Trabalho Prático I - INF01017

Modelos Interpretáveis para Previsão de Gravidade de Casos de Dengue

Gabriel Carvalho Tavares
Maximus Borges da Rosa

Profa. Dra. Mariana Recamonde Mendoza

Universidade Federal do Rio Grande do Sul

18 de novembro de 2025

Resumo

Este relatório apresenta o desenvolvimento de um modelo de aprendizado de máquina para classificação da gravidade de casos de dengue. O trabalho abrange desde a definição do problema até a avaliação de diferentes algoritmos, incluindo análise exploratória dos dados, pré-processamento, e comparação de desempenho entre modelos.

Sumário

1 Definição do Problema e Coleta de Dados

1.1 Contextualização

A dengue faz parte de um grupo de doenças denominadas arboviroses, que se caracterizam por serem causadas por vírus transmitidos por vetores artrópodes. No Brasil, o vetor da dengue é a fêmea do mosquito Aedes aegypti (significa "odioso do Egito"). Os vírus dengue (DENV) estão classificados cientificamente na família Flaviviridae e no gênero Orthoflavivirus. Até o momento são conhecidos quatro sorotipos – DENV-1, DENV-2, DENV-3 e DENV-4 –, que apresentam distintos materiais genéticos (genótipos) e linhagens.

Aspectos como a urbanização, o crescimento desordenado da população, o saneamento básico deficitário e os fatores climáticos mantêm as condições favoráveis para a presença do vetor, com reflexos na dinâmica de transmissão desses arbovírus. A dengue possui padrão sazonal, com aumento do número de casos e o risco para epidemias, principalmente entre os meses de outubro de um ano a maio do ano seguinte.

Todas as faixas etárias são igualmente suscetíveis à doença, porém as pessoas mais velhas e aquelas que possuem doenças crônicas, como diabetes e hipertensão arterial, têm maior risco de evoluir para casos graves e outras complicações que podem levar à morte.

Em 2014, o Brasil começou a adotar a nova classificação de casos de dengue da Organização Mundial da Saúde (OMS), sendo estes atualmente classificados como dengue, dengue com sinais de alarme e dengue grave.

1.2 Definição do Problema

Nosso trabalho tem como objetivo desenvolver um modelo baseado em AM para prever a gravidade de um caso de dengue - segundo a classificação da OMS - a partir de informações gerais do paciente, dados de residência, características da notificação individual, dados clínicos e dados laboratoriais de pacientes.

1.3 Fonte dos Dados

Para tanto, utilizamos o conjunto de dados do Sistema de Informação de Agravos de Notificação (SINAN), disponibilizado pelo Ministério da Saúde.

1.3.1 Processo de Coleta e Filtragem dos Dados

O dataset foi construído através de uma estratégia de filtragem temporal para balancear melhor a representação das classes:

- **Classe low_risk**: dados coletados exclusivamente do ano de 2021
- **Classes alarm e severe**: dados coletados de todo o período disponível, cuja classificação segue o padrão da OMS (01/01/2014 a 11/08/2025)

Esta abordagem foi adotada devido ao volume significativamente maior de casos de baixo risco em comparação com as classes minoritárias.

O ano de 2021 foi escolhido por ser recente e conter uma quantidade substancial de registros, garantindo uma amostra representativa da classe low_risk.

1.4 Caracterização do Dataset

O conjunto de dados é composto por **716.917 registros** de casos de dengue notificados no Brasil. O dataset está estruturado para uma tarefa de **aprendizado supervisionado**, especificamente para um problema de **classificação multiclasse**.

O objetivo é construir um modelo preditivo capaz de classificar a gravidade de um caso de dengue (`severity`) com base em um conjunto de atributos demográficos, comorbidades e sintomas apresentados pelo paciente.

O atributo alvo (`severity`) define a gravidade do caso de dengue. Ele está dividido em três classes, cujas distribuições são mostradas na Tabela 2.

A principal característica deste atributo é o **forte desbalanceamento de classes**. A classe `severe`, que representa os casos de maior criticidade e é, presumivelmente, a mais importante de se prever corretamente, é significativamente minoritária. Este desbalanceamento é o maior desafio do dataset, e torna indispensável o uso de técnicas de *resampling*.

O dataset contém 42 atributos preditores (*features*) que fornecem o contexto para o modelo. Dentre estes atributos há dados nominais (codificados como one-hot), binários e numéricos contínuos. A Tabela 1 descreve estes atributos.

Nome do Atributo	**Codificação**	**Dimensão**
idade_paciente	real	-
dias_sintomas_notificacao	real	-
possui_doenca_autoimune	binary	-
possui_diabetes	binary	-
possui_doencas_hematologicas	binary	-
possui_hepatopatias	binary	-
possui_doenca_renal	binary	-
possui_hipertensao	binary	-
possui_doenca_acido_peptica	binary	-
apresenta_febre	binary	-
apresenta_cefaleia	binary	-
apresenta_exantema	binary	-
apresenta_dor_costas	binary	-
apresenta_mialgia	binary	-
apresenta_vomito	binary	-
apresenta_conjutivite	binary	-
apresenta_dor_retroorbital	binary	-
apresenta_artralgia	binary	-
apresenta_artrite	binary	-
apresenta_leucopenia	binary	-
apresenta_petequias	binary	-
prova_laco	binary	-
sigla_uf_residencia	one-hot	5
sexo_paciente	one-hot	2
raca_cor_paciente	one-hot	6
gestante_paciente	one-hot	7

Tabela 1: Caracterização dos atributos do dataset.

Classe	Número de Exemplos
low_risk	420.850
alarm	267.544
severe	28.523
Total	**716.917**

Tabela 2: Distribuição de exemplos por classe no dataset.

2 Análise Exploratória

A Figura 1 mostra a correlação, indicada pela medida de associação V de Cramer, entre cada par de features do dataset. Visto que a maioria dos atributos dos dados são categóricos, convertemos os atributos originalmente numéricos para categóricos (utilizando intervalos de valores) para possibilitar o uso da métrica V de Cramer.

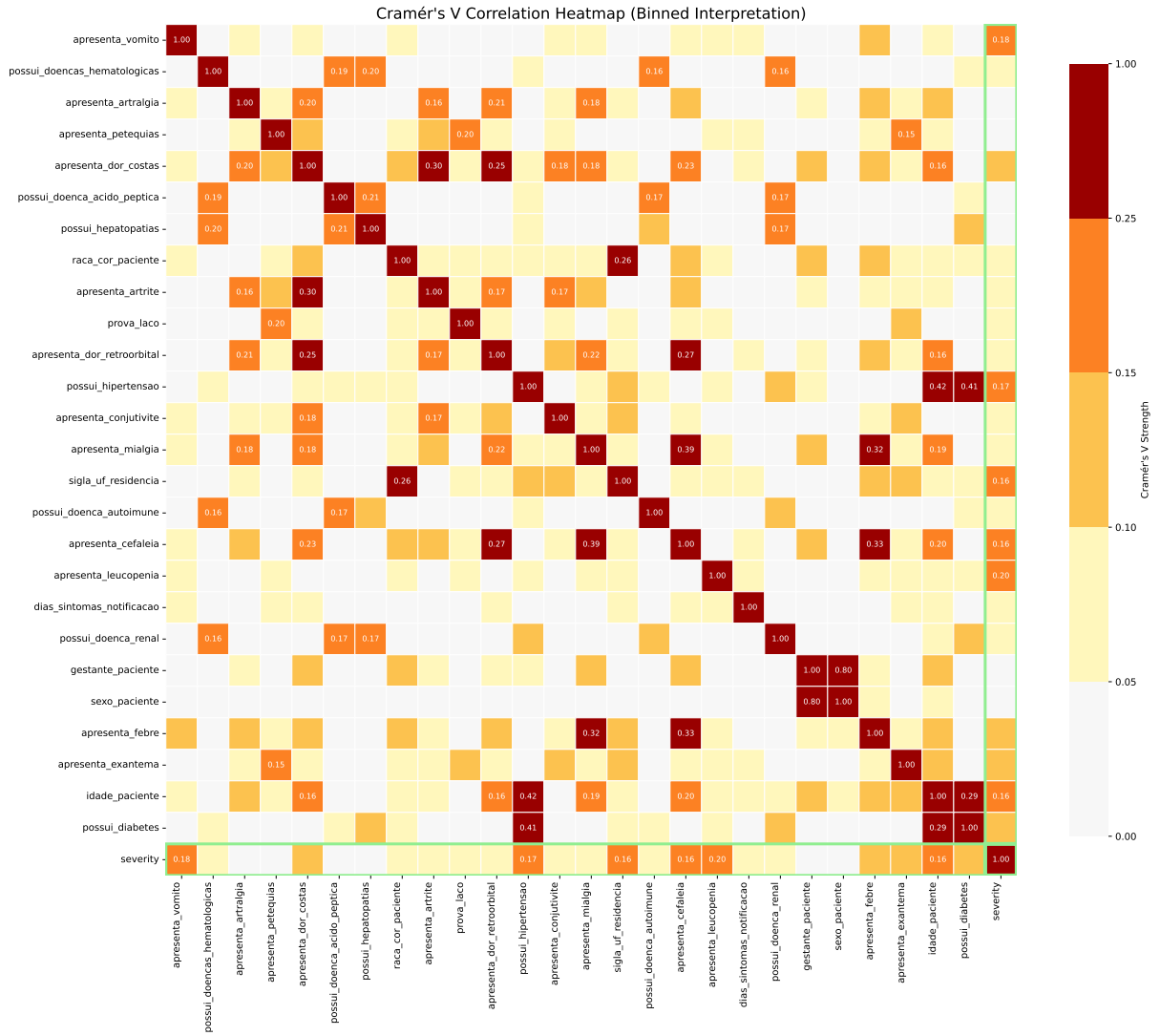

Figura 1: Matriz de correlação entre features do dataset.

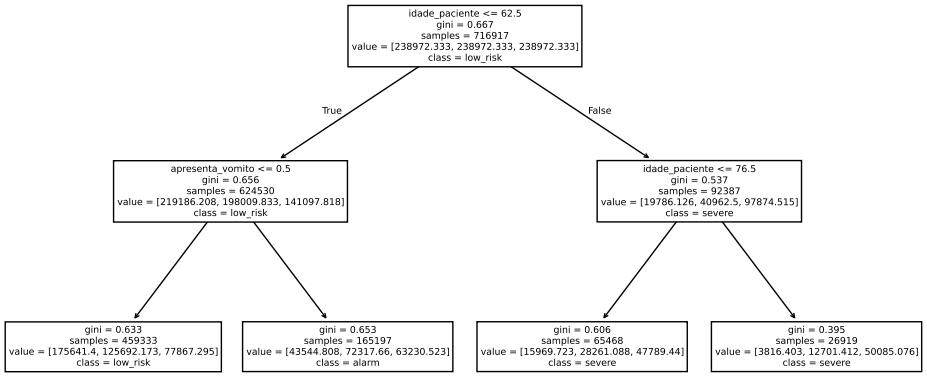

Figura 2: Exemplo gráfico de uma árvore de decisão.

É interessante notar que existe uma variável com forte correlação com a variável alvo (`severity`), mensurada pelo V de Cramer Esse achado aponta `apresenta_leucopenia` como uma variável-chave para prever a severidade dos casos.

3 Pré-processamento

3.1 Tratamento de Valores Ausentes

O conjunto de *features* selecionado, composto por dados demográficos e clínicos básicos, apresentou uma baixa incidência de valores faltantes. Adotamos uma estratégia de tratamento segmentada, dependendo do atributo com dado ausente:

- **Remoção de Instâncias:** Para atributos essenciais à classificação, onde a imputação de valores artificiais não é confiável (como, por exemplo, `idade_paciente`), optou-se pela remoção completa das instâncias (linhas) com valores faltantes.
- **Imputação por Valor Negativo (Moda):** Nos casos de atributos binários referentes a comorbidades ou sintomas, os valores ausentes foram imputados com **0** (negativo). Esta abordagem se justifica por ser o valor mais frequente (a moda) para todas essas *features*.
- **Imputação por Categoria Específica:** Para atributos categóricos que já possuem um valor específico para "não informado"em seu domínio (como `raca_cor_paciente` e `gestante_paciente`), este valor foi atribuído aos campos faltantes.

3.2 Codificação, Normalização e Padronização de Variáveis

Foi utilizada a codificação *One-Hot Encoding* para variáveis categóricas que não poderiam ser interpretadas como binárias. Variáveis referentes à comorbidades e sintomas do paciente, cujos valores correspondem à respostas "sim"e "não"foram codificadas como binárias.

As únicas variáveis efetivamente numéricas do nosso dataset são as variáveis como `idade_paciente` e `dias_sintomas_notificacao`, e estas foram representadas como números reais. Removemos do conjunto de dados todos os exemplos contendo valores negativos para estes atributos e, no caso do atributo `idade_paciente`, removemos também os exemplos com valor maior que 120 neste atributo. As variáveis numéricas foram normalizadas utilizando *Standard Scaler*, com estatísticas computadas exclusivamente do conjunto de treinamento.

O atributo alvo (`severity`) for codificado com *Label Encoding*.

3.3 Tratamento de Desbalanceamento

O desbalanceamento de classes é um dos principais desafios deste dataset, conforme evidenciado na Tabela 2.

Este desbalanceamento severo pode levar os algoritmos de aprendizado a apresentarem *bias* em favor das classes majoritárias, resultando em baixo desempenho na predição de casos graves. Para mitigar este problema, adotamos uma abordagem combinada que integra três técnicas complementares:

3.3.1 Random Under Sampler

Para reduzir a dominância da classe majoritária (`low_risk`), aplicamos subamostragem aleatória (*Random Under Sampling*), reduzindo o número de exemplos desta classe para igualar à quantidade de exemplos da classe intermediária (`alarm`). Esta técnica tem como objetivo:

- Reduzir o viés do modelo em favor da classe majoritária
- Diminuir o custo computacional do treinamento
- Balancear a influência das classes no processo de aprendizado

3.3.2 SMOTE (Synthetic Minority Over-sampling Technique)

Para a classe minoritária (`severe`), utilizamos SMOTE, uma técnica de sobreamostragem sintética que gera novos exemplos através de interpolação entre exemplos existentes no espaço de características. O SMOTE foi configurado para aumentar o número de exemplos da classe `severe` até igualar às demais classes.

As principais vantagens do SMOTE sobre a simples duplicação de exemplos incluem:

- Redução do risco de *overfitting*, pois não replica exemplos idênticos
- Expansão das regiões de decisão da classe minoritária
- Melhoria na capacidade de generalização do modelo

3.3.3 Class Weight Balancing

Complementarmente, utilizamos o parâmetro `class_weight='balanced'` nos algoritmos que suportam esta configuração. Este parâmetro ajusta automaticamente os pesos das classes inversamente proporcionais às suas frequências, penalizando mais os erros nas classes minoritárias durante o treinamento.

3.3.4 Pipeline de Amostragem

O tratamento foi implementado através de um pipeline sequencial que aplica primeiro a subamostragem e depois a sobreamostragem sintética. A Tabela 3 mostra a distribuição das classes antes e após a aplicação das técnicas de amostragem no conjunto de treinamento.

Classe	Original	Após Amostragem
low_risk	294.594	187.280
alarm	187.280	187.280
severe	19.967	187.280
Total	**501.841**	**561.840**

Tabela 3: Distribuição de classes antes e após amostragem com SMOTE e Random Under Sampler no conjunto de treinamento.

Após a aplicação da estratégia combinada, todas as três classes ficaram perfeitamente balanceadas com 187.280 exemplos cada. É importante ressaltar que estas técnicas foram aplicadas **exclusivamente no conjunto de treinamento**, mantendo os conjuntos de validação e teste com suas distribuições originais para garantir uma avaliação realista do desempenho do modelo em dados do mundo real.

4 Definição da Abordagem, Algoritmos e Estratégia de Avaliação

4.1 Algoritmos Selecionados

Foi escolhido um conjunto diversificado de algoritmos que representam diferentes paradigmas do aprendizado de máquina. O objetivo é comparar modelos lineares, probabilísticos, baseados em árvores e métodos de *ensemble*, permitindo analisar o equilíbrio entre interpretabilidade - crucial para em um trabalho relacionado à área da saúde - e desempenho preditivo.

5 Definição da Abordagem, Algoritmos e Estratégia de Avaliação

5.1 Algoritmos Selecionados

Foi escolhido um conjunto diversificado de algoritmos que representam diferentes paradigmas do aprendizado de máquina. O objetivo é comparar modelos lineares, probabilísticos, baseados em árvores e métodos de *ensemble*, permitindo analisar o equilíbrio entre interpretabilidade - crucial para em um trabalho relacionado à área da saúde - e desempenho preditivo.

5.1.1 Árvore de Decisão

As Árvores de Decisão são modelos não paramétricos que aprendem regras de decisão hierárquicas extraídas dos dados.

- **Características:** São modelos de "caixa-branca", altamente interpretáveis e capazes de lidar nativamente com dados numéricos e categóricos.
- **Vantagens e desvantagens:** Destacam-se pela interpretabilidade, mas tendem ao *overfitting* e podem ser instáveis, pois pequenas mudanças nos dados podem gerar árvores significativamente diferentes.
- **Hiperparâmetros principais:** profundidade máxima da árvore (`max_depth`) e número mínimo de amostras para divisão de um nó (`min_samples_split`), que controlam a complexidade do modelo.

5.1.2 Random Forest

O Random Forest é um método de *ensemble* baseado na construção de múltiplas árvores de decisão.

- **Ensemble Learning:** Utiliza *bagging*, onde cada árvore é treinada em uma subamostra aleatória dos dados. A predição final resulta da votação entre as árvores.
- **Redução de variância:** A combinação das árvores reduz a variância e mitiga o *overfitting* típico de árvores individuais, aumentando a robustez do modelo.
- **Hiperparâmetros principais:** número de árvores (`n_estimators`) e profundidade máxima (`max_depth`).

5.1.3 Gradient Boosting

O Gradient Boosting também é um método de *ensemble*, mas baseado em *boosting*.

- **Boosting vs. Bagging:** Ao contrário do *bagging*, o *boosting* constrói árvores de forma sequencial, onde cada nova árvore busca corrigir os erros da anterior, focando nas amostras mais difíceis.
- **Implementações:** Foi utilizada a biblioteca XGBoost, reconhecida pelo alto desempenho e eficiência.
- **Hiperparâmetros principais:** número de estimadores (`n_estimators`), taxa de aprendizado (`learning_rate`) e profundidade das árvores.

5.1.4 Naive Bayes

O Naive Bayes é um classificador probabilístico baseado no Teorema de Bayes, assumindo independência condicional entre os atributos.

- **Características:** Treinamento e predição extremamente rápidos. Lida bem com conjuntos de dados de alta dimensionalidade e é adequado como *baseline*.
- **Vantagens e desvantagens:** Embora simples e eficiente, a suposição de independência raramente se verifica na prática, podendo limitar seu desempenho preditivo.

5.1.5 Regressão Logística Multinomial

A Regressão Logística é um modelo linear generalizado usado para classificação, cuja versão multinomial atende problemas com mais de duas classes.

- **Características:** Modelo interpretável, em que os coeficientes indicam a influência de cada atributo sobre a probabilidade de cada classe.
- **Vantagens e desvantagens:** É eficiente e fornece uma boa *baseline* linear. Por outro lado, não captura relações complexas ou não lineares entre os atributos.

5.1.6 MLP (Multi-layer Perceptron)

O Multi-layer Perceptron (MLP) é uma rede neural artificial feedforward composta por camadas totalmente conectadas. É um modelo capaz de aprender relações não lineares complexas a partir dos dados.

- **Características:** Utiliza uma ou mais camadas ocultas com funções de ativação não lineares (como ReLU ou Tanh), permitindo aprender representações internas mais expressivas do que modelos lineares. Requer normalização dos atributos de entrada para um bom desempenho.
- **Vantagens e desvantagens:** Possui alta capacidade de modelar relações não lineares e interações entre atributos. Entretanto, é menos interpretável do que modelos lineares e métodos baseados em árvores, e pode exigir maior tempo de treinamento, ser sensível à escolha dos hiperparâmetros e propenso a *overfitting* em bases pequenas.
- **Hiperparâmetros principais:** número e tamanho das camadas ocultas (`hidden_layer_sizes`), taxa de aprendizado, função de ativação, regularização (`alpha`) e número máximo de iterações de treinamento.

5.2 Estratégia de Avaliação

5.2.1 Particionamento dos Dados

Para avaliar adequadamente o desempenho dos modelos, o dataset foi dividido em três conjuntos independentes seguindo uma estratégia de *holdout* estratificada:

- **Conjunto de Teste (15%):** Separado inicialmente e mantido isolado durante todo o processo de desenvolvimento. Este conjunto é utilizado exclusivamente para a avaliação final do modelo otimizado, garantindo uma estimativa não enviesada do desempenho em dados nunca vistos.
- **Conjunto de Treinamento (70%):** Utilizado para o ajuste dos parâmetros dos modelos. É neste conjunto que são aplicadas as técnicas de balanceamento (SMOTE e Random Under Sampling).
- **Conjunto de Validação (15%):** Utilizado para avaliar o desempenho dos modelos durante a seleção de hiperparâmetros e para detectar *overfitting*. Mantém a distribuição original das classes para refletir melhor o cenário real.

A divisão foi realizada de forma **estratificada**, preservando a proporção original das classes em cada conjunto. Isso é especialmente importante dado o forte desbalanceamento do dataset, garantindo que cada partição seja representativa da distribuição geral.

Para o treinamento final do modelo selecionado, os conjuntos de treinamento e validação são combinados (85% dos dados totais), maximizando a quantidade de informação disponível para o aprendizado, enquanto o conjunto de teste permanece isolado para avaliação.

5.2.2 Métricas de Avaliação

Dado o forte desbalanceamento de classes do dataset e a importância clínica diferenciada de cada classe (especialmente `severe`), as seguintes métricas foram utilizadas:

- **F1-Score Macro:** Métrica principal de avaliação. Calcula o F1-Score de cada classe separadamente e depois computa a média não ponderada. Esta métrica trata todas as classes com igual importância, sendo adequada para problemas desbalanceados onde o desempenho nas classes minoritárias é crítico.
- **F1-Score por Classe:** Média harmônica entre precisão e revocação para cada classe individual, permitindo análise detalhada do desempenho.
- **Recall:** Proporção de casos reais da classe que foram corretamente identificados. Alta revocação é crucial para a classe `severe`, pois falsos negativos podem ter consequências graves.
- **Precision:** Proporção de predições positivas corretas. Alta precisão indica baixa taxa de falsos positivos, importante para evitar alarmes desnecessários.
- **Accuracy:** Proporção total de predições corretas. Embora seja uma métrica intuitiva, é menos informativa em problemas desbalanceados, servindo apenas como referência complementar.
- **F1-Score Weighted:** Média ponderada do F1-Score de cada classe pelo número de exemplos. Fornece uma visão do desempenho geral considerando a distribuição real das classes.

6 Spot-checking de Algoritmos

6.1 Metodologia

Para comparar os modelos selecionados de forma justa, realizamos o treinamento e avaliação com as mesmas partições do conjunto de dados e utilizando exatamente o mesmo *pipeline* de amostragem.

Para os modelos baseados em árvore (Árvore de Decisão, Random Forest e XGBoost), realizamos uma rápida exploração sobre diferentes valores de profundidade máxima, escolhendo aquele cujos resultados foram melhores no conjunto de validação para avaliar o modelo sobre o conjunto de teste.

6.2 Resultados do Spot-checking

6.2.1 Árvore de Decisão

A Figura 3 mostra a performance (medida através da métrica F1-Score) do modelo com diferentes valores de profundidade máxima. A partir da profundidade máxima 15, podemos observar um forte *overfitting*, visto que a performance do modelo melhora substancialmente no conjunto de treinamento enquanto, no conjunto de validação, pouca (ou nenhuma, a partir da profundidade máxima 20) melhoria é observada.

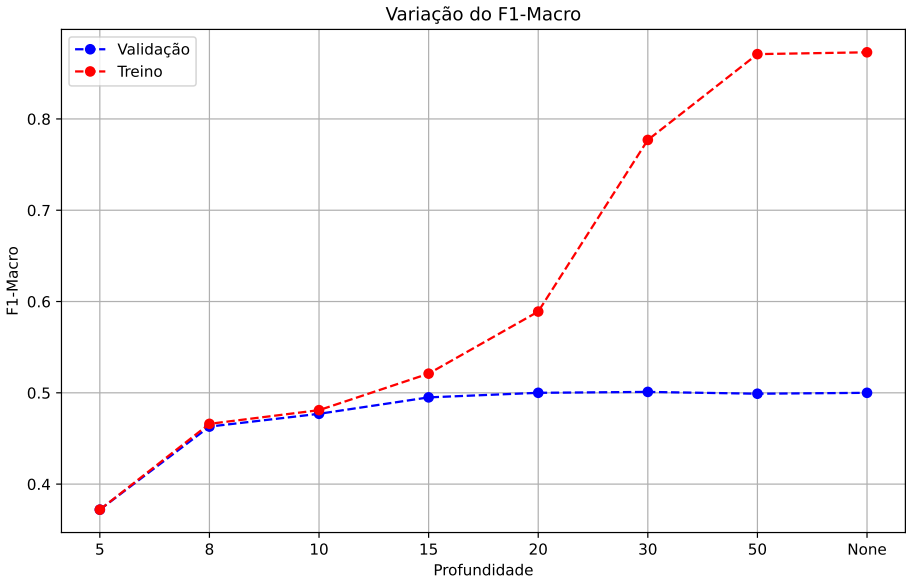

Figura 3: Variação do F1-Macro em função da profundidade máxima.

A Figura 4 mostra a matriz de confusão da Árvore de Decisão sobre o conjunto de teste. É possível perceber que a classe minoritária (`severe`), mesmo com a utilização de técnicas de amostragem durante o treinamento, ainda representa a classe cujo modelo possui maior dificuldade em classificar corretamente. Em particular, observa-se que o modelo tende a classificar exemplos da classe `severe` como `alarm`.

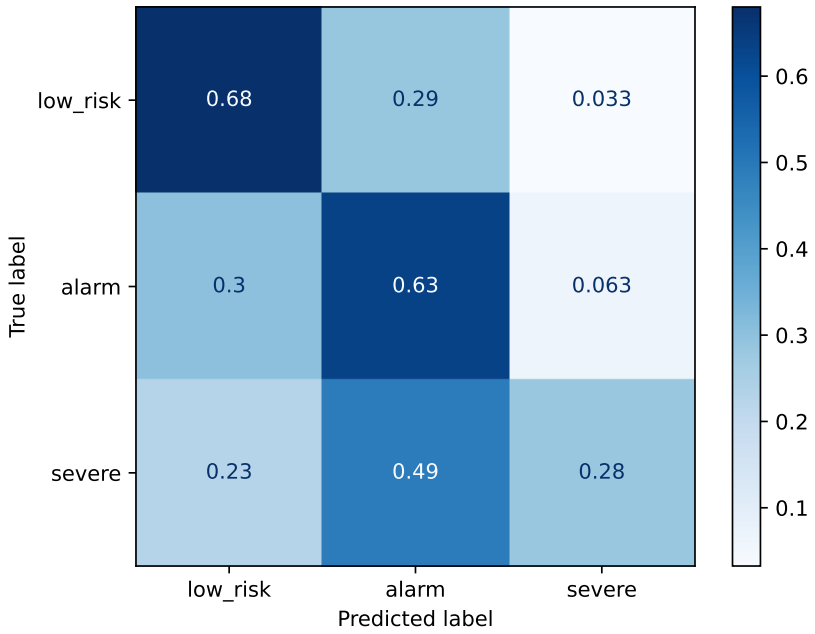

Figura 4: Matriz de confusão da Árvore de Decisão.

A Figura 5 mostra a performance, por classe, da Árvore de Decisão no conjunto de teste. Essa Árvore de Decisão foi instanciada com o valor de profundidade máxima que resultou na melhor performance no conjunto de validação. É possível verificar que houve um *overfitting* significativo, mesmo com o *pruning* sendo realizado.

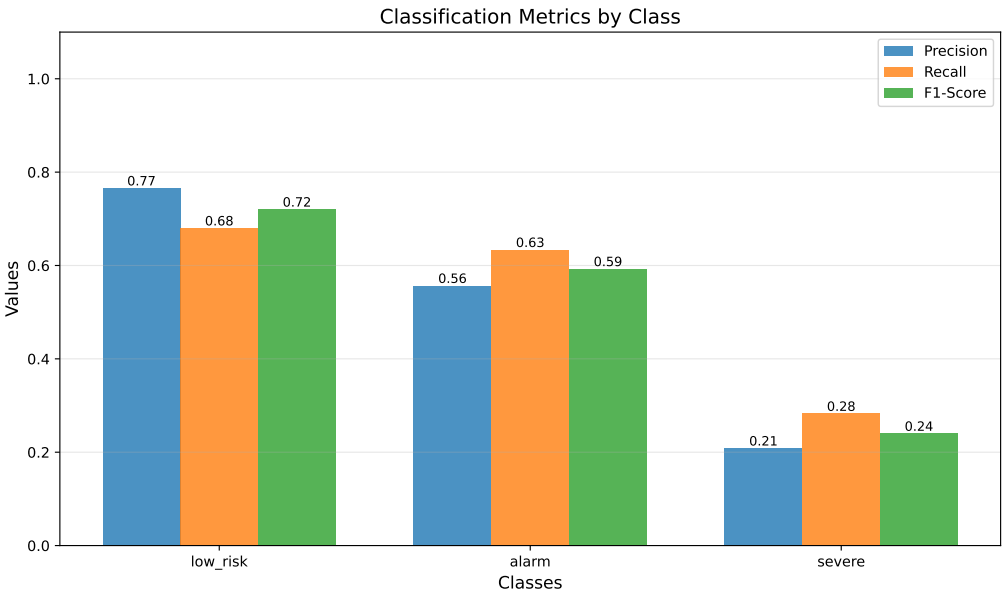

Figura 5: Métricas de performance da Árvore de Decisão por classe.

Tabela 4: Métricas de performance da Árvore de Decisão em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.510	0.532	0.518
Weighted Avg	0.666	0.647	0.654
Accuracy: 0.647			

6.2.2 Random Forest

A Figura 6 mostra a performance (medida através da métrica F1-Score) do modelo com diferentes valores de profundidade máxima. Assim como ocorre com as Árvores de Decisão, a partir da profundidade máxima 15 observamos a ocorrência de *overfitting*.

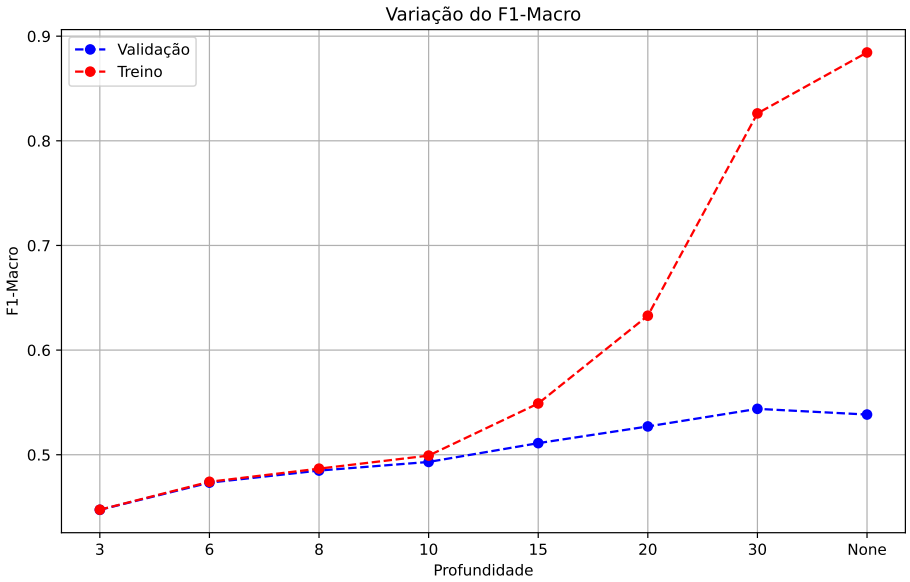

Figura 6: Variação do F1-Macro em função da profundidade máxima.

A Figura 7 mostra a matriz de confusão do Random Forest sobre o conjunto de teste. Assim como observado para a Árvore de Decisão, o modelo teve uma dificuldade bastante significativa de classificar as instâncias da classe `severe`, onde a maioria destas foram classificadas como `alarm`.

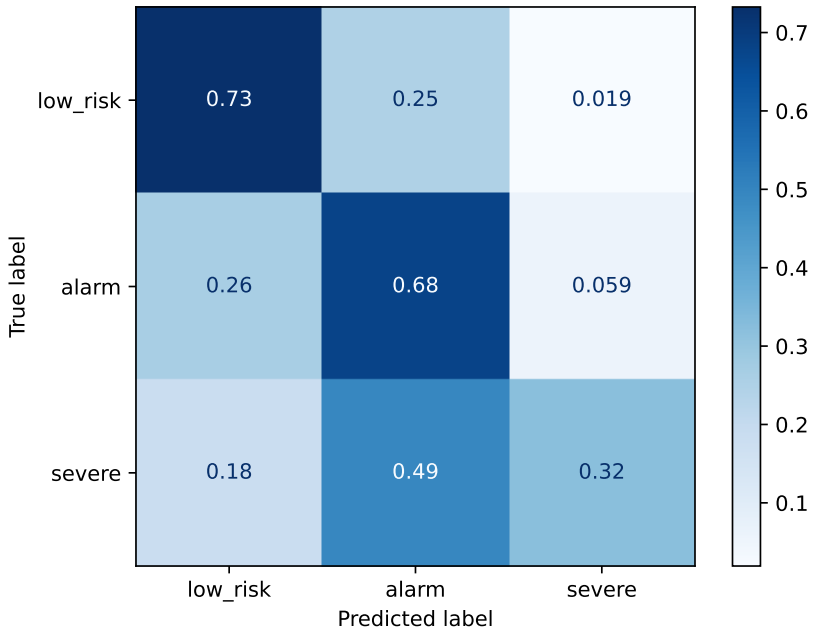

Figura 7: Matriz de confusão do Random Forest.

Assim como na Árvore de Decisão, o modelo Random Forest sofreu *overfitting* mesmo com a realização do *pruning*, como mostra a Figura 8.

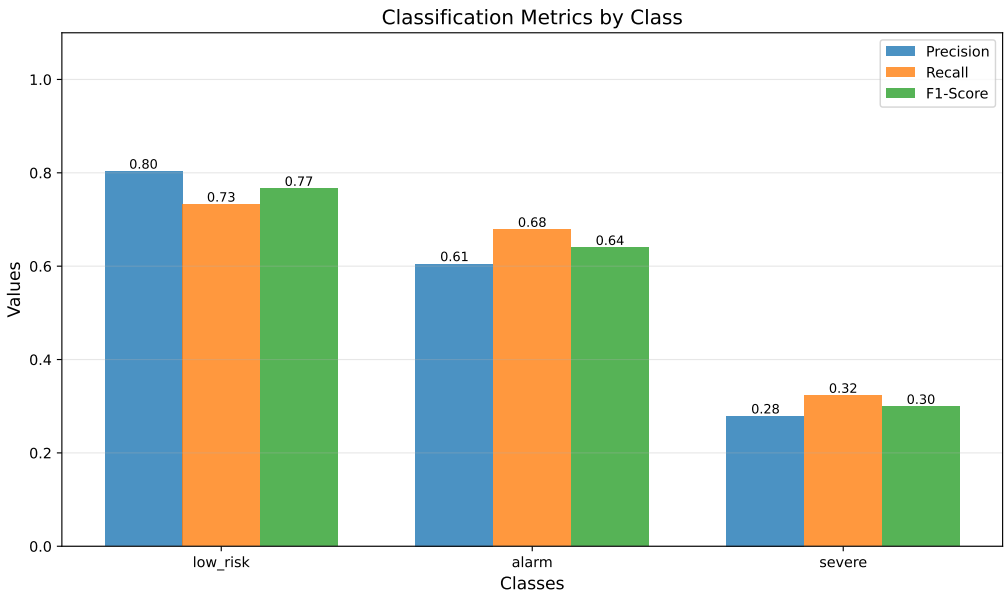

Figura 8: Métricas de performance do Random Forest por classe.

Tabela 5: Métricas de performance do Random Forest em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.562	0.578	0.568
Weighted Avg	0.708	0.696	0.701
Accuracy: 0.696			

6.2.3 Gradient Boosting

A Figura 9 mostra a performance (medida através da métrica F1-Score) do modelo com diferentes valores de profundidade máxima. Aqui, o *overfitting* ocorre a partir de uma profundidade menor do que o observado nos modelos anteriores (Árvore de Decisão e Random Forest).

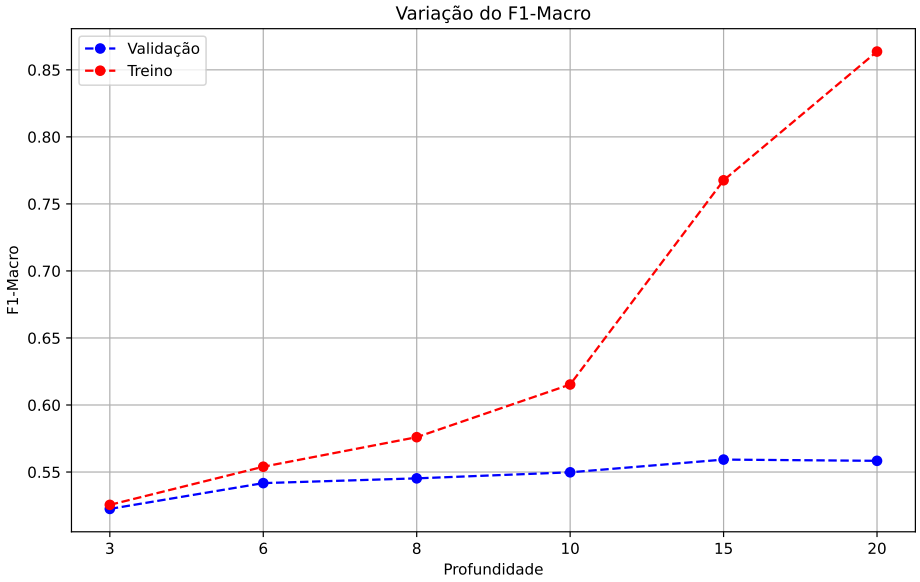

Figura 9: Variação do F1-Macro em função da profundidade máxima.

A Figura 10 mostra a matriz de confusão do XGBoost sobre o conjunto de teste. Assim como observado para os modelos anteriores, a classificação das instâncias da classe minoritária (`severe`) representa um desafio considerável.

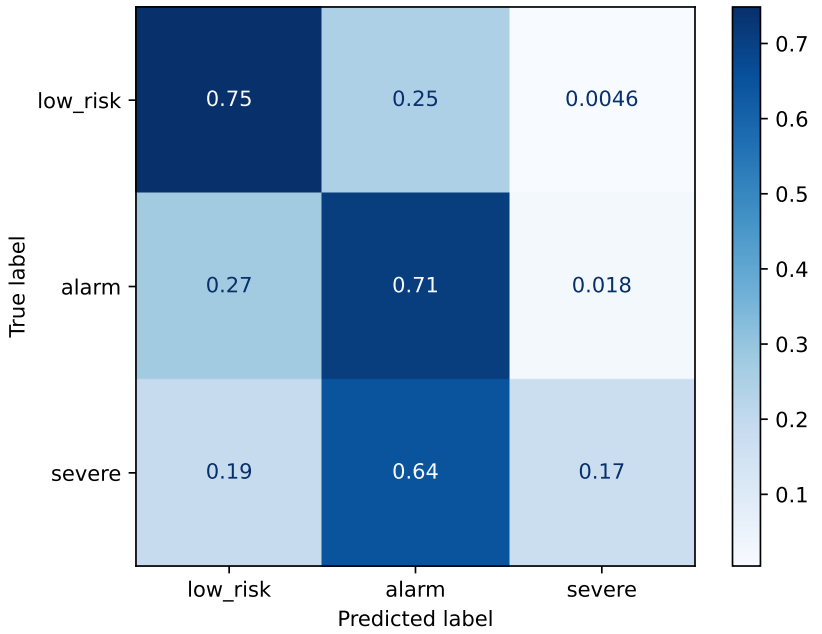

Figura 10: Matriz de confusão do XGBoost.

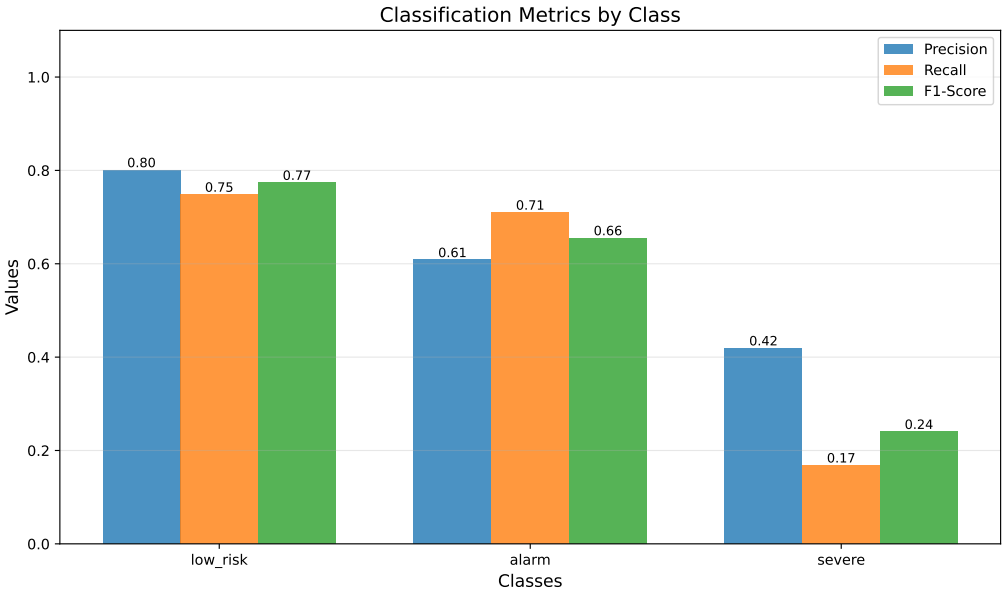

Figura 11: Métricas de performance do XGBoost por classe.

Tabela 6: Métricas de performance do XGBoost em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.610	0.542	0.557
Weighted Avg	0.714	0.711	0.709
Accuracy: 0.711			

6.2.4 Regressão Logística Multinomial

A Figura 12 mostra a matriz de confusão da Regressão Logística Multinomial sobre o conjunto de teste. Diferentemente do observado nos modelos baseados em árvore, a matriz de confusão mostra que a performance deste modelo não foi significativamente impactada pelo desbalanceamento da classe minoritária.

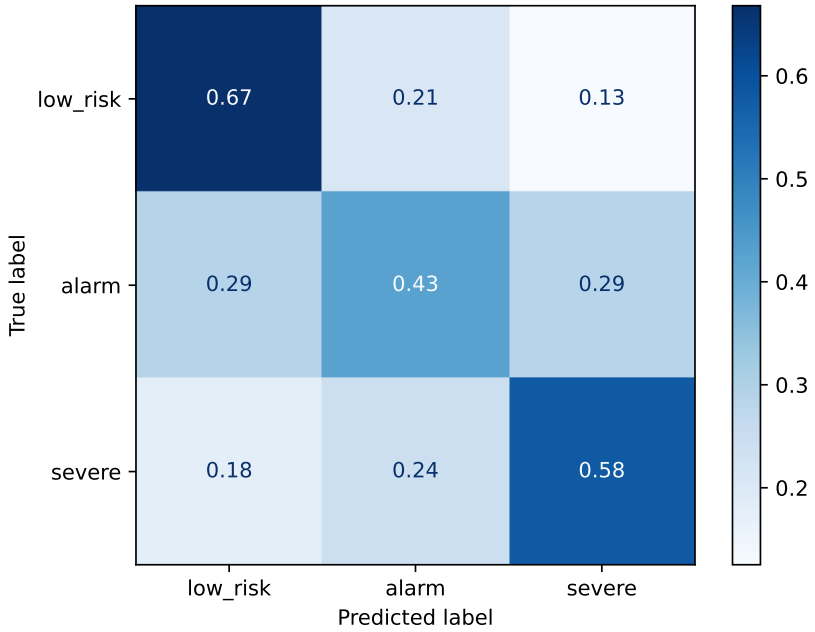

Figura 12: Matriz de confusão da Regressão Logística Multinomial.

Observa-se, na Figura 13, que a Regressão Logística Multinomial, embora apresente métricas de Precisão e F1-Macro significativamente inferiores às dos modelos baseados em árvore, obteve um ***recall* superior para a classe minoritária** (`severe`).

Isso sugere que a menor complexidade da Regressão Logística atuou como uma regularização, **impedindo um sobreajuste (overfitting) às classes majoritárias** (`alarm` e `low_risk`). Ao não se adaptar excessivamente aos padrões dos dados majoritários, o modelo linear manteve uma sensibilidade (recall) maior para a classe `severe`, embora ao custo de uma menor precisão geral.

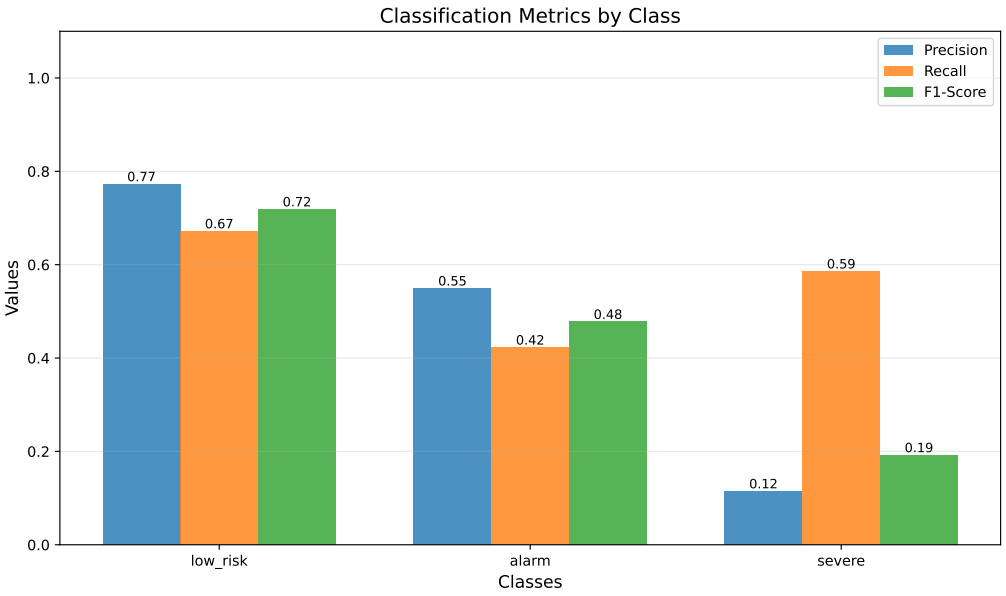

Figura 13: Métricas de performance da Regressão Logística Multinomial por classe.

Tabela 7: Métricas de performance da Regressão Logística Multinomial em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.479	0.561	0.463
Weighted Avg	0.664	0.576	0.608
Accuracy: 0.576			

6.2.5 Naive Bayes Gaussiano

A Figura 14 mostra a matriz de confusão do modelo Naive Bayes Gaussiano sobre o conjunto de teste. Aqui, o modelo não possui, como maior fonte de erro, os exemplos da classe `severe`, mas sim, os exemplos da classe `alarm`.

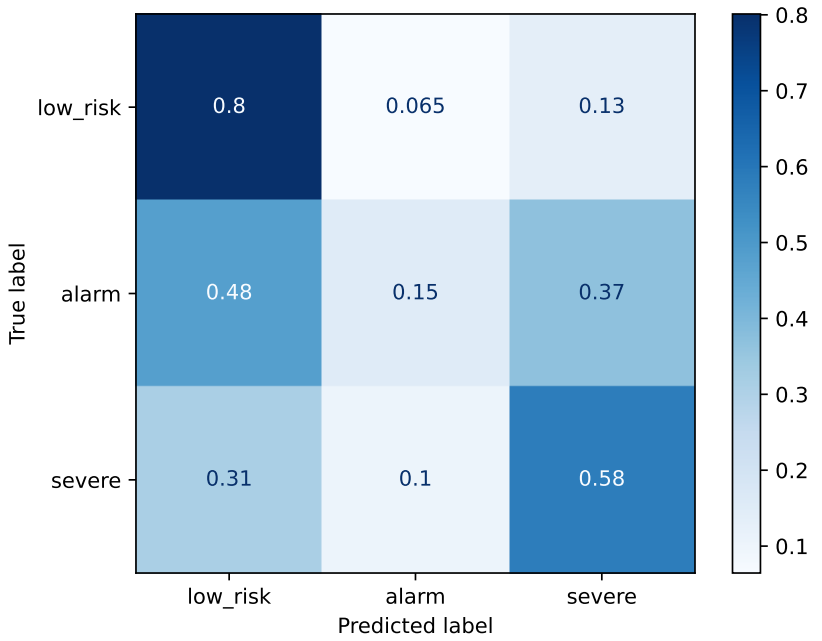

Figura 14: Matriz de confusão do Naive Bayes Gaussiano.

De forma análoga à Regressão Logística, os resultados do Naive Bayes Gaussiano (Figura 15) indicam que a menor complexidade do modelo **impede** o *overfitting* às classes majoritárias. Contudo, embora esse comportamento aumente o *recall* da classe minoritária, ele o faz ao custo de uma redução na precisão geral. No caso do Naive Bayes, **essa penalidade na precisão foi significativamente mais pronunciada**.

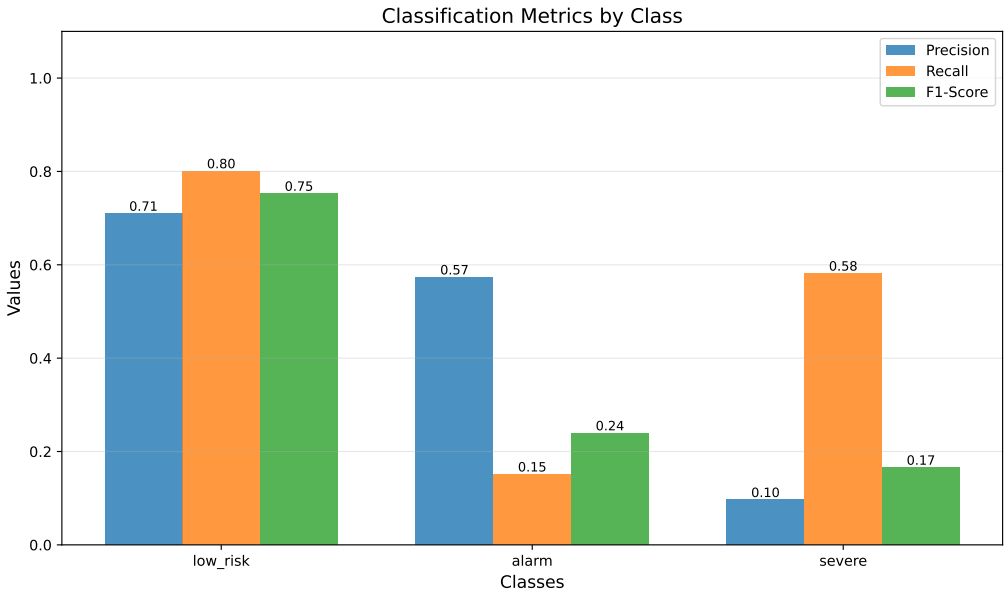

Figura 15: Métricas de performance do Naive Bayes Gaussiano por classe.

Tabela 8: Métricas de performance do Naive Bayes Gaussiano em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.455	0.505	0.365
Weighted Avg	0.632	0.530	0.517
Accuracy: 0.530			

6.2.6 MLP (Multi-layer Perceptron)

A Figura 16 mostra a matriz de confusão do MLP sobre o conjunto de teste. A maior fonte de erro para o MLP, assim como ocorre com os modelos baseados em árvore, é a classe minoritária `severe`.

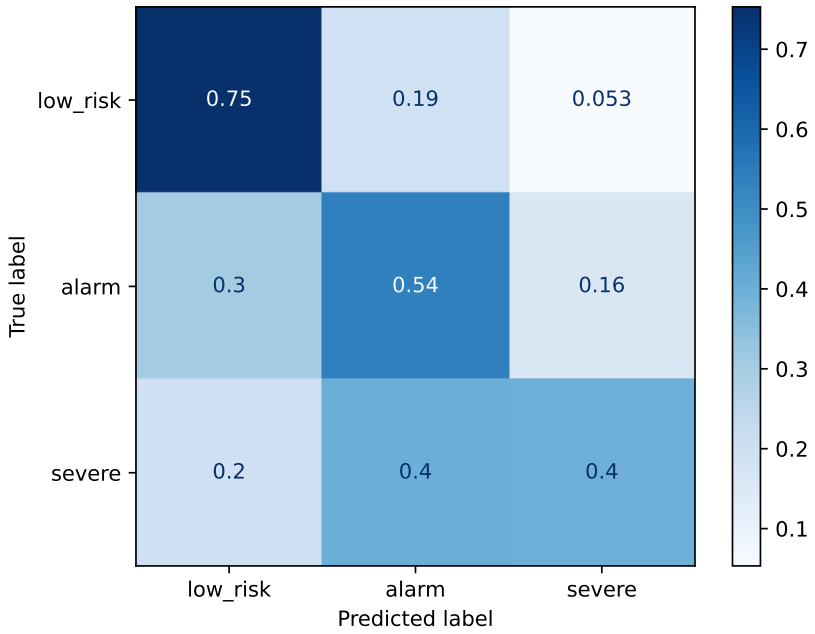

Figura 16: Matriz de confusão do MLP.

Os resultados apresentados na Figura 17 mostram que este modelo se posiciona em um "ponto intermediário"entre os modelos de alto viés (Regressão Logística e Naive Bayes) e os de alta variância (baseados em árvores). Para avaliar o modelo MLP, utilizamos um número fixo de camadas ocultas, de dimensões também fixas (`(64, 32)`). Essa escolha de hiper-parâmetros representa uma configuração razoavelmente rasa, e o modelo resultante não possui uma grande quantidade de parâmetros aprendíveis, o que explica o, bastante provável, alto viés apresentado pelo modelo.

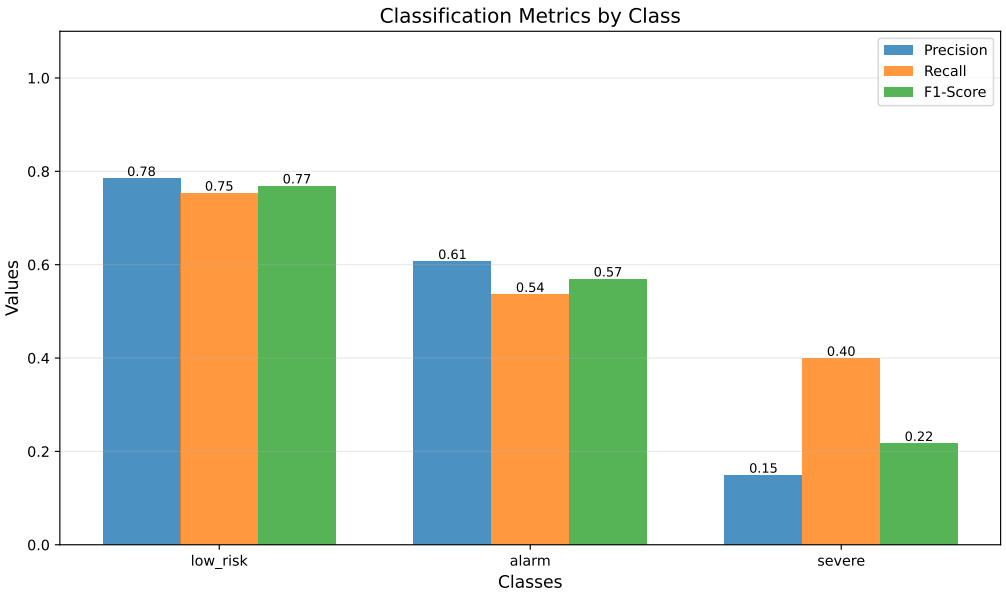

Figura 17: Métricas de performance do MLP por classe.

Tabela 9: Métricas de performance do MLP em geral.

	Precision	**Recall**	**F1-Score**
Macro Avg	0.513	0.563	0.518
Weighted Avg	0.693	0.658	0.672
Accuracy: 0.658			

6.3 Estratégias Mais Promissoras

Para a seleção final dos modelos, o Random Forest e o XGBoost foram escolhidos por apresentarem o melhor desempenho na métrica F1-macro. Este resultado indica uma performance geral superior e um melhor balanceamento na classificação de todas as classes.

Contudo, também pretendemos explorar o comportamento de modelos mais simples e com alto viés, como a Regressão Logística. O objetivo é analisar os diferentes *trade-offs* que esses modelos oferecem em comparação com as abordagens baseadas em árvore, que são mais complexas.

Referências

1. Base dos Dados - SINAN (Dengue). https://basedosdados.org/dataset/f51134c2-5ab9-4b table=9bdbca38-d97f-47fa-b422-84477a6b68c8

2. Scikit-learn Documentation. https://scikit-learn.org/

3. Imbalanced-learn Documentation. https://imbalanced-learn.org/

4. XGBoost Documentation. https://xgboost.readthedocs.io/

5. SINAN - Sistema de Informação de Agravos de Notificação. Ministério da Saúde. https://portalsinan.saude.gov.br/

6. ScienceDirect - User's guide to correlation coefficients. https://www.sciencedirect.com/science/article/pii/S2452247318302164

7. Dengue - Ministério da Saúde. https://www.gov.br/saude/pt-br/assuntos/saude-de-a-a-z/d/dengue#:~:text=Preven%C3%A7%C3%A3o,no%20sistema%20p%C3%BAblico%20de%20sa%C3%BAde.

8. Dengue - Agência Fiocruz de Notícias. https://agencia.fiocruz.br/dengue