Métricas de Classificação e Validação Cruzada

Accuracy, Matriz de Confusão, F1-Score e Cross-Validation (com exemplos no Colab)

Objetivos da aula

O que você deve conseguir fazer ao final 🙂

- Entender o que é Accuracy e quando ela pode enganar
- Ler uma Matriz de Confusão e identificar TP, TN, FP e FN
- Calcular e interpretar Precision, Recall e F1-Score
- Aplicar Validação Cruzada (k-fold) para estimar generalização
- Gerar gráficos no Colab para comunicar resultados

🧭 Regra de ouro: uma única métrica raramente conta a história inteira.

Exercício Prático

Classifique as pessoas exibidas a seguir em uma das classes: “feliz” ou “triste”

A

B

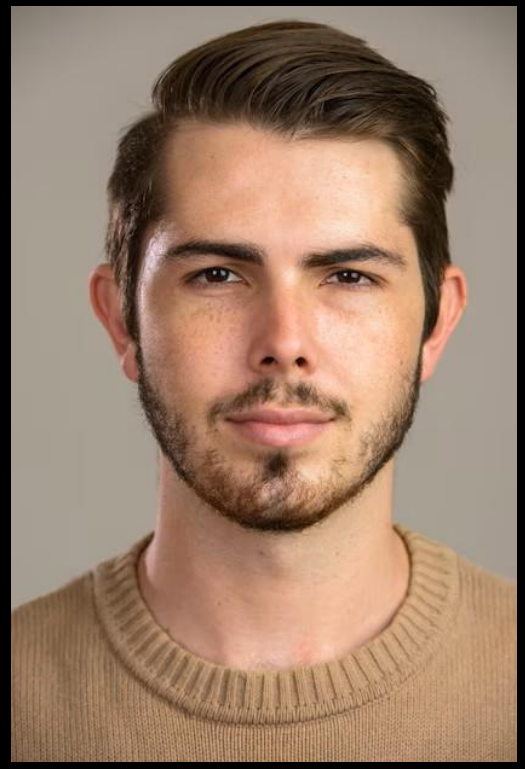
C

D

E

F

G



H

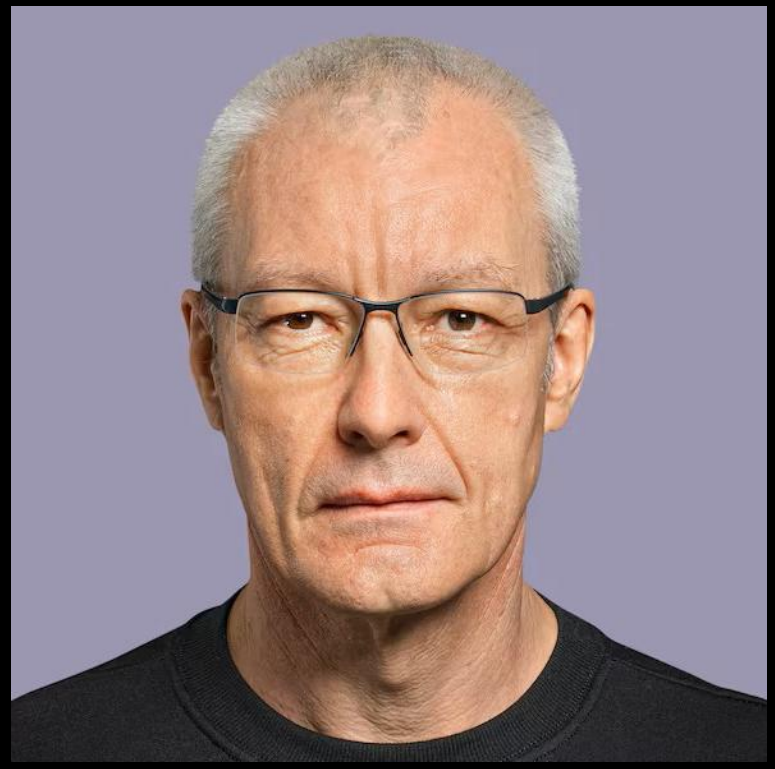
J

K

L

M

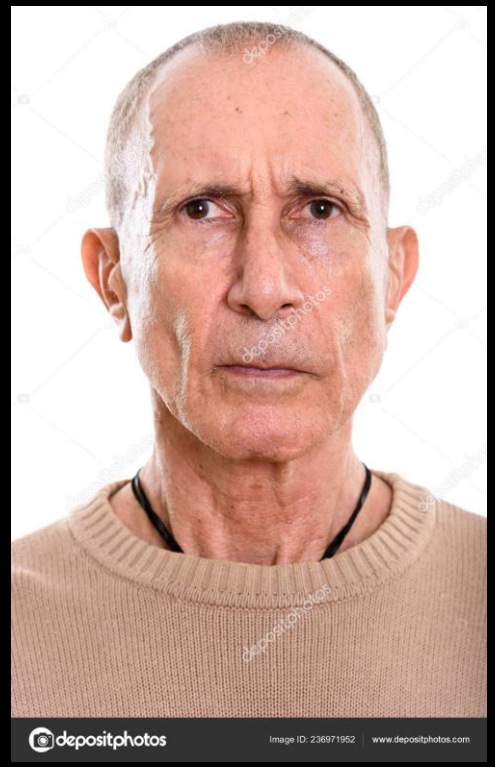
N



o

P

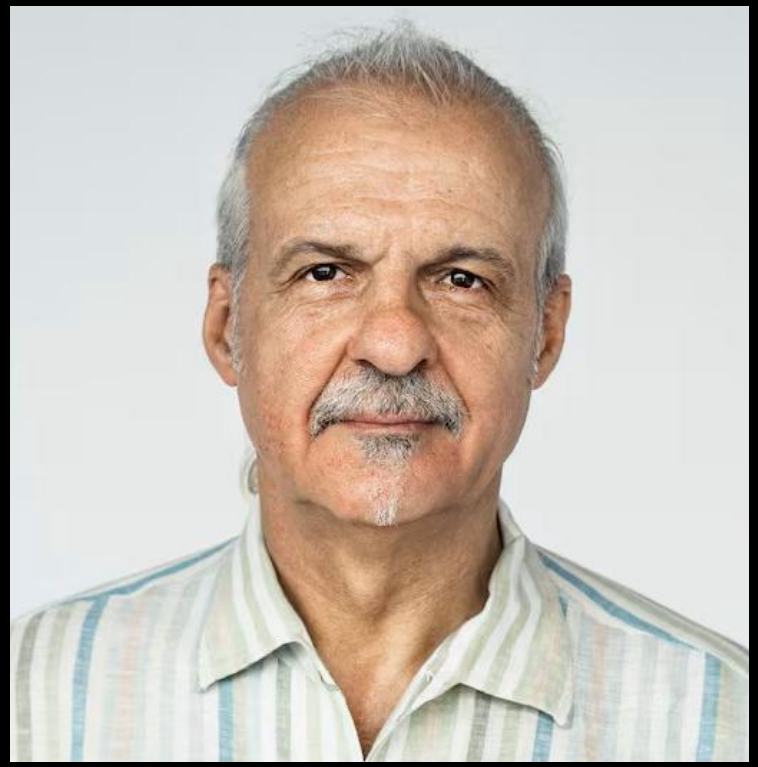
Q



R

S

T



Exercício Prático

Anote a classe real de cada exemplo como: “feliz” ou “triste”

GABARITO

A

FELIZ

B

TRISTE

C

FELIZ

D

TRISTE

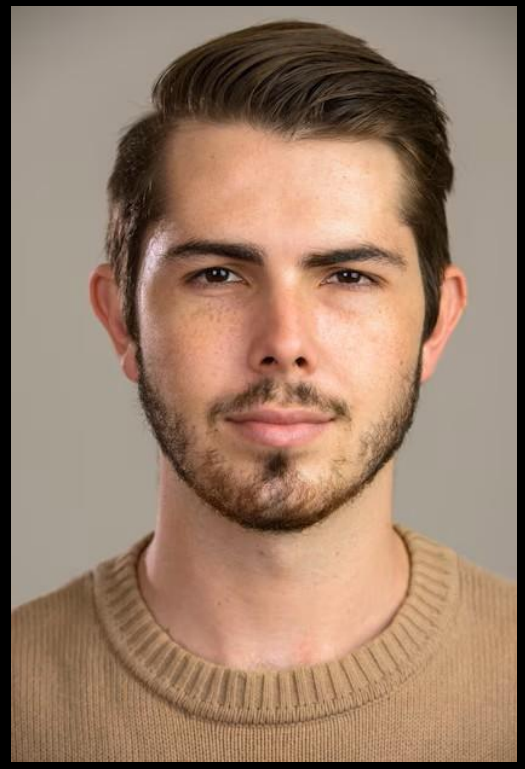
E

TRISTE

F

FELIZ

G



FELIZ

H

FELIZ

I

TRISTE

J

TRISTE

K

FELIZ

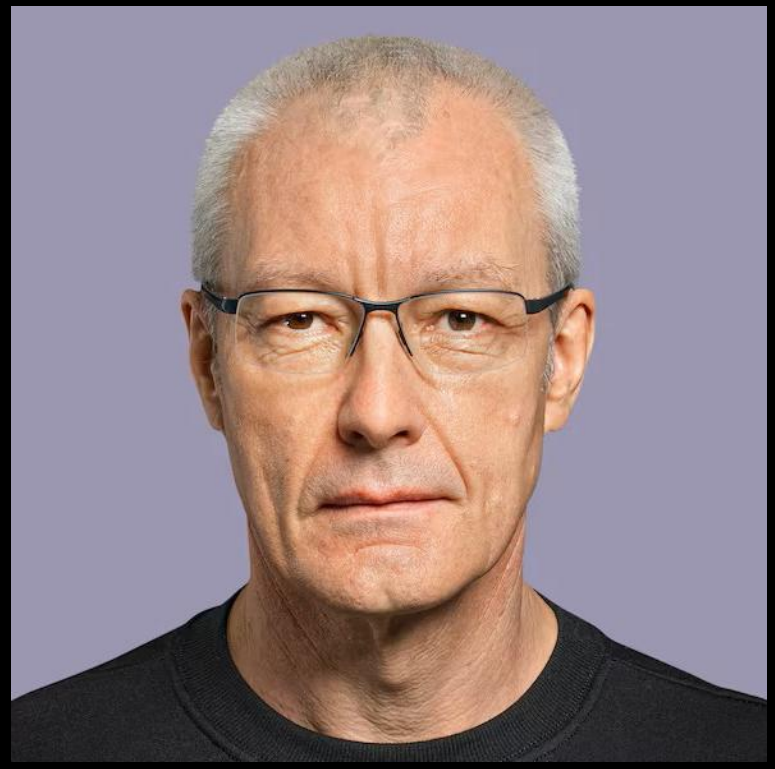
L

FELIZ

M

TRISTE

N



TRISTE

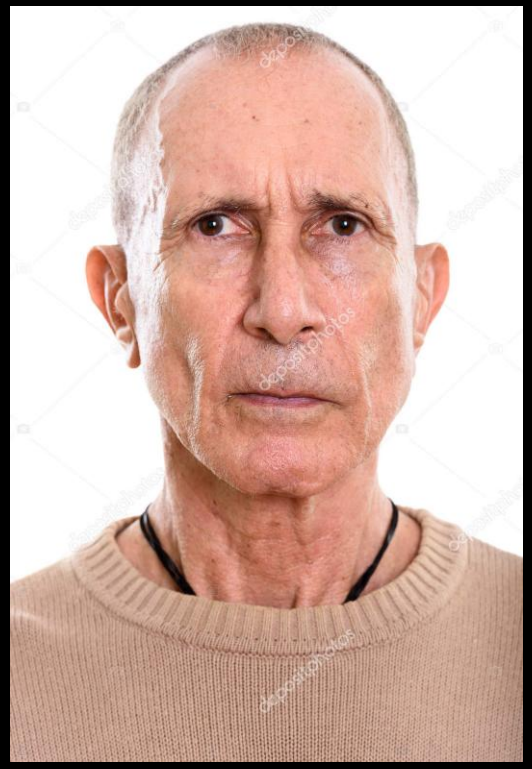
o

FELIZ

P

FELIZ

Q



TRISTE

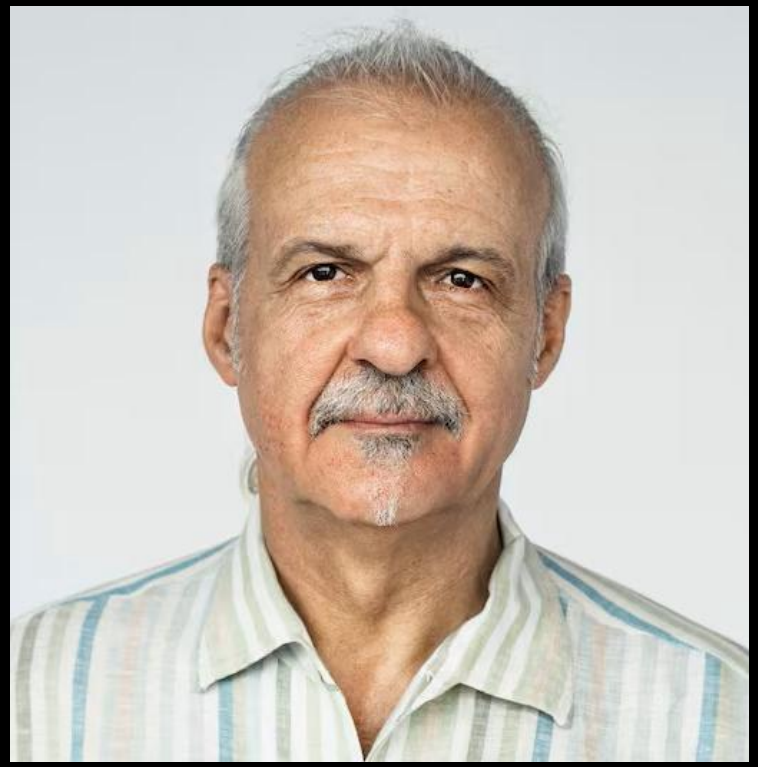
R

TRISTE

S

TRISTE

T



FELIZ

Matriz de Confusão

Matriz de Confusão

Entendendo as nomenclaturas

Tenho um sistema treinado em reconhecer Maças.
Eu apresento para meu sistema uma Maça.

Eu apresento pro sistema um abacaxi.

	Sistema retorna: Maça !	Sistema retorna: Não é Maça !
	 ACERTOU !	 ERROU !
	VERDADEIRO POSITIVO (VP)	FALSO NEGATIVO (FN)
	Sistema retorna: Maça !	Sistema retorna: Não é Maça !
	 ERROU !	 ACERTOU !
	FALSO POSITIVO (FP)	VERDADEIRO NAGATIVO (VN)

Exercício Prático

Monte a sua matriz de confusão

Accuracy (Acurácia)

A porcentagem de acertos no conjunto avaliado

1. Está doente e o teste é positivo: **verdadeiro positivo →** (VP)
2. Está saudável e o teste é negativo: **verdadeiro negativo →** (VN)
3. Está saudável e o teste é positivo: **falso positivo →** (FP)
4. Está doente e o teste é negativo: **falso negativo →** (FN)

$$\textbf{ACURÁCIA:} \quad \frac{\textbf{VP + VN}}{\textbf{TOTAL (VP+VN+FN+FP)}}$$

Imagem de: https://www.blogs.unicamp.br/linhadefundo/2020/06/16/meu-teste-deu-positivo-e-agora-entendendo-a-sensibilidade-e-a-especificidade-dos-testes-diagnosticos/

Accuracy (Acurácia)

A porcentagem de acertos no conjunto avaliado

Definição

```
Accuracy = (VP + VN) / (VP + VN + FP + FN)
```

Quando funciona bem

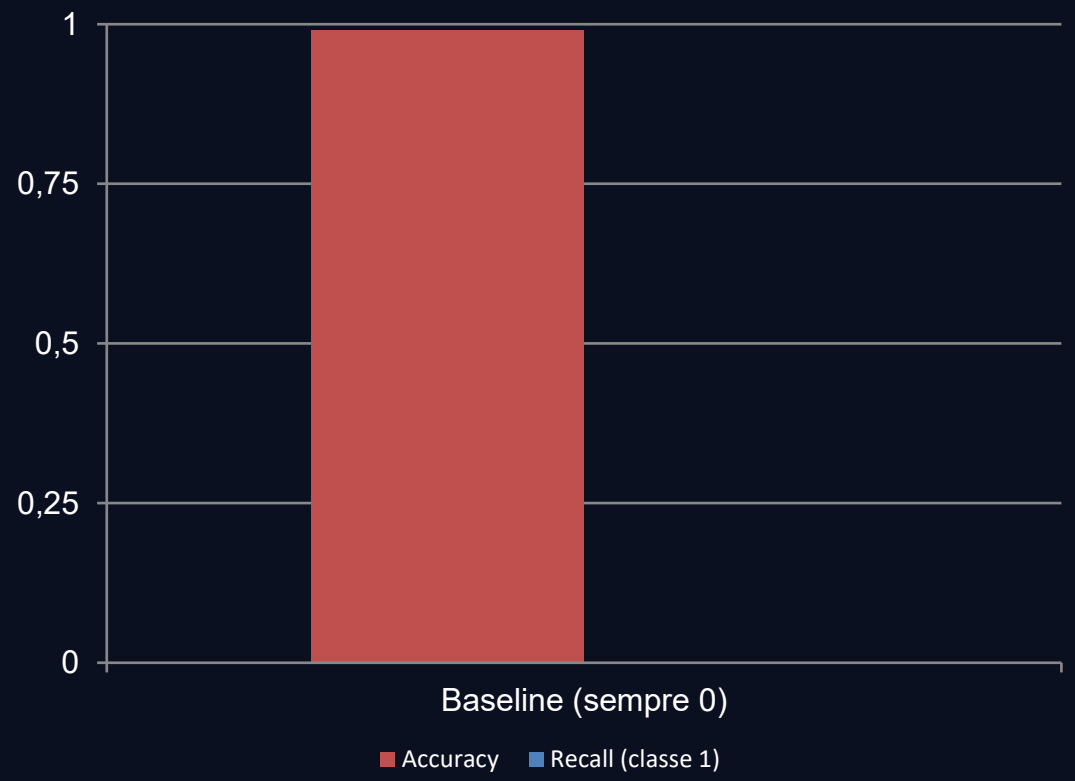
- Boa quando as classes são balanceadas
- Boa quando o custo de FP e FN é parecido

Armadilha clássica

- Dados desbalanceados: 99% da classe 0
- Modelo “sempre 0”: accuracy alta, mas recall da classe 1 = 0

⚠️ Se o seu problema tem “raros importantes”, olhe além da accuracy.

Mini-exemplo (classe rara)



Accuracy é útil, mas não é onisciente. Combine com recall e F1.

Exercício Prático

Calcule a sua acurácia

Matriz de Confusão

O mapa dos acertos e erros por classe

Para binária (classe positiva = 1):

VN (Verdadeiro Negativo)	**FP (Falso Positivo)**
FN (Falso Negativo)	**VP (Verdadeiro Positivo)**

Linha = real | Coluna = previsto

🎯 Cada célula é um tipo de erro. O que custa mais no seu caso: FP ou FN?

Como interpretar

- FP alto: muitos alarmes falsos (custo: tempo, atrito)
- FN alto: muitos casos perdidos (custo: risco, perda)
- Threshold muda o equilíbrio entre FP e FN

Onde isso aparece

- A matriz é base para Precision, Recall (sensibilidade) e F1
- Em multi-classe, vira uma matriz N×N

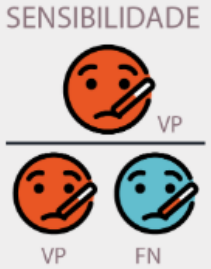
Leia a matriz antes de confiar em qualquer número “bonito”.

Além da Acurácia

Precisão e Sensibilidade

Fórmulas (classe positiva)

Dos positivos reais, quantos eu consegui achar (RECALL) ?



A **Sensibilidade** do teste é a proporção entre o número de doentes que o teste consegue detectar (VP) e o número total de doentes (ND). Em outras palavras, é a probabilidade de o teste ser positivo para uma pessoa doente: P(Teste+doente).

Dos positivos que eu marquei, quantos eram de verdade (PRECISION)?

A **Precisão** (ou valor preditivo positivo) é a relação entre a quantidade de pessoas doentes que testaram positivo (VP) e o número total de testes positivos (N+). Em outras palavras, é a probabilidade de você estar doente, dado que o teste deu positivo: P(Doente/Teste+).

Exercício Prático

Calcule a sua precisão e sensibilidade

F1-Score

Média Harmônica entre precisão e sensibilidade

Fórmulas (classe positiva)

```
Precision = TP / (TP + FP)
Recall    = TP / (TP + FN)
```

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

🧠 Pune o modelo quando um dos dois (precision ou recall) é baixo.

Exemplos práticos

- Spam: priorize precision
- Diagnóstico: priorize recall
- Quando em dúvida: F1 (e olhe a matriz)

Intuição rápida

- F1: equilíbrio quando você precisa dos dois

É crucial para conjuntos de dados desbalanceados, equilibrando o custo de falsos positivos e falsos negativos. A pontuação varia de 0 (péssimo) a 1 (excelente).

Por que usar: Diferente da acurácia, o F1-Score não é enganado se uma classe dominar o conjunto de dados.

Aplicações: Ideal para diagnósticos médicos, recuperação de informações e classificação binária onde o equilíbrio é necessário.

Exercício Prático

Calcule o seu F1-score

Exemplo no Colab: treino e métricas

Dataset: Breast Cancer (scikit-learn)

O que faremos

- Carregar dados prontos (sem download)
- Treinar Logistic Regression com Pipeline
- Gerar predições no teste

⚙️ Rode em um Runtime CPU no Colab. É rápido.

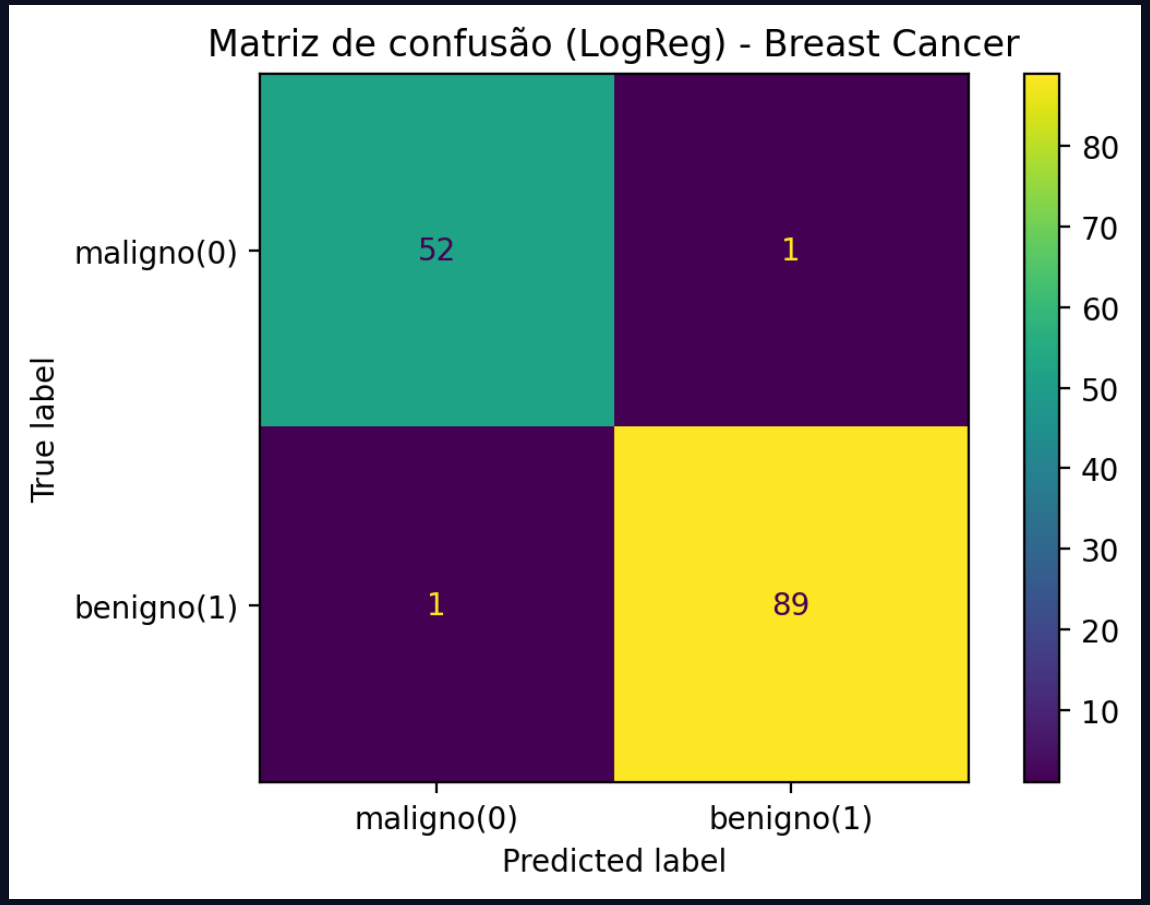
Código (Colab)

```
from sklearn.datasets import load_breast_cancer
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression

X, y = load_breast_cancer(return_X_y=True)
X_tr, X_te, y_tr, y_te = train_test_split(
    X, y, test_size=0.25, stratify=y, random_state=42
)

model = make_pipeline(StandardScaler(),
LogisticRegression(max_iter=2000))
model.fit(X_tr, y_tr)
pred = model.predict(X_te)
```

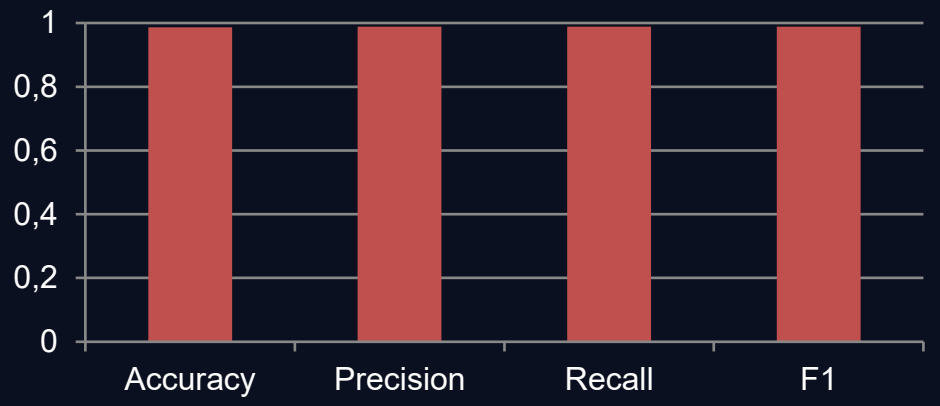
Exemplo: resultados e gráfico

Matriz de confusão + métricas no conjunto de teste

Matriz de confusão (teste)



Métricas (teste)



Código (Colab)

```
from sklearn.metrics import (accuracy_score,
precision_score,
recall_score, f1_score, confusion_matrix)

acc  = accuracy_score(y_te, pred)
prec = precision_score(y_te, pred)
rec  = recall_score(y_te, pred)
f1   = f1_score(y_te, pred)
cm   = confusion_matrix(y_te, pred)

acc, prec, rec, f1, cm
```

Validação Cruzada (k-fold)

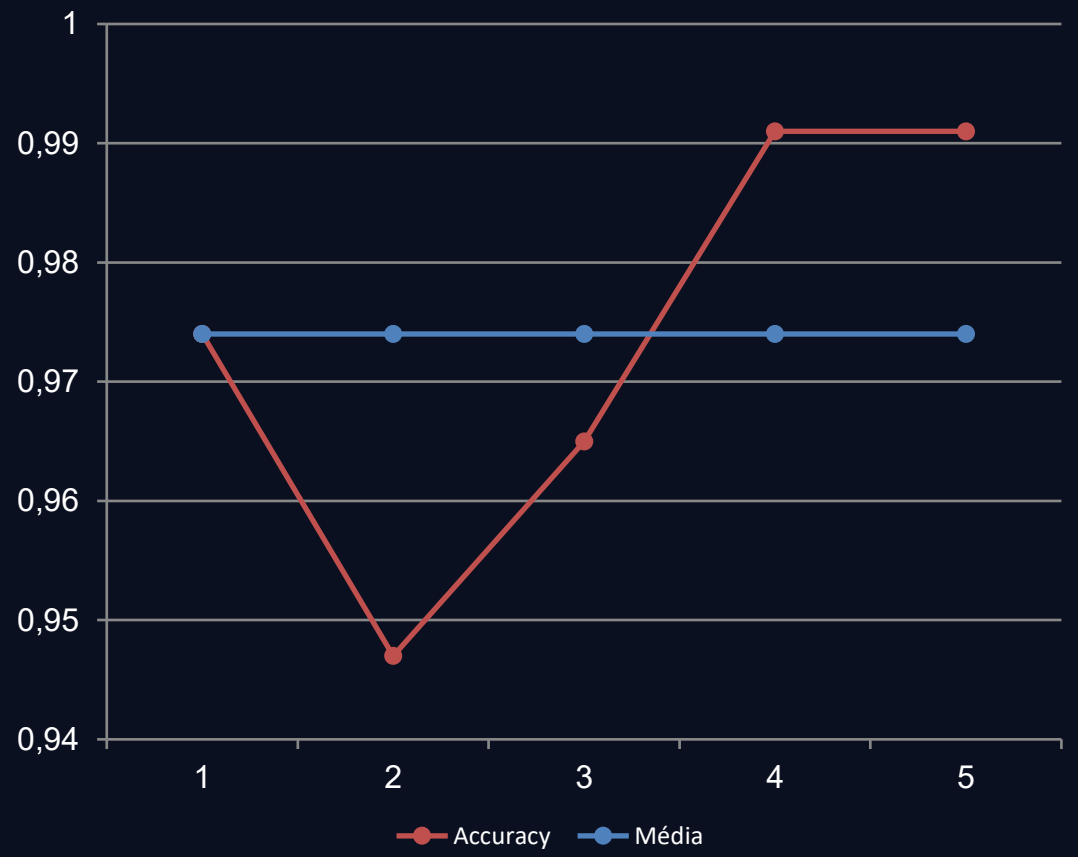
Estimando performance de generalização com menos “sorte”

Ideia

- Divida o dataset em k partes (folds)
- Treine em k-1 folds e teste no fold restante
- Repita k vezes e compute média e desvio padrão
- Use StratifiedKFold quando houver desbalanceamento

🧰 CV é ótima para comparar modelos e hiperparâmetros. Guarde um teste final separado.

Exemplo: Accuracy por fold (k=5)



Olhe média e variância. Dois modelos podem empatar na média e diferir no risco.

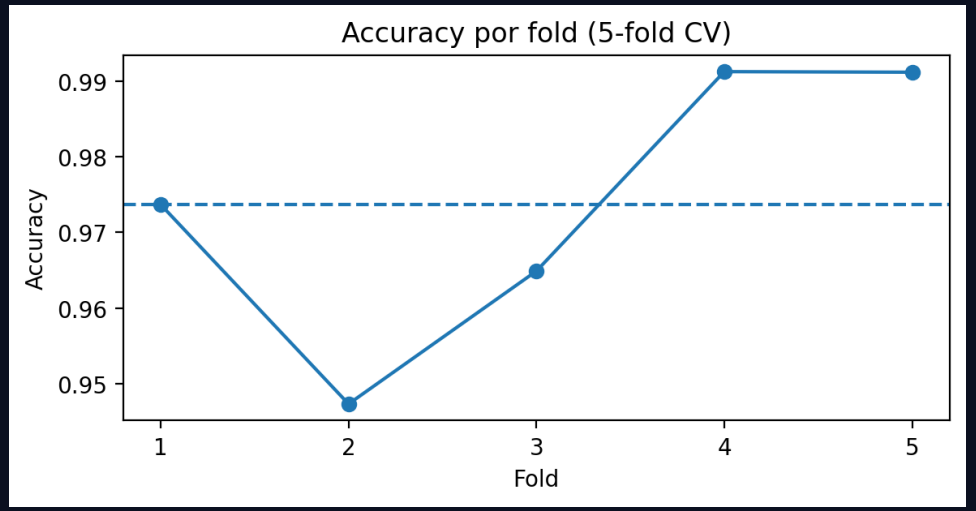
Exemplo no Colab: cross_validate

Calculando métricas em k-fold e gerando gráficos

Código (Colab)

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import StratifiedKFold, cross_validate

cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
res = cross_validate(model, X, y, cv=cv,
                     scoring=['accuracy','precision','recall','f1'])

accs = res['test_accuracy']
print('accuracy média:', accs.mean(), '±', accs.std())

plt.plot(range(1, len(accs)+1), accs, marker='o')
plt.axhline(accs.mean(), linestyle='--')
plt.xticks(range(1, len(accs)+1))
plt.xlabel('Fold')
plt.ylabel('Accuracy')
plt.title('Accuracy por fold (CV)')
plt.show()
```

Gráfico (exemplo)



Dica: salve os resultados em um DataFrame para facilitar gráficos mais bonitos.

Exercício (mão na massa)

Dataset: Wine (3 classes) para vocês criarem seus próprios gráficos

Tarefas

- Carregue o dataset Wine e faça um train/test split estratificado
- Treine um modelo (LogReg, RandomForest, SVM...)
- Calcule accuracy e matriz de confusão (3×3) e plote o heatmap
- Calcule F1 macro e F1 weighted e compare
- Faça validação cruzada (k=5) para accuracy e F1 macro
- Gere 2 gráficos: matriz de confusão e performance por fold

Starter code (Colab)

```python
from sklearn.datasets import load_wine
from sklearn.model_selection import train_test_split,
StratifiedKFold, cross_validate
from sklearn.metrics import ConfusionMatrixDisplay, f1_score
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt

X, y = load_wine(return_X_y=True)
X_tr, X_te, y_tr, y_te = train_test_split(
    X, y, test_size=0.25, stratify=y, random_state=42
)

clf = RandomForestClassifier(random_state=42)
clf.fit(X_tr, y_tr)
pred = clf.predict(X_te)

# 1) Matriz de confusão
ConfusionMatrixDisplay.from_predictions(y_te, pred)
plt.title('Wine: matriz de confusão')
plt.show()

# 2) F1 macro vs weighted
print('F1 macro   :', f1_score(y_te, pred, average='macro'))
print('F1 weighted:', f1_score(y_te, pred, average='weighted'))

# 3) CV + gráfico por fold
cv = StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
res = cross_validate(clf, X, y, cv=cv,
scoring=['accuracy','f1_macro'])

plt.plot(res['test_accuracy'], marker='o', label='accuracy')
plt.plot(res['test_f1_macro'], marker='o', label='f1_macro')
plt.legend(); plt.title('Performance por fold')
plt.xlabel('Fold'); plt.show()
```

Boas práticas (checklist rápido)

Para não cair em “números bonitos” 😳

- Separe um conjunto de teste final e não “olhe” para ele durante ajustes
- Use Pipeline para evitar vazamento de dados (scaler só com treino)
- Sempre inspecione a matriz de confusão (por classe)
- Em desbalanceamento: prefira F1, precision/recall e métricas macro
- Em CV: reporte média e desvio padrão, não só a média
- Escolha a métrica alinhada ao custo real do erro (FP vs FN)

Resumo em 4 frases

- Accuracy é um resumo, não um veredito.
- A matriz de confusão mostra onde o modelo erra.
- F1 combina precision e recall quando ambos importam.
- Validação cruzada reduz a dependência de um único split.

Fim.