Projeto de Aprendizado de Máquina - Documentação

Amanda Magalhães Lima / RA: 11201920607
Ferdinando Massimo Kessin Longoni / RA: 11202020600
Nayara Valéria Joca Gonçalves / RA: 11201921427

1. Introdução

Este projeto foi inicialmente concebido para explorar a interseção entre processamento de linguagem natural e geração de áudio, visando criar música a partir de descrições textuais detalhadas. A ideia era utilizar a base de dados **MusicCaps**, que fornece pares de texto e clipes musicais, para desenvolver um modelo capaz de traduzir nuances textuais em características musicais, como gênero, humor e ritmo. Apesar de inovadora, a abordagem apresentou desafios significativos em termos de processamento computacional e complexidade do modelo, levando-nos a reconsiderar o escopo do trabalho.

Dessa forma, decidimos redirecionar o foco para um problema prático e mais alinhado com nossos recursos: a análise de dados educacionais do **ENEM (Exame Nacional do Ensino Médio)**. Essa nova proposta explora a identificação de padrões e a previsão de aprovação de alunos com base em informações contidas na base do ENEM e em dados históricos de notas de corte de nossa faculdade.

O ENEM é uma prova ampla e de grande importância no Brasil, servindo como porta de entrada para inúmeras universidades. A base de dados do exame oferece informações detalhadas sobre as questões, alternativas, assuntos abordados e características dos alunos que realizaram a prova. Com essa riqueza de informações, vislumbramos a oportunidade de aplicar técnicas de aprendizado de máquina para identificar agrupamentos (clusters) de estudantes, prever notas e simular aprovações de maneira eficiente.

A proposta atual visa não apenas uma aplicação prática de aprendizado de máquina no contexto educacional, mas também a criação de um modelo que possa ser escalado para auxiliar instituições de ensino na análise de desempenho de candidatos, potencializando processos seletivos e identificando oportunidades de melhorias pedagógicas.

2. Base de Dados

2.1 Base Inicial - MusicCaps

A **MusicCaps** é uma base de dados voltada para a geração de música a partir de descrições textuais e foi introduzida no artigo "MusicLM: Generating Music From Text", de Agostinelli et al. A base é composta por **5,5 mil pares de música e texto**, onde cada trecho musical de **10 segundos** é acompanhado por uma descrição detalhada fornecida por **especialistas humanos**.

2.1.1 Características da Base MusicCaps

1. **Descrições Textuais**
   Para cada clipe musical, a base inclui uma legenda em **texto livre**, composta em média por quatro frases. Essas legendas descrevem aspectos da música, incluindo a sua estrutura emocional e técnica. As descrições são cuidadosamente elaboradas para capturar a essência musical e incluem informações sobre o gênero, ritmo, instrumentos, humor e mais.
2. **Aspectos Musicais**
   Além da legenda em texto livre, cada clipe também possui uma lista detalhada de aspectos musicais. Essa lista descreve elementos importantes da música, como:
   - **Gênero**: o tipo de música (ex.: pop, rock, jazz);
   - **Humor**: a sensação emocional da música (ex.: alegre, melancólica, introspectiva);
   - **Tempo**: a velocidade da música (ex.: rápido, moderado, lento);
   - **Vozes**: se há vozes cantando, e a descrição delas;
   - **Instrumentação**: quais instrumentos estão presentes (ex.: guitarra, bateria, piano);
   - **Dissonâncias**: elementos harmônicos que criam tensão musical;
   - **Ritmo**: características rítmicas e métricas da música.
3. **Formato de Análise e Aplicações**
   A estrutura rica da base facilita a análise de como descrições textuais complexas podem ser convertidas em música. Os dados fornecem um ponto de partida para o treinamento de modelos de aprendizado de máquina que

visam gerar música baseada em descrições. A descrição detalhada dos aspectos musicais e emocionais oferece uma oportunidade única para explorar como diferentes elementos do texto podem influenciar a composição musical.

2.1.2 Motivação e Desistência da Base MusicCaps

A **MusicCaps** foi escolhida inicialmente por sua riqueza textual, que permite uma correspondência detalhada entre palavras e sons. Essa correspondência é fundamental para o desenvolvimento de modelos de aprendizado de máquina que buscam gerar música com base em descrições, um campo de pesquisa promissor e inovador. A base oferece a possibilidade de analisar como diferentes características musicais podem ser manipuladas e geradas a partir de uma simples descrição textual, o que abriria portas para novas aplicações na indústria da música e em áreas criativas. Contudo, devido à complexidade computacional exigida para implementar modelos compatíveis, optamos por mudar o escopo do projeto.

2.2 Base Atual - ENEM

A nova base de dados escolhida é relacionada ao **Exame Nacional do Ensino Médio (ENEM)**, uma das principais avaliações educacionais no Brasil, que serve como porta de entrada para diversas instituições de ensino superior. Essa base é rica em informações e oferece possibilidades amplas de análise no contexto educacional.

2.2.1 Descrição da Base ENEM

1. **Fonte e Disponibilidade**:
    - A base do **ENEM 2019** está disponível no Kaggle e inclui dados demográficos, de desempenho e informações educacionais dos participantes.
    - Outra fonte relevante é o Automated Essay Score (AES) Dataset, que se foca na pontuação automática de redações.
2. **Estrutura da Prova**:

- O ENEM é composto por uma redação (não incluída em nossa análise) e uma parte objetiva com 180 questões de múltipla escolha, divididas em quatro áreas:
  - Ciências Humanas
  - Linguagens e Códigos
  - Ciências da Natureza
  - Matemática
- A base utilizada contém apenas informações relacionadas às questões objetivas e suas alternativas, bem como os resultados dos participantes.

Características Relevantes da Base

- **Informações sobre os alunos**:
  Dados demográficos, socioeconômicos e de desempenho individual.
- **Questões da prova**:
  Textos das questões objetivas, alternativas, temas e categorias (ex.: matemática, ciências, etc.).
- **Desempenho geral**:
  Notas em cada área do conhecimento e a pontuação final do aluno.

Motivação para Escolha

- **Contexto Nacional**: O ENEM é amplamente utilizado no Brasil, tornando os resultados do projeto relevantes e de fácil aplicação prática.
- **Riqueza dos Dados**: A base é rica em atributos, o que possibilita análises profundas e diversas.
- **Facilidade de Integração**: Os dados incluem informações de desempenho e categorias claras, facilitando a aplicação de aprendizado de máquina para identificar padrões e realizar previsões.

3. Sobre o Projeto

3.1 Objetivo do Projeto

O objetivo principal deste projeto é realizar uma análise de agrupamento (clusterização) no contexto da base de dados do **ENEM** e prever a aprovação dos alunos com base na **nota de corte** histórica da faculdade. A análise se concentra em entender o comportamento do desempenho dos alunos a partir das variáveis da base de dados e como essas informações podem ser usadas para prever a aprovação dos alunos, considerando a nota de corte.

3.1 Identificação de Clusters de Desempenho

O primeiro passo será **identificar agrupamentos (clusters)** significativos dentro da base de dados do ENEM. Utilizamos uma técnica de **aprendizado não supervisionado**, o algoritmo **K-means**, para identificar grupos de alunos com características de desempenho semelhantes.

- **Características analisadas**:
    - Desempenho nas questões de cada área (Ciências Humanas, Linguagens, Ciências da Natureza e Matemática).
    - Informações demográficas, como **idade**, **sexo**, **localização** e **situação socioeconômica**.
    - **Rendimento geral**, que envolve o desempenho total nas diferentes áreas da prova.

Essa análise ajudará a entender melhor como diferentes fatores influenciam o desempenho dos alunos e se existem padrões de resultados que podem ser explorados para prever suas chances de aprovação.

3.2 Previsão da Aprovação dos Alunos

Após a identificação dos clusters e a definição da nota de corte, construímos um **modelo preditivo** para prever a aprovação ou reprovação dos alunos com base nas suas características de desempenho. Utilizamos uma técnica de **classificação supervisionada**, com **árvores de decisão**, para treinar o modelo utilizando dados históricos de alunos.

- **Entrada para o modelo**: Variáveis de desempenho (questões de cada área), dados demográficos e socioeconômicos.

- **Saída do modelo**: Probabilidade de o aluno ser aprovado ou reprovado, com base na nota de corte definida.
- **Validação**: A precisão do modelo será avaliada utilizando **dados de teste**, que incluirão informações de alunos que não foram parte do treinamento inicial. Isso permitirá testar a capacidade do modelo de generalizar as previsões para novos dados.

3.3 Impacto Esperado

- **Identificação de Padrões de Desempenho:** A análise de clusters permite categorizar alunos em grupos com características de desempenho semelhantes, o que pode auxiliar instituições de ensino a:
    - **Criar programas personalizados de apoio:** Alunos em clusters com desempenho abaixo da média podem receber suporte específico, como reforço em áreas com maior dificuldade.
    - **Desenvolver políticas inclusivas:** Compreender os fatores demográficos e socioeconômicos associados ao desempenho pode ajudar na implementação de políticas que promovam a equidade no acesso ao ensino superior.
- **Otimização de Recursos Educacionais:** Com um modelo preditivo bem calibrado, as instituições podem antecipar resultados e investir de maneira estratégica em:
    - **Programas preparatórios para o ENEM:** Baseados nos fatores mais influentes para aprovação.
    - **Apoio direcionado:** Alocação de recursos, como bolsas de estudo ou orientação pedagógica, para alunos com maior potencial de alcançar a nota de corte.
- **Uso em Processos Seletivos:** O modelo preditivo pode ser integrado a processos de triagem e seleção, especialmente em casos de análise prévia de candidatos ao ingresso em faculdades:
    - **Auxílio no planejamento acadêmico:** Prevendo quais perfis de alunos podem demandar mais suporte, como tutoria ou acompanhamento pedagógico.

- **Identificação de talentos:** Destacar alunos com alto potencial, mesmo que seu desempenho inicial esteja próximo à nota de corte.

Durante o desenvolvimento, percebemos que algumas variáveis, como as notas por área do ENEM e certas características demográficas, estavam relacionadas ao status de aprovação. Decidimos excluir essas variáveis do modelo, pois fugiam do foco da nossa análise e adicionariam maior complexidade ao estudo.

Reconhecemos, no entanto, a importância dessas informações, que refletem desigualdades estruturais no acesso e desempenho educacional. Embora a análise dessas variáveis não tenha sido contemplada neste estudo, o projeto abre espaço para futuras reflexões e iniciativas, como investigar o impacto de intervenções pedagógicas e explorar como os dados podem ser usados para melhorar o desempenho acadêmico em políticas públicas.

4. Seleção e Limpeza da Base de Dados

4.1 Objetivo

O objetivo deste processo foi preparar os dados do ENEM 2019 para análise e uso em tarefas subsequentes, selecionando apenas as colunas relevantes e aplicando limpezas e transformações adequadas. A base de dados original contém muitas variáveis, das quais apenas uma parte é necessária para a análise.

4.2 Etapas do Processo

4.2.1 Carregamento da Base de Dados

- A base foi carregada a partir do arquivo “MICRODADOS_ENEM_2019.csv”, disponível no Kaggle.
- **Configurações de carregamento**:
  - **Encoding**: latin1 (necessário para lidar com caracteres especiais no português).
  - **Delimitador**: ; (separador padrão da base de dados fornecida).

```
df = pd.read_csv(CSV_FILE, encoding=ENCODING, delimiter=DELIMITER)
```

4.2.2 Seleção de Colunas Relevantes

- A base original possui 157 colunas. Para fins de desempenho, apenas aquelas consideradas úteis para a análise foram mantidas.
- **Lista de colunas selecionadas:**
  - Dados demográficos: `'TP_FAIXA_ETARIA'`, `'TP_SEXO'`, `'TP_ESTADO_CIVIL'`, `'TP_COR_RACA'`, `'TP_NACIONALIDADE'`.
  - Informações escolares: `'TP_ESCOLA'`, `'TP_ENSINO'`, `'TP_DEPENDENCIA_ADM_ESC'`, `'TP_LOCALIZACAO_ESC'`, `'TP_SIT_FUNC_ESC'`.
  - Notas: `'NU_NOTA_CN'`, `'NU_NOTA_CH'`, `'NU_NOTA_LC'`, `'NU_NOTA_MT'`, `'NU_NOTA_REDACAO'`.
  - Dados do questionário socioeconômico: `'Q001'` a `'Q025'`.
- Para evitar erros de referência a colunas inexistentes, o código verificou a existência das colunas no DataFrame antes de aplicá-las:

```
colunas_existentes = [col for col in COLUNAS_RELEVANTES if col in df.columns]
df = df[colunas_existentes]
```

4.2.3 Remoção de Valores Ausentes

- Para garantir a integridade dos dados, todas as linhas com valores ausentes foram removidas.

```
df = df.dropna()
```

4.2.4 Adição de Novas Colunas

- **Média das Notas**: Foi calculada a média das notas em Ciências da Natureza, Ciências Humanas, Linguagens, Matemática e Redação.
- **Status de Aprovação**: Foi adicionada uma coluna indicando se o aluno foi "APROVADO" (média acima de 728,77) ou "REPROVADO".

```
df['MEDIA_NOTAS'] = df[['NU_NOTA_CN', 'NU_NOTA_CH', 'NU_NOTA_LC',
'NU_NOTA_MT', 'NU_NOTA_REDACAO']].mean(axis=1)
df['STATUS'] = df['MEDIA_NOTAS'].apply(lambda x: 'APROVADO' if x >
728.77 else 'REPROVADO')
```

4.2.5 Ajustes no Questionário Socioeconômico

- **Substituições em questões com cardinalidade:**
  - Substituiu respostas `'h'` por `'a'` nas questões com hierarquia, indicando o menor valor lógico.
- **Substituições em questões sem cardinalidade:**
  - Substituiu respostas `'f'` por valores mais frequentes (moda).

```
for col in cardinalidade_cols:
    if 'h' in df[col].unique():
        df[col] = df[col].replace('h', 'a')

for col in no_cardinalidade_cols:
    if 'f' in df[col].unique():
        most_frequent = df[col].mode()[0]
        df[col] = df[col].replace('f', most_frequent)
```

4.2.6 Transformação com Label Encoding

- Variáveis categóricas foram transformadas em valores numéricos usando o **LabelEncoder**.
- Isso foi feito em colunas específicas, incluindo as questões do questionário e outras variáveis hierárquicas:

```
label_encode_cols = ['Q001', 'Q002', ..., 'TP_FAIXA_ETARIA',
'TP_ANO_CONCLUIU']
for col in label_encode_cols:
    if col in df.columns:
        df[col] = label_encoder.fit_transform(df[col])
```

4.2.7 Salvamento dos Resultados

- O DataFrame processado foi salvo em dois arquivos:
    - **Arquivo completo (`v4.csv`)**: Contém todas as linhas e colunas processadas.
    - **Amostra reduzida (`v4_mini.csv`)**: Contém uma amostra aleatória de 10 linhas para inspeção.

```
df.to_csv("v4.csv", index=False)
df.sample(n=10).to_csv("v4_mini.csv", index=False)
```

5. Desenvolvimento do Modelo

5.1 Carregamento dos Dados

- **Fonte dos Dados**: A base de dados utilizada foi extraída do arquivo `v4.csv`.
- **Seleção de Colunas**: Foram mantidas apenas as colunas relevantes para o problema, garantindo que o modelo fosse eficiente e evitasse ruído causado por atributos irrelevantes.
- **Tratamento de Valores Ausentes**: Todos os valores nulos foram substituídos por 0, uma abordagem prática para lidar com dados ausentes e evitar erros no processamento subsequente.

5.2 Análise Inicial e Balanceamento de Classes

- **Desbalanceamento de Classes**: Inicialmente, a base de dados apresentava um desbalanceamento severo, com 873.211 exemplos de alunos reprovados contra apenas 8.519 aprovados. Esse desbalanceamento poderia levar o modelo a priorizar a classe majoritária, resultando em baixa eficácia.
- **Técnica de Balanceamento**: Optamos pelo **undersampling**, reduzindo a quantidade de exemplos da classe majoritária (reprovados) para igualar ao número de exemplos da classe minoritária (aprovados). Essa abordagem foi escolhida devido ao tamanho significativo da base de dados, que tornava o oversampling menos prático. Após o balanceamento, cada classe passou a contar com 8.519 exemplos.

- **Impacto do Balanceamento**: O balanceamento garantiu que o modelo fosse treinado com uma distribuição equilibrada de exemplos, reduzindo vieses e melhorando sua capacidade de generalização.

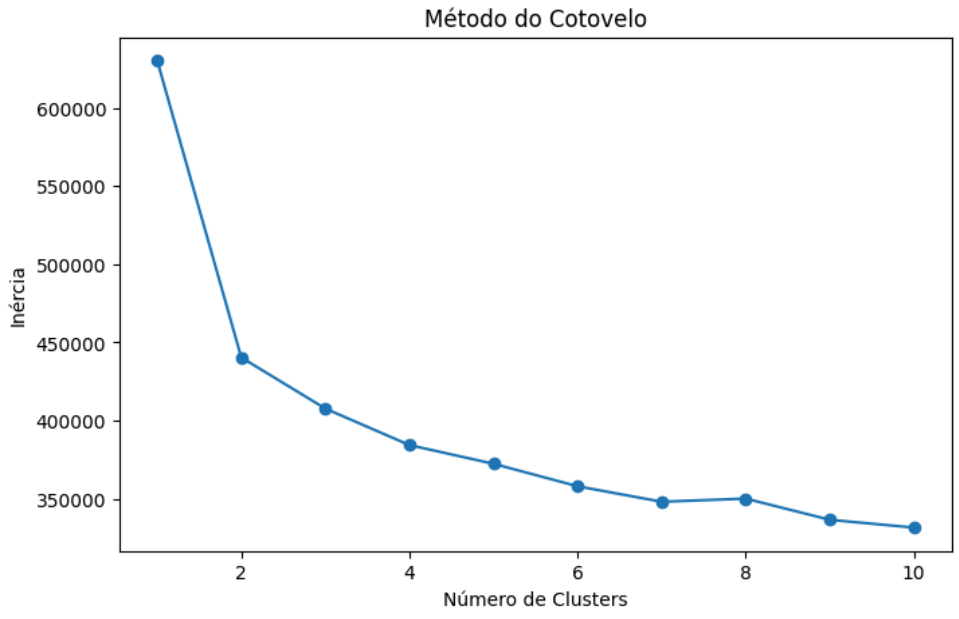
5.3 Normalização

- Após o balanceamento, os dados foram escalados utilizando técnicas de normalização. Essa etapa foi essencial para garantir que todas as colunas numéricas tivessem o mesmo peso nos cálculos de distância, especialmente relevantes para o algoritmo de clusterização e para evitar que características com magnitudes maiores dominassem o modelo.

5.4 Clusterização com K-means

- **Método do Cotovelo (Elbow)**: Utilizamos o método do cotovelo para determinar o melhor número de clusters (k). Os passos incluíram:
    - Treinar o modelo K-means com valores de k variando de 1 a 10.
    - Calcular a soma das distâncias ao quadrado dentro dos clusters (inertia).
    - Gerar o gráfico da inércia em função de k.
    - Identificar o "cotovelo" no gráfico, que no nosso caso indicou k=7, conforme imagem abaixo.

Gráfico 1: Gráfico do Método de Elbow



- **Método da Silhueta**: Outra abordagem para validar a escolha do número de clusters (k) seria a aplicação da métrica da silhueta. Essa métrica avalia a qualidade do agrupamento ao considerar tanto a proximidade dos pontos dentro de um cluster quanto a distância deles para os outros clusters. A pontuação da silhueta varia de -1 a 1:
    - Valores próximos a **1** indicam que os pontos estão bem agrupados.
    - Valores próximos a **0** sugerem que os pontos estão na fronteira entre os clusters.
    - Valores negativos indicam que os pontos foram agrupados de forma inadequada.

  Embora a métrica da silhueta fosse uma ferramenta útil para confirmar a escolha de k, enfrentamos limitações de capacidade de processamento que impediram sua aplicação eficaz. Por isso, seguimos com o valor de k=7, determinado pelo método do cotovelo (Elbow).

- **Resultado**: Os clusters gerados foram adicionados como uma nova coluna no dataset para auxiliar na análise subsequente e no treinamento do modelo.

5.5 Análise dos Clusters

Em busca de entender melhor o perfil dos alunos agrupados em cada um dos clusters, seguimos com uma análise exploratória dos dados para avaliar e definir perfis para eles, o que nos gerou os gráficos a seguir.

Gráfico 2: Distribuição Racial por Cluster

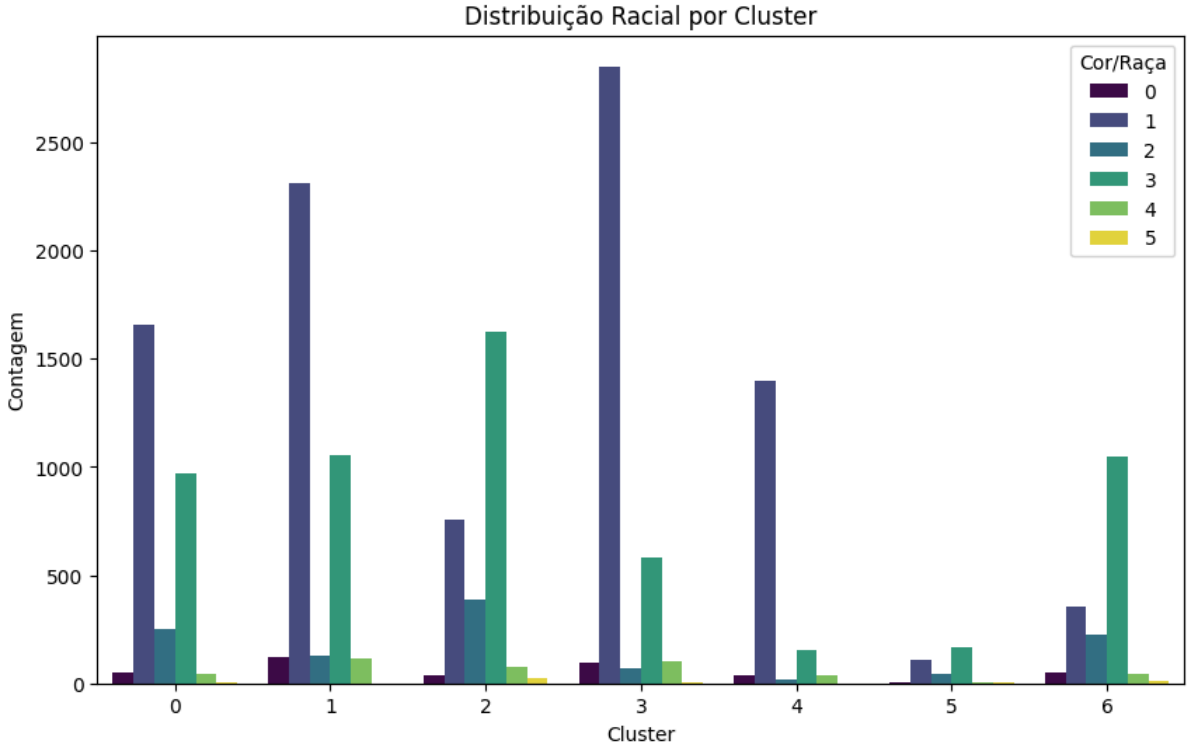


Gráfico 3: Prevalência de Tipo Escolar por Cluster

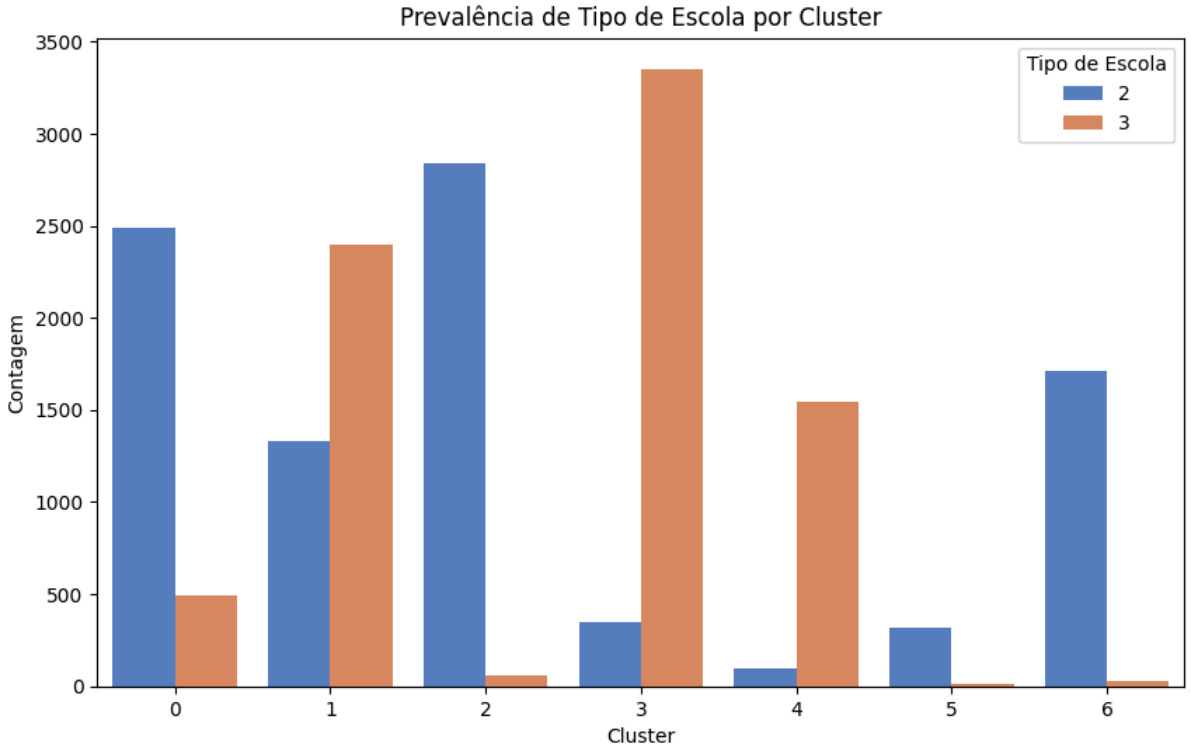


Gráfico 4: Dependência Administrativa das Escolas por Cluster

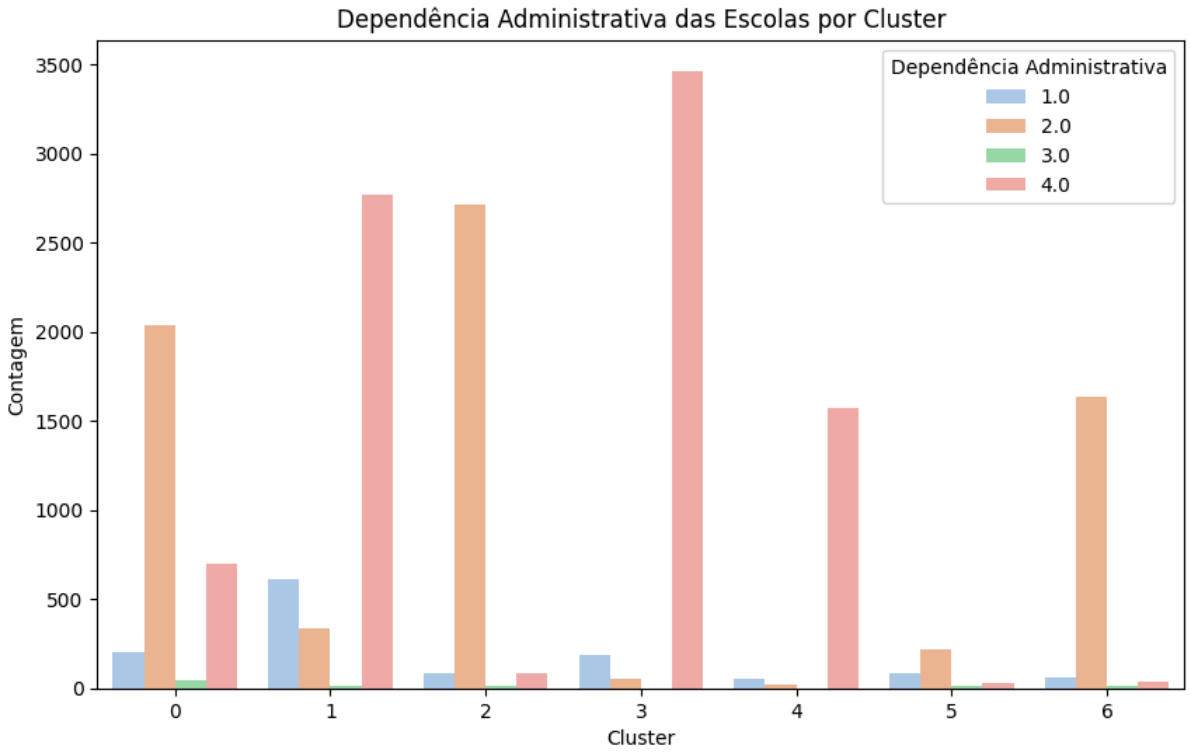


Gráfico 5: Distribuição de Renda Familiar por Cluster

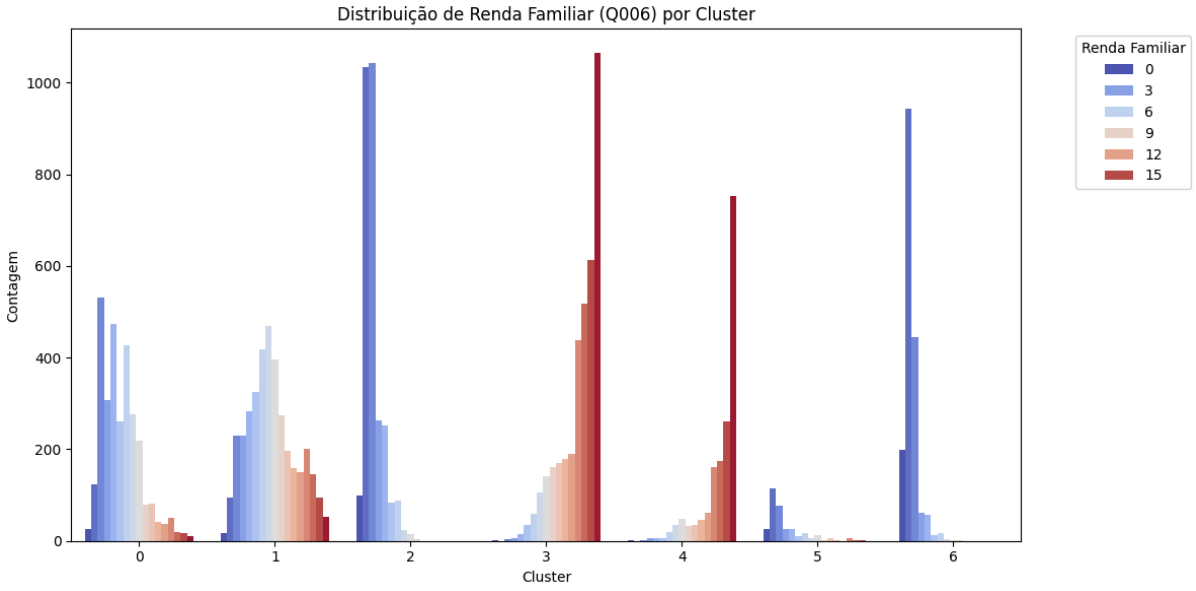


Gráfico 6: Ocupação do Pai por Cluster

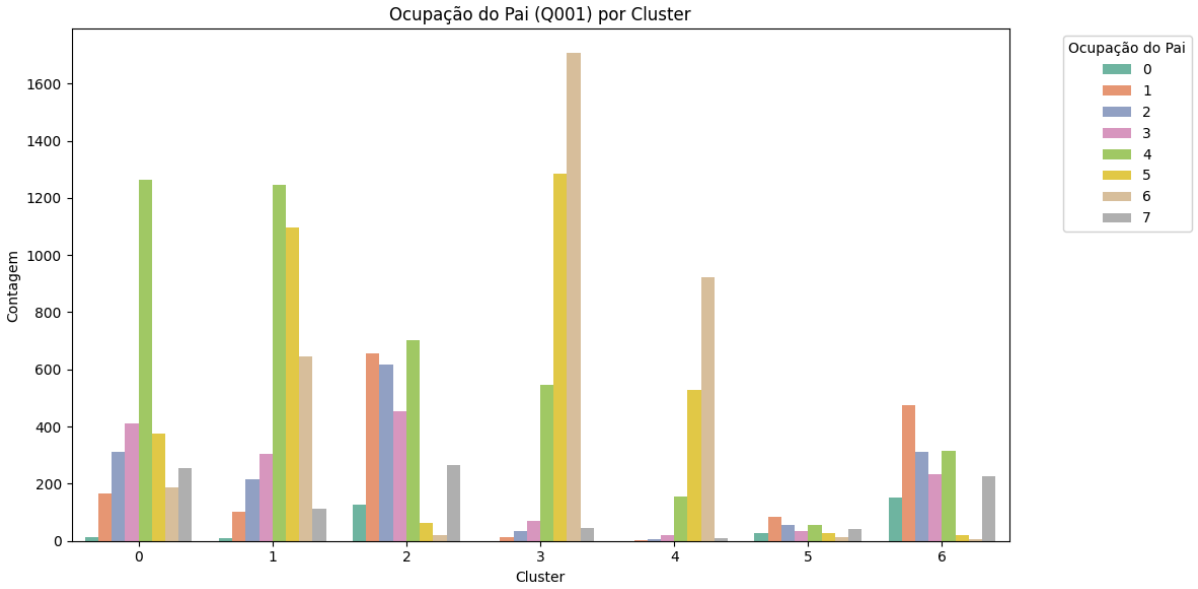
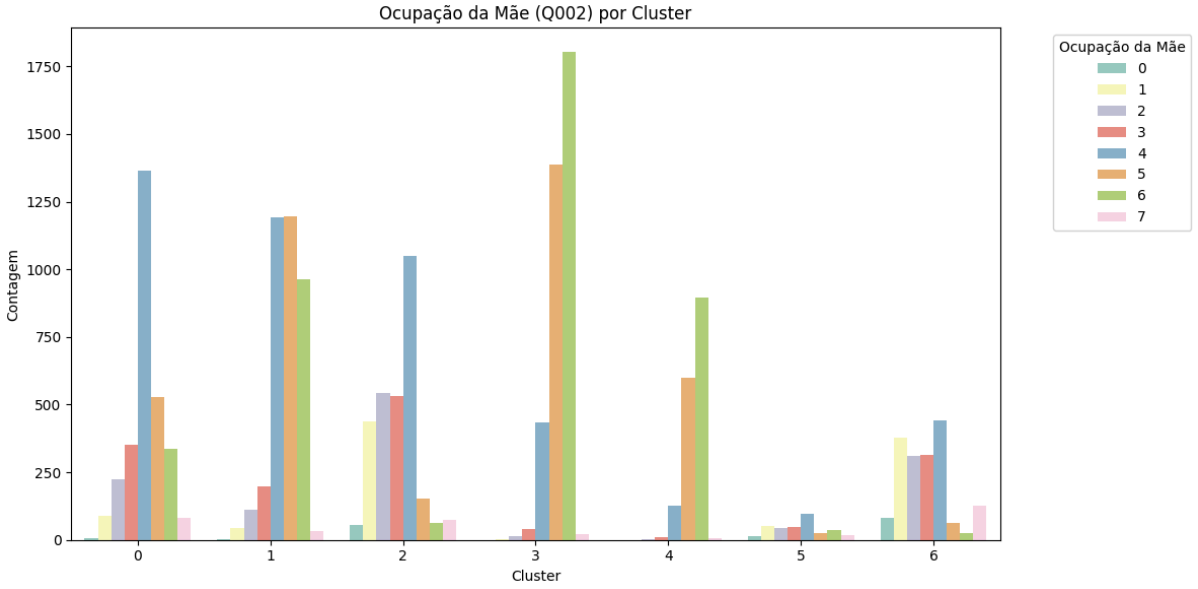


Gráfico 7: Ocupação da Mãe por Cluster:



As análises geradas nos permitiram destacar algumas características para os clusters, conforme abaixo:

- **Cluster 0:** Majoritariamente branco, de escola pública estadual, com pais e mães que não completaram o Ensino Médio.
- **Cluster 1:** Majoritariamente branco, de escola privada, com pais e mães que completaram o Ensino Médio.
- **Cluster 2:** Majoritariamente pardo, de escola pública estadual, com renda familiar abaixo de R$ 3.960,00, com pais e mães que completaram o ensino médio.
- **Cluster 3:** Majoritariamente branco, de escola privada, com renda familiar entre R$ 19.800,01 e R$ 26.400,00, com pais e mães que possuem Graduação e/ou Pós Graduação completa.

- **Cluster 4:** Majoritariamente branco, de escola privada, com renda familiar entre R$ 19.800,01 e R$ 26.400,00, com pais e mães que possuem Graduação Completa.
- **Cluster 5:** Majoritariamente pardo, de escola privada, com renda familiar entre R$ 3.300,01 e R$ 3.960,00, com pais e mães que não completaram o ensino médio.
- **Cluster 6:** Majoritariamente pardo, de escola pública estadual, com renda familiar abaixo de R$ 3.960,00.

5.6 Separação do Conjunto de Dados

- **Limpeza de Colunas**: Antes da separação, foram removidas colunas irrelevantes. Após a primeira tentativa, que resultou em **overfitting**, removemos identificadores e características altamente correlacionadas com o rótulo como medida de mitigação.
- **Divisão dos Dados**: O conjunto de dados foi dividido em treino (70%) e teste (30%), utilizando **estratificação** para manter o balanceamento de classes nos dois subconjuntos.

5.7 Treinamento e Avaliação com Árvore de Decisão

- **Problema Inicial**: Durante o treinamento inicial, o modelo apresentou acurácia perfeita, indicando **overfitting**. Isso ocorreu devido à complexidade intrínseca das árvores de decisão, que podem memorizar dados de treino quando não há limitações no número de divisões ou na profundidade da árvore.
- **Medidas de Mitigação**:
    - Regularizamos o modelo limitando a profundidade da árvore e ajustando o número mínimo de amostras por nó e folha:

```
clf = DecisionTreeClassifier(min_samples_split=10,
min_samples_leaf=5, random_state=42)
```

    - Verificamos a separação correta entre `X_train` e `X_test` para garantir que não houvesse interseção de dados entre os dois conjuntos:

```
print(set(X_train.index).intersection(set(X_test.index)))
```

- Avaliamos a correlação entre as características dos dados (`X_train`) e o rótulo (`y_train`) para identificar possíveis relações diretas que poderiam causar viés:

```
correlations = pd.concat([X_train, y_train],
axis=1).corr()
print(correlations['STATUS'].sort_values(ascending=False))
```

- Visualizamos as regras da árvore para identificar possíveis dependências excessivas em colunas específicas:

```
from sklearn.tree import export_text
tree_rules = export_text(clf,
feature_names=list(X_train.columns))
print(tree_rules)
```

- Removemos colunas altamente correlacionadas com o rótulo, incluindo `cluster`, `STATUS`, e notas específicas (`NU_NOTA_CN`, `NU_NOTA_CH`, etc.).

- **Resultados Após Ajustes**:
  O modelo apresentou métricas mais realistas e consistentes:
  - **Precision**: 84% para aprovados e 85% para reprovados.
  - **Recall**: 87% para aprovados e 83% para reprovados.
  - **F1-Score**: 85% para ambas as classes.
  - **Acurácia Geral**: 85%.

6. Conclusão

Ao longo do desenvolvimento deste modelo, enfrentamos desafios importantes relacionados à escolha das variáveis e à interpretação dos resultados, o que nos permitiu refletir sobre como as características dos dados influenciam diretamente o desempenho e a generalização do modelo.

Um dos passos mais significativos foi a decisão de remover colunas altamente correlacionadas com o status de aprovação ou reprovação, como `'STATUS'`, que

define aprovação ou reprovação na faculdade, `'cluster'`, que define diretamente o cluster em que o aluno está inserido, e as notas do ENEM (`'NU_NOTA_CN'`, `'NU_NOTA_CH'`, `'NU_NOTA_LC'`, `'NU_NOTA_MT'`, `'NU_NOTA_REDACAO'`, `'MEDIA_NOTAS'`). Embora essas variáveis fossem preditoras diretas ou indiretamente relacionadas ao rótulo, mantê-las no modelo levaria ao risco de superestimação da performance (overfitting), conforme ocorreu na nossa primeira tentativa. Isso porque o modelo pode basear suas decisões exclusivamente em características específicas que, embora úteis no conjunto de treinamento, não necessariamente refletem padrões generalizáveis em conjuntos não vistos.

Essa decisão de exclusão foi essencial para garantir que o modelo aprendesse padrões mais amplos e relevantes, considerando interações complexas entre as variáveis restantes. O objetivo final era desenvolver um classificador que pudesse ser aplicado a novos dados com maior confiança, sem depender de informações que, na prática, já carregam implicitamente o rótulo.

Outra etapa interessante foi a análise dos clusters gerados pelo K-means. Apesar de termos utilizado os clusters apenas como uma etapa exploratória no pipeline, eles revelaram perfis distintos de alunos. A identificação desses perfis pode fornecer insights valiosos para futuras ações institucionais, como políticas de suporte acadêmico ou estratégias de intervenção. Por exemplo, clusters com médias de notas mais baixas poderiam representar alunos com dificuldades consistentes em determinadas áreas do ENEM, sugerindo oportunidades para reforço específico. Clusters mais próximos do limiar de aprovação poderiam indicar alunos que se beneficiariam de apoio direcionado para superar a barreira da nota de corte.

Em resumo, nosso modelo final apresentou um desempenho consistente, com métricas como precisão, recall e f1-score equilibrados entre as classes, demonstrando que ele consegue generalizar bem ao prever aprovação ou reprovação. A análise dos dados e os ajustes realizados ao longo do processo, como a exclusão de variáveis problemáticas e a normalização, foram fundamentais para alcançar esses resultados.

O aprendizado gerado por este projeto não se limita apenas ao modelo em si, mas também à compreensão de como decisões no pré-processamento e na escolha de

variáveis influenciam diretamente o sucesso de um sistema de machine learning. Com essas bases, estamos bem posicionados para expandir ou adaptar esse modelo para outros contextos relacionados ao desempenho acadêmico ou mesmo para questões mais amplas no âmbito educacional.