Análise de Dados – Exploratória e ML

Sumário

[Introdução 4](#_Toc181822029)

[Objetivo do aplicativo 4](#_Toc181822030)

[Objetivo da Análise Exploratória de Dados 4](#_Toc181822031)

[Objetivo da Análise de Dados (para Aprendizado de Máquina) 4](#_Toc181822032)

[Levantamento dos dados na análise exploratória 4](#_Toc181822033)

[Busca dos dados 4](#_Toc181822034)

[Justificativa de uso 5](#_Toc181822035)

[Descrição da base de dados de trabalho 5](#_Toc181822036)

[Limpeza dos dados e Preparação dos dados 6](#_Toc181822037)

[Condicionamento para alimentar o modelo de ML 8](#_Toc181822038)

[Condicionamento inicia 8](#_Toc181822039)

[Definição dos objetivos e das classes 8](#_Toc181822040)

[Definição dos modelos mais adequados para analisar os dados 10](#_Toc181822041)

[Descrição dos modelos selecionados 11](#_Toc181822042)

[Modelo 1: Naive Bayes – GaussianNB 11](#_Toc181822043)

[Critérios de Seleção: 11](#_Toc181822044)

[Modelo 2: Árvore de Decisão – Gini 11](#_Toc181822045)

[Critérios de Seleção: 11](#_Toc181822046)

[Modelo 3: Árvore de Decisão – Entropia 12](#_Toc181822047)

[Critérios de Seleção: 12](#_Toc181822048)

[Modelo 4: K-Nearest Neighbors (KNN) 12](#_Toc181822049)

[Critérios de Seleção: 12](#_Toc181822050)

[Aplicação dos modelos selecionados 12](#_Toc181822051)

[Modelo 1: Naive Bayes – GaussianNB 13](#_Toc181822052)

[Modelo 2: Árvore de Decisão – Gini e Entropia 13](#_Toc181822053)

[Modelo 3: K-Nearest Neighbors (KNN) 13](#_Toc181822054)

[Análise dos Resultados 14](#_Toc181822055)

[Modelo 1: Naive Bayes – GaussianNB 14](#_Toc181822056)

[Modelo 2: Árvore de Decisão – Gini 14](#_Toc181822057)

[Modelo 3: Árvore de Decisão – Entropia 15](#_Toc181822058)

[Modelo 4: K-Nearest Neighbors (KNN) 16](#_Toc181822059)

[Comparação Entre Modelos: 17](#_Toc181822060)

[Análise Comparativa 17](#_Toc181822061)

[Forças e Fraquezas 18](#_Toc181822062)

[Concluindo os Modelos 19](#_Toc181822063)

[Ajustes Necessários 20](#_Toc181822064)

[Identificação de Problemas: 20](#_Toc181822065)

[Mudanças na Base de Dados: 20](#_Toc181822066)

[Ajustes nos Modelos: 21](#_Toc181822067)

[Impacto das Modificações 21](#_Toc181822068)

[Modelo 1: Naive Bayes – GaussianNB 21](#_Toc181822069)

[Modelo 2: Árvore de Decisão – Gini 22](#_Toc181822070)

[Modelo 3: Árvore de Decisão – Entropia 23](#_Toc181822071)

[Modelo 4: K-Nearest Neighbors (KNN) 24](#_Toc181822072)

[Comparação Entre Modelos: 25](#_Toc181822073)

[Análise Comparativa 25](#_Toc181822074)

[Forças e Fraquezas 26](#_Toc181822075)

[Conclusão 27](#_Toc181822076)

[Principais Descobertas e Eficácia dos Modelos 27](#_Toc181822077)

[Etapas recomendadas para aprimorar a análise 28](#_Toc181822078)

[Conclusão Final 28](#_Toc181822079)

[Modelo no APP 28](#_Toc181822080)

[RPAs 29](#_Toc181822081)

[Manuais 29](#_Toc181822082)

[AWS: 29](#_Toc181822083)

Análise de Dados para EducaEco

Gats

Integrantes:

Carlos Henrique de Godoy Santos 2°G N°1

Murilo de Oliveira Moreira 2°G N°15

# Introdução

O EducaEco é um aplicativo dedicado a incentivar e educar jovens e adolescentes sobre Sustentabilidade e ESG. Nosso objetivo é resolver a lacuna de conhecimento sobre esses temas entre crianças e adolescentes, proporcionando oportunidades para seu desenvolvimento pessoal e profissional.

## Objetivo do aplicativo

O EducaEco busca conscientizar os jovens sobre a importância de proteger e preservar o meio ambiente. Através desse aprendizado, o aplicativo oferece uma base para o desenvolvimento pessoal, podendo também abrir portas para futuras oportunidades profissionais.

O aplicativo conta com diversas funcionalidades projetadas para transmitir nossa visão, incluindo missões diárias, atividades práticas, quizzes e vídeos com profissionais especializados que discutem esses temas de forma acessível e envolvente.

Os integrantes do 1° ano desenvolveram um material com especificações detalhadas sobre o EducaEco. Informações mais completas e técnicas podem ser encontradas nele.

# Objetivo da Análise Exploratória de Dados

**Montar, analisar e limpar um conjunto de dados** relacionado ao aplicativo para ser utilizado em análise de dados exploratória e para fins de aprendizado de máquina (ML).

O objetivo da Análise Exploratório de Dados entra para conseguirmos montar, analisar e limpar nosso conjunto de dados, para termos uma análise sem anormalidades e conseguirmos tirar insights precisos sobre o que queremos analisar. Os dados que analisamos podem ser de bases de dados governamentais, científicas ou de sondagens próprias das equipes envolvidas, então é muito importante, deixarmos as informações ajustadas e sem anormalidades.

# Objetivo da Análise de Dados (para Aprendizado de Máquina)

Dentro de nossa análise de dados, buscamos prever a motivação do usuário para o uso de um aplicativo com foco de incentivar e educar jovens e adolescentes sobre Sustentabilidade e ESG.

Em nossa análise usamos quatro modelos de classificação:

* Naive Bayes – GaussianNB
* Árvore de Decisão – Gini
* Árvore de Decisão – Entropia
* KNN – K-Nearest Neighbors

# Levantamento dos dados na análise exploratória

## Busca dos dados

Para termos informações se as pessoas realmente iriam utilizar um aplicativo com foco de incentivar e educar jovens e adolescentes sobre Sustentabilidade e ESG.

Nós do 2° ano da equipe de Análise de Dados, criamos um formulário para conhecermos mais sobre o usuário em si, para analisarmos o nível de comprometimento da pessoa com Sustentabilidade e ESG, e perguntas mais sobre os diferenciais do nosso aplicativo para vermos se eles gostam ou não.

Link do formulário: <https://forms.office.com/Pages/DesignPageV2.aspx?origin=NeoPortalPage&subpage=design&id=TPFIsZcjLECrahtHERd6wEVGLLc70q9HqryorVE7pNNURFhaSTc4QTNRODNXU0NCWkVOVjg2UU9TMy4u>

No total obtivemos 252 respostas. Lembrando que esses forms não foi enviado só para as pessoas do nosso público-alvo, queríamos avaliar o todo, então temos variedades de faixa-etárias.

## Justificativa de uso

As perguntas foram cuidadosamente elaboradas para avaliar o comprometimento dos usuários com temas de Sustentabilidade e ESG e explorar o interesse nos diferenciais que o nosso aplicativo oferece. Com isso, podemos identificar potenciais usuários mais engajados e avaliar como suas percepções e práticas alinhadas à sustentabilidade refletem no interesse pelas funcionalidades abordadas. Essa análise nos permite prever o grau de adesão e engajamento esperado, direcionando melhorias que aumentem o valor e a relevância do aplicativo para nosso público-alvo.

# Descrição da base de dados de trabalho

Em nossa base de dados, temos 11 colunas, onde todas são do tipo string:

* 1. Faixa etária: Qual e a sua faixa etária ?
  + Menos de 18 anos
  + Entre 18 e 24 anos
  + Entre 25 e 35 anos
  + Entre 35 e 45 anos
  + Entre 46 a 59 anos
  + Acima de 60 anos
* 2. Renda familiar mensal: Qual é a sua renda familiar mensal, considerando as seguintes faixas.
  + Classe A: Acima de R$19.801,00
  + Classe B: Entre R$7.921,00 E R$19.800,00
  + Classe C: Entre R$2.641,00 e R$7.920,00
  + Classe D: Entre R$1.321,00 e R$2.640,00
  + Classe E: Até R$1.320,00
* 3. Práticas sustentáveis: Você adota alguma **prática sustentável** (ações que visam a utilização eficiente dos recursos naturais, sem os esgotar ou prejudicar os ecossistemas) no seu dia a dia?
  + Sim
  + Não
* 4. Frequência das práticas sustentáveis: Com que frequência você **pratica** uma ação sustentável ?
  + Alta
  + Média
  + Baixa
* 5. Aprendizado sobre sustentabilidade na escola: Você aprendeu ou teve aulas sobre **sustentabilidade**na escola?
  + Sim
  + Não
* 6. Uso de jogos no aprendizado escolas: Se você aprendeu **sustentabilidade**na escola, inclui-a jogos para a sua aprendizagem?
  + Sim
  + Não
* 7. Opinião sobre jogos para aprendizado: Você acredita que **jogos**podem ajudar no aprendizado sobre sustentabilidade?
  + Sim
  + Não
  + Talvez
* 8. Consumo de outros aplicativos/plataformas: Você faz ou já fez o consumo de outros **aplicativos / plataformas** de sustentabilidade e ESG (Governança ambiental, social e corporativa) ?
  + Sim
  + Não
* 9. Importância do conhecimento em Sustentabilidade/ESG: Você acha que o **conhecimento** sobre Sustentabilidade / ESG (Governança ambiental, social e corporativa) é crucial para o seu desenvolvimento profissional ?
  + Sim
  + Não
* 10. Funcionalidades úteis em um aplicativo: Quais das seguintes **funcionalidades**você consideraria mais útil em um aplicativo de aprendizado sobre Sustentabilidade e ESG (Governança ambiental, social e corporativa) com gamificação.
  + Checkbox: Jogos para seu aprendizado
  + Checkbox: Atividades práticas para seu desenvolvimento
  + Checkbox: Quizzes para testar seu conhecimento
  + Checkbox: Vídeos com profissionais falando sobre Sustentabilidade e ESG
* 11. Motivação da Pessoa: Qual seria sua **motivação**para usar um aplicativo de Sustentabilidade / ESG (Governança ambiental, social e corporativa)
  + Pessoal
  + Em um ambiente Profissional
  + Estudantil
  + Não tenho motivação

Fizemos nossa análise e tratamento da base de dados em python, usando bibliotecas para estes serviços como: Pandas, Numpy, Matplotlib e Seaborn.

# Limpeza dos dados e Preparação dos dados

Para que a base de dados seja adequada à nossa análise, tratamos os seguintes pontos:

* Nossa base de dados conterá valores nulos devido a perguntas condicionais no formulário, onde algumas pessoas podem não responder. Portanto, preencheremos esses valores nulos com 'Não' ou '0'.
* Uma das perguntas do nosso formulário é de múltipla escolha, o que significa que algumas linhas específicas da base podem ter mais de um valor. Para resolver isso, vamos dividir essa coluna em várias a partir da vírgula, de acordo com o número de respostas. A coluna foi quebrado em 4 colunas, pois tinham 4 funcionalidades que podiam ser escolhidas, com o nosso tratamento, fizemos uma condição para se a resposta tiver na coluna retorna True, se não retorna False.
* Em alguns casos, formulários como Google Forms ou similares adicionam colunas padrão que não são relevantes para a nossa análise. Essas colunas podem conter valores nulos ou dados que não agregam valor, podendo até comprometer a qualidade da análise. Portanto, é essencial identificar e excluir essas colunas para garantir que o conjunto de dados seja mais limpo e focado nas informações que realmente importam. Removendo essas colunas desnecessárias, podemos evitar ruídos e possíveis erros na análise, além de simplificar o processamento dos dados.
* Nosso objetivo é avaliar as pessoas classificadas como 'Estudantil'. Para isso, transformaremos as outras classes ('Pessoal', 'Em um ambiente Profissional', e 'Não tenho motivação') em uma única categoria chamada 'Não tenho motivação'

Após todos esses tratamentos, obtivemos uma base de dados totalmente tratada, sem valores nulos. Ao final do tratamento ficamos com 14 colunas, porque a coluna de funcionalidades foi quebrada em 4, onde 10 colunas são do tipo object e 4 do tipo boolean.

Após isso tratamos a base para a Machine Learning, então fizemos os seguintes passos:

* Separamos a base de dados em atributos e resposta.
* Criamos e aplicamos o pré-processador na base, transformando as classes dentro das colunas em numéricas.
* Separamos em Treino e Teste.

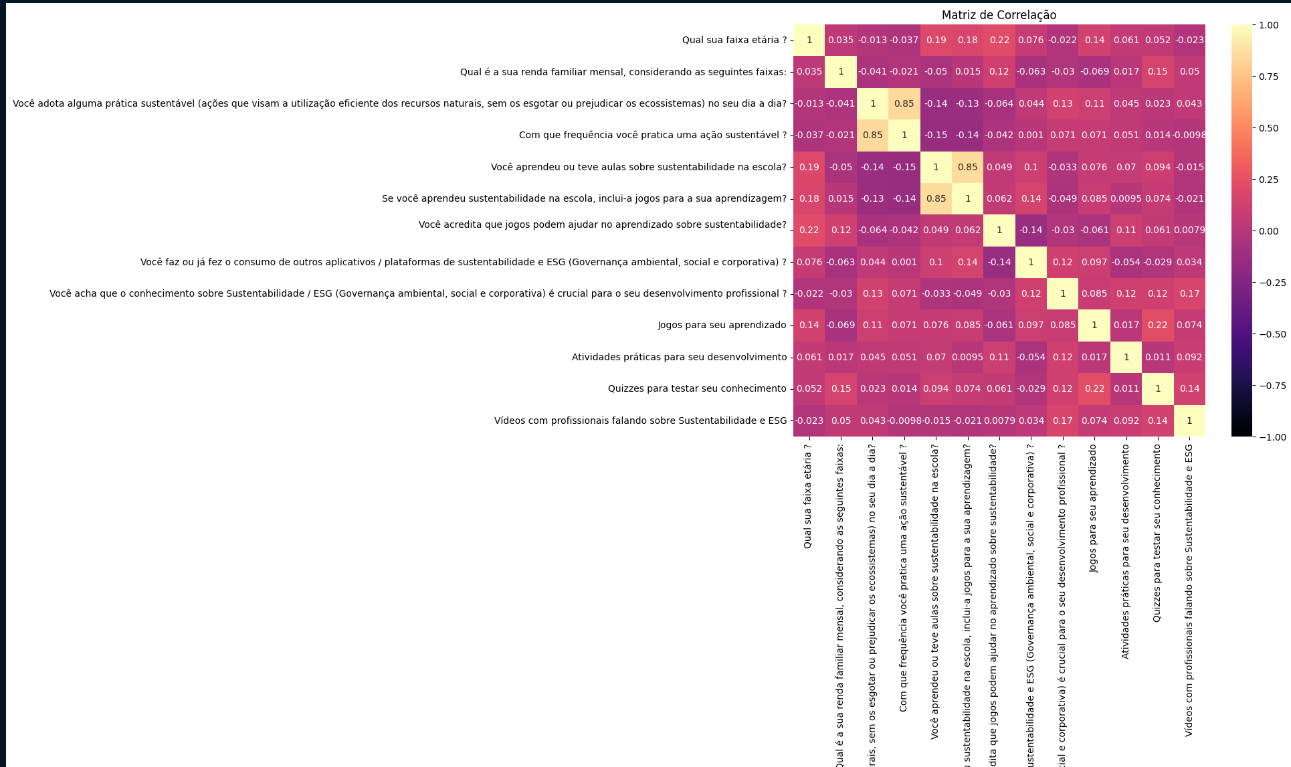
Após essas etapas, plotamos uma matriz de correlação para verificarmos se as perguntas tem relações ou não.

A matriz de correlação é uma ferramenta estatística que ajuda a entender como variáveis numéricas se relacionam entre si em um conjunto de dados. O valor em cada célula indica o grau de correlação entre o par de variáveis que se cruzam naquela posição. <br>

**Correlação positiva (+1):** Indica que as duas variáveis aumentam ou diminuem juntas. <br>

**Correlação negativa (-1):** Indica que uma variável aumenta enquanto a outra diminui. <br>

**Correlação próxima de zero (0):** Indica pouca ou nenhuma relação linear entre as variáveis. Elas podem variar independentemente.



# Condicionamento para alimentar o modelo de ML

Rodamos os modelos de Machine Learning no “puro”, ou seja, fomos apenas com a base tratada, claro que com as métricas obtidas, concluímos que precisamos de uma aplicação mais profunda, para aumentarmos as métricas dos nossos modelos.

## Condicionamento inicia

Após a primeira rodagem dos modelos de Machine Learning, como já disse concluímos que precisariam ser tratadas, então a partir disso fizemos:

* Implementamos o PCA, para reduzir a dimensionalidade dos dados. Mas infelizmente não obtivemos um resultado bom, com ele diminuindo nossas métricas, então não usamos ele.
* Duplicando os dados com o método SMOTE, ele cria dados a partir da base de dados igualmente, se baseando nos pontos (especificar a quantidade de pontos) mais próximos para criar um novo dado. Com ele obtivemos um resultado muito bom.

# Definição dos objetivos e das classes

Como já comentei, a princípio nossa IA dentro da variável resposta tinham 4 classes: Pessoal; Em um ambiente Profissional; Estudantil e Não tenho motivação. Mas só queremos avaliar as pessoas que usariam nosso aplicativo para uso Estudantil, pelo menos por enquanto em nosso MVP. Tratamos isso, e assumimos que as motivações: Pessoal e Em um ambiente Profissional transformamos na motivação Estudantil.

Com os modelos de Machine Learning queremos prever a motivação dos usuários para usar um aplicativo de ensino sobre Sustentabilidade e ESG. Escolhemos essa variável resposta, pois podemos analisar literalmente o porquê de o cliente usar nosso aplicativo.

A partir disso vamos ter uma melhor observação sobre realmente qual seria a motivação do usuário para usar nosso aplicativo.

Nossa base de dados ficou com 14 colunas, sendo 13 colunas atributos e 1 coluna resposta, ficando assim:

**Atributos:**

* 1. Faixa etária: Qual e a sua faixa etária ? Tipo da coluna: Object
  + Menos de 18 anos
  + Entre 18 e 24 anos
  + Entre 25 e 35 anos
  + Entre 35 e 45 anos
  + Entre 46 a 59 anos
  + Acima de 60 anos
* 2. Renda familiar mensal: Qual é a sua renda familiar mensal, considerando as seguintes faixas. Tipo da coluna: Object
  + Classe A: Acima de R$19.801,00
  + Classe B: Entre R$7.921,00 E R$19.800,00
  + Classe C: Entre R$2.641,00 e R$7.920,00
  + Classe D: Entre R$1.321,00 e R$2.640,00
  + Classe E: Até R$1.320,00
* 3. Práticas sustentáveis: Você adota alguma **prática sustentável** (ações que visam a utilização eficiente dos recursos naturais, sem os esgotar ou prejudicar os ecossistemas) no seu dia a dia? Tipo da coluna: Object
  + Sim
  + Não
* 4. Frequência das práticas sustentáveis: Com que frequência você **pratica** uma ação sustentável ? Tipo da coluna: Object
  + Alta
  + Média
  + Baixa
* 5. Aprendizado sobre sustentabilidade na escola: Você aprendeu ou teve aulas sobre **sustentabilidade**na escola? Tipo da coluna: Object
  + Sim
  + Não
* 6. Uso de jogos no aprendizado escolas: Se você aprendeu **sustentabilidade**na escola, inclui-a jogos para a sua aprendizagem? Tipo da coluna: Object
  + Sim
  + NÃo
* 7. Opinião sobre jogos para aprendizado: Você acredita que **jogos**podem ajudar no aprendizado sobre sustentabilidade? Tipo da coluna: Object
  + Sim
  + Não
  + Talvez
* 8. Consumo de outros aplicativos/plataformas: Você faz ou já fez o consumo de outros **aplicativos / plataformas** de sustentabilidade e ESG (Governança ambiental, social e corporativa) ? Tipo da coluna: Object
  + Sim
  + Não
* 9. Importância do conhecimento em Sustentabilidade/ESG: Você acha que o **conhecimento** sobre Sustentabilidade / ESG (Governança ambiental, social e corporativa) é crucial para o seu desenvolvimento profissional ? Tipo da coluna: Object
  + Sim
  + Não
* 10. Funcionalidade Jogos para seu Aprendizado: Jogos para seu aprendizado. Tipo da coluna: Boolean
  + True
  + False
* 11. Funcionalidade Atividades práticas para seu desenvolvimento: Atividades práticas para seu desenvolvimento. Tipo da coluna: Boolean
  + True
  + False
* 12. Funcionalidade Quizzes para testar seu conhecimento: Quizzes para testar seu conhecimento. Tipo da coluna: Boolean
  + True
  + False
* 13. Funcionalidade Vídeos com profissionais falando sobre sustentabilidade e ESG: Vídeos com profissionais falando sobre sustentabilidade e ESG. Tipo da coluna: Boolean
  + True
  + False

**Resposta:**

* 14. Motivação da Pessoa: Qual seria sua **motivação**para usar um aplicativo de Sustentabilidade / ESG (Governança ambiental, social e corporativa). Tipo da coluna: Object
  + Estudantil
  + Não tenho motivação

# Definição dos modelos mais adequados para analisar os dados

Para realizar as análises, levantamos quatro modelos de classificação: Naive Bayes (GaussianNB), Árvore de Decisão (com os critérios de Gini e Entropia), e K-Nearest Neighbors (KNN). Estes modelos foram considerados potenciais candidatos devido a sua diversidade em termos de complexidade e adequação ao tipo de dados e ao objetivo da tarefa.

**Natureza dos Dados:** O conjunto de dados analisado possui características tanto categóricas quanto contínuas, que precisam ser classificadas em categorias específicas. Naive Bayes, em particular, é eficiente com dados categóricos, enquanto Árvore de Decisão e KNN oferecem flexibilidade para ambas as naturezas de dados.

**Complexidade dos Modelos:**

* **Naive Bayes:** Um modelo simples e eficaz para tarefas de classificação rápida. Foi escolhido devido à sua capacidade de produzir bons resultados em dados que possuem independência entre as características, característica esperada no nosso conjunto de dados após a análise exploratória.
* **Árvore de Decisão:** Testamos com os critérios de Gini e Entropia. As Árvores de Decisão oferecem interpretabilidade e são eficientes em detectar padrões com critérios de decisão específicos, como o Gini para maximizar a pureza dos nós e a Entropia para maximizar a informação.
* **KNN**: Um modelo não paramétrico que pode se adaptar bem em contextos em que a relação entre as classes é mais complexa. Foi incluído por sua simplicidade de implementação e sua natureza flexível ao ajustar o parâmetro de número de vizinhos mais próximos.

**Objetivo de Classificação:** O objetivo da tarefa de classificação é **binário**, ou seja, categorizamos os dados em duas classes. Isso nos permitiu ajustar cada um dos modelos para otimizar o desempenho especificamente para este tipo de classificação, simplificando o problema e direcionando os recursos computacionais para maximizar a separação entre essas duas classes.

# Descrição dos modelos selecionados

## Modelo 1: Naive Bayes – GaussianNB

O Naive Bayes é um modelo probabilístico baseado no Teorema de Bayes, que assume a independência entre as características. A versão GaussianNB é particularmente adequada para dados contínuos que seguem uma distribuição normal.

### Critérios de Seleção:

* **Precisão e Robustez**: O Naive Bayes é conhecido por sua precisão em problemas onde a suposição de independência entre características é razoável. É robusto para lidar com grandes conjuntos de dados categorizados rapidamente.
* **Interpretação e Explicabilidade**: Esse modelo é bastante interpretável, pois cada variável contribui independentemente para a classificação final. Isso facilita a compreensão do processo de classificação.
* **Desempenho Computacional**: O Naive Bayes é um dos modelos mais leves computacionalmente, sendo rápido para treinar e aplicar, mesmo em grandes volumes de dados.
* **Capacidade de Generalização**: Naive Bayes tem uma boa capacidade de generalização em dados com independência aproximada entre variáveis, sendo adequado para tarefas de classificação binária.

## Modelo 2: Árvore de Decisão – Gini

A Árvore de Decisão é um modelo que usa uma estrutura de árvore para dividir os dados em subconjuntos com base em características específicas. O critério Gini mede a impureza de um nó na árvore.

### Critérios de Seleção:

* **Precisão e Robustez**: A Árvore de Decisão com critério Gini permite uma classificação precisa ao dividir os dados em nós homogêneos. É robusta, pois pode lidar com dados sem sentidos.
* **Interpretação e Explicabilidade**: Oferece interpretabilidade visual e intuitiva, com regras claras que mostram como cada decisão é tomada.
* **Desempenho Computacional**: A Árvore de Decisão pode ser um pouco mais custosa computacionalmente em comparação com o Naive Bayes, especialmente em dados grandes, mas seu custo é administrável em nossa base.
* **Capacidade de Generalização**: A capacidade de generalização da Árvore de Decisão depende da profundidade da árvore. Em uma profundidade controlada, ela pode generalizar bem.

## Modelo 3: Árvore de Decisão – Entropia

Similar à árvore com critério Gini, mas usando a entropia para medir a pureza dos nós, maximizando a informação.

### Critérios de Seleção:

* **Precisão e Robustez**: É preciso em diferenciar os dados baseando-se em informação adicional (entropia), o que pode melhorar a precisão em certos casos.
* **Interpretação e Explicabilidade**: Mantém a facilidade de interpretação e visualização típica de Árvores de Decisão.
* **Desempenho Computacional**: Comparável ao critério Gini, mas com uma leve diferença no cálculo de entropia. Em geral, o desempenho computacional é equilibrado.
* **Capacidade de Generalização**: Similar ao critério Gini, pode generalizar bem com a profundidade ajustada.

## Modelo 4: K-Nearest Neighbors (KNN)

O KNN classifica uma nova amostra com base na proximidade dela com as amostras de treinamento mais próximas.

### Critérios de Seleção:

* **Precisão e Robustez**: O KNN é robusto e preciso em situações em que os dados possuem agrupamentos claros entre as classes.
* **Interpretação e Explicabilidade**: Sua explicabilidade é moderada, pois depende da proximidade com os vizinhos, o que pode ser intuitivo, mas não fornece uma regra clara.
* **Desempenho Computacional**: Pode ser custoso quando aplicado a dados grandes, pois precisa calcular a distância para cada ponto de treino, mas funciona bem com conjuntos menores.
* **Capacidade de Generalização**: A capacidade de generalização depende do número de vizinhos escolhidos, podendo ser ajustada para evitar sobreajuste.

# Aplicação dos modelos selecionados

**Divisão dos Dados**

Para avaliar a eficácia de cada modelo de maneira imparcial, dividimos o conjunto de dados em **conjuntos de treinamento e teste**, de modo que o modelo aprenda a partir de uma parte dos dados e seja avaliado em uma parte separada e inédita para ele. Nesta aplicação:

* **Proporção de Divisão:** Utilizamos uma divisão de 80% dos dados para treinamento e 20% dos dados para teste. Essa divisão permite que o modelo tenha uma quantidade adequada de dados para aprender.
* **Validação Cruzada:** Aplicamos a validação cruzada para garantir o teste no seu maior potencial. Com esse método dividimos o conjunto de dados em 5 (k) folds, onde o modelo e treinado em k – 1 partes e testado na parte restante, evitando o viés de uma única divisão.

**Métricas de Avaliação**

Para medir o desempenho e a eficácia dos modelos, utilizamos estas métricas:

* **Acurácia:** A taxa de acerto do modelo, indicando a porcentagem de previsões corretas.
* **Precisão**: Mede a qualidade das previsões positivas do modelo, verificando quantas das previsões positivas estão corretas para uma determinada classe.
* **Recall:** Mede a capacidade do modelo de identificar todas as instâncias positivas reais, ou seja, quantos dos verdadeiros positivos foram corretamente identificados.
* **F1-Score:** Encontra um equilíbrio entre Precision e Recall, sendo ideal quando há necessidade de balancear as duas métricas. Um F1-Score próximo de 1 indica um bom equilíbrio entre Precision e Recall, enquanto um F1-Score próximo de 0 indica desempenho insatisfatório.

## Modelo 1: Naive Bayes – GaussianNB

* **Treinamento:** O treinamento deste modelo foi direto, pois o Naive Bayes não possui hyperparâmetros. Apenas foi necessário garantir que os dados fossem padronizados para que o modelo pudesse interpretar as distribuições gaussianas corretamente.

## Modelo 2: Árvore de Decisão – Gini e Entropia

* **Treinamento:** O treinamento deste modelo foi usando seus hyperparâmetros, para conseguirmos o maior potencial do modelo.

## Modelo 3: K-Nearest Neighbors (KNN)

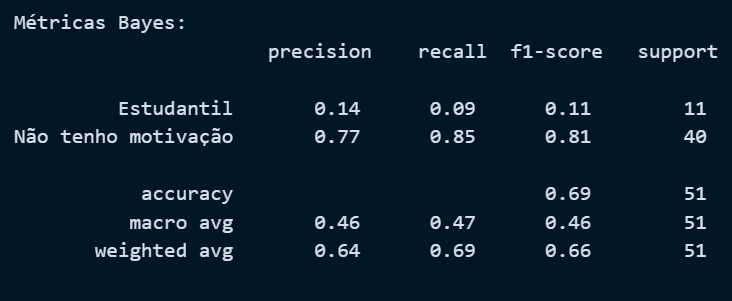
* **Treinamento:** O treinamento desde modelo foi testando o melhor número de k (números de vizinhos mais próximos) para selecionar o número que proporcionasse o melhor equilíbrio entre precisão e generalização.

# Análise dos Resultados

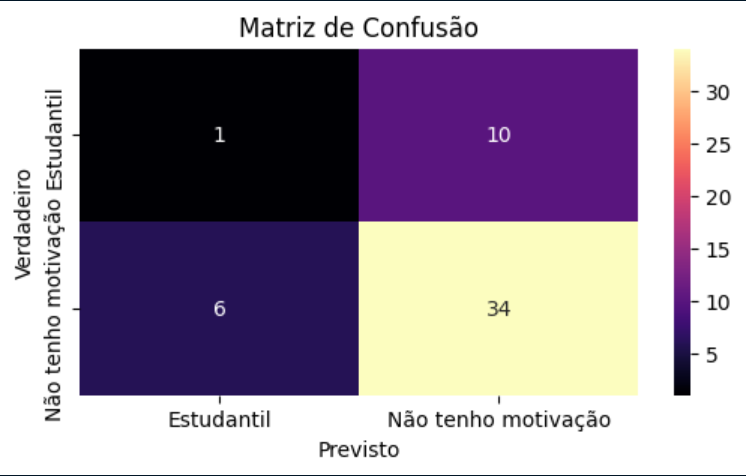
Em todos os modelos avaliamos todas as métricas mencionadas acima, vale lembrar que estamos analisando os modelos sem o balanceamento de classes e nem redução de dimensionalidades, vamos avaliar agora:

## Modelo 1: Naive Bayes – GaussianNB

**Métricas de Avaliação**



**Matriz de Confusão**



**Discussão dos Resultados**

* **Equilíbrio entre Precisão e Recall**: O modelo apresenta uma boa precisão e recall para a classe "Não tenho motivação" (a classe majoritária), indicando que está bem adaptado a ela. No entanto, o desempenho na classe "Estudantil" é muito baixo, com uma baixa precisão e recall, sugerindo dificuldade em identificar essa classe.
* **Classes Desbalanceadas**: Como esperado, o modelo tem dificuldades para classificar a classe menos frequente ("Estudantil"), refletindo o desbalanceamento de classes.
* **Erro e Overfitting**: A acurácia de 69% é moderada, mas o baixo f1-score para a classe minoritária sugere que o modelo está enviesado para a classe majoritária, o que pode indicar underfitting ao não captar bem as características da classe minoritária.

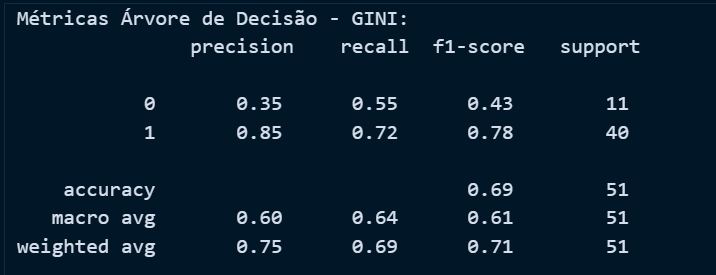
## Modelo 2: Árvore de Decisão – Gini

**Métricas de Avaliação**

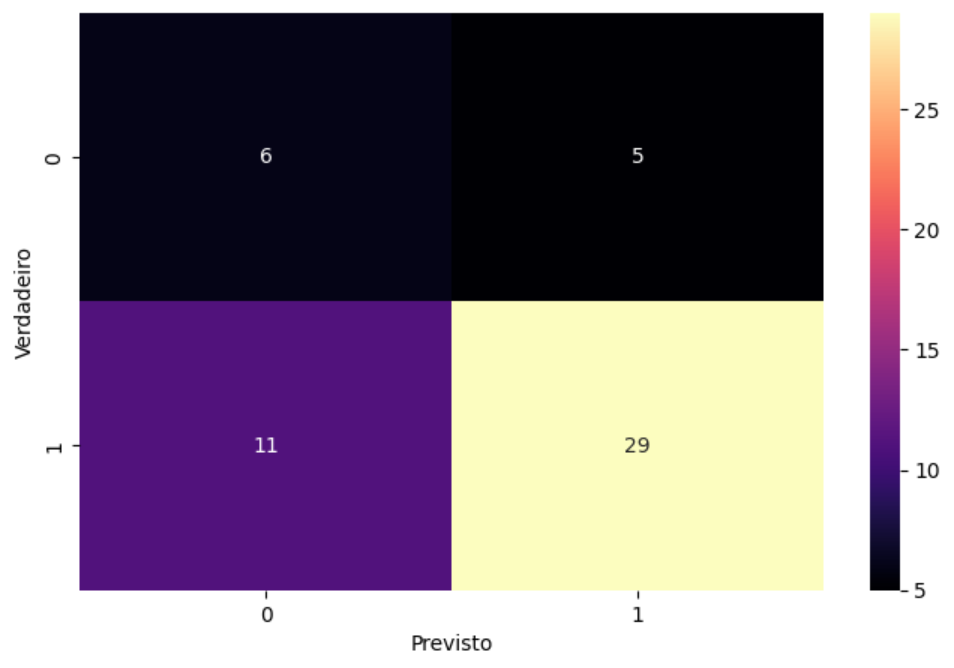
Legenda:

0 – Estudantil

1 – Não tenho motivação



**Matriz de Confusão**

****

**Discussão dos Resultados:**

* **Equilíbrio entre Precisão e Recall**: O modelo mostrou um bom equilíbrio entre precisão e recall para a classe "1", indicando que consegue identificar corretamente a classe majoritária. No entanto, a precisão e o recall para a classe "0" são baixos, com um recall ligeiramente menor o que evidencia a dificuldade do modelo em identificar corretamente a classe minoritária.
* **Classes Desbalanceadas**: Assim como o Naive Bayes, a árvore de decisão também encontra desafios em lidar com a classe menos frequente, embora seu desempenho seja um pouco melhor para a classe "0", evidenciando que o modelo tem dificuldades para classificar corretamente a classe "0".
* **Erro e Overfitting**: Com uma acurácia de 69%, o modelo não apresenta sinais claros de overfitting. No entanto, o desempenho desigual entre as classes indica que ele poderia se beneficiar de técnicas de balanceamento para melhorar a performance na classe minoritária.

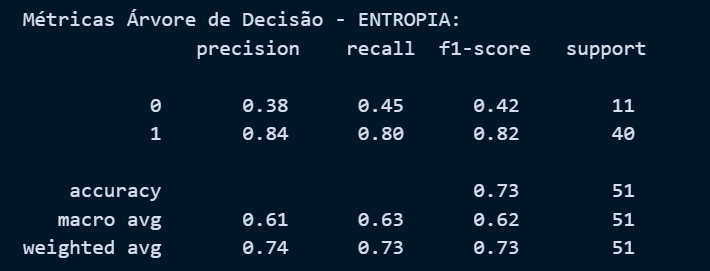
## Modelo 3: Árvore de Decisão – Entropia

**Métricas de Avaliação**

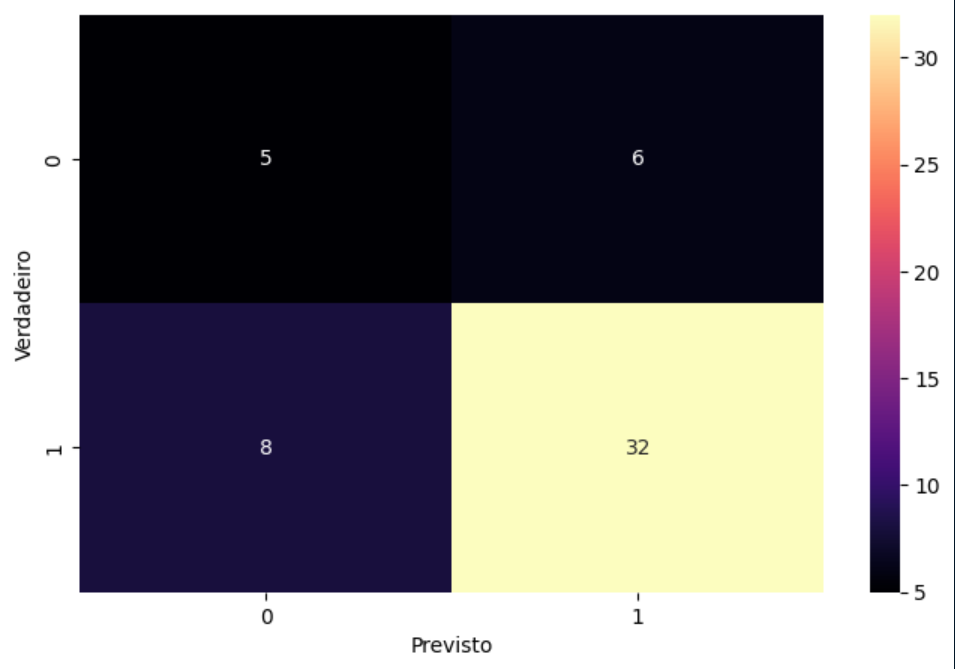
Legenda:

0 – Estudantil

1 – Não tenho motivação



**Matriz de Confusão**

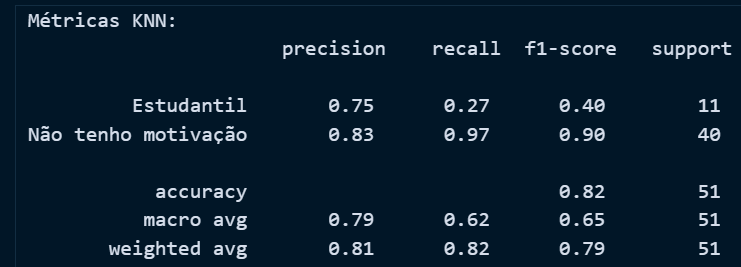
****

**Discussão dos Resultados:**

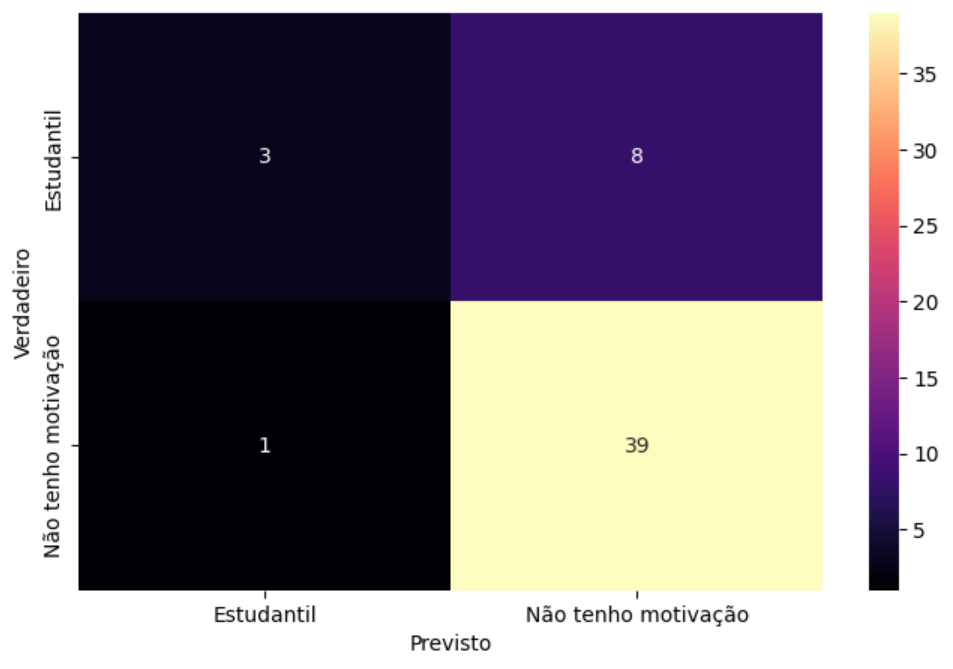
* **Equilíbrio entre Precisão e Recall**: Este modelo apresenta um equilíbrio relativamente bom entre precisão e recall para a classe "1", indicando que consegue classificar corretamente. O desempenho na classe "0" ainda são baixos, mas um pouco mais equilibrados em comparação ao modelo Gini, mostrando uma leve melhora na captura de exemplos da classe minoritária.
* **Classes Desbalanceadas**: Este modelo apresente algumas melhorias em comparação ao Gini, ele ainda enfrenta dificuldades em lidar com a classe minoritária, indicando que o balanceamento de classes poderia ajudar a melhorar os resultados.
* **Erro e Overfitting**: A acurácia de 73% e o desempenho mais equilibrado nas métricas sugerem que este modelo está um pouco mais adaptado ao conjunto de dados do que o modelo com Gini, possivelmente indicando uma menor tendência de underfitting.

## Modelo 4: K-Nearest Neighbors (KNN)

**Métricas de Avaliação**

****

**Matriz de Confusão**

****

**Discussão dos Resultados:**

* **Equilíbrio entre Precisão e Recall**: Este modelo apresenta uma alta precisão e recall para a classe majoritária (“Não tenho motivação"), o que é positivo. No entanto, a classe minoritária tem um recall muito baixo (0.27), indicando que o modelo está "tendendo" a classificar instâncias como pertencentes à classe majoritária.
* **Classes Desbalanceadas**: O KNN tem dificuldades com a classe minoritária, refletindo sua maior sensibilidade ao desbalanceamento de classes, pois classifica a maioria das instâncias como pertencentes à classe dominante.
* **Erro e Overfitting**: A alta acurácia de 82% sugere que o modelo está adequado para a maioria das instâncias, mas o baixo desempenho na classe minoritária indica que ele pode estar enviesado, caracterizando-se como um possível caso de underfitting para essa classe.

## Comparação Entre Modelos:

### Análise Comparativa

**Acurácia:**

* **Melhor Modelo:** K-Nearest Neighbors com 82% de acurácia.
* **Explicação:** O KNN teve a maior acurácia de todos os modelos, o que indica um bom desempenho. Isso é um reflexo da sua capacidade de capturar a maior parte das instâncias corretamente, principalmente na classe majoritária.

**Precisão e Recall para a Classe 0:**

* **Melhor Precisão:** K-Nearest Neighbors (KNN) com 75% de precisão.
* **Melhor Recall:** Árvore de Decisão – GINI com 55% de recall.
* **Explicação:** A Árvore de Decisão Gini apresenta o maior recall para a classe 0, o que significa que consegue identificar mais instâncias para a classe minoritária em comparação com os outros modelos. Entretando, o KNN tem a maior precisão para a classe 0 entre os modelos, embora com um recall muito baixo, indicando que, quando classifica corretamente, é preciso, mas perde muitos casos da classe minoritária.

**Precisão e Recall para a Classe 1:**

* **Melhor Precisão:** Árvore de Decisão – Gini com 85% de precisão.
* **Melhor Recall:** K-Nearest Neighbors (KNN) com 97% de recall.
* **Explicação:** O KNN se destaca com um alto recall na classe 1, indicando que ele captura quase todas as instâncias dessa classe. A alta precisão da Árvore de Decisão – Gini sugere que ele é bom em evitar falsos positivos para essa classe, embora com um recall um pouco mais baixo.

**F1-Score**

* **Classe 0:** Árvore de Decisão – Gini, tem o melhor f1-score com 43%.
* **Classe 1:** K-Nearest Neigbors (KNN), tem o melhor f1-score com 90%, se destacando-se como o mais equilibrado em precisão e recall para essa classe.
* **Explicação:** O f1-score do KNN para a classe 1 mostra que ele é excelente para esta classe, mas sua performance para a classe 0 é limitada. A Árvore de Decisão – Gini apresenta o melhor equilíbrio para a classe 0, tornando-se potencialmente mais útil quando a classe minoritária é relevante.

### Forças e Fraquezas

#### Modelo 1: Naive Bayes – GaussianNB

**Forças:**

* Bom desempenho na classe majoritária (classe 1), com precisão e recall altos.

**Fraquezas:**

* Baixo desempenho na classe minoritária (classe 0), com baixa precisão e recall, refletindo uma dificuldade em lidar com desbalanceamentos de classes.

**Conclusão:** Esse modelo é adequado quando o foco está em capturar corretamente a classe majoritária, mas ele apresenta baixa adaptabilidade para a classe minoritária.

#### Modelo 2: Árvore de Decisão – Gini

**Forças:**

* Apresenta um equilíbrio razoável entre precisão e recall para a classe majoritária, com boa precisão e recall, indicando que consegue capturar a maioria dos exemplos desta classe.
* Desempenho moderado em recall para a classe minoritária, conseguindo identificar algumas instâncias desta classe.

**Fraqueza:**

* A precisão para a classe minoritária é baixa, o que indica uma dificuldade em classificar corretamente essa classe em um conjunto desbalanceado, prejudicando o desempenho geral do modelo.

**Conclusão:** É uma opção viável quando se busca um equilíbrio entre precisão e recall para a classe majoritária, mas tem limitações ao identificar corretamente a classe minoritária. Ele pode ser útil quando a prioridade é capturar a classe majoritária.

#### Modelo 3: Árvore de Decisão – Entropia

**Forças:**

* Maior acurácia geral 73% em comparação com o critério Gini, o que sugere um desempenho mais adaptado ao conjunto de dados.
* Melhor equilíbrio entre precisão e recall para a classe majoritária, indicando uma boa capacidade de classificação para essa classe.

**Fraqueza:**

* Embora apresente uma leve melhora no desempenho para a classe minoritária em relação ao modelo com critério Gini, ainda enfrenta dificuldades em classificá-la adequadamente, com recall e precisão baixos para essa classe.

**Conclusão:** Esse modelo é uma escolha equilibrada, com uma leve vantagem em acurácia sobre o critério Gini, mas ainda limitado em capturar adequadamente a classe minoritária.

#### Modelo 4: K-Nearest Neighbors (KNN)

**Forças:**

* Maior acurácia entre todos os modelos, com excelente desempenho na classe majoritária.
* Alto recall na classe majoritária, o que é útil quando se prioriza evitar falsos negativos para essa classe.

**Fraquezas:**

* Baixíssima recall para a classe minoritária, o que indica uma dificuldade severa em capturar instâncias dessa classe. Isso limita seu uso em cenário onde a classe minoritária é relevante.

**Conclusão:** O KNN é ideal para cenários onde o foco está na classe majoritária e onde a acurácia geral é mais importante do que o equilíbrio entre classes. No entanto, ele é o menos recomendado para lidar com o desbalanceamento de classes.

### Concluindo os Modelos

**Se o foco é capturar a classe minoritária (classe 0):** A Árvore de Decisão com critério Gini oferece o melhor equilíbrio para essa classe e é mais adequada.

**Se a prioridade é a acurácia geral e o foco está na classe majoritária:** O KNN apresenta o melhor desempenho em termos de acurácia e recall para a classe majoritária.

**Para um equilíbrio entre as classes:** A Árvore de Decisão com critério Entropia oferece um bom compromisso entre acurácia e desempenho equilibrado entre as classes.

# Ajustes Necessários

Com base na análise dos problemas identificados, vamos explorar estratégias para ajustar os modelos e os dados a fim de melhorar o desempenho, especialmente para a classe minoritária.

## Identificação de Problemas:

**Problemas de Desempenho:**

* **Baixa precisão e recall na classe minoritária**: Em todos os modelos, observamos que a classe minoritária teve um desempenho inferior, com baixa precisão e baixo recall, especialmente em comparação com a classe majoritária.
* **Desempenho desbalanceados nas métricas:** Alguns modelos apresentaram boa acurácia, mas outras métricas, como precisão e recall, não foram equilibradas entre as classes, indicando que a acurácia sozinha pode não refletir a real performance do modelo.

**Dados Desbalanceados:**

* **Predominância da classe majoritária:** O desbalanceamento de classes está influenciando os modelos a "chutarem" predominantemente na classe majoritária, resultando em baixa precisão e recall para a classe minoritária.
* **Impacto negativo no desempenho geral:** Este desbalanceamento faz com que os modelos apresentem métricas desproporcionais, com acurácia alta mas com fraco desempenho para a classe minoritária, o que reduz a utilidade do modelo para casos da classe menos comum.

## Mudanças na Base de Dados:

Para melhorar as métricas, podemos aplicar alguns métodos para isso:

**Redução de dimensionalidade:** Podemos usar o método do PCA, ele é usado para reduzir a dimensionalidade do conjunto de dados. Com ele podemos identificar as variações dos dados, ajudando para reduzir a dimensionalidade dele. Em nosso projeto, eu apliquei o PCA, mas tive uma perda das métricas, piorando todas elas, ou seja, priorizei não usar ele.

**Balanceamento de Classes:**

* **Oversampling da Classe minoritária (Smote):** Estamos aplicando o método Smote na base de treino da nossa IA, esse método aumenta a quantidade de amostras da classe minoritária, criando instâncias sintéticas semelhantes às existentes. Isso ajuda a balancear a distribuição das classes, permitindo que o modelo aprenda de forma mais equilibrada e aumenta a performance na classe minoritária. Conseguimos um ótimo resultado com o método Smote, melhorando as métricas.
* **RandomUnderSample para diminuir a classe majoritária e balancear as classes:** Aplicamos em nosso projeto o RandomUnderSample, esse método reduz o número de amostras na classe majoritária ao subamostrar, mantendo o mesmo número de exemplos para as classes minoritárias e majoritárias. Com este método, em todos os modelos tivemos uma perda gigantesca das métricas. Resolvemos não levar ele para frente, pois esse método por conta de reduzir a classe majoritária ao remover exemplos aleatórios, causa uma perda de informações, e pode acontecer de amostrar importantes, que contribuem para uma melhor generalização do modelo, serem removidos. Isso pode levar a um modelo com menor precisão, especialmente se os dados excluídos tivessem características importante para a identificação das classes.

## Ajustes nos Modelos:

Para melhorar o desempenho dos modelos, os ajustes nos hiperparâmetros e a utilização de validação cruzada são essenciais.

**Ajuste de Hiperparâmetros**

**Grid Search:** Utilizar Grid Search para testar combinações de hiperparâmetros e encontrar a configuração ideal para cada modelo, sempre buscando as melhores combinações:

* **Naive Bayes:** Ajustar os parâmetros de suavização.
* **Árvore de Decisão:** Ajustar a profundidade máxima, o número mínimo de amostras por folha. Buscar a melhor combinação dos hiperparâmetros.
* **K-Nearest Neighbors (KNN):** Testar o melhor k (número de vizinhos) possível para a melhor acurácia e métodos de ponderação de vizinhos (uniforme ou distância).

**Validação Cruzada**

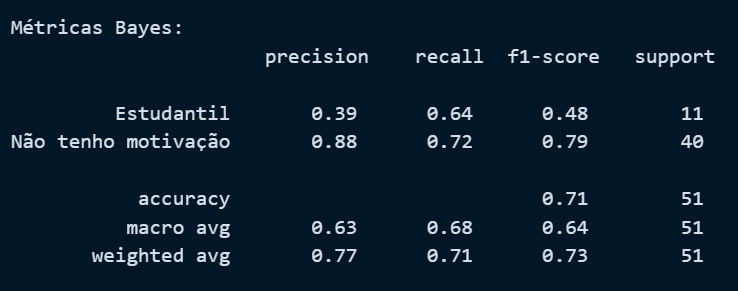
* **K-Fold Cross-Validation:** Implementar a validação cruzada, em nosso caso usamos 5 folds, para avaliar o desempenho dos modelos de forma mais robusta. Essa técnica ajuda a reduzir o risco de overfitting, pois cada parte dos dados é usado tanto para treinamento quanto para validação.

# Impacto das Modificações

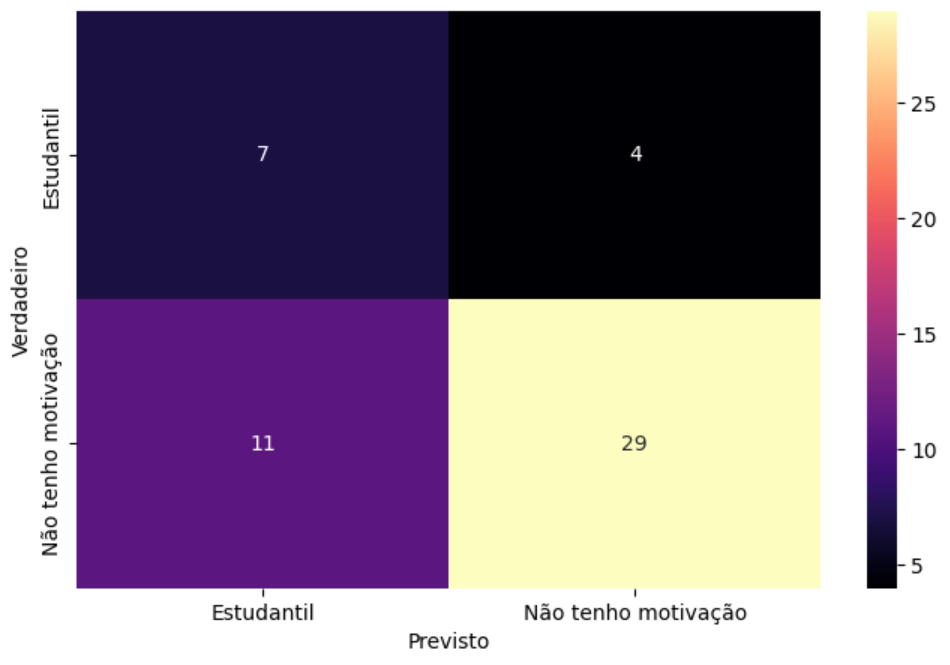
Para cada modelo, foram aplicadas técnicas de balanceamento de dados, como o SMOTE, além de otimizações via Grid Search com Cross-Validation. Essas modificações ajudaram a melhorar as métricas, permitindo avaliar o desempenho considerando diferentes aspectos, como o equilíbrio entre precisão e recall, o impacto do desbalanceamento de classes e potenciais sinais de overfitting ou underfitting.

## Modelo 1: Naive Bayes – GaussianNB

**Métricas de Avaliação**

****

**Matriz de Confusão**

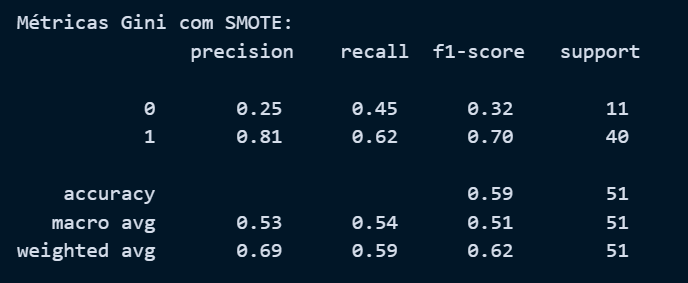
****

**Discussão dos Resultados:**

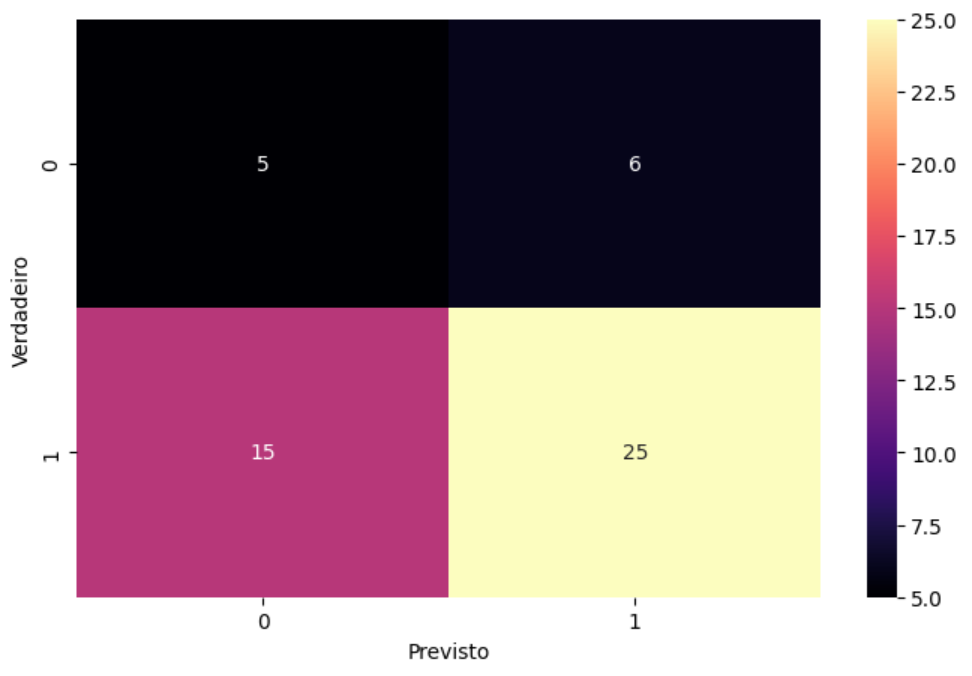
* **Equilíbrio entre Precisão e Recall:** O modelo teve um desempenho muito bom para a classe "Não tenho motivação", especialmente no recall (0.97). No entanto, a baixa precisão e recall na classe "Estudantil" indicam que o modelo tende a focar na classe majoritária, não capturando bem a classe minoritária.
* **Classes Desbalanceadas:** O modelo mostra dificuldade em classificar corretamente a classe menos frequente ("Estudantil"), um problema comum em dados desbalanceados, mesmo após a aplicação do SMOTE.
* **Erro e Overfitting:** Com uma acurácia de 80%, o modelo parece adequado para a classe dominante, mas o baixo f1-score para a classe minoritária sugere underfitting em relação a essa classe, refletindo que ele não conseguiu captar bem as suas características.

## Modelo 2: Árvore de Decisão – Gini

**Métricas de Avaliação**

****

**Matriz de Confusão**

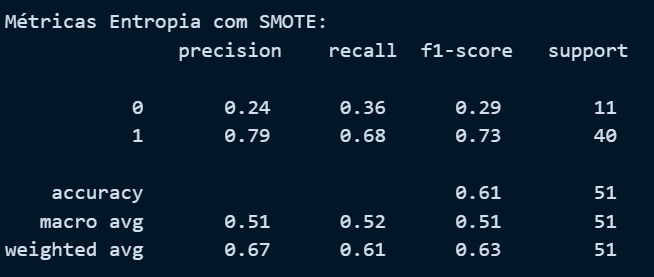
****

**Discussão dos Resultados:**

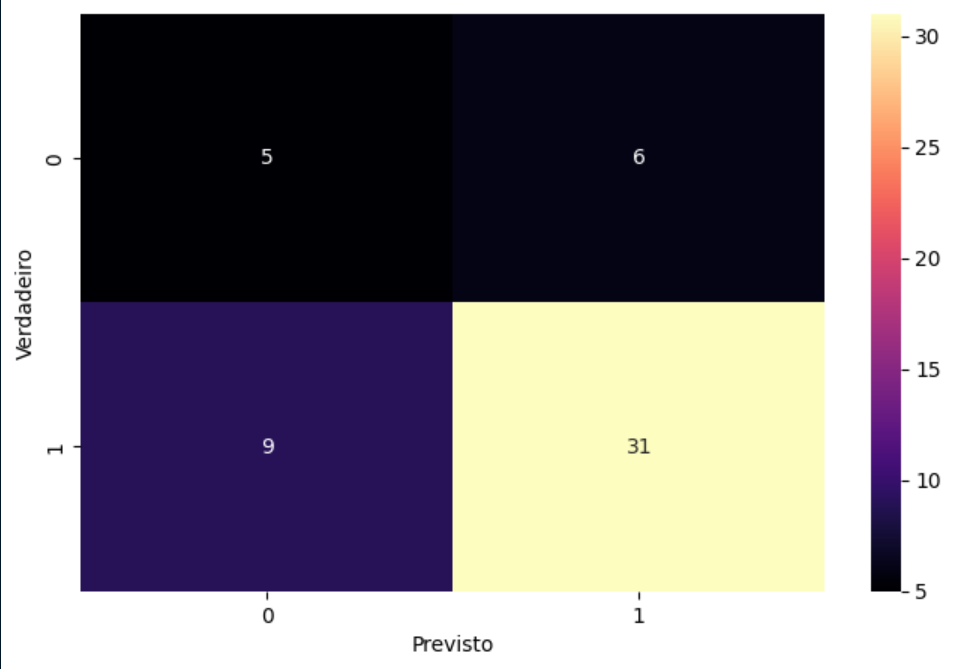
* **Equilíbrio entre Precisão e Recall:** O modelo tem um bom equilíbrio para a classe majoritária (1), com precisão e recall de 0.80. A precisão e recall baixos para a classe minoritária (0) indicam que o modelo continua a ter dificuldades em identificar essa classe.
* **Classes Desbalanceadas:** A aplicação do SMOTE ajudou, mas o desempenho para a classe minoritária ainda é limitado, mostrando que a árvore de decisão com critério Gini tem dificuldades em lidar com o desbalanceamento, embora melhor que o Naive Bayes.
* **Erro e Overfitting:** A acurácia de 69% é satisfatória e o modelo não apresenta sinais evidentes de overfitting, mas o desempenho desigual entre as classes indica que ele ainda não consegue captar plenamente as características da classe minoritária.

## Modelo 3: Árvore de Decisão – Entropia

**Métricas de Avaliação**

****

**Matriz de Confusão**

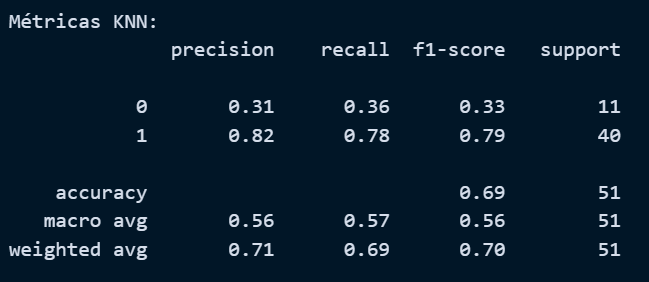
****

**Discussão dos Resultados:**

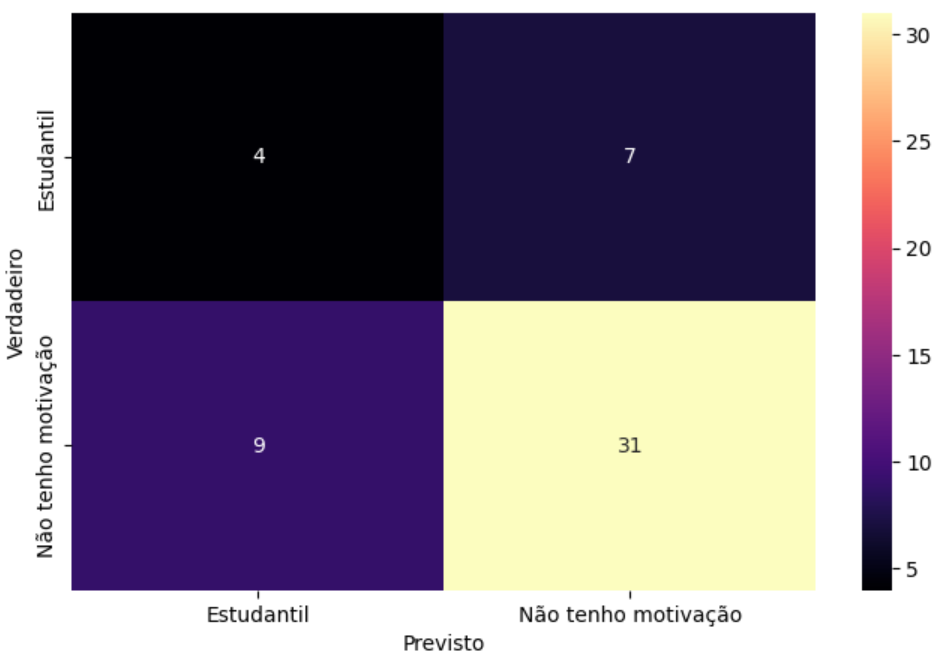
* **Equilíbrio entre Precisão e Recall:** Com um equilíbrio mais favorável entre precisão e recall na classe majoritária (1) e um desempenho um pouco melhor para a classe minoritária (0) do que o modelo Gini, o critério de Entropia permitiu um ganho nas métricas.
* **Classes Desbalanceadas:** Ainda que o modelo continue a ter desafios com a classe menos frequente, o desempenho é ligeiramente melhor do que com o critério Gini. A maior precisão e recall na classe minoritária mostram que o critério de Entropia conseguiu captar um pouco mais as características dessa classe.
* **Erro e Overfitting:** A acurácia de 71% e as métricas balanceadas sugerem que o modelo com Entropia está menos sujeito a underfitting em comparação com o critério Gini, apresentando uma melhor adaptação às duas classes.

## Modelo 4: K-Nearest Neighbors (KNN)

**Métricas de Avaliação**

****

**Matriz de Confusão**

****

**Discussão dos Resultados**

* **Equilíbrio entre Precisão e Recall:** O modelo KNN mostrou-se eficaz em capturar a classe majoritária com alta precisão e recall, mas teve uma precisão e recall baixos para a classe minoritária, indicando tendência a favorecer a classe dominante.
* **Classes Desbalanceadas:** Como um modelo sensível a instâncias próximas, o KNN reflete um impacto maior do desbalanceamento de classes, mostrando limitações em classificar corretamente a classe menos frequente, mesmo com SMOTE.
* **Erro e Overfitting:** A acurácia de 69% é sólida, mas a dificuldade do modelo em captar a classe minoritária sugere um leve underfitting para essa classe. Ele ainda não consegue generalizar bem para todas as instâncias, sendo mais adequado para a classe dominante.

## Comparação Entre Modelos:

### Análise Comparativa

**Acurácia**

* **Melhor Modelo:** Naive Bayes com 71% de acurácia.
* **Explicação:** O modelo Bayes teve a maior acurácia, indicando uma boa capacidade de generalização ao classificar as instâncias, especialmente em relação à classe majoritária.

**Precisão e Recall para a Classe 0:**

* **Melhor Precisão:** KNN com SMOTE, com 31% de precisão.
* **Melhor Recall:** Bayes com SMOTE, com 64% de recall.
* **Explicação:** Embora o KNN apresente uma precisão relativamente baixa para a classe 0, o modelo Bayes destaca-se ao identificar um número maior de instâncias dessa classe, o que sugere uma capacidade superior para detectar casos da classe minoritária.

**Precisão e Recall para a Classe 1:**

* **Melhor Precisão:** Bayes com SMOTE, com 88% de precisão.
* **Melhor Recall:** KNN com SMOTE, com 78% de recall.
* **Explicação:** O modelo Bayes é eficaz em evitar falsos positivos para a classe 1, apresentando a maior precisão. No entanto, o KNN captura uma porcentagem significativa de instâncias dessa classe, refletindo um desempenho robusto.

**F1-Score:**

* **Classe 0:** Bayes com SMOTE, com f1-score de 48%.
* **Classe 1:** Bayes com SMOTE, com f1-score de 79%.
* **Explicação:** O modelo Bayes é o mais equilibrado em termos de f1-score para a classe 1, destacando-se na precisão e no recall. Para a classe 0, embora o desempenho não seja ideal, ainda é superior ao dos outros modelos, sugerindo que ele é mais confiável em geral.

### Forças e Fraquezas

#### Modelo 1: Naive Bayes – GaussianNB

**Forças:**

* Alto desempenho em precisão e recall para a classe "Não tenho motivação", mostrando a capacidade do modelo de identificar corretamente a classe majoritária.
* Boa acurácia geral de 71%, sugerindo um bom balanceamento nas previsões.

**Fraquezas:**

* Desempenho moderado na classe "Estudantil", com precisão de 0.39 e recall de 0.64, indicando que o modelo pode perder instâncias importantes da classe minoritária.

**Conclusão:** Este modelo é adequado para cenários onde a detecção da classe majoritária é prioritária, mas pode não ser ideal para contextos que requerem uma identificação eficaz da classe minoritária.

#### Modelo 2: Árvore de Decisão – Gini

**Forças:**

* Melhoria na identificação da classe 1, minimizando os falsos positivos.
* Alta precisão para classe 1.

**Fraquezas:**

* Baixa precisão e recall para a classe 0, resultando em desempenho geral insatisfatório com acurácia de apenas 59%.

**Conclusão:** Esse modelo é útil em situações onde a prioridade é evitar falsos positivos na classe 1, mas sua capacidade de reconhecer a classe minoritária é limitada, comprometendo sua eficácia em aplicações que exigem um equilíbrio entre ambas as classes.

#### Modelo 3: Árvore de Decisão – Entropia

**Forças**

* Oferece um equilíbrio moderado entre precisão e recall para a classe 1.

**Fraqueza**

* Desempenho fraco na classe 0, com baixa precisão e recall, limitando sua utilidade em situações críticas.

**Conclusão:** Esse modelo pode ser aplicado em situações onde um desempenho equilibrado é necessário, mas não é ideal para cenários que requerem alta eficácia na detecção da classe minoritária.

#### Modelo 4: K-Nearest Neighbors

**Forças**

* Bom recall para a classe “Não tenho motitvação”, capturando uma parte significativa das instâncias relevantes.

**Fraquezas**

* Baixa precisão para a classe 0, resultando em uma taxa de falsos positivos relativamente alta.

**Conclusão:** Esse modelo é adequado quando o foco está em identificar corretamente a classe 1, mas sua eficácia na classe minoritária é preocupante, o que limita sua utilidade em casos onde ambas as classes são críticas.

# Conclusão

## Principais Descobertas e Eficácia dos Modelos

**Naive Bayes (GaussianNB) com SMOTE**: Esse modelo alcançou a maior acurácia, apresentando resultados sólidos tanto em precisão quanto em recall para a classe majoritária ("Não tenho motivação") e um desempenho aceitável na classe minoritária ("Estudantil"). Com 88% de precisão e 64% de recall para a classe majoritária, ele demonstra robustez na identificação correta das instâncias de ambas as classes, embora com uma ligeira fraqueza na classe minoritária. Esse equilíbrio o torna adequado para aplicações que exigem alta eficácia para ambas as classes, mantendo uma precisão consistente e minimizando falsos positivos.

**Árvore de Decisão - Gini**: O modelo com critério de Gini apresentou um recall e uma precisão modestos para a classe majoritária, mas seu desempenho geral ficou abaixo do esperado, com uma acurácia de apenas 59%. A identificação limitada da classe minoritária e uma baixa precisão para a classe 0 fazem com que este modelo não seja o ideal para cenários que demandam um balanceamento entre as classes.

**Árvore de Decisão - Entropia**: Embora tenha apresentado um equilíbrio aceitável entre precisão e recall para a classe majoritária, o desempenho na classe minoritária foi insuficiente, refletindo baixa acurácia e falta de sensibilidade para casos minoritários. Assim, esse modelo é menos indicado para situações que exigem uma alta taxa de identificação de todas as classes.

**K-Nearest Neighbors (KNN) com SMOTE**: O KNN demonstrou um bom recall na classe majoritária, capturando adequadamente as instâncias relevantes. No entanto, a precisão baixa para a classe minoritária e uma taxa elevada de falsos positivos limitam sua aplicabilidade, tornando-o menos confiável em cenários onde é necessário um balanceamento eficaz entre as classes.

Podemos observar que em todos os modelos, tivemos uma performance melhor na classe majoritária, mas em contrapartida tiveram alguns modelos que a classe minoritária ficou bem abaixo do esperado, o que contribuiu para uma acurácia baixa. O principal problema disso foi a falta de balanceamento entre as classes.

## Etapas recomendadas para aprimorar a análise

Para aprimorar a análise, podemos ajustar alguns métodos para conseguirmos melhores métricas:

* **Ajustar parâmetros:** Realizar ajustes adicionais com o Grid Search para explorar ainda mais possibilidades de aprimoramento, focando em um balanceamento ainda mais potente entre as classes.
* **Reavaliar o balanceamento:** Testar outras técnicas de balanceamento além do Smote e RandomUnderSample para verificar se há impacto positivo nas métricas da classe minoritária.

## Conclusão Final

A análise dos quatro modelos Naive Bayes - GaussianNB, Árvore de Decisão - Gini, Árvore de Decisão - Entropia e K-Nearest Neighbors (KNN) indicou que, apesar das variações nos desempenhos em diferentes métricas, o Naive Bayes se destacou como o modelo mais equilibrado e eficaz para atender uma acurácia relevante e um balanceamento entre as classes. Com uma acurácia de 71%, ele demonstrou a melhor capacidade de generalização, identificando corretamente a maioria das instâncias e equilibrando adequadamente as classes majoritárias e minoritárias.

A escolha do modelo Naive Bayes traz um impacto positivo no projeto, pois ele assegura um equilíbrio entre as classes e uma precisão consistente para a classe majoritária, o que é crucial no contexto em questão. Esse equilíbrio é essencial para garantir que o aplicativo ofereça previsões que atendam tanto aos usuários com motivação estudantil quanto àqueles que podem não ter tanta motivação, sem comprometer o desempenho geral.

# Modelo no APP

Para implementar o modelo Naiva Bayes – GaussianNB no aplicativo EducaEco, seguimos uma série de etapas, detalhadas a seguir:

* **Serialização do Modelo e Pré-processador:** Iniciamos com a serialização do modelo Naive Bayes e do preprocessador de dados. Esse passo foi fundamental para garantir que ambos pudessem ser carregados posteriormente no aplicativo.
* **Criação do Formulário:** Desenvolvemos um formulário em HTML, CSS e JavaScript, cujas perguntas correspondem às colunas da nossa base de dados. Esse formulário coleta as respostas dos usuários, que serão usadas para realizar a previsão.
* **Integração com Flask:** Em seguida, construímos uma aplicação Flask para consumir o formulário. Nessa aplicação, realizamos a desserializção do modelo e do preprocessador para que fiquem prontos para a classificação.
* **Processamento dos Dados no Formulário:** As respostas do formulário são transformadas em um DataFrame, para que o modelo possa processá-las adequadamente. Aplicamos o preprocessador para converter variáveis categóricas em numéricas, preparando as entradas para o modelo.
* **Classificação com o Modelo:** Com o modelo desserializado, realizamos a previsão. A IA retorna “0” para a classe “Estudantil” e “1” para a classe “Não tenho motivação”.
* **Armazenamento dos Resultados:** A previsão da IA, junto com o nome e e-mail do usuário, é enviada para uma API desenvolvida pela equipe de desenvolvimento. Essa API armazena os dados no banco de dados MongoDB, nosso sistema de armazenamento não relacional.
* **Deploy da Aplicação no Render: Finalmente, hospedamos a aplicação Flask no serviço Render e a incorporamos no aplicativo na área restrita, utilizamos uma WebView para fácil acesso dos usuários.**

# RPAs

Neste projeto, implementamos três RPAs para automatizar processos e otimizar o fluxo de trabalho. As automações foram divididas em duas categorias: Manuais e AWS.

## Manuais

Nesta categoria, temos dois RPAs que funcionam manualmente, sem integração com a AWS ou a pipeline. Eles foram desenvolvidos para tarefas que não exigem uma execução contínua e automática.

* **RPA para cadastro de usuários no Firebase**: Este RPA automatiza o processo de cadastramento dos alunos no Firebase para o aplicativo, gerando automaticamente uma estrutura de e-mail e senha para cada aluno a partir de um arquivo Excel. Em vez de realizar cadastros individuais, basta alimentar o programa com o arquivo que ele processa rapidamente todos os cadastros. Como os alunos não fazem o cadastro diretamente, esse RPA otimiza significativamente o tempo, automatizando a inclusão deles no aplicativo.
* **RPA para inserção de dados nas tabelas do Servlets**: O segundo RPA manual insere informações diretamente nas tabelas do Servlets. A partir de um arquivo Excel, ele realiza a inserção automática, garantindo economia de tempo e padronização dos dados nos bancos. Essa padronização agiliza processos e evita erros manuais na inserção das informações.

## AWS:

A categoria AWS abrange o RPA configurado para rodar automaticamente na infraestrutura da AWS, integrado com a pipeline do projeto para execução contínua.

* **RPA para sincronização entre bancos de dados**: Este RPA é essencial para garantir a consistência de dados entre dois bancos de dados, um do primeiro ano e outro do segundo ano. Ele busca dados novos no banco do segundo ano, os insere no banco do primeiro ano e, em seguida, atualiza o banco do segundo ano com as informações do primeiro ano (mas sem alterar o conteúdo original do primeiro). Dessa forma, o RPA automatiza a troca de dados, mantendo ambos os bancos sincronizados e atualizados.