Relatório Sprint IDP

Edição 1.2024

Empresa: Jusbrasil

Integrantes do grupo:

Lucas Fiche Ungarelli Borges
Mateus de Araujo Almeida
Rafael Henrique Garcia de Souza

Orientador:

Prof. Me. Klayton Rodrigues de Castro

Desafio: Desenvolvimento de uma abordagem de Inteligência Artificial para Análise e Predição de Medidas Liminares em Dados Jurídicos.

1. Introdução

O Jusbrasil é uma empresa de tecnologia que tem como proposta de valor oferecer acesso rápido e fácil às informações jurídicas. As informações disponibilizadas em seu site incluem andamentos processuais, jurisprudência, doutrina, modelos de peças jurídicas, contatos de advogados e diários oficiais.

Para cumprir seus objetivos, a empresa coleta um volume massivo de dados diariamente, a partir de diversas fontes, os quais devem ser processados, analisados e formatados para disponibilização de informações e dar conhecimento aos seus assinantes. Dentre os diversos tipos de dados coletados, os andamentos processuais tem uma importância singular, pois são objeto de consulta por seus assinantes, em geral advogados e escritórios jurídicos que, conforme a fase do processo, precisam atuar tempestivamente na defesa dos interesses de seus clientes, além das partes de um processo eventualmente interessadas em acompanhar seus casos em julgamento nos tribunais.

Como a atuação jurídica dos advogados é orientada por prazos que, se não forem observados fielmente, podem causar até mesmo a perda do direito de seus clientes, a integridade e confiabilidade das informações disponibilizadas pela

empresa é fundamental. Nesse cenário, o Jusbrasil apresentou um desafio à equipe do Sprint do IDP, conforme detalhado nas seções a seguir.

2. Definição do Problema

Um momento crucial no trâmite dos processos judiciais é a concessão de medidas liminares. Em casos de tutela de urgência, a parte de um processo pode pedir ao juiz que decida de imediato sobre determinada questão, para evitar que o direito objeto de discussão venha a se tornar de impossível aplicação prática em situações delicadas ou de extrema ordem.

Por exemplo, se uma pessoa estiver prestes a ser despejada de um imóvel devido a uma disputa de propriedade, a decisão final pode levar meses ou até anos. No entanto, se durante esse período a pessoa for forçada a deixar o imóvel, pode haver um dano irreparável, se eventualmente for comprovado que ela tinha o direito de permanecer na propriedade.

Para evitar essa situação, o advogado desta parte pode solicitar uma medida liminar ao juiz, pedindo que ele emita uma decisão imediata que permita à pessoa permanecer no imóvel até que o caso seja resolvido. O juiz, considerando a urgência e a possibilidade de danos irreparáveis, poderia conceder essa medida liminar, protegendo temporariamente o direito da pessoa enquanto o processo judicial segue seu curso normal.

Nesse tipo de situação, geralmente são exigidas providências imediatas dos advogados das partes. Havendo uma falha na prestação desse serviço, tanto o cliente pode ver o seu direito frustrado, quanto o próprio advogado pode ser responsabilizado pela falha na prestação de seus serviços.

É justamente para impedir a hipótese em que a parte veja o direito discutido no processo se tornar irrealizável que o conhecimento oportuno desse tipo de decisão é essencial para a atuação dos advogados. Ou seja, é fundamental para o Jusbrasil disponibilizar a ocorrência de decisão liminar aos assinantes interessados de forma completa, confiável e tempestiva, pois eles confiam na empresa para receber notificações sobre os processos em que atuam ou são parte envolvida.

Para que este serviço possa ser realizado de forma mais célere e efetiva pelo Jusbrasil, o desafio proposto foi a criação de uma abordagem de Inteligência Artificial para análise de dados jurídicos, incluindo a modelagem preditiva de medidas liminares, visto que há disponibilidade de dados massivos cujo volume, velocidade e variedade são potencialmente adequados ao treinamento de uma Inteligência Artificial para executar a tarefa.

A literatura aponta que isso pode ser hipoteticamente realizado com a adoção de mecanismos de aprendizado de máquina (Machine Learning) e uma infraestrutura (Big Data) que seja capaz de ingerir, processar e analisar os dados sumarizados dos tribunais e os andamentos processuais de maneira automatizada e tempestiva.

Nos contatos então realizados com os representantes da empresa, obtivemos a informação que eles ainda não utilizam uma abordagem semelhante para a finalidade que aqui se propôs e que o processo atualmente adotado pela empresa é passível de otimização, incluindo a melhoria de desempenho com a diminuição da dependência de um fluxo diverso de tarefas, as quais poderiam ser coordenadas a partir dessa inovação, possibilitando a construção um sistema de identificação da probabilidade de ocorrência de decisões liminares mais eficiente e eficaz.

3. Descrição dos Objetivos

O desafio proposto pelo Jusbrasil é a construção de uma solução para classificação dos processos, a partir de seus andamentos, indicando a existência ou não de uma decisão liminar. Dessa forma, os assinantes interessados em determinado processo poderiam ser notificados oportunamente acerca da eventual existência de uma medida liminar automaticamente, de modo que possam ser adotadas as providências cabíveis de forma cada vez mais tempestiva, eficiente e eficaz.

Assim, o objetivo geral definido para este projeto é o desenvolvimento de uma abordagem de Inteligência Artificial para análise de dados jurídicos, incluindo a modelagem preditiva de medidas liminares. Os objetivos específicos são:

- Análise de Dados para entendimento do fluxo de trabalho viável para identificação de decisões liminares em andamentos de processos judiciais;
- Pesquisa e desenvolvimento de uma solução de Big Data para ingestão e processamento de dados jurídicos;
- Pesquisa e desenvolvimento de uma abordagem de Machine Learning para modelar a probabilidade de ocorrência de decisões liminares;
- Desenvolvimento de uma API (Application Programming Interface) para classificação de processos segundo a probabilidade de ocorrência de decisões liminares.

4. Metodologia

Para alcançar os objetivos definidos neste projeto, a partir da orientação do mentor, Professor Klayton R. Castro, foi utilizado o CRISP-DM (Cross Industry Standard Process for Data Mining), framework de melhores práticas amplamente adotado em projetos de mineração de dados e modelagem preditiva, além de um conjunto de ferramentas de Big Data, NoSQL e de desenvolvimento de REST APIs, para ingerir, processar e analisar os dados, estabelecendo assim o Pipeline de Dados. O CRISP-DM preconiza as seguintes etapas:

a) Entendimento do Negócio: visa compreender os requisitos do projeto e definir claramente os objetivos e metas a serem alcançados.
b) Definição das métricas de sucesso e critérios de avaliação: visa a coleta, descrição e exploração dos dados disponíveis para determinar sua qualidade e relevância.
c) Preparação dos Dados: visa a realização do pré-processamento dos dados para garantir sua qualidade e adequação para análise, incluindo limpeza e transformação para um formato adequado.
d) Modelagem: pesquisa, seleção e treinamento de algoritmos de aprendizado supervisionado para desenvolvimento de uma abordagem capaz de prever a probabilidade de ocorrência de medidas liminares.

e) Avaliação: visa observar o desempenho dos modelos desenvolvidos para garantir que atendam aos critérios definidos.
f) Implementação: visa a construção de um protótipo em estágio de MVP (Mínimo Produto Viável) que possa ser implementado pelo Jusbrasil.

5. Desenvolvimento da Proposta de Solução

O desenvolvimento da solução partiu da análise da base de dados inicialmente fornecida pelo Jusbrasil. Esta base de dados é composta de cinco arquivos em formato .CSV (Comma Separated Values), que podem ser subdivididos em dois grupos.

O primeiro grupo (I) é composto de três arquivos, e possui as amostras para o treinamento do modelo de classificação baseado em aprendizagem supervisionada, visto que os processos já estão rotulados com a existência ou não de medida liminar. O segundo grupo (II) é composto de dois arquivos, e deve ser utilizado para realização dos testes do modelo preditivo a ser construído.

O grupo I possui a seguinte composição: a) processos_amostra.csv, em que constava, um “id” de registro, o número de identificação (“cnj”), a natureza e o assunto de cada processo; b) movimentações_amostra.csv, em que constava o “id” de registro de cada processo do arquivo anterior, e os textos de todos os andamentos processuais de cada um dos processos; e c) result_amostra.csv, em que constava o número de identificação do processo (“cnj”) e a classificação se o processo tinha liminar ou não.

5.1 Entendimento do Negócio

Os arquivos fornecidos pelo Jusbrasil foram importados para a ferramenta de apresentação e análise de dados Microsoft PowerBI, o que possibilitou a construção de um painel (Dashboard) capaz de apoiar uma avaliação inicial do cenário apresentado, conforme disposto nas Figuras 1 e 2.

Figura 1: Dashboard para Análise Inicial de Dados da Base de Treinamento

Observa-se então que a base de treinamento possui mais de 13,1 mil processos, sendo que apenas 12,94% deles, cerca de 1,7 mil processos, indicam decisões liminares em seus andamentos processuais.

Por sua vez, o grupo II apresentou a seguinte composição: a) processos.csv, em que constava, um “id” de registro, o número de identificação (“cnj”), a natureza e o assunto de cada processo; e b) movimentações.csv, em que constava o “id” de registro de cada processo do arquivo anterior, e os textos de todos os andamentos processuais de cada um dos processos. Após a importação para o PowerBI, obtivemos as informações dispostas na Figura 2:

Figura 2: Dashboard para Análise Inicial de Dados da Base de Testes

Observa-se, portanto, que a base de testes possui mais do que o dobro da base de treinamento, contando com mais de 27,8 mil processos. Não há repetição de dados entre os processos relacionados na base de treino e na base de teste. Dessa forma, entendemos que o volume e qualidade dos dados são relevantes e suficientes para iniciar a abordagem de mineração de dados e modelagem do aprendizado de máquina.

5.2 Definição das métricas de sucesso e critérios de avaliação

Como trata-se de um modelo preditivo de uma classificação binária, foi utilizada a matriz de confusão para apurar a acurácia, precisão, recall, f1-score e AUC-ROC (Area Under Curve - Receiver Operating Characteristic), que são as métricas clássicas estabelecidas na literatura especializada.

Inicialmente, optou-se por rodar o fluxo de trabalho do CRISP-DM integralmente em uma primeira iteração apenas na base de treinamento já rotulada. Após apuração dos resultados de aprendizagem supervisionada para validação da

abordagem que foi construída em nível de protótipo, foram estabelecidos os seguintes critérios de sucesso para cada uma das métricas :

- Acurácia: > 0,95
- Precisão: > 0,95 para não liminares e > 0,90 para liminares
- Recall: > 0,95 para não liminares e > 0,85 para liminares
- F1-Score: > 0,95 para não liminares e > 0,85 para liminares
- AUC-ROC: > 0,90

5.3 Preparação dos Dados

Após a análise inicial da base de dados e considerando o volume de dados e a necessidade de manipulação para limpeza e disposição em um formato adequado à aplicação de algoritmos de aprendizado de máquina, decidiu-se pela utilização do sistema gerenciador de banco de dados MongoDB, uma implementação de NoSQL da família de documentos, bastante robusta e conhecida por sua capacidade de lidar com armazenamento e processamento dos dados nessas condições.

Antes da importação da base de treinamento para o MongoDB, foi realizado o pré-processamento inicial de dados, incluindo a limpeza de caracteres desnecessários e a remoção de acentos. Após a importação, considerando que os dados necessários para a utilização dos algoritmos de *machine learning* ainda estavam espalhados em três coleções, foi gerada uma nova coleção relacionando os seguintes campos: “cnj”, “assunto”, “natureza”, “movimentações” e “liminar”.

Além disso, refinou-se a etapa de pré-processamento com a limpeza de dados, removendo a pontuação e palavras não representativas com a finalidade de aplicação de algoritmos de *machine learning*, as chamadas *stop words*, tais como: “a”, “o”, “de”, "que", etc. Também foram removidos os números que constavam nos textos dos campos “natureza”, “assunto” e “movimentações” e representam ruído aos dados.

A fase de pré-processamento encerrou-se quando restaram em cada um dos campos “natureza”, “assunto” e “movimentações” apenas palavras separadas por espaço, pois dessa forma é possível vetorizar todos os campos da base de treino,

transformando todos os dados em números, a fim de possibilitar um uso mais adequado dos algoritmos de machine learning.

Nas iterações seguintes, foi realizado o refinamento da etapa de pré-processamento, com a utilização da biblioteca Python NLTK (Natural Language Toolkit) para complementar o processo de limpeza e remoção de stopwords.

Por fim, foi realizada a junção das palavras separadas dos campos “natureza”, “assunto” e “movimentações” em um campo único, para fins de treinamento dos modelos com base nos algoritmos selecionados.

5.4 Modelagem

Com a base de dados já preparada, utilizou-se entre 65% e 70% da base de treinamento para vetorizar as palavras com a implementação da técnica de TF-IDF (Term Frequency - Inverse Document Frequency), que avalia a importância de cada termo em relação a todo o conjunto de documentos, penalizando termos comuns em muitos documentos.

Foi explorada a adoção dos seguintes algoritmos na modelagem:

- Decision Tree
- Random Forest
- Naive Bayes
- k-NearestNeighbours
- SVM (SVC)
- ExtraTrees
- GradientBoost
- LightGBM
- XGBoost
- Regressão Logística

Os algoritmos que melhor performaram foram o LightGBM, GradientBoost, XGBoost e Decision Tree. Os resultados apurados na modelagem inicial podem ser verificados nas Figuras 3 e 4.

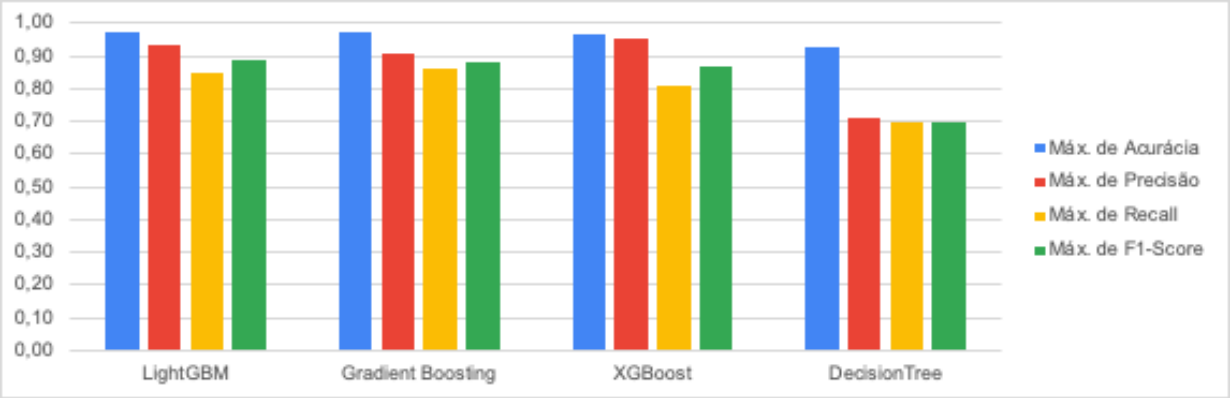


Figura 3: Apresentação das Métricas de Desempenho dos Top 4 Algoritmos (Treino/Teste: 0.65/0.35)

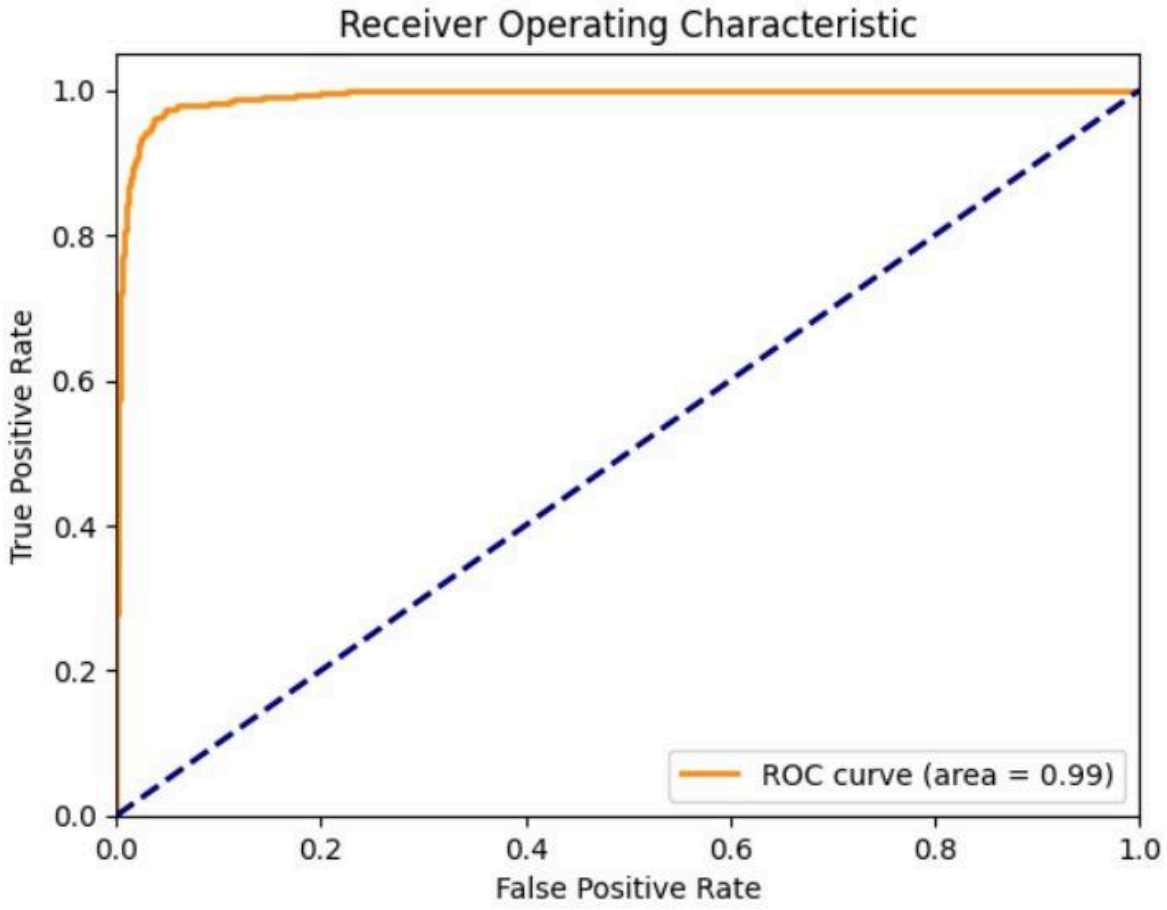


Figura 4: Curva ROC obtida com algoritmo LightGBM (Treino/Teste: 0.7/0.3)

Definidos os algoritmos de aprendizagem de máquina mais assertivos, passou-se para a fase de otimização dos modelos resultantes desses algoritmos. Para tanto, foi utilizada a técnica de otimização bayesiana. Essa técnica se utiliza de análise de regressão para escolher iterativamente o melhor conjunto de

hiperparâmetros, que são configurações externas definidas antes do processo de treinamento, que influenciam o comportamento e desempenho dos algoritmos utilizados no modelo preditivo. A escolha correta dos hiperparâmetros é crucial para obter um modelo de alta qualidade.

Além disso, fez-se uso de validação cruzada, uma técnica usada em aprendizado de máquina para avaliar a capacidade de generalização de um modelo, ou seja, como ele se comporta em dados não vistos. O objetivo é evitar o *overfitting* (quando o modelo se ajusta muito bem aos dados de treinamento, mas não generaliza bem para novos dados) e *underfitting* (quando o modelo não se ajusta bem nem aos dados de treinamento nem aos dados novos). Para isso, no processo de validação cruzada, o conjunto de dados é dividido em várias partes (ou dobras) e usa diferentes partes para treinamento e validação em várias iterações, a fim de corroborar seus resultados.

Os hiperparâmetros e os valores utilizados na busca pela otimização foram, tanto na base de treino em sua completude, como também dividindo-a em cinco partes:

- max_depth: None, 50, 100, 150
- n_estimators: 300, 600, 1200, 1800
- learning_rate: 0.001, 0.01, 0.1

5.5 Avaliação

Em relação ao algoritmo XGBoost, o melhor resultado se deu com a base de treino completa com uma acurácia de 0,96769, enquanto que na base dividida a referida métrica variou de 0,94197 a 0,96331, conforme pode-se ver na Figura 5:

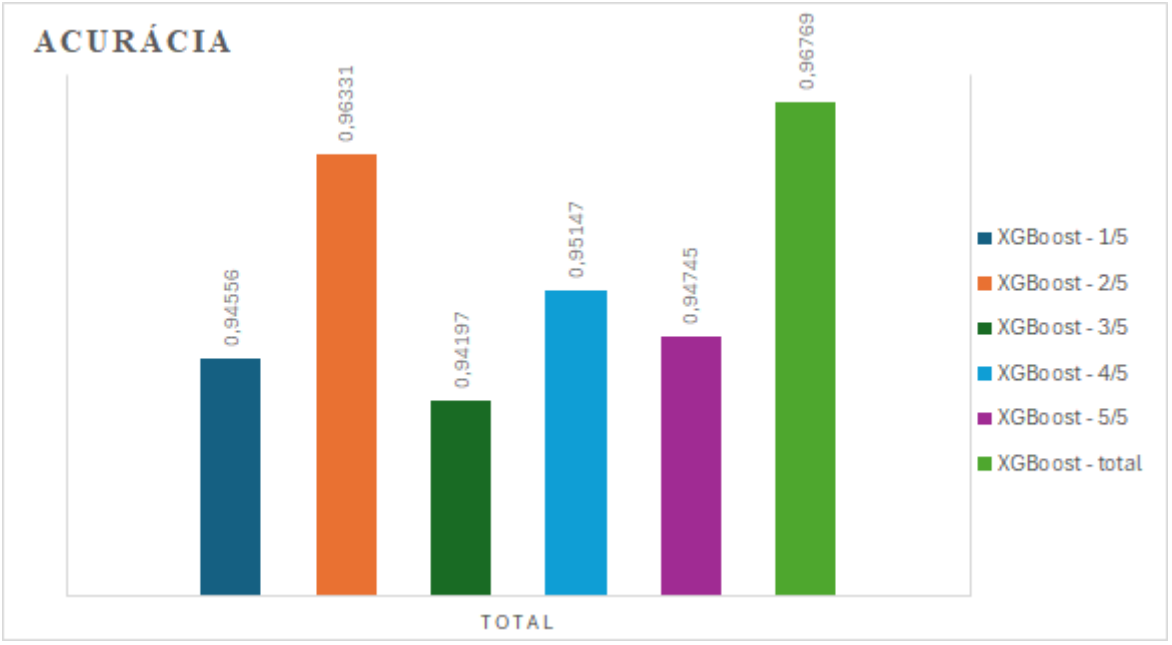


Figura 5: Acurácia do algoritmo XGBoost (Treino/Teste: 0.65/0.35) nas bases parciais e total

Em relação aos melhores hiperparâmetros, os encontrados na otimização do algoritmo XGBoost foram:

- learning_rate: 0,1
- max_depth: None
- n_estimators: 600

Por sua vez, o melhor resultado do algoritmo LightGBM se deu com a base de treino completa com uma acurácia de 0,96734, enquanto que na base dividida a referida métrica variou entre 0,94079 e 0,95976, conforme pode-se ver na Figura 6:

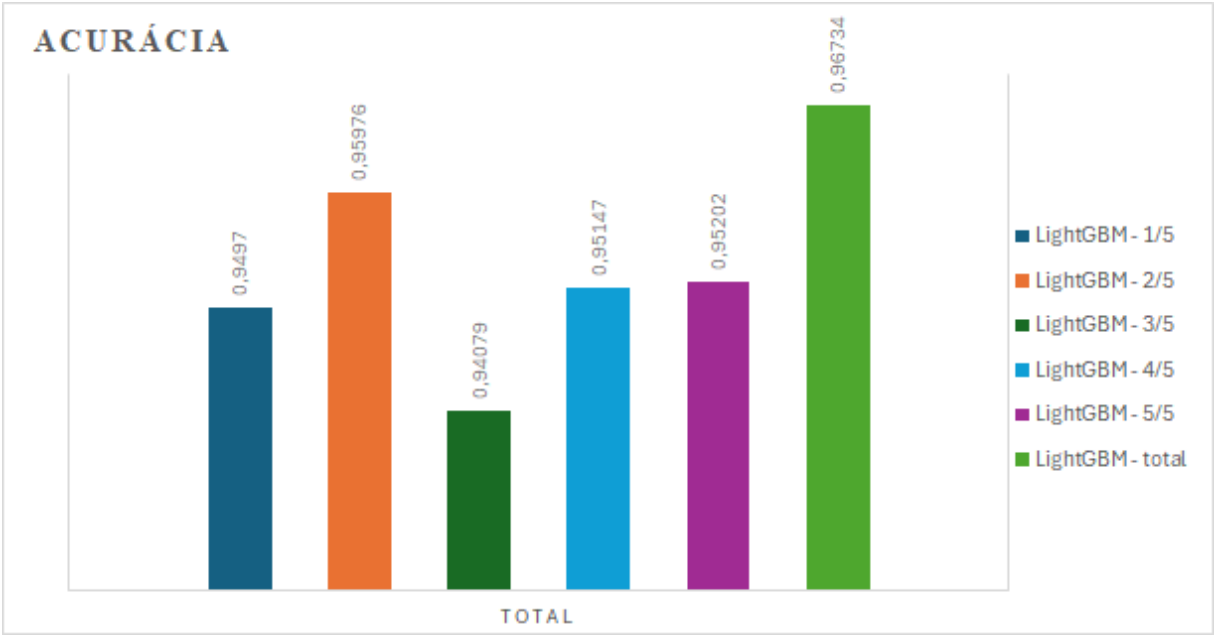
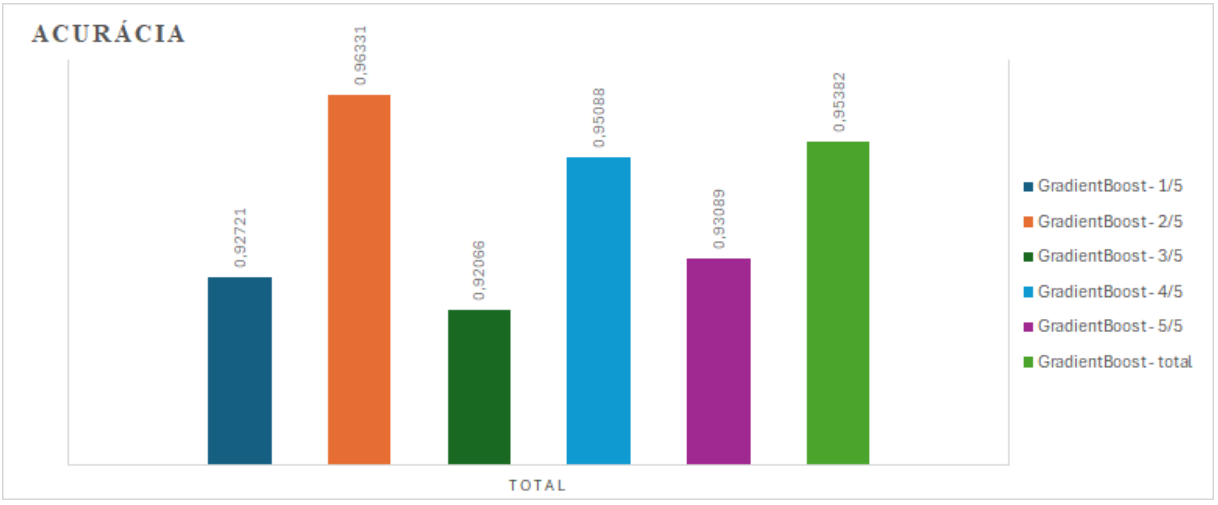


Figura 6: Acurácia do algoritmo LightGBM (Treino/Teste: 0.65/0.35) nas bases parciais e total

Em relação aos melhores hiperparâmetros, os encontrados na otimização do algoritmo LightGBM foram:

- learning_rate: 0,01
- max_depth: 150
- n_estimators: 1800

Por fim, o melhor resultado do algoritmo GradientBoost, se deu com uma das bases parciais com uma acurácia de 0,96331, enquanto nas demais avaliações a referida métrica variou de 0,92066 a 0,95382, conforme pode-se ver na Figura 7:



Dessa forma, observamos que o XGBoost apresentou a melhor consistência de acurácia na base completa (0,96769) e também na base dividida (variando de 0,94197 a 0,96331). O LightGBM teve um desempenho muito próximo ao XGBoost na base completa (0,96734) e um desempenho um pouco inferior na base dividida (variando de 0,94079 a 0,95976). Por sua vez, o GradientBoost teve um desempenho geral um pouco inferior, com a melhor acurácia em uma base parcial de 0,96331 e outras avaliações variando de 0,92066 a 0,95382.

Considerando que a classe de interesse é a 1, que representa a existência de medida liminar, elaborou-se a tabela abaixo, que apresenta as métricas de Precisão, Recall e F1-score obtidas para os algoritmos XGBoost, LightGBM e Gradient Boost:

Algoritmo	Precision (1)	Recall (1)	F1-score (1)
XGBoost	0.92	0.83	0.87
LightGBM	0.93	0.86	0.89
Gradient Boost	0.92	0.84	0.88

Tabela 1: Algoritmos e métricas obtidas para classe de interesse

Embora o XGBoost tenha obtido a maior acurácia na base completa e mostrado bons resultados na base dividida, observamos que as demais métricas não estão bem equilibradas se comparadas ao LightGBM e Gradient Boost. Conforme exposto na Tabela 1, elas apresentaram as maiores variações, o que pode indicar que sua aplicação tende a não generalizar tão bem se exposta a dados ainda não vistos.

Dessa forma, para determinar o algoritmo de melhor desempenho, é importante adotar como critério decisivo não somente a acurácia, mas também a métrica F1-score, pois ela representa a média harmônica entre precisão e recall, considerando como fator principal a classe de interesse (1). Ou seja, quanto maior o índice F1-score, melhor é o desempenho do modelo em identificar a classe de interesse. Por essa razão, optamos pelo LightGBM como algoritmo base para implementação do MVP, pois ele teve o desempenho mais equilibrado se consideradas todas as métricas levantadas, alcançando com sucesso os critérios de

desempenho elencados no início do projeto.

5.6 Implementação

Após a exaustiva experimentação e avaliação que resultou na opção pelo algoritmo LightGBM, conforme descrito na seção anterior, foi realizado o treinamento do modelo utilizando todo o conjunto de dados de treinamento. A etapa de vetorização utilizada no pré-processamento e a geração do modelo treinado foram salvos em arquivos no formato pickle (.pkl), uma forma de serializar objetos em Python, permitindo que o contexto de vetorização das palavras e o modelo treinado sejam armazenados em disco e carregados posteriormente sem a necessidade de retreinamento, ou seja, prontos para serem utilizados pela API (Interface de Programação de Aplicações) que implementa a solução de classificação.

A fim de melhorar a experiência do Jusbrasil, esta API foi desenvolvida para servir como interface entre o modelo preditivo e outras aplicações ou usuários, permitindo que predições sejam feitas em novos dados. Para o desenvolvimento da API foi utilizado o framework web Python Flask, amplamente reconhecido por sua simplicidade e flexibilidade, facilitando o desenvolvimento rápido e eficiente de protótipos de APIs, aliado a robustez e confiabilidade para ser usado em ambientes de produção, com capacidade de implementação de rápida escalabilidade, seja em infraestruturas on-premises ou baseadas em nuvem.

Além disso, Flask é ideal para arquiteturas de microsserviços, onde pequenos serviços independentes são desenvolvidos e implantados separadamente, podendo ser distribuídos para gerenciar todo o fluxo de ingestão dos dados de processos, processamento, análise e classificação desses processos como tendo ou não medida liminar. Esta arquitetura permite que o Jusbrasil integre esta solução de forma eficiente em seus sistemas existentes, proporcionando predições que auxiliem na atuação dos assinantes.

Para isso, é necessário realizar uma requisição web via HTTP/POST (Hypertext Transfer Protocol) com o JSON (JavaScript Object Notation) de entrada fornecido. JSON é um formato de dados leve e fácil de ler e escrever para humanos

e de fácil tratamento e geração para computadores, sendo amplamente suportado por muitas linguagens de programação, pois permite representar dados complexos de forma estruturada, tornando-o ideal para a troca de informações entre microsserviços e arquiteturas cliente/servidor web.

A API desenvolvida retorna a requisição HTTP/POST com um JSON de resposta, indicando se o processo tem uma medida liminar e a probabilidade associada. Este padrão de comunicação REST (Representational State Transfer) é fortemente utilizado para a criação de APIs web, permitindo a comunicação entre sistemas de forma simples e eficiente, sendo amplamente aceito no mercado. As APIs RESTful utilizam métodos HTTP padrão (GET, POST, PUT, DELETE), tornando-as fáceis de integrar com clientes web e outras aplicações que já utilizam essas tecnologias. O fluxo de trabalho estabelecido na API está descrito nas seções seguintes.

5.6.1 Recepção dos Dados:

- A API recebe uma solicitação POST contendo um JSON com `id_cnj` e `movimentacao`.
- As movimentações são uma lista de textos descrevendo os andamentos, que são os eventos processuais.

5.6.2 Pré-processamento:

- O texto das movimentações é tratado, com remoção de acentuação e stopwords, sendo combinado em uma única string e vetorizado segundo o arquivo base de pré-processamento em formato pickle.

5.6.2 Predição:

- O vetor resultante é examinado pelo modelo treinado, também já carregado em um arquivo em formato pickle, para assim gerar uma previsão.
- A previsão indica se o processo tem ou não uma medida liminar (1 para sim, 0 para não).
- A probabilidade associada à previsão é calculada.

5.6.3 Resposta:

- A API responde com um JSON contendo o `id_cnj`, a `previsão` e a `probabilidade.`
- Essa resposta pode ser usada por outras aplicações ou diretamente por advogados para tomar decisões informadas.

Com esta implementação, foi possível criar uma ferramenta eficaz para prever medidas liminares, fornecendo ao Jusbrasil uma vantagem significativa na análise de dados de processos judiciais. As escolhas de implementação do pipeline de dados e da API não apenas atendem aos requisitos atuais, mas também estão preparadas para futuras expansões e melhorias, garantindo sua utilidade a longo prazo.

6. Conclusão

A correção e tempestividade das informações disponibilizadas pelo Jusbrasil sobre os andamentos processuais são de suma importância, pois está em jogo a atuação oportuna do advogado em defesa dos interesses de seus representados. Os resultados obtidos neste projeto apontam que é possível calcular as probabilidades da ocorrência de medidas liminares adotando uma abordagem de processamento e análise de dados com Machine Learning, resultando em uma ferramenta eficaz para a previsão de medidas liminares.

Assim, o objetivo de desenvolver uma abordagem de Inteligência Artificial para a análise de dados jurídicos e modelagem preditiva de medidas liminares foi alcançado com sucesso. A implementação atendeu aos critérios estabelecidos, alcançando as métricas de desempenho definidas para o projeto. As métricas obtidas, incluindo acurácia, precisão, recall, F1-score e AUC-ROC, indicam que o modelo não só é capaz de prever com alta acurácia as medidas liminares, mas também possui um bom equilíbrio entre as diferentes métricas de avaliação.

Com isso, a abordagem proposta mostrou-se robusta para ser potencialmente

integrada aos sistemas do Jusbrasil, fornecendo uma ferramenta confiável para a antecipação e gestão de decisões liminares. Portanto, a solução desenvolvida não só atingiu os objetivos principais, como também atende as expectativas em termos de desempenho preditivo, estabelecendo uma base para futuras melhorias e implementações em larga escala.

Referências Bibliográficas

MAYER-SCHÖNBERGER, Viktor; CUKIER, Kenneth. Big Data: A Revolution That Will Transform How We Live, Work, and Think. Boston: Houghton Mifflin Harcourt, 2013.

SIMON, Phil. Too Big to Ignore: The Business Case for Big Data. New Jersey: John Wiley & Sons, 2013.

DAVENPORT, Thomas H. Big Data at Work: Dispelling the Myths, Uncovering the Opportunities. Boston: Harvard Business Review Press, 2014.

HOWS, David et al. The Definitive Guide to MongoDB: A complete guide to dealing with Big Data using MongoDB. 2. ed. New York: Apress, 2013.

MANNING, Christopher D.; RAGHAVAN, Prabhakar; SCHÜTZE, Hinrich. Introduction to Information Retrieval. Cambridge: Cambridge University Press, 2008.

BIRD, Steven; KLEIN, Ewan; LOPER, Edward. Natural Language Processing with Python. Sebastopol: O'Reilly Media, 2009.

BISHOP, Christopher M. Pattern Recognition and Machine Learning. New York: Springer, 2006.

HAN, Jiawei; KAMBER, Micheline; PEI, Jian. Data Mining: Concepts and Techniques. 3. ed. Waltham: Morgan Kaufmann, 2012.

BEAN, Randy. How Big Data Is Empowering AI and Machine Learning at Scale. MIT Sloan Management Review, 2017. Disponível em: https://sloanreview.mit.edu/article/how-big-data-is-empowering-ai-and-machine-learning-at-scale/. Acesso em: 17 maio 2024.

SVYATKOVSKIY, A.; IMAI, K.; KROEGER, M.; SHIRAI, Y. Large-scale text processing pipeline with Apache Spark. In: IEEE INTERNATIONAL CONFERENCE ON BIG DATA (BIG DATA), 2016, Washington, DC. Anais... Washington, DC: IEEE, 2016. p.

3928-3935. DOI: 10.1109/BigData.2016.7841068. Disponível em: https://ieeexplore.ieee.org/document/7841068. Acesso em: 17 maio 2024.

SAYED, Safynaz Abdel-Fattah; ELKORANY, Abeer Mohamed; SAYED MOHAMMAD, Sabah. Applying Different Machine Learning Techniques for Prediction of COVID-19 Severity. IEEE Access, v. 9, p. 135697–135707, 2021. DOI: 10.1109/access.2021.3116067. Disponível em: http://dx.doi.org/10.1109/ACCESS.2021.3116067. ISSN: 2169-3536.

BARTZ, Eva; BARTZ-BEIELSTEIN, Thomas; ZAEFFERER, Martin; MERSMANN, Olaf (Eds.). Hyperparameter Tuning for Machine and Deep Learning with R: A Practical Guide. Singapore: Springer, 2023. DOI: 10.1007/978-981-19-5170-1. Disponível em: https://doi.org/10.1007/978-981-19-5170-1. Acesso em: 17 maio 2024.

SHEARER, Colin et al. The CRISP-DM Model: The New Blueprint for Data Mining. Journal of Data Warehousing, v. 5, n. 4, p. 13-22, 2000. Disponível em: https://www.ibm.com/docs/en/spss-modeler/saas?topic=dm-crisp-help-overview. Acesso em: 17 maio 2024.