Pré-Processamento de Dados

Lucas F. Lan[1], Herick Marcio M. Brito[2], Juary dos Santos L. Júnior[3]

[1]Sistemas de Informação – Centro Universitário de Excelência (UNEX) Caixa Postal 44.085-370 – Feira de Santana – BA – Brasil

Lucaslan10@hotmail.com, herickmarcio356@gmail.com, juaryjuniorr@gmail.com

Abstract. *This article presents the implementation of a data preprocessing library using only native Python resources. The work was developed in the context of the Data Mining course, with the objective of applying fundamental techniques such as handling missing values, removing duplicates, normalizing, standardizing, and encoding variables, thus preparing the data for future modeling stages in recommendation systems.*

Resumo. *Este artigo apresenta a implementação de uma biblioteca de pré-processamento de dados utilizando apenas recursos nativos da linguagem Python. O trabalho foi desenvolvido no contexto da disciplina de Mineração de Dados, com o objetivo de aplicar técnicas fundamentais de tratamento de valores ausentes, remoção de duplicados, normalização, padronização e codificação de variáveis, preparando os dados para futuras etapas de modelagem em sistemas de recomendação.*

1. Introdução

Tabela 1. Exemplo de base de dados brutos

ID_Pedido	ID_Cliente	Categoria_Culinaria	Valor_Pedido (R$)	Avaliacao (1-5)	Tempo_Entrega (min)	Forma_Pagamento
101	58	Italiana	85.50	5	99999	Cartão de Crédito
102	72	Brasileira	55.00	None	None	Pix
103	65	Japonesa	120.00	"Bom"	48	Cartão de Crédito
101	58	Italiana	85.50	5	35	Cartão de Crédito
104	81	Italiano	92.30	7	29	Cartão
105	95	Hambúrguer	210.70	1	240	Dinheiro
106	72	Brasileira	?	3	51	Pix

A etapa de pré-processamento de dados é fundamental em projetos de ciência de dados, pois os dados brutos raramente estão em condições ideais para análise. O pré-processamento tem, portanto, o papel de preparar essas informações, garantindo consistência e qualidade. A Tabela 1 apresenta um exemplo de base de dados brutos, na qual é possível identificar diferentes problemas.

Dentre esses problemas, destaca-se a incompletude, caracterizada pela ausência de valores em determinados campos, representados por None ou interrogações. Outro problema recorrente é a inconsistência, que ocorre quando os dados estão presentes, mas violam regras lógicas ou de formatação, como a falta de padronização e erros de

digitação. Um exemplo disso é a categoria culinária registrada de formas distintas, como “Italiana” e “Italiano”.

Além disso, observam-se os chamados ruídos, que correspondem a informações incorretas, corrompidas ou irrelevantes. Nesse caso, um tempo de entrega de 99999 minutos, embora válido numericamente, é irreal, assim como uma avaliação registrada com nota 7 quando o intervalo permitido é de 1 a 5. Caso tais problemas não sejam tratados adequadamente, o resultado da análise de dados será comprometido, produzindo interpretações imprecisas e distorcidas.

Diante desse cenário, este trabalho tem como objetivo implementar, sem o uso de bibliotecas externas como pandas e numpy, uma biblioteca de pré-processamento de dados para resolver os problemas encontrados em bases reais, como a duplicação de dados, valores categóricos em texto, valores ausentes e valores com diferenças de escalas extremas. Portanto, os métodos implementados servirão para tratar esses dados inadequados e fornecê-los já prontos para as próximas etapas de análise de dados.

2. Fundamentação Teórica

2.1. Tratamento de Valores Ausentes

Valores ausentes são campos que não possuem registro, ou seja, campos com valores nulos e é um dos problemas mais frequentes nas bases de dados, isso pode ocorrer por falhas de coleta, erro humano ou omissão proposital. Se não tratados, comprometem a análise para um modelo de classificação.

Para o tratamento desses valores ausentes, temos duas principais abordagens: a remoção que consiste na exclusão das linhas que contêm valores nulos. Apesar de simples, essa abordagem pode reduzir significativamente o tamanho da base de dados, consequentemente uma perda significativa de informações, caso a quantidade de nulos seja grande em relação ao total da base. Portanto, a remoção de dados é ideal ser usada quando a quantidade de valores nulos é pequena em relação ao total da base. Já a outra abordagem chamada de Imputação, consiste no preenchimento desses campos nulos por estimativas ou um valor padrão. As principais estimativas usadas para o preenchimento são os métodos básicos de estatística como média, mediana e moda. Destarte, a imputação tem como característica a preservação das informações, porém como há a adição de valores artificiais, pode distorcer a análise com falsos dados.

Tabela 2. Base de clientes

NOME	IDADE	SALÁRIO
Ana	25	3000
Bruno	None	2500
Carla	30	None

A Tabela 2 ilustra uma base de dados de cliente que alguns não informaram a idade ou o salário. Nesse contexto, aplicando o tratamento de valores ausentes com a abordagem de remoção, a única linha restante seria [Ana - 25 - 3000], pois é a única sem nenhum valor nulo. Em contrapartida, utilizando a abordagem de Imputação com o preenchimento pela média, teríamos:

Tabela 3. Base de dados preenchida pela média

NOME	IDADE	SALÁRIO
Ana	25	3000
Bruno	27.5	2500
Carla	30	2750

2.2. Tratamento de Dados Duplicados

Os dados duplicados são registros idênticos ou quase idênticos que aparecem mais de uma vez na base de dados. A presença de dados duplicados traz consequências como a distorção da análise, devido aos valores que não representam a realidade por serem duplicados. Considere que um cliente estava com a conexão instável e clicou duas vezes para finalizar o pedido, enviando assim dois registros do mesmo pedido, se não forem tratados, o sistema de recomendação concluiria que esse cliente tem preferência por esse prato, todavia é uma análise incorreta devido ao pedido duplicado. Portanto para o tratamento desses dados duplicados, é feito a identificação das linhas duplicadas e em seguida é removida, assim, restando apenas um registro.

2.3. Normalização e Padronização de Atributos Numéricos

As técnicas de normalização e padronização são usadas com objetivo de transformar todas as variáveis na mesma ordem de grandeza. Por exemplo, atributos como valor de um pedido podem variar de 0 a 1000, enquanto a avaliação pode ir de 0 a 5, caso não seja realizado a padronização ou a normalização desses dados, o algoritmo de análise vai dar maior importância para os dados com maior ordem de grandeza, destarte, prejudicando a análise desses dados. A diferença entre elas é que a normalização coloca as variáveis em um intervalo de 0 e 1 (ou -1 e 1, caso tenha resultado negativo), enquanto a padronização transforma as variáveis fazendo com que elas resultem em uma média igual a 0 e desvio padrão igual a 1. Em síntese, utiliza-se a padronização quando os dados têm uma distribuição gaussiana, ou seja, os dados se concentram próximo da média. Já a normalização, usa-se quando não sabemos a distribuição dos dados, ou sabemos que não é gaussiana.

2.3.1. Normalização Min-Max

A normalização Min-Max transforma as variáveis em um intervalo de 0 a 1, sendo o valor mínimo do conjunto transformado em 0 e o valor máximo em 1.

$$X' = \frac{X - Xmin}{Xmax - Xmin}$$

Figura 1. Fórmula da normalização Min-Max

Onde X’ representa o resultado da normalização (um valor entre 0 e 1), X é o valor original que vai ser normalizado, Xmin é o valor mínimo do conjunto de dados e Xmax o valor máximo do conjunto de dados. Por exemplo, considere o conjunto de dados: 50, 100, 200 e 400, temos que o Xmin = 50, e Xmax = 400. Aplicando a normalização Min-Max:

Para o valor 50: 50 – 50 / 400 – 50 = 0

Para o valor 100: 100 – 50 / 400 – 50 = 0.1429

Para o valor 200: 200 – 50 / 400 – 50 -> 0.4286

Para o valor 400: 400 – 50 / 400 – 50 = 1

Assim, o conjunto normalizado fica: 0, 0.1429, 0.4286, 1

2.3.2. Padronização z-score

A padronização z-score transforma os dados de forma que tenham uma média de zero e um desvio padrão de um.

$$z = \frac{x - \mu}{\sigma}$$

Figura 2. Fórmula da padronização z-score

Em que, z é o resultado da padronização, x o valor a ser padronizado, μ a média e σ representa o desvio padrão. Considere o conjunto de dados: 10, 20 e 30, tendo a média aritmética: 20 e o desvio padrão: 8.165, aplicando a fórmula para cada valor:

Para 10: 10 – 20 / 8.165 = -1.225

Para 20: 20 – 20 / 8.165 = 0

Para 30: 30 – 20 / 8.165 = 1.225

Assim, depois de aplicar a fórmula o conjunto passa a ser -1.225, 0 e 1.225, além disso todos os valores passam a ter média = 0 e desvio padrão = 1.

2.4. Codificação de Variáveis Categóricas

A maioria dos algoritmos de aprendizado de máquina exigem receber dados numéricos, ou seja, para usar variáveis categóricas é preciso convertê-las em números para que o algoritmo consiga analisá-los. Portanto, os codificadores são métodos usados para transformar atributos categóricos em representações numéricas, iremos discutir sobre a Codificação de Rótulo (Label Enconding) e Codificação One Hot (One Hot Encoding).

2.4.1 Codificação de Rótulo (Label Enconding)

A técnica de codificação de rótulo consiste na atribuição de um número inteiro para cada categoria única de uma variável categórica. Por exemplo, se tivermos uma variável “categoria culinária” que possui as categorias: “italiana”, “japonesa” e “brasileira”,

utilizando a codificação em rótulo, essas categorias seriam representadas da seguinte forma: Italiana -> 0, japonesa -> 1, brasileira -> 2. Essa técnica introduz uma relação ordinal artificial entre as categorias (2 > 1 > 0), o que pode levar o modelo a interpretar erroneamente e atribuir maior peso para a comida brasileira, por exemplo. Nesse sentido, essa abordagem de codificação é mais bem utilizada com variáveis categóricas ordinais, onde existe uma ordem entre as categorias, como pequeno, médio e grande.

2.4.2 Codificação One Hot (One Hot Enconding)

O método de codificação One Hot transforma cada categoria única de uma variável em uma nova coluna binária, sendo o valor 1 indicando a presença e 0 a ausência da categoria na nova coluna. Por exemplo, suponha que temos a variável “cor” com três categorias únicas: “vermelho”, “azul” e “verde”. Ao utilizarmos o método One Hot nesse conjunto: ["vermelho", "azul", "verde", "azul"], teremos o seguinte resultado:

Tabela 4. Dados após a codificação One Hot

COR	COR_VERMELHO	COR_AZUL	COR_VERDE
vermelho	1	0	0
azul	0	1	0
verde	0	0	1
azul	0	1	0

Embora o One Hot ser uma técnica muito eficaz, ela pode aumentar muito o tamanho do conjunto de dados se a variável tiver muitas categorias únicas, o que pode levar a um aumento no tempo de computação e na memória necessária.

3. Metodologia

A implementação do código foi estruturada com base na separação de responsabilidades entre as classes MissingValueProcessor, responsável pelo tratamento de valores ausentes; Scaler, que aplica as transformações de escala em variáveis numéricas; e Encoder, responsável pela codificação das variáveis categórica. As classes MissingValueProcessor e Scaler contam com um método auxiliar chamado _get_target_columns, utilizado para definir as colunas a serem processadas. Este método funciona da seguinte maneira: se um conjunto de nomes de colunas é fornecido, ele retorna uma lista contendo apenas esses nomes; caso contrário, retorna uma lista com todas as colunas do dataset.

A classe MissingValueProcessor possui os métodos isna, notna, fillna e dropna. Todos recebem como argumento um conjunto de nomes de colunas, que é gerenciado pelo método auxiliar _get_target_columns. O método isna retorna um novo dataset contendo apenas as linhas que possuem pelo menos um valor nulo em uma das colunas especificadas, enquanto o notna retorna um novo dataset com as linhas que não possuem nenhum valor nulo. Ambos operam de modo semelhante: eles iteram sobre as linhas do dataset e aplicam sua respectiva verificação. O isna verifica se qualquer item na linha é nulo e, em caso afirmativo, adiciona a linha inteira ao novo dataset. A única diferença para o notna é que ele verifica se todos os itens são não nulos para então adicionar a linha.

Já o método fillna preenche os valores nulos nas colunas especificadas, utilizando estratégias como a média, moda, mediana ou um valor padrão, definidas via argumentos. Para isso, o método percorre as colunas e utiliza uma estrutura condicional para aplicar a estratégia de preenchimento escolhida, recorrendo à classe Statistics para os cálculos estatísticos. Após determinar o valor de preenchimento, o dataset original é alterado, substituindo os valores nulos. Por fim, o método dropna remove as linhas com valores nulos, aplicando o método notna e atualizando o dataset original com o resultado.

Ademais, a classe Scaler contém os métodos minMax_scaler e standard_scaler. O minMax_scaler aplica a normalização Min-Max, modificando o dataset. O processo consiste em percorrer cada coluna especificada e isolar seus valores numéricos, uma vez que a normalização se aplica apenas a eles. Em seguida, calculam-se os valores máximo e mínimo para obter o intervalo (Xmax - Xmin). Se o intervalo for zero, o que significa que todos os valores são iguais, eles são padronizados para 0.0. Caso contrário, a fórmula de normalização Min-Max (Figura 1) é aplicada a cada valor numérico. Os valores não numéricos são mantidos inalterados. Ao final, a coluna do dataset original é substituída pela nova lista com os valores escalonados.

O método standard_scaler, por sua vez, aplica a padronização Z-score. Sua implementação é semelhante à do minMax_scaler, porém, em vez dos valores máximo e mínimo, calculam-se a média aritmética e o desvio padrão com o auxílio da classe Statistics. Um detalhe importante é a necessidade de usar o método notna para filtrar previamente os valores nulos antes de passá-los para a classe Statistics. Após o cálculo, verifica-se se o desvio padrão é igual a zero para evitar uma divisão por zero. Se for o caso, o que indica que todos os valores são constantes, eles são padronizados para 0.0. Do contrário, a fórmula da padronização Z-score (Figura 2) é aplicada aos valores numéricos, enquanto os não numéricos são mantidos. Por fim, o dataset original é atualizado com a lista de valores escalonados.

Por fim, a classe Encoder é responsável pela conversão de variáveis categóricas em representações numéricas. Ela implementa dois métodos de codificação distintas. A primeira, label_encode, implementa a técnica de Label Encoding, que mapeia cada categoria única de uma coluna para um valor inteiro único. O processo inicia com a identificação de todas as categorias únicas, que são ordenadas. Em seguida, é criado um dicionário que atribui um número inteiro único a cada categoria. Ao percorrer a coluna, cada valor é substituído pelo seu inteiro correspondente em uma nova lista contendo os novos valores codificados, enquanto valores nulos são preservados. No final, a coluna original no dataset é substituída por esta nova lista.

O outra método chamado oneHot_encode, implementa a técnica de One-Hot Encoding. Esta técnica transforma uma única coluna categórica em múltiplas colunas binárias, para cada categoria única identificada, uma nova coluna é adicionada ao dataset, utilizando um nome composto pelo nome original e o da categoria, como Cor_Vermelho, por exemplo. Os valores nesta nova coluna são binários: 1 se a linha na coluna original continha a respectiva categoria, e 0 caso contrário. Após a criação de todas as novas colunas para uma variável, o dataset original é atualizado com as novas colunas e novos valores e a coluna categórica original é removida do dataset.

4. Resultados

A fim de garantir um funcionamento adequado, foram conduzidos testes unitários para cada método. Na execução dos testes unitários, todos os métodos se comportaram de maneira esperada e passaram nos testes.

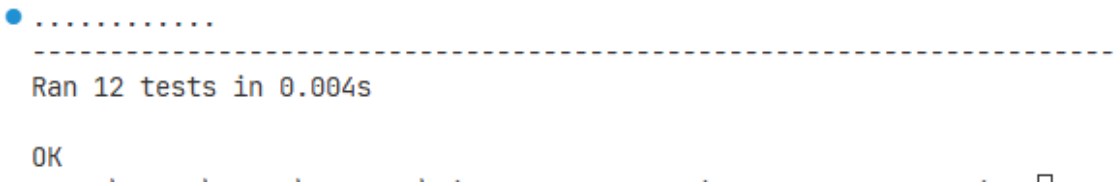

Figura 3. Resultado dos testes unitários.

Em um teste unitário para o método isna foi encontrado um erro nos testes fornecidos, na linha 27 é feito a verificação do length de uma das colunas após o isna. O resultado esperado no teste original era 4, todavia não há como o length ser 4 após uso do isna, pois o dataset apresenta apenas 3 linhas com valores nulos e uma sem valores nulos, portanto foi feito a alteração do resultado esperado de 4 para 3.

Além disso, o "Patch" dos testes estava apontando para um arquivo inexistente, portanto foi feito a alteração para os arquivos existentes.

```
# Patch indica o local ONDE o objeto é usado, não onde ele é definido.
# @patch('preprocessing_lib.MissingValueProcessor')
# @patch('preprocessing_lib.Scaler')
# @patch('preprocessing_lib.Encoder')
# @patch('preprocessing_lib.Statistics')
@patch('preprocessing.MissingValueProcessor')
@patch('preprocessing.Scaler')
@patch('preprocessing.Encoder')
@patch('food_statistics.Statistics')
```

Figura 4. Alteração do patch

5. Considerações Finais

Um dos pontos que acho importante a ser tratado no futuro, é o tratamento de erros mais desenvolvido e elaborado, além disso é crucial pensar sempre no desempenho e performance do algoritmo, então como melhorias principais para o futuro são melhora do desempenho e tratamento de erros.

No começo, foi difícil entender como faríamos para verificar uma linha inteira do dataset, porém à medida que fomos progredindo e estudando mais sobre as funções nativas da linguagem Python, se tornou mais fácil. Além disso, a didática do Python de Dictionary e List Comprehension era estranha, mas depois foi ficando mais normal.

Não teve nenhum método mais difícil que outro, todas apresentaram seus desafios iniciais, porém todos foram superados ao longo do tempo e dedicação.

6. Referências

STACK OVERFLOW. *Como funciona o any e o all em Python?* Disponível em: https://pt.stackoverflow.com/questions/193622/como-funciona-o-any-e-o-all-em-python. Acesso em: 7 set. 2025.

FERRARI, Daniel Gomes; CASTRO, Leandro Nunes de. *Introdução à Mineração de Dados: conceitos básicos, algoritmos e aplicações.* Rio de Janeiro: LTC, 2018. E-book. Disponível em: https://integrada.minhabiblioteca.com.br/reader/books/978-85-472-0100-5. Acesso em: 8 set. 2025.

ESTATÍSTICA FÁCIL. *O que é normalização Min-Max.* Disponível em: https://estatisticafacil.org/glossario/o-que-e-normalizacao-min-max/. Acesso em: 8 set. 2025.

DATACAMP. *Normalization in machine learning.* Disponível em: https://www.datacamp.com/pt/tutorial/normalization-in-machine-learning. Acesso em: 8 set. 2025.

ESTATÍSTICA FÁCIL. *O que é Z-Score Normalization.* Disponível em: https://estatisticafacil.org/glossario/o-que-e-z-score-normalization/. Acesso em: 10 set. 2025.

W3SCHOOLS. *Python enumerate() function.* Disponível em: https://www.w3schools.com/python/ref_func_enumerate.asp. Acesso em: 11 set. 2025.

DATACAMP. *One-hot encoding in Python: tutorial.* Disponível em: https://www.datacamp.com/pt/tutorial/one-hot-encoding-python-tutorial. Acesso em: 20 set. 2025.

PEDRORP. *Guia de codificadores de atributos categóricos em Machine Learning.* Medium, 2020. Disponível em: https://medium.com/@pedrorp/guia-de-codificadores-de-atributos-categ%C3%B3ricos-em-machine-learning-60a9f22c9a3b. Acesso em: 20 set. 2025.

ESTATÍSTICA FÁCIL. *O que é normalização Min-Max.* Disponível em: https://estatisticafacil.org/glossario/o-que-e-normalizacao-min-max/. Acesso em: 20 set. 2025.

DATACAMP. *Normalization vs Standardization.* Disponível em: https://www.datacamp.com/pt/tutorial/normalization-vs-standardization. Acesso em: 20 set. 2025.