CURSO BÁSICO DE INTELIGÊNCIA ARTIFICIAL E BATE-PAPO COM CONVIDADOS ESPECIAIS

INTELIGÊNCIA ARTIFICIAL PARA TODOS

DE 08/06 A 12/08

COM OS PROFESSORES DO LABORATÓRIO ARIA/UFPB: TELMO FILHO, THAÍS GAUDENCIO E YURI MALHEIROS

Centro de Informática UFPB

Departamento de ESTATÍSTICA

aria artificial intelligence applications

- CURSO SEM PRÉ-REQUISITOS
- HTTP://ARIA.CI.UFPB.BR/IAPARATODOS
- INSCRIÇÃO PARA CERTIFICADO - DE 01/06 A 07/06: HTTP://BIT.LY/SIGEVENTOS
- ENCONTROS: SEGUNDAS E QUARTAS
- HORÁRIO: 19:00 ÀS 20:00

aria

Início Sobre Projetos Membros Parceiros Publicações Contato

LABORATÓRIO DE APLICAÇÕES EM INTELIGÊNCIA ARTIFICIAL

As experiências definem a aprendizagem. Assim, o ARIA constrói experiências para máquinas e para pessoas, formando especialistas na área de inteligência artificial e ciência de dados, desenvolvendo aplicações e pesquisando seus métodos.

SAIBA MAIS

SE INSCREVE E JÁ APERTA NO SININHO, QUE VOCÊS PASSAM A RECEBER AS NOTIFICAÇÕES.

NOSSOS ENCONTROS DURARÃO 1 HORA E, ASSIM QUE POSSÍVEL, DEIXAREMOS OS VÍDEOS GRAVADOS NO CANAL.

NÃO PRECISA SE PREOCUPAR EM ESTAR LIGADO ÀS 19:00, MAS ESTANDO, ROLA TIRAR DÚVIDA E PARTICIPAR, O QUE JÁ DEIXA A AULA MAIS ANIMADA.

SOBRE O MATERIAL DE ACOMPANHAMENTO: O ALUNO PRECISA SE LOGAR EM:
CLASSROOM.GOOGLE.COM
DEPOIS, CLICAR EM PARTICIPAR DA TURMA (ÍCONE COM UM MAIS +)

POR FIM, ENTRAR COM O CÓDIGO DA TURMA: PXV3ANW

TAMBÉM EM: HTTPS://ARIA.CI.UFPB.BR/IA-PARA-TODOS-MATERIAL/
ESPERAMOS QUE VOCÊS REALMENTE CURTAM O CURSO E APROVEITEM-NO AO MÁXIMO. NÃO DEIXEM DE INTERAGIR CONOSCO, TAMBÉM, POR E-MAIL OU MENSAGEM NO NOSSO INSTAGRAM (@APRENDIZAGEMDEMAQUINA)

Avaliação de modelos

Agora que sabemos treinar modelos de aprendizagem de máquina, como sabemos se um modelo é bom em resolver uma tarefa?

Como comparar modelos?

Avaliação de modelos

Agora que sabemos treinar modelos de aprendizagem de máquina, como sabemos se um modelo é bom em resolver uma tarefa?

Como comparar modelos?

Avaliação de modelos

No fim das contas, queremos saber quão bem nossos modelos generalizam o que aprenderam quando vêem novos dados

Avaliação de modelos

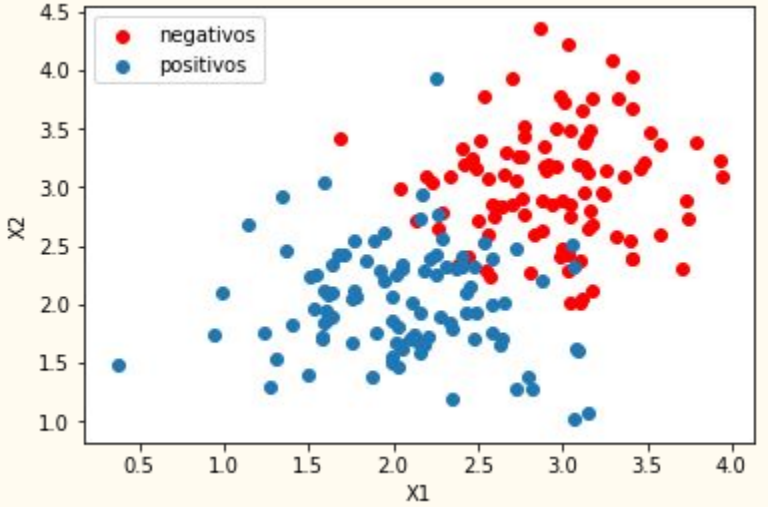
No fim das contas, queremos saber quão bem nossos modelos generalizam o que aprenderam quando vêem novos dados

Conjunto de teste

Classificação

Avaliação de classificadores binários

Como vimos antes, um problema binário contém apenas duas classes: positiva e negativa

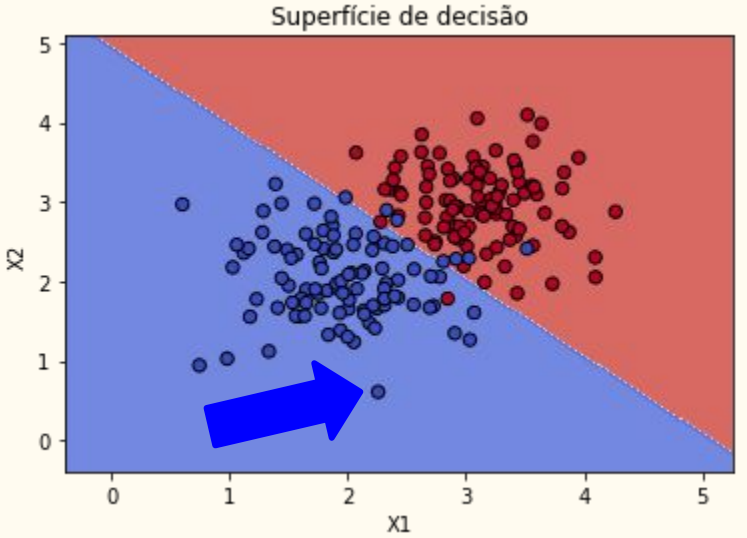


Avaliação de classificadores binários

Quando perguntamos a um classificador binário qual é a classe de um indivíduo, temos quatro resultados possíveis:

Verdadeiro positivo (TP)

Positivo identificado como **positivo**

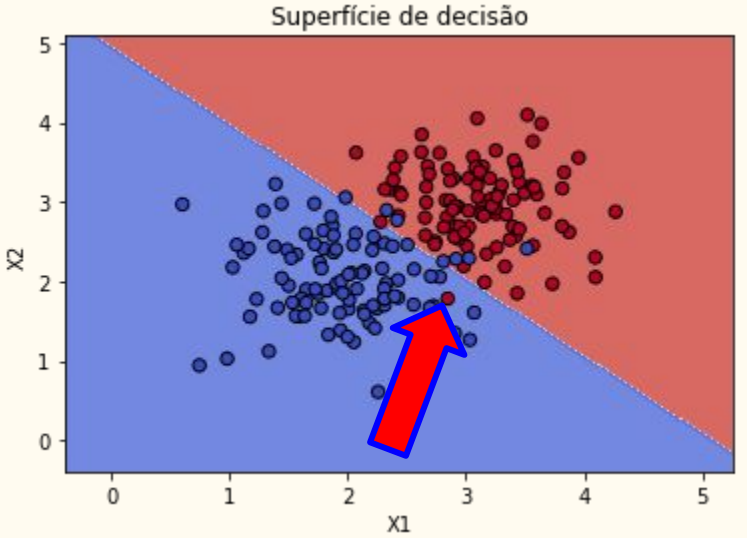


Avaliação de classificadores binários

Quando perguntamos a um classificador binário qual é a classe de um indivíduo, temos quatro resultados possíveis:

Falso positivo (FP) (erro tipo I)

Negativo identificado como **positivo**

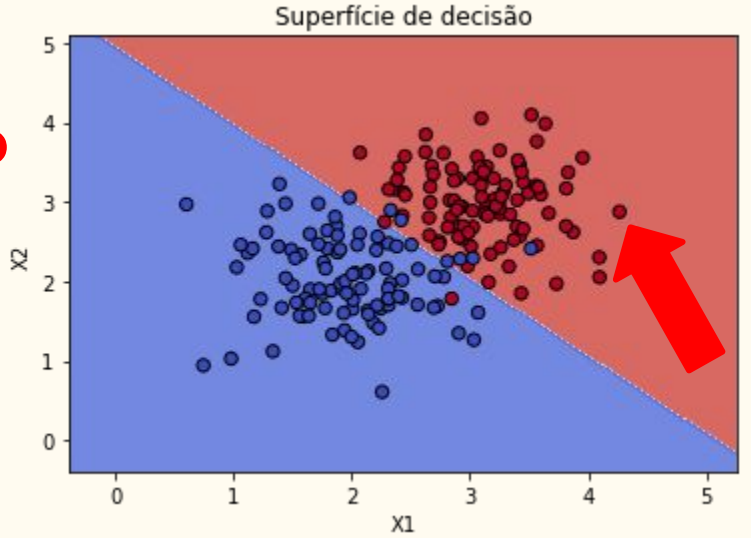


Avaliação de classificadores binários

Quando perguntamos a um classificador binário qual é a classe de um indivíduo, temos quatro resultados possíveis:

Verdadeiro negativo (TN)

Negativo identificado como **negativo**

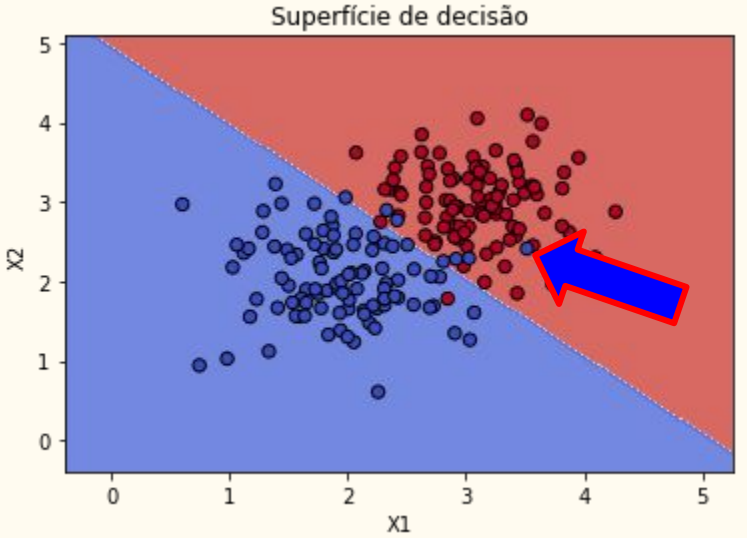


Avaliação de classificadores binários

Quando perguntamos a um classificador binário qual é a classe de um indivíduo, temos quatro resultados possíveis:

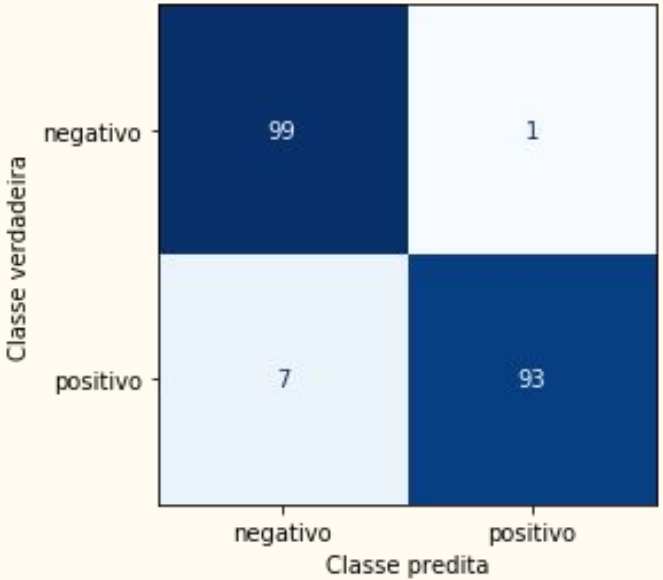
Falso negativo (FN) (erro tipo II)

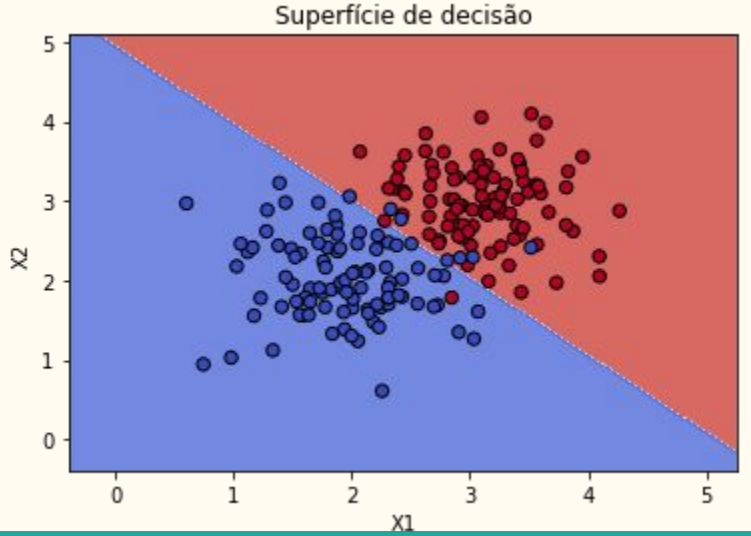
Positivo identificado como **negativo**



Matriz de confusão

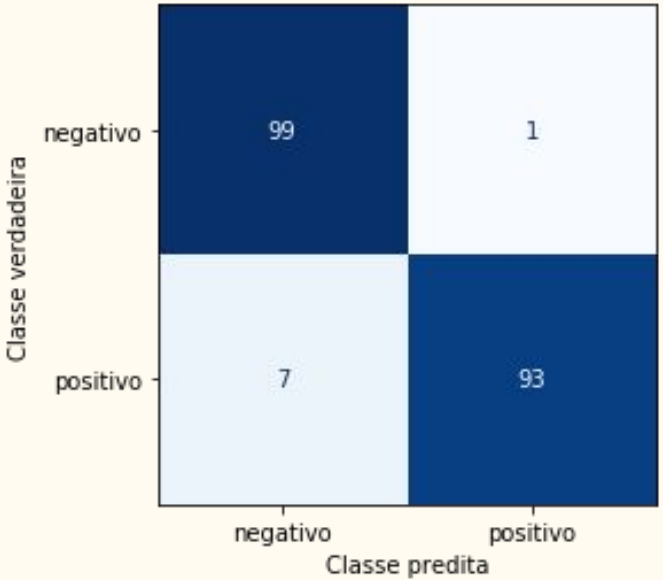
As quantidades desses quatro resultados possíveis costumam ser apresentadas em uma **matriz de confusão**



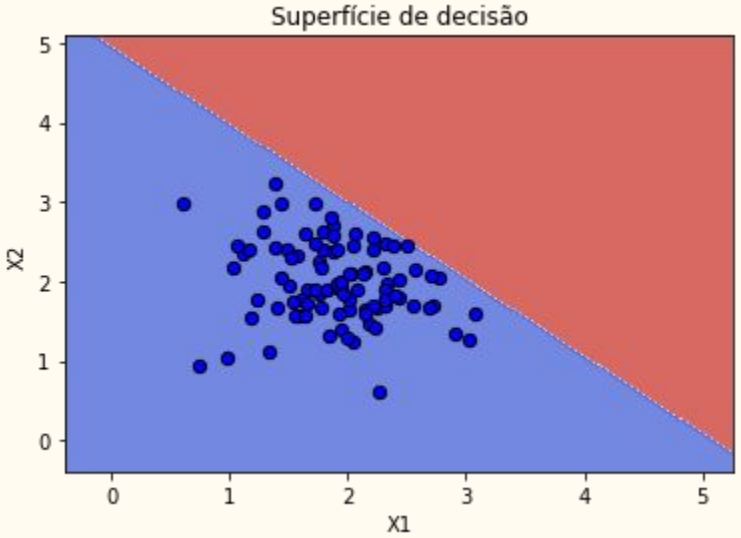


Matriz de confusão

As quantidades desses quatro resultados possíveis costumam ser apresentadas em uma **matriz de confusão**

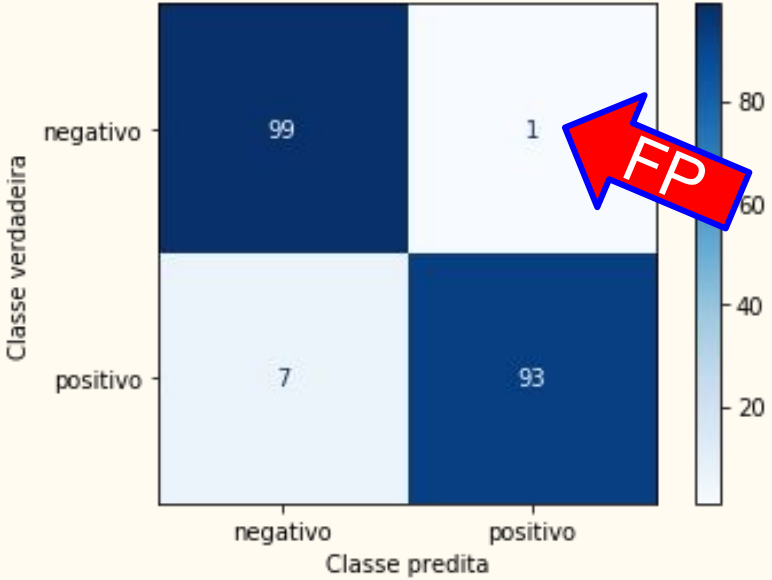


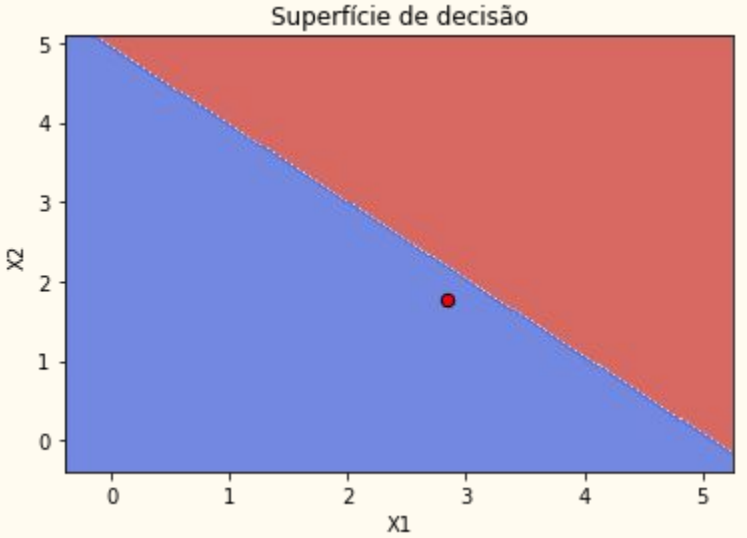
TP



Matriz de confusão

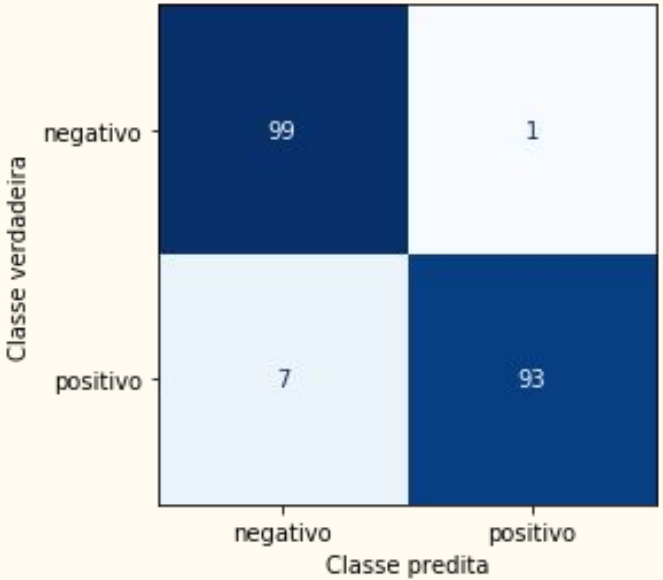
As quantidades desses quatro resultados possíveis costumam ser apresentadas em uma **matriz de confusão**

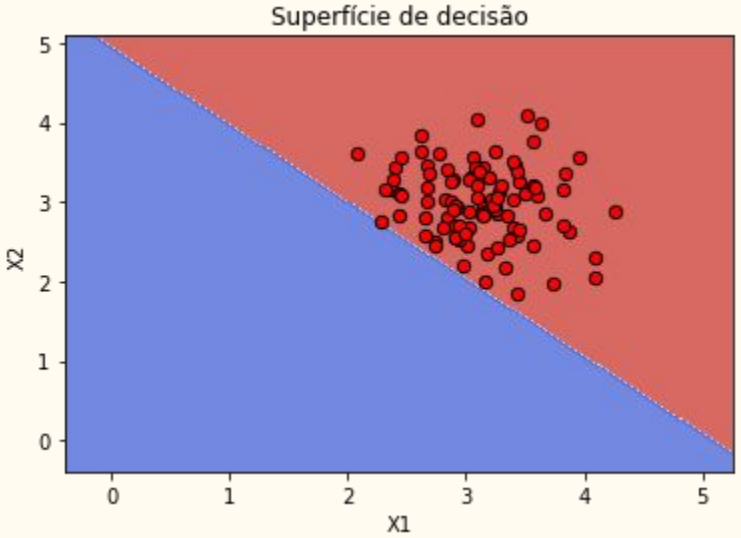




Matriz de confusão

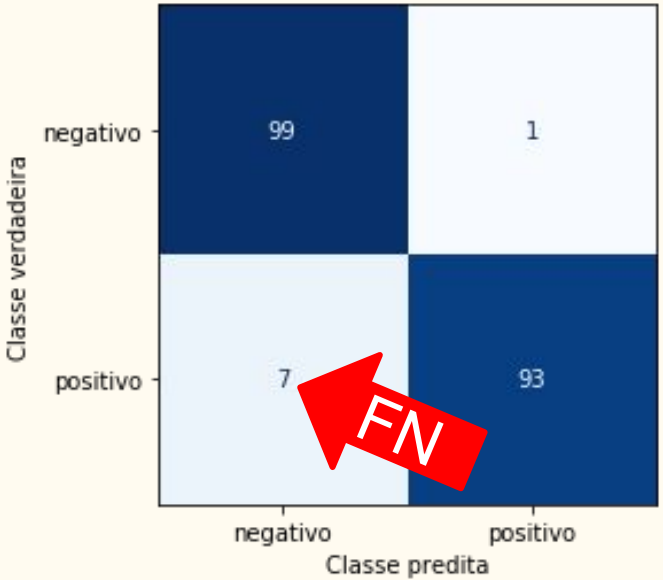
As quantidades desses quatro resultados possíveis costumam ser apresentadas em uma **matriz de confusão**

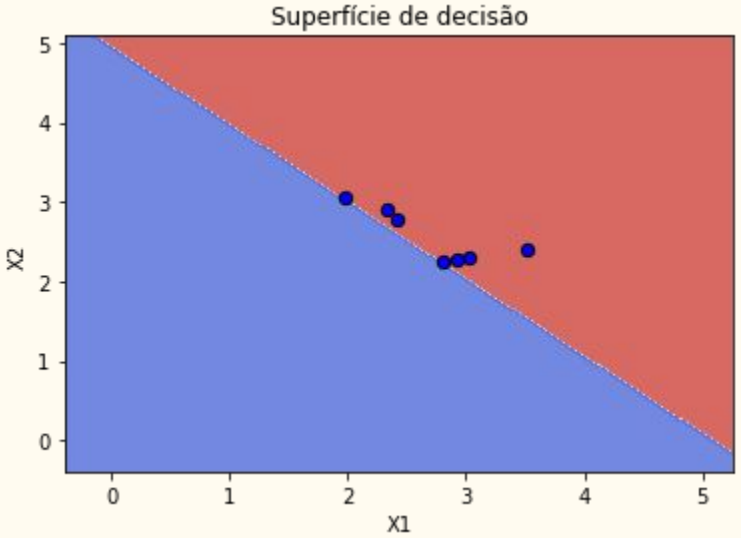




Matriz de confusão

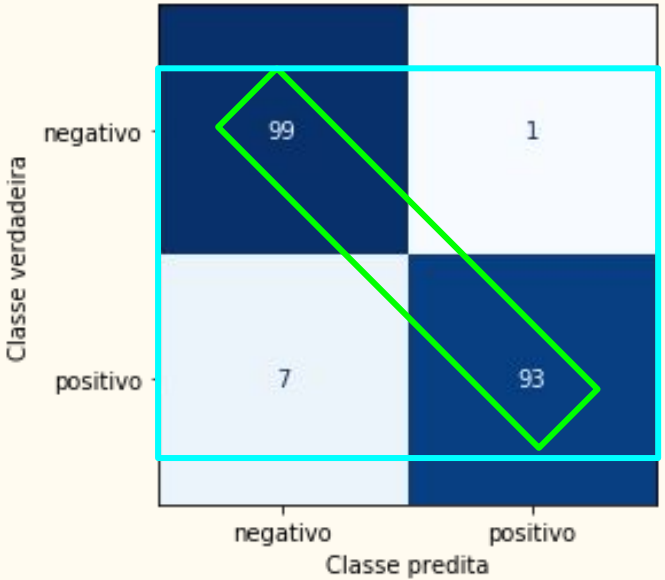
As quantidades desses quatro resultados possíveis costumam ser apresentadas em uma **matriz de confusão**





Acurácia

Os valores de TP, FP, TN e FN podem ser usados para calcular medidas de performance, como a Acurácia



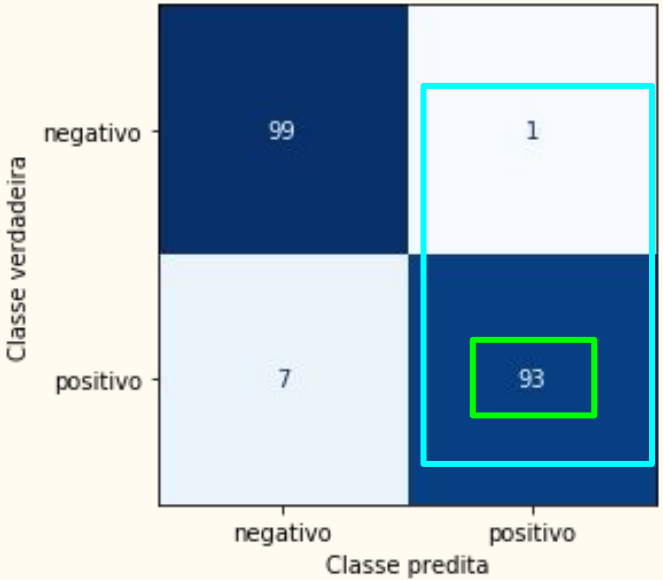
$$Acc = \frac{TP+TN}{TP+FP+TN+FN}$$

$$Acc = \frac{93+99}{93+1+99+7}$$

$$Acc = \frac{192}{200} = 0{,}96$$

Precisão

Entre as minhas predições positivas, quantos indivíduos são realmente positivos?



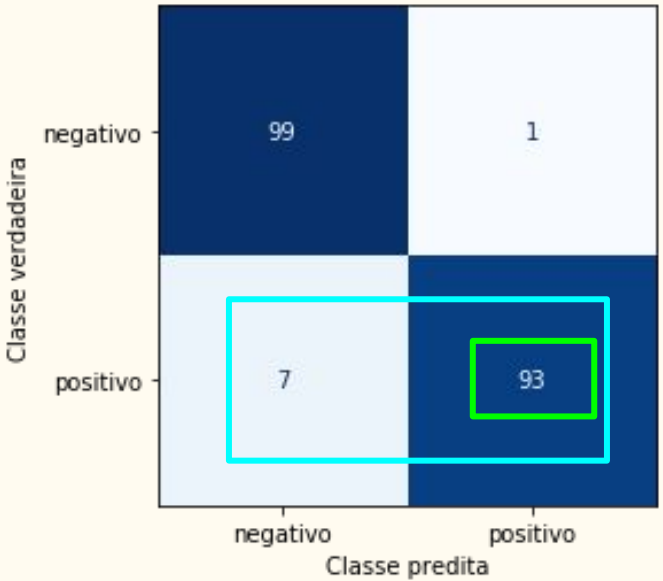
$$Prec = \frac{TP}{TP+FP}$$

$$Prec = \frac{93}{93+1}$$

$$Prec = \frac{93}{94} = 0{,}99$$

Cobertura, revocação, sensibilidade (recall)

Entre os meus indivíduos positivos, quantos eu identifiquei corretamente?



$$Rec = \frac{TP}{TP+FN}$$

$$Rec = \frac{93}{93+7}$$

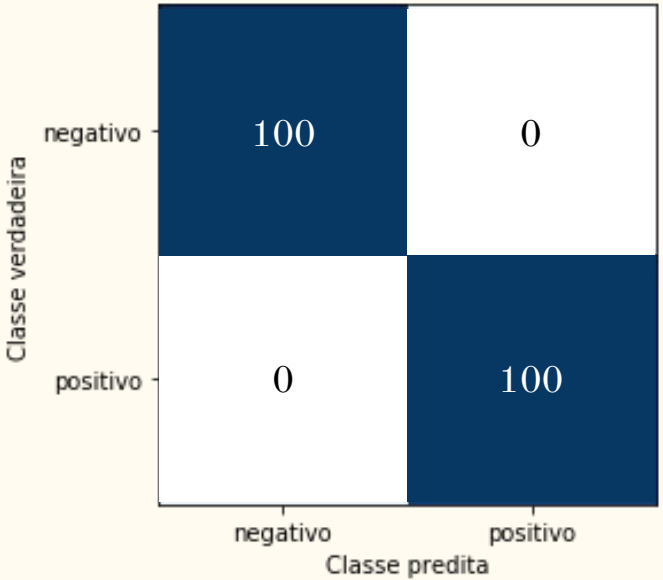
$$Rec = \frac{93}{100} = 0{,}93$$

Muitas outras medidas

		True condition			
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$
Predicted condition	Predicted condition positive	**True positive**	**False positive**, Type I error	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$
	Predicted condition negative	**False negative**, Type II error	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$
		True positive rate (TPR), Recall, Sensitivity, probability of detection, Power = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out, probability of false alarm = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	Specificity (SPC), Selectivity, True negative rate (TNR) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$	F_1 score = $2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

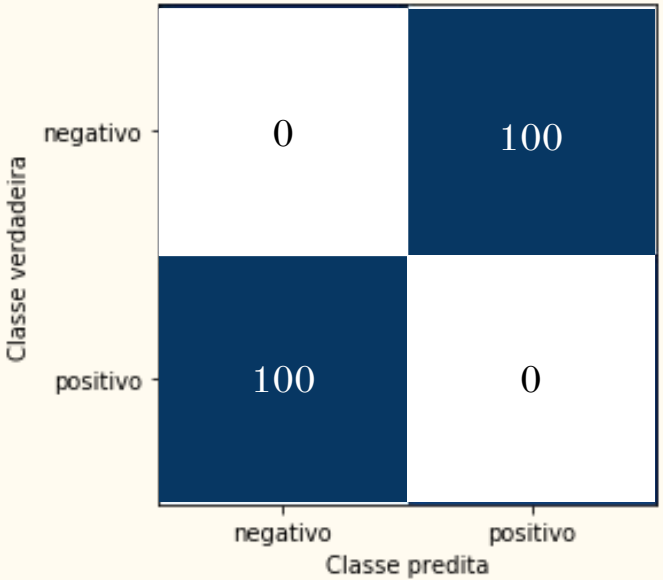
O sonho da matriz de confusão perfeita

Isso pode acontecer em treinamento, mas é muito improvável em teste e costuma indicar sobreajuste (*overfitting*)



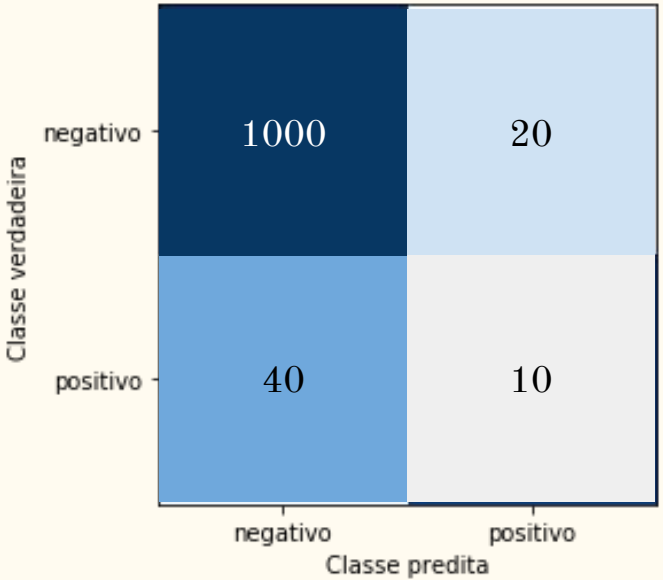
O classificador mentiroso

Se um classificador binário hipotético **sempre errar**, é possível inverter suas predições e **sempre acertar**



Identificando problemas na matriz de confusão

Neste exemplo, temos alta acurácia: só erramos 60 elementos de 1070



$$Acc = \frac{TP+TN}{TP+FP+TN+FN}$$

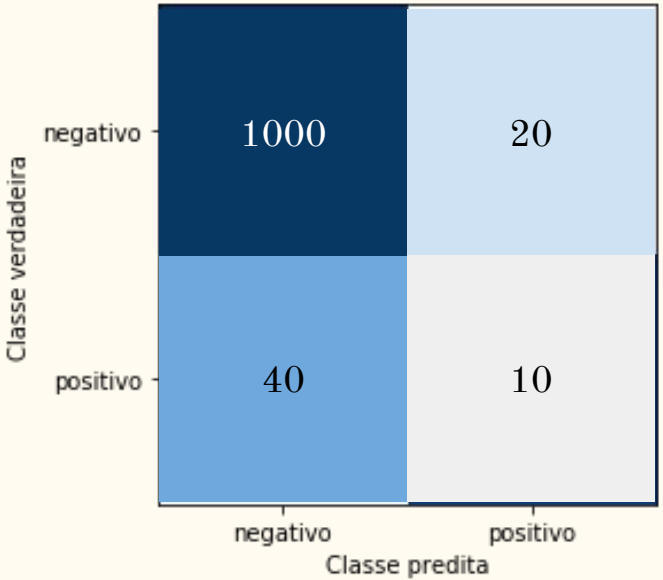
$$Acc = \frac{10+1000}{10+20+1000+40}$$

$$Acc = \frac{1010}{1070}$$

$$Acc = 0{,}94$$

Identificando problemas na matriz de confusão

Vamos calcular a precisão e a cobertura (*recall*), para avaliar o desempenho em relação à classe positiva



$$Prec = \frac{TP}{TP+FP}$$

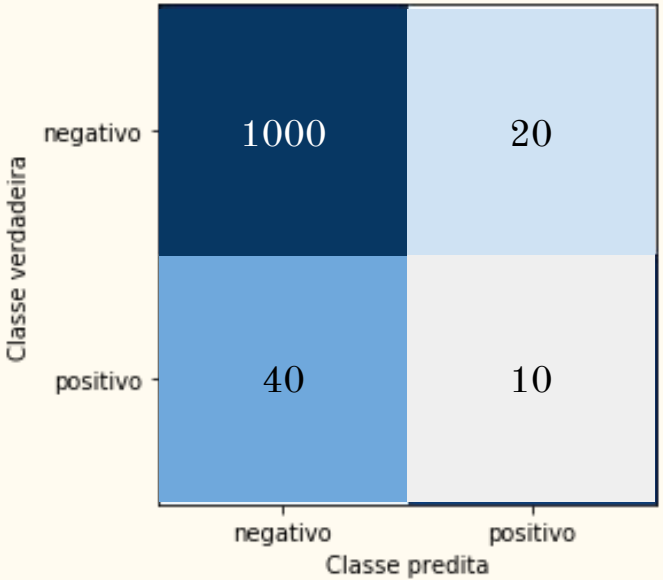
$$Prec = \frac{10}{10+20} = 0{,}33$$

$$Rec = \frac{TP}{TP+FN}$$

$$Rec = \frac{10}{10+40} = 0{,}2$$

Identificando problemas na matriz de confusão

Apesar da alta acurácia, o classificador tem péssimo desempenho para a classe positiva



$$Prec = \frac{TP}{TP+FP}$$

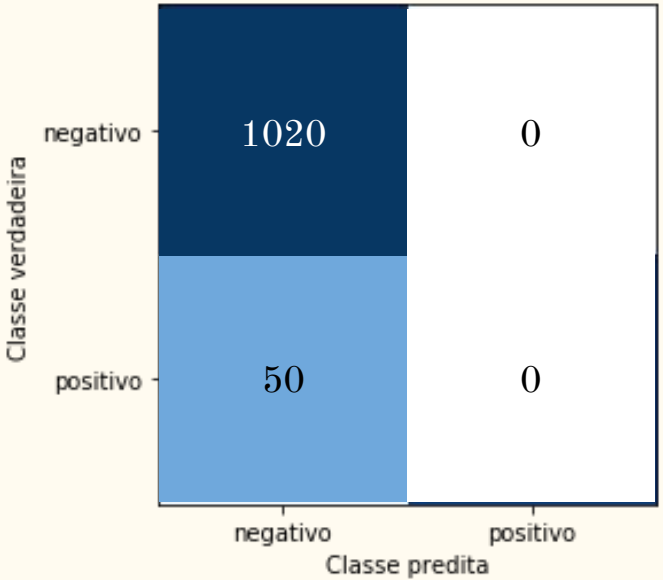
$$Prec = \frac{10}{10+20} = 0{,}33$$

$$Rec = \frac{TP}{TP+FN}$$

$$Rec = \frac{10}{10+40} = 0{,}2$$

Identificando problemas na matriz de confusão

Na verdade, um classificador que apenas chute que todos os elementos são negativos tem maior acurácia



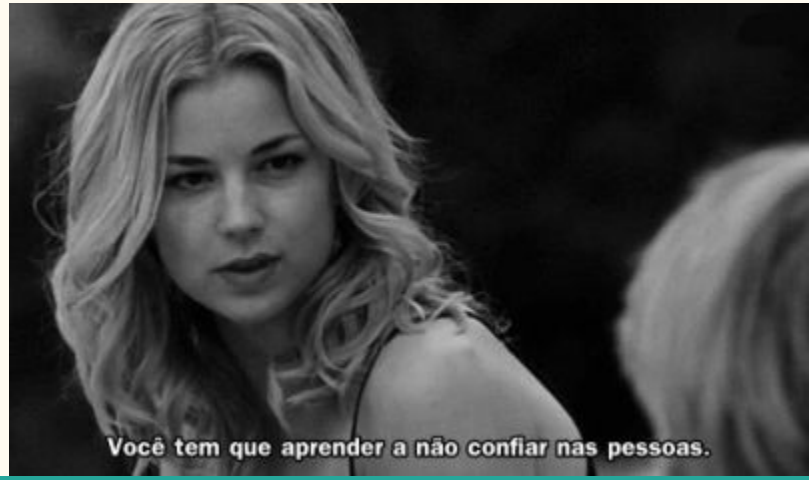
$$Acc = \frac{TP+TN}{TP+FP+TN+FN}$$

$$Acc = \frac{1020}{1070}$$

$$Acc = 0{,}95$$

Identificando problemas na matriz de confusão

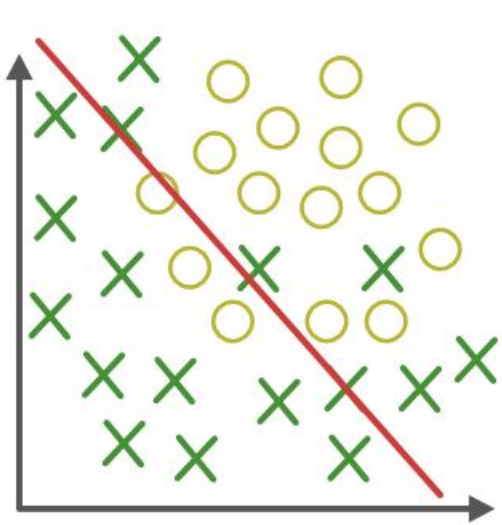
Portanto, é extremamente importante não **confiar cegamente** no resultado de acurácia do seu classificador, principalmente se uma classe for bem maior do que outra



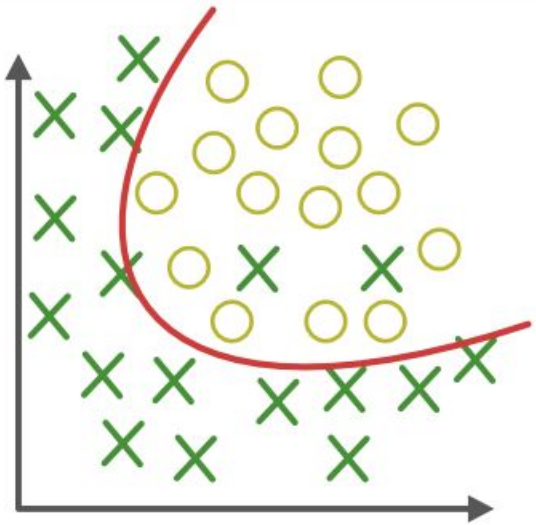
Identificando problemas na matriz de confusão

Uma boa regra é ter em mente as proporções das classes no conjunto de dados: você deve esperar que seu classificador tenha **acurácia pelo menos maior do que a proporção da maior classe**

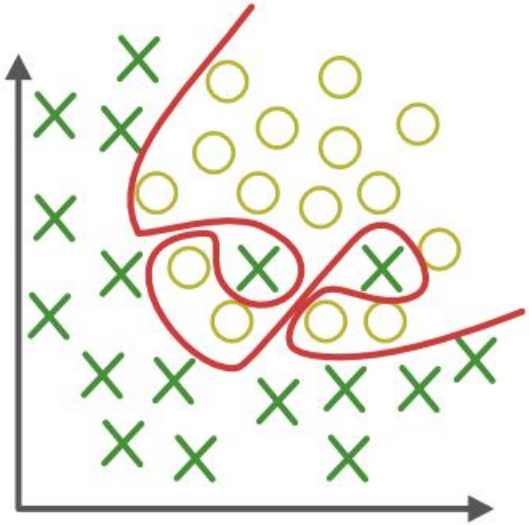
Over/underfitting (Sobre/sub-ajuste)



Sub-ajuste
(Simples demais pra explicar a variância)



Ajuste apropriado



Sobre-ajuste
(Bom demais pra ser verdade)

Over/underfitting (Sobre/sub-ajuste)

Essas condições de ajuste podem ser detectadas se compararmos as acurácias ou matrizes de confusão de treinamento e de teste

Underfitting mostrará desempenhos de treinamento e teste parecidos e ruins

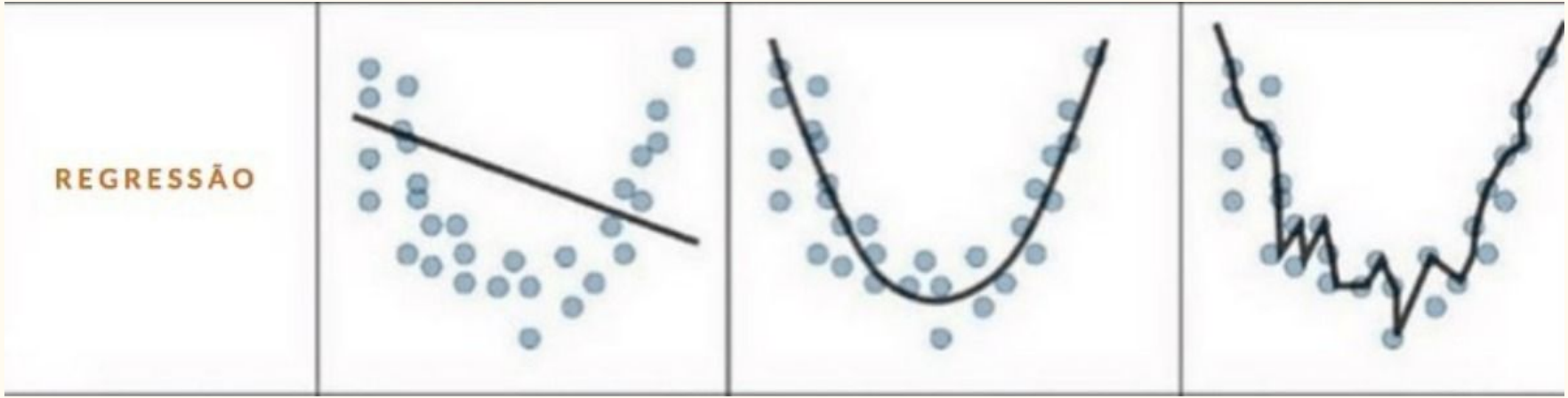
Over/underfitting (Sobre/sub-ajuste)

Sobre-ajuste aparecerá como um desempenho de treinamento muito bom (às vezes perfeito) acompanhado de desempenho de teste péssimo

Over/underfitting (Sobre/sub-ajuste)

Ajuste apropriado aparece como desempenhos de treinamento e teste parecidos e razoáveis/bons

Over/underfitting em regressão



Sub-ajuste
(Simples demais pra explicar a variância)

Ajuste apropriado

Sobre-ajuste
(Bom demais pra ser verdade)

Fronteira de decisão

Classificadores binários probabilísticos decidem suas predições com base na seguinte regra:

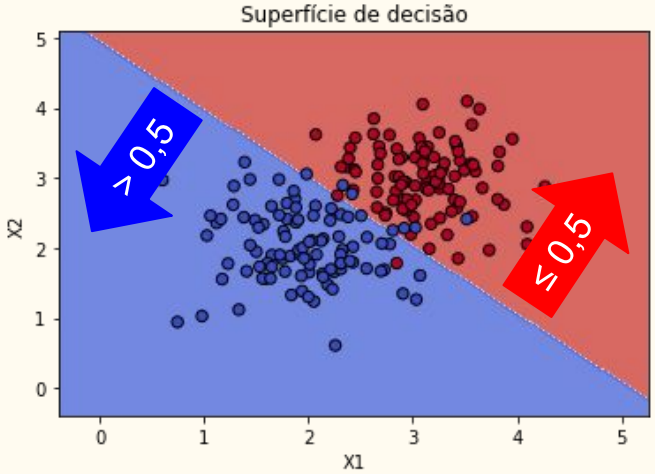
$$\text{Predição} = \begin{cases} +, \text{ se } \mathrm{P}(\mathrm{Y} = + \,|\mathrm{X} = \mathbf{x}) > \mathrm{P}(\mathrm{Y} = - \,|\mathrm{X} = \mathbf{x}) \\ -, \text{ se } \mathrm{P}(\mathrm{Y} = + \,|\mathrm{X} = \mathbf{x}) \leq \mathrm{P}(\mathrm{Y} = - \,|\mathrm{X} = \mathbf{x}) \end{cases}$$

Essencialmente, isso quer dizer que o indivíduo será positivo se

$$\mathrm{P}(\mathrm{Y} = + \,|\mathrm{X} = \mathbf{x}) > 0{,}5$$

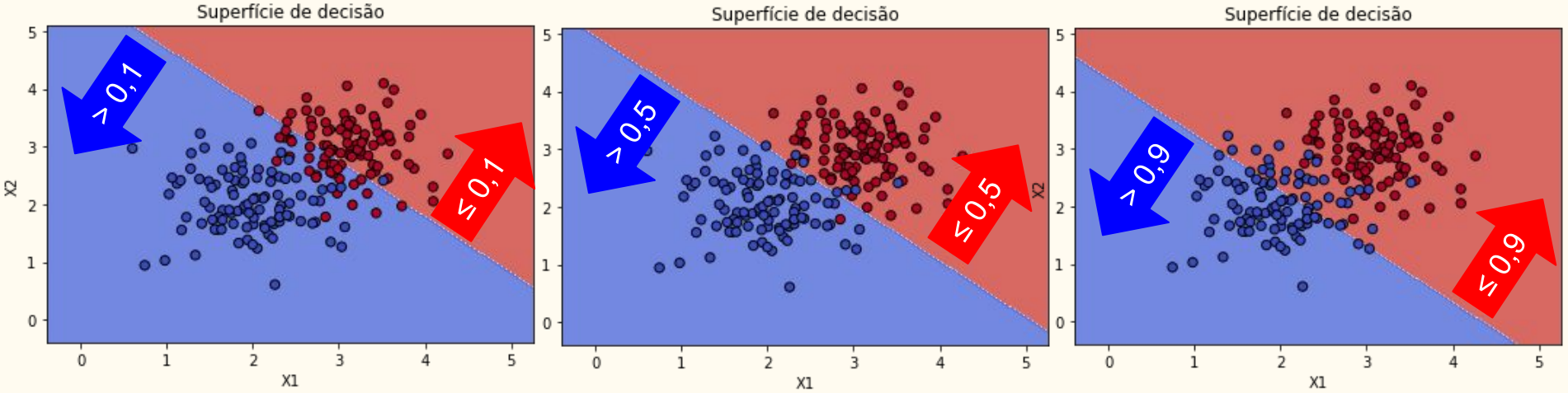
Fronteira de decisão

Dessa forma, 0,5 define um limiar, formando uma fronteira de decisão, como podemos ver na imagem abaixo



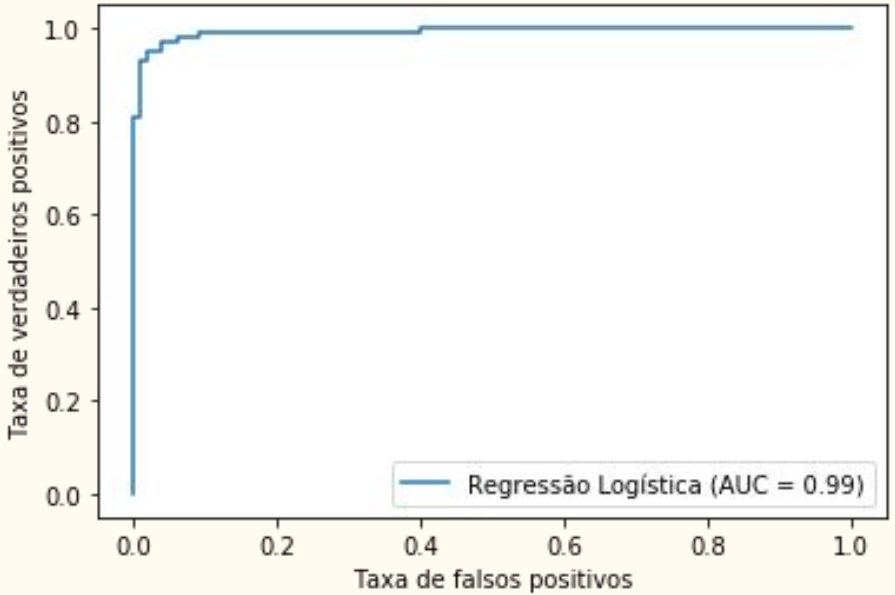
Fronteira de decisão

Essa fronteira de decisão pode ser movida para obter valores quantidades diferentes de Verdadeiros/Falsos positivos



Curva ROC

As diferentes taxas de verdadeiros/falsos positivos de acordo com os limiares podem ser visualizadas usando a curva ROC

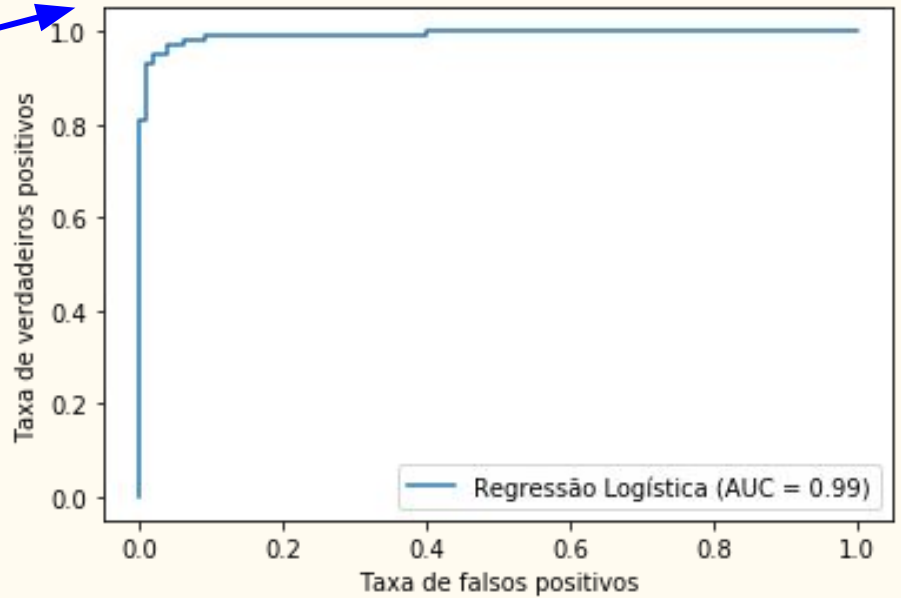


Curva ROC

Idealmente, queremos selecionar um limiar com a maior taxa de verdadeiros positivos e a menor taxa de falsos positivos

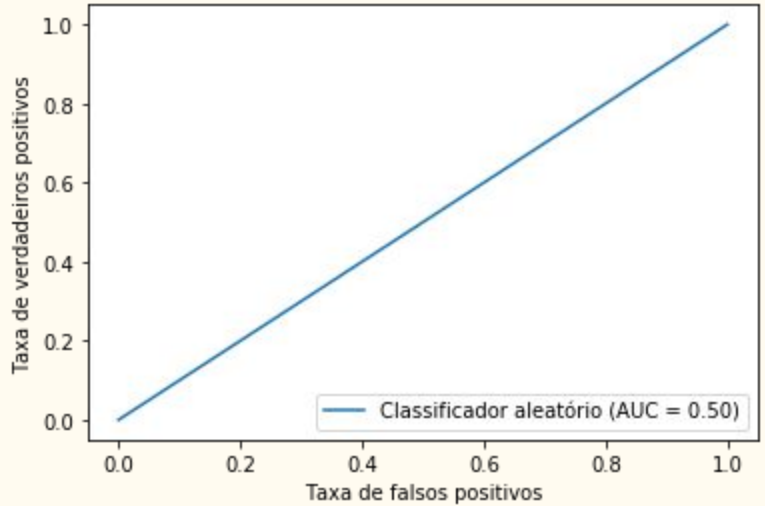
O mais próximo possível daqui

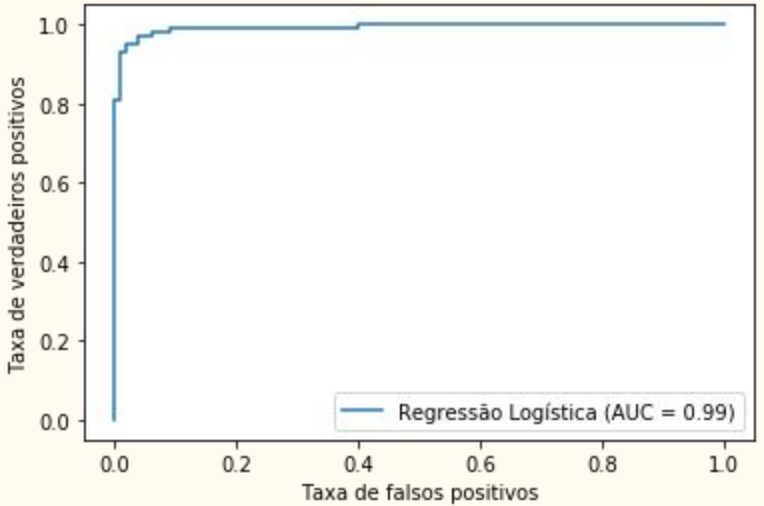
Aqui, o limiar ótimo é 0,47



Área embaixo da curva ROC

A área embaixo da curva ROC (AUC ou AUROC) indica quão bom é o desempenho do classificador independente do limiar
Um classificador aleatório tem AUC = 0,5





Classificação multiclasse

Em um problema multiclasse, temos verdadeiros/falsos de cada classe, ficando com uma matriz de confusão como segue:

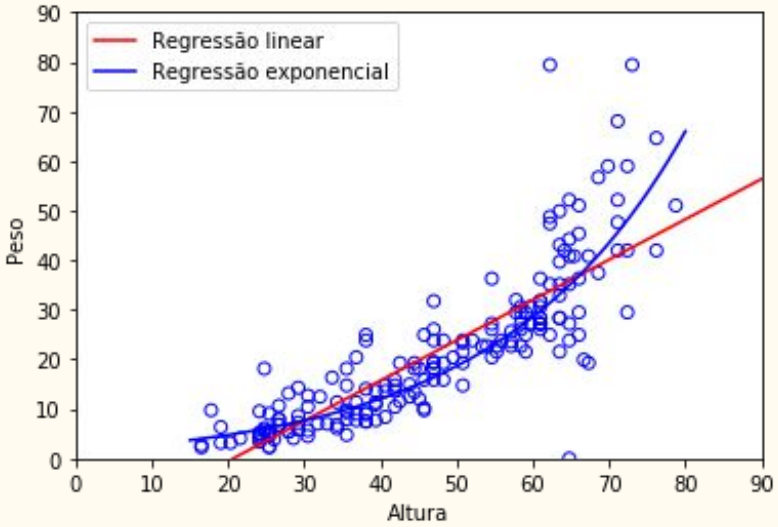
$$Acc = \frac{16+21+14}{16+21+14+2+1}$$
$$Acc = \frac{51}{54}$$
$$Acc = 0{,}94$$

Regressão

Avaliação de modelos de regressão

Na avaliação de regressão, estamos interessados em predições numéricas

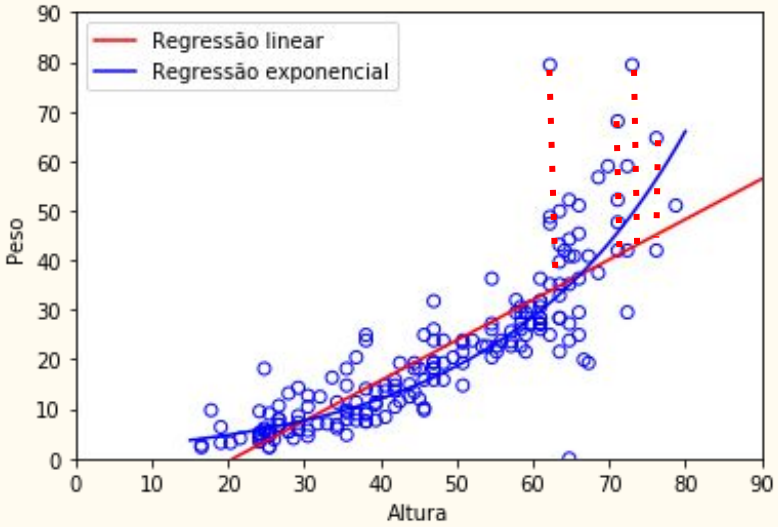
Assim, o que nos interessa é o quão **distantes** estamos de um bom ajuste



Avaliação de modelos de regressão

Na avaliação de regressão, estamos interessados em predições numéricas

Assim, o que nos interessa é o quão **distantes** estamos de um bom ajuste



Avaliação de modelos de regressão

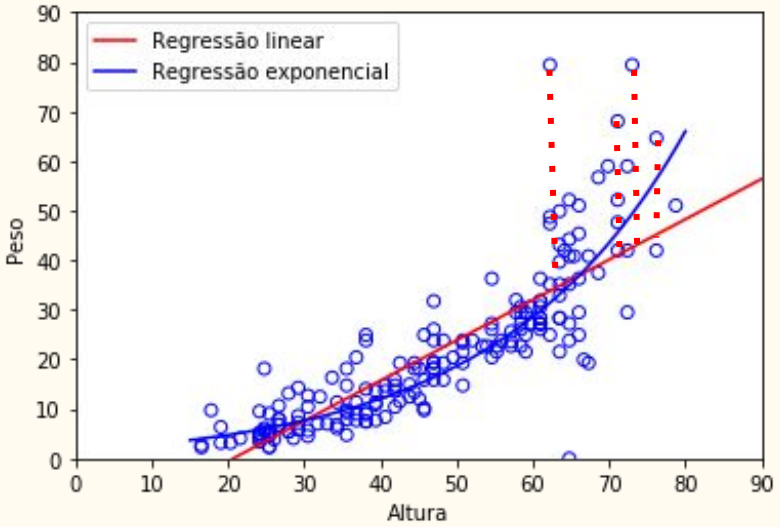
Essas distâncias são comumente calculadas usando duas medidas:

Erro quadrático médio:

$$MSE(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{f}(x_i))^2$$

Erro absoluto médio:

$$MAE(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{f}(x_i)\right|$$



Exemplo

i	y_i	$\hat{f}(x_i)$	$\left(y_i - \hat{f}(x_i)\right)^2$	$\left\lvert y_i - \hat{f}(x_i)\right\rvert$
1	1	1,2	0,04	0,2
2	-1	-0,2	0,64	0,8
3	2	1,9	0,01	0,1
4	0	0,1	0,01	0,1
5	3	3	0	0
6	-2	-1,9	0,01	0,1

$$MSE(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{f}(x_i)\right)^2$$

$$MSE(\hat{f}) = \frac{0,04+0,64+0,01+0,01+0+0,01}{6}$$

$$\boxed{MSE(\hat{f}) = 0,12}$$

$$MAE(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{f}(x_i)\right|$$

$$MAE(\hat{f}) = \frac{0,2+0,8+0,1+0,1+0+0,1}{6}$$

$$\boxed{MAE(\hat{f}) = 0,22}$$

Validação de modelos

Validação de modelos

Aí você treinou seus modelos usando o conjunto de treinamento e agora quer testá-los. Se você não tem um conjunto de teste, como proceder?

E se você tiver um conjunto de teste à disposição, será que um treino e um teste são suficientes para decidir o melhor modelo para seu problema?

Validação de modelos

Primeiro, precisamos lembrar que os conjuntos que usamos para treinar e testar nossos modelos são **amostras** e não a **população inteira**

Além disso, raramente podemos garantir que as amostras foram obtidas seguindo processos estatisticamente corretos

Validação de modelos

Para tomarmos decisões com significância estatística, precisamos usar nossas amostras de forma que possamos extrair o máximo de informação

A seguir, veremos estratégias que nos ajudam simular quão bem um modelo de aprendizagem de máquina é capaz de generalizar seu aprendizado na prática

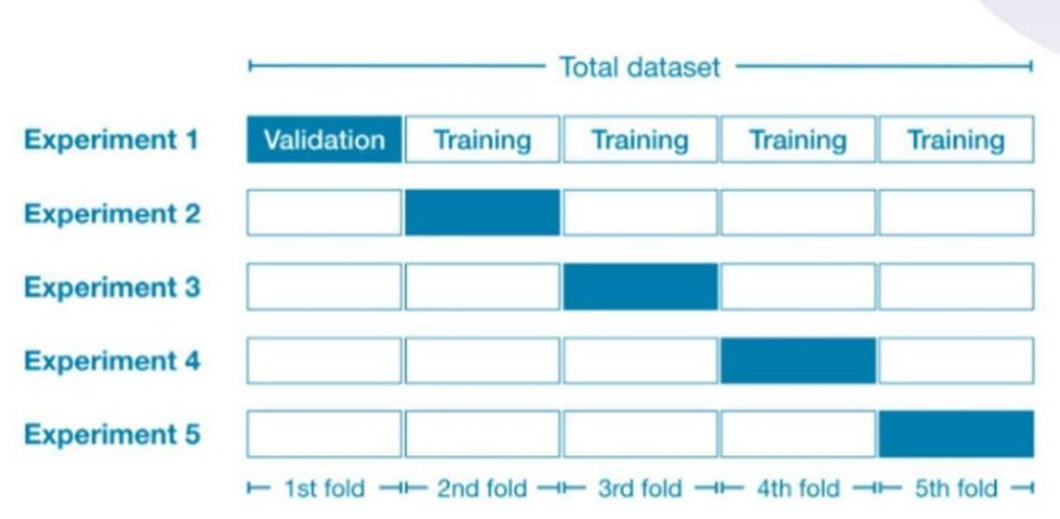
Validação cruzada

Usada para avaliar a capacidade do modelo de reagir corretamente a dados que não foram usados para treiná-lo

Pode ajudar a detectar problemas como over/underfitting e seleção enviesada de amostras

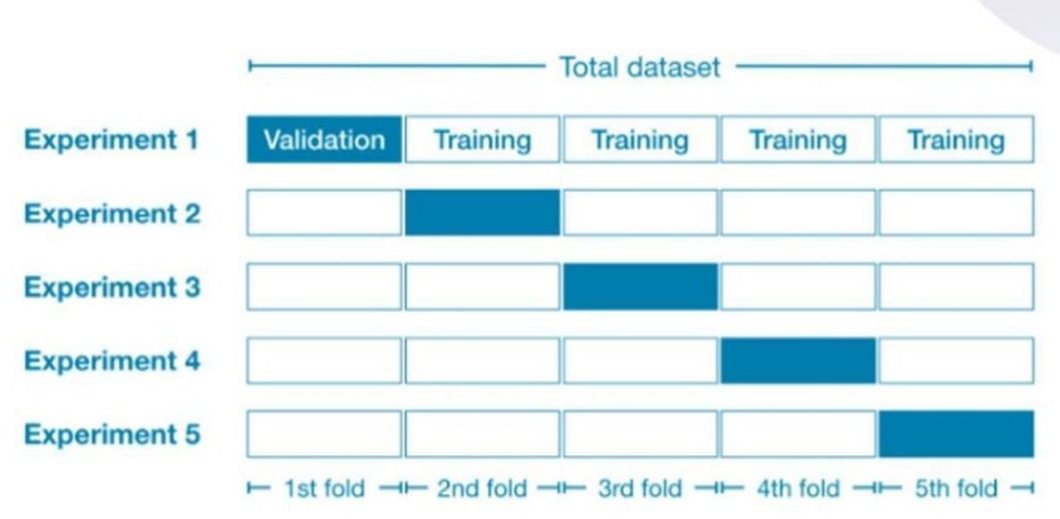
Validação cruzada

Seu funcionamento é simples: divide-se o conjunto em k subconjuntos (*folds*) e, a cada rodada, usa-se um subconjunto para teste (validação) e os outros para treino



Validação cruzada

Esse processo faz com que os conjuntos de treinamento e validação sejam disjuntos



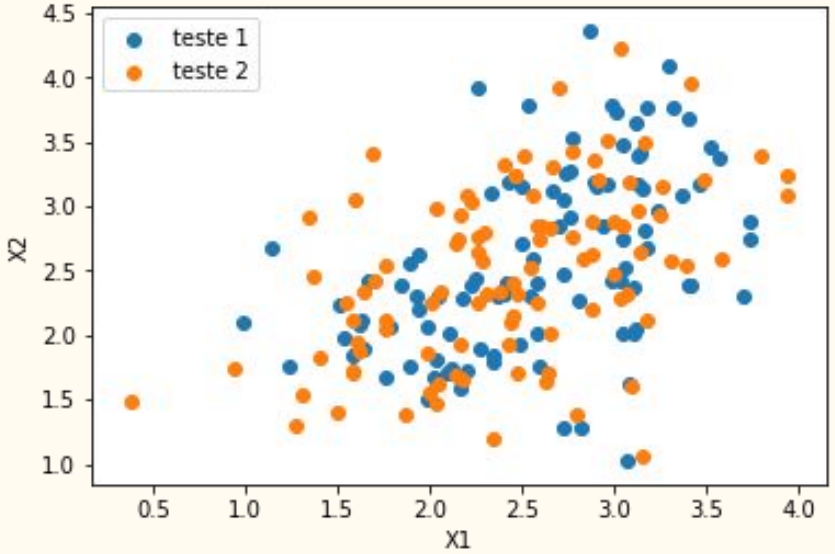
Validação cruzada

```
import numpy as np
from sklearn.model_selection import KFold
from sklearn.tree import DecisionTreeClassifier

accuracies = []
kf = KFold(n_splits=2)
for train_index, test_index in kf.split(X):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        dt = DecisionTreeClassifier()
        dt.fit(X_train, y_train)
        acc = dt.score(X_test, y_test)
        accuracies.append(acc)

print(np.mean(accuracies), np.std(accuracies))
```



Validação cruzada

Em problemas de classificação, é interessante manter as freqüências das classes em cada conjunto de treinamento/teste

Na validação cruzada tradicional, isso não é garantido

Validação cruzada estratificada

Para isso, deve-se usar a validação cruzada estratificada

```
import numpy as np
from sklearn.model_selection import StratifiedKFold
from sklearn.tree import DecisionTreeClassifier

accuracies = []
skf = StratifiedKFold(n_splits=5)
for train_index, test_index in skf.split(X, y):
        X_train, X_test = X[train_index], X[test_index]
        y_train, y_test = y[train_index], y[test_index]

        dt = DecisionTreeClassifier().fit(X_train, y_train)
accuracies.append(dt.score(X_test, y_test))

print(np.mean(accuracies), np.std(accuracies))
```

Leave-one-out

Às vezes o conjunto de dados que temos à disposição é tão pequeno que não conseguimos dividi-lo em subconjuntos de maneira satisfatória

Nesses casos, usamos um tipo de validação cruzada em que cada elemento é um subconjunto

Leave-one-out

A cada rodada do *Leave-one-out* (deixe um de fora), separamos **um indivíduo para teste** e **treinamos com todos os outros**

Isso permite que ainda tenhamos dados suficientes para treinar o modelo

Holdout

Indicado para a situação inversa à do *Leave-one-out:* Temos **dados demais** para executar diversas rodadas de validação cruzada

Neste caso, selecionamos uma amostra aleatória de com p% dos dados (por exemplo 70%) para treinamento

Os dados restantes são usados para teste (por exemplo 30%)

Holdout

Holdout não é ideal, pois só avalia o desempenho do algoritmo usando uma possível combinação dos objetos, que já são obtidos de uma amostra

Uma possível solução é obter diversas subamostras a partir dos dados (*random subsampling*)

FAQ

FAQ

“A competição que tou participando no *kaggle* disponibiliza um conjunto de treinamento e um de teste. Preciso fazer validação cruzada?”

FAQ

“A competição que tou participando no *kaggle* disponibiliza um conjunto de treinamento e um de teste. Preciso fazer validação cruzada?”

- Não, **só se você quiser ganhar**. Nesse caso, recomenda-se realizar validação cruzada usando o conjunto de treinamento, para selecionar o melhor modelo. Com o modelo selecionado, treina-se usando o conjunto de treinamento inteiro e aí sim, usa-se o conjunto de teste.

FAQ

“Meu modelo tem diferentes (hiper)parâmetros e seus valores influenciam bastante o desempenho. Devo escolhê-los usando o resultado da validação cruzada?”

FAQ

“Meu modelo tem diferentes (hiper)parâmetros e seus valores influenciam bastante o desempenho. Devo escolhê-los usando o resultado da validação cruzada?”

- Não. Isso é equivalente a otimizar em tempo de teste. O correto é fazer uma validação cruzada interna em cada conjunto de treinamento ou separar um dos *folds* para teste, um para validação e o resto para treinamento.

O fim justifica os memes.
-Albert Einstein

15/07 - 19:00 - AO VIVO

youtube.com/arialab

BATE-PAPO COM RENAN MOIOLI

IA, NEUROCIÊNCIA E ROBÓTICA

Graduação em Engenharia Elétrica pela Universidade Estadual de Campinas (2006), mestrado pela mesma universidade (2008) com ênfase em inteligência computacional e robótica, e doutorado em Ciências Cognitivas pelo Centre for Computational Neuroscience and Robotics na Universidade de Sussex, Reino Unido. Professor pesquisador do Instituto Internacional de Neurociências Edmond e Lily Safra entre 2013 e 2018. Em 2018, tornou-se Newton Advanced Fellow da Royal Society/ UK. Desde 2019 é professor no Instituto Metrópole Digital da Universidade Federal do Rio Grande do Norte.