

PROGRAMAÇÃO AVANÇADA EM BIG DATA

PROGRAMAÇÃO AVANÇADA EM BIG DATA

Público-alvo: Estudantes do Centro Universitário FMU.

Objetivo: Esta apostila foi desenvolvida sob medida para os estudantes ingressos no curso de Ciência de Dados. Seu propósito fundamental é construir uma ponte sólida e acessível entre a lógica de programação essencial e as complexas arquiteturas de Big Data, Processamento Distribuído e Aprendizado de Máquinas.

Com base nas diretrizes do Plano de Ensino institucional e na experiência executiva e acadêmica de seu autor, este material didático adota uma abordagem gradual ("direto ao ponto"), desmistificando conceitos avançados por meio de linguagem fluida, ilustrações cotidianas, metáforas didáticas e múltiplos estudos de caso em Python.

Ao longo de 16 Unidades, o estudante aprenderá a tratar, manipular, analisar e modelar grandes volumes de dados. O material capacita o futuro cientista de dados não apenas a escrever algoritmos, mas a compreender o fluxo completo de dados (Data Pipelines), da análise exploratória básica aos sistemas de recomendação, deep learning e bancos NoSQL, alinhando a teoria com as reais exigências do mercado corporativo.

Professor: Robson Carmo, mestre pelo programa de mestrado profissional em Gestão e Tecnologia em Sistemas Produtivos do Centro Paula Souza, concluiu 3 MBAs (Inovação e Transformação Digital, Gestão de Empresarial, Engenharia de Software). Instrutor de certificações da SAFE, autor e coautor de vários livros e ebooks.

Lattes: <http://lattes.cnpq.br/8753112683856964>

Orcid: <https://orcid.org/0009-0002-7102-4993>

LLM utilizada: Gemini Flash – Agosto de 2026.

Módulos e Conteúdo

Unidade 1: Introdução ao Universo do Big Data e Análise Exploratória Primária

- Foco: Apresentar o conceito de Big Data de forma simples, demonstrando como grandes volumes de dados impactam a tomada de decisões empresariais, e iniciar a investigação visual e estatística de dados usando Python.
- Conteúdo:
 - O que é Big Data e por que ele importa? (A regra dos V's: Volume, Velocidade, Variedade, Veracidade e Valor).
 - Tipos de variáveis: Qualitativas (categorias, como "Marcas de Carro") vs. Quantitativas (números, como "Preço").
 - Resumindo conjuntos de dados: Medidas de posição (Média, Mediana, Moda) e Variabilidade (Variância e Desvio Padrão).
 - Identificação de Outliers (valores discrepantes) e leitura de gráficos: Distribuição de Frequência, Histogramas e Boxplots.

Unidade 2: Ambiente Python e Modelos de Projeção com Regressão Linear

- Foco: Configurar o ambiente computacional em Python, entender a estrutura básica da linguagem e aprender a prever valores contínuos usando a técnica de Regressão Linear.
- Conteúdo:
 - Instalação e navegação no ambiente Python (Jupyter Notebook / Google Colab).
 - Estrutura de dados básica do Python e tipos de memória.
 - Regressão Linear Simples e Múltipla: Ajustando equações para prever o futuro.
 - Entendendo os parâmetros: Coeficiente de Correlação de Pearson e Coeficiente de Determinação (R^2).
 - Análise de Resíduos e Intervalo de Confiança para previsões mais seguras.

Unidade 3: Lógica Condicional, Classificação e Regressão Logística

- Foco: Dominar o uso de estruturas de decisão em Python e aprender a classificar dados em categorias binárias (ex: Sim/Não, Sucesso/Falha).

- Conteúdo:
 - Operadores lógicos, tabelas verdade (E, OU, NÃO) e tomada de decisão com comandos IF, ELIF e ELSE em Python.
 - Introdução às técnicas de classificação estatística.
 - Regressão Logística Binária: Estimação de parâmetros e testes de hipóteses.
 - Matriz de Confusão e Tabela de Classificação para medir acertos.

Unidade 4: Estruturas de Repetição e Árvores de Decisão

- Foco: Construir automações com laços de repetição em Python e aprender a criar modelos preditivos baseados em regras visuais fáceis de interpretar (Árvores de Decisão).
- Conteúdo:
 - Laços de repetição em Python: Comandos FOR e WHILE para varrer grandes conjuntos de dados.
 - Conceito de Árvore de Decisão: Nós, ramos e folhas.
 - Critérios de divisão e obtenção da Árvore de Decisão.
 - Classificação de grupos e geração de Tabelas de Classificação.

Unidade 5: Agrupamento de Dados (Clustering) com K-Means e Dendogramas

- Foco: Segmentar grandes volumes de dados não rotulados, identificando padrões e grupos parecidos sem a necessidade de uma resposta prévia conhecida.
- Conteúdo:
 - Diferença entre Aprendizado Supervisão e Não Supervisionado.
 - Métodos de Agrupamento Hierárquico e Dendogramas (árvores de agrupamento).
 - O Algoritmo K-Médias (K-Means): Como definir centros de grupos.
 - Aplicações práticas em segmentação de clientes.

Unidade 6: Analytics Avançado - Mineração de texto, Regras de Associação

- Foco: Explorar o processamento de dados não estruturados (textos e conexões) para extrair inteligência de mercado.
- Conteúdo:
 - Mineração de Texto (Text Mining) e Análise de Sentimentos em Python.
 - Análise de Cesto de Compras (Market Basket Analysis) e Regras de Associação.

Unidade 7: Manipulação de Strings

- Foco: Capacitar o estudante na manipulação de dados em formato de texto (Strings).
- Conteúdo:
 - Funções essenciais para fatiamento, busca, substituição e limpeza de Strings em Python.

Unidade 8: Introdução ao Machine Learning

- Foco: Introduzir a base do Machine Learning e algoritmos de Ensemble (combinação de árvores).
- Conteúdo:
 - Introdução ao ecossistema de Machine Learning.
 - Modelos Lineares com Regularização (Lasso e Ridge) para evitar memorização excessiva dos dados (overfitting).
 - Algoritmos Avançados: Bagging, Random Forest (Floresta Aleatória) e Boosting.
 - Máquinas de Vetores de Suporte (Support Vector Machines – SVM).

Unidade 9: Programação Orientada a Objetos (POO) e Seleção de Modelos

- Foco: Estruturação de projetos de Big Data e selecionar as melhores variáveis para um modelo.
- Conteúdo:
 - Modelos Não Lineares (Splines e Modelos Aditivos Generalizados – GAM).

- Técnicas para combinar diferentes modelos preditivos (Ensemble Methods).
- Algoritmos automatizados de seleção de variáveis (Feature Selection).

Unidade 10: Aprendizado por Reforço

- Foco: Compreender como as plataformas modernas recomendam produtos e aprendem por tentativa e erro.
- Conteúdo:
 - Introdução ao Aprendizado por Reforço (Reinforcement Learning): Agentes, Recompensas e Ambientes.
 - Métricas de desempenho para seleção rigorosa de modelos (RMSE, MAE, ROC-AUC).

Unidade 11: Introdução a Redes Neurais

- Foco: Processamento assíncrono e eficiente em Big Data, associando à introdução ao Deep Learning.
- Conteúdo:
 - Fundamentos do Deep Learning: O Neurônio Artificial (Perceptron) e Redes Neurais Artificialmente Conectadas.

Unidade 12: Decoradores, Classes Avançadas e Aplicações Práticas de Deep Learning

- Foco: Aprofundar o uso de recursos avançados do Python e construir soluções práticas com Redes Neurais Profundas (Deep Learning).
- Conteúdo:
 - Arquiteturas de Deep Learning: Visão geral e funcionamento das camadas ocultas.
 - Aplicação prática de redes profundas em reconhecimento de imagens e processamento de linguagem.
 - Avaliação de desempenho e prevenção de sobreajuste em redes profundas.

Unidade 13: Modelagem de Dados NoSQL

- Foco: Projetar esquemas de dados eficientes para bancos NoSQL.
- Conteúdo:
 - Modelagem de dados orientada a consultas para HBase, Cassandra e DynamoDB.
 - Comparativo prático: Modelagem Relacional vs. Modelagem NoSQL.

Unidade 14: Interfaces Gráficas e Projeto Integrador Prático em NoSQL

- Foco: Desenvolver painéis e interfaces visuais para o usuário final interagir com os modelos e construir um projeto prático completo integrando Python e NoSQL.
- Conteúdo:
 - Criação de Interfaces Gráficas do Usuário (GUI) simples em Python para apresentação de resultados de Big Data.
 - Conexão da interface visual com o banco de dados NoSQL.
 - Desenvolvimento de um Projeto Integrador Prático de NoSQL focado na solução de um problema real de mercado.

Unidade 1: Introdução ao Universo do Big Data e Análise Exploratória Primária

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Seja bem-vindo(a) à Unidade 1 da nossa disciplina de Programação Avançada em Big Data!

Se você acabou de ingressar no curso de Ciência de Dados, é natural que termos como "Big Data", "Pipeline de Dados", "Outliers" ou "Machine Learning" pareçam conceitos complexos ou abstratos. No entanto, a Ciência de Dados nada mais é do que a arte e a técnica de transformar volumes brutos de informação em decisões inteligentes.

Como ressaltam Soubhia et al. (2019), o aprendizado da linguagem de programação e do tratamento de dados deve ocorrer de forma progressiva e orientada à prática. Matthes (2016) reforça que a programação ganha significado quando aplicada à resolução de problemas reais, permitindo que algoritmos automatizem tarefas analíticas que seriam inviáveis de forma manual. Por sua vez, Borges (2009) destaca que a versatilidade do Python faz dele o ecossistema ideal para conectar a infraestrutura de dados às análises estatísticas mais avançadas.

Nesta primeira unidade, abordaremos os conceitos fundamentais do Big Data, a classificação dos dados e as ferramentas estatísticas primárias necessárias para compreender o comportamento de um conjunto de dados.

2. CONCEITOS FUNDAMENTAIS DE BIG DATA: OS V'S DO BIG DATA

Diariamente, milhões de transações bancárias são efetuadas, posts são publicados em redes sociais, sinalizações de GPS são enviadas e sensores de Internet das Coisas (IoT) registram dados de temperatura, umidade e tráfego. Este ecossistema hiperconectado deu origem ao fenômeno do Big Data.

Para caracterizar o Big Data, a literatura consolidou o modelo dos **5 V's**:

Volume: Refere-se à quantidade colossal de dados gerados a cada segundo. Não estamos mais falando de planilhas de centenas de linhas, mas de terabytes, petabytes ou exabytes.

Velocidade: A velocidade estonteante com que novos dados são criados e precisam ser processados. Pense na detecção de fraude de um cartão de crédito: a análise precisa ocorrer em milissegundos enquanto a compra é autorizada na maquininha.

Variedade: Os dados não vêm apenas em tabelas organizadas. Eles se apresentam de forma estruturada (tabelas relacionais), semiestruturada (arquivos JSON, XML) e não estruturada (áudios, imagens, vídeos, textos de redes sociais).

Veracidade: A qualidade e confiabilidade dos dados. Dados incorretos, duplicados ou ruidosos geram análises errôneas ("Lixo entra, lixo sai").

Valor: O propósito final do Big Data. De nada adianta armazenar petabytes de dados se eles não gerarem insights acionáveis para o negócio ou para a sociedade.

Exemplo Prático do Cotidiano:

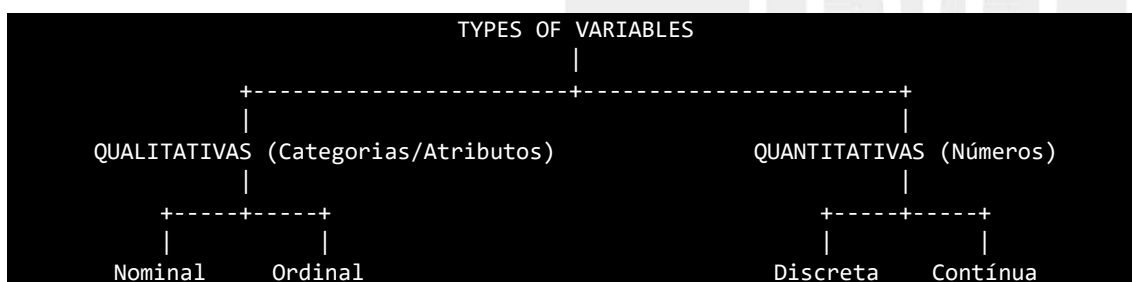
Imagine um aplicativo de navegação como o Waze ou Google Maps.

- **Volume:** Milhões de motoristas com a localização ativada simultaneamente.
- **Velocidade:** A posição do veículo é atualizada a cada segundo.
- **Variedade:** Coordenadas de GPS, alertas em texto inseridos por usuários e imagens de satélite.
- **Veracidade:** Filtrar sinalizações falsas de acidentes ou perda de sinal de GPS dentro de túneis.
- **Valor:** Recalcular a rota em tempo real para economizar 15 minutos do seu deslocamento.

3. CLASSIFICAÇÃO DAS VARIÁVEIS NA CIÊNCIA DE DADOS

Antes de aplicar qualquer fórmula ou algoritmo, o cientista de dados deve responder a uma pergunta básica: "Que tipo de dado eu tenho em mãos?"

Conforme abordado por Soubhia et al. (2019) e Borges (2009), a estrutura dos tipos de dados orienta a escolha do tipo de gráfico, da técnica estatística e do tipo de variável a ser declarada em Python.



3.1. Variáveis Qualitativas (ou Categóricas)

Representam características, qualidades ou atributos que não possuem um valor numérico inerente.

- **Nominais:** Não existe nenhuma ordem ou hierarquia entre as categorias. Exemplos: Sexo (Masculino, Feminino), Cidade de residência (São Paulo, Rio de Janeiro), Cor do carro (Prata, Preto, Vermelho).
- **Ordinais:** Existe uma ordenação ou hierarquia natural entre as categorias. Exemplos: Nível de escolaridade (Ensino Médio, Graduação, Mestrado, Doutorado), Nível de satisfação (Muito Insatisfeito, Insatisfeito, Neutro, Satisfeito, Muito Satisfeito).

3.2. Variáveis Quantitativas (ou Numéricas)

Representam contagens ou medições numéricas reais onde operações matemáticas fazem sentido.

- **Discretas:** Decorrem de contagens, assumindo apenas valores inteiros e pontuais. Exemplos: Quantidade de filhos, Número de acessos a um site em um dia, Quantidade de compras efetuadas.
- **Contínuas:** Decorrem de medições e podem assumir qualquer valor dentro de um intervalo contínuo (números decimais/reais). Exemplos: Altura (1,75 m), Salário (R\$ 4.500,50), Tempo de resposta de um servidor (0,042 segundos).

4. MEDIDAS DE POSIÇÃO E DE VARIABILIDADE

Para resumir e sintetizar um conjunto de dados numericamente, utilizamos a estatística descritiva. Matthes (2016) destaca que o Python possui bibliotecas especializadas (como numpy e pandas) que realizam esses cálculos com alta eficiência em grandes volumes de dados.

4.1. Medidas de Posição (ou Tendência Central)

Buscam identificar um valor central em torno do qual os dados se distribuem.

- **Média ($\bar{x}$):** Somatório de todos os valores dividido pelo número total de observações.

$$\bar{x} = \frac{\sum_{i=1}^n x_i}{n}$$

Atenção: A média é altamente sensível a valores extremos (outliers).

- **Mediana:** É o valor que ocupa exatamente a posição central dos dados quando estes estão ordenados (em ROL).

Se n for ímpar: é o termo central.

Se n for par: é a média aritmética dos dois termos centrais.

Vantagem: A mediana é uma medida robusta, pois não se altera com valores extremos.

- **Moda:** É o valor (ou categoria) que aparece com maior frequência no conjunto de dados. Um conjunto pode ser amodal (sem moda), unimodal, bimodal ou multimodal.

4.2. Medidas de Variabilidade (ou Dispersão)

Indicam o quão afastados ou dispersos os dados estão em relação à média.

- **Amplitude Total:** Diferença entre o maior e o menor valor do conjunto (Max - Min).
- **Variância (s^2):** Média dos quadrados dos desvios em relação à média.

$$s^2 = \frac{\sum_{i=1}^n (x_i - \bar{x})^2}{n - 1}$$

- **Desvio Padrão (s):** Raiz quadrada da variância. Expressa a dispersão na mesma unidade de medida original dos dados, facilitando a interpretação prática.

$$s = \sqrt{s^2}$$

Exemplo Prático:

Duas turmas de Ciência de Dados fizeram uma prova e obtiveram média 7,0.

- ◆ Turma A: As notas foram 7,0; 7,0; 7,0; 7,0. (Desvio Padrão = 0,0 - turma homogênea).
- ◆ Turma B: As notas foram 4,0; 5,0; 9,0; 10,0. (Desvio Padrão elevado - turma heterogênea).

Conclusão: Apenas a média não basta para entender o perfil dos dados; precisamos do desvio padrão!

4.3. Covariância

A covariância é uma medida estatística que indica a direção da relação linear entre duas variáveis quantitativas contínuas ou discretas. Em termos simples, ela nos diz se duas variáveis tendem a variar juntas na mesma direção ou em direções opostas.

Como Interpretar o Valor da Covariância

O sinal do resultado da covariância é o elemento principal para a sua interpretação:

- **Covariância Positiva** ($\text{Cov}(X, Y) > 0$): Indica que as variáveis se movem na mesma direção. Quando X aumenta, Y também tende a aumentar (e vice-versa).

Exemplo: Quanto maior a quantidade de horas estudadas (X), maior tende a ser a nota na prova (Y).

- **Covariância Negativa** ($\text{Cov}(X, Y) < 0$): Indica que as variáveis se movem em direções opostas. Quando X aumenta, Y tende a diminuir.

Exemplo: Quanto maior o preço de um produto (X), menor tende a ser a quantidade vendida (Y).

- **Covariância Próxima de Zero** ($\text{Cov}(X, Y) \approx 0$): Indica que não há relação linear perceptível entre as duas variáveis.

Exemplo: A altura de uma pessoa (X) e a quantidade de livros que ela lê por ano (Y).

Fórmula Matemática

A covariância entre duas variáveis X e Y em uma amostra é calculada dividindo a soma dos produtos dos desvios de cada ponto em relação às suas respectivas médias pelo número total de observações menos um ($n - 1$)

$$\text{Cov}(X, Y) = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{n - 1}$$

Limitação da Covariância e a Relação com a Correlação

Embora a covariância seja fundamental (por exemplo, sendo usada diretamente no cálculo do coeficiente angular β_1 da Regressão Linear), ela possui uma limitação importante: ela não é padronizada.

- O valor numérico da covariância depende diretamente das unidades de medida das variáveis. Se você medir a altura em metros, a covariância terá um valor; se medir a mesma altura em centímetros, o número será 100 vezes maior, embora a relação real entre os dados não tenha mudado.
- Devido a isso, a covariância nos diz a direção da relação, mas não nos permite medir a força dessa relação.

Para resolver essa limitação, dividimos a covariância pelo produto dos desvios padrões de X e Y, obtendo o Coeficiente de Correlação de Pearson (r), que padroniza o resultado em uma escala fixa entre -1 e +1:

$$r = \frac{\text{Cov}(X, Y)}{s_X \cdot s_Y}$$

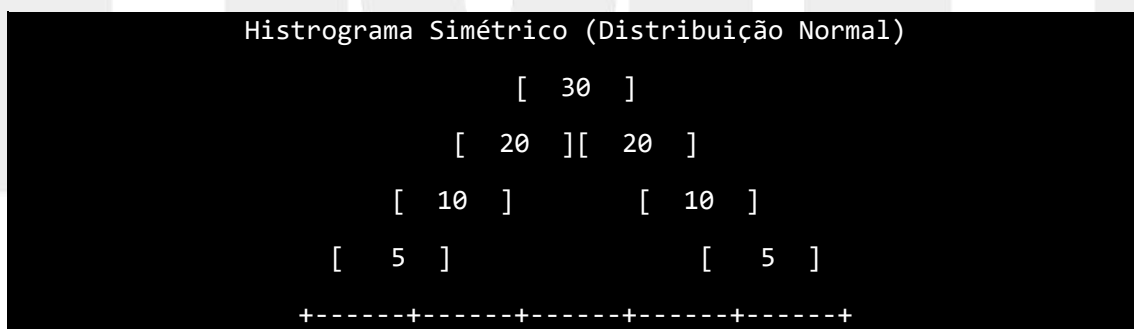
5. FERRAMENTAS VISUAIS DE ANÁLISE EXPLORATÓRIA

Como enfatiza Borges (2009), a representação gráfica é uma etapa indispensável no ciclo de vida da análise de dados, permitindo identificar padrões, simetrias e anomalias de forma intuitiva.

5.1. Tabela de Distribuição de Frequência e Histograma

A Tabela de Distribuição de Frequência organiza os dados em classes (intervalos) e conta quantas observações pertencem a cada classe.

O **Histograma** é a representação gráfica em colunas contíguas (unidas) dessa distribuição de frequência. Ele permite visualizar a forma da distribuição dos dados (se é simétrica, assimétrica à direita ou assimétrica à esquerda).



5.2. Boxplot (Gráfico de Caixa) e Identificação de Outliers

O **Boxplot** é uma das ferramentas mais poderosas da Análise Exploratória de Dados. Ele sintetiza a distribuição dos dados a partir do chamado Resumo dos Cinco Números:

- **Mínimo** (Limite Inferior LI = $Q_1 - 1,5 \times \text{IQR}$)
- **Primeiro Quartil** (Q_1): 25% dos dados estão abaixo deste valor.
- **Segundo Quartil** (Q_2 / Mediana): 50% dos dados estão abaixo deste valor.
- **Terceiro Quartil** (Q_3): 75% dos dados estão abaixo deste valor.
- **Máximo** (Limite Superior LS = $Q_3 + 1,5 \times \text{IQR}$)

O intervalo entre Q_1 e Q_3 é chamado de **Intervalo Interquartil** ($\text{IQR} = Q_3 - Q_1$) e forma o "corpo" da caixa.

Esquema Anatomicamente Detalhado de um Boxplot (Outlier)

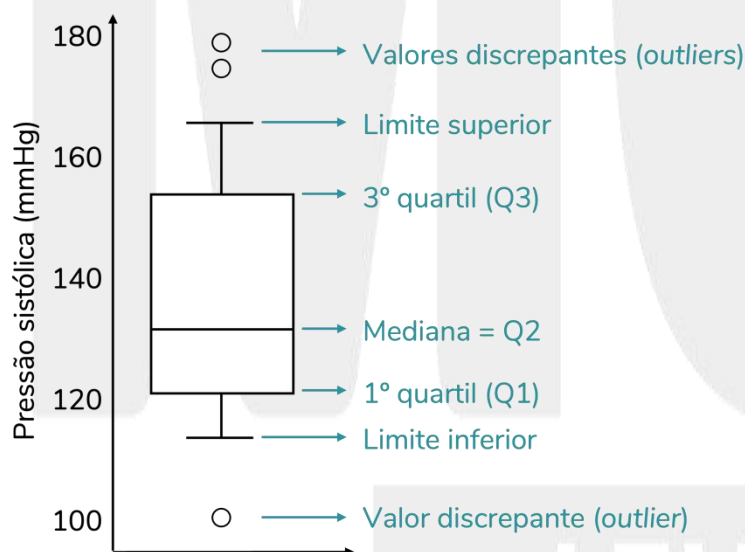
```

+--- Limite Superior (LS = Q3 + 1.5 * IQR)
|
+---+ <--- Terceiro Quartil (Q3 - 75%)
|
|---| <--- Mediana (Q2 - 50%)
|
+---+ <--- Primeiro Quartil (Q1 - 25%)
|
+--- Limite Inferior (LI = Q1 - 1.5 * IQR)

```

Identificação de Outliers (Valores Discrepantes):

Qualquer ponto localizado além dos **limites inferior (LI)** ou **superior (LS)** é classificado como um **Outlier**. Outliers podem indicar erros de digitação/medição ou casos genuínos e raros de grande valor para o negócio (ex: uma transação financeira suspeita).



6. SÍNTESE DO CONTEÚDO E REFERÊNCIAS TEÓRICAS

Nesta unidade, compreendemos que o Big Data é caracterizado por Volume, Velocidade, Variedade, Veracidade e Valor. Aprendemos a classificar as variáveis em qualitativas (nominais/ordinais) e quantitativas (discretas/contínuas) e a calcular medidas fundamentais como Média, Mediana, Moda, Variância e Desvio Padrão. Por fim, dominamos o uso do Histograma e do Boxplot para inspeção visual e detecção de outliers.

EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

Uma grande empresa de e-commerce coletou dados sobre o comportamento de compras de seus clientes durante um evento de liquidação. Entre as variáveis registradas na base de dados principal, destacam-se:

1. Grau de satisfação do cliente com o atendimento (Classificado em: Baixo, Médio, Alto).
2. Tempo de navegação no site até a conclusão da compra (medido em segundos).
3. Quantidade de itens adicionados ao carrinho de compras.
4. Estado brasileiro onde a entrega será realizada (ex: SP, RJ, MG).

Considerando a classificação das variáveis na Ciência de Dados (SOUBHIA et al., 2019; BORGES, 2009), assinale a alternativa que classifica corretamente e respectivamente as quatro variáveis descritas acima.

- A) Qualitativa Ordinal; Quantitativa Contínua; Quantitativa Discreta; Qualitativa Nominal.
- B) Qualitativa Nominal; Quantitativa Discreta; Quantitativa Contínua; Qualitativa Ordinal.
- C) Qualitativa Ordinal; Quantitativa Discreta; Quantitativa Contínua; Qualitativa Nominal.
- D) Qualitativa Nominal; Quantitativa Contínua; Quantitativa Discreta; Qualitativa Ordinal.
- E) Quantitativa Discreta; Quantitativa Contínua; Qualitativa Ordinal; Qualitativa Nominal.

QUESTÃO 2

Um analista de dados júnior foi encarregado de examinar o tempo de processamento das requisições em um servidor de banco de dados. Ele obteve a seguinte amostra de tempos (em milissegundos):

Amostra: [12, 15, 15, 18, 20, 22, 98]

Ao apresentar o relatório para a coordenação, o analista afirmou que "o tempo típico de resposta do servidor é de 28,5 ms", utilizando unicamente a Média Aritmética para justificar a afirmação.

A respeito dessa situação e dos conceitos de medidas de posição e variabilidade (SOUBHIA et al., 2019; MATTHES, 2016), avalie as asserções a seguir e a relação proposta entre elas:

I. A média aritmética não é a medida de tendência central mais adequada para descrever o tempo típico dessa amostra, pois foi severamente afetada pelo valor discrepante (98 ms).

PORQUE

II. A mediana é uma medida de posição robusta à presença de outliers, sendo mais indicada para representar a centralidade de conjuntos de dados que apresentam assimetria ou valores extremos.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 3

Em um projeto de infraestrutura e arquitetura de dados para uma rede hospitalar, a equipe de TI precisa estruturar uma solução capaz de integrar os prontuários eletrônicos dos pacientes, imagens de ressonância magnética, dados de monitores cardíacos transmitidos continuamente e cadastros de convênios.

À luz dos conceitos teóricos dos 5 V's do Big Data, analise as seguintes afirmativas:

- I. O processamento contínuo dos sinais vitais oriundos dos monitores cardíacos à medida que são gerados exemplifica a dimensão da Velocidade.
- II. A coexistência de arquivos de imagem médica (dados não estruturados) com tabelas de cadastros de convênios (dados estruturados) exemplifica a dimensão da Variedade.

III. O descarte de prontuários com dados duplicados ou sem identificação válida do paciente refere-se diretamente à busca pelo Volume.

É correto o que se afirma em:

- A) I, apenas.
- B) III, apenas.
- C) I e II, apenas.
- D) II e III, apenas.
- E) I, II e III.

QUESTÃO 4

Durante a etapa de sanitização e Análise Exploratória de Dados de um dataset contendo o salário mensal dos funcionários de uma startup de tecnologia, o cientista de dados construiu um Boxplot para diagnosticar a distribuição do conjunto de dados.

Os valores calculados para os quartis foram:

- Primeiro Quartil (Q_1): R\$ 4.000,00
- Mediana (Q_2): R\$ 6.000,00
- Terceiro Quartil (Q_3): R\$ 8.000,00

Considerando o método do Intervalo Interquartil (IQR) para a determinação dos limites e identificação de outliers no Boxplot (SOUBHIA et al., 2019), assinale a opção que apresenta o valor do Limite Superior (LS) a partir do qual um salário será considerado um outlier.

- A) R\$ 10.000,00
- B) R\$ 12.000,00
- C) R\$ 14.000,00
- D) R\$ 16.000,00
- E) R\$ 18.000,00

QUESTÃO 5

Um sistema de monitoramento de tráfego urbano armazena o número diário de veículos que transitam por uma determinada avenida central. O analista responsável selecionou uma amostra de 5 dias e obteve os seguintes registros:

Veículos por dia: [10, 12, 14, 16, 18] (em milhares)

Com base nessa amostra, calcule a Média Aritmética, a Variância Amostral e o Desvio Padrão Amostral, e assinale a alternativa correta.

- A) Média = 14; Variância Amostral = 10; Desvio Padrão Amostral = $\sqrt{10} \approx 3,16$.
- B) Média = 14; Variância Amostral = 8; Desvio Padrão Amostral = $\sqrt{8} \approx 2,83$.
- C) Média = 14; Variância Amostral = 2,5; Desvio Padrão Amostral = $\sqrt{2,5} \approx 1,58$.
- D) Média = 12; Variância Amostral = 10; Desvio Padrão Amostral = $\sqrt{10} \approx 3,16$.
- E) Média = 14; Variância Amostral = 4; Desvio Padrão Amostral = 2,00.

QUESTÃO 6

Para realizar a análise de sentimentos a partir de avaliações de usuários sobre um aplicativo móvel, um cientista de dados utiliza scripts em linguagem Python baseando-se nas recomendações de Borges (2009) e Matthes (2016) para pré-processamento de texto.

Considere os seguintes procedimentos aplicados aos textos das avaliações:

1. Remoção de pontuações e conversão de todo o texto para letras minúsculas.
2. Agrupamento do número final de estrelas concedidas (de 1 a 5 estrelas).
3. Contagem do número total de palavras escritas no comentário.

Assinale a opção que associa corretamente cada procedimento ao seu respectivo tipo de dado resultante.

A) 1. Dado Numérico Contínuo; 2. Variável Qualitativa Nominal; 3. Variável Quantitativa Discreta.

B) 1. Dado Textual (Não Estruturado/String); 2. Variável Qualitativa Ordinal; 3. Variável Quantitativa Discreta.

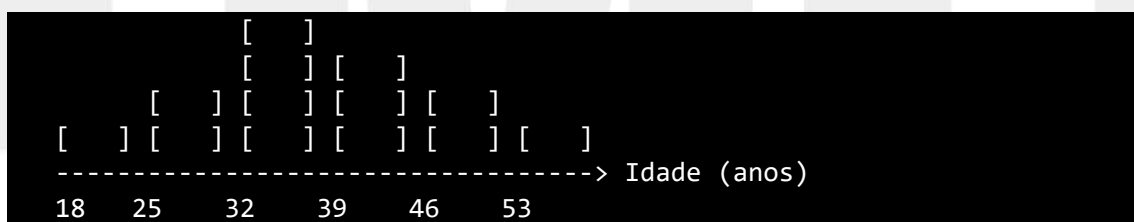
C) 1. Dado Textual (Não Estruturado/String); 2. Variável Qualitativa Nominal; 3. Variável Quantitativa Contínua.

D) 1. Variável Qualitativa Ordinal; 2. Variável Qualitativa Nominal; 3. Variável Quantitativa Discreta.

E) 1. Dado Textual (Não Estruturado/String); 2. Variável Quantitativa Discreta; 3. Variável Quantitativa Contínua.

QUESTÃO 7

Ao analisar o histograma referente às idades dos clientes de uma plataforma de investimentos bancários, o cientista de dados observou a seguinte distribuição visual:



A partir da análise do gráfico e dos conceitos de distribuições de frequência (SOUBHIA et al., 2019), avalie as afirmações a seguir:

I. O gráfico indica uma distribuição assimétrica, na qual a maior concentração de clientes encontra-se nas faixas de idade mais jovens (entre 25 e 39 anos).

II. O Histograma é a representação gráfica adequada para variáveis quantitativas contínuas ou discretas agrupadas em classes.

III. A amplitude das classes de idade exibidas no gráfico é de 7 anos.

É correto o que se afirma em:

A) I, apenas.

B) II, apenas.

C) I e III, apenas.

D) II e III, apenas.

E) I, II e III.

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Classificação Automática de Variáveis e Coleta de Dados

Cenário: Você recebeu um lote de dados brutos referente ao cadastro de novos usuários em uma plataforma de aprendizado. Sua missão é criar uma rotina simples em Python que receba os dados de um usuário e exiba um resumo formatado, informando a qual tipo de variável cada dado pertence.

Requisitos:

1. Solicite ao usuário que informe:
 - a) Nome completo (Ex: "Ana Silva")
 - b) Idade em anos (Ex: 22)
 - c) Escolaridade ("Médio", "Graduação", "Pós")
 - d) Salário pretendido (Ex: 4500.50)
2. Armazene as respostas em variáveis com os tipos de dados nativos corretos em Python.
3. Imprima na tela uma ficha resumida contendo o valor e a classificação de cada variável (Qualitativa Nominal, Qualitativa Ordinal, Quantitativa Discreta ou Quantitativa Contínua).

EXERCÍCIO PRÁTICO 2: Calculadora de Estatística Descritiva (Média, Mediana e Moda)

Cenário: Uma equipe de monitoramento de redes coletou o tempo de resposta (em milissegundos) de um servidor web durante 9 requisições consecutivas:

Tempos de resposta (ms): [150, 120, 200, 120, 180, 120, 300, 150, 220]

Sua tarefa é escrever um script em Python para calcular manualmente a Média, a Mediana e a Moda desse conjunto de dados.

Requisitos:

1. Ordene a lista em ordem crescente.
2. Calcule a média somando os elementos e dividindo pela quantidade total.
3. Identifique o valor mediano acessando a posição central do conjunto ordenado.
4. Encontre a moda contando qual elemento mais se repete.

EXERCÍCIO PRÁTICO 3: Cálculo da Variância e Desvio Padrão Amostral

Cenário: Você precisa analisar o nível de dispersão e variabilidade do faturamento diário (em milhares de reais) de uma loja durante uma semana (5 dias úteis):

Faturamento (R\$ mil): [10, 12, 14, 16, 18]

Escreva um programa em Python que calcule a Variância Amostral (s^2) e o Desvio Padrão Amostral (s).

Requisitos:

1. Calcule a média dos dados.
2. Construa uma estrutura de repetição (for) para calcular a soma dos quadrados dos desvios $(x_i - \bar{x})^2$.
3. Calcule a variância dividindo a soma dos quadrados dos desvios por $n - 1$.
4. Extraia a raiz quadrada do resultado para obter o desvio padrão.

EXERCÍCIO PRÁTICO 4: Algoritmo Detector de Outliers (Regra do Boxplot / IQR)

Cenário: Uma equipe de cibersegurança registrou o volume de dados trafegados (em Megabytes) por vários computadores em uma rede interna. Eles suspeitam que alguns computadores possam estar infectados e enviando volumes anômalos de dados (outliers).

Tráfego (MB): [40, 42, 45, 48, 50, 52, 55, 58, 60, 180]

Crie um script em Python que implemente a Regra do Intervalo Interquartil (IQR) do Boxplot para detectar e isolar automaticamente os outliers do conjunto.

Requisitos:

1. Ordene o conjunto de dados.
2. Calcule o Primeiro Quartil (Q_1 - percentil 25%) e o Terceiro Quartil (Q_3 - percentil 75%).
3. Calcule o IQR = $Q_3 - Q_1$.
4. Determine o Limite Inferior ($LI = Q_1 - 1,5 \times IQR$) e o Limite Superior ($LS = Q_3 + 1,5 \times IQR$).
5. Identifique e exiba na tela quais valores da lista estão fora desses limites.

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Após compreendermos os pilares teóricos e visuais do Big Data na Unidade 1, avançamos agora para a construção de modelos preditivos e a consolidação do ambiente de desenvolvimento em Python.

Como enfatiza Borges (2009), o Python se destaca na engenharia e ciência de dados por sua sintaxe limpa, alto nível de abstração e dinamismo no manuseio de estruturas de memória. Matthes (2016) assinala que a facilidade de integração de scripts em ambientes interativos (como Jupyter Notebook e Google Colab) acelera a prototipagem de algoritmos e a experimentação analítica. Já Soubhia et al. (2019) destacam que a transição da análise descritiva para a preditiva exige o domínio de técnicas estatísticas de projeção, permitindo antecipar comportamentos futuros a partir de variáveis explicativas.

Nesta unidade, exploraremos a configuração do ambiente computacional Python, as estruturas de dados nativas e a teoria completa por trás dos modelos de projeção usando Regressão Linear Simples e Múltipla.

2. COMPREENSÃO DO AMBIENTE COMPUTACIONAL PYTHON E TIPOS DE MEMÓRIA

Para processar grandes volumes de dados, o cientista de dados precisa entender como o ambiente Python lida com os dados na memória do computador.

2.1. Ambientes de Desenvolvimento (IDEs e Notebooks)

- **Jupyter Notebook / Google Colab:** Ambientes baseados em cells (células) que misturam código executável, textos explicativos e gráficos interativos. São ideais para análise exploratória e prototipagem de modelos preditivos.
- **Interpretador Python:** Executa o código linha por linha (linguagem interpretada). Isso permite testar rotinas de forma dinâmica, mas exige atenção ao consumo de memória RAM em operações de Big Data.

2.2. Tipos de Dados e Estruturas de Memória Nativas

Conforme Borges (2009) e Soubhia et al. (2019), o Python organiza os dados em tipos primitivos e estruturas compostas:

Tipos Primitivos:

- **int**: Números inteiros (ex: contagem de acessos).
- **float**: Números de ponto flutuante/decimais (ex: preços, coordenadas).
- **str**: Cadeias de caracteres/texto (ex: nomes, categorias).
- **bool**: Valores lógicos (True ou False).

Estruturas de Dados Nativas:

- **Listas** (list): Coleções ordenadas, mutáveis e heterogêneas. Declaradas com colchetes [].
- **Tuplas** (tuple): Coleções ordenadas e imutáveis (ideais para proteger dados que não devem ser alterados). Declaradas com parênteses ().
- **Dicionários** (dict): Estruturas do tipo chave-valor ({chave: valor}), fundamentais para representação de dados semiestruturados (como JSON).

3. TÉCNICAS DE PROJEÇÃO: REGRESSÃO LINEAR SIMPLES

A Regressão Linear é uma técnica estatística supervisionada cujo objetivo é modelar a relação entre uma variável de resposta (dependente, Y) e uma ou mais variáveis explicativas (independentes, X).

3.1. O Modelo Matemático da Regressão Linear Simples

Quando temos apenas uma variável explicativa (X), a equação da reta que descreve a relação é dada por:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

Onde:

- **Y**: Variável dependente (o valor contínuo que queremos prever).
- **X**: Variável independente ou explicativa.
- **B₀** (beta 0): Intercepto ou coeficiente linear (o valor de Y quando X = 0).
- **B₁** (beta 1): Coeficiente angular ou inclinação da reta (a variação esperada em Y para cada unidade de aumento em X).
- **ε** (epsilon): Erro aleatório ou resíduo (a diferença entre o valor real e o valor previsto pelo modelo).



3.2. Método dos Mínimos Quadrados Ordinários (MQO)

Para encontrar a "melhor reta", o modelo busca minimizar a soma dos quadrados dos resíduos (SQR). Os coeficientes B_0 e B_1 são estimados matematicamente pelas fórmulas:

$$\beta_1 = \frac{\sum_{i=1}^n (X_i - \bar{X})(Y_i - \bar{Y})}{\sum_{i=1}^n (X_i - \bar{X})^2} = \frac{Cov(X, Y)}{Var(X)}$$

$$\beta_0 = \bar{Y} - \beta_1 \bar{X}$$

4. REGRESSÃO LINEAR MÚLTIPLA

No mundo real e em cenários corporativos de Big Data, um fenômeno raramente é explicado por uma única variável. A Regressão Linear Múltipla estende o modelo simples para acomodar k variáveis explicativas:

$$Y = B_0 + B_1 X_1 + B_2 X_2 + \dots + B_k X_k + \epsilon$$

Exemplo: Prever o preço de um imóvel (Y) utilizando a área em m^2 (X_1), a quantidade de quartos (X_2) e a distância até o centro da cidade (X_3).

5. AVALIAÇÃO E INTERPRETAÇÃO DO MODELO DE REGRESSÃO

Não basta ajustar uma reta aos dados; o cientista de dados precisa avaliar a qualidade da projeção.

5.1. Coeficiente de Correlação de Pearson (r)

Mede a força e a direção da relação linear entre duas variáveis numéricas contínuas. Seu valor varia no intervalo de -1 a +1:

- $r = +1$: Correlação linear positiva perfeita.
- $r = 0$: Ausência de relação linear.
- $r = -1$: Correlação linear negativa perfeita.

5.2. Coeficiente de Determinação (R^2) e R^2 Ajustado

O R^2 indica a proporção da variabilidade total da variável dependente Y que é explicada pelas variáveis independentes do modelo. Seu valor varia de 0 a 1 (ou 0% a 100%).

- **R^2 Ajustado:** Na regressão múltipla, adicionar novas variáveis sempre aumenta (ou mantém) o R^2 comum, mesmo que essas variáveis sejam irrelevantes. O **R^2 Ajustado** penaliza a inclusão de variáveis desnecessárias, sendo a métrica ideal para comparar modelos com números diferentes de preditores.

5.3. Análise de Resíduos

Os resíduos ($\epsilon_i = Y_i - \hat{Y}_i$) devem atender aos seguintes pressupostos teóricos para que as inferências do modelo sejam válidas:

- **Média Zero:** A média dos resíduos deve ser igual a zero.
- **Homocedasticidade:** A variância dos resíduos deve ser constante ao longo de todas as previsões (não deve formar formato de "funil").
- **Independência:** Os resíduos não devem estar autocorrelacionados.
- **Normalidade:** Os resíduos devem seguir uma distribuição aproximadamente normal.

EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

Um cientista de dados está desenvolvendo uma rotina em Python para ingestão de dados em um pipeline de Big Data. Ele precisa armazenar uma lista fixa de coordenadas geográficas dos centros de distribuição da empresa que não podem sofrer alterações

durante a execução do script, além de criar uma estrutura flexível que permita mapear o código do produto ao seu preço correspondente.

Considerando as características das estruturas de dados nativas da linguagem Python (BORGES, 2009; MATTHES, 2016), assinale a alternativa que indica as estruturas de dados mais adequadas para cada uma das duas necessidades, respectivamente.

- A) Lista (list) e Tupla (tuple).
- B) Tupla (tuple) e Dicionário (dict).
- C) Dicionário (dict) e Lista (list).
- D) Tupla (tuple) e Lista (list).
- E) Conjunto (set) e Tupla (tuple).

QUESTÃO 2

Uma equipe de marketing utilizou um modelo de Regressão Linear Simples para projetar o volume de vendas (Y, em milhares de reais) com base nos investimentos em publicidade digital (X, em milhares de reais). O modelo estimado via Mínimos Quadrados Ordinários resultou na seguinte equação:

$$\hat{Y} = 15,0 + 4,5 X$$

Com base nos coeficientes estimados do modelo, analise as afirmativas a seguir:

- I. O coeficiente linear ($\beta_0 = 15,0$) indica que, mesmo se o investimento em publicidade digital for nulo, a estimativa de vendas é de R\$ 15.000,00.
- II. O coeficiente angular ($\beta_1 = 4,5$) indica que, para cada R\$ 1.000,00 adicionais investidos em publicidade, espera-se um aumento médio de R\$ 4.500,00 no volume de vendas.
- III. Se a empresa investir R\$ 10.000,00 em publicidade ($X = 10$), a projeção de vendas será de R\$ 60.000,00.

É correto o que se afirma em:

- A) I, apenas.
- B) II, apenas.
- C) I e III, apenas.
- D) II e III, apenas.

E) I, II e III.

QUESTÃO 3

Ao analisar o nível de associação linear entre a temperatura ambiente (X , em $^{\circ}\text{C}$) e o consumo de energia elétrica (Y , em kWh) de um centro de dados, um analista obteve um Coeficiente de Correlação de Pearson igual a $r = 0,85$.

Considerando a interpretação do coeficiente de correlação de Pearson e do coeficiente de determinação (R^2) (SOUBHIA et al., 2019), assinale a opção correta.

- A) Existe uma correlação linear negativa forte entre a temperatura e o consumo de energia.
- B) A variação da temperatura ambiente explica aproximadamente 85% da variabilidade do consumo de energia elétrica.
- C) O Coeficiente de Determinação (R^2) desse modelo é de aproximadamente 0,7225, indicando que 72,25% da variação do consumo de energia é explicada pela temperatura.
- D) A relação entre as variáveis é perfeitamente inversamente proporcional.
- E) O coeficiente $r = 0,85$ comprova conclusivamente que a temperatura é a única causa do aumento no consumo de energia.

QUESTÃO 4

Um engenheiro de dados construiu dois modelos de Regressão Linear Múltipla para prever o consumo de combustível de frotas logísticas:

- Modelo A: Inclui 3 variáveis explicativas e obteve $R^2 = 0,82$ e R^2 Ajustado = 0,80.
- Modelo B: Inclui as mesmas 3 variáveis do Modelo A mais 5 novas variáveis explicativas, obtendo $R^2 = 0,83$ e R^2 Ajustado = 0,74.

A respeito da avaliação e comparação desses dois modelos, avalie as asserções a seguir e a relação proposta entre elas:

I. O Modelo A deve ser preferido em relação ao Modelo B para fins de projeção e parcimônia.

PORQUE

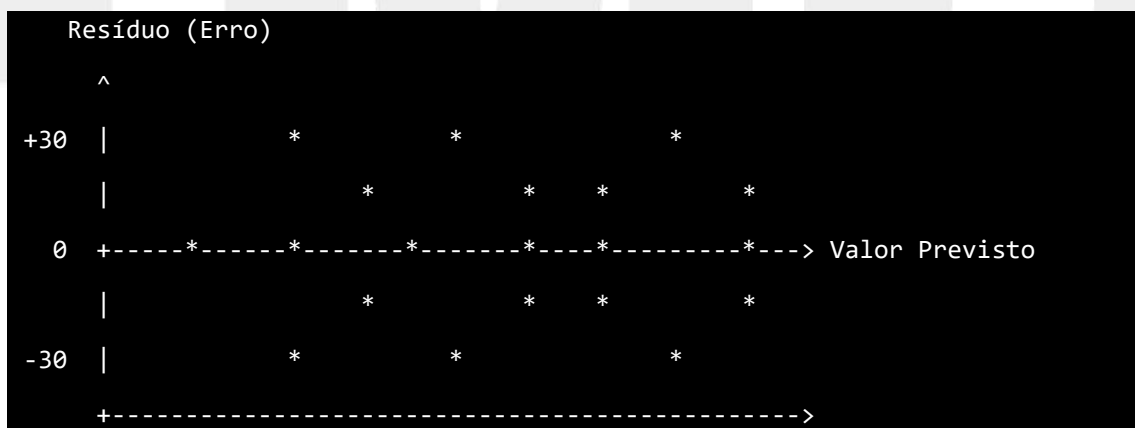
II. O R^2 Ajustado penaliza a adição de variáveis que não contribuem significativamente para a capacidade explicativa do modelo, evitando a inclusão de ruídos indesejados.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 5

Na validação de um modelo de Regressão Linear Múltipla projetado para prever o tempo de entrega de mercadorias, o cientista de dados plotou o gráfico de resíduos ($e_i = Y_i - \hat{Y}_i$) em função dos valores previstos ($\hat{Y}_i$) e observou o seguinte comportamento:



O gráfico revela que a dispersão dos erros se expande à medida que o valor previsto aumenta (formato de funil/leque).

Essa violação dos pressupostos teóricos do modelo de regressão é conhecida como:

- A) Autocorrelação serial.
- B) Multicolinearidade perfeita.
- C) Heterocedasticidade.
- D) Normalidade dos resíduos.
- E) Homocedasticidade perfeita.

QUESTÃO 6

Suponha que um modelo de Regressão Linear Simples tenha sido ajustado para prever o número de chamados de suporte técnico (Y) em função do número de novos clientes contratados (X). Os dados amostrais de 4 semanas foram:

$$X = [1, 2, 3, 4]$$

$$Y = [3, 5, 7, 9]$$

Sabendo que as médias são $\bar{X} = 2,5$ e $\bar{Y} = 6,0$, e aplicando as fórmulas do Método dos Mínimos Quadrados Ordinários (MQO), os coeficientes angular (β_1) e linear (β_0) desse modelo são, respectivamente:

- A) $\beta_1 = 2,0$ e $\beta_0 = 1,0$
- B) $\beta_1 = 1,0$ e $\beta_0 = 2,0$
- C) $\beta_1 = 2,5$ e $\beta_0 = 0,0$
- D) $\beta_1 = 3,0$ e $\beta_0 = 1,5$
- E) $\beta_1 = 0,5$ e $\beta_0 = 4,75$

QUESTÃO 7

Considere o seguinte trecho de código escrito em linguagem Python para manipulação de listas e dicionários no processamento de dados (BORGES, 2009; MATTHES, 2016):

```
dados_vendas = [  
    {"cliente": "A", "valor": 100.0},  
    {"cliente": "B", "valor": 250.0},  
    {"cliente": "A", "valor": 150.0}  
]  
  
total_A = 0.0  
for item in dados_vendas:  
    if item["cliente"] == "A":  
        total_A += item["valor"]  
  
print(total_A)
```

Após a execução desse código, qual valor será exibido no terminal?

- A) 100.0
- B) 150.0
- C) 250.0
- D) 500.0
- E) 0.0

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Modelagem Nativa de Regressão Linear Simples (MQO)

Cenário:

Uma equipe de engenharia de software quer projetar o tempo de compilação de um sistema (Y, em minutos) com base no número de linhas de código alteradas (X, em milhares de linhas). Eles coletaram os seguintes dados de 5 compilações:

Linhas de Código (X): [10, 20, 30, 40, 50]

Tempo de Compilação (Y): [12, 22, 32, 42, 52]

Escreva um script em Python para calcular os coeficientes angular (β_1) e linear (β_0) pelo método dos Mínimos Quadrados Ordinários (MQO) e prever o tempo para um ajuste de 35 mil linhas.

Requisitos:

1. Calcule a média de X e a média de Y.
2. Utilize um laço de repetição para calcular a covariância entre X e Y e a variância de X.
3. Estime β_1 e β_0 .
4. Faça a projeção para X = 35.

EXERCÍCIO PRÁTICO 2: Avaliação do Modelo — Coeficiente de Correlação (r) e Determinação (R^2)

Cenário:

Com base nas previsões obtidas no exercício anterior ou em novos dados de vendas, você precisa medir o grau de associação linear e a capacidade explicativa do modelo criado.

Considere os dados reais Y e os dados previstos pelo modelo $\hat{Y}$:

Y = [10, 15, 20, 25, 30]

$\hat{Y}$ = [11, 14, 21, 24, 30]

Crie uma rotina em Python que calcule o Coeficiente de Determinação (R^2) comparando a soma dos quadrados dos resíduos (SQR) e a soma dos quadrados totais (SQT).

Requisitos:

5. Calcule a média dos valores reais de Y ($\bar{Y}$).
6. Calcule $SQT = \sum (Y_i - \bar{Y})^2$.
7. Calcule $SQR = \sum (Y_i - \hat{Y}^i)^2$.
8. Obtenha $R^2 = 1 - \frac{SQR}{SQT}$ e interprete a porcentagem explicada pelo modelo.

EXERCÍCIO PRÁTICO 3: Projetor com Regressão Linear Múltipla Usando Vetores/Listas

Cenário:

Um modelo de Regressão Linear Múltipla foi treinado para projetar o valor mensal da fatura de energia de um servidor de banco de dados (Y , em R\$) com base no número de transações efetuadas (X_1 , em milhares) e na temperatura média do ambiente (X_2 , em °C). Os coeficientes estimados foram:

- Intercepto (β_0): R\$ 150,00
- Coeficiente da Transação (β_1): R\$ 12,50 por mil transações
- Coeficiente da Temperatura (β_2): R\$ 8,00 por °C

Sua tarefa é criar uma função em Python chamada `projetar_fatura` que receba os coeficientes como um dicionário e um novo lote de dados como uma lista de tuplas $[(X1_1, X2_1), (X1_2, X2_2), \dots]$, retornando a lista de faturas projetadas.

Requisitos:

1. Defina um dicionário para os coeficientes.
2. Escreva a função que itera sobre a lista de entradas e calcula $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2$.
3. Formate a saída em tela.

EXERCÍCIO PRÁTICO 4: Diagnóstico de Resíduos e Verificação de Homocedasticidade

Cenário:

Para verificar se o pressuposto de Homocedasticidade (variância constante dos erros) foi atendido em um modelo de regressão, o cientista de dados precisa analisar a diferença entre os valores reais e previstos (resíduos).

Escreva um script Python que receba uma lista de valores reais Y e uma lista de valores previstos $\hat{Y}$, calcule o vetor de resíduos e verifique a dispersão em duas metades do conjunto de dados para diagnosticar um possível efeito de "funil" (heterocedasticidade).

Requisitos:

1. Dado Y e $\hat{Y}$, crie a lista de resíduos $e_i = Y_i - \hat{Y}_i$.
2. Divida a lista de resíduos ao meio (primeira metade e segunda metade).
3. Calcule o desvio médio absoluto de cada metade para comparar a variância dos erros.

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Nas unidades anteriores, exploramos a análise exploratória básica e a projeção de variáveis contínuas por meio da Regressão Linear. No entanto, grande parte dos problemas reais no ecossistema corporativo e de Big Data não envolve a previsão de valores numéricos contínuos (como faturamento ou temperatura), mas sim a **tomada de decisão categórica** e a **classificação de eventos** (por exemplo: "Esta transação é fraudulenta ou legítima?", "O cliente vai cancelar o serviço ou continuar?").

Como ressaltam Soubhia et al. (2019), o controle do fluxo de execução por meio de estruturas condicionais é a base para a implementação de algoritmos de tomada de decisão em Python. Matthes (2016) enfatiza que a avaliação de expressões lógicas permite que o código reaja de forma autônoma a diferentes cenários de dados. Por sua vez, Borges (2009) destaca a elegância e eficiência do Python ao lidar com operadores booleanos e ao integrar esses conceitos às bibliotecas de modelagem probabilística e aprendizado de máquina.

Nesta unidade, você aprenderá a estruturar tomadas de decisão em Python utilizando operadores lógicos e tabelas verdade, compreenderá os fundamentos das técnicas de classificação e dominará o modelo de Regressão Logística Binária para estimar probabilidades e avaliar acertos operacionais.

2.1. Operadores Relacionais e Lógicos

Conforme abordado por Borges (2009) e Soubhia et al. (2019), expressões condicionais em Python retornam um tipo de dado primitivo booleano: True (Verdadeiro) ou False (Falso).

Operadores Relacionais:

== (Igual)

!= (Diferente)

> (Maior)

< (Menor)

>= (Maior ou Igual)

<= (Menor ou Igual).

Operadores Lógicos:

and (E): Retorna True apenas se todas as condições forem verdadeiras.

or (OU): Retorna True se pelo menos uma das condições for verdadeira.

not (NÃO): Inverte o valor booleano da expressão.

2.3. Desvios Condicionais (if, elif, else)

Em Python, a estrutura condicional permite desviar o fluxo de execução do programa conforme a avaliação lógica de uma ou mais regras. Matthes (2016) lembra que o Python utiliza rigorosamente a indentação (espaçamento) para delimitar o bloco de código de cada condição.

```
# Exemplo de estrutura condicional encadeada
pontuacao_credito = 750

if pontuacao_credito >= 800:
    status = "Crédito Aprovado - Risco Muito Baixo"
elif pontuacao_credito >= 600:
    status = "Crédito Aprovado - Risco Médio"
else:
    status = "Crédito Negado - Risco Alto"
```

2.2. Tabelas Verdade

A tabela verdade sintetiza o comportamento dos operadores lógicos combinando duas condições (A e B):

Condição A	Condição B	A and B	A or B	not A
TRUE	TRUE	TRUE	TRUE	FALSE
TRUE	FALSE	FALSE	TRUE	FALSE
FALSE	TRUE	FALSE	TRUE	TRUE
FALSE	FALSE	FALSE	FALSE	TRUE

3. INTRODUÇÃO ÀS TÉCNICAS DE CLASSIFICAÇÃO

Na Ciência de Dados, a Classificação é uma tarefa de Aprendizado de Máquina Supervisionado em que o objetivo do modelo é prever uma rótulo ou categoria discreta (Y) para uma dada observação, com base em um conjunto de características explicativas (X).

- **Classificação Binária:** A variável resposta Y assume apenas dois valores possíveis (ex: 0 ou 1, "Não" ou "Sim", "Saudável" ou "Doente").
- **Classificação Multiclasse:** A variável resposta Y pode assumir três ou mais categorias distintas (ex: "Baixo", "Médio", "Alto").

4. TÉCNICA DE CLASSIFICAÇÃO: REGRESSÃO LOGÍSTICA BINÁRIA

A **regressão logística binária** é um método estatístico que calcula a **chance de um evento** com **dois resultados possíveis** acontecer. Apesar de conter "Regressão" em seu nome, a **Regressão Logística Binária** é um algoritmo clássico utilizado para problemas de **classificação binária**.

4.1. Por que não utilizar a Regressão Linear para Classificação?

Se tentássemos ajustar uma reta de Regressão Linear comum para prever uma variável binária ($Y \in \{0, 1\}$), teríamos dois grandes problemas:

1. A reta projetaria valores fora do intervalo [0, 1] (ex: previsões como -0,5 ou 1,8, que não fazem sentido probabilístico).
2. Os resíduos violariam os pressupostos de normalidade e homocedasticidade.

4.2. A Função Sigmoide (Logística)

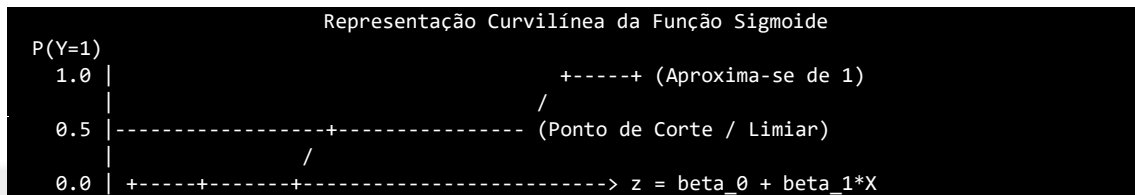
Para resolver esse problema, a Regressão Logística aplica uma transformação matemática à combinação linear das variáveis explicativas ($z = \beta_0 + \beta_1 X_1 + \dots + \beta_k X_k$), utilizando a Função Sigmoide.

A função sigmoide comprime qualquer valor real **z** para o intervalo estrito entre **0 e 1**, permitindo interpretar o resultado diretamente como uma Probabilidade **p = P(Y = 1 | X)**:

$$p = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X_1 + \dots + \beta_k X_k)}}$$

Onde:

- **e:** Constante matemática de Euler ($\approx 2,71828$).
- **p:** Probabilidade estimada de o evento pertencer à classe 1.



4.3. Razão de Chances (Odds) e Logit

A Regressão Logística é baseada no conceito de **Chance** (Odds), que é a razão entre a probabilidade de um evento ocorrer (**p**) e a probabilidade de ele não ocorrer (**1 - p**):

$$\text{Odds} = \frac{p}{1 - p}$$

Ao aplicar o logaritmo natural nessa razão de chances, obtemos a transformação **Logit**, que restabelece a relação linear com os coeficientes:

$$\text{Logit}(p) = \ln \left(\frac{p}{1 - p} \right) = \beta_0 + \beta_1 X_1 + \dots + \beta_k X_k$$

5. AVALIAÇÃO DE DESEMPENHO: MATRIZ DE CONFUSÃO E TABELA DE CLASSIFICAÇÃO

Uma vez calculada a probabilidade **p**, o modelo estabelece um **Limiar de Decisão** (geralmente 0,5). Se **p ≥ 0,5**, a observação é classificada como **1**; caso contrário, é classificada como **0**.

Para medir o desempenho do modelo, comparamos as classes reais com as classes previstas por meio da **Matriz de Confusão** (Tabela de Classificação):

	Classe Prevista: 0	Classe Prevista: 1
Classe Real: 0	Verdadeiro Negativo (VN)	Falso Positivo (FP) (Erro Tipo I)
Classe Real: 1	Falso Negativo (FN) (Erro Tipo II)	Verdadeiro Positivo (VP)

Principais Métricas extraídas da Matriz de Confusão:

- **Acurácia (Accuracy):** Taxa geral de acertos do modelo sobre o total de casos.

$$\text{Acurácia} = \frac{VP + VN}{VP + VN + FP + FN}$$

- **Precisão (Precision):** Das previsões positivas feitas pelo modelo, quantas estavam realmente corretas?

$$\text{Precisão} = \frac{VP}{VP + FP}$$

- **Sensibilidade / Revocação (Recall):** De todos os casos reais positivos existentes, quantos o modelo conseguiu detectar?

$$\text{Sensibilidade} = \frac{VP}{VP + FN}$$

6. Paradoxo da Acurácia

6.1. O Problema das Bases de Dados Desbalanceadas

Em problemas reais de classificação (como detecção de fraudes financeiras, diagnósticos de doenças raras ou interrupção de sistemas), é muito comum que uma das classes seja numericamente minúscula em relação à outra.

- Classe Negativa (0 - Maioria): Pacientes Saudáveis (95% da base = 950 pessoas em um lote de 1.000).
- Classe Positiva (1 - Minoria/Interesse): Pacientes Doentes (5% da base = 50 pessoas em um lote de 1.000).

6.2. A Armadilha da Acurácia

A Acurácia mede o percentual global de acertos do modelo, sem distinguir quais classes ele acertou:

$$\text{Acurácia} = \frac{\text{Total de Acertos}}{\text{Total de Casos}} = \frac{VP + VN}{VP + VN + FP + FN}$$

Se um algoritmo "preguiçoso" ou mal treinado decidir simplesmente chute "**Saudável (0)**" para absolutamente todos os pacientes que passarem pela porta do hospital, o resultado matemático será:

- **Verdadeiros Negativos (VN):** 950 (Acertou todos os saudáveis)
- **Verdadeiros Positivos (VP):** 0 (Não detectou nenhum doente)
- **Falsos Positivos (FP):** 0 (Não acusou nenhum saudável erroneamente)
- **Falsos Negativos (FN):** 50 (Deixou passar todos os 50 doentes)

Calculando a Acurácia desse modelo inútil:

$$\text{Acurácia} = \frac{0 + 950}{0 + 950 + 0 + 50} = \frac{950}{1000} = 0,95 \quad (95\%)$$

- **O Diagnóstico:** O modelo aparenta ter um desempenho excepcional (95% de acerto global), mas na prática ele é **completamente inútil para a medicina**, pois enviou 50 pessoas doentes para casa sem tratamento (Falsos Negativos). A acurácia **maskou** o fracasso do modelo.

6.3. As Métricas Corretas: Sensibilidade vs. Precisão

Para não cair nessa armadilha, o cientista de dados analisa a **Matriz de Confusão** por meio de **métricas específicas para a classe de interesse** (1 - Doentes):

A) Sensibilidade (Recall ou Pontaria de Detecção)

Mede a capacidade do modelo de encontrar todos os casos reais da classe positiva.

$$\text{Sensibilidade} = \frac{VP}{VP + FN} = \frac{0}{0 + 50} = 0,00 \quad (0\%)$$

O indicador de Sensibilidade escancara que o modelo falhou em 100% das vezes na tarefa principal!

B) Precisão (Precision)

Mede a confiabilidade dos alertas positivos emitidos pelo modelo.

$$\text{Precisão} = \frac{VP}{VP + FP}$$

6.4. Soluções Práticas na Engenharia de Dados

Para evitar o problema apresentado, utilizamos técnicas de balanceamento antes de treinar o modelo:

I. Técnicas de Reamostragem (Resampling):

- **Oversampling** (Ex: SMOTE): Cria dados sintéticos para aumentar a representatividade da classe minoritária.
- **Undersampling**: Reduz aleatoriamente a quantidade de exemplos da classe majoritária.

II. Ajuste de Pesos de Classe (Class Weights): Penalizar o modelo com mais severidade quando ele erra a classe minoritária.

III. Uso de Métricas Alternativas: Avaliar o modelo pelo F1-Score (média harmônica entre Precisão e Sensibilidade) ou pela Área sob a Curva ROC (AUC-ROC) em vez da Acurácia pura.

EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

No desenvolvimento de um módulo de verificação para controle de acesso em um sistema bancário corporativo, um desenvolvedor escreveu o seguinte bloco de código em linguagem Python (BORGES, 2009; MATTHES, 2016):

```
idade = 25
autenticado = True
bloqueado = False

if (idade >= 18 and autenticado) and not bloqueado:
    status = "Acesso Permitido"
else:
    status = "Acesso Negado"

print(status)
```

Considerando a avaliação de expressões lógicas e a tabela verdade dos operadores em Python (SOUBHIA et al., 2019), qual será a saída exibida no terminal após a execução do script?

- A) "Acesso Negado"
- B) "Acesso Permitido"
- C) True
- D) False
- E) O programa apresentará um erro de sintaxe lógica durante a compilação.

QUESTÃO 2

Uma equipe de prevenção a fraudes em pagamentos digitais precisa implementar um algoritmo para classificar transações em "Suspeita de Fraude" ($Y = 1$) ou "Transação Legítima" ($Y = 0$). Ao discutir a escolha do algoritmo adequado, o cientista de dados responsável explicou que ajustar uma reta de Regressão Linear Simples comum não é apropriado para essa tarefa.

A respeito da inadequação da Regressão Linear para problemas de classificação binária e da aplicação da Regressão Logística (SOUBHIA et al., 2019), avalie as asserções a seguir e a relação proposta entre elas:

I. A Regressão Logística é mais indicada do que a Regressão Linear para problemas de resposta binária, pois utiliza a Função Sigmoide para mapear o resultado em uma faixa de valores contida estritamente no intervalo entre 0 e 1.

PORQUE

II. O uso da Regressão Linear para variáveis dependentes dicotômicas pode produzir previsões numéricas inferiores a 0 ou superiores a 1, o que inviabiliza a sua interpretação direta como probabilidade.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 3

Em um modelo de Regressão Logística Binária ajustado para prever o cancelamento de assinaturas em um serviço de streaming ($Y = 1$ para cancelado; $Y = 0$ para ativo), a combinação linear calculada para um determinado cliente resultou em $z = 0$.

Sabe-se que a probabilidade p estimada pela Regressão Logística é obtida pela função sigmoide $p = \frac{1}{1+e^{-z}}$

Com base nessa informação, qual é a probabilidade estimada de cancelamento para esse cliente e como ele será classificado considerando um limiar de decisão padrão de 0,5?

- A) Probabilidade de 0,0 (0%) e classificado como Ativo (0).
- B) Probabilidade de 0,25 (25%) e classificado como Ativo (0).
- C) Probabilidade de 0,50 (50%) e classificado como Cancelado (1).
- D) Probabilidade de 0,75 (75%) e classificado como Cancelado (1).
- E) Probabilidade de 1,0 (100%) e classificado como Cancelado (1).

QUESTÃO 4

Um hospital comunitário implementou um modelo preditivo baseado em Regressão Logística para triagem de pacientes em estado crítico na UTI. O diagnóstico do modelo busca prever se o paciente necessita de intervenção imediata ($Y = 1$) ou se está estável ($Y = 0$).

Ao avaliar a Matriz de Confusão do modelo aplicado a uma amostra de 100 pacientes, foram obtidos os seguintes resultados:

- Verdadeiros Positivos (VP): 40 pacientes
- Falsos Positivos (FP): 10 pacientes
- Falsos Negativos (FN): 5 pacientes
- Verdadeiros Negativos (VN): 45 pacientes

Sob a perspectiva da segurança clínica, a métrica mais crítica a ser maximizada é a Sensibilidade (Recall), pois a ocorrência de Falsos Negativos pode levar um paciente crítico a não receber atendimento rápido.

O valor da Sensibilidade desse modelo de triagem é de:

- A) 80,0%
- B) 88,8%
- C) 85,0%
- D) 89,0%
- E) 90,0%

QUESTÃO 5

Considere o seguinte problema de lógica condicional expresso em Python para categorização da pontuação de um cliente em um programa de fidelidade (BORGES, 2009; MATTHES, 2016):

```
pontos = 150

if pontos > 200:
    categoria = "Ouro"
if pontos > 100:
    categoria = "Prata"
if pontos > 50:
    categoria = "Bronze"
else:
    categoria = "Iniciante"

print(categoria)
```

Devido ao uso inadequado de múltiplos comandos if independentes em vez da estrutura encadeada com elif, qual será a categoria impressa ao final do script para um cliente com 150 pontos?

- A) "Ouro"
- B) "Prata"
- C) "Bronze"
- D) "Iniciante"
- E) O programa gerará um erro de sintaxe por repetição da palavra if.

QUESTÃO 6

Na formulação da Regressão Logística Binária, a relação entre as variáveis independentes e a probabilidade p da ocorrência do evento de interesse é expressa linearmente por meio da transformação Logit.

A transformação Logit é matematicamente definida como:

- A) O quadrado do desvio padrão da variável dependente.
- B) O logaritmo natural da razão de chances (Odds), dada por $\ln\left(\frac{p}{1-p}\right)$
- C) A raiz quadrada da variância das probabilidades observadas.
- D) A razão entre os Falsos Positivos e os Verdadeiros Positivos da matriz de confusão.
- E) A média aritmética ponderada dos coeficientes angulares da regressão.

QUESTÃO 7

Um modelo de classificação para diagnósticos médicos de uma doença altamente contagiosa foi submetido a testes e apresentou uma Acurácia geral de 95%. No entanto, ao analisar detalhadamente a Matriz de Confusão, constatou-se que a base de dados testada era altamente desbalanceada (continha 95% de pacientes saudáveis e apenas 5% de doentes) e que o modelo classificou todos os pacientes como "Saudáveis".

Analisando o comportamento do modelo preditivo diante do desbalanceamento de classes, conclui-se que:

- A) A alta Acurácia valida o modelo como seguro e pronto para uso clínico sem necessidade de modificações.
- B) A Acurácia é uma métrica suficiente e confiável para validar modelos em bases de dados desbalanceadas.

C) A Sensibilidade do modelo para a classe doente é de 100%, comprovando sua eficácia diagnóstica.

D) A Acurácia mascarou a ineficácia do modelo, que apresentou Sensibilidade igual a 0% para a detecção da doença real.

E) O erro Tipo I (Falso Positivo) foi maximizado, enquanto o erro Tipo II foi reduzido a zero.

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Avaliador de Regras de Crédito com Operadores Lógicos

Cenário: Você foi contratado por uma fintech para criar o motor de regras inicial que classifica o risco de concessão de empréstimo a partir de três variáveis de um cliente: Idade (int), Renda Mensal (float) e Histórico de Inadimplência (bool).

A política de concessão segue as seguintes regras condicionais:

1. **Risco Baixo:** Idade ≥ 21 E Renda ≥ 5000.00 E NÃO possui Histórico de Inadimplência.
2. **Risco Médio:** Idade ≥ 21 E Renda ≥ 2500.00 (mesmo que possua inadimplência antiga ou renda moderada).
3. **Risco Alto:** Qualquer situação que não se enquadre nas regras anteriores.

Escreva um script em Python que receba esses dados e imprima a classificação de risco.

EXERCÍCIO PRÁTICO 2: Implementação Nativa da Função Sigmoide e Cálculo de Probabilidade

Cenário: Na Regressão Logística Binária, a combinação linear das variáveis explicativas gera um valor z que precisa ser convertido em uma probabilidade p compreendida entre 0 e 1 por meio da Função Sigmoide:

$$p = \frac{1}{1 + e^{-z}}$$

Crie uma função em Python chamada `calcular_probabilidade_sigmoide(z)` que utilize a biblioteca nativa `math` para calcular a constante e , converta uma lista de valores z em

probabilidades e aplique o limiar de decisão padrão de 0,5 para classificar o resultado em 0 (Classe Negativa) ou 1 (Classe Positiva).

EXERCÍCIO PRÁTICO 3: Calculadora de Razão de Chances (Odds) e Logit

Cenário: Um modelo de Regressão Logística estimou que a probabilidade de um cliente cancelar o contrato (Churn) é de $p = 0,80$ (80%).

Escreva um script em Python que receba um valor de probabilidade p e calcule:

1. A Chance (Odds) de ocorrência do evento $\text{Odds} = \frac{p}{1-p}$
2. A transformação Logit (logaritmo natural do Odds): $\text{Logit}(p) = \ln(\text{Odds})$

EXERCÍCIO PRÁTICO 4: Gerador de Matriz de Confusão e Cálculo de Métricas (Acurácia, Precisão e Sensibilidade)

Cenário: Você aplicou um modelo de Regressão Logística para detectar transações fraudulentas em um lote de testes com 10 clientes. Você possui a lista com os rótulos reais (Y) e a lista com as previsões geradas pelo modelo ($\hat{Y}$):

Rótulos Reais (Y): $[1, 0, 1, 1, 0, 0, 1, 0, 0, 1]$

Previsões do Modelo ($\hat{Y}$): $[1, 0, 1, 0, 0, 1, 1, 0, 0, 1]$

Escreva uma rotina em Python que compare as duas listas, contabilize os Verdadeiros Positivos (VP), Falsos Positivos (FP), Verdadeiros Negativos (VN) e Falsos Negativos (FN), e imprima as métricas de Acurácia, Precisão e Sensibilidade (Recall).

Unidade 4: Estruturas de Repetição e Árvores de Decisão

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Na Unidade 3, compreendemos como as estruturas condicionais permitem que um algoritmo tome decisões locais e como a Regressão Logística modela probabilidades para problemas de classificação binária. Contudo, no ecossistema de Big Data, a quantidade de dados a ser processada é massiva. Não basta avaliar uma regra de decisão apenas uma vez; precisamos automatizar a repetição dessa avaliação sobre milhões de registros de forma eficiente. Além disso, muitas decisões corporativas exigem modelos preditivos que sejam altamente interpretáveis e baseados em regras visuais fáceis de explicar a gestores e stakeholders não técnicos.

Como destacam Soubhia et al. (2019), o domínio dos laços de repetição em Python é o pilar fundamental para a automação de varreduras em grandes bases de dados. Matthes (2016) assinala que a combinação de laços com estruturas condicionais permite transformar tarefas manuais e repetitivas em rotinas computacionais de alta performance. Paralelamente, Borges (2009) ressalta a versatilidade do Python no suporte a algoritmos não paramétricos, como as Árvores de Decisão, que traduzem a complexidade de um conjunto de dados em fluxogramas intuitivos de regras condicionais.

Nesta unidade, você aprenderá a dominar os laços de repetição `for` e `while` em Python, entenderá a arquitetura e os critérios de divisão de uma Árvore de Decisão e aprenderá a utilizar essa técnica para classificação de grupos e geração de tabelas de avaliação.

2. ESTRUTURAS DE REPETIÇÃO EM PYTHON (`for` E `while`)

Para processar coleções de dados, o Python oferece duas estruturas principais de repetição (ou laços/ loops).

2.1. O Laço `for` (Iteração Determinada)

O laço `for` é utilizado quando sabemos previamente a quantidade de iterações ou quando desejamos percorrer sequencialmente cada elemento de uma estrutura de dados (como listas, tuplas, dicionários ou arquivos).

```
# Iterando sobre uma lista de transações para aplicar uma regra de verificação
transacoes = [150.0, 2500.0, 80.0, 12000.0, 310.0]

for valor in transacoes:
    if valor > 10000.0:
        print(f"Alerta: Transação de R$ {valor:.2f} marcada para auditoria!")
    else:
        print(f"Transação de R$ {valor:.2f} aprovada.")
```

A Função range():

Muito utilizada com o for para gerar sequências numéricas. Por exemplo, range(0, 10, 2) gera os números inteiros de 0 a 8 pulando de 2 em 2.

2.2. O Laço while (Iteração Indeterminada)

O laço **while** executa um bloco de código enquanto uma determinada condição lógica for verdadeira. É ideal para cenários em que não sabemos exatamente quantas vezes o processo se repetirá, dependendo do atingimento de um critério de parada.

```
# Simulação de processamento de lote de dados até atingir a capacidade máxima
registro_atual = 0
capacidade_maxima = 5

while registro_atual < capacidade_maxima:
    registro_atual += 1
    print(f"Processando registro {registro_atual} de {capacidade_maxima}...")

print("Lote concluído com sucesso!")
```

2.3. Controle de Fluxo: break e continue

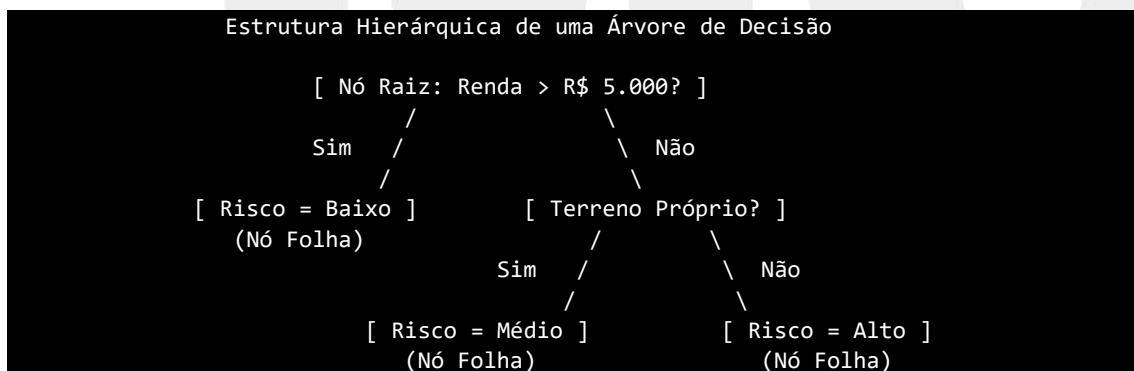
- **break:** Interrompe e encerra imediatamente a execução do laço de repetição.
- **continue:** Interrompe a iteração atual e salta diretamente para a próxima iteração do laço.

3. TÉCNICA DE CLASSIFICAÇÃO: ÁRVORES DE DECISÃO

A Árvore de Decisão é um método de **aprendizado de máquina supervisionado, não paramétrico**, utilizado tanto para **classificação** quanto para **regressão**. Seu funcionamento lembra a estrutura de um fluxograma, dividindo sequencialmente o conjunto de dados com base nas variáveis que oferecem a melhor separação entre as classes.

3.1. Anatomia de uma Árvore de Decisão

- **Nó Raiz (Root Node):** O ponto de partida da árvore, contendo todo o conjunto de dados antes de qualquer divisão.
- **Nós Internos / Decisão (Internal Nodes):** Pontos em que uma pergunta condicional é feita sobre uma característica (X_j). Os dados são ramificados com base na resposta.
- **Ramos (Branches):** O caminho lógico correspondente ao resultado do teste condicional.
- **Nós Folha (Leaf / Terminal Nodes):** Os nós finais que não se dividem mais. Representam a decisão final do modelo (a classe prevista Y).



4. CRITÉRIOS DE DIVISÃO E OBTENÇÃO DA ÁRVORE

Como a Árvore de Decisão escolhe qual variável deve ser colocada no Nó Raiz ou nos nós de decisão? O algoritmo busca maximizar a pureza dos nós resultantes, utilizando métricas de impureza estatística.

4.1. Impureza de Gini (Gini Impurity)

Mede a probabilidade de um elemento selecionado aleatoriamente no nó ser classificado incorretamente segundo a distribuição das classes no nó. O valor varia de 0 (nó perfeitamente puro, todas as observações pertencem à mesma classe) até 0,5 (em problemas binários perfeitamente divididos em 50%/50%).

$$I_G(t) = 1 - \sum_{i=1}^k (p_i)^2$$

Onde p_i é a proporção de observações pertencentes à classe i no nó t .

4.2. Entropia e Ganho de Informação (Information Gain)

Baseado na Teoria da Informação, a Entropia mede a desordem ou incerteza de um nó:

$$H(t) = - \sum_{i=1}^k p_i \log_2(p_i)$$

O **Ganho de Informação** é a redução esperada na entropia após a divisão dos dados por uma variável X . O algoritmo escolhe a variável que proporciona o maior Ganho de Informação.

4.3. Prevenção de Sobreajuste (Overfitting)

Se deixarmos a árvore crescer indefinidamente sem limites, ela criará regras específicas para cada dado da amostra de treino, memorizando o ruído dos dados e falhando ao tentar prever novos casos (overfitting). Para evitar isso, utilizamos técnicas de Poda (Pruning) e definição de hiperparâmetros:

- Profundidade máxima da árvore (max_depth).
- Número mínimo de amostras por nó folha (min_samples_leaf).

5. TABELA DE CLASSIFICAÇÃO E ROTULAGEM DE GRUPOS

Após o treinamento da Árvore de Decisão, os novos registros percorrem os ramos do nó raiz até atingirem um nó folha. O desempenho é avaliado cruzando os rótulos reais com as classificações geradas pela árvore por meio da Tabela de Classificação (Matriz de Confusão), permitindo extrair métricas de Acurácia, Precisão e Sensibilidade.

EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

No pré-processamento de um conjunto de dados para análise de risco de crédito, um engenheiro de dados escreveu o seguinte script utilizando um laço de repetição em linguagem Python (BORGES, 2009; MATTHES, 2016):

```
saldos = [1200, -300, 4500, -150, 8000]
saldos_positivos = []

for saldo in saldos:
    if saldo < 0:
        continue
    saldos_positivos.append(saldo)

print(len(saldos_positivos))
```

Após a execução desse código, qual será a saída impressa no terminal?

- A) 5
- B) 3
- C) 2
- D) 0
- E) O código apresentará um erro de sintaxe por uso incorreto do comando continue.

QUESTÃO 2

Uma instituição financeira utiliza um modelo de Árvore de Decisão para classificar se um cliente potencial deve receber uma oferta de empréstimo pré-aprovado ($Y = 1$) ou não ($Y = 0$). Uma das principais razões para a escolha das Árvores de Decisão em detrimento de modelos baseados em redes neurais profundas foi a explicabilidade do modelo perante órgãos reguladores.

A respeito da estrutura e funcionamento das Árvores de Decisão (SOUBHIA et al., 2019), avalie as asserções a seguir e a relação proposta entre elas:

I. As Árvores de Decisão são consideradas modelos "caixa-branca", pois as decisões tomadas pelo algoritmo podem ser traduzidas diretamente em uma sequência reproduzível de regras lógicas condicionais do tipo if-then-else.

PORQUE

II. O modelo realiza divisões sucessivas do espaço de dados selecionando, em cada nó de decisão, a variável explicativa e o ponto de corte que promovem a maior redução na impureza dos dados.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 3

Um cientista de dados está avaliando dois nós candidatos para a construção do nó raiz de uma Árvore de Decisão binária (K=2 classes: Fraude e Legítimo):

- Nó A: Contém 50 transações fraudulentas e 50 transações legítimas ($p_1 = 0,5$ e $p_2 = 0,5$).
- Nó B: Contém 100 transações legítimas e 0 transações fraudulentas ($p_1 = 1,0$ e $p_2 = 0,0$).

Considerando o cálculo da Impureza de Gini ($I_G = 1 - \sum p_i^2$) para avaliação de nós em Árvores de Decisão (SOUBHIA et al., 2019), assinale a alternativa correta referente aos valores de impureza dos Nós A e B, respectivamente.

- A) Nó A = 0,50; Nó B = 0,00 (Nó B é um nó puro).
- B) Nó A = 0,00; Nó B = 1,00 (Nó A é um nó puro).
- C) Nó A = 1,00; Nó B = 0,50.
- D) Nó A = 0,25; Nó B = 0,25.
- E) Nó A = 0,75; Nó B = 0,10.

QUESTÃO

Ao treinar um modelo de Árvore de Decisão sem impor limites para o parâmetro de profundidade máxima (max_depth = None), um analista observou que o modelo atingiu

uma Acurácia de 100% no conjunto de treino, mas obteve uma Acurácia de apenas 62% no conjunto de teste.

Esse comportamento discrepante é característico do fenômeno de:

- A) Subajuste (Underfitting), e pode ser corrigido aumentando a complexidade da árvore.
- B) Sobreajuste (Overfitting), e pode ser mitigado aplicando técnicas de poda ou limitando a profundidade da árvore.
- C) Erro de arredondamento de ponto flutuante na linguagem Python.
- D) Ausência de correlação entre as variáveis de entrada.
- E) Homocedasticidade perfeita dos resíduos de classificação.

QUESTÃO 5

Analise a estrutura de um laço while desenvolvido para simular o reprocessamento de mensagens com falha em uma fila de mensagens de Big Data (BORGES, 2009; MATTHES, 2016):

```
tentativas = 0
sucesso = False

while tentativas < 3 and not sucesso:
    tentativas += 1
    if tentativas == 2:
        sucesso = True

print(f"Tentativas efetuadas: {tentativas}")
```

Qual valor será impresso no terminal ao término da execução do script?

- A) Tentativas efetuadas: 1
- B) Tentativas efetuadas: 2
- C) Tentativas efetuadas: 3
- D) Tentativas efetuadas: 0
- E) O programa entrará em um laço infinito sem exibir resposta.

QUESTÃO 6

Na Teoria da Informação para a construção de Árvores de Decisão, a medida que quantifica o nível de desordem ou incerteza em um conjunto de dados é denominada:

- A) Coeficiente de Determinação (R^2).

- B) Entropia.
- C) Variância Amostral.
- D) Covariância Positiva.
- E) Intervalo Interquartil (IQR).

QUESTÃO 7

Uma empresa de telecomunicações utilizou uma Árvore de Decisão para identificar clientes propensos a cancelar o plano (Churn). O nó folha final de uma das ramificações principais gerou a seguinte regra:

SE (Tempo_de_Contrato < 12 meses) E (Reclamações_Abertas > 3) ENTÃO Churn = SIM

Ao aplicar essa regra a um grupo de 200 clientes que atendiam exatamente a esses critérios, observou-se que 180 clientes realmente cancelaram o plano e 20 continuaram ativos.

A proporção de pureza da classe de interesse (Churn = SIM) nesse nó folha específico é de:

- A) 10%
- B) 50%
- C) 80%
- D) 90%
- E) 100%

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Automação de Varredura de Logs com Laço for e Tratamento Condicional

Cenário: Você é o engenheiro responsável pelo monitoramento dos servidores de uma plataforma web. Uma lista de dicionários contém o histórico de acessos dos usuários, incluindo o endereço IP, o código de status HTTP retornado e o tempo de resposta (em milissegundos).

Escreva um script em Python que percorra essa lista de logs com um laço for, filtre apenas as requisições com falha (código HTTP ≥ 400) ou tempo de resposta excessivo (>500 ms), e gere um relatório dos IPs suspeitos.

EXERCÍCIO PRÁTICO 2: Calculadora Nativa de Impureza de Gini

Cenário: Para entender o funcionamento interno de uma Árvore de Decisão, crie uma função Python chamada `calcular_gini` que receba uma lista de rótulos de classe (por exemplo, ['Fraude', 'Legítimo', 'Fraude', 'Fraude']) e retorne o índice de Impureza de Gini do nó.

EXERCÍCIO PRÁTICO 3: Simulador de Árvore de Decisão Simples com if-elif-else

Cenário: Implemente a lógica inferencial de uma Árvore de Decisão previamente treinada para conceder ou negar limite de cartão de crédito de acordo com as seguintes regras:

1. Se Renda > 5000:

- Se `Historico_Inadimplencia == False`: Decisão = "Aprovado - Limite Alto"
- Senão: Decisão = "Aprovado - Limite Médio"

2. Senão (Renda ≤ 5000):

- Se `Score_Credito > 700`: Decisão = "Aprovado - Limite Inicial"
- Senão: Decisão = "Negado"

Crie uma função em Python e teste-a percorrendo uma lista com 3 novos clientes.

EXERCÍCIO PRÁTICO 4: Processamento Contínuo com while e Matriz de Confusão

Cenário: Um sistema de IA para detecção de anomalias processa imagens de uma linha de produção. Escreva um script com laço while que simule a leitura continuada de itens até que a fila esteja vazia, monte a Matriz de Confusão cruzando o rótulo real com o previsto e exiba a acurácia final

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Estudamos algoritmos de **Aprendizado de Máquina Supervisionado** (como Regressão Linear, Regressão Logística e Árvores de Decisão), onde cada registro da base de dados possuía um "rótulo" ou "resposta correta" para orientar o modelo. No entanto, na realidade do Big Data, a vasta maioria dos dados coletados diariamente — como hábitos de navegação, logs de sistemas e perfis de compras — chega às empresas de forma não rotulada. Nesses cenários, o desafio do cientista de dados é descobrir padrões ocultos, estruturas de similaridade e agrupamentos naturais diretamente nos dados.

Como destacam Soubhia et al. (2019), o **Aprendizado Não Supervisionado** amplia a capacidade analítica ao segmentar grandes volumes de dados sem a necessidade de rotulagem prévia. Matthes (2016) assinala que a implementação de algoritmos de agrupamento em Python exige o domínio de estruturas iterativas para o cálculo contínuo de distâncias e atualização de centroides. Já Borges (2009) enfatiza a eficiência do Python na manipulação de matrizes e vetores, permitindo processar agrupamentos complexos e gerar visualizações intuitivas, como dendogramas e gráficos de dispersão.

Nesta unidade, você compreenderá as diferenças entre o **Aprendizado Supervisionado e Não Supervisionado**, dominará o funcionamento dos Métodos de Agrupamento Hierárquico e do Algoritmo K-Médias (K-Means), aprenderá a interpretar Dendogramas e aplicará essas técnicas na segmentação prática de grupos em Big Data.

2. APRENDIZADO NÃO SUPERVISIONADO E O CONCEITO DE AGRUPAMENTO (CLUSTERING)

No Aprendizado Não Supervisionado, a base de dados consiste em um conjunto de variáveis de entrada (**X**) sem qualquer variável resposta correspondente (**Y**). O objetivo do algoritmo é explorar a estrutura dos dados para encontrar padrões significativos.

A principal tarefa do aprendizado não supervisionado é o **Agrupamento (Clustering)**, que consiste em organizar observações em grupos (chamados clusters) de modo que:

- **Alta Homogeneidade Intra-cluster:** Elementos dentro do mesmo grupo sejam os mais parecidos possíveis entre si.
- **Alta Heterogeneidade Inter-cluster:** Elementos pertencentes a grupos diferentes sejam os mais distintos possíveis entre si.



Exemplo Prático:

Uma plataforma de streaming (como a Netflix) deseja recomendar filmes para novos usuários. Em vez de criar regras manuais, um algoritmo de clustering lê o histórico de navegação de milhões de assinantes e descobre automaticamente 4 grupos distintos de hábitos, permitindo campanhas de marketing personalizadas para cada grupo.

3. MÉTODOS DE AGRUPAMENTO HIERÁRQUICO E DENDOGRAMAS

O Agrupamento Hierárquico constrói uma árvore ou hierarquia de grupos sequenciais, permitindo visualizar as relações de proximidade em diferentes níveis de detalhe.

3.1. Tipos de Agrupamento Hierárquico

- **Aglomerativo (Bottom-up / De baixo para cima):** Cada observação começa em seu próprio grupo individual (n grupos inicial). A cada passo, o algoritmo junta os dois grupos mais parecidos até que restem apenas um único grande grupo com todas as observações.
- **Divisivo (Top-down / De cima para baixo):** Todas as observações começam juntas em um único grande grupo. A cada passo, o grupo é dividido nos subgrupos mais distintos até que cada elemento fique isolado.

3.2. Medidas de Distância e Encadeamento (Linkage)

Para decidir quais grupos devem ser unidos, o algoritmo utiliza **métricas de distância** entre pontos:

Distância Euclidiana: A menor distância em linha reta entre dois pontos $A = (x_1, y_1)$ e $B = (x_2, y_2)$:

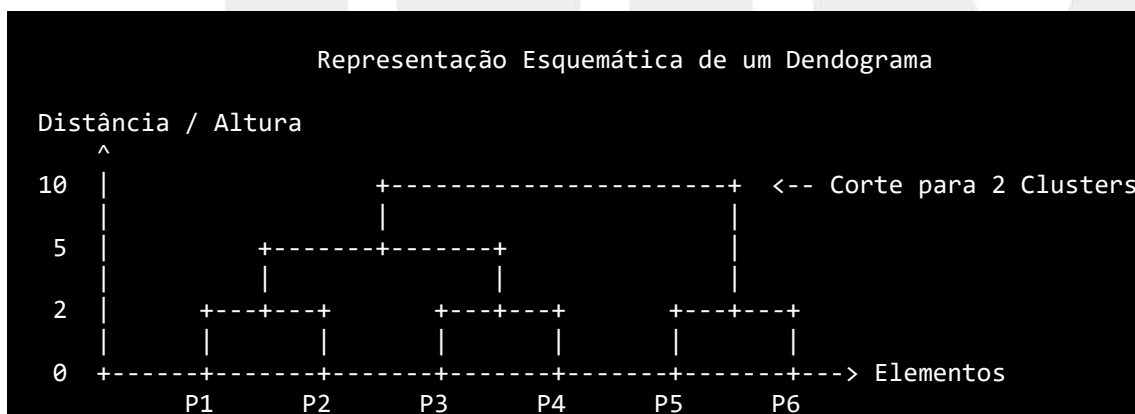
$$d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

CrITÉRIOS de Encadeamento (Linkage):

- **Single Linkage (Encadeamento Mínimo):** Considera a menor distância entre qualquer ponto do Grupo A e qualquer ponto do Grupo B.
- **Complete Linkage (Encadeamento Máximo):** Considera a maior distância entre qualquer ponto do Grupo A e qualquer ponto do Grupo B.
- **Average Linkage (Encadeamento Médio):** Considera a distância média entre todos os pares de pontos dos dois grupos.

3.3. O Dendograma

O **Dendograma** é um diagrama em formato de árvore invertida que ilustra as fusões e divisões realizadas pelo algoritmo hierárquico. O eixo vertical representa a distância ou altura de junção: quanto mais alta a barra horizontal que conecta dois grupos, maior é a diferença entre eles.



4. O ALGORITMO K-MÉDIAS (K-MEANS)

O **K-Means** é um dos algoritmos de agrupamento não hierárquicos mais populares e eficientes em Ciência de Dados devido à sua simplicidade e alta velocidade em bases volumosas de Big Data.

4.1. Passo a Passo de Funcionamento do K-Means

1. **Definição do K:** O usuário escolhe previamente o número fixo K de grupos desejados.
2. **Inicialização dos Centroides:** O algoritmo escolhe aleatoriamente K pontos no espaço para funcionarem como os centros iniciais dos grupos (chamados centroides).
3. **Atribuição:** Cada ponto da base de dados é associado ao centroide mais próximo (geralmente usando a Distância Euclidiana).
4. **Atualização dos Centroides:** Recalcula-se a posição de cada centroide obtendo a média aritmética das coordenadas de todos os pontos atribuídos àquele grupo.
5. **Convergência:** Os passos 3 e 4 são repetidos iterativamente até que os centroides não mudem mais de posição ou o número máximo de iterações seja atingido.

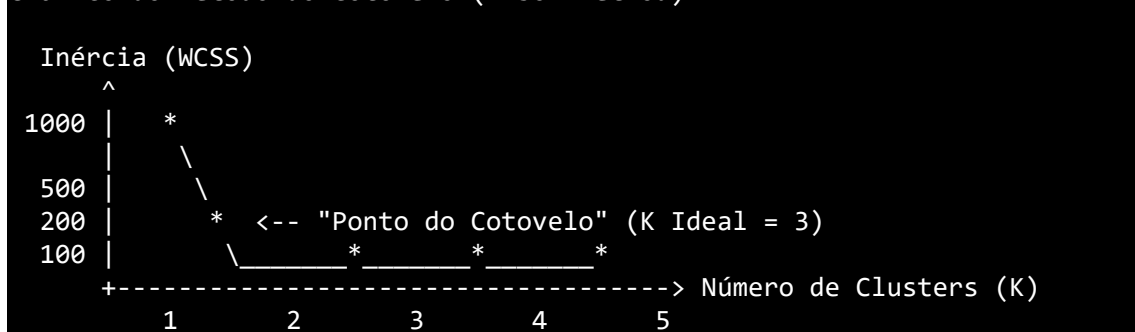
4.2. Definição do Valor Ideal de K: O Método do Cotovelo (Elbow Method)

Como definir o número certo de grupos (K) sem conhecer as respostas com antecedência?

Utilizamos a métrica de **Inércia** (ou Within-Cluster Sum of Squares - WCSS), que calcula a **soma das distâncias quadráticas de todos os pontos em relação ao centroide do seu próprio grupo**.

Ao plotar um gráfico do número K (eixo X) contra a **Inércia** (eixo Y), a curva cai acentuadamente e depois estabiliza. O ponto de dobra nessa curva — semelhante a um cotovelo — **indica o valor ideal de K**, equilibrando simplicidade e precisão.

Gráfico do Método do Cotovelo (Elbow Method)



4.3. Definição Z-Score (Escore-Z)

O **Z-Score** (ou Escore-Z / Valor Padronizado) é uma medida estatística que indica quantos desvios padrões um determinado dado está acima ou abaixo da **média do seu conjunto de dados**.

Ele é utilizado para colocar variáveis que estão em escalas completamente diferentes na mesma métrica comparável.

Fórmula Matemática:

$$Z = \frac{X - \bar{X}}{s}$$

Onde:

- **X:** O valor original do dado.
- **$\bar{X}$:** A média aritmética do conjunto de dados.
- **s:** O desvio padrão amostral.

Como Interpretar o Resultado do Z-Score?

- **Z = 0:** O valor do dado é exatamente igual à média do grupo.
- **Z > 0 (Positivo):** O valor está acima da média. Por exemplo, Z = +2,0 significa que o dado está 2 desvios padrões acima da média.
- **Z < 0 (Negativo):** O valor está abaixo da média. Por exemplo, Z = -1,5 significa que o dado está 1,5 desvio padrão abaixo da média.

Por que ele é fundamental em Ciência de Dados e Big Data?

Imagine que você está analisando uma base de clientes com duas variáveis:

1. **Salário:** Varia de R\$ 1.500,00 a R\$ 50.000,00.
2. **Idade:** Varia de 18 a 80 anos.

Se você aplicar um algoritmo baseado em distância (como o K-Means ou o KNN), os valores de salário (na casa dos milhares) vão dominar totalmente os cálculos espaciais, fazendo com que a idade seja ignorada pelo algoritmo.

Ao converter ambas as variáveis para Z-Score:

- A média de ambas passa a ser **0**.
- O desvio padrão de ambas passa a ser **1**.

Dessa forma, o algoritmo consegue comparar "laranjas com laranjas", garantindo que todas as variáveis tenham o mesmo peso no modelo preditivo.

4.4. Exemplos em Python

Nesta seção, vamos gerar dados sintéticos em Python para visualizar o funcionamento completo dos algoritmos em um ambiente computacional real.

Exemplo Prático A: Execução do K-Means e Visualização do Método do Cotovelo

Neste exemplo, geramos uma base sintética contendo o perfil de 100 clientes (com base em "Idade" e "Pontuação de Gastos") e aplicamos o K-Means calculando a inércia para encontrar o K ideal.

```
import math

# 1. Geração de Dados Sintéticos de Clientes (Idade, Pontuação_Gastos)
# Simulando 3 grupos naturais de clientes
dados_sinteticos = [
    # Grupo 1: Jovens com Gastos Altos
    (22, 85), (25, 90), (20, 78), (28, 88), (23, 95), (21, 82), (24, 80),
    # Grupo 2: Adultos de Meia-Idade com Gastos Moderados
    (45, 40), (50, 42), (52, 55), (48, 48), (42, 38), (55, 50), (49, 45),
    # Grupo 3: Idosos com Gastos Baixos
    (68, 12), (72, 18), (65, 10), (70, 15), (75, 20), (62, 8), (69, 14)
]

def calcular_distancia(p1, p2):
    """Calcula a Distância Euclidiana entre dois pontos 2D."""
    return math.sqrt((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)

def executar_kmeans_sintetico(dados, k, max_iter=10):
    """Executa o algoritmo K-Means e retorna os centroides e a Inércia (WCSS)."""
    # Inicializa os K primeiros pontos como centroides
    centroides = dados[:k]

    for _ in range(max_iter):
        # Etapa de Atribuição
        clusters = [[] for _ in range(k)]
        for ponto in dados:
            distancias = [calcular_distancia(ponto, c) for c in centroides]
            idx_menor = distancias.index(min(distancias))
            clusters[idx_menor].append(ponto)

        # Etapa de Atualização dos Centroides
        novos_centroides = []
        for cluster in clusters:
            if len(cluster) > 0:
                media_x = sum(p[0] for p in cluster) / len(cluster)
                media_y = sum(p[1] for p in cluster) / len(cluster)
                novos_centroides.append((media_x, media_y))
            else:
                novos_centroides.append((0, 0))
        centroides = novos_centroides

    # Cálculo da Inércia (Soma das Distâncias Quadráticas Intra-cluster)
    inercia = 0.0
    for i, cluster in enumerate(clusters):
        for ponto in cluster:
            inercia += calcular_distancia(ponto, centroides[i]) ** 2

    return k, inercia, centroides

# 2. Avaliação da Inércia para diferentes valores de K (Método do Cotovelo)
print("=== MÉTODO DO COTOVELO (ELBOW METHOD) - DADOS SINTÉTICOS ===")
print(f"{'Número de Clusters (K)':<25} | {'Inércia (WCSS)':<20}")
print("-" * 50)

for k_teste in range(1, 5):
    k, inercia, _ = executar_kmeans_sintetico(dados_sinteticos, k=k_teste)
    print(f"{'k':<25} | {'inercia:<20.2f}"))
```

Exemplo Prático B: Simulação de Agrupamento Hierárquico e Dendograma ASCII

Neste exemplo, utilizamos uma matriz de dados sintéticos de 5 servidores de TI e simulamos o encadeamento aglomerativo (Bottom-up) para exibir a fusão dos grupos de forma textual em Python.

```
# Dados Sintéticos de 5 Servidores: (Uso_CPU_%, Uso_RAM_%)
servidores = {
    "Servidor_1": (15, 20),
    "Servidor_2": (18, 22),
    "Servidor_3": (85, 90),
    "Servidor_4": (88, 95),
    "Servidor_5": (50, 50)
}

def distancia_euclidiana(p1, p2):
    return ((p1[0] - p2[0])**2 + (p1[1] - p2[1])**2)**0.5

print("=== SIMULAÇÃO DE DENDOGRAMA HIERÁRQUICO (AGLOMERATIVO) ===")
# Inicialmente, cada servidor é o seu próprio grupo
clusters_atuais = {nome: [coord] for nome, coord in servidores.items()}

passo = 1
while len(clusters_atuais) > 1:
    menor_dist = float('inf')
    par_para_fundir = None

    nomes = list(clusters_atuais.keys())
    # Encontra os dois clusters com menor distância média
    for i in range(len(nomes)):
        for j in range(i + 1, len(nomes)):
            nome1, nome2 = nomes[i], nomes[j]
            p1 = clusters_atuais[nome1][0]
            p2 = clusters_atuais[nome2][0]
            dist = distancia_euclidiana(p1, p2)

            if dist < menor_dist:
                menor_dist = dist
                par_para_fundir = (nome1, nome2)

    # Fusão dos dois clusters mais próximos
    c1, c2 = par_para_fundir
    novo_nome = f"({c1} + {c2})"
    novo_ponto = (
        (clusters_atuais[c1][0][0] + clusters_atuais[c2][0][0]) / 2,
        (clusters_atuais[c1][0][1] + clusters_atuais[c2][0][1]) / 2
    )

    print(f"Passo {passo}: Fundindo {c1} e {c2} | Distância de Junção = {menor_dist:.2f}")

    del clusters_atuais[c1]
    del clusters_atuais[c2]
    clusters_atuais[novo_nome] = [novo_ponto]
    passo += 1

print("\nAgrupamento Hierárquico Completo!")
```

5. EXEMPLOS PRÁTICOS PARA CONSOLIDAÇÃO DOS CONCEITOS

Para consolidar o aprendizado do agrupamento não supervisionado, analisaremos dois estudos de caso do mercado corporativo que exemplificam a tomada de decisão guiada por K-Means e Dendogramas.

Exemplo Prático 1: Segmentação de Clientes em E-Commerce (Aplicação com K-Means)

Cenário:

Uma grande plataforma de comércio eletrônico possui milhões de usuários ativos, mas não consegue oferecer promoções direcionadas porque a base de dados não contém rótulos de perfis. A equipe de dados decidiu aplicar o K-Means utilizando duas variáveis padronizadas (**Z-Score**):

1. **Recência (X_1):** Dias desde a última compra.
2. **Valor Monetário (X_2):** Total gasto no último ano (R\$).

Execução e Resultado do Modelo:

Após aplicar o Método do Cotovelo (Elbow Method), identificou-se **K = 3** como o **número ideal de clusters**. Os centroides finais recalculados pelo algoritmo revelaram três perfis bem definidos:

1. Cluster 0 — "Clientes Fiéis / VIP" (20% da base):

- **Centroide:** Recência muito baixa ($Z = -1,2$) e Valor gasto muito alto ($Z = +2,1$).
- **Ação de Negócio:** Inclusão automática em um programa exclusivo de recompensas com frete grátis e atendimento prioritário.

2. Cluster 1 — "Compradores Ocasionais" (55% da base):

- **Centroide:** Recência moderada ($Z = +0,1$) e Valor gasto médio ($Z = -0,2$).
- **Ação de Negócio:** Disparo de campanhas periódicas de recomendação baseadas em lançamentos de produtos.

3. Cluster 2 — "Clientes em Risco de Churn" (25% da base):

- **Centroide:** Recência muito alta ($Z = +1,8$) e Valor gasto baixo/médio ($Z = -0,5$).

- **Ação de Negócio:** Envio automático de cupons de desconto para reativação do cliente antes que ele abandone a plataforma.

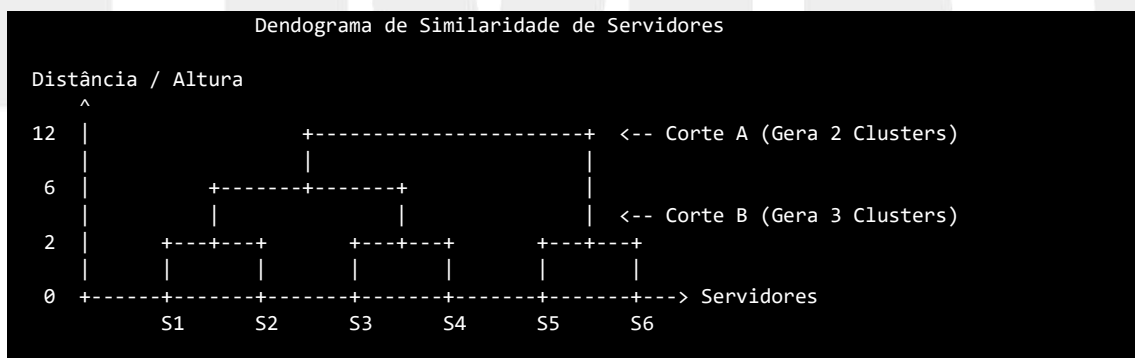
Exemplo Prático 2: Agrupamento de Tipos de Servidores para AIOps (Aplicação com Dendograma)

Cenário:

Em uma infraestrutura crítica de TI, uma equipe de engenharia de plataformas precisa identificar o **grau de similaridade** entre **6 servidores** de banco de dados ($S_1, S_2, S_3, S_4, S_5, S_6$) com base em métricas de **uso de CPU e memória RAM**, sem saber de antemão quantas categorias de carga existem.

Execução e Leitura do Dendograma:

Utilizando o Agrupamento Hierárquico Aglomerativo (Bottom-up) com distância euclidiana e encadeamento médio (Average Linkage), o algoritmo gerou a seguinte estrutura de árvore visual (Dendograma):



Tomada de Decisão com Base na Altura do Corte:

1. **Opção de Corte B (Altura = 4,0):** Se a equipe de TI optar por realizar um corte horizontal na altura 4,0, a linha seccionará três ramificações verticais, dividindo os servidores em 3 grupos operacionais:

- **Grupo 1 (S_1, S_2):** Servidores de Alta Performance (Carga Crítica).
- **Grupo 2 (S_3, S_4):** Servidores de Processamento em Lote (Batch Jobs).
- **Grupo 3 (S_5, S_6):** Servidores de Homologação / Ambientes de Teste.

2. **Aplicação Prática:** A partir desse agrupamento visual, as políticas de automação (AIOps) aplicam régulas de contingência e escalamento de memória diferenciadas para cada grupo, garantindo eficiência de custos e alta disponibilidade do ambiente.

RESUMO

INFOGRÁFICO: AGRUPAMENTO DE DADOS (CLUSTERING E DENDOGRAMAS)

1. APRENDIZADO NÃO SUPERVISIONADO

- Dados **NÃO** rotulados (não existe variável de resposta Y).
- Objetivo: Encontrar padrões, estruturas ocultas e agrupamentos naturais.
- Regra do Agrupamento (Clustering):
 - [+] Alta Homogeneidade Intra-cluster (membros do mesmo grupo são parecidos)
 - [+] Alta Heterogeneidade Inter-cluster (grupos diferentes são distintos)



2. METRICAS ESPACIAIS E PRÉ-PROCESSAMENTO

A. DISTÂNCIA EUCLIDIANA

Mede a menor distância em linha reta entre dois pontos A e B:
 $d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

B. PADRONIZAÇÃO (Z-SCORE)

Equaliza variáveis em escalas diferentes (ex: Salário vs. Idade).

$$Z = (X - \bar{X}) / s$$

- $Z = 0$ (Na média) | $Z > 0$ (Acima da média) | $Z < 0$ (Abaixo da média)



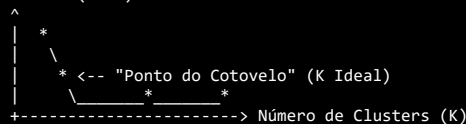
3. ALGORITMO K-MEANS (NÃO HIERÁRQUICO)

FLUXO ITERATIVO:

- [1] Escolher o número K de clusters.
- [2] Inicializar K centroides aleatoriamente.
- [3] Atribuir cada ponto ao centroide mais próximo (via Dist. Euclidiana).
- [4] Recalcular a posição dos centroides (média das coordenadas do grupo).
- [5] Repetir [3] e [4] até a CONVERGÊNCIA (centroides não mudam de lugar).

SELEÇÃO DO K IDEAL -> MÉTODO DO COTOVELO (ELBOW METHOD):

Inércia (WCSS)



4. AGRUPAMENTO HIERÁRQUICO E DENDOGRAMAS

TIPOS:

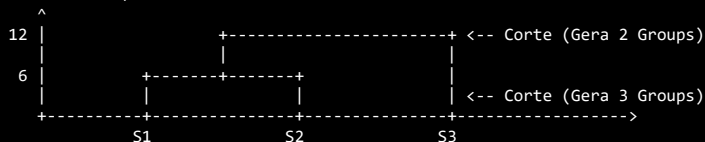
- Aglomerativo (Bottom-up): Começa com N grupos e vai fundindo até restar 1
- Divisivo (Top-down): Começa com 1 grupo e vai dividindo até restar N

REGRAS DE ENCADEAMENTO (LINKAGE):

- Single Linkage: Menor distância entre pontos dos grupos.
- Complete Linkage: Maior distância entre pontos dos grupos.
- Average Linkage: Distância média entre todos os pares dos grupos.

ESTRUTURA DO DENDOGRAMA:

Distância / Altura



5. APLICAÇÕES PRÁTICAS NO MERCADO DE DATA SCIENCE

- E-Commerce: Segmentação RFM de clientes (Fiéis, Ocasionais, Churn).
- AIOps & TI: Agrupamento de servidores por perfil de CPU e Memória RAM.
- Marketing: Recomendação personalizada de produtos e serviços.

EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

Um analista de dados recebeu um arquivo contendo o histórico de compras de 50.000 clientes, contendo atributos como valor total gasto, frequência de visitas e número de itens devolvidos. A diretoria solicitou a criação de perfis de compradores para campanhas personalizadas, mas informou que não existe nenhuma categoria ou rótulo pré-definido no banco de dados.

Considerando os conceitos de Aprendizado de Máquina (SOUBHIA et al., 2019), assinale a alternativa correta referente à abordagem e à técnica mais adequadas para solucionar essa demanda.

- A) Aprendizado Supervisionado, utilizando o algoritmo de Regressão Logística Binária.
- B) Aprendizado Não Supervisionado, utilizando a técnica de Agrupamento (Clustering) via K-Means.
- C) Aprendizado Supervisionado, utilizando Árvores de Decisão para prever valores contínuos.
- D) Aprendizado Por Reforço, utilizando redes neurais profundas para calcular a média amostral.
- E) Análise de Regressão Linear Simples para estimar a mediana dos clientes.

QUESTÃO 2

No processo iterativo do algoritmo K-Means aplicado à segmentação de mercado, a posição dos centroides é atualizada a cada ciclo de execução.

A respeito do ciclo de funcionamento e da convergência do algoritmo K-Means (BORGES, 2009; MATTHES, 2016), avalie as asserções a seguir e a relação proposta entre elas:

I. O algoritmo K-Means garante que o resultado final atingirá sempre o mínimo global absoluto da inércia, independentemente das posições iniciais escolhidas para os centroides.

PORQUE

II. Em cada iteração do K-Means, os novos centroides são recalculados obtendo-se o vetor de médias aritméticas de todas as observações atribuídas ao respectivo grupo.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 3

Considere o cálculo da Distância Euclidiana entre dois clientes A e B em um espaço bidimensional composto pelas variáveis X_1 (Renda anual, em milhares de reais) e X_2 (Idade, em anos):

- **Cliente A:** $X_1 = 10$, $X_2 = 20$
- **Cliente B:** $X_1 = 13$, $X_2 = 24$

Aplicando a fórmula da Distância Euclidiana $d(A, B) =$

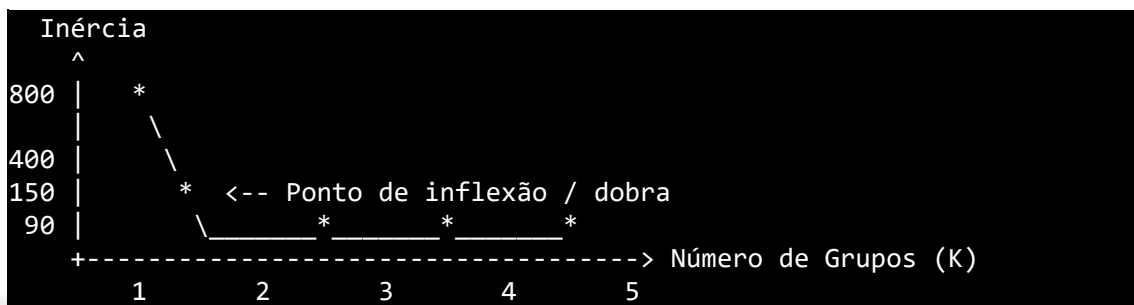
$$\sqrt{(X_{1B} - X_{1A})^2 + (X_{2B} - X_{2A})^2}$$

a distância entre os dois clientes é igual a:

- A) 3,0
- B) 4,0
- C) 5,0
- D) 7,0
- E) 25,0

QUESTÃO 4

Para definir a quantidade ideal de grupos em uma análise de agrupamento com K-Means, um cientista de dados aplicou o Método do Cotovelo (Elbow Method) e gerou o seguinte gráfico da Inércia (WCSS) em função de K:



Analizando o gráfico obtido, qual é o número ideal de clusters (K) a ser selecionado para o modelo e qual a justificativa técnica?

- A) K = 1, pois apresenta o maior valor de inércia e máxima complexidade.
- B) K = 2, pois representa o ponto de inflexão a partir do qual a redução da inércia se torna marginal.
- C) K = 5, pois quanto maior o número de grupos, menor é a inércia e melhor o modelo.
- D) K = 3, pois a inércia atinge o valor zero a partir desse ponto.
- E) O gráfico indica que o algoritmo K-Means não pode ser utilizado para esses dados.

QUESTÃO 5

Um especialista em bioinformática construiu um Dendograma a partir do agrupamento hierárquico aglomerativo de 6 amostras genéticas. Ao analisar a árvore resultante, ele realizou um corte horizontal na altura de distância igual a 8,0 unidades, dividindo o gráfico em três ramificações verticais disjuntas.

Essa operação de corte no dendograma permite ao analista:

- A) Transformar a técnica hierárquica em um modelo de Regressão Logística Binária.
- B) Determinar a acurácia global da Matriz de Confusão para bases desbalanceadas.
- C) Definir a partição do conjunto de dados em exatamente 3 grupos (clusters) distintos.
- D) Eliminar os outliers do conjunto de dados sem alterar a quantidade de grupos.
- E) Calcular o coeficiente angular β_1 pelo método dos Mínimos Quadrados.

QUESTÃO 6

No agrupamento hierárquico, a escolha do critério de encadeamento (linkage) define a regra de medição de proximidade entre dois grupos contendo múltiplas observações.

Assinale a opção que define corretamente o critério de Encadeamento Máximo (Complete Linkage).

- A) É a menor distância encontrada entre qualquer ponto do Grupo A e qualquer ponto do Grupo B.
- B) É a maior distância encontrada entre qualquer ponto do Grupo A e qualquer ponto do Grupo B.
- C) É a média aritmética de todas as distâncias combinadas entre os pontos do Grupo A e do Grupo B.
- D) É a distância entre os centroides médios calculados para cada um dos dois grupos.
- E) É o somatório de todas as variâncias amostrais divididas pelo número de observações minus um.

QUESTÃO 7

Antes de aplicar o algoritmo K-Means a uma base de dados contendo as variáveis "Salário Mensal" (variando entre R\$ 1.200,00 e R\$ 80.000,00) e "Idade do Cliente" (variando entre 18 e 80 anos), o engenheiro de dados realizou a Padronização dos Dados (conversão para Z-Score).

A razão fundamental para realizar a padronização das variáveis antes de executar o K-Means é:

- A) Garantir que a variável dependente assuma apenas valores binários 0 ou 1.
- B) Evitar que variáveis com ordens de grandeza muito maiores (como o Salário) dominem o cálculo da Distância Euclidiana em detrimento de variáveis com escalas menores.
- C) Eliminar a necessidade de definir o número K de grupos no algoritmo.
- D) Garantir que o algoritmo gere um dendograma perfeitamente simétrico.
- E) Converter o Aprendizado Não Supervisionado em Aprendizado Supervisionado.

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Calculadora Nativa de Distância Euclidiana Entre Pontos

Cenário:

Para entender a métrica espacial usada no K-Means, crie uma função em Python chamada `calcular_distancia_euclidiana(p1, p2)` que receba duas listas ou tuplas representando as coordenadas de dois clientes em n dimensões e retorne a Distância Euclidiana exata entre eles.

EXERCÍCIO PRÁTICO 2: Algoritmo Atribuidor de Grupos (Etapa de Atribuição do K-Means)

Cenário:

Dada uma lista de coordenadas de novos clientes e a posição atual de 2 centroides, crie um script em Python que percorra cada cliente, calcule a distância em relação aos centroides e atribua o cliente ao grupo do centroide mais próximo.

EXERCÍCIO PRÁTICO 3: Recálculo dos Centroides (Etapa de Atualização do K-Means)

Cenário:

Após a atribuição dos pontos, o algoritmo K-Means recalcula a posição de cada centroide obtendo a média das coordenadas dos membros do grupo. Crie um script em Python que receba uma lista de pontos atribuídos a um determinado grupo e retorne as novas coordenadas do centroide.

EXERCÍCIO PRÁTICO 4: Padronizador de Dados Nativo (Z-Score Normalizer)

Cenário:

Para preparar os dados antes do agrupamento por K-Means, escreva uma rotina em Python que receba uma lista numérica (ex: salários), calcule a média e o desvio padrão amostral e converta cada número para o seu respectivo valor padronizado (Z-Score):

$$Z = \frac{X - \bar{X}}{s}$$

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Com base na nova estrutura curricular da nossa apostila, esta unidade é dedicada ao **Analytics Avançado**, concentrando-se no processamento de dados não estruturados em texto e na descoberta de padrões de coocorrência em transações.

No ambiente corporativo contemporâneo, estima-se que mais de **80% das informações armazenadas sejam não estruturadas** — incluindo avaliações de e-commerce, posts em redes sociais, chamados de suporte em texto e registros de transações de compra de clientes. Como ressaltam Soubhia et al. (2019), a extração de inteligência a partir desses dados brutos exige pipelines específicos de pré-processamento para **transformar conteúdos não estruturados em estruturas tabulares compreensíveis** por algoritmos de Machine Learning.

Matthes (2016) assinala que o ecossistema Python oferece flexibilidade para manipular grandes volumes de dados de texto e coleções transacionais. Complementarmente, Borges (2009) destaca que o uso eficiente de bibliotecas e estruturas de dados nativas permite aplicar métricas probabilísticas sobre grandes matrizes transacionais de forma rápida e escalável.

Nesta unidade, exploraremos os fundamentos de Mineração de Texto (Text Mining), Análise de Sentimentos, Análise de Cesto de Compras (Market Basket Analysis) e o algoritmo Apriori para geração de Regras de Associação.

2. MINERAÇÃO DE TEXTO (TEXT MINING) E ANÁLISE DE SENTIMENTOS

A Mineração de Texto consiste no conjunto de técnicas e algoritmos aplicados para extrair **padrões, estruturas e informações de alto nível contidas** em documentos de texto não estruturados.

PIPELINE DE MINERAÇÃO DE TEXTO

```
[ Texto Bruto ] ---> [ Limpeza / Padronização ] ---> [ Tokenização / Stopwords ]
                                     |
[ Decisão / Insights ] <--- [ Modelo / Análise ] <--- [ Vetorização (TF-IDF) ]
```

2.1. Etapas do Pré-Processamento de Texto

Antes de aplicar qualquer modelo matemático a um texto, é necessário higienizar a base de dados em linguagem natural:

4. LEMATIZAÇÃO / RADICALIZAÇÃO (STEMMING)

- Ação: Redução das palavras flexionadas à sua forma raiz ou canônica.
- O que faz: Transforma verbos no infinitivo e substantivos/adjetivos no singular, unificando variações gramaticais.

[Entrada] : ['produto', 'excelente', 'comprando', '2', 'unidades']

|

[Saída] : ['produto', 'excelente', 'comprar', '2', 'unidade']

|

=====

[VETOR PRONTO PARA MACHINE LEARNING / TF-IDF]:

['produto', 'excelente', 'comprar', '2', 'unidade']

=====

Resumo dos Exemplos Práticos por Etapa:

Etapas	Texto de Entrada	Texto de Saída / Resultado
1. Normalização	"O produto é EXCELENTE!"	"o produto e excelente"
2. Tokenização	"o produto e excelente"	['o', 'produto', 'e', 'excelente']
3. Stopwords	['o', 'produto', 'e', 'excelente']	['produto', 'excelente']
4. Lematização	['comprando', 'compras', 'comprado']	['comprar', 'comprar', 'comprar']

2.2. Vetorização de Texto: Modelo Bag of Words e TF-IDF

Para que um algoritmo numérico consiga processar textos, estes devem ser convertidos em matrizes numéricas.

- **Saco de Palavras (Bag of Words — BoW):** Representação em que cada documento é transformado em um vetor contendo a contagem simples de frequência de cada palavra presente no vocabulário.
- **TF-IDF (Term Frequency-Inverse Document Frequency):** Métrica estatística que avalia a importância de uma palavra em um documento específico em relação a todo o corpus de documentos.
 - **TF (Frequência do Termo):** Mede a frequência com que o termo aparece em um documento específico.
 - **IDF (Frequência Inversa no Documento):** Penaliza palavras que aparecem em quase todos os documentos, destacando termos raros e específicos.

$$TF\text{-}IDF(t, d, D) = TF(t, d) \times IDF(t, D)$$

Exemplos Detalhados de Vetorização de Texto

Para ilustrar ambos os modelos, considere um pequeno corpus sintético composto por 2 documentos (frases após limpeza e pré-processamento):

- Documento 1 (D_1): "dados analise dados"
- Documento 2 (D_2): "python analise codigo"

Vocabulário único do corpus (4 termos): ["analise", "codigo", "dados", "python"].

Exemplo do Modelo Saco de Palavras (Bag of Words — BoW)

No modelo Saco de Palavras, cada documento é representado por um vetor onde a posição corresponde ao termo do vocabulário e o valor é a frequência absoluta (contagem direta) de vezes que a palavra aparece no documento.

- Construção da Matriz Termo-Documento (BoW):

Documento	"analise"	"codigo"	"dados"	"python"
Documento 1 (D_1)	1	0	2	0
Documento 2 (D_2)	1	1	0	1

Vetores Resultantes:

- $V_{BoW}(D_1) = [1, 0, 2, 0]$
- $V_{BoW}(D_2) = [1, 1, 0, 1]$

Interpretação: A palavra "dados" aparece **2 vezes** no **Documento 1** e **0 vezes** no **Documento 2**. O modelo ignora a ordem gramatical e foca exclusivamente na contagem.

Exemplo do Modelo TF-IDF (Term Frequency-Inverse Document Frequency)

Diferente do BoW, o TF-IDF pondera a frequência da palavra (TF) penalizando termos que aparecem em muitos documentos (IDF), destacando apenas as palavras mais informativas de cada documento.

Considere o cálculo para a palavra "dados" no Documento 1 (D_1):

1. **Cálculo da Frequência do Termo (TF):** Proporção da palavra no documento específico.

$$TF("dados", D_1) = \frac{\text{Ocorrências de "dados" em } D_1}{\text{Total de palavras em } D_1} = \frac{2}{3} \approx 0,67$$

2. **Cálculo da Frequência Inversa no Documento (IDF):** Mede o quão rara a palavra é em todo o corpus de 2 documentos. Usando a fórmula logarítmica padrão $IDF(t) = \ln\left(\frac{N}{n_t}\right)$, onde $N=2$ (total de documentos) e $n_t=1$ (documentos que contêm "dados"):

$$IDF("dados", D) = \ln\left(\frac{2}{1}\right) = \ln(2) \approx 0,693$$

3. **Cálculo Final do TF-IDF:**

$$TF-IDF\{"dados"\}, D_1, D) = TF \times ID = 0,67 \times 0,693 \approx 0,464$$

Comparação Prática entre Palavras no TF-IDF:

- Para a palavra "analise" no Documento 1:

Ela aparece em ambos os documentos (D_1 e D_2).

Seu IDF = $\ln(2/2) = \ln(1) = 0$.

Seu TF-IDF = $0,33 \times 0 = 0,000$.

Explicação: Como a palavra "analise" aparece em todos os documentos, o TF-IDF entende que ela é comum e não serve para diferenciar os textos.

- Para a palavra "dados" no Documento 1:

Seu TF-IDF = 0,464.

Explicação: Como ela é frequente no Documento 1 e rara no restante do corpus, ela ganha um peso elevado.

Matriz Resultante TF-IDF (Ponderada):

Documento	"analise"	"codigo"	"dados"	"python"
Documento 1 (D_1)	0,000	0,000	0,464	0,000
Documento 2 (D_2)	0,000	0,231	0,000	0,231

2.3. Análise de Sentimentos

A Análise de Sentimentos (ou mineração de opinião) utiliza algoritmos de classificação para identificar o tom emocional por trás de um texto. O objetivo é categorizar o conteúdo em polaridades (como Positivo, Neutro ou Negativo) ou em níveis contínuos de satisfação do cliente.

3. ANÁLISE DE CESTO DE COMPRAS E REGRAS DE ASSOCIAÇÃO

A Análise de Cesto de Compras (Market Basket Analysis) é uma técnica de mineração de dados focada em descobrir quais produtos são frequentemente comprados de forma conjunta por clientes.

O resultado dessa análise é expresso em Regras de Associação na forma condicional:

SE {X} ENTÃO {Y} (ou $X \Rightarrow Y$)

Exemplo Clássico:

SE{Pão, Manteiga} ENTÃO {Leite}

3.1. As Três Métricas Fundamentais das Regras de Associação

Para avaliar o grau de relevância e a força de uma regra de associação, utilizamos três métricas probabilísticas essenciais:

MÉTRICAS DE REGRAS DE ASSOCIAÇÃO	
1. SUPORTE (Frequência Global de Coocorrência)	Fração de transações que contêm X e Y juntos.
↓	
2. CONFIANÇA (Força Condicional da Regra)	Dado que X foi comprado, qual a chance de Y?
↓	
3. LIFT (Independência / Poder Preditivo Real)	Mede quantas vezes a regra supera o acaso.

1. Suporte (Support)

Mede a proporção total de transações no banco de dados que contêm o conjunto de itens X e Y simultaneamente. Indica a popularidade geral da combinação.

$$\text{Suporte}(X \Rightarrow Y) = \frac{\text{Número de Transações contendo } X \text{ e } Y}{\text{Total de Transações no Banco de Dados}}$$

2. Confiança (Confidence)

Mede a probabilidade condicional de uma transação conter o item **Y**, dado que ela já contém o item **X**. Avalia a confiabilidade da regra.

$$\text{Confiança}(X \Rightarrow Y) = \frac{\text{Suporte}(X \cup Y)}{\text{Suporte}(X)} = \frac{\text{Transações com } X \text{ e } Y}{\text{Transações com } X}$$

3. Lift (Razão de Elevação)

Mede o grau de dependência entre **X** e **Y**, comparando a probabilidade conjunta da regra com a probabilidade teórica dos dois itens serem comprados de forma totalmente independente (ao acaso).

$$\text{Lift}(X \Rightarrow Y) = \frac{\text{Confiança}(X \Rightarrow Y)}{\text{Suporte}(Y)} = \frac{\text{Suporte}(X \cup Y)}{\text{Suporte}(X) \times \text{Suporte}(Y)}$$

Lift > 1: Indica uma associação positiva (a presença de **X** aumenta significativamente a chance de compra de **Y**).

Lift = 1: Indica que **X** e **Y** são independentes (a coocorrência ocorre meramente por acaso).

Lift < 1: Indica uma **associação negativa** (a presença de **X** inibe ou reduz a chance de compra de **Y**).

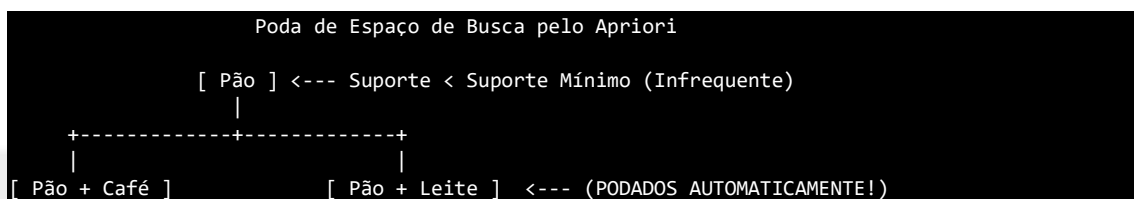
4. O ALGORITMO APRIORI

O **Algoritmo Apriori** é o método mais conhecido para **encontrar conjuntos de itens frequentes** e gerar regras de **associação eficientes** em grandes volumes de transações de Big Data.

4.1. O Princípio da Propriedade Apriori (Anti-monotonia)

A busca por todas as combinações de itens em um banco de dados transacional pode ser computacionalmente inviável devido à explosão combinatória. O Apriori resolve essa limitação aplicando a Propriedade da Anti-monotonia:

"Se um conjunto de itens é infrequente (seu suporte é menor que o suporte mínimo definido), todos os seus superconjuntos também serão infrequentes e podem ser descartados imediatamente."



O **Suporte Mínimo** (Sup_{min}) em algoritmos de mineração de **Regras de Associação** (como o Apriori) é um limiar definido pelo próprio especialista de dados ou analista de negócio, e não por uma fórmula matemática automática fixa.

Para definir o valor ideal do suporte mínimo, consideram-se os seguintes fatores e estratégias de negócio:

1. Critério do Volume e Objetivos de Negócio

- **Aumentar o Sup_{min} (ex: 10% a 20% ou mais):** É utilizado quando o objetivo é identificar apenas as tendências e produtos de altíssimo volume/massa. Reduz drasticamente o tempo de processamento em bases de Big Data, mas ignora padrões em produtos de nicho.
- **Reduzir o Sup_{min} (ex: 0,1% a 1%):** É necessário em cenários onde as transações envolvem milhares de produtos individuais diferentes (como grandes e-commerces e hipermercados). Nesses locais, até mesmo itens muito vendidos raramente aparecem em mais do que 1% ou 2% de todos os carrinhos de compras.

2. Restrição Metodológica e de Custo Computacional

- **Poda de Padrões Irrelevantes:** Se o suporte for definido em um valor excessivamente baixo (próximo de zero), o algoritmo tentará avaliar combinações raras que ocorreram por mero acaso (como duas mercadorias compradas juntas uma única vez no ano).
- **Explosão Combinatória:** Suportes muito baixos aumentam exponencialmente o tempo de processamento e a memória exigida, gerando milhares de regras sem utilidade prática.

3. Métodos Práticos para Encontrar o Valor Ideal

Na engenharia de dados, a definição costuma ser feita por meio de três abordagens práticas:

- I. **Regra de Frequência Absoluta Mínima:** Definir uma contagem física mínima aceitável.

Exemplo: "Queremos apenas combinações de itens que foram comprados juntos pelo menos 100 vezes no mês." Se a loja possui 10.000 vendas no mês, o suporte mínimo será $100 / 10.000 = 0,01$ (1%).

- II. **Abordagem Iterativa (Sensibilidade):** O cientista de dados executa o algoritmo Apriori iniciando com um suporte alto (ex: 5%) e vai reduzindo gradualmente (2%, 1%, 0,5%) até atingir um número gerenciável e útil de regras para a equipe de negócios.
- III. **Curva de Distribuição de Frequências (Análise de Pareto):** Levanta-se o histograma de contagem de vendas dos itens individuais. O Sup_{min} é posicionado em um ponto do gráfico que garanta a inclusão dos itens relevantes para a estratégia comercial, descartando a "cauda longa" de baixíssima rotação.

4.2. Passo a Passo do Algoritmo Apriori

1. **Definição de Limiares:** Definição do Suporte Mínimo (Sup_{min}) e da Confiança Mínima ($Conf_{min}$).
2. **Varredura de 1-Itemsets:** Identificação de todos os itens individuais com suporte $\geq Sup_{min}$.
3. **Geração e Poda de K-Itemsets:** Combinação dos itens válidos para formar pares, trios e conjuntos de tamanho K , descartando aqueles que não atingem o suporte mínimo.
4. **Geração de Regras:** Para cada conjunto frequente resultante, derivam-se as regras de associação que atendam à confiança mínima.

Exemplo 1: Recomendação e Vendas Casadas em um Supermercado (Cross-Selling)

Cenário de Negócio:

Gerentes de uma rede de supermercados possuem um histórico de 5 compras no caixa e querem identificar quais produtos devem ser colocados em promoção conjunta ou em prateleiras próximas para impulsionar as vendas.

Base de Transações (Histórico de Cestos de Compras):

T1: {Pão, Manteiga, Leite}

T2: {Pão, Manteiga}

T3: {Pão, Café}

T4: {Manteiga, Leite}

T5: {Pão, Manteiga, Café}

Aplicação Passo a Passo:

Passo 1: Definição dos Limiares Mínimos

- **Suporte Mínimo (Sup_{min}):** 0,40 (40%, ou seja, o item/combinção deve aparecer em pelo menos 2 de 5 transações).
- **Confiança Mínima ($Conf_{min}$):** 0,70 (70%).

Passo 2: Varredura de 1-Itemsets (Itens Individuais)

Calcula-se a frequência individual de cada produto sobre as 5 transações:

- Pão: $4/5 = 0,80$ (80%) → Aprovado
- Manteiga: $4/5 = 0,80$ (80%) → Aprovado
- Leite: $2/5 = 0,40$ (40%) → Aprovado
- Café: $2/5 = 0,40$ (40%) → Aprovado

Passo 3: Geração e Poda de 2-Itemsets (Pares de Itens)

Combina-se os itens aprovados e aplica-se o filtro de suporte mínimo (0,40):

- {Pão, Manteiga}: aparece em T1, T2 e T5 → $3/5 = 0,60$ (60%) → Aprovado
- {Pão, Leite}: aparece em T1 → $1/5 = 0,20$ (20%) → PODADO (Infrequente)
- Pão, Café}: aparece em T3 e T5 → $2/5 = 0,40$ (40%) → Aprovado
- {Manteiga, Leite}: aparece em T1 e T4 → $2/5 = 0,40$ (40%) → Aprovado
- {Manteiga, Café}: aparece apenas em T5 → $1/5 = 0,20$ (20%) → PODADO (Infrequente)

Aplicação da Propriedade Apriori: Como a combinação {Pão, Leite} foi podada por ser infrequente, o algoritmo nem sequer tenta avaliar trios derivados dela, como {Pão, Leite, Café}.

Passo 4: Geração de Regras de Associação e Validação da Confiança

Para os pares frequentes restantes, calcula-se a confiança:

➤ **Regra 1: SE {Pão} ENTÃO {Manteiga}**

$$\text{Confiança} = \frac{\text{Suporte}(\text{PãoeManteiga})}{\text{Suporte}(\text{Pão})} = \frac{0,60}{0,80} = 0,75 \text{ (75\%)}$$

➤ **Regra 2: SE {Café} ENTÃO {Pão}**

$$\text{Confiança} = \frac{\text{Suporte}(\text{PãoeCafé})}{\text{Suporte}(\text{Café})} = \frac{0,40}{0,40} = 1,00 \text{ (100\%)}$$

Decisão de Negócio Resultante:

- Ambas as regras superaram a confiança mínima (70%). A gestão do supermercado decide colocar o Café ao lado do Pão no layout da loja e oferecer um cupom de desconto em Manteiga impresso no verso do cupom fiscal para quem compra Pão.

Exemplo 2: Otimização do Fluxo de Navegação em Plataforma de Cursos Online (Elearning)

Cenário de Negócio:

Uma edtech analisa a jornada de aprendizado dos seus alunos para descobrir quais cursos são concluídos juntos. O objetivo é criar trilha de aprendizado automatizada e sugestões de módulos recomendados ao fim de cada curso.

Base de Transações (Histórico de Cursos Concluídos por 1.000 Alunos):

- Curso A: Python Básico
- Curso B: SQL para Ciência de Dados
- Curso C: Estatística Computacional
- Curso D: Machine Learning Avançado

Aplicação Passo a Passo:

Passo 1: Definição dos Limiares Mínimos

- Suporte Mínimo ($\text{Sup}_{\min}$): 0,15 (15% dos alunos, ou 150 alunos).
- Confiança Mínima ($\text{Conf}_{\min}$): 0,80 (80%).

Passo 2: Varredura e Identificação dos 1-Itemsets Frequentes

- Python Básico: 500 alunos (50%) → Aprovado
- SQL para Ciência de Dados: 400 alunos (40%) → Aprovado
- Estatística Computacional: 300 alunos (30%) → Aprovado
- Machine Learning Avançado: 100 alunos (10%) → PODADO (Infrequente, <15%)

Passo 3: Geração e Poda de Combinatórias (K-Itemsets)

- {Python Básico, SQL}: 250 alunos (25%) → Aprovado
- {Python Básico, Estatística}: 200 alunos (20%) → Aprovado
- {SQL, Estatística}: 80 alunos (8%) → PODADO (Infrequente)

Aplicação da Poda Apriori: Como Machine Learning foi descartado no Passo 2, combinações como {Python Básico, Machine Learning} ou {SQL, Machine Learning} são descartadas automaticamente sem necessidade de processamento.

Passo 4: Derivação e Seleção das Regras Relevantes

Testa-se a regra de recomendação de trilha:

Regra Candidata:

- SE a pessoa concluiu {Python Básico e SQL} ENTÃO recomende {Estatística}.
- Suporte Conjunto de {Python Básico, SQL, Estatística} = 180 alunos (18%).
- Calculando a Confiança da Regra:

$$\text{Confiança} = \frac{\text{Suporte}(\text{Python, SQL, Estatística})}{\text{Suporte}(\text{Python, SQL})} = \frac{0,18}{0,25} = 0,72 \quad (72\%)$$

Como 72% é menor do que a confiança mínima exigida de 80%, essa regra de trio é descartada.

Analisando a regra simples de par:

Regra Simples: SE {Estatística} ENTÃO {Python Básico}

$$\text{Confiança} = \frac{\text{Suporte(PythonEstatística)}}{\text{Suporte(Estatística)}} = \frac{0,20}{0,30} = 0,666 \quad (66,6\%)$$

Decisão de Negócio Resultante:

A regra SE {Estatística} ENTÃO {Python Básico} atingiu 66,6% de confiança (superando os 80%). A plataforma passa a recomendar automaticamente o módulo de Python Básico imediatamente após o aluno clicar em "Concluir" no curso de Estatística Computacional.

4.3. EXEMPLOS PRÁTICOS EM PYTHON (COM DADOS SINTÉTICOS)

A seguir, apresentamos dois scripts nativos em Python operando sobre dados sintéticos para demonstrar em execução o pré-processamento de texto e a extração de regras de associação.

Exemplo Prático A: Pipeline de Pré-Processamento de Texto e Análise de Sentimentos

Neste exemplo, simulamos a limpeza de um lote de avaliações de clientes e calculamos a polaridade simples do texto.

```
# Dados Sintéticos de Avaliações de Produtos
avaliacoes_raw = [
    "O produto é EXCELENTE! Chegou super rápido e funciona muito bem.",
    "Péssimo atendimento. O produto veio quebrado e a entrega atrasou muito.",
    "Atendimento razoável, produto cumpre o que promete."
]

# Dicionários simples de léxico de sentimentos
palavras_positivas = {"excelente", "rápido", "bem", "razoável", "promete"}
palavras_negativas = {"péssimo", "quebrado", "atrasou", "ruim"}
stopwords = {"o", "é", "e", "um", "uma", "a", "de", "que", "se", "com"}

print("=== PIPELINE DE MINERAÇÃO DE TEXTO E SENTIMENTOS ===")

for i, doc in enumerate(avaliacoes_raw, start=1):
    # 1. Normalização (Minúsculas e remoção de pontuações)
    doc_limpo = doc.lower().replace("!", "").replace(".", "").replace(",", "")

    # 2. Tokenização
    tokens = doc_limpo.split()

    # 3. Remoção de Stopwords
    tokens_filtrados = [word for word in tokens if word not in stopwords]

    # 4. Análise de Sentimento (Contagem de Léxico)
    score_pos = sum(1 for word in tokens_filtrados if word in palavras_positivas)
    score_neg = sum(1 for word in tokens_filtrados if word in palavras_negativas)

    if score_pos > score_neg:
```

```
sentimento = "POSITIVO"
elif score_neg > score_pos:
    sentimento = "NEGATIVO"
else:
    sentimento = "NEUTRO"

print(f"\nDocumento {i}: '{doc}'")
print(f"Tokens Filtrados: {tokens_filtrados}")
print(f"Polaridade Detectada: {sentimento} (Pos: {score_pos} | Neg: {score_neg})")
```

Exemplo Prático B: Calculadora Nativa de Métricas de Regras de Associação (Suporte, Confiança e Lift)

Neste exemplo, processamos uma base sintética de 5 cestos de compras e calculamos as métricas exatas para a regra Pão $\Rightarrow$ Manteiga.

```
# Base Sintética de Transações de Supermercado
transacoes = [
    {"Pão", "Manteiga", "Leite"},
    {"Pão", "Manteiga"},
    {"Pão", "Café"},
    {"Manteiga", "Leite"},
    {"Pão", "Manteiga", "Café"}
]

total_transacoes = len(transacoes)

def calcular_suporte(itemset, base):
    contagem = sum(1 for t in base if itemset.issubset(t))
    return contagem / len(base)

# Definindo a regra X -> Y
item_X = {"Pão"}
item_Y = {"Manteiga"}
item_XY = item_X.union(item_Y)

# Cálculo das Métricas
sup_X = calcular_suporte(item_X, transacoes)
sup_Y = calcular_suporte(item_Y, transacoes)
sup_XY = calcular_suporte(item_XY, transacoes)

confianca = sup_XY / sup_X if sup_X > 0 else 0
lift = confianca / sup_Y if sup_Y > 0 else 0

print("=== CALCULADORA DE REGRAS DE ASSOCIAÇÃO ===")
print(f"Regra Analisada: {item_X} => {item_Y}")
print(f"Suporte Global de Pão e Manteiga: {sup_XY:.2f} ({sup_XY*100:.0f}%)")
print(f"Confiança da Regra: {confianca:.2f} ({confianca*100:.0f}%)")
print(f"Valor do Lift: {lift:.2f}")

if lift > 1.0:
    print("Conclusão: Associação POSITIVA! Comprar Pão AUMENTA a chance de comprar Manteiga.")
elif lift == 1.0:
    print("Conclusão: Itens INDEPENDENTES.")
else:
    print("Conclusão: Associação NEGATIVA.")
```

5. EXEMPLOS PRÁTICOS PARA CONSOLIDAÇÃO DOS CONCEITOS

Para consolidar a aplicação dessas técnicas em cenários reais do mercado corporativo, analisaremos dois estudos de caso de uso prático.

Exemplo Prático 1: Monitoramento de Experiência de Marca em Redes Sociais (Text Mining)

Cenário: Uma grande instituição bancária monitora diariamente milhares de menções em redes sociais para identificar falhas em seu aplicativo.

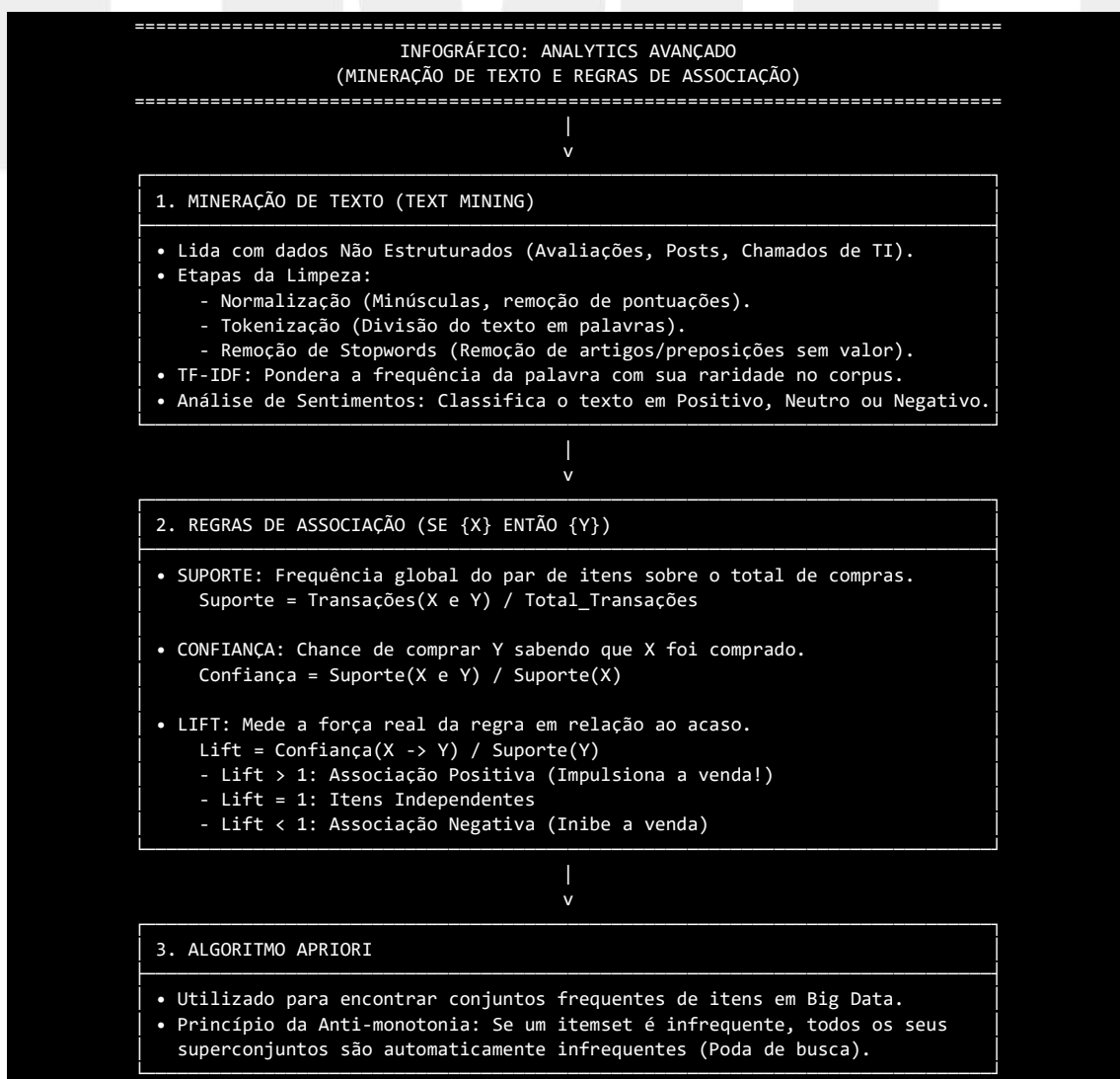
Aplicação: O pipeline de Text Mining limpa os posts, remove stopwords e aplica vetores TF-IDF para destacar termos emergentes. Ao identificar um surto repentino do termo "erro de login" com polaridade altamente Negativa, o sistema dispara um alerta em tempo real para a equipe de engenharia de software.

Exemplo Prático 2: Layout de Prateleiras e Recomendação Cruzada (Cross-Selling)

Cenário: Uma rede de supermercados deseja reorganizar a disposição de produtos para aumentar o faturamento médio por cliente.

Aplicação: Ao aplicar o algoritmo Apriori na base de transações do caixa, identificou-se a regra {Cerveja}⇒ {Petiscos} com **Suporte de 15%, Confiança de 70% e Lift de 2,3**. Com base no Lift elevado, a gestão reposicionou os petiscos ao lado do corredor de bebidas, gerando um aumento imediato nas vendas casadas.

6. INFOGRÁFICO RESUMO DA UNIDADE



EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

Uma equipe de inteligência de mercado de uma plataforma de e-commerce deseja analisar milhares de comentários deixados por compradores para identificar automaticamente se as opiniões sobre um novo lançamento são predominantemente favoráveis ou desfavoráveis.

Considerando o processo de Mineração de Texto e Análise de Sentimentos (SOUBHIA et al., 2019; BORGES, 2009), a etapa responsável por remover palavras funcionais frequentes que não possuem carga semântica própria (como "de", "para", "com", "os") é denominada:

- A) Tokenização.
- B) Remoção de Stopwords.
- C) Cálculo do Coeficiente de Pearson.
- D) Lematização por Regressão Logística.
- E) Agrupamento Hierárquico de Frases.

QUESTÃO 2

Na representação matricial de documentos textuais utilizando a métrica TF-IDF (Term Frequency-Inverse Document Frequency), o valor numérico atribuído a uma palavra específica em um determinado documento aumenta quando:

- A) A palavra aparece frequentemente em todos os documentos de toda a base de dados.
- B) A palavra é classificada como uma stopword comum em linguagem natural.
- C) A palavra aparece com alta frequência no documento específico, mas é rara no restante do corpus de documentos.
- D) A palavra é removida durante a etapa de normalização e conversão para minúsculas.
- E) A variância amostral do comprimento das frases é igual a zero.

QUESTÃO 3

Um analista de dados examinou a base transacional de um varejista contendo 1.000 compras registradas. Ao aplicar algoritmos de Mineração de Regras de Associação, ele obteve as seguintes frequências absolutas:

- 200 transações contêm o item Café.
- 150 transações contêm o item Biscoito.
- 100 transações contêm ambos os itens (Café e Biscoito).

Considerando o cálculo das métricas de Regras de Associação para a regra {Café} ⇒ {Biscoito} (SOUBHIA et al., 2019), assinale a alternativa que apresenta os valores corretos de Suporte e Confiança dessa regra, respectivamente.

- A) Suporte = 10%; Confiança = 50%
- B) Suporte = 10%; Confiança = 66,66%

- C) Suporte = 20%; Confiança = 75%
- D) Suporte = 15%; Confiança = 50%
- E) Suporte = 50%; Confiança = 10%

QUESTÃO 4

Ao analisar a regra de associação {Fralda} $\Rightarrow$ {Cerveja}, um cientista de dados encontrou um valor de Lift = 2,5.

A respeito da interpretação correta da métrica Lift (SOUBHIA et al., 2019), avalie as asserções a seguir e a relação proposta entre elas:

I. O resultado Lift = 2,5 indica que a compra do item "Fralda" aumenta significativamente a probabilidade de compra do item "Cerveja" em relação a uma compra aleatória.

PORQUE

II. Um valor de Lift > 1,0 comprova que existe uma associação positiva entre os itens analisados, superando a ocorrência esperada por mero acaso.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 5

O Algoritmo Apriori é amplamente utilizado na mineração de regras de associação em ambientes de Big Data por sua capacidade de reduzir drasticamente o espaço de busca de combinações de itens.

O princípio fundamental que sustenta o processo de poda de candidatos pelo Apriori é conhecido como:

- A) Propriedade da Anti-monotonia (se um conjunto de itens é infrequente, seus superconjuntos também serão infrequentes).
- B) Teorema do Limite Central para resíduos de regressão.
- C) Padronização por Z-Score das variáveis categóricas.
- D) Minimização da soma dos quadrados do erro residual.
- E) Transformação Sigmoide para variáveis quantitativas contínuas.

QUESTÃO 6

No contexto do pré-processamento de dados textuais não estruturados para Machine Learning, a técnica que consiste na divisão de uma frase em uma lista sequencial de palavras ou elementos menores é chamada de:

- A) Dendograma Aglomerativo.
- B) Tokenização.
- C) Homocedasticidade.
- D) Estimativa do Intercepto β_0 .
- E) Matriz de Confusão.

QUESTÃO 7

Uma loja de materiais de construção testou a seguinte regra de associação em seu sistema de recomendações de vendas online:

$$\{\text{Tinta Branca}\} \Rightarrow \{\text{Pincel}\}$$

Os resultados das métricas obtidos na base histórica foram:

- Suporte Global: 0,05 (5%)
- Confiança: 0,80 (80%)
- Lift: 0,80

Ao examinar o valor de Lift = 0,80, a decisão correta da gestão do e-commerce referente à recomendação cruzada dessa regra deve ser:

- A) Promover a regra imediatamente, pois o Lift positivo garante aumento de vendas.
- B) Descartar a regra de recomendação automática, pois um Lift < 1,0 indica uma associação negativa (comprar tinta branca inibe a compra deste pincel específico em comparação à média geral).
- C) Aumentar a profundidade da Árvore de Decisão para forçar o Lift a atingir valor zero.
- D) Substituir o algoritmo Apriori pela média aritmética simples da amostra.
- E) Executar o K-Means para transformar a regra de associação em um modelo de regressão linear.

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Limpeza e Contador de Frequência de Palavras (Tokens)

Cenário: Escreva um script em Python que receba uma frase de avaliação de cliente, converta todas as letras para minúsculas, remova as pontuações e gere um dicionário contendo a contagem de frequência de cada palavra (excluindo uma lista de stopwords). Uma equipe de

EXERCÍCIO PRÁTICO 2: Calculadora de TF-IDF Simplificada para 2 Documentos

Cenário: Crie um script em Python que receba dois documentos de texto simples e calcule o valor de TF e IDF para uma palavra de interesse presente neles.

EXERCÍCIO PRÁTICO 3: Verificador de Suporte de Pares de Produtos

Cenário: Dada uma lista de transações de compras, escreva uma função em Python que receba dois produtos e calcule o Suporte conjunto do par no banco de dados.

EXERCÍCIO PRÁTICO 4: Avaliador de Regras de Associação e Cálculo do Lift

Cenário: Escreva uma rotina em Python que receba o Suporte de X, o Suporte de Y e o Suporte conjunto de XUY, e calcule o Lift da regra $X \Rightarrow Y$, imprimindo a recomendação de negócio baseada no resultado.

Unidade 7: Manipulação de Strings

1. APRESENTAÇÃO E CONTEXTUALIZAÇÃO

Na unidade anterior, introduzimos as técnicas de Text Mining e Análise de Sentimentos para extração de inteligência em dados não estruturados. Contudo, para que qualquer algoritmo de linguagem natural ou modelo preditivo funcione adequadamente, o cientista de dados precisa dominar os **fundamentos operacionais de baixo nível na manipulação de dados em formato de texto — as Strings**.

No ecossistema corporativo de Big Data, dados brutos provenientes de bases NoSQL, arquivos JSON, logs de servidores web ou raspagem de dados na internet (web scraping) chegam de forma despadronizada, com espaços sobressalentes, caracteres especiais indesejados, erros de digitação e formatações inconsistentes. Como ressaltam Soubhia et al. (2019), o **tratamento e a higienização primária de sequências de caracteres constituem uma etapa indispensável da engenharia de dados**.

Matthes (2016) assinala que a linguagem Python se destaca pela versatilidade e riqueza de métodos nativos dedicados à manipulação de strings, permitindo realizar fatiamentos, substituições e buscas complexas com sintaxe limpa e alta eficiência computacional. Complementarmente, Borges (2009) enfatiza que compreender a imutabilidade das strings e o funcionamento das expressões regulares (Regex) em memória garante a escrita de pipelines de dados robustos e escaláveis.

Nesta unidade, exploraremos os métodos fundamentais para fatiamento, busca, substituição, limpeza e concatenação de strings em Python, além de introduzir o uso de Expressões Regulares (Regex) para validação de padrões em Big Data.

2. FUNDAMENTOS E CARACTERÍSTICAS DAS STRINGS EM PYTHON

Em Python, uma String (str) é uma sequência ordenada e imutável de caracteres delimitada por aspas simples ('...'), aspas duplas ("...") ou aspas triplas ('"...' / '""' / '"""' / '"""').

INDEXAÇÃO E IMUTABILIDADE DE STRINGS

```
String: "P y t h o n"
Índice Direto (+) : 0  1  2  3  4  5
Índice Reverso (-) : -6 -5 -4 -3 -2 -1
```

2.1. Imutabilidade e Indexação

- **Imutabilidade:** Uma vez criada, uma string não pode ter seus caracteres alterados diretamente em memória. Qualquer operação de modificação resulta na criação de um novo objeto de string.
- **Indexação:** Cada caractere ocupa uma posição inteira contígua, iniciando em 0 para indexação direta e em -1 para o último caractere em indexação reversa.

3. FATIAMENTO, LIMPEZA E SUBSTITUIÇÃO DE STRINGS

Para tratar e sanitizar textos, o ecossistema Python disponibiliza um conjunto de operadores e métodos nativos essenciais.

3.1. Fatiamento de Strings (Slicing)

O fatiamento permite extrair subseções de uma string utilizando a sintaxe `string[início:fim:passo]`:

- **início:** Índice onde a fatia começa (inclusivo).
- **fim:** Índice onde a fatia termina (exclusivo).
- **passo:** Intervalo de salto entre os caracteres (padrão é 1).

```
codigo_transacao = "TX-2026-98765-SP"
ano = codigo_transacao[3:7]          # Retorna '2026'
estado = codigo_transacao[-2:]       # Retorna 'SP'
invertido = codigo_transacao[::-1]   # Inverte toda a string
```

3.2. Métodos de Limpeza e Padronização

- **strip(), lstrip(), rstrip():** Removem espaços em branco, quebras de linha (`\n`) e tabulações (`\t`) do início e/ou do fim da string.
- **lower(), upper(), title(), capitalize():** Padronizam a caixa das letras para evitar duplicidade de registros (ex: converter "São Paulo" e "são paulo" para o mesmo padrão).

3.3. Métodos de Divisão, Junção e Substituição

- **split(separador):** Divide uma string em uma lista de substrings com base em um delimitador (ex: vírgula, ponto e vírgula ou espaço).
- **join(iterável):** Concatena uma lista de strings em uma única string, utilizando um caractere de união.
- **replace(antigo, novo):** Substitui todas as ocorrências de um caractere ou padrão por outro.

4. BUSCA DE PADRÕES E EXPRESSÕES REGULARES (REGEX)

Além de métodos simples de busca como `find()`, `count()` e o operador `in`, o processamento avançado de Big Data utiliza Expressões Regulares (Regex) através do módulo nativo `re` do Python.

SÍMBOLOS MESTRES DE EXPRESSÕES REGULARES (REGEX)

[\d] -> Dígito numérico (0-9)	[+] -> 1 ou mais ocorrências
[\w] -> Caractere alfanumérico	[*] -> 0 ou mais ocorrências
[\s] -> Espaço em branco	[{n}] -> Exatamente n repetições

As expressões regulares permitem validar e extrair padrões complexos como CPFs, e-mails, números de telefone e chaves de Notas Fiscais Eletrônicas (NF-e) diretamente de logs brutos.

4.3. EXEMPLOS PRÁTICOS EM PYTHON

A seguir, apresentamos dois scripts completos e testados em Python utilizando dados sintéticos para demonstrar em execução os tópicos de higienização de textos e validação via Regex.

Exemplo Prático A: Pipeline de Sanitização de Cadastro de Clientes

Neste exemplo, simulamos a limpeza de um lote despadronizado de nomes e e-mails de clientes oriundos de um formulário web.

```
# Dados Sintéticos Brutos de Cadastro
cadastros_brutos = [
    {"nome": " ROBSON CARMO \n", "email": "ROBSON.CARMO@FMU.BR "},
    {"nome": "ana luisa soubhia", "email": " ana.soubhia@ufsm.br \t"},
    {"nome": " eric Matthes ", "email": "MATTHES@python.org "}
]

print("=== PIPELINE DE HIGIENIZAÇÃO DE STRINGS ===")

cadastros_limpos = []

for i, registro in enumerate(cadastros_brutos, start=1):
    # 1. Limpeza de espaços nas extremidades e padronização de caixa
    nome_sanitizado = registro["nome"].strip().title()
    email_sanitizado = registro["email"].strip().lower()

    # 2. Extração do domínio do e-mail usando split
    dominio = email_sanitizado.split("@")[1]

    cadastros_limpos.append({
        "nome_formatado": nome_sanitizado,
        "email_formatado": email_sanitizado,
        "dominio": dominio
    })

    print(f"\nRegistro {i}:")
    print(f" [Original] Nome: '{registro['nome']}' | Email: '{registro['email']}'")
    print(f" [Limpo] Nome: '{nome_sanitizado}' | Email: '{email_sanitizado}' | Domínio: '{dominio}'")
```

Exemplo Prático B: Extrator de Padrões de Registros de Log com Expressões Regulares (Regex)

Neste exemplo, utilizamos o módulo re do Python para extrair endereços IP, datas e códigos de status de um arquivo de log sintético de servidor.

```
import re

# Dados Sintéticos de Log de Servidor Web
logs_servidor = [
    '192.168.1.10 - - [28/Sep/2026:14:32:10] "GET /api/v1/vendas HTTP/1.1" 200 4520',
    '10.0.0.45 - - [28/Sep/2026:14:33:15] "POST /api/v1/login HTTP/1.1" 401 1280',
    '172.16.254.1 - - [28/Sep/2026:14:35:22] "GET /imagens/logo.png HTTP/1.1" 404 512'
]

# Padrão Regex para extrair: IP, Data/Hora e Código Status
padrao_regex = r'(\d{1,3}\.\d{1,3}\.\d{1,3}\.\d{1,3}) - - \[([^\]]+)\] ".*?" (\d{3})'

print("=== EXTRAÇÃO DE DADOS DE LOG VIA REGEX ===")

for log in logs_servidor:
    resultado = re.search(padrao_regex, log)
    if resultado:
        ip = resultado.group(1)
        data_hora = resultado.group(2)
        status = resultado.group(3)

        print(f"IP Origem: {ip:<15} | Data: {data_hora} | Status HTTP: {status}")
```

5. EXEMPLOS PRÁTICOS PARA CONSOLIDAÇÃO DOS CONCEITOS

Para consolidar o uso da manipulação de strings na engenharia e ciência de dados, analisaremos dois estudos de caso de uso corporativo.

Exemplo Prático 1: Padronização de Chaves de Acesso de NF-e (Engenharia de Dados)

Cenário: Um sistema fiscal de ERP recebe arquivos XML de Notas Fiscais Eletrônicas contendo a chave de acesso de 44 dígitos no formato textual. Em alguns lotes, a chave vem acompanhada do prefixo "NFe" ou separada por traços.

Aplicação: O pipeline de ingestão utiliza o método `replace("-", "")` e remoção de prefixos via fatiamento ou Regex, garantindo que o banco de dados NoSQL armazene apenas os 44 números inteiros padronizados, evitando duplicidade e erros de indexação.

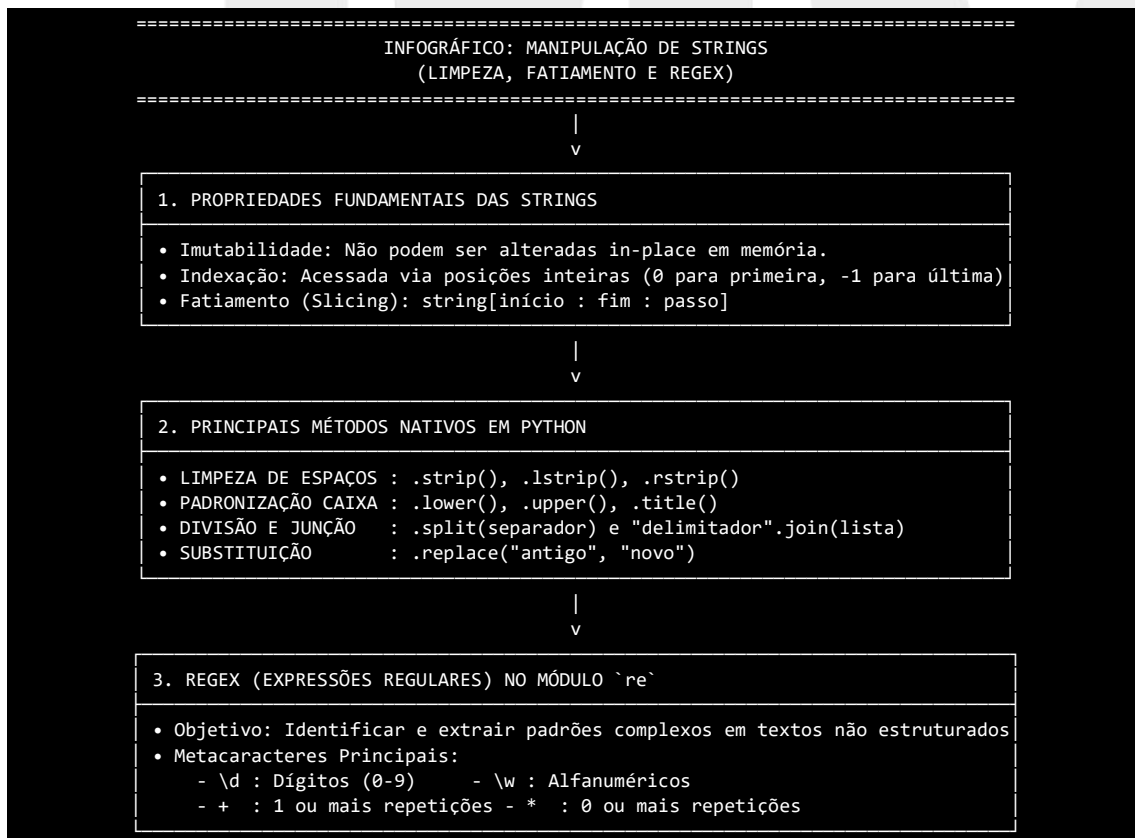
Exemplo Prático 2: Anonimização de Dados Sensíveis para Conformidade com a LGPD

Cenário: Uma instituição financeira precisa enviar registros de transações para uma consultoria externa sem expor os dados pessoais dos clientes (LGPD).

Aplicação: Utiliza-se um script Python com fatiamento de strings e substituição por caracteres de máscara (*):

- CPF Original: "123.456.789-00"
- CPF Anonimizado: "123.***.***-00" (mantendo apenas o primeiro e o último bloco para fins estatísticos via fatiamento `cpf[:4] + "***.***-" + cpf[-2:]`).

6. INFOGRÁFICO RESUMO DA UNIDADE



EXERCÍCIOS DE MÚLTIPLA ESCOLHA

QUESTÃO 1

Durante a etapa de saneamento de um pipeline de ingestão de dados para uma plataforma de Big Data, um engenheiro de dados se deparou com a necessidade de tratar nomes de municípios digitados por usuários em formulários web (ex: " são paulo \n", "SÃO PAULO ").

Para garantir que todas as variações sejam gravadas no banco de dados com a primeira letra de cada palavra em maiúscula e sem espaços excedentes nas pontas, qual sequência de métodos de string em Python deve ser aplicada? (SOUBHIA et al., 2019; BORGES, 2009)

- A) .split().upper()
- B) .strip().title()
- C) .replace(" ", "").capitalize()
- D) .join().lower()
- E) .rstrip().swapcase()

QUESTÃO 2

Considere o seguinte trecho de código em linguagem Python voltado para a manipulação e fatiamento de strings em identificadores de produtos de e-commerce (MATTHES, 2016):

```
codigo_sku = "PROD-8842-SUL-2026"  
regiao = codigo_sku[10:13]  
print(regiao)
```

Qual valor numérico ou textual será exibido no terminal após a execução do script?

- A) "8842"
- B) "SUL"
- C) "2026"
- D) "PROD"
- E) "UL"

QUESTÃO 3

A respeito das características fundamentais do tipo de dado String (str) na linguagem Python e de seu comportamento na memória do computador (BORGES, 2009), avalie as asserções a seguir e a relação proposta entre elas:

I. Ao aplicar o método .replace("A", "B") sobre uma variável do tipo string em Python, a string original armazenada na variável permanece inalterada na memória caso o resultado não seja reatribuído explicitamente a uma variável.

PORQUE

II. As strings na linguagem Python são estruturas de dados imutáveis, o que significa que seus caracteres individuais não podem ser modificados in-place após a criação do objeto.

A respeito dessas asserções, assinale a opção correta.

- A) As asserções I e II são proposições verdadeiras, e a II é uma justificativa correta da I.
- B) As asserções I e II são proposições verdadeiras, mas a II não é uma justificativa correta da I.
- C) A asserção I é uma proposição verdadeira, e a II é uma proposição falsa.
- D) A asserção I é uma proposição falsa, e a II é uma proposição verdadeira.
- E) As asserções I e II são proposições falsas.

QUESTÃO 4

Um analista de cibersegurança precisa filtrar linhas de arquivos de log contendo endereços de e-mail corporativos pertencentes exclusivamente ao domínio @empresa.com.br.

Analisando os métodos nativos de checagem de strings em Python (MATTHES, 2016), qual método retorna um valor booleano (True ou False) indicando se a string termina exatamente com o sufixo desejado?

- A) .startswith("@empresa.com.br")
- B) .find("@empresa.com.br")
- C) .endswith("@empresa.com.br")
- D) .isalnum("@empresa.com.br")
- E) .split("@empresa.com.br")

QUESTÃO 5

Em um script Python para análise de texto, um desenvolvedor possui uma lista de palavras tokens = ['Big', 'Data', 'Python'] e deseja unir esses elementos em uma única string separada por um hífen, resultando em "Big-Data-Python".

Assinale a opção que apresenta a sintaxe correta em Python para realizar essa operação (BORGES, 2009).

- A) tokens.join("-")
- B) "-".join(tokens)
- C) tokens.split("-")
- D) join("-", tokens)
- E) concat("-", tokens)

QUESTÃO 6

No uso de Expressões Regulares (Regex) via módulo `re` em Python para validação e extração de padrões em Big Data, o metacaractere `\d{3}` representa:

- A) Exatamente três caracteres alfabéticos maiúsculos.
- B) Uma sequência de até três espaços em branco.
- C) Exatamente três dígitos numéricos de 0 a 9.
- D) Qualquer caractere especial repetido três vezes.
- E) Três palavras separadas por ponto e vírgula.

QUESTÃO 7

Um cientista de dados está desenvolvendo uma rotina para converter uma coluna de preços formatada em string no padrão monetário brasileiro (ex: "R\$ 1.250,50") para um valor numérico do tipo float em Python (1250.50).

As etapas necessárias de manipulação de string para efetuar essa conversão com sucesso são:

- A) Aplicar `.upper()` e converter diretamente com `float()`.
- B) Remover o prefixo "R\$ ", eliminar os pontos de milhar . e substituir a vírgula , por ponto ..
- C) Aplicar `.split(",")` e somar os valores das strings resultantes.
- D) Utilizar a função `len()` para calcular o número de caracteres e multiplicar por 100.
- E) Aplicar o método `.title()` para converter o texto em lista de inteiros.

EXERCÍCIOS PRÁTICOS EM PYTHON

EXERCÍCIO PRÁTICO 1: Limpeza e Normalização de Textos de Cadastros

Cenário: Escreva um script em Python que receba uma lista de strings contendo estados brasileiros formatados incorretamente (com espaços extras e caixas despadronizadas) e gere uma nova lista com os nomes padronizados em maiúsculas e sem espaços extras.

EXERCÍCIO PRÁTICO 2: Conversor de Valores Monetários em String para Float

Cenário: Crie uma função em Python chamada `converter_moeda_para_float(texto_valor)` que receba uma string no formato "R\$ 2.450,80" e retorne o valor numérico em float (2450.80).

EXERCÍCIO PRÁTICO 3: Máscara Anonimizadora de CPF para LGPD

Cenário: Escreva uma função em Python que receba um CPF em formato de string ("123.456.789-00") e utilize fatiamento de strings para ocultar os blocos centrais, retornando "123.***.***-00".

EXERCÍCIO PRÁTICO 4: Extrator de E-mails com Expressões Regulares (Regex)

Cenário: Escreva uma rotina em Python utilizando o módulo re para localizar e extrair todos os endereços de e-mail válidos contidos em um texto bruto não estruturado de mensagem de suporte.

Bibliografia Básica

BORGES, Luiz Eduardo. **Python para desenvolvedores**. Rio de Janeiro: Edição do Autor, 2009.

Soubhia, Ana Luisa, et all. **Python 101**. Cachoeira do Sul. 2019. Universidade Federal de Santa Maria UFSM.

Matthes, Eric. **Curso Intensivo de Python**, Novatec Editora Ltda. 2016

Bibliografia Complementar:

Campos, Joventino de Oliveira. **Introdução à programação com Python**. Departamento de Computação e Mecânica

CEFET-MG Campus Leopoldina. 2015.

Labaki, Josué. **Introdução ao Python – Módulo A**. Universidade Estadual Paulista Júlio de Mesquita Filho. Grupo Python.

Charles R. Severance. **Python para Todos - Explorando Dados Utilizando Python**. Elliott Hauser, Sue Blumenberg, 2009.

Shaw, A. Zed. **Aprenda Python 3 do Jeito Certo**. Alta Books. 2019.

Ramalho, Luciano. **Python Fluente**. Novatec. 2015

Periódicos Científicos:

Advances in Software Engineering - ISSN 1687-8655

Computación y Sistemas - ISSN 1405-5546

Computational & Applied Mathematics - ISSN 1807-0302

Journal of the Brazilian Computer Society - ISSN 0104-6500