Relatório Técnico – Projeto de Previsão de Nível de Obesidade

1. Introdução e Objetivo do Projeto

O objetivo fundamental deste projeto é desenvolver um modelo preditivo capaz de estimar o nível de obesidade de indivíduos com base em características físicas, hábitos alimentares e variáveis comportamentais.

A iniciativa está inserida no contexto de desenvolvimento de competências em Data Analytics e Machine Learning, abrangendo todas as etapas de um pipeline moderno de dados: extração, tratamento, preparação dos dados, construção, avaliação e apresentação de um modelo preditivo funcional.

A justificativa do projeto reside na importância de compreender como fatores de estilo de vida e padrões alimentares influenciam diretamente o risco de obesidade. Dessa forma, o modelo proposto visa não só atingir alto desempenho preditivo, mas também fornecer subsídios analíticos para decisões estratégicas em saúde pública, nutrição e bem-estar.

O desenvolvimento foi pautado por uma visão de **Data Science aplicada**, integrando conceitos de engenharia de dados, estatística e aprendizado de máquina em um pipeline reprodutível e escalável.

2. Arquitetura de Dados e Ferramentas Selecionadas

Para garantir organização, reprodutibilidade e escalabilidade, foi projetada uma arquitetura de dados modular, inspirada na **Arquitetura Medalhão (Medallion Architecture)**. Essa abordagem estrutura o fluxo de dados em camadas lógicas (Bronze, Silver e Gold), permitindo gestão eficiente das transformações e assegurando qualidade, rastreabilidade e governança ao longo do ciclo analítico.

A implementação ocorreu em ambiente local, utilizando o PostgreSQL como base de dados principal e o VS Code como IDE de desenvolvimento em Python, proporcionando flexibilidade, transparência e baixo custo operacional.

O projeto está organizado em pastas modulares, conforme as boas práticas de Engenharia de Dados:

- src/,
- data/,
- models/,
- reports/,
- docs/.

Figura 1 – Estrutura modular do projeto e organização dos diretórios principais

Figura 1 - Estrutura modular do projeto e organização dos diretórios principais

A Figura 1 apresenta a estrutura modular do projeto, evidenciando a separação das responsabilidades entre os diretórios de dados, código-fonte, modelos, relatórios e documentação.

A Figura 2, por sua vez, ilustra o fluxo analítico adotado, estruturado segundo a Arquitetura Medalhão, que organiza o ciclo de vida dos dados desde a ingestão até a disponibilização para consumo no dashboard interativo.

2.1. Desenho da Arquitetura

A estrutura do pipeline de dados contempla desde a ingestão bruta e armazenamento dos dados brutos no PostgreSQL (Camada Bronze), passando pelo pré-processamento e exploração no VS Code com Python (Camada Silver), até a geração do modelo e do dashboard interativo em Streamlit (Camada Gold).

Essa arquitetura híbrida — combinando **banco de dados relacional (PostgreSQL)** e **ambiente analítico em Python** — assegura persistência, rastreabilidade e flexibilidade para análises e reprocessamentos futuros.

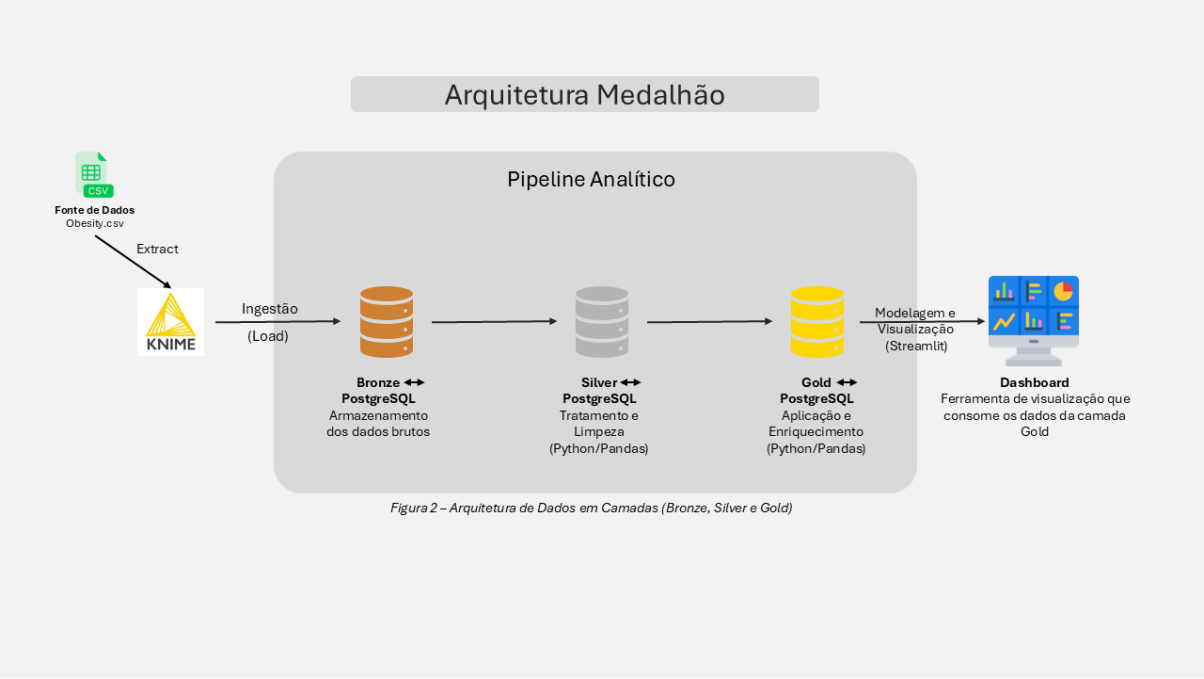

Figura 2 – Arquitetura de Dados em Camadas (Bronze, Silver e Gold)

2.2. Lógica e Justificativa da Arquitetura

A arquitetura modular garante clareza e reprodutibilidade em todas as etapas do projeto:

- **Camada Bronze (PostgreSQL):** Responsável pelo armazenamento dos dados brutos extraídos do arquivo obesity.csv, carregados por meio do **KNIME**, que executa o *Extract & Load* no banco de dados.
  Essa camada mantém os dados em seu estado original, funcionando como a "**fonte única da verdade**" (*Single Source of Truth*), possibilitando auditoria e reprocessamentos a qualquer momento.
- **Camada Silver (Python + PostgreSQL):** Etapa de pré-processamento e limpeza dos dados diretamente a partir do banco de dados. São aplicadas técnicas de tratamento de valores ausentes, padronização de tipos, normalização e criação de variáveis derivadas (ex.: IMC).
  O dataset tratado é então salvo como uma nova tabela no PostgreSQL, garantindo persistência e consistência das transformações.
- **Camada Gold (Python + PostgreSQL):** Responsável pelo enriquecimento dos dados e preparação para a modelagem preditiva. Nessa camada, ocorre a separação dos conjuntos de treino e teste, além da seleção das variáveis mais relevantes.
  Os resultados são armazenados em uma nova tabela ("Gold"), que serve como base para o **dashboard interativo desenvolvido em Streamlit**, garantindo acesso rápido e estruturado às previsões e métricas de desempenho do modelo.

Nota Técnica – Padronização, Persistência e Reprodutibilidade:
Durante a etapa de ingestão, o **KNIME** foi utilizado para realizar o *Extract & Load* do dataset original no **PostgreSQL**, aplicando o **mapeamento de colunas conforme o dicionário de dados oficial**.

Essa harmonização semântica (ex.: FAF → frequencia_semanal_atividade_fisica) visa **melhorar a legibilidade e reduzir ambiguidades analíticas**, sem alterar a estrutura ou o conteúdo informacional dos dados.

Para assegurar **reprodutibilidade total**, uma cópia do arquivo original (obesity.csv) foi mantida na pasta data/raw/. Assim, caso o acesso ao banco de dados PostgreSQL não seja possível, o pipeline pode ser **replicado localmente** sem perda de fidelidade aos resultados.

Essa estratégia combina **persistência (via PostgreSQL)** e **portabilidade (via CSV local)**, mantendo a integridade conceitual da **Arquitetura Medalhão** e garantindo que o projeto possa ser executado em diferentes ambientes com consistência.

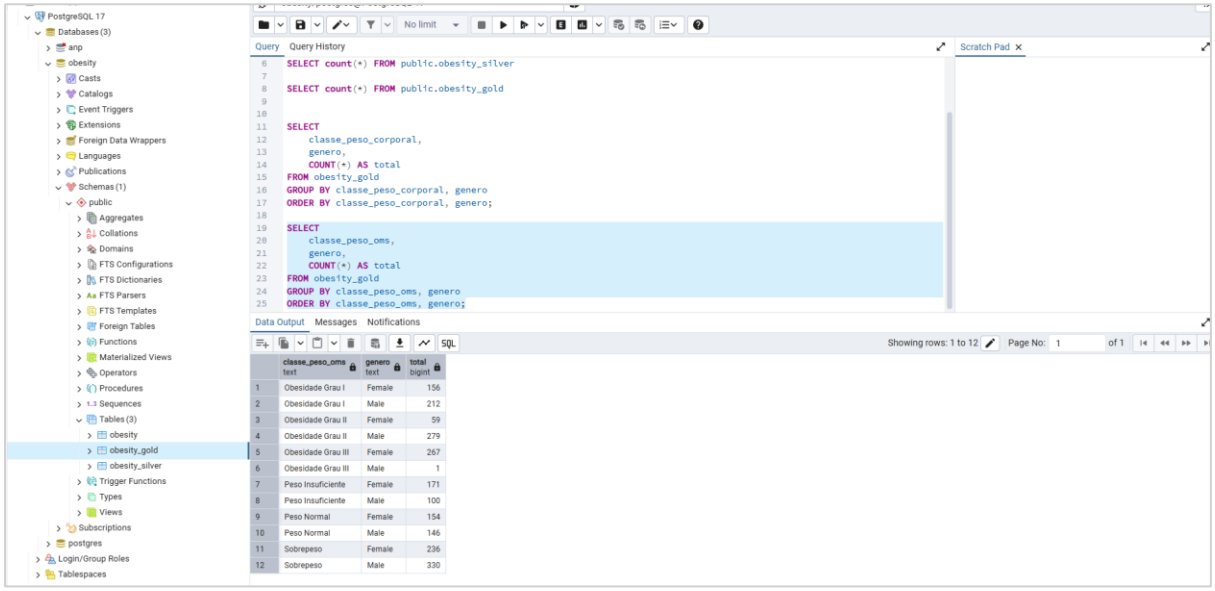

Figura 3 - Estrutura final das tabelas no banco de dados PostgreSQL.

Complementando a arquitetura de dados, a estrutura modular do projeto assegura separação clara de responsabilidades:

- **src/**: contém todo o código-fonte modularizado da aplicação, incluindo scripts para treinamento de modelos (src/models) e a aplicação interativa (src/app).
- **models/**: esta pasta é designada para armazenar os artefatos de modelo final treinados e serializados (ex.: arquivos .pkl ou .joblib). O aplicativo Streamlit carrega o modelo desta pasta para realizar as previsões, garantindo que o app não precise retreinar o modelo a cada execução.
- **reports/**: armazena as saídas analíticas estáticas geradas durante o desenvolvimento. A subpasta figures/ guarda visualizações-chave, como gráficos da análise exploratória, matrizes de confusão e gráficos SHAP de interpretabilidade, facilitando a consulta e a apresentação dos resultados.

Essa arquitetura permite reprodutibilidade total, possibilitando que qualquer etapa seja refeita de forma independente, garantindo rastreabilidade e governança sobre as transformações realizadas.

2.3. Justificativa da Ferramenta de Visualização

A camada de apresentação foi desenvolvida com Streamlit, biblioteca Python voltada para criação de interfaces analíticas interativas. A escolha se baseou na integração direta com o pipeline Python, baixo custo e simplicidade de deploy (via Streamlit

Cloud) e na capacidade de gerar dashboards dinâmicos com gráficos explicativos, métricas e interpretações do modelo (explainability, via SHAP).

Assim, o Streamlit atua como o front-end analítico, traduzindo resultados técnicos em insights acessíveis e visualmente claros. Durante o deploy, a aplicação consome os dados da camada Gold exportada em CSV, garantindo acessibilidade a todos os avaliadores, independentemente da conexão direta ao banco.

Figura 4 – Dashboard interativo desenvolvido em Streamlit para visualização dos resultados.

3. Desenvolvimento do Projeto: Da Ingestão à Modelagem

O desenvolvimento foi guiado por uma abordagem incremental, em que cada módulo do pipeline foi construído e validado de forma independente, garantindo rastreabilidade e fácil manutenção do código.

O pipeline foi estruturado em quatro etapas principais:

1. **Ingestão de Dados (Camada Bronze):**
   O dataset original (obesity.csv) foi carregado para o banco de dados PostgreSQL utilizando o KNIME como ferramenta de *Extract & Load*.
   Essa abordagem permite garantir persistência e integridade dos dados, além de facilitar futuras atualizações da base sem necessidade de manipulação manual de arquivos.
   Uma cópia do CSV foi mantida em data/raw/ apenas para garantir reprodutibilidade total do projeto em ambiente local.
2. **Tratamento e Pré-Processamento:** Nessa etapa, os dados armazenados na camada Bronze foram conectados ao **VS Code via Python** utilizando bibliotecas como SQLAlchemy e Pandas. Foram aplicadas rotinas de limpeza, tratamento de valores ausentes, padronização de tipos, encoding de variáveis categóricas e criação de atributos derivados (ex.: cálculo de IMC). O conjunto resultante foi gravado novamente no PostgreSQL como **tabela Silver**, consolidando os dados prontos para exploração e modelagem.

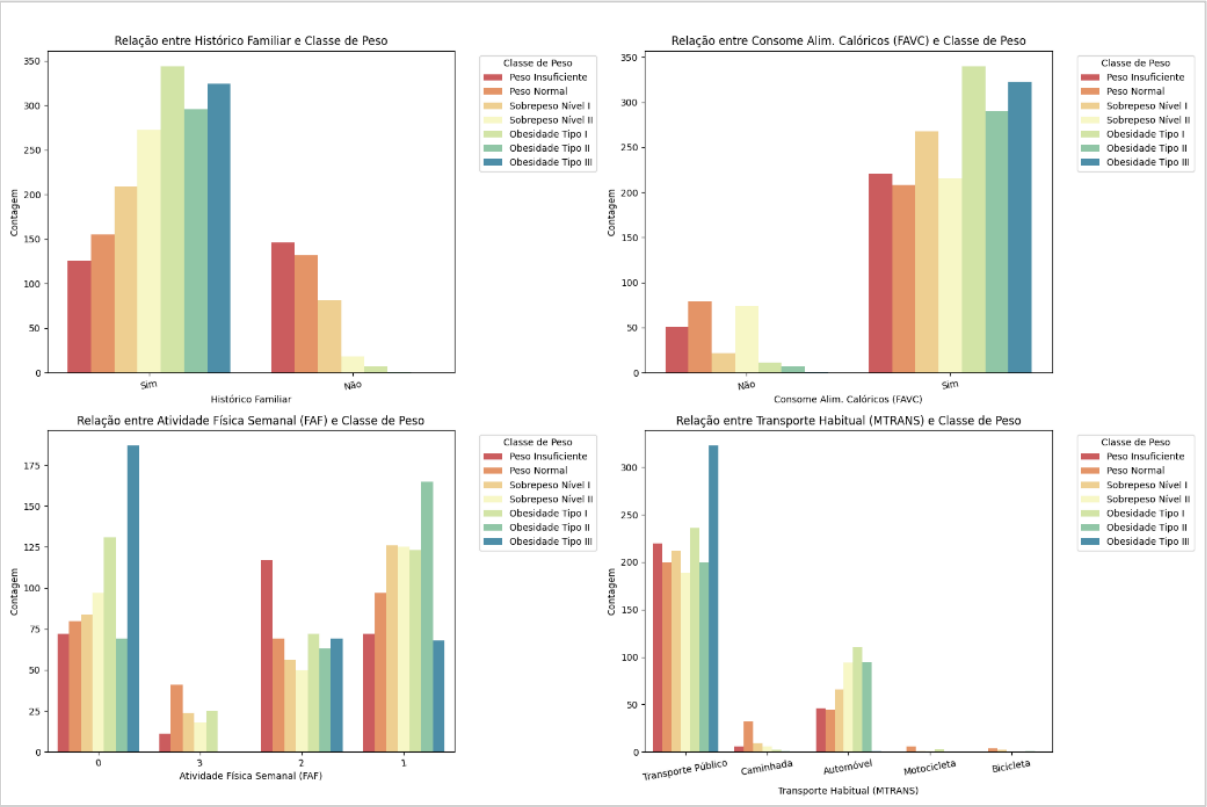

Figura 5 – Análise exploratória: countplots da relação entre hábitos específicos e classes de peso.

3. **Treinamento e Avaliação de Modelos:** A partir dos dados processados da camada Silver, foram testados diferentes algoritmos de *Machine Learning* supervisionado — **Random Forest e XGBoost** — utilizando a biblioteca scikit-learn. A seleção do modelo final considerou a **acurácia mínima de 75%**, complementada pelas métricas **F1-Score, Precision, Recall e Matriz de Confusão**. O modelo final foi serializado (random_forest_pipeline.joblib) e armazenado na pasta models/. Paralelamente, o dataset final da camada Gold foi **exportado em formato CSV** (data/processed/) para permitir sua integração com o dashboard.

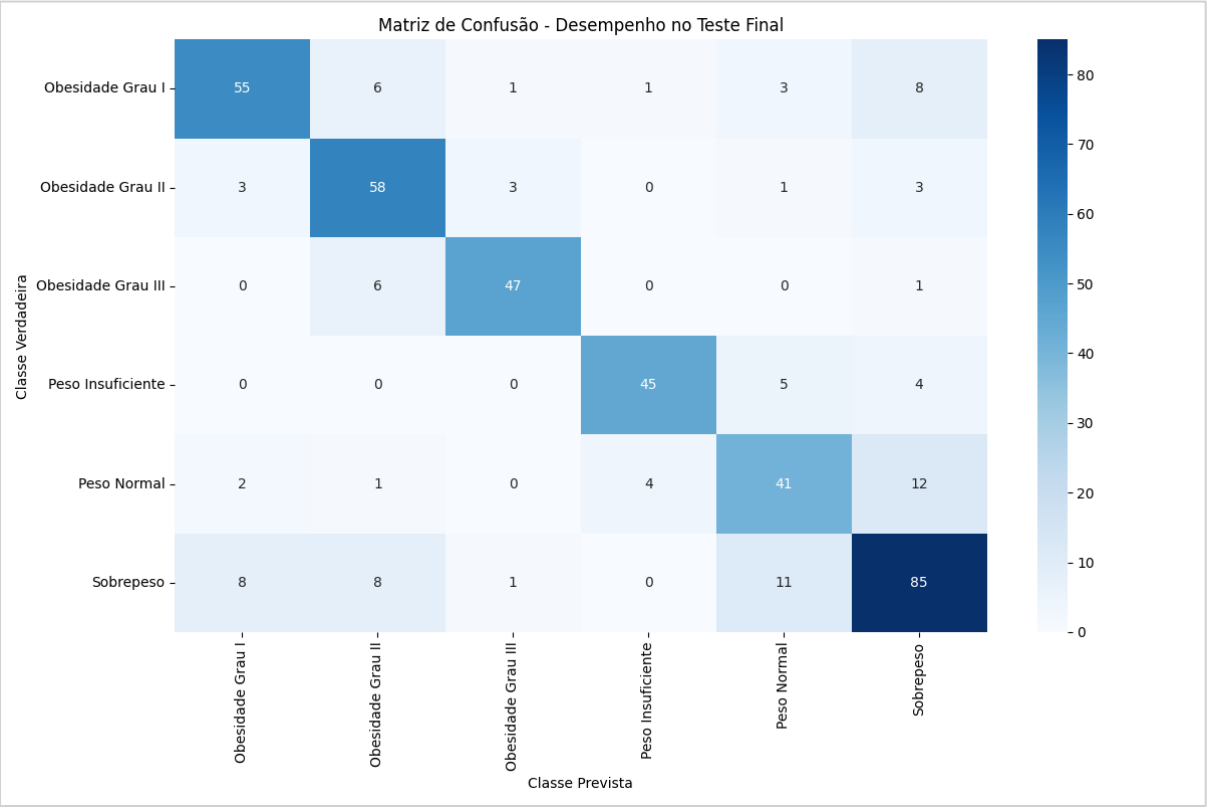

Figura 6 – Avaliação do modelo: matriz de confusão

4. **Apresentação e Interpretação:** Os resultados foram integrados em um **dashboard desenvolvido em Streamlit**, alimentado diretamente pelo **dataset CSV da camada Gold**. Essa decisão garante **total acessibilidade e portabilidade**, permitindo que o dashboard seja publicado no **Streamlit Cloud** e acessado pelos avaliadores **sem necessidade de conexão ao banco de dados PostgreSQL**. A interface permite a visualização interativa de métricas de desempenho, gráficos explicativos (SHAP, Matriz de Confusão) e previsões em tempo real, traduzindo os resultados técnicos em *insights* interpretáveis.

Esse pipeline assegura a fluidez entre todas as etapas — da ingestão à visualização —, consolidando uma **solução analítica de ponta a ponta**, com governança, desempenho e foco na interpretabilidade dos resultados.

4. Descoberta e Ajuste da Variável Alvo

Durante a construção da visão analítica interativa (em Streamlit), foi identificada uma inconsistência relevante na variável-alvo original (**classe_peso_corporal**), responsável pela classificação dos níveis de obesidade.

Ao analisar a distribuição de IMC por gênero, observou-se que indivíduos com o mesmo valor de IMC recebiam classificações distintas. Por exemplo:

- Homens com IMC ≈ 37 eram classificados como Obesidade Tipo II
- Mulheres com IMC ≈ 37 eram classificadas como Obesidade Tipo III

Essa divergência indicou a presença de vazamento de dados e viés no dataset original, uma vez que a classificação dependia implicitamente do gênero — variável que não deve interferir nos critérios de IMC definidos por órgãos de saúde.

Diante desse cenário, optou-se por revisar a variável-alvo, substituindo-a por uma nova coluna denominada **classe_peso_oms**, criada com base exclusivamente no valor de IMC e nas faixas de referência estabelecidas pela Organização Mundial da Saúde (OMS):

ClassificaçãoFaixa de IMC (kg/m^2)

- Peso Insuficiente < 18.5
- Peso Normal 18.5 – 24.9
- Sobrepeso 25.0 – 29.9
- Obesidade Grau I 30.0 – 34.9
- Obesidade Grau II 35.0 – 39.9
- Obesidade Grau III ≥ 40.0

A criação dessa nova variável elimina o viés identificado e garante padronização científica e imparcialidade no processo de modelagem preditiva.

Para fins de rastreabilidade, ambas as variáveis foram mantidas na camada Silver e comparadas antes da atualização da camada Gold. O gráfico abaixo ilustra essa diferença:

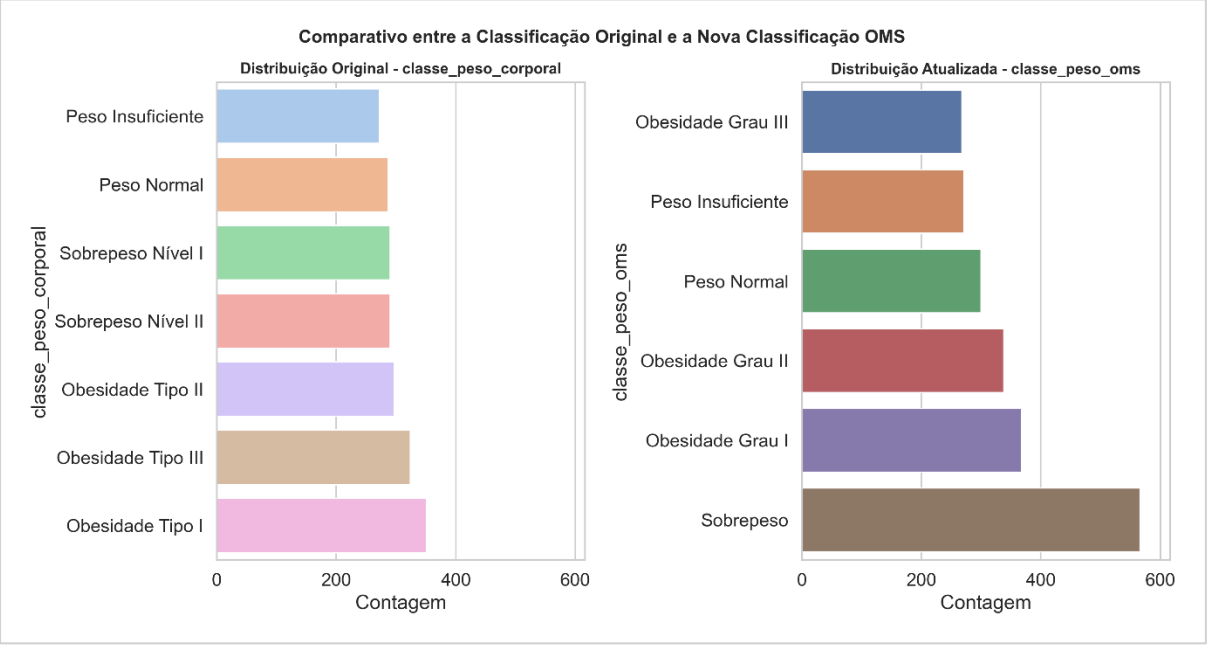

Figura 7 – Comparativo entre Classificação Original e Classificação OMS.

Essa nova abordagem permitiu corrigir o problema identificado, consolidando o critério OMS como padrão único e reproduzível de classificação.

5. Análise dos Resultados e Insights

O pipeline de treinamento foi executado nos dados corrigidos (utilizando classe_peso_oms como alvo) e validado de forma robusta. Após uma comparação via Validação Cruzada K-Fold, o modelo Random Forest foi selecionado como o modelo com melhor desempenho, superando o XGBoost com uma acurácia média de 75.35%. Após o re-treino final, o modelo foi avaliado no conjunto de teste (20% dos dados, "trancados" para a prova final), atingindo os seguintes resultados:

- **Acurácia Final: 78.25%** (Superando o requisito de 75% do projeto).
- **F1-Score (Ponderado):** 0.78 (Indicando um excelente equilíbrio geral entre Precisão e Recall).

A análise detalhada do classification_report e da Matriz de Confusão (Figura 6) revelou os pontos fortes e fracos do modelo:

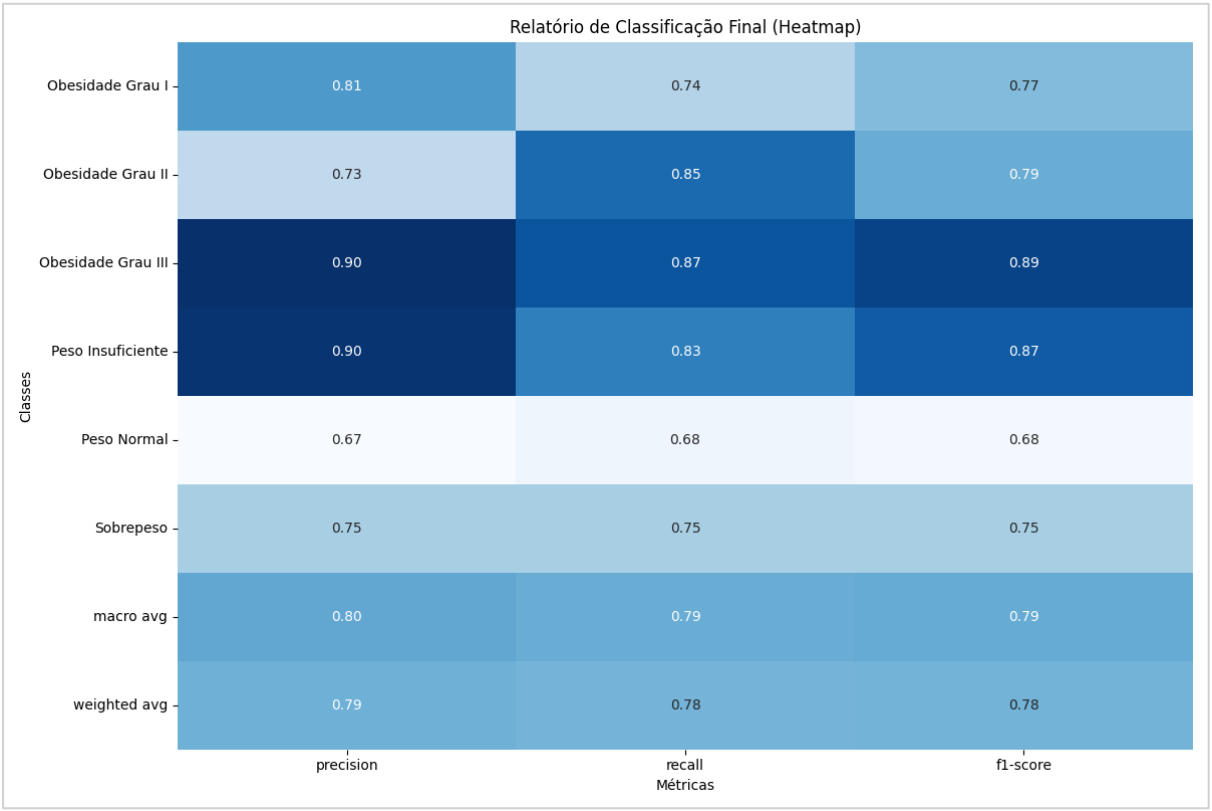

Figura 8 – Relatório de Classificação do modelo final, detalhando Precision, Recall e F1-Score por classe.

1. **Alta Confiabilidade nos Extremos:** O modelo demonstrou ser extremamente eficaz na identificação das classes de ponta, atingindo um F1-Score de 0.89 para 'Obesidade Grau III' (Classe 2) e 0.87 para 'Peso Insuficiente' (Classe 3). Isso o torna uma ferramenta de triagem muito confiável para os casos mais críticos.

2. **Desafio de Classificação:** O principal desafio do modelo continua sendo a distinção entre 'Peso Normal' (Classe 4) e 'Sobrepeso' (Classe 5). No entanto, o refinamento das *features* melhorou o desempenho nesta classe mais difícil, elevando seu **F1-Score para 0.68**

Os gráficos de importância de variáveis (SHAP Values) evidenciaram que as features de engenharia criadas na camada Silver — indice_estilo_vida e indice_risco_alimentar — foram os fatores mais determinantes para o desempenho preditivo, superando o impacto de variáveis comportamentais isoladas.

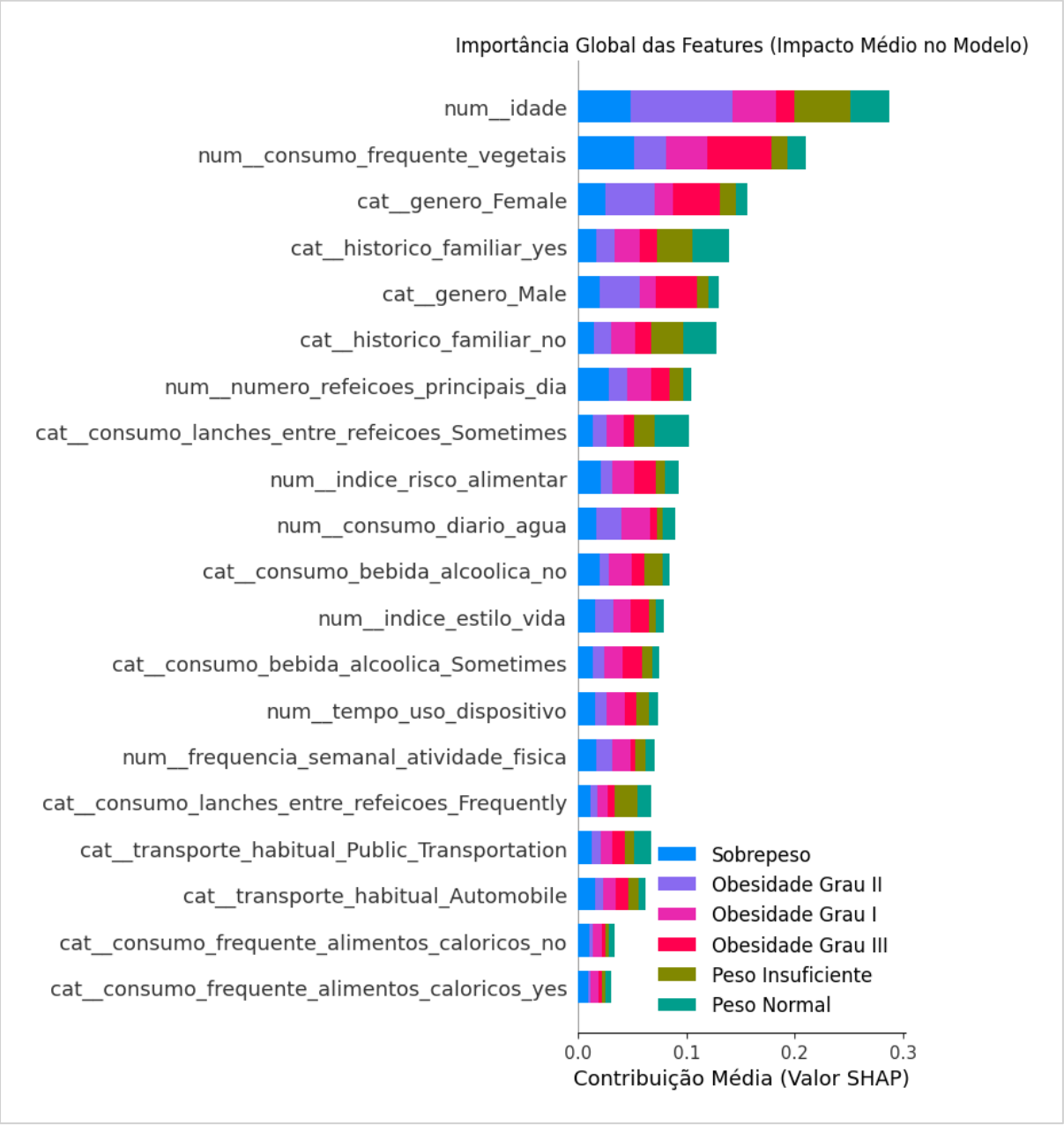

Figura 9 – Interpretação dos resultados do modelo via SHAP Values.

6. Aplicabilidade e Recomendações Estratégicas

O modelo desenvolvido, com acurácia validada de **78.25%** e forte desempenho na identificação de casos graves (F1-Score de 0.89), pode servir de base para ferramentas preditivas de risco individual integradas a sistemas de saúde.

Dado que o modelo se mostrou altamente confiável nos extremos, ele é particularmente aplicável como uma ferramenta de triagem (apoio à decisão), ajudando profissionais de saúde a identificar rapidamente pacientes em 'Obesidade Grau III' que necessitam de intervenção imediata, ao mesmo tempo em que sinaliza pacientes na fronteira 'Peso Normal'/'Sobrepeso' para acompanhamento preventivo.

A arquitetura modular e escalável permite adaptação fácil para ambientes em nuvem, e o deploy via Streamlit válida a solução como um produto analítico ponta-a-ponta.

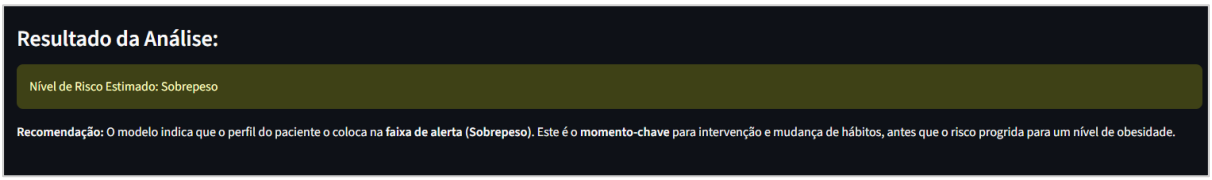

Figura 10 – Exemplo de insight visual disponível no dashboard.

7. Conclusão

O projeto atingiu todos os objetivos propostos, entregando um pipeline completo de ciência de dados que abrange extração, tratamento, modelagem preditiva, estruturação de dados em camadas (Bronze, Silver e Gold) e desenvolvimento de dashboard interativo para interpretação dos resultados.

O diferencial do projeto foi a identificação e correção de um *data leakage* crítico na variável-alvo original. Ao invés de prosseguir com um modelo de métricas falsamente infladas, foi realizada uma rigorosa engenharia de features para **criar um alvo cientificamente válido** (classe_peso_oms), garantindo a integridade da análise.

Mesmo executado em ambiente local, o projeto seguiu padrões profissionais de arquitetura e engenharia de dados, garantindo reprodutibilidade, clareza e escalabilidade. O resultado é um produto analítico **robusto e validado**, capaz de gerar insights relevantes e acionáveis sobre obesidade e hábitos de vida.

Dessa forma, o projeto consolida-se como uma solução de análise preditiva completa, integrando boas práticas de engenharia de dados, modelagem estatística e comunicação analítica, servindo como base escalável para estudos futuros sobre determinantes da obesidade.