Classificação Automática de Documentos Utilizando Técnicas de Inteligência Artificial

Luciano Lopes de Sousa[1], **José Vigno Moura Sousa**[1]

[1]Universidade Estadual do Piauí (UESPI)

lucianosousa@aluno.uespi.br, josevigno@prp.uespi.br

Abstract. *The organization of documents in an institution is extremely important, as it brings practicality and efficiency in retrieval. Manually separating these files according to their type takes many hours of work on the part of the responsible professional. Therefore, this work aims to develop a system capable of classifying institutional documents of the State University of Piauí (UESPI), according to their specific type. The database, initially, was composed of PDF files with the photo of the physical document. For the recognition of the texts contained in the images, the software Tesseract was used with some image processing techniques, in order to improve the performance of text extraction. From this, a database was built with the textual information of the documents and their corresponding class. After defining the database, pre-processing was performed using natural language processing techniques, preparing for the classification phases. The classification phase was carried out by two layers of classifiers, one through regular expressions (seeking to locate the type of document by the title) and another using machine learning models (analyzing the textual content of the text). These layers work with the classification flow going through the regular expressions first, if it cannot identify, it passes to the classifier model. In the final application, an API was built that performs the efficient classification process, using both regular expressions and the machine learning model with an accuracy of 98 % in cross-validation.*

Resumo. *A organização de documentos em uma instituição é de extrema importância, pois traz praticidade e eficiência na recuperação. Separar esses arquivos de acordo com seu tipo de forma manual, ocupa bastante horas de trabalho por parte do profissional responsável. Portanto, este trabalho tem como objetivo o desenvolvimento de um sistema capaz de classificar documentos institucionais da Universidade Estadual do Piauí (UESPI), de acordo com seu determinado tipo. A base de dados, inicialmente, estava dispostas de arquivos PDFs possuindo a foto do documento físico. Para o reconhecimento dos textos contidos nas imagens, foi utilizado o software Tesseract com algumas técnicas de processamentos de imagens, a fim de melhorar o desempenho da extração dos textos. A partir disso, foi construído uma base de dados com as informações textuais dos documentos e sua classe correspondente. Após a definição da base de dados, foi realizado o pré-processamento utilizando técnicas de processamento de linguagem natural, preparando para as fases de classificação. A fase de classificação se deu por duas camadas de identificação, uma através de expressões regulares (buscando localizar o tipo de documento pelo título) e outra utilizando modelos de aprendizado de máquina (analisando o conteúdo textual*

do texto). Essas camadas funciona com o fluxo de classificação passando primeiro pela expressões regulares, se não conseguir identificar, passa para o modelo classificador. Na aplicação final, foi construído uma API que realiza o processo de classificação eficiente, utilizando tanto as expressões regulares quanto o modelo de aprendizado de máquina com a acurácia de 98 % na validação cruzada.

1. Introdução

O crescente aumento no números de documentos, nos últimos anos, vem se expandindo cada vez mais [Espinosa Touzon 2020]. Serviços de armazenamento e gerenciamento digitais são indispensáveis para alocação desses tipos de dados, até por questões de segurança e confiabilidade. Várias instituições e serviços governamentais utilizam bastante esse tipo de sistemas para guardar suas informações. Além da possibilidade de armazenamento, a preservação e a integridade dos documentos é um fator importante a ser considerado. Pois a partir de um certo período de tempo, os documentos físicos podem acabar sofrendo algum tipo de desgaste e ocorrer a perda de suas informações [Arellano 2004]. Dessa forma, dispositivos que façam o escaneamento e transformações dos documentos físicos para virtuais de forma automatizada, é importante para preservar suas informações.

Na era da digitalização e armazenamento virtual, a organização é um fator primordial para conseguir gerenciar esses documentos de forma eficiente [Guha and Samanta 2019]. Assim, a classificação e separação de acordo com seu tipo é um requisito fundamental a ser cumprido, principalmente em ambientes onde há a geração de grandes quantidades de documentos digitais, como por exemplo: instituição de ensino e serviços governamentais. Esse processo de rotulação está sujeito a erros e demanda tempo (quando realizada de forma manual), necessita de bastante atenção, pois enganos na classificação podem resultar em problemas na organização e localização desses arquivos [Ciecierski and Kamola 2020]. Dessa forma, um meio que pode ajudar nesse processo, seria por intermédio de sistemas automatizados, atuando principalmente na classificação e detecção desses documentos.

Os sistemas automatizados para essa tarefa de classificação e detecção, alia tecnologia e o gerenciamento de negócios para otimizar resultados e contribuir para o alcance de objetivos globais [Ernst 2020]. Ao adotar esse tipo de sistema, uma companhia consegue reduzir a rotina de execução de uma determinada tarefa, acelerar a execução das atividades, reduzir erros de atenção e substituir alguns processos manuais por sistemas automatizados [Ernst 2020]. Um grande aliado para esse tipo de automação é com o uso de um *software* que possa aprender a realizar esse tipo de tarefa de forma automatizada.

Com a utilização de sistemas automatizados, o processo de classificação dos documentos textuais se tornaria mais eficiente. Alguns trabalhos na literatura propuseram métodos para a classificação de documentos, como em [Yesenia and Veronica 2021], [Luo 2021]. No trabalho de [Yesenia and Veronica 2021], foi proposto um sistema de classificação de trabalhos de pesquisa publicados na Scopus, onde após o processamento dos textos, foi utilizado o *K-Nearest Neighbors* e Análise de Discriminante como classificadores. Por outro lado, no projeto de [Luo 2021], criou-se um método para classificação de textos e documentos em inglês, onde utilizou-se o modelo de Máquina de Vetor de

Suporte como classificador.

Nesse projeto, é proposto a construção de um sistema que faça a classificação de documentos institucionais provenientes da UESPI, desenvolvendo um sistema que possa realizar a detecção e classificação de documentos de acordo com seu determinado tipo. Para a realização desse projeto, será necessário seguir seis passos primordiais: (*i*) Construção e preparação da base de dados; (*ii*) pré-processamento dos dados dos documentos, utilizando técnicas de Processamento de Linguagem Natural (NLP); (*iii*) Análise de expressões regulares para filtrar o documento pelo seu título, funcionando como uma camada que irá realizar o processo de filtragem; (*iv*) implementação de classificadores disponíveis na literatura para a tarefa de classificação dos documentos institucionais, caso a camada de filtragem falhe ao encontrar o rótulo do documento; (*v*) implementar e analisar as métricas de avaliação recomendadas na literatura, para obter dados estatísticos dos modelos classificadores escolhidos; (*vi*) desenvolvimento de uma API para que possa oferecer um bom sistema automatizado para a classificação de documentos, possuindo uma camada de filtragem contendo expressões regulares que irá buscar pelo título do documento de entrada, caso esse processo não consiga identificar, será utilizado um modelo classificador que irá analisar todo o conteúdo textual e retornar um resultado.

O restante deste documento está organizado como segue: o Capítulo 2 apresenta os trabalhos relacionados, que inspiraram na construção desse projeto; o Capítulo 3 apresenta a fundamentação teórica, com conceitos importantes para esse trabalho; o Capítulo 4 mostra como ocorreu todo o processo de implementação da API proposta por esse trabalho; o Capítulo 5 apresenta os resultados obtido com o projeto presente nesse artigo; o Capítulo 6 apresenta a conclusão do trabalho, mostrando os pontos obtidos e alcançada com a realização desse projeto.

2. Trabalhos relacionados

Nessa seção, será apresentado alguns trabalhos onde terá como base para a criação do projeto proposto neste trabalho. Há trabalhos onde se assemelham com essa proposta desenvolvida nesse artigo, resolvendo outras questões envolvendo o processamento e detecção de documentos.

Em [Mittal et al. 2020], foi proposto um estudo em que buscou a classificação, utilizando técnicas de linguagem de processamento natural, de um conjunto de dados que classifica os cargos com base na descrição da consulta. Para essa tarefa, foi utilizado alguns algoritmos de classificação para realização de uma análise comparativa. Os algoritmos utilizados foram: *Naive Bayes de Bernoulli*; *Naive Bayes Multinomial*; Floresta Aleatória; Linear SVM e LSVM. De acordo com as precisões avaliadas, o algoritmo de LSVM teve melhor desempenho, onde foi capaz de atingir 96,25 % de precisão para um conjunto contendo 55 mil amostras.

Na tarefa de análise de textos e documentos para classificação, o trabalho de [Luo 2021] propôs uma abordagem para a classificação de textos e documentos. Para a tarefa de classificação, foi fornecida uma análise comparativa de diferentes algoritmos de aprendizado de máquina, incluindo *Naive Bayes*, Máquina de Vetor de Suporte e Regressão logística. Dentro das abordagens analisadas, a Máquina de Vetor de Suporte obteve melhor desempenho em relação aos outros algoritmos. De acordo com [Luo 2021], os resultados obtidos ultrapassam os 90 % ao usar um conjunto de dados com uma quan-

tidade superior a 4000 recursos.

Um sistema para fazer análises de layouts em documentos foi proposto no trabalho de [Kosaraju et al. 2019]. O sistema recebeu o nome de DoT-Net, onde é capaz de fazer classificação de multiclasses e pode identificar blocos de componentes de documentos, como textos, imagens, tabelas, expressões matemáticas, entre outros. O principal foco do trabalho seria na problemática de identificar textos contra não textos. De acordo com os resultados obtidos em [Kosaraju et al. 2019], teve promissores performance, com as métricas utilizadas como acurácia, F1 score e AUC.

No projeto de [Kim and Gil 2019], tem como proposta um sistema de classificação de artigos de pesquisas que podem agrupar artigos em uma determinada classe significativa, onde esses tenham assuntos semelhantes. O sistema proposto extrai palavras chaves do resumo de cada artigo, utilizando o esquema Alocação de *Dirichlet latente*. Após isso, usa-se o algoritmo de agrupamento de *K-means* em que vai classificar todos os documentos por assuntos similares, baseado na frequência inversa da frequência do documento.

Um sistema que classifica notícias da BBC através do processamento de texto, é proposto em [Shah et al. 2020]. Para a tarefa de classificação, foram testados e comparados os algoritmos de Regressão Logística, Floresta Aleatória, *K-Nearest Neighbors*. De acordo apurados em [Shah et al. 2020], o modelo classificador de notícias da BBC obteve resultados satisfatórios. As métricas usadas para medir a performance dos algoritmos testados foram a precisão, acurácia, medida F1 e matriz de confusão. Assim, de acordo com as métricas de validação, o melhor algoritmo de acordo com os resultados, foi o escolhido para o sistema.

Descobrir o melhor tipo de algoritmo de aprendizado de máquina para a tarefa de classificação de documentos, é proposto pelo trabalho de [Basha et al. 2019]. A frequência de termo nos documentos foi usada como extração de características. O *Human-Aids* e *Mouse Câncer* foram as bases de dados usadas, com 150 instâncias cada, utilizando como método de validação as métricas de *Recall*, Precisão e medida F1. De acordo com as pesquisas de [Basha et al. 2019], mostrou que 98 % tem melhor performance do que outros algoritmos como Árvore de Decisão com 96 %, seguido do *K-Nearest Neighbor* (KNN) com 84 % em termos de acurácia.

3. Fundamentação teórica

3.1. Documentos textuais digitais

Os documentos digitais já "nascem" no ambiente digital, ou seja, não contém sua forma física no papel. Os digitalizados por sua vez, tem seu conteúdo convertido para a forma digital, onde é passado por um processo de codificação binária das informações, possibilitando o entendimento das máquinas. Os escaneados, são documentos físicos que foram passados pelo processo de escaneamento, onde gera uma imagem do documento original físico, não podendo ser editado e compreendido pela máquina.

Neste projeto, irá utilizar uma base de dados de documentos institucionais na sua forma escaneada, contendo somente a imagem do documento original. Os dados são provenientes da instituição UESPI, divididos em alguns tipos utilizados pela própria instituição, sendo: lista de espera, fichas de acompanhamento, fichas cadastrais, lista de

confirmações, reintegração, comprovantes de matrícula, documentos recebidos, fichas de AACC, histórico escolar e pedagogia de transição.

3.2. Inteligência artificial

A inteligência artificial (IA) se trata de uma tecnologia onde procura simular o processo humano de aprendizado e tomada de decisões. O termo é frequentemente aplicado ao projeto de desenvolver sistemas dotados dos processos intelectuais característicos dos seres humanos, como a habilidade de raciocinar, descobrir significados, generalizar ou aprender com a experiência passada [Haider 2021]. Ainda assim, os algoritmos inteligentes não conseguem corresponder a flexibilidade humana em diversas áreas de um ambiente, onde requer bastante conhecimento diário. Contudo, em tarefas bem específicas, uma IA pode atingir níveis de desempenho que pode se comparar a especialistas e profissionais humanos na realização de uma determinada tarefa, como por exemplo em diagnósticos médicos [Saraiva et al. 2019].

Uma IA pode, a partir de uma coleção de dados expostos, fazer previsões através de vários tipos de abordagem, como aprendizado de máquina, redes neurais, processamento de linguagem natural (através de textos), redes neurais convolucionais (através de imagem), e outras técnicas que abrangem o contexto da inteligência artificial [Haenlein and Kaplan 2019]. Em geral, os sistemas que utilizam algoritmos de IA funcionam ingerindo grandes quantidades de dados de treinamento rotulados, analisando os dados para correlações, fazendo a busca de padrões e usando esses padrões para fazer previsões sobre os estados futuros.

O aprendizado de máquina (AM), mais conhecido pela sua forma em inglês *machine learning*, é um subconjunto da IA. Nesse campo de aprendizagem, possibilitam que os sistemas inteligentes aprendam a partir de um conjunto de dados de forma autônoma, sem exigir qualquer tipo de programação específica [Borgers 2021]. Dessa forma, a partir dos dados que são expostos, aprendem a tomar decisão de forma independente. A forma de aprendizado pelos algoritmos de AM, refere-se a sua capacidade de buscar constantemente minimizar os erros de previsão e maximizar as previsões de serem verdadeiras [Rudin et al. 2022].

Na área de AM, contém uma diversidade de algoritmos para resolver os mais variados tipos de problemas de dados. Dessa forma, o problema em questão pode ditar qual tipo de algoritmo utilizar, pois pode conter vários fatores que influenciam qual o melhor tipo de modelo de AM escolher [Mahesh 2020]. Na imagem 1 podemos observar a família de algoritmos que a área a AM pode conter, podendo resolver uma vasta gama de problemas.

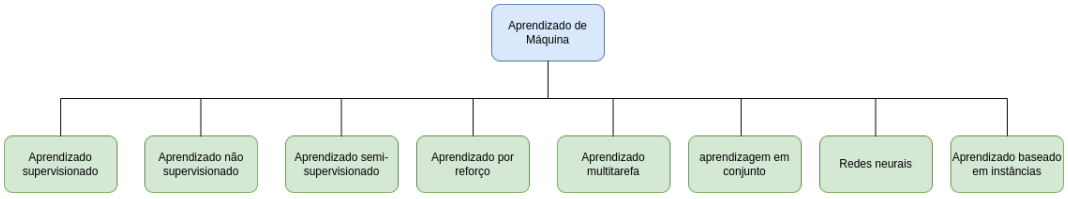


Figura 1. Família de algoritmos pertencentes a área do aprendizado de máquina.

3.2.1. Floresta Aleatória

Nesse algoritmo supervisionado, como o próprio nome sugere, ele é formado por um grande número de árvores de decisão individuais, onde operam em conjunto [Breiman 2001]. A ideia geral de utilizar uma floresta de árvores, é a combinação geral dos resultados dos modelos que aumentam o resultado geral. A floresta aleatória adiciona aleatoriedade adicional ao modelo, na medida que as árvores crescem [Probst et al. 2019]. Ao invés de sempre procurar o melhor recurso ao dividir o nó, ele procura o recurso mais importante entre um subconjunto aleatório de recursos. Como cada árvore no conjunto é treinada em um subconjunto aleatório do conjunto de treinamento geral, o conjunto como um todo tem menos probabilidade de super ajustar o conjunto de treinamento [Brophy and Lowd 2021].

3.2.2. Máquina de Vetor de Suporte

A Máquina de Vetor de Suporte (SVM, do inglês *Support Vector Machine*) é um algoritmo de máquina supervisionado utilizado principalmente para problemas de classificação. Por motivos de simplicidade e flexibilidade, é um método que pode abordar uma série de problemas, proporcionando um equilibrado desempenho preditivo. No entanto, a SVM é matematicamente complexa e computacionalmente pesado [Christmann and Steinwart 2008]. O SVM funciona criando um hiperplano ou diversos hiperplanos em um espaço de alta dimensão, e o melhor hiperplano é o que divide os dados de maneira otimizada em diferentes classes com maior separação entre elas [Zhou et al. 2021]. Quando utilizado para a classificação não linear, o SVM usa várias funções do *kernel* para estimar as margens, assim, maximizando as margens entre os hiperplanos [Ahmad et al. 2018a].

3.2.3. Perceptron de Multicamadas

O Perceptron de multicamadas (MLP, do inglês *Multilayer Perceptron*) é uma rede neural totalmente conectada com pelo menos três camadas: uma camada de entrada; uma ou mais camadas ocultas e uma camada de saída [Ahmadlou et al. 2021]. A regra de propagação é dada pelo produto interno das entradas ponderadas pelos pesos com adição do termo bias, e a saída da camada anterior é a entrada da camada atual, dessa forma, todas as camadas estão conectadas umas nas outras [Šegota et al. 2021]. Cada camada está alimentando a próxima com o resultado de sua computação. Esse processo percorre todo o caminho através das camadas ocultas até chegar na camada de saída [Lorencin et al. 2020]. A atualização dos pesos da rede neural ocorre com o algoritmo de retropropagação, onde com base no cálculo do erro obtido pela camada de saída da rede neural, recalcula o valor dos pesos da última camada de neurônios e procede para as camadas anteriores, no fluxo de trás para frente [Car et al. 2020].

3.2.4. *XGBoost*

O nome do algoritmo vem do inglês *Extreme Gradient Boosting* (Aumento de Gradiente Extremo), e trata-se de um método baseado em árvores de decisão com aumento de gradiente. Dessa forma, o *XGBoost* é capaz de combinar resultados de muitos classificadores ”fracos”, utilizando a técnica de aumento de gradiente, formando assim um forte modelo [Chen and Guestrin 2016]. O aumento de gradiente é uma técnica no qual novos modelos são criados para prever os resíduos de modelos anteriores e depois adicionados para fazer a previsão final. Dessa forma, ele usa um algoritmo de descida de gradiente para minimizar a perda ao adicionar novos modelos [Ogunleye and Wang 2019].

3.3. Reconhecimento Óptico de Caracteres

O reconhecimento óptico de caracteres (OCR, do inglês *Optical Character Recognition*) é uma técnica amplamente usada para reconhecer textos em imagens, como fotos e documentos digitalizados [Patel et al. 2012]. Assim, mesmo que muitos documentos já sejam digitais, ainda sim existem documentos que estão apenas impressos, na forma de imagem, e essa tecnologia ajudaria a identificar os textos contidos nas impressões digitalizadas.

O OCR transforma a imagem obtida em um conteúdo legível e editável de letras, palavras ou frases, sendo bastante similar ao que estava no documento original [Memon et al. 2020]. Por exemplo, se digitalizar um documento ou uma fotografia com uma impressora, ela possivelmente irá gerar um arquivo como uma imagem digital. Esse arquivo pode assumir alguns tipos de formatos, porém esse novo arquivo ainda pode ser apenas uma imagem do documento original, sem a possibilidade de editá-lo. Dessa forma, com a técnica de OCR, poderia carregar esse documento gerado pela impressora em um *software* OCR. O programa irá identificar os símbolos e textos contidos no documento, e transformará em um texto editável. Esse exemplo pode ser visto através da figura 2.

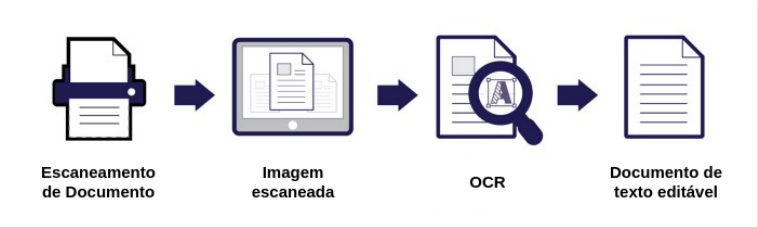


Figura 2. Processo de extração de texto com a técnica de OCR.

3.4. Otimização de hiperparâmetros

A otimização de hiperparâmetros refere-se à busca automática dos hiperparâmetros de um modelo de Aprendizado de Máquina. Os hiperparâmetros são todas os parâmetros configuráveis de um modelo, essas opções não são atualizados durante o processo de treinamento e estão diretamente ligadas na construção do modelo [Feurer and Hutter 2019]. Assim, esse método testa várias combinações onde busca-se aquela com melhor desempenho no processo de treinamento do modelo.

O objetivo de uma busca otimizada de hiperparâmetros é: reduzir o trabalho do desenvolvedor de IA; melhorar a precisão e eficiência do treinamento dos modelos; tornar

a escolha do conjunto mais favorável de hiperparâmetro mais convincente e os resultados dos treinamentos mais reprodutíveis [Yang and Shami 2020]. A otimização de hiperparâmetros demanda uma grande quantidade de recursos computacionais, principalmente quando vários hiperparâmetros são otimizados juntos [Yu and Zhu 2020].

Existem diversas abordagens na literatura para a busca otimizada de hiperparâmetros. Onde tem-se como objetivo em comum, achar o melhor conjunto de recursos para um determinado modelo. A partir da literatura, podemos citar como exemplo: pesquisa manual, o desenvolvedor de IA irá definir manualmente os parâmetros da rede; pesquisa em grade, realiza uma busca exaustiva nos conjuntos de hiperparâmetros especificados [Liashchynskyi and Liashchynskyi 2019]; pesquisa aleatória, diferente da pesquisa em grade que irá buscar em todas as combinações de hiperparâmetro, verificando um número fixo de combinações aleatoriamente [Bergstra and Bengio 2012]; otimização bayesiana, os métodos bayesianos tentam construir uma função que estima quão bom seu modelo pode ser para uma certa escolha de hiperparâmetros [Frazier 2018].

3.5. Processamento de linguagem natural

O processamento de linguagem natural (NLP, do inglês *Natural Language Processing*) trata-se da área da inteligência artificial que tem como o objetivo em dar aos computadores a habilidade de entender textos e palavras faladas, da mesma maneira que os seres humanos [Chowdhary 2020]. A NLP fundamenta-se em muitas áreas, incluindo a ciência da computação e linguística computacional, com o objetivo de buscar preencher a janela entre a comunicação humana e a compreensão do computador.

Com a utilização da NLP, foi possível realizar uma série de tarefas que condiz ao processamento de texto e fala. Dessa forma, existem diversas aplicações que utilizam essas técnicas, como: detecção de spam [Ora 2020], utilizada geralmente em filtragem de emails; máquina de tradução, tendo o serviço mais conhecido como o Google Tradutor; agentes virtuais e chatbots [Vasilateanu et al. 2018], [Ahmad et al. 2018b], geralmente utilizados para automatizar os atendimentos por telefone e chat na internet; análise de sentimentos [Hussein 2018], onde tem o objetivo de identificar o humor ou opiniões em grandes quantidades de texto, realizando a mineração de opinião; resumo de textos [Mishra et al. 2014], onde pode digerir uma grande quantidade de dados textuais, e criar resumos e sinopses, uma versão menor contendo a ideia chave principal do texto.

3.5.1. Tratamentos dos dados textuais

Para poder modelar a língua e os dados textuais de um determinado problema, são necessários pré-processamentos que abstraem e estruturam a linguagem, deixando somente a informação relevante para o entendimento da máquina [Vijayarani et al. 2015]. O objetivo deste pré-processamento é reduzir as características textuais, dado que um texto pode haver uma grande quantidade de características, removendo informações que serão irrelevantes ou redundantes no processo de classificação textual. Nos Tratamentos dos dados, existem dois processos fundamentais utilizados nesse projeto: limpeza e pré-processamento dos dados.

- **Limpeza dos dados**: Nessa fase de tratamento dos dados, o objetivo é realizar uma filtragem nos textos que compõem a base de dados. Para isso, é necessário

a utilização de algumas técnicas que removam alguns caracteres que possam influenciar negativamente no processo de classificação. Dentre as diversas técnicas na literatura para limpeza de dados textuais, neste trabalho foram utilizados os seguintes processos: Remoção de caracteres/palavras desconhecido; Remoção de pontuação; Remoção de *URLs*/*Tags* e Remoção de palavras raras.

- **Pré-processamento dos dados**: Nessa fase, tem como objetivo a diminuição das características textuais. Nesse processo de tratamento, irá deixar o texto mais ”legível”para a máquina. Portanto, extrair as ideias principais contidas nos textos é o principal objetivo dessa fase de pré-processamento. Neste projeto, foram utilizadas as seguintes técnicas de pré-processamento, dentre as diversas contidas na literatura: Remoção de *stopwords* (elementos do texto que geralmente possuem “pouca informação”); *Stemming* (reduz a palavra à seu radical); Extração de palavras-chave (extrair as informações mais relevantes do texto) e Codificação TF-IDF (avalia a frequência local e a frequência geral de um determinado termo, além de avaliar um fator de ponderação de forma para os que mais aparecem nos documentos tenham uma representatividade menor).

3.6. Métricas de avaliação

Ao desenvolver projetos de aprendizado de máquina, é crucial a utilização de métricas que possam medir o desempenho do modelo projetado [Dalianis 2018]. Essas técnicas de avaliação estatística são utilizadas logo no término da construção e treinamento do modelo. Portanto, existem na literatura vários métodos que possam medir a performance para diferentes tipos de algoritmos de aprendizado de máquina [Hossin and Sulaiman 2015].

3.6.1. Validação cruzada *K-Fold*

A técnica de validação cruzada *k-fold*, é utilizada quando se espera avaliar a capacidade de generalização do modelo, dado um conjunto de dados. A partir desse método, ele tem altas chances de detectar se o seu modelo está realmente ajustado aos seus dados de treinamento, ou seja, sofrendo *overfitting* [Berrar 2019]. Esse método possui apenas um único parâmetro a ser determinado, chamado de k. Esse parâmetro k, refere-se ao número de grupos que uma determinada base de dados será divida. Por exemplo, se $k = 10$, então quer dizer que a base de dados será dividida em 10 grupos.

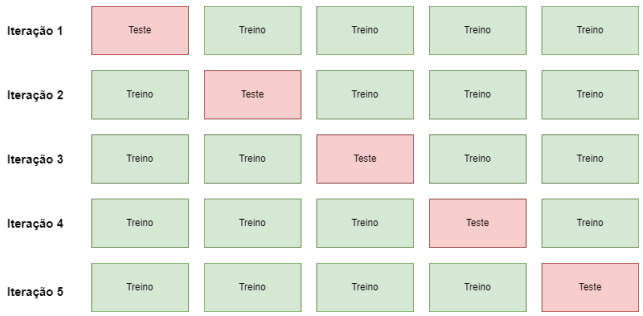


Figura 3. Técnica de validação cruzada *K-Fold*, utilizando $k = 5$.

Na figura 3 é possível ver como funciona a utilização dessa técnica de validação cruzada. Esse método consiste em dividir a base de dados em um número de k partes,

aproximadamente na mesma quantidade de amostra para cada parte. Em cada iteração, as partes de treino e teste será mudado para medir a capacidade de generalização do modelo. Dessa forma, garante que cada subconjunto será utilizado para teste em algum momento da avaliação do modelo.

3.6.2. Precisão

A precisão é umas das métricas mais utilizadas na literatura para medir a performance de um modelo de aprendizado de máquina [Russo et al. 2018]. Nesse método, ele considera os exemplos classificados como pertencentes a uma classe, que realmente são daquela classe (verdadeiros positivos), dividido pelo número de exemplos classificados nesta classe, mas que pertencem a outras (falsos positivos) [Davis and Goadrich 2006]. A equação pode ser vista logo abaixo:

$$precisao = \frac{VP}{VP + FP} \tag{1}$$

Na expressão acima, temos VP, que são os verdadeiros positivos, ou seja, classificados corretamente pelo modelo pertencendo a uma determinada classe. Enquanto FP expressa os falsos positivos, onde são os exemplos classificados em uma determinada classe que pertence a outra.

3.6.3. Recall

O *recall*, conhecido também como sensibilidade, é o cálculo da proporção dos verdadeiros positivos entre todas as observações que realmente são positivas no seu conjunto de dados [Miao and Zhu 2021]. Esta métrica é definida pela razão entre a quantidade de exemplos classificados corretamente como positivos e a quantidade de exemplos que são de fato positivos, de acordo com a expressão abaixo:

$$recall = \frac{VP}{VP + FN} \tag{2}$$

Na expressão acima, que define como o *recall* é calculado, temos VP sendo os exemplos classificados positivamente pelo modelo, onde sendo realmente positivo. Em FN, define os exemplos que foram classificados negativamente, onde sendo de classe positiva.

3.6.4. Medida F1

A medida F1 é um método que utiliza tanto a métrica de precisão quanto também o *recall*. Portanto, ela é definida pela média harmônica entre essas duas métricas. Abaixo pode ser vista sua equação.

$$F1 = 2 * \frac{precisao * recall}{precisao + recall} \tag{3}$$

Nessa medida, a partir da média harmônica, se a precisão ou o *recall* for baixo, isso implicará diretamente no resultado da medida F1. Quando se tem um resultado alto da medida F1, isso quer dizer que tanto a precisão quanto o *recall* foram altas [Dalianis 2018].

4. Metodologia

Nesse capítulo da metodologia, será apresentado os métodos e passos que irão compor o projeto. Portanto, foi necessário 6 passos primordiais: (*i*) Construção e preparação da base de dados; (*ii*) pré-processamento dos dados dos documentos; (*iii*) Análise de expressões regulares para filtrar o documento pelo seu título; (*iv*) implementação de classificadores disponíveis na literatura para a tarefa de classificação dos documentos institucionais; (*v*) implementar e analisar as métricas de avaliação e (*vi*) desenvolvimento de uma API. Na figura 4, pode ser vista todo o fluxo do sistema proposto e discutido no decorrer desta seção.

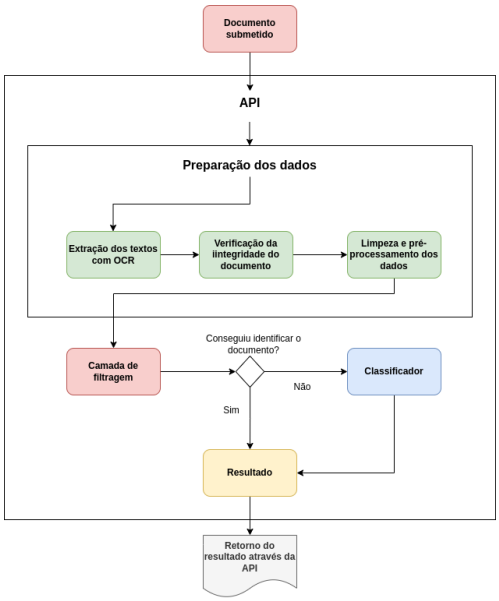


Figura 4. Funcionamento do sistema da API no processo de classificação de documentos.

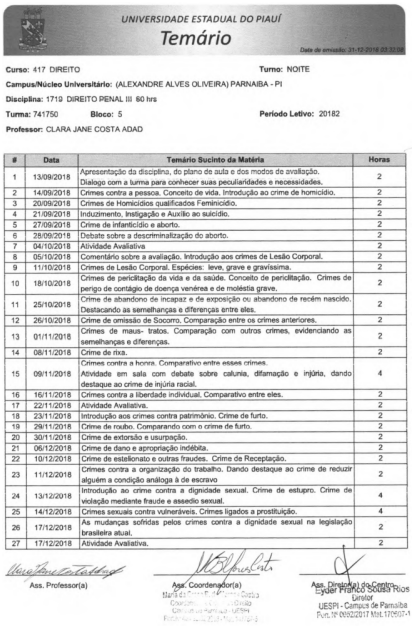
4.1. Criação da base de dados

Nessa seção será apresentado os passos para a construção da base de dados, mostrando como serão organizados os documentos, assim como o processo de extração dos textos com a técnica de OCR e como irá acontecer a rotulação para cada classe. A partir da conclusão de todos os passos, será possível conseguir uma base de dados pronta para a utilização das técnicas de limpeza e pré-processamento dos dados, preparando para a fase de treinamento dos modelos propostos.

A base de dados é composta de documentos institucionais provenientes da UESPI. Nela, possui documentos relacionados a informações acadêmicas sobre os alunos, dividido em diferentes tipos de categorias. Inicialmente, a base de dados veio composta por documentos dispostos em pastas, de acordo com seu ano de criação e seu rótulo. Todos os documentos estão em formato PDF, porém de forma escaneada, ou seja, uma imagem com a foto original do documento.

4.1.1. Processamento das imagens dos documentos

Inicialmente, antes da utilização das técnicas de OCR para a extração dos textos das imagens, a página do documento passa por um processamento de imagem, a fim de garantir melhor qualidade da imagem e ter uma melhor performance na utilização do OCR. Os processamentos realizados nas páginas dos documentos foram: *Median Blur* e *Otsu threshold*. O *Median Blur* tem o objetivo de remover alguns ruídos, especialmente o sal e pimenta, além de alguns arranhões. Por outro lado, o *Otsu threshold* tem o objetivo de deixar a zona de interesse em maior evidência, ou seja, destacar as palavras contida no documento. Nas figuras 5 e 6, pode ser vista o resultado desses processamentos.



UNIVERSIDADE ESTADUAL DO PIAUÍ

Temário

Curso: 417 DIREITO Turno: NOITE

Campus/Núcleo Universitário: (ALEXANDRE ALVES OLIVEIRA) PARNAIBA - PI

Disciplina: 1719 DIREITO PENAL III 60 hrs

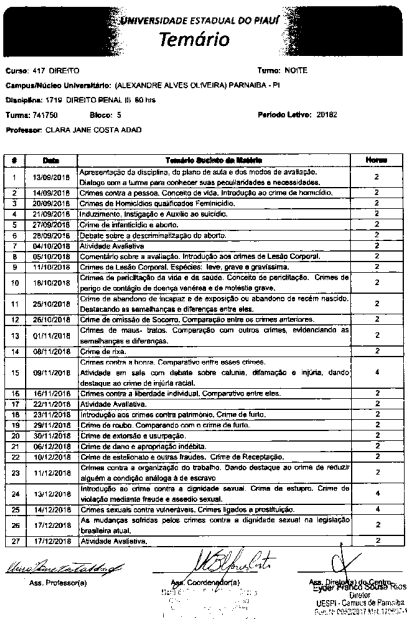
Turma: 741750 Bloco: 5 Período Letivo: 20182

Professor: CLARA JANE COSTA ADAD

#	Data	Temário Sucinto da Matéria	Horas
1	13/09/2018	Apresentação da disciplina, do plano de aula e dos modos de avaliação. Dialogo com a turma para conhecer suas peculiaridades e necessidades.	2
2	14/09/2018	Crimes contra a pessoa. Conceito de vida. Introdução ao crime de homicídio.	2
3	20/09/2018	Crimes de Homicídios qualificados Feminicídio.	2
4	21/09/2018	Induzimento, Instigação e Auxílio ao suicídio.	2
5	27/09/2018	Crime de infanticídio e aborto.	2
6	28/09/2018	Debate sobre a descriminalização do aborto.	2
7	04/10/2018	Atividade Avaliativa.	2
8	05/10/2018	Comentário sobre a avaliação. Introdução aos crimes de Lesão Corporal.	2
9	11/10/2018	Crimes de Lesão Corporal. Espécies: leve, grave e gravíssima.	2
10	18/10/2018	Crimes de periclitação da vida e da saúde. Conceito de periclitação. Crimes de perigo de contágio de doença venérea e de moléstia grave.	2
11	25/10/2018	Crime de abandono de incapaz e de exposição ou abandono de recém nascido. Destacando as semelhanças e diferenças entre eles.	2
12	26/10/2018	Crime de omissão de Socorro. Comparação entre os crimes anteriores.	2
13	01/11/2018	Crimes de maus- tratos. Comparação com outros crimes, evidenciando as semelhanças e diferenças.	2
14	08/11/2018	Crime de rixa.	2
15	09/11/2018	Crimes contra a honra. Comparativo entre esses crimes. Atividade em sala com debate sobre calunia, difamação e injúria, dando destaque ao crime de injúria racial.	4
16	16/11/2018	Crimes contra a liberdade individual. Comparativo entre eles.	2
17	22/11/2018	Atividade Avaliativa.	2
18	23/11/2018	Introdução aos crimes contra patrimônio. Crime de furto.	2
19	29/11/2018	Crime de roubo. Comparando com o crime de furto.	2
20	30/11/2018	Crime de extorsão e usurpação.	2
21	06/12/2018	Crime de dano e apropriação indébita.	2
22	10/12/2018	Crime de estelionato e outras fraudes. Crime de Receptação.	2
23	11/12/2018	Crimes contra a organização do trabalho. Dando destaque ao crime de reduzir alguém a condição análoga à de escravo	2
24	13/12/2018	Introdução ao crime contra a dignidade sexual. Crime de estupro. Crime de violação mediante fraude e assedio sexual.	4
25	14/12/2018	Crimes sexuais contra vulneráveis. Crimes ligados a prostituição.	4
26	17/12/2018	As mudanças sofridas pelos crimes contra a dignidade sexual na legislação brasileira atual.	2
27	17/12/2018	Atividade Avaliativa.	2

Ass. Professor(a) Ass. Coordenador(a) Ass. Diretor(a) do Centro: Eyder Franco Sousa Rios, Diretor, UESPI - Campus de Parnaíba

Figura 5. Documento antes do processamento.



UNIVERSIDADE ESTADUAL DO PIAUÍ

Temário

Curso: 417 DIREITO Turno: NOITE

Campus/Núcleo Universitário: (ALEXANDRE ALVES OLIVEIRA) PARNAIBA - PI

Disciplina: 1719 DIREITO PENAL III 60 hrs

Turma: 741750 Bloco: 5 Período Letivo: 20182

Professor: CLARA JANE COSTA ADAD

#	Data	Temário Sucinto da Matéria	Horas
1	13/09/2018	Apresentação da disciplina, do plano de aula e dos modos de avaliação. Dialogo com a turma para conhecer suas peculiaridades e necessidades.	2
2	14/09/2018	Crimes contra a pessoa. Conceito de vida. Introdução ao crime de homicídio.	2
3	20/09/2018	Crimes de Homicídios qualificados Feminicídio.	2
4	21/09/2018	Induzimento, Instigação e Auxílio ao suicídio.	2
5	27/09/2018	Crime de infanticídio e aborto.	2
6	28/09/2018	Debate sobre a descriminalização do aborto.	2
7	04/10/2018	Atividade Avaliativa	2
8	05/10/2018	Comentário sobre a avaliação. Introdução aos crimes de Lesão Corporal.	2
9	11/10/2018	Crimes de Lesão Corporal. Espécies: leve, grave e gravíssima.	2
10	18/10/2018	Crimes de periclitação da vida e da saúde. Conceito de periclitação. Crimes de perigo de contágio de doença venérea e de moléstia grave.	2
11	25/10/2018	Crime de abandono de incapaz e de exposição ou abandono de recém nascido. Destacando as semelhanças e diferenças entre eles.	2
12	26/10/2018	Crime de omissão de Socorro. Comparação entre os crimes anteriores.	2
13	01/11/2018	Crimes de maus- tratos. Comparação com outros crimes, evidenciando as semelhanças e diferenças.	2
14	08/11/2018	Crime de rixa.	2
15	09/11/2018	Crimes contra a honra. Comparativo entre esses crimes. Atividade em sala com debate sobre calunia, difamação e injúria, dando destaque ao crime de injúria racial.	4
16	16/11/2018	Crimes contra a liberdade individual. Comparativo entre eles.	2
17	22/11/2018	Atividade Avaliativa.	2
18	23/11/2018	Introdução aos crimes contra patrimônio. Crime de furto.	2
19	29/11/2018	Crime de roubo. Comparando com o crime de furto.	2
20	30/11/2018	Crime de extorsão e usurpação.	2
21	06/12/2018	Crime de dano e apropriação indébita.	2
22	10/12/2018	Crime de estelionato e outras fraudes. Crime de Receptação.	2
23	11/12/2018	Crimes contra a organização do trabalho. Dando destaque ao crime de reduzir alguém a condição análoga à de escravo	2
24	13/12/2018	Introdução ao crime contra a dignidade sexual. Crime de estupro. Crime de violação mediante fraude e assedio sexual.	4
25	14/12/2018	Crimes sexuais contra vulneráveis. Crimes ligados a prostituição.	4
26	17/12/2018	As mudanças sofridas pelos crimes contra a dignidade sexual na legislação brasileira atual.	2
27	17/12/2018	Atividade Avaliativa.	2

Ass. Professor(a) Ass. Coordenador(a) Ass. Diretor(a) do Centro: Eyder Franco Sousa Rios, Diretor, UESPI - Campus de Parnaíba

Figura 6. Documento após passar pelo processamento.

4.1.2. Rotulação dos documentos e extração dos textos com OCR

A primeira parte do processo de construção da base de dados, será pelo rastreamento dos caminhos de diretório de cada documento. Com isso, será possível associar qual a classe e onde estava localizado cada arquivo da base de dados, já que a pasta onde está o documento contém seu respectivo rótulo, além do ano de criação. Entretanto, apesar dos dados está organizados pelos seus rótulos, ainda continha diversas pastas com relação à data de criação, onde um mesmo tipo de rótulo está espalhado pelas pastas de data de criação. A partir desse processo, o rastreamento dos caminhos irá facilitar bastante na localização dos documentos.

Após ter todos os documentos com seus respectivos caminhos, a segunda parte do processo se dá pela extração dos textos que contém cada documento escaneado. Nesse processo, irá utilizar a técnica de OCR, como mostrado na seção 3.3, para reconhecimento de caracteres. Foi usado o software *Tesseract* para a utilização da técnica de OCR, disponibilizado na linguagem Python, onde tem um papel fundamental no reconhecimento dos

textos nos documentos. Com a extração dos textos dos documentos digitalizados, foi gerado um arquivo do tipo CSV, onde ficam todas as informações textuais dos documentos, juntamente com suas classes.

Logo depois de gerar o arquivo final contendo todas as informações textuais dos documentos, juntamente com suas classes, será realizada uma verificação de integridade do documento, tanto no processo de preparação da base de dados, como também de forma automatizada no fluxo da API do sistema. Isso deve-se ao fato que existem documentos com algum tipo de rasura, normalmente devido seu ano de criação ser antigo. Assim, já que seu conteúdo não está legível, será realizada uma limpeza nesses tipos de documentos, deixando somente os que possuem seus dados compreensíveis. Após todos esses processamentos, a base de dados está pronta para ser analisada.

4.1.3. Análise da base de dados

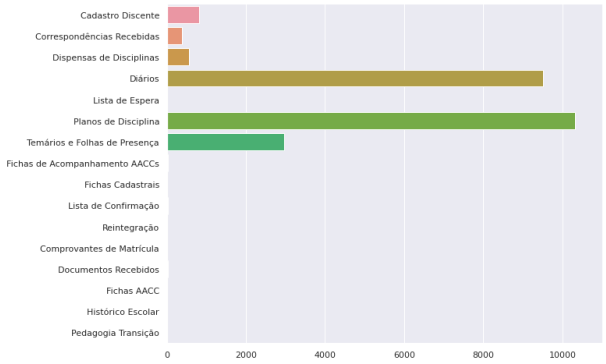


Figura 7. Base de dados com relação ao número de amostra para cada classe.

A base de dados final possui todos os documentos na qual é possível extrair suas informações a partir da técnica de OCR. Nela, contém vários rótulos que classificam os documentos. Entretanto, possuíam classes que tinham poucas amostras. Dessa forma, ao passar para a fase de classificação com essas classes possuindo poucas amostras, iria afetar a performance do classificador, em razão do grande desbalanceamento no banco de dados. Foi necessário, para balancear a base de dados, retirar os rótulos na quais não possuía uma boa quantidade de amostras, como pode ser vista de acordo com a figura 7.

A primeira tarefa no processo de análise da base de dados será rastrear quais classes serão utilizadas para a tarefa de classificação. Após ter estudado quais classes compõem a base de dados, será feita uma análise com relação ao número de amostras para cada rótulo. Dessa forma, é possível estimar se a base de dados está balanceada para o processo de treinamento. Como visto em análises prévias, há classes que possuem um pequeno número de amostras, onde será necessário removê-las, a fim de balancear o conjunto de dados.

Em uma análise acerca da base de dados, foi possível perceber quais classes irão ser utilizadas no processo de classificação dos documentos. Infelizmente, não será possível utilizar todas as classes constituintes no conjunto de documentos, pois não há amostra suficiente para o modelo classificador poder aprender e fazer a generalização dos

dados. Em um primeiro estudo, como visto na figura 7, é possível observar as possíveis classes que farão parte do processo de treinamento, sendo elas: Cadastro Discente, correspondências recebidas, dispensas de disciplinas, diários, planos de disciplina, temários e folhas de presença.

4.2. Treinamento e testes dos modelos de IA

Nessa seção, será mostrados os passos tomados para o processo de treinamento e testes de modelos de IA. Nesta etapa, ocorrerá a preparação da base de dados, treinamento dos modelos com a técnica de otimização de hiperparâmetros, construção da camada de filtragem a partir de expressões regulares, e o processo que o sistema classifica um documento.

4.2.1. Preparação da base de dados

Com a criação da base de dados, como mostrados os passos a serem tomados na seção 4.1, nesse processo de preparação dos dados, serão aplicados as técnicas de NLP, como vistas na seção 3.5, em todos os documentos que compõem o conjunto de dados. Primeiramente, os dados precisa passar pela fase de limpeza (dada alguns conceitos mostrados na seção 3.5.1), onde serão aplicadas as técnicas de remoção de caracteres ou palavras desconhecidas; remoção de qualquer tipo de pontuação; remoção de *tags* e *URLs* que esteja presente em algum documento, além de remoção de palavras raras onde tem somente uma ocorrência.

Após a fase de limpeza dos dados, será realizado as técnicas de pré-processamento dos textos (falado um pouco dos conceitos na seção 3.5.1), voltado mais para o treinamento dos modelos. Nessa fase, se dá pela aplicação dos métodos de remoção de *stopwords*, realização da técnica de *stemming* nos textos contido nos documentos, extração das palavras-chave e aplicação da codificação TF-IDF. A codificação TF-IDF ocorre na última parte do processo de pré-processamento, pois após realizar todas as técnicas de limpeza e pré-processamento, será originada a matriz de característica a partir do cálculo do método de codificação TF-IDF.

No processo de treinamento dos modelos, após gerado a matriz de característica dos documentos da base de dados, pretende-se dividir os dados em treinamento e teste. Nesse processo de divisão, será destinado 70 % para o treinamento do classificador, enquanto o resto de 30 %, será destinado para o processo de testes, de acordo com as métricas de avaliação do modelo [Saraiva et al. 2019]. Ao dividir a base de dados, a realização dos testes acontece com dados que não foram ”visto”no processo de treinamento, medindo a capacidade de generalização do modelo. Após todas essas etapas vistas nessa fase de preparação, a base de dados está pronta para ser utilizada nos treinamentos dos classificadores.

4.2.2. Treinamento com otimização de hiperparâmetros

Para o treinamento dos modelos nesse projeto, será com o auxílio de algumas das técnicas de otimização de hiperparâmetros como mostrado na seção 3.4. As técnicas de otimização irão buscar o melhor conjunto de hiperparâmetros, dado uma distribuição de valores para

cada variável de configuração de um determinado modelo. Os modelos que foram implementados e testados, como mostrados um pouco dos conceitos na seção 3.2 para a classificação de documentos foram: Floresta Aleatória, Máquina de Vetor de Suporte, *XGBoost* e Perceptron de Multicamadas.

Para cada classificador, foi configurado um dicionário contendo uma lista de possíveis valores de hiperparâmetros, para a busca através da técnica escolhida de otimização de hiperparâmetro. O treinamento realizado ocorreu com algumas configurações da base de dados, a fim de testar qual abordagem se sairia melhor na classificação dos documentos. Primeiramente, foram criados dois tipos de configurações da base de dados, a fim de conseguir obter melhor performance dos modelos implementados. Para cada uma das configurações, foi realizado o treinamento de todos os classificadores propostos, utilizando a técnica de otimização de hiperparâmetros. Cada configuração ficou da seguinte forma: Documentos completos, com todas as páginas e Documentos possuindo somente a primeira página.

Para o treinamento utilizando o otimizador de hiperparâmetros, foi realizado com o auxílio da biblioteca do *scikit-learning*. O método de otimização escolhido foi o *Random Search* (Pesquisa Aleatória), como mostrado na seção 3.4, ele verifica um número fixo de combinações aleatoriamente, dado um número máximo de iterações. A partir da busca desses hiperparâmetros, resultará na melhor configuração encontrada para cada modelo proposto. A escolha de cada parâmetro para realizar o ajuste através da otimização, foi através da documentação dos modelos, disposta na página das bibliotecas: *scikit-learning* (encontra todos os modelos, exceto o *XGBoost*) e *XGBoost*.

Nas tabelas 1, 2, 3, 4, é possível observar os conjuntos de hiperpâmetros testados para cada modelo implementado nesse projeto, a fim de obter o classificador com melhor desempenho na tarefa de classificação de documentos. É possível observar que para cada um dos modelos, há uma lista de parâmetros que diferem uns dos outros na sua construção, por isso a necessidade da definição manual da lista de configurações para cada um. Foi testado e mesclados várias configurações dentre as dispostas nas tabelas, de acordo com o otimizador, a fim de encontrar a melhor entre elas. Nas tabelas de hiperparâmetros de cada modelo, tem o resultado de melhor conjunto para cada tipo de configuração da base de dados.

Tabela 1. Tabela de hiperparâmetros do modelo Máquina de Vetor de Suporte.

Máquina de Vetor de Suporte			
	C	**gamma**	**kernel**
	0.1	1	poly
	1	0.1	rbf
	10	0.01	sigmoid
	100	0.001	-
	1000	0.0001	-
melhor conjunto config. 1	**100**	**0.01**	**sigmoid**
melhor conjunto config. 2	**1000**	**0.0001**	**rbf**

Tabela 2. Tabela de hiperparâmetros do modelo Floresta Aleatória.

Floresta Aleatória							
	max_depth	**n_estimators**	**min_samples_split**	**min_samples_leaf**	**max_features**	**criterion**	**oob_score**
	80	160	2	1	auto	gini	True
	90	180	3	2	sqrt	entropy	False
	100	200	4	3	log2	-	-
	110	220	5	4	None	-	-
	120	240	6	5	-	-	-
	150	260	-	-	-	-	-
	180	280	-	-	-	-	-
melhor conjunto config. 1	**150**	**260**	**5**	**3**	**None**	**entropy**	**False**
melhor conjunto config. 2	**90**	**220**	**4**	**2**	**sqrt**	**entropy**	**True**

Tabela 3. Tabela de hiperparâmetros do modelo XGBoost.

XGBoost				
	max_depth	**learning_rate**	**n_estimators**	**colsample_bytree**
	3	0.01	100	0.3
	6	0.05	500	0.7
	10	0.1	1000	-
melhor conjunto config. 1	**3**	**0.1**	**500**	**0.7**
melhor conjunto config. 2	**6**	**0.1**	**500**	**0.3**

Tabela 4. Tabela de hiperparâmetros do modelo Perceptron de Multicamadas.

Perceptron de Multicamadas				
	hidden_layer_sizes	**activation**	**solver**	**learning_rate**
	2	identity	lbfgs	constant
	3	logistic	sgd	invscaling
	4	tanh	adam	adaptive
	5	relu	-	-
melhor conjunto config. 1	**5**	**tanh**	**lbfgs**	**adaptive**
melhor conjunto config. 2	**5**	**logistic**	**lbfgs**	**constant**

Todo o processo de treinamento realizado neste trabalho, ocorreu com o auxílio da ferramenta *MLFlow* de gerenciamento de modelos de AM. O *MLflow* é uma plataforma de código aberto para gerenciar o ciclo de vida de modelos de AM, incluindo experimentação, reprodutibilidade, implantação e um registro de um modelo de produção [Zaharia et al. 2018]. A partir dessa ferramenta, os testes feitos com os modelos se tornou bem mais prático, pois além de ter toda a assistência na criação dos modelos, todos os resultados estatísticos obtidos ficam armazenados para fácil comparação.

4.3. Camada de filtragem através de expressão regular

No processo para realizar a classificação dos documentos, já na parte do fluxo da aplicação, foi definido uma camada de filtragem, onde tem o objetivo de rastrear o título presente no documento, após os processos de preparação e limpeza da base de dados. Essa camada funciona percorrendo o documento de entrada e através de expressões regulares, busca pelo título no corpo do documento. Para cada classe, existem títulos que, ao longo dos anos, foram sofrendo alterações. Dessa forma, foram analisados os títulos que, frequentemente, mais apareciam na base de dados para cada tipo de documento.

Na figura 8, é possível observar a distribuição de documentos identificados pela camada de filtragem. Em comparação com a distribuição total da base de dados, é visível que não foi possível detectar todos os documentos de acordo com seus títulos. Na classe Cadastro de Discente, por exemplo, quase todos os documentos foram identificados com sucesso pela camada de expressão regular, restando poucas amostras que não foram identificadas. Na classe Correspondências Recebidas, por sua vez, nenhum documento foi detectado pela camada, isso devido ao documento não possuir um título padrão que o identifique, assim, é necessário analisar o conteúdo restante do documento para identificar sua classe.

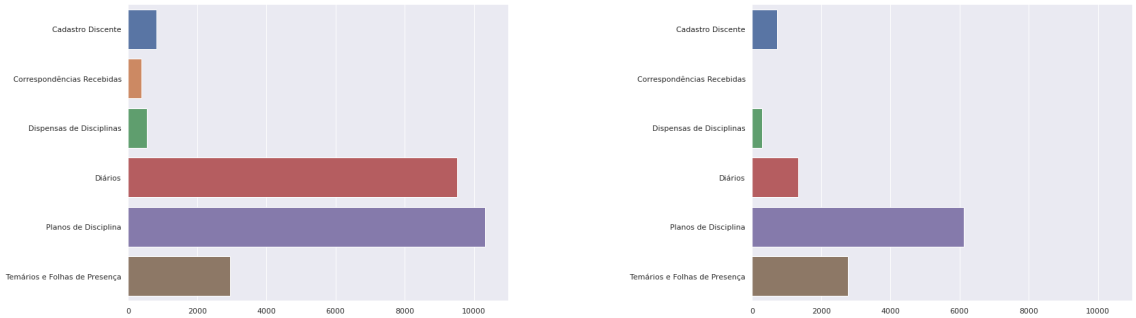


Figura 8. (A) Distribuição do amostras na base de dados. (B) Distribuição de documentos identificados pela camada de filtragem.

Na camada de filtragem, nem sempre vai conseguir encontrar o documento pelo título, pois há ocorrências em que o título do documento não é corretamente identificado devido a alguns fatores: erros na detecção através da técnica de OCR, como título do documento com rasuras; erro de ortografia (presente em documentos antigos) e variações nos títulos. A partir desses problemas para a detecção através da camada de filtragem, é necessário analisar e identificar o texto através do conteúdo presente no corpo do documento com técnicas de AM. Dessa forma, as técnicas de AM vem como garantia para que o documento possa ser corretamente classificado.

4.4. Classificação dos documentos

A partir das definições dos processos para realizar a classificação dos documentos, o documento terá que seguir uma série de passos, como: rastreamentos dos textos através de técnica de ORC; limpeza e pré-processamento dos textos; camada de filtragem através de expressões regulares e modelo de AM para realizar a classificação, caso a camada de

filtragem não consiga rastrear o título. Na figura 9, pode ser visto os passos para um documento ser classificado.

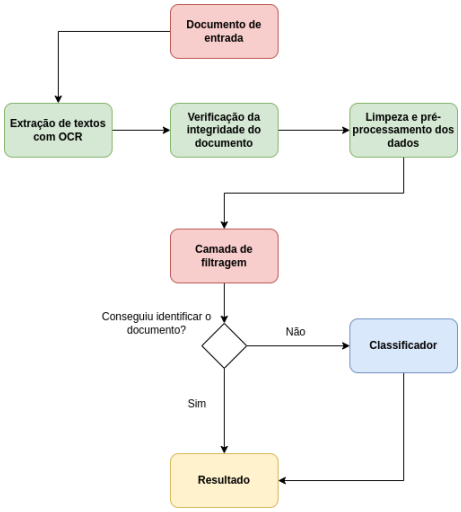


Figura 9. Passos que o documento terá que seguir para poder ser categorizado pelo classificador.

4.5. Criação da API

Para a criação da API, foi utilizado o *framework FastAPI*, disponibilizado para a linguagem Python. A escolha do *FastAPI* veio por ser intuitivo, rápido e fácil para se criar uma API. Além disso, ele possui um sistema de documentação e testes de requisições que torna bem prático o seu uso. A API proposta neste trabalho possui somente um *endpoint* de requisição, com o objetivo de ser fácil e objetivo o seu uso.

Na criação da API, terá que ser integrado todo os processos, assim como mostrado seção 4.4. Portanto, na figura 4, pode ser vista todos os processos que um documento de entrada será submetido na API. Primeiramente, o documento é submetido na aplicação, logo depois ele passa pelo processo de preparação dos dados, e segue para a camada de filtragem, caso não encontre o tipo de documento pela filtragem, ele continua o fluxo seguindo para o classificador, onde irá categorizar o documento. Após obter a resposta de qual o rótulo do documento, a API retorna com o resultado obtido.

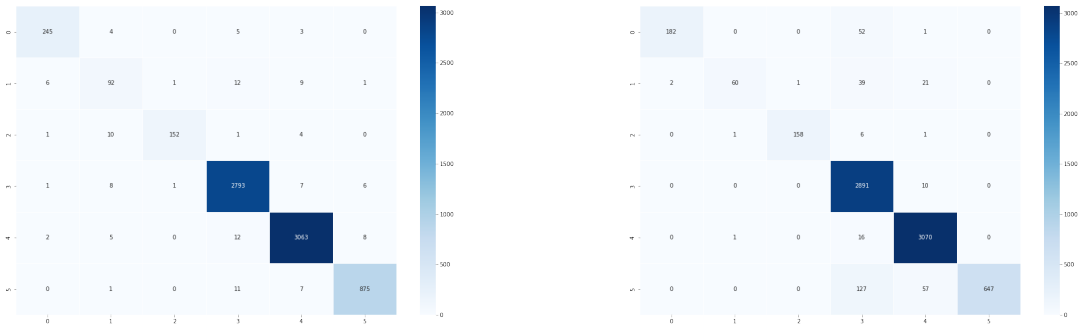
5. Resultados

Neste capítulo, serão apresentados os resultados obtidos com esse projeto. Primeiramente, foi construído a base de dados a partir dos dados obtidos nos documentos, para que possa ficar preparado para a fase de aplicação das técnicas de NLP. Após toda a fase de preparação dos dados, foi criada uma camada de filtragem, onde tem o objetivo de detectar o título do documento através de expressões regulares. Em seguida, foi o processo destinado ao treinamento e testes dos modelos de AM, onde foram extraídos os resultados estatísticos de performance das métricas de avaliação. Após a implementação dos modelos, finalmente, no último processo, foi a construção da API, utilizando todas as etapas desenvolvidas neste trabalho.

Nesta seção irá focar na descrição das métricas para todos os modelos testados. Dessa forma, busca-se comparar os diferentes modelos testados na tarefa de classificação

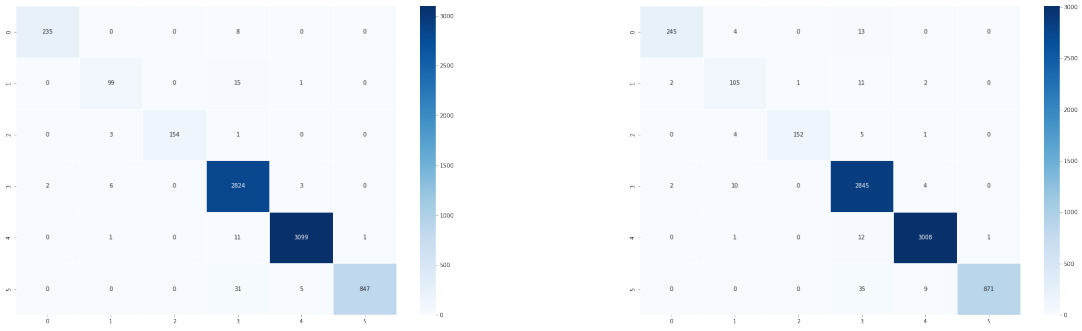
de documentos, a partir das métricas apresentadas na seção 3.6. A partir da análise das tabelas e gráficos de desempenho do modelo, fique claro qual classificador que será utilizado, dado seus resultados obtidos. Além disso, foi construído testes de configuração da base de dados, onde se estabeleceram da seguinte maneira: Documentos completos, com todas as páginas e Documentos possuindo somente a primeira página.

Nas figuras 10, 11, 12, 13, pode ser observado as matrizes de confusão para cada modelo testado neste projeto. A matriz de confusão exibe a distribuição dos registros em termos de suas classes atuais e de suas classes previstas, apontando os resultados corretos e errados. Na diagonal principal, é vista os acertos corretos para cada classe, dessa forma, fora da diagonal principal, tem os valores preditos errados pelo modelo. No eixo y, são os verdadeiros rótulos dos documentos, enquanto no eixo x, são os rótulos previstos pelo modelo. Dessa forma, tem-se, respectivamente, de 0 à 5: Cadastro Discente, Correspondências Recebidas, Dispensas de Disciplinas, Diários, Planos de Disciplina, Temários e Folhas de Presença.



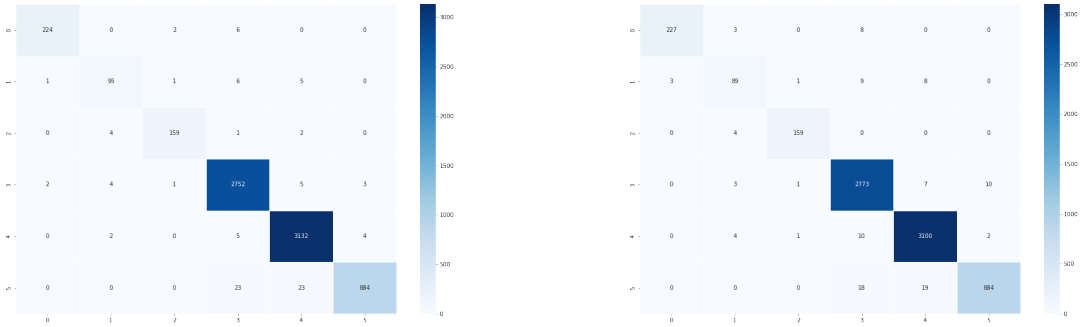
(A) (B)

Figura 10. Matriz de confusão do modelo Floresta aleatória. (A) é a configuração 1 e (B) é a configuração 2.



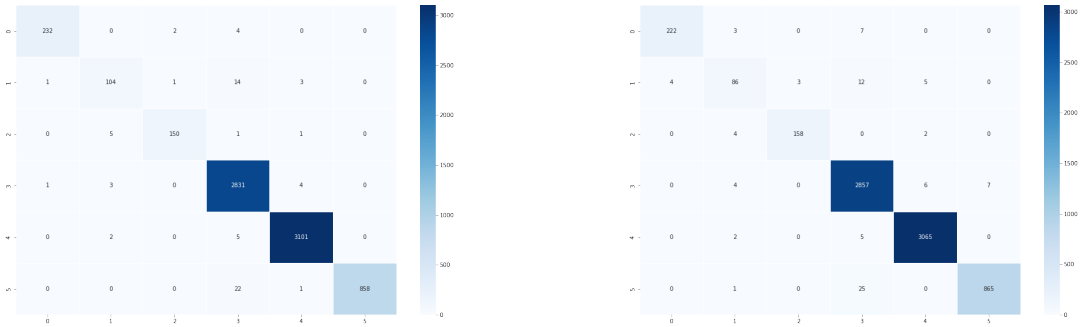
(A) (B)

Figura 11. Matriz de confusão do modelo Máquina vetor de suporte. (A) é a configuração 1 e (B) é a configuração 2.



(A) (B)

Figura 12. Matriz de confusão do modelo Perceptron de multicamadas. (A) é a configuração 1 e (B) é a configuração 2.



(A) (B)

Figura 13. Matriz de confusão do modelo XGBoost. (A) é a configuração 1 e (B) é a configuração 2.

Nas figuras da matriz de confusão para cada modelo, possui o resultado com as duas configurações diferentes sobre a base de dados, sendo A para configuração 1 (todas as páginas) e B para configuração 2 (somente primeira página). Em relação às configurações propostas, os resultados permaneceram bastantes equilibrados, possuindo somente uma pouquíssima vantagem para a configuração 1 da base de dados. É possível observar, em poucos casos, que em todos os modelos houve classes onde previu o documento erroneamente, principalmente no rótulo 3, para os documentos de Diários.

Nas tabelas 5 e 6, é possível observar os resultados das métricas de avaliação para cada modelo implementado. Na tabela 5, é os resultados da configuração da base de dados contendo todas as páginas do documento, enquanto na 6, é a configuração da base de dados apresentando somente a primeira página. É possível observar que os classificadores, em modo geral, obtiveram bom desempenho para classificar os documentos, sendo o Máquina de vetor de suporte, Perceptron de multicamadas e *XGBoost*, com melhores resultados e bem parecidos em suas pontuações.

Tabela 5. Tabela dos resultados das métricas de avaliação da config. 1.

Modelo	Precisão	Recall	Medida F1	Validação cruzada
Floresta Aleatória	0.9455	0.9300	0.9375	0.9842
Máquina de vetor de suporte	0.9788	0.9589	0.9686	0.9848
Perceptron de multicamadas	0.9722	0.9574	0.9646	0.9850
XGBoost	0.9775	0.9574	0.9671	0.9900

Tabela 6. Tabela dos resultados das métricas de avaliação da config. 2.

Modelo	Precisão	Recall	Medida F1	Validação cruzada
Floresta Aleatória	0.9742	0.8306	0.8844	0.9495
Máquina de vetor de suporte	0.9652	0.9471	0.9558	0.9844
Perceptron de multicamadas	0.9653	0.9475	0.9562	0.9813
XGBoost	0.9657	0.9441	0.9545	0.9885

Para a escolha do modelo, foram necessárias algumas questões com relação aos resultados dos modelos e das configurações propostas neste trabalho. Primeiro, as configurações 1 e 2, em relação aos resultados, ficaram bastante semelhantes para todos os classificadores. Assim, deve ser levado em consideração o melhor modelo quanto aos resultados e o tipo de configuração adotado. A escolha pelo modelo da configuração 2, obteve maior praticidade para classificar um determinado documento, pois necessita somente da primeira página em relação a configuração 1, que necessita de todas as páginas do documento. Portanto, o modelo final adotado para o sistema da API, poderia ser a Máquina de vetor de suporte, Perceptron de multicamadas e *XGBoost*, pelos seus bons resultados e semelhantes. Foi escolhido o modelo *XGBoost*, levando em consideração a precisão e validação cruzada para a aplicação final.

6. Conclusão

Neste trabalho, foi apresentado um sistema de API capaz de realizar a classificação e detecção de documentos institucionais provenientes da UESPI. Neste projeto, foi utilizado algumas técnicas necessárias para que esse sistema conseguisse corretamente rotular os documentos. Para o reconhecimento dos textos nos documentos, foi utilizado a técnica de OCR através do software *Tesseract*, disponibilizado para a linguagem *Python*. Foi utilizado algumas técnicas de limpeza e pré-processamento dos textos, para que possa ficar mais evidente as características textuais de cada rótulo. Foi adicionado uma camada de filtragem através de expressões regulares, com o objetivo de identificar a classe do documento pelo título. Além disso, foram implementados quatro modelos classificadores, afim de realizar comparações estatísticas e escolher o melhor modelo para servir o sistema da API. Com todos esses processo, o sistema obteve bom desempenho na tarefa de classificação e identificação de documentos, sendo que, além da camada de filtragem, possui modelo com acurácia de 98 % na validação cruzada, caso não fosse identificado pelas expressões regulares.

Referências

Ahmad, I., Basheri, M., Iqbal, M. J., and Rahim, A. (2018a). Performance comparison of support vector machine, random forest, and extreme learning machine for intrusion detection. *IEEE access*, 6:33789–33795.

Ahmad, N. A., Che, M. H., Zainal, A., Abd Rauf, M. F., and Adnan, Z. (2018b). Review of chatbots design techniques. *International Journal of Computer Applications*, 181(8):7–10.

Ahmadlou, M., Al-Fugara, A., Al-Shabeeb, A. R., Arora, A., Al-Adamat, R., Pham, Q. B., Al-Ansari, N., Linh, N. T. T., and Sajedi, H. (2021). Flood susceptibility mapping and assessment using a novel deep learning model combining multilayer perceptron and autoencoder neural networks. *Journal of Flood Risk Management*, 14(1):e12683.

Arellano, M. A. (2004). Preservation of digital documents. *Ciência da Informação*, 33(2):15–27.

Basha, S. M., Bagyalakshmi, K., Ramesh, C., Rahim, R., Manikandan, R., and Kumar, A. (2019). Comparative study on performance of document classification using supervised machine learning algorithms: Knime. *International Journal on Emerging Technologies*, 10(1):148–153.

Bergstra, J. and Bengio, Y. (2012). Random search for hyper-parameter optimization. *Journal of machine learning research*, 13(2).

Berrar, D. (2019). Cross-validation.

Borgers, L. (2021). The role of artificial intelligence (ai) in radiology: The current ctatus of fda approved systems.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brophy, J. and Lowd, D. (2021). Machine unlearning for random forests. In *International Conference on Machine Learning*, pages 1092–1104. PMLR.

Car, Z., Baressi Šegota, S., Anđelić, N., Lorencin, I., and Mrzljak, V. (2020). Modeling the spread of covid-19 infection using a multilayer perceptron. *Computational and mathematical methods in medicine*, 2020.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chowdhary, K. (2020). Natural language processing. *Fundamentals of artificial intelligence*, pages 603–649.

Christmann, A. and Steinwart, I. (2008). Support vector machines.

Ciecierski, K. A. and Kamola, M. (2020). Comparison of text classification methods for government documents. In *International Conference on Artificial Intelligence and Soft Computing*, pages 39–49. Springer.

Dalianis, H. (2018). Evaluation metrics and evaluation. In *Clinical text mining*, pages 45–53. Springer.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *Proceedings of the 23rd international conference on Machine learning*, pages 233–240.

Ernst, C. (2020). Artificial intelligence and autonomy: self-determination in the age of automated systems. In *Regulating Artificial Intelligence*, pages 53–73. Springer.

Espinosa Touzon, M. (2020). Digital transformation in companies: Invoice processing as a document manager.

Feurer, M. and Hutter, F. (2019). Hyperparameter optimization. In *Automated machine learning*, pages 3–33. Springer, Cham.

Frazier, P. I. (2018). A tutorial on bayesian optimization. *arXiv preprint arXiv:1807.02811*.

Guha, A. and Samanta, D. (2019). Real-time application of document classification based on machine learning. In *International Conference on Information, Communication and Computing Technology*, pages 366–379. Springer.

Haenlein, M. and Kaplan, A. (2019). A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. *California management review*, 61(4):5–14.

Haider, L. (2021). Artificial intelligence in erp.

Hossin, M. and Sulaiman, M. N. (2015). A review on evaluation metrics for data classification evaluations. *International journal of data mining & knowledge management process*, 5(2):1.

Hussein, D. M. E.-D. M. (2018). A survey on sentiment analysis challenges. *Journal of King Saud University-Engineering Sciences*, 30(4):330–338.

Kim, S.-W. and Gil, J.-M. (2019). Research paper classification systems based on tf-idf and lda schemes. *Human-centric Computing and Information Sciences*, 9(1):1–21.

Kosaraju, S. C., Masum, M., Tsaku, N. Z., Patel, P., Bayramoglu, T., Modgil, G., and Kang, M. (2019). Dot-net: Document layout classification using texture-based cnn. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1029–1034. IEEE.

Liashchynskyi, P. and Liashchynskyi, P. (2019). Grid search, random search, genetic algorithm: A big comparison for nas. *arXiv preprint arXiv:1912.06059*.

Lorencin, I., Anđelić, N., Španjol, J., and Car, Z. (2020). Using multi-layer perceptron with laplacian edge detector for bladder cancer diagnosis. *Artificial Intelligence in Medicine*, 102:101746.

Luo, X. (2021). Efficient english text classification using selected machine learning techniques. *Alexandria Engineering Journal*, 60(3):3401–3409.

Mahesh, B. (2020). Machine learning algorithms-a review. *International Journal of Science and Research (IJSR).[Internet]*, 9:381–386.

Memon, J., Sami, M., Khan, R. A., and Uddin, M. (2020). Handwritten optical character recognition (ocr): A comprehensive systematic literature review (slr). *IEEE Access*, 8:142642–142668.

Miao, J. and Zhu, W. (2021). Precision–recall curve (prc) classification trees. *Evolutionary intelligence*, pages 1–25.

Mishra, R., Bian, J., Fiszman, M., Weir, C. R., Jonnalagadda, S., Mostafa, J., and Del Fiol, G. (2014). Text summarization in the biomedical domain: a systematic review of recent research. *Journal of biomedical informatics*, 52:457–467.

Mittal, S., Gupta, S., Shamma, A., Sahni, I., Thakur, D., et al. (2020). A performance comparisons of machine learning classification techniques for job titles using job descriptions.

Ogunleye, A. and Wang, Q.-G. (2019). Xgboost model for chronic kidney disease diagnosis. *IEEE/ACM transactions on computational biology and bioinformatics*, 17(6):2131–2140.

Ora, A. (2020). *Spam detection in short message service using natural language processing and machine learning techniques*. PhD thesis, Dublin, National College of Ireland.

Patel, C., Patel, A., and Patel, D. (2012). Optical character recognition by open source ocr tool tesseract: A case study. *International Journal of Computer Applications*, 55(10):50–56.

Probst, P., Wright, M. N., and Boulesteix, A.-L. (2019). Hyperparameters and tuning strategies for random forest. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(3):e1301.

Rudin, C., Chen, C., Chen, Z., Huang, H., Semenova, L., and Zhong, C. (2022). Interpretable machine learning: Fundamental principles and 10 grand challenges. *Statistics Surveys*, 16:1–85.

Russo, D. P., Zorn, K. M., Clark, A. M., Zhu, H., and Ekins, S. (2018). Comparing multiple machine learning algorithms and metrics for estrogen receptor binding prediction. *Molecular pharmaceutics*, 15(10):4361–4370.

Saraiva, A. A., Ferreira, N. M. F., de Sousa, L. L., Costa, N. J. C., Sousa, J. V. M., Santos, D., Valente, A., and Soares, S. (2019). Classification of images of childhood pneumonia using convolutional neural networks. In *BIOIMAGING*, pages 112–119.

Šegota, S. B., Anđelić, N., Mrzljak, V., Lorencin, I., Kuric, I., and Car, Z. (2021). Utilization of multilayer perceptron for determining the inverse kinematics of an industrial robotic manipulator. *International Journal of Advanced Robotic Systems*, 18(4):1729881420925283.

Shah, K., Patel, H., Sanghvi, D., and Shah, M. (2020). A comparative analysis of logistic regression, random forest and knn models for the text classification. *Augmented Human Research*, 5(1):1–16.

Vasilateanu, A., Ene, R., et al. (2018). Call-center virtual assistant using natural language processing and speech recognition. *Journal of ICT, Design, Engineering and Technological Science*, 2(2):40–46.

Vijayarani, S., Ilamathi, M. J., Nithya, M., et al. (2015). Preprocessing techniques for text mining-an overview. *International Journal of Computer Science & Communication Networks*, 5(1):7–16.

Yang, L. and Shami, A. (2020). On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, 415:295–316.

Yesenia, O. and Veronica, S.-F. (2021). Automatic classification of research papers using machine learning approaches and natural language processing. In *International Conference on Information Technology & Systems*, pages 80–87. Springer.

Yu, T. and Zhu, H. (2020). Hyper-parameter optimization: A review of algorithms and applications. *arXiv preprint arXiv:2003.05689*.

Zaharia, M., Chen, A., Davidson, A., Ghodsi, A., Hong, S. A., Konwinski, A., Murching, S., Nykodym, T., Ogilvie, P., Parkhe, M., et al. (2018). Accelerating the machine learning lifecycle with mlflow. *IEEE Data Eng. Bull.*, 41(4):39–45.

Zhou, J., Qiu, Y., Zhu, S., Armaghani, D. J., Li, C., Nguyen, H., and Yagiz, S. (2021). Optimization of support vector machine through the use of metaheuristic algorithms in forecasting tbm advance rate. *Engineering Applications of Artificial Intelligence*, 97:104015.