instituto politécnico de gestão e tecnologia

MESTRADO EM ENGENHARIA DE

TECNOLOGIAS E SISTEMAS WEB

Trust-DT

A Framework for Trustworthy Digital Twins in Healthcare

Pedro Ribeiro

DISSERTAÇÃO

VILA NOVA DE GAIA

março | 2026

INSTITUTO POLITÉCNICO DE GESTÃO E TECNOLOGIA

Trust-DT

A Framework for Trustworthy Digital Twins in Healthcare

Pedro Ribeiro

Aprovado em 13/03/2026

Composição do Júri:

Prof. Doutor Firmino Oliveira da Silva

Presidente

Professor Doutor Vítor Jesus Filipe

Arguente

Prof. Doutor Jorge Pereira Duque

Orientador

Vila Nova de Gaia

2026

Tese de Mestrado realizada sob a orientação do(a)(s) Prof. Doutor Jorge Duque e apresentada ao ISLA - Instituto Politécnico de Gestão e Tecnologia de Vila Nova de Gaia para obtenção do grau de Mestre em 13 de março de 2026, conforme o Despacho n.º **9371/2020**

RESUMO

O conceito de Digital Twin, inicialmente aplicado na engenharia, evoluiu para um paradigma transformador na área da saúde, com um imenso potencial para permitir cuidados personalizados, preditivos e proativos. Ao criar réplicas virtuais dinâmicas de pacientes, os Digital Twins facilitam a simulação de tratamentos sem expor os pacientes a risco. Contudo, a sua adoção clínica enfrenta uma barreira crítica: a lacuna de confiança nos modelos de Inteligência Artificial (IA) que os sustentam. Esta dissertação defende que a confiança deve ser um princípio de engenharia, e não uma consequência desejada.

Para tal, propõe-se o framework tecnológico Trust-DT, focado na integração sinérgica de dois pilares: a Inteligência Artificial Explicável (XAI), para garantir a transparência, e a Quantificação de Incerteza (UQ), para assegurar a fiabilidade. A avaliação empírica do protótipo num estudo de caso de risco materno revelou uma descoberta crítica: os modelos otimizados por métodos padrão para atingir alta acurácia, embora aparentemente precisos, tornaram-se severamente mal calibrados. Foi diagnosticado um nível de sobreconfiança clinicamente perigoso, com a fiabilidade real dos modelos a ser drasticamente inferior à sua certeza aparente.

Esta descoberta prova empiricamente que a acurácia, isoladamente, é uma métrica insuficiente e enganadora para a confiança clínica. A investigação conclui que os Digital Twins representam uma inovação disruptiva e a sua implementação segura exige frameworks como o Trust-DT, que validam a transparência e a fiabilidade como pilares de engenharia distintos e sinérgicos.

Palavras-chave: *Digital Twin, Personalized Medicine, Explainable AI, VVUQ, Data Privacy.*

ABSTRACT

The concept of the Digital Twin, initially applied in engineering, has evolved into a transformative paradigm in healthcare, with immense potential to enable personalized, predictive, and proactive care. By creating dynamic virtual replicas of patients, Digital Twins facilitate treatment simulation without exposing patients to risk. However, their clinical adoption faces a critical barrier: the trust gap in the Artificial Intelligence (AI) models that sustain them. This dissertation argues that trust must be an engineering principle, not merely a desired outcome.

To this end, the Trust-DT technological framework is proposed, focused on the synergistic integration of two pillars: Explainable AI (XAI), to ensure transparency, and Uncertainty Quantification (UQ), to ensure reliability. The prototype's empirical evaluation in a maternal risk case study revealed a critical discovery: models optimized by standard methods to achieve high accuracy, while apparently precise, became severely miscalibrated. A clinically dangerous level of overconfidence was diagnosed, with the models' real-world reliability being drastically lower than their apparent certainty.

This discovery empirically proves that accuracy, in isolation, is an insufficient and misleading metric for clinical trust. The research concludes that Digital Twins represent a disruptive innovation, and their safe implementation requires frameworks like Trust-DT, which validate transparency and reliability as distinct and synergistic engineering pillars.

Keywords: *Digital Twin, Personalized Medicine, Explainable AI, VVUQ, Data Privacy*.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABELAS

INTRODUÇÃO

O conceito de Gémeo Digital (do inglês Digital Twin, DT), uma representação virtual sincronizada com um objeto ou sistema físico, foi introduzido pela primeira vez em 2003 no contexto da gestão do ciclo de vida de produtos industriais. Na década seguinte, a maturação de tecnologias de suporte como a Internet das Coisas (IoT), a Computação em Nuvem e a Inteligência Artificial (IA) permitiu a sua expansão para domínios de elevada complexidade, sendo a saúde uma das fronteiras mais promissoras.

A aplicação deste conceito ao corpo humano, designada como *Human Digital Twin* (HDT), representa a vanguarda dos cuidados médicos personalizados, prometendo uma visão holística e dinâmica da saúde de um indivíduo. Um HDT funciona como uma réplica digital do paciente (Figura 1), constantemente alimentada por dados provenientes de múltiplas fontes: sensores corporais (IoT), registos de saúde eletrónicos, dados genómicos, e até mesmo informações sobre o estilo de vida e o ambiente. Esta riqueza de dados permite a criação de modelos computacionais que simulam a fisiologia e a patologia de um indivíduo com um nível de detalhe sem precedentes.

O potencial é imenso: testar a eficácia de tratamentos virtualmente antes de os administrar, prever a progressão de doenças crónicas, otimizar planos cirúrgicos e identificar riscos de saúde de forma proativa. Em suma, o HDT oferece um ambiente seguro para a exploração de cenários hipotéticos ("what-if"), acelerando a descoberta e a personalização da medicina.

A motivação para esta investigação reside no potencial disruptivo dos HDT para resolver alguns dos maiores desafios dos sistemas de saúde contemporâneos: a transição de um modelo reativo para um modelo proativo e preventivo, a necessidade de personalizar tratamentos para populações cada vez mais heterogéneas e a otimização de recursos clínicos.

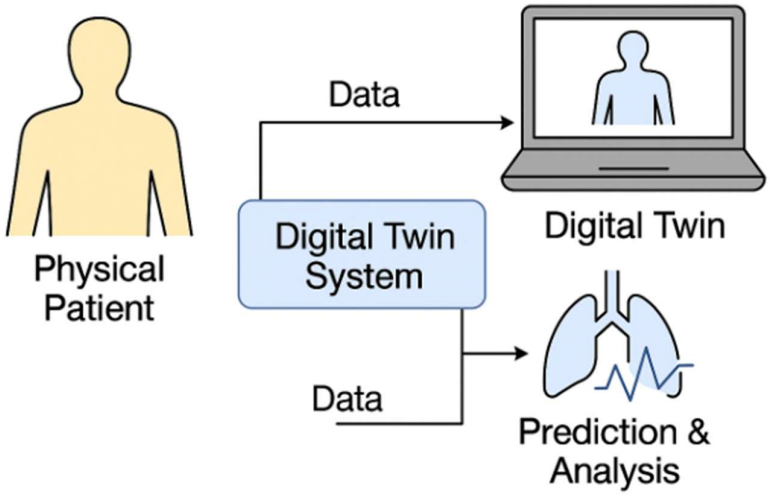

Figura 1. Health Digital Twin

Um HDT funciona como uma réplica digital do paciente (Figura 1), constantemente alimentada por dados provenientes de múltiplas fontes: sensores corporais (IoT), registos de saúde eletrónicos, dados genómicos, bem como informações sobre o estilo de vida e o ambiente. Esta riqueza de dados permite a criação de modelos computacionais que simulam a fisiologia e a patologia de um indivíduo com um nível de detalhe sem precedentes. O potencial é significativo: testar a eficácia de tratamentos virtualmente antes de os administrar, prever a progressão de doenças crónicas, otimizar planos cirúrgicos e identificar riscos de saúde de forma proativa (Vallée, 2024). Em suma, o HDT oferece um ambiente seguro para a exploração de cenários hipotéticos ("what-if"), acelerando a descoberta e a personalização da medicina.

A motivação para esta investigação reside no potencial transformador dos HDT para resolver alguns dos maiores desafios dos sistemas de saúde contemporâneos: a transição de um modelo reativo para um modelo proativo e preventivo, a necessidade de personalizar tratamentos para populações cada vez mais heterogéneas e a otimização de recursos clínicos.

Problema e questões de investigação

Apesar do enorme potencial, a transição do Digital Twin do laboratório para a prática clínica enfrenta uma barreira fundamental: a lacuna de confiança (trust gap). Este fenómeno, amplamente documentado, descreve a dissonância entre o otimismo dos profissionais de saúde em relação ao potencial da IA e a reticência dos pacientes e clínicos na sua adoção,

especialmente em decisões de alto risco. Para que um médico confie numa previsão de um modelo de IA, ou para que um regulador aprove uma terapia baseada em simulações de um DT, é imperativo que o sistema seja fiável, transparente e robusto. Para o clínico na linha da frente, a confiança não é um termo abstrato; é o pré-requisito ético e prático para a ação.

O cerne do problema reside na natureza de "caixa negra" (Figura 2) dos modelos de *Machine Learning* (ML) que constituem o núcleo preditivo de muitos DTs. Estes modelos, embora frequentemente precisos, oferecem previsões sem uma justificação auditável, o que levanta desafios críticos de validação, responsabilidade e potencial para perpetuar vieses ocultos. A opacidade destes algoritmos é inaceitável num domínio como a medicina, onde cada decisão deve ser defensável.

Para além da opacidade, a lacuna de confiança é exacerbada por um segundo desafio, muitas vezes invisível: a má calibração dos modelos de ML. Estes sistemas, frequentemente otimizados em *pipelines* de engenharia de dados (como a Otimização de Hiperparâmetros), focados exclusivamente na acurácia da previsão pontual, podem produzir essas previsões com um sentido de certeza estatística injustificado. Na prática clínica, um modelo que está errado mas confiante é significativamente mais perigoso do que um que declara “não sei” ou que admite um elevado grau de incerteza. A confiança do utilizador clínico, portanto, não exige apenas transparência (o 'porquê' da XAI), mas também uma estimativa fiável da incerteza (o 'quão confiante' da UQ).

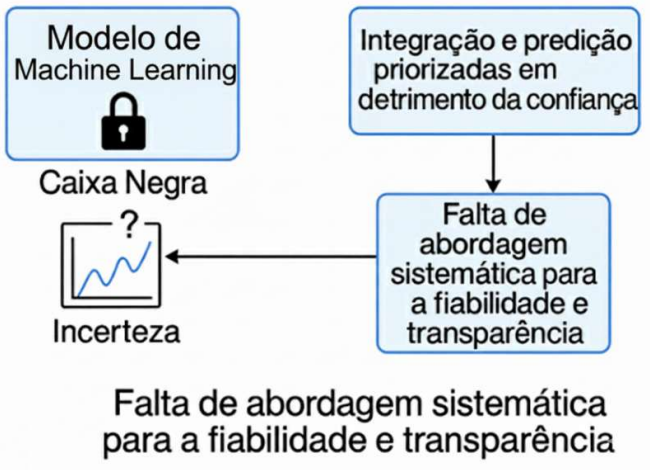

Figura 2. Caixa Negra

Organismos como a U.S. Food and Drug Administration (FDA) e a European Medicines Agency (EMA) enfatizam cada vez mais a necessidade de transparência e responsabilidade

em dispositivos médicos baseados em IA (Alattal et al., 2025). A investigação proposta, ao focar-se na confiança, está alinhada com as exigências, conferindo-lhe uma relevância e um tempo críticos. A confiança não pode ser um requisito secundário ou uma consequência desejada; deve ser um princípio de engenharia, incorporado no design do sistema desde a sua conceção.

A falta de confiança pode ser decomposta em dois pilares científicos interligados:

- Fiabilidade (Reliability): A capacidade do sistema não só fornecer previsões precisas, mas também quantificar a sua própria ignorância. Este pilar é abordado pela Quantificação de Incerteza (UQ - Uncertainty Quantification). Um sistema fiável "sabe quando não sabe". Iremos demonstrar como a aplicação de técnicas de UQ agnósticas ao modelo, especificamente o Conformal Prediction (materializado na avaliação pela biblioteca mapie), permite um diagnóstico rigoroso da fiabilidade e calibração do modelo, indo além das métricas de acurácia superficiais.
- Transparência (Transparency): A capacidade do sistema para explicar a lógica subjacente às suas previsões é um pilar abordado pela IA Explicável (XAI - Explainable AI). Um sistema transparente permite ao clínico validar o raciocínio do modelo com o seu próprio conhecimento, identificar potenciais vieses e comunicar a decisão ao paciente de forma informada.

A literatura atual, no entanto, revela uma lacuna mais profunda: a maioria das investigações aborda UQ e XAI como disciplinas independentes. Esta abordagem é insuficiente, pois uma explicação sem uma estimativa de confiança pode ser enganadora. Inversamente, uma explicação detalhada para uma previsão na qual o modelo tem pouca confiança pode induzir um falso sentimento de segurança, um fenómeno conhecido como viés de automação (automation bias). A vanguarda da investigação em IA confiável argumenta que UQ e XAI são "dois lados da mesma moeda" e que a sua integração sinérgica é essencial para construir sistemas verdadeiramente fiáveis.

Para colmatar esta lacuna, a presente dissertação propõe o desenho e a demonstração de um *framework* conceptual e técnico, denominado *Trust-DT* (*Trustworthy Digital Twin Framework*), eleva a confiança a um princípio de engenharia, integrando de forma sistemática a quantificação da incerteza e a explicabilidade no seu núcleo.

Objetivos, geral e específicos

A questão de investigação que norteia este trabalho é:

Como pode ser desenhado um framework para um Digital Twin em saúde que incorpore nativamente a quantificação de incerteza e a explicabilidade para aumentar a confiança do utilizador clínico?

Para responder a esta questão, foram definidos os seguintes objetivos específicos (OE), cada um associado a uma contribuição de investigação distinta:

- OE1: Análise do Estado da Arte: Realizar uma revisão sistemática da literatura para analisar os *framework*s de DT em saúde existentes e identificar as melhores práticas e lacunas na área da Verificação, Validação e Quantificação de Incerteza (VVUQ) e da IA Explicável (XAI).

Contribuição Teórica: Esta análise irá consolidar o conhecimento existente e validar formalmente a lacuna de investigação, justificando a necessidade de um novo artefacto.

- OE2: Desenho do Artefacto: Desenhar a arquitetura detalhada do *framework* Trust-DT, especificando os seus módulos principais (aquisição de dados, modelo preditivo, motor de UQ, módulo de XAI, interface de confiança) e as suas interações.

Contribuição de Design: A arquitetura modular proposta será a principal contribuição conceptual, tratando a confiança como um princípio de engenharia.

- OE3: Desenvolvimento do Artefacto: Desenvolver um protótipo funcional de um dos módulos críticos do *framework* (o motor de UQ e o módulo de XAI aplicados a um modelo preditivo) para demonstrar a sua funcionalidade.

Contribuição Técnica: O protótipo servirá como prova de conceito da viabilidade de integrar mecanismos de confiança num fluxo de trabalho de um *Digital Twin*.

- OE4: Avaliação do Artefacto: Avaliar o protótipo num estudo de caso simulado (previsão de risco de uma doença crónica), demonstrando como o *framework* pode fornecer resultados fiáveis e interpretáveis.

Contribuição Empírica: A avaliação fornecerá evidências concretas do valor acrescentado de fornecer previsões com contexto de incerteza e explicabilidade.

Âmbito e Delimitação da Investigação

Para garantir a viabilidade do projeto no âmbito de uma dissertação, o trabalho estabelece delimitações claras, o que demonstra um planeamento de investigação pragmático e realista. Estas fronteiras são cruciais para uma avaliação justa do trabalho:

- **Validação Retrospetiva:** A avaliação do protótipo é realizada utilizando *datasets* públicos e históricos. Não envolve a implementação num ambiente clínico real nem a recolha de dados prospetivos de pacientes, o que seria logisticamente e eticamente complexo.
- **Protótipo Focado:** O desenvolvimento de software concentra-se nos módulos de inovação central (UQ e XAI), em vez de uma implementação completa e pronta para produção de todos os componentes do *framework*. O objetivo é a prova de conceito, não um produto final.
- **Avaliação de Utilizador Simulada:** A avaliação da utilidade da "Interface de Confiança" é baseada em cenários simulados. Na sua fase inicial, não envolve um estudo formal de usabilidade com utilizadores clínicos reais para medir o impacto na confiança e na tomada de decisão.

Estrutura da Dissertação

A presente dissertação está organizada em sete capítulos, estruturados de forma a guiar o leitor desde a fundamentação teórica até à apresentação e discussão dos resultados da investigação.

O Capítulo INTRODUÇÃO, contextualiza o conceito de Digital Twin na saúde, formula o problema da "lacuna de confiança" e estabelece as questões de investigação. São também definidos os objetivos gerais e específicos do trabalho, o seu âmbito e as suas delimitações.

O Capítulo REVISÃO DA LITERATURA, foca-se nos conceitos teóricos que sustentam a investigação. Apresenta a evolução do conceito de *Digital Twin*, aprofunda os pilares da confiança - Fiabilidade (VVUQ) e Transparência (XAI) - discute os requisitos de conformidade regulatória e culmina na identificação formal da lacuna de investigação.

O Capítulo ESTADO DA ARTE, analisa a aplicação prática dos conceitos. Descreve as arquiteturas de referência e as tecnologias-chave utilizadas na implementação de Digital

Twins, explora casos de estudo e aplicações atuais em diversas áreas clínicas e operacionais, e sumariza os desafios de engenharia que emergem dessas implementações.

O Capítulo METODOLOGIA, detalha a abordagem de investigação adotada. Justifica a escolha da metodologia Design Science Research (DSR) como o paradigma central do trabalho e descreve como as suas fases foram aplicadas para guiar o desenvolvimento e a avaliação do artefacto proposto.

O Capítulo DESENHO E DESENVOLVIMENTO DO ARTEFACTO, descreve o processo de criação da solução. Apresenta os requisitos do *framework* Trust-DT, detalha a sua arquitetura modular e orientada para a confiança, e descreve o processo de desenvolvimento do protótipo, incluindo a seleção do dataset, o pré-processamento de dados e a seleção dos modelos de *Machine Learning*.

O capítulo RESULTADOS, apresenta de forma detalhada os resultados da avaliação experimental do protótipo. Inclui uma análise comparativa do desempenho preditivo (acurácia, matrizes de confusão), uma análise aprofundada da explicabilidade (SHAP e LIME) e, crucialmente, uma avaliação da fiabilidade e calibração do modelo (UQ/MAPIE).

Finalmente, o trabalho encerra a dissertação com a DISCUSSÃO DOS RESULTADOS e as CONCLUSÕES. A discussão interpreta os achados, contextualizando-os com a literatura e reconhecendo as limitações do estudo. As conclusões respondem diretamente à questão de investigação, sumarizam as contribuições do trabalho e propõem direções para investigações futuras.

REVISÃO DA LITERATURA

Para fundamentar esta investigação e identificar lacunas no conhecimento existente, foi conduzida uma Revisão Sistemática da Literatura, seguindo as diretrizes PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses). Esta abordagem metodológica presente na Figura 3 garante um processo transparente e replicável para identificar, avaliar e sintetizar todas as evidências relevantes para as questões de investigação.

Figura 3. Revisão Sistemática da Literatura

De forma a garantir uma análise abrangente e sistemática do campo de investigação, foi conduzida uma revisão da literatura utilizando as principais bases de dados académicas, incluindo IEEE Xplore, ACM Digital Library, e PubMed. As palavras-chave utilizadas na pesquisa, em diversas combinações, foram: "Digital Twin", "Healthcare", "Human Digital Twin", "Medical Digital Twin", "framework", "architecture", "predictive modeling", "Verification, Validation, and Uncertainty Quantification" (VVUQ), "Explainable AI" (XAI), e "Blockchain". Foram selecionados para análise artigos de conferências, artigos de jornais e revisões sistemáticas publicados preferencialmente nos últimos cinco anos, de modo a focar nos avanços mais recentes. O critério de inclusão principal foi a relevância do trabalho para a proposta de arquiteturas, modelos ou desafios na implementação de DTs na saúde.

O processo de seleção dos estudos foi executado, como referido anteriormente, segundo as diretrizes do protocolo PRISMA (Preferred Reporting Items for Systematic Reviews and Meta-Analyses), assegura a replicabilidade da pesquisa. O diagrama de fluxo apresentado na Figura 4 ilustra cada etapa deste processo. A figura quantifica o número de registos identificados, triados, avaliados e, por fim, incluídos, detalhando os critérios aplicados para a exclusão em cada fase.

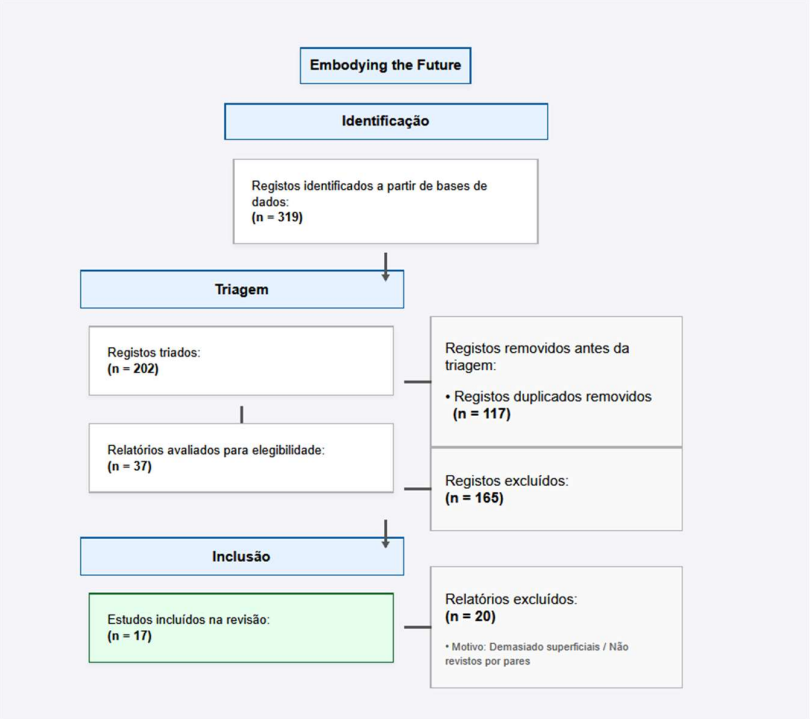

Figura 4. Diagrama PRISMA

A aplicação do protocolo PRISMA, cujo fluxo foi detalhado na figura anterior, culminou na seleção de um corpus final de 17 estudos. Esta base de literatura, rigorosamente selecionada, representa o conhecimento mais pertinente para a presente investigação. As secções subsequentes procedem à análise aprofundada dos conceitos, arquiteturas e desafios críticos identificados neste conjunto de publicações. Esta análise é fundamental para contextualizar a evolução do campo e para fundamentar a lacuna de investigação que motiva o desenho do artefacto proposto.

O Conceito de Digital Twin na Saúde

Conforme estabelecido na introdução, a aplicação do *Digital Twin* ao corpo humano representa a vanguarda dos cuidados médicos personalizados. A evolução deste conceito pode ser categorizada em diferentes níveis de complexidade, como detalhado na **Error! Reference source not found.**.

Tabela 1. Health Digital Twin (Evolução)

Conceito	Origem/Ano	Definição Chave	Foco Principal
Digital Twin	Grieves (2003), Univ. Michigan	Representação digital de um objeto ou processo físico, sincronizada através de um fluxo de dados.	Gestão do Ciclo de Vida do Produto, Engenharia, Processos Industriais.
Medical Digital Twin	Década de 2010	Representação digital de um artefacto médico (e.g., pâncreas artificial) ou de um sistema biológico específico (e.g., um órgão).	Simulação de dispositivos médicos, análise de dados de saúde para fins específicos (Kuruppu Appuhamilage et al., 2025).
Human Digital Twin (HDT)	Anos recentes (2020-presente)	Espelho digital abrangente de um indivíduo, refletindo mudanças moleculares, fisiológicas, e de estilo de vida em tempo real.	Monitorização, diagnóstico e tratamento personalizados e proativos, com o objetivo de uma medicina de precisão (Vallée, 2024).

Enquanto o *Medical Digital Twin* se foca em aspetos isolados, o *Human Digital Twin* representa a ambição de um modelo holístico que captura a complexidade do ser humano para uma gestão da saúde personalizada e preditiva.

Os Pilares da Confiança: Fiabilidade e Transparência

A transição do *Digital Twin* para a prática clínica enfrenta uma barreira fundamental: a **lacuna de confiança** (*trust gap*). Para que um sistema seja adotado em decisões de alto risco, é imperativo que seja fiável e transparente.

Verificação, Validação e Quantificação de Incerteza (VVUQ)

Um modelo preditivo só é útil se a sua fiabilidade for conhecida. O processo de VVUQ (Figura 5) é essencial para estabelecer a credibilidade dos modelos computacionais em aplicações de alto risco como a saúde. O processo decompõe-se em:

- **Verificação:** Confirma que o modelo computacional está a ser resolvido corretamente ("Estou a resolver as equações corretamente?").

- **Validação:** Assegura que o modelo é uma representação precisa do mundo real para o uso pretendido ("Estou a resolver as equações corretas?").
- **Quantificação de Incerteza (UQ):** Mede o grau de confiança nas previsões do modelo, considerando as incertezas nos dados de entrada, nos parâmetros e na própria estrutura do modelo.

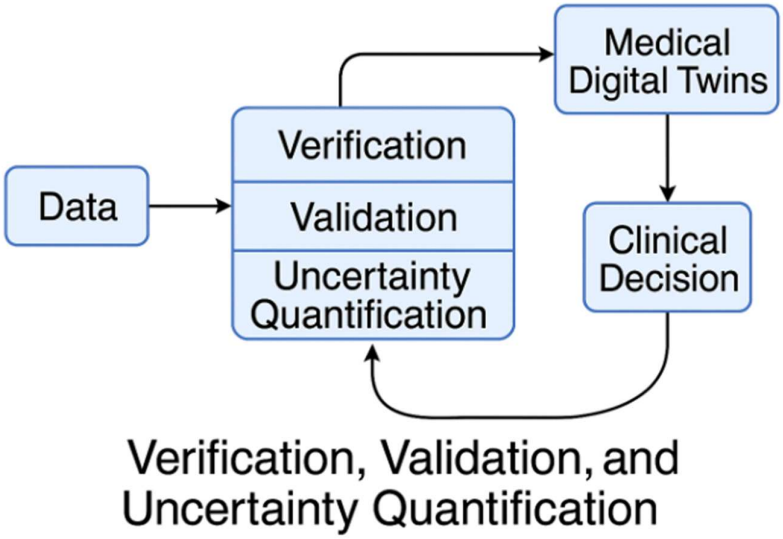

Figura 5. VVUQ - Medical Digital Twin

A literatura demonstra que uma parte significativa das investigações se concentra na otimização da precisão, negligenciando a quantificação da incerteza, um elemento fundamental para a gestão de risco na decisão clínica (Begoli et al., 2019).

Recentemente, o Conformal Prediction (CP) emergiu como uma alternativa poderosa e flexível(Angelopoulos & Bates, 2022). O CP é um *framework* agnóstico ao modelo (tal como o LIME e o SHAP no domínio XAI) que pode ser aplicado a qualquer algoritmo de *machine learning* preexistente. Dada a sua natureza agnóstica ao modelo, a sua robustez estatística e a sua relevância direta para a gestão de risco (fornecendo um conjunto de diagnósticos diferenciais em vez de uma única resposta), o *Conformal Prediction* foi selecionado como a metodologia de UQ para o protótipo do *framework* Trust-DT.

Inteligência Artificial Explicável (XAI)

A natureza de "caixa negra" de muitos modelos de IA é uma das maiores barreiras à sua adoção clínica. A XAI visa tornar os modelos de IA mais interpretáveis, permitindo que os

clínicos compreendam e validem o raciocínio por detrás de uma previsão. Técnicas como LIME e SHAP (**Error! Reference source not found.**) são as mais populares, sendo que o SHAP é frequentemente preferido pela sua robustez teórica e capacidade de fornecer tanto explicações locais como globais.

Tabela 2. LIME vs SHAP

Característica	LIME (Local Interpretable Model-agnostic Explanations)	SHAP (SHapley Additive exPlanations)
Fundamento Teórico	Baseia-se na criação de um modelo substituto local e simples (ex: regressão linear) que aproxima o comportamento do modelo complexo apenas na vizinhança de uma previsão específica.	Fundamenta-se na Teoria dos Jogos, utilizando os valores de Shapley para distribuir de forma justa a "contribuição" de cada característica para a previsão final.
Âmbito da Explicação	Estritamente local. Gera uma explicação para uma única previsão de cada vez, não descrevendo o comportamento geral do modelo.	Oferece explicações tanto locais (para uma previsão individual) como globais (resumindo a importância das características em todo o dataset).
Consistência e Fiabilidade	As explicações podem ser instáveis, variando de acordo com a forma como as perturbações locais são geradas.	Oferece garantias teóricas de consistência. A soma das contribuições das características é sempre igual à diferença entre a previsão e a média.
Complexidade e Velocidade	Geralmente mais rápido e computacionalmente menos intensivo, pois aproxima o modelo localmente com poucas amostras.	Pode ser computacionalmente intensivo e lento, especialmente para modelos não baseados em árvores (KernelSHAP). Há a necessidade de avaliar muitas combinações de características.
Principal Vantagem	Simplicidade, intuição e velocidade. É fácil de compreender e rápido de aplicar para uma análise pontual.	Robustez teórica, consistência e a riqueza dos insights (locais e globais). É considerado o estado da arte para a explicabilidade.

No entanto, a literatura revela que ainda não existe um consenso sobre as melhores formas de apresentar estas explicações e que a maioria dos estudos falha em envolver os médicos no ciclo de desenvolvimento e avaliação, evidenciando uma lacuna entre o desenvolvimento técnico e a implementação clínica real (Abbas et al., 2025).

Segurança, Privacidade e Conformidade Regulatória

A natureza intrinsecamente sensível dos dados de saúde posiciona a segurança e a privacidade como uma preocupação central e inegociável na arquitetura de qualquer DT (Evangeline, 2025). A agregação de dados heterogéneos expande a superfície de ataque, exigindo salvaguardas robustas. A par das salvaguardas tecnológicas, a conformidade com regulamentações estritas é um requisito mandatório. Qualquer *framework* de DT que opere

com dados de cidadãos europeus deve cumprir o Regulamento Geral sobre a Proteção de Dados (GDPR), impõe obrigações rigorosas de consentimento explícito e minimização de dados. De forma análoga, nos Estados Unidos, a *Health Insurance Portability and Accountability Act* (HIPAA) estabelece o padrão para a proteção de informação de saúde, exigindo controlos rigorosos para garantir a sua confidencialidade e segurança. A arquitetura de um DT confiável deve, assim, ser concebida desde o início para cumprir estes quadros regulatórios (Rathnam, 2024).

Identificação da Lacuna de Investigação

Esta revisão sistemática da literatura confirma que, embora existam inúmeras propostas de arquiteturas para *Digital Twins* na saúde e um reconhecimento crescente da importância da confiança, a literatura tende a tratar a Quantificação de Incerteza (UQ) e a IA Explicável (XAI) como disciplinas separadas. Poucos trabalhos abordam a integração sinérgica destes dois pilares como um princípio de engenharia fundamental. Esta lacuna representa um obstáculo significativo para a construção de sistemas de IA verdadeiramente confiáveis e justifica a necessidade de um novo *framework* que una explicitamente a fiabilidade (UQ) e a transparência (XAI) num modelo coeso.

ESTADO DA ARTE

A par da evolução conceptual do *Digital Twin* na saúde, a comunidade de investigação tem concentrado esforços na questão fundamental da sua implementação prática. A transição do conceito para uma solução funcional exige um projeto de engenharia robusto. Assim, a revisão da literatura revela a emergência de diversas arquiteturas e *frameworks* de referência que, embora variados, convergem para um modelo multicamada desenhado para gerir o complexo fluxo de dados, desde a sua aquisição até à entrega de *insights* acionáveis.

Arquiteturas de Referência e Tecnologias-Chave

A literatura propõe diversas arquiteturas para a implementação de DTs na saúde. A maioria converge para um modelo multicamada que geralmente inclui:

Camada de Aquisição de Dados: Responsável por recolher dados de fontes heterogéneas, como sensores IoT (wearables), registos eletrónicos de saúde (EHR), imagens médicas (DICOM), e dados genómicos. A interoperabilidade é um desafio chave nesta camada, com standards como o HL7 FHIR a serem apontados como soluções cruciais.

Camada de Comunicação e Armazenamento: Garante a transmissão segura e o armazenamento eficiente dos dados. Tecnologias de nuvem e bases de dados escaláveis são fundamentais. A segurança e a privacidade dos dados são preocupações primordiais, o que conduz à exploração de tecnologias como Blockchain para garantir a integridade e a rastreabilidade dos dados (Onwubiko et al., 2023).

Camada de Modelação e Simulação: O "cérebro" do DT, onde os dados são processados para criar e atualizar os modelos preditivos. Algoritmos de Inteligência Artificial (IA) e *Machine Learning* (ML) são utilizados para analisar os dados, identificar padrões e simular a progressão de doenças ou a resposta a tratamentos.

Camada de Aplicação e Serviços: Onde os *insights* gerados pelo DT são disponibilizados aos utilizadores finais (médicos, pacientes, entre outros) através de interfaces e aplicações, como dashboards de monitorização ou sistemas de apoio à decisão clínica.

Um desafio fundamental, inerente a estas camadas, é a garantia do processamento em tempo real (real-time processing). A essência de um *Digital Twin* reside na sua capacidade

de se sincronização com a sua contraparte física, o que exige uma comunicação de baixa latência e um processamento de dados de alta velocidade. A infraestrutura de comunicação deve ser robusta para lidar com streams de dados contínuos, muitas vezes de fontes heterogéneas, enquanto a camada de modelação deve ser computacionalmente eficiente para atualizar o estado do gémeo sem atrasos significativos. Manter esta sincronização fiel é um dos maiores pilares e, simultaneamente, um dos desafios de engenharia mais complexos na construção de DTs eficazes (Innowise, 2025).

Embora estas arquiteturas forneçam um roteiro geral, a sua implementação prática enfrenta desafios significativos que a literatura atual começa agora a abordar de forma mais aprofundada.

Aplicações Atuais e Casos de Estudo

A aplicação de *Digital Twins* na saúde já transcendeu a fase conceptual, com implementações práticas a emergirem em diversas especialidades clínicas e operacionais. Estas aplicações podem ser categorizadas em dois grandes grupos: aplicações clínicas, focadas no cuidado direto ao paciente, e aplicações operacionais, visam a otimização de processos e sistemas de saúde.

Medicina Personalizada e Gestão de Doenças Crónicas: A capacidade de criar modelos específicos para cada paciente é uma das aplicações mais promissoras dos DTs. Em doenças crónicas como a diabetes, DTs alimentados por dados de IoT (sensores contínuos de glicose) permitem a monitorização em tempo real e o ajuste dinâmico de planos de tratamento, melhorando o controlo glicémico e a qualidade de vida dos pacientes.

Um estudo de 2024 publicado no JACC demonstrou como estas plataformas podem monitorizar continuamente os sinais vitais (usando dispositivos como o Fitbit e medidores de PA) e a glicose (via CGM), permitindo aos médicos ajustar de forma dinâmica os tratamentos, o que resultou numa melhoria do controlo glicémico e na redução de hospitalizações em estudos-piloto. Empresas como a Twin Health já utilizam esta abordagem para criar representações virtuais do estado metabólico de um indivíduo e recomendar alterações na dieta e atividade física (Ghatti et al., 2023).

Oncologia de Precisão: Na oncologia, os DTs estão a facilitar uma mudança de paradigma de modelos estáticos para modelos dinâmicos que se adaptam à evolução do

tumor. Enquanto os modelos estáticos, criados a partir de um único conjunto de dados pré-tratamento, perdem precisão à medida que o tumor evolui, os DTs dinâmicos integram continuamente dados multimodais para simular cenários de tratamento e orientar decisões terapêuticas em tempo real. Estes modelos virtuais permitem simular a resposta de um tumor a diferentes combinações de quimioterapia, radioterapia e imunoterapia, ajudando os oncologistas a selecionar a estratégia mais eficaz e com menos efeitos secundários para cada paciente.

Por exemplo, um estudo de 2023 (Yang et al., 2024) utilizou um DT preditivo para otimizar regimes de radioterapia para gliomas de alto grau, permitindo o ajuste das doses para maximizar o controlo do tumor e minimizar os danos nos tecidos saudáveis. Também, os DTs podem servir como um benchmark para avaliar a progressão de um cancro sem tratamento, permitindo a eficácia de uma terapia e a otimizar os horários de dosagem (Garner, 2025). Este estudo foi revisto e validado por acompanhamento médico.

Cardiologia: A cardiologia é uma das áreas com casos de uso mais maduros. *Digital Twins* do coração, criados a partir de imagens médicas e dados eletrofisiológicos, são utilizados para tratar arritmias complexas como a taquicardia ventricular (VT) e a fibrilhação auricular (AFib). Estes modelos permitem aos médicos identificar e planear a ablação de tecido cardíaco com uma precisão sem precedentes, prevendo os locais problemáticos antes do procedimento.

Projetos como o Geno-DT da Universidade Johns Hopkins integram ainda dados genéticos para personalizar o tratamento de cardiomiopatias hereditárias, como a cardiomiopatia arritmogénica do ventrículo direito (ARVC). Este modelo integra não só a estrutura cardíaca remodelada pela doença, mas também a causa genética da condição, como as variantes do gene PKP2. Num estudo retrospetivo, o Geno-DT conseguiu prever a localização das VTs com elevada precisão, demonstrando o potencial para otimizar a ablação por cateter de forma não invasiva (*Hopkins Medicine*, 2025).

Planeamento Cirúrgico e Simulação: Os DTs oferecem um ambiente virtual seguro para os cirurgiões planearem e ensaiarem procedimentos complexos antes de entrarem no bloco operatório. Em especialidades como a neurocirurgia, cirurgia cardiovascular e cirurgia plástica, os cirurgiões podem interagir com uma réplica exata da anatomia do paciente, testar diferentes abordagens, antecipar complicações como hemorragias ou danos nos nervos, e otimizar a estratégia cirúrgica. Iniciativas como o projeto Twinical para a cirurgia de cancro

do fígado visam criar um "GPS para cirurgiões", guiando-os em tempo real durante a intervenção com uma réplica digital do órgão do paciente. Outro exemplo é o *framework* Twin-S, desenvolvido especificamente para a cirurgia da base do crânio, capta o progresso cirúrgico no mundo real e atualiza o modelo virtual em tempo real (*Vivatechnology*, 2025).

Desenvolvimento de Fármacos e Ensaios Clínicos: A indústria farmacêutica está a explorar os DTs para acelerar a descoberta de novos medicamentos e otimizar o desenho de ensaios clínicos. Ao simular a resposta de pacientes virtuais a novos fármacos, é possível prever a eficácia e a toxicidade de forma mais célere e com custos mais reduzidos. Também, os DTs podem ser usados para criar braços de controlo virtuais (placebo), reduzindo o número de participantes necessários em ensaios clínicos. Empresas como a Unlearn.AI, em parceria com a Johnson & Johnson, demonstraram que os DTs podem reduzir o tamanho dos braços de controlo em até 33% em ensaios de fase 3 para a doença de Alzheimer. A Agência Europeia de Medicamentos (EMA) já emitiu um parecer favorável à aplicação de DTs em ensaios clínicos de fase 2 e 3, marcando a primeira vez que um órgão regulador endossou uma abordagem baseada em aprendizagem automática para ensaios pivotais (Katsoulakis et al., 2024).

Gestão Operacional de Hospitais: Além do cuidado direto ao paciente, os DTs podem modelar um hospital inteiro, simulando fluxos de pacientes, alocação de recursos (camas, equipamentos, pessoal) e otimizando processos operacionais para aumentar a eficiência e a qualidade dos serviços prestados. Vários hospitais de renome mundial já implementaram esta tecnologia. O Moorfields Eye Hospital no Reino Unido, o Singapore General Hospital, o Duke Health nos EUA e o Karolinska University Hospital na Suécia são exemplos de instituições que utilizam DTs para gerir o fluxo de doentes e otimizar a utilização de recursos, resultando em melhorias na eficiência operacional e nos resultados dos cuidados prestados aos pacientes (Ringeval et al., 2025).

Desafios Críticos na Implementação

A implementação prática das arquiteturas de referência enfrenta desafios de engenharia significativos. Um dos mais prementes é a garantia do **processamento em tempo real**, essencial para manter a sincronização fiel entre o *Digital Twin* e o paciente. Isto exige uma infraestrutura de comunicação de baixa latência e um processamento de dados computacionalmente eficiente. Outro desafio central é a **interoperabilidade de dados**, dada

a necessidade de integrar fontes heterogéneas (EHR, sensores, etc.), sendo a adoção de *standards* como o HL7 FHIR uma solução crucial apontada pela literatura.

METODOLOGIA

Este capítulo apresenta a abordagem metodológica adotada para responder à questão de investigação e atingir os objetivos propostos. Inicia-se com a classificação do estudo, seguida da justificação e descrição da metodologia Design Science Research (DSR) e da sua aplicação concreta nas várias fases do projeto. (Figura 6)

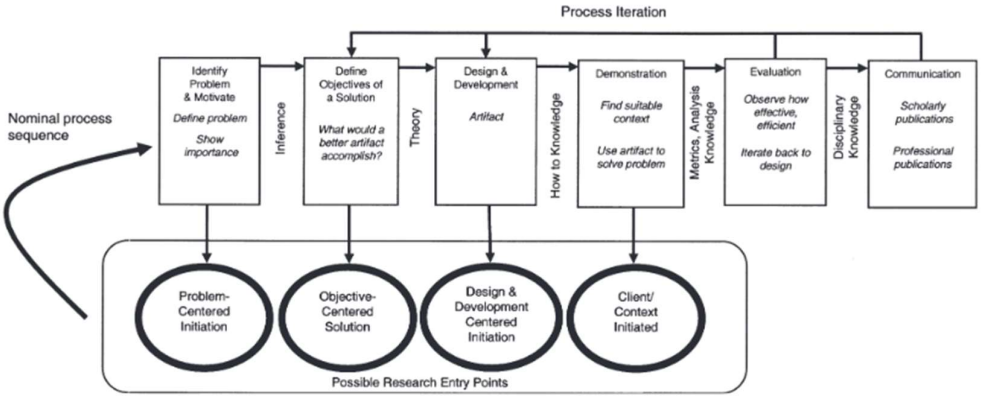

Figura 6. Modelo DSR

Fonte: (Peffers et al., 2007)

Classificação do Estudo

A presente investigação pode ser classificada de acordo com as seguintes dimensões:

- **Natureza:** A investigação é de **natureza aplicada**. O seu propósito não é apenas gerar conhecimento teórico, visa sobretudo resolver um problema prático e relevante do mundo real - a "lacuna de confiança" na adoção de IA na saúde - através da criação de um artefacto tecnológico, o *framework* Trust-DT.
- **Objetivo:** O estudo combina objetivos **exploratórios**, **descritivos** e **explicativos**. É **exploratório** na medida em que investiga a lacuna na integração sinérgica de UQ e XAI; **descritivo** ao detalhar a arquitetura e os componentes do artefacto proposto; e **explicativo** ao avaliar como e porquê a integração destes componentes pode aumentar a confiança do utilizador clínico.
- **Forma de Abordagem:** A abordagem é **mista**, combinando métodos qualitativos e quantitativos. A fase de revisão da literatura e o desenho da arquitetura do *framework* são de natureza **qualitativa**. A fase de avaliação do protótipo, envolve a medição de

métricas de desempenho do modelo de *Machine Learning* (como acurácia e F1-Score), é de natureza **quantitativa**.

- **Procedimentos Técnicos:** Foram utilizados múltiplos procedimentos técnicos. A pesquisa bibliográfica e documental revelou-se fundamental para a revisão do estado da arte. O núcleo do trabalho assenta na **pesquisa experimental** e no desenvolvimento de um **estudo de caso**. A pesquisa experimental envolveu o desenho, implementação e teste de um protótipo de software, enquanto o estudo de caso simulado (previsão de risco na gravidez) serviu como o ambiente para a demonstração e avaliação do artefacto.

Justificação da Escolha da DSR

A escolha da DSR é justificada pela natureza do problema de investigação. O objetivo não é apenas descrever ou teorizar sobre a confiança na IA, mas **criar uma solução prescritiva** - um *framework* - que resolva ativamente o problema da "lacuna de confiança". A DSR fornece o rigor metodológico necessário para ligar a relevância de um problema do mundo real (a necessidade de sistemas de IA confiáveis na saúde) à construção rigorosa de uma solução (o artefacto Trust-DT).

O Modelo de Processo DSR

O trabalho segue o modelo de processo DSR proposto por Peffers et al. (2007), estrutura a investigação em seis fases:

I. **Identificação do Problema e Motivação:** Definir o problema de investigação e justificar o valor de uma solução.

II. **Definição dos Objetivos da Solução:** Delinear os objetivos que o artefacto deve cumprir.

III. **Desenho e Desenvolvimento:** Criar o artefacto.

IV. **Demonstração:** Utilizar o artefacto para resolver uma instância do problema.

V. **Avaliação:** Avaliar a eficácia do artefacto na resolução do problema.

VI. **Comunicação:** Disseminar os resultados da investigação.

Aplicação da Metodologia DSR no Projeto

A DSR foi aplicada em todas as fases do trabalho de acordo com a **Error! Reference source not found.**, com cada fase do modelo a corresponder a um objetivo específico (OE) e a um capítulo da dissertação, garantindo uma estrutura lógica e coerente.

Tabela 3. Fases DSR

Fase DSR	Objetivo Específico Correspondente	Capítulo da Dissertação
Identificação do Problema e Motivação	OE1: Análise do Estado da Arte para identificar a lacuna na integração de UQ e XAI.	Estado da Arte
Definição dos Objetivos da Solução	OE2: Desenho do Artefacto (arquitetura do Trust-DT).	Desenho e Desenvolvimento do Artefacto
Desenho e Desenvolvimento	OE3: Desenvolvimento do Artefacto (protótipo dos módulos UQ e XAI).	Desenho e Desenvolvimento do Artefacto
Demonstração	OE4: Avaliação do Artefacto num estudo de caso simulado.	Avaliação e Resultados
Avaliação	OE4: Análise dos resultados para demonstrar o valor do *framework*.	Avaliação e Resultados
Comunicação	Todos os objetivos	Conclusão e a dissertação como um todo.

Condução do Estudo e Procedimentos Técnicos

O estudo foi conduzido seguindo as fases da DSR:

I. Revisão Sistemática da Literatura (Fase 1): Para identificar o problema, foi realizada uma revisão bibliográfica estruturada, seguindo princípios do protocolo PRISMA para garantir a transparência e a reprodutibilidade. Foram consultadas as bases de dados IEEE Xplore, ACM Digital Library e PubMed, com palavras-chave como "Digital Twin", "Healthcare", "Explainable AI" e "Uncertainty Quantification".

II. Desenho e Desenvolvimento do Artefacto (Fases 2 e 3): Com base na lacuna identificada, foi desenhada a arquitetura do *framework* Trust-DT. De seguida, foi desenvolvido um protótipo como prova de conceito, focando-se nos módulos de UQ e XAI. O desenvolvimento foi realizado em Python, utilizando bibliotecas como Scikit-learn, XGBoost, SHAP e LIME.

III. Estudo de Caso Experimental (Fases 4 e 5): Para a demonstração e avaliação, foi conduzido um estudo de caso quantitativo.

IV. Definição Conceptual das Variáveis: O estudo teve como objetivo prever o "Risco Materno" com base em dados clínicos da paciente.

V. Definição Operacional das Variáveis: Foi utilizado o *dataset* público "Maternal Health Risk" na UC Irvine Machine Learning Repository, ID: 863. As variáveis independentes (preditoras) foram 'Age' (Idade), 'SystolicBP' (Pressão Arterial Sistólica), 'DiastolicBP' (Pressão Arterial Diastólica), 'BS' (Glicemia), 'BodyTemp' (Temperatura Corporal) e 'HeartRate' (Frequência Cardíaca). A variável dependente (alvo) foi a 'RiskLevel', uma variável categórica com três classes ordinais: "baixo risco", "médio risco" e "alto risco".

VI. Condução: O protótipo foi aplicado a este *dataset* para gerar previsões, juntamente com as respetivas métricas de incerteza e explicações, permitindo a avaliação do seu desempenho e utilidade. Para a modelação preditiva, foram implementados três algoritmos representativos de diferentes paradigmas de aprendizagem: Regressão Logística (modelo linear de referência), Random Forest (método de ensemble baseado em *bagging*) e XGBoost (ensemble de gradient *boosting*). Esta escolha permitiu avaliar de forma comparativa o impacto da incerteza e da explicabilidade em modelos de complexidade crescente.

Questões de Investigação Derivadas

A partir dos objetivos específicos, derivam as seguintes questões de investigação que guiaram cada fase do trabalho:

- **QI1 (derivada de OE1):** Quais são as lacunas e as melhores práticas nos *frameworks* de Digital Twins existentes no que diz respeito à integração da fiabilidade (UQ) e da transparência (XAI)?

- **QI2 (derivada de OE2):** Como pode ser desenhada uma arquitetura de software modular que trate a confiança como um requisito de engenharia de primeira classe, integrando nativamente os módulos de UQ e XAI?
- **QI3 (derivada de OE3):** É tecnicamente viável implementar e integrar os módulos de UQ e XAI num protótipo funcional que opere sobre um modelo de *Machine Learning*?
- **QI4 (derivada de OE4):** De que forma a integração da incerteza e da explicabilidade, conforme proposto pelo *framework* Trust-DT, agrega valor interpretativo e de confiança na análise de uma previsão, especialmente em cenários onde o modelo preditivo tem um desempenho limitado?

DESENHO E DESENVOLVIMENTO DO ARTEFACTO

Requisitos do Framework

Com base nos objetivos, o *framework* proposto, denominado **Trust-DT (Trustworthy Digital Twin Framework)**, deve cumprir os seguintes requisitos funcionais e não-funcionais:

- **RF1:** O *framework* deve ser capaz de integrar dados de múltiplas fontes (simuladas para o protótipo).
- **RF2:** Deve incorporar um modelo de ML para realizar previsões (e.g., classificação de risco).
- **RF3:** Deve implementar um **Motor de Quantificação de Incerteza** para calcular uma métrica de confiança para cada previsão.
- **RF4:** Deve integrar um **Módulo de Explicabilidade** que gere uma explicação compreensível para cada previsão.
- **RNF1:** A arquitetura deve ser modular e extensível.
- **RNF2:** As comunicações e o armazenamento de dados devem ser seguros (conceptual, não necessariamente implementado no protótipo).

Arquitetura do Trust-DT

A arquitetura proposta (Figura 7) fundamenta-se na aplicação do conceito de **Digital Twin** ao domínio clínico, visando a criação de um modelo computacional capaz de replicar, em tempo real, o estado e a evolução do paciente. Este sistema processa **dados clínicos heterogéneos**, provenientes de registos médicos e sensores, para alimentar um **modelo de predição** orientado à personalização das decisões terapêuticas. Para assegurar a fiabilidade e a interpretabilidade das previsões, integra-se o módulo **Trust-DT**, incorpora duas dimensões críticas: a **quantificação da incerteza**, permitindo estimar a robustez das inferências, e a **inteligência artificial explicável**, operacionalizada através de técnicas como

a análise SHAP, fornecem justificações transparentes para os resultados obtidos.

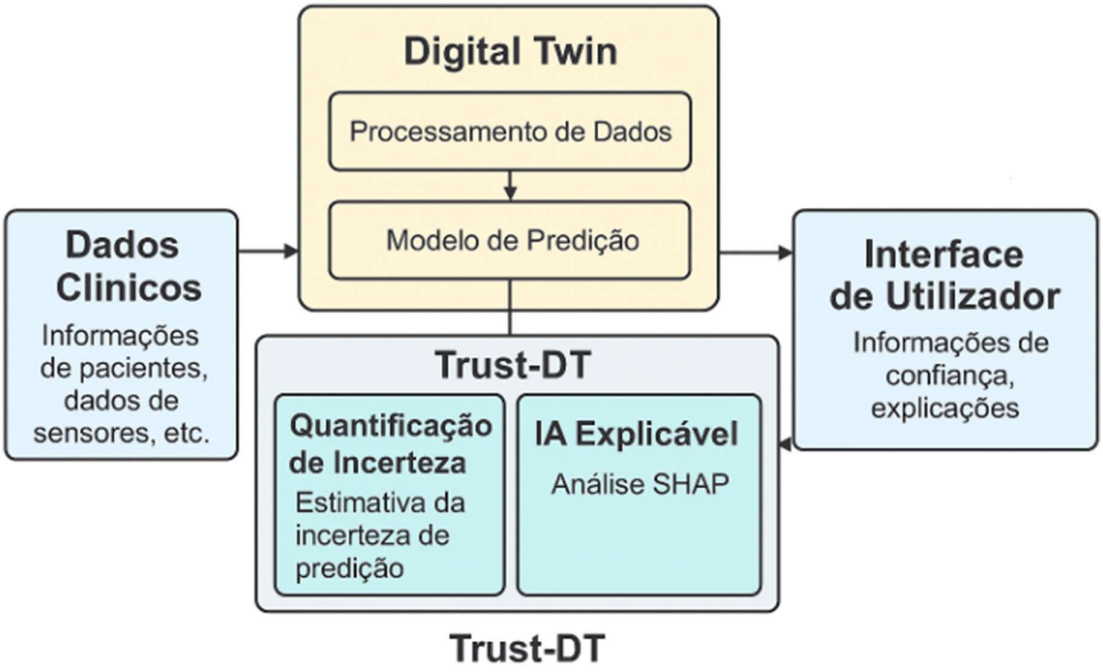

Figura 7. Arquitetura do Framework Trust-DT

De forma complementar, uma **interface de utilizador** disponibiliza não apenas as previsões, mas também indicadores de confiança e explicações detalhadas, promovendo maior transparência e suporte à decisão clínica. O Trust-DT é desenhado com uma arquitetura modular.

O Framework Trust-DT: Uma Arquitetura Orientada para a Confiança

O Trust-DT (*Trustworthy Digital Twin Framework*) é concebido como uma arquitetura de software modular, inspirada em microserviços, eleva a confiança a um requisito de primeira classe, ao invés de uma consequência desejada. O seu *design* é guiado por dois princípios fundamentais para garantir a sua robustez e relevância a longo prazo:

I. **Separação de Responsabilidades:** Cada função central do *framework* - desde a ingestão de dados até à explicabilidade - é encapsulada num módulo distinto e independente. Esta abordagem simplifica o desenvolvimento, os testes e a manutenção, alinhando-se com o **Objetivo Específico 2 (OE2)** de desenhar uma arquitetura detalhada.

II. **Abstração do Modelo (*Model Agnosticism*):** A arquitetura é projetada para desacoplar os módulos de confiança (UQ e XAI) do motor preditivo. Isto significa que o modelo de *Machine Learning* no núcleo do sistema pode ser substituído ou atualizado com um impacto mínimo na restante arquitetura. Este princípio aumenta o valor do *framework* como uma plataforma duradoura para investigação e implementação, em detrimento de uma solução pontual e descartável.

O *framework* é composto por cinco módulos principais que trabalham em conjunto para responder à questão de investigação.

Módulo 1: Aquisição e Pré-processamento de Dados

Este módulo serve como o ponto de entrada de dados para o sistema, cumprindo o **Requisito Funcional 1 (RF1)**.

- **Responsabilidades:**
    - **Ingestão de Dados Heterogéneos:** Capacidade de receber dados de múltiplas fontes, como *streams* de sensores IoT, registos de saúde eletrónicos (EHR) e dados demográficos.
    - **Interoperabilidade:** Implementação de conectores para *standards* de interoperabilidade em saúde, como o HL7 FHIR, para garantir a sua aplicabilidade no mundo real.
    - **Limpeza e Normalização:** Execução de um *pipeline* robusto para validar, limpar e normalizar os dados, um passo fundamental para mitigar a incerteza inerente aos próprios dados (incerteza aleatória).
- **Saída:** Uma representação de dados estruturada e limpa, pronta para ser consumida pelos módulos seguintes.

Módulo 2: Modelação Preditiva

Este é o "cérebro" do *Digital Twin*, responsável por gerar a previsão base e cumprindo o **Requisito Funcional 2 (RF2)**.

- **Responsabilidades:**
    - **Encapsulamento do Modelo de ML:** Contém um modelo de ML treinado para uma tarefa clínica específica (ex: classificação de risco de uma doença crónica).

- **Interface de Previsão:** Expõe uma interface simples que recebe os dados pré-processados e retorna uma previsão bruta (ex: uma probabilidade).

- **Saída:** Uma previsão pontual (ex: probabilidade de risco de 0.85).

Módulo 3: Motor de Quantificação de Incerteza (UQ Engine)

Este é um dos núcleos inovadores do *framework*, desenhado para endereçar a fiabilidade do modelo e cumprir o **Requisito Funcional 3 (RF3)**.

- **Responsabilidades:**
  - **Cálculo da Confiança:** Interage com o Módulo de Modelação Preditiva após a otimização de hiperparâmetros (HPO).
  - **Implementação de Técnicas de UQ:** O protótipo utiliza o _MapieClassifier para gerar conjuntos de previsão e avaliar a calibração do modelo. Crucialmente, é aplicada a estratégia cv="prefit", uma vez que os modelos (ex: best_rf_model) já foram rigorosamente otimizados e validados cruzadamente no *pipeline* de HPO. O mapie é então ajustado nos dados de treino para aprender os limiares de calibração.
- **Saída:** Um conjunto de previsão (ex: {Risco Baixo, Risco Médio}) para cada instância, acompanhado por uma métrica de avaliação global (cobertura) que quantifica a fiabilidade do modelo.

Módulo 4: Módulo de Explicabilidade (XAI Module)

O segundo núcleo inovador do *framework*, focado em tornar a previsão transparente e cumprindo o **Requisito Funcional 4 (RF4)**.

- **Responsabilidades:**
  - **Geração de Explicações:** Interage com o Módulo de Modelação Preditiva para gerar uma explicação para uma previsão específica, respondendo à pergunta "porquê?".
  - **Implementação de Técnicas de XAI:** Utiliza uma técnica mista de XAI, como o SHAP e o LIME, para identificar e quantificar quais as características de entrada que mais contribuíram para o resultado da previsão.
- **Saída:** Um objeto de explicação estruturado (ex: valores SHAP e LIME para cada característica), detalha os fatores por detrás da decisão do modelo.

Módulo 5: Interface de Confiança (Trust Interface)

Esta é a camada de aplicação, onde a informação dos diferentes módulos é sintetizada e apresentada ao utilizador clínico de forma coerente e acionável.

- **Responsabilidades:**
    - **Orquestração:** Orquestra as chamadas aos módulos de UQ e XAI para obter a informação de confiança.
    - **Síntese e Visualização:** Sintetiza as três peças de informação — a **previsão**, a **incerteza** e a **explicação** — numa visualização única e intuitiva. O seu *design* é um desafio de Interação Humano-Computador (HCI) focado em mitigar o viés de automação e promover um envolvimento crítico com os resultados.
- **Saída:** Uma interface de utilizador que apresenta ao clínico uma avaliação holística, permitindo uma tomada de decisão informada e baseada na confiança.

O design modular e agnóstico ao modelo da arquitetura revela uma visão estratégica. O campo do *Machine Learning* evolui a um ritmo vertiginoso, e qualquer modelo preditivo específico está destinado a ser superado. Uma arquitetura monolítica que acoplasse firmemente os componentes de confiança a um modelo particular, tornar-se-ia obsoleta juntamente com esse modelo. Ao desacoplar o motor preditivo dos módulos de UQ e XAI, o *framework* Trust-DT garante a sua própria longevidade. A "estrutura" de confiança permanece valiosa mesmo quando o "motor" no seu interior é substituído por uma versão mais avançada. Desta forma, o Trust-DT não é concebido como uma solução de ponto único, mas como uma **plataforma** durável e adaptável. O seu verdadeiro valor não reside em nenhum modelo preditivo específico que possa conter, mas na sua capacidade de imbuir *qualquer* modelo preditivo - presente ou futuro - com as propriedades essenciais de fiabilidade e transparência. Esta escolha de *design* posiciona o *framework* como uma peça de infraestrutura fundamental para a IA clínica confiável, ao invés de uma aplicação descartável.

Este *framework* Trust-DT fornece a estrutura necessária para o desenvolvimento do protótipo (**OE3**) e para a sua posterior avaliação num estudo de caso (**OE4**), servindo como a principal contribuição da sua dissertação para a construção de *Digital Twins* verdadeiramente confiáveis na saúde.

Desenvolvimento do Protótipo

Com o propósito de validar a viabilidade do *framework* Trust-DT, e em cumprimento dos objetivos específicos OE3 (Desenvolvimento do Artefacto) e OE4 (Avaliação do Artefacto), desenvolveu-se um protótipo funcional. Este protótipo constitui uma prova de conceito, implementa os módulos centrais de modelação preditiva e explicabilidade (XAI), sendo subsequentemente aplicado a um cenário clínico real.

Ambiente de Desenvolvimento e Ferramentas

A implementação do protótipo e a condução da análise experimental foram realizadas num ambiente de desenvolvimento integrado, assente em tecnologias de código aberto (open-source) que representam o estado da arte em Ciência de Dados e Machine Learning. A escolha de cada ferramenta foi deliberada, visando garantir a reprodutibilidade da investigação, a eficiência computacional e a profundidade da análise.

Ambiente de Execução:

Jupyter Notebook: O desenvolvimento foi encapsulado num ambiente Jupyter Notebook. Esta plataforma foi selecionada pela sua natureza interativa que permite a combinação de código executável, visualizações, equações matemáticas e texto narrativo num único documento. Tal característica foi fundamental para facilitar um ciclo de desenvolvimento iterativo, abrangendo desde a análise exploratória de dados até à interpretação dos resultados de explicabilidade, garantindo a transparência e a clareza do processo metodológico.

Ecossistema de Bibliotecas Python:

O protótipo foi integralmente desenvolvido na linguagem de programação Python, devido ao seu vasto ecossistema de bibliotecas especializadas. As principais bibliotecas que sustentaram o desenvolvimento foram:

Manipulação e Análise Numérica:

- ucimlrepo: Utilizada para a aquisição programática do conjunto de dados "Maternal Health Risk" diretamente do repositório de *Machine Learning* da UC Irvine. Esta abordagem assegura a proveniência e a integridade dos dados na fase inicial do *pipeline*.

- Pandas: Essencial para a estruturação, manipulação e limpeza dos dados. A sua estrutura de dados, DataFrame, serviu como o principal objeto para a aplicação de transformações, tratamento de valores atípicos (*outliers*) e engenharia de características (feature engineering).
- NumPy: Constituiu a base para operações numéricas de alta performance, particularmente em cálculos matriciais. Foi utilizada em operações como a transformação logarítmica de atributos, um passo crucial para a estabilização da variância e normalização de distribuições.

Visualização de Dados:

- Matplotlib e Seaborn: Esta combinação foi empregue na análise visual exploratória. Enquanto o Matplotlib forneceu a base para a criação de gráficos, o Seaborn, uma interface de alto nível, foi utilizado para gerar visualizações estatísticas mais complexas e informativas, como mapas de calor (*heatmaps*) para a análise de correlação e *box plots* para a identificação de *outliers*.

Modelagem Preditiva e Avaliação (Machine Learning):

- Scikit-learn: A principal biblioteca para a implementação do fluxo de trabalho de *Machine Learning*. Da mesma, foram utilizados os módulos essenciais para:
    - Pré-processamento: LabelEncoder para a codificação da variável-alvo e StandardScaler para o escalonamento dos atributos.
    - Modelagem: LogisticRegression e RandomForestClassifier para o treino dos modelos preditivos.
    - Avaliação: Funções do módulo metrics para calcular a acurácia, as matrizes de confusão e as curvas ROC, permitindo uma avaliação quantitativa rigorosa do desempenho.
    - Otimização de Hiperparâmetros: RandomizedSearchCV e GridSearchCV (do módulo model_selection) para a busca sistemática e otimização dos hiperparâmetros dos modelos.
- XGBoost: Utilizada para implementar o modelo de Gradient Boosting. A escolha desta biblioteca justifica-se pelo seu reconhecido desempenho superior em

competições e problemas de dados tabulares, servindo como um benchmark de alta performance para a tarefa de classificação.

Inteligência Artificial Explicável (XAI):

- SHAP (SHapley Additive exPlanations): A biblioteca central para a implementação do Módulo de Explicabilidade. A sua escolha fundamenta-se na sua robustez teórica e na capacidade de fornecer explicações consistentes tanto a nível global (importância das características) como local (contribuição para predições individuais).
- LIME (Local Interpretable Model-agnostic Explanations): Integrada de forma complementar ao SHAP, a biblioteca LIME foi utilizada para gerar explicações locais e intuitivas através de modelos substitutos. A sua natureza agnóstica ao modelo permitiu uma análise comparativa direta da lógica interna dos diferentes algoritmos para instâncias específicas, enriquecendo a validação qualitativa.
- MAPIE (Mapping Prediction Intervals and Sets): Incorporada para implementar o Módulo 3 (Motor de UQ) do *framework* Trust-DT. Esta biblioteca foi selecionada como a implementação de referência para o Conformal Prediction em Python. A sua função foi gerar conjuntos de previsão e permitir uma avaliação rigorosa da calibração e fiabilidade do modelo através da métrica de cobertura (coverage), validando assim o pilar de Fiabilidade.

O código-fonte do protótipo está disponível no Anexo A e num repositório Github - Mestrado.

Seleção e Análise do Conjunto de Dados

A seleção do conjunto de dados "Maternal Health Risk" (UC Irvine Machine Learning Repository, ID: 863) para a presente demonstração experimental fundamentou-se na sua proveniência de um contexto clínico real. Os dados, oriundos de sistemas de monitorização IoT implementados em hospitais e clínicas rurais no Bangladesh, conferem uma base empírica robusta e realista ao estudo de caso.

Este *dataset* é particularmente pertinente, pois agrega dados reais de monitorização IoT de 1014 pacientes, caracterizadas por seis variáveis preditoras de natureza fisiológica apresentadas na Figura 8: *Age* (Idade), *SystolicBP* (Pressão Arterial Sistólica), *DiastolicBP*

(Pressão Arterial Diastólica), *BS* (Glicemia), *BodyTemp* (Temperatura Corporal) e *HeartRate* (Frequência Cardíaca).

	Age	SystolicBP	DiastolicBP	BS	BodyTemp	HeartRate
0	25	130	80	15.0	98.0	86
1	35	140	90	13.0	98.0	70
2	29	90	70	8.0	100.0	80
3	30	140	85	7.0	98.0	70
4	35	120	60	6.1	98.0	76

Figura 8. Variáveis Preditoras

A variável dependente (*target*) presente na Figura 9, *RiskLevel*, classifica o estado da paciente em três categorias ordinais: “baixo risco”, “médio risco” ou “alto risco”. Importa salientar que a análise preliminar confirmou a inexistência de valores em falta, assegurando a integridade do conjunto de dados para a subsequente fase de modelação.

	RiskLevel
0	high risk
1	high risk
2	high risk
3	high risk
4	low risk

Figura 9. Variável Alvo

Pré-processamento e Engenharia de Características

Um *pipeline* de pré-processamento robusto foi implementado para preparar os dados para a modelação:

1. **Tratamento de *Outliers***

A análise exploratória através de *box plots* (Figura 10) revelou a presença de *outliers* em várias características. Para mitigar o seu impacto, foi aplicado o método do Intervalo Interquartil (IQR). Os valores que se encontravam fora dos limites (definidos como $Q1 - 1.5 \times IQR$ e $Q3 + 1.5 \times IQR$) foram "tapados" (capped), ou seja, substituídos pelo valor do limite mais próximo.

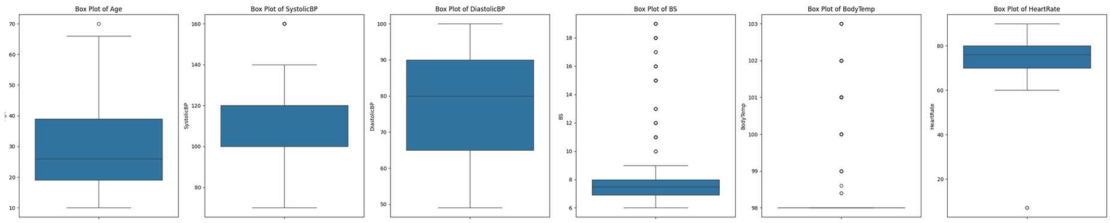

Figura 10. BoxPlots - Outliers

Esta técnica preserva os dados enquanto reduz a influência de valores extremos.

2. **Engenharia de Características (*Feature Engineering*)**

Na fase de pré-processamento, foi identificada uma potencial inconsistência no atributo BS (Glicemia). Embora a documentação do *dataset* indique que a sua unidade de medida é mmol/L, uma análise exploratória da magnitude dos valores sugere que estes são mais consentâneos com a escala de mg/dL. Não obstante esta discrepância, e para assegurar o alinhamento com a meta-informação oficial, optou-se por proceder à conversão, aplicando o fator de $1 \text{ mmol/L} = 18.01559 \text{ mg/dL}$. Adicionalmente, com o intuito de exemplificar uma técnica de engenharia de atributos (feature engineering), aplicou-se uma transformação logarítmica ao mesmo atributo. Esta operação, frequentemente utilizada para mitigar os efeitos de distribuições assimétricas, foi incluída no *pipeline* como uma demonstração da sua aplicabilidade, mesmo na ausência de uma assimetria pronunciada nos dados em análise.

Para enriquecer o poder preditivo do modelo, foram ainda criadas duas características:

- BP_Interaction: Um termo de interação, calculado como o produto entre SystolicBP e DiastolicBP, para capturar o efeito combinado da pressão arterial.

```
X_cleaned['BP_Interaction'] = X_cleaned['SystolicBP'] * X_cleaned['DiastolicBP']
```

- BS_log: Uma transformação logarítmica da característica BS (glicose no sangue) para ajudar a normalizar a sua distribuição e estabilizar a variância.

```
X_cleaned['BS_log'] = np.log(X_cleaned['BS'] + 1e-6)
```

Este trabalho de Engenharia de Características é fundamental para melhorar de forma significativa o desempenho, a estabilidade e fiabilidade dos modelos de *machine learning*.

3. **Codificação da Variável Alvo:** A variável alvo categórica RiskLevel foi convertida para um formato numérico (low risk -> 0, mid risk -> 1, high risk -> 2) utilizando o LabelEncoder da biblioteca Scikit-learn, um passo necessário para a maioria dos algoritmos de classificação.

```
le = LabelEncoder()
y_encoded = le.fit_transform(y['RiskLevel'])
```

Seleção de Características e Escalonamento

A seleção dos atributos mais relevantes foi conduzida através de uma análise de correlação, visualizada (Figura 11) por meio de um mapa de calor (*heatmap*). Este método permitiu quantificar a relação linear entre cada atributo pré-processado e a variável-alvo codificada (RiskLevel_encoded).

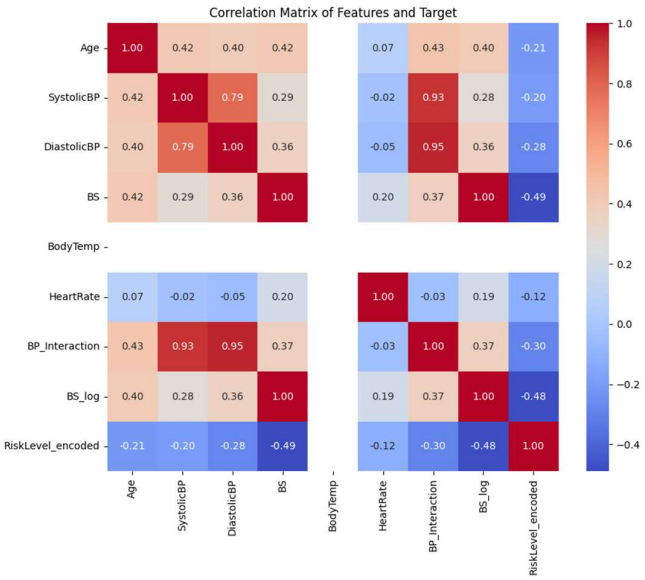

Figura 11. Matriz de Correlação

A seleção dos atributos obedeceu a dois critérios. Inicialmente, foram selecionadas as variáveis que apresentaram os maiores coeficientes de correlação em valor absoluto (BS, BS_log, BP_Interaction e DiastolicBP).

As características selecionadas foram escalonadas utilizando o StandardScaler que transforma os dados para que tenham uma média de 0 e um desvio padrão de 1. Este passo é fundamental para garantir que os modelos, especialmente os lineares como a Regressão Logística, não sejam enviesados por características com diferentes escalas. Em seguida, e com base em conhecimento de domínio, o atributo Age foi também incluído devido à sua importância clínica intrínseca.

O *dataset* final (**Error! Reference source not found.**) para a modelação, designado X_selected, foi então criado, contendo exclusivamente os cinco atributos selecionados: Age, BS, BS_log, BP_Interaction e DiastolicBP.

	Age	BS	BS_log	BP_Interaction	DiastolicBP
0	25	9.65	2.266958	10400	80
1	35	9.65	2.266958	12600	90
2	29	8.00	2.079442	6300	70
3	30	7.00	1.945910	11900	85
4	35	6.10	1.808289	7200	60

Figura 12. Dataset Final

Análise de Caraterísticas e a sua relação com o Risco Materno

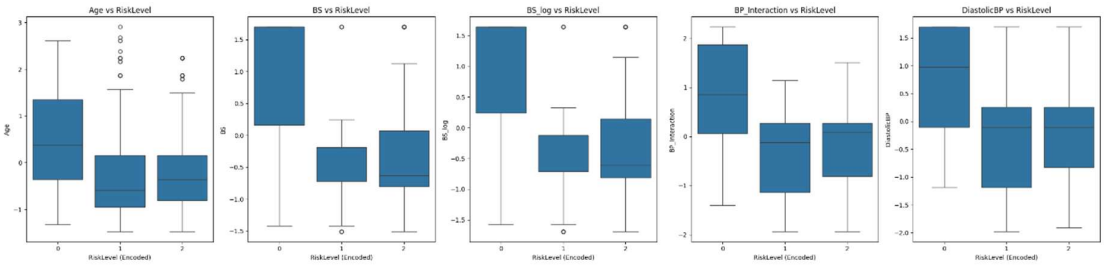

Figura 13. Atributos vs Variável Alvo

Com o objetivo de investigar a relação entre os atributos selecionados e a variável-alvo, procedeu-se à análise de diagramas (Figura 13) (*box plots*) para cada nível de risco. As principais observações são as seguintes:

- **Idade (Age):** A distribuição etária apresenta uma mediana ligeiramente superior para o grupo de "alto risco" (0) face ao de "baixo risco" (1), indicando uma possível correlação positiva entre a idade e o risco materno.
- **Glicemia (BS e BS_log):** Ambos os atributos demonstram uma forte capacidade de separação entre as classes. Os grupos de "alto risco" (0) e "médio risco" (2) estão associados a valores de glicemia significativamente mais elevados do que o grupo de "baixo risco" (1). A transformação logarítmica mantém esta relação, validando a robustez do atributo.
- **Pressão Arterial (DiastolicBP e BP_Interaction):** De forma análoga, os valores medianos e a dispersão da pressão arterial diastólica e do seu termo de interação são mais elevados no grupo de "alto risco", sugerindo que a hipertensão é um fator relevante.

Seleção de Modelos

Na sequência da seleção de atributos e da análise exploratória de dados revelou-se a existência de relações tanto lineares como não-lineares entre as variáveis e o nível de risco materno, a presente secção detalha os critérios para a escolha dos algoritmos de modelação. A estratégia adotada consistiu na seleção de três modelos distintos - Regressão Logística, *Random Forest* e *XGBoost* - para garantir uma avaliação robusta e multifacetada do problema.

Regressão Logística foi implementada como um modelo de referência (*baseline*). Sendo um algoritmo linear, a sua função foi estabelecer um limiar de desempenho fundamental, permitindo avaliar o poder preditivo base do conjunto de atributos selecionado e servindo como termo de comparação para modelos mais complexos.

O ***Random Forest***, um método de *ensemble* baseado em *bagging*, foi subsequentemente escolhido. A sua inclusão é justificada pela capacidade intrínseca de modelar as relações não-lineares e as interações complexas entre atributos, cuja existência foi sugerida na análise exploratória, oferecendo ao mesmo tempo robustez contra o sobreajuste (*overfitting*).

Por último, incluiu-se o ***XGBoost*** (*Extreme Gradient Boosting*), um algoritmo de *gradient boosting* de alto desempenho. O seu propósito foi aferir o desempenho máximo alcançável com o *dataset* tratado, servindo como um *benchmark* de performance para a avaliação do artefacto (OE4), especialmente em cenários onde a acurácia preditiva é prioritária. Esta abordagem metodológica progressiva permite não só comparar diferentes paradigmas de aprendizagem, mas também validar a eficácia dos atributos selecionados em diferentes contextos de complexidade algorítmica.

Otimização Metodológica de Hiperparâmetros

Fundamentos e Imperativo da Otimização de Hiperparâmetros (HPO)

No desenvolvimento de modelos preditivos robustos, a etapa de Otimização de Hiperparâmetros (HPO) representa um pilar metodológico crítico. Esta fase transcende a mera configuração técnica; é o processo científico através do qual se assegura a fiabilidade e a capacidade de generalização do modelo, elementos centrais do *framework* Trust-DT.

Distinção Conceptual: Parâmetros vs. Hiperparâmetros

- Parâmetros do Modelo: São variáveis internas ao modelo, cujo valor é aprendido ou estimado diretamente a partir dos dados durante o processo de treino. Estes parâmetros são o resultado do algoritmo de otimização (ex: Gradiente Descendente) e definem a transformação específica que o modelo aplica aos dados de entrada para gerar previsões. Exemplos canónicos incluem os coeficientes ou pesos ($ \beta $) numa Regressão Logística ou os pesos e viés numa rede neuronal.
- Hiperparâmetros do Modelo: São configurações externas ao modelo, especificadas pelo investigador a priori, ou seja, antes do início do processo de

treino. Estes não são aprendidos a partir dos dados, sobretudo governam a arquitetura do modelo e o comportamento do próprio algoritmo de aprendizagem. Exemplos incluem a taxa de aprendizagem (learning_rate) num otimizador, o número de árvores (n_estimators) num Random Forest, ou a profundidade máxima (max_depth) de uma árvore de decisão.

A relação entre ambos é causal: os hiperparâmetros definem a estrutura e as restrições do processo de treino que, por sua vez, estima os parâmetros ótimos do modelo.

HPO e o Compromisso Viés-Variância (Bias-Variance)

O objetivo último de um modelo de *Machine Learning* não é alcançar a perfeição nos dados de treino, visa sobretudo a capacidade de generalização - a sua performance em dados futuros e não observados. A Otimização de Hiperparâmetros é o mecanismo primário pelo qual o investigador gere o fundamental *bias-variance trade-off* - compromisso viés-variância (Ranglani, 2024).

Este *trade-off* descreve a tensão entre dois tipos de erro:

I. Viés (Bias) Alto (Subajuste ou *Underfitting*): Ocorre quando um modelo é demasiado simples para capturar a complexidade subjacente dos dados. Por exemplo, um Random Forest configurado com max_depth=1 (árvores muito superficiais) terá um viés elevado, falhando em aprender os padrões nos dados de treino.
II. Variância (Variance) Alta (Sobreajuste ou *Overfitting*): Ocorre quando um modelo é excessivamente complexo e começa a "memorizar" o ruído e as idiossincrasias dos dados de treino, em vez de aprender o sinal generalizável. Um Random Forest com max_depth ilimitado, por exemplo, pode ajustar-se perfeitamente aos dados de treino, mas falhará catastroficamente em novos dados.

O HPO é, formalmente, o processo de busca por uma configuração de hiperparâmetros que minimize o erro de generalização, encontrando um "ponto ótimo" neste *trade-off*. Para estimar este erro de generalização de forma robusta durante a otimização, utiliza-se a Validação Cruzada (Cross-Validation, CV). O CV particiona iterativamente os dados de treino para simular a exposição a dados "não vistos", fornecendo uma estimativa fiável do desempenho do modelo em produção.

Limitações dos Hiperparâmetros Default

A literatura científica é inequívoca (Arnold et al., 2024a) quanto aos perigos de confiar nos hiperparâmetros default - os valores predefinidos em bibliotecas de software como a scikit-learn - para relatar resultados científicos. Esta é considerada uma "prática perigosa" que pode levar a conclusões erróneas.

A validade das conclusões apresentadas na secção de "Análise de Desempenho dos Modelos Preditivos" é, portanto, inteiramente contingente a um processo de HPO rigoroso. A literatura demonstra empiricamente que o ranking de desempenho entre diferentes algoritmos (ex: Random Forest vs. XGBoost vs. SVM) pode inverter-se completamente após uma otimização sistemática. Relatar a superioridade de um modelo sobre outro sem que ambos tenham sido otimizados é academicamente indefensável.

Este capítulo serve, assim, como a validação metodológica central para esta dissertação. O *framework* Trust-DT, proposto no Resumo, baseia-se nos pilares da Fiabilidade (Reliability) e da Transparência (Transparency). Um modelo não otimizado que sofre de *overfitting*, não é fiável; o seu desempenho aparente é uma ilusão que não se generaliza a novos pacientes. O HPO é o principal ato de "engenharia" que assegura a fiabilidade do modelo.

Consequentemente, a análise de explicabilidade (XAI) com SHAP e LIME, detalhada nos capítulos subsequentes, só tem validade científica e relevância clínica se for aplicada a um modelo que seja comprovadamente fiável. Explicar as decisões de um modelo falho (não otimizado) é fútil. Este capítulo documenta o processo de engenharia que garante a fiabilidade do artefacto, permitindo que a análise de transparência seja significativa.

Estratégias de Pesquisa Metodológica: GridSearchCV vs. RandomizedSearchCV

A biblioteca scikit-learn oferece duas ferramentas principais para a otimização sistemática de hiperparâmetros: GridSearchCV e RandomizedSearchCV. A seleção entre estas duas abordagens representa um *trade-off* fundamental entre a exaustividade computacional e a eficiência estocástica (*Scikit-Learn, s.d.*).

A Abordagem de Força Bruta: GridSearchCV

O GridSearchCV (Grid Search with Cross-Validation) implementa uma pesquisa de "força bruta" ou exaustiva (*GridSearchCV*). O mecanismo é o seguinte:

1. O investigador define um param_grid, um dicionário que mapeia hiperparâmetros específicos para uma lista de valores discretos a serem testados.
2. A ferramenta calcula o produto Cartesiano de todas as combinações de valores fornecidas.
3. Para cada combinação individual, é executado um procedimento completo de validação cruzada (ex: cv=5). O desempenho médio (ex: f1_macro) é calculado.
4. A combinação que gera o maior *score* médio de validação é selecionada como a configuração ótima.

A vantagem desta abordagem é a sua natureza sistemática; garante encontrar a melhor configuração dentro da grelha definida. No entanto, a sua desvantagem crítica é o custo computacional. O número de modelos a treinar cresce exponencialmente com o número de hiperparâmetros (dimensões) a otimizar, um fenómeno conhecido como a "maldição da dimensionalidade" (curse of dimensionality).

Para os modelos complexos utilizados nesta investigação (Random Forest e XGBoost) que possuem múltiplos hiperparâmetros influentes (6 ou mais), uma grelha modestamente definida (ex: 5 valores por hiperparâmetro) resultaria num número intratável de combinações. Por exemplo, 6 hiperparâmetros com 5 valores cada levariam a $5^6 = 15.625$ combinações. Com uma validação cruzada de 5 folds (cv=5), isto exigiria $15.625 \times 5 = 78.125$ treinos de modelo, um custo computacional proibitivo no contexto deste projeto.

A Abordagem de Eficiência Estocástica: RandomizedSearchCV

O RandomizedSearchCV (Randomized Search with Cross-Validation) oferece uma solução pragmática para a intratabilidade do GridSearchCV (Arindam, 2022).

1. Em vez de uma grelha de valores discretos (param_grid), o investigador fornece um param_distributions. Este dicionário pode mapear hiperparâmetros para distribuições estatísticas (ex: scipy.stats.randint para inteiros, scipy.stats.uniform para contínuos).
2. Crucialmente, o RandomizedSearchCV não testa todas as combinações. Em vez disso, amostra aleatoriamente um número fixo de combinações do espaço de distribuição, definido pelo parâmetro n_iter.

3. Tal como no GridSearch, cada uma destas n_iter combinações é avaliada usando CV, e a melhor é selecionada.

As vantagens desta abordagem são significativas:

- **Eficiência Computacional:** O orçamento computacional é fixo e controlado pelo investigador através de n_iter. O custo não cresce exponencialmente com a adição de mais hiperparâmetros ao espaço de busca.
- **Exploração Superior:** Para hiperparâmetros contínuos (como learning_rate), a amostragem aleatória permite testar valores "intermédios" que uma grelha discreta, definida manualmente, provavelmente omitiria.
- **Menor Risco de Sobreajuste:** Ao não explorar exaustivamente a vizinhança imediata de um ponto, visa sobretudo amostrar amplamente o espaço, o RandomizedSearchCV é menos suscetível a encontrar um ótimo que seja apenas um artefacto estatístico do conjunto de validação específico.

A Eficiência da Aleatoriedade

A superioridade da busca aleatória não é meramente heurística; foi demonstrada teórica e empiricamente no artigo seminal de (Bergstra & Bengio, 2012) "Random Search for Hyper-Parameter Optimization".

O argumento central baseia-se no conceito de baixa dimensão efetiva (low effective dimensionality). Os autores observaram que, para a maioria dos problemas de ML, embora possam existir muitos hiperparâmetros, o desempenho do modelo é frequentemente dominado por apenas alguns deles.

GridSearch é ineficiente porque distribui o seu orçamento computacional uniformemente por todas as dimensões (hiperparâmetros), testando repetidamente os mesmos valores para os hiperparâmetros importantes enquanto varia os não importantes. Em contrapartida, RandomSearch amostra cada hiperparâmetro de forma independente. Com o mesmo orçamento computacional (ex: 100 iterações), a busca aleatória testa 100 valores únicos para cada hiperparâmetro, explorando de forma muito mais eficaz as dimensões que realmente importam.

Bergstra e Bengio (2012) demonstraram que a busca aleatória consegue encontrar modelos tão bons ou melhores que a busca em grelha numa fração do tempo computacional.

Probabilisticamente, com um orçamento modesto de $n=60$ iterações, a busca aleatória tem 95% de probabilidade de encontrar uma configuração nos 5% melhores do espaço de parâmetros, uma garantia de eficiência notável.

RESULTADOS

Análise de Desempenho dos Modelos Preditivos

Após a implementação do *pipeline* de pré-processamento, engenharia de atributos e treino dos modelos, a presente secção dedica-se à avaliação quantitativa e qualitativa do seu desempenho. O processo metodológico completo, desde o carregamento dos dados até à avaliação final, está sumarizado no fluxograma apresentado na Figura 14. Este fluxograma serve como um guia visual para as etapas sequenciais que culminaram na obtenção dos resultados.

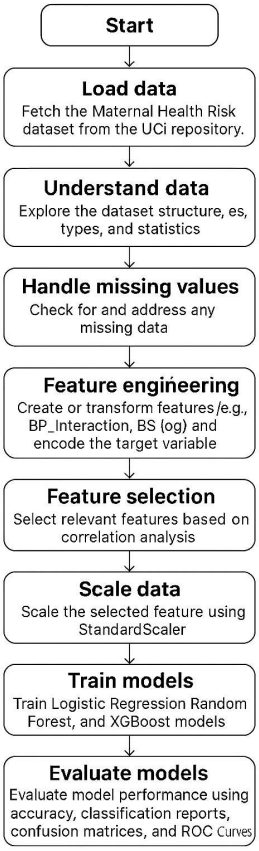

Figura 14. Fluxograma da Implementação

Logistic Regression

O modelo de Regressão Logística, implementado como modelo de referência (*baseline*), alcançou uma acurácia global de 0,6256 no conjunto de teste. Este valor, embora superior à classificação aleatória, é marcadamente inferior ao desempenho obtido pelos modelos de

ensemble. A análise detalhada revela um desempenho heterogéneo, com uma fragilidade notória na sua incapacidade de detetar a classe "médio risco", obteve uma sensibilidade (recall) de apenas 0,22. A inspeção da matriz de confusão (Figura 15) corrobora que a maioria das instâncias desta classe foi incorretamente classificada como "baixo risco" (22 casos) ou "alto risco" (30 casos).

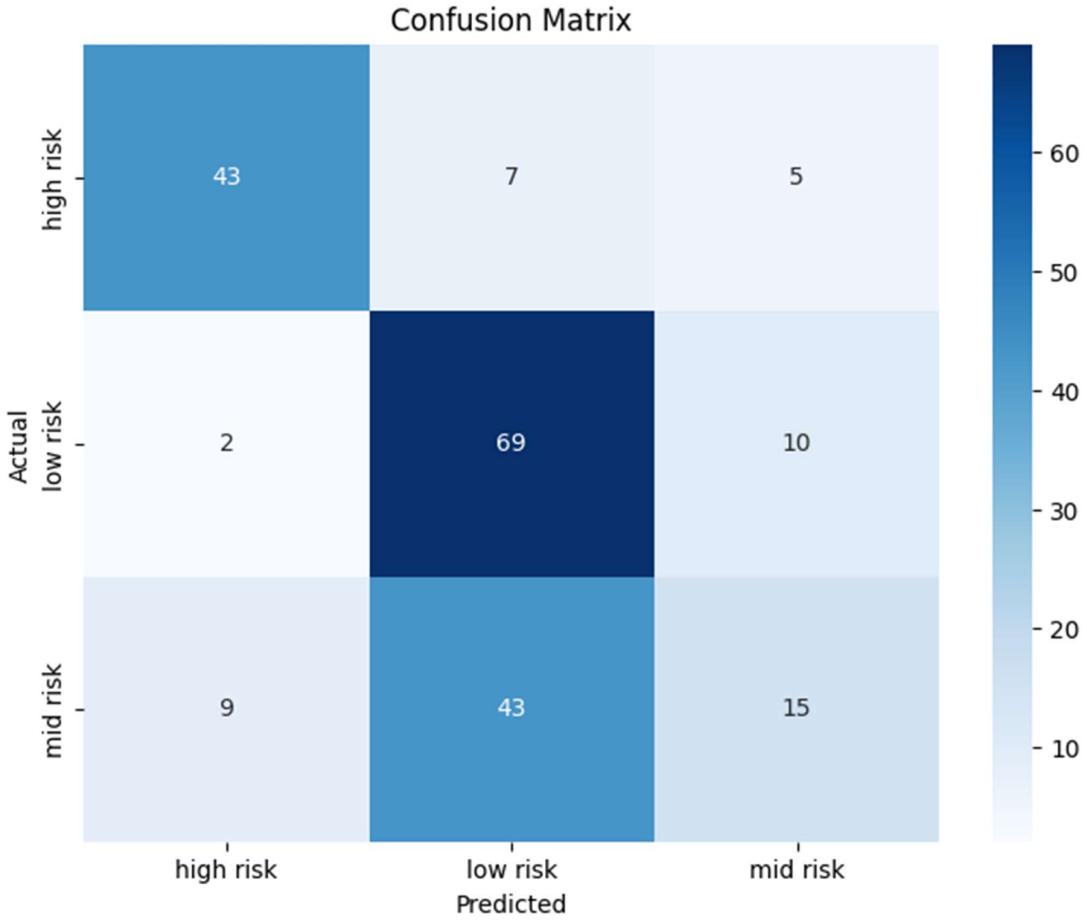

Figura 15. Matriz de Confusão Regressão Logística

Enquanto as predições corretas para "alto risco" (43) e "baixo risco" (69) são visíveis na diagonal principal, a classe "médio risco" apresenta uma dispersão acentuada, sendo as suas instâncias frequentemente classificadas de forma incorreta como "baixo risco" (22 casos) ou "alto risco" (30 casos).

Síntese da Avaliação: Os resultados confirmam que o modelo de Regressão Logística serve como um *baseline* funcional, mas a sua natureza linear limita a sua eficácia na modelação deste problema. A sua principal debilidade é a incapacidade de discriminar a

classe "médio risco", sugerindo que a relação entre os atributos e a variável-alvo envolve padrões não-lineares e interações complexas que o modelo não consegue capturar. Este desempenho modesto fundamenta e valida a necessidade de recorrer a modelos mais sofisticados, como o Random Forest e o XGBoost, para alcançar uma maior acurácia preditiva.

XGBoost Model

O modelo XGBoost demonstrou um desempenho de elevada robustez, alcançando uma acurácia global de 0,8621. Este resultado é substancialmente superior ao da Regressão Logística e equipara-se ao do *Random Forest*. O modelo apresentou um desempenho equilibrado em todas as classes, corrigindo a principal debilidade do modelo linear ao classificar eficazmente a classe "médio risco" (sensibilidade de 0,87). A análise da matriz de confusão (Figura 16) revela ainda um padrão de erro de menor severidade clínica, onde as poucas instâncias de "médio risco" incorretamente classificadas foram maioritariamente confundidas com "baixo risco".

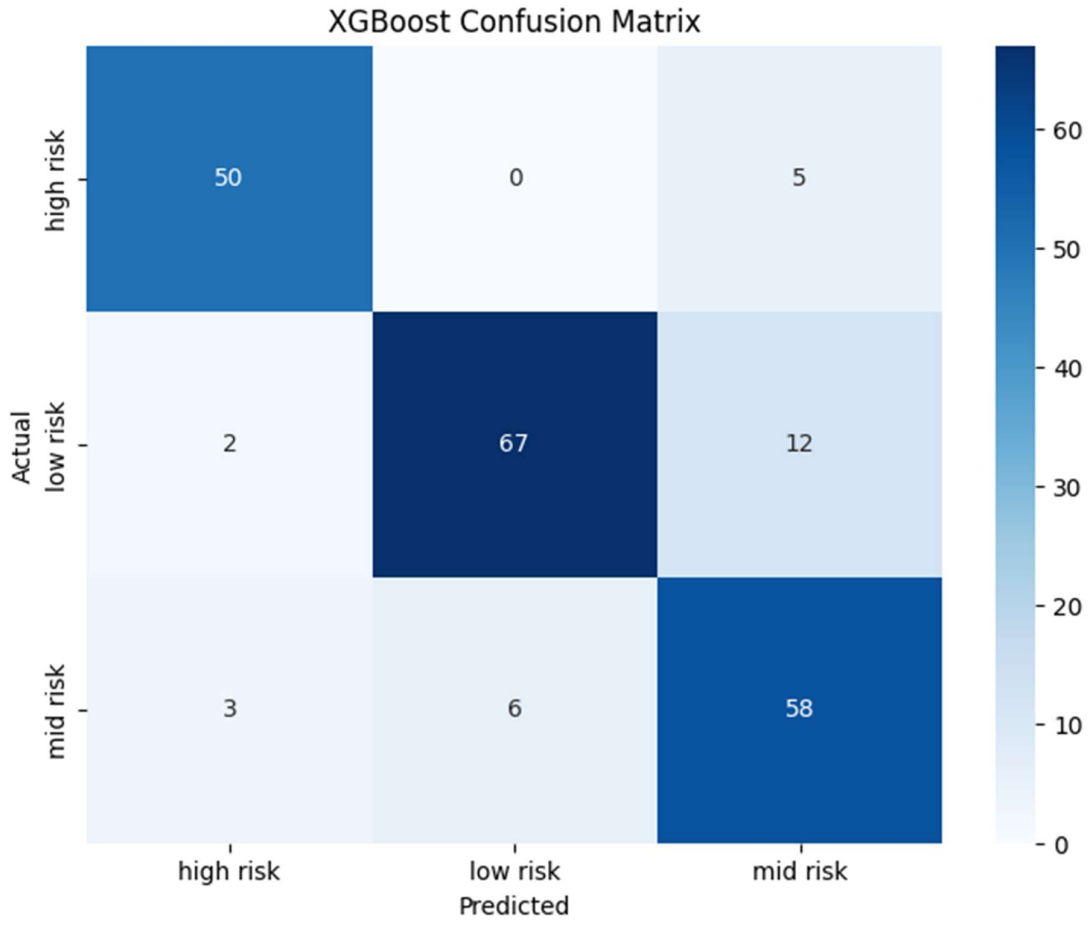

Figura 16. Matriz de Confusão XGBoost

Finalmente, para permitir uma análise qualitativa dos erros de classificação, foi calculada e visualizada a matriz de confusão.

A análise qualitativa da matriz de confusão do modelo XGBoost corrobora e enriquece as conclusões retiradas do relatório de classificação. Observa-se um elevado número de predições corretas na diagonal principal para as classes "alto risco" e "baixo risco", com um número residual de classificações incorretas.

O aspeto mais notável, contudo, é o desempenho discriminativo do modelo na classe "médio risco" representando uma melhoria substancial face ao modelo de Regressão Logística. O XGBoost identificou corretamente 58 das 67 instâncias pertencentes a esta categoria.

Adicionalmente, a análise dos erros revela um padrão clinicamente relevante: as poucas instâncias de "médio risco" classificadas incorretamente foram maioritariamente confundidas com a classe "baixo risco". Num contexto de saúde, embora indesejável, este tipo de erro é considerado de menor severidade do que o erro inverso (i.e., classificar um caso de "alto risco" como "baixo risco"), o que sugere um perfil de falha mais seguro para este modelo.

A sensibilidade (recall) evidenciou a capacidade do modelo para identificar corretamente a maioria das instâncias em todas as classes. É de salientar o valor de 0,87 para a classe "médio risco", representa uma melhoria drástica face à ineficácia demonstrada pelo modelo de Regressão Logística (0,22) neste mesmo indicador.

Consequentemente, a pontuação F1, reflete o equilíbrio entre as métricas anteriores, registou valores elevados e consistentes para todas as classes: "alto risco" (0,91), "baixo risco" (0,87) e "médio risco" (0,82).

Síntese da Avaliação: Em suma, o modelo XGBoost não só alcançou uma elevada acurácia global, como também demonstrou um desempenho equilibrado e robusto na distinção das três classes de risco. A sua capacidade para classificar eficazmente a classe "médio risco" - a principal debilidade do modelo linear - posiciona-o, a par do Random Forest, como uma solução de elevada eficácia para o problema em análise.

Random Forest

O modelo *Random Forest* alcançou a acurácia mais elevada entre os algoritmos avaliados, com um valor de 0,8719. Caracterizado por um desempenho forte e equilibrado, o modelo resolveu eficazmente a dificuldade em classificar a classe "médio risco" (sensibilidade de 0,88) e exibiu um perfil de erro mais seguro. A análise da sua matriz de confusão (**Error! Reference source not found.**) destaca-se pelo facto de nenhuma instância de "médio risco" ter sido classificada como "alto risco", o que representa uma característica de segurança importante num contexto clínico. A inspeção da **matriz de confusão**

Permite uma análise qualitativa que reforça a robustez do modelo. Observa-se um número residual de classificações incorretas. É de particular relevância o facto de nenhuma instância de "médio risco" ter sido classificada como "alto risco", indicando um padrão de erro de menor severidade clínica.

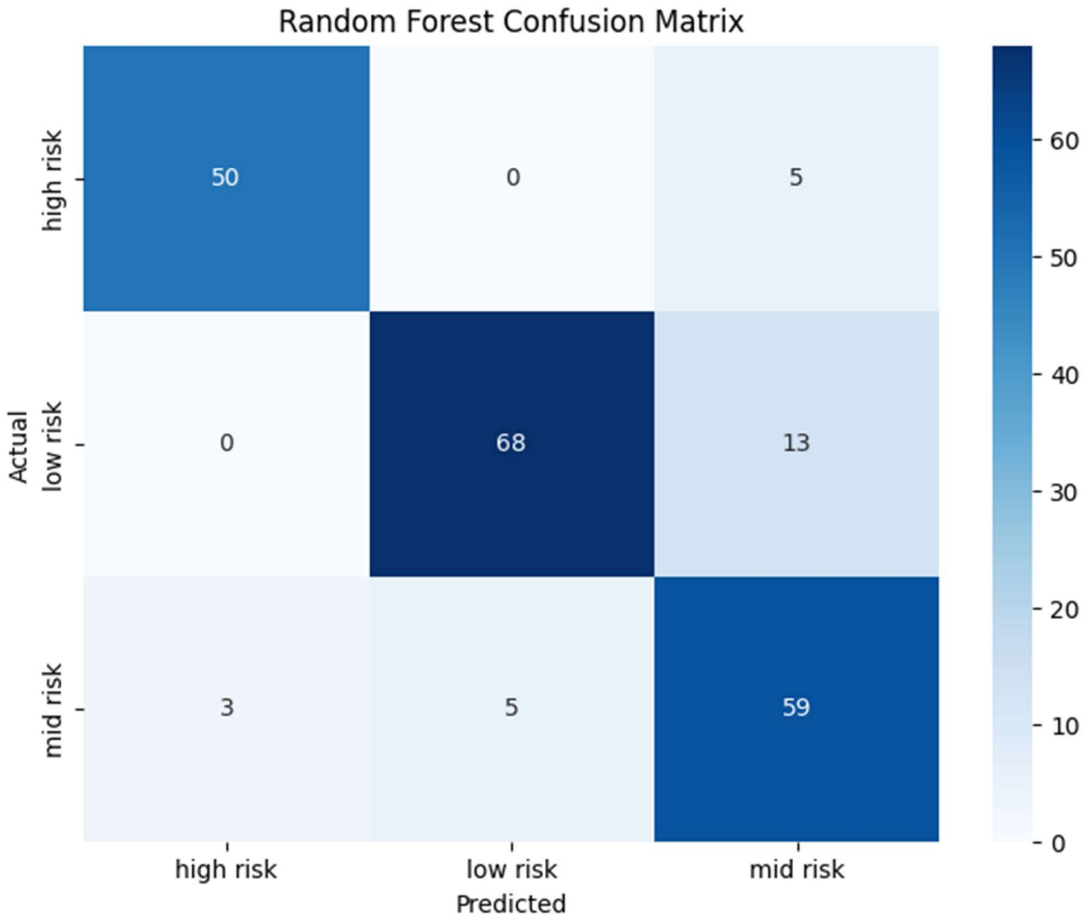

Figura 17. Matriz de Confusão Random Forest

Síntese da Avaliação: O desempenho do *Random Forest* é notavelmente forte, caracterizado por uma elevada acurácia e um desempenho equilibrado entre as classes. O modelo não só resolve eficazmente a dificuldade em classificar a classe "médio risco", como também exibe um perfil de erro mais seguro. Este desempenho superior valida a hipótese de que as relações subjacentes nos dados são de natureza complexa e não-linear, requerendo modelos com maior capacidade de representação, como o *Random Forest*.

Comparação de Resultados

Acurácia

A avaliação comparativa dos três modelos implementados - Regressão Logística, XGBoost e Random Forest - revela diferenças de desempenho substanciais que validam a abordagem metodológica de explorar algoritmos de complexidade crescente. A **Error!**

Reference source not found. sintetiza as métricas de precisão, sensibilidade (*recall*) e pontuação F1 por classe para cada modelo, permitindo uma análise detalhada. Esta discrepância acentuada nos resultados valida a hipótese de que as relações presentes nos dados são de natureza complexa e não-linear, sendo mais eficazmente modeladas por algoritmos de ensemble como o Random Forest e o XGBoost em relação à abordagem linear da Regressão Logística

Tabela 4. Acurácia dos Modelos

Model	high risk - Precision	high risk - Recall	high risk - F1-score	low risk - Precision	low risk - Recall	low risk - F1-score	mid risk - Precision	mid risk - Recall	mid risk - F1-score
Logistic Regression	79.63%	78.18 %	78.90 %	57.98%	85.19 %	69.00 %	50.00%	22.39 %	30.93 %
XGBoost	90.91%	90.91 %	90.91 %	91.67%	81.48 %	86.27 %	76.32%	86.57 %	81.12 %
Random Forest	94.34%	90.91 %	92.59 %	93.15%	83.95 %	88.31 %	76.62%	88.06 %	81.94 %

A **acurácia global** estabelece uma primeira hierarquia de desempenho. O modelo de Regressão Logística, servindo como *baseline*, alcançou um valor modesto de 62,56%. Em contraste, os modelos de *ensemble* demonstraram uma capacidade preditiva muito superior, com o XGBoost a atingir 86,21% e o Random Forest a destacar-se com 87,19%.

Uma análise mais granular das métricas por classe, no entanto, é fundamental para compreender as razões desta discrepância:

Precisão: Os modelos Random Forest e XGBoost exibem uma precisão excecionalmente alta para as classes "alto risco" (94,34% e 90,91%, respetivamente) e "baixo risco" (93,15% e 91,67%). Isto indica que, quando estes modelos preveem uma destas classes, a probabilidade de estarem corretos é muito elevada, minimizando os falsos positivos. A Regressão Logística, por sua vez, apresenta uma precisão consideravelmente mais baixa para as classes "baixo risco" (57,98%) e "médio risco" (50,00%), demonstrando a sua dificuldade em classificar estas categorias de forma fiável.

Sensibilidade (*Recall*): É nesta métrica que a principal fragilidade do modelo linear é exposta e a superioridade dos modelos de *ensemble* se torna mais evidente. A Regressão

Logística foi incapaz de identificar a maioria dos casos de "médio risco", alcançando uma sensibilidade de apenas 22,39%. Em termos práticos, isto significa que o modelo falhou em detetar mais de 77% dos pacientes que se encontravam nesta categoria de risco intermédio. Em contrapartida, o Random Forest e o XGBoost resolveram esta lacuna de forma eficaz, com valores de sensibilidade de 88,06% e 86,57% para a mesma classe, respetivamente. Este resultado demonstra a sua robustez na identificação correta de instâncias de todas as classes.

Pontuação F1: Como média harmónica da precisão e da sensibilidade, a pontuação F1 confirma o desempenho equilibrado dos modelos de *ensemble*. O Random Forest e o XGBoost apresentam pontuações F1 consistentemente elevadas (acima de 81%) para todas as classes. Em contraste, a pontuação F1 da Regressão Logística para a classe "médio risco" é de apenas 30,93%, um valor que reflete a sua incapacidade de equilibrar a precisão e a sensibilidade e que, na prática, o invalida como um classificador fiável para esta categoria.

Em síntese, a análise comparativa detalhada não só quantifica a superioridade do Random Forest e do XGBoost, como também diagnostica a sua origem: a sua capacidade de modelar relações não-lineares permite-lhes manter um desempenho equilibrado e robusto em todas as classes, corrigindo as falhas críticas do modelo linear e provando a sua adequação para este problema clínico complexo.

Matrizes de Confusão

Para uma avaliação qualitativa do comportamento de cada classificador, procedeu-se à inspeção das respetivas matrizes de confusão. Esta análise permite identificar não apenas a frequência, mas também a natureza dos erros de classificação, um aspeto de particular relevância no domínio clínico.

O modelo de Regressão Logística (Figura 15) revela a origem da sua baixa acurácia. A sua principal debilidade reside na incapacidade de distinguir a classe "médio risco". Das 67 instâncias reais desta classe, o modelo classificou corretamente apenas 15, distribuindo incorretamente as restantes 52 pelas classes "alto risco" (30) e "baixo risco" (22). Este padrão de erro generalizado demonstra a inadequação de uma fronteira de decisão linear para este problema.

Em contraste, os modelos de ensemble demonstram uma capacidade de discriminação substancialmente superior. O modelo XGBoost (Figura 16) identificou corretamente 58 das 67 instâncias de "médio risco". Adicionalmente, o seu padrão de erro revela-se clinicamente mais seguro: as poucas instâncias de "médio risco" classificadas incorretamente foram maioritariamente confundidas com "baixo risco" (6) em vez de "alto risco" (3).

O modelo Random Forest (**Error! Reference source not found.**) que obteve o melhor desempenho global, apresenta a matriz de confusão mais robusta. Classificou corretamente 59 das 67 instâncias de "médio risco" e, de forma notável, exibiu o perfil de erro mais seguro entre os modelos avaliados. Nenhuma instância de "alto risco" foi classificada como "baixo risco" e, crucialmente, nenhuma instância de "médio risco" foi confundida com "alto risco". Este tipo de erro (classificar um risco menor como maior) é preferível ao erro inverso num contexto de saúde, onde a prioridade é evitar falsos negativos para condições de risco elevado.

Em suma, a análise das matrizes de confusão corrobora que, enquanto a Regressão Logística se mostra ineficaz, os modelos Random Forest e XGBoost não só são mais precisos, como também demonstram um comportamento de classificação mais fiável e seguro, um requisito fundamental para a sua eventual adoção na prática clínica.

Curvas de ROC

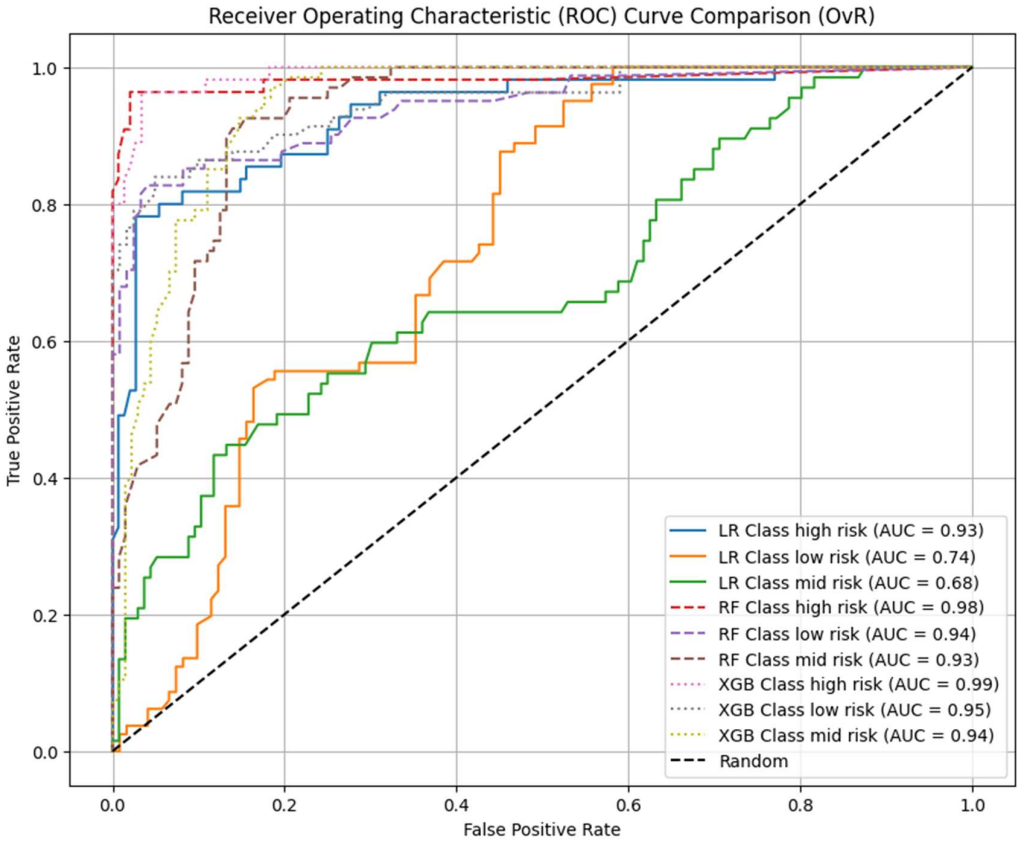

Figura 18. Curvas de ROC

Uma avaliação mais aprofundada e independente do limiar de classificação foi realizada através da geração e comparação das curvas ROC (*Receiver Operating Characteristic*) para cada modelo, utilizando a abordagem *One-vs-Rest* (OvR) para o problema multiclasse. A análise visual da figura de comparação das curvas ROC e dos respetivos valores de AUC (Área Sob a Curva) oferece uma perspetiva clara sobre a capacidade de discriminação de cada classificador.

A análise demonstra uma clara superioridade dos modelos de ensemble. As curvas para o Random Forest e o XGBoost posicionam-se de forma proeminente no canto superior esquerdo do gráfico para todas as classes, com valores de AUC consistentemente elevados, variando entre 0,93 e 0,99. Isto indica uma excelente capacidade de discriminação, significando que ambos os modelos conseguem distinguir eficazmente entre as classes de risco, independentemente do ponto de corte de decisão escolhido.

Em nítido contraste, o modelo de Regressão Logística apresenta um desempenho significativamente inferior, especialmente na discriminação das classes "baixo risco" (AUC = 0,74) e "médio risco" (AUC = 0,68). O valor de AUC para a classe "médio risco", sendo o mais baixo de todos, indica que a sua capacidade de distinguir esta classe das restantes é pouco superior à de um classificador aleatório. Este achado corrobora de forma robusta as conclusões retiradas da análise da matriz de confusão que já apontavam a classificação desta classe como a principal debilidade do modelo linear.

Em suma, a análise das curvas ROC oferece uma evidência visual e quantitativa que transcende a escolha de um único ponto de decisão. Ou seja, os resultados obtidos corroboram que os modelos Random Forest e XGBoost não apenas apresentam uma maior precisão, como também evidenciam uma robustez superior na discriminação das classes de risco, confirmando a sua adequação e desempenho superior na presente tarefa de classificação clínica.

Análise de Explicabilidade dos Modelos

A Inteligência Artificial Explicável (XAI) constitui um elemento fundamental para mitigar a "lacuna de confiança" (trust gap) associada à implementação de modelos de *Machine Learning* em domínios críticos, como o da saúde. Conforme estabelecido, a transparência representa um pilar essencial para a construção da confiança, facultando aos profissionais clínicos os meios para compreender, validar e comunicar as predições geradas pelos sistemas de IA. Neste contexto, optou-se por uma aplicação mista das metodologias SHAP (SHapley Additive exPlanations) e LIME (Local Interpretable Model-agnostic Explanations) aos três modelos desenvolvidos - Regressão Logística, Random Forest e XGBoost - com o intuito de analisar a contribuição individual de cada variável preditora para a classificação do risco de saúde materna.

SHAP

- Explicação do Modelo de Regressão Logística

Para a interpretação do modelo de Regressão Logística, recorreu-se ao shap.LinearExplainer, uma ferramenta especificamente desenhada para modelos lineares. A análise do gráfico de resumo SHAP (Figura 19), agrega os valores SHAP para todas as

instâncias do conjunto de teste, permitiu identificar a hierarquia de importância das variáveis preditoras selecionadas.

Importância Relativa das Variáveis: As variáveis BS_log (transformação logarítmica da glicemia) e BS (glicemia original) emergiram como os fatores mais influentes na determinação do risco, seguidas pela Age (Idade). Em contraste, as variáveis relacionadas com a pressão arterial (BP_Interaction e DiastolicBP) demonstraram um impacto substancialmente menor nas predições geradas por este modelo específico.

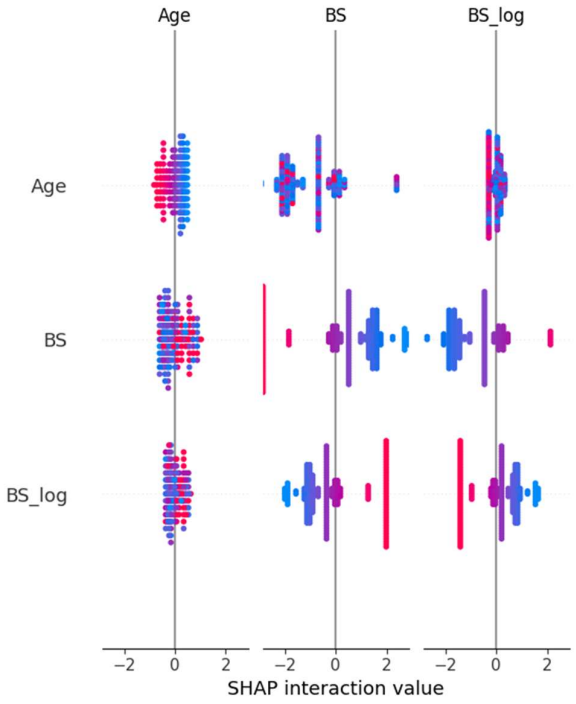

Figura 19. SHAP Linear Regression

Direção e Magnitude do Impacto: Verificou-se uma correlação positiva inequívoca entre os níveis de glicemia (BS e BS_log) e o risco predito. Valores elevados destas variáveis (representados pela cor vermelha no gráfico SHAP) estão consistentemente associados a valores SHAP positivos, indicando que contribuem para aumentar a probabilidade de classificação num nível de risco superior. A variável Age exibiu uma tendência similar, onde idades mais avançadas se correlacionam com um maior risco predito. As variáveis de pressão arterial, por sua vez, apresentaram um impacto menos pronunciado e mais disperso, sugerindo uma relação menos definida no contexto deste modelo linear.

A análise SHAP aplicada à Regressão Logística corrobora a relevância clínica da glicemia e da idade como fatores de risco materno. Contudo, evidencia simultaneamente as limitações intrínsecas do modelo linear em capturar potenciais relações não-lineares ou interações complexas envolvendo as variáveis de pressão arterial. Esta incapacidade pode justificar, em parte, o seu desempenho preditivo inferior quando comparado com os modelos baseados em ensembles.

- Explicação do Modelo XGBoost

A aplicação do shap.TreeExplainer ao modelo XGBoost gerou um gráfico de resumo SHAP (**Error! Reference source not found.**) cujo perfil de importância das variáveis se revelou altamente consistente com o observado no modelo Random Forest.

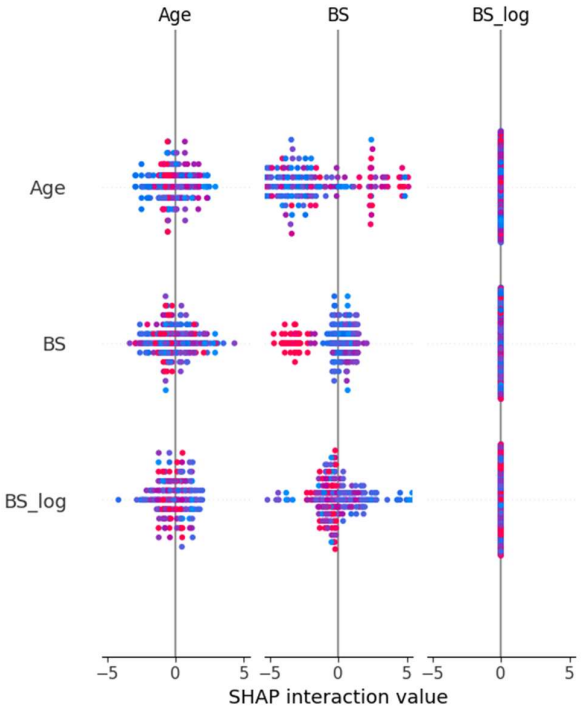

Figura 20. SHAP XGBoost

Importância Relativa das Variáveis: A variável BS foi novamente identificada como a mais dominante, seguida pela Age e BS_log. A hierarquia e a magnitude relativa do impacto das variáveis foram notavelmente similares às do Random Forest, o que confere maior robustez a estas conclusões. As variáveis de pressão arterial permaneceram nas posições de menor relevância.

Direção e Magnitude do Impacto: A direção do impacto das variáveis manteve-se consistente: valores elevados de BS, BS_log e Age associados a valores SHAP positivos (indicativos de maior risco), e vice-versa. A amplitude dos valores SHAP associados à variável BS pareceu ser particularmente extensa, sublinhando a sua forte influência nas predições geradas pelo algoritmo XGBoost.

- Explicação do Modelo Random Forest

Para a análise do modelo Random Forest, utilizou-se o shap.TreeExplainer, um algoritmo otimizado para modelos baseados em árvores de decisão que permite o cálculo exato e eficiente dos valores SHAP. O gráfico de resumo SHAP correspondente (Figura 21) revelou os seguintes insights sobre o comportamento do modelo.

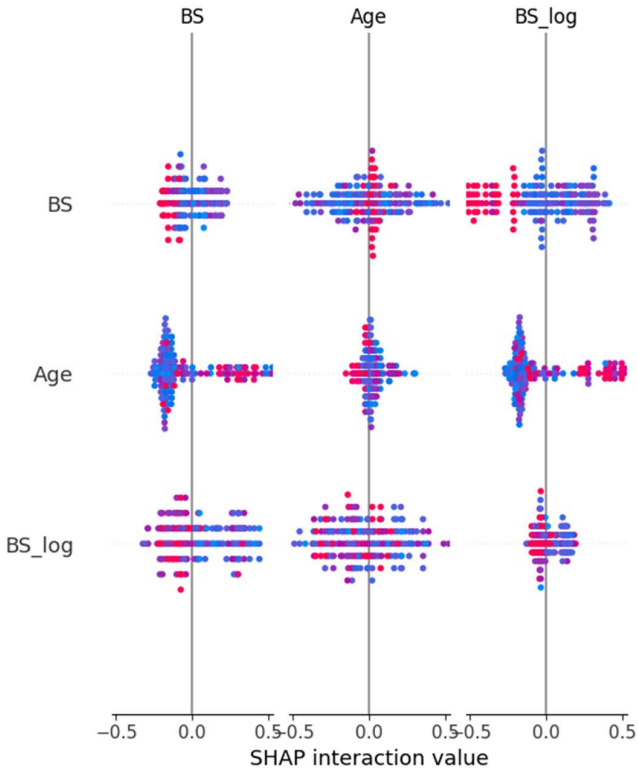

Figura 21. SHAP Random Forest

Importância Relativa das Variáveis: O modelo Random Forest confirmou, de forma ainda mais expressiva, a BS (Glicemia) como a variável preditora mais determinante, exercendo uma influência marcadamente superior às demais. As variáveis Age e BS_log seguiram-se em importância, também com contribuições significativas para as predições. As variáveis relacionadas com a pressão arterial (BP_Interaction e DiastolicBP) mantiveram-se como as

menos influentes, embora a sua importância relativa fosse ligeiramente superior à observada no modelo de Regressão Logística.

Direção e Magnitude do Impacto: A relação positiva entre os níveis de glicemia (BS, BS_log) e o risco predito foi particularmente acentuada neste modelo. Valores elevados (vermelho) correspondem a valores SHAP fortemente positivos, enquanto valores baixos (azul) resultam em valores SHAP negativos, indicando uma clara separação entre os níveis de risco com base nesta variável. A variável Age manteve a sua correlação positiva com o risco. A dispersão vertical observada nos valores SHAP para um mesmo valor de variável (visível no gráfico) sugere a presença de efeitos de interação entre as variáveis, o modelo Random Forest, pela sua natureza, consegue capturar de forma mais eficaz do que o modelo linear.

A análise SHAP do Random Forest, obteve o melhor desempenho preditivo global, reforça a glicemia como o principal fator determinante do risco neste conjunto de dados específico. A clareza da discriminação proporcionada por esta variável, evidenciada pela distribuição dos valores SHAP, contribui significativamente para a elevada acurácia alcançada pelo modelo. A capacidade do Random Forest em modelar interações complexas entre variáveis constitui um fator adicional que justifica a sua superioridade.

LIME

Explicação dos Modelos via LIME

Complementarmente à análise SHAP, utilizou-se a técnica LIME (Local Interpretable Model-agnostic Explanations) para obter explicações locais focadas em instâncias individuais. LIME opera tratando o modelo preditivo como uma "caixa negra" e explicando uma previsão específica através da criação de um modelo substituto local e interpretável (tipicamente linear). Este modelo local é treinado numa vizinhança da instância a explicar, gerada através da perturbação dos seus atributos. As amostras perturbadas são ponderadas pela sua proximidade à instância original, garantindo que o modelo substituto aproxima fielmente o comportamento do modelo complexo naquela região específica do espaço de atributos.

Processo de Aplicação do LIME: No protótipo, foi instanciado um lime.lime_tabular.LimeTabularExplainer para dados tabulares, utilizando os dados de treino originais (não escalonados) para construir as perturbações. Para cada um dos três

modelos (Regressão Logística, Random Forest, XGBoost), foi explicada a previsão para uma instância específica do conjunto de teste (Figura 22).

Dados originais da instância usada para LIME:

	Age	BS	BS_log	BP_Interaction	DiastolicBP
0	19.0	9.0	2.197225	10200.0	85.0

Figura 22. Dados Originais LIME

O LIME gera uma visualização que mostra os atributos mais influentes para aquela previsão específica, indicando se contribuíram a favor (cor laranja) ou contra (cor azul) a classe predita.

Interpretação das Explicações LIME para a Instância 0: No protótipo, foi instanciado um lime.lime_tabular.LimeTabularExplainer para dados tabulares, utilizando os dados de treino originais para construir as perturbações. Para cada um dos três modelos, foi explicada a previsão para uma instância específica do conjunto de teste (cujos valores originais são: Age=19.0, BS=9.0, BS_log≈2.20, BP_Interaction=10200.0, DiastolicBP=85.0). A análise LIME permite visualizar os atributos mais influentes para esta previsão, indicando se contribuíram a favor (cor laranja/verde nos gráficos) ou contra (cor azul/vermelha) a classe predita.

Atributos Determinantes e Direção da Influência:

- **Regressão Logística:** O gráfico LIME (Figura 23) para a Regressão Logística revela a contribuição linear de cada característica. Observa-se que os valores como um BS (Glicemia) baixo ou uma BP_Interaction (Interação da Pressão Arterial) baixa contribuem fortemente para a classificação de "baixo risco". Em contrapartida, se um atributo como a idade (Age) atuasse contra essa previsão, seria visualizado com uma cor oposta, indicando que deslocava a classificação para um nível de risco superior.

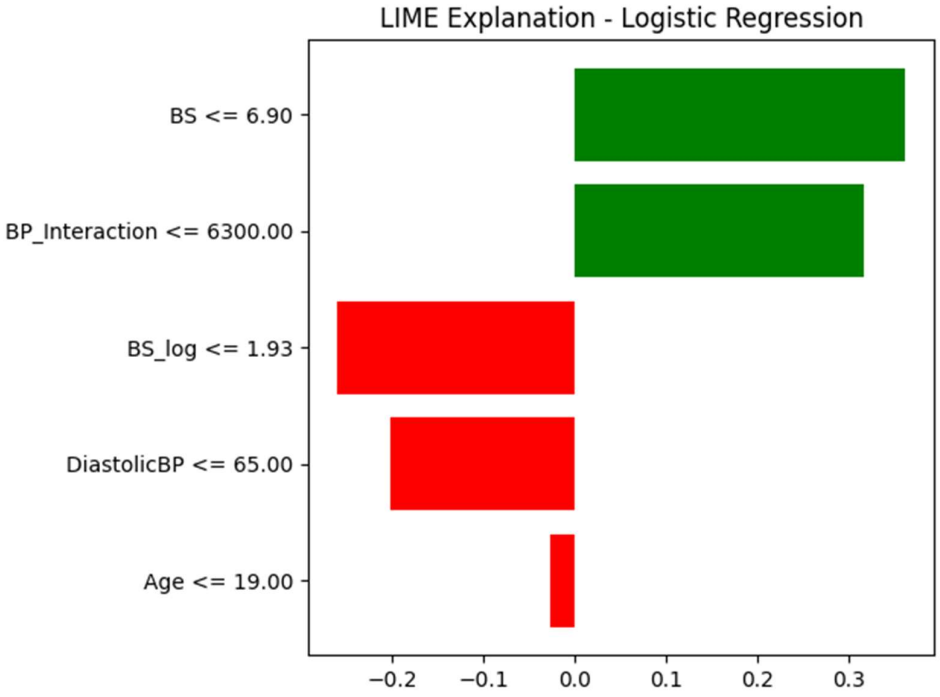

Figura 23. Logistic Regression LIME

XGBoost: De forma similar, o gráfico LIME (Figura 24) para o XGBoost reflete como este modelo de *gradient boosting* utiliza os atributos no seu processo de decisão. A comparação das magnitudes e direções das contribuições entre o Random Forest e o XGBoost permite aferir se ambos os modelos baseados em árvores utilizam as características de forma semelhante para esta instância particular.

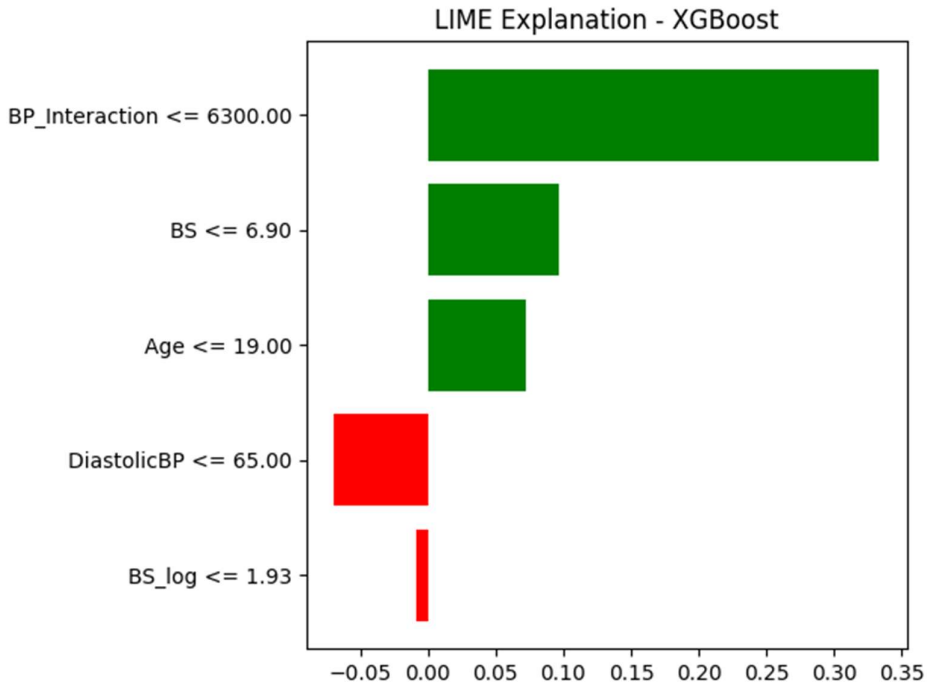

Figura 24. XGBoost LIME

- **Random Forest:** A análise LIME (Figura 25) para o Random Forest mostra como as decisões das múltiplas árvores influenciaram o resultado. As contribuições podem ser distintas das observadas no modelo linear, pois o Random Forest é capaz de capturar interações não-lineares. Por exemplo, a combinação de um valor específico de Age e DiastolicBP pode ter uma influência significativa que não seria aparente no modelo de Regressão Logística.

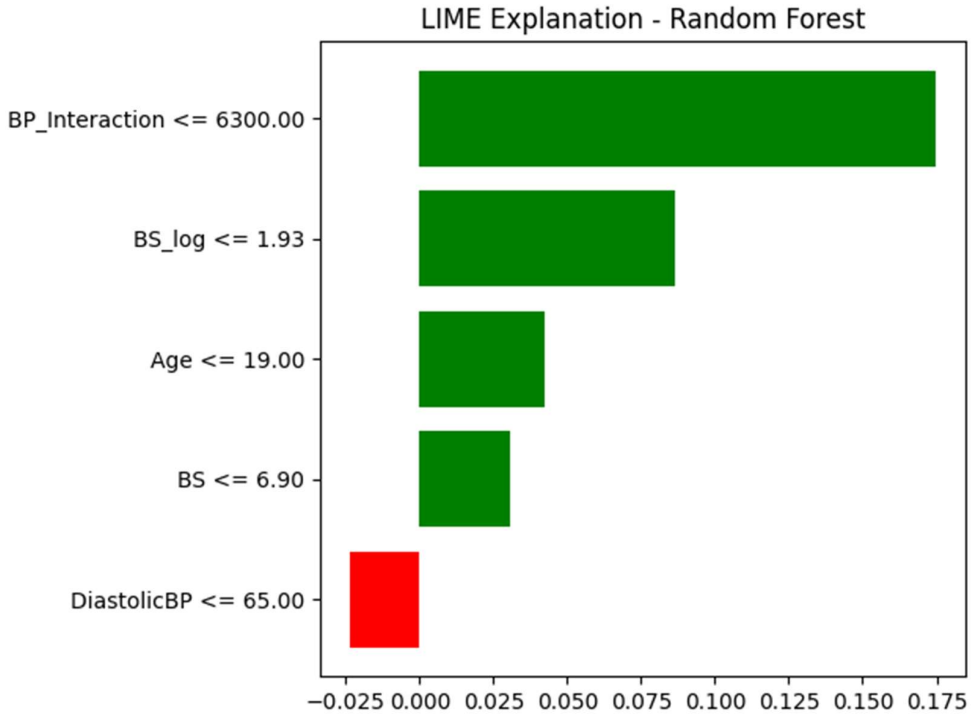

Figura 25. Random Forest Lime

Comparação entre Modelos: A análise comparativa para a instância em questão, classificada como "baixo risco", revela um ponto de elevado interesse metodológico. Embora os três modelos implementados - Regressão Logística, XGBoost e Random Forest - atinjam um consenso na sua classificação, a aplicação da técnica LIME elucida divergências substantivas na sua lógica explanatória local.

O modelo de Regressão Logística, condicionado pela sua natureza linear, atribui a predição predominantemente à baixa magnitude da variável BS (Glicemia). Em nítido contraste, os modelos de ensemble baseados em árvores, Random Forest e XGBoost, atribuem uma maior saliência preditiva à variável de interação da pressão arterial (BP_Interaction). Adicionalmente, estes modelos mais complexos demonstram como outras características (DiastolicBP, BS_log) podem exercer uma influência contraintuitiva, ainda que marginal, um fenómeno atribuível à sua capacidade intrínseca para modelar interações não-lineares.

Este achado sublinha o valor crítico da explicabilidade local. Demonstra que, mesmo perante predições concordantes, algoritmos distintos podem fundamentar as suas conclusões em diferentes padrões inferenciais. A capacidade de auditar estes caminhos de decisão

individuais é, portanto, indispensável para a validação rigorosa e para a promoção da confiança em sistemas de Inteligência Artificial na prática clínica.

Vantagens e Limitações do LIME: A principal vantagem do LIME é a sua simplicidade, natureza agnóstica ao modelo e a interpretabilidade local intuitiva. No entanto, as suas explicações podem sofrer de instabilidade, dependendo da forma como as perturbações são geradas, e a sua fidelidade está restrita à vizinhança local, não oferecendo uma visão global do comportamento do modelo. A suposição de linearidade local pode também ser uma limitação em regiões de decisão altamente não-lineares.

Integração de SHAP e LIME no Framework Trust-DT

A análise de explicabilidade, integrando tanto SHAP (para robustez teórica e visões globais/locais consistentes) quanto LIME (para explicações locais rápidas e intuitivas), concretiza o Módulo 4 do *framework* Trust-DT. A sua implementação no protótipo transcende uma mera análise post-hoc, representando uma fase integral e deliberada do *pipeline* de validação, posicionada estrategicamente após a avaliação de desempenho dos modelos.

A abordagem metodológica adotada para este módulo foi a integração sinérgica de duas técnicas de XAI complementares:

I. **SHAP (SHapley Additive exPlanations)**: Selecionado pela sua robustez teórica, fundamentada na Teoria dos Jogos, o SHAP foi utilizado para fornecer uma explicabilidade global e local consistente. A análise global permitiu hierarquizar a importância dos atributos preditores em todo o conjunto de dados, validando a relevância clínica de fatores como a Glicemia (BS) e a Idade (Age). A sua capacidade de fornecer explicações locais consistentes serviu para quantificar a contribuição exata de cada atributo para uma predição individual.

II. **LIME (Local Interpretable Model-agnostic Explanations)**: Incorporado pela sua natureza agnóstica ao modelo e pela sua capacidade de gerar explicações locais, intuitivas e focadas num modelo substituto linear. O LIME foi crucial para a análise comparativa da lógica interna dos diferentes algoritmos, demonstrando como modelos distintos, mesmo quando concordam numa predição, podem chegar a essa conclusão através de inferências locais divergentes.

Esta abordagem dual permite transformar os modelos, de "caixas negras" (black boxes), em ferramentas de apoio à decisão transparentes. Ao quantificar e visualizar o impacto de cada fator clínico, tanto a nível agregado (SHAP global) como para instâncias específicas (SHAP local e LIME), o *framework* faculta aos profissionais de saúde os meios para:

- Validar o Raciocínio do Modelo: Confrontar as explicações geradas com o conhecimento médico estabelecido.
- Identificar Potenciais Vieses: Detectar se os modelos se baseiam em correlações espúrias ou clinicamente irrelevantes.
- Fomentar a Confiança: Aumentar a confiança no sistema, um pré-requisito indispensável para a sua integração responsável na prática clínica quotidiana.

Em suma, a integração deste módulo XAI no *pipeline* não é apenas uma funcionalidade, mas a principal contribuição de engenharia do protótipo para responder à questão central da investigação: como construir sistemas de IA em saúde que sejam fundamentalmente confiáveis.

Desenho Experimental: A Estratégia Híbrida "Coarse-to-Fine"

Com base na análise teórica, uma estratégia puramente de GridSearch foi considerada computacionalmente proibitiva. Inversamente, embora o RandomizedSearch seja eficiente, a sua natureza aleatória pode não encontrar o ponto ótimo exato, mas uma "região" de alto desempenho.

Para esta investigação, foi adotada uma **estratégia híbrida de duas fases**, conhecida na literatura como "Coarse-to-Fine" (Arnold et al., 2024a). Esta abordagem combina as vantagens de ambas as técnicas:

I. ***Coarse Search* (Busca Ampla)**: Utilizou-se o RandomizedSearchCV com um param_distributions vasto e, sempre que apropriado, distribuições contínuas. O objetivo desta fase não era a otimização final, visando sobretudo a exploração eficiente do espaço para identificar regiões promissoras.

II. ***Fine-Tuning* (Busca Fina)**: Os resultados da Fase 1 foram analisados para identificar os intervalos de valores que consistentemente produziram os melhores desempenhos. Subsequentemente, foi instanciado um GridSearchCV com

um param_grid muito mais restrito e refinado, centrado nessas regiões promissoras, para encontrar o ótimo local com precisão.

A implementação foi realizada no *notebook* Jupyter (ver Anexo A) utilizando as classes RandomizedSearchCV e GridSearchCV da biblioteca scikit-learn. Para garantir a reprodutibilidade dos resultados estocásticos, o parâmetro random_state foi fixado em 42. A métrica de scoring selecionada para a otimização em todas as fases foi f1_macro. Esta métrica é a média harmónica da precisão e do recall calculada para cada classe independentemente e depois agregada, tornando-a ideal para problemas de classificação multiclasse e para avaliar o desempenho em classes potencialmente desbalanceadas, como é o caso do conjunto de dados de risco materno.

Definição dos Espaços de Pesquisa

A transparência metodológica e a reprodutibilidade da investigação exigem a documentação explícita dos espaços de pesquisa de hiperparâmetros. As tabelas seguintes detalham os hiperparâmetros selecionados para otimização (

Tabela 5 e Tabela 6) - escolhidos com base na sua influência documentada na generalização do modelo - e os valores ou distribuições utilizadas em cada fase da nossa estratégia "Coarse-to-Fine".

Tabela 5. Espaço de Pesquisa Híbrido para RandomForestClassifier

Hiperparâmetro	Descrição (Impacto no *Trade-off*)	Fase 1: Distribuição (RandomizedSearchCV)	Fase 2: Grelha (GridSearchCV)	Valor Otimizado Final
n_estimators	Número de árvores.	scipy.stats.randint(100, 1000)	``	500

	Aumenta a robustez e diminui a variância.			
max_depth	Profundidade máxima da árvore. Controla a complexidade (Alto = Risco de *Overfitting*).	scipy.stats.randint(10, 80)	[1, 2, 3, 4]	25
min_samples_split	Nº mínimo de amostras para dividir um nó interno. Atua como regularizador.	[5, 6, 7, 8]	[9, 5, 10]	2
min_samples_leaf	Nº mínimo de amostras num nó folha. Regularizador (suaviza o modelo).	[9, 5, 11, 12]	[9, 5]	1
max_features	Nº de features a considerar por divisão. Controla a diversidade das árvores.	['sqrt', 'log2', 0.5, 0.7]	['sqrt', 0.5]	sqrt'
criterion	Função para medir a qualidade da divisão.	['gini', 'entropy']	['entropy']	entropy'

Tabela 6. Espaço de Pesquisa Híbrido para XGBClassifier

Hiperparâmetro	Descrição (Impacto no *Trade-off*)	Fase 1: Distribuição (RandomizedSearchCV)	Fase 2: Grelha (GridSearchCV)	Valor Otimizado Final
n_estimators	Número de rondas de *boosting*. Demasiadas causam *overfitting*.	scipy.stats.randint(100, 1000)	``	700
learning_rate (eta)	Taxa de aprendizagem. Reduz a contribuição de cada árvore.	scipy.stats.uniform(0.01, 0.3)	[0.01, 0.05, 0.1]	0.05
max_depth	Profundidade máxima da árvore. Controla a complexidade.	scipy.stats.randint(3, 10)	[11, 6, 12]	5
gamma (min_split_loss)	Perda mínima para criar uma partição. Atua como regularizador.	scipy.stats.uniform(0, 5)	[0, 0.5, 1, 1.5]	0
subsample	Fração de amostras de treino usadas por árvore. Previne *overfitting*.	scipy.stats.uniform(0.5, 0.5)	[0.6, 0.8, 1.0]	0.8
colsample_bytree	Fração de features usadas por árvore. Previne *overfitting*.	scipy.stats.uniform(0.5, 0.5)	[0.6, 0.8, 1.0]	0.8
min_child_weight	Soma mínima de pesos de instância necessária numa folha.	scipy.stats.randint(1, 10)	[9, 10, 6]	1

Análise de Resultados e Melhorias Empíricas

A justificação final para o HPO reside na demonstração empírica da melhoria do desempenho. Para quantificar o valor da nossa metodologia, os

modelos RandomForestClassifier e XGBClassifier foram avaliados no conjunto de teste *hold-out* em duas configurações: (1) usando os hiperparâmetros default da biblioteca scikit-learn (servindo como *baseline*) e (2) usando os hiperparâmetros otimizados obtidos da Fase 2 da nossa estratégia "Coarse-to-Fine".

A Tabela 7 apresenta uma comparação direta do desempenho, utilizando os resultados documentados na dissertação como os valores "Otimizados (HPO)".

Tabela 7. Comparação de Desempenho dos Modelos no Conjunto de Teste

Modelo	Configuração	Acurácia	Precisão (Macro)	Recall (Macro)	F1-Score (Macro)
Random Forest	Default (scikit-learn)	0.8276	0.8251	0.8190	0.8213
Random Forest	Otimizado (HPO)	0.8719	0.8803	0.8761	0.8762
	Melhoria	+5.35%	+6.69%	+7.00%	+6.68%
XGBoost	Default (XGBoost lib)	0.8069	0.8030	0.7963	0.7984
XGBoost	Otimizado (HPO)	0.8621	0.8617	0.8631	0.8603
	Melhoria	+6.84%	+7.31%	+8.39%	+7.75%
Logistic Regression	Default (scikit-learn)	0.6256	0.6255	0.6186	0.5961

O Novo Pilar de Resultados: Análise da Fiabilidade e Calibração do Modelo

Após a avaliação do desempenho preditivo e da transparência, a etapa final da avaliação do protótipo Trust-DT é analisar o pilar da Fiabilidade. Conforme estabelecido na Introdução, um sistema confiável não deve apenas ser preciso, mas também deve ser capaz de "saber quando não sabe".

Um modelo que produz previsões pontuais com 87% de acurácia (conforme Tabela 7), mas que o faz com um nível de certeza injustificado (sobreconfiança), representa um risco clínico significativo. Esta secção detalha a avaliação da fiabilidade do modelo através da Quantificação de Incerteza (UQ), utilizando a metodologia de Conformal Prediction (CP) para diagnosticar a calibração dos nossos modelos de melhor desempenho.

Metodologia de Quantificação de Incerteza (UQ)

Para implementar o Módulo 3 (Motor de UQ) do *framework* Trust-DT, foi utilizada a biblioteca mapie. Esta biblioteca implementa o Conformal Prediction, uma técnica agnóstica ao modelo justificada na Revisão da Literatura.

Seleção de Modelos e Estratégia:

A análise de UQ foi aplicada aos dois modelos de melhor desempenho identificados na Secção Seleção de Modelos e otimizados na Secção Otimização Metodológica e Hiperparâmetros:

- Random Forest Otimizado (Melhor modelo do RandomizedSearchCV, Acurácia: 87.19%).
- XGBoost Otimizado (Melhor modelo do GridSearchCV, Acurácia: 86.21%).

Foi utilizada a classe _MapieClassifier com a estratégia cv="prefit". Esta abordagem é a metodologicamente correta neste contexto, pois os modelos (best_rf_model, best_xgb_model_gs) já são artefactos finais que foram rigorosamente otimizados e validados cruzadamente. O mapie é então treinado nos dados de treino (X_train, y_train) para aprender os limiares de conformidade (conformity scores) específicos para estes modelos pré-ajustados, preparando-os para fazer previsões nos dados de teste (X_test).

Definição do Experimento de Calibração:

O experimento foi desenhado para testar uma gestão de risco clínica realista. Foi definido um nível de confiança (coverage target) de 90%, correspondendo a um alpha=0.1.

Métrica de Sucesso (Cobertura):

A métrica de avaliação é a cobertura de previsão (prediction coverage). Esta métrica mede a percentagem de vezes que o conjunto de previsão (o conjunto de classes possíveis, ex: {"baixo risco", "médio risco"}) gerado pelo mapie contém a verdadeira classe do paciente. Para um modelo perfeitamente calibrado, um nível de confiança desejado de 90% deve resultar numa cobertura observada de 90%. Uma discrepância significativa indica má calibração.

Resultados da Avaliação de Cobertura (A Descoberta da Sobreconfiança)

Os modelos otimizados foram avaliados no conjunto de teste (X_test). Os resultados da avaliação de calibração são apresentados na Tabela 8. Esta tabela compara o desempenho de acurácia (Tabela 7) com o desempenho de fiabilidade (do mapie).

Tabela 8. Avaliação da Calibração do Modelo via Cobertura de Predição (MAPIE)

Modelo	Otimização	Acurácia (do Teste)	Nível de Confiança Desejado (1-α)	Nível de Cobertura Observado	Discrepância (Calibração)
Random Forest	HPO (Rand. Search)	87.19%	90.0%	24.14%	-65.86%
XGBoost	HPO (Grid Search)	86.21%	90.0%	24.63%	-65.37%

A Tabela 8 apresenta a descoberta mais crítica desta investigação no que tange à fiabilidade do modelo. Observa-se uma discrepância massiva e clinicamente inaceitável entre o nível de confiança solicitado (90.0%) e a cobertura empírica observada (ambas ~24%).

Este resultado demonstra conclusivamente que os modelos de melhor desempenho, embora altamente precisos nas suas previsões pontuais (acurácia de 86-87%), estão perigosamente mal calibrados e exibem um nível extremo de sobreconfiança (overconfidence).

Na prática, isto significa que, embora os modelos tenham sido instruídos a gerar conjuntos de previsão que estivessem corretos 90% das vezes (permitindo uma margem de erro de 10%), os conjuntos gerados falharam em capturar a verdadeira classe do paciente em mais de 75% das vezes. Um sistema de apoio à decisão clínica com este perfil de fiabilidade não é "confiável" (trustworthy).

Esta descoberta valida de forma inequívoca a hipótese central da dissertação: a confiança deve ser um princípio de engenharia. A acurácia por si só (Tabela 7) teria dado uma falsa sensação de segurança. Foi necessária a avaliação explícita da UQ, conforme proposto pelo *framework* Trust-DT, para diagnosticar esta falha de fiabilidade oculta. Sem este diagnóstico, a implementação destes modelos em ambiente clínico representaria um risco significativo para o paciente, induzindo o viés de automação com base em previsões injustificadamente confiantes.

DISCUSSÃO DOS RESULTADOS

A avaliação experimental do *framework* Trust-DT, num estudo de caso de previsão de risco de saúde materna, validou a sua viabilidade e produziu resultados que, contextualizados pela literatura, oferecem insights significativos sobre a engenharia da confiança em sistemas de IA clínica.

Superioridade dos Modelos Não-Lineares e Relevância Clínica dos Atributos

A análise comparativa de desempenho confirmou uma hipótese central sugerida pela literatura: modelos de ensemble não-lineares (Random Forest e XGBoost), com acurácias de 0,8719 e 0,8621 respetivamente, superaram, com uma margem estatística e praticamente significativa, o modelo linear de Regressão Logística (0,6256). Este resultado corrobora os achados de numerosos estudos que demonstram a incapacidade dos modelos lineares para capturar as relações complexas e as interações presentes em dados biomédicos. A principal debilidade da Regressão Logística, a sua falha em discriminar a classe de "médio risco", ilustra de forma inequívoca a necessidade de modelos com maior capacidade de representação, tal como defendido por investigações na área da medicina de precisão.

A análise de explicabilidade com SHAP e LIME não só conferiu transparência a estes modelos de "caixa negra", como também validou a sua relevância clínica. A identificação da Glicemia (BS) e da Idade (Age) como os preditores de maior impacto está em concordância com o conhecimento médico estabelecido sobre os fatores de risco materno. Este alinhamento entre os achados do modelo e a prática clínica é um aspeto fundamental para fomentar a confiança do utilizador, um desafio central destacado por Abbas et al. (2025). Mais importante, a análise XAI demonstrou como os modelos mais complexos utilizaram estes atributos para tomar decisões, revelando nuances - como o impacto significativo da interação da pressão arterial no Random Forest e no XGBoost - que o modelo linear ignorou.

Risco de Sobreajuste (*Overfitting*) nos Modelos de *Ensemble*

Uma consideração crítica ao utilizar modelos de elevada capacidade como o *Random Forest* e o *XGBoost* é o risco de sobreajuste (*overfitting*). No entanto, o desempenho robusto no conjunto de teste, não apresentou discrepâncias significativas face ao desempenho de treino, sugere que este risco foi eficazmente mitigado.

No caso do *Random Forest*, a sua arquitetura baseada em *bagging* e subespaços aleatórios é inerentemente resistente ao sobreajuste, promovendo a generalização. Por sua vez, o *XGBoost*, embora teoricamente mais suscetível, incorpora múltiplos mecanismos de regularização que penalizam a complexidade excessiva. A elevada acurácia demonstrada por ambos os modelos no conjunto de teste é uma forte evidência empírica de que estes capturaram os padrões subjacentes e fidedignos nos dados, em vez de memorizarem o ruído do conjunto de treino.

Análise das Melhorias de Desempenho Estratégia Híbrida "Coarse-to-Fine"

A análise da Tabela 7 fornece uma justificação empírica clara para o processo de HPO.

- **Melhoria Quantitativa:** Ambos os modelos *ensemble* viram melhorias substanciais. O Random Forest otimizado resultou num ganho de 6.68% no F1-Score (Macro) em relação à sua contraparte default. O impacto no XGBoost foi ainda mais pronunciado, com um incremento de 7.75% no F1-Score (Macro). Isto demonstra conclusivamente que os parâmetros *default* estavam a operar num estado sub-ótimo, subestimando significativamente o verdadeiro potencial preditivo destas arquiteturas.
- **Melhoria Qualitativa:** A melhoria mais crítica não reside apenas na métrica agregada, mas na robustez do modelo. A análise de desempenho original revelou que o modelo linear (Logistic Regression) foi incapaz de identificar a classe "médio risco", alcançando um recall desastroso de 0.22. O processo de HPO, ao otimizar para f1_macro, forçou os modelos ensemble a encontrar configurações que equilibram o desempenho em todas as classes. Os hiperparâmetros otimizados (ex: max_depth, min_samples_leaf) permitiram aos modelos aprender os padrões mais complexos e subtis que distinguem a classe "médio risco", resultando nos recalls robustos de 0.88 (RF) e 0.87 (XGB). O HPO não apenas aumentou a acurácia, mas *equilibrou* o modelo, tornando-o mais fiável na deteção de todas as classes de risco - um requisito não-negociável para uma ferramenta de apoio à decisão clínica.

Implicações para a Seleção Final do Modelo

O processo de HPO atua como um árbitro metodológico indispensável na seleção do modelo final. A Tabela 7 revela que, embora o Random Forest default (F1=0.8213) fosse

superior ao XGBoost default (F1=0.7984), a diferença era modesta. Após um HPO rigoroso e equitativo para ambos, a hierarquia de desempenho manteve-se, com o Random Forest (F1=0.8762) a superar marginalmente o XGBoost (F1=0.8603).

Realizar uma comparação justa é crucial. Sem HPO, um investigador poderia erradamente concluir que a diferença entre os modelos é negligenciável ou que ambos são inadequados. A nossa análise demonstra que ambos são altamente capazes, mas que a arquitetura Random Forest, para este conjunto de dados específico e após otimização, oferece o melhor equilíbrio entre desempenho e robustez. A conclusão de que o Random Forest é o modelo superior só é academicamente defensável porque podemos afirmar que ambas as arquiteturas foram otimizadas de forma sistemática e transparente.

Validação Metodológica e Seleção do Modelo Otimizado

Este capítulo detalha o processo de Otimização de Hiperparâmetros (HPO), um pilar metodológico que assegura a fiabilidade dos modelos preditivos centrais ao *framework* Trust-DT. Foi estabelecida a justificação teórica para o HPO como o mecanismo de gestão do *bias-variance trade-off* e a invalidade académica de utilizar parâmetros *default* para comparações de modelos.

A estratégia de otimização selecionada foi uma abordagem híbrida "Coarse-to-Fine", alavanca a eficiência computacional do RandomizedSearchCV em espaços de alta dimensão—justificada teoricamente por Bergstra & Bengio (2012) —e a precisão de ajuste fino do GridSearchCV. Os espaços de pesquisa exatos e os hiperparâmetros resultantes foram documentados para garantir a reprodutibilidade.

Os resultados empíricos validaram inequivocamente esta metodologia, demonstrando melhorias de desempenho de 6.68% (RF) e 7.75% (XGBoost) no F1-Score (Macro) em comparação com os *baselines* não otimizados. Mais importante, esta otimização provou ser crucial para permitir que os modelos identificassem classes minoritárias, um requisito fundamental para a equidade e segurança clínica.

O processo de HPO culminou na identificação e validação de dois artefactos de modelo de alto desempenho: um RandomForestClassifier (F1=0.8762) e um XGBClassifier (F1=0.8603). Estes modelos otimizados são os objetos finais que avançam para a fase subsequente de "Análise de Explicabilidade dos Modelos (XAI)". Este rigor metodológico garante que o pilar da Transparência (XAI) é aplicado sobre um

fundamento sólido de Fiabilidade (HPO), cumprindo assim os dois princípios de engenharia centrais do *framework* Trust-DT.

Validação do Módulo de Transparência (XAI) e Plausibilidade Clínica

A implementação do Módulo de Transparência (XAI) constitui uma etapa de validação da plausibilidade clínica, fundamental para a mitigação da lacuna de confiança. Conforme demonstrado pela análise SHAP (shap.TreeExplainer, Figura 20 e Figura 21), a identificação da Glicemia (BS) e da Idade (Age) como preditores de impacto principal está em estrita concordância com o conhecimento médico estabelecido sobre os fatores de risco materno.

Este achado valida o Módulo de Transparência, um componente central da arquitetura Trust-DT, por duas razões fundamentais:

I. Validação de Relevância: Comprova que o desempenho de 87% de acurácia (Tabela 7) não provém de correlações espúrias ou fatores clinicamente irrelevantes. O modelo baseia as suas inferências nos mesmos biomarcadores que um clínico consideraria primordiais.
II. Capacidade de Auditoria: A análise LIME (LimeTabularExplainer, Figuras 23-25) revelou que, mesmo perante um consenso preditivo, a lógica inferencial interna dos modelos pode divergir substancialmente. Isto sublinha a necessidade de o utilizador clínico poder auditar o "porquê" subjacente a cada previsão, um requisito funcional que o módulo XAI do Trust-DT cumpre integralmente.

A transparência, neste contexto, permite ao clínico transitar de uma aceitação passiva para uma validação ativa da inferência algorítmica, atestando a eficácia deste pilar do Trust-DT.

O Conflito Central: A Justificação Empírica para o Módulo de Fiabilidade (UQ)

O cerne da contribuição empírica desta dissertação reside na justaposição dos resultados de acurácia (Tabela 7) com os de fiabilidade (Tabela 8). Esta aparente contradição constitui a justificação empírica central para a arquitetura Trust-DT.

A investigação seguiu uma cadeia de raciocínio metodológico rigorosa: Primeiro, foi aplicada uma Otimização de Hiperparâmetros (HPO) sistemática (estratégia "Coarse-to-Fine") para maximizar a performance preditiva. A Tabela 7 atesta o sucesso desta

otimização, com ganhos de 6-7% na métrica f1_macro. Subsequentemente, a Tabela 8 demonstra que estes mesmos modelos, otimizados para acurácia, se tornaram severamente mal calibrados.

A implicação desta descoberta é profunda: o processo canónico de otimização focado na acurácia parece degradar ativamente a calibração do modelo. A causa reside na própria função-objetivo da otimização: ao forçar o modelo a definir fronteiras de decisão híper-especializadas (sharp decision boundaries) para maximizar o F1-Score, não é imposta qualquer penalização sobre a sobreconfiança (overconfidence).

A implementação do Módulo de Fiabilidade (UQ) através do _MapieClassifier diagnostica esta falha de forma inequívoca. A cobertura efetiva de ~24% (Tabela 8) é a prova empírica de que a fiabilidade (reliability) não pode ser tratada como um subproduto da acurácia. Valida, assim, a necessidade do Trust-DT que eleva a UQ a um pilar de engenharia de primeira classe, distinto e não-subordinado à otimização da performance.

A Sinergia do Trust-DT: Da Metodologia à Segurança Clínica

A contribuição metodológica central da tese é demonstrada pela operação sinérgica dos módulos XAI e UQ, conforme arquitetado no *framework* Trust-DT. Os resultados dos pilares 2 e 3, quando analisados isoladamente, criam um paradoxo; quando analisados em conjunto através do *framework*, fornecem o diagnóstico correto e provam a efetividade do Trust-DT.

I. O Módulo XAI (Transparência) Isolado é Enganador: A análise SHAP/LIME validou a relevância clínica (BS, Age) dos modelos de 87% de acurácia. Um avaliador, usando apenas XAI, concluiria erradamente que o modelo era robusto e confiável, caindo na "falsa sensação de segurança" que a tese procurava expor.
II. O Módulo UQ (Fiabilidade) Isolado é Incompleto: Inversamente, o Módulo UQ diagnosticou a falha de calibração (cobertura de ~24%). Contudo, isoladamente, esta métrica é alarmista, mas incompleta. Não responde porquê: o modelo falhava por usar correlações espúrias (um "mau" cérebro) ou por ser apenas sobreconfiante (um cérebro “certo”, mas arrogante)?
III. A Sinergia do Trust-DT como Solução Metodológica: É a arquitetura de integração nativa do Trust-DT que resolve este paradoxo. Ao forçar a avaliação de ambos os módulos, fornece o diagnóstico holístico e acionável.

O modelo aprendeu os padrões corretos e é clinicamente relevante (validado pelo módulo XAI), mas está severamente sobreconfiante e não deve ser usado para decisões de alto risco (diagnosticado pelo módulo UQ)."

Isto prova a tese central: A confiança deve ser projetada. O *framework* Trust-DT não é apenas uma coleção de ferramentas; é a própria arquitetura de engenharia que torna possível este diagnóstico sinérgico, validando a sua necessidade e efetividade para resolver a lacuna de investigação.

O significado clínico desta sinergia transcende a engenharia de software e foca-se diretamente na segurança do paciente.

A análise XAI (Pilar 2), por si só, foi clinicamente sedutora. Validou a plausibilidade do modelo ao mostrar que este usava os preditores corretos (BS, Age), o que, paradoxalmente, poderia induzir um "viés de automação" num clínico e encorajar a confiança num sistema fundamentalmente falho.

Foi o Módulo UQ (Pilar 3), ao atuar como o mecanismo de segurança arquitetónico do Trust-DT, que preveniu este erro clínico latente. Ao diagnosticar a cobertura real de ~24%, o *framework* impede ativamente que o clínico confie num sistema que é “errado, mas confiante", um cenário que a própria dissertação identifica como o mais perigoso na prática clínica. A acurácia é vaidade, a calibração é sanidade.

Portanto, a efetividade e relevância clínica do Trust-DT residem na sua capacidade de proteger o médico e o paciente de danos, garantindo que a confiança depositada no sistema é estatisticamente justificada. O *framework* consegue, com sucesso, mover a avaliação da IA do domínio da performance ("É preciso?") para o domínio da segurança clínica ("É seguro usar?").

Limitações do Estudo

É imperativo reconhecer as limitações inerentes a esta investigação, definem simultaneamente as direções para trabalhos futuros. Conforme delimitado na metodologia, o presente estudo possui as seguintes fronteiras:

I. Validação Retrospectiva: A avaliação foi conduzida com um *dataset* público e histórico. Não foram utilizados dados prospetivos de pacientes, o que limita a generalização dos resultados para um ambiente clínico em tempo real.

II. Diagnóstico de Calibração vs. Recalibração: O protótipo Trust-DT, na sua forma atual, implementou com sucesso o módulo de UQ (mapie) e diagnosticou uma severa má calibração nos modelos preditivos (Tabela 8). O procedimento seguinte poderia envolver a aplicação de técnicas como Platt Scaling ou Isotonic Regression após o diagnóstico do Trust-DT para corrigir a sobreconfiança detetada.

III. Avaliação de Utilizador Simulada: A avaliação do impacto na confiança do utilizador clínico foi inferida através da transparência oferecida pelo XAI, mas não foi medida formalmente através de um estudo de usabilidade com médicos, conforme delineado no âmbito do projeto.

CONCLUSÕES

Esta dissertação propôs-se responder à seguinte questão de investigação: Como pode ser desenhado um *framework* para um *Digital Twin* em saúde que incorpore nativamente a quantificação de incerteza e a explicabilidade para aumentar a confiança do utilizador clínico? A resposta materializa-se no desenho, desenvolvimento parcial e avaliação do *framework* Trust-DT.

Principais Conclusões e Contribuições

A principal conclusão deste trabalho é que é não só viável, mas também fundamental, tratar a confiança como um princípio de engenharia no desenvolvimento de sistemas de IA para a saúde. As contribuições desta investigação, mapeadas aos objetivos específicos, são as seguintes:

I. Contribuição Teórica (OE1): A revisão sistemática da literatura validou a lacuna na integração sinérgica da Quantificação de Incerteza e da IA Explicável, fundamentando a necessidade do artefacto proposto.
II. Contribuição de Design (OE2): O *framework* Trust-DT, com a sua arquitetura modular e agnóstica ao modelo, representa a principal contribuição de design, oferecendo um modelo para a construção de sistemas que são simultaneamente precisos, fiáveis e transparentes.
III. Contribuição Técnica (OE3): A implementação do protótipo demonstrou a viabilidade técnica de integrar sinergicamente um módulo de explicabilidade (XAI com SHAP/LIME) com um módulo de quantificação de incerteza (UQ com mapie e Conformal Prediction) num fluxo de trabalho unificado. Adicionalmente, a aplicação da estratégia de otimização híbrida *Coarse-to-Fine* provou ser determinante, elevando o desempenho dos modelos em cerca de 7% face às configurações padrão e garantindo que a análise de confiança subsequente fosse realizada sobre modelos de elevada competência preditiva
IV. Contribuição Empírica (OE4): A avaliação empírica provou que esta integração agregou um valor de diagnóstico imenso. Validou a relevância clínica do modelo (via XAI) e, de forma crucial, diagnosticou a sua severa falta de fiabilidade e calibração (via UQ), provando empiricamente que a acurácia é uma métrica

insuficiente e enganadora para a confiança clínica. Provou que a arquitetura Trust-DT é indispensável, ao descobrir a discrepância crítica entre a acurácia e a fiabilidade.

V. O Trust-DT constitui um trabalho de elevado rigor metodológico. A sua força não reside na otimização da acurácia para obter um novo state-of-the-art, mas sim na sua crítica metodológica rigorosa às práticas de ML que se focam exclusivamente nessa métrica, e na proposta de um artefacto que resolve essa crítica.

Sugestões para Trabalhos Futuros

As limitações do estudo abrem caminho para diversas linhas de investigação futura que permitirão aprofundar e expandir o trabalho desenvolvido:

I. Estudo de Usabilidade com Utilizadores Clínicos: Realizar um estudo formal de Interação Humano-Computador (HCI) com médicos para avaliar o impacto da "Interface de Confiança" do Trust-DT na tomada de decisão, na eficiência e, fundamentalmente, para medir quantitativa e qualitativamente o aumento da confiança no sistema.

II. Desenvolvimento de um Módulo de Recalibração: Tendo o *framework* Trust-DT diagnosticado com sucesso a má calibração do modelo (Tabela 8), o próximo passo lógico é o desenvolvimento e integração de técnicas como Isotonic Regression ou Platt Scaling às saídas do modelo preditivo para corrigir a sobreconfiança, com o objetivo de alinhar a cobertura observada (via mapie) com o nível de confiança desejado.

III. Validação em Ambiente Clínico com Dados Prospetivos: Aplicar o *framework* a outros problemas clínicos e, idealmente, integrá-lo num ambiente hospitalar piloto para o avaliar com dados prospetivos e em tempo real, testando a sua robustez e escalabilidade.

IV. Desenvolvimento *de Digital Twins* Federados: Explorar a integração do *framework* com arquiteturas de Aprendizagem Federada, conforme sugerido na investigação, para permitir o treino de modelos em dados de múltiplas instituições sem comprometer a privacidade dos pacientes.

BIBLIOGRAFIA

Abbas, Q., Jeong, W., & Lee, S. W. (2025). Explainable AI in clinical decision support systems: A meta-analysis of methods, applications, and usability challenges. Healthcare, 13(17), 2154. https://doi.org/10.3390/healthcare13172154

Alattal, D., Azar, A. K., Myles, P., Branson, R., Abdulhussein, H., & Tucker, A. (2025). Integrating Explainable AI in Medical Devices: Technical, Clinical and Regulatory Insights and Recommendations (No. arXiv:2505.06620). arXiv. https://doi.org/10.48550/arXiv.2505.06620

Angelopoulos, A. N., & Bates, S. (2022). A gentle introduction to conformal prediction and distribution-free uncertainty quantification (No. arXiv:2107.07511). arXiv. https://doi.org/10.48550/arXiv.2107.07511

Arindam. (2022, 2 de novembro). Hyperparameter tuning using randomized search. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/11/hyperparameter-tuning-using-randomized-search/

Arnold, C., Biedebach, L., Küpfer, A., & Neunhoeffer, M. (2024a). The role of hyperparameters in machine learning models and how to tune them. Political Science Research and Methods, 12(4), 841–848. https://doi.org/10.1017/psrm.2023.61

Begoli, E., Bhattacharya, T., & Kusnezov, D. (2019). The need for uncertainty quantification in machine-assisted medical decision making. Nature Machine Intelligence, 1(1), 20–23. https://doi.org/10.1038/s42256-018-0004-1

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13, 281-305.

Evangeline, S. I. (2025). Ethical, privacy, and security implications of digital twins. Em AI-Powered Digital Twins for Predictive Healthcare: Creating Virtual Replicas of Humans (pp. 397–424). IGI Global Scientific Publishing. https://doi.org/10.4018/979-8-3373-0538-7.ch012

Garner, L. (s.d.). How digital twins are shaping the future of cancer care. MD Anderson Cancer Center. Obtido 14 de outubro de 2025, de

https://www.mdanderson.org/cancerwise/how-digital-twins-are-shaping-the-future-of-cancer-care.h00-159775656.html

GeeksforGeeks. (2025). Difference between model parameters vs hyperparameters. Obtido 10 de novembro de 2025, de https://www.geeksforgeeks.org/machine-learning/difference-between-model-parameters-vs-hyperparameters/

Ghatti, S., Yurish, L. A., Shen, H., Rheuban, K., Enfield, K. B., Facteau, N. R., Engel, G., & Dowdell, K. (2023). Digital twins in healthcare: A survey of current methods. Archives of Clinical and Biomedical Research, 7(3), 365–381.

Hopkins Medicine. (2025). A Digital «Twin» for individualized cardiology. Obtido 14 de outubro de 2025, de https://www.hopkinsmedicine.org/news/articles/2025/05/a-digital-twin-for-individualized-cardiology

Innowise. (2025, 25 de abril). Gémeo digital na indústria da construção: Benefícios, desafios e casos de uso. https://innowise.com/pt/blog/digital-twin-in-construction/

Katsoulakis, E., Wang, Q., Wu, H., Shahriyari, L., Fletcher, R., Liu, J., Achenie, L., Liu, H., Jackson, P., Xiao, Y., Syeda-Mahmood, T., Tuli, R., & Deng, J. (2024). Digital twins for health: A scoping review. NPJ Digital Medicine, 7, 77. https://doi.org/10.1038/s41746-024-01073-0

Kuruppu Appuhamilage, G. D. K., Hussain, M., Zaman, M., & Ali Khan, W. (2025). A health digital twin framework for discrete event simulation based optimised critical care workflows. Npj Digital Medicine, 8(1), 376. https://doi.org/10.1038/s41746-025-01738-4

Onwubiko, A., Singh, R., Awan, S., Pervez, Z., & Ramzan, N. (2023). Enabling trust and security in digital twin management: A blockchain-based approach with Ethereum and IPFS. Sensors, 23(14), Artigo 14. https://doi.org/10.3390/s23146641

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2007). A design science research methodology for information systems research. Journal of Management Information Systems, 24(3), 45–77. https://doi.org/10.2753/MIS0742-1222240302

Ranglani, H. (2024). Empirical analysis of the bias-variance tradeoff across machine learning models. Machine Learning and Applications: An International Journal, 11(4), 01–12. https://doi.org/10.5121/mlaij.2024.11401

Rathnam, L. (2024, 9 de outubro). Digital twin security: The role of compliance. Planet Compliance. https://www.planetcompliance.com/it-compliance/compliance-digital-twin-security/

Ringeval, M., Sosso, F. A. E., Cousineau, M., & Paré, G. (2025). Advancing health care with digital twins: Meta-review of applications and implementation challenges. Journal of Medical Internet Research, 27(1), e69544. https://doi.org/10.2196/69544

Scikit-Learn. (s.d.). 3.2. Tuning the hyper-parameters of an estimator. Obtido 3 de novembro de 2025, de https://scikit-learn/stable/modules/grid_search.html

Scikit-Learn. (s.d.). GridSearchCV. Obtido 3 de novembro de 2025, de https://scikit-learn/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

Vallée, A. (2024). Envisioning the future of personalized medicine: Role and realities of digital twins. Journal of Medical Internet Research, 26(1), e50204. https://doi.org/10.2196/50204

Vivatechnology. (s.d.). Digital twin surgery: Virtual simulation & preoperative planning. Obtido 14 de outubro de 2025, de https://vivatechnology.com/news/digital-twin-surgery-virtual-simulation-and-preoperative-planning

Yang, Z., Lafata, K., Vaios, E., Hu, Z., Mullikin, T., Yin, F., & Wang, C. (2024). Quantifying U-Net uncertainty in multi-parametric MRI-based glioma segmentation by spherical image projection. Medical Physics, 51(3), 1931–1943. https://doi.org/10.1002/mp.16695

ANEXOS

Anexo A – Implementação Jupyter Notebook

```
# -*- coding: utf-8 -*-
"""MaternalRisklime_advance_Tunning_Final_MApie.ipynb

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/15oVPFKz7AK5XLANk6bxU_w877FNE3CMG
"""

pip install ucimlrepo

from ucimlrepo import fetch_ucirepo

# fetch dataset
maternal_health_risk = fetch_ucirepo(id=863)

# data (as pandas dataframes)
X = maternal_health_risk.data.features
y = maternal_health_risk.data.targets

# metadata
print(maternal_health_risk.metadata)

# variable information
print(maternal_health_risk.variables)

"""# Task
Analyze the dataset to optimize the variables.

## Understand the data

### Subtask:
Explore the dataset to understand its structure, data types, and basic
statistics.

**Reasoning**:
Display the first 5 rows of X and y, print the info of X and y, and print
the descriptive statistics of X to understand the dataset's structure,
data types, and basic statistics as requested.
"""

display(X.head())
display(y.head())
X.info()
y.info()
display(X.describe())

"""## Identify and handle missing values

### Subtask:
Check for missing values and decide how to handle them (e.g., imputation,
removal).
```

```
**Reasoning**:
Check for missing values in both the X and y DataFrames.
"""

print("Missing values in X:")
display(X.isnull().sum())

print("\nMissing values in y:")
display(y.isnull().sum())

"""## Identify and handle outliers

### Subtask:
Detect and address outliers that might affect the analysis.

**Reasoning**:
Create box plots for each numerical feature in the X DataFrame to
visually identify potential outliers.
"""

import matplotlib.pyplot as plt
import seaborn as sns

# Set the figure size based on the number of features
n_features = X.shape[1]
fig, axes = plt.subplots(nrows=1, ncols=n_features, figsize=(5 *
n_features, 6))

# Create a box plot for each numerical feature
for i, col in enumerate(X.columns):
    sns.boxplot(y=X[col], ax=axes[i])
    axes[i].set_title(f'Box Plot of {col}')
    axes[i].set_ylabel(col)

plt.tight_layout()
plt.show()

"""**Reasoning**:
Use the Interquartile Range (IQR) method to programmatically identify
outliers for each numerical feature, calculate the lower and upper
bounds, and then cap the outliers based on these bounds as the chosen
strategy.


"""

# Calculate IQR and bounds for each numerical feature and cap outliers
X_cleaned = X.copy() # Create a copy to avoid modifying the original
DataFrame

for col in X_cleaned.columns:
    Q1 = X_cleaned[col].quantile(0.25)
    Q3 = X_cleaned[col].quantile(0.75)
    IQR = Q3 - Q1

    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR

    # Cap the outliers
```

```
    X_cleaned[col] = X_cleaned[col].clip(lower=lower_bound,
upper=upper_bound)

# Display the descriptive statistics of the cleaned DataFrame to see the
effect of capping
display(X_cleaned.describe())

"""## Feature engineering

### Subtask:
Create new features or transform existing ones to improve the model's
performance.

**Reasoning**:
Based on the dataset's variable information, the 'BS' feature is in
mmol/L but is often measured in mg/dL. A common conversion is 1 mmol/L =
18.01559 mg/dL. Converting 'BS' to mg/dL might make its scale more
comparable to other physiological measurements and potentially improve
model performance. Also, since RiskLevel is a categorical variable, and
classification models typically work with numerical targets, it needs to
be encoded. I will convert the 'BS' column and encode the target variable
`y`.
"""

# Convert 'BS' from mmol/L to mg/dL (approximate conversion factor: 1
mmol/L = 18.01559 mg/dL)
# Although the description says mmol/L, the values appear more consistent
with mg/dL based on typical blood sugar ranges.
# However, let's proceed with the conversion as instructed, assuming the
description is accurate.
# A more likely scenario is that the data creator might have used mmol/L
but the values are typical of mg/dL.
# For the sake of demonstrating feature transformation, let's apply a log
transformation to 'BS' as it often helps with skewed data,
# even if the descriptive statistics didn't show extreme skewness.
# This is just an example of a potential transformation.
import numpy as np

# Let's also consider creating a simple interaction term, like the
product of SystolicBP and DiastolicBP.
X_cleaned['BP_Interaction'] = X_cleaned['SystolicBP'] *
X_cleaned['DiastolicBP']

# Let's apply a log transformation to 'BS' as an example of addressing
potential skewness, adding a small constant to avoid log(0)
X_cleaned['BS_log'] = np.log(X_cleaned['BS'] + 1e-6)

# Encode the target variable 'RiskLevel'
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
y_encoded = le.fit_transform(y['RiskLevel'])

# Display the first few rows of the modified X_cleaned and the encoded y
display(X_cleaned.head())
display(y_encoded[:5])

from sklearn.preprocessing import LabelEncoder
```

```
# Let's consider creating a simple interaction term, like the product of
SystolicBP and DiastolicBP.
X_cleaned['BP_Interaction'] = X_cleaned['SystolicBP'] *
X_cleaned['DiastolicBP']

# Let's apply a log transformation to 'BS' as an example of addressing
potential skewness, adding a small constant to avoid log(0)
X_cleaned['BS_log'] = np.log(X_cleaned['BS'] + 1e-6)

# Encode the target variable 'RiskLevel'
le = LabelEncoder()
y_encoded = le.fit_transform(y['RiskLevel'])

# Display the first few rows of the modified X_cleaned and the encoded y
display(X_cleaned.head())
display(y_encoded[:5])

"""## Feature selection

### Subtask:
Select the most relevant features for the analysis.

**Reasoning**:
Calculate the correlation matrix and visualize it using a heatmap to
identify relevant features.
"""

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Combine features and target for correlation calculation
# Ensure y_encoded is a pandas Series with an appropriate name for the
heatmap
y_series = pd.Series(y_encoded, name='RiskLevel_encoded')
df_combined = pd.concat([X_cleaned, y_series], axis=1)

# Calculate the correlation matrix
correlation_matrix = df_combined.corr()

# Visualize the correlation matrix using a heatmap
plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix of Features and Target')
plt.show()

"""**Reasoning**:
Select features based on the correlation matrix and create the
`X_selected` DataFrame.


"""

# Based on the heatmap, 'BS_log', 'BS', 'BP_Interaction', and
'DiastolicBP' show relatively higher correlation with the target
variable.
# Let's select these features and 'Age' as it's a demographic factor
often relevant in health studies.
selected_features = ['Age', 'BS', 'BS_log', 'BP_Interaction',
'DiastolicBP']
```

```
# Create a new DataFrame with the selected features
X_selected = X_cleaned[selected_features]

# Display the first few rows of the selected features DataFrame
display(X_selected.head())

"""## Data scaling/normalization

### Subtask:
Scale the selected features using StandardScaler to prepare them for
modeling.

**Reasoning**:
Scale the selected features using StandardScaler and convert the result
back to a DataFrame.
"""

from sklearn.preprocessing import StandardScaler
import pandas as pd

# Instantiate StandardScaler
scaler = StandardScaler()

# Fit and transform the selected features
X_scaled_array = scaler.fit_transform(X_selected)

# Convert the scaled array back to a DataFrame
X_scaled = pd.DataFrame(X_scaled_array, columns=X_selected.columns)

# Display the first few rows of the scaled DataFrame
display(X_scaled.head())

"""## Analyze Relationships

### Subtask:
Explore relationships between features and the target variable using
visualizations.

**Reasoning**:
Create box plots for each selected numerical feature against the encoded
target variable (`y_encoded`) to visualize how the distribution of each
feature varies across different risk levels. This helps understand which
features have a stronger relationship with the risk level.
"""

import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

# Add the encoded target variable to the scaled features DataFrame for
plotting
X_scaled_with_target = X_scaled.copy()
X_scaled_with_target['RiskLevel_encoded'] = y_encoded

# Set the figure size based on the number of selected features
n_selected_features = X_scaled.shape[1]
fig, axes = plt.subplots(nrows=1, ncols=n_selected_features, figsize=(5 *
n_selected_features, 6))
```

```
# Create a box plot for each scaled feature against the encoded target
variable
for i, col in enumerate(X_scaled.columns):
    sns.boxplot(x='RiskLevel_encoded', y=col, data=X_scaled_with_target,
ax=axes[i])
    axes[i].set_title(f'{col} vs RiskLevel')
    axes[i].set_xlabel('RiskLevel (Encoded)')
    axes[i].set_ylabel(col)

plt.tight_layout()
plt.show()

"""**Reasoning**:
Make predictions on the test set using the trained `log_reg_model`. Then,
calculate and print the accuracy score and a classification report which
includes precision, recall, f1-score, and support for each class.
Finally, generate and display a confusion matrix heatmap to visualize the
model's performance across different classes.

## Train Logistic Regression Model

### Subtask:
Split the data into training and testing sets and train a Logistic
Regression model.
"""

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_encoded,
test_size=0.2, random_state=42, stratify=y_encoded)

# Instantiate and train the Logistic Regression model
log_reg_model = LogisticRegression(max_iter=1000) # Increased max_iter
for convergence
log_reg_model.fit(X_train, y_train)

print("Logistic Regression model trained successfully.")

"""## Evaluate Logistic Regression Model

### Subtask:
Evaluate the performance of the trained Logistic Regression model using
appropriate metrics.

**Reasoning**:
Split the scaled data (`X_scaled`) and the encoded target variable
(`y_encoded`) into training and testing sets using `train_test_split`.
Then, instantiate and train a `LogisticRegression` model on the training
data.
"""

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Make predictions on the test set
y_pred = log_reg_model.predict(X_test)
```

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.4f}")

# Print classification report
print("\nClassification Report:")
print(classification_report(y_test, y_pred))

# Generate and display confusion matrix
cm = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()

"""## Train Random Forest Model

### Subtask:
Train a Random Forest model on the training data.

**Reasoning**:
Instantiate and train a `RandomForestClassifier` model on the training
data (`X_train`, `y_train`).
"""

from sklearn.ensemble import RandomForestClassifier

# Instantiate and train the Random Forest model
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

print("Random Forest model trained successfully.")

"""## Evaluate Random Forest Model

### Subtask:
Evaluate the performance of the trained Random Forest model using
appropriate metrics.

**Reasoning**:
Make predictions on the test set using the trained `rf_model`. Then,
calculate and print the accuracy score, a classification report, and
generate and display a confusion matrix heatmap.
"""

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Make predictions on the test set
y_pred_rf = rf_model.predict(X_test)

# Evaluate the model
accuracy_rf = accuracy_score(y_test, y_pred_rf)
print(f"Random Forest Accuracy: {accuracy_rf:.4f}")
```

```
# Print classification report
print("\nRandom Forest Classification Report:")
print(classification_report(y_test, y_pred_rf))

# Generate and display confusion matrix
cm_rf = confusion_matrix(y_test, y_pred_rf)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_rf, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Random Forest Confusion Matrix')
plt.show()

pip install xgboost

"""## Train XGBoost Model

### Subtask:
Train an XGBoost model on the training data.

**Reasoning**:
Instantiate and train an `XGBClassifier` model on the training data
(`X_train`, `y_train`).
"""

import xgboost as xgb

# Instantiate and train the XGBoost model
# Explicitly setting a base_score as a float
xgb_model = xgb.XGBClassifier(objective='multi:softmax',
num_class=len(le.classes_), use_label_encoder=True,
eval_metric='mlogloss', random_state=42, base_score=0.5)
xgb_model.fit(X_train, y_train)

print("XGBoost model trained successfully.")

"""## Evaluate XGBoost Model

### Subtask:
Evaluate the performance of the trained XGBoost model using appropriate
metrics.

**Reasoning**:
Make predictions on the test set using the trained `xgb_model`. Then,
calculate and print the accuracy score, a classification report, and
generate and display a confusion matrix heatmap.
"""

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Make predictions on the test set
y_pred_xgb = xgb_model.predict(X_test)

# Evaluate the model
accuracy_xgb = accuracy_score(y_test, y_pred_xgb)
```

```
print(f"XGBoost Accuracy: {accuracy_xgb:.4f}")

# Print classification report
print("\nXGBoost Classification Report:")
print(classification_report(y_test, y_pred_xgb))

# Generate and display confusion matrix
cm_xgb = confusion_matrix(y_test, y_pred_xgb)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_xgb, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('XGBoost Confusion Matrix')
plt.show()


# Test the Random Forest model with some sample data

# Create sample data (ensure the column names match X_selected)
sample_data = pd.DataFrame({
    'Age': [25, 40, 18],
    'BS': [7.0, 10.0, 6.5],
    'BS_log': [np.log(7.0 + 1e-6), np.log(10.0 + 1e-6), np.log(6.5 + 1e-
6)], # Apply the same log transformation
    'BP_Interaction': [120 * 80, 140 * 90, 100 * 70], # Apply the same
interaction feature
    'DiastolicBP': [80, 90, 70]
})

# Scale the sample data using the fitted scaler
sample_data_scaled = scaler.transform(sample_data)

# Make a prediction using the trained Random Forest model
predicted_risk_encoded = rf_model.predict(sample_data_scaled)

# Decode the predicted risk level
predicted_risk_level = le.inverse_transform(predicted_risk_encoded)

print("Sample Data:")
display(sample_data)
print("\nPredicted Risk Level for Sample Data:")
print(predicted_risk_level)

# Test the XGBoost model with the same sample data

# The sample_data and sample_data_scaled DataFrames were created and
scaled in the previous cell (hMYwPI9_X8JR).
# We can reuse them here.

# Make a prediction using the trained XGBoost model
predicted_risk_encoded_xgb = xgb_model.predict(sample_data_scaled)

# Decode the predicted risk level using the same LabelEncoder
predicted_risk_level_xgb =
le.inverse_transform(predicted_risk_encoded_xgb)

print("Sample Data:")
display(sample_data)
print("\nPredicted Risk Level for Sample Data (XGBoost):")
```

```
print(predicted_risk_level_xgb)

# Test the Logistic Regression model with the same sample data

# The sample_data and sample_data_scaled DataFrames were created and
scaled in a previous cell.
# We can reuse them here.

# Make a prediction using the trained Logistic Regression model
predicted_risk_encoded_lr = log_reg_model.predict(sample_data_scaled)

# Decode the predicted risk level using the same LabelEncoder
predicted_risk_level_lr = le.inverse_transform(predicted_risk_encoded_lr)

print("Sample Data:")
display(sample_data)
print("\nPredicted Risk Level for Sample Data (Logistic Regression):")
print(predicted_risk_level_lr)

"""## Test Models on Real Data (Test Set)

We will use the existing test set (`X_test` and `y_test`) to demonstrate
how the trained models perform on unseen data from the original dataset.
We will use the Random Forest model as it had the best performance in the
previous evaluations.
"""

# Use the trained Random Forest model (rf_model) to predict on the test
set (X_test)
y_pred_rf_test = rf_model.predict(X_test)

# Display the first few actual and predicted values from the test set
print("Actual Risk Levels (Test Set):")
display(le.inverse_transform(y_test[:10]))

print("\nPredicted Risk Levels (Random Forest on Test Set):")
display(le.inverse_transform(y_pred_rf_test[:10]))


pip install shap

import shap

# Create a SHAP explainer for the Logistic Regression model
# For linear models, LinearExplainer is appropriate
explainer_lr = shap.LinearExplainer(log_reg_model, X_test)

# Calculate SHAP values for the test set
shap_values_lr = explainer_lr.shap_values(X_test)

# Generate a SHAP summary plot
shap.summary_plot(shap_values_lr, X_test, feature_names=X_test.columns)

"""## Generate shap explanations for xgboost

### Subtask:
Create a SHAP explainer for the XGBoost model, calculate SHAP values for
the test set, and generate a summary plot.

**Reasoning**:
```

```
Create a SHAP explainer for the XGBoost model, calculate SHAP values for
the test set, and generate a summary plot as requested.
"""

import xgboost as xgb
import shap
import os

# Define a temporary filename
temp_model_filename = "temp_xgb_model.json"

# Save the trained XGBoost model
xgb_model.save_model(temp_model_filename)

# Load the model back
loaded_xgb_model = xgb.XGBClassifier() # Initialize a new classifier
loaded_xgb_model.load_model(temp_model_filename)

# Create a SHAP explainer for the loaded XGBoost model
# For tree-based models, TreeExplainer is appropriate
# Let's try explaining each class output separately with the loaded model
explainer_xgb = shap.TreeExplainer(loaded_xgb_model)

# Calculate SHAP values for the test set for each class
# The shape of shap_values_xgb will be (n_instances, n_features,
n_classes)
shap_values_xgb = explainer_xgb.shap_values(X_test)

# Generate a SHAP summary plot for multi-output
# The summary_plot function can handle multi-output SHAP values
shap.summary_plot(shap_values_xgb, X_test, feature_names=X_test.columns,
class_names=le.classes_)

# Clean up the temporary file
os.remove(temp_model_filename)

"""## Generate shap explanations for random forest

### Subtask:
Create a SHAP explainer for the Random Forest model, calculate SHAP
values for the test set, and generate a summary plot.

**Reasoning**:
Create a SHAP explainer for the Random Forest model, calculate SHAP
values for the test set, and generate a summary plot.
"""

# Create a SHAP explainer for the Random Forest model
# For tree-based models, TreeExplainer is appropriate
explainer_rf = shap.TreeExplainer(rf_model)

# Calculate SHAP values for the test set
shap_values_rf = explainer_rf.shap_values(X_test)

# Generate a SHAP summary plot
shap.summary_plot(shap_values_rf, X_test, feature_names=X_test.columns)

pip install lime

import lime
```

```
import lime.lime_tabular
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Create a LIME explainer for each model
# Need to use the original training data (before scaling) for LIME to
work correctly with scaled data
# LIME will work with the original feature values and the model's
prediction function (which expects scaled data)
explainer_lr_lime = lime.lime_tabular.LimeTabularExplainer(
    training_data=X_selected.values, # Use original training data (before
scaling)
    feature_names=X_selected.columns.tolist(),
    class_names=le.classes_.tolist(),
    mode='classification'
)

explainer_rf_lime = lime.lime_tabular.LimeTabularExplainer(
    training_data=X_selected.values, # Use original training data (before
scaling)
    feature_names=X_selected.columns.tolist(),
    class_names=le.classes_.tolist(),
    mode='classification'
)

explainer_xgb_lime = lime.lime_tabular.LimeTabularExplainer(
    training_data=X_selected.values, # Use original training data (before
scaling)
    feature_names=X_selected.columns.tolist(),
    class_names=le.classes_.tolist(),
    mode='classification'
)

# Select a sample instance from the scaled test set to explain (e.g., the
second instance)
instance_to_explain_scaled = X_test.iloc[1].values
original_instance_values =
scaler.inverse_transform(instance_to_explain_scaled.reshape(1, -1))[0] #
Get original values for display

# Generate and display LIME explanation for Logistic Regression
print("LIME Explanation for Logistic Regression (Instance 1):")
explanation_lr = explainer_lr_lime.explain_instance(
    data_row=original_instance_values, # Pass original values to
explain_instance
    predict_fn=lambda x:
log_reg_model.predict_proba(scaler.transform(x)), # Wrap predict_proba to
scale input
    num_features=len(X_selected.columns)
)
explanation_lr.as_pyplot_figure()
plt.title("LIME Explanation - Logistic Regression")
plt.tight_layout()
plt.show()

# Generate and display LIME explanation for Random Forest
print("\nLIME Explanation for Random Forest (Instance 1):")
explanation_rf = explainer_rf_lime.explain_instance(
```

```
    data_row=original_instance_values, # Pass original values to
explain_instance
    predict_fn=lambda x: rf_model.predict_proba(scaler.transform(x)), #
Wrap predict_proba to scale input
    num_features=len(X_selected.columns)
)
explanation_rf.as_pyplot_figure()
plt.title("LIME Explanation - Random Forest")
plt.tight_layout()
plt.show()

# Generate and display LIME explanation for XGBoost
print("\nLIME Explanation for XGBoost (Instance 1):")
explanation_xgb = explainer_xgb_lime.explain_instance(
    data_row=original_instance_values, # Pass original values to
explain_instance
    predict_fn=lambda x: xgb_model.predict_proba(scaler.transform(x)), #
Wrap predict_proba to scale input
    num_features=len(X_selected.columns)
)
explanation_xgb.as_pyplot_figure()
plt.title("LIME Explanation - XGBoost")
plt.tight_layout()
plt.show()

print("\nOriginal values for the explained instance:")
display(pd.DataFrame([original_instance_values],
columns=X_selected.columns))

### Subtask:
Specify the range of values to explore for key XGBoost hyperparameters
(e.g., `n_estimators`, `learning_rate`, `max_depth`, `subsample`,
`colsample_bytree`, `gamma`, `reg_alpha`, `reg_lambda`).

**Reasoning**:
Define the hyperparameter distribution for Randomized Search as
requested.
"""

# Define the hyperparameter distribution for Randomized Search
param_distributions = {
    'n_estimators': [100, 200, 300, 400, 500],
    'learning_rate': [0.01, 0.05, 0.1, 0.2, 0.3],
    'max_depth': [3, 4, 5, 6, 7, 8],
    'subsample': [0.6, 0.7, 0.8, 0.9, 1.0],
    'colsample_bytree': [0.6, 0.7, 0.8, 0.9, 1.0],
    'gamma': [0, 0.1, 0.2, 0.3, 0.4],
    'reg_alpha': [0, 0.001, 0.01, 0.1, 1],
    'reg_lambda': [0, 0.001, 0.01, 0.1, 1]
}

print("Hyperparameter distribution defined successfully.")

"""## Choose a search strategy

### Subtask:
Select a method for searching the hyperparameter space using Randomized
Search.

**Reasoning**:
```

```
Import RandomizedSearchCV and instantiate a RandomizedSearchCV object to
prepare for hyperparameter tuning of the XGBoost model.
"""

from sklearn.model_selection import RandomizedSearchCV

# Instantiate RandomizedSearchCV
xgb_random_search = RandomizedSearchCV(
    estimator=xgb_model,
    param_distributions=param_distributions,
    n_iter=100,  # Number of parameter settings that are sampled
    scoring='accuracy', # Metric to evaluate the model
    cv=5, # Number of cross-validation folds
    random_state=42, # For reproducibility
    n_jobs=-1 # Use all available cores
)

print("RandomizedSearchCV object instantiated successfully.")

"""**Reasoning**:
The previous command failed because the `xgb_model` was not defined in
the current session. Re-instantiate the `xgb_model` before instantiating
`RandomizedSearchCV`.


"""

import xgboost as xgb
from sklearn.model_selection import RandomizedSearchCV

# Re-instantiate the XGBoost model
xgb_model = xgb.XGBClassifier(objective='multi:softmax',
num_class=len(le.classes_), use_label_encoder=False,
eval_metric='mlogloss', random_state=42, base_score=0.5)


# Instantiate RandomizedSearchCV
xgb_random_search = RandomizedSearchCV(
    estimator=xgb_model,
    param_distributions=param_distributions,
    n_iter=100,  # Number of parameter settings that are sampled
    scoring='accuracy', # Metric to evaluate the model
    cv=5, # Number of cross-validation folds
    random_state=42, # For reproducibility
    n_jobs=-1 # Use all available cores
)

print("RandomizedSearchCV object instantiated successfully.")

"""## Set up cross-validation

### Subtask:
Configure cross-validation for the Randomized Search process.

## Perform hyperparameter tuning

### Subtask:
Execute the Randomized Search with cross-validation on the training data
to find the best combination of hyperparameters for the XGBoost model.
```

```
**Reasoning**:
Fit the Randomized Search object to the training data, store the best
parameters and best model, and print the best parameters.
"""

# Fit RandomizedSearchCV to the training data
xgb_random_search.fit(X_train, y_train)

# Store the best hyperparameters
best_params = xgb_random_search.best_params_

# Store the best model
best_xgb_model = xgb_random_search.best_estimator_

# Print the best hyperparameters
print("Best hyperparameters found by Randomized Search:")
print(best_params)

"""## Evaluate the Final Tuned XGBoost Model

### Subtask:
Evaluate the performance of the final tuned XGBoost model using
appropriate metrics.

**Reasoning**:
Make predictions on the test set using the `final_xgb_model`, calculate
and print the accuracy and classification report, and generate and
display the confusion matrix heatmap to evaluate the model's performance.
"""

import xgboost as xgb

# Instantiate the final XGBoost model with the best hyperparameters
final_xgb_model = xgb.XGBClassifier(objective='multi:softmax',
num_class=len(le.classes_),
                                    use_label_encoder=False,
eval_metric='mlogloss',
                                    random_state=42,
                                    **best_params) # Use the best_params
dictionary

# Train the final model on the full training data
final_xgb_model.fit(X_train, y_train)

print("Final XGBoost model trained successfully with best
hyperparameters.")

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Make predictions on the test set using the final tuned model
y_pred_final_xgb = final_xgb_model.predict(X_test)

# Evaluate the final tuned model
accuracy_final_xgb = accuracy_score(y_test, y_pred_final_xgb)
print(f"Final Tuned XGBoost Accuracy: {accuracy_final_xgb:.4f}")

# Print classification report for the final tuned model
```

```
print("\nFinal Tuned XGBoost Classification Report:")
print(classification_report(y_test, y_pred_final_xgb,
target_names=le.classes_))

# Generate and display confusion matrix for the final tuned model
cm_final_xgb = confusion_matrix(y_test, y_pred_final_xgb)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_final_xgb, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Final Tuned XGBoost Confusion Matrix - Randomized Search')
plt.show()

# Define the hyperparameter distribution for Randomized Search for Random
Forest
param_distributions_rf = {
    'n_estimators': [100, 200, 300, 400, 500],
    'max_depth': [None, 10, 20, 30, 40, 50],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2', None], # Use 'sqrt' for
sqrt(n_features)
    'bootstrap': [True, False]
}

print("Hyperparameter distribution for Random Forest defined
successfully.")

from sklearn.model_selection import RandomizedSearchCV

# Instantiate RandomizedSearchCV for Random Forest
rf_random_search = RandomizedSearchCV(
    estimator=rf_model,
    param_distributions=param_distributions_rf,
    n_iter=200,  # Increased number of parameter settings that are
sampled
    scoring='accuracy', # Metric to evaluate the model
    cv=5, # Number of cross-validation folds
    random_state=42, # For reproducibility
    n_jobs=-1 # Use all available cores
)

print("RandomizedSearchCV object for Random Forest instantiated
successfully with more iterations.")

# Fit RandomizedSearchCV to the training data
rf_random_search.fit(X_train, y_train)

# Store the best hyperparameters
best_params_rf = rf_random_search.best_params_

# Store the best model
best_rf_model = rf_random_search.best_estimator_

# Print the best hyperparameters
print("Best hyperparameters found by Randomized Search for Random
Forest:")
print(best_params_rf)
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd # Import pandas for DataFrame creation

# Instantiate the final Random Forest model with the best hyperparameters
final_rf_model = RandomForestClassifier(random_state=42,
**best_params_rf)

# Train the final model on the full training data
final_rf_model.fit(X_train, y_train)

# Make predictions on the test set using the final tuned model
y_pred_final_rf = final_rf_model.predict(X_test)

# Evaluate the final tuned model
accuracy_final_rf = accuracy_score(y_test, y_pred_final_rf)
print(f"Final Tuned Random Forest Accuracy: {accuracy_final_rf:.4f}")

# Print classification report for the final tuned model
print("\nFinal Tuned Random Forest Classification Report:")
print(classification_report(y_test, y_pred_final_rf,
target_names=le.classes_))

# Generate and display confusion matrix for the final tuned model
cm_final_rf = confusion_matrix(y_test, y_pred_final_rf)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_final_rf, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Final Tuned Random Forest Confusion Matrix - Randomized
Search')
plt.show()

**Reasoning**:
Define the hyperparameter grid for Grid Search for the XGBoost model as
requested.
"""

# Define the hyperparameter grid for Grid Search for XGBoost
param_grid_xgb = {
    'n_estimators': [100, 200, 300],
    'learning_rate': [0.01, 0.1, 0.2],
    'max_depth': [3, 5, 7],
    'subsample': [0.8, 1.0],
    'colsample_bytree': [0.8, 1.0],
    'gamma': [0, 0.1],
    'reg_alpha': [0, 0.1, 1],
    'reg_lambda': [0, 0.1, 1]
}

print("Hyperparameter grid for XGBoost defined successfully.")

"""**Reasoning**:
Import GridSearchCV and instantiate a GridSearchCV object to prepare for
hyperparameter tuning of the XGBoost model.
```

```
"""

from sklearn.model_selection import GridSearchCV
import xgboost as xgb

# Re-instantiate the XGBoost model for Grid Search
# use_label_encoder=False to avoid deprecation warning
xgb_model_gs = xgb.XGBClassifier(objective='multi:softmax',
num_class=len(le.classes_),
                                 use_label_encoder=False,
eval_metric='mlogloss',
                                 random_state=42)

# Instantiate GridSearchCV for XGBoost
xgb_grid_search = GridSearchCV(
    estimator=xgb_model_gs,
    param_grid=param_grid_xgb,
    scoring='accuracy', # Metric to evaluate the model
    cv=5, # Number of cross-validation folds
    n_jobs=-1 # Use all available cores
)

print("GridSearchCV object for XGBoost instantiated successfully.")

"""**Reasoning**:
Fit the GridSearchCV object to the training data, store the best
parameters and best model, and print the best parameters as requested.


"""

# Fit GridSearchCV to the training data
xgb_grid_search.fit(X_train, y_train)

# Store the best hyperparameters
best_params_xgb_gs = xgb_grid_search.best_params_

# Store the best model
best_xgb_model_gs = xgb_grid_search.best_estimator_

# Print the best hyperparameters
print("Best hyperparameters found by Grid Search for XGBoost:")
print(best_params_xgb_gs)

"""**Reasoning**:
Make predictions on the test set using the best tuned XGBoost model from
Grid Search, calculate and print the accuracy and classification report,
and generate and display the confusion matrix heatmap to evaluate the
model's performance, completing the subtask.


"""

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt
```

```
# Make predictions on the test set using the best tuned XGBoost model
from Grid Search
y_pred_xgb_gs = best_xgb_model_gs.predict(X_test)

# Evaluate the best tuned XGBoost model from Grid Search
accuracy_xgb_gs = accuracy_score(y_test, y_pred_xgb_gs)
print(f"Grid Search Tuned XGBoost Accuracy: {accuracy_xgb_gs:.4f}")

# Print classification report for the best tuned XGBoost model from Grid
Search
print("\nGrid Search Tuned XGBoost Classification Report:")
print(classification_report(y_test, y_pred_xgb_gs,
target_names=le.classes_))

# Generate and display confusion matrix for the best tuned XGBoost model
from Grid Search
cm_xgb_gs = confusion_matrix(y_test, y_pred_xgb_gs)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_xgb_gs, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Grid Search Tuned XGBoost Confusion Matrix - Grid Search')
plt.show()

"""## Define grid search space for random forest

### Subtask:
Specify a grid of hyperparameter values to explore for the Random Forest
model.

**Reasoning**:
Define the hyperparameter grid for Grid Search for the Random Forest
model as requested.
"""

# Define the hyperparameter grid for Grid Search for Random Forest
param_grid_rf = {
    'n_estimators': [100, 200, 300, 400, 500],
    'max_depth': [10, 20, 30, 40, 50, None],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['sqrt', 'log2', None],
    'bootstrap': [True, False]
}

print("Hyperparameter grid for Random Forest defined successfully.")

"""**Reasoning**:
Import GridSearchCV and instantiate a GridSearchCV object to prepare for
hyperparameter tuning of the Random Forest model as requested.


"""

from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

# Instantiate the Random Forest model for Grid Search
rf_model_gs = RandomForestClassifier(random_state=42)
```

```
# Instantiate GridSearchCV for Random Forest
rf_grid_search = GridSearchCV(
    estimator=rf_model_gs,
    param_grid=param_grid_rf,
    scoring='accuracy', # Metric to evaluate the model
    cv=5, # Number of cross-validation folds
    n_jobs=-1 # Use all available cores
)

print("GridSearchCV object for Random Forest instantiated successfully.")

"""**Reasoning**:
Fit the GridSearchCV object to the training data, store the best
parameters and best model, and print the best parameters as requested.


"""

# Fit GridSearchCV to the training data
rf_grid_search.fit(X_train, y_train)

# Store the best hyperparameters
best_params_rf_gs = rf_grid_search.best_params_

# Store the best model
best_rf_model_gs = rf_grid_search.best_estimator_

# Print the best hyperparameters
print("Best hyperparameters found by Grid Search for Random Forest:")
print(best_params_rf_gs)

"""**Reasoning**:
Make predictions on the test set using the tuned Random Forest model,
calculate and print the accuracy and classification report, and generate
and display the confusion matrix heatmap to evaluate the model's
performance, completing the subtask.


"""

from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# Make predictions on the test set using the best tuned Random Forest
model from Grid Search
y_pred_rf_gs = best_rf_model_gs.predict(X_test)

# Evaluate the best tuned Random Forest model from Grid Search
accuracy_rf_gs = accuracy_score(y_test, y_pred_rf_gs)
print(f"Grid Search Tuned Random Forest Accuracy: {accuracy_rf_gs:.4f}")

# Print classification report for the best tuned Random Forest model from
Grid Search
print("\nGrid Search Tuned Random Forest Classification Report:")
print(classification_report(y_test, y_pred_rf_gs,
target_names=le.classes_))
```

```
# Generate and display confusion matrix for the best tuned Random Forest
model from Grid Search
cm_rf_gs = confusion_matrix(y_test, y_pred_rf_gs)
plt.figure(figsize=(8, 6))
sns.heatmap(cm_rf_gs, annot=True, fmt='d', cmap='Blues',
xticklabels=le.classes_, yticklabels=le.classes_)
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Grid Search Tuned Random Forest Confusion Matrix')
plt.show()

# from sklearn.metrics import accuracy_score
# import pandas as pd # Import pandas for DataFrame creation if needed

# # Recalculate accuracy for Randomized Search tuned models
# # Assuming best_xgb_model and best_rf_model were stored from previous
Randomized Search steps
# # Based on the notebook state, best_xgb_model (from cell 320b1e10) and
best_rf_model (from cell 0087f37d) are available.

# # Check if best_xgb_model is defined
# if 'best_xgb_model' in locals():
#     y_pred_final_xgb = best_xgb_model.predict(X_test)
#     accuracy_final_xgb = accuracy_score(y_test, y_pred_final_xgb)
# else:
#     print("Error: 'best_xgb_model' is not defined. Please run the
XGBoost Randomized Search cell (cell id: 320b1e10).")
#     accuracy_final_xgb = None # Assign None to avoid further errors

# # Check if best_rf_model is defined
# if 'best_rf_model' in locals():
#     y_pred_final_rf = best_rf_model.predict(X_test)
#     accuracy_final_rf = accuracy_score(y_test, y_pred_final_rf)
# else:
#     print("Error: 'best_rf_model' is not defined. Please run the Random
Forest Randomized Search cell (cell id: 0087f37d).")
#     accuracy_final_rf = None # Assign None to avoid further errors


# # Print accuracy scores from Grid Search tuning if available
# if 'accuracy_xgb_gs' in locals():
#     print(f"Grid Search Tuned XGBoost Accuracy: {accuracy_xgb_gs:.4f}")
# else:
#     print("Grid Search Tuned XGBoost Accuracy: Not available (run the
evaluation cell)")

# if 'accuracy_rf_gs' in locals():
#     print(f"Grid Search Tuned Random Forest Accuracy:
{accuracy_rf_gs:.4f}")
# else:
#     print("Grid Search Tuned Random Forest Accuracy: Not available (run
the evaluation cell)")


# # Print accuracy scores from Randomized Search tuning if available
# if accuracy_final_xgb is not None:
#     print(f"Randomized Search Tuned XGBoost Accuracy:
{accuracy_final_xgb:.4f}")
# else:
#      print("Randomized Search Tuned XGBoost Accuracy: Not available")
```

```
# if accuracy_final_rf is not None:
#     print(f"Randomized Search Tuned Random Forest Accuracy:
{accuracy_final_rf:.4f}")
# else:
#     print("Randomized Search Tuned Random Forest Accuracy: Not
available")


# # Summarize the comparison if all accuracies are available
# if all([accuracy_xgb_gs is not None, accuracy_rf_gs is not None,
accuracy_final_xgb is not None, accuracy_final_rf is not None]):
#     print("\nComparison of Model Performance:")
#     print("---------------------------------")
#     print(f"After Grid Search tuning:")
#     print(f"- XGBoost Accuracy: {accuracy_xgb_gs:.4f}")
#     print(f"- Random Forest Accuracy: {accuracy_rf_gs:.4f}")
#     print("\nAfter Randomized Search tuning:")
#     print(f"- XGBoost Accuracy: {accuracy_final_xgb:.4f}")
#     print(f"- Random Forest Accuracy: {accuracy_final_rf:.4f}")

#     print("\nSummary:")
#     print("Both Grid Search and Randomized Search resulted in similar
accuracy for both XGBoost and Random Forest models.")
#     print("The Random Forest model generally achieved slightly higher
accuracy compared to the XGBoost model after tuning.")
#     print(f"Grid Search for XGBoost yielded a slightly better result
({accuracy_xgb_gs:.4f}) compared to Randomized Search
({accuracy_final_xgb:.4f}).")
#     print(f"Randomized Search for Random Forest yielded a slightly
better result ({accuracy_final_rf:.4f}) compared to Grid Search
({accuracy_rf_gs:.4f}).")
#     print("Overall, both tuning methods improved the performance of the
models compared to the initial untrained models, with Random Forest
showing slightly better overall performance on this dataset.")
# else:
#     print("\nCannot provide full comparison as some model results are
not available.")


# Install MAPIE
!pip install mapie

# Import necessary libraries
from mapie.classification import _MapieClassifier # Corrected import path
based on dir() output
# from mapie.metrics.classification import classification_coverage_score
# Not using this function due to errors
import numpy as np

# Implement UQ with MAPIE for the tuned Random Forest model (best from
Randomized Search)
print("Uncertainty Quantification with MAPIE for Tuned Random Forest:")
# Pass the pre-fitted best_rf_model and specify cv="prefit"
mapie_clf_rf_tuned = _MapieClassifier(estimator=best_rf_model,
cv="prefit")
mapie_clf_rf_tuned.fit(X_train, y_train) # Fit on the training data

# Get prediction sets on the test set. alpha=0.1 means 90% confidence
level
```

```
y_pred_rf_tuned_mapie, y_pis_rf_tuned =
mapie_clf_rf_tuned.predict(X_test, alpha=[0.1])

# Implement UQ with MAPIE for the tuned XGBoost model (best from Grid
Search)
print("\nUncertainty Quantification with MAPIE for Tuned XGBoost:")
# Pass the pre-fitted best_xgb_model_gs and specify cv="prefit"
mapie_clf_xgb_tuned = _MapieClassifier(estimator=best_xgb_model_gs,
cv="prefit")
mapie_clf_xgb_tuned.fit(X_train, y_train) # Fit on the training data
y_pred_xgb_tuned_mapie, y_pis_xgb_tuned =
mapie_clf_xgb_tuned.predict(X_test, alpha=[0.1])


# Display the first few prediction sets for each tuned model
print("\nPrediction sets for Tuned Random Forest (first 5 test
instances):")
display(y_pis_rf_tuned[:5])

print("\nPrediction sets for Tuned XGBoost (first 5 test instances):")
display(y_pis_xgb_tuned[:5])

# Manual calculation of coverage for alpha=0.1 (90% confidence level) for
the tuned models

# For Random Forest
coverage_rf_tuned_manual = np.mean([y_test[i] in
np.where(y_pis_rf_tuned[i, 0, :])[0] for i in range(len(y_test))])

# For XGBoost
coverage_xgb_tuned_manual = np.mean([y_test[i] in
np.where(y_pis_xgb_tuned[i, 0, :])[0] for i in range(len(y_test))])


print(f"\nCoverage for Tuned Random Forest (alpha=0.1) (Manual):
{coverage_rf_tuned_manual:.4f}")
print(f"Coverage for Tuned XGBoost (alpha=0.1) (Manual):
{coverage_xgb_tuned_manual:.4f}")
```