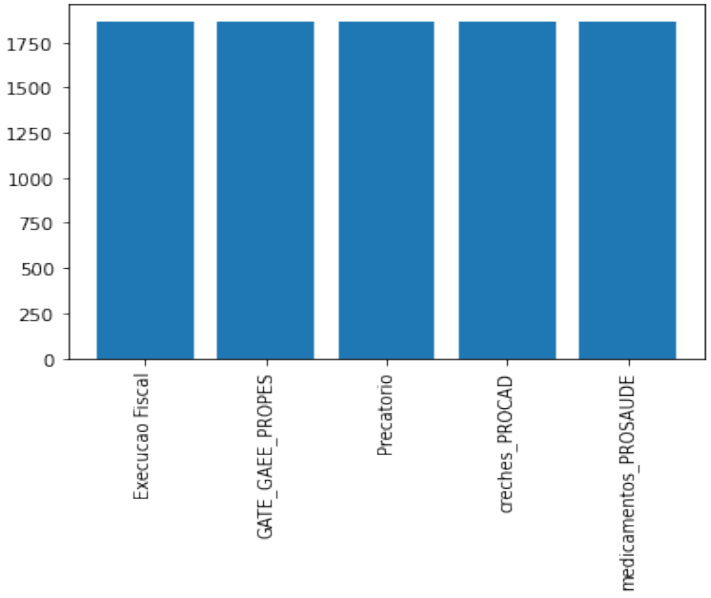

Figura 6.3: Aplicação de Random Under-Sampling (RUS) na nova base

Execução da PGDF[1] se coletou alguns dados neste tema. Chegando ao levantamento presente na Tabela 6.2.

A nova base de dados resultado das coletas de códigos do processo no sistema de informação e gestão de documentos da PGDF e, em seguida, das petições iniciais no sistema do Tribunal de Justiça do Distrito Federal e dos Territórios (TJDFT) via Modelo Nacional de Interoperabilidade (MNI) resultou na Tabela 6.3.

[1] `http://www.pg.df.gov.br/inteligenciaartificial/`

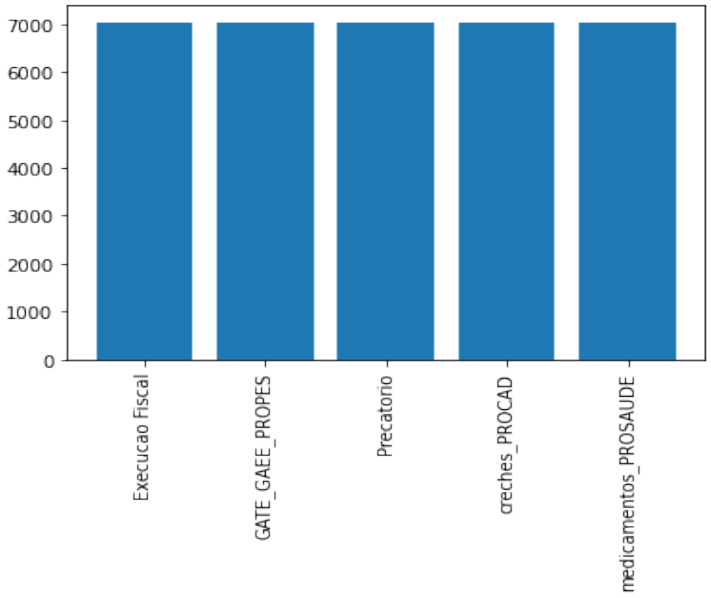

Figura 6.4: Aplicação de Synthetic Minority Oversampling Technique (SMOTE) na nova base

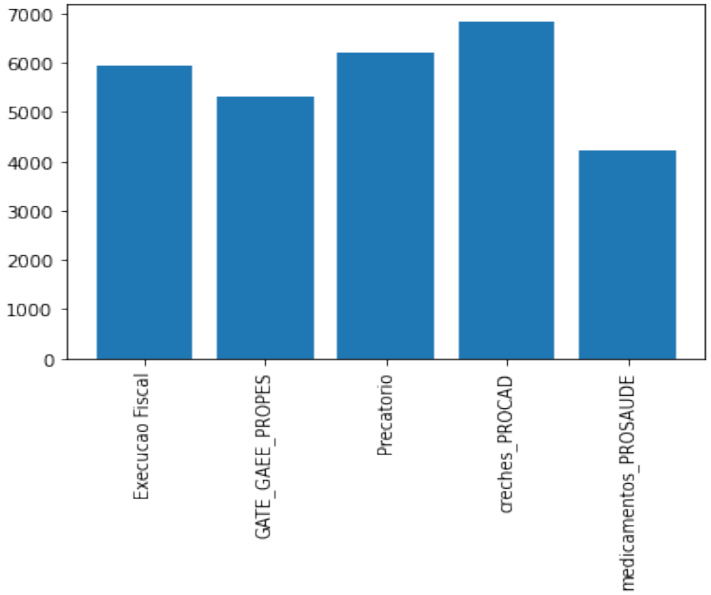

Figura 6.5: Aplicação da técnica SMOTE and Edited Nearest Neighbors (SMOTEENN) na nova base

Ou seja, após a coleta dos códigos dos processos rotulados, revisados e certificados no sistema de informação e gestão de documentos da PGDF para posterior coleta das petições iniciais via MNI foi feito o pré-processamento. Pode-se visualizar que ainda existe um desbalanceamento nos dados, por meio da figura 6.1 que mostra as quantidades de processos por classe.

Seguiu-se o pré-processamento aplicando-se as técnicas de amostragem em bases desbalanceadas. O primeiro passo foi dividir os dados em dois para treinamento e teste. A

Figura 6.2 é a visualização da parte extraída dos dados para o treinamento. Nesta que se vai trabalhar as amostragens, pois o desbalanceamento impacta o treinamento e não os testes, visto que os testes são feitos unitariamente sem reajustes do modelo.

A primeira técnica de re-amostragem em bases desbalanceadas foi a Random Under-Sampling (RUS), que pode ser verificada na Figura 6.3. Nela um o corte foi feito em aproximadamente 1750(um mil setecentos e cinquenta) observações para cada classe. Se comparado com a Figura 5.13 pode-se concluir que nesta base a quantidade de observações por classe minoritária é mais significativa.

Após esses resultados mais significativos, segui-se para a aplicação das outras técnicas de re-amostragem. As Figuras 6.4 e 6.5, respectivamente, apresentam os resultados das técnicas Synthetic Minority Oversampling Technique (SMOTE) e SMOTE and Edited Nearest Neighbors (SMOTEENN). Como dito, a primeira é uma técnica de *oversampling*, simplesmente criando dados sintéticos para as classes minoritárias e a segunda é uma técnica híbrida.

6.2 Modelagem e resultados finais

Uma primeira execução dos algoritmos selecionados foi feita. A Tabela 6.4 demonstra que os resultados foram melhores que com a base anterior. A Tabela 5.19 tem uma precisão macro média(avg) que não passou de 40%. O algoritmo *Gradient Boosting Classifier* foi o que deu cabo aos resultados presentes na Tabela 6.4 com F_1-Score macro médio de 89%, mas todos os outros algoritmos selecionados para os experimentos finais, presentes na Tabela 5.19, também apresentaram desempenho similar. O que pode ser observado na Tabela 6.5, discutida mais adiante.

Tabela 6.4: Primeira execução da terceira bateria de experimentos

Gradient Boosting Classifier			
	precision	recall	f1-score
Execucao Fiscal	0.99	1.00	0.99
GATE_GAEE_PROPES	0.99	0.98	0.99
Precatorio	0.94	0.93	0.93
creches_PROCAD	0.68	0.98	0.80
medicamentos_PROSAUDE	0.95	0.58	0.72
accuracy			0.94
macro avg	0.91	0.89	0.89
weighted avg	0.95	0.94	0.94

A Tabela 6.5 demonstra a aplicação dos algoritmos selecionados e as técnicas de re-amostragem. Pode-se observar uma melhoria significativa. Inclusive, já se pode perceber

que a expectativa de precisão do trabalho foi atendida. Mesmo assim, partiu-se para a implementação das redes neurais em *deep learnig* utilizando esta nova base com *10-Fold cross-validation* na intensão de se ter mais garantias da qualidade dos resultados.

Tabela 6.5: Resultados do desempenho dos modelos na nova base em F_1-Score macro médio

Nova base e as técnicas *GridSearchCV, OneVsRestClassifier* e *10-Fold cross-validation*						
Inbalance X Classifier	Logistic Regression	SVC	K Neighbors Classifier	Decision Tree Classifier	Random Forest Classifier	Gradient Boosting Classifier
RUS	0.88	0.88	0.70	0.86	0.88	0.89
SMOTE	0.88	0.89	0.83	0.87	0.88	0.89
SMOTEEN	0.87	0.88	0.83	0.86	0.87	0.87

6.3 Comparação dos algoritmos e as redes neurais em *deep learnig*

Para comparar os algoritmos foi utilizada uma técnica de 5(cinco) vezes 2(duas) validações cruzadas com teste t de *student*. Isso, com todas as técnicas de amostragem em bases desbalanceadas, para RUS, SMOTE e SMOTEENN. A comparação foi feita par-a-par em cada um dos algoritmos selecionados para os experimentos finais: *Logistic Regression*, Support Vector Machine (SVM), *K Neighbors Classifier*, *Decision Tree Classifier*, *Random Forest Classifier* e o *Gradient Boosting Classifier*. Ou seja, a combinação de $3\binom{6}{2}$ resultou em 45 execuções de comparações.

As Tabelas 6.6, 6.7 e 6.8 apresentam os p-valor do teste de hipótese que foram maiores que 0.05. O teste de hipóteses tinha como hipótese nula que não havia diferença significativa entre os modelos. Esse p-valor maiores que 0.05 demostram que os modelos com os algoritmos *Logistic Regression* e *Random Forest* não refutaram a hipóteses de que os modelos são similares. Assim, como os modelos com os algoritmos *Gradient Boosting* e Support Vector Machine (SVM). Desse modo, se elegeu o algoritmo *Gradient Boosting Classifier* arbitrariamente para comparar com os demais algoritmos com redes neurais em *deep learning*.

Avançando nos experimentos têm-se os resultados da Tabela 6.9, que demonstra que o *deep learning* com *10-Fold cross-validation* acompanha a melhoria de desempenho. Aparentemente com melhores resultados que todos os outros modelos. O que restou a ser feito foi realizar uma comparação estatística com esses resultados das classificações, de modo a eleger o melhor modelo. Por hora, tem-se que registar uma tentativa de classificação com um algoritmo de classificação em *deep learning* pré-treinado. O algoritmo pré-treinado

selecionado foi o Bidirectional Encoder Representations from Transformers (BERT) da Google[2].

Tabela 6.6: Comparação com 5 X 2-Fold-cross-validation e RUS

p-valor - RUS	Logistic Regression	SVM
Random Forest	0.309	-
Gradient Boosting	-	0.908

Tabela 6.7: Comparação com 5 X 2-Fold-cross-validation e SMOTE

p-valor - SMOTE	Logistic Regression	SVM
Random Forest	0.352	-
Gradient Boosting	-	0.884

Tabela 6.8: Comparação com 5 X 2-Fold-cross-validation e SMOTEENN

p-valor - SMOTEENN	Logistic Regression	SVM
Random Forest	0.382	-
Gradient Boosting	-	1

Tabela 6.9: Resultados do desempenho do *deep learning* em F_1-Score macro médio para o BERT

Deep learning(tensorflow e keras)	
SMOTEENN	0.95
RUS	0.94
SMOTE	0.95

Tabela 6.10: Resultados do desempenho dos modelos na nova base em F_1-Score macro médio para o BERT

	precision	recall	f1-score
0	0.91	0.85	0.88
1	0.83	0.85	0.84
2	0.92	0.93	0.92
3	0.93	0.98	0.95
4	0.83	0.82	0.83
accuracy			0.89
macro avg	0.89	0.89	0.89
weighted avg	0.89	0.89	0.89

[2]https://github.com/google-research/bert

Tabela 6.11: Resultados do desempenho dos modelos na nova base em F_1-Score macro médio para o *Gradient Boosting Classifier*

	precision	recall	f1-score
0	0.72	0.98	0.83
1	0.96	0.62	0.75
2	0.99	0.97	0.98
3	0.98	0.99	0.99
4	0.93	0.93	0.93
accuracy			0.90
macro avg	0.92	0.90	0.90
weighted avg	0.92	0.90	0.90

Na comparação o algoritmo *Gradient Boosting Classifier* teve o resultado da Tabela 6.11. Neste teste, o que se mostrou mais interessante foi o resultado da comparação estatística. Quando da aplicação do teste estatístico para comparação da significância da diferença, realmente não se refuta a hipótese de diferença significativa entre os 90% do *Gradient Boosting Classifier* com One-versus-All (OVA) e o *GridSearchCV* do *Scikit Learn* e os 89% do algoritmo pré-treinado BERT. O teste resultou em um p-valor de 0.268.

Vale também observar as matrizes de confusão dos dois modelos. Comparando as Tabelas 6.12 e a 6.13 pode-se perceber que a primeira, que corresponde ao modelo do algoritmo pré-treinado BERT tem mais erros que o do modelo do algoritmo *Gradient Boosting Classifier*, expresso na Tabela 6.13. Porém a diferença não foi considerada significativa estatisticamente.

Tabela 6.12: Matriz de confusão para o BERT

289	35	3	7	5
16	278	4	3	27
2	4	293	1	16
1	3	0	335	3
9	15	18	14	258

Tabela 6.13: Matriz de confusão para o *Gradient Boosting Classifier*

329	2	0	6	2
123	198	0	0	7
2	0	308	0	6
0	0	0	340	2
10	7	0	1	296

Quando comparados com teste estatístico Cochran's Q Test para verificação da significância da diferença entre os 3(três) modelos *Gradient Boosting Classifier* com One-versus-

All (OVA) e o *GridSearchCV* do *Scikit Learn*, o BERT e o *deep learnig* com tensorFlow e keras o teste resultou em um p-valor de 0. Ou seja, como o p-valor foi menor que 0.05 podemos refutar a hipótese estatística de que os modelos são significativamente similares. Portanto, o modelo *deep learnig* com tensorFlow e keras tem diferença com seus 95% de precisão.

Para certificar foram feitos testes par-a-par com o BERT e o *deep learnig* com tensorFlow e keras, também, resultando em p-valor igual a 0(zero). Além disso, foi feito outro teste par-a-par com *Gradient Boosting Classifier* se valendo de One-versus-All (OVA) com *GridSearchCV* do *Scikit Learn* e o *deep learnig* com tensorFlow e keras, da mesma forma, resultando em p-valor igual a 0(zero).

Com isso, os experimentos foram encerrados e o algoritmo com redes neurais em *deep learnig* com tensorFlow e keras foi eleito para montar o melhor modelo para a classificação dos documentos no contexto deste estudo. Importante notar que a melhora de desempenho teve mais relação com a base de dados estar mais susceptível a uma classificação supervisionada do que propriamente a técnica ou algoritmo utilizado.

Com isso, para a PGDF é amplamente sabido que urge a necessidade de utilização do aprendizado de máquina nos trabalhos processuais repetitivos. Enquanto a mão de obra especializada se torna cada dia mais dispendiosa os recursos computacionais vão se tornando mais baratos, de mais fácil acesso. Sendo assim, o principal impacto deste trabalho para a instituição foram as possibilidades que ficam claras com o uso de seus próprios recursos. Muito mais que um protótipo funcional e documentação a casa jurídica sabe que pode avançar na estruturação para um implementação de modelos de aprendizado de máquina que trarão real benefício para sua atividade fim.

Capítulo 7

Conclusão, limitações e Trabalhos futuros

Este capítulo trata das conclusões retiradas a partir da execução do trabalho. Serão apresentas as limitações que foram impostas no decorrer da pesquisa. Ao final será apresentada as intenções de trabalhos futuros.

7.1 Conclusão e limitações

As conclusões são iniciadas apresentando que este trabalho experimentou várias técnicas de pré-processamento em bases de dados desbalanceadas e comparou classificadores computacionais utilizando dados da Procuradoria Geral do Distrito Federal (PGDF). Sabe-se que remanesce uma necessidade latente de se otimizar o uso de recursos humanos na atividade de classificação de processos judiciais, que hoje é feita manualmente. A expectativa seria de que o aprendizado de máquina seria mais rápido que a classificação manual.

Sabendo que em média uma pessoa pode levar cerca de 20 (vinte) a 30 (trinta) minutos para classificar um processo judicial de natureza simples e conhecida no contexto deste trabalho, este trabalho conseguiu demonstrar viabilidade para um aprofundamento na implementação de um classificador com o auxílio do computador em produção. Pois, o classificador com aprendizado de máquina que consegue classificar um processo na ordem de segundos. Certamente a manutenção do modelo deve influenciar, pois o retrino é necessário. Entretanto, o retreino com o modelo de melhor desempenho comparativo, *deep learning* é retreinado em horas, o que daria para classicar apenas algumas dezenas de processo manualmente.

Ou seja, depois do modelo estar treinado uma classificação está sendo feita na ordem de tempo de segundos. Lembrado que a automatização das classificações em si já se mostra vantajosa na hipótese de se fazer a atividade automaticamente no mesmo tempo

que uma pessoa, pois retiraria as pessoas das atividades repetitivas de classificação para posicioná-la em outras atividades mais intelecto-produtivas.

Com relação às análises descritivas pode-se concluir que dependendo dos dados coletados podemos encontrar grande desbalanceamento. A expectativa do órgão no início do trabalho seria de que o computador fizesse a classificação sem ter dados previamente rotulados. Por isso que a base de dados inicial estava muito desbalanceada e com dados sem garantia de rotulagem adequada. Pode-se considerar esta a primeira limitação. Mas, os métodos de análise descritiva dos dados trouxeram à tona alguns problemas que não estavam visíveis para a área de negócio. Por exemplo, a má classificação dos anos anteriores para os documentos de precatório.

A expectativa inicial do negócio seria de que o computador corrigisse as possíveis más classificações que foram feitas no passado. Foi um árduo trabalho conseguir entender isso e explicar como seria o real processo de classificação com o uso do computador. Principalmente com algoritmos de classificação supervisionados, pois necessitavam de rótulos bem feito para se realizar a fase de treinamento. O trabalho avançou na medida que o processo de entendimento de negócio e entendimento dos dados foi sendo explicitado para ambas as partes e se equalizou a necessidade de negócio com as possibilidades técnicas.

Pode-se concluir que mesmo se utilizando as técnicas ditas estado da arte a condição dos dados é muito importante para o aprendizado de máquina. Também pode-se concluir que modelos não tão modernos combinados podem gerar resultados bem interessantes. Visto que a fase de treinamentos de modelos mais tradicionais pode ser considerada mais rápida que modelos mais modernos como o *Bidirectional Encoder Representations from Transformers (BERT)* que demorou dias para finalizar o treinamento. Se o modelo precisa ser retreinado com frequência e não se dispõe de altos recursos computacionais, pode-se pensar em utilizar um modelo mais tradicional como o *Support Vector Machine (SVM).* Os modelos mais antigos são treinados em questão de horas e a velocidade de classificação é similar. O desempenho que variou na diferença máxima de 5%. Ou seja, no pior caso tivemos 90% para o *Gradient Boosting Classifier* e 95% para o *deep learnig.*

Outra conclusão são de que se verificou a eficiência do cascateamento de modelos binários perante outros métodos. O modelo que apresentou melhor desempenho foi o *deep learning* com score de 95% de acerto, na métrica F_1-Score macro médio. Foi nítida a melhoria promovida pelo uso das técnicas de amostragem em dados desbalanceados. Não foi possível realizar análises descritivas mais profundas com os dados encontrados de modo a entender os ganhos com a classificação automática, pois grande quantidade de dados estavam muito desbalanceados. Pode-se considerar essa a segunda limitação.

Conclui-se que os documentos jurídicos de processos repetitivos são melhor classificados automaticamente com o uso de redes neurais em *deep learnig*, refutando a hipótese

que os modelos em *ensemble learning* seriam mais eficientes no caso estudado. Não foi possível quantificar distribuição do insucesso do contencioso utilizando os agrupamentos de documentos jurídicos. Isso será tema de trabalho futuro. A técnica de amostragem que melhor se adequou aos modelos dependeu do modelo e dos dados. Não foi possível eleger uma melhor técnica de amostragem para este problema.

Por fim, diante dos melhores resultados com a troca de base de dados podemos concluir que o desempenho do modelo vai depender muito da qualidade dos dados. Esta, por sua vez, está relacionada à exatidão com que os rótulos nos dados foram feitos. Podemos perceber que na primeira base todos os esforços não resultaram em uma melhoria de desempenho. Utilizando as mesmas técnicas na nova base de dados, com mais garantias de qualidade na rotulagem, o trabalho chegou nos resultados esperados.

7.2 Trabalhos futuros

Para os trabalhos futuros se pretende aumentar os números de classes a serem identificadas. Estudar melhor os dados de precatório para, talvez, apresentar uma sugestão de rotulação assistida pelo computador, verificar a aplicação de reconhecimento de entidades nomeadas nos documentos classificados para coletar as partes do processo e criar um mecanismo de sugestão de petições iniciais.

Para aumentar o número de classes a serem identificadas percebe-se que serão necessárias mais rodadas de entrevistas, pois é visível a necessidade de se aumentar o entendimento do negócio e dos dados. Visto que a qualidade dos dados rotulados é fundamental para uma rápida transição da classificação manual para a automática. Inicialmente pode-se focar nos casos mais simples para melhor aproveitar as pessoas humanas em outras atividades mais intelecto-produtivas.

Com relação aos dados de precatório inicialmente verificados permanece a necessidade da casa de se ajustar aquelas classificações. Então se precisa estudar melhor os dados de precatório talvez seja necessário utilizar classificadores não supervisionados, como foi tentado com o Latent Dirichlet Allocation (LDA) neste trabalho, sem sucesso. Assim, uma equipe poderia certificar os agrupamentos automáticos e direcionar para a melhor classe dentre as permitidas na ontologia no Conselho Nacional de Justiça (CNJ). Assim, talvez aceleraria o processo de ajuste dessas classificações.

Outra contribuição seria na evolução de algum conhecimento na verificação das entidades nomeadas dos documentos, pois reconhecendo as entidades nomeadas, será possível se verificar as partes do processo e, possivelmente, se criar petições iniciais automáticas. A ideia é que se crie uma base de petições para que o sistema faça sugestões de petições iniciais ou outras peças. Ou seja, entendendo de que tema é a intimação e entendendo,

por exemplo, que o Governo do Distrito Federal (GDF) é polo passivo na ação, se poderá sugerir algumas peças para que o procurador selecione aquela que melhor se adeque ao caso repetitivo e o sistema completa as peças com as entidades nomeadas reconhecidas. Visto que já se tem na PGDF petições iniciais feitas manualmente para alguns casos conhecidos e repetitivos. A partir da identificação do tema repetitivo já se tem um modelo pronto, porém, atualmente, é preenchido manualmente.

Referências Bibliográficas

[1] Andrade, Patrícia Helena Maia Alves de: *Aplicação de técnicas de mineração de textos para classificação de documentos : um estudo da automatização da triagem de denúncias na CGU*. Dissertação (Mestrado Profissional em Computação Aplicada)—Universidade de Brasília, setembro 2015. `http://repositorio.unb.br/bitstream/10482/21004/1/2015_PatríciaHelenaMaiaAlvesdeAndrade.pdfhttp://repositorio.unb.br/handle/10482/21004`. 4, 22, 47

[2] Rocha, Ana Carolina Pereira: *Universidade de Brasília Mineração de Textos para Classificação de Processos Judiciais Trabalhistas no PJe-JT*. Dissertação (Mestrado Profissional em Computação Aplicada) — Universidade de Brasília, página 148, 2019. `https://repositorio.unb.br/handle/10482/37933`. 4, 22, 40, 47

[3] Dermot Turing: *A História da computação: do Ábaco à Inteligência Artificial*. M. Books do Brasil Editora Ltda., 2019, ISBN 9788576803188. 6

[4] Turing, A. M.: *Computing Machinery and Intelligence*. Mind, 49:433–460, 1950, ISSN 25436031. 6

[5] McCulloch, Warren S. e Walter H. Pitts: *A Logical Calculus of Ideas Immanent in Nervous Activity*. 5:115–133, 1943. 7

[6] Russell, Stuart e Peter Norvig: *Artificial Intelligence: A Modern Approach*. Prentice Hall, 3ª edição, 2010. 7, 8, 9, 10, 15, 16, 34, 75

[7] Mitchell, Tom M: *Machine Learning*. McGraw-Hill, New York, 1997, ISBN 978-0-07-042807-2. 7, 8, 9

[8] Bishop, Christopher M: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2006, ISBN 978-0-387-31073-2. 8, 9, 10, 15

[9] Christian, Hans, Mikhael Pramodana Agus e Derwin Suhartono: *Single Document Automatic Text Summarization using Term Frequency-Inverse Document Frequency (TF-IDF)*. ComTech: Computer, Mathematics and Engineering Applications, 7(4):285, 2016, ISSN 2087-1244. 8

[10] Sen, Pratap Chandra, Mahimarnab Hajra e Mitadru Ghosh: *Supervised Classification Algorithms in Machine Learning: A Survey and Review*. Em *Advances in Intelligent Systems and Computing*, volume 937, páginas 99–111. 2020, ISBN 9789811374029. `http://link.springer.com/10.1007/978-981-13-7403-6_11`. 10, 11, 12

[11] Shah, Kanish, Henil Patel, Devanshi Sanghvi e Manan Shah: *A Comparative Analysis of Logistic Regression, Random Forest and KNN Models for the Text Classification.* Augmented Human Research, 5(1):12, dezembro 2020, ISSN 2365-4317. `https://doi.org/10.1007/s41133-020-00032-0http://link.springer.com/10.1007/s41133-020-00032-0`. 10, 11, 12

[12] Lorena, Ana Carolina e André C. P. L. F. De Carvalho: *Uma Introdução às Support Vector Machines.* Revista de Informática Teórica e Aplicada, 14(2):43–67, dezembro 2007, ISSN 21752745. `https://seer.ufrgs.br/rita/article/view/rita_v14_n2_p43-67`. 11

[13] Hearst, Marti. A., Bernhard. Scholkopf, Susan. Dumais, Edgar. Osuna e John Platt: *Trends and controversies- Support Vector Machines.* IEEE Intelligent Systems and their Applications, 13(4):18 – 28, 1998, ISSN 1094-7167. 11

[14] Salsburg, D S e J M Gradel: *Uma senhora toma chá : como a estatística revolucionou a ciência no século XX.* Ciência da Vida Comum. Zahar, 2009, ISBN 9788537801161. `https://books.google.com.br/books?id=Q9CePgAACAAJ`. 12

[15] Sulzmann, Jan Nikolas, Johannes Fürnkranz e Eyke Hüllermeier: *On pairwise naive bayes classifiers.* Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 4701 LNAI:371–381, 2007, ISSN 16113349. 12

[16] Natekin, Alexey e Alois Knoll: *Gradient boosting machines, a tutorial.* Frontiers in Neurorobotics, 7(DEC), 2013, ISSN 16625218. 12

[17] Friedman, Jerome H.: *Greedy function approximation: A gradient boosting machine.* Annals of Statistics, 29(5):1189–1232, 2001, ISSN 00905364. 13

[18] Goodfellow, Ian, Yoshua Bengio e Aaron Courville: *Deep Learning.* MIT Press, 2016, ISBN 9780262035613. `http://www.deeplearningbook.org`. 13, 14

[19] Neha Bansal, Arun Sharma e R.K. Singh: *A Review on the Application of Deep Learning in Legal Domain*, volume 2. Springer International Publishing, 2019. 14, 40

[20] Blei, David M: *Latent Dirichlet Allocation.* Journal of Machine Learning Research, 3:30, 2003. 15

[21] Ghosal, Attri, Arunima Nandy, Amit Kumar Das, Saptarsi Goswami e Mrityunjoy Panday: *A Short Review on Different Clustering Techniques and Their Applications*, volume 937. Springer Singapore, 2020, ISBN 978-981-13-7402-9. `http://link.springer.com/10.1007/978-981-13-7403-6`. 15

[22] Ghosh, Debraj: *A Sentiment-Based Hotel Review Summarization.* Em *Advances in Intelligent Systems and Computing*, volume 1, páginas 39–44. 2020. `http://link.springer.com/10.1007/978-981-13-7403-6_5`. 16, 17

[23] Irshad, Areeba e Malay Kishore Dutta: *Identification of Windows-Based Malware by Dynamic Analysis Using Machine Learning Algorithm*. páginas 207–218. 2021. http://link.springer.com/10.1007/978-981-15-1275-9_18. 16, 17

[24] Forman, George: *An Extensive Empirical Study of Feature Selection Metrics for Text Classification*. CrossRef Listing of Deleted DOIs, 1:1289–1305, 2000, ISSN 0003-6951. 17

[25] Kosmajac, Dijana e Vlado Keselj: *Slavic language identification using cascade classifier approach*. 2018 17th International Symposium on INFOTEH-JAHORINA, INFOTEH 2018 - Proceedings, 2018-Janua(March):1–6, 2018. 17, 38

[26] Gao, Xiao zhi: *Advances in Intelligent Systems and Computing 1086 Advances in Computational Intelligence and Communication Technology*, volume 2. 2019, ISBN 9789811512742. 17

[27] Bishop, Christopher M: *Model-based machine learning*. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 371(1984):20120222, fevereiro 2013, ISSN 1364-503X. https://royalsocietypublishing.org/doi/10.1098/rsta.2012.0222. 18

[28] Newman, David, Jey Han Lau, Karl Grieser e Timothy Baldwin: *Automatic evaluation of topic coherence*. NAACL HLT 2010 - Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Proceedings of the Main Conference, (June):100–108, 2010. 18

[29] McNemar, Quinn: *Note on the sampling error of the difference between correlated proportions or percentages*. Psychometrika, 12(2):153–157, 1947, ISSN 00333123. 18

[30] Bagui, Subhash C: *Combining Pattern Classifiers: Methods and Algorithms*, volume 47. 2005, ISBN 9786468600. 19, 21

[31] Dietterich, Thomas G: *Statistical Tests for Comparing Supervised Classi cation Learning Algorithms 1 Introduction*. Science, 10(7):1–24, 1997. http://dx.doi.org/10.1162/089976698300017197. 19, 20

[32] Yin, R K: *Estudo de Caso - 5.Ed.: Planejamento e Métodos*. Bookman Editora, 2015, ISBN 9788582602324. https://books.google.com.br/books?id=EtOyBQAAQBAJ. 23

[33] Dresch, Aline, Daniel Pacheco Lacerda e José Antônio Valle Antunes Jr.: *Design Science Research: A Method for Science and Technology Advancement*. Bookman Editora, 2015, ISBN 9783319073736. 29, 31, 42

[34] Rout, Neelam, Debahuti Mishra e Manas Kumar Mallick: *Handling imbalanced data: A survey*. Advances in Intelligent Systems and Computing, 628(January):431–443, 2018, ISSN 21945357. 34

[35] Fernández, Alberto, Victoria López, Mikel Galar, María José Del Jesus e Francisco Herrera: *Analysing the classification of imbalanced data-sets with multiple classes: Binarization techniques and ad-hoc approaches.* Knowledge-Based Systems, 42:97–110, 2013, ISSN 09507051. 34, 35, 36

[36] Wang, Shuo e Xin Yao: *Multiclass imbalance problems: Analysis and potential solutions.* IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 42(4):1119–1130, 2012, ISSN 10834419. 34

[37] Haixiang, Guo, Li Yijing, Jennifer Shang, Gu Mingyun, Huang Yuanyue e Gong Bing: *Learning from class-imbalanced data: Review of methods and applications.* Expert Systems with Applications, 73(December):220–239, 2017, ISSN 09574174. `http://dx.doi.org/10.1016/j.eswa.2016.12.035`. 34, 46

[38] Gao, Xin, Yang He, Mi Zhang, Xinping Diao, Xiao Jing, Bing Ren e Weijia Ji: *A multiclass classification using one-versus-all approach with the differential partition sampling ensemble.* Engineering Applications of Artificial Intelligence, 97(July 2019):104034, 2021, ISSN 09521976. `https://doi.org/10.1016/j.engappai.2020.104034`. 35, 36, 37

[39] Hasib, Khan Md, Md Sadiq Iqbal, Faisal Muhammad Shah, Jubayer Al Mahmud, Mahmudul Hasan Popel, Md Imran Hossain Showrov, Shakil Ahmed e Obaidur Rahman: *A Survey of Methods for Managing the Classification and Solution of Data Imbalance Problem.* Journal of Computer Science, 16(11):1546–1557, 2020, ISSN 15526607. 35

[40] Gama, João e Pavel Brazdil: *Cascade Generalization.* Machine Learning - Kluwer Academic Publishers. Manufactured in The Netherlands, 41(3):315–343, 2000, ISSN 08856125. 36, 37

[41] Calvo, Hiram e Omar Juárez Gambino: *Cascading classifiers for twitter sentiment analysis with emotion lexicons.* Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9624 LNCS:270–280, 2018, ISSN 16113349. 37, 38

[42] Rifkin, Ryan e Aldebaro Klautau: *In Defense of One-Vs-All Classification.* Journal Of Machine Learning Research, 5:2–6, 2004. 38

[43] Wang, Yefeng e Jon Patrick: *Cascading Classifiers for Named Entity Recognition in Clinical Notes.* Workshop Biomedical Information Extraction, Annual Meeting of the Association for Computational Linguistics, páginas 42–49, 2009. `http://www.ihtsdo.org/publications/`. 38

[44] Halgrim, Scott Russell, Fei Xia, Imre Solti, Eithon Cadag e Özlem Uzuner: *A cascade of classifiers for extracting medication information from discharge summaries.* Journal of Biomedical Semantics, 2(3), 2011, ISSN 20411480. 38

[45] Pławiak, Paweł, Moloud Abdar e U. Rajendra Acharya: *Application of new deep genetic cascade ensemble of SVM classifiers to predict the Australian credit scoring.*

Applied Soft Computing Journal, 84:105740, 2019, ISSN 15684946. https://doi.org/10.1016/j.asoc.2019.105740. 39

[46] Nikolić, Nikola, Olivera Grljević e Aleksandar Kovačević: *Aspect-based sentiment analysis of reviews in the domain of higher education.* Electronic Library, 38(1):44–64, 2020, ISSN 02640473. 39

[47] Surden, Harry: *Machine Learning and Law.* Wash. L. Rev, 87:89, 2014. http://scholar.law.colorado.edu/articleshttp://scholar.law.colorado.edu/articles/81. 39

[48] Araujo, Pedro Henrique Luz de, , Teófilo Emídio de Campos, , Fabricio Ataides Braz, e Nilton Correia da Silva: *{VICTOR}: a Dataset for {B}razilian Legal Documents Classification.* Em *Proceedings of the 12th Language Resources and Evaluation Conference*, número May, páginas 1449–1458. 2020, ISBN 979-10-95546-34-4. https://aclanthology.org/2020.lrec-1.181. 39

[49] Braz, Fabricio Ataides, Nilton Correia da Silva, Teofilo Emidio de Campos, Felipe Borges S. Chaves, Marcelo H. S. Ferreira, Pedro Henrique Inazawa, Victor H. D. Coelho, Bernardo Pablo Sukiennik, Ana Paula Goncalves Soares de Almeida, Flavio Barros Vidal, Davi Alves Bezerra, Davi B. Gusmao, Gabriel G. Ziegler, Ricardo V. C. Fernandes, Roberta Zumblick e Fabiano Hartmann Peixoto: *Document classification using a Bi-LSTM to unclog Brazil's supreme court.* 2018. http://arxiv.org/abs/1811.11569. 40

[50] Braz, Fabricio Ataides, Nilton Correia da Silva, Teofilo Emidio de Campos, Felipe Borges S. Chaves, Marcelo H. S. Ferreira, Pedro Henrique Inazawa, Victor H. D. Coelho, Bernardo Pablo Sukiennik, Ana Paula Goncalves Soares de Almeida, Flavio Barros Vidal, Davi Alves Bezerra, Davi B. Gusmao, Gabriel G. Ziegler, Ricardo V. C. Fernandes, Roberta Zumblick e Fabiano Hartmann Peixoto: *Document type classification for Brazil's supreme court using a Convolutional Neural Network.* páginas 2–5, 2018. http://arxiv.org/abs/1811.11569. 40

[51] Araujo, Pedro Henrique Luz de, Teófilo E. de Campos, Renato R.R. de Oliveira, Matheus Stauffer, Samuel Couto e Paulo Bermejo: *LeNER-Br: A Dataset for Named Entity Recognition in Brazilian Legal Text.* Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 11122 LNAI:313–323, 2018, ISSN 16113349. 40

[52] Waltl, Bernhard, Johannes Muhr, Ingo Glaser, Georg Bonczek, Elena Scepankova e Florian Matthes: *Classifying legal norms with active machine learning.* Frontiers in Artificial Intelligence and Applications, 302:11–20, 2017, ISSN 09226389. 40

[53] Lorenzo, Catania, Di Silvestro, Daria Spampinato e Alessandro Torrisi: *Automatic Classification of Legal Textual Documents using C4.5.* 2009. http://snowball.tartarus.org/algorithms/italian/stemmer.html. 40

[54] Undavia, Samir, Adam Meyers e John E. Ortega: *A comparative study of classifying legal documents with neural networks.* Proceedings of the 2018 Federated Conference on Computer Science and Information Systems, FedCSIS 2018, 15:515–522, 2018. 40

[55] Elnaggar, Ahmed, Christoph Gebendorfer, Ingo Glaser e Florian Matthes: *Multi-task deep learning for legal document translation, summarization and multi-label classification.* ACM International Conference Proceeding Series, (September):9–15, 2018. 40

[56] Landthaler, Jörg, Bernhard Waltl, Patrick Holl e Florian Matthes: *Extending full text search for legal document collections using word embeddings.* Frontiers in Artificial Intelligence and Applications, 294:73–82, 2016, ISSN 09226389. 41

[57] John, Adebayo Kolawole, Luigi Di Caro, Livio Robaldo e Guido Boella: *Legalbot: A deep learning-based conversational agent in the legal domain.* Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10260 LNCS:267–273, 2017, ISSN 16113349. 41

[58] Kowsrihawat, Kankawin, Peerapon Vateekul e Prachya Boonkwan: *Predicting judicial decisions of criminal cases from Thai supreme court using bi-directional gru with attention mechanism.* Proceedings of the 5th Asian Conference on Defence Technology, ACDT 2018, páginas 50–55, 2018. 41

[59] Li, Shang, Hongli Zhang, Lin Ye, Xiaoding Guo e Binxing Fang: *Evaluating the rationality of judicial decision with LSTM-based case modeling.* Proceedings - 2018 IEEE 3rd International Conference on Data Science in Cyberspace, DSC 2018, páginas 392–397, 2018. 41

[60] Polo, Felipe Maia, Itamar Ciochetti e Emerson Bertolo: *Predicting Legal Proceedings Status: an Approach Based on Sequential Text Data.* 2020. `http://arxiv.org/abs/2003.11561`. 41

[61] Wei, Fusheng, Han Qin, Shi Ye e Haozhen Zhao: *Empirical Study of Deep Learning for Text Classification in Legal Document Review.* Proceedings - 2018 IEEE International Conference on Big Data, Big Data 2018, páginas 3317–3320, 2019. 41

[62] Creswell, John W: *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches.* 2013. 42

[63] Kauark, Fabiana da Silva Kauark, Fernanda Castro Manhães Carlos e Henrique Medeiros: *Metodologia da Pesquisa: Um guia prático.* Livro, páginas 99 –117, 1390. 42

[64] Mariano, Ari Melo e Maíra Rocha Santos: *Revisão da Literatura: Apresentação de uma Abordagem Integradora.* AEDEM International Conference, (September):427–443, 2017. 42

[65] Lopez, André Porto Ancona: *Diretrizes para o desenvolvimento de projetos de cunho científico.*, 2011. 42

[66] Azevedo, Ana e Manuel Filipe Santos: *KDD, SEMMA AND CRISP-DM: A PARALLEL OVERVIEW.* 2008. 43

[67] Chapman, Pete, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer e Wirth Rudiger: *Crisp-Dm 1.0*. CRISP-DM Consortium, SPSS inc., página 76, 2000. 43

[68] Sievert, Carson e Kenneth Shirley: *LDAvis: A method for visualizing and interpreting topics*. páginas 63–70, 2015. 44

[69] McDonald, Gary C.: *Ridge Regression*. wires computational statistics, 2009. 45

[70] Gallant, Stephen I.: *Perceptron-Based Learning Algorithms*. IEEE Transactions on Neural Networks, 1(2):179–191, 1990, ISSN 19410093. 45

[71] Crammer, Koby, Ofer Dekel, Joseph Keshet, Shai Shalev Shwartz Singer e Yoram Singer: *Online Passive-Aggressive Algorithms Koby*. Journal ofMachine Learning Research 7, 7:551–585, 2006. 45

[72] Altman, N. S.: *An introduction to kernel and nearest-neighbor nonparametric regression*. American Statistician, 46(3):175–185, 1992, ISSN 15372731. 45

[73] Ho, Tin Kam: *Random decision forests*. Proceedings of the International Conference on Document Analysis and Recognition, ICDAR, 1:278–282, 1995, ISSN 15205363. 45

[74] Cortes, Corinna e Vladimir Vapnik: *Support-vector networks*. Machine Learning, 20(3):273–297, setembro 1995, ISSN 0885-6125. `http://link.springer.com/10.1007/BF00994018`. 45

[75] T.M. COVER, P.E. HART: *Nearest Neighbor Pattern Classfication*. I:1–28, 2012. 45

[76] Rennie, Jason D.M., Lawrence Shih, Jaime Teevan e David Karger: *Tackling the Poor Assumptions of Naive Bayes Text Classifiers*. Proceedings, Twentieth International Conference on Machine Learning, 2(1973):616–623, 2003. 45

[77] Vu Bui, Quang, Karim Sayadi, Soufian Ben Amor e Marc Bui: *Combining latent dirichlet allocation and K-means for documents clustering: Effect of probabilistic based distance measures*. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 10191 LNAI:248–257, 2017, ISSN 16113349. 71