UNIVERSIDADE FEDERAL DE OURO PRETO
DEPARTAMENTO DE CIÊNCIA DE COMPUTAÇÃO
SISTEMAS DE RECOMENDAÇÃO

**Projeto de Pesquisa**

# ESTRATÉGIAS DE DIVISÃO DE BASE DE DADOS EM TREINO E TESTE

**Alunos:**

Clara Loris de Sales Gomes

Eduardo Matosinhos Florinda

Nicolle Canuto Nunes

Thiago Dornelas Borba

# 1 Introdução

No aprendizado de máquina (ML), a divisão de dados entre treino e teste é uma prática comum utilizada para avaliar a eficácia do modelo de recomendação. O objetivo é dividir o conjunto de dados em dois grupos: um grupo para treinar o modelo e outro grupo para avaliar o desempenho do modelo.

Nos sistemas de recomendação, a divisão de dados é especialmente importante devido à natureza dos dados. Os dados em sistemas de recomendação geralmente apresentam alta dimensionalidade e esparsidade, o que pode levar a um *overfitting* do modelo se não forem tomadas medidas adequadas. Além disso, os sistemas de recomendação geralmente têm como objetivo fornecer recomendações personalizadas para cada usuário, o que requer um modelo que seja capaz de generalizar para novos usuários e itens. Usar uma estratégia apropriada de divisão de dados ajuda a garantir que o modelo generalize bem para novos dados, levando a melhores recomendações. Além disso, ajuda a prevenir problemas como a fuga de dados, onde o modelo pode inadvertidamente aprender padrões que só estão presentes nos dados de teste, levando a resultados imprecisos quando aplicado a novos dados.

Existem várias abordagens para dividir os dados, incluindo a divisão aleatória, a divisão temporal e a validação cruzada. A divisão aleatória é a abordagem mais simples, em que os dados são divididos aleatoriamente em conjuntos de treinamento e teste. A divisão temporal é uma abordagem que leva em conta a ordem temporal dos dados, dividindo os dados mais antigos para treinamento e os mais recentes para teste. A validação cruzada é uma técnica mais avançada que envolve a divisão dos dados em vários conjuntos de treinamento e teste, permitindo uma avaliação mais robusta do modelo.

## 1.1 Justificativa

Existem várias estratégias de divisão de dados que podem ser usadas em um modelo de recomendação. Uma pesquisa pode ajudar a identificar a estratégia mais adequada para um determinado conjunto de dados, levando em consideração fatores como tamanho do conjunto de dados, distribuição de dados, objetivos de modelagem, entre outros.

Dessa forma, é necessário a realização de trabalhos que analisem o impacto de diferentes estra-

tégias de divisão de dados em modelos de recomendação combinado com diferentes algoritmos de filtragem, permitindo uma melhor compreensão das vantagens e desvantagens de cada abordagem.

### 1.2 Objetivos

O objetivo da pesquisa é avaliar e comparar diferentes estratégias de divisão de dados e algoritmos de filtragem aplicados a uma base de dados pequena para oferecer recomendação de ingredientes de pizza. A pesquisa busca identificar a estratégia de divisão de dados mais adequada e o algoritmo de filtragem mais eficaz para o contexto, a fim de melhorar o desempenho do modelo de recomendação.

## 2 Trabalhos relacionados

O artigo Faraway (2016) investiga se a divisão dos dados em conjuntos de treinamento e teste é realmente necessária para obter uma boa precisão de previsão em modelos de aprendizado de máquina. Os autores examinam várias estratégias de treinamento, incluindo treinamento com todos os dados, treinamento com validação cruzada e treinamento com validação cruzada repetida. Eles descobrem que, para alguns conjuntos de dados, o treinamento com todos os dados leva a uma precisão de previsão tão boa quanto ou melhor do que o treinamento com validação cruzada. No entanto, em geral, a validação cruzada ainda é a melhor opção para garantir uma boa precisão de previsão e evitar o sobreajuste. Os autores também destacam a importância de ajustar corretamente os hiperparâmetros do modelo e enfatizam a necessidade de realizar mais pesquisas sobre esse assunto para entender melhor o comportamento dos modelos de aprendizado de máquina em diferentes conjuntos de dados.

Tan et al. (2021) discute as limitações da abordagem atual de dividir aleatoriamente os dados em conjuntos de treinamento e teste para avaliar modelos de aprendizado de máquina. Os autores propõem uma nova técnica de divisão de treinamento/teste baseada em um método de amostragem de densidade, que considera a distribuição dos dados e permite uma divisão mais precisa. Os pontos principais da abordagem são:

- Estimar a densidade dos dados usando um método de estimação de densidade não-

paramétrico.

- Definir uma função de custo que considera a distribuição dos dados, baseada na diferença entre a distribuição do conjunto de treinamento e a distribuição geral dos dados.
- Encontrar a divisão dos dados em conjuntos de treinamento e teste que minimiza a função de custo definida, utilizando um algoritmo de otimização.
- Avaliar o modelo de aprendizado de máquina treinado com a nova divisão de treinamento/teste proposta.

Os resultados dos experimentos realizados pelos autores mostram que essa abordagem pode melhorar significativamente a precisão dos modelos de aprendizado de máquina em comparação com a abordagem atual de divisão aleatória.

No artigo Roshankhah et al. (2021), são exploradas diferentes estratégias de divisão de dados de treinamento e teste para a segmentação e pontuação automatizadas de imagens de ultrassom pulmonar de pacientes com COVID-19. Os autores argumentam que a segmentação automatizada e a pontuação das imagens de ultrassom podem ser úteis na avaliação da gravidade da doença e no monitoramento da resposta ao tratamento. No entanto, para que essas técnicas sejam eficazes, é essencial que o modelo seja treinado e testado em conjuntos de dados independentes e bem equilibrados. Por meio de experimentos, foram avaliadas três estratégias de divisão de dados - a divisão aleatória, a divisão por paciente e a divisão por imagem. Com isso, os autores observam que a estratégia de divisão por paciente produz os resultados mais estáveis e confiáveis nesse contexto, em comparação com as outras duas estratégias. Além disso, eles observam que a precisão do modelo de segmentação e pontuação aumenta com o tamanho do conjunto de treinamento.

Meng et al. (2020) apresenta uma análise comparativa entre seis estratégias de divisão de dados - utilizadas na avaliação de modelos de recomendação - em três conjuntos: treinamento, validação e teste. As estratégias testadas foram: divisão aleatória, divisão temporal, divisão por usuário, divisão por Item, divisão por cluster de usuários e divisão por cluster de itens. Os autores realizaram testes com três bases de dados distintas e concluíram que a escolha da estratégia de divisão de dados pode influenciar significativamente a avaliação do modelo. As estratégias de divisão por *cluster* de usuários e *cluster* de itens apresentaram melhores resultados do que as outras estratégias. Por fim, os autores enfatizam a importância de escolher uma estratégia de divisão de dados

que se adapte bem às características da base de dados e do modelo de recomendação para garantir uma avaliação justa e precisa do desempenho do modelo.

A ideia principal do método proposto em Farias et al. (2020) é dividir os dados em conjuntos de treinamento e teste de forma estratificada e balanceada em relação às classes, utilizando uma medida de similaridade entre as amostras. Para implementar o método, é necessário calcular uma matriz de similaridade entre as amostras, que pode ser calculada de diversas maneiras, como a distância euclidiana ou o coeficiente de correlação de Pearson, por exemplo. Em seguida, a matriz de similaridade é utilizada para agrupar as amostras em subgrupos mais similares. A partir desses subgrupos, é possível construir os conjuntos de treinamento e teste de forma estratificada e balanceada, ou seja, garantindo que cada subgrupo esteja representado em ambos os conjuntos, na mesma proporção. Esse processo é repetido diversas vezes, para gerar diferentes divisões dos dados, e os resultados são combinados para obter uma estimativa mais robusta da performance do modelo. Comparado com outras abordagens de divisão de dados, o método proposto pelo artigo Farias et al. (2020) demonstrou resultados promissores em diversos conjuntos de dados de referência, e pode ser uma alternativa interessante para melhorar a performance de classificadores em problemas de classificação.

# 3 Fundamentação teórica

## 3.1 *Underfitting* e *Overfitting*

*Overfitting* e *underfitting* são problemas comuns que podem ocorrer durante o treinamento de modelos de aprendizado de máquina.

O *overfitting* ocorre quando um modelo se ajusta muito bem aos dados de treinamento, mas não generaliza bem para novos dados. Isso significa que o modelo se torna muito específico para os dados de treinamento e não consegue capturar a variabilidade dos dados do mundo real. Isso pode levar a uma performance pior do modelo em dados de teste ou em situações reais.

Por outro lado, o *underfitting* ocorre quando um modelo não se ajusta adequadamente aos dados de treinamento. Isso significa que o modelo não consegue capturar as relações complexas entre as variáveis do problema e, portanto, não consegue modelar os dados de treinamento com precisão

suficiente. Isso pode levar a uma performance pior do modelo em dados de treinamento e em dados de teste.

Ambos os problemas podem ser resolvidos com técnicas adequadas de ajuste de modelo. Para combater o *overfitting*, pode-se usar técnicas como regularização, *dropout* e aumento de dados, que ajudam a reduzir a complexidade do modelo e a evitar que ele se ajuste demais aos dados de treinamento. Para combater o *underfitting*, pode-se tentar aumentar a complexidade do modelo, aumentar o tamanho do conjunto de treinamento ou ajustar os hiperparâmetros do modelo.

A divisão de dados em conjuntos de treinamento, validação e teste é uma técnica importante para avaliar a capacidade de generalização de um modelo de aprendizado de máquina e também pode ter um grande impacto na ocorrência de *overfitting* e *underfitting*.

Se o conjunto de treinamento for muito pequeno, o modelo pode não conseguir capturar todas as nuances dos dados e ocorrerá *underfitting*. Por outro lado, se o conjunto de treinamento for muito grande e o conjunto de validação e teste forem muito pequenos, o modelo pode se ajustar demais aos dados de treinamento e ocorrerá *overfitting*.

## 3.2 Estratégias de divisão dos dados

### 3.2.1 K-fold

A técnica k-fold é comumente usada para avaliar a eficácia de modelos de aprendizado de máquina e sistemas de recomendação, por meio da divisão do conjunto de dados em k partições de tamanho semelhante. Durante k iterações, uma das k partições é usada para teste e as outras k-1 são usadas para treinamento. Ao final de cada iteração, os resultados são combinados para produzir uma estimativa mais precisa do desempenho do modelo.

Essa técnica é especialmente útil para avaliar a robustez do modelo, uma vez que é testado em k conjuntos de dados diferentes. Além disso, é uma alternativa viável quando o conjunto de dados é pequeno e não é possível fazer a divisão aleatória em treinamento e teste. No entanto, deve-se lembrar que essa técnica pode ser computacionalmente intensiva, tornando-se mais demorada com conjuntos de dados maiores ou modelos mais complexos. Por isso, é importante garantir que os dados sejam embaralhados aleatoriamente antes da aplicação da técnica k-fold para evitar vieses no conjunto de dados.

Em suma, a técnica k-fold é uma técnica de validação cruzada útil para avaliar modelos de aprendizado de máquina e sistemas de recomendação, sendo uma alternativa viável quando a divisão aleatória em treinamento e teste não é possível. No entanto, é preciso considerar a complexidade do modelo e a capacidade de processamento disponível antes de escolher a técnica de validação cruzada adequada.

### 3.2.2 Divisão aleatória

O processo de divisão aleatória envolve a seleção de uma proporção aleatória de observações do conjunto de dados original para serem incluídas no conjunto de treinamento e no conjunto de teste. Geralmente, a divisão é feita de forma que a maioria dos dados sejam alocados para o conjunto de treinamento e uma pequena parte para o conjunto de teste.

Uma vez que a divisão foi feita, o modelo é treinado no conjunto de treinamento e avaliado no conjunto de teste. Isso permite que se tenha uma ideia de quão bem o modelo é capaz de generalizar para novos dados que não foram usados no treinamento.

No entanto, é importante lembrar que a seleção aleatória pode levar a uma variância alta no desempenho do modelo, dependendo da aleatoriedade dos dados. Portanto, é comum usar a validação cruzada, uma técnica mais robusta, em vez de apenas uma única divisão aleatória para avaliar o modelo de forma mais precisa.

## 3.3 Algoritmos de classificação e regressão

### 3.3.1 K-vizinhos mais próximos

K-vizinhos mais próximos (KNN) é um algoritmo de aprendizado de máquina supervisionado usado para classificação e regressão. O objetivo do KNN é classificar uma nova amostra com base nas amostras mais próximas em um conjunto de treinamento.

O KNN opera usando a distância Euclidiana entre pontos. Quando uma nova amostra é apresentada, o algoritmo calcula a distância Euclidiana entre a nova amostra e todas as outras amostras no conjunto de treinamento. Em seguida, o algoritmo seleciona os K pontos mais próximos da nova amostra, onde K é um número inteiro positivo definido pelo usuário.

Para a classificação, o algoritmo faz uma contagem de votos dos rótulos das amostras selecionadas. O rótulo mais comum entre as K amostras mais próximas é atribuído à nova amostra. No caso de regressão, o algoritmo retorna a média dos valores dos K vizinhos mais próximos.

O KNN é um algoritmo simples, fácil de implementar e interpretar. No entanto, ele pode ser sensível a *outliers*, já que as amostras mais próximas podem ser dominadas por dados incomuns. O desempenho do KNN também pode ser afetado pelo número de vizinhos selecionados (K). Um K pequeno pode levar a um modelo instável, enquanto um K grande pode levar a um modelo menos sensível às características locais dos dados.

Em sistemas de recomendação, a abordagem mais comum é usar o KNN para criar um sistema de recomendação baseado em item. Isso envolve o cálculo das distâncias entre itens com base nas classificações de usuários. Em seguida, o algoritmo seleciona os K itens mais próximos, com base nessas distâncias, e recomenda esses itens ao usuário. Por exemplo, se um usuário avaliou positivamente alguns filmes de ação e aventura, o sistema de recomendação baseado em item pode recomendar outros filmes semelhantes em termos de gênero, diretor, ator, etc.

Outra abordagem é usar o KNN para criar um sistema de recomendação baseado em usuário. Nesse caso, o algoritmo calcula as distâncias entre usuários com base em suas classificações de itens. Em seguida, o algoritmo seleciona os K usuários mais semelhantes e recomenda itens avaliados positivamente por esses usuários, mas que o usuário em questão ainda não avaliou.

### 3.3.2 Árvore de decisão

A árvore de decisão representa uma estrutura em forma de árvore na qual cada nó interno corresponde a uma decisão baseada em um atributo e cada folha corresponde a uma classificação ou valor de regressão.

A construção de uma árvore de decisão começa com a seleção do atributo mais importante para dividir o conjunto de treinamento em duas ou mais subconjuntos. O processo de seleção de atributo pode ser feito por diferentes critérios, como ganho de informação, índice Gini ou razão de verossimilhança.

Uma vez que o atributo de divisão é escolhido, a árvore de decisão é dividida em dois ramos, cada um correspondendo a um valor do atributo de divisão. Esse processo é repetido recursivamente para cada subconjunto até que os subconjuntos resultantes contenham apenas uma classe ou

um número mínimo de amostras é alcançado.

A árvore de decisão pode ser usada para classificação, onde a classe de uma nova amostra é determinada seguindo o caminho da árvore até chegar a uma folha. Para regressão, o valor predito é a média dos valores dos exemplos de treinamento correspondentes ao caminho da árvore.

Em sistemas de recomendação, existem três abordagens que usam árvores de decisão para recomendação:

- Abordagem baseada em conteúdo: uma árvore de decisão pode ser construída para representar as características dos itens. Os atributos dos itens, como gênero, atores, diretores, entre outros, são usados para construir a árvore de decisão. O usuário pode então especificar suas preferências em relação aos atributos e percorrer a árvore para encontrar itens que correspondam a essas preferências.

- Abordagem baseada em usuário: uma árvore de decisão pode ser construída para representar as preferências dos usuários. Os atributos dos itens avaliados pelos usuários são usados para construir a árvore de decisão. A árvore de decisão é então usada para prever as classificações que um usuário daria a itens que ainda não foram avaliados.

- Abordagem baseada em conteúdo: uma árvore de decisão pode ser construída para representar a relação entre os itens. Os atributos dos itens são usados para construir a árvore de decisão. A árvore de decisão é então usada para encontrar itens semelhantes a um item de interesse, com base em seus atributos.

Uma das principais vantagens de usar árvores de decisão para recomendação é que elas são fáceis de entender e interpretar. Além disso, as árvores de decisão são relativamente rápidas e escaláveis para grandes conjuntos de dados e também podem lidar com dados faltantes. No entanto, a qualidade das recomendações depende da escolha dos atributos e do algoritmo de construção da árvore de decisão. É importante escolher atributos relevantes e usar técnicas eficazes para construir a árvore de decisão.

### 3.3.3 Regressão Logística

A Regressão Logística é um tipo de análise de regressão usada para modelar a probabilidade de um determinado evento ou resultado ocorrer, com base em um conjunto de variáveis predito-

ras. É comumente usada em problemas de classificação binária, onde o objetivo é prever se uma observação pertence a uma das duas classes possíveis.

Nesse algoritmo, a variável de resposta é modelada como uma função das variáveis preditoras, usando a função logística, que mapeia qualquer valor de entrada para um valor entre 0 e 1. Isso permite interpretar a saída do modelo como uma probabilidade de pertencer a uma determinada classe.

Em sistemas de recomendação, a Regressão Logística pode ser usada para prever a probabilidade de um usuário gostar de um item ou não. O modelo é treinado com base nas informações de histórico de navegação e preferências do usuário e nas características dos itens recomendados. As características podem incluir informações como o gênero do filme, o autor do livro ou o preço do produto. Com base nas probabilidades calculadas pelo modelo, o sistema de recomendação pode classificar os itens em ordem de relevância e recomendar os itens mais relevantes para o usuário. Também pode ser usada em sistemas de recomendação baseados em conteúdo, onde as recomendações são feitas com base nas características dos itens recomendados. Nesse caso, o modelo é treinado para prever a probabilidade de um usuário gostar de um item com base em suas características, como o gênero de um filme, o autor de um livro ou o tipo de música.

Uma vantagem dessa técnica é sua simplicidade e interpretabilidade, o que a torna uma escolha popular para aplicações onde a interpretabilidade é importante, como em ciências médicas ou sociais. Além disso, a Regressão Logística tem muitas extensões, como a regressão logística multinomial, que pode lidar com problemas de classificação multiclasse, e a regressão logística ordinal, que pode lidar com variáveis de resposta ordinais No entanto, pode ser limitada em sua capacidade de modelar relacionamentos complexos entre as variáveis preditoras e a variável de resposta.

### 3.3.4 Florestas aleatórias

A técnica de florestas aleatórias (RandomForest) é uma abordagem de aprendizado de máquina que utiliza várias árvores de decisão para realizar a classificação. RandomForest cria várias árvores de decisão independentes, utilizando diferentes subconjuntos dos dados de treinamento, além de diferentes subconjuntos das variáveis de entrada. Essas árvores são treinadas usando uma amostra aleatória dos dados de treinamento e uma amostra aleatória das variáveis de entrada.

Durante a etapa de classificação, cada árvore individual é utilizada para prever a classe de um

novo conjunto de dados. A classe final atribuída é determinada por votação entre as diferentes árvores de decisão.

Uma das principais vantagens do RandomForest é a sua capacidade de lidar com dados de alta dimensionalidade, bem como com dados que apresentam correlação entre as variáveis de entrada. Além disso, ele é capaz de lidar com dados desbalanceados, onde uma ou mais classes possuem um número significativamente menor de exemplos.

Não é o modelo mais comum utilizado em sistemas de recomendação, já que normalmente esses sistemas se baseiam em técnicas de filtragem colaborativa ou de fatoração de matriz. No entanto, é possível utilizar o RandomForest em sistemas de recomendação de maneiras diferentes, dependendo do contexto específico.

Uma possibilidade seria utilizá-lo para classificar usuários com base em seu comportamento de navegação no site. Nesse caso, poderíamos coletar dados de navegação dos usuários, como páginas visitadas, tempo gasto em cada página, ações realizadas (como cliques ou compras), entre outros, e utilizar esses dados como variáveis de entrada do modelo.

Após treinar com os dados de navegação, o modelo poderia ser utilizado para prever a classe (ou categoria) de um novo usuário, com base em seu comportamento de navegação. Por exemplo, poderíamos classificar usuários em categorias como "interessados em produtos esportivos", "interessados em produtos de beleza", "interessados em eletrônicos", entre outros.

A partir dessa classificação, seria possível recomendar produtos ou conteúdos personalizados para cada usuário, de acordo com a categoria à qual ele foi atribuído. Por exemplo, um usuário classificado como "interessado em produtos esportivos"poderia receber recomendações de tênis, roupas de ginástica, ou suplementos alimentares voltados para atletas.

É importante ressaltar que a eficácia desse tipo de sistema de recomendação dependeria da qualidade dos dados de navegação coletados, bem como da capacidade do modelo em classificar corretamente os usuários em suas respectivas categorias.

### 3.3.5 Aumento do gradiente

Aumento do gradiente (GradientBoosting) é um algoritmo de conjunto (ensemble) que combina vários modelos de aprendizado de máquina fracos para criar um modelo mais forte. O algoritmo funciona adicionando novas árvores de decisão ao conjunto, onde cada nova árvore tenta

corrigir os erros dos modelos anteriores. O processo continua até que um número especificado de árvores seja atingido ou até que um critério de parada seja alcançado. A previsão final é feita combinando as previsões de todas as árvores individuais.

O GradientBoosting pode ser usado para construir modelos de aprendizado de máquina que preveem as classificações dos usuários para itens que ainda não foram avaliados. Essas previsões são então usadas para recomendar novos itens aos usuários.

Por exemplo, é possível usar o Gradient Boosting para criar um modelo que preveja as classificações que um usuário daria a um determinado item, com base nas classificações que ele já deu a outros itens e nas classificações de outros usuários que possuem padrões de classificação semelhantes.

Os hiperparâmetros do Gradient Boosting podem ser ajustados para maximizar a precisão das previsões do modelo, e esses modelos podem ser atualizados continuamente à medida que mais dados se tornam disponíveis.

Por fim, o algoritmo é conhecido por sua alta precisão e se tornou uma escolha popular para muitas aplicações, como detecção de fraude, filtragem de spam e previsão de *churn* de clientes. No entanto, pode ser computacionalmente caro e propenso a *overfitting* se não for ajustado corretamente. Portanto, é importante selecionar cuidadosamente os hiperparâmetros para evitar *overfitting* e obter o melhor desempenho possível.

# 4 Desenvolvimento

## 4.1 Métricas adotadas

### 4.1.1 Score

Foi utilizada a métrica accuracy_score do pacote sklearn.metrics. A partir dela é calculada o acurácia dado um conjunto de rótulos previstos em relação aos rótulos verdadeiros. Essa função retorna a fração das previsões corretas. A seguir um exemplo:

```python
from sklearn.metrics import accuracy_score

# define true labels
true_labels = ["a", "c", "b", "a"]

# define corresponding predicted labels
pred_labels = ["c", "c", "b", "a"]

# find accuracy scores
accuracy = accuracy_score(true_labels, pred_labels)
print("The accuracy of prediction is: ", accuracy)

```

Run

Output

```
The accuracy of prediction is:  0.75
```

Figura 1: Exemplo da métrica score.

### 4.1.2 RMSE

A métrica RMSE mede a diferença entre os valores previstos pelo modelo e os valores reais, ao quadrado. Em seguida, é calculada a média desses erros quadráticos e a raiz quadrada do resultado é tomada para retornar a unidade original.

$$RMSE\left(y, \hat{y}\right) = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2}$$

Equação 6 — Equação da raiz do erro quadrático médio. Nesta equação há o cálculo da diferença entre o valor y e ŷ, contudo com a elevação do resultado ao quadrático. Mas para deixar o resultado na mesma escala que os dados, é aplicado a raiz quadrada no resultado.

Figura 2: Fórmula da métrica RMSE.

#### 4.1.3 MAE

A métrica MAE mede a diferença absoluta entre os valores previstos pelo modelo e os valores reais, e em seguida é calculada a média dessas diferenças.

$$MAE\left(y, \hat{y}\right) = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right|$$

Equação 3 — Equação do erro médio absoluto. Nesta equação há o calculo da média da diferença entre o valor predito ŷ e o real y. Quanto menor o valor de MAE, significa que melhor são os resultados preditos pelo modelo de machine learning.

Figura 3: Fórmula da métrica MAE.

## 4.2 Base de dados

Questionário realizado com 200 pessoas, fazendo perguntas como:

- Gênero

- Idade
- Quanto picante você gostaria que sua pizza fosse?
- Você gosta de Pepperoni?
- Você prefere tomates?
- Você gosta de Queijo?
- Você gosta de Capsicum?
- Tem algum problema de saúde ou dieta?

Desta forma, é possível identificar as preferências de cada usuário e tentar predizer para novos usuários.

# 5 Resultados

Dado o banco de dados utilizado, a seguir os resultados das predições realizados pelos algoritmos e métricas adotados.

Tabela 1: Divisão aleatória - 80/20

| Método | SCORE | RMSE | MAE |
|---|---|---|---|
| KNN | 0,55 | 0,67 | 0,45 |
| DECISION TREE | 0,50 | 0,71 | 0,50 |
| LOGISTIC REGRESSION | 0,40 | 0,77 | 0,60 |
| RANDOM FOREST | 0,50 | 0,71 | 0,50 |
| GRADIENT BOOSTING CLASSIFIER | 0,60 | 0,63 | 0,40 |

Tabela 2: K fold com 5 folds

| Método | SCORE | RMSE | MAE |
|---|---|---|---|
| KNN | 0,55 | 0,67 | 0,45 |
| DECISION TREE | 0,50 | 0,71 | 0,50 |
| LOGISTIC REGRESSION | 0,40 | 0,77 | 0,60 |
| RANDOM FOREST | 0,60 | 0,63 | 0,40 |
| GRADIENT BOOSTING CLASSIFIER | 0,60 | 0,63 | 0,40 |

Tabela 3: Stratified k folds com 10 folds

| Método | SCORE | RMSE | MAE |
|---|---|---|---|
| KNN | 0,55 | 0,67 | 0,45 |
| DECISION TREE | 0,50 | 0,71 | 0,50 |
| LOGISTIC REGRESSION | 0,40 | 0,77 | 0,60 |
| RANDOM FOREST | 0,50 | 0,71 | 0,50 |
| GRADIENT BOOSTING CLASSIFIER | 0,60 | 0,63 | 0,40 |

Tabela 4: Stratified shuffle split com 2 folds

| Método | SCORE | RMSE | MAE |
|---|---|---|---|
| KNN | 0,55 | 0,67 | 0,45 |
| DECISION TREE | 0,50 | 0,71 | 0,50 |
| LOGISTIC REGRESSION | 0,40 | 0,77 | 0,60 |
| RANDOM FOREST | 0,50 | 0,71 | 0,50 |
| GRADIENT BOOSTING CLASSIFIER | 0,60 | 0,63 | 0,40 |

# 6 Conclusão

O relatório apresenta uma análise da aplicação de algoritmos em um banco de dados, utilizando diferentes estratégias de divisão dos dados para avaliar a performance dos modelos de predição. Os resultados foram inconclusivos para afirmar que a escolha da estratégia de divisão pode impactar

significativamente a performance do modelo, sendo importante realizar uma avaliação cuidadosa das diferentes opções disponíveis a partir de mais experimentos.

Os algoritmos utilizados não apresentaram desempenho satisfatório, com alto erro médio e baixa correlação com os valores reais. Contudo, não foi possível identificar diferenças significativas na performance dos modelos, variando a estratégia de divisão dos dados adotada.

Considerando as limitações do estudo, é recomendado que futuras pesquisas explorem outras base de dados, além de avaliar o desempenho de diferentes algoritmos. Com uma avaliação criteriosa das estratégias de divisão dos dados e dos algoritmos, é possível obter modelos precisos e robustos para a predição em diferentes contextos.

# Referências


Faraway, J. J. (2016). Does data splitting improve prediction? *Statistics and computing*, *26*, 49–60.

Farias, F., Ludermir, T., & Bastos-Filho, C. (2020). Similarity based stratified splitting: an approach to train better classifiers. *arXiv preprint arXiv:2010.06099*.

Meng, Z., McCreadie, R., Macdonald, C., & Ounis, I. (2020). Exploring data splitting strategies for the evaluation of recommendation models. In *Proceedings of the 14th acm conference on recommender systems* (pp. 681–686).

Roshankhah, R., Karbalaeisadegh, Y., Greer, H., Mento, F., Soldati, G., Smargiassi, A., ... others (2021). Investigating training-test data splitting strategies for automated segmentation and scoring of covid-19 lung ultrasound images. *The Journal of the Acoustical Society of America*, *150*(6), 4118–4127.

Tan, J., Yang, J., Wu, S., Chen, G., & Zhao, J. (2021). A critical look at the current train/test split in machine learning. *arXiv preprint arXiv:2106.04525*.