## Features dos datasets

**TARGET**

target

**DIVERSOS**

os\_timestamp, node\_name, iteration

**CPU**

mean\_os\_cpu\_ctx\_switches, mean\_os\_cpu\_guest, mean\_os\_cpu\_guest\_nice,mean\_os\_cpu\_idle, mean\_os\_cpu\_interrupts, mean\_os\_cpu\_iowait, mean\_os\_cpu\_irq, mean\_os\_cpu\_nice, mean\_os\_cpu\_soft\_interrupts, mean\_os\_cpu\_softirq, mean\_os\_cpu\_steal, mean\_os\_cpu\_syscalls, mean\_os\_cpu\_system, mean\_os\_cpu\_user,

**DISCO**

mean\_os\_disk\_discard\_io, mean\_os\_disk\_discard\_merges, mean\_os\_disk\_discard\_sectors, mean\_os\_disk\_discard\_ticks, mean\_os\_disk\_in\_flight, mean\_os\_disk\_io\_ticks, mean\_os\_disk\_read\_io, mean\_os\_disk\_read\_merge, mean\_os\_disk\_read\_sectors, mean\_os\_disk\_read\_ticks, mean\_os\_disk\_time\_in\_queue, mean\_os\_disk\_write\_io, mean\_os\_disk\_write\_merge, mean\_os\_disk\_write\_sectors, mean\_os\_disk\_write\_ticks,

**MEMÓRIA**

mean\_os\_mem\_nr\_active\_anon, mean\_os\_mem\_nr\_active\_file, mean\_os\_mem\_nr\_inactive\_anon, mean\_os\_mem\_nr\_inactive\_file, mean\_os\_mem\_nr\_mapped, mean\_os\_mem\_pgfault, mean\_os\_mem\_pgfree, mean\_os\_mem\_pgmajfault, mean\_os\_mem\_pgpgin, mean\_os\_mem\_pgpgout, mean\_os\_mem\_pgreuse,

**REDE**

mean\_os\_net\_bytes\_recv, mean\_os\_net\_bytes\_sent, mean\_os\_net\_dropin, mean\_os\_net\_dropout, mean\_os\_net\_errin, mean\_os\_net\_errout, mean\_os\_net\_num\_connections, mean\_os\_net\_packets\_recv, mean\_os\_net\_packets\_sent,

**PROCESSO**

mean\_process\_cpu\_children\_system, mean\_process\_cpu\_children\_user, mean\_process\_cpu\_iowait, mean\_process\_cpu\_system, mean\_process\_cpu\_user, mean\_process\_disk\_read\_bytes, mean\_process\_disk\_read\_chars, mean\_process\_disk\_read\_count, mean\_process\_disk\_write\_bytes, mean\_process\_disk\_write\_chars, mean\_process\_disk\_write\_count, mean\_process\_mem\_data, mean\_process\_mem\_dt, mean\_process\_mem\_lib, mean\_process\_mem\_resident, mean\_process\_mem\_shared, mean\_process\_mem\_size, mean\_process\_mem\_text, mean\_process\_net\_rx\_bytes, mean\_process\_net\_rx\_compressed, mean\_process\_net\_rx\_drop, mean\_process\_net\_rx\_errs, mean\_process\_net\_rx\_fifo, mean\_process\_net\_rx\_frame, mean\_process\_net\_rx\_multicast, mean\_process\_net\_rx\_packets, mean\_process\_net\_tx\_bytes, mean\_process\_net\_tx\_carrier, mean\_process\_net\_tx\_colls, mean\_process\_net\_tx\_compressed, mean\_process\_net\_tx\_drop, mean\_process\_net\_tx\_errs, mean\_process\_net\_tx\_fifo, mean\_process\_net\_tx\_packets,

**CONTAINER**

mean\_container\_cpu\_nr\_periods, mean\_container\_cpu\_nr\_throttled, mean\_container\_cpu\_system, mean\_container\_cpu\_throttled\_time, mean\_container\_cpu\_user, mean\_container\_disk\_8:0\_async, mean\_container\_disk\_8:0\_discard, mean\_container\_disk\_8:0\_read, mean\_container\_disk\_8:0\_sync, mean\_container\_disk\_8:0\_total, mean\_container\_disk\_8:0\_write, mean\_container\_mem\_active\_anon, mean\_container\_mem\_active\_file, mean\_container\_mem\_cache, mean\_container\_mem\_inactive\_anon, mean\_container\_mem\_inactive\_file, mean\_container\_mem\_mapped\_file, mean\_container\_mem\_pgfault, mean\_container\_mem\_pgmajfault, mean\_container\_mem\_pgpgin, mean\_container\_mem\_pgpgout, mean\_container\_mem\_rss, mean\_container\_mem\_unevictable, mean\_container\_net\_rx\_bytes, mean\_container\_net\_rx\_compressed, mean\_container\_net\_rx\_drop, mean\_container\_net\_rx\_errs, mean\_container\_net\_rx\_fifo, mean\_container\_net\_rx\_frame, mean\_container\_net\_rx\_multicast, mean\_container\_net\_rx\_packets, mean\_container\_net\_tx\_bytes, mean\_container\_net\_tx\_carrier, mean\_container\_net\_tx\_colls, mean\_container\_net\_tx\_compressed, mean\_container\_net\_tx\_drop, mean\_container\_net\_tx\_errs, mean\_container\_net\_tx\_fifo, mean\_container\_net\_tx\_packets,

## Bases de dados

As bases de dados svm.csv e terasort.csv tem a mesma estrutura de dados.

# PREPARAÇÃO DOS DADOS

**Relatório de Análise - Célula 5: Carregamento e Análise do Target**

**🎯 Objetivo da Célula**

A célula 5 é responsável pelo **carregamento do dataset** e **análise inicial das classes target**, sendo uma etapa fundamental na preparação dos dados para modelagem de machine learning.

**📋 Processos Executados**

**1. Carregamento Robusto do Dataset**

* **Estratégia**: Carregamento com tratamento de exceções em duas etapas
* **Primeira tentativa**: Carrega o dataset completo usando pd.read\_csv(arq\_dataset\_csv)
* **Fallback**: Se falhar, tenta carregar uma amostra de 10.000 linhas para análise
* **Saída**: Exibe dimensões do dataset (linhas × colunas) com formatação numérica

**2. Análise Detalhada do Target**

**Distribuição das Classes**

* **Contagem**: Calcula quantidade de amostras por classe usando value\_counts()
* **Percentuais**: Converte contagens em percentuais para melhor visualização
* **Formato de saída**: Exibe classes em formato dicionário e percentuais individuais

**Cálculo de Balanceamento**

* **Métrica principal**: balance\_ratio = classe\_minoritária / classe\_majoritária
* **Conversão**: balance\_percentage = balance\_ratio × 100
* **Interpretação**: Quanto mais próximo de 100%, mais balanceado está o dataset

**3. Classificação do Nível de Balanceamento**

A célula implementa uma **escala de 5 níveis** para classificar o balanceamento:

| **Ratio** | **Nível** | **Indicador** | **Interpretação** |
| --- | --- | --- | --- |
| ≥ 80% | ✅ BALANCEADO | Verde | Dataset bem balanceado |
| 60-79% | ⚠️ LEVEMENTE DESBALANCEADO | Amarelo | Pode funcionar sem correção |
| 40-59% | ⚠️ MODERADAMENTE DESBALANCEADO | Laranja | Recomenda-se balanceamento |
| 20-39% | 🔴 SEVERAMENTE DESBALANCEADO | Vermelho | Balanceamento necessário |
| < 20% | 🔴 EXTREMAMENTE DESBALANCEADO | Vermelho escuro | Balanceamento crítico |

**4. Relatório de Informações Gerais**

* **Valores ausentes**: Conta células vazias na coluna target
* **Total de features**: Calcula número de colunas menos a coluna target
* **Métricas de balanceamento**: Exibe ratio, classes minoritária e majoritária
* **Formatação**: Usa separadores de milhares para melhor legibilidade

**📊 Saídas Geradas**

1. **Status do carregamento** com dimensões do dataset
2. **Distribuição detalhada** das classes com contagens e percentuais
3. **Classificação do balanceamento** com indicadores visuais
4. **Resumo estatístico** das classes e qualidade dos dados

**🎯 Importância para o Pipeline**

Esta célula é **crítica** pois:

* ✅ Valida a integridade do carregamento dos dados
* ✅ Identifica problemas de balanceamento que podem afetar o modelo
* ✅ Fornece métricas essenciais para decisões de pré-processamento
* ✅ Estabelece baseline para monitoramento da qualidade dos dados

**🔧 Melhorias Implementadas**

* **Cálculo correto** do ratio de balanceamento (corrigido da versão anterior)
* **Escala lógica** de classificação (valores altos = mais balanceado)
* **Tratamento robusto** de exceções no carregamento
* **Formatação clara** das saídas com emojis e separadores

A célula está **bem estruturada** e fornece todas as informações necessárias para a próxima etapa de preparação dos dados.

PREPARAÇÃO DOS DADOS

**🎯 Objetivo da Célula**

A célula 22 (execution count 22) é responsável pela **preparação e limpeza dos dados** que serão utilizados para treinamento dos modelos de machine learning. Esta é uma etapa crucial que transforma os dados brutos em um formato adequado para algoritmos de ML.

**📋 Processos Executados**

**1. Identificação e Exclusão de Colunas Irrelevantes**

* **Estratégia**: Remove colunas que não são adequadas para modelagem
* **Colunas removidas**:
  + os\_timestamp: Dados temporais (não são features preditivas)
  + node\_name: Identificadores categóricos (pode causar overfitting)
  + iteration: Números sequenciais (não têm valor preditivo)
  + target: Variável dependente (separada das features)
* **Validação**: Verifica se as colunas existem antes de tentar excluí-las

**2. Seleção de Features Numéricas**

* **Filtro automático**: Seleciona apenas colunas com tipos numéricos (int, float)
* **Justificativa**: Algoritmos de ML tradicionais funcionam melhor com dados numéricos
* **Resultado**: Lista final de features válidas para o modelo

**3. Separação Features vs Target (X, y)**

* **X (Features)**: Variáveis independentes/preditoras
* **y (Target)**: Variável dependente a ser prevista
* **Uso de .copy()**: Evita modificações no DataFrame original

**4. Tratamento de Valores Ausentes**

* **Detecção**: Conta total de valores nulos em todas as features
* **Estratégia de imputação**: Substitui valores ausentes pela **mediana**
* **Vantagens da mediana**:
  + ✅ Robusta a outliers
  + ✅ Mantém distribuição original
  + ✅ Funciona bem para dados numéricos

**5. Codificação do Target**

* **Verificação automática**: Testa se target é categórico (object) ou numérico
* **LabelEncoder**: Converte strings em números (ex: "normal"→0, "anomaly"→1)
* **Mapeamento**: Salva a correspondência original→codificado para interpretação posterior
* **Flexibilidade**: Funciona tanto com targets categóricos quanto numéricos

**6. Validação Final**

* **Dimensionalidade**: Confirma número de amostras e features
* **Consistência**: Verifica se X e y têm o mesmo número de amostras
* **Classes**: Mostra valores únicos do target codificado

**📊 Saídas Geradas**

1. **Contagem de features** selecionadas para o modelo
2. **Lista de colunas excluídas** para auditoria
3. **Status do tratamento** de valores ausentes (se aplicável)
4. **Mapeamento de classes** (se target foi codificado)
5. **Resumo final** com dimensões dos dados preparados

**🎯 Boas Práticas Implementadas**

**✅ Pontos Fortes**

* **Tratamento robusto** de valores ausentes com mediana
* **Validação de existência** de colunas antes da exclusão
* **Codificação automática** do target baseada no tipo de dados
* **Preservação dos dados originais** com uso de .copy()
* **Documentação clara** com prints informativos

**⚠️ Possíveis Melhorias**

* **Feature engineering**: Poderia incluir criação de novas features
* **Análise de correlação**: Para detectar features redundantes
* **Detecção de outliers**: Para tratamento de valores extremos
* **Validação de tipos**: Para garantir consistência dos dados

**🔧 Impacto no Pipeline**

Esta célula é **fundamental** porque:

* ✅ **Limpa os dados** removendo informações irrelevantes
* ✅ **Padroniza o formato** para algoritmos de ML
* ✅ **Trata problemas** como valores ausentes
* ✅ **Prepara o target** para classificação
* ✅ **Valida a integridade** dos dados finais

**📈 Preparação para Próximas Etapas**

Os dados resultantes estão prontos para:

* **Divisão treino/teste/validação**
* **Normalização/padronização**
* **Aplicação de algoritmos de ML**
* **Validação cruzada**

## ESTRATÉGIA DE REMOÇÃO DE FEATURES

**Explicação da Célula 7: Remoção de Features Altamente Correlacionadas**

**🎯 Objetivo da Célula**

Esta célula implementa um processo de **seleção de features** removendo variáveis que apresentam **alta correlação** entre si, evitando redundância e melhorando a eficiência do modelo de machine learning.

**📋 Análise Detalhada do Código**

**1. Função drop\_campos\_altamente\_correlacionados()**

**Parâmetros de Entrada:**

* X: DataFrame com as features
* threshold=0.95: Limiar de correlação (padrão 95%)

**Processo de Cálculo:**

* **Matriz de correlação**: Calcula correlação de Pearson entre todas as features
* **.abs()**: Usa valores absolutos (considera correlações positivas e negativas)
* **Interpretação dos valores**:
  + 1.0 = Correlação perfeita (positiva ou negativa)
  + 0.0 = Sem correlação linear
  + 0.95+ = Alta correlação (redundância)

**Identificação de Pares Correlacionados:**

* **Triângulo superior**: Evita duplicação de análise (correlação A→B = B→A)
* **k=1**: Exclui a diagonal principal (correlação de uma variável consigo mesma = 1.0)

**Seleção de Features para Remoção:**

* **List comprehension**: Identifica colunas com correlação > 95%
* **Estratégia**: Remove uma das features do par correlacionado

**2. Visualização da Matriz de Correlação**

* **Heatmap**: Visualização colorida da matriz de correlação
* **coolwarm**: Escala de cores (azul=negativa, vermelho=positiva)
* **center=0**: Centraliza a escala de cores no zero

**3. Execução Principal**

**Carregamento dos Dados:**

* Recupera os datasets preparados (treino, teste, validação)

**Verificação de Configuração:**

* **Configuração flexível**: Permite ativar/desativar o processo
* **Padrão True**: Por padrão, executa a remoção

**Atualização das Features:**

* **Filtragem**: Remove features correlacionadas da lista original
* **Atualização**: Aplica mudanças aos datasets de treino/teste/validação

**Persistência e Relatório:**

* **Salva**: Persiste as mudanças em arquivo
* **Relatório**: Exibe informações atualizadas
* **Recarrega**: Obtém dados atualizados

**4. Informações de Debug:**

**🎯 Importância para Machine Learning**

**✅ Benefícios da Remoção:**

1. **Reduz multicolinearidade**: Evita problemas numéricos em algoritmos
2. **Melhora interpretabilidade**: Menos features redundantes
3. **Reduz overfitting**: Menos parâmetros para ajustar
4. **Acelera treinamento**: Menos dimensões para processar
5. **Economia de memória**: Datasets menores

**⚠️ Considerações:**

* **Perda de informação**: Pode remover features úteis
* **Threshold crítico**: 95% é conservador, poderia ser ajustado
* **Correlação linear**: Não detecta dependências não-lineares

**🔧 Pontos de Melhoria Sugeridos:**

1. **Threshold adaptativo**: Testar diferentes valores (0.85, 0.90, 0.95)
2. **Importância das features**: Manter a feature mais importante do par correlacionado
3. **Análise de correlação não-linear**: Usar mutual information
4. **Validação cruzada**: Testar impacto no desempenho do modelo

**📊 Resultado Esperado:**

* **Dataset otimizado** com menos redundância
* **Matriz de correlação** mais balanceada
* **Features mais independentes** para melhor generalização

Esta célula implementa uma **técnica fundamental** de feature selection que prepara os dados para uma modelagem mais eficiente e robusta.