UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO
FACULDADE DE CIÊNCIAS
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

TRABALHO DE CONCLUSÃO DE CURSO

Gabriel de Almeida Farias Cassola
Pedro Israel da Rocha Smith

SISTEMA DE GEOLOCALIZAÇÃO E ANÁLISE DE RISCO DE ACIDENTES DE TRÂNSITO PARA EMISSÃO DE ALERTAS EM BAURU

Bauru, 2025

Universidade Estadual Paulista Júlio de Mesquita Filho
Faculdade de Ciências

SISTEMAS DE INFORMAÇÃO

GABRIEL DE ALMEIDA FARIAS CASSOLA
PEDRO ISRAEL DA ROCHA SMITH

SISTEMA DE GEOLOCALIZAÇÃO E ANÁLISE DE RISCO DE ACIDENTES DE TRÂNSITO PARA EMISSÃO DE ALERTAS EM BAURU

Trabalho apresentado como exigência parcial para a conclusão do Curso de Bacharelado em Sistemas de Informação da Faculdade de Ciências – UNESP Campus Bauru.
Orientador: Prof. Dr. Clayton Reginaldo Pereira.

Bauru, 2025

Smith, Pedro Israel da Rocha.
Sistema de Geolocalização e Análise de Risco de Acidentes de Trânsito para Emissão de Alertas em Bauru. / Pedro Israel da Rocha Smith, Gabriel de Almeida Farias Cassola. – Bauru, 2025
67 f. : il.

Trabalho de conclusão de curso (Bacharelado – Sistemas de Informação)–Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru
Orientador: Clayton Reginaldo Pereira

1. Acidentes de Trânsito. 2. Análise preditiva. 3. Aprendizado de Máquina. 4. Geolocalização. 5. Segurança Viária I. Cassola, Gabriel de Almeida Farias. II. Smith, Pedro Israel da Rocha. III. Universidade Estadual Paulista. Faculdade de Ciências. II. Título.

RESUMO

O presente projeto desenvolveu um sistema de geolocalização e análise de risco de acidentes de trânsito, com o intuito de alertar os usuários em tempo real sobre zonas perigosas na cidade de Bauru. O problema norteador deste trabalho esta relacionado ao elevado índice de acidentes de trânsito, que acarreta sérios impactos na saúde pública, na economia e na mobilidade urbana, agravados pela falta de um monitoramento efetivo e de intervenções precisas em pontos críticos. A solução implementada consistiu na integração de dados históricos oriundos de fontes públicas, com técnicas de análise espacial e algoritmos de *machine learning*, para identificar padrões de acidentes e antecipar situações de risco. O sistema foi desenvolvido com módulos dedicados à coleta e pré-processamento de dados, análise preditiva, geolocalização interativa e comunicação via API, permitindo uma resposta rápida e eficaz. Os objetivos do projeto incluíram, dentre outros, a redução dos índices de acidentes por meio da emissão de alertas preventivos, o aprimoramento da gestão do tráfego urbano por parte dos órgãos públicos e a criação de um modelo de intervenção que pode ser replicado em outras cidades com desafios similares. Com isso, o trabalho contribuiu para a melhoria da segurança viária e a promoção de um ambiente urbano mais seguro e organizado.

Palavras-chave: acidentes de trânsito; análise preditiva; aprendizado de máquina; geolocalização; segurança viária.

ABSTRACT

This project developed a geolocation and traffic accident risk analysis system aimed at alerting users in real-time about hazardous zones in the city of Bauru. The guiding problem of this study addresses the high rate of traffic accidents, which causes serious impacts on public health, the economy, and urban mobility. These issues are exacerbated by the lack of effective monitoring and precise interventions at critical points. The implemented solution involved integrating historical data from public sources with spatial analysis techniques and machine learning algorithms to identify accident patterns and anticipate risk situations. The system was developed with dedicated modules for data collection and pre-processing, predictive analysis, interactive geolocation, and API communication, enabling a rapid and effective response. Project objectives included, among others, reducing accident rates by issuing preventive alerts, improving urban traffic management by public agencies, and creating an intervention model that can be replicated in other cities facing similar challenges. Consequently, this work contributed to improving road safety and promoting a safer, more organized urban environment.

Keywords: geolocation; machine learning; predictive analysis; road safety; traffic accidents;.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AED Análise Exploratória de Dados

API Application Programming Interface (Interface de Programação de Aplicações)

ASGI Asynchronous Server Gateway Interface (Interface de Gateway de Servidor Assíncrono)

CSS Cascading Style Sheets (Folhas de estilo em cascata)

CV Cross-Validation (Validação Cruzada)

CSV Comma-Separated Values (Valores Separados por Vírgula)

EMDURB Empresa Municipal de Desenvolvimento Urbano e Rural de Bauru

ES6+ ECMAScript 6+

GIL Global Interpreter Lock (Trava do Interpretador Global)

GIS Geographic Information System (Sistema de Informações Geográficas)

GPS Global Positioning System (Sistema de Posicionamento Global)

I/O Input/Output (Entrada/Saída)

IBGE Instituto Brasileiro de Geografia e Estatística

INFOSIGA Sistema de Informações Gerenciais de Acidentes de Trânsito

INMET Instituto Nacional de Meteorologia

JSON JavaScript Object Notation (Notação de Objeto JavaScript)

ML Machine Learning (Aprendizado de Máquina)

MLP Multilayer Perceptron (Perceptron de Múltiplas Camadas)

OMS Organização Mundial da Saúde

PLANMOB Plano de Mobilidade Urbana

PR-AUC Area Under the Precision-Recall Curve (Área sob a Curva Precisão-Recall)

REST Representational State Transfer (Transferência de Estado Representacional)

RF Random Forest (Floresta Aleatória)

ROC-AUC Area Under the Receiver Operating Characteristic Curve (Área sob a Curva ROC)

SHAP SHapley Additive exPlanations (Explicações Aditivas de Shapley)

SVM Support Vector Machine (Máquina de Vetores de Suporte)

TCC Trabalho de Conclusão de Curso

UI User Interface (Interface de Usuário)

UML Unified Modeling Language (Linguagem de Modelagem Unificada)

UTC Coordinated Universal Time (Tempo Universal Coordenado)

XGBoost Extreme Gradient Boosting (Aprimoramento de Gradiente Extremo)

SUMÁRIO

1 INTRODUÇÃO

Os acidentes de trânsito representam atualmente um dos maiores desafios globais relacionados à saúde pública e à mobilidade urbana. De acordo com a Organização Mundial da Saúde (OMS), mais de 1,3 milhão de pessoas morrem todos os anos em decorrência de acidentes viários, enquanto outras 50 milhões sofrem lesões, muitas delas com sequelas permanentes. Além das perdas humanas, esses incidentes geram impactos profundos no sistema de saúde, na economia e na segurança das cidades.

No Brasil, a situação é igualmente preocupante. O país ocupa uma das posições mais críticas no cenário mundial, figurando como o terceiro em número absoluto de mortes no trânsito, atrás apenas da Índia e da China, conforme aponta o relatório *Global Status Report on Road Safety 2018* da OMS. Diversos fatores contribuem para esse quadro alarmante, como o crescimento desordenado dos centros urbanos, a sobrecarga da infraestrutura viária e o aumento acelerado da frota de veículos sem o devido planejamento em segurança e mobilidade.

A cidade de Bauru, localizada no interior do Estado de São Paulo, exemplifica de forma clara os desafios enfrentados pelos centros urbanos em relação à mobilidade e à segurança viária. De acordo com o Plano de Mobilidade Urbana (PLANMOB), a população do município cresceu de 341.621 habitantes (IBGE, 2007) para 371.690 (estimativa IBGE, 2017), representando um aumento de 8,8%. No mesmo intervalo, a frota de veículos motorizados passou de 161.275 para 274.306, o que corresponde a um crescimento de aproximadamente 70%, atingindo a média de 1,36 habitante por veículo. Esse crescimento acelerado da frota, quando não acompanhado por investimentos estruturais, tecnológicos e educacionais, contribui diretamente para o aumento da ocorrência de acidentes de trânsito.

Esse cenário é evidenciado por dados do portal Infosiga-SP, que posicionam Bauru como a 10ª cidade com maior número de óbitos no trânsito entre os municípios paulistas com mais de 300 mil habitantes, registrando uma taxa de mortalidade de 12,33 mortes por 100 mil habitantes.

Diante desse panorama, este trabalho apresenta a proposta do desenvolvimento de um sistema de geolocalização e análise preditiva de acidentes de trânsito em tempo real para a cidade de Bauru. Ao integrar dados georreferenciados do Infosiga-SP com técnicas de análise espacial, a ferramenta visa identificar padrões de acidentes, como horários de pico ou locais recorrentes de colisões, permitindo que gestores públicos priorizem intervenções em pontos críticos e que motoristas recebam alertas preventivos. Dessa forma, o sistema busca não apenas reduzir os índices de acidentes, mas também servir como modelo para outras cidades de porte médio com desafios similares de mobilidade.

No entanto, o desenvolvimento do sistema enfrenta desafios significativos, começando pela qualidade dos dados, que precisam ser precisos, atualizados e georreferenciados para garantir a eficácia dos alertas. Além disso, a análise preditiva exige capacidade técnica para

processar grandes volumes de informações em tempo real, o que demanda infraestrutura computacional adequada. Outro obstáculo é a criação de modelos de análise robustos capazes de identificar padrões confiáveis no cenário dinâmico do trânsito de Bauru. Por fim, a adoção do sistema depende do engajamento de gestores públicos e da população, exigindo estratégias de conscientização e treinamento para sua efetiva implementação. A superação desses desafios será fundamental para assegurar o impacto positivo da solução proposta.

A organização deste trabalho estrutura-se nas seções subsequentes da seguinte forma. No Capítulo 2, é apresentado o detalhamento do problema, expondo a natureza dos acidentes em Bauru, os segmentos da sociedade afetados, os impactos econômicos e a definição dos requisitos funcionais, de desempenho e de custo da solução. Em seguida, o Capítulo 3 mapeia as soluções existentes, baseadas em veículos, infraestrutura e análise de dados, realizando uma crítica comparativa que fundamenta os objetivos do sistema híbrido de predição e alerta proposto.

A metodologia e a implementação técnica são descritas no Capítulo 4, que detalha a arquitetura modular do sistema, abrangendo desde a coleta e pré-processamento de dados até a análise preditiva, geolocalização e o desenvolvimento da API e do frontend. Posteriormente, o Capítulo 5 descreve os resultados e os procedimentos adotados para a validação quantitativa da eficácia do modelo preditivo, bem como a validação funcional e de usabilidade do sistema. Por fim, o Capítulo 6 sintetiza as conclusões, os desafios superados e o impacto do projeto, seguido pelo Capítulo 7, que reúne as referências bibliográficas utilizadas ao longo do desenvolvimento.

2 DETALHAMENTO DO PROBLEMA

Os acidentes de trânsito são um problema recorrente em centros urbanos de médio e grande porte, com consequências diretas na saúde pública, na mobilidade urbana e na economia. No caso específico da cidade de Bauru, a situação tem se agravado nas últimas décadas em virtude do aumento significativo da frota de veículos, do crescimento desordenado da malha urbana e da ausência de mecanismos eficientes de prevenção e monitoramento. Essa realidade evidencia a necessidade de soluções tecnológicas capazes de mitigar os impactos dos acidentes e tornar o trânsito mais seguro para todos os cidadãos.

2.1 **A natureza do problema**

A natureza do problema está diretamente relacionada à recorrência de acidentes de trânsito em áreas críticas da cidade de Bauru. Tais ocorrências, muitas vezes, manifestam-se de forma concentrada em trechos viários específicos, como cruzamentos, rotatórias e vias com histórico de má sinalização, má conservação ou alta velocidade. No entanto,

esses pontos críticos não são facilmente identificados pelos condutores, especialmente por aqueles que não possuem familiaridade com a região. A ausência de sistemas de informação e alerta em tempo real contribui para a carência de consciência situacional dos motoristas, elevando a exposição ao risco.

Além disso, a subutilização de dados públicos sobre acidentes, como os disponibilizados pelo Infosiga-SP, impede a formulação de políticas preventivas e o uso estratégico dessas informações para alertar motoristas em tempo real.

Em 2023, Bauru registrou 5.181 multas por uso de celular ao volante, sendo 2.997 por manuseio e 2.184 por uso durante a condução (BASTOS, 2023), infrações graves segundo o Código de Trânsito Brasileiro (BRASIL, 1997, art. 252). Esses números, reportados pelo Jornal da Cidade de Bauru (JCNET), evidenciam comportamentos de risco que elevam a probabilidade de acidentes. Para pedestres, a falta de atenção às faixas de segurança é um problema recorrente, com iniciativas como o "Faixa Viva"em Lençois Paulista sugerindo a necessidade de educação no trânsito (PREFEITURA MUNICIPAL DE LENÇÓIS PAULISTA, 2023).

2.2 Segmentos da sociedade afetados pelos acidentes de trânsito

O problema dos acidentes de trânsito afeta diversos segmentos da sociedade de maneira direta e indireta. Motoristas, passageiros, pedestres e ciclistas estão expostos a riscos constantes ao se deslocarem pela cidade, sobretudo em regiões que apresentam maior índice de sinistros. Famílias são impactadas emocionalmente e financeiramente quando ocorrem acidentes graves, especialmente em casos que resultam em mortes ou sequelas permanentes. O sistema público de saúde também é afetado, uma vez que precisa atender às vítimas, muitas vezes em situações de emergência, o que sobrecarrega os hospitais e consome recursos significativos. As seguradoras e empresas privadas sofrem prejuízos com indenizações, reparos e perda de produtividade. Além disso, gestores públicos enfrentam dificuldades para alocar recursos de forma eficiente quando não há ferramentas precisas de monitoramento e análise preditiva para auxiliar na tomada de decisões.

2.3 Impacto econômico e benefícios esperados com a solução

Os impactos econômicos causados pelos acidentes de trânsito são elevados e abrangem diversos setores. No Brasil, estima-se que os custos anuais com acidentes de trânsito superam os R$50 bilhões, considerando despesas com atendimento médico, perda de produtividade, danos materiais e processos judiciais (PEREIRA et al., 2020). Em Bauru, embora os números locais sejam proporcionalmente menores, a cidade ainda figura entre as que mais registram óbitos no trânsito no estado de São Paulo, conforme dados do Infosiga-SP, que declarou Bauru como a 10ª cidade com maior número de óbitos no trânsito entre os municípios paulistas com mais de 300 mil habitantes.

A adoção de uma solução baseada em geolocalização e análise de dados pode gerar benefícios significativos. Espera-se que a implementação de um sistema de alertas em tempo real contribua para a redução dos acidentes, o que diminuiria os custos hospitalares, os prejuízos materiais e os impactos sociais. Além disso, a melhoria da segurança viária em regiões estratégicas pode tornar o ambiente urbano mais seguro, atraindo investimentos e promovendo maior qualidade de vida à população.

2.4 Requisitos funcionais, de desempenho e de custo

Para que a solução tecnológica proposta seja efetiva na mitigação dos acidentes de trânsito em Bauru e atenda às demandas de segurança viária descritas anteriormente, é imprescindível estabelecer critérios técnicos e operacionais rigorosos que norteiem o seu desenvolvimento. A definição destes requisitos visa assegurar que o sistema não apenas processe os dados corretamente, mas que o faça com a agilidade necessária para um contexto de trânsito e dentro de uma realidade orçamentária que permita sua escalabilidade.

Dessa forma, esta seção detalha as especificações do projeto em três dimensões fundamentais: os requisitos funcionais, que descrevem as capacidades de interação com dados do Infosiga-SP e geolocalização; os requisitos de desempenho, que estipulam os limites de latência e precisão para garantir a utilidade dos alertas em tempo real; e os requisitos de custo, que orientam a escolha por tecnologias acessíveis para viabilizar a implementação em larga escala.

- **Requisitos funcionais:** O sistema deverá ser capaz de coletar, processar e analisar dados históricos de acidentes provenientes do Infosiga-SP e integrar essas informações com a localização atual do usuário por meio de um sistema de geolocalização. Também deverá emitir alertas em tempo real sempre que o usuário estiver se aproximando de uma zona considerada de alto risco;

- **Requisitos de desempenho:** A aplicação deve apresentar respostas rápidas, com tempo mínimo de latência entre a identificação da localização do usuário e o envio do alerta. A precisão da localização e da análise espacial também deve ser alta, garantindo confiabilidade nas informações fornecidas; e

- **Requisitos de custo:** A solução deverá ser de baixo custo para viabilizar sua implementação e uso em larga escala, considerando a possibilidade de adoção por órgãos públicos ou integração com sistemas de navegação já utilizados pelos motoristas. O uso de tecnologias livres, como APIs gratuitas de mapas e plataformas de dados abertos, pode contribuir para a redução dos custos totais do projeto.

Com o cumprimento desses requisitos, espera-se que a ferramenta proposta tenha grande potencial de impacto positivo, contribuindo não apenas para a diminuição dos

índices de acidentes em Bauru, mas também servindo como base para a replicação do modelo em outras cidades com características semelhantes.

3 SOLUÇÕES EXISTENTES E OBJETIVOS DA PROPOSTA

Este capítulo apresenta uma pesquisa sobre soluções existentes relacionadas ao uso de sistemas de geolocalização e análise de acidentes de trânsito para emissão de alertas, com foco em sua aplicabilidade em Bauru. O aumento dos acidentes na cidade, conforme dados recentes, exige soluções que combinem tecnologia e análise de dados para melhorar a segurança viária. Serão descritos sistemas baseados em veículos, em infraestrutura e de análise de dados para previsão de acidentes, com profundidade suficiente para avaliar sua competência, aspectos positivos e negativos. Ao final, uma análise crítica dessas soluções fundamentará os objetivos do sistema proposto, destacando suas diferenças. Abaixo, são descritos alguns pontos importantes do sistema:

3.1 Soluções existentes

3.1.1 Sistemas baseados em veículos

Sistemas baseados em veículos utilizam sensores integrados, como GPS e acelerômetros, para detectar acidentes e emitir alertas automáticos. Um exemplo é o *"IoT Based Automatic Vehicle Accident Detection and Rescue System"* (Hashstudioz, 2024), que emprega sensores de impacto e geolocalização para identificar colisões e enviar coordenadas a serviços de emergência. Abaixo, são descritos alguns pontos importantes do sistema:

- Competência: Detecta acidentes em tempo real com alta precisão, fornecendo localização exata para resposta rápida.
- Aspectos positivos: Integração direta ao veículo reduz falhas humanas, e a automação agiliza o socorro. É eficaz em áreas urbanas como Bauru, onde o tráfego é intenso.
- Aspectos negativos: O custo de instalação é elevado, especialmente em veículos mais antigos, comuns na frota de Bauru (JCNET, 2024). Requer manutenção regular e adesão ampla para ser eficiente.

3.1.2 Sistemas baseados em infraestrutura

Esses sistemas utilizam sensores e câmeras instalados nas vias para monitorar o tráfego e emitir alertas. O estudo *"Smart Roads for Autonomous Accident Detection and Warnings"* (PMC, 2022) propõe um sistema com luzes e sons para alertar motoristas sobre acidentes

ou condições adversas, usando infraestrutura viária inteligente. Abaixo, são descritos alguns pontos importantes do sistema:

- Competência: Oferece cobertura ampla, detectando acidentes independentemente dos veículos envolvidos, ideal para vias críticas de Bauru.
- Aspectos positivos: Permite monitoramento contínuo e pode ser integrado a sistemas de gestão urbana, beneficiando o planejamento viário.
- Aspectos negativos: Os custos de implantação e manutenção são altos, e a infraestrutura viária limitada de Bauru (G1 Bauru, 2025) pode dificultar a implementação. Há também questões de privacidade com o uso de câmeras.

3.1.3 Sistemas de análise de dados para previsão de acidentes

Esses sistemas utilizam dados históricos e geográficos para prever áreas de risco e emitir alertas preventivos. O *"GIS-Based Accident Location and Analysis System (GIS-ALAS)"* (TRID, 1998) mapeia pontos críticos com base em sistemas de informação geográfica, enquanto *"Geographical Information Systems Aided Traffic Accident Analysis"* (ScienceDirect, 2007) usa análise de densidade para identificar hotspots.

- Competência: Identifica padrões de acidentes, permitindo alertas proativos em áreas de alto risco, como cruzamentos perigosos em Bauru.
- Aspectos positivos: Baixo custo operacional após implementação e foco na prevenção, alinhado às necessidades de planejamento urbano.
- Aspectos negativos: Depende de dados precisos e atualizados, que podem ser escassos em Bauru devido à subnotificação. Não oferece resposta em tempo real a acidentes.

3.2 Análise crítica das soluções existentes

Os sistemas baseados em veículos são eficazes para detecção em tempo real, mas sua aplicação em Bauru é limitada pelo custo e pela frota envelhecida, que cresceu 26,2% em veículos com mais de 11 anos entre 2021 e 2023 (JCNET, 2024). Sistemas baseados em infraestrutura têm potencial para monitoramento amplo, mas a falta de recursos para modernizar as vias de Bauru (G1 Bauru, 2025) e os altos custos os tornam inviáveis a curto prazo. Já os sistemas de análise de dados são valiosos para prevenção, mas sua dependência de dados históricos e a ausência de alertas em tempo real limitam sua resposta a emergências imediatas. Nenhuma solução integra completamente detecção em tempo real, análise preditiva e emissão de alertas adaptados às condições locais de Bauru, como o aumento de acidentes com motociclistas (G1 Bauru, 2025).

3.3 Objetivos do sistema proposto

O presente projeto teve como objetivo desenvolver um sistema inteligente de geolocalização e análise preditiva de acidentes de trânsito na cidade de Bauru. A proposta visou unir tecnologias de ciência de dados, *machine learning*, geoprocessamento e desenvolvimento *web* para fornecer uma ferramenta útil na prevenção de acidentes e no auxílio à tomada de decisões por parte de condutores e autoridades.

A seguir, são apresentados os principais objetivos definidos para o desenvolvimento deste sistema:

- Desenvolver e treinar um **modelo preditivo** de *machine learning* capaz de identificar padrões e prever zonas com maior risco de acidentes.
- Criar um **sistema de emissão de alertas geográficos** com base nas predições do modelo e nos dados de localização dos usuários.
- Implementar uma **API REST**, utilizando FastAPI, para interligar o modelo, os dados e a interface web do sistema.
- Desenvolver uma **interface web interativa**, utilizando Vue.js e ferramentas de visualização geoespacial, para exibir os dados e alertas em tempo real.

Diferentemente dos sistemas baseados em veículos, o sistema proposto não exigiu instalação universal, reduzindo custos. Comparado aos sistemas de infraestrutura, evitou investimentos massivos em vias, aproveitando tecnologias existentes. E, em relação aos sistemas de análise de dados, adicionou detecção em tempo real, oferecendo uma abordagem híbrida que atendeu às necessidades específicas de Bauru.

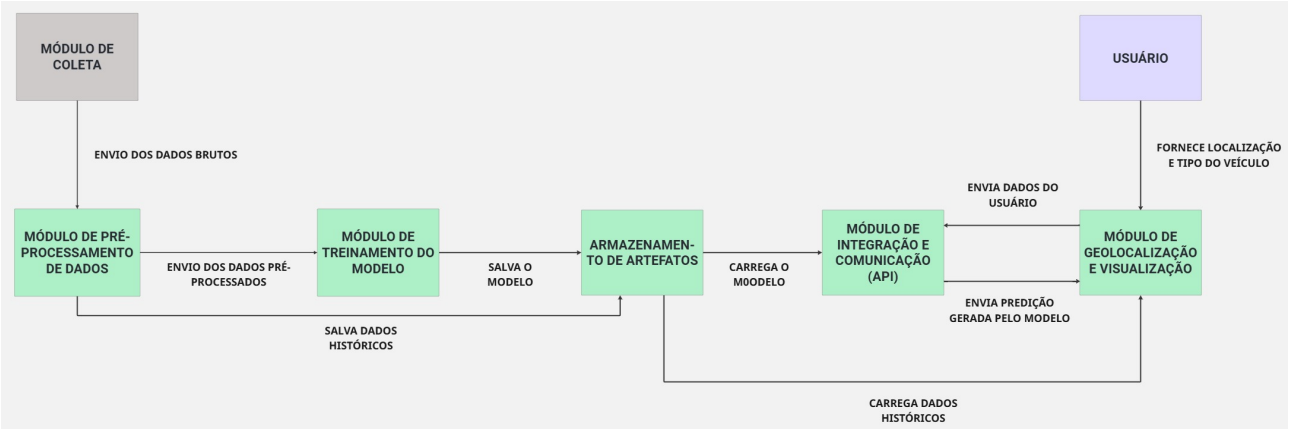
4 IMPLEMENTAÇÃO E USO

Este capítulo apresenta a descrição detalhada da metodologia de pesquisa e desenvolvimento adotada para a concepção e implementação do sistema proposto. Com o intuito de garantir a replicabilidade e a validade científica do projeto, toda a estrutura do sistema foi modularizada em componentes distintos. A seção subsequente destrincha cada um desses módulos, desde a coleta e o pré-processamento dos dados até o treinamento do modelo preditivo e a arquitetura da aplicação, descrevendo suas funções específicas, a lógica de programação inerente e as ferramentas e tecnologias empregadas. Adicionalmente, são justificadas as escolhas técnicas realizadas, demonstrando o alinhamento entre as necessidades do projeto e as soluções tecnológicas de ponta.

A figura 1 ilustra o diagrama de blocos da arquitetura completa do sistema, desde a aquisição dos dados brutos até a interface do usuário final. Os blocos representam

os principais componentes da solução, cujo fluxo e implementação são detalhados nas subseções a seguir.

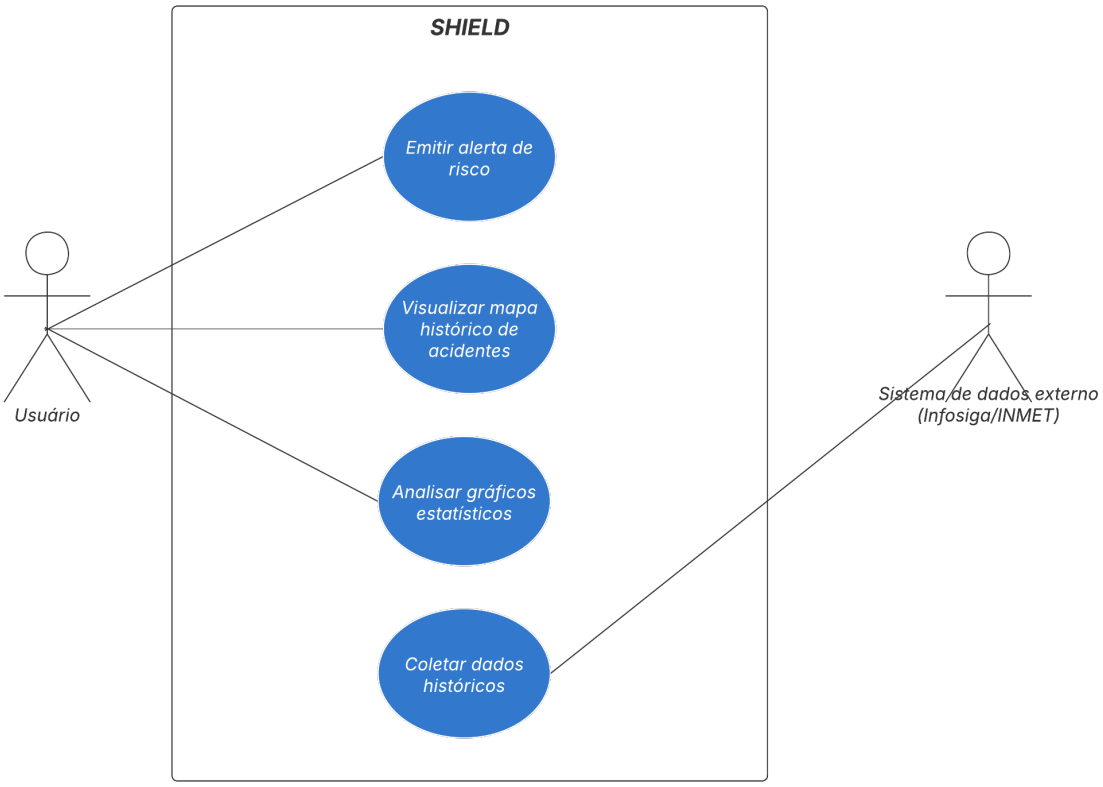
Figura 1 – Diagrama arquitetural



Fonte: Elaborada pelos autores.

Após a definição da arquitetura macro do sistema, o detalhamento do projeto é realizado através da Linguagem de Modelagem Unificada (UML). A dimensão funcional da aplicação é explorada por meio de um Diagrama de Casos de Uso (Figura 2), o qual delimita o escopo e as interações do principal agente (o usuário) com as capacidades sistêmicas, como a emissão de alertas e a consulta de informações históricas.

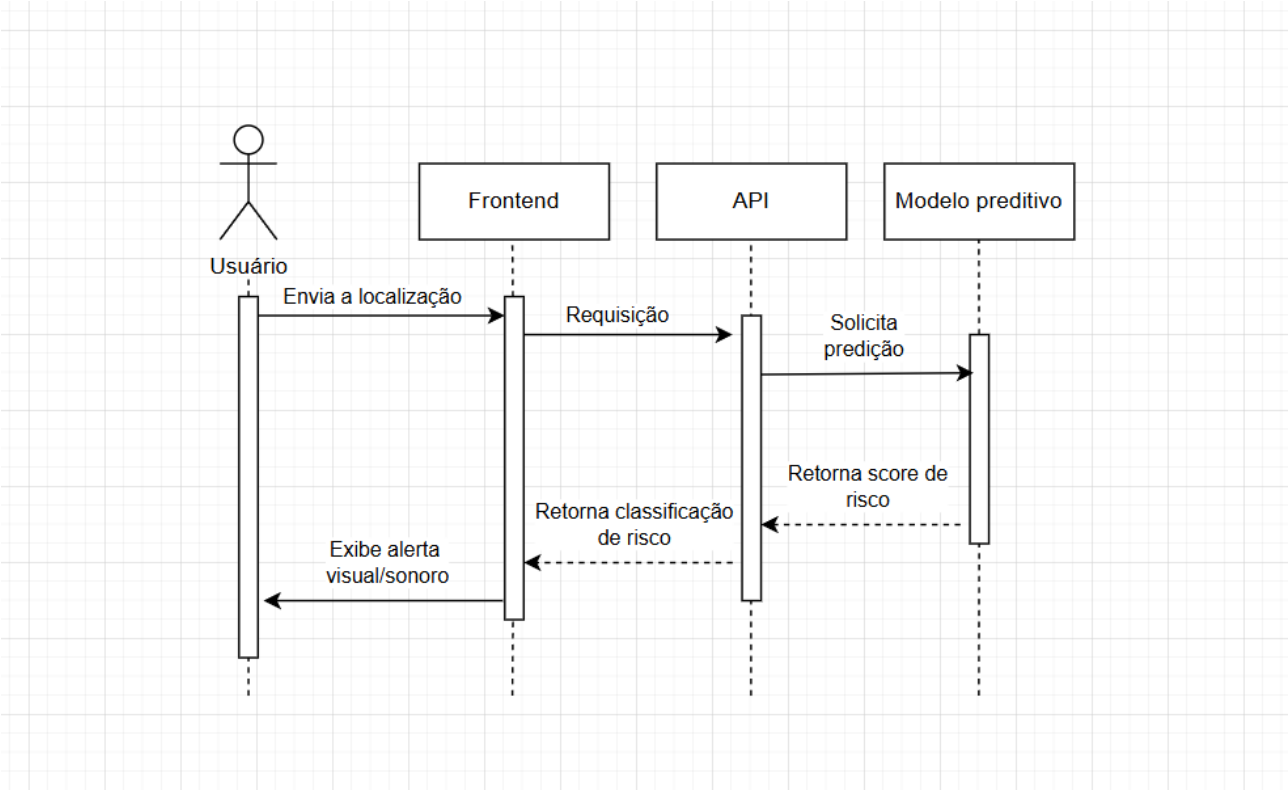
Figura 2 – Casos de uso do sistema



Fonte: Elaborada pelos autores.

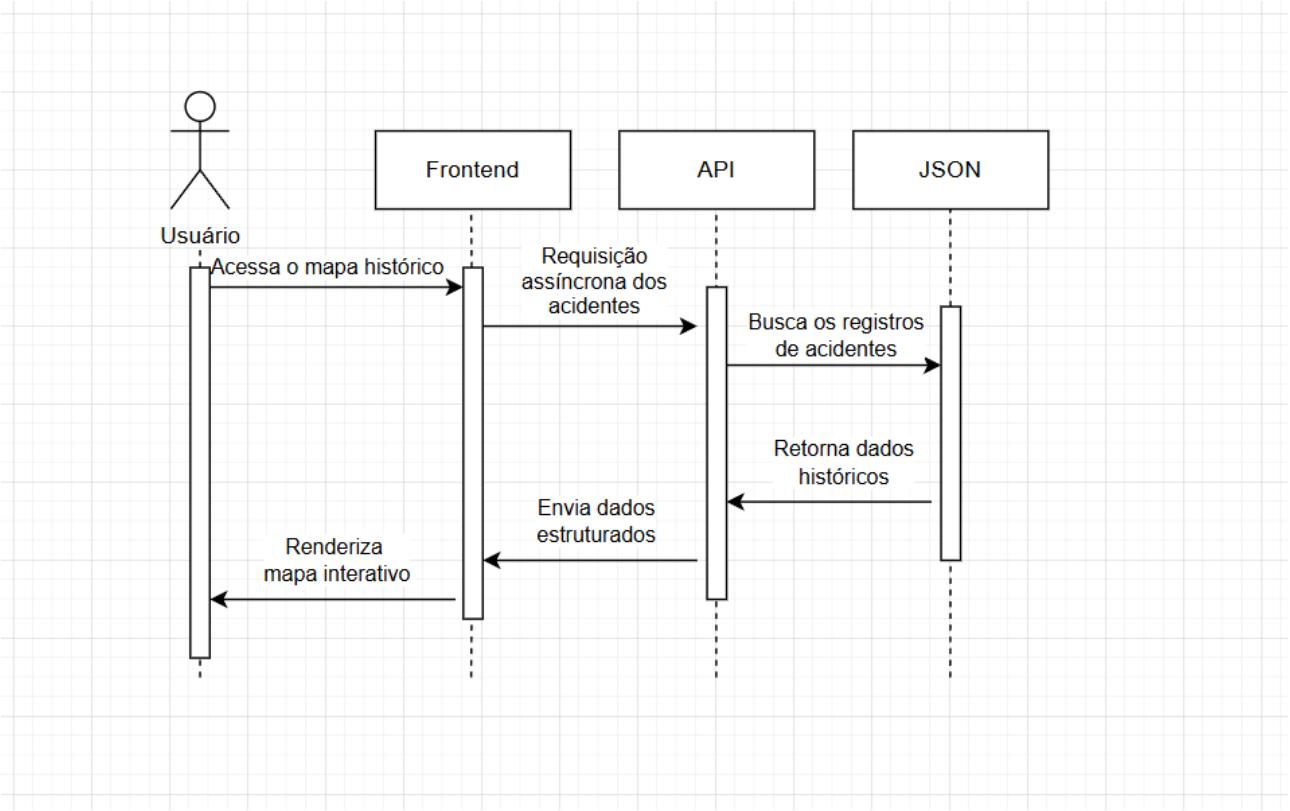
Na sequência, a dinâmica comportamental do *software* é especificada por meio de Diagramas de Sequência, ilustrando a ordem temporal das mensagens e chamadas entre os componentes do *backend* e *frontend* para cada funcionalidade central, representados nas Figuras 3,4 e 5.

Figura 3 – Diagrama de sequência: Emitir alerta



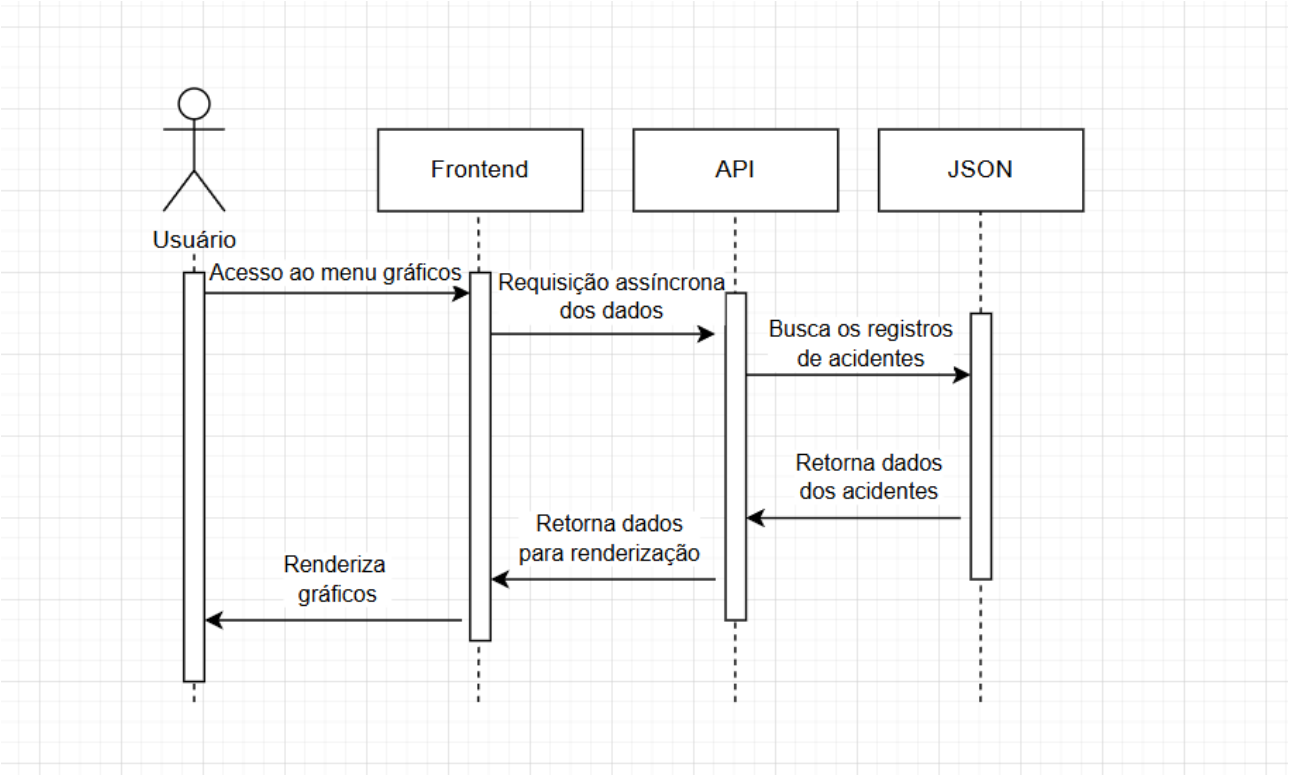
Fonte: Elaborada pelos autores.

Figura 4 – Diagrama de sequência: Consultar mapa interativo



Fonte: Elaborada pelos autores.

Figura 5 – Diagrama de sequência: Consultar página de gráficos

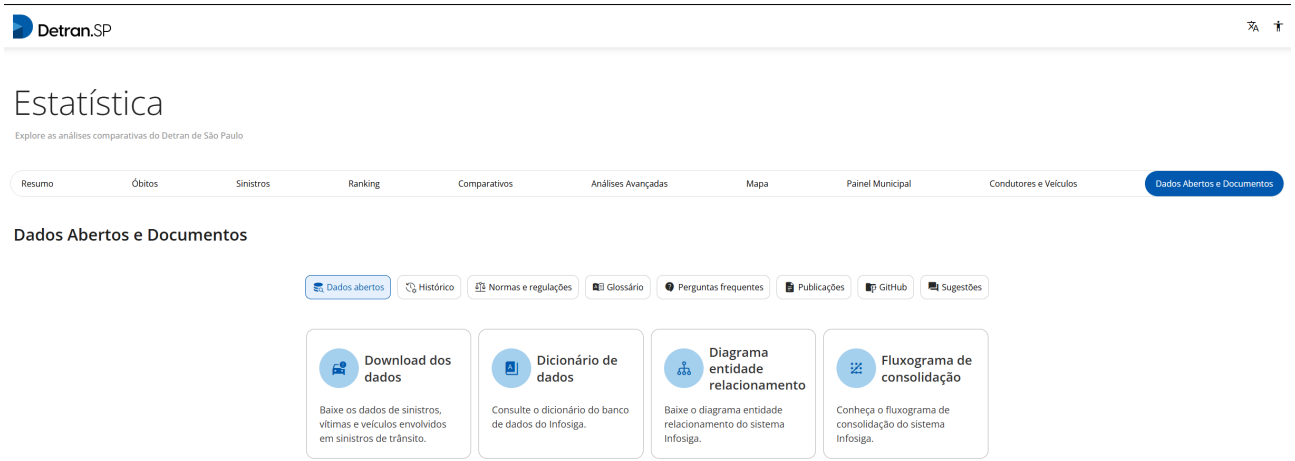


Fonte: Elaborada pelos autores.

4.1 Coleta de dados

O primeiro estágio da metodologia consiste no módulo de coleta de dados, uma etapa fundamental que estabelece a base empírica e quantitativa do modelo preditivo. Para o desenvolvimento deste projeto, foram utilizados três conjuntos de dados brutos provenientes do portal Infosiga-SP, abrangendo o período de 2022 a 2025. A escolha desta janela temporal justifica-se pela proximidade com o contexto atual da infraestrutura urbana de Bauru. Optou-se por priorizar dados mais recentes em detrimento de períodos mais longos (como a opção disponível de 2015 a 2021), sob o pressuposto de que correções de infraestrutura viária e mudanças nas dinâmicas urbanas ao longo do tempo poderiam tornar as causas de sinistros de períodos mais antigos menos representativas da realidade atual. Adicionalmente, a escolha desta fonte justifica-se pela sua alta granularidade, credibilidade oficial e pelo detalhamento georreferenciado das ocorrências. A Figura 6 ilustra o processo de extração dos dados brutos do Infosiga-SP.

Figura 6 – Portal do Infosiga-SP com dados históricos de sinistros.



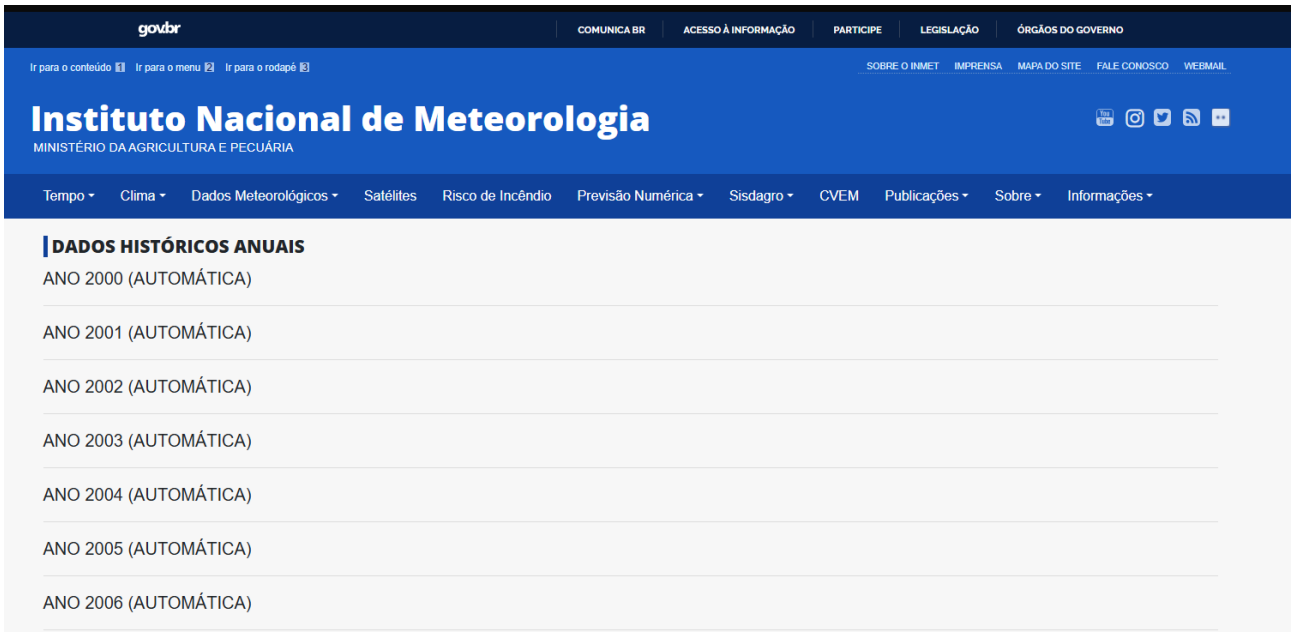
Fonte: https://infosiga.detran.sp.gov.br/#referencia.

Cada *dataset* possui uma função específica e se interliga pelo identificador único de sinistro (`id_sinistro`). O conjunto `sinistros_2022-2025` serve como base primária, fornecendo a data, o horário, a localização precisa (latitude e longitude) e a classificação da ocorrência, essenciais para a análise espacial e temporal das áreas de risco. O *dataset* `pessoas_2022-2025` enriquece essa base com variáveis demográficas, como sexo, idade e tipo de vítima, enquanto o `veiculos_2022-2025` complementa o quadro com características dos veículos envolvidos (ano de fabricação, cor, tipo). A utilização combinada destes arquivos é imperativa para a construção do conjunto completo de características do modelo, fornecendo tanto as coordenadas espaciais das amostras positivas (acidentes) quanto o contexto socioeconômico e veicular das ocorrências.

O segundo componente do módulo de coleta envolve a aquisição e integração de dados ambientais, essenciais para a investigação de fatores exógenos que podem influenciar a

sinistralidade viária. Para tal, foram utilizados documentos públicos disponibilizados no portal do INMET, extraindo-se os registros da estação meteorológica de Bauru (Estação A705) para o mesmo período de abrangência dos dados do Infosiga-SP (2022 a 2025). O portal para a obtenção dos dados é representado na Figura 7.

Figura 7 – Portal do INMET com dados históricos de clima.



Fonte: https://portal.inmet.gov.br/dadoshistoricos

Essa coleta horária de informações foi crucial para permitir o cruzamento temporal preciso das ocorrências de acidentes com as condições climáticas exatas do momento. O *dataset* do INMET fornece uma ampla gama de variáveis atmosféricas, dentre as quais se destacam a precipitação total horária e a umidade relativa do ar horária. Esta escolha foi metodologicamente orientada pela hipótese inicial de que existe uma correlação significativa entre a ocorrência de sinistros e a precipitação pluviométrica, um fator que comprovadamente afeta a aderência dos pneus e a visibilidade dos condutores, constituindo um atributo preditivo de alto valor. A integração desses dados de diferentes naturezas (geográfica e climática) na mesma base foi um passo fundamental para o enriquecimento do conjunto de dados e para a robustez da análise preditiva subsequente.

Em síntese, a coleta de dados desta seção garantiu a obtenção de uma base de informações robusta, multidimensional e cronologicamente alinhada, composta por três conjuntos de dados do Infosiga-SP (sinistros, pessoas e veículos) e um conjunto de dados climáticos do INMET. A combinação dessas fontes é crucial para a construção de características preditivas que atendam aos requisitos do modelo de *machine learning*. Contudo, a natureza distinta e a potencial heterogeneidade inerente aos dados brutos exigem um tratamento rigoroso e uma etapa de padronização antes da modelagem.

Dessa forma, a próxima etapa da metodologia, conforme detalhado na subseção 4.2,

será dedicada à limpeza e agrupamento dos dados, onde esses insumos serão processados, padronizados e organizados em uma estrutura unificada. Este passo é fundamental para permitir o levantamento de hipóteses preditivas e a engenharia de características e, somente então, preparar o *dataset* final para o treinamento e a validação do modelo.

4.2 Limpeza e agrupamento dos conjuntos de dados

Este módulo dedicou-se ao pré-processamento dos dados, um estágio crítico para assegurar a qualidade, consistência e interoperabilidade das informações brutas coletadas do Infosiga-SP e do INMET. Dada a heterogeneidade das fontes, foi estabelecido um pipeline rigoroso de higienização utilizando a biblioteca Pandas, focado em três eixos principais: filtragem espacial, tratamento de valores ausentes e sincronização temporal.

O processo foi iniciado com a delimitação geográfica dos dados. Para os *datasets* de "Sinistros"e "Pessoas", aplicou-se um filtro direto pelo município de "Bauru". Para o *dataset* de "Veículos", que originalmente não possui atributo de município, desenvolveu-se uma estratégia de filtragem relacional: manteve-se apenas os veículos cujos identificadores de sinistro (id_sinistro) possuíssem correspondência no conjunto já filtrado de acidentes locais, garantindo a integridade referencial da base.

No tratamento de qualidade dos dados, adotou-se uma política de padronização para valores nulos e inconsistentes. Em variáveis categóricas (como cor do veículo ou profissão da vítima), termos variados indicando ausência de informação (ex: "NAO INFORMADO", campos vazios) foram unificados sob o rótulo "NAO DISPONIVEL". Para variáveis numéricas críticas, utilizou-se a imputação de valores sentinela (como -1 para idade ou -9999 para coordenadas geográficas), permitindo que o modelo diferencie matematicamente a ausência de dado de um valor real. Adicionalmente, foi realizada uma normalização semântica no atributo de cores dos veículos, agrupando variações lexicais (ex: "Amarela"e "AMARELO") em categorias únicas.

Por fim, o processamento dos dados climáticos do INMET exigiu uma etapa adicional de engenharia de dados: a correção de fuso horário. Como os registros meteorológicos são disponibilizados em UTC (Tempo Universal Coordenado), foi realizada a conversão de todos os timestamps para o horário local de Bauru (UTC-3), assegurando que o cruzamento entre as condições de chuva e o momento exato do acidente fosse preciso.

As Tabelas 1, 2 e 3 resumem as principais operações de limpeza e transformação aplicadas a cada conjunto de dados para a constituição da base final.

Tabela 1 – Regras de filtragem espacial

Dataset	Filtragem Espacial
Sinistros	Município: "BAURU"
Pessoas	Filtro herdado de Sinistros
Veículos	*Inner Join* com Sinistros
Clima	Janela temporal: 2022–2025

Fonte: Elaborada pelos autores.

Tabela 2 – Regras de tratamento de nulos

Dataset	Tratamento de Nulos
Sinistros	• Coordenadas: `-9999` • Hora: `"99:99"` • Logradouro: `"S/N"` • Tipo veículo: `0` • Gravidade: `0` • Logradouro: "NAO DISPONIVEL" • Tipo Sinistro: `"N"`
Pessoas	• Idade: `-1` • Ano óbito: `-1` • Mês óbito: `-1` • Dia óbito: `-1` • Ano_mês_óbito: 1900-01-01 • Tipo vitima:"NAO DISPONIVEL" • Profissão: "NAO DISPONIVEL" • Tipo veiculo vitima: "NAO DISPONIVEL" • Data Óbito: `1900-01-01`
Veículos	• Ano Fabricação: `0` • Ano Modelo: `0` • Cor Veículo: "NAO DISPONIVEL."
Clima	• Precipitação: `0.0`

Fonte: Elaborada pelos autores.

Tabela 3 – Regras de padronização

Dataset	Padronização
Sinistros	• "NAO INFORMADO"→ "NAO DISPONIVEL"
Pessoas	• "NAO INFORMADO"→ "NAO DISPONIVEL" • Renomeação de coluna: tipo_de_vítima → tipo_vitima
Veículos	• "NAO INFORMADO"→ "NAO DISPONIVEL" • Normalização de Cores: Unificação semântica
Clima	• Fuso Horário: Subtração de 3h (UTC → UTC-3)

Fonte: Elaborada pelos autores.

Destaca-se a estratégia de filtragem dos veículos, que, por não possuírem atributo de município, foram selecionados via cruzamento relacional (Inner Join) com os sinistros já filtrados. Outro ponto crítico foi a correção do fuso horário nos dados climáticos (de UTC para o horário local), eliminando o risco de associar um acidente a uma condição meteorológica futura ou passada.

Após estas transformações, os quatro conjuntos foram consolidados em uma estrutura tabular única, exportada em formato CSV, servindo de base confiável para a Análise Exploratória e o treinamento dos modelos.

4.3 Análise exploratória de dados

Após a conclusão das etapas de coleta (4.1) e pré-processamento (4.2), os conjuntos de dados do Infosiga-SP e do INMET , agora limpos e unificados, foram submetidos a uma análise exploratória dos dados. O objetivo desta etapa é investigar os dados de sinistros de trânsito em Bauru, no período de 2022 a 2025, para identificar padrões, tendências, correlações e anomalias.

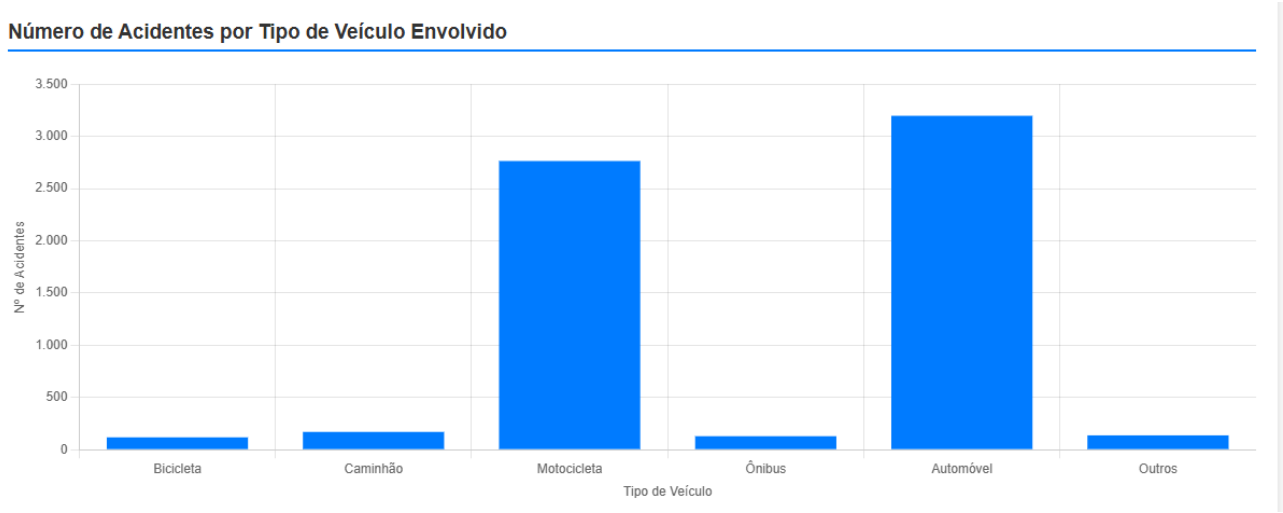
Os *insights* visuais gerados nesta seção são fundamentais para a formulação de hipóteses sobre as causas e dinâmicas dos acidentes. Essas hipóteses, por sua vez, justificarão a seleção das características (variáveis preditivas) para o treinamento e a comparação dos modelos de *machine learning* (seção 4.4). As visualizações de dados discutidas a seguir foram implementadas na interface do sistema, conforme detalhado no item 4.8.2.

4.3.1 Análise de fatores de risco e contexto viário

A primeira análise buscou compreender o contexto onde os acidentes ocorrem, focando

nos tipos de veículos envolvidos. A Figura 8 demonstra a distribuição de sinistros por tipo de veículo. Fica evidente que a vasta maioria dos acidentes envolve "Automóvel"e "Motocicleta". Este dado não apenas quantifica o problema, mas também alinha-se com a problemática descrita no Capítulo 3, que destaca o aumento de acidentes com motociclistas em Bauru. A alta frequência desses dois modais os qualifica como variáveis de alta relevância para a previsão de risco.

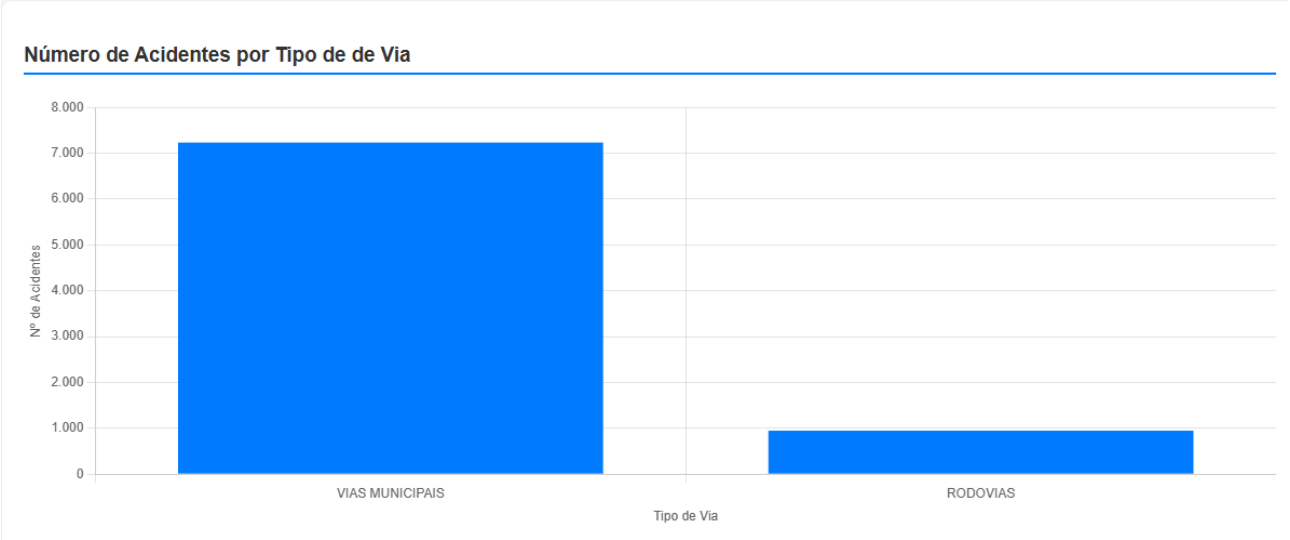
Figura 8 – Acidentes por tipo de veículo



Fonte: Elaborada pelos autores.

De forma complementar, a Figura 9 segmenta os acidentes por tipo de via. Observa-se uma concentração esmagadora de ocorrências em "VIAS MUNICIPAIS", em detrimento de "RODOVIAS". Esta descoberta é crucial, pois valida o escopo do projeto como um problema intrinsecamente urbano e justifica a utilização da malha viária municipal como base para a geração de amostras negativas.

Figura 9 – Acidentes por tipo de via

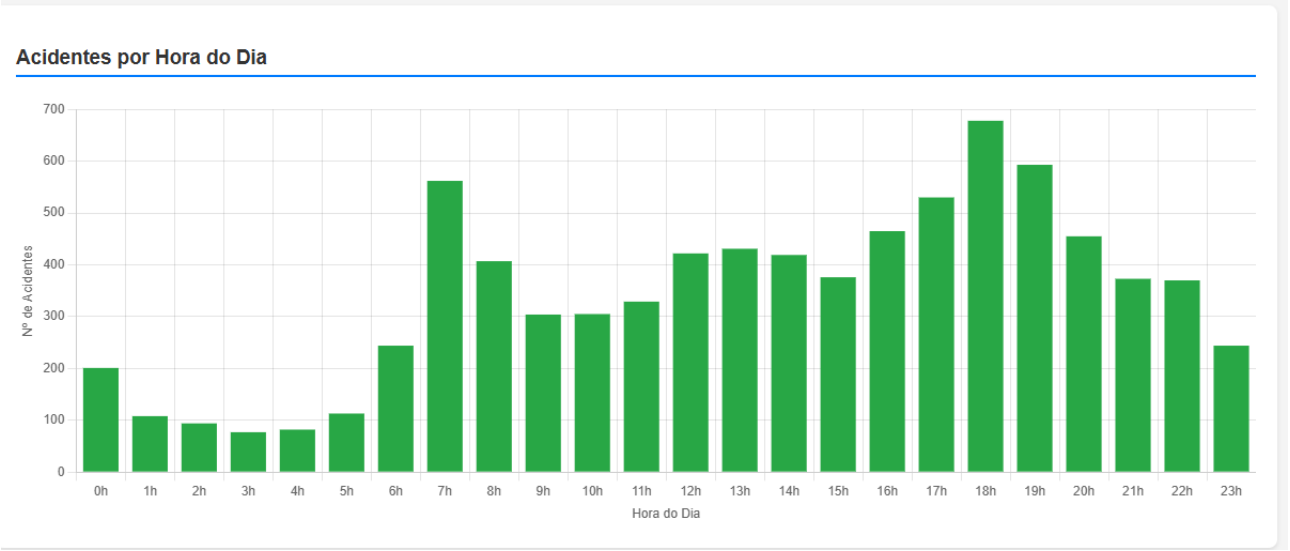


Fonte: Elaborada pelos autores.

4.3.2 Análise temporal dos sinistros

A análise temporal é vital para entender quando os riscos aumentam. A Figura 10 plota a contagem de acidentes por hora do dia.

Figura 10 – Acidentes por hora do dia



Fonte: Elaborada pelos autores.

O gráfico revela padrões não aleatórios, mas sim fortemente correlacionados com a atividade urbana. Identificam-se três picos principais:

- **Pico da manhã:** Um aumento notável entre 07:00 e 08:00, coincidente com o início do horário comercial e escolar.

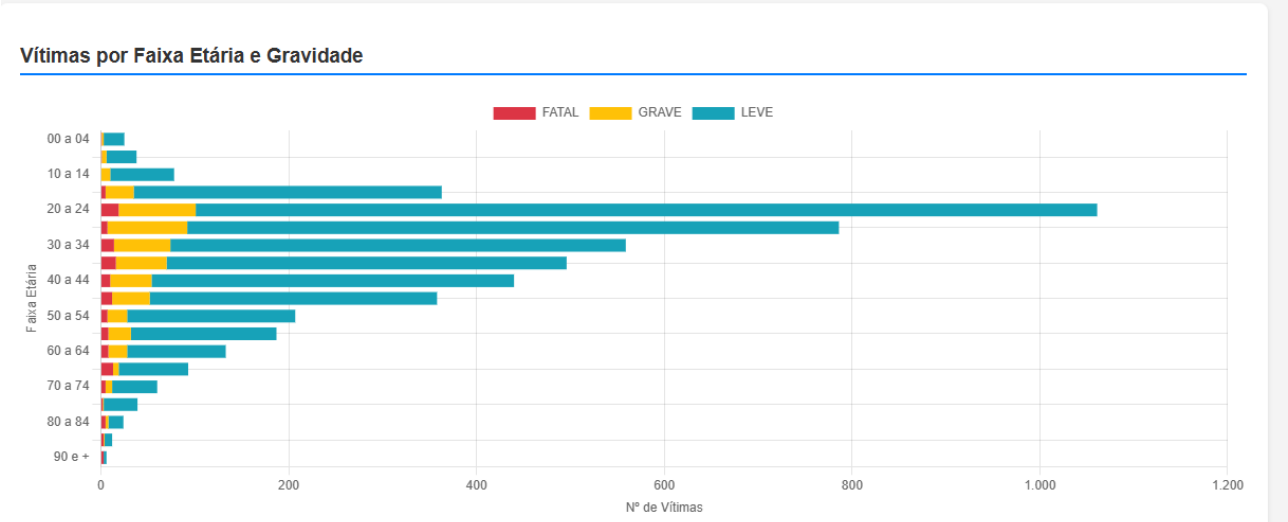
- **Pico do almoço:** Uma elevação menor, porém clara, por volta das 12:00 e 13:00.
- **Pico da noite:** O período mais crítico do dia, entre 17:00 e 19:00, que corresponde ao principal horário de retorno do trabalho e universidades.

Essa distribuição bimodal (manhã e noite) reforça que a variável hora é um preditor fundamental, sugerindo que o risco de acidente é dinâmico e dependente do fluxo de tráfego.

4.3.3 Análise demográfica e de gravidade

Compreender o perfil das vítimas é essencial para dimensionar o impacto social do problema, alinhando-se aos objetivos de saúde pública do projeto. A Figura 11 demonstra que, embora os acidentes ocorram em todas as idades, há uma concentração significativa de vítimas nas faixas de "20 a 24"e "30 a 34"anos. Isso indica que a população economicamente ativa e jovem é a mais vulnerável. A análise de gravidade (Fatal, Grave, Leve) sobreposta a essa distribuição permite a futuros gestores públicos direcionar campanhas de conscientização para os grupos de maior risco.

Figura 11 – Vítimas por faixa etária e gravidade

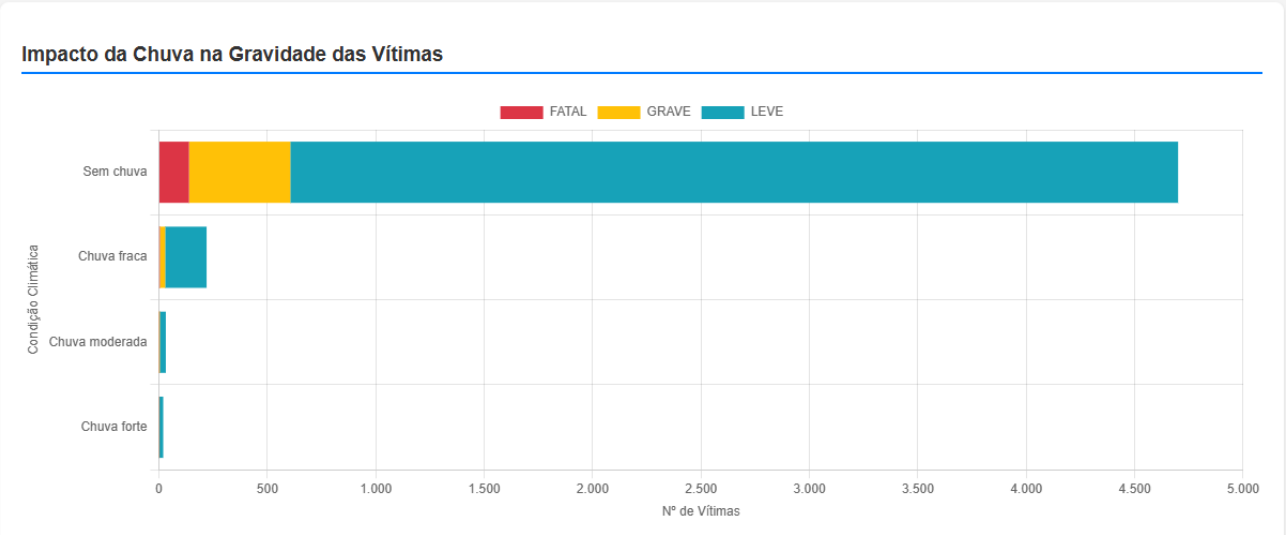


Fonte: Elaborada pelos autores.

4.3.4 Análise de fatores ambientais

A Figura 12 apresenta a distribuição absoluta dos sinistros por condição climática. À primeira vista, observa-se uma predominância massiva de ocorrências em dias "Sem chuva", o que é estatisticamente esperado, dado que a precipitação é um evento intermitente e menos frequente na série temporal.

Figura 12 – Impacto da chuva na gravidade dos acidentes



Fonte: Elaborada pelos autores.

Contudo, a análise crítica deve focar na proporcionalidade. Embora o volume total de acidentes com chuva seja menor, a literatura indica que o risco relativo por quilômetro rodado aumenta sob precipitação. O gráfico revela que, quando ocorrem as condições de "Chuva Fraca"ou "Moderada", a incidência de vítimas não é desprezível. A hipótese sustentada é que a chuva atua como um catalisador de risco: ela reduz a aderência e a visibilidade, transformando erros que seriam leves em pista seca em sinistros de maior gravidade.

4.3.5 Conclusão da AED e levantamento de hipóteses

A Análise Exploratória de Dados permitiu extrair *insights* valiosos e transformar dados brutos em informação acionável. Com base nos padrões observados, foram formuladas as seguintes hipóteses centrais que nortearão a modelagem preditiva:

- **Hipótese 1 (Temporal):** O risco de acidente não é estático, mas flutua ao longo do dia, atingindo seu pico nos horários de rush (7h-8h e 17h-19h);
- **Hipótese 2 (Contextual):** O tipo de via e o tipo de veículo são preditores significativos da ocorrência de sinistros;
- **Hipótese 3 (Ambiental):**A presença de precipitação, conforme medido pelo INMET, tem correlação positiva com a ocorrência e a gravidade dos acidentes; e
- **Hipótese 4 (Demográfica):** Existe um perfil demográfico de maior risco, concentrado em adultos jovens (20-34 anos).

Confirmadas estas hipóteses e identificadas as variáveis-chave, o projeto avança para a etapa de desenvolvimento do modelo (4.5), onde estas *features* serão utilizadas para

treinar e comparar diferentes algoritmos de *machine learning* capazes de prever o risco de acidente em tempo real.

4.4 Comparação entre modelos preditivos

Após a análise dos dados no capítulo 4.3, que identificou as variáveis (*features*) de maior relevância, esta seção detalha os algoritmos de *machine learning* selecionados para a tarefa central do projeto: estimar a probabilidade de risco de acidente.

O problema é formulado como uma abordagem probabilística (ou uma classificação focada na probabilidade). O objetivo não é prever se um acidente vai ocorrer (Classe 1) ou não vai ocorrer (Classe 0) de forma binária, mas sim gerar um *score* de risco contínuo (entre 0 e 1) para um determinado conjunto de condições (local, hora, clima, etc.). Este score é a própria "análise de risco"que o sistema fornecerá.

Para isso, selecionamos algoritmos de classificação que são capazes de gerar saídas probabilísticas, permitindo uma comparação de diferentes abordagens.

4.4.1 Random Forest

O Random Forest, proposto por Leo Breiman (2001), constitui um método de *ensemble learning* baseado na técnica de *Bootstrap Aggregating (Bagging)*, que visa aprimorar a precisão e a estabilidade das estimativas preditivas. Sua arquitetura compreende uma coleção de múltiplas árvores de decisão, cada uma treinada em uma amostra *bootstrap* dos dados e utilizando apenas um subconjunto aleatório das variáveis em cada ponto de divisão. Esta dupla aleatorização é crucial, pois descorrelaciona as árvores individuais, prevenindo que um atributo dominante monopolize todas as predições. Para a formulação probabilística do problema, a estimativa de risco para uma nova observação não é obtida por uma classificação majoritária simples, mas sim pela agregação ponderada dos votos de todas as árvores. Dessa forma, o score gerado representa a fração de árvores na floresta que categorizaram a entrada como o evento de interesse (acidente).

Para a construção de cada árvore de decisão e a determinação da melhor divisão (*split*) em cada nó, o algoritmo utiliza o critério de Impureza de Gini. Dado um nó t com amostras de C classes, a impureza é calculada conforme a Equação 1:

$$Gini(t) = 1 - \sum_{i=1}^{C} p(i|t)^2 \tag{1}$$

onde $p(i|t)$ representa a proporção de amostras da classe i no nó t. O objetivo do algoritmo durante o treinamento é minimizar essa impureza a cada divisão.

A robustez do Random Forest, aliada à sua capacidade de reduzir a variância sem um aumento significativo no bias (BREIMAN, 2001), justifica sua seleção como um algoritmo

eficiente e confiável para mapear as condições contextuais a uma medida de incerteza contínua.

4.4.2 XGBoost

O E*xtreme Gradient Boosting (XGBoost)* é uma implementação altamente otimizada e escalável do *Gradient Boosting* (CHEN & GUESTRIN, 2016). Seu poder reside na construção sequencial de árvores de decisão, onde cada nova árvore é treinada para corrigir os erros residuais (gradientes) da composição anterior.

Matematicamente, o modelo XGBoost é construído de forma aditiva. A predição final para uma entrada x_i na iteração t é dada pela soma das predições das árvores anteriores mais a nova árvore f_t, conforme a Equação 2:

$$\hat{y}_i^{(t)} = \hat{y}_i^{(t-1)} + f_t(x_i) \tag{2}$$

Para garantir a generalização e evitar o sobreajuste (*overfitting*), o algoritmo minimiza uma função objetivo regularizada. Esta função, apresentada na Equação 3, combina a função de perda l (que mede a diferença entre o valor real y_i e o predito $\hat{y}_i$) com um termo de penalização Ω:

$$Obj(\Theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k) \tag{3}$$

em que $\Omega(f_k) = \gamma T + \frac{1}{2}\lambda||w||^2$ controla a complexidade da árvore, penalizando o número de folhas T e a magnitude dos pesos w das folhas.

Esta abordagem aditiva permite que o modelo aprimore iterativamente sua performance, minimizando uma função de perda regularizada que controla a complexidade do modelo, mitigando o sobreajuste (*overfitting*). Na estimativa de risco, o XGBoost calcula, para cada instância, um valor acumulado na escala logarítmica , que é o *output* não transformado. Este valor é posteriormente mapeado para o intervalo [0, 1] por meio da função logística, traduzindo a escala em uma medida de verossimilhança. Sua inclusão é justificada pela reconhecida alta performance na calibração de scores e pela eficiência no processamento de dados tabulares e esparsos, tornando-o um forte candidato para o modelo preditivo final.

4.4.3 Multilayer Perceptron (MLP)

O *Multilayer Perceptron*, uma classe fundamental de redes neurais artificiais, é especialmente desenhado para a tarefa de estimativa de risco devido à sua capacidade de modelar relações não-lineares complexas entre os atributos (HAYKIN, 2009). Sua arquitetura, que tipicamente consiste em uma camada de entrada, uma ou mais camadas ocultas e uma camada de saída, executa transformações matriciais e ativações não-lineares sucessivas.

Para problemas de estimação de risco, a rede é configurada com um único neurônio na camada de saída que emprega a função logística como função de ativação.

Matematicamente, a função sigmoide mapeia o valor agregado proveniente das camadas anteriores para um intervalo estritamente entre 0 e 1. Este valor de saída é interpretado como a verossimilhança ou score de risco, sendo uma representação nativa e direta da probabilidade condicional de ocorrência do evento-alvo, conforme aprendido e generalizado pela rede.

Para garantir que a saída do modelo seja uma probabilidade válida no intervalo $[0, 1]$, a camada final da rede aplica a função de ativação Sigmoide (ou Logística), definida matematicamente pela Equação 4:

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4}$$

onde z representa a soma ponderada das entradas da última camada oculta acrescida do viés (*bias*). Esta função mapeia qualquer valor real para o intervalo probabilístico desejado para a estimativa de risco.

4.4.4 Definição das métricas de avaliação

A avaliação do desempenho dos modelos classificadores fundamenta-se na Matriz de Confusão, uma estrutura tabular que confronta as classes preditas pelo modelo com as classes reais do conjunto de dados. Os quatro resultados elementares derivados dessa matriz são: Verdadeiros Positivos (VP), Falsos Positivos (FP), Falsos Negativos (FN) e Verdadeiros Negativos (VN).

Com base nestes componentes, definem-se as métricas primárias de *Precision* e *Recall*. A *Precision* avalia a qualidade das predições positivas, enquanto o *Recall* mede a capacidade do modelo de cobrir o total de eventos de interesse. As definições matemáticas são apresentadas nas Equações 5 e 6:

$$Precision = \frac{VP}{VP + FP} \tag{5}$$

$$Recall = \frac{VP}{VP + FN} \tag{6}$$

Para obter uma visão unificada do desempenho que pondere o equilíbrio entre a precisão e a cobertura, utiliza-se o *F1-Score*, definido como a média harmônica entre as duas métricas anteriores, conforme a Equação 7:

$$F1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{7}$$

4.4.4.1 Justificativa Matemática da Escolha da Métrica PR-AUC

A seleção da métrica global de avaliação é crítica em cenários de aprendizado supervisionado com classes severamente desbalanceadas, como é o caso da previsão de acidentes de trânsito. Embora a Área sob a Curva ROC (ROC-AUC) seja amplamente utilizada na literatura, ela apresenta limitações matemáticas quando a classe negativa (não-eventos) é majoritária, o que pode conduzir a interpretações otimistas enganosas sobre a eficácia do modelo.

A curva ROC relaciona a Taxa de Verdadeiros Positivos (TPR) com a Taxa de Falsos Positivos (FPR). A definição formal da FPR é apresentada na Equação 8:

$$FPR = \frac{FP}{FP + VN} \tag{8}$$

No contexto de risco viário urbano, o número de Verdadeiros Negativos (VN), veículos que trafegam sem sofrer acidentes, é ordens de magnitude superior ao número de acidentes. Observando o denominador da Equação 8, nota-se que um valor de VN excessivamente grande dilui o impacto dos Falsos Positivos (FP). Isso resulta em um valor de FPR artificialmente baixo, mesmo quando o modelo gera um número elevado de falsos alarmes, inflacionando a métrica ROC-AUC e mascarando deficiências operacionais do sistema.

Em contraste, a métrica PR-AUC (*Area Under the Precision-Recall Curve*) avalia a relação entre a *Precision* e *Recall*. Retomando a fórmula da *Precision* (Equação 5), observa-se que o termo VN não participa de sua composição.

Dessa forma, a métrica PR-AUC penaliza diretamente o modelo pelo aumento de Falsos Positivos, independentemente do volume da classe majoritária. Para o presente trabalho a PR-AUC constitui o estimador mais rigoroso da confiabilidade do sistema, garantindo que um alto *score* de avaliação reflita, de fato, a capacidade de identificar riscos reais sem gerar fadiga de alarmes no usuário (DAVIS; GOADRICH, 2006; SAITO; REHMSMEIER, 2015).

4.5 Desenvolvimento do modelo

Esta seção detalha o processo prático de como foram construídos e otimizados. O desenvolvimento do modelo foi dividido em duas etapas críticas: primeiro, a criação de um conjunto de dados de classificação viável através da amostragem negativa e, segundo, a otimização dos hiperparâmetros de cada modelo por meio do *GridSearchCV*.

4.5.1 Amostragem negativa

Um desafio fundamental deste projeto reside na natureza dos dados de origem (Infosiga-SP), que constituem um conjunto de dados inerentemente positivos (acidentes), registrando apenas onde e quando os eventos ocorreram. Esta configuração se enquadra no paradigma

de aprendizado de máquina conhecido como *Positive-Unlabeled Learning*, amplamente estudado em cenários de detecção de eventos raros. Para que um modelo probabilístico possa aprender a estabelecer fronteiras de decisão e diferenciar um local de alto risco de um local seguro, é mandatório que ele seja exposto a exemplos de ambos os estados.

Como os dados negativos (representando tráfego normal sem sinistros) não existem nativamente, foi indispensável criá-los artificialmente por meio de uma estratégia de amostragem negativa, conforme prática estabelecida na literatura para modelagem de risco espacial e previsão de acidentes com dados desbalanceados (ESMALIFARD et al., 2020).

O processo de construção do *dataset* negativo foi executado da seguinte forma, visando mitigar o viés de seleção inerente ao método:

- **Geração de coordenadas aleatórias:** Foram gerados milhares de pontos de coordenadas (latitude/longitude) aleatórios, porém espacialmente restritos aos segmentos de rua válidos do mapa de Bauru, obtidos a partir da plataforma *OpenStreetMap*. Essa restrição garante que os pontos gerados representem locais de tráfego plausível, e não áreas inacessíveis.

- **Tipos de amostras**: Para aumentar a robustez do treinamento, foram gerados dois tipos de amostras negativas:

    1. Amostras aleatórias: Aplicadas aos pontos de coordenadas aleatórias, consistiram em atribuir um carimbo de data, hora e tipo de veículo aleatórios, mantendo-se dentro do período de análise e seguindo a proporção de veículos observada no conjunto de dados de acidentes.
    2. Amostras de contraste temporal: Foram utilizadas as coordenadas exatas dos acidentes reais, porém combinadas com horários, tipo de veículo aleatório e dias diferentes daqueles em que os eventos ocorreram. A premissa metodológica é que, se em um determinado local um evento não foi registrado com essas *features*, essa combinação representa com alta probabilidade um ponto de não-acidente para fins de treinamento.

- **Enriquecimento de features:** Todos os pontos negativos sintéticos foram então submetidos ao mesmo processo de enriquecimento e *feature engineering* dos acidentes reais, cruzando-os com as fontes de dados meteorológicos, garantindo a consistência.

4.5.1.1 Análise de sensibilidade da amostragem e escolha da proporção

É crucial notar que a proporção de amostragem negativa (eventos positivos:eventos negativos) é tratada como um hiperparâmetro de treinamento, e não como uma tentativa de replicar a proporção real de acidentes e não-acidentes no mundo real. Uma proporção

realista (que, em termos de volume de viagens, poderia exceder 1:40.000) seria computacionalmente inviável e metodologicamente falha, visto que o desbalanceamento extremo forçaria o modelo a prever a classe majoritária ('não-acidente') em quase 100% dos casos, resultando em um *Recall* nulo para a classe de interesse (acidente) (CHAWLA, 2005). Este desafio é amplamente reconhecido na literatura de modelagem de risco espacial e previsão de acidentes de trânsito que lidam com dados desbalanceados (YE et al., 2023).

É importante ressaltar que a alteração da distribuição natural das classes introduz um viés intencional nas probabilidades estimadas pelo modelo. Consequentemente, os scores de saída deixam de representar a probabilidade absoluta de um acidente ocorrer e passam a funcionar como índices de risco relativo.

O objetivo desta etapa, portanto, é encontrar um ponto de equilíbrio (*trade-off*) que forneça ao modelo exemplos negativos suficientes para aprender a discriminar os padrões de risco, garantindo a usabilidade na produção, onde o custo de um falso positivo (alerta desnecessário) deve ser minimizado. A proporção ideal é aquela que maximiza a capacidade de generalização do modelo em cenários desbalanceados. A escolha da proporção final é, portanto, justificada pela performance do modelo na métrica de avaliação PR-AUC, que é a métrica mais indicada para problemas com desbalanceamento severo, explicada na Subseção 4.4.4.1.

Para a seleção, foi utilizado o *XGBoost* como um modelo de *benchmark*, por ser um algoritmo de alta performance para dados tabulares, para avaliar como a performance do modelo se comportava em *datasets* com níveis crescentes de desbalanceamento. A Tabela 4 mostra as diferentes performances do PR-AUC com diferentes proporções de dados.

Tabela 4 – Comparação de performance PR-AUC

Proporção	**PR-AUC**
1:2	0.8654
1:4	0.7887
1:10	0.7161
1:20	0.6676
1:50	0.6233
1:20	0.6676
1:100	0.8623
1:200	0.8558

Fonte: Elaborada pelos autores.

A análise de sensibilidade da Tabela 4 revela uma complexa relação entre o nível de desbalanceamento e a capacidade discriminatória do modelo. A proporção $1:2$ (um evento positivo para dois negativos) apresentou o PR-AUC numericamente mais elevado (0.8654). Contudo, essa proporção reflete uma divergência extrema do equilíbrio de classes real, sendo metodologicamente insustentável para um cenário de produção. Um modelo treinado

em 1 : 2 estaria enviesado a prever a classe positiva com alta frequência, resultando em uma taxa de falsos positivos inaceitavelmente alta no mundo real, onde a proporção de não-acidentes é amplamente superior.

O objetivo da amostragem não é maximizar a métrica em um ambiente artificialmente balanceado, mas sim garantir a generalização robusta no ambiente de produção. Neste sentido, a proporção 1 : 100 obteve o melhor resultado em termos de equilíbrio metodológico e performance preditiva. A variação de apenas 0.0031 no PR-AUC (0.8654 em 1 : 2 para 0.8623 em 1 : 100) é considerada estatisticamente irrelevante e compatível com ruído. O fato de o modelo, ao ser exposto a um cenário 50 vezes mais desbalanceado (1 : 100 vs. 1 : 2), manter um poder preditivo praticamente idêntico ao cenário mais fácil comprova sua robustez e sua capacidade de aprender os padrões preditivos intrínsecos (horário, clima, composição veicular) sem estar viciado na proporção do dataset de treinamento.

Finalmente, embora a proporção 1 : 200 também demonstre alta performance (0.8558), o modelo 1 : 100 já havia validado o ponto principal de generalização. A adoção de uma proporção de 1 : 200 implicaria um aumento desnecessário no custo computacional e no tempo de treinamento, sem oferecer ganho marginal de performance que justificasse o investimento de recursos. Portanto, o modelo final foi ajustado para a proporção 1 : 100 por representar o ponto ótimo de trade-off entre viabilidade computacional e desempenho preditivo em um contexto realista.

4.5.2 Refinamento e escolha dos parâmetros com *GridSearchCV*

O desempenho dos algoritmos de *machine learning* depende diretamente da calibração de seus hiperparâmetros. Para identificar a configuração ideal para cada modelo, adotou-se a técnica de busca exaustiva em grade, conhecida como *GridSearchCV*.

Para assegurar a robustez estatística dessa otimização, o processo foi acoplado à validação cruzada com cv = 3. Nesse método, o conjunto de dados de treinamento é particionado em três subconjuntos distintos. Cada combinação é então treinada e avaliada três vezes, utilizando-se a média das pontuações para determinar a melhor configuração.

Conforme estabelecido na subseção 4.4.4, a métrica escolhida para orientar a seleção dos melhores modelos foi a *Average Precision* (que maximiza a área sob a curva *Precision-Recall*, ou PR-AUC). Essa escolha é fundamental dada a natureza desbalanceada do problema, onde a prioridade é a correta identificação da classe minoritária (acidentes).

A seguir, são detalhados os espaços de busca explorados e os parâmetros finais selecionados para cada um dos três algoritmos avaliados: *Random Forest*, *XGBoost* e *Multilayer Perceptron* (MLP).

4.5.2.1 Otimização do Random Forest

Para o *Random Forest*, a busca concentrou-se no equilíbrio entre a complexidade das árvores e a capacidade de generalização do *ensemble*. A Tabela 5 apresenta os valores testados.

Tabela 5 – Espaço de busca e valores explorados para o Random Forest

Hiperparâmetro	**Valores explorados**
n_estimators	[200, 300, 400]
max_depth	[10, 20, None]
min_samples_leaf	[2, 4, 6]
min_samples_split	[5, 10, 15]
class_weight	['balanced']

Fonte: Elaborada pelos autores.

Os hiperparâmetros que apresentaram a melhor performance média estão descritos na Tabela 6, acompanhados de sua justificativa técnica.

Tabela 6 – Hiperparâmetros otimizados do Random Forest

Hiperparâmetro	**Escolha**	**Explicação**
n_estimators	400	**Quantidade de árvores**. O valor alto garante estabilidade estatística por votação majoritária, reduzindo a variância das predições.
max_depth	None	**Limite de crescimento vertical**. "None"deixa a árvore crescer livremente até as folhas puras, essencial para capturar padrões complexos.
min_samples_leaf	2	**Mínimo de dados por folha**. Exigir 2 evita a criação de regras para acidentes únicos, forçando a generalização por pares.
min_samples_split	5	**Mínimo de dados para divisão**. Impede a fragmentação excessiva em nós intermediários, evitando overfitting em pequenos grupos.
class_weight	'balanced'	**Peso das classes**. Aumenta automaticamente a importância do erro na classe minoritária (acidentes) para corrigir o desbalanceamento.

Fonte: Elaborada pelos autores.

4.5.2.2 Otimização do XGBoost

No caso do *XGBoost*, a estratégia focou no controle da taxa de aprendizado e no balanceamento de pesos para lidar com a classe positiva. A Tabela 7 detalha o espaço de busca.

Tabela 7 – Espaço de busca e valores explorados para o XGBoost

Hiperparâmetro	**Valores explorados**
learning_rate	[0.1, 0.05]
scale_pos_weight	[100]
max_depth	[5, 7]
subsample	[0.8, 1.0]
colsample_bytree	[0.8, 1.0]
gamma	[0.5, 1]

Fonte: Elaborada pelos autores.

A configuração final, que maximizou a métrica de avaliação, é apresentada na Tabela 8.

Tabela 8 – Hiperparâmetros otimizados do XGBoost

Hiperparâmetro	**Escolha**	**Explicação**
learning_rate	0.05	**Velocidade do aprendizado**. Uma taxa baixa (0.05) evita pular o ótimo global, exigindo mais passos (árvores) para um ajuste fino e preciso.
max_depth	5	**Complexidade da árvore**. Limitada a 5 para evitar que o modelo decore os dados de treino, mantendo a capacidade de generalização.
scale_pos_weight	100	**Peso da classe positiva**. Compensa a proporção 1:100 dos dados, penalizando erros em acidentes 100x mais para priorizar o Recall.
n_estimators	400	**Número de iterações**. Quantidade necessária para compensar a taxa de aprendizado lenta e convergir para o erro mínimo.
subsample	1.0	**Amostras por árvore**. Uso de 100% das linhas disponíveis para maximizar o aprendizado, já que o dataset de treino é controlado.
colsample_bytree	0.8	**Colunas por árvore**. Seleciona 80% das features aleatoriamente, forçando árvores diferentes a aprenderem caminhos distintos (diversidade).
gamma	1	**Redução mínima de perda**. Valor conservador que impede a criação de novos nós se o ganho de informação não for significativo.

Fonte: Elaborada pelos autores.

4.5.2.3 Otimização do Multilayer Perceptron (MLP)

Para a rede neural (MLP), a busca priorizou a definição da arquitetura da rede (camadas e neurônios) e a função de ativação. A Tabela 9 mostra os parâmetros explorados.

Tabela 9 – Espaço de busca e valores explorados para o MLP

Hiperparâmetro	**Valores explorados**
hidden_layer_sizes	[(50,), (100,), (50, 50), (100, 50)]
activation	['relu', 'tanh', 'logistic', 'identity']
alpha	[0.0001, 0.001, 0.01]
learning_rate_init	[0.001, 0.01]

Fonte: Elaborada pelos autores.

A Tabela 10 detalha a arquitetura final selecionada pelo processo de otimização.

Tabela 10 – Hiperparâmetros otimizados do MLP

Hiperparâmetro	**Escolha**	**Explicação**
hidden_layer_sizes	(50,)	**Estrutura da rede**. Uma única camada com 50 neurônios provou-se suficiente para mapear a não-linearidade sem custo computacional excessivo.
activation	'logistic'	**Função de disparo**. A sigmoide ('logistic') foi ideal por gerar saídas naturalmente probabilísticas no intervalo [0, 1].
alpha	0.01	**Regularização L2**. Penalidade aplicada a pesos grandes. O valor 0.01 restringe a magnitude dos pesos para evitar overfitting.
learning_rate_init	0.01	**Passo inicial do otimizador**. Valor ligeiramente maior para acelerar a descida do gradiente no início do treinamento.
early_stopping	True	**Parada automática**. Interrompe o treino se a validação parar de melhorar, economizando recursos e evitando o superajuste.

Fonte: Elaborada pelos autores.

4.6 Análise das métricas do modelo

Após o processo de desenvolvimento e otimização dos três modelos candidatos (*Random Forest, XGBoost* e *MLP*), esta seção apresenta a análise comparativa de suas métricas de desempenho. O objetivo é avaliar objetivamente os resultados obtidos no conjunto de dados de teste, selecionar o modelo final mais performático para a tarefa de estimativa de risco e analisar seu comportamento prático para a implementação no sistema de alertas.

4.6.1 Comparação de desempenho no conjunto de teste

Para garantir uma avaliação imparcial, os três modelos, já otimizados com os hiperparâmetros encontrados na subseção 4.5.2, foram avaliados em um conjunto de dados de teste que não foi utilizado durante o treinamento ou a validação cruzada.

O desempenho foi medido utilizando as métricas de avaliação de ranking probabilístico definidas na subseção 4.4.4, com foco principal na PR-AUC (DAVIS, 2006), dada a natureza desbalanceada do problema, assim como as métricas de *Precision, Recall* e *F1-Score* para a classe positiva.

A Tabela 11 mostra os resultados das métricas para visualização e comparação entre os modelos.

Tabela 11 – Comparação de métricas dos modelos no conjunto de teste

Modelo	**PR-AUC**	**ROC-AUC**	**Precision**	**Recall**	**F1-score**
Random Forest	0.8606	0.9661	0.98	0.81	0.89
XGBoost	0.8532	0.9655	0.37	0.85	0.52
MLP	0.8280	0.8537	0.82	0.63	0.71

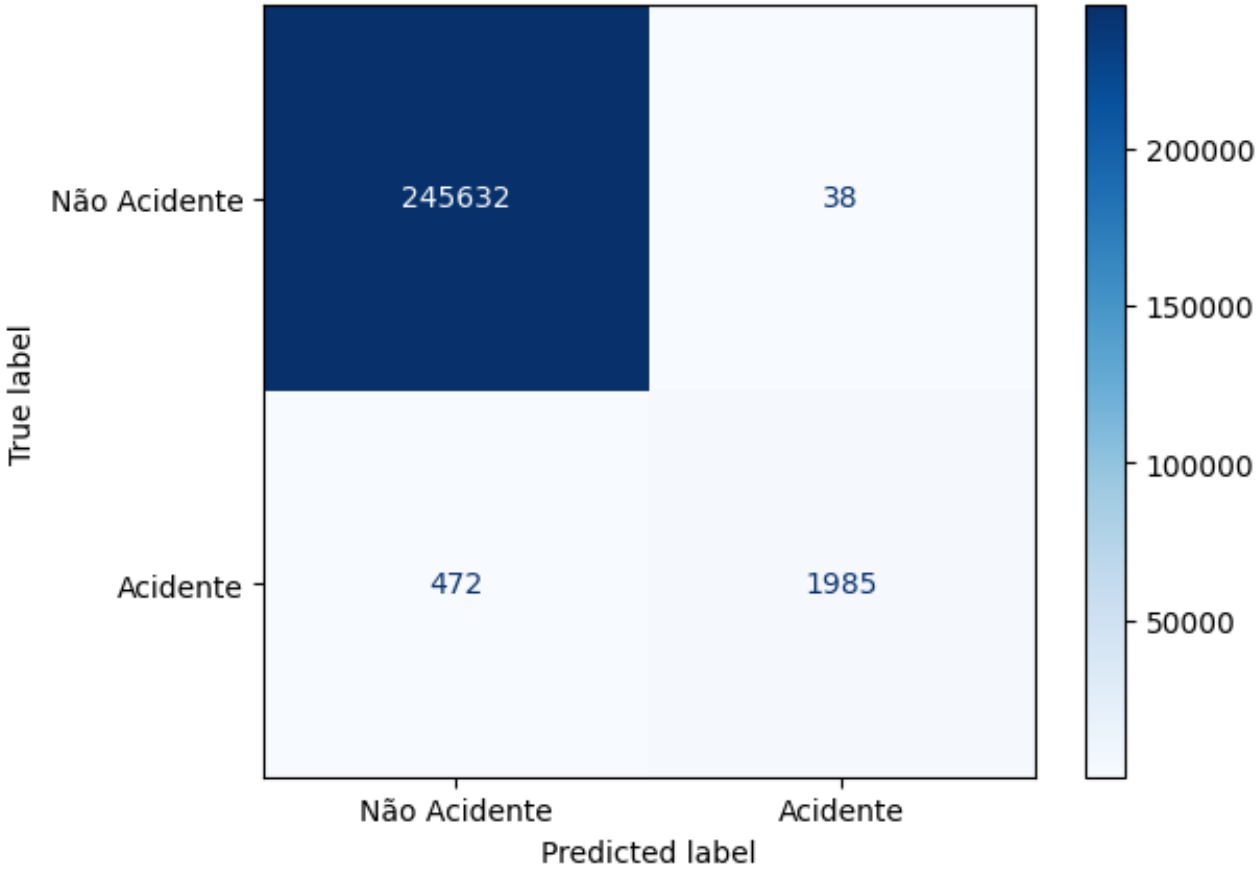
Fonte: Elaborada pelos autores.

4.6.2 Seleção do modelo final

Com base nos resultados comparativos apresentados, o modelo *Random Forest* foi selecionado como o artefato preditivo final do sistema. O modelo alcançou um desempenho superior nas métricas de avaliação, atingindo uma PR-AUC de 0.8606, ROC-AUC de 0.9661, *Precision* de 0.98, *Recall* de 0.81 e *F1-score* de 0.89, perdendo somente por 0.04 para o *XGBoost* no *Recall*, o que é irrelevante. Este modelo é, portanto, o motor da API de predição, carregado no *backend* para ser consumido pelo sistema de alertas em tempo real.

A análise da matriz de confusão (Figura 13) complementa o entendimento das métricas, traduzindo o desempenho em termos de impacto operacional. No conjunto de testes, com o *threshold* definido para otimizar o *Recall* sem comprometer excessivamente a *Precision*, o modelo classificou corretamente 1.985 casos de alto risco (Verdadeiros Positivos) e 245.632 casos de baixo risco (Verdadeiros Negativos). O sistema geraria 38 Falsos Positivos (alertas desnecessários), um custo aceitável para mitigar a ocorrência de 472 Falsos Negativos, que representam falhas críticas do sistema em prever acidentes reais. Essa matriz demonstra o equilíbrio operacional alcançado pelo modelo na identificação de áreas críticas em Bauru, priorizando a segurança com um número reduzido de alertas falsos.

Figura 13 – Matriz de confusão do *Random Forest*



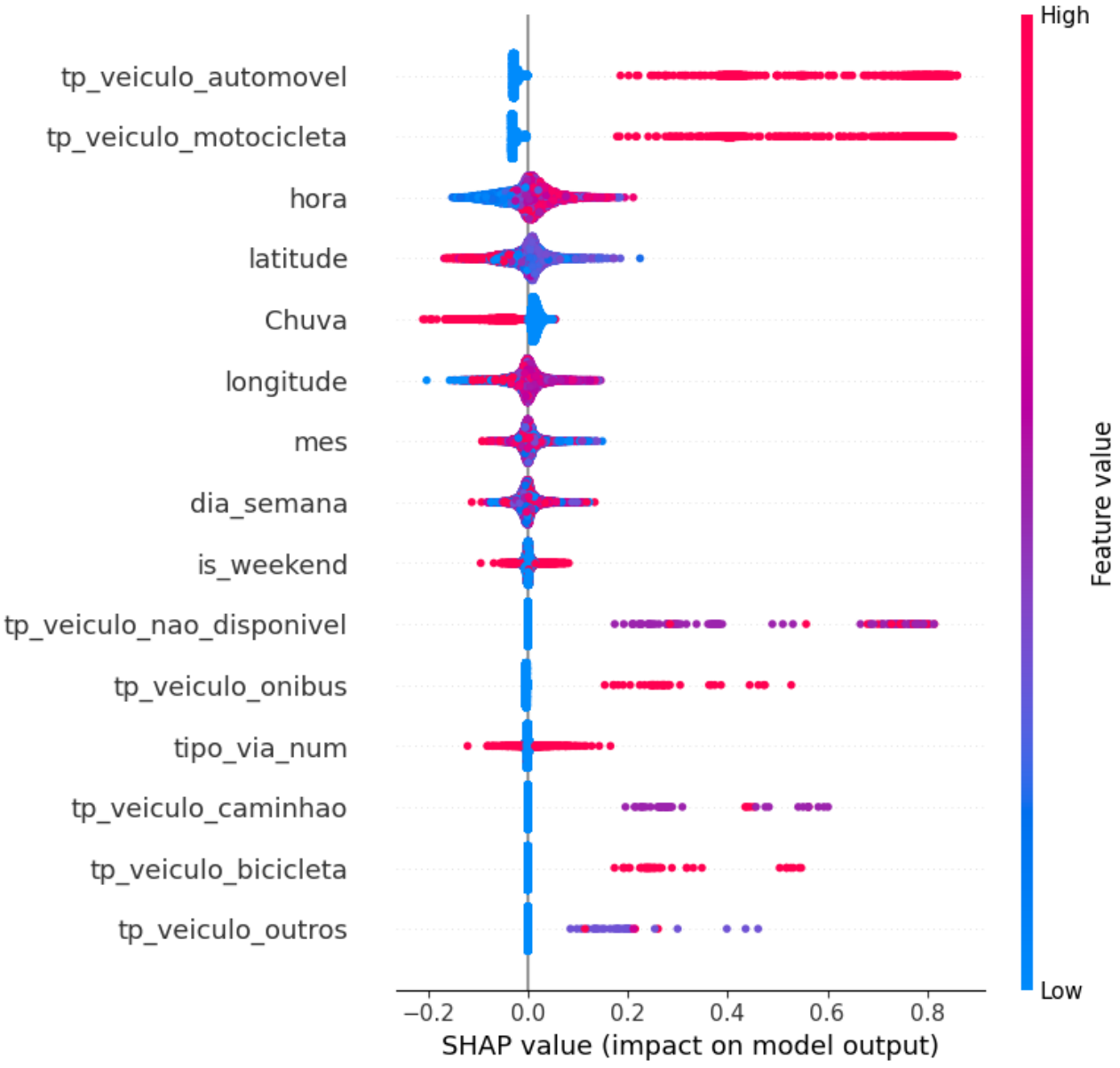
Fonte: Elaborada pelos autores.

4.6.3 Análise da interpretabilidade e importância das variáveis

Para além das métricas de desempenho, a compreensão da forma como o modelo realiza suas predições é vital para garantir a confiabilidade e a validade causal da solução. O método SHAP (SHapley Additive exPlanations), utilizado neste trabalho, permite decompor a predição de risco e analisar a contribuição marginal de cada *feature* individualmente e em conjunto.

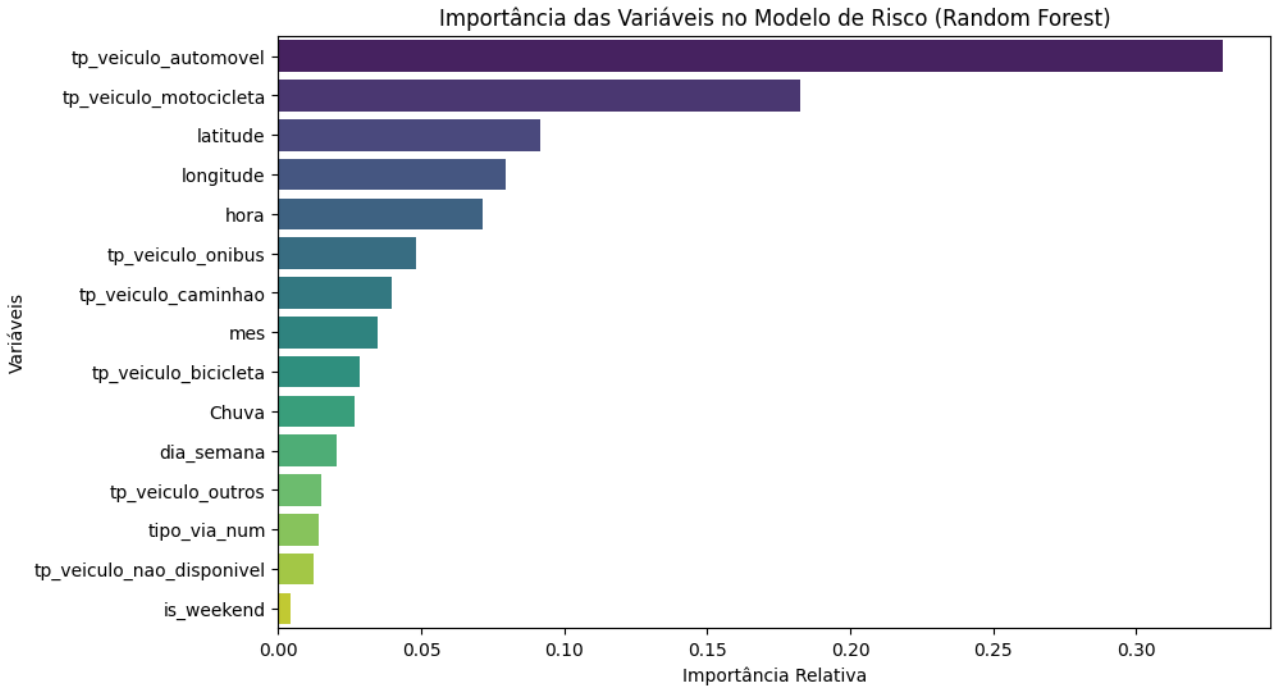
A análise da importância das variáveis, calculada via redução média de impureza (MDI) do modelo, permitiu validar a coerência interna do algoritmo. As Figuras 14 e 15 ilustram a hierarquia dos atributos preditivos através da análise de SHAP, que fornece o entendimento final da importância e da influência de cada *feature* na predição do modelo. A partir desta visualização, é possível confirmar que as variáveis mais relevantes para a predição do risco de acidentes são o tipo de veículo e o contexto temporal.

Figura 14 – SHAP



Fonte: Elaborada pelos autores.

Figura 15 – Importância das variáveis no modelo



Fonte: Elaborada pelos autores.

Especificamente, as *features* "tp_veiculo_motocicleta"e "tp_veiculo_automovel"(indicando a presença ou envolvimento desses veículos no sinistro) demonstram ter o maior impacto positivo no score de risco (pontos vermelhos mais à direita), validando a hipótese inicial sobre a vulnerabilidade do primeiro e a alta frequência de envolvimento do segundo. Em seguida, a variável "hora"(hora exata do sinistro) também é uma forte preditora, com valores mais altos (horas de pico, representadas em vermelho) empurrando a previsão para um risco maior. As variáveis geográficas ("latitude"e "longitude") e contextuais ("Chuva", "mes", "dia_semana") seguem como fatores relevantes, com impacto moderado e equilibrado, atuando como modificadores do risco base. Por fim, *features* como "tp_veiculo_caminhao", "tp_veiculo_bicicleta"e "tp_veiculo_outros"(outros tipos de modais) demonstram um impacto menor na média, mas ainda contribuem para a predição em casos específicos.

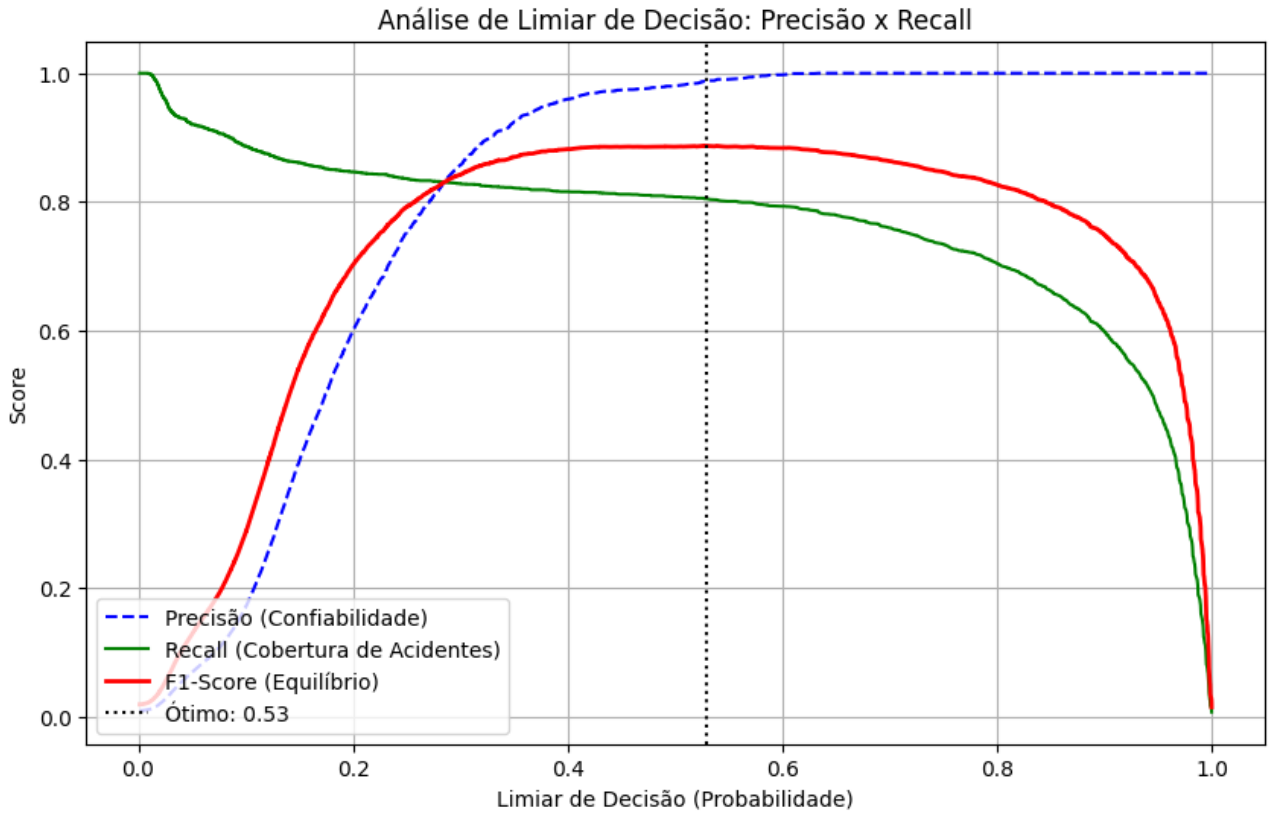
Em síntese, a análise SHAP não apenas confirma que o modelo está aprendendo os padrões esperados, mas revela que o risco de acidentes em Bauru é primariamente contextual e temporal, e não apenas espacial. Embora o georreferenciamento seja o insumo essencial para a localização e o alerta, a intensidade do risco é determinada pelas variáveis de contexto (tipo de veículo e hora). Essa conclusão valida a robustez do conjunto de variáveis selecionado e a natureza preditiva da solução de alerta em tempo real.

4.6.4 Análise e definição do Threshold

Para determinar o limiar operacional do sistema SHIELD, foi realizada uma análise da

curva *Precision-Recall* no conjunto de teste do modelo final. O objetivo foi maximizar o *F1-Score*, métrica que representa a média harmônica entre precisão e revocação, indicando o ponto de equilíbrio ideal onde o sistema detecta o maior número de acidentes possível sem gerar um volume excessivo de falsos alertas. A Figura 16 apresenta a variação dessas métricas conforme o limiar de probabilidade aumenta.

Figura 16 – Análise de limiar de decisão: Precisão x *Recall*



Fonte: Elaborada pelos autores.

A análise quantitativa indicou que o *F1-Score* máximo é atingido com um limiar ótimo de 0.5294.

Com base nessa análise empírica e visando a simplificação da regra de negócio sem perda significativa de performance, optou-se por arredondar este valor para 0.5 como o ponto de corte para alertas críticos. Essa escolha é estrategicamente fundamentada na gestão do custo do erro: ao selecionar um limiar próximo ao *F1-Score* máximo, o sistema mantém uma alta precisão (confiabilidade), minimizando a fadiga de alarmes, enquanto ainda garante um alto *recall* (cobertura de acidentes), o que é crucial para mitigar acidentes não previstos. Além disso, esse arrendondamento também se da por uma questão de segurança, onde é melhor errar alertando um pouco mais (*Recall mais elevado* do que deixar passar um risco real.

A lógica de decisão da *API* foi então configurada com três níveis de alerta para fornecer uma melhor experiência ao usuário:

- **Risco ALTO (Limiar $\geq$ 0.5):** Definido com base no limiar ótimo encontrado. Neste nível, a probabilidade de acidente é estatisticamente significativa e a confiança do modelo é alta, exigindo atenção imediata do condutor;
- **Risco MÉDIO (Limiar entre 0.3 e 0.5):** Definido com o propósito de criar um estado de atenção preventiva no condutor sem necessariamente disparar um alerta.
- **Risco BAIXO (Limiar $<$ 0.3):** Abaixo deste valor, o sistema considera as condições normais de tráfego e não emite alertas para evitar a fadiga de alarmes no usuário.

Essa abordagem estratificada, fundamentada nos dados de teste, permite que o sistema SHIELD seja robusto (detectando a maioria dos riscos reais) e confiável (minimizando interrupções desnecessárias), validando sua aplicabilidade em um cenário real de trânsito urbano.

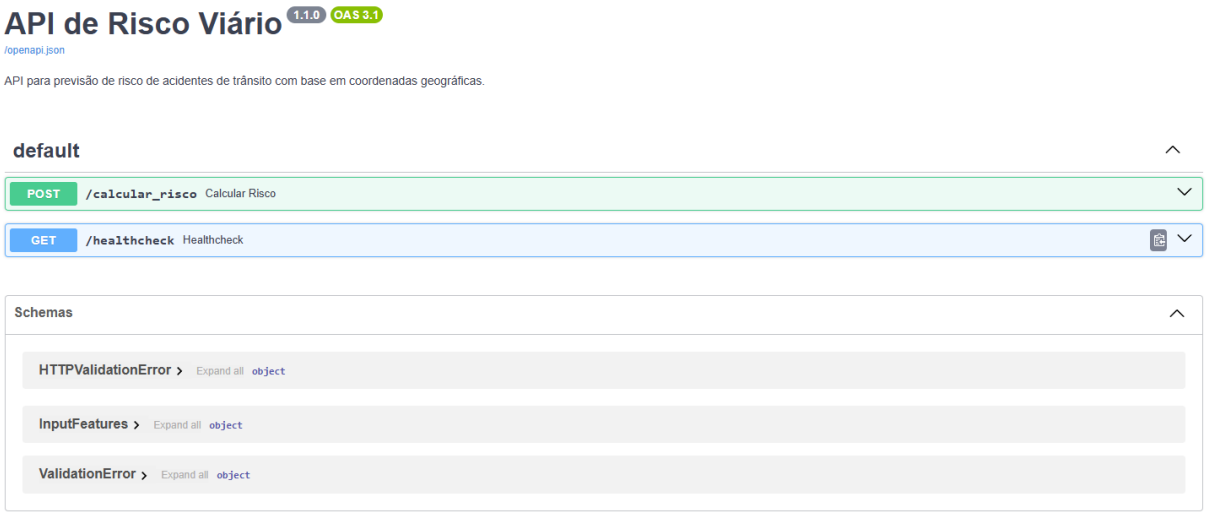
4.7 Desenvolvimento da *API*

A arquitetura do *backend* é centralizada na *API* de predição, estabelecida pelo *framework FastAPI*, com o objetivo primário de fornecer a funcionalidade de alerta de risco em tempo real. Sua função essencial é operacionalizar o modelo de *machine learning* (Seção 4.5) para a emissão de zonas de risco dinâmicas, o que demandou a escolha de uma solução de alto desempenho.

Neste contexto, a decisão de empregar o *FastAPI*, em detrimento de estruturas síncronas tradicionais como *Flask* ou *Django*, foi estritamente determinada pela necessidade de alta concorrência em um ambiente sujeito a limitações de *I/O Bound*. Embora a linguagem *Python* possua o *GIL*, que restringe a execução paralela de *threads* em operações de CPU, o *FastAPI* mitiga essa latência ao operar sobre o padrão *ASGI* e utilizar a sintaxe nativa *async/await*. Essa arquitetura permite que a repetição de eventos libere o processamento para atender novas requisições de risco enquanto o sistema aguarda respostas de serviços externos latentes, como a *API* da *Open-Meteo* e a *Overpass API*. Dessa forma, o tempo de espera pela rede não bloqueia o servidor, assegurando que a consulta de dados climáticos e viários ocorra de maneira não-bloqueante, garantindo a performance em tempo real e a escalabilidade necessárias para um sistema crítico de segurança viária.

Adicionalmente, a ferramenta oferece validação automática de dados (via *Pydantic*) e documentação interativa integrada (via *Swagger UI*), ilustrada na Figura 17, atributos que facilitam a manutenção e a rastreabilidade do projeto, justificando plenamente sua seleção como a tecnologia ideal para o serviço de dados.

Figura 17 – Documentação da *API* (*Swagger UI*)



Fonte: Elaborada pelos autores.

O *endpoint* principal, denominado "/calcular_risco", aceita dados de entrada validados pelo esquema *InputFeatures* do *Pydantic*, que exige as coordenadas geográficas "latitude"e "longitude"em formato de ponto flutuante, obtidas diretamente da geolocalização do usuário no navegador, e o "tp_veiculo_selecionado"definido na interface e selecionado pelo usuário (cuja obtenção será detalhada na subseção 4.8.3). Este esquema de validação pode ser observado na Figura 18).

Figura 18 – Esquema de validação da API

```python
# SCHEMA DE ENTRADA
class InputFeatures(BaseModel):
    latitude: float
    longitude: float
    tp_veiculo_selecionado: str = Field(..., description="Tipo de veículo selecionado")
```

Fonte: Elaborada pelos autores.

Dentro deste *endpoint*, ocorre o processo crucial de enriquecimento de dados e preparação do *payload*. Utilizando a função "datetime.now()", o sistema obtém o contexto temporal imediato, extraindo variáveis essenciais para a predição, como o "dia_semana", "mes", o estado binário "is_weekend"e a "hora_int". Essas variáveis temporais e contextuais são as mesmas utilizadas na fase de treinamento do modelo e descritas na subseção (4.5). Tais variáveis, juntamente com o tratamento *one-hot encoding* para o tipo de veículo selecionado, são reunidas em um *dataframe* de entrada. Para fins de rastreabilidade e depuração do sistema, o conteúdo exato deste conjunto de dados é registrado nos *logs* da aplicação (logging.info) antes do processamento final, garantindo a validação dos atributos enriquecidos. Em seguida, os dados são formatados no objeto df_processed (*dataframe*) do

Pandas, seguindo a ordem exata dos atributos esperados pelo modelo carregado via "joblib". A construção detalhada deste *payload* e a criação do *dataframe* estão representadas na Figura 19. Finalmente, o modelo realiza a predição da probabilidade ("prob"), e o sistema aplica um "limiar"para categorizar e retornar o nível de risco estimado como "ALTO", "MÉDIO"ou "BAIXO"na chave "interpretacao", o resultado da predição é sumarizado na Figura 20.

Figura 19 – Construção do *payload* no *endpoint* para cálculo de risco

```python
# ENDPOINT PRINCIPAL
@app.post("/calcular_risco")
async def calcular_risco(features: InputFeatures):
    if model is None:
        raise HTTPException(status_code=500, detail="Modelo não disponível no servidor.")

    if not model_features:
        logging.error("O modelo foi carregado, mas a lista de 'model_features' está vazia. Verifique o artefato .pkl.")
        raise HTTPException(status_code=500, detail="Configuração de modelo inválida no servidor.")

    try:
        now = datetime.now()
        dia_semana = now.weekday()
        mes = now.month
        is_weekend = 1 if dia_semana >= 5 else 0
        hora_int = now.hour

        input_data = {
            "latitude": features.latitude,
            "longitude": features.longitude,
            "dia_semana": dia_semana,
            "mes": mes,
            "is_weekend": is_weekend,
            "hora": hora_int,
            "tp_veiculo_bicicleta": 0,
            "tp_veiculo_caminhao": 0,
            "tp_veiculo_motocicleta": 0,
            "tp_veiculo_nao_disponivel": 0,
            "tp_veiculo_onibus": 0,
            "tp_veiculo_outros": 0,
            "tp_veiculo_automovel": 0,
            "tipo_via_num": response_via,
            "chuva": response_chuva
        }

        if features.tp_veiculo_selecionado in input_data:
            input_data[features.tp_veiculo_selecionado] = 1
        elif features.tp_veiculo_selecionado != "tp_veiculo_nao_disponivel":
            logging.warning(f"Tipo de veículo '{features.tp_veiculo_selecionado}' não é uma feature esperada. Usando 'nao_disponivel'.")
            input_data["tp_veiculo_nao_disponivel"] = 1


        df_input = pd.DataFrame([input_data])
        logging.info(f"Dados para predição: {df_input.to_dict()}")
```

Fonte: Elaborada pelos autores.

Figura 20 – Predição da API

```
    # Realiza a predição
    prob = model.predict_proba(df_processed[model_features])[:, 1]
    risco = float(prob[0])

    limiar = 0.45

    if risco >= limiar:
        interpretacao = "ALTO"
    elif risco >= 0.2:
        interpretacao = "MÉDIO"
    else:
        interpretacao = "BAIXO"

    return {
        "risco_estimado": risco,
        "interpretacao": interpretacao,
        "timestamp": datetime.now().isoformat(),
    }

except Exception as e:
    logging.error(f"Erro na predição: {traceback.format_exc()}")
    raise HTTPException(status_code=500, detail=f"Erro interno: {str(e)}")
```

Fonte: Elaborada pelos autores.

Para fins de diagnóstico e validação manual no ambiente de testes, foi implementado o *endpoint* "/calcular_risco_teste". Diferentemente da rota principal, que exige apenas dados brutos e realiza todo o enriquecimento internamente, esta *API* de teste foi desenhada para receber o conjunto completo de *features* diretamente no corpo da requisição. Sua função é facilitar o teste de regressão e a depuração do pipeline ao permitir que seja inserido um *payload* totalmente especificado, contornando a lógica de obtenção de dados externos e temporais. A lógica desta rota garante a integridade dimensional do conjunto de dados de entrada, tratando variáveis ausentes (que seriam provenientes do *one-hot encoding*) ao preenchê-las com valor zero. O retorno deste *endpoint* de teste inclui o risco estimado e a lista exata de variáveis utilizadas, garantindo a rastreabilidade e a validação do processo de predição em cenários específicos, demonstrado na Figura 21.

Figura 21 – *Endpoint* para testes

```python
# ============================
# ENDPOINT PARA TESTES
# ============================
@app.post("/calcular_risco_teste")
async def calcular_risco_teste(input_data: dict):
    """
    Endpoint para testes manuais — recebe todas as features diretamente no corpo.
    """
    if model is None:
        raise HTTPException(status_code=500, detail="Modelo não disponível no servidor.")

    if not model_features:
        logging.error("O modelo foi carregado, mas 'model_features' está vazio.")
        raise HTTPException(status_code=500, detail="Configuração de modelo inválida no servidor.")

    try:
        if not isinstance(input_data, dict):
            raise HTTPException(status_code=400, detail="O corpo deve ser um dicionário com as features do modelo.")

        df_input = pd.DataFrame([input_data])
        logging.info(f"Dados brutos recebidos para teste: {df_input.to_dict()}")

        for col in model_features:
            if col not in df_input.columns:
                df_input[col] = 0

        df_processed = df_input[model_features].astype(float)

        # Predição
        prob = model.predict_proba(df_processed)[:, 1]
        risco = float(prob[0])

        limiar = 0.5
        interpretacao = "ALTO" if risco >= limiar else "MÉDIO" if risco >= 0.3 else "BAIXO"

        # Retorno completo
        return {
            "risco_estimado": risco,
            "interpretacao": interpretacao,
            "features_recebidas": list(df_input.columns),
            "features_utilizadas": df_processed.to_dict(orient="records")[0],
            "timestamp": datetime.now().isoformat(),
        }

    except Exception as e:
        logging.error(f"Erro na predição (teste): {traceback.format_exc()}")
        raise HTTPException(status_code=500, detail=f"Erro interno: {str(e)}")
```

Fonte: Elaborada pelos autores.

4.8 Desenvolvimento do Front-end

O desenvolvimento da interface de usuário constituiu um módulo crucial, responsável por apresentar os resultados da análise preditiva e permitir a interação georreferenciada em tempo real com o sistema. A escolha da *stack* tecnológica foi precedida por uma análise comparativa, em que *frameworks* robustos como *React* e *Angular* foram avaliados e posteriormente descartados, principalmente devido à sua maior complexidade de configuração e à estrutura excessivamente rígida para o escopo e agilidade almejados pelo projeto. A arquitetura final foi estabelecida sobre duas tecnologias complementares. Como espinha

dorsal, utilizou-se o *JavaScript* moderno (ES6+), essencial para a manipulação assíncrona dos dados via *Axios* e para a integração com bibliotecas de visualização, como *Leaflet.js*. Para gerenciar o estado da aplicação e construir a interface reativa, foi selecionado o *framework* progressivo *Vue.js* 3. Essa escolha foi metodologicamente justificada pela sua curva de aprendizado suave, pelo seu foco em simplicidade e performance. A eficiência no desenvolvimento foi garantida pelo uso de *Single File Components* (.vue) e de diretivas nativas do *framework*, permitindo a criação de uma experiência de usuário rica, responsiva e intuitiva, especialmente na exibição de dados geográficos e filtros dinâmicos.

4.8.1 Página de dados históricos

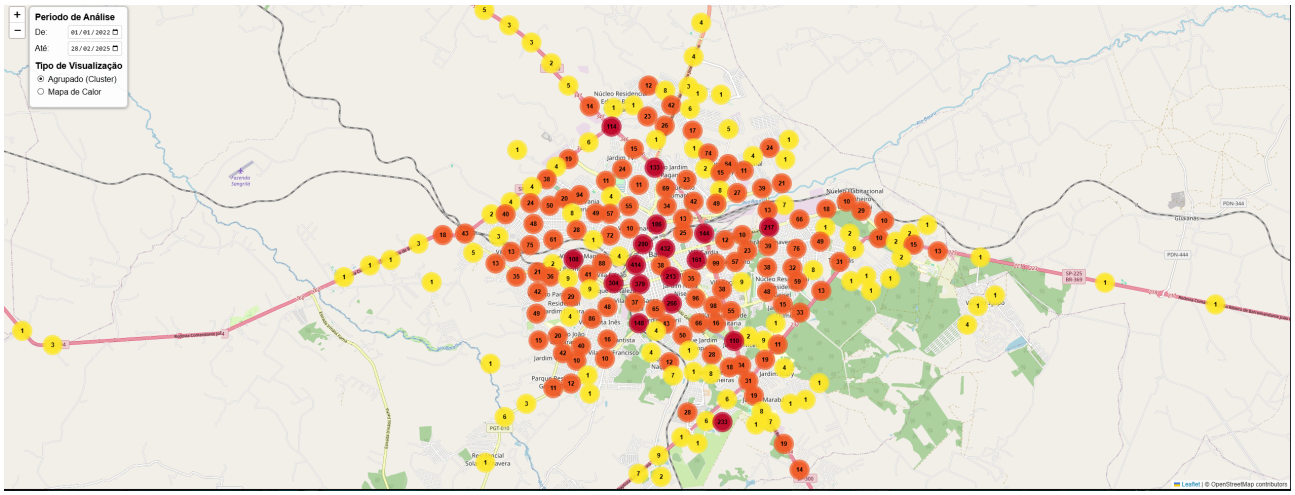
O componente MapaBauru visa a visualização dinâmica e interativa do histórico de sinistros em Bauru, incorporando funcionalidades avançadas de filtragem e diferentes modalidades de representação espacial. Estruturalmente, o componente gerencia os principais estados da aplicação, como o carregamento dos dados, o modo de visualização escolhido e o intervalo temporal da análise. A integração é realizada com a biblioteca *Leaflet.js* para o mapa e com o *Supercluster* para o agrupamento dos pontos, sendo complementada pela *Leaflet.heat* para o mapa de calor, permitindo a alternância entre diferentes visualizações.

A manipulação de dados é iniciada com a requisição assíncrona do arquivo "coordenadas.json"(obtido na subseção 4.3), que consolida todos os pontos de sinistros válidos. Esta etapa inclui uma validação e normalização inicial rigorosa, convertendo as coordenadas para ponto flutuante e verificando-as contra limites geográficos válidos, de modo a descartar registros anômalos. Após o carregamento, o sistema determina automaticamente o intervalo temporal mínimo e máximo dos dados, utilizando o registro mais antigo e o mais recente para configurar os filtros de data da interface. O núcleo da filtragem é uma propriedade reativa que aplica o intervalo de datas selecionado pelo usuário ao conjunto completo de dados, garantindo que o mapa e as camadas de visualização sejam atualizados dinamicamente sempre que os filtros temporais forem alterados.

O sistema oferece ao usuário a escolha entre dois modos de visualização: o Agrupamento (*Cluster*) ou o Mapa de Calor (*Heatmap*), selecionáveis diretamente na interface. A manipulação de dados é responsável por pré-processar os dados filtrados: ela recarrega o índice Supercluster com os pontos atualizados, essencial para o modo *cluster*, e, simultaneamente, constrói a camada de calor. A customização do mapa de calor define parâmetros de densidade e estilo, como raio, desfoque e um gradiente de cores que varia de azul (menor densidade) a vermelho (maior densidade). A alternância entre as visualizações é gerenciada por uma função que limpa a camada inativa e adiciona ao mapa apenas o modo selecionado. A renderização dos agrupamentos é finalizada com o uso do *Supercluster* apenas quando o modo *cluster* está ativo, mantendo o padrão de categorização visual dos ícones baseado na contagem de sinistros.

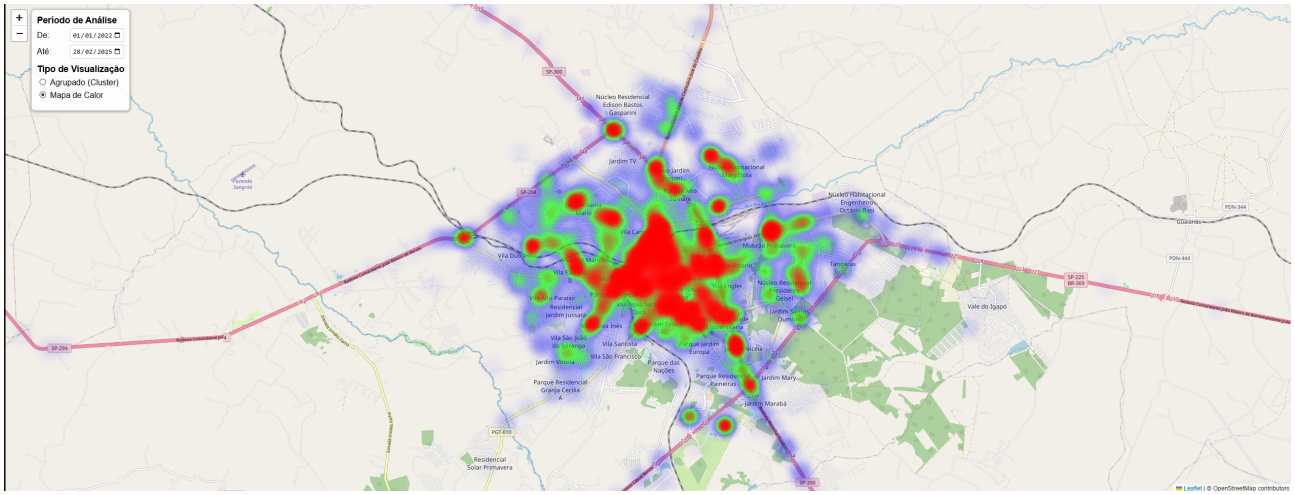
Essa página de visualização de dados históricos, com sua robusta capacidade de filtragem temporal e alternância de modos de representação, transforma o volume de dados de acidentes em inteligência espacial acionável, permitindo aos analistas identificar padrões de concentração pontual e áreas de risco generalizado no tecido urbano por meio da flexibilidade em apresentar os sinistros como agrupamentos de clusters ou como um mapa de calor de densidade. A seguir, a Figura 22 ilustra a visualização dos acidentes por meio do modo de agrupamento, e a Figura 23 demonstra a representação pela zona de calor.

Figura 22 – Visualização dos dados históricos por *clusters*



Fonte: Elaborada pelos autores.

Figura 23 – Visualização dos dados históricos por zonas de calor



Fonte: Elaborada pelos autores.

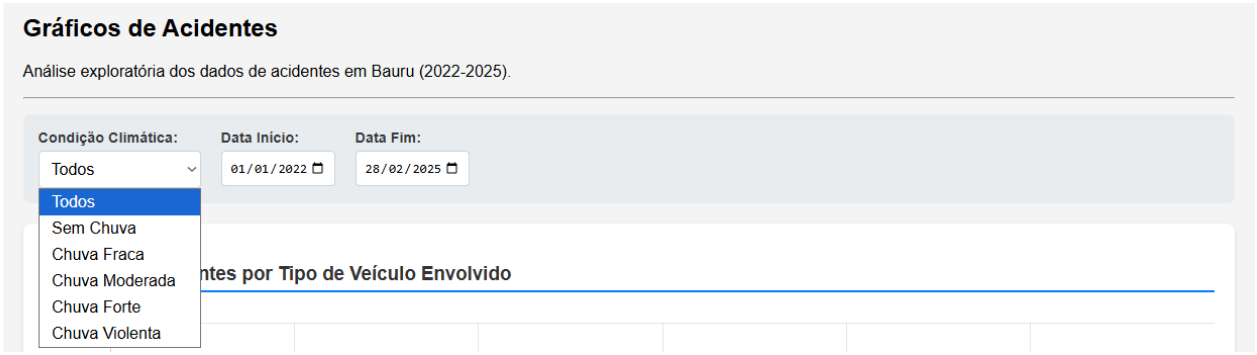
4.8.2 Página de gráficos

O componente de gráficos de acidentes estrutura a seção de análise exploratória de dados, cujo objetivo primordial é prover uma visualização quantitativa e dinâmica dos

sinistros em Bauru entre 2022 e 2025. A funcionalidade é iniciada pelo carregamento assíncrono de dois conjuntos de dados primários, relativos a acidentes e vítimas, que são armazenados em estados reativos da aplicação.

Em seguida, o sistema determina o intervalo temporal de validade dos dados, estabelecendo o período mínimo e máximo para a interface de filtragem. O sistema de interação oferece ao usuário a capacidade de refinar a análise temporal por meio de inputs de data de início e fim, além de permitir a aplicação de filtros baseados na condição climática (como "Sem chuva", "Chuva Fraca"e "Chuva Moderada", ilustrado na Figura 24).

Figura 24 – Filtros de data e clima



Fonte: Elaborada pelos autores.

Esta filtragem é processada de forma otimizada por propriedades computadas, que são observadas reativamente. Essa lógica garante que as visualizações sejam dinamicamente atualizadas sempre que os critérios de filtro do usuário forem alterados. A partir dos dados filtrados, são geradas cinco visualizações estatísticas principais, que agregam as informações por: tipo de veículo envolvido, hora do dia, perfil demográfico da vítima e gravidade da lesão, tipo de via e impacto da chuva na gravidade dos sinistros.

Tais visualizações fornecem um panorama estatístico abrangente dos fatores de risco, horários de pico e o perfil das vítimas, essenciais para a compreensão das dinâmicas do acidentário. O conjunto completo destas análises quantitativas é apresentado e detalhado nas Figuras 8 a 12, as quais ilustram a distribuição dos acidentes por diferentes fatores contextuais e demográficos.

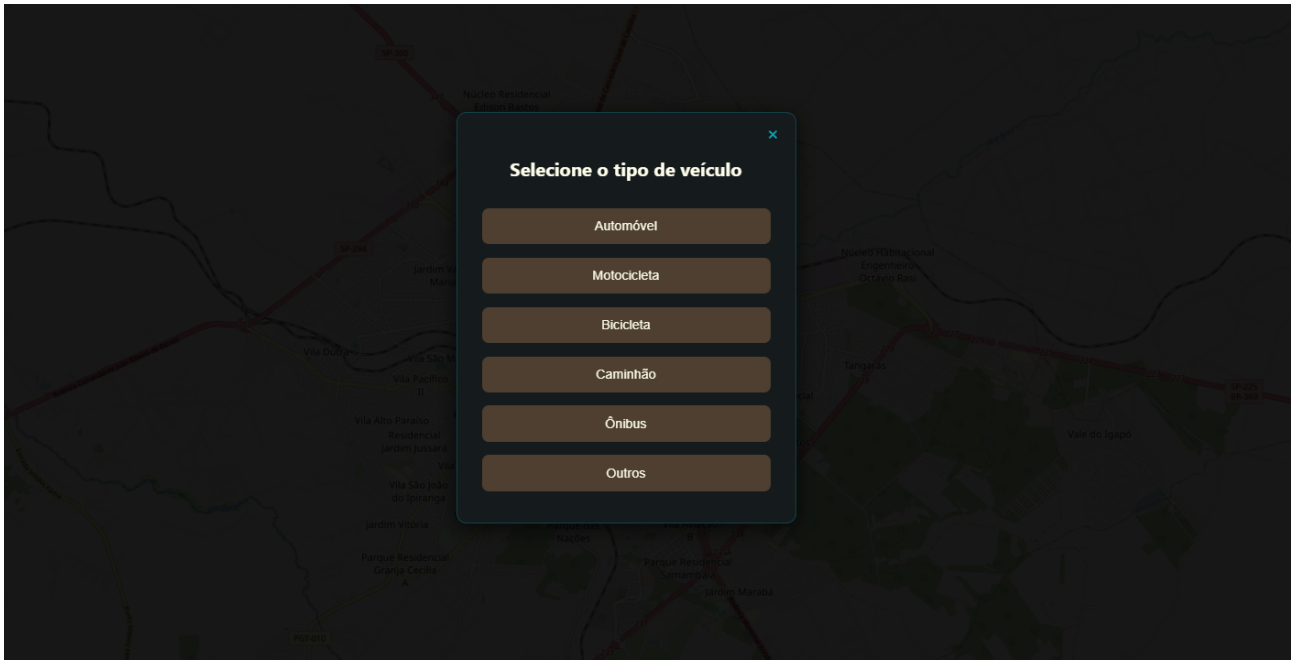
4.8.3 Página de alertas

O componente *AlertasBauru.vue* estabelece o sistema de emissão de alertas em tempo real. Este componente é dinâmico e consome diretamente o modelo preditivo (subseção 4.6) por meio da *API* (subseção 4.7), fechando o ciclo do sistema e cumprindo o objetivo principal do projeto: entregar o alerta de segurança viária.

O processo de interação do usuário é iniciado pela obtenção da localização geográfica. Para isso, o componente invoca a *API* de geolocalização nativa do navegador, solicitando

a permissão do usuário. Uma vez obtidas, as coordenadas de latitude e longitude são armazenadas no estado reativo da aplicação. Simultaneamente, o usuário interage com a interface selecionando o tipo de veículo que está utilizando, valor que é armazenado para compor o *payload* de predição, como representado na Figura 25.

Figura 25 – Lista de veículos



Fonte: Elaborada pelos autores.

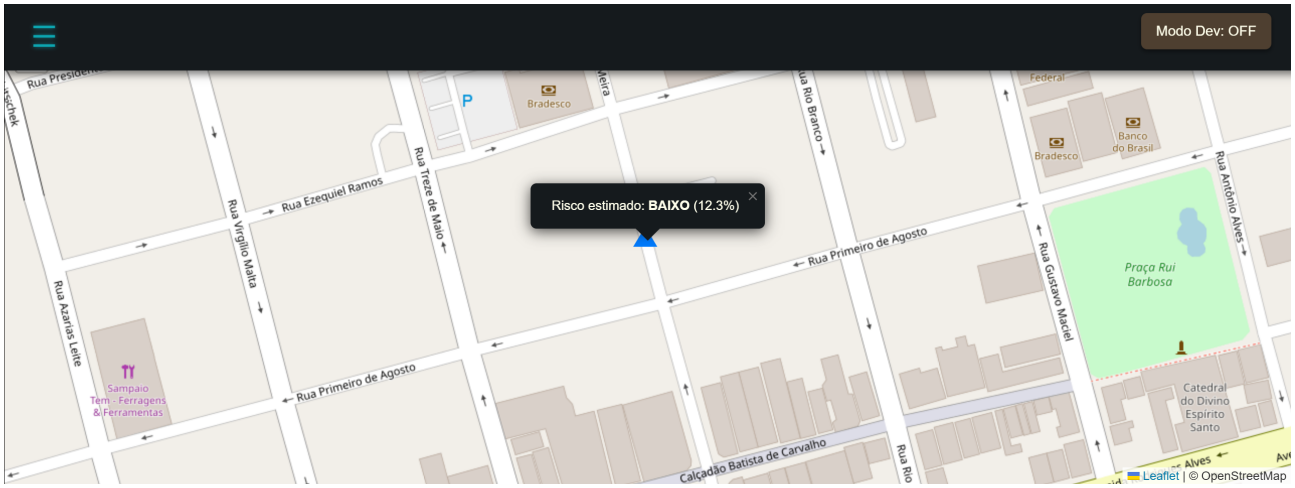
Com a localização geográfica atual e o tipo de veículo definidos e armazenados no estado reativo do componente, a aplicação inicia sua operação de monitoramento disparando a primeira requisição à *API* para calcular o risco, utilizando a biblioteca *Axios* para enviar os dados. Este *endpoint* atua como o núcleo da lógica de *backend*: ele recebe os dados básicos e, para completar a inferência do modelo, os enriquece imediatamente com informações contextuais. Primeiramente, são adicionados atributos temporais (como hora, mês e dia da semana). Adicionalmente, a capacidade preditiva do sistema é complementada pela integração com fontes de dados exógenas mediante consultas assíncronas a serviços externos: a precipitação é obtida da *API Open Meteo* em intervalos de cinco minutos, enquanto a classificação da infraestrutura viária (municipal ou rodoviária) é determinada pela *Overpass API* da *OpenStreetMap*, com atualização a cada sessenta segundos. Esses atributos enriquecidos, que refletem as variáveis treinadas no modelo preditivo, são cruciais e garantem que o *payload* submetido ao algoritmo de *machine learning* para a inferência de risco contenha todas as variáveis contextuais dinâmicas necessárias para maximizar a precisão do *score* (Figuras 19 e 20).

A *API*, então, retorna uma resposta *JSON* contendo o score probabilístico e a interpretação categórica ("BAIXO", "MÉDIO", "ALTO"), definida por limiares pré-estabelecidos no servidor. Estes dados retornados são usados para atualizar o estado da aplicação, que

exibe o resultado ao usuário de forma visual e intuitiva. A visualização é complementada por um sistema de cores na seta de geolocalização do usuário, que é alterada para azul quando o risco é "BAIXO", amarelo para "MÉDIO"e vermelho para "ALTO". Esta diferenciação visual, essencial para a experiência do usuário e a tomada de decisão em trânsito, pode ser observada detalhadamente nas Figuras 26, 27 e 28, respectivamente.

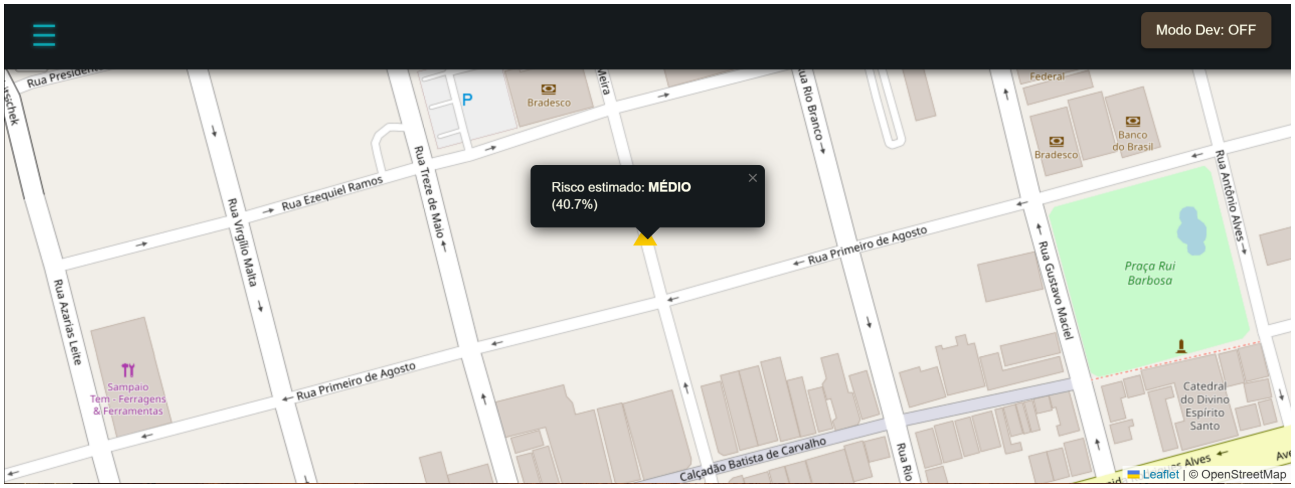
Adicionalmente, para maximizar a atenção e o engajamento do condutor em cenários críticos, a detecção de risco "ALTO"aciona um alerta sonoro que é emitido em conjunto com a mudança da cor para vermelho. Esse reforço multimodal (visual e auditivo) eleva a eficácia do sistema como uma ferramenta preventiva.

Figura 26 – Representação visual para risco baixo



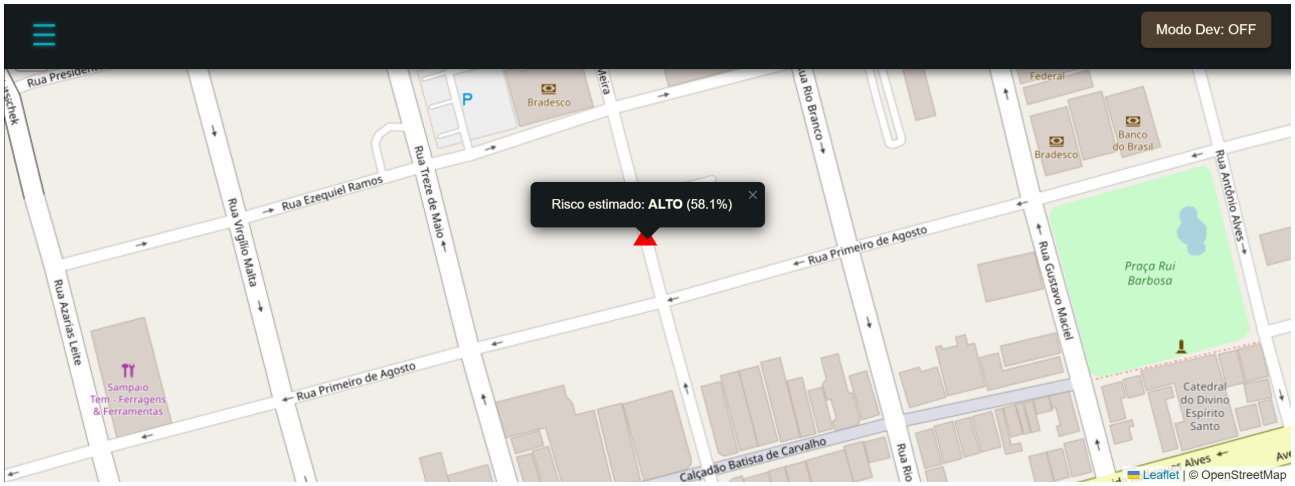
Fonte: Elaborada pelos autores.

Figura 27 – Representação visual para risco médio



Fonte: Elaborada pelos autores.

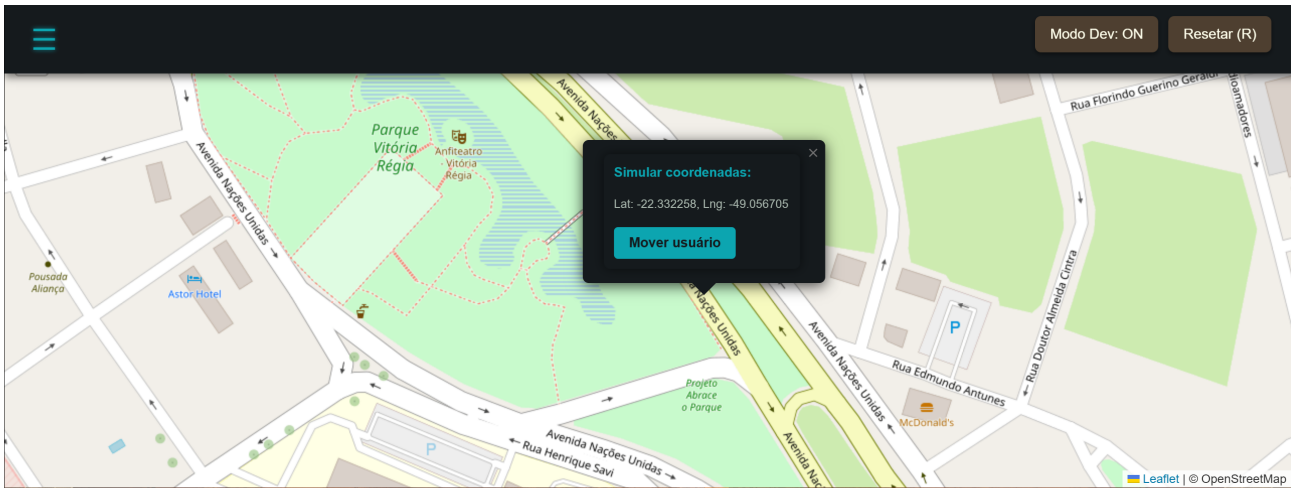
Figura 28 – Representação visual para risco alto



Fonte: Elaborada pelos autores.

Para fins de teste e garantia da confiabilidade do sistema, foi desenvolvido um modo de desenvolvedor que permite simular a localização do usuário. Ao acionar esta funcionalidade, o usuário pode clicar em qualquer ponto do mapa para definir uma nova coordenada. Essa simulação injeta artificialmente a latitude e longitude escolhidas no *payload* da requisição, substituindo a geolocalização nativa do navegador. Esse mecanismo é fundamental para validar a resposta preditiva em diferentes zonas da cidade, assegurando que a lógica de enriquecimento de dados e a inferência do modelo estejam funcionando corretamente em diversas condições espaciais e contextuais, conforme ilustrado na Figura 29.

Figura 29 – Representação visual do modo de desenvolvedor



Fonte: Elaborada pelos autores.

4.8.4 Menu principal

Para garantir a usabilidade e uma arquitetura de navegação clara, o *frontend* foi

estruturado em torno do componente Menu Inicial (Figura 30), que serve como o ponto de entrada principal para o sistema SHIELD.

Figura 30 – Menu inicial do projeto

Fonte: Elaborada pelos autores.

Este componente central é implementado com base na lógica de reatividade do *Vue.js*. Ele utiliza o mecanismo de emissão de eventos para controlar, de forma eficiente, a transição entre as diversas vistas da aplicação.

O menu oferece ao usuário acesso direto a três funcionalidades principais, que correspondem aos módulos de visualização definidos no sistema: o módulo de Alertas em tempo real (4.8.3), o Mapa Histórico de Acidentes (4.8.1) e a seção de Gráficos de Análise Exploratória (4.8.2). Cada opção na interface está vinculada a um evento de interação (clique) que comunica o destino desejado a um componente gerenciador, que, então, se encarrega de renderizar a interface correspondente. Essa abordagem, que se baseia na componentização e no gerenciamento de estados reativos do Vue.js, assegura uma navegação fluida, coesa e modular. Além disso, permite que a aplicação escale de forma eficiente, evitando a necessidade de recarregar a página a cada transição. A definição visual e o estilo do menu são aplicados por meio da importação de um arquivo CSS externo, mantendo a separação ideal entre a apresentação e a lógica de manipulação dos componentes.

5 PROCEDIMENTOS DE VALIDAÇÃO DO PROJETO

Este capítulo descreve os procedimentos adotados para validar a eficácia e a robustez da solução proposta, assegurando que os objetivos definidos foram atingidos. A validação deste projeto foca em duas frentes principais: a validação quantitativa da performance do modelo preditivo e a validação funcional da aplicação de interface com o usuário.

5.1 Validação quantitativa do modelo preditivo

A validação central deste TCC consiste em comprovar quantitativamente que o modelo de *machine learning* desenvolvido é capaz de estimar o risco de acidentes de trânsito. Esta etapa é fundamental, pois serve como a prova de conceito da hipótese central do trabalho: que existe um padrão predizível nos sinistros de Bauru baseado em fatores temporais, espaciais e climáticos.

Conforme detalhado extensivamente no Capítulo 4.6 (Análise de Métricas e Seleção do Modelo Final), esta validação foi realizada através da avaliação rigorosa dos modelos candidatos em um conjunto de dados de teste. O desempenho foi medido utilizando métricas robustas para dados desbalanceados, com foco especial na PR-AUC, conforme definido na subseção 4.4.4.

A superioridade do modelo Random Forest nesta métrica validou tecnicamente sua capacidade de discriminar com eficácia entre cenários de baixo e alto risco. Portanto, esta seção confirma que o artefato central do sistema (o modelo preditivo) não apenas funciona, mas foi otimizado e selecionado com base em desempenho empírico, validando a abordagem de ciência de dados adotada.

5.2 Validação funcional e de usabilidade do sistema

Uma vez validado o núcleo preditivo, a segunda etapa de validação focou em garantir que a aplicação front-end cumpre os objetivos funcionais e de usabilidade do projeto. Esta validação, de natureza mais qualitativa, verificou se os *insights* gerados pelo modelo e pela análise de dados (Capítulo 4.3) são apresentados ao usuário de forma clara, correta e acionável.

Esta validação funcional foi centrada em três componentes principais da interface:

- **Validação dos módulos de análise (páginas de gráficos e mapa histórico)** Verificou-se se os dados exploratórios das seções 4.8.2 e 4.8.1 estavam sendo carregados corretamente e se os filtros de data e clima respondiam dinamicamente, permitindo ao usuário explorar os padrões históricos conforme esperado;

- **Validação da API e integridade do modelo:** O componente central do sistema é a API FastAPI, responsável por servir o modelo preditivo. Para isso, foi utilizado o *endpoint* healthcheck, implementado especificamente para este fim. Este *endpoint* não se limita a confirmar que o servidor está operacional; ele ativamente verifica se o artefato do modelo foi carregado com sucesso na memória e reporta o número de atributos que o modelo espera. A resposta positiva deste healthcheck foi a validação formal de que o modelo estava corretamente conectado à API;

- **Validação do fluxo de alertas:** Após a confirmação da integridade da API pelo healthcheck, a validação final seguiu para o *endpoint* principal calcular_risco.

Verificou-se que o *payload* (latitude, longitude, tipo de veículo) enviado pela interface front-end era corretamente processado, enriquecido com dados temporais (hora, mês, etc.) e passado ao modelo. O retorno bem-sucedido de um score de risco e sua respectiva interpretação ("BAIXO", "MÉDIO", "ALTO") validou funcionalmente todo o fluxo de alerta, desde a entrada do usuário até a resposta final, concluindo o ciclo de validação da solução.

6 CONCLUSÃO

O desenvolvimento deste trabalho permitiu comprovar a hipótese central de que os acidentes de trânsito em Bauru, embora complexos, seguem padrões predizíveis. A aplicação de técnicas de *machine learning* em dados públicos (Infosiga-SP e INMET) culminou no desenvolvimento de um modelo preditivo (*Random Forest*) validado, que alcançou uma PR-AUC de 0.8606 em dados de teste. Mais importante, o projeto entregou uma solução completa e funcional, composta pelo modelo preditivo, uma API de alto desempenho e uma interface interativa, demonstrando a aplicação prática e bem-sucedida da ciência de dados na segurança viária de Bauru.

No entanto, o contraste entre o pré-projeto e o desenvolvimento real revelou que os maiores desafios encontrados não foram os esperados. O risco de disponibilidade de dados provou ser o obstáculo central de toda a pesquisa. Na tentativa de mitigar este risco, foi realizada uma visita técnica à EMDURB (Empresa Municipal de Desenvolvimento Urbano e Rural de Bauru) para solicitar dados de fluxo de tráfego; nesta visita, foi constatada a inexistência de tais dados para o município. Essa escassez de dados de "não acidentes"e de tráfego tornou a criação de um conjunto de dados sintético robusto a tarefa mais crítica do projeto. Uma falha de atenção inicial nesta etapa, a não inclusão de dados de tipo de veículo nas amostras negativas, levou a dois meses de atraso no cronograma. Durante este período, os modelos (incluindo *SVMs*, que se mostraram ineficazes) apresentavam resultados contraintuitivos, com probabilidades de acidente consistentemente acima de 98%. A percepção de que o erro não estava na complexidade do modelo, mas na simplicidade dos dados de entrada, foi a lição mais desafiadora e valiosa do projeto.

Este desafio reconfigurou drasticamente o cronograma planejado. A etapa de estudo e seleção de modelos, prevista para dois meses, estendeu-se até o final do projeto. Em contrapartida, o desenvolvimento da *API*, estimado em meses, foi concluído em poucos dias graças à eficiência do *framework FastAPI*. Essa compressão forçou uma mudança de metodologia: em vez de um desenvolvimento linear começando com a criação de um modelo, *API* e por fim a aplicação web, as frentes de aprimoramento do modelo e desenvolvimento da interface de usuário precisaram ser executadas em paralelo para cumprir o prazo.

Diante desses desafios, as perspectivas de trabalhos futuros também foram recontextualizadas. O pré-projeto previa a integração com *APIs* de tráfego em tempo real (como

Waze ou *Google Maps*). Contudo, a constatação da inexistência de dados públicos de fluxo na EMDURB sugere que uma contribuição futura mais impactante seria uma parceria com o órgão público para utilizar o próprio sistema SHIELD como uma ferramenta piloto para iniciar a coleta e estruturação desses dados na cidade.

Em síntese, este trabalho entregou uma solução completa e funcional (modelo preditivo, *API* de alto desempenho e interface interativa) que comprova a aplicação bem-sucedida da ciência de dados na segurança viária de Bauru. Ao calcular a probabilidade de ocorrência absoluta como métrica viável para alerta em tempo real, dada a ausência de dados de fluxo, o projeto não apenas forneceu um artefato técnico validado e de baixo custo para a melhoria da mobilidade urbana, mas também serviu como uma investigação prática sobre os desafios reais da ciência de dados no setor público brasileiro, oferecendo um modelo replicável para outras cidades com desafios semelhantes.

7 REFERÊNCIAS

ORGANIZAÇÃO MUNDIAL DA SAÚDE. **Lesões causadas pelo trânsito**. 13 dez. 2023. Disponível em: https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries. Acesso em: 5 abr. 2025.

ASSOCIAÇÃO PAULISTA DE MEDICINA. **Brasil é o terceiro país com mais mortes de trânsito**. 2023. Disponível em: https://www.apm.org.br/brasil-e-o-terceiro-pais-com-mais-mortes-de-transito/. Acesso em: 5 abr. 2025.

BAURU (SP). **Decreto nº 14.446, de 22 de novembro de 2019**. Institui o Plano de Mobilidade Urbana de Bauru (PLANMOB) e dá outras providências. *Diário Oficial do Município de Bauru*, Bauru, 22 nov. 2019. Disponível em: `https://www2.bauru.sp.gov.br/arquivos/sist_juridico/documentos/decretos/dec14446.pdf`. Acesso em: 5 abr. 2025.

SÃO PAULO (Estado). Departamento Estadual de Trânsito. **Infosiga SP – Sistema de Informações Gerenciais de Acidentes de Trânsito do Estado de São Paulo**. Ranking de óbitos por município. Disponível em: `https://www.infosiga.sp.gov.br/#ranking`. Acesso em: 5 abr. 2025.

SÃO PAULO (Estado). Departamento Estadual de Trânsito. **Infosiga SP – Sistema de Informações Gerenciais de Acidentes de Trânsito do Estado de São Paulo**. Ranking de óbitos por município. Disponível em: `https://www.infosiga.sp.gov.br/#ranking`. Acesso em: 5 abr. 2025.

BASTOS, L. **Multas por uso de celular ao volante lideram estatística em Bauru**. Disponível em: https://sampi.net.br/bauru/noticias/2772548/bauru-e-regiao/2023/07/multas-por-uso-de-celular-ao-volante-lideram-estatistica-em-bauru. Acesso em: 6 abr. 2025.

BRASIL. **Lei nº 9.503, de 23 de setembro de 1997**. Institui o Código de Trânsito Brasileiro. Diário Oficial da União, Brasília, DF, 24 set. 1997. Disponível em: `https://www.planalto.gov.br/ccivil_03/leis/l9503.htm`. Acesso em: 6 abr. 2025.

PREFEITURA MUNICIPAL DE LENÇÓIS PAULISTA. **Município implanta Programa de Trânsito Faixa Viva**. Lençóis Paulista, 7 fev. 2023. Disponível em: https://www.lencoispaulista.sp.gov.br/v2/noticia/7583/municipio-implanta-programa-de-transito-faixa-viva.html. Acesso em: 6 abr. 2025.

HASHSTUDIOZ. **IoT Based Automatic Vehicle Accident Detection and Rescue System. 2024.** Disponível em: https://www.hashstudioz.com/blog/iot-based-automatic-vehicle-accident-detection-and-rescue-system/. Acesso em: 6 abr. 2025.

PEREIRA, Rafael H. M.; GONÇALVES, Daniel R.; BRAGA, Carlos Kaue V. **Mortes no trânsito: uma análise das informações do Datasus entre 1996 e 2017**. Brasília: Ipea, jul. 2020. Disponível em: `https://www.ipea.gov.br/atlasviolencia/arquivos/artigos/7018-td2565.pdf`. Acesso em: 6 abr. 2025.

PMC. **Smart Roads for Autonomous Accident Detection and Warnings. 2022.** Disponível em: https://pmc.ncbi.nlm.nih.gov/articles/PMC8953218/. Acesso em: 6 abr. 2025.

TRID. **GIS-Based Accident Location and Analysis System (GIS-ALAS). 1998.** Disponível em: `https://trid.trb.org/View/484578`. Acesso em: 6 abr. 2025.

SCIENCEDIRECT. **Geographical Information Systems Aided Traffic Accident Analysis. 2007.** Disponível em: `https://www.sciencedirect.com/science/article/abs/pii/S0001457507000863`. Acesso em: 6 abr. 2025.

JCNET. **Frota de Bauru envelhece e veículos com mais de 11 anos registram alta. 2024.** Disponível em: https://sampi.net.br/bauru/noticias/2811364/bauru-e-regiao/2024/01/frota-de-bauru-envelhece-e-veiculos-com-mais-de-11-anos-registram-alta. Acesso em: 6 abr. 2025.

G1 BAURU. **Número de mortes no trânsito de Bauru cresce 20% em 2024; motociclistas são maioria. 2025.** Disponível em: https://g1.globo.com/sp/bauru-marilia/noticia/2025/01/08/numero-de-mortes-no-transito-de-bauru-cresce-20percent-em-2024-motociclistas-sao-maioria.ghtml. Acesso em: 6 abr. 2025.

FLASK. **Documentação oficial**. Disponível em: `https://flask.palletsprojects.com/en/latest/`. Acesso em: 1 jul. 2025.

DJANGO. **Documentação oficial**. Disponível em: `https://docs.djangoproject.com/en/stable/`. Acesso em: 1 jul. 2025.

NODE.JS COM EXPRESS. **Documentação oficial do Express**. Disponível em: `https://expressjs.com/`. Acesso em: 1 jul. 2025.

PYTHON 3. **Documentação oficial**. Disponível em: `https://docs.python.org/3/`. Acesso em: 1 jul. 2025.

PANDAS. **Documentação oficial**. Disponível em: `https://pandas.pydata.org/docs/`. Acesso em: 1 jul. 2025.

NUMPY. **Documentação oficial**. Disponível em: `https://numpy.org/doc/`. Acesso em: 1 jul. 2025.

SCIKIT-LEARN. **Documentação oficial**. Disponível em: `https://scikit-learn.org/stable/documentation.html`. Acesso em: 1 jul. 2025.

XGBOOST. **Documentação oficial**. Disponível em: `https://xgboost.readthedocs.io/en/stable/`. Acesso em: 1 jul. 2025.

GEOPANDAS. **Documentação oficial**. Disponível em: `https://geopandas.org/en/stable/docs.html`. Acesso em: 1 jul. 2025.

PYTEST. **Documentação oficial**. Disponível em: `https://docs.pytest.org/en/latest/`. Acesso em: 1 jul. 2025.

FASTAPI. **Documentação oficial**. Disponível em: `https://fastapi.tiangolo.com/`. Acesso em: 1 jul. 2025.

POSTGRESQL. **Documentação oficial**. Disponível em: `https://www.postgresql.org/docs/`. Acesso em: 1 jul. 2025.

POSTGIS. **Documentação oficial**. Disponível em: https://postgis.net/documentation/. Acesso em: 1 jul. 2025.

REACT. **Documentação oficial**. Disponível em: `https://react.dev/learn`. Acesso em: 1 jul. 2025.

ANGULAR. **Documentação oficial**. Disponível em: `https://angular.io/docs`. Acesso em: 1 jul. 2025.

JAVASCRIPT. **Documentação oficial**. Disponível em: `https://developer.mozilla.org/pt-BR/docs/Web/JavaScript`. Acesso em: 1 jul. 2025.

VUE.JS 3. **Documentação oficial**. Disponível em: https://vuejs.org/guide/introduction.html. Acesso em: 1 jul. 2025.

AXIOS. **Documentação oficial**. Disponível em: `https://axios-http.com/docs/intro`. Acesso em: 1 jul. 2025.

LEAFLET.JS. **Documentação oficial**. Disponível em: https://leafletjs.com/reference.html. Acesso em: 1 jul. 2025.

CHART.JS. **Documentação oficial**. Disponível em: `https://www.chartjs.org/docs/latest/`. Acesso em: 1 jul. 2025.

SÃO PAULO (Estado). **Tabela de estaçõesl**. Disponível em: `https://tempo.inmet.gov.br/TabelaEstacoes/A001`. Acesso em: 1 jul. 2025.

BREIMAN, Leo. Random forests. **Machine Learning**, v. 45, n. 1, p. 5-32, 2001.

CHEN, Tianqi; GUESTRIN, Carlos. XGBoost: A Scalable Tree Boosting System. **Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. San Francisco, 2016. p. 785-794.

HAYKIN, Simon. **Neural networks and learning machines**. 3. ed. Upper Saddle River: Pearson Education, 2009.

OPEN-METEO. **Open-Meteo Weather API**. Disponível em: `https://open-meteo.com`. Acesso em: 5 nov. 2025.

OVERPASS API. In: **OpenStreetMap Wiki**. Disponível em: `https://wiki.openstreetmap.org/wiki/Overpass_API`. Acesso em: 6 nov. 2025.

CHAWLA, N. V. **Data mining para conjuntos de dados desbalanceados: uma visão geral**. In: MENDEZ, L.; BEZRUKOV, N. (Org.). Manual de Mineração de Dados e Descoberta de Conhecimento. New York: Springer, 2005.

YE, L. et al. **Previsão da severidade de lesões em acidentes considerando o desbalanceamento de dados: uma rede adversária generativa de Wasserstein com abordagem de penalidade de gradiente**. Accident Analysis & Prevention, Amsterdam, v. 192, p. 107271, 2023. Disponível em: `https://doi.org/10.1016/j.aap.2023.107271`. Acesso em: 27 out. 2025.

DAVIS, J. **Goadrich, M. (2006). "The relationship between Precision-Recall and ROC curves"**. Proceedings of the 23rd international conference on Machine learning. `https://www.researchgate.net/publication/215721831_The_Relationship_Between_Precision-Recall_and_ROC_Curves`. Acesso em: 27 out. 2025.

ESMALIFARD, A. et al. **Uma estrutura de fusão para previsão de acidentes em tempo real: uma estratégia consciente do desbalanceamento para sistemas de prevenção de colisões**. Transportation Research Part C: Emerging Technologies, Amsterdam, v. 118, p. 102708, 2020. Disponível em: `https://doi.org/10.1016/j.trc.2020.102708`. Acesso em: 28 out. 2025.

SAITO, T.; REHMSMEIER, M. **The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets**. PLOS ONE, San Francisco, v. 10, n. 3, p. e0118432, 2015. Disponível em: `https://doi.org/10.1371/journal.pone.0118432`. Acesso em: 03 dez. 2025.