Separação Treino-Validation-Teste

# Aula	9
☑ Ready	☑
☑ Finished	☑
≡ Ciclos	Ciclo 01: Fundamentos

Objetivo da Aula:

- [ ] O que é o método Train-Validation-Test Split
- [ ] A divisão ideal
- [ ] Atenção

Conteúdo:

▼ 1. O que é o método Train-Validation-Test Split

O método **Train, Validation e Test Split** é uma técnica usada para dividir os dados em três conjuntos distintos, permitindo a **construção, ajuste e avaliação** de modelos de machine learning e séries temporais. Essa abordagem é essencial para **avaliar corretamente o desempenho do modelo** e garantir que ele **generalize bem** para novos dados. Sem essa divisão, podemos enfrentar problemas como **overfitting** e **avaliações irreais** do desempenho do modelo.

▼ 1.1. Conjunto de Treinamento (Train set)

O **conjunto de treinamento** é responsável por ensinar o modelo. Ele contém a maior parte dos dados e é utilizado para ajustar os parâmetros internos do modelo. Durante essa etapa, o modelo aprende padrões a partir das informações disponíveis.

No entanto, avaliar o modelo apenas com os dados de treinamento pode gerar uma falsa sensação de bom desempenho, pois o modelo pode simplesmente ter decorado os exemplos em vez de aprender padrões generalizáveis.

Para evitar esse problema, é necessário um segundo conjunto de dados, chamado de **conjunto de validação**

▼ 1.2. Conjunto de Validação (Validation set)

O **conjunto de validação** serve para avaliar o modelo enquanto ele ainda está sendo ajustado.

Ele não é utilizado para treinar o modelo, mas sim para testar diferentes combinações de hiperparâmetros e estratégias de modelagem.

Essa etapa é fundamental porque permite identificar ajustes que podem melhorar a precisão e a robustez do modelo antes da avaliação final.

Se o desempenho do modelo no conjunto de validação for muito inferior ao desempenho no conjunto de treinamento, isso indica que o modelo está sofrendo de overfitting.

▼ 1.3. Conjunto de Teste (test set)

o **conjunto de teste** é utilizado para fazer a avaliação final do modelo.

Esse conjunto contém dados totalmente novos, que não foram utilizados nem no treinamento nem na validação.

Dessa forma, ele simula o cenário real onde o modelo será aplicado. A performance do modelo no conjunto de teste fornece uma estimativa mais precisa de como ele irá se comportar em dados reais.

Caso o desempenho no teste seja significativamente pior do que nos dados de validação, pode haver indícios de que o modelo foi ajustado em excesso para o conjunto de validação, levando a um fenômeno chamado **overfitting ao conjunto de validação**.

Separar os dados nesses três conjuntos é essencial para garantir que o modelo seja treinado corretamente, ajustado de maneira adequada e avaliado de forma justa. Dessa maneira, é possível construir um modelo

mais confiável, capaz de fornecer previsões precisas e úteis no mundo real.

▼ 2. A divisão ideal

Diferente de outros tipos de dados, as **séries temporais** possuem uma característica fundamental: **a dependência temporal**.

Isso significa que a ordem dos dados importa, pois as observações passadas influenciam as futuras. Portanto, a divisão dos conjuntos de treinamento, validação e teste deve ser feita respeitando a sequência temporal.

A divisão clássica para séries temporais segue o seguinte formato:

Conjunto	Divisão	Divisão mais usada	Observação
Treinamento	60% a 80%	80%	Dados iniciais.
Validação	10% a 20%	10%	Dados subsequente ao treinamento
Teste	10% a 20%	10%	Dados finais

Atenção:

📌 **Nunca dividir os dados aleatoriamente:** Em problemas convencionais de machine learning, é comum dividir os dados de forma aleatória. Em séries temporais, isso é um erro grave, pois os dados passados influenciam diretamente os futuros. A divisão deve sempre seguir a ordem cronológica.

📌 **Evitar vazamento de dados:** O data leakage acontece quando informações do futuro são acidentalmente usadas para treinar o modelo. Como as séries temporais possuem dependência temporal, qualquer vazamento pode gerar previsões artificialmente boas, mas que falham na prática.

📌 **Cuidado com sazonalidade e tendências:** Se a série possui padrões sazonais (exemplo: vendas mensais, temperatura ao longo do ano), é importante que todos os conjuntos contenham pelo menos um ciclo completo para que o modelo aprenda corretamente.

Respeitar essas diretrizes ajuda a garantir que o modelo treinado seja confiável, generalize bem para dados futuros e não sofra com problemas de overfitting ou data leakage.

▼ 3. Atenção

📌 **Nunca dividir os dados aleatoriamente:** Em problemas convencionais de machine learning, é comum dividir os dados de forma aleatória. Em séries temporais, isso é um erro grave, pois os dados passados influenciam diretamente os futuros. A divisão deve sempre seguir a ordem cronológica.

📌 **Evitar vazamento de dados:** O data leakage acontece quando informações do futuro são acidentalmente usadas para treinar o modelo. Como as séries temporais possuem dependência temporal, qualquer vazamento pode gerar previsões artificialmente boas, mas que falham na prática.

📌 **Cuidado com sazonalidade e tendências:** Se a série possui padrões sazonais (exemplo: vendas mensais, temperatura ao longo do ano), é importante que todos os conjuntos contenham pelo menos um ciclo completo para que o modelo aprenda corretamente.

Respeitar essas diretrizes ajuda a garantir que o modelo treinado seja confiável, generalize bem para dados futuros e não sofra com problemas de overfitting ou data leakage.