INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE RONDÔNIA
CAMPUS COLORADO DO OESTE
CURSO SUPERIOR DE TECNOLOGIA EM GESTÃO PÚBLICA

ROBSON PEREIRA DA SILVA

PROJETO DE ANÁLISE DE DADOS EDUCACIONAIS COM PYTHON: UM ESTUDO DE CASO SOBRE DESEMPENHO ACADÊMICO E APRENDIZADO DE MÁQUINA

COLORADO DO OESTE
2025

ROBSON PEREIRA DA SILVA

PROJETO DE ANÁLISE DE DADOS EDUCACIONAIS COM PYTHON: UM ESTUDO DE CASO SOBRE DESEMPENHO ACADÊMICO E APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado ao curso Superior de Tecnologia em Gestão Pública do Instituto Federal de Educação, Ciência e Tecnologia de Rondônia (IFRO) - *Campus* Colorado do Oeste, como requisito parcial para obtenção do Título de Tecnólogo em Gestão Pública. Sob a orientação do Prof. Me. Glicerinaldo de Sousa Gomes.

COLORADO DO OESTE

2025

Ficha catalográfica elaborada pelo Sistema Gerador de Ficha Catalográfica do IFRO.

Silva, Robson Pereira da.
Projeto de análise de dados educacionais com Python: um estudo de caso sobre desempenho acadêmico e aprendizado de máquina / Robson Pereira da Silva. - Colorado do Oeste, 2025.
69 f. : il.

Orientador(a): Prof. Me. Glicerinaldo de Sousa Gomes.

Trabalho de Conclusão de Curso (Superior de Tecnologia em Gestão Pública) – Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO, Colorado do Oeste, 2025.

1. Análise de dados. 2. Aprendizado de máquina. 3. Gestão em educação pública. 4. Tomada de decisão. I. Gomes, Glicerinaldo de Sousa (orient.). II. Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - IFRO. III. Título.

Bibliotecário(a) Responsável: Roseni Santos Rodrigues, CRB-11/916

ROBSON PEREIRA DA SILVA

PROJETO DE ANÁLISE DE DADOS EDUCACIONAIS COM PYTHON: UM ESTUDO DE CASO SOBRE DESEMPENHO ACADÊMICO E APRENDIZADO DE MÁQUINA

Trabalho de Conclusão de Curso apresentado ao Curso Superior de Tecnologia em Gestão Pública, do Instituto Federal de Educação, Ciência e Tecnologia de Rondônia - *Campus* Colorado do Oeste, como parte das exigências para obtenção do título de Tecnólogo em Gestão Pública, sob a orientação do Prof. Me. Glicerinaldo de Sousa Gomes

Aprovado em: 30/10/2025, pela banca examinadora.

gov.br Documento assinado digitalmente
GLICERINALDO DE SOUSA GOMES
Data: 30/10/2025 21:44:54-0300
Verifique em https://validar.iti.gov.br

Glicerinaldo de Sousa Gomes
Orientador(a)

gov.br Documento assinado digitalmente
IVELTYMA ROOSEMALEN PASSOS IBIAPINA
Data: 31/10/2025 18:02:14-0300
Verifique em https://validar.iti.gov.br

Iveltyma Roosemalen Passos Ibiapina Examinador

gov.br Documento assinado digitalmente
JOSILEIDE CARMEM BELO GOMES
Data: 17/11/2025 22:42:37-0300
Verifique em https://validar.iti.gov.br

Josileide Carmem Belo Gomes
Examinadora

PROJETO DE ANÁLISE DE DADOS EDUCACIONAIS COM PYTHON: UM ESTUDO DE CASO SOBRE DESEMPENHO ACADÊMICO E APRENDIZADO DE MÁQUINA

Robson Pereira da Silva[1]

Glicerinaldo de Sousa Gomes[2]

RESUMO

Este trabalho tem como objetivo aplicar técnicas de Análise de Dados combinado com um Modelo de Aprendizagem de Máquina através de uma pesquisa quantitativa, exploratória e aplicada. O projeto foi desenvolvido através da Linguagem de Programação Python, a partir de uma base de dados simulada para fins didáticos, contendo 2.392 registros estudantis, sugerindo uma análise exploratória de dados educacionais dos estudantes tais como, variáveis demográficas, acadêmicas e comportamentais, gerando insights a partir dos dados e também um modelo de Inteligência Artificial que consiga ler as informações dos estudantes e dizer automaticamente a classificação antecipada de suas notas. A partir da análise de dados, será possível evidenciar o impacto de fatores como faltas escolares, tempo de estudo semanal e apoio familiar no desempenho dos alunos. Os resultados apontam que a análise de dados e modelos de aprendizado de máquina, podem ser uma aliada estratégica na gestão educacional, permitindo identificar padrões de risco para tomada de decisões mais assertivas baseadas em evidências através de dados, obtendo resultados mais eficazes para o ambiente escolar.

Palavras-chave: Análise de Dados; Aprendizado de Máquina; Gestão em Educação Pública; Tomada de Decisão.

ABSTRACT

This work aims to apply Data Analysis techniques combined with a Machine Learning Model through quantitative, exploratory, and applied research. The project was developed using the Python Programming Language, based on a simulated database for didactic purposes, containing 2,392 student records, suggesting an exploratory analysis of students' educational data such as demographic, academic, and behavioral variables, generating insights from the data and also an Artificial Intelligence model that can read student information and automatically predict their grades. From the data analysis, it will be possible to highlight the impact of factors such as school absences, weekly study time, and family support on student performance. The results indicate that data analysis and machine learning models can be a strategic ally in educational management, allowing the identification of risk patterns for more assertive decision-making based on data evidence, obtaining more effective results for the school environment.

Keywords: Data Analysis; Machine Learning; Public Education Management; Decision Making.

[1] Aluno do curso de Tecnologia em Gestão Pública do IFRO. E-mail: robsonprogramadordelphi@gmail.com

[2] Professor de TCC do curso de Tecnologia em Gestão Pública do IFRO e orientador do trabalho. Mestre em Gestão em Organizações Aprendentes, UFPB. E-mail: glicerinaldo@gmail.com.

1 INTRODUÇÃO

O setor público está se tornando cada vez mais digitalizado e dinâmico, esta evolução e a demanda por gestão pública baseada em dados, tem impulsionado a adoção de tecnologias inovadoras como a ciência de dados e a inteligência artificial (IA) na formulação de políticas públicas. Na esfera educacional, essas tecnologias podem oferecer possibilidades concretas para compreender padrões de desempenho estudantil e apoiar decisões mais estratégicas por parte de gestores.

Este trabalho busca explorar como ferramentas de análise e ciência de dados, aliadas a modelos de aprendizado de máquina desenvolvidos em Python[3], podem ser aplicadas ao contexto da educação pública brasileira para analisar e prever o desempenho acadêmico de estudantes do ensino médio. Com base em um conjunto de dados fictício, mas representativo e didático, foi desenvolvida uma Análise Exploratória (EDA) através da manipulação e visualização gráfica estatística, na busca de padrões baseadas nos dados coletados, e também um modelo de aprendizado de máquina capaz de classificar estudantes em faixas de notas, a partir de dados já analisados ou através da coleta de novos dados, obtendo a classificação antecipada das notas.

A proposta deste trabalho inspira-se na necessidade de modernização da gestão pública educacional buscando demonstrar que a análise de dados e modelos de aprendizado de máquina podem ser um recurso útil para a identificação precoce de estudantes em risco, possibilitando a elaboração de estratégias mais eficazes e antecipadas.

Baseado na análise dos dados da base de dados dados_estudantes.csv, foi possível avaliar que a evasão escolar e o baixo desempenho acadêmico ainda figuram entre os principais desafios da educação pública mundial, uma vez que a base de dados utilizada neste projeto foi elaborada em outro país. Iniciativas que promovam a análise inteligente de dados educacionais podem fortalecer as políticas de acompanhamento e suporte ao aluno, especialmente quando se consideram variáveis além do fator econômico, como comportamento, hábitos de estudo e apoio familiar.

Este trabalho tem por objetivo geral investigar como a aplicação de técnicas de análise de dados e modelos de aprendizado de máquina, utilizando a linguagem Python, pode contribuir para decisões estratégicas na gestão educacional e assim melhorar o

[3] Python é uma linguagem de programação de alto nível, interpretada de script, imperativa, orientada a objetos, funcional, de tipagem dinâmica e forte e foi lançada por Guido van Rossun em 1991.

desempenho acadêmico de estudantes da rede pública de ensino, através da análise exploratória sobre dados acadêmicos e comportamentais de estudantes, identificando os principais fatores (insights) que influenciam o desempenho escolar. Desenvolver um modelo de aprendizado de máquina capaz de prever e classificar estudantes por desempenho acadêmico e por fim avaliar o potencial de uso da análise de dados como ferramenta de apoio à gestão pública educacional.

O presente estudo justifica-se por contribuir com uma abordagem prática e tecnicamente fundamentada para a análise do desempenho escolar com base em dados simulados e posteriormente em dados reais e evidentes. O uso de Python e suas bibliotecas de ciência de dados (Pandas, Numpy, Matplotlib, Seaborn e Scikit-Learn), aproxima o campo da pesquisa das ferramentas utilizadas no mercado, ao mesmo tempo em que se alinha aos princípios da gestão pública orientada por dados. Além disso, a utilização de modelos de aprendizado de máquina pode ser vista como uma evolução no uso da tecnologia na gestão educacional, propondo soluções proativas em vez de reativas.

2 REFERENCIAL TEÓRICO

2.1 Ciência de dados e gestão pública

A ciência de dados tem se consolidado como uma ferramenta estratégica na gestão pública, proporcionando decisões baseadas em evidências através de dados com maior eficiência na gestão de políticas públicas. Segundo Provost e Fawcett (2013), a ciência de dados combina estatística, programação e conhecimento de domínio para extrair valor de grandes volumes de dados.

Nessa dinâmica, um governo baseado em dados, utiliza informações e análises inteligentes para fundamentar decisões estratégicas e operacionais, resultando em serviços públicos mais eficientes, personalizados, com políticas públicas mais eficazes e transparentes, que atendam as reais necessidades dos cidadãos, ou seja, ao invés de se basear em suposições, o governo adota uma cultura orientada a dados para entender a sociedade, otimizando recursos e melhorando a qualidade de vida da população.

Na gestão pública esse conceito se faz necessário para enfrentar problemas complexos que envolve a utilização de fatos e métricas, onde as decisões, a formulação de políticas públicas e a prestação de serviços são orientadas por evidências extraídas de dados brutos, ao invés de intuição ou experiência prática. A integração de dados enfatiza a

organização e a unificação de bases de dados governamentais para então permitir uma visão abrangente e sistêmica da realidade ocasionando a melhoria da eficiência e desempenho, onde o objetivo é otimizar a eficiência operacional, melhorando o desempenho das instituições públicas e alcançar melhores resultado para os cidadãos.

Por fim a inovação no ciclo de políticas públicas, onde a modelagem de sistemas, muitas vezes envolvendo inteligência artificial, é usada para alterar e acelerar o ciclo de produção de políticas, com interação entre humanos e máquinas na geração de conhecimento.

Davenport e Ronanki (2018) destacam que a inteligência artificial, aliada à ciência de dados, está transformando organizações públicas e privadas ao permitir decisões velozes e precisas. Essa transformação é evidenciada pelo conceito de governo baseado em dados (Janssen & Helbig, 2018), onde a análise preditiva e prescritiva orienta políticas públicas eficazes, através da coleta, análise e uso de dados fundamentais para tomada de decisões estratégicas e operacionais no setor público.

Assim sendo, os autores destacam um governo baseado em dados como cultura de gestão que utiliza a análise de dados mensuráveis para orientar a administração pública e a formulação de políticas bem informadas e eficazes.

2.2 Análise de dados educacionais

A análise de dados educacionais tem se destacado como uma poderosa ferramenta para avaliar o desempenho de estudantes. Este processo utiliza informações coletadas em diferentes níveis entre eles, aluno, escola e rede de ensino, para entender e melhorar a qualidade da educação. Essa análise envolve a coleta de dados como desempenho em avaliações, frequência, dados demográficos e socioeconômicos, utilizando técnicas estatísticas para identificar padrões, tendências e áreas de melhoria.

Segundo Baker e Yacef (2009), a tecnologia contribui para uma coleta de grande quantidade de dados educacionais de forma precisa e eficiente. A análise de dados também pode ajudar a personalizar o ensino, otimizar a alocação de recursos, desenvolver políticas mais eficazes e em última análise, aperfeiçoar a experiência de aprendizado dos alunos.

Luckesi (2002) argumenta que a avaliação de aprendizagem deve considerar o contexto do aluno, indo além de métricas quantitativas, para ele, avaliar é um ato rigoroso de acompanhamento da aprendizagem, pois é ela que permite tomar conhecimento do que se aprendeu e do que não se aprendeu e reorientar o educando para que supere suas dificuldades, na medida em que o que importa é aprender. Brito, Costa e Andrade (2020)

reforçam que fatores como apoio familiar, assiduidade e hábitos de estudo influenciam diretamente o desempenho escolar. Neste cenário, a análise de dados também pode ser utilizada para identificar as necessidades individuais dos alunos, padrões e tendências, revelando informações valiosas que podem influenciar o desempenho dos alunos.

A análise de dados educacionais deve continuar a evoluir e desempenhar um papel cada vez mais importante na melhoria e qualidade do ensino, a medida em que a tecnologia avança, através do uso de sistemas de gerenciamento de aprendizagem, plataformas online e aplicativos educacionais, teremos cada vez mais, um grande número de dados a serem analisados que permitirão um entendimento mais específico do processo educacional.

Siemens (2013) destaca a importância da utilização de técnicas de aprendizado de máquina (machine learning) e algoritmos de análise de dados para identificar padrões e tendências nos dados educacionais, com essas técnicas é possível descobrir informações valiosas podendo ser utilizadas na personalização do ensino de acordo com as necessidades de cada aluno.

Kulkarni (2016) discorre que a personalização do ensino é um dos benefícios mais significativos da análise de dados educacionais, com a análise de dados permitindo identificar alunos que possuem dificuldades em certas áreas e assim oferecer intervenções específicas para ajudá-los em seu processo de aprendizagem, e por outro lado identificar alunos com habilidades notáveis e desafiá-los com atividades mais avançadas.

Sendo assim, a análise de dados educacionais desempenha um papel significativo na exploração de informações a respeito do desempenho dos alunos, possibilitando a coleta e armazenamento através de uma análise de dados eficiente, que permite uma personalização do ensino atrelada a melhoria contínua da educação.

2.3 Aprendizado de máquina aplicado à educação

O aprendizado de máquina (machine learning) é uma subárea da inteligência artificial que permite a construção de modelos preditivos a partir de dados históricos. Na educação pública, essas técnicas têm sido aplicadas para prever evasão escolar, personalizar o ensino e avaliar políticas públicas.

Davenport e Ronanki (2018) destacam que a inteligência artificial pode ser usada para classificar estudantes em grupos de risco, sugerir intervenções personalizadas e automatizar processos de avaliação. Russel e Norvig (2020), apresentam os fundamentos da inteligência artificial, que inclui algoritmos supervisionados como por exemplo, a árvore

de decisão, um classificador de floresta aleatória, que é um meta-estimador que se encaixa em várias árvores de decisão classificadas em várias subamostras do conjunto de dados e usa a média para melhorar a precisão preditiva e controlar o sobreajuste[4] (overfitting), utilizando a melhor estratégia de divisão.

Segundo o Portal F10 (2021), aprendizado de máquina é uma tecnologia que utiliza a inteligência artificial e algoritmos para treinar as ferramentas para poderem realizar certas tarefas de forma autônoma, as ferramentas aprendem de forma interativa, através de dados coletados em suas interações e com isso, é possível identificar padrões e anomalias em grandes conjuntos de dados.

Na gestão escolar, modelos preditivos ajudam a identificar alunos em risco de evasão ou com baixo desempenho, permitindo que as instituições tomem decisões antecipadas. Dessa maneira, através do aprendizado de máquina, é possível identificar os padrões e prever quais alunos podem precisar de suporte adicional levando a intervenções mais eficazes.

2.4 Python e ferramentas de análise de dados

Entre as linguagens interpretadas, por diversos motivos históricos e culturais, Python desenvolveu uma comunidade grande e ativa de processamento científico e análise de dados. Nos últimos anos, Python passou de uma linguagem de computação científica e inovadora para uma das mais importantes em ciência de dados, aprendizado de máquina e desenvolvimento de softwares em geral (MCKINNEY, 2018).

Python com seu amplo suporte e suas bibliotecas desenvolvidas para análise e ciência de dados aliados a sua linguagem de alto nível, o transformou em uma opção popular para essas tarefas, sendo amplamente utilizado tanto no meio acadêmico quanto no mercado.

Ainda segundo McKinney (2018), o Python utiliza as seguintes bibliotecas[5] para a análise de dados e construção de modelos de IA:

- Pandas e Numpy: são utilizados para manipulação, limpeza e otimização da base de dados, além de análise exploratória (descritiva, diagnóstica, preditiva e prescritiva);
- Matplotlib e Seaborn: usados para visualizar os dados através de gráficos estatísticos;

[4] Sobreajuste (overfitting) é um fenômeno que ocorre em modelos de aprendizado de máquina e estatística quando o modelo se ajusta excessivamente bem aos dados de treinamento, a ponto de capturar não apenas os padrões gerais, mas também o ruído e as flutuações aleatórias presentes nesses dados.

[5] Bibliotecas Python são coleções de códigos pré-programados (módulos e funções) que os desenvolvedores podem usar para realizar tarefas específicas, evitando ter que escrever tudo do zero.

- Scikit-learn: fortemente utilizado em implementação de modelos de aprendizado de máquina (algoritimos para classificação e regressão).

Além dessas bibiliotecas, também é utilizado o Jupyter Notebook**,** que é uma aplicação web utilizada em ciência de dados, aprendizado de máquina, análise numérica e outras áreas para explorar dados, desenvolver algoritmos e apresentar resultados de forma clara e organizada.

Além dessas bibliotecas, McKinney (2018) também destaca inúmeras outras bibliotecas que podem ser utilizadas em diversos tipos de análises especificamente, mas para análises de dados exploratórias, as relatadas anteriormente são suficientes para se ter ótimos resultados como na análise de dados educacionais, onde se busca por padrões e tendências, os quais podem ser posteriormente utilizados em sistemas de classificações automáticas produzidos através do aprendizado de máquina.

3 METODOLOGIA

Este estudo adota uma abordagem quantitativa, exploratória e aplicada, com o objetivo de investigar padrões no desempenho acadêmico de estudantes do ensino médio da rede pública, por meio da aplicação de técnicas de análise de dados e modelos de aprendizado de máquina desenvolvidos em Python para prever futuros resultados através de novos dados.

Quantitativa**:** este método de pesquisa coleta e analisa dados numéricos para testar hipóteses, encontrar padrões, fazer previsões e generalizar resultados para populações maiores, uma vez que este trabalho utiliza uma base de dados com dados estruturados e sugere análise estatística.

Segundo (MATTAR; RAMOS, 2021) a quantidade de recursos tecnológicos e softwares que apoiam esse tipo de análise, não é preciso realizar cálculos para utilizar estatística; muito mais importante é desenvolver um raciocínio estatístico que inclua a compreensão dos conceitos utilizados, a identificação dos tipos de variáveis e a aplicação dos diferentes tipos de análises e testes estatísticos.

Exploratória: este método de estudo visa familiarizar o pesquisador com um assunto pouco conhecido buscando fornecer uma compreensão inicial sobre o tema, preencher lacunas de informação e gerar hipóteses para pesquisas futuras, na medida em que o trabalho busca identificar padrões e relações entre variáveis acadêmicas e comportamentais.

Segundo Ferreria, Mirando e Pinto (2021) cada vez mais temos acesso a dados dos mais variados tipos e formatos. Além disso, a quantidade de dados com acesso à informação vem aumentando. Para um projeto de ciência de dados (data science), é necessário tratar e analisar os dados brutos até se transformarem em informação. O uso correto dessa informação a transforma em conhecimento, e o ato de utilizarmos esse conhecimento em benefício da tomada de decisão gera o conhecimento como sabedoria;

Aplicada: este método utiliza investigação científica que visa solucionar problemas práticos e concretos, ou desenvolver novas tecnologias, produtos ou serviços, já que este trabalho se propõe a resolver um problema real com potencial impacto social, ou seja, a melhora no desempenho educacional.

Tendo compromisso com a aplicação do conhecimento, mas com enfoque estratégico de aplicação para a solução de problemas práticos. Espera-se, nesse sentido, que as pesquisas em educação tenham impacto social, oferecendo retorno à comunidade (MATTAR; RAMOS, 2021).

3.1 A Base de Dados

A base de dados **dados_estudantes.csv**, contém informações abrangentes sobre 2.392 alunos do ensino médio, detalhando seus dados demográficos, hábitos de estudo, envolvimento dos pais, atividades extracurriculares e desempenho acadêmico.

A variável-alvo classifica as notas dos alunos em classes de desempenho distintas, fornecendo um conjunto de dados bem elaborado e estruturado para pesquisa educacional, modelagem preditiva e análise estatística. Esta base de dados foi originada através do site Kaggle[6], que é uma plataforma de competição de ciência de dados e comunidade on-line para cientistas de dados e profissionais de aprendizado de máquina da Google LLC, e foi disponibilizado por Rabie El Kharoua[7], sob licença CC BY 4.0.

3.2 Etapas de Análise de Dados e construção do Modelo de Inteligência Artificial

a) **Limpeza e padronização dos dados**: renomeação de colunas, tratamento de valores inconsistentes e exclusão de campos irrelevantes (por exemplo, ID_Aluno).
b) **Classificação das notas**: os estudantes foram categorizados em notas de A a F, com base no GPA (por exemplo, A = nota >= 3.5).

[6] Kaggle: https://www.kaggle.com/
[7] Link da base de dados: https://www.kaggle.com/datasets/rabieelkharoua/students-performance-dataset

c) **Correção de classificações**: a base de dados ao ser analisada, observou-se que existiam inconsistências nos valores em relação a coluna Media das Notas e Classificação das Notas. Para corrigir esse problema e não haver divergências entre as análises, foi aplicada uma função do Python para fazer a correção das classificações de acordo com as notas do GPA.

d) **Análise Exploratória**: geração de estatísticas descritivas e visualizações para identificação de padrões e correlações.

e) **Análise Preditiva do Desempenho Acadêmico**: sugestões de acordo com a visualização dos dados, para novas tomadas de decisões baseada em funções condicionais e assim melhorar o desempenho acadêmico.

f) **Codificação das variáveis categóricas**: para dar inicio ao treinamento do modelo através dos algoritmos de classificação.

g) **Separação dos dados**: 70% para teste e 30% para treino.

h) **Modelos treinados**: vizinhos próximos (K-Nearest Neighbors (KNN)) e árvore de de decisão (Random Forest).

i) **Avaliação de desempenho**: o modelo árvore de decisão obteve 98,32% de acurácia[8], contra 71,34% do vizinhos próximos — sendo, portanto, selecionado.

Com o modelo final, foram feitas previsões em uma nova base de dados com 258 novos estudantes, permitindo a classificação antecipada de suas prováveis notas com base nos dados coletados.

4 RESULTADOS E DISCUSSÃO

Este projeto trata-se de uma avaliação curricular para um curso de Python no foco em Análise de Dados, utilizando um conjunto de dados simulado para fins didáticos e foi desenvolvido totalmente de forma prática. A definição do problema é avaliar: “Quais fatores demográficos, acadêmicos e comportamentais mais influenciam a classificação e a média das notas dos estudantes?” Tendo como base a variável-alvo Classificação das Notas. Os passos tomados para o inicio do projeto seguem na seguinte ordem:

- Passo 1 – Importar a base de dados;
- Passo 2 - Visualizar a base de dados;
- Passo 3 – Corrigir os erros na base de dados;

[8] Acurácia (accuracy): Métrica de avaliação que mede a proporção total de predições corretas feitas por um modelo em relação ao número total de predições realizadas.

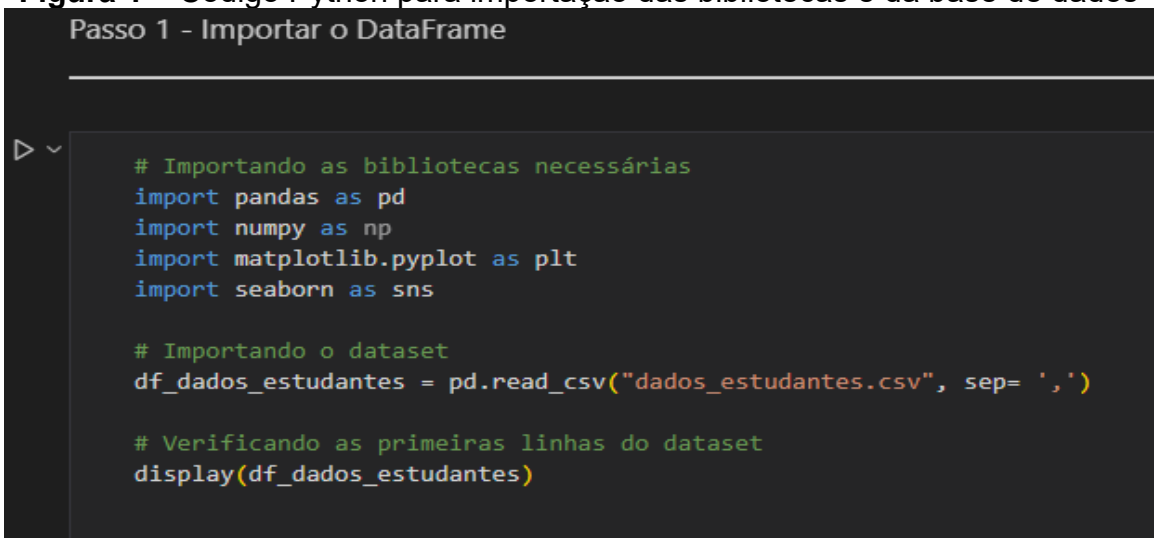
- Passo 4 – Analisar o desempenho acadêmico;
- Passo 5 – Analisar as variáveis que influenciam o desempenho acadêmico;
- Passo 6 – Fazer uma análise preditiva para melhorar o desempenho acadêmico, caso seja necessário.

O objetivo deste artigo não é explicar os códigos e funcionalidades do Python, mas sim detalhar como foi aplicada a análise de dados e a criação do modelo de aprendizado de máquina no projeto, com foco nos resultados para a gestão pública.

Análise exploratória (EDA)

Passo 1 – Importar a base de dados

Figura 1 – Código Python para importação das bibliotecas e da base de dados



Fonte: Elaborado pelo autor (2025).

Para dar inicio ao projeto, é necessário através do código da figura 1, importar as bibliotecas do Python necessárias para a Análise de Dados e assim importar a base de dados **dados_estudantes.csv** e seguir ao passo 2.

Figura 2 – Visualização da base de dados

	StudentID	Age	Gender	Ethnicity	ParentalEducation	StudyTimeWeekly	Absences	Tutoring
0	1001	17	1	0	2	19.833723	7	1
1	1002	18	0	0	1	15.408756	0	0
2	1003	15	0	2	3	4.210570	26	0
3	1004	17	1	0	3	10.028829	14	0
4	1005	17	1	0	2	4.672495	17	1
...	...	...	...	...	...	...	...	...
2387	3388	18	1	0	3	10.680555	2	0
2388	3389	17	0	0	1	7.583217	4	1
2389	3390	16	1	0	2	6.805500	20	0
2390	3391	16	1	1	0	12.416653	17	0
2391	3392	16	1	0	2	17.819907	13	0

2392 rows × 15 columns

Fonte: Elaborado pelo autor (2025).

A figura 2 exibe a base de dados gerada através do código da figura 1. Observa-se que as colunas da base de dados estão em inglês e que os registros de todas as colunas estão no formato de números, o que será explicado na tabela 1.

Para seguir os padrões apresentados na Descrição das Colunas da base de dados, é necessário alterar o formato de dados de algumas colunas, seguindo o padrão descrito na Tabela 1:

Tabela 1 – Mapa de descrição das colunas da base de dados

Descrição das Colunas da Base de Dados	
Identificação do Estudante	
ID do Aluno	Identificador único para cada aluno (de 1001 a 3392).
Detalhes Demográficos	
Idade	Idade dos estudantes, variando de 15 a 18 anos.
Gênero (gênero do estudante)	0: Masculino - 1: Feminino
Etnia (etnia dos estudantes)	0: Caucasiano - 1: Afrodescendente - 2: Asiático - 3: Outra
Escolaridade dos pais	
Escolaridade dos Pais (Nível de escolaridade dos pais)	0: Nenhum - 1: Ensino Médio - 2: Alguma - 3: Graduação Faculdade - 4: Pós-Graduação ou Superior
Hábitos de Estudo	
Horas de Estudo Semanais	Tempo de estudo semanal em horas (0 a 20).
Faltas	Número de faltas no ano letivo (0 a 30).
Aulas Particulares (Participação em aulas particulares)	0: Não - 1: Sim
Envolvimento Parental	

Apoio dos Pais (nível de apoio dos pais)	0: Nenhum - 1: Baixo - 2: Moderado - 3: Alto 4: Muito Alto
Atividades Extracurriculares	
Participação em atividades extracurriculares	0: Não - 1: Sim
Esportes (participação em esportes)	0: Não - 1: Sim
Musica (participação em atividades musicais)	0: Não - 1: Sim
Voluntariado (participação em atividades de voluntariado)	0: Não - 1: Sim
Desempenho Acadêmico	
Media das Notas (média das notas (GPA))	variando de 2.0 a 4.0.
Classificação das Notas (classificação do aluno com base na média das notas	0: 'A' (Média >= 3.5) 1: 'B' (3.0 <= Média < 3.5) 2: 'C' (2.5 <= Média < 3.0) 3: 'D' (2.0 <= Média < 2.5) 4: 'F' (Média < 2.0)

Fonte: Elaborado pelo autor (2025).

Ao desenvolver um Banco de Dados, é natural a criação de alguns campos do tipo numérico, a fim de que o sistema possa funcionar de maneira consistente, uma vez que os processadores de computadores processam as informações através de números binários (0 e 1), isso facilita o trabalho do computador e do programador que desenvolve ou administra bancos de dados, tornando o desempenho e a manutenção mais eficazes.

Neste caso o autor da base de dados informou o que significa cada número em seus referidos campos através deste mapa de descrição das colunas como mostrado na tabela 1, e o Python possui funções específicas para tratar destes ajustes que será mostrado nos próximos passos.

Passo 2 – Visualizar a base de dados

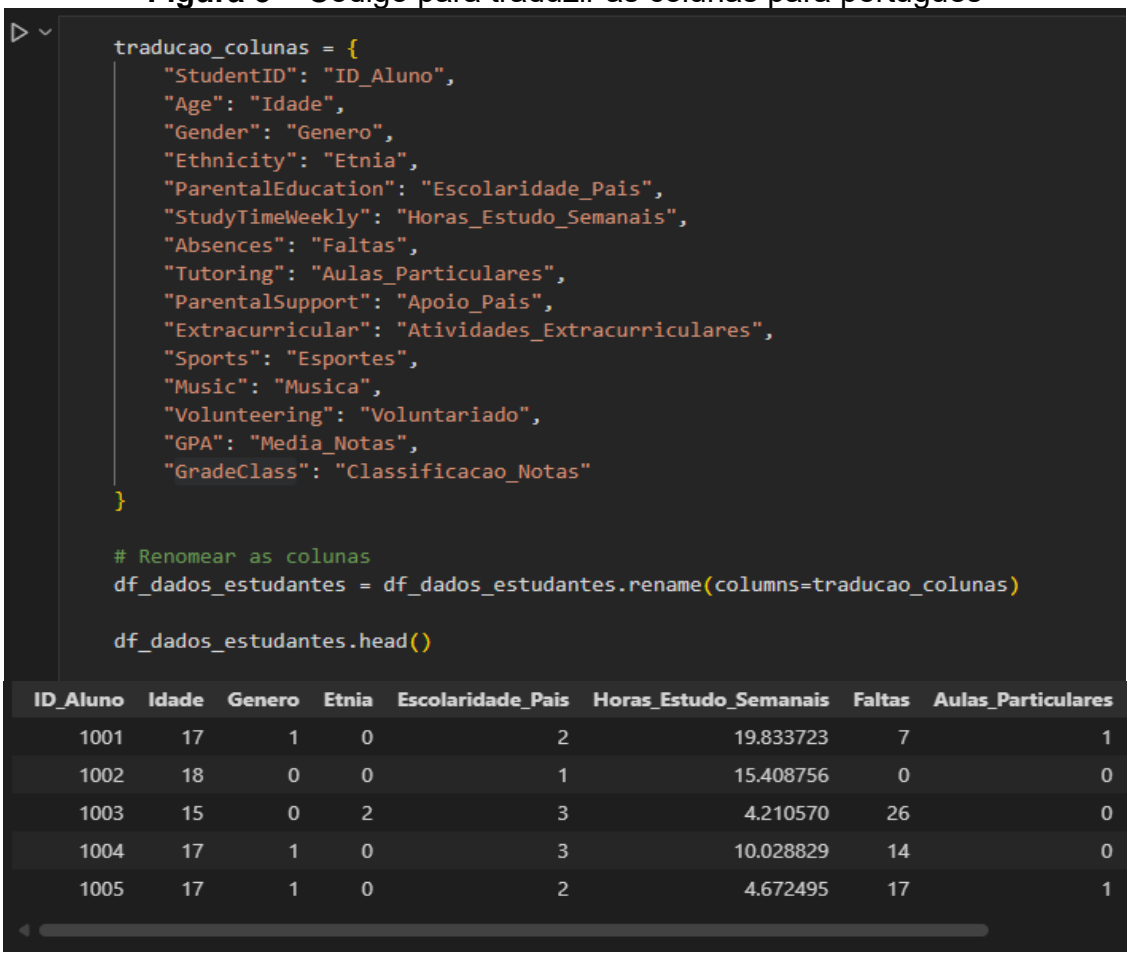
Neste passo é feita a análise geral da base de dados em busca de erros, valores nulos e ausentes, como também informações sobre o tamanho da base de dados, tipos de dados, se esses campos convergem com os tipos de dados registrados, etc. O Python e sua biblioteca Pandas, possui uma gama de métodos e funções para manipular e corrigir base de dados. É normal encontrar diariamente, bases de dados inconsistentes e com muitos erros de digitação, uma vez que os mesmos são cadastrados por humanos e assim são passiveis a erros.

No caso desta base de dados, não foram encontrados erros ou falhas nos registros, mas ainda assim é necessário ajustar algumas configurações para facilitar o entendimento da análise para quem for utilizar as informações para a tomada de decisões.

Passo 3 – Corrigir erros na base de dados e remodelar os dados

No passo 3, como não existe erros na base de dados para correção, somente será necessário remodelar a base de dados.

Figura 3 – Código para traduzir as colunas para português



```
traducao_colunas = {
    "StudentID": "ID_Aluno",
    "Age": "Idade",
    "Gender": "Genero",
    "Ethnicity": "Etnia",
    "ParentalEducation": "Escolaridade_Pais",
    "StudyTimeWeekly": "Horas_Estudo_Semanais",
    "Absences": "Faltas",
    "Tutoring": "Aulas_Particulares",
    "ParentalSupport": "Apoio_Pais",
    "Extracurricular": "Atividades_Extracurriculares",
    "Sports": "Esportes",
    "Music": "Musica",
    "Volunteering": "Voluntariado",
    "GPA": "Media_Notas",
    "GradeClass": "Classificacao_Notas"
}

# Renomear as colunas
df_dados_estudantes = df_dados_estudantes.rename(columns=traducao_colunas)

df_dados_estudantes.head()
```

ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares
1001	17	1	0	2	19.833723	7	1
1002	18	0	0	1	15.408756	0	0
1003	15	0	2	3	4.210570	26	0
1004	17	1	0	3	10.028829	14	0
1005	17	1	0	2	4.672495	17	1

Fonte: Elaborado pelo autor (2025).

Como esta base de dados foi originada em inglês, se faz necessário traduzi-la para o português através do código na figura 3.

Figura 4 – Código para substituir valores numérico pelos valores do mapa de descrições das colunas

```
# Para alterar os valores numéricos pelos seus respectivos valores
# informados na descrição, usarei o método replace
# para reverter os números cadastrados em seus devidos valores informados.

# Alterando os valores da descrição da coluna Gênero
df_dados_estudantes["Genero"] = df_dados_estudantes["Genero"].replace({
    0: "Masculino",
    1: "Feminino"
})
```

[42]

```
# Alterando os valores da descrição da coluna Etnia
df_dados_estudantes["Etnia"] = df_dados_estudantes["Etnia"].replace({
    0: "Caucasiano",
    1: "Afrodescendente",
    2: "Asiático",
    3: "Outra"
})
```

[43]

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0
2	3	15	Masculino	Asiático	Graduação	4	26
3	4	17	Feminino	Caucasiano	Graduação	10	14
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17
...	...	...	...	...	...	...	...
2387	2388	18	Feminino	Caucasiano	Graduação	10	2
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13

2392 rows × 15 columns

Fonte: Elaborado pelo autor (2025).

Com o código através do método **replace()**[9] do Python é possível trocar qualquer registro para o formato desejado, neste caso trocar os números pelo texto. Por exemplo, no campo gênero, onde for 0 é substituído por Masculino, e onde for 1 por Feminino.

[9] Método replace() em Python é usado para substituir todas as ocorrências de uma substring por outra dentro de uma string retornando uma nova string com as alterações.
String: sequência de caracteres usada para representar texto em programação.

Após todos os ajustes realizados, a base de dados foi analisada novamente para verificar se tudo ficou corretamente ajustado, apenas por um detalhe que foi notado na análise em relação as médias das notas e a classificação, que no caso desta base de dados estatavam divergindo entres os valores.

Figura 5 – Notas das médias não estão de acordo com a classificação

Fonte: Elaborado pelo autor (2025).

De acordo com a figura 5, é possível observar que ao se comparar a nota da classificação com a nota da média, há uma incompatibilidade nos valores. A média 1.8 por exemplo está classificada como letra B, sendo que B é para notas maior ou igual 3.0, assim como C é para notas maior que 2.5 e há uma nota 1.1 classificada como C.

Figura 6 – Aplicação do método Apply com Lambda para ajuste das classificações em relação ao GPA

Ao analisar o novo DataFrame, observei que existem inconsistências nos valores em relação a coluna Media_Notas e Classificacao_Notas. Para corrigir esse problema e não haver divergências entre as análises, irei corrigir os valores usando o método apply com lambda de acordo com as informações descritas na Descrição do Desempenho Acadêmico.

```
# Aplicando o lambda para corrigir os valores da Classificacao_Notas em relação a Média_Notas
novo_df_estudantes["Classificacao_Notas_Lambda"] = novo_df_estudantes["Media_Notas"].apply(lambda x:
                                                    "A" if x >= 3.5 else
                                                    "B" if x >= 3.0 else
                                                    "C" if x >= 2.5 else
                                                    "D" if x >= 2.0 else
                                                    "F")

display(novo_df_estudantes)
```

Python

ıres	Apoio_Pais	Atividades_Extracurriculares	Esportes	Musica	Voluntariado	Media_Notas	Classificacao_Notas	Classificacao_Notas_Lambda
Sim	Moderado	Não	Não	Sim	Não	2.9	C	C
Não	Baixo	Não	Não	Não	Não	3.0	B	B
Não	Moderado	Não	Não	Não	Não	0.1	F	F
Não	Alto	Sim	Não	Não	Não	2.1	D	D
Sim	Alto	Não	Não	Não	Não	1.3	F	F
...	...	...	...	...	...	...	...	...
Não	Muito Alto	Sim	Não	Não	Não	3.5	A	A
Sim	Muito Alto	Não	Sim	Não	Não	3.3	F	B
Não	Moderado	Não	Não	Não	Sim	1.1	C	F
Não	Moderado	Não	Sim	Sim	Não	1.8	B	F
Não	Moderado	Não	Não	Não	Sim	2.1	B	D

Fonte: Elaborado pelo autor (2025).

Com o método **Apply**[10] e **Lambda**[11] do Python é possível ajustar os valores de uma coluna com referência a outra. O código executa o comando dizendo que quando **x**, que é representado pela coluna Média das Notas, for maior ou igual 3.5 por exemplo, então a coluna Classificação das Notas recebe a letra A e assim segue até o fim da tabela.

Se fizermos a comparação entre as colunas Classificação das Notas e Classificação das Notas Lambda na base de dados da figura 6, veremos que agora os valores foram ajustados corretamente. Agora com a base de dados devidamente ajustada e remodelada, partiremos para o Passo 4, o mais importante e o centro de todo o projeto, que é a Análise de Dados.

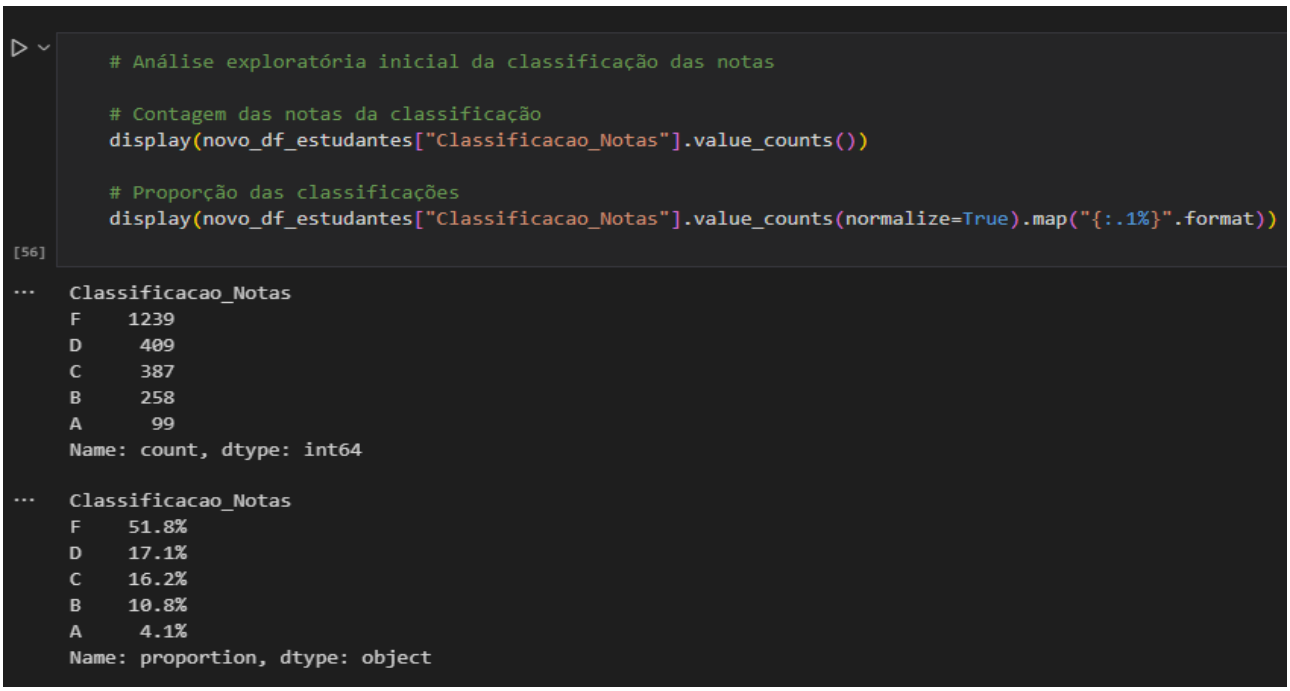
Passo 4 – Análise de dados

Este passo é iniciado com uma análise exploratória para identificar como os alunos estão devidamente classificados como é observado na figura 7.

Figura 7 – Análise exploratória das classificações dos estudantes (Notas A–F)

[10] O método apply do pandas aplica uma função a uma serie ou ao longo de um eixo (linhas ou colunas) de um dataframe (base de dados)

[11] Uma função `lambda` em Python é uma função anônima e curta, definida em uma única linha com a palavra-chave lambda.



```
# Análise exploratória inicial da classificação das notas

# Contagem das notas da classificação
display(novo_df_estudantes["Classificacao_Notas"].value_counts())

# Proporção das classificações
display(novo_df_estudantes["Classificacao_Notas"].value_counts(normalize=True).map("{:.1%}".format))
```

[56]

```
Classificacao_Notas
F    1239
D     409
C     387
B     258
A      99
Name: count, dtype: int64

Classificacao_Notas
F    51.8%
D    17.1%
C    16.2%
B    10.8%
A     4.1%
Name: proportion, dtype: object
```

Fonte: Elaborado pelo autor (2025).

De acordo com a análise exploratória, os alunos foram classificados com 51,8% na nota F, 17,1% na nota D, 16,2% na nota C, 10,8% na nota B e 4,1% na nota A. Os dados indicam que mais de 50% dos alunos estão classificados em F mostrando que o desempenho acadêmico destes estudantes possui um desempenho insatisfatório e que há uma grande exigência de se criar estratégias eficazes a fim de elevar o desempenho destes estudantes.

A partir de agora é necessário verificar quais variáveis mais afetaram esse desempenho para assim poder responder a pergunta definida no problema e poder tomar as decisões estratégicas eficazes, a fim de aumentar o desempenho dos alunos.

Passo 5 – Análise e visualização das variáveis que influenciam no desempenho acadêmico.

Através da visualização de dados em forma de gráficos estatísticos, os dados serão apresentados de forma visível e clara, baseado no método de correlação entre todas as variáveis e a variável-alvo que é a coluna Classificação das Notas, descobrindo quais variáveis mais impactam a classificação. Para visualizar os dados, o Python possui as bibliotecas Matplotlib e Seaborn para criar gráficos funcionais, modernos e atraentes.

Figura 8 – Código para execução dos gráficos estatísticos

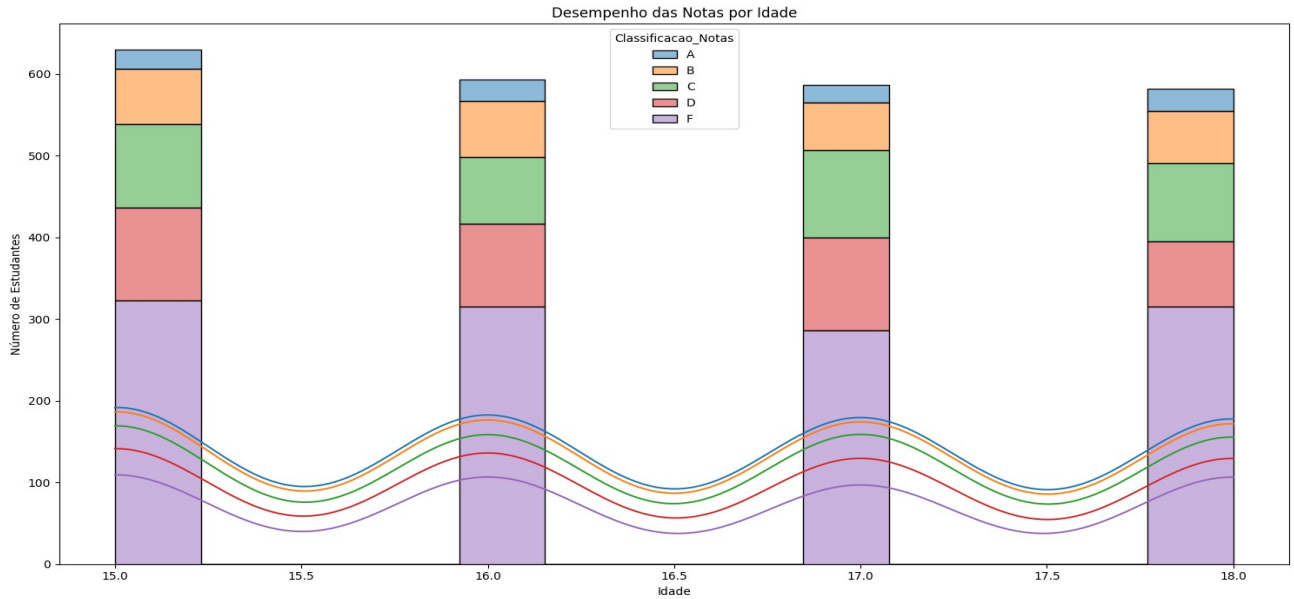
```
# Visualização e Análise do desempenho da classificação das notas (como as outras colunas do DataFrame impactam no
# desempenho das notas)

# Criação dos Gráficos
for coluna in novo_df_estudantes.columns:
    # Configuração da figura
    plt.figure(figsize=(15, 8))

    # gráficos de histograma - Desempenho das Notas VS Variáveis
    sns.histplot(data=novo_df_estudantes, x=coluna, hue="Classificacao_Notas", kde=True, multiple="stack")
    plt.title("Desempenho das Notas por " + coluna)
    plt.xlabel(coluna)
    plt.ylabel('Número de Estudantes')

    plt.tight_layout()
    plt.show()
```

Fonte: Elaborado pelo autor (2025).

Com este código, todos os gráficos são gerados de uma só vez. Para fazer a correlação entre as variáveis, foi utilizado o gráfico de Histograma, este tipo de gráfico montado em pilhas, ajuda a ter uma visualização abrangente entre o número de envolvidos em relação a variável consultada, com a possibilidade de relacionar a variável-alvo através de uma legenda com cores características de acordo com o valor da variável alvo, destacando essas cores diretamente nas pilhas do gráfico.

Na sequência, será observado o que cada gráfico deve revelar e os insights por traz de cada um deles.

Figura 9 – Desempenho das notas por idade

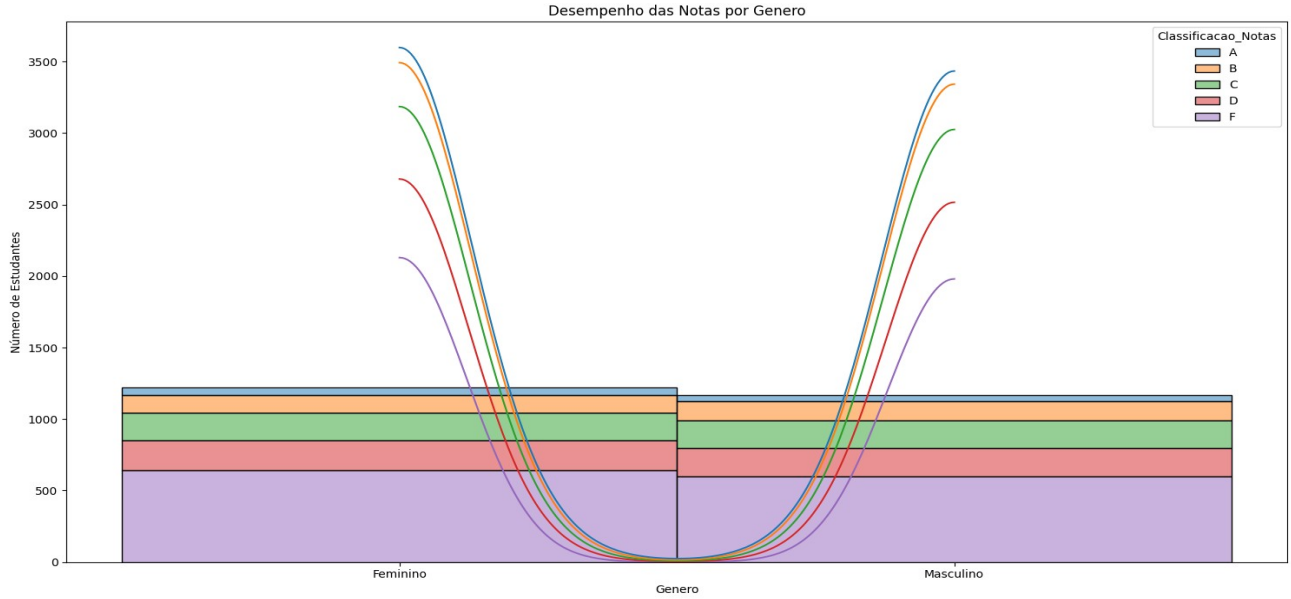


Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 9, indicam que as idades entre os alunos são próximas e estão na faixa entre 15 e 18 anos. As idades estão na mesma proporção em número de alunos. Quanto a classificação das notas, os desempenhos são muito próximos entre as

idades, indicando que a idade não impacta o desempenho, porém esta análise demonstra uma distribuição homogênea entre os alunos.

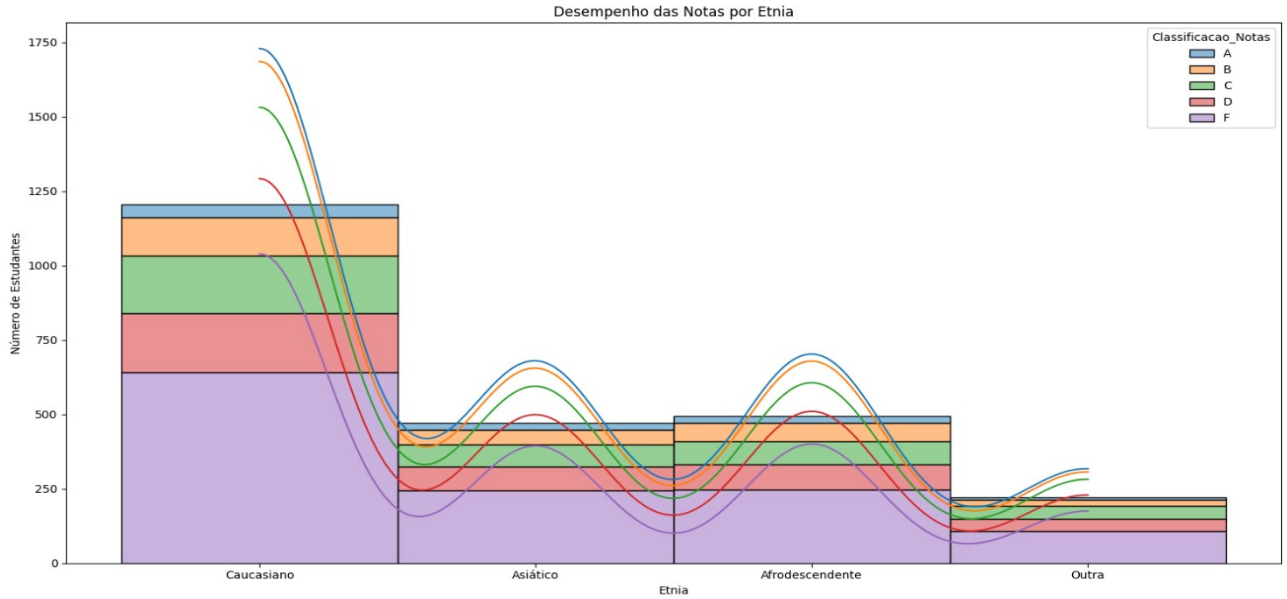
Figura 10 – Desempenho das notas por gênero



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 10, indicam que em relação ao gênero dos alunos, há um equilíbrio no número de alunos. A classificação das notas também está em equilíbrio, definindo que o gênero não tem relação direta com o desempenho acadêmico dos estudantes indicando uma distribuição homogênea entre os grupos.

Figura 11 – Desempenho das notas por etnia



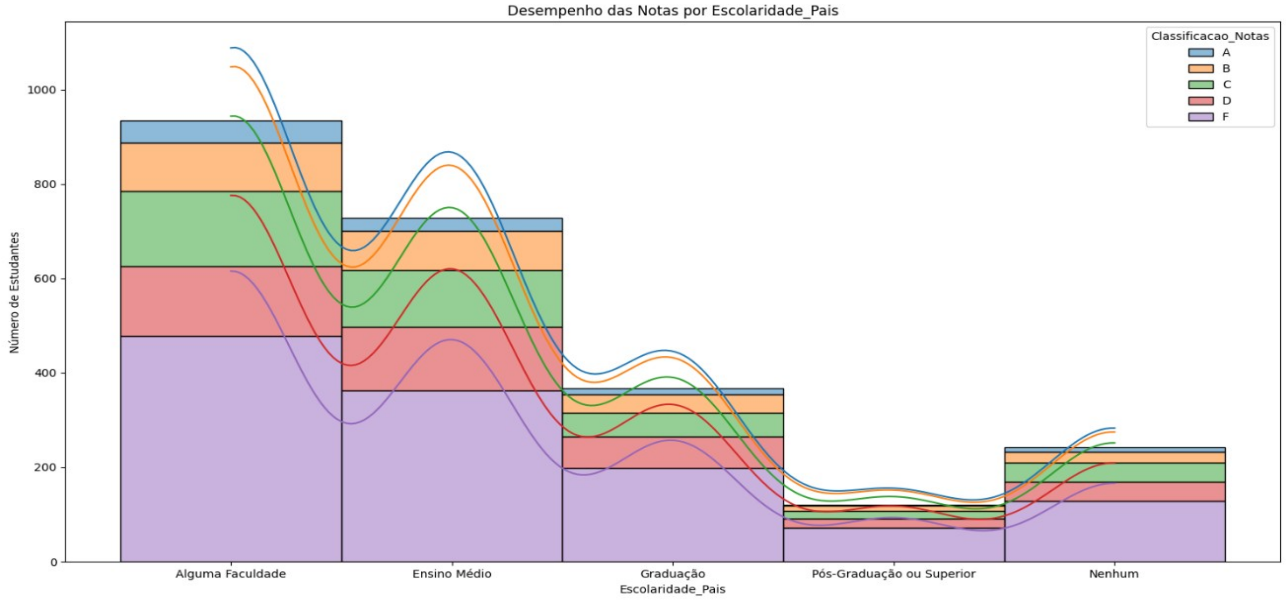
Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 11, indicam um contraste entre os grupos étnicos, com o maior número de alunos concentrada no grupo Caucasiano e os grupos Asiáticos e

Afrodescendentes em equilíbrio, mas muito abaixo do grupo Caucasiano. O grupo Outros é praticamente irrelevante.

Referente a classificação das notas, todos os grupos também estão bem equilibrados, comprovando que a etnia também não tem influencia sobre o desempenho acadêmico. Somente observar a diferença na distribuição entre o grupo caucasiano e as demais.

Figura 12 – Desempenho das notas por escolaridade dos pais

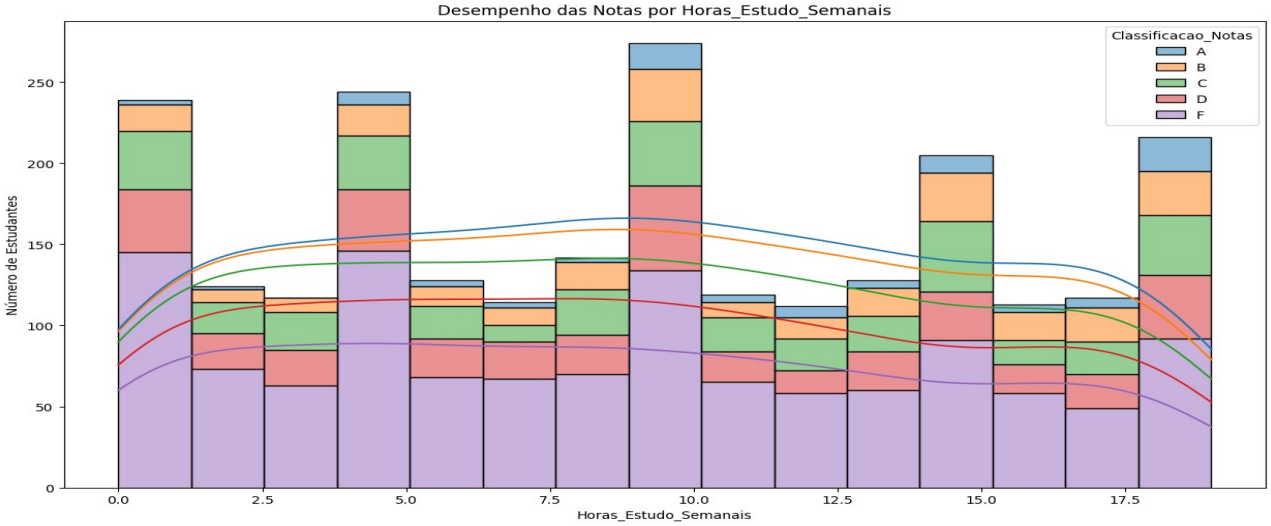


Fonte: Elaborado pelo autor (2025).

Nos dados do gráfico da figura 12, observa-se que o grupo Alguma Faculdade é onde estão agrupados a maior parte dos alunos, com o número de alunos no grupo Ensino Médio na sequência. Já os grupos Graduação, Pós-graduação ou Superior e Nenhum estão agrupados a menor parte dos estudantes.

Em relação a classificação, os grupos alguma faculdade e ensino médio possuem uma leve frequência na nota A, mas ao mesmo tempo irrelevante em relação aos outros grupos, revelando que a escolaridade dos pais também possui uma certa influencia na classificação onde a nota A possui um leve aumento nesses dois grupos, indicando que alunos com pais mais graduados tendem a ter notas melhores.

Figura 13 – Desempenho das notas por horas de estudos semanais

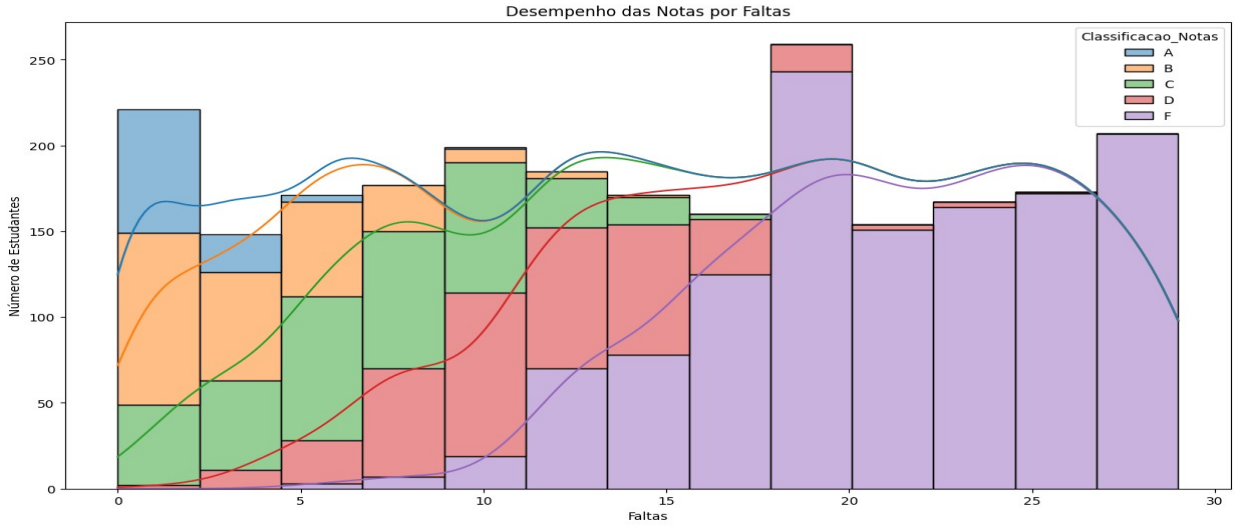


Fonte: Elaborado pelo autor (2025).

Nos dados do gráfico da figura 13, observa-se uma certa dispersão entre as horas de estudos, resumindo, uma grupo está entre 1 e 2 horas, outro está entre 4 e 5 horas, já outro está entre 8 e 10 horas, sendo estes os grupos com o maior número de alunos concentrados. Depois há uma certa concentração entre 14 e 15 horas e outra acima de 17 horas. Mas em relação a classificação há um um grupo que se destaca que é o grupo entre 8 e 10 horas possuindo uma certa elevação em todas as classificações e sofrendo uma leve queda na nota F e tendo um aumento discreto na nota A, indicando que as horas de estudo influenciam de forma direta o desempenho dos estudantes,

Mas de forma equilibrada, ou seja, acima de 10 horas o desempenho sofre forte queda, assim como abaixo de 8 horas. Dessa forma, a análise deste gráfico, indica que o ideal de horas de estudos semanais está entre 8 e 10 horas de estudos para que os estudantes alcancem melhores desempenhos.

Figura 14 – Desempenho das notas por faltas



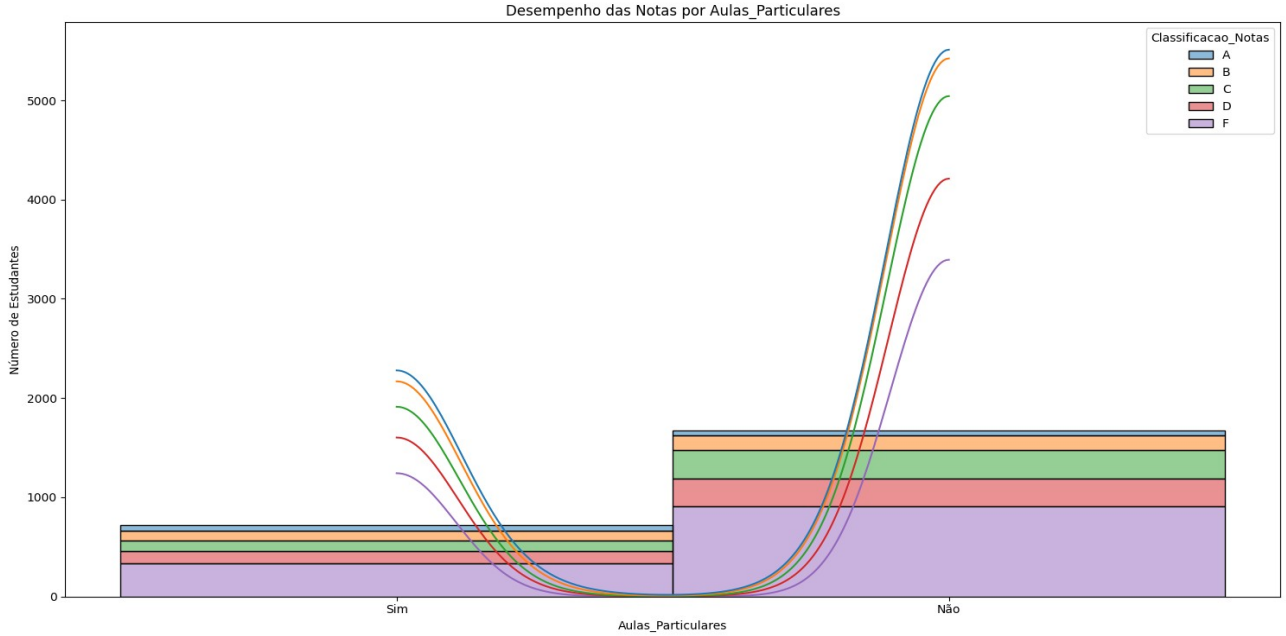
Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 14, indicam 2 faixas de grupos com a maior parte dos alunos concentradas entre as faixas de 0 a 2 faltas e outro grupo na faixa entre18 e 20 faltas. Na faixa entre 5 e 15 faltas há um certo equilíbrio no número de alunos, indicando que o nível de frequência é até satisfatório, assim como nas faixas entre 20 e 30.

Isso quer dizer que nessa análise em específico, é possível dizer que até existe uma certa frequência de alunos nas aulas, pois existe uma distribuição equilibrada de alunos em todas as faixas, exceto nas duas em destaque. Mas o ponto crítico está nas classificações, onde as classificações A e B praticamente desaparecem acima de 7 faltas com a nota C sumindo a partir de 18, porém as notas D e F predominam em todas as faixas.

Com certeza a falta de presença em sala de aula é um fator que impacta de forma considerada no desempenho dos estudantes, embora haja uma frequência razoável dos estudantes, o que exige estratégias eficazes para evitar a baixa frequência nas salas de aulas.

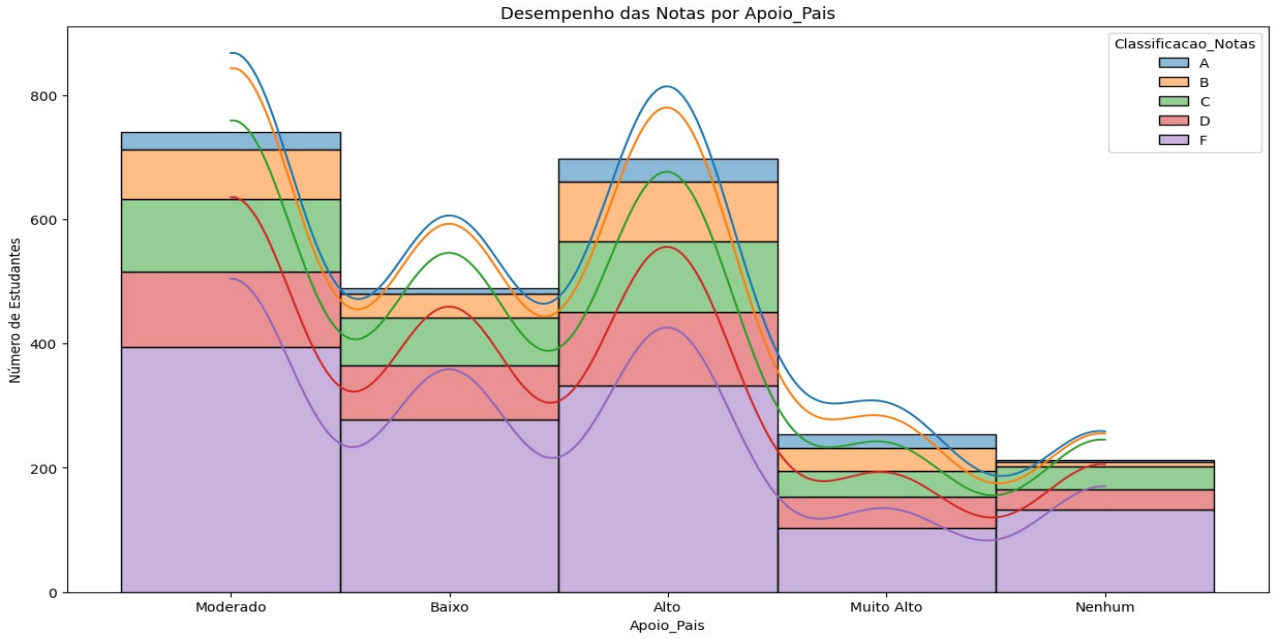
Figura 15 – Desempenho das notas por aulas particulares



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 15, indicam que a maioria dos alunos não pratica aulas particulares, embora a classificação das notas esteja equilibra entre os que praticam e não praticam, indicando que esta variável não impacta no desempenho dos estudantes, mas pode favorecer em um aumento representativo no desempenho se for incentivada.

Figura 16 – Desempenho das notas por apoio dos pais

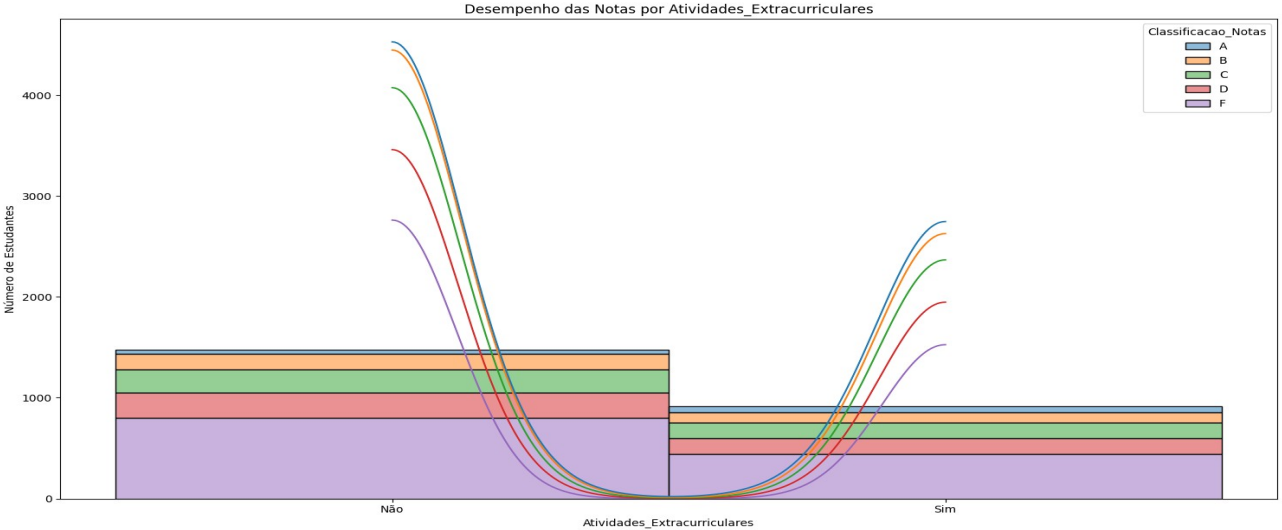


Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 16, indicam que a maioria dos alunos que tem apoio dos pais, estão nos grupos Alto, Moderado e Baixo, confirmando desta forma, que o apoio dos pais influencia muito no desempenho dos estudantes, apesar do grupo Muito Alto estar muito abaixo dos outros, quase empatado com aqueles que não possui Nenhum Apoio.

Esta variável indica o quanto o apoio dos pais é importante e impacta diretamente no desempenho dos estudantes, basta notar que o grupo baixo mesmo com um maior número de alunos, teve uma classificação muito pior que o grupo muito alto que está abaixo de todos os outros grupos, além desse grupo estar com uma classificação bem equilibrada e próxima aos outros grupos.

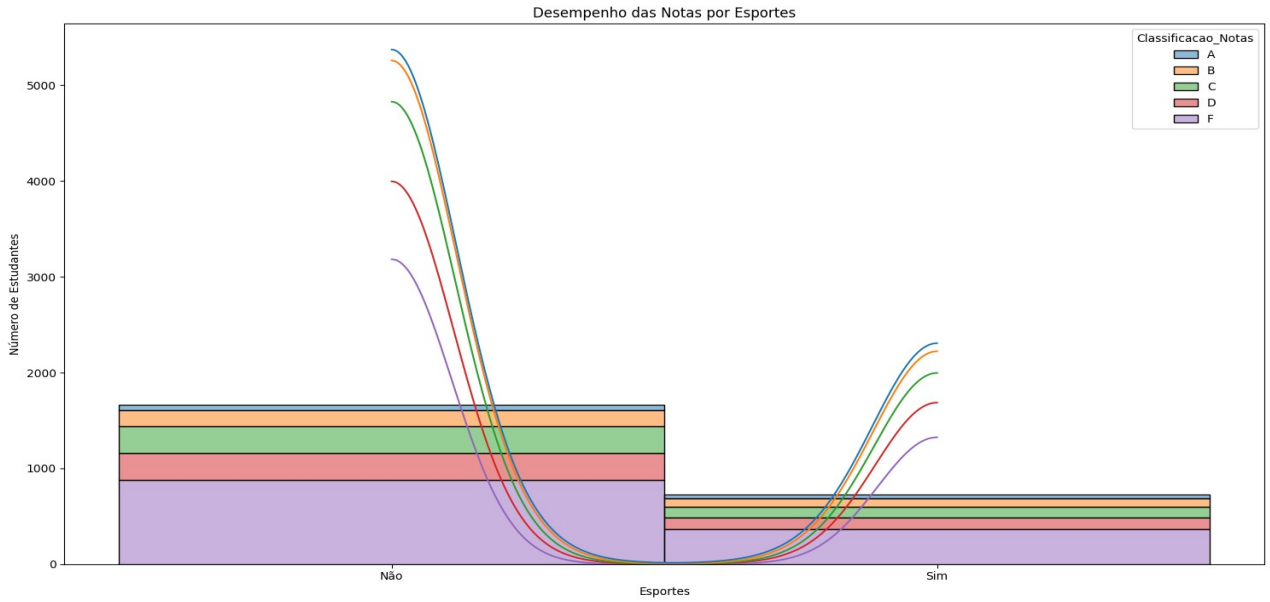
Figura 17 – Desempenho das notas por atividades extracurriculares



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 17, indicam que a maioria dos estudantes não pratica esta atividade. Apesar das classificações estarem equilibradas entres os grupos Sim e Não, e esta variável não impactar diretamente o desempenho, o grupo Sim evidencia que os alunos que praticam atividades extracurriculares alcançaram melhores classificações.

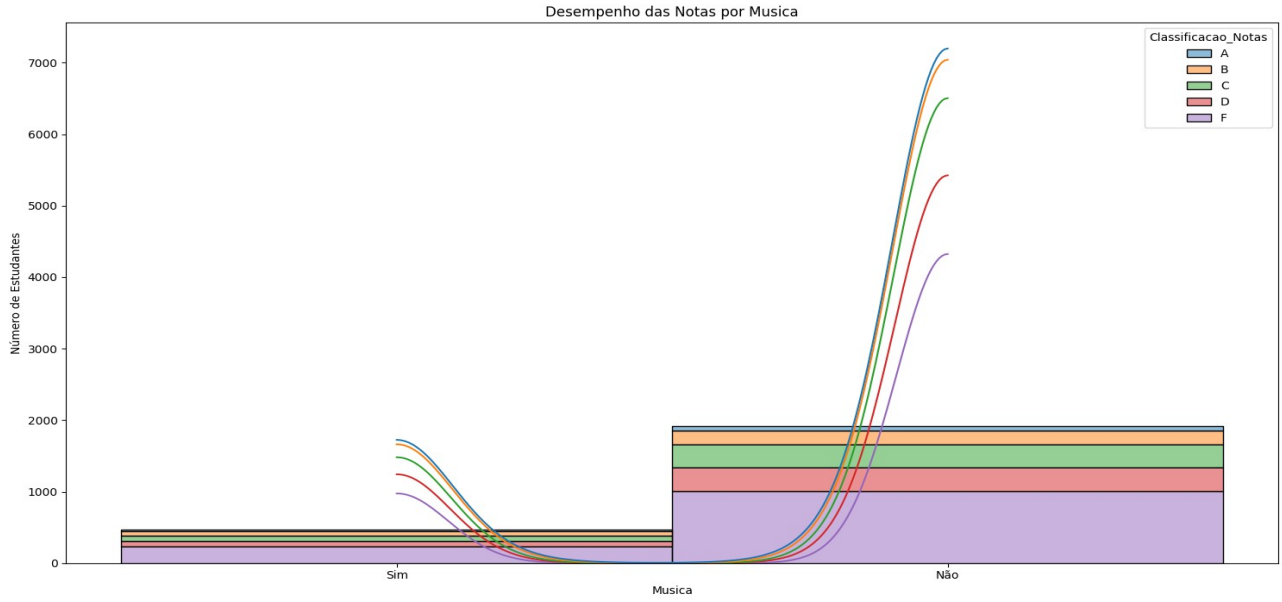
Figura 18 – Desempenho das notas por esportes



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 18, indicam que a maioria dos estudantes também não pratica esportes. Já o desempenho das classificações são bastante equilibrados nos dois grupos revelando que a variável esportes também não impacta no desempenho dos estudantes.

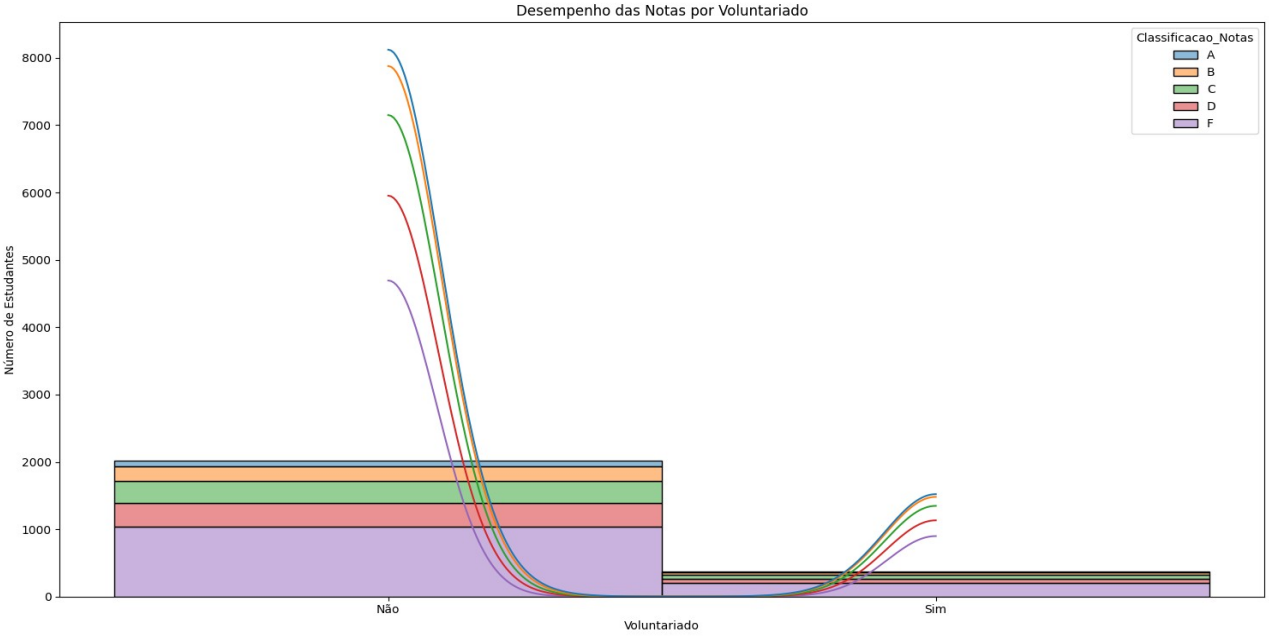
Figura 19 – Desempenho das notas por música



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 19, indicam que a grande maioria dos estudantes não pratica esta atividade. Nos dois grupos Sim e Não, as notas das classificações são equilibradas, também evidenciando que esta variável não influencia no desempenho dos estudantes.

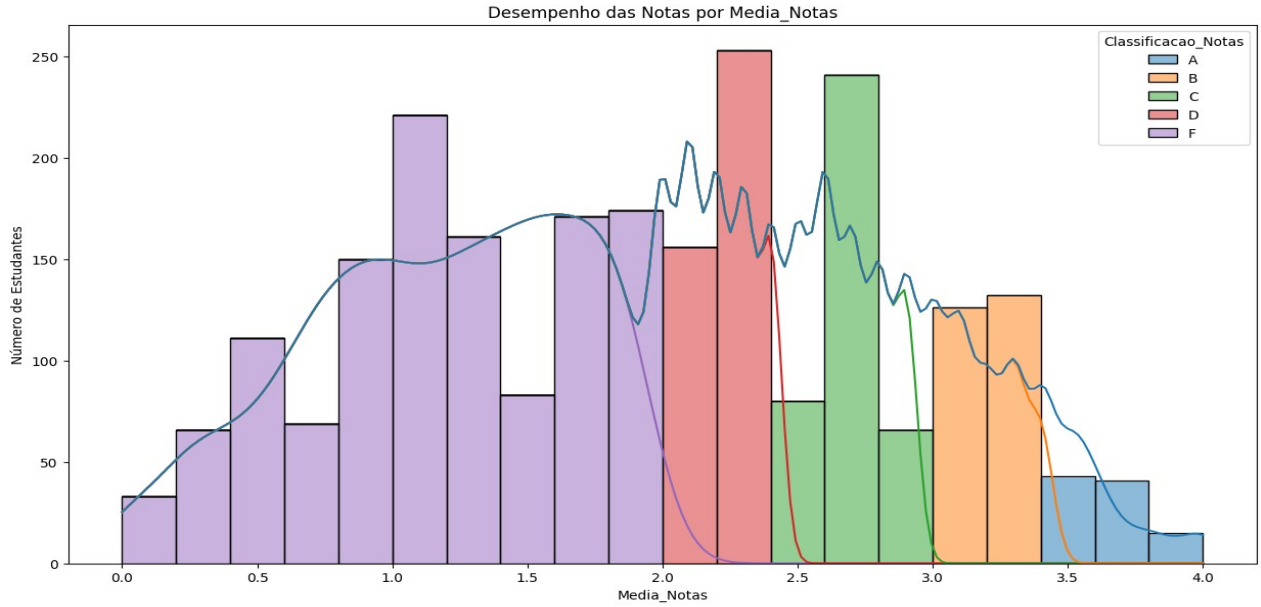
Figura 20 – Desempenho das notas por voluntariado



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 20, indicam que a grande maioria dos alunos não participa desta atividade. As notas das classificações também estão bem próximas entre os dois grupos, mostrando também que esta variável não afeta o desempenho dos estudantes.

Figura 21 – Desempenho das notas por média (GPA)

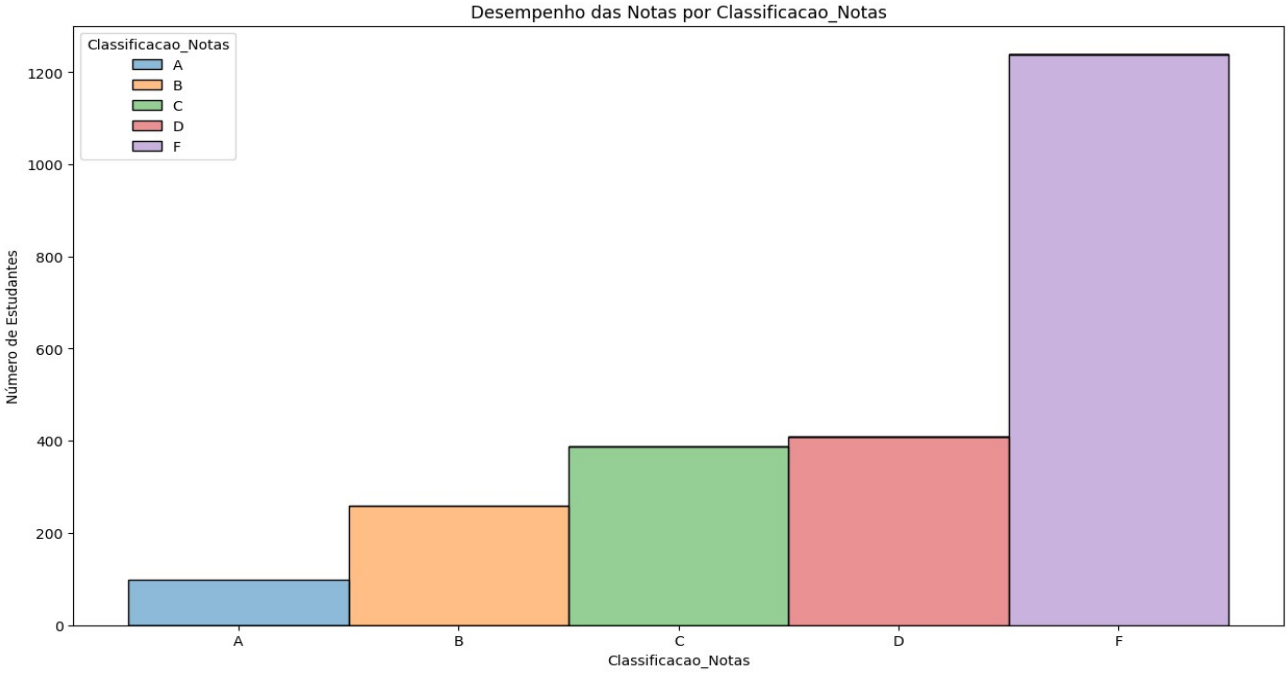


Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 21, indicam o baixo desempenho dos estudantes. As notas de 3 a baixo figuram entre a maioria dos estudantes, enquanto as que estão acima de 3 figuram entre o menor número de alunos, impactando diretamente na classificação e no desempenho dos estudantes.

Vale destacar que esta variável é afetada pelas demais variáveis, demonstrando que mesmo as variáveis que não tiveram influência direta no desempenho, as mesmas devem ser avaliadas, para talvez estarem gerando na variável notas por média, uma melhor classificação do desempenho dos estudantes, resumindo, esta variável esclarece que todas as variáveis estão correlacionadas direta e indiretamente influenciando esta variável.

Figura 22 – Desempenho das notas por classificação das notas



Fonte: Elaborado pelo autor (2025).

Os dados do gráfico da figura 22, indicam apenas como o desempenho dos estudantes foi afetado através das outras variáveis. No gráfico é possível notar como a nota F é predominante e figura praticamente entre o maior número de alunos, com as demais notas muito abaixo e caindo de forma decrescente até a nota A. Este gráfico evidencia uma forte tomada de decisão baseada em estratégias relacionadas aos insights obtidos das demais variáveis analisadas através dessa Análise Exploratória de Dados para que seja obtido um melhor desempenho dos estudantes.

Conclusão e entrega do resultado a partir dos dados

De modo geral, a história contada através dos dados analisados de acordo com os insights extraídos, deixa claro que as variáveis impactam diretamente a variável média das notas, esta por sua vez define o desempenho através da variável-alvo classificação das notas.

Variáveis que impactaram a classificação das notas:

- **Horas de estudos semanais:** alunos que estudam entre 8 e 10 horas alcançam notas maiores, abaixo de 8 e acima de 10 as notas tendem a cair.

Sugestão: alunos devem estudar no mínimo 8 horas e no máximo 10 horas por semana.

- **Faltas:** Alunos com mais de 6 faltas tem desempenho sofrível na classificação.

Sugestão: Melhorar a frequência dos alunos, criando um limite máximo de faltas que atingida ou extrapolada, desclassifique o aluno.

- **Aulas Particulares:** Em Aulas particulares o desempenho das notas foi bem pior para os estudantes que não participaram.

Sugestão: incentivar os alunos a fazerem aulas particulares.

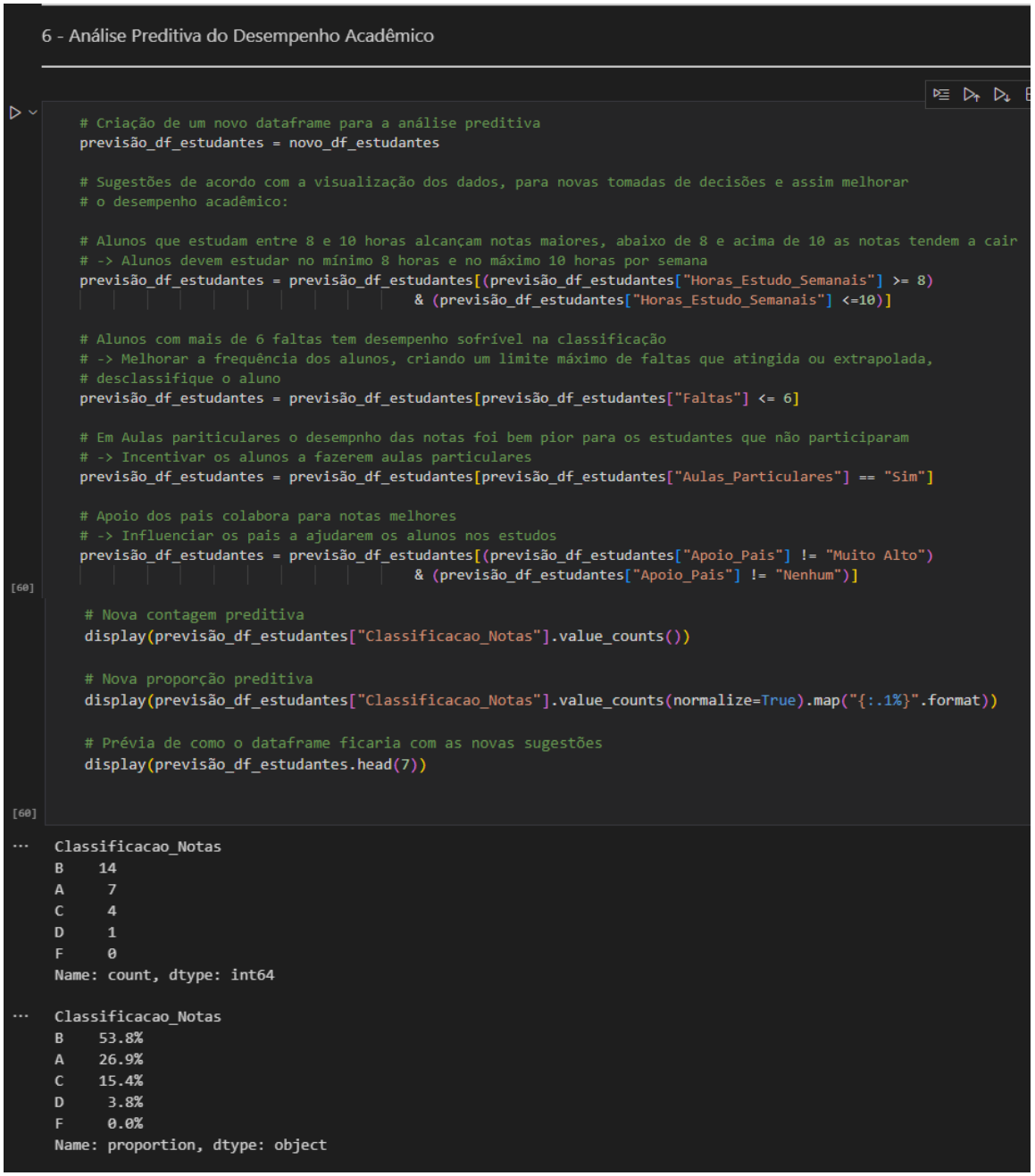
- **Apoio dos pais:** Apoio dos pais colabora para notas melhores.

Sugestão: influenciar os pais a ajudarem os alunos nos estudos.

Passo 6 – Análise preditiva do desempenho acadêmico

Com as variáveis impactantes definidas, é hora de desenvolver as simulações baseadas nestas através de funções condicionais. Para fazer essa análise preditiva, são criadas situações através de alguns códigos do Python para ver como será afetado o resultado segundo essas novas situações conforme segue na figura 23.

Figura 23 – Códigos para criar novos cenários baseados no insights encontrados na análise



Fonte: Elaborado pelo autor (2025).

Como mostra a figura 23, através de novos cenários criados de forma simulada, foi possível alcançar 53,8% com a nota B e 26,9% com a nota A, sendo zerada a nota F, lembrando que no desempenho original, o problema começou com uma taxa de 51,7% de classificação na nota F e apenas 4,1% para a nota A.

Claro que essa é uma simulação elaborada por meio de funções condicionais do Python, de forma que os resultados podem trazer uma realidade muito além das expectativas, pois se analisarmos bem, devemos notar que a quantidade de registros caiu de 2392 para 26 registros, ou seja, nas condições propostas o Python conseguiu extrair da base de dados apenas esses 26 registros baseados nas condições extraídas na análise.

Esse tipo de simulação preditiva é proposta para se ter uma noção do que pode ser feito ao se tomar decisões baseadas em dados, o mais sensato é continuar com a entrada de novos dados na base de dados, com as novas estratégias já tomadas, realizando novas análises para verificar o comportamento das classificações de acordo com a entrada desses novos dados.

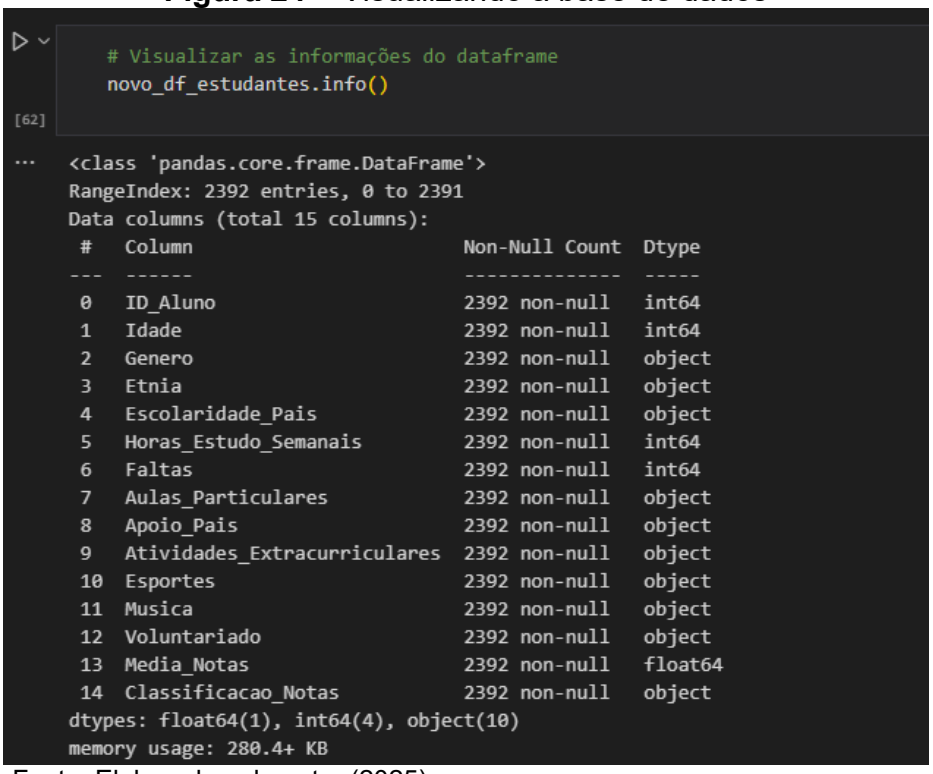
Ou então para criar previsões mais eficazes, também podem ser desenvolvidos modelos de aprendizado de máquina, o famoso machine learning, que usa inteligência artificial a partir de algoritmos baseados em dados e treinados para prever novas situações através de dados já disponíveis ou de novos dados coletados. Essa será a segunda parte deste projeto que será observado a seguir.

Criação da Inteligência Artificial e Previsões (Modelo de Aprendizado de Máquina)

Agora será dodo início a segunda parte do projeto deste artigo científico, que é o desenvolvimento dos modelos de aprendizado de máquina em Python, através da biblioteca Scikit-learn, para a realização das previsões de novas classificações baseadas nos dados analisados. A finalidade do projeto é criar um modelo que consiga ler as informações dos estuantes e dizer automaticamente a classificação de suas notas: A, B, C, D e F. Para os próximos passos, será seguida a sequência da análise de dados.

Passo 7 – Preparar a base de dados para a inteligência artificial

Figura 24 – Visualizando a base de dados



```
[62]
    # Visualizar as informações do dataframe
    novo_df_estudantes.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   ID_Aluno                   2392 non-null   int64
 1   Idade                      2392 non-null   int64
 2   Genero                     2392 non-null   object
 3   Etnia                      2392 non-null   object
 4   Escolaridade_Pais          2392 non-null   object
 5   Horas_Estudo_Semanais      2392 non-null   int64
 6   Faltas                     2392 non-null   int64
 7   Aulas_Particulares         2392 non-null   object
 8   Apoio_Pais                 2392 non-null   object
 9   Atividades_Extracurriculares  2392 non-null   object
 10  Esportes                   2392 non-null   object
 11  Musica                     2392 non-null   object
 12  Voluntariado               2392 non-null   object
 13  Media_Notas                2392 non-null   float64
 14  Classificacao_Notas        2392 non-null   object
dtypes: float64(1), int64(4), object(10)
memory usage: 280.4+ KB
```

Fonte: Elaborado pelo autor (2025).

Analisando a figura 24, que contempla as informações atuais da base de dados, através da função **info()** do Python, função esta que traz informações como tamanho da base de dados, suas colunas, e para o que interessa no momento, os tipos de dados de cada coluna.

No começo do projeto da análise de dados, no momento da importação da base de dados, onde os registros aparecem todos numerados na figura 2, e tiveram que ser transformadas em textos de acordo como os valores no mapa de informações. Agora observe na figura que no Dtypes (tipo dos dados), a maior parte das colunas estão no formato de texto, o que impedirá o treino da inteligência artificial nesse formato, pois como mencionado anteriormente, processadores de computadores não conseguem ler textos e sim números binários.

Então o que é preciso ser feito aqui neste momento, é o mesmo que foi feito no início, mas agora o oposto, aqui vamos converter os campos em texto para valores numéricos novamente. Mas nesse caso, será feito de uma forma diferente da anterior, onde antes foi utilizado o método replace() do Python para transformar os números em valores de texto informados no mapa de descrições, aqui será utilizado um método do scikit-learn, chamado de LabelEncoder[12], que fará isso de uma forma mais eficaz.

[12] `LabelEncoder` da biblioteca scikit-learn é uma classe utilitária usada para converter rótulos (labels) ou variáveis categóricas não numéricas em valores numéricos inteiros.

Figura 25 – Codificando as colunas através do método LabelEncoder para a inteligência artificial

```
# Codificar as colunas para tipo numérico para a Inteligência Artificial

from sklearn.preprocessing import LabelEncoder

# Codificar coluna Genero
codificador_genero = LabelEncoder()
novo_df_estudantes["Genero"] = codificador_genero.fit_transform(novo_df_estudantes["Genero"])

# Codificar coluna Etnia
codificado_etnia = LabelEncoder()
novo_df_estudantes["Etnia"] = codificado_etnia.fit_transform(novo_df_estudantes["Etnia"])

# Codificar coluna Escolaridade_Pais
codificador_escola_pais = LabelEncoder()
novo_df_estudantes["Escolaridade_Pais"] = codificador_escola_pais.fit_transform(novo_df_estudantes["Escolaridade_Pais"])

# Codificar coluna Aulas Particulares
codificador_particulares = LabelEncoder()
novo_df_estudantes["Aulas_Particulares"] = codificador_particulares.fit_transform(novo_df_estudantes["Aulas_Particulares"])

# Codificar coluna Apoio_Pais
codificador_apoio = LabelEncoder()
novo_df_estudantes["Apoio_Pais"] = codificador_apoio.fit_transform(novo_df_estudantes["Apoio_Pais"])
```

Fonte: Elaborado pelo autor (2025).

Veja agora o resultado na figura 26:

Figura 26 – Informações da base de dados com as colunas todas no tipo numérico

```
novo_df_estudantes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID_Aluno                    2392 non-null   int64
 1   Idade                       2392 non-null   int64
 2   Genero                      2392 non-null   int64
 3   Etnia                       2392 non-null   int64
 4   Escolaridade_Pais           2392 non-null   int64
 5   Horas_Estudo_Semanais       2392 non-null   int64
 6   Faltas                      2392 non-null   int64
 7   Aulas_Particulares          2392 non-null   int64
 8   Apoio_Pais                  2392 non-null   int64
 9   Atividades_Extracurriculares 2392 non-null  int64
 10  Esportes                    2392 non-null   int64
 11  Musica                      2392 non-null   int64
 12  Voluntariado                2392 non-null   int64
 13  Media_Notas                 2392 non-null   float64
 14  Classificacao_Notas         2392 non-null   object
dtypes: float64(1), int64(13), object(1)
memory usage: 280.4+ KB
```

Fonte: Elaborado pelo autor (2025).

Todas as colunas agora estão no formato numérico, dos tipos inteiros e decimais. Perceba que a única coluna que e ainda está com o tipo texto é a Classificação das Notas, e não por acaso, isso porque será ela a coluna à ser prevista e que exibirá as notas de A a F pelo modelo de aprendizado de máquina.

Figura 27 – Novo formato da base de dados

```
# Visualização do dataframe após a codificação
display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais
0	1	17	0	2	0	19	7	1	2
1	2	18	1	2	1	15	0	0	1
2	3	15	1	1	2	4	26	0	2
3	4	17	0	2	2	10	14	0	0
4	5	17	0	2	0	4	17	1	0
...	...	...	...	...	...	...	...	...	...
2387	2388	18	0	2	2	10	2	0	3
2388	2389	17	1	2	1	7	4	1	3
2389	2390	16	0	2	0	6	20	0	2
2390	2391	16	0	0	3	12	17	0	2
2391	2392	16	0	2	0	17	13	0	2

2392 rows × 15 columns

Fonte: Elaborado pelo autor (2025).

A base de dados continua com os mesmo registros, mas agora interpretados através de números como estava originalmente para que seja possível treinar o modelo.

Figura 28 – Preparar a base de dados para o treino

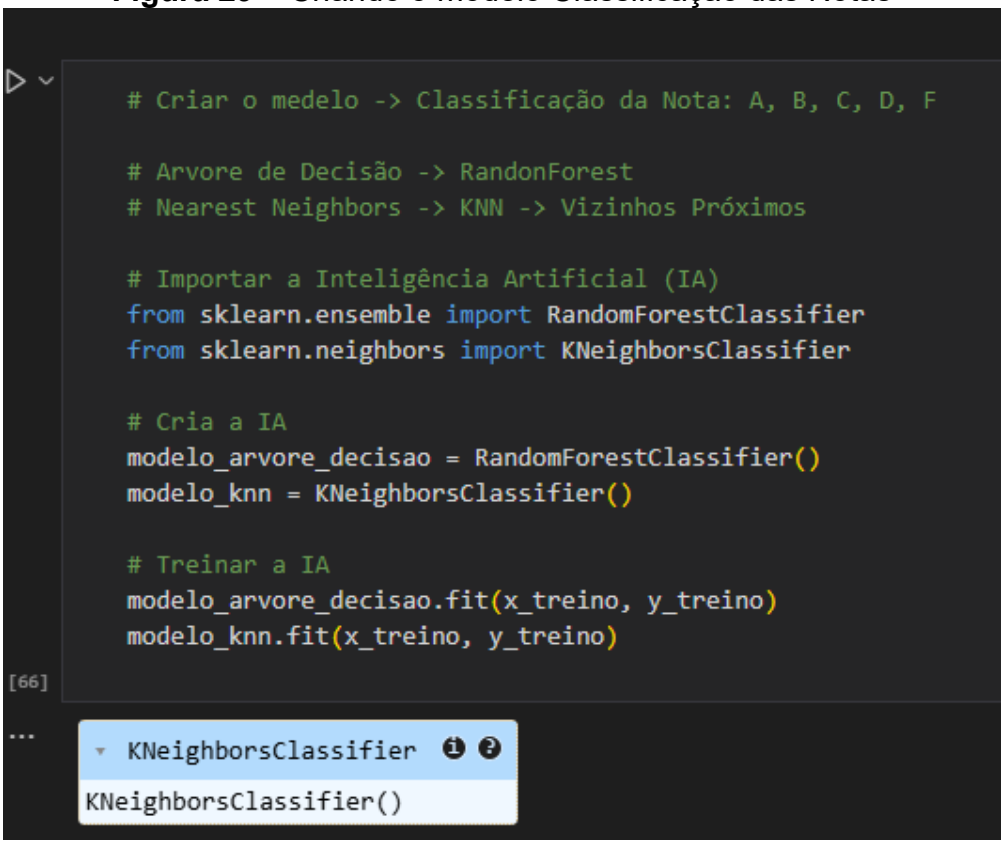
```
# Preparar o dataframe para o treino

# y é a coluna do dataframe a ser prevista
y = novo_df_estudantes["Classificacao_Notas"]

# x são as colunas do dataframe que serão usadas para a previsão
x = novo_df_estudantes.drop(columns=["Classificacao_Notas", "ID_Aluno"])

# Separar os dados de treino e os dados de teste
from sklearn.model_selection import train_test_split

x_treino, x_teste, y_treino, y_teste = train_test_split(x, y, test_size=0.7)
```

Fonte: Elaborado pelo autor (2025).

Com o código exibido na figura 28, separamos a base de dados em duas partes que são **x** e **y**, onde x será todas as colunas da base de dados que serão utilizadas para a previsão, e y será a coluna da base de dados que será prevista, no caso a coluna Classificação das Notas.

Passo 8 – Treinar a inteligência artificial

Figura 29 – Criando o modelo Classificação das Notas



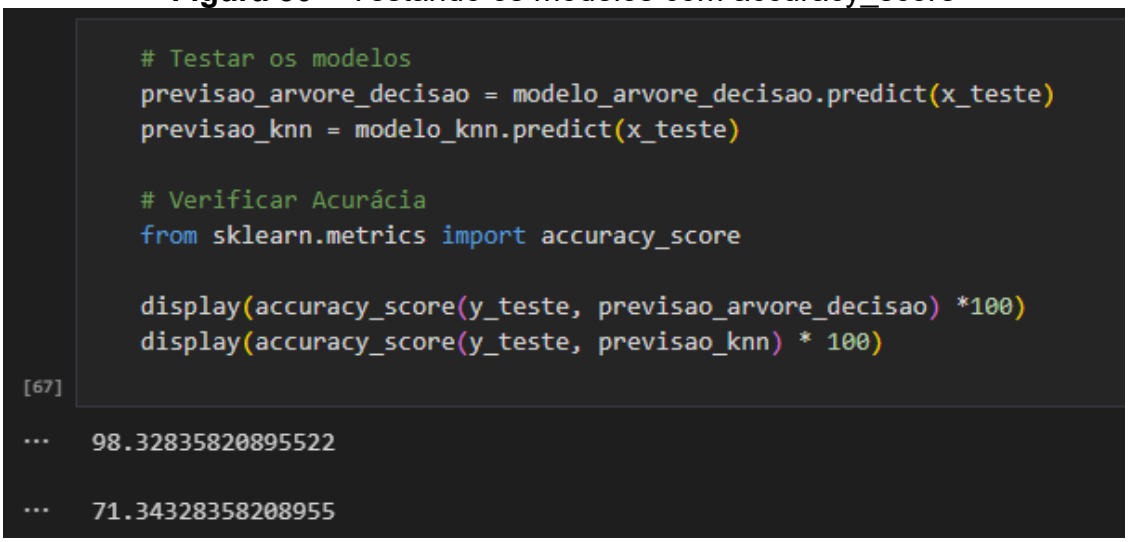
Fonte: Elaborado pelo autor (2025).

Neste passo conforme a figura 29, é treinada a inteligência artificial a partir dos modelos de classificação disponibilizadas pela biblioteca scikit-learn, as quais foram escolhidas dois modelos mais utilizados: **árvore de decisão** e **vizinhos próximos**.

Passo 9 – Escolher o melhor modelo

Com os dois modelos treinados, é preciso testá-los e verificar qual dos dois se sai melhor nas previsões, para isso será utilizado a função **accuracy_score** da biblioteca scilkit learn, que é uma métrica de avaliação de acurácia, que serve para medir a taxa de acerto do modelo como pode ser visto na figura 30.

Figura 30 – Testando os modelos com accuracy_score



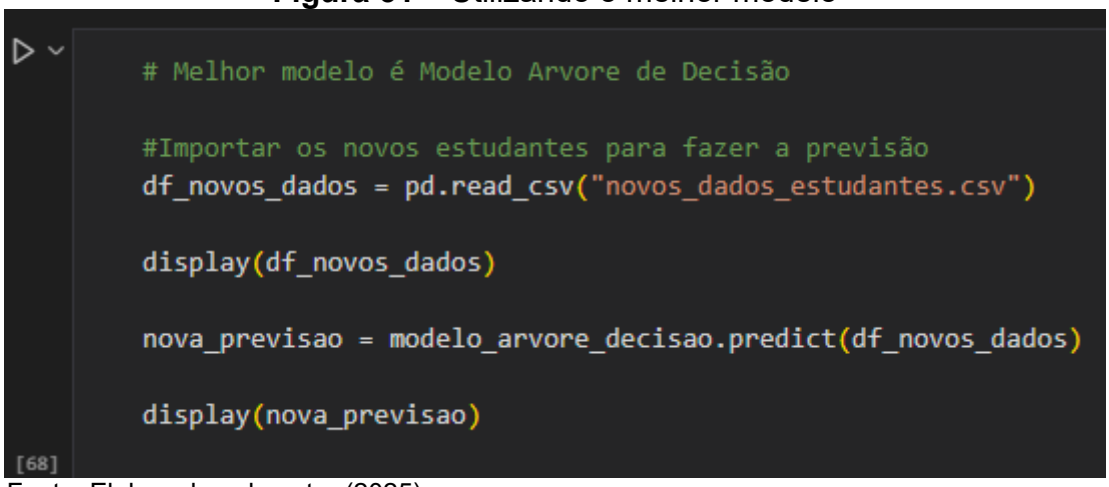
Fonte: Elaborado pelo autor (2025).

Como pode ser visto na figura 30, o modelo árvore de decisão se saiu melhor que o modelo vizinhos próximos. O árvore de decisão alcançou uma acurácia de 98,3% contra 71,3% do vizinhos próximos. Isso significa que no modelo árvore de decisão a cada 100 previsões, ele conseguiu acertar 98 delas, que para este projeto está excelente. Por essa razão será utilizado o modelo árvore de decisão.

Passo 10 – Utilizar o melhor modelo

Agora será utilizado o modelo escolhido para fazer as previsões a partir de novos dados através de um nova base de dados com 258 registros como demonstrado na figura 31.

Figura 31 – Utilizando o melhor modelo



Fonte: Elaborado pelo autor (2025).

Figura 32 – Resultado das previsões

```
array(['F', 'F', 'B', 'F', 'F', 'C', 'C', 'C', 'A', 'C', 'C', 'F', 'D',
       'C', 'C', 'D', 'F', 'F', 'F', 'F', 'F', 'F', 'C', 'F', 'F', 'D',
       'D', 'D', 'B', 'B', 'F', 'F', 'F', 'B', 'C', 'F', 'C', 'F', 'F',
       'C', 'F', 'F', 'F', 'D', 'F', 'F', 'F', 'C', 'B', 'A', 'B', 'F',
       'D', 'F', 'F', 'F', 'F', 'A', 'D', 'B', 'F', 'B', 'A', 'C', 'A',
       'C', 'D', 'F', 'D', 'B', 'A', 'D', 'B', 'D', 'D', 'F', 'C', 'D',
       'B', 'F', 'F', 'D', 'D', 'C', 'A', 'D', 'F', 'C', 'C', 'F', 'F',
       'F', 'B', 'D', 'F', 'C', 'F', 'F', 'B', 'F', 'A', 'C', 'D', 'F',
       'C', 'D', 'D', 'F', 'F', 'F', 'D', 'C', 'F', 'F', 'C', 'F', 'C',
       'B', 'B', 'F', 'D', 'B', 'B', 'F', 'B', 'C', 'F', 'F', 'F', 'D',
       'D', 'D', 'C', 'F', 'B', 'F', 'F', 'F', 'F', 'F', 'B', 'D', 'F',
       'D', 'F', 'F', 'F', 'F', 'C', 'F', 'B', 'F', 'D', 'F', 'F', 'C',
       'F', 'F', 'F', 'D', 'D', 'C', 'F', 'F', 'F', 'F', 'F', 'B', 'F',
       'F', 'F', 'D', 'F', 'C', 'B', 'F', 'F', 'F', 'B', 'F', 'C', 'D',
       'F', 'D', 'F', 'F', 'F', 'C', 'F', 'D', 'B', 'B', 'F', 'D', 'B',
       'C', 'F', 'A', 'D', 'F', 'F', 'F', 'F', 'D', 'F', 'F', 'F', 'F',
       'F', 'F', 'D', 'F', 'D', 'C', 'F', 'F', 'D', 'F', 'F', 'B', 'F',
       'D', 'B', 'F', 'F', 'F', 'C', 'D', 'F', 'F', 'B', 'C', 'B', 'F',
       'B', 'D', 'C', 'F', 'F', 'F', 'F', 'D', 'B', 'D', 'D', 'B', 'F',
       'F', 'F', 'B', 'F', 'F', 'C', 'F', 'F', 'B', 'C', 'D'],
      dtype=object)
```

Fonte: Elaborado pelo autor (2025).

Na figura 32, temos uma **array**, que em Python, são estruturas de dados multidimensionais que armazenam uma coleção de elementos de um mesmo tipo. As informações da figura são os valores da coluna Classificação das Notas previstas pelo modelo árvore de decisão, que no caso, são as 258 previsões dos dados da nova base de dados **novos_dados_estudantes.csv**.

Para melhor entender o funcionamento do modelo de previsão, será feito uma análise exploratória desses novos dados como demonstrado na figura 33.

Figura 33 – Análise exploratória inicial das novas classificações de notas

```
# Análise exploratória inicial da classificação das notas

# Contagem das notas da classificação

df_novos_dados["Classificacao_Notas"] = nova_previsao

display(df_novos_dados["Classificacao_Notas"].value_counts())

# Proporção das classificações
display(df_novos_dados["Classificacao_Notas"].value_counts(normalize=True).map("{:.1%}".format))

display(df_novos_dados)
```

[69]

```
Classificacao_Notas
F    126
D     48
C     39
B     36
A      9
Name: count, dtype: int64

Classificacao_Notas
F    48.8%
D    18.6%
C    15.1%
B    14.0%
A     3.5%
Name: proportion, dtype: object
```

Fonte: Elaborado pelo autor (2025).

Na análise da nova base de dados, observa-se que os novos dados ainda demonstram que o desempenho dos estudantes continua baixo e que é preciso serem tomadas melhores decisões estratégicas para elevar o aumento do desempenho.

Mas neste caso, houve uma melhora significativa no desempenho em relação com a base de dados original, onde a nota F estava acima de 50% ante 48,8% desta nova base de dados, o que prova que o modelo de aprendizado de máquina traz proporcionalmente um resultado fiel, ainda que a nova base de dados tenha menos dados que o original.

Um recurso interessante nos modelos de aprendizado de máquina é a função **feature_importances_** da biblioteca scikit-learn do Python. Com essa função é possível analisar quais as características são mais importantes para definir, neste caso, a classificação das notas a partir das outras variáveis, como é exibido na figura 34.

Figura 34 – Características mais importantes para definir a classificação das notas

```
# Características mais importantes para definir a Classificação das Notas

colunas = list(x_teste.columns)
importancia = pd.DataFrame(index=colunas, data=modelo_arvore_decisao.feature_importances_)
importancia = importancia * 100
print(importancia)
```

```
[70]
...
                                    0
Idade                        2.239083
Genero                       1.108586
Etnia                        1.949609
Escolaridade_Pais            2.229692
Horas_Estudo_Semanais        4.809006
Faltas                      23.406254
Aulas_Particulares           1.337815
Apoio_Pais                   2.784290
Atividades_Extracurriculares 1.119025
Esportes                     1.012912
Musica                       0.965540
Voluntariado                 0.768735
Media_Notas                 56.269454
```

Fonte: Elaborado pelo autor (2025).

A função feature_importance_ apontou que as variáveis mais importantes a serem observadas na classificação das notas, resultando no baixo desempenho dos estudantes foi em primeiro lugar, as médias das notas com 56,2% de atenção, seguida das faltas com 23,4%, e na sequência as horas de estudos semanais com 4,8% e o apoio dos pais com 2,7%.

Observe que a função retornou os mesmos insights extraídos na análise exploratória dos dados, indicando que a variável média notas é a mais importante e é definida de acordo com o desempenho das outras variáveis. É claro que essa função não substitui a análise

exploratória, mas pode indicar onde é necessário dar mais atenção na hora da análise de dados.

5 CONSIDERAÇÕES FINAIS

Este trabalho de Análise de Dados combinado com um Modelo de Aprendizado de Máquina, em modo experimental atendeu aos objetivos esperados, sejam eles, no campo cientifico, didático e técnico.

No campo cientifico por se tratar de um projeto baseado em funções estatísticas e matemáticas, com base em pesquisa quantitativa, exploratória e aplicada; no campo didático, por utilizar uma base de dados, que embora seja simulada para fins de estudo e treinamento, foi muito bem elaborada trazendo resultados significativos após a análise, podendo futuramente na prática, ser aplicada em toda esfera da educação pública; e no campo técnico, por se tratar de um projeto totalmente prático desenvolvido através de uma linguagem de programação poderosa e popular no mundo da programação e do desenvolvimento de sistemas, o Python, obtendo resultados precisos e expressivos.

Apesar do trabalho ter evidenciado um sério problema na educação pública, existe uma certa limitação no estudo por se tratar de uma base de dados simulada que ainda necessite de uma aplicabilidade à realidade brasileira.

É sugerido para futuras pesquisas o uso de dados reais originados da rede pública de ensino, integração com sistemas educacionais, ou comparações entre modelos de aprendizado de máquina.

O projeto deste artigo científico, deve servir de inspiração para a gestão pública não somente no setor educacional, mas em diversos setores do serviço público em geral, para se orientarem através de estratégias e decisões baseadas em dados evidentes através da análise e ciência de dados.

REFERÊNCIAS

BAKER, R. S.; YACEF, K. **The state of educational data mining in 2009: A review and future visions.** Journal of Educational Data Mining, v. 1, n. 1, p. 3-17, 2009.

BRITO, L. M.; COSTA, A. C.; ANDRADE, M. F. Fatores associados ao desempenho escolar: uma análise com estudantes do ensino fundamental. **Estudos e Pesquisas em Psicologia,** Rio de Janeiro, v. 20, n. 1, p. 9–27, 2020.

DAVENPORT, T. H.; RONANKI, R. Artificial Intelligence for the Real World. **Harvard Business Review,** Boston, v. 96, n. 1, p. 108–116, jan./fev. 2018.

FERREIRA, R. G C.; MIRANDA, L. B. A de; PINTO, R. A.; et al. **Preparação e Análise Exploratória de Dados.** Porto Alegre: SAGAH, 2021. E-book. p.14. ISBN 9786556902890. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9786556902890/. Acesso em: 18 ago. 2025.

JANSSEN, M.; HELBIG, N. Innovating and changing the policy-cycle: Policy-makers be prepared! **Government Information Quarterly**, v. 35, n. 2, p. S1–S4, 2018.

KULKARNI, C.; CAMBRE, J.; CHRISTOPHER, J. **Personalized e-learning systems: Analysis of current research challenges and directions for future work.** Computers in Human Behavior, v. 55, p. 1185-1202, 2016.

LUCKESI, C. C. **Avaliação da aprendizagem escolar: estudos e proposições.** 15. ed. São Paulo: Cortez, 2002.

MATTAR, J.; RAMOS, D. K. **Metodologia da pesquisa em educação: Abordagens Qualitativas, Quantitativas e Mistas.** São Paulo: Almedina Brasil, 2021. E-book. p.118; 289. ISBN 9786586618518. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/9786586618518/. Acesso em: 18 ago. 2025.

MCKINNEY, W. **Python para Análise de Dados: Tratamento de Dados com Pandas, Numpy e Ipython.** São Paulo: Novatec Editora, 2018. E-book p.26.

PORTAL F10. **Como o Machine Learning está transformando a educação**. Curitiba, 2021. Disponível em: <https://blog.f10.com.br/como-o-machine-learning-esta-transformando-a-educacao/>. Acesso em: 10 novembro 2025.

PROVOST, F.; FAWCETT, T. **Data Science for Business: What You Need to Know About Data Mining and Data-Analytic Thinking.** Sebastopol: O'Reilly Media, 2013.

RUSSELL, S.; NORVIG, P. **Inteligência Artificial.** 3. ed. São Paulo: Pearson, 2020.

SIEMENS, G. **Learning analytics: The emergence of a discipline.** American Behavioral Scientist, v. 57, n. 10, p. 1380-1400, 2013.

APÊNDICES

APÊNDICE A – CADERNOS JUPYTER NOTEBOOK (ESTUDANTES.ipynb E ESTUDANTES_IA.ipynb)

Este apêndice apresenta os cadernos Jupyter Notebook, que são os arquivos executáveis, utilizados na execução prática do projeto de artigo científico de TCC. O documento foi exportado diretamente do Visual Studio Code no formato PDF, e contém o código-fonte completo e totalmente bem documentado, visualizações gráficas, estatísticas descritivas, simulações das previsões realizadas através de funções condicionais e a construção dos modelos de aprendizado de máquina. O notebook (caderno) foi desenvolvido com a linguagem de programação Python, e executado no ambiente Jupyter Notebook integrado ao Visual Studio Code.

A inclusão deste material tem como objetivo, garantir a transparência metodológica, permitir a reprodutibilidade dos resultados e demonstrar o domínio técnico do autor na aplicação de análise de dados voltada à gestão pública.

🚀 Projeto - Análise de Dados Desempenho Acadêmico

Pergunta Norteadora:

"Quais fatores demográficos, acadêmicos e comportamentais mais influenciam a classificação e a média de notas dos estudantes?"

A variável de interesse é `Classificacao_Notas`. Investigar como elas se relacionam com outros fatores, como idade, gênero, faltas, horas de estudo e participação em atividades extracurriculares.

Este conjunto de dados contém informações abrangentes sobre 2.392 alunos do ensino médio, detalhando seus dados demográficos, hábitos de estudo, envolvimento dos pais, atividades extracurriculares e desempenho acadêmico. A variável-alvo, , classifica as notas dos alunos em categorias distintas, fornecendo um conjunto de dados robusto para pesquisa educacional, modelagem preditiva e análise estatística.

📚 Origem dos Dados

Este dataset foi disponibilizado por Rabie El Kharoua no Kaggle e está licenciado sob CC BY 4.0:

> El Kharoua, R. (2024). Students Performance Dataset.
> Disponível em: Kaggle Dataset Link.

Passos do Projeto

Passo 1 - Importar o DataFrame

Passo 2 - Visualizar o DataFrame

Passo 3 - Corrigir erros no DataFrame

Passo 4 - Analisar o desempenho acadêmico

Passo 5 - Analisar as variáveis que influenciam no desempenho acadêmico

Passo 6 - Fazer uma análise preditiva para melhorar o desempenho acadêmico, caso seja necessário

---

Passo 1 - Importar o DataFrame

In [1]:
```
# Importando as bibliotecas necessárias
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Importando o dataset
df_dados_estudantes = pd.read_csv("dados_estudantes.csv", sep= ',')

# Verificando as primeiras linhas do dataset
display(df_dados_estudantes)
```

	StudentID	Age	Gender	Ethnicity	ParentalEducation	StudyTimeWeekly	Absences	Tutoring	ParentalSupport	Extracurricular	S
0	1001	17	1	0	2	19.833723	7	1	2	0	
1	1002	18	0	0	1	15.408756	0	0	1	0	
2	1003	15	0	2	3	4.210570	26	0	2	0	
3	1004	17	1	0	3	10.028829	14	0	3	1	
4	1005	17	1	0	2	4.672495	17	1	3	0	
...	...	...	...	...	...	...	...	...	...	...	
2387	3388	18	1	0	3	10.680555	2	0	4	1	
2388	3389	17	0	0	1	7.583217	4	1	4	0	
2389	3390	16	1	0	2	6.805500	20	0	2	0	
2390	3391	16	1	1	0	12.416653	17	0	2	0	
2391	3392	16	1	0	2	17.819907	13	0	2	0	

2392 rows × 15 columns

In [2]:
```
traducao_colunas = {
    "StudentID": "ID_Aluno",
    "Age": "Idade",
    "Gender": "Genero",
    "Ethnicity": "Etnia",
    "ParentalEducation": "Escolaridade_Pais",
    "StudyTimeWeekly": "Horas_Estudo_Semanais",
    "Absences": "Faltas",
    "Tutoring": "Aulas_Particulares",
    "ParentalSupport": "Apoio_Pais",
    "Extracurricular": "Atividades_Extracurriculares",
    "Sports": "Esportes",
    "Music": "Musica",
    "Volunteering": "Voluntariado",
    "GPA": "Media_Notas",
    "GradeClass": "Classificacao_Notas"
}

# Renomear as colunas
df_dados_estudantes = df_dados_estudantes.rename(columns=traducao_colunas)

df_dados_estudantes.head()
```

Out[2]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extra
0	1001	17	1	0	2	19.833723	7	1	2	
1	1002	18	0	0	1	15.408756	0	0	1	
2	1003	15	0	2	3	4.210570	26	0	2	
3	1004	17	1	0	3	10.028829	14	0	3	
4	1005	17	1	0	2	4.672495	17	1	3	

Passo 2 - Visualizar o DataFrame

In [3]:
```
# Usando shape para conferir as dimensões do dataframe
df_dados_estudantes.shape
```

Out[3]: (2392, 15)

In [4]:
```
# Conferindo os tipos de dados utilizados no dataframe com dtypes
df_dados_estudantes.dtypes
```

Out[4]:
```
ID_Aluno                          int64
Idade                             int64
Genero                            int64
Etnia                             int64
Escolaridade_Pais                 int64
Horas_Estudo_Semanais           float64
Faltas                            int64
Aulas_Particulares                int64
Apoio_Pais                        int64
Atividades_Extracurriculares      int64
Esportes                          int64
Musica                            int64
Voluntariado                      int64
Media_Notas                     float64
Classificacao_Notas             float64
dtype: object
```

In [5]:
```
# Visualizando as informaçoes do dataframe com info
df_dados_estudantes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                       Non-Null Count  Dtype  
---  ------                       --------------  -----  
 0   ID_Aluno                     2392 non-null   int64  
 1   Idade                        2392 non-null   int64  
 2   Genero                       2392 non-null   int64  
 3   Etnia                        2392 non-null   int64  
 4   Escolaridade_Pais            2392 non-null   int64  
 5   Horas_Estudo_Semanais        2392 non-null   float64
 6   Faltas                       2392 non-null   int64  
 7   Aulas_Particulares           2392 non-null   int64  
 8   Apoio_Pais                   2392 non-null   int64  
 9   Atividades_Extracurriculares 2392 non-null   int64  
 10  Esportes                     2392 non-null   int64  
 11  Musica                       2392 non-null   int64  
 12  Voluntariado                 2392 non-null   int64  
 13  Media_Notas                  2392 non-null   float64
 14  Classificacao_Notas          2392 non-null   float64
dtypes: float64(3), int64(12)
memory usage: 280.4 KB
```

In [6]:
```
# Conferir se existe valores ausentes no dataframe
df_dados_estudantes.isna().sum()
```

Out[6]:
```
ID_Aluno                        0
Idade                           0
Genero                          0
Etnia                           0
Escolaridade_Pais               0
Horas_Estudo_Semanais           0
Faltas                          0
Aulas_Particulares              0
Apoio_Pais                      0
Atividades_Extracurriculares    0
Esportes                        0
Musica                          0
Voluntariado                    0
Media_Notas                     0
Classificacao_Notas             0
dtype: int64
```

In [7]:
```
# Por fim fazer uma análise descritiva do dataframe
df_dados_estudantes.describe()
```

Out[7]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares
count	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000
mean	2196.500000	16.468645	0.510870	0.877508	1.746237	9.771992	14.541388	0.301421
std	690.655244	1.123798	0.499986	1.028476	1.000411	5.652774	8.467417	0.458971
min	1001.000000	15.000000	0.000000	0.000000	0.000000	0.001057	0.000000	0.000000
25%	1598.750000	15.000000	0.000000	0.000000	1.000000	5.043079	7.000000	0.000000
50%	2196.500000	16.000000	1.000000	0.000000	2.000000	9.705363	15.000000	0.000000
75%	2794.250000	17.000000	1.000000	2.000000	2.000000	14.408410	22.000000	1.000000
max	3392.000000	18.000000	1.000000	3.000000	4.000000	19.978094	29.000000	1.000000

Passo 3 - Corrigir erros no DataFrame e remodelar os dados

Para seguir os padrões apresentados na Descrição das Colunas do DataFrame, irei alterar o tipo de dados de algumas colunas, seguindo o padrão descrito abaixo:

📊 Descrição das Colunas do DataFrame

Identificação do Estudante

- **ID_Aluno**: Identificador único para cada aluno (de 1001 a 3392).

👥 Detalhes Demográficos

- **Idade**: Idade dos estudantes, variando de 15 a 18 anos.
- **Genero**: Gênero do estudante:
  - 0: Masculino
  - 1: Feminino
- **Etnia**: Etnia dos estudantes:
  - 0: Caucasiano
  - 1: Afrodescendente
  - 2: Asiático
  - 3: Outra

👫 Educação dos Pais

- **Escolaridade_Pais**: Nível de escolaridade dos pais:
  - 0: Nenhum
  - 1: Ensino Médio
  - 2: Alguma Faculdade
  - 3: Graduação
  - 4: Pós-Graduação ou Superior

📚 Hábitos de Estudo

- **Horas_Estudo_Semanais**: Tempo de estudo semanal em horas (0 a 20).
- **Faltas**: Número de faltas no ano letivo (0 a 30).
- **Aulas_Particulares**: Participação em aulas particulares:
  - 0: Não
  - 1: Sim

👫 Envolvimento Parental

- **Apoio_Pais**: Nível de apoio dos pais:
  - 0: Nenhum
  - 1: Baixo
  - 2: Moderado
  - 3: Alto
  - 4: Muito Alto

🏫 Atividades Extracurriculares

- **Atividades_Extracurriculares**: Participação em atividades extracurriculares:
  - 0: Não
  - 1: Sim
- **Esportes**: Participação em esportes:
  - 0: Não
  - 1: Sim
- **Musica**: Participação em atividades musicais:
  - 0: Não
  - 1: Sim
- **Voluntariado**: Participação em atividades de voluntariado:
  - 0: Não
  - 1: Sim

📈 Desempenho Acadêmico

- **Media_Notas**: Média das notas (GPA), variando de 2.0 a 4.0.
- **Classificacao_Notas**: Classificação do aluno com base na média das notas:
  - 0: 'A' (Média >= 3.5)
  - 1: 'B' (3.0 <= Média < 3.5)
  - 2: 'C' (2.5 <= Média < 3.0)
  - 3: 'D' (2.0 <= Média < 2.5)
  - 4: 'F' (Média < 2.0)

In [8]:
```
# Alterando a coluna Media_Notas para float com 1 casa decimal
df_dados_estudantes[["Media_Notas"]] = df_dados_estudantes[["Media_Notas"]].astype(float).round(1)
```

In [9]:
```
# Alterando as colunas Horas_Estudo_Semanais e Classificacao_Notas para Int
df_dados_estudantes[["Horas_Estudo_Semanais",
                     "Classificacao_Notas"]] = df_dados_estudantes[["Horas_Estudo_Semanais",
                                                                    "Classificacao_Notas"]].astype(int)

# Visualizando a alteração das 3 colunas
df_dados_estudantes.dtypes
```

Out[9]:
```
ID_Aluno                      int64
Idade                         int64
Genero                        int64
Etnia                         int64
Escolaridade_Pais             int64
Horas_Estudo_Semanais         int64
Faltas                        int64
Aulas_Particulares            int64
Apoio_Pais                    int64
Atividades_Extracurriculares  int64
Esportes                      int64
Musica                        int64
Voluntariado                  int64
Media_Notas                 float64
Classificacao_Notas           int64
dtype: object
```

In [10]:
```
# Visualizando o dataframe após as alterações
df_dados_estudantes.head()
```

Out[10]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extra
0	1001	17	1	0	2	19	7	1	2	
1	1002	18	0	0	1	15	0	0	1	
2	1003	15	0	2	3	4	26	0	2	
3	1004	17	1	0	3	10	14	0	3	
4	1005	17	1	0	2	4	17	1	3	

In [11]:
```
# O ID_Aluno está sendo contando de 1001 à 3392, para alterar a contagem para 1 à 2392 usarei o método apply lambda para
# fazer essa alteração.

# Alterando os valores do ID_Aluno para contar do número 1 em diante
df_dados_estudantes["ID_Aluno"] = df_dados_estudantes["ID_Aluno"].apply(lambda x: x - 1000)
```

In [12]:
```
# Para alterar os valores numéricos pelos seus respectivos valores informados na descrição, usarei o método replace
# para reverter os números cadastrados em seus devidos valores informados.

# Alterando os valores da descrição da coluna Gênero
df_dados_estudantes["Genero"] = df_dados_estudantes["Genero"].replace({
    0: "Masculino",
    1: "Feminino"
})
```

In [13]:
```
# Alterando os valores da descrição da coluna Etnia
df_dados_estudantes["Etnia"] = df_dados_estudantes["Etnia"].replace({
    0: "Caucasiano",
    1: "Afrodescendente",
    2: "Asiático",
    3: "Outra"
})
```

In [14]:
```
# Alterando os valores da descrição da coluna Escolaridade dos Pais
df_dados_estudantes["Escolaridade_Pais"] = df_dados_estudantes["Escolaridade_Pais"].replace({
    0: "Nenhum",
    1: "Ensino Médio",
    2: "Alguma Faculdade",
```

```
    3: "Graduação",
    4: "Pós-Graduação ou Superior"
})
```

In [15]:
```
# Alterando os valores da descrição da coluna Aulas Particulares
df_dados_estudantes["Aulas_Particulares"] = df_dados_estudantes["Aulas_Particulares"].replace({
    0: "Não",
    1: "Sim"
})
```

In [16]:
```
# Alterando os valores da descrição da coluna Apoio dos Pais
df_dados_estudantes["Apoio_Pais"] = df_dados_estudantes["Apoio_Pais"].replace({
    0: "Nenhum",
    1: "Baixo",
    2: "Moderado",
    3: "Alto",
    4: "Muito Alto"
})
```

In [17]:
```
# Alterando os valores da descrição da coluna Atividades Extracurriculares
df_dados_estudantes["Atividades_Extracurriculares"] = df_dados_estudantes["Atividades_Extracurriculares"].replace({
    0: "Não",
    1: "Sim"
})
```

In [18]:
```
# Alterando os valores da descrição da coluna Esportes
df_dados_estudantes["Esportes"] = df_dados_estudantes["Esportes"].replace({
    0: "Não",
    1: "Sim"
})
```

In [19]:
```
# Alterando os valores da descrição da coluna Musica
df_dados_estudantes["Musica"] = df_dados_estudantes["Musica"].replace({
    0: "Não",
    1: "Sim"
})
```

In [20]:
```
# Alterando os valores da descrição da coluna Voluntariado
df_dados_estudantes["Voluntariado"] = df_dados_estudantes["Voluntariado"].replace({
    0: "Não",
    1: "Sim"
})
```

In [21]:
```
# Alterando os valores da descrição da coluna Classificacao_Notas
df_dados_estudantes["Classificacao_Notas"] = df_dados_estudantes["Classificacao_Notas"].replace({
    0: "A",
    1: "B",
    2: "C",
    3: "D",
    4: "F"
})
```

In [22]:
```
# Com a alteração de todos os valores necessários para suas devidas descrições, é hora de visualizar o resultado em
# um novo dataframe.

#Visualização do novo dataframe após a alteração das descrições das colunas
novo_df_estudantes = df_dados_estudantes

display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	At
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	
...	...	...	...	...	...	...	...	...	...	
2387	2388	18	Feminino	Caucasiano	Graduação	10	2	Não	Muito Alto	
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4	Sim	Muito Alto	
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20	Não	Moderado	
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17	Não	Moderado	
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13	Não	Moderado	

2392 rows × 15 columns

Ao analisar o novo DataFrame, observei que existem inconsistências nos valores em relação a coluna Media_Notas e Classificacao_Notas. Para corrigir esse problema e não haver divergências entre as análises, irei corrigir os valores usando o método apply com lambda de acordo com as informações descritas na Descrição do Desempenho Acadêmico.

In [23]:
```
# Aplicando o lambda para corrigir os valores da Classificacao_Notas em relação a Média_Notas
novo_df_estudantes["Classificacao_Notas_Lambda"] = novo_df_estudantes["Media_Notas"].apply(lambda x:
                                                                                           "A" if x >= 3.5 else
                                                                                           "B" if x >= 3.0 else
                                                                                           "C" if x >= 2.5 else
                                                                                           "D" if x >= 2.0 else
                                                                                           "F")

display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	At
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	
...	...	...	...	...	...	...	...	...	...	
2387	2388	18	Feminino	Caucasiano	Graduação	10	2	Não	Muito Alto	
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4	Sim	Muito Alto	
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20	Não	Moderado	
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17	Não	Moderado	
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13	Não	Moderado	

2392 rows × 16 columns

Como eu havia notado, os valores estavam divergentes. Isso pode ser observado no dataframe comparando-se os valores entre Classificacao_Notas e Classificacao_Notas_Lambda que eu criei.

In [24]:
```
# Excluindo a coluna Classificacao_Notas (com os valores errados)
novo_df_estudantes = novo_df_estudantes.drop(columns=["Classificacao_Notas"])

novo_df_estudantes.head()
```

Out[24]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividade
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	

In [25]:
```
# Renomeando a coluna Classificacao_Notas_Lambda para o nome original Classificacao_Notas
novo_df_estudantes = novo_df_estudantes.rename(columns={"Classificacao_Notas_Lambda": "Classificacao_Notas"})

novo_df_estudantes.head()
```

Out[25]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividade
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	

Agora que todas as colunas e seus valores foram remodelados e corrigidos, temos uma visão melhorada e precisa das informações.

É hora de extrair os insights.

Passo 4 - Análise dos dados

In [26]:
```
# Análise exploratória inicial da classificação das notas

# Contagem das notas da classificação
display(novo_df_estudantes["Classificacao_Notas"].value_counts())

# Proporção das classificações
display(novo_df_estudantes["Classificacao_Notas"].value_counts(normalize=True).map("{:.1%}".format))
```

```
Classificacao_Notas
F    1239
D     409
C     387
B     258
A      99
Name: count, dtype: int64
Classificacao_Notas
F    51.8%
D    17.1%
C    16.2%
B    10.8%
A     4.1%
Name: proportion, dtype: object
```

De acordo com a análise exploratória, observa-se que o desempenho dos estudantes é considerado sofrível, com 51.8% dos alunos classificados com a nota F e apenas 4.1% atingiram a nota A.

Passo 5 - Análise e visualização das variáveis que influenciam no desempenho acadêmico

---

In [27]:
```
# Configuração da ordem da categoria Classificação das Notas, que nos gráficos são mostrados fora da ordem de A à F.

# Variável para ajustar o parâmetro categories no historgrama
ordem_categoria = ["A", "B", "C", "D", "F"]

# Ordenando a coluna Classificaçao_Notas
novo_df_estudantes['Classificacao_Notas'] = pd.Categorical(novo_df_estudantes['Classificacao_Notas'], categories=ordem_cat
```

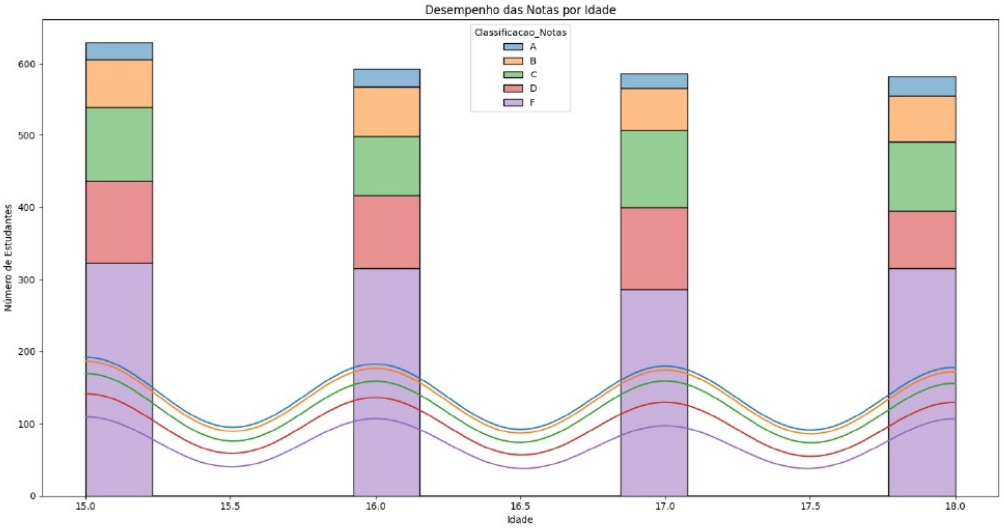
In [28]:
```
# Excluir campo ID_Aluno que é desnessário para a visualização dos dados
novo_df_estudantes = novo_df_estudantes.drop(columns="ID_Aluno")

novo_df_estudantes.head()
```

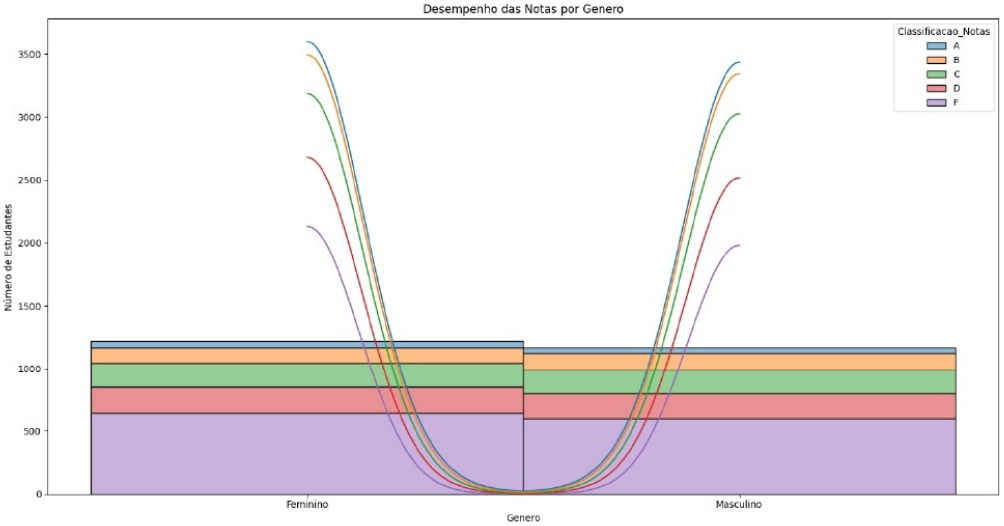
Out[28]:

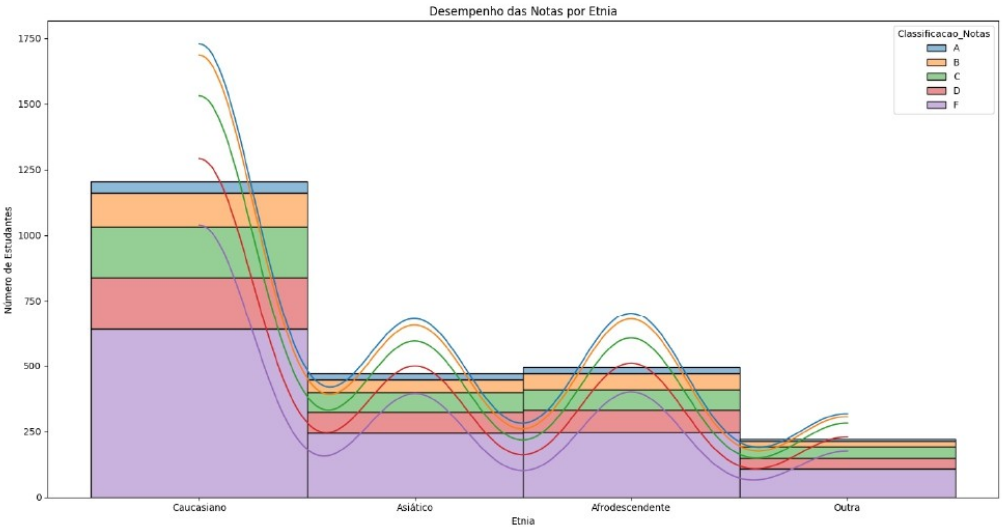
	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extracurr
0	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	

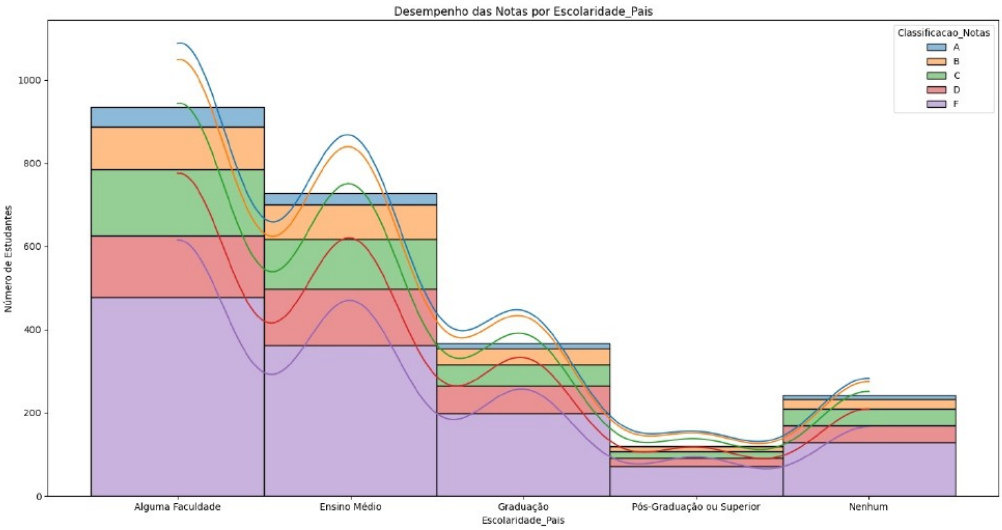
In [29]:

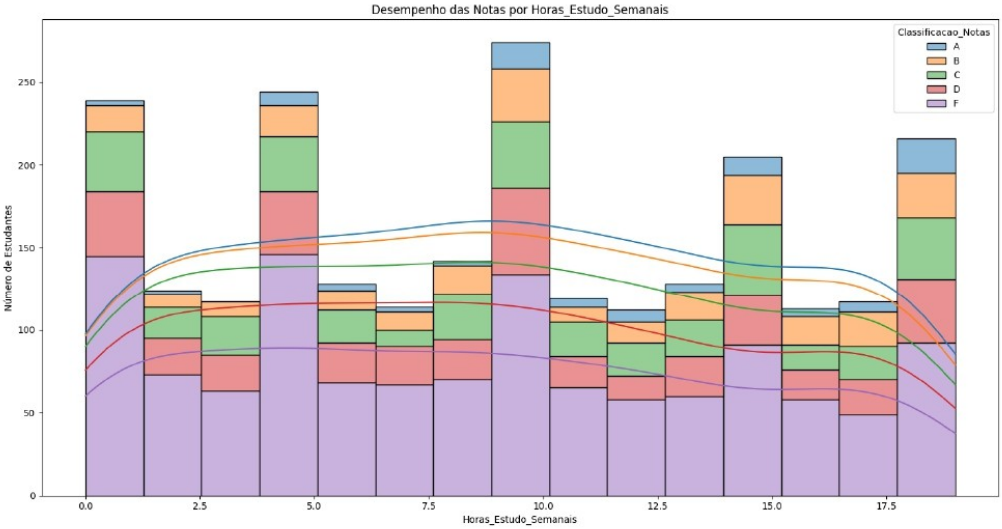
```
# Visualização e Análise do desempenho da classificação das notas (como as outras colunas do DataFrame impactam no
# desempenho das notas)

# Criação dos Gráficos
for coluna in novo_df_estudantes.columns:
    # Configuração da figura
    plt.figure(figsize=(15, 8))

    # gráficos de histograma - Desempenho das Notas VS Variáveis
    sns.histplot(data=novo_df_estudantes, x=coluna, hue="Classificacao_Notas", kde=True, multiple="stack")
    plt.title("Desempenho das Notas por " + coluna)
    plt.xlabel(coluna)
    plt.ylabel('Número de Estudantes')

    plt.tight_layout()
    plt.show()
```

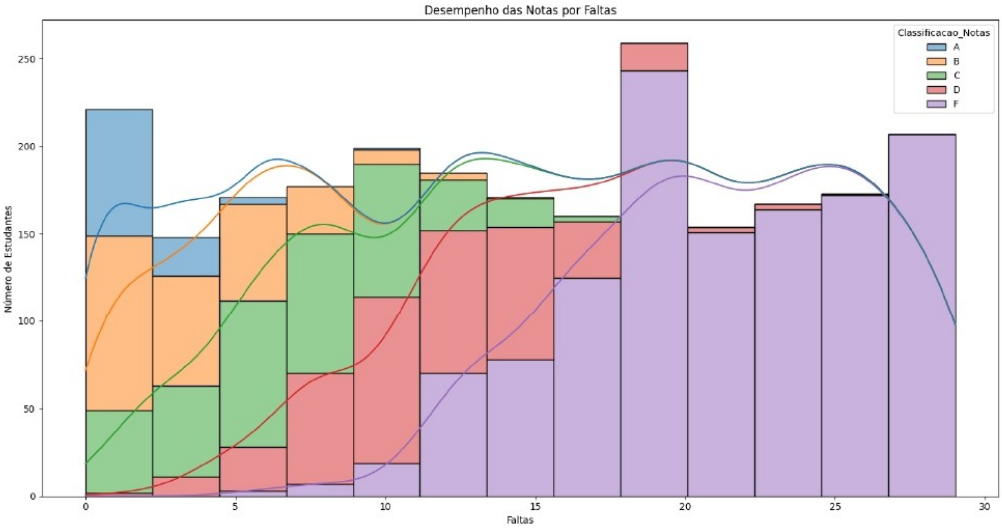


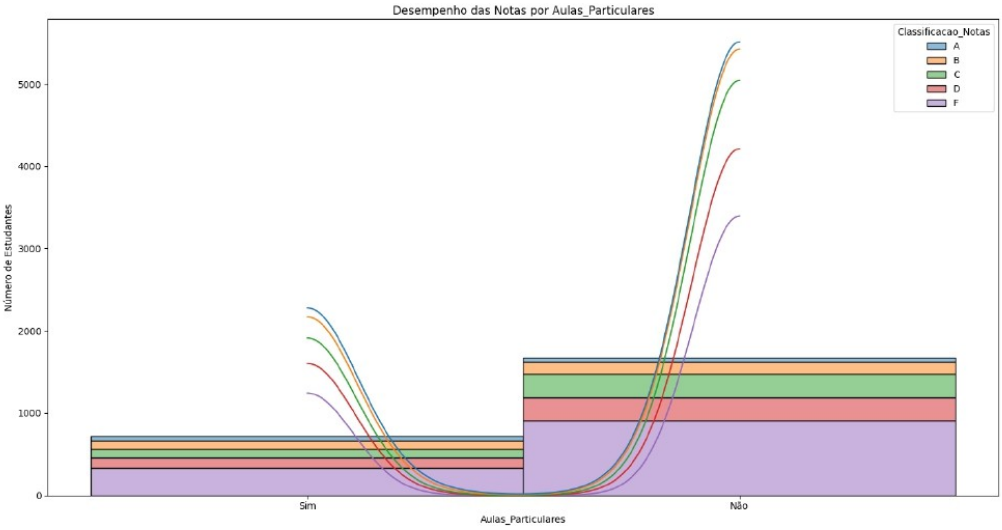


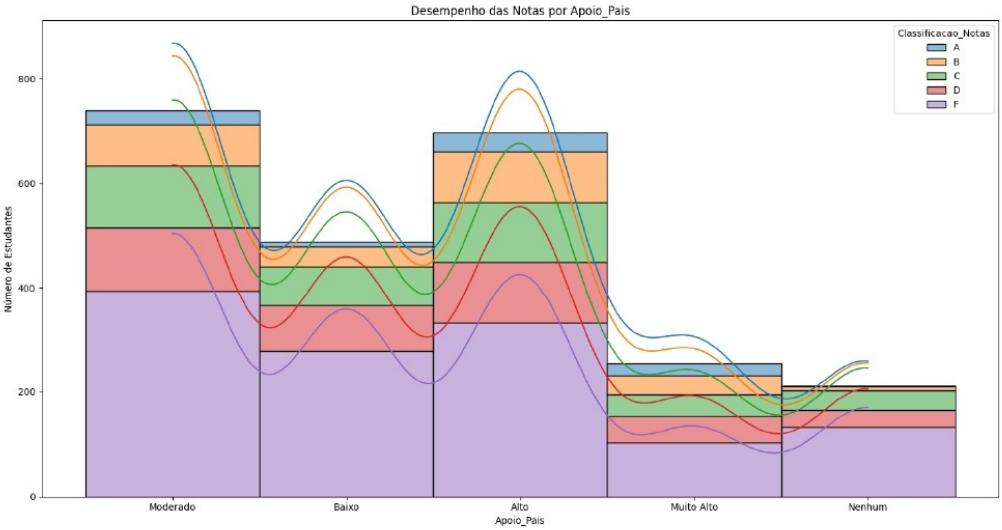

Desempenho das Notas por Etnia
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
250
500
750
1000
1250
1500
1750
Caucasiano
Asiático
Afrodescendente
Outra
Etnia

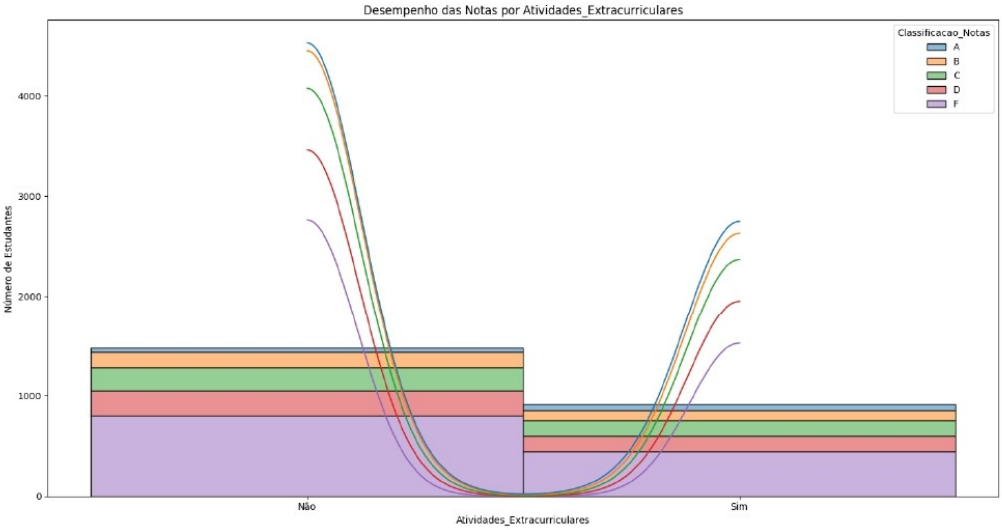

Desempenho das Notas por Escolaridade_Pais
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
200
400
600
800
1000
Alguma Faculdade
Ensino Médio
Graduação
Pós-Graduação ou Superior
Nenhum
Escolaridade_Pais

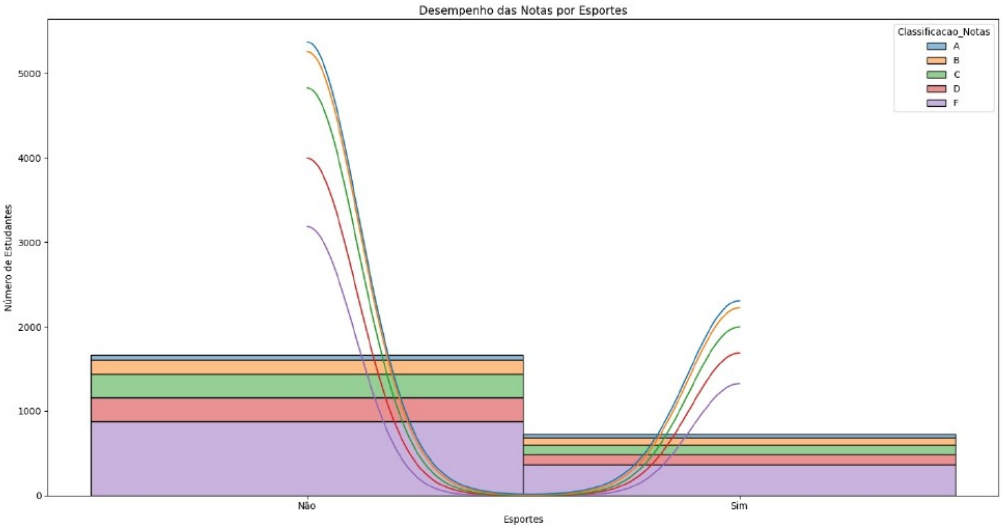

Desempenho das Notas por Horas_Estudo_Semanais
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
50
100
150
200
250
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
Horas_Estudo_Semanais

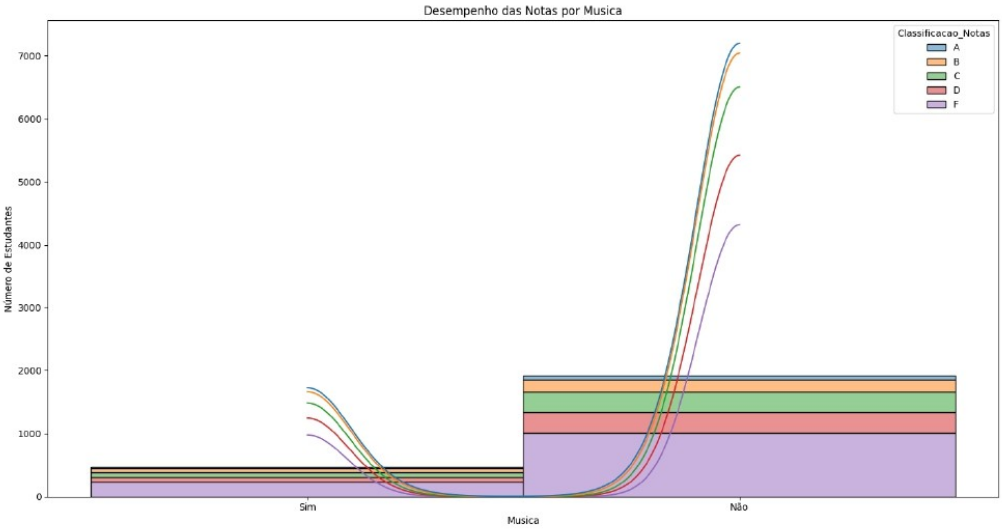

Desempenho das Notas por Faltas
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
Faltas
0
50
100
150
200
250
0
5
10
15
20
25
30

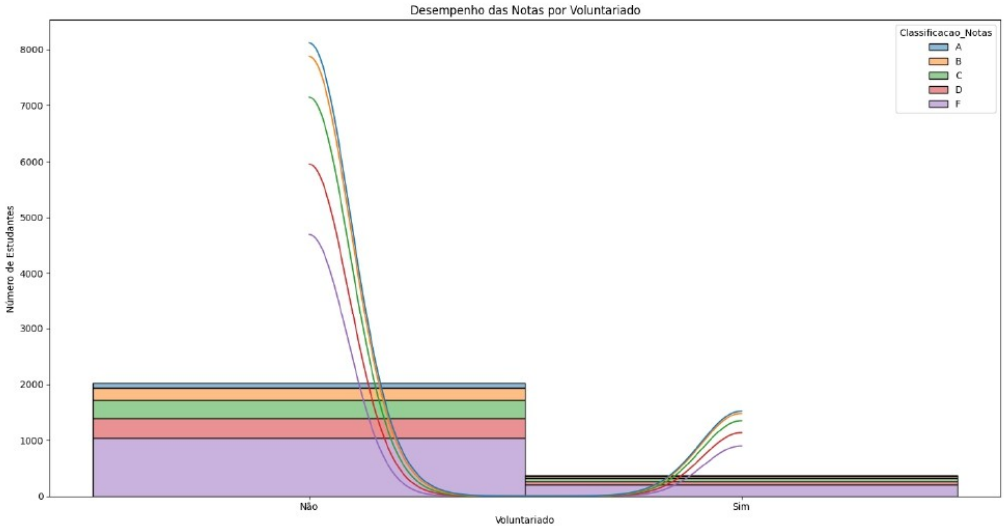

Desempenho das Notas por Aulas_Particulares
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
Aulas_Particulares
Sim
Não
0
1000
2000
3000
4000
5000

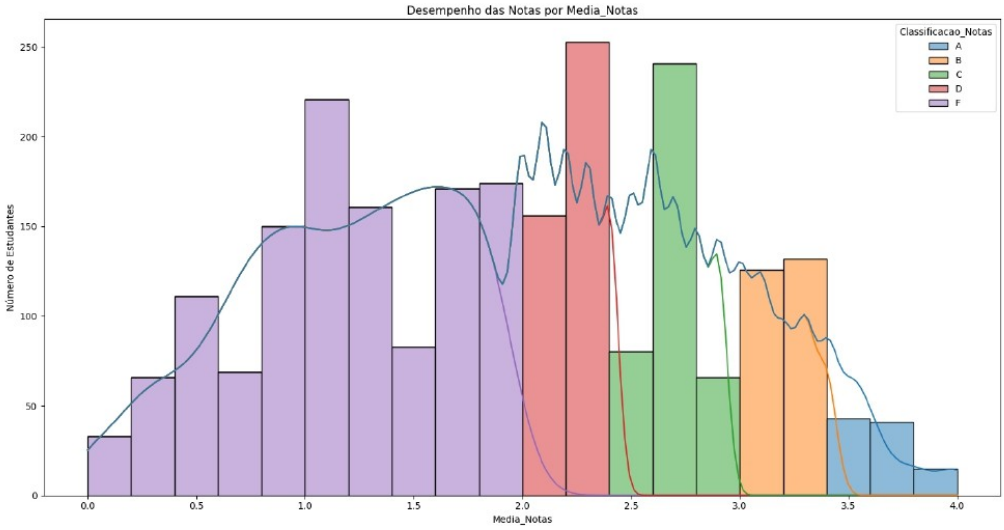

Desempenho das Notas por Apoio_Pais
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
Apoio_Pais
Moderado
Baixo
Alto
Muito Alto
Nenhum
0
200
400
600
800

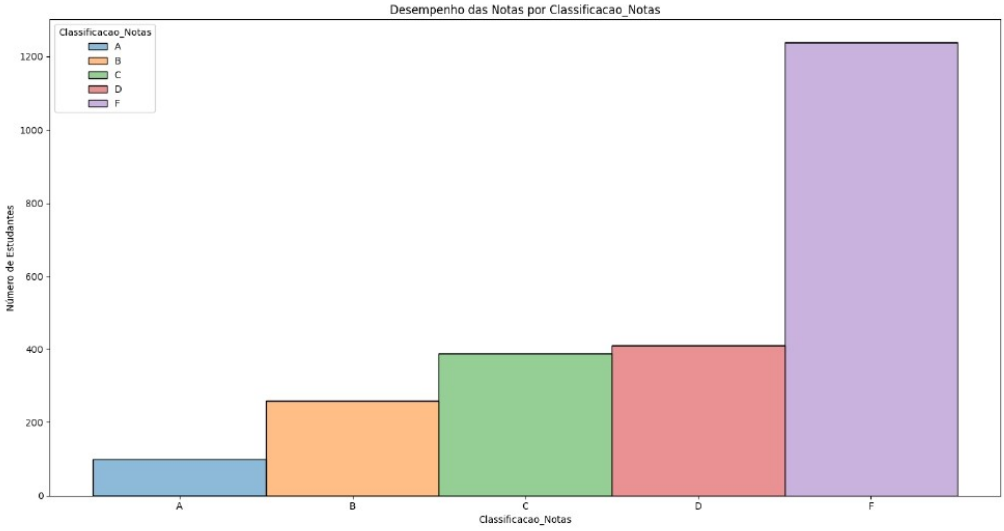

Desempenho das Notas por Atividades_Extracurriculares
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
1000
2000
3000
4000
Não
Sim
Atividades_Extracurriculares


Desempenho das Notas por Esportes
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
1000
2000
3000
4000
5000
Não
Sim
Esportes


Desempenho das Notas por Musica
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
1000
2000
3000
4000
5000
6000
7000
Sim
Não
Musica


Desempenho das Notas por Voluntariado
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
1000
2000
3000
4000
5000
6000
7000
8000
Não
Sim
Voluntariado


Desempenho das Notas por Media_Notas
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
50
100
150
200
250
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Media_Notas


Desempenho das Notas por Classificacao_Notas
Classificacao_Notas
A
B
C
D
F
Número de Estudantes
0
200
400
600
800
1000
1200
A
B
C
D
F
Classificacao_Notas

Insights Extraídos:

Os fatores demográficos não influenciaram a classificação e a média dos estudantes de modo direto, os dados só mostram um certo contraste na diversidade dos grupos etnicos, que provavalmente são influenciados pelos níveis culturais e sócio-econômicos.

Os fatores acadêmicos não tiveram uma forte influência na classificação e na média dos estudantes, uma vez que a grande maioria dos estudantes não participaram de atividades extracurriculares, provavelmente por falta de interesse dos estudantes nestas atividades, isso com certeza impactou muito o desempenho da classificação e da média, num futuro pode-se criar um incentivo à pratica de tais atividades.

Os fatores comportamentais tiveram uma influência muito siginificativa na classificacação e na média dos estudantes, sem dúvida as faltas, que segundo os dados, apesar dos alunos terem uma boa frequência nas aulas, as notas dos que faltam, por serem as mais baixas a nível de notas, impactaram muito o desempenho da classificação e da média de modo geral. Outro fator que também pode ter certa influência é o baixo número de participantes nas aulas particulares, que apesar das notas desse grupo não terem influenciado diretamente nas notas, mas impactaram a classificação e a média de modo geral. Já o que pode ter contribuído para melhores notas foram o apoio dos pais e as horas de estudos semanais desde que se mantenham entre 8 e 10 horas de estudos semanais.

---

6 - Análise Preditiva do Desempenho Acadêmico

In [30]:
```
# Criação de um novo dataframe para a análise preditiva
previsão_df_estudantes = novo_df_estudantes

# Sugestões de acordo com a visualização dos dados, para novas tomadas de decisões e assim melhorar
# o desempenho acadêmico:

# Alunos que estudam entre 8 e 10 horas alcançam notas maiores, abaixo de 8 e acima de 10 as notas tendem a cair
# -> Alunos devem estudar no mínimo 8 horas e no máximo 10 horas por semana
previsão_df_estudantes = previsão_df_estudantes[(previsão_df_estudantes["Horas_Estudo_Semanais"] >= 8)
                                    & (previsão_df_estudantes["Horas_Estudo_Semanais"] <=10)]

# Alunos com mais de 6 faltas tem desempenho sofrível na classificação
# -> Melhorar a frequência dos alunos, criando um limite máximo de faltas que atingida ou extrapolada,
# desclassifique o aluno
previsão_df_estudantes = previsão_df_estudantes[previsão_df_estudantes["Faltas"] <= 6]

# Em Aulas particulares o desempenho das notas foi bem pior para os estudantes que não participaram
# -> Incentivar os alunos a fazerem aulas particulares
previsão_df_estudantes = previsão_df_estudantes[previsão_df_estudantes["Aulas_Particulares"] == "Sim"]

# Apoio dos pais colabora para notas melhores
# -> Influenciar os pais a ajudarem os alunos nos estudos
previsão_df_estudantes = previsão_df_estudantes[(previsão_df_estudantes["Apoio_Pais"] != "Muito Alto")
                                    & (previsão_df_estudantes["Apoio_Pais"] != "Nenhum")]

# Nova contagem preditiva
display(previsão_df_estudantes["Classificacao_Notas"].value_counts())

# Nova proporção preditiva
display(previsão_df_estudantes["Classificacao_Notas"].value_counts(normalize=True).map("{:.1%}".format))

# Prévia de como o dataframe ficaria com as novas sugestões
display(previsão_df_estudantes.head(7))
```

```
Classificacao_Notas
B    14
A     7
C     4
D     1
F     0
Name: count, dtype: int64
Classificacao_Notas
B    53.8%
A    26.9%
C    15.4%
D     3.8%
F     0.0%
Name: proportion, dtype: object
```

	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Ext
92	16	Masculino	Caucasiano	Alguma Faculdade	9	1	Sim	Alto	
110	18	Masculino	Afrodescendente	Ensino Médio	10	4	Sim	Alto	
146	17	Feminino	Asiático	Alguma Faculdade	9	1	Sim	Moderado	
324	16	Feminino	Afrodescendente	Alguma Faculdade	10	5	Sim	Baixo	
505	15	Masculino	Caucasiano	Alguma Faculdade	10	4	Sim	Moderado	
542	17	Masculino	Asiático	Nenhum	8	6	Sim	Alto	
554	15	Masculino	Afrodescendente	Graduação	10	4	Sim	Baixo	

Por último fizemos uma análise preditiva após a visualização como visto nos resultados acima.

Nós começamos o problema com um taxa de 51.7% de classificação na nota F e somente 4.0% para a nota A, o que nos mostra um desempenho sofrível na classificação das notas dos alunos. Com ajuda dos gráficos e as sugestões de ajustes para melhorar o desempenho acadêmico, conseguimos ajustar nosso dataframe e prever um percentual de 26.9% para a nota A, com a nota F excluída da classificação e conseguindo obter 53.8% para a nota B e a nota D ficando em 3.8%.

Resposta da Pergunta Norteadora:

"Quais fatores demográficos, acadêmicos e comportamentais mais influenciam a classificação e a média de notas dos estudantes?"

Resposta:

De modo geral, a história contada através dos dados analisados de acordo com os insigths extraídos, deixa claro que as variáveis impactam diretamente a variável média das notas, esta por sua vez define o desempenho através da variável alvo classificação das notas. Os fatores comportamentais representados pelas variáveis faltas, apoio dos pais e horas de estudos semanais, são classificadas como as vilãs do mau desempenho acadêmico, sendo elas as que mais influenciaram na classificação das notas. Os fatores acadêmicos representados pelas variáveis música, esporte, voluntariado e atividades extracurriculares, apesar de não terem influenciado diretamente na classificação das notas, as mesmas devem ser dada uma atenção especial pelo baixíssimo número de participantes em todas essas atividades, possivelmente um maior número de alunos nessas atividades contribuiria para um aumento nas médias e até influenciar variáveis comportamentais como as faltas por exemplo, aumentando a frequência dos alunos. Os fatores demográficos representados pelas variáveis idade, gênero e etnia, apesar de irrelevantes para a classificação das notas, elas chamam atenção ao exibirem uma relação harmoniosa entre os alunos, evidenciando que o problema pode estar relacionado com a gestão do ensino público, a própria variável etnia alerta sobre a questão da inclusão social, que abrange até mesmo questões culturais e socioeconômicas.

Análise e Visualização de Dados concluídos!

📈 **Análise de Dados**

🚀 Projeto - Desempenho Acadêmico

Inteligência Artificial e Previsões:

Criar um modelo que consiga ler as informações dos estudantes e dizer automaticamente a classificação de suas notas: A, B, C, D, F

Analisar o dataframe e com base na análise criar o modelo de Inteligência Artificial e Previsões.

📚 Origem dos Dados

Este dataset foi disponibilizado por Rabie El Kharoua no Kaggle e está licenciado sob CC BY 4.0:

> El Kharoua, R. (2024). Students Performance Dataset.
> Disponível em: Kaggle Dataset Link.

Passos do Projeto

Passo 1 - Importar o DataFrame

Passo 2 - Visualizar o DataFrame

Passo 3 - Corrigir erros no DataFrame, modelar e entender os dados

Passo 4 - Preparar o dataframe para a Inteligência Artificial

Passo 5 - Treinar a Inteligência Artificial

Passo 6 - Escolher o melhor modelo

Passo 7 - Usar o melhor modelo

Passo 1 - Importar o DataFrame

In [36]:
```
# Importando as bibliotecas necessárias
import pandas as pd

# Importando o dataset
df_dados_estudantes = pd.read_csv("dados_estudantes.csv", sep= ',')

# Verificando as primeiras linhas do dataset
display(df_dados_estudantes)
```

	StudentID	Age	Gender	Ethnicity	ParentalEducation	StudyTimeWeekly	Absences	Tutoring	ParentalSupport	Extracurricular	S
0	1001	17	1	0	2	19.833723	7	1	2	0	
1	1002	18	0	0	1	15.408756	0	0	1	0	
2	1003	15	0	2	3	4.210570	26	0	2	0	
3	1004	17	1	0	3	10.028829	14	0	3	1	
4	1005	17	1	0	2	4.672495	17	1	3	0	
...	...	...	...	...	...	...	...	...	...	...	
2387	3388	18	1	0	3	10.680555	2	0	4	1	
2388	3389	17	0	0	1	7.583217	4	1	4	0	
2389	3390	16	1	0	2	6.805500	20	0	2	0	
2390	3391	16	1	1	0	12.416653	17	0	2	0	
2391	3392	16	1	0	2	17.819907	13	0	2	0	

2392 rows × 15 columns

In [37]:
```
traducao_colunas = {
    "StudentID": "ID_Aluno",
    "Age": "Idade",
```

```
    "Gender": "Genero",
    "Ethnicity": "Etnia",
    "ParentalEducation": "Escolaridade_Pais",
    "StudyTimeWeekly": "Horas_Estudo_Semanais",
    "Absences": "Faltas",
    "Tutoring": "Aulas_Particulares",
    "ParentalSupport": "Apoio_Pais",
    "Extracurricular": "Atividades_Extracurriculares",
    "Sports": "Esportes",
    "Music": "Musica",
    "Volunteering": "Voluntariado",
    "GPA": "Media_Notas",
    "GradeClass": "Classificacao_Notas"
}

# Renomear as colunas
df_dados_estudantes = df_dados_estudantes.rename(columns=traducao_colunas)

df_dados_estudantes.head()
```

Out[37]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extrac
0	1001	17	1	0	2	19.833723	7	1	2	
1	1002	18	0	0	1	15.408756	0	0	1	
2	1003	15	0	2	3	4.210570	26	0	2	
3	1004	17	1	0	3	10.028829	14	0	3	
4	1005	17	1	0	2	4.672495	17	1	3	

Passo 2 - Visualizar o DataFrame

In [38]:
```
# Usando shape para conferir as dimensões do dataframe
df_dados_estudantes.shape
```

Out[38]:
```
(2392, 15)
```

In [39]:
```
# Conferindo os tipos de dados utilizados no dataframe com dtypes
df_dados_estudantes.dtypes
```

Out[39]:
```
ID_Aluno                      int64
Idade                         int64
Genero                        int64
Etnia                         int64
Escolaridade_Pais             int64
Horas_Estudo_Semanais       float64
Faltas                        int64
Aulas_Particulares            int64
Apoio_Pais                    int64
Atividades_Extracurriculares  int64
Esportes                      int64
Musica                        int64
Voluntariado                  int64
Media_Notas                 float64
Classificacao_Notas         float64
dtype: object
```

In [40]:
```
# Visualizando as informaçoes do dataframe com info
df_dados_estudantes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                        Non-Null Count  Dtype  
---  ------                        --------------  -----  
 0   ID_Aluno                      2392 non-null   int64  
 1   Idade                         2392 non-null   int64  
 2   Genero                        2392 non-null   int64  
 3   Etnia                         2392 non-null   int64  
 4   Escolaridade_Pais             2392 non-null   int64  
 5   Horas_Estudo_Semanais         2392 non-null   float64
 6   Faltas                        2392 non-null   int64  
 7   Aulas_Particulares            2392 non-null   int64  
 8   Apoio_Pais                    2392 non-null   int64  
 9   Atividades_Extracurriculares  2392 non-null   int64  
 10  Esportes                      2392 non-null   int64  
 11  Musica                        2392 non-null   int64  
 12  Voluntariado                  2392 non-null   int64  
 13  Media_Notas                   2392 non-null   float64
 14  Classificacao_Notas           2392 non-null   float64
dtypes: float64(3), int64(12)
memory usage: 280.4 KB
```

In [41]:
```
# Conferir se existe valores ausentes no dataframe
df_dados_estudantes.isna().sum()
```

Out[41]:
```
ID_Aluno                      0
Idade                         0
Genero                        0
Etnia                         0
Escolaridade_Pais             0
Horas_Estudo_Semanais         0
Faltas                        0
Aulas_Particulares            0
Apoio_Pais                    0
Atividades_Extracurriculares  0
Esportes                      0
Musica                        0
Voluntariado                  0
Media_Notas                   0
Classificacao_Notas           0
dtype: int64
```

In [42]:
```
# Por fim fazer uma análise descritiva do dataframe
df_dados_estudantes.describe()
```

Out[42]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares
count	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000	2392.000000
mean	2196.500000	16.468645	0.510870	0.877508	1.746237	9.771992	14.541388	0.301421
std	690.655244	1.123798	0.499986	1.028476	1.000411	5.652774	8.467417	0.458971
min	1001.000000	15.000000	0.000000	0.000000	0.000000	0.001057	0.000000	0.000000
25%	1598.750000	15.000000	0.000000	0.000000	1.000000	5.043079	7.000000	0.000000
50%	2196.500000	16.000000	1.000000	0.000000	2.000000	9.705363	15.000000	0.000000
75%	2794.250000	17.000000	1.000000	2.000000	2.000000	14.408410	22.000000	1.000000
max	3392.000000	18.000000	1.000000	3.000000	4.000000	19.978094	29.000000	1.000000

Passo 3 - Corrigir erros no DataFrame, modelar e entender os dados

---

Para seguir os padrões apresentados na Descrição das Colunas do DataFrame, irei alterar o tipo de dados de algumas colunas, seguindo o padrão descrito abaixo:

📊 Descrição das Colunas do DataFrame

Identificação do Estudante

- **ID_Aluno**: Identificador único para cada aluno (de 1001 a 3392).

👥 Detalhes Demográficos

- **Idade**: Idade dos estudantes, variando de 15 a 18 anos.
- **Genero**: Gênero do estudante:

    - 0: Masculino
    - 1: Feminino
- **Etnia**: Etnia dos estudantes:
    - 0: Caucasiano
    - 1: Afrodescendente
    - 2: Asiático
    - 3: Outra

👪 Educação dos Pais

- **Escolaridade_Pais**: Nível de escolaridade dos pais:
    - 0: Nenhum
    - 1: Ensino Médio
    - 2: Alguma Faculdade
    - 3: Graduação
    - 4: Pós-Graduação ou Superior

📚 Hábitos de Estudo

- **Horas_Estudo_Semanais**: Tempo de estudo semanal em horas (0 a 20).
- **Faltas**: Número de faltas no ano letivo (0 a 30).
- **Aulas_Particulares**: Participação em aulas particulares:
    - 0: Não
    - 1: Sim

👪 Envolvimento Parental

- **Apoio_Pais**: Nível de apoio dos pais:
    - 0: Nenhum
    - 1: Baixo
    - 2: Moderado
    - 3: Alto
    - 4: Muito Alto

🏠 Atividades Extracurriculares

- **Atividades_Extracurriculares**: Participação em atividades extracurriculares:
    - 0: Não
    - 1: Sim
- **Esportes**: Participação em esportes:
    - 0: Não
    - 1: Sim
- **Musica**: Participação em atividades musicais:
    - 0: Não
    - 1: Sim
- **Voluntariado**: Participação em atividades de voluntariado:
    - 0: Não
    - 1: Sim

📈 Desempenho Acadêmico

- **Media_Notas**: Média das notas (GPA), variando de 2.0 a 4.0.
- **Classificacao_Notas**: Classificação do aluno com base na média das notas:
    - 0: 'A' (Média >= 3.5)
    - 1: 'B' (3.0 <= Média < 3.5)
    - 2: 'C' (2.5 <= Média < 3.0)
    - 3: 'D' (2.0 <= Média < 2.5)
    - 4: 'F' (Média < 2.0)

In [43]:
```
# Alterando a coluna Media_Notas para float com 1 casa decimal
df_dados_estudantes[["Media_Notas"]] = df_dados_estudantes[["Media_Notas"]].astype(float).round(1)
```

In [44]:
```
# Alterando as colunas Horas_Estudo_Semanais e Classificacao_Notas para Int
df_dados_estudantes[["Horas_Estudo_Semanais",
                     "Classificacao_Notas"]] = df_dados_estudantes[["Horas_Estudo_Semanais",
                                                                    "Classificacao_Notas"]].astype(int)
```

```
# Visualizando a alteração das 3 colunas
df_dados_estudantes.dtypes
```

Out[44]:
```
ID_Aluno                        int64
Idade                           int64
Genero                          int64
Etnia                           int64
Escolaridade_Pais               int64
Horas_Estudo_Semanais           int64
Faltas                          int64
Aulas_Particulares              int64
Apoio_Pais                      int64
Atividades_Extracurriculares    int64
Esportes                        int64
Musica                          int64
Voluntariado                    int64
Media_Notas                   float64
Classificacao_Notas             int64
dtype: object
```

In [45]:
```
# Visualizando o dataframe após as alterações
df_dados_estudantes.head()
```

Out[45]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extra
0	1001	17	1	0	2	19	7	1	2	
1	1002	18	0	0	1	15	0	0	1	
2	1003	15	0	2	3	4	26	0	2	
3	1004	17	1	0	3	10	14	0	3	
4	1005	17	1	0	2	4	17	1	3	

In [46]:
```
# O ID_Aluno está sendo contando de 1001 à 3392, para alterar a contagem para 1 à 2392 usarei o método apply lambda para
# fazer essa alteração.

# Alterando os valores do ID_Aluno para contar do número 1 em diante
df_dados_estudantes["ID_Aluno"] = df_dados_estudantes["ID_Aluno"].apply(lambda x: x - 1000)
```

In [47]:
```
# Para alterar os valores numéricos pelos seus respectivos valores informados na descrição, usarei o método replace
# para reverter os números cadastrados em seus devidos valores informados.

# Alterando os valores da descrição da coluna Gênero
df_dados_estudantes["Genero"] = df_dados_estudantes["Genero"].replace({
    0: "Masculino",
    1: "Feminino"
})
```

In [48]:
```
# Alterando os valores da descrição da coluna Etnia
df_dados_estudantes["Etnia"] = df_dados_estudantes["Etnia"].replace({
    0: "Caucasiano",
    1: "Afrodescendente",
    2: "Asiático",
    3: "Outra"
})
```

In [49]:
```
# Alterando os valores da descrição da coluna Escolaridade dos Pais
df_dados_estudantes["Escolaridade_Pais"] = df_dados_estudantes["Escolaridade_Pais"].replace({
    0: "Nenhum",
    1: "Ensino Médio",
    2: "Alguma Faculdade",
    3: "Graduação",
    4: "Pós-Graduação ou Superior"
})
```

In [50]:
```
# Alterando os valores da descrição da coluna Aulas Particulares
df_dados_estudantes["Aulas_Particulares"] = df_dados_estudantes["Aulas_Particulares"].replace({
    0: "Não",
    1: "Sim"
})
```

In [51]:
```
# Alterando os valores da descrição da coluna Apoio dos Pais
df_dados_estudantes["Apoio_Pais"] = df_dados_estudantes["Apoio_Pais"].replace({
    0: "Nenhum",
    1: "Baixo",
    2: "Moderado",
    3: "Alto",
    4: "Muito Alto"
})
```

In [52]:
```
# Alterando os valores da descrição da coluna Atividades Extracurriculares
df_dados_estudantes["Atividades_Extracurriculares"] = df_dados_estudantes["Atividades_Extracurriculares"].replace({
    0: "Não",
    1: "Sim"
})
```

In [53]:
```
# Alterando os valores da descrição da coluna Esportes
df_dados_estudantes["Esportes"] = df_dados_estudantes["Esportes"].replace({
     0: "Não",
     1: "Sim"
})
```

In [54]:
```
# Alterando os valores da descrição da coluna Musica
df_dados_estudantes["Musica"] = df_dados_estudantes["Musica"].replace({
    0: "Não",
    1: "Sim"
})
```

In [55]:
```
# Alterando os valores da descrição da coluna Voluntariado
df_dados_estudantes["Voluntariado"] = df_dados_estudantes["Voluntariado"].replace({
    0: "Não",
    1: "Sim"
})
```

In [56]:
```
# Alterando os valores da descrição da coluna Classificacao_Notas
df_dados_estudantes["Classificacao_Notas"] = df_dados_estudantes["Classificacao_Notas"].replace({
    0: "A",
    1: "B",
    2: "C",
    3: "D",
    4: "F"
})
```

In [57]:
```
# Com a alteração de todos os valores necessários para suas devidas descrições, é hora de visualizar o resultado em
# um novo dataframe.

#Visualização do novo dataframe após a alteração das descrições das colunas
novo_df_estudantes = df_dados_estudantes

display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	At
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	
...	...	...	...	...	...	...	...	...	...	
2387	2388	18	Feminino	Caucasiano	Graduação	10	2	Não	Muito Alto	
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4	Sim	Muito Alto	
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20	Não	Moderado	
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17	Não	Moderado	
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13	Não	Moderado	

2392 rows × 15 columns

Ao analisar o novo DataFrame, observei que existem inconsistências nos valores em relação a coluna Media_Notas e Classificacao_Notas. Para corrigir esse problema e não haver divergências entre as análises, irei corrigir os valores usando o método apply com lambda de acordo com as informações descritas na Descrição do Desempenho Acadêmico.

In [58]:
```
# Aplicando o lambda para corrigir os valores da Classificacao_Notas em relação a Média_Notas
novo_df_estudantes["Classificacao_Notas_Lambda"] = novo_df_estudantes["Media_Notas"].apply(lambda x:
                                                                                           "A" if x >= 3.5 else
                                                                                           "B" if x >= 3.0 else
                                                                                           "C" if x >= 2.5 else
                                                                                           "D" if x >= 2.0 else
                                                                                           "F")

display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	At
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	
...	...	...	...	...	...	...	...	...	...	
2387	2388	18	Feminino	Caucasiano	Graduação	10	2	Não	Muito Alto	
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4	Sim	Muito Alto	
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20	Não	Moderado	
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17	Não	Moderado	
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13	Não	Moderado	

2392 rows × 16 columns

Como eu havia notado, os valores estavam divergentes. Isso pode ser observado no dataframe comparando-se os valores entre Classificacao_Notas e Classificacao_Notas_Lambda que eu criei.

In [59]:
```
# Exluindo a coluna Classificacao_Notas (com os valores errados)
novo_df_estudantes = novo_df_estudantes.drop(columns=["Classificacao_Notas"])

novo_df_estudantes.head()
```

Out[59]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividade
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	

In [60]:
```
# Renomeando a coluna Classificacao_Notas_Lambda para o nome original Classificacao_Notas
novo_df_estudantes = novo_df_estudantes.rename(columns={"Classificacao_Notas_Lambda": "Classificacao_Notas"})

novo_df_estudantes.head()
```

Out[60]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividade
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado	
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo	
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado	
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto	
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto	

Agora que todas as colunas e seus valores foram remodelados e corrigidos, temos uma visão melhorada e precisa das informações, irei preparar o dataframe para a inteligência artifical.

In [61]:
```
novo_df_estudantes
```

Out[61]:

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais
0	1	17	Feminino	Caucasiano	Alguma Faculdade	19	7	Sim	Moderado
1	2	18	Masculino	Caucasiano	Ensino Médio	15	0	Não	Baixo
2	3	15	Masculino	Asiático	Graduação	4	26	Não	Moderado
3	4	17	Feminino	Caucasiano	Graduação	10	14	Não	Alto
4	5	17	Feminino	Caucasiano	Alguma Faculdade	4	17	Sim	Alto
...	...	...	...	...	...	...	...	...	...
2387	2388	18	Feminino	Caucasiano	Graduação	10	2	Não	Muito Alto
2388	2389	17	Masculino	Caucasiano	Ensino Médio	7	4	Sim	Muito Alto
2389	2390	16	Feminino	Caucasiano	Alguma Faculdade	6	20	Não	Moderado
2390	2391	16	Feminino	Afrodescendente	Nenhum	12	17	Não	Moderado
2391	2392	16	Feminino	Caucasiano	Alguma Faculdade	17	13	Não	Moderado

2392 rows × 15 columns

Passo 4 - Preparar o dataframe para a Inteligência Artifical

In [62]:
```
# Visualizar as informações do dataframe
novo_df_estudantes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   ID_Aluno                     2392 non-null   int64
 1   Idade                        2392 non-null   int64
 2   Genero                       2392 non-null   object
 3   Etnia                        2392 non-null   object
 4   Escolaridade_Pais            2392 non-null   object
 5   Horas_Estudo_Semanais        2392 non-null   int64
 6   Faltas                       2392 non-null   int64
 7   Aulas_Particulares           2392 non-null   object
 8   Apoio_Pais                   2392 non-null   object
 9   Atividades_Extracurriculares 2392 non-null   object
 10  Esportes                     2392 non-null   object
 11  Musica                       2392 non-null   object
 12  Voluntariado                 2392 non-null   object
 13  Media_Notas                  2392 non-null   float64
 14  Classificacao_Notas          2392 non-null   object
dtypes: float64(1), int64(4), object(10)
memory usage: 280.4+ KB
```

In [ ]:
```
# Codificar as colunas para tipo numérico para a Inteligência Artificial

from sklearn.preprocessing import LabelEncoder

# Codificar coluna Genero
codificador_genero = LabelEncoder()
novo_df_estudantes["Genero"] = codificador_genero.fit_transform(novo_df_estudantes["Genero"])

# Codificar coluna Etnia
codificado_etnia = LabelEncoder()
novo_df_estudantes["Etnia"] = codificado_etnia.fit_transform(novo_df_estudantes["Etnia"])

# Codificar coluna Escolaridade_Pais
codificador_escola_pais = LabelEncoder()
novo_df_estudantes["Escolaridade_Pais"] = codificador_escola_pais.fit_transform(novo_df_estudantes["Escolaridade_Pais"])

# Codificar coluna Aulas Particulares
codificador_particulares = LabelEncoder()
novo_df_estudantes["Aulas_Particulares"] = codificador_particulares.fit_transform(novo_df_estudantes["Aulas_Particulares"]

# Codificar coluna Apoio_Pais
codificador_apoio = LabelEncoder()
novo_df_estudantes["Apoio_Pais"] = codificador_apoio.fit_transform(novo_df_estudantes["Apoio_Pais"])

# Codificar coluna Atiividades Extracurriculares
codificador_atividade = LabelEncoder()
novo_df_estudantes["Atividades_Extracurriculares"] = codificador_atividade.fit_transform(novo_df_estudantes["Atividades_Ex

# Codificar coluna Esportes
```

```
codificador_esporte = LabelEncoder()
novo_df_estudantes["Esportes"] = codificador_esporte.fit_transform(novo_df_estudantes["Esportes"])

# Codificar coluna Musica
codificador_musica = LabelEncoder()
novo_df_estudantes["Musica"] = codificador_musica.fit_transform(novo_df_estudantes["Musica"])

# Codificar coluna Voluntariado
codificador_voluntariado = LabelEncoder()
novo_df_estudantes["Voluntariado"] = codificador_voluntariado.fit_transform(novo_df_estudantes["Voluntariado"])

novo_df_estudantes.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2392 entries, 0 to 2391
Data columns (total 15 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   ID_Aluno                    2392 non-null   int64
 1   Idade                       2392 non-null   int64
 2   Genero                      2392 non-null   int64
 3   Etnia                       2392 non-null   int64
 4   Escolaridade_Pais           2392 non-null   int64
 5   Horas_Estudo_Semanais       2392 non-null   int64
 6   Faltas                      2392 non-null   int64
 7   Aulas_Particulares          2392 non-null   int64
 8   Apoio_Pais                  2392 non-null   int64
 9   Atividades_Extracurriculares  2392 non-null   int64
 10  Esportes                    2392 non-null   int64
 11  Musica                      2392 non-null   int64
 12  Voluntariado                2392 non-null   int64
 13  Media_Notas                 2392 non-null   float64
 14  Classificacao_Notas         2392 non-null   object
dtypes: float64(1), int64(13), object(1)
memory usage: 280.4+ KB
```

In [64]:
```
# Visualização do dataframe após a codificação
display(novo_df_estudantes)
```

	ID_Aluno	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extr
0	1	17	0	2	0	19	7	1	2	
1	2	18	1	2	1	15	0	0	1	
2	3	15	1	1	2	4	26	0	2	
3	4	17	0	2	2	10	14	0	0	
4	5	17	0	2	0	4	17	1	0	
...	...	...	...	...	...	...	...	...	...	
2387	2388	18	0	2	2	10	2	0	3	
2388	2389	17	1	2	1	7	4	1	3	
2389	2390	16	0	2	0	6	20	0	2	
2390	2391	16	0	0	3	12	17	0	2	
2391	2392	16	0	2	0	17	13	0	2	

2392 rows × 15 columns

In [65]:
```
# Preparar o dataframe para o treino

# y é a coluna do dataframe a ser prevista
y = novo_df_estudantes["Classificacao_Notas"]

# x são as colunas do dataframe que serão usadas para a previsão
x = novo_df_estudantes.drop(columns=["Classificacao_Notas", "ID_Aluno"])

# Separar os dados de treino e os dados de teste
from sklearn.model_selection import train_test_split

x_treino, x_teste, y_treino, y_teste = train_test_split(x, y, test_size=0.7)
```

Passo 5 - Treinar a Inteligência Artificial

In [66]:
```
# Criar o medelo -> Classificação da Nota: A, B, C, D, F

# Arvore de Decisão -> RandonForest
```

```
# Nearest Neighbors -> KNN -> Vizinhos Próximos

# Importar a Inteligência Artificial (IA)
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier

# Cria a IA
modelo_arvore_decisao = RandomForestClassifier()
modelo_knn = KNeighborsClassifier()

# Treinar a IA
modelo_arvore_decisao.fit(x_treino, y_treino)
modelo_knn.fit(x_treino, y_treino)
```

Out[66]:

```
KNeighborsClassifier()
```

Passo 6 - Escolher o melhor modelo

In [67]:

```
# Testar os modelos
previsao_arvore_decisao = modelo_arvore_decisao.predict(x_teste)
previsao_knn = modelo_knn.predict(x_teste)

# Verificar Acurácia
from sklearn.metrics import accuracy_score

display(accuracy_score(y_teste, previsao_arvore_decisao) *100)
display(accuracy_score(y_teste, previsao_knn) * 100)
```

```
98.32835820895522
71.34328358208955
```

Passo 7 - Usar melhor modelo

In [68]:

```
# Melhor modelo é Modelo Arvore de Decisão

#Importar os novos estudantes para fazer a previsão
df_novos_dados = pd.read_csv("novos_dados_estudantes.csv")

display(df_novos_dados)

nova_previsao = modelo_arvore_decisao.predict(df_novos_dados)

display(nova_previsao)
```

	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extracurriculares
0	18	0	0	1	5.344198	26	1	1	0
1	15	0	2	1	10.182268	21	1	1	0
2	15	1	1	1	2.949078	3	1	1	1
3	17	1	2	1	12.233114	21	1	3	1
4	17	1	3	2	11.314946	23	1	1	0
...	...	...	...	...	...	...	...	...	...
253	18	1	0	0	7.945826	15	1	0	0
254	17	1	0	2	6.736954	27	0	4	0
255	15	1	0	2	11.389648	3	1	3	1
256	17	1	2	2	17.222270	11	0	3	0
257	16	0	1	1	1.566764	10	0	3	0

258 rows × 13 columns

```
array(['F', 'F', 'B', 'F', 'F', 'C', 'C', 'C', 'A', 'C', 'C', 'F', 'D',
       'C', 'C', 'D', 'F', 'F', 'F', 'F', 'F', 'F', 'C', 'F', 'F', 'D',
       'D', 'D', 'B', 'B', 'F', 'F', 'F', 'B', 'C', 'F', 'C', 'F', 'F',
       'C', 'F', 'F', 'F', 'D', 'F', 'F', 'F', 'C', 'B', 'A', 'B', 'F',
       'D', 'F', 'F', 'F', 'F', 'A', 'D', 'B', 'F', 'B', 'A', 'C', 'A',
       'C', 'D', 'F', 'D', 'B', 'A', 'D', 'B', 'D', 'D', 'F', 'C', 'D',
       'B', 'F', 'F', 'D', 'D', 'C', 'A', 'D', 'F', 'C', 'C', 'F', 'F',
       'F', 'B', 'D', 'F', 'C', 'F', 'F', 'B', 'F', 'A', 'C', 'D', 'F',
       'C', 'D', 'D', 'F', 'F', 'F', 'D', 'C', 'F', 'F', 'C', 'F', 'C',
       'B', 'B', 'F', 'D', 'B', 'B', 'F', 'B', 'C', 'F', 'F', 'F', 'D',
       'D', 'D', 'C', 'F', 'B', 'F', 'F', 'F', 'F', 'F', 'B', 'D', 'F',
       'D', 'F', 'F', 'F', 'F', 'C', 'F', 'B', 'F', 'D', 'F', 'F', 'C',
       'F', 'F', 'F', 'D', 'D', 'C', 'F', 'F', 'F', 'F', 'F', 'B', 'F',
       'F', 'F', 'D', 'F', 'C', 'B', 'F', 'F', 'F', 'B', 'F', 'C', 'D',
       'F', 'D', 'F', 'F', 'F', 'C', 'F', 'D', 'B', 'B', 'F', 'D', 'B',
       'C', 'F', 'A', 'D', 'F', 'F', 'F', 'F', 'D', 'F', 'F', 'F', 'F',
       'F', 'F', 'D', 'F', 'D', 'C', 'F', 'F', 'D', 'F', 'F', 'B', 'F',
       'D', 'B', 'F', 'F', 'F', 'C', 'D', 'F', 'F', 'B', 'C', 'B', 'F',
       'B', 'D', 'C', 'F', 'F', 'F', 'F', 'D', 'B', 'D', 'D', 'B', 'F',
       'F', 'F', 'B', 'F', 'F', 'C', 'F', 'F', 'B', 'C', 'D'],
      dtype=object)
```

In [69]:
```
# Análise exploratória inicial da classificação das notas

# Contagem das notas da classificação

df_novos_dados["Classificacao_Notas"] = nova_previsao

display(df_novos_dados["Classificacao_Notas"].value_counts())

# Proporção das classificações
display(df_novos_dados["Classificacao_Notas"].value_counts(normalize=True).map("{:.1%}".format))

display(df_novos_dados)
```

```
Classificacao_Notas
F    126
D     48
C     39
B     36
A      9
Name: count, dtype: int64
Classificacao_Notas
F    48.8%
D    18.6%
C    15.1%
B    14.0%
A     3.5%
Name: proportion, dtype: object
```

	Idade	Genero	Etnia	Escolaridade_Pais	Horas_Estudo_Semanais	Faltas	Aulas_Particulares	Apoio_Pais	Atividades_Extracurriculares
0	18	0	0	1	5.344198	26	1	1	0
1	15	0	2	1	10.182268	21	1	1	0
2	15	1	1	1	2.949078	3	1	1	1
3	17	1	2	1	12.233114	21	1	3	1
4	17	1	3	2	11.314946	23	1	1	0
...	...	...	...	...	...	...	...	...	...
253	18	1	0	0	7.945826	15	1	0	0
254	17	1	0	2	6.736954	27	0	4	0
255	15	1	0	2	11.389648	3	1	3	1
256	17	1	2	2	17.222270	11	0	3	0
257	16	0	1	1	1.566764	10	0	3	0

258 rows × 14 columns

Quais as caracteristicas mais importantes para definir a Classificação das Notas?

In [70]:
```
# Características mais importantes para definir a Classificação das Notas

colunas = list(x_teste.columns)
importancia = pd.DataFrame(index=colunas, data=modelo_arvore_decisao.feature_importances_)
importancia = importancia * 100
print(importancia)
```

	0
Idade	2.239083
Genero	1.108586
Etnia	1.949609
Escolaridade_Pais	2.229692
Horas_Estudo_Semanais	4.809006
Faltas	23.406254
Aulas_Particulares	1.337815
Apoio_Pais	2.784290
Atividades_Extracurriculares	1.119025
Esportes	1.012912
Musica	0.965540
Voluntariado	0.768735
Media_Notas	56.269454

As características mais importantes são as médias das notas com 56,2% e faltas com 23,4% e não menos importante as horas de estudos semanais com 4.8%.

📈 **Análise de Dados e Machine Learning**
