Análise de Dados para o Trocatine

**Empresa:** Trocatine

**Integrantes:**

Gabriela Machado Ribeiro (N°7) – 2°H Tech

Sarah de Moraes Ferreira Batista (N°17) – 2°H Tech

Sumário

[Introdução 2](#_Toc181770215)

[Objetivo do aplicativo 3](#_Toc181770216)

[Objetivo da Análise Exploratória de Dados 3](#_Toc181770217)

[Objetivo da Análise de Dados (para Aprendizado de Máquina) 3](#_Toc181770218)

[Levantamento dos dados na análise exploratória 4](#_Toc181770219)

[Busca dos dados 4](#_Toc181770220)

[Descrição da base de dados de trabalho 4](#_Toc181770221)

[Faixa Etária 4](#_Toc181770222)

[Flag Filho 5](#_Toc181770223)

[Faixa Etária Filho 6](#_Toc181770224)

[Cidade 6](#_Toc181770225)

[Compra Roupa Usada 7](#_Toc181770226)

[Dificuldade Compra 7](#_Toc181770227)

[Dificuldade Infantil 8](#_Toc181770228)

[Faixa Salarial 8](#_Toc181770229)

[Forma Trocou 9](#_Toc181770230)

[Gênero 9](#_Toc181770231)

[Objetivo Compra 10](#_Toc181770232)

[Reciclagem Tecido 10](#_Toc181770233)

[Segurança Aplicativo 11](#_Toc181770234)

[Interesse 11](#_Toc181770235)

[Visualização dos Dados Sujos 12](#_Toc181770236)

[Limpeza dos dados 13](#_Toc181770237)

[Visualização dos Dados (após serem limpos) 14](#_Toc181770238)

[Condicionamento para alimentar o modelo de ML 14](#_Toc181770239)

[Condicionamento inicial 15](#_Toc181770240)

[Definição dos objetivos e das classes 15](#_Toc181770241)

[Definição dos modelos mais adequados para analisar os dados 15](#_Toc181770242)

[Descrição dos modelos selecionados 16](#_Toc181770243)

[Modelo 1: Naive Bayes (GaussianNB) 16](#_Toc181770244)

[Modelo 2: Tree (DecisionTreeClassifier) 16](#_Toc181770245)

[Modelo 3: KNN (KNeighborsClassifier) 16](#_Toc181770246)

[Aplicação dos modelos selecionados 16](#_Toc181770247)

[Análise dos Resultados 17](#_Toc181770248)

[Modelo 1: Naive Bayes 17](#_Toc181770249)

[Modelo 2: Tree 18](#_Toc181770250)

[Modelo 3: KNN 19](#_Toc181770251)

[Comparação Entre Modelos: 20](#_Toc181770252)

[Ajustes Necessários 20](#_Toc181770253)

[Identificação de Problemas: 20](#_Toc181770254)

[Mudanças na Base de Dados: 20](#_Toc181770255)

[Ajustes nos Modelos: 22](#_Toc181770256)

[Impacto das Modificações 22](#_Toc181770257)

[Modelo 1: [Nome do Modelo] 22](#_Toc181770258)

[Modelo 2: [Nome do Modelo] 22](#_Toc181770259)

[Modelo n [Nome do Modelo] 22](#_Toc181770260)

[Comparação Entre Modelos: 22](#_Toc181770261)

[Conclusão 22](#_Toc181770262)

# Introdução

O Trocatine é uma plataforma que visa oferecer um ambiente seguro e acessível para a compra, venda e troca de artigos infantis, direcionada especialmente a responsáveis por crianças. Com o objetivo de atender às necessidades dessas famílias, que frequentemente precisam substituir itens devido ao desenvolvimento constante das crianças, o Trocatine propõe uma alternativa sustentável ao descarte de produtos ainda em bom estado. Apresentando aos responsáveis opções alternativas para lidar com esses itens, para evitar que sejam levados a descartar objetos no lixo comum e a adquirir novos produtos a altos custos. Dessa forma, a plataforma apoia os responsáveis no consumo consciente e contribui para a diminuição de resíduos têxteis.

## Objetivo do aplicativo

O objetivo do aplicativo é auxiliar a introduzir a sustentabilidade na vida de famílias e incentivar a troca de itens infantis entre as pessoas que realmente precisam, desta forma evitando o descarte indevido desses itens e incentivando a economia circular. Além de disponibilizar os produtos por um preço mais acessível já que é algo essencial para as famílias as quais o nosso aplicativo atende, pela perspectiva dos preços exorbitantes os quais são ofertados os objetos no mercado infantil.

# Objetivo da Análise Exploratória de Dados

# A Análise Exploratória de Dados para o projeto Trocatine tem como objetivo preparar e otimizar um conjunto de dados coletados por meio de formulários, a fim de gerar novas ideias que sejam valiosas e para aprimorar os modelos de aprendizado de máquina (ML) usados para identificar potenciais clientes do aplicativo. O Trocatine é uma plataforma que visa facilitar a compra, venda e troca de artigos infantis, oferecendo uma solução sustentável e prática para o descarte de itens que as famílias não utilizam mais.

# O conjunto de dados contém informações detalhadas sobre os perfis e comportamentos de consumo das famílias que buscam trocar ou adquirir produtos infantis de segunda mão. A análise e a limpeza desses dados são etapas cruciais para garantir que o aplicativo atenda de forma eficaz às necessidades de seus usuários, proporcionando uma experiência digital que se ajuste ao perfil específico de quem busca opções mais acessíveis e ecológicas.

# Além disso, os dados coletados permitirão o desenvolvimento de modelos preditivos mais precisos, ajudando o Trocatine a identificar com maior eficácia as pessoas com maior probabilidade de se tornarem clientes da plataforma. Com isso, o aplicativo poderá oferecer uma experiência personalizada, promovendo o consumo consciente e facilitando o acesso a produtos infantis reutilizáveis, alinhados às necessidades e preferências dos usuários.

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

A partir da base de dados estruturada durante a fase de Análise Exploratória de Dados, a aplicação de inteligência artificial terá como objetivo principal a identificação de indivíduos com o perfil próximo as características de um cliente em potencial para o aplicativo. Para atingir esse objetivo, serão realizados experimentos com diferentes modelos, que foram apresentados e discutidos ao longo das aulas. A escolha desses modelos visa comparar o desempenho de diferentes abordagens de classificação, buscando a mais adequada para identificar os potenciais clientes. Os modelos a serem testados incluem:

* KNeighborsClassifier (KNN)
* GaussianNB (Naive Bayes)
* DecisionTreeClassifier (Decision Tree ou Árvore de Decisão)

# Levantamento dos dados na análise exploratória

Os processos relacionados a base de dados podem ser observados no notebook chamado balanceamento\_base.ipynb dentro do link do github.

## Busca dos dados

A coleta de dados realizada pela equipe foi efetuada por meio de um formulário online, projetado especificamente para atender ao objetivo do modelo de previsão, que é identificar potenciais clientes para o aplicativo Trocatine. O questionário foi elaborado com o intuito de compreender o perfil, preferências e características dos indivíduos interessados na proposta do aplicativo.

A pesquisa foi conduzida ao longo de um período de **68 dias**, durante os quais foram obtidas **232 respostas**. Cada formulário levou, em média, **8 minutos e 3 segundos** para ser preenchido, com **19 perguntas** abordando diversos aspectos sobre o público-alvo. Essas informações são essenciais para a construção do modelo de aprendizado de máquina, pois fornecem dados valiosos sobre as necessidades e comportamentos dos usuários, permitindo a identificação dos indivíduos mais propensos a se tornar clientes do Trocatine.

A importância desses dados está no fato de que eles são diretamente relacionados à definição de estratégias eficazes para atrair e engajar os usuários. Ao entender melhor o perfil do público, será possível ajustar as funcionalidades e a abordagem do aplicativo para atender de forma mais precisa às necessidades dos usuários, proporcionando a melhor experiência possivel.

# Descrição da base de dados de trabalho

Priorizando um entendimento detalhado a respeito da base de dados, as colunas utilizadas vão ser demonstradas como seções que apresentarão: tipo dos dados, o que ele é, valores-limite e parâmetros estatísticos. Os parâmetros estatísticos foram encontrados a partir da função **describe** utilizando a biblioteca **pandas**, como todas as colunas presentes na base de dados são colunas categóricas os parâmetros estatísticos eles seriam:

* **count**: o número de entradas não nulas.
* **unique**: o número de valores únicos.
* **top**: o valor mais frequente.
* **freq**: a frequência do valor mais comum.

As colunas apresentadas serão apenas as utilizadas após a limpeza por conta da quantidade exacerbada de colunas que havia anteriormente, ademais ao final será apresentado como está se comportando os dados dessas colunas da forma a qual eles foram coletados.

### Faixa Etária

**Tipo**: String

**O que ele é:** Indica a idade da pessoa que está respondendo a pesquisa.

**Valores Limites**:

Menos de 18 anos

Entre 18 e 24 anos

Entre 25 e 34 anos

Entre 35 e 44 anos

Entre 45 e 54 anos

Entre 55 e 59 anos

60+

**Parâmetros Estatísticos:**

Texto

Descrição gerada automaticamente

### Flag Filho

**Tipo**: Boolean

**O que ele é**: Uma indicação se o indivíduo tem filhos ou não.

**Valores Limites**:

Sim – 0

Não – 1

**Parâmetros Estatísticos**:

Tabela

Descrição gerada automaticamente com confiança baixa

### Faixa Etária Filho

**Tipo**: String

**O que ele é**: Se ele tiver filhos, uma indicação da idade do filho dele.

**Valores Limites**:

Menos de 8 anos

Entre 9 e 14 anos

Entre 15 e 17 anos

Entre 18 e 24 anos

Entre 25 e 34 anos

Entre 35 e 44 anos

Entre 45 e 54 anos

Entre 55 e 59 anos

60+

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Cidade

**Tipo**: String

**O que ele é**: A cidade que a pessoa que está respondendo mora.

**Valores Limites** :

Poderia ser escrito de forma livre.

**Parâmetros Estatísticos**:

Uma imagem contendo Texto

Descrição gerada automaticamente

### Compra Roupa Usada

**Tipo**: Boolean

**O que ele é**: Verificação de aderência a compra de roupas usadas, logo se o indivíduo compraria roupas usadas.

**Valores Limites**:

Sim – 1

Não – 0

**Parâmetros Estatísticos**:

Uma imagem contendo Texto

Descrição gerada automaticamente

### Dificuldade Compra

**Tipo**: String

**O que ele é**: Dificuldades encontradas ao comprar suas próprias roupas baseado nas opções disponíveis.

**Valores Limites**:

Preço

Crescimento

Qualidade

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Dificuldade Infantil

**Tipo**: String

**O que ele é**: Dificuldades encontradas ao comprar roupas infantis baseado nas opções disponíveis.

**Valores Limites**:

Preço

Crescimento

Qualidade

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Faixa Salarial

**Tipo**: String

**O que ele é**: A faixa salarial que se encaixa a renda do individuo.

**Valores Limites**:

Classe A: superior a R$ 22 mil

Classe B: entre R$7,1 mil e R$ 22 mil

Classe C: entre R$ 2,9 mil e R$ 7,1 mil

Classe D/E: até R$ 2,9 mil

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Forma Trocou

**Tipo**: String

**O que ele é**: Se a pessoa já obteve alguma experiencia com qualquer forma de troca de itens.

**Valores Limites**:

Trocou

Doou

Recebeu

**Parâmetros Estatísticos**:

Uma imagem contendo Texto

Descrição gerada automaticamente

### Gênero

**Tipo**: String

**O que ele é**: O gênero que a pessoa se identifica.

**Valores Limites**:

Feminino

Masculino

Prefiro não me identificar

**Parâmetros Estatísticos**:

Uma imagem contendo Texto

Descrição gerada automaticamente

### Objetivo Compra

**Tipo**: String

**O que ele é**: Qual seria o objetivo principal da pessoa ao realizar uma compra online.

**Valores Limites**:

Chegue mais rápido

Comprar mais barato

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Reciclagem Tecido

**Tipo**: Boolean

**O que ele é**: Se o individuo conhece locais de coleta de tecido.

**Valores Limites**:

Sim – 1

Não – 0

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Segurança Aplicativo

**Tipo**: Boolean

**O que ele é**: Se a pessoa se sente segura ao realizar compras online.

**Valores Limites**:

Sim – 1

Não – 0

**Parâmetros Estatísticos**:

Texto

Descrição gerada automaticamente

### Interesse

**Tipo**: Boolean

**O que ele é**: Se o objetivo do aplicativo seria algo que interessa a pessoa.

**Valores Limites**:

Sim – 1

Não – 0

**Parâmetros Estatísticos**:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

### Visualização dos Dados Sujos

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

# Limpeza dos dados

Para a preparação dos dados, foi analisado como se comportavam os valores de cada coluna

e com base nesta visão geral foram realizadas as ações seguintes. Durante o procedimento, foi excluído as colunas de: ID, Nome, Hora da última modificação, Email, Hora de início, Hora de conclusão. Além dessas foram excluídas outras 5 colunas que eram de escrita livre, os motivos da remoção foram: ID’s duplicados, nulos em quantidade massiva, descritivos que não agregavam ao modelo. Após a exclusão destas colunas, foi excluída também uma linha a qual os dados claramente não foram preenchidos da forma correta.

Após alterar o nome das colunas, foi encontrado inconformidades em 4 colunas: Dificuldade infantil, Dificuldade compra, Forma Trocou, Cidade. As 3 primeiras colunas estavam com informações duplicadas por conta da ordem as quais elas apareciam, logo foi criado uma ordem padrão para essas colunas, as quais seriam:

Dificuldade Infantil e Dificuldade Compra:

* Preco;
* Qualidade;
* Crescimento;
* Preco;Qualidade;
* Preco;Crescimento;
* Qualidade;Crescimento;
* Preco;Qualidade;Crescimento;

Forma Trocou:

* Doou;
* Trocou;
* Recebeu;
* Doou;Trocou;
* Doou;Recebeu;
* Trocou;Recebeu;
* Doou;Trocou;Recebeu;

A respeito da coluna cidade, foi criada uma função para consertar o regex já que os valores se repetiam com pequenas diferenças, por exemplo um espaço, e ele foi criado de acordo com as inconformidades presentes no dataset.

### Visualização dos Dados (após serem limpos)

Interface gráfica do usuário, Site

Descrição gerada automaticamente

# Condicionamento para alimentar o modelo de ML

A preparação dos dados ocorrerá antes da execução do modelo de aprendizado de máquina. Além disso, será feita novamente após a execução do modelo, caso seja identificado que ajustes são necessários para um novo processamento.

O condicionamento dos dados será realizado em duas etapas distintas: antes e após os ajustes do modelo de aprendizado de máquina. Na primeira etapa será realizado os testes que levam em consideração apenas os passos de validação cruzada, treino e teste. Após os ajustes, será adicionado mais passos, como de redução de dimensionalidade e padronização dos dados. Cada uma dessas etapas de condicionamento será detalhada em subseções específicas.

## Condicionamento inicial

* Normalização do banco de dados (Ordinal Encoder e Label Encoder)
* A técnica de OverSampling foi aplicada (já que foi apresentado uma pequena amostra de registros, não foi considerado o teste de UnderSampling), e para saber qual é a função mais adequada realizará diversos testes de qual seria a melhor forma de realizar a técnica de gerar novos dados genéricos, desta forma 7 bases geradas de funções distintas são testadas.

ANÁLISE DE DADOS (MÓDOLO)

# Definição dos objetivos e das classes

O objetivo principal deste trabalho é identificar os padrões comportamentais das pessoas interessadas na proposta do aplicativo, a fim de prever quais indivíduos têm maior probabilidade de se interessar pela plataforma. A aplicação dos modelos de aprendizado de máquina visa responder à pergunta: **"Você estaria interessado(a) em um aplicativo com essas funcionalidades?"** O modelo será responsável por prever duas classes de resposta: **"Sim"** e **"Não"** que serão representadas da forma binaria na ordem: 0 e 1. Essas respostas irão auxiliar na compreensão do perfil do público-alvo, proporcionando ideias valiosos sobre a viabilidade e aceitação do aplicativo, e ajudando na definição de estratégias de marketing e aprimoramento do produto.

# Definição dos modelos mais adequados para analisar os dados

Os dados fornecidos para este trabalho consistem exclusivamente em dados quantitativos. Ou seja, todas as respostas no formulário foram baseadas em escolhas pré-definidas, sem espaço para respostas abertas. Isso facilita as análises estatísticas e a aplicação de modelos preditivos, pois as respostas são numéricas e padronizadas, permitindo que os dados sejam processados de forma eficiente para prever os interesses dos potenciais usuários do aplicativo.

Considerando a natureza dos dados e os objetivos do projeto, foram avaliados tanto modelos simples quanto modelos mais complexos.

* **Modelos Simples**: São mais fáceis de implementar e interpretar. Modelos como **Naive Bayes** e **Árvores de Decisão** são exemplos de algoritmos que, mesmo sendo simples, podem gerar resultados rápidos e eficientes para dados com estrutura mais clara e simples.
* **Modelos Mais Complexos**: Embora possam captar interações mais complexas entre as variáveis, os modelos mais sofisticados, como o **K-Nearest Neighbors (KNN)**, exigem mais tempo de treinamento e ajustes finos, além de serem mais exigentes em termos computacionais. Esses modelos são adequados quando há a necessidade de identificar padrões não lineares ou interações mais sutis entre as variáveis, mas podem ser mais difíceis de interpretar.

O objetivo da aplicação dos modelos é realizar uma **classificação binária**, ou seja, prever se um indivíduo está ou não interessado no aplicativo, com base nas respostas fornecidas no questionário. As classes a serem previstas são **"Sim"** e **"Não"**.

# Descrição dos modelos selecionados

Descrever cada modelo selecionado em uma subseção a seguir (criar quantas subseções forem necessárias para os modelos selecionados). Escrever os critérios que utilizou para seleção do modelo, considerando aspectos como:

* Precisão e Robustez: capacidade do modelo de classificar corretamente novos dados e lidar com variações nos dados.
* Interpretação e Explicabilidade: facilidade em interpretar os resultados e entender como as previsões são feitas.
* Desempenho Computacional: tempo e recursos necessários para treinar e aplicar o modelo.
* Capacidade de Generalização: habilidade do modelo de manter um bom desempenho em dados não vistos.

## Modelo 1: Naive Bayes (GaussianNB)

## **Precisão e Robustez:** O Naive Bayes é um modelo robusto, especialmente eficaz em cenários com grande número de características categóricas. Ele pressupõe que as variáveis independentes são condicionalmente independentes, o que pode ser uma simplificação excessiva, mas, na prática, ainda oferece resultados satisfatórios. Embora não seja o modelo mais preciso comparado a opções mais complexas, sua capacidade de fornecer resultados consistentes o torna valioso para problemas com dados mais simples.

## **Interpretação e Explicabilidade**: O Naive Bayes é altamente interpretável. A probabilidade de cada classe é calculada com base nas probabilidades das variáveis, permitindo uma fácil explicação dos resultados, o que é útil, especialmente quando os usuários não são especialistas em modelos de aprendizado de máquina.

## **Desempenho Computacional**: O modelo é muito eficiente em termos de tempo de treinamento e inferência. Seu baixo custo computacional faz dele uma excelente escolha em cenários onde a velocidade de processamento e a resposta rápida são cruciais.

## **Capacidade de Generalização**: Embora o Naive Bayes seja eficaz em dados balanceados, ele pode ter dificuldades com dados desbalanceados, caso não haja ajustes para lidar com a distribuição desigual entre as classes (como balanceamento de classes).

## Modelo 2: Decision Tree (DecisionTreeClassifier)

## **Natureza dos Dados**: A Árvore de Decisão pode lidar tanto com dados categóricos quanto contínuos, sem a necessidade de transformações complexas, o que a torna uma escolha prática para dados mistos.

## **Complexidade do Modelo**: Sua complexidade é moderada. Embora seja mais sofisticada que o Naive Bayes, ela ainda é relativamente simples de interpretar. A árvore pode capturar interações entre variáveis e representar decisões complexas em uma estrutura hierárquica clara, o que facilita sua interpretação.

## **Objetivo de Classificação**: A Árvore de Decisão é eficaz tanto para problemas de classificação binária quanto para multi-classe. Ela oferece flexibilidade, e sua estrutura pode ser ajustada por meio de técnicas como a poda para evitar o overfitting e melhorar a generalização.

## Modelo 3: KNN (KNeighborsClassifier)

**Natureza dos Dados**: O KNN pode lidar com dados tanto categóricos quanto contínuos. No entanto, é importante que as variáveis contínuas sejam padronizadas para garantir que as distâncias calculadas entre os pontos de dados sejam significativas e precisas.

**Complexidade do Modelo**: O KNN é considerado mais complexo em termos de computação, especialmente para grandes volumes de dados. O desempenho do modelo depende da escolha do parâmetro k, que define o número de vizinhos mais próximos a serem considerados. Ajustar k corretamente é crucial para a eficácia do modelo, e, em dados grandes, pode tornar o treinamento mais lento e computacionalmente exigente.

**Objetivo de Classificação**: O KNN é adequado para problemas de classificação binária e multi-classe, e tem a vantagem de ser intuitivo. Contudo, em problemas de classes desbalanceadas, o modelo pode ser tendencioso, favorecendo a classe majoritária nas previsões.

## Conclusão

A escolha dos modelos de aprendizado de máquina para o projeto foi baseada na natureza dos dados e nos objetivos específicos de classificação binária (prever as classes "Sim" e "Não"). O **Naive Bayes**, a **Árvore de Decisão** e o **K-Nearest Neighbors (KNN)** foram selecionados por suas características complementares: simplicidade e rapidez (Naive Bayes), interpretabilidade e flexibilidade (Árvore de Decisão), e capacidade de capturar interações complexas (KNN). Cada modelo oferece vantagens distintas, sendo importante testá-los para determinar qual proporciona os melhores resultados no contexto específico deste aplicativo.

# Aplicação dos modelos selecionados

A aplicação do modelo foi estruturada em etapas sequenciais, com o objetivo de otimizar o desempenho do modelo. A primeira etapa consistiu na divisão dos dados em conjuntos de treinamento (80%) e teste (20%). Essa separação foi realizada para oito diferentes conjuntos de dados, sendo o primeiro a base original sem modificações e os outros sete correspondentes às bases geradas previamente.

Após a divisão dos dados, foi realizada uma validação cruzada com 5 divisões (folds), juntamente com a seleção dos melhores hiperparâmetros utilizando a função **GridSearchCV**, aplicada exclusivamente sobre os dados de treinamento. Essa abordagem foi utilizada para identificar quais bases de dados ofereciam os melhores resultados para cada modelo.

Os resultados de cada divisão foram analisados, e os hiperparâmetros foram ajustados com base na métrica **F1-score**, que oferece uma visão equilibrada entre **precision** e **recall**, permitindo avaliar adequadamente o desempenho do modelo em termos de precisão e capacidade de recuperação.

# Análise dos Resultados

Na etapa de teste é feita uma bateria de testes, visando verificar se haveria oscilações nas métricas e seu tempo de execução. Durante a realização, foi possivel observar que as métricas não mudaram significativamente, logo apenas um dos testes será mostrado como exemplo. Enquanto os tempos de execução foram:

Gráfico, Gráfico de barras

Descrição gerada automaticamente

## Modelo 1: Naive Bayes

Métricas de Avaliação:

Gráfico

Descrição gerada automaticamente

*Resultados do f1 score apresentado durante a validação cruzada realizado em cada uma das bases com o modelo de Naive Bayes.*

Calendário

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com o modelo de Naive Bayes que apresentou o melhor desempenho abaixo de 98% com a base que interagiu melhor.*

Gráfico

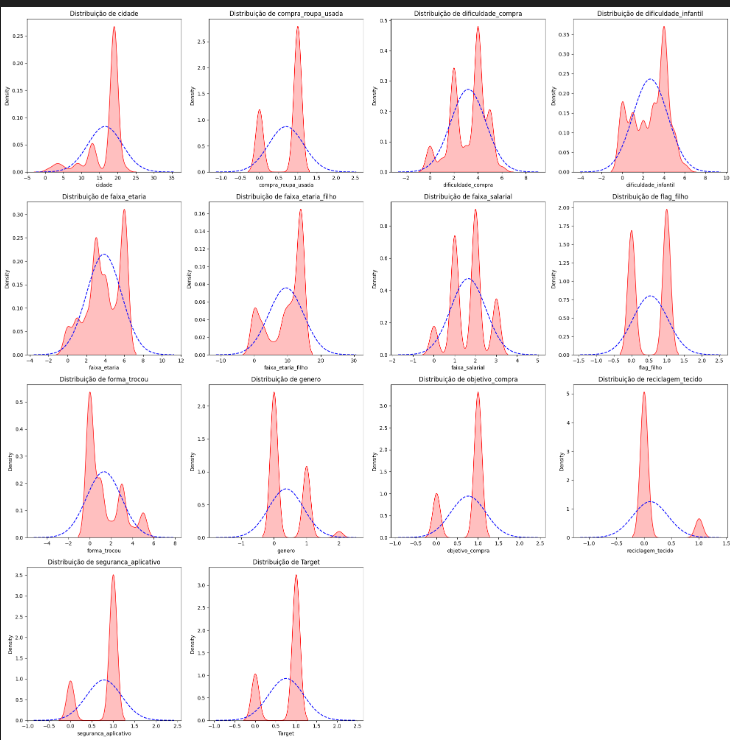
Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

Discussão dos Resultados:

Por conta da falta de equilíbrio entre as classes, foi apresentado uma acurácia maior, todavia os resultados apresentados pela precisão e o recall da classe 0 (yes) são extremamente baixos. Essas informações demonstram a dificuldade do modelo em lidar com a classe minoritária, mas para além disso é possivel observar no gráfico de F1 score, que sua interação com todas as bases, sua dificuldade de se adaptar ao contexto do aplicativo. Isso é detectável por seu desempenho baixo em bases balanceadas, alcançando métricas maiores apenas quando está desbalanceado, pois desta forma quando aponta apenas para a classe majoritária tem a falsa sensação de um bom resultado.

Nos princípios do modelo GaussianNB os dados quantitativos deveriam seguir uma distribuição normal, quando normalizamos a base com o OrdinalEncoder ele passa a tratar as colunas qualitativas da mesma forma. A base utilizada na construção desse modelo é feita somente por colunas qualitativas, logo para verificar se este era o problema foi feito um gráfico com as distribuições dos valores de cada coluna.



*Gráfico com a distribuição dos valores de cada coluna da base de dados que o modelo Naive Bayes utiliza, sendo a linha azul a distribuição esperada e a área vermelha o que é realmente encontrado na base.*

Analisando os resultados apresentados é possivel concluir que as colunas não seguem o padrão esperado, podemos deduzir que este seria um dos motivos para o modelo não conseguir se adaptar da melhor forma.

## Modelo 2: Decision Tree

Métricas de avaliação:

Gráfico, Gráfico de barras

Descrição gerada automaticamente com confiança média

*Resultados do f1 score apresentado durante o cross-validation realizado em cada uma das bases com o modelo de DecisionTreeClassifier.*

Calendário

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com o modelo de DecisionTreeClassifier que apresentou o melhor desempenho abaixo de 98% com a base que interagiu melhor.*

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

Discussão dos Resultados:

O cross-validation do modelo foi seguida uma boa estabilidade do F1 score nas bases, entre 85% e 100%, conquanto tenha apresentado algumas vezes a métrica alta demais de forma propicia a um overfitting. Durante o teste da melhor combinação entre hiperparâmetros e base, foi demonstrado um ótimo equilíbrio entre a precisão e o recall de ambas as classes. Desta forma, resultou em ótimos resultados de acurácia também e nenhuma métricas ficou abaixo de 80%.

## 

## Modelo 3: KNN

Métricas de Avaliação:

Gráfico

Descrição gerada automaticamente com confiança média

*Resultados do f1 score apresentado durante o cross-validation realizado em cada uma das bases com o modelo de KNeighborsClassifier.*

Tela preta com letras brancas

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com o modelo de KNN que apresentou o melhor desempenho (abaixo de 98%) com a base que interagiu.*

Gráfico

Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

Discussão dos Resultados:

O modelo apresentou estar com o F1 score elevado na interação da maioria das bases, apresentando ter se adaptado muito bem em relação a base. Isso é possivel ser comprovado durante o teste também. Entretanto apesar de estar apresentando métricas muito boas, elas estarem tão elevadas não são necessariamente significa algo bom, o mais adequado seria estar menor que 98%.

## Comparação Entre Modelos:

O ponto forte do modelo de Naive Bayes seria exatamente o tempo que sua execução demora, mas não apresenta bons resultados nas suas métricas da classe 0 (yes). O modelo de KNN apresenta métricas extremamente altas, demonstrando ter grande potencial para o contexto do aplicativo, mas não alcançando ainda o seu melhor estado. Em paralelo, o modelo que mais se adequou seria o de árvore, demonstrando estar exatamente entre os dois extremos apontados nos outros dois modelos com um bom balanceamento do F1 score de ambas as classes. Apesar disso a árvore seria a que tem o maior tempo de execução.

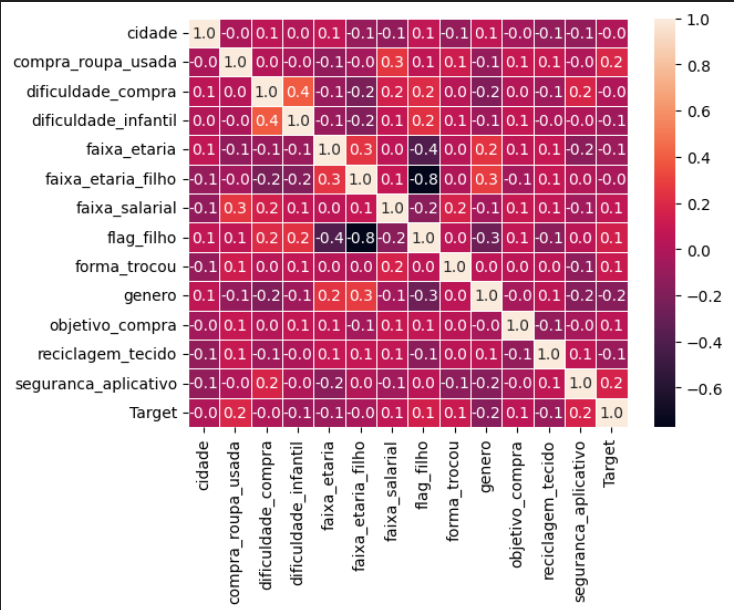
# Ajustes Necessários

## Identificação de Problemas:

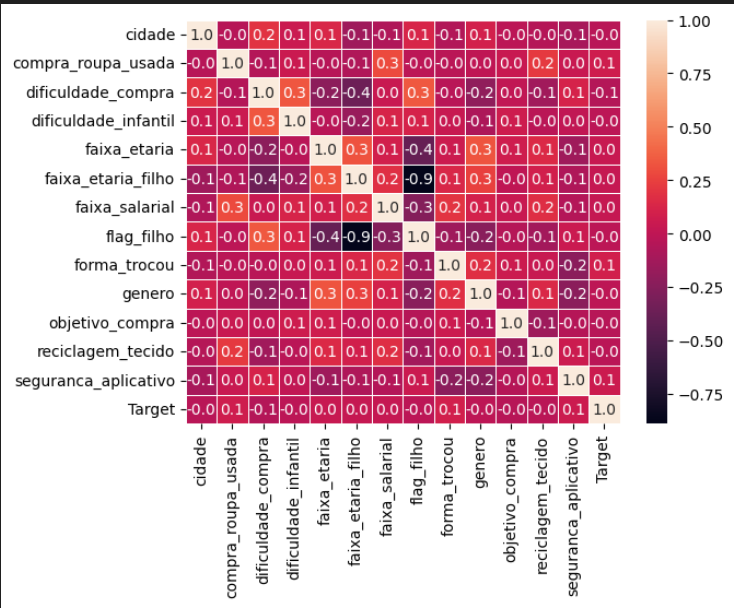
As principais questões a serem solucionadas seriam a dificuldade de classificação da classe 0 (yes) no modelo de Naive Bayes e as métricas extremamente altas do modelo de KNN.

## Mudanças na Base de Dados:

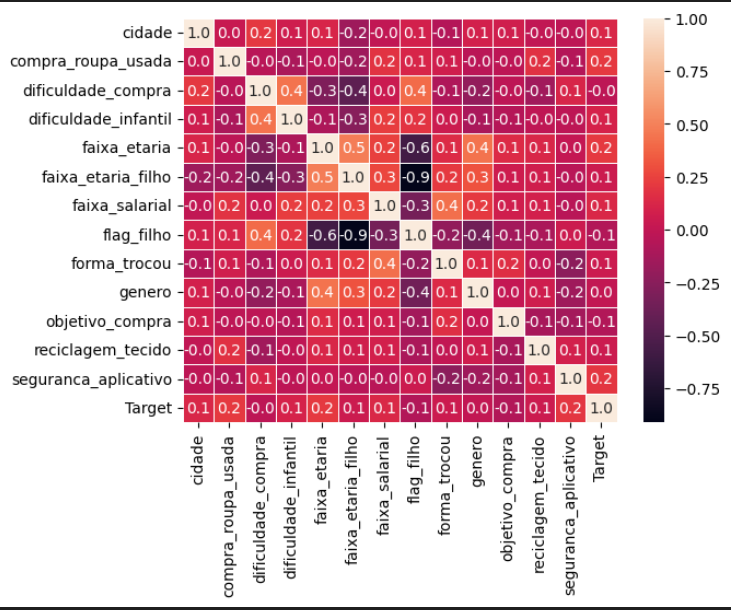
A quantidade e a maneira a qual as informações passadas podem estar impactando o desempenho do modelo do Naive Bayes e do KNN. Para identificar isso foi feito um gráfico de correlação das features de cada uma das bases:



*Base utilizada para a classificação do modelo de Naive Bayes.*



*Base utilizada para a classificação do modelo de Tree.*



*Base utilizada para a classificação do modelo de KNN.*

Foi possivel concluir que as colunas podem ser reduzidas, já que apresenta correlações em sua base. A redução de dimensionalidade auxiliaria também a fazer o modelo generalizar melhor a base e podendo levar a uma execução mais rapida, evitando o overffiting apresentado pelo modelo de KNN e diminuindo o tempo de execução do modelo de tree. Isso será alcançado utilizando o método PCA e o SelectKBest em conjunto.

A dificuldade apresentada pelo Naive Bayes poderia ser solucionada pensando em sua lógica de distribuição normal. Como foi apresentado anteriormente os valores estão fugindo do padrão que o modelo assume, todavia escalonar os dados pode ajudar a manter os valores das variáveis em uma faixa similar, melhorando o ajuste à suposição de normalidade. Desta forma padronizar a base fornecida poderia tornar ele menos influenciável por variáveis em escalas diferentes. E além de poder ajudar também no modelo de KNN que é baseado em distancias e pode ser sensível a escala de dados. Isso será alcançado utilizando o método StandardScaler.

## Ajustes nos Modelos:

A forma a qual vai ser decidido os melhores parâmetros para os métodos, será por meio de uma pipeline que contêm as etapas que passara pela função GridSearchCV, a função também realiza uma validação cruzada com cinco divisões avaliando o F1 score utilizando apenas os dados de treino. Os parâmetros seriam:

Texto

Descrição gerada automaticamente

*Parâmetros que são testados na função GridSearchCV, utilizando uma estrutura de repetição como meio de informar quantas colunas seriam selecionadas pelo SelectKBest e quantas colunas teriam para serem reavaliadas pelo PCA após essa etapa, com o número de colunas variando de 1 a 6 que representa por volta de metade da base.*

Foi testado fazer uma reavaliação dos parâmetros dos modelos e das bases em conjunto com as novas etapas, entretanto o tempo de execução exigido para realizar essa reavaliação tornava inviável realizá-la. Logo para essas melhoras foram mantidos os parâmetros e a base do modelo que teve o maior desempenho abaixo de 98%. Além disso também foi testado ampliar o intervalo de números para o SelectKBest, entretanto não gerava o resultado esperado de combater a presença do overffiting no modelo de KNN, como a redução para no máximo 6 colunas não causaram grandes impactos nos outros modelos foi estabelecido desta forma.

# Impacto das ModificaçõesGráfico, Gráfico de barras Descrição gerada automaticamente

*Média dos F1 scores que cada modelo alcançou com a nova pipeline durante a validação cruzada.*

Gráfico, Gráfico de barras, Gráfico de mapa de árvore

Descrição gerada automaticamente

*Média dos tempos de execução de cada modelo extraída durante uma bateria de 6 vezes de teste.*

## Modelo 1: Naive Bayes (GaussianNB)

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

*As colunas selecionadas após o estágio do PCA e suas principalidades.*

Calendário

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com a pipeline, utilizando o modelo de Naive Bayes que apresentou o melhor desempenho (abaixo de 98%) com a base que interagiu na etapa anterior.*

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

É possivel notar que o modelo conseguiu entender melhor a base de dados fornecida a ele após a sua padronização, desta forma a classe 0 (yes) teve melhores métricas de recall e precisão do que anteriormente, entretanto ainda não alcançou um resultado satisfatório em ambas as métricas o que resulta em um resultado baixo também no F1 score.

Discussão dos Resultados: explicar como as mudanças na base de dados e os ajustes nos modelos ajudaram a melhorar as métricas de avaliação.

## Modelo 2: Decision Tree (DecisionTreeClassifier)

Gráfico

Descrição gerada automaticamente

*As colunas selecionadas após o estágio do PCA e suas principalidades.*

Calendário

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com a pipeline, usando o modelo de decision tree que apresentou o melhor desempenho (abaixo de 98%) com a base que interagiu na etapa anterior.*

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

Após a mudança, é possivel observar uma piora do desempenho do modelo de árvore, apesar de não ser uma grande piora. Isso provavelmente se deve a limitação de colunas até 6, tanto que é possivel observar na seleção de colunas que foram escolhidas a quantidade máxima.

## Modelo 3: KNN (KNeighborsClassifier)

Interface gráfica do usuário, Gráfico

Descrição gerada automaticamente

*As colunas selecionadas após o estágio do PCA e suas principalidades.*

Uma imagem contendo Calendário

Descrição gerada automaticamente

*Resultados das métricas apresentadas durante o teste realizado com a pipeline, o modelo de KNN seria o que apresentou o melhor desempenho (abaixo de 98%) com a base que interagiu na etapa anterior.*

Gráfico

Descrição gerada automaticamente

*Matriz de confusão gerada a partir da predição do modelo de aprendizado.*

O modelo de KNN não demonstrou novamente overffiting, obtendo 94% de F1 score, apesar também selecionar a maior quantidade possivel apresenta não ter dificuldades com a redução de dimensionalidade e conseguir ter um ótimo desempenho apenas com as novas seis colunas escolhidas.

## Comparação Entre Modelos:

O modelo de Naive Bayes aumentou suas métricas na classe 0 (yes) em relação ao que foi apresentado anteriormente, mas não suficientemente para alcançar resultados bons. Enquanto o modelo de arvore diminuiu suas métricas, além de aumentar o seu tempo de execução piorando nos dois quesitos em relação a anteriormente. Por fim foi solucionado o overffiting apresentado no modelo de KNN e houve uma grande melhora em seu tempo de execução, desta forma, em comparação aos outros modelos testados nessa pipeline, foi o modelo que obteve os melhores resultados.

Em uma comparação ampla, os dois melhores modelos testados que obtiveram os melhores resultados foram o modelo de KNN com a pipeline e o modelo de Árvore de Decisão antes da pipeline. Ambos tiveram F1 scores bons, com o modelo de arvore ligeiramente maior, o mesmo cenário é encontrado quando discutido em relação ao tempo de execução. Para o auxílio desta decisão foi realizado um [teste A/B](#_Comparação_Entre_Modelos:) levando em consideração as métricas obtidas durante a validação cruzada realizada e o tempo de execução coletado durante a bateria de testes.

Após a coleta do resultado do teste A/B, foi determinado que o modelo de Árvore de Decisão seria o mais adequado, já que ele tem indícios de ter uma performance significativamente maior e não tem indícios de ter um tempo de execução significativamente mais lento.

# Conclusão

Para finalizar este documento sobre a análise de dados e os modelos aplicados no projeto Trocatine, é importante destacar que o processo de exploração, limpeza e aplicação de modelos de aprendizado de máquina permitiu identificar padrões relevantes e potenciais clientes para a plataforma. A aplicação e comparação entre os modelos Gaussian Naive Bayes, Decision Tree e K-Nearest Neighbors (KNN) revelaram a necessidade de ajustes específicos para lidar com a natureza dos dados e os desafios de classificação da base.

A análise comprovou que a Decision Tree e o KNN foram os modelos mais eficazes para prever o interesse dos usuários no Trocatine, apresentando um bom equilíbrio entre precisão e desempenho. Apesar das dificuldades iniciais com o Naive Bayes, as modificações realizadas foram fundamentais para melhorá-lo, ainda que não o suficiente para superá-lo em comparação com os outros modelos.

O impacto potencial dessas previsões no projeto é significativo, pois permite que o Trocatine personalize estratégias de marketing e desenvolvimento de funcionalidades, melhor alinhando o produto ao perfil e necessidades do público-alvo. Como próximos passos, recomenda-se expandir o conjunto de dados, realizar novas otimizações nos modelos e explorar abordagens mais complexas para maximizar a precisão e generalização das previsões.

**Relatório para RPA de Transferência de Dados entre Bancos de Dados**

**Objetivo**

Este RPA (Automação de Processos Robóticos) foi desenvolvido para transferir informações do banco de dados de **extração** (Banco do Primeiro Ano) para o banco de **destino** (Banco do Segundo Ano). O objetivo principal é garantir que apenas informações que ainda não estejam presentes no banco de destino sejam transferidas, evitando duplicidade de dados. A automação realiza esse processo de forma programática e controlada, garantindo que o conteúdo seja sempre atualizado sem sobrecarregar o banco de destino com registros repetidos.

**Estrutura e Explicação do Código**

Para alcançar o objetivo, o código foi dividido em uma série de funções responsáveis por extrair, transformar e carregar os dados (processo ETL). Essas funções permitem modularidade e facilitam a manutenção do código. Abaixo, seguem as explicações detalhadas das principais funções e do funcionamento do código como um todo.

**Funções Principais**

1. **extraindo\_info\_tabela\_origem**: Esta função conecta-se ao banco de dados do Primeiro Ano, extrai os dados completos da tabela especificada e retorna esses dados em um DataFrame do Pandas. É essencial para obter as informações que serão processadas e transferidas.
2. **colunas\_tabela\_destino**: Responsável por retornar a lista de colunas da tabela de destino no banco de dados do Segundo Ano. Essa lista de colunas é usada nas funções de inserção para garantir que os dados inseridos estejam alinhados com a estrutura da tabela de destino, evitando erros por incompatibilidade.
3. **inserir\_info\_tabela\_destino\_category**: Esta função transfere dados da coluna **tipo\_produto** da tabela **categoria** (do Primeiro Ano) para o campo **name** da tabela **categories** no Segundo Ano. A função verifica se cada valor já existe na tabela de destino para evitar duplicação.
4. **inserir\_info\_tabela\_destino\_tag**: Neste caso, os títulos das colunas na tabela **tag** do Primeiro Ano são usados para preencher o campo **type** na tabela **tags** do Segundo Ano. Os valores individuais nas colunas de **tag** do Primeiro Ano são transferidos para o campo **name** da tabela **tags** no Segundo Ano. Essa função inclui verificações de duplicação para garantir que apenas novos valores sejam inseridos.
5. **transferindo\_info\_usuario**: Responsável por gerenciar o processo completo de transferência de dados de usuários e seus respectivos telefones. Esta função coordena a extração dos dados de usuários e a inserção das informações na tabela de usuários e na tabela de telefones no banco do Segundo Ano.
6. **Funções de Inserção e Validação de Duplicidade**: Cada uma das funções de inserção (como inserir\_info\_tabela\_destino\_category e inserir\_info\_tabela\_destino\_tag) possui validações internas para garantir que os dados não sejam duplicados. A estrutura geral da função verifica se o valor específico já existe na tabela de destino e, caso contrário, realiza a inserção.

**Dependência da Estrutura do Banco do Primeiro Ano**

Este RPA depende totalmente da estrutura do banco de dados do Primeiro e do Segundo ano para funcionar corretamente. As funções de extração e de inserção foram desenvolvidas com base nos nomes e na estrutura atual das tabelas e colunas desse banco. Caso qualquer alteração na estrutura do banco do Primeiro ou do Segundo ano ocorra (como renomeação de tabelas, mudanças nos nomes das colunas ou alteração nos tipos de dados), o código atual deixará de funcionar corretamente.

Essa dependência exige uma manutenção contínua para acompanhar mudanças no banco de origem. Caso o banco de dados de extração seja alterado, será necessário ajustar o código para que o processo ETL do RPA continue funcionando conforme o esperado.

**Conclusão**

O RPA criado automatiza o processo de transferência de dados entre o banco do Primeiro Ano e o banco do Segundo Ano, garantindo que apenas informações novas sejam adicionadas ao banco de destino, eliminando duplicações. Esse processo facilita a atualização de dados e permite maior integridade e consistência no banco de destino, essencial para a operação contínua do sistema.