# Previsão de probabilidade de diabetes em estágio inicial usando técnicas de mineração de dados

1st Jonas v. Moreira
*TP555 - Inteligência Artificial e Aprendizado de Máquina*
*Inatel - Instituto Nacional de Telecomunicações*
*Santa Rita do Sapucaí, Brasil*
*jonas.vbm@gmail.com*
*Matrícula 50*

***Abstract*—This work aimed to reproduce, validate, and expand the results of the paper "Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques" [1]. Initially, the methods described in the original paper were reproduced, confirming the effectiveness of the Random Forest algorithm as the most efficient for predicting the risk of early-stage diabetes, with 97.4% accuracy in cross-validation and 99% in the percentage split. Subsequently, other machine learning approaches using AutoGluon were explored, identifying that neural networks can achieve slightly better performance. Finally, a REST API and a graphical interface were developed to operationalize the model in practical applications. The results indicate that machine learning can be a powerful tool for the early screening of diabetes, especially in regions with limited access to laboratory tests.**

***Index Terms*—Diabetes Risk, Symptom, Early Stage, Data Mining, KDD, Dataset, Evaluation Model, Supervised Learning Algorithms, Unsuper- vised Learning Algorithms, Dataset, Mining Tools**

## I. Introdução

A diabetes mellitus é uma das doenças crônicas mais prevalentes e impactantes no mundo. Segundo a Organização Mundial da Saúde (OMS), mais de 422 milhões de pessoas em todo o mundo sofrem com essa condição, sendo que grande parte desses casos não é diagnosticada precocemente [2]. Essa ausência de diagnóstico precoce está associada à progressão silenciosa da doença, que pode levar a complicações graves, como doenças cardiovasculares, insuficiência renal, perda da visão e amputações.

A detecção precoce da diabetes é um desafio global, especialmente em países de baixa e média renda, onde os recursos para testes laboratoriais detalhados, como o teste oral de tolerância à glicose (OGTT) e os níveis de hemoglobina glicada (identificadas através do exame HbA1c), são escassos. Além disso, esses testes, embora eficazes, podem ser caros, demorados e inacessíveis em áreas rurais. Como resultado, muitos pacientes não recebem intervenções adequadas a tempo, aumentando os custos do sistema de saúde e reduzindo significativamente a qualidade de vida das pessoas afetadas.

Nesse contexto, a tecnologia de mineração de dados surge como uma ferramenta poderosa para prever a probabilidade de diabetes com base em sintomas relatados e fatores de risco associados. Algoritmos de aprendizado de máquina têm demonstrado alta eficiência em identificar padrões em grandes volumes de dados, permitindo o desenvolvimento de modelos preditivos precisos. Esses modelos são especialmente úteis para triagem inicial, onde apenas sintomas básicos este disponíveis.

O artigo "Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques" aborda esse problema ao aplicar algoritmos como *Random Forest*, *Naive Bayes* e Regressão Logística para prever o risco de diabetes com base em um conjunto de dados coletado em Bangladesh [1]. O estudo destaca que o *Random Forest* foi o algoritmo com melhor desempenho, alcançando uma precisão de até 97,4% com validação cruzada e 99% com divisão percentual de 80:20.

Dada a relevância do tema, este trabalho teve como objetivos principais: reproduzir os resultados do artigo [1] e validar a eficácia do *Random Forest* como o algoritmo mais preciso; explorar outras técnicas de aprendizado de máquina utilizando o *AutoGluon* [7], a fim de identificar algoritmos que possam superar o desempenho do *Random Forest*; e desenvolver uma aplicação prática que permita predições em tempo real, utilizando uma interface de programação que segue o estilo arquitetural de transferência de estado representacional (API REST) e uma interface gráfica simples.

## II. Metodologia

O artigo [1] analisou um conjunto de dados coletado no Sylhet Diabetes Hospital, em Bangladesh, contendo 520 instâncias e 17 atributos, apresentados na Tabela I. Após um pré-processamento básico, foram aplicados os algoritmos *Naive Bayes*, *Logistic Regression* e *Random Forest*, avaliados por meio de validação cruzada com 10 *folds* e divisão percentual de 80:20 para treino e teste. O *Random Forest* destacou-se como o algoritmo mais preciso, com acurácia de 97,4% na validação cruzada e 99% na divisão percentual [1].

Com base nesse contexto, a metodologia utilizada neste trabalho foi dividir o desafio total em quatro etapas principais: reprodução dos resultados do artigo original, exploração do uso de outros algoritmos com *AutoGluon*, treinamento e serialização do modelo, e desenvolvimento de uma API REST e interface gráfica.

Inicialmente, os dados foram pré-processados para converter valores categóricos, como *Yes/No* e *Male/Female*, em números

TABLE I
ATRIBUTOS DO DATABASE

| Attributes | Values |
|---|---|
| Age | 1. 20-35, 2. 36-45, 3. 46-55, 4. 56-65, 5. above 65 |
| Sex | 1. Male, 2. Female |
| Polyuria | 1. Yes, 2. No |
| Polydipsia | 1. Yes, 2. No |
| Sudden weight loss | 1. Yes, 2. No |
| Weakness | 1. Yes, 2. No |
| Polyphagia | 1. Yes, 2. No |
| Genital thrush | 1. Yes, 2. No |
| Visual blurring | 1. Yes, 2. No |
| Itching | 1. Yes, 2. No |
| Irritability | 1. Yes, 2. No |
| Delayed healing | 1. Yes, 2. No |
| Partial paresis | 1. Yes, 2. No |
| Muscle stiffness | 1. Yes, 2. No |
| Alopecia | 1. Yes, 2. No |
| Obesity | 1. Yes, 2. No |
| Class | 1. Positive, 2. Negative |

binários. Realizou-se a divisão dos dados em conjuntos de treino (75%) e teste (25%), garantindo a reprodutibilidade com uma semente fixa (random_state=50). Os algoritmos *Naive Bayes*, *Logistic Regression* e *Random Forest* foram treinados e avaliados conforme o artigo original. A avaliação incluiu validação cruzada e divisão percentual, confirmando o *Random Forest* como o modelo mais eficiente.

Posteriormente, utilizou-se o *AutoGluon* para explorar uma variedade maior de algoritmos, incluindo redes neurais e métodos baseados em árvores de decisão. A métrica de avaliação escolhida foi *accuracy*, e o *AutoGluon* testou automaticamente diferentes configurações e hiperparâmetros.

Na etapa final, o *Random Forest* foi treinado novamente utilizando o conjunto completo de treino e salvo como um arquivo .pkl utilizando a biblioteca *joblib* [6]. Para operacionalizar o modelo, foi desenvolvida uma API REST em um *microframework web* escrito em *python* chamado *Flask*, capaz de receber dados no formato de notação de objeto *JavaScript* (JSON) e retornar predições binárias (diabetes positivo ou negativo). A API foi integrada a uma interface gráfica simples, que permite que usuários insiram sintomas e recebam previsões de forma rápida e direta.

## III. EXPLORANDO OUTROS ALGORITMOS

Para explorar modelos de algoritmos diferentes dos apresentados em [1], foi desenvolvido um código em *python* para realizar uma análise sobre os dados utilizados em [1], e então tentar encontrar um algoritmo que fosse mais eficiente que o *Random Forest*, opção encontrada pelos autores de [1].

O processo de análise inicia-se com o carregamento do conjunto de dados, que contém as informações sobre diabetes, que é carregado no ambiente de execução e então lido para um *DataFrame*, proporcionando uma estrutura adequada para a manipulação e análise dos dados.

Em seguida, é realizada a conversão das variáveis necessárias para o processamento, e então o conjunto de dados é dividido em duas partes: uma para o treinamento do modelo e outra para a avaliação de sua performance. A divisão é realizada de maneira aleatória, com 80% dos dados sendo alocados para treinamento e 20% para testes. A semente aleatória, especificada pelo parâmetro é utilizada para garantir a reprodutibilidade dos resultados, permitindo que o experimento seja replicado com a mesma divisão dos dados.

O modelo de aprendizado de máquina é então configurado utilizando o *TabularPredictor* do *AutoGluon*, que recebe como parâmetro a coluna *class* da base de dados como variável alvo e a métrica de avaliação definida como acurácia. O treinamento do modelo é realizado por meio de um método que executa de maneira automatizada a seleção do melhor modelo e a otimização dos hiperparâmetros. O *AutoGluon* utiliza técnicas avançadas de *auto machine learning* para avaliar uma série de modelos e configurações, proporcionando uma solução eficiente e prática para problemas de classificação.

Após o treinamento, a avaliação do modelo é realizada utilizando os dados de teste. Adicionalmente, um método de liderança é utilizado para gerar um *ranking* detalhado dos modelos testados durante o processo de treinamento, permitindo uma comparação entre os diversos modelos avaliados e facilitando a análise da performance. Este *ranking* proporciona uma visão clara sobre o desempenho de cada modelo, possibilitando a escolha do modelo mais eficaz para a tarefa de previsão em questão.

Os resultados indicaram que o modelo *NeuralNetFastAI* superou ligeiramente o *Random Forest* em acurácia (99,04% contra 98,85%), destacando-se como uma alternativa viável, como mostra a Figura 1.

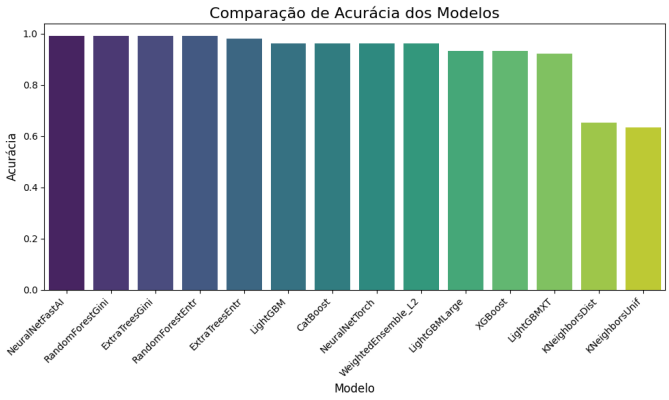


Fig. 1. Comparação de Acurácia dos Modelos.

## IV. TREINANDO O MODELO

Para criar uma aplicação que permita a interação do usuário com um modelo de *machine learning*, é necessário primeiro realizar o treinamento deste modelo, e então exportar o modelo treinado, para que este possa ser utilizado em outras aplicações.

O código desenvolvido para treinar o modelo realiza um processo de preparação dos dados, treinamento de um modelo de aprendizado de máquina e salvamento do modelo treinado para uso posterior.

Inicialmente, os dados são pré-processados para transformar valores categóricos binários, como '*Yes*' e '*No*', em valores numéricos (1 e 0, respectivamente), o que é necessário para o processamento de dados por algoritmos de aprendizado de máquina. Em seguida, a coluna '*Gender*' é convertida de forma similar, onde '*Male*' é transformado em 1 e '*Female*' em 0.

Após a conversão dos dados, as variáveis independentes (atributos) e a variável dependente (a variável alvo a ser prevista) são separadas, com as variáveis independentes armazenadas em X e a variável dependente em y.

O conjunto de dados é então dividido em dois subconjuntos: um para treinamento (75% dos dados) e outro para teste (25%). A seguir, um modelo de *Random Forest* é inicializado, para avaliar as divisões das árvores de decisão, com 100 árvores de decisão sendo utilizadas no modelo. O modelo é treinado com os dados de treinamento, onde o algoritmo aprende a partir das variáveis independentes (x) para prever a variável dependente (y).

Por fim, o modelo treinado é salvo em um arquivo utilizando a biblioteca *joblib* [6], permitindo que o modelo seja carregado posteriormente para fazer previsões sem a necessidade de reexecutar o treinamento. O arquivo salvo é denominado 'diabetes_rf_model.pkl', e pode ser reutilizado para outras análises ou previsões em novos dados.

## V. Aplicação Web

Para validar o modelo de modo prático, foi criada uma aplicação *web* interativa que permite aos usuários submeter dados sobre diversos sintomas e características relacionadas ao diabetes para uma previsão de diagnóstico. A arquitetura do sistema é composta por duas partes principais: a implementação do *backend*, que utiliza a biblioteca *Flask* para expor uma API de previsão, e a interface *frontend*, desenvolvida em linguagem de marcação de hipertexto (HTML), folhas de estilo em cascata (CSS) e *JavaScript*, para a interação com o usuário. A Imagem 2 apresenta o *frontend* da aplicação web.

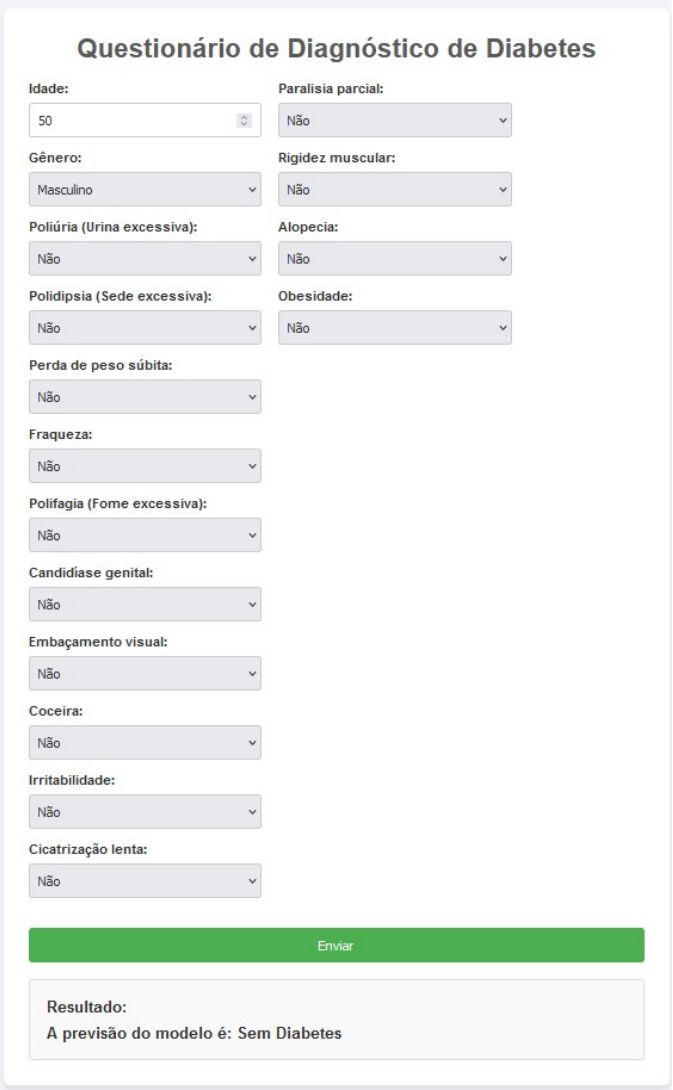


Fig. 2. Frontend da Aplicação Web.

No lado do *backend*, o sistema utiliza o *framework Flask* para criar um servidor *web* que oferece uma rota de previsão, acessível por meio de uma requisição através do protocolo de transferência de hipertexto (HTTP) do tipo postagem (POST). Quando o usuário envia os dados através do *frontend*, esses dados são recebidos pela API na rota *predict*. Os dados são então extraídos do corpo da requisição, em formato JSON, e organizados de maneira a corresponder ao formato exigido pelo modelo de *machine learning* previamente treinado e armazenado no arquivo diabetes_rf_model.pkl. O modelo é carregado utilizando a biblioteca *joblib*, que permite carregar modelos serializados de *python*.

Os dados recebidos são estruturados em um *DataFrame* utilizando a biblioteca Pandas, que é uma estrutura de dados eficiente para manipulação de grandes volumes de informações tabulares. Esse *DataFrame* é então passado ao modelo de aprendizado de máquina, que realiza a predição sobre a probabilidade do paciente ter diabetes, baseada nos sintomas fornecidos. O resultado da predição é retornado ao *frontend* no formato JSON, com a chave *prediction* indicando se o resultado foi positivo ou negativo para diabetes.

A aplicação *Flask* foi configurada com o *Flask-CORS* para garantir que o *backend* possa ser acessado por clientes externos, independentemente de seu domínio, facilitando a interação com o *frontend*. A API pode ser testada em um servidor local, como por exemplo um ambiente de desenvolvimento. Em um ambiente de produção, a única diferença seria necessário ajustar o localizador uUniforme de recursos (URL) de acesso para refletir o endereço público da aplicação.

O *frontend* da aplicação consiste em um formulário HTML que coleta os dados do usuário, como idade, gênero, e presença de sintomas como poliúria (urina excessiva), polidipsia (sede excessiva), e outros sinais clássicos do diabetes. O formulário é estruturado de maneira clara e interativa, com campos para o usuário selecionar suas respostas, utilizando entradas de texto e menus suspensos. Quando o usuário preenche o formulário e clica no botão de envio, o *JavaScript* captura os dados e

os envia para a API *backend*. Essa função faz uma requisição assíncrona ao servidor, enviando os dados no formato JSON.

Ao receber a resposta da API, o *frontend* exibe a previsão de forma clara para o usuário, informando se o diagnóstico aponta para a presença ou ausência de diabetes. Em caso de erro, uma mensagem é exibida para que o usuário saiba que ocorreu um problema ao processar os dados.

Este sistema proporciona uma interface simples e eficaz para a utilização de modelos de *machine learning* em ambientes clínicos, permitindo a previsão de condições de saúde com base em sintomas reportados pelos pacientes. A integração entre o *backend* em *Flask* e o *frontend* em HTML/*JavaScript* torna a aplicação acessível e fácil de usar, facilitando a disseminação e aplicação de tecnologias de diagnóstico automatizado na área da saúde.

O modelo implementado neste experimento utilizou uma interface HTML simples, mas a API desenvolvida para realizar a predição, poderia ser integrada a qualquer tipo de sistema, ou então servir de backend para um aplicativo de *smartphone*, por exemplo.

## VI. Resultados

Na reprodução dos resultados do artigo original, os algoritmos *Naive Bayes*, *Logistic Regression* e *Random Forest* foram avaliados com os mesmos métodos descritos no estudo. O *Random Forest* confirmou-se como o mais eficiente, com 97,4% de acurácia na validação cruzada e 99% na divisão percentual.

A exploração com o *AutoGluon* revelou que redes neurais, como o modelo *NeuralNetFastAI*, podem alcançar desempenho ligeiramente superior, com 99,04% de acurácia, como apresenta a Tabela II. Entretanto, o *Random Forest* foi escolhido para a implementação prática devido à sua combinação de simplicidade, robustez e interpretabilidade.

Por fim, a integração do modelo em uma API REST e uma interface gráfica mostrou-se funcional e eficiente, com testes simulados confirmando a precisão das predições em tempo real.

TABLE II
Comparação de desempenho dos algoritmos.

| Model | Score Test | Score Val | Eval Metric |
|---|---|---|---|
| NeuralNetFastAI | 0.990385 | 0.988095 | accuracy |
| RandomForestGini | 0.990385 | 0.976190 | accuracy |
| RandomForestEntr | 0.990385 | 0.976190 | accuracy |
| ExtraTreesGini | 0.990385 | 0.976190 | accuracy |
| ExtraTreesEntr | 0.980769 | 0.988095 | accuracy |
| LightGBM | 0.961538 | 0.988095 | accuracy |
| CatBoost | 0.961538 | 0.976190 | accuracy |
| NeuralNetTorch | 0.961538 | 1.000000 | accuracy |
| WeightedEnsemble_L2 | 0.961538 | 1.000000 | accuracy |
| LightGBMLarge | 0.932692 | 0.988095 | accuracy |
| XGBoost | 0.932692 | 0.988095 | accuracy |
| LightGBMXT | 0.923077 | 0.988095 | accuracy |
| KNeighborsDist | 0.653846 | 0.607143 | accuracy |
| KNeighborsUnif | 0.634615 | 0.583333 | accuracy |

## VII. Discussão

Os resultados reproduzidos confirmaram a robustez do *Random Forest* para prever o risco de diabetes. A inclusão do *AutoGluon* destacou a importância de explorar múltiplas abordagens para otimizar a precisão do modelo. A API REST e a interface gráfica expandiram a aplicabilidade prática do modelo, demonstrando potencial para uso em triagens clínicas iniciais.

Entretanto, algumas limitações foram identificadas. O conjunto de dados é relativamente pequeno e específico de uma região, o que pode comprometer a generalização. Além disso, o modelo mostrou-se dependente de sintomas como poliúria e polidipsia, o que pode afetar sua eficácia em cenários onde esses sintomas não estejam claramente presentes.

## VIII. Conclusão

Este trabalho contribuiu para a validação dos resultados do artigo original e expandiu a análise ao incorporar novas abordagens. A integração prática do modelo em uma API e interface gráfica demonstrou que algoritmos de aprendizado de máquina podem oferecer soluções acessíveis para a triagem de diabetes em estágio inicial.

Pesquisas futuras podem focar na ampliação do conjunto de dados, desenvolvimento de modelos híbridos e validação em ambientes clínicos. Esses esforços têm o potencial de aumentar a robustez e aplicabilidade das soluções propostas, promovendo diagnósticos mais precoces e eficazes.

## Acknowledgment

Este artigo foi escrito como trabalho final da disciplina TP555 - Inteligência Artificial e Aprendizado de Máquina, do Instituto Nacional de Telecomunicações - Inatel, no segundo semestre de 2024.

## References

[1] M. M. F. Islam, R. Ferdousi, S. Rahman, and H. Y. Bushra, "Likelihood Prediction of Diabetes at Early Stage Using Data Mining Techniques," in *Proceedings of the 2020 International Conference on Data Mining and Big Data*, 2020. Available at: https://doi.org/10.1007/978-981-15-1910-9_4.

[2] Organização Mundial da Saúde (OMS), "Diabetes," Available at: https://www.who.int/news-room/fact-sheets/detail/diabetes.

[3] AutoGluon Team, "AutoGluon: AutoML for Text, Image, and Tabular Data," Available at: https://auto.gluon.ai, 2023.

[4] F. Pedregosa, et al., "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825-2830, 2011. Available at: https://scikit-learn.org.

[5] Flask Documentation, "Flask: Web Application Framework," Available at: https://flask.palletsprojects.com, 2023.

[6] Joblib Team, "Joblib: Lightweight Pipelines in Python," Available at: https://joblib, 2023.

[7] AutoGluon Team, "AutoGluon: AutoML for Text, Image, and Tabular Data," Available at: https://auto.gluon.ai, 2023.