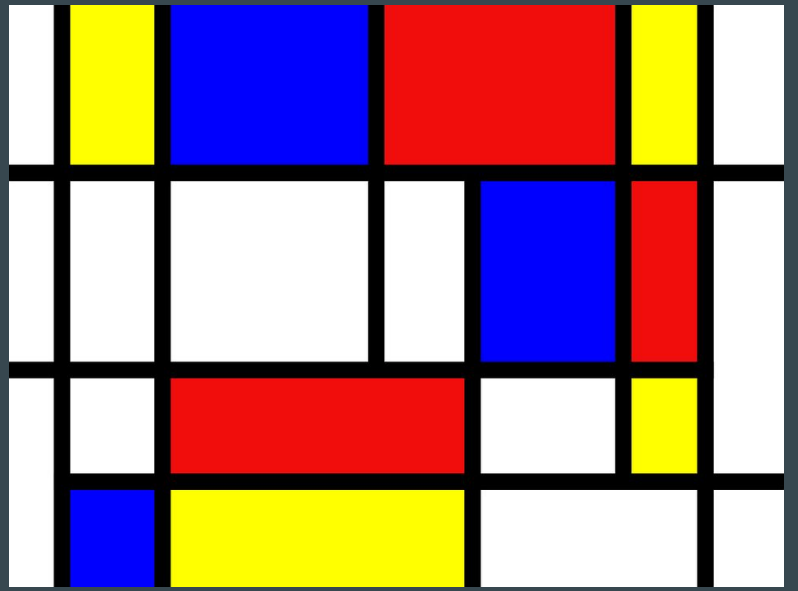
Árvores de classificação

Renato Assunção - DCC, UFMG
2020

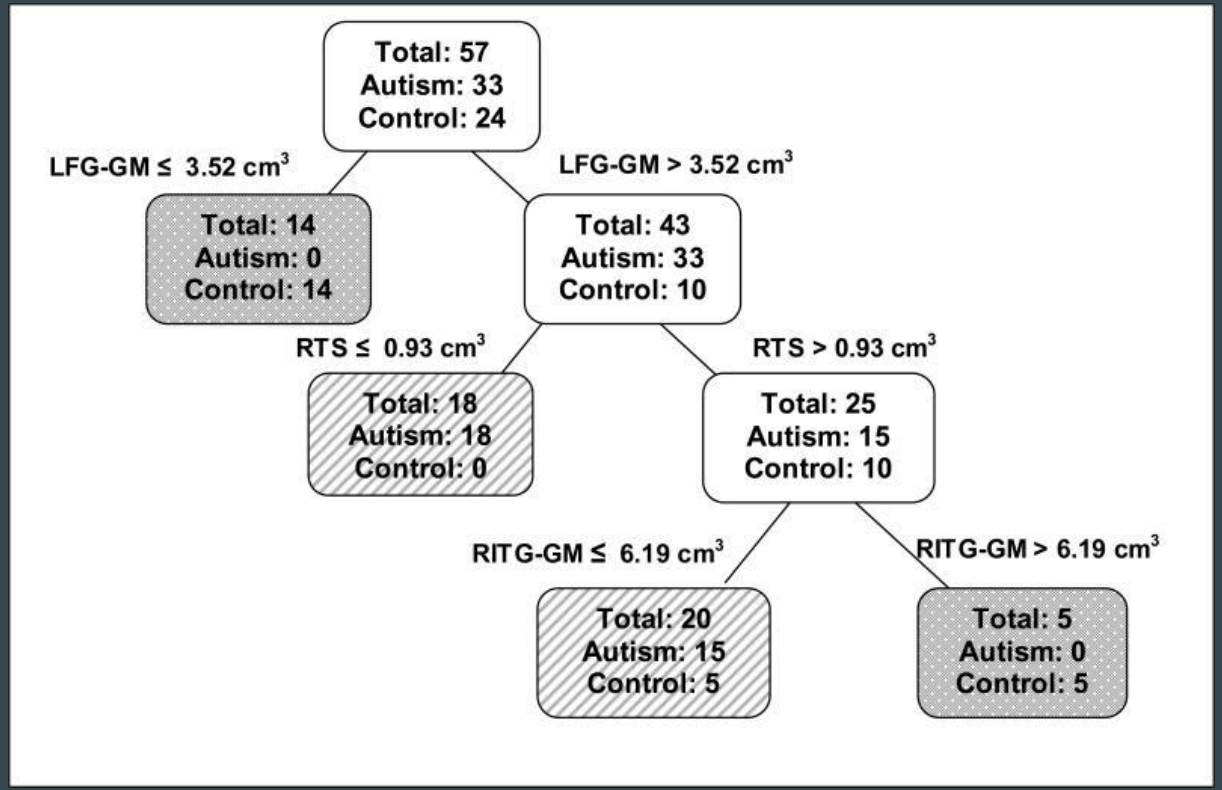
Árvores de classificação e regressão...

Usar qual dessas duas imagens ilustrar as árvores de classificação?

Árvores reais ou obra de Piet Mondrian? Resposta no final



O que vamos obter



Neeley et al. (2007) - Classifying austim

ELSEVIER

Brain & Development 29 (2007) 389–399

BRAIN & DEVELOPMENT
Official Journal of the Japanese Society of Child Neurology

www.elsevier.com/locate/braindev

Original article

Quantitative temporal lobe differences: Autism distinguished from controls using classification and regression tree analysis

E. Shannon Neeley [a], Erin D. Bigler [b,c,e,*], Lori Krasny [c], Sally Ozonoff [d], William McMahon [c], Janet E. Lainhart [c]

[a] *Department of Statistics, Brigham Young University, USA*
[b] *Departments of Psychology and Neuroscience, Brigham Young University, Provo, UT 84602, USA*
[c] *Department of Psychiatry, The University of Utah, USA*
[d] *The MIND Institute, University of California at Davis, USA*
[e] *The Brain Institute, University of Utah, USA*

Received 20 January 2006; received in revised form 31 October 2006; accepted 14 November 2006

Árvore para autismo

Pensa-se que o lobo temporal seja anormal no autismo.

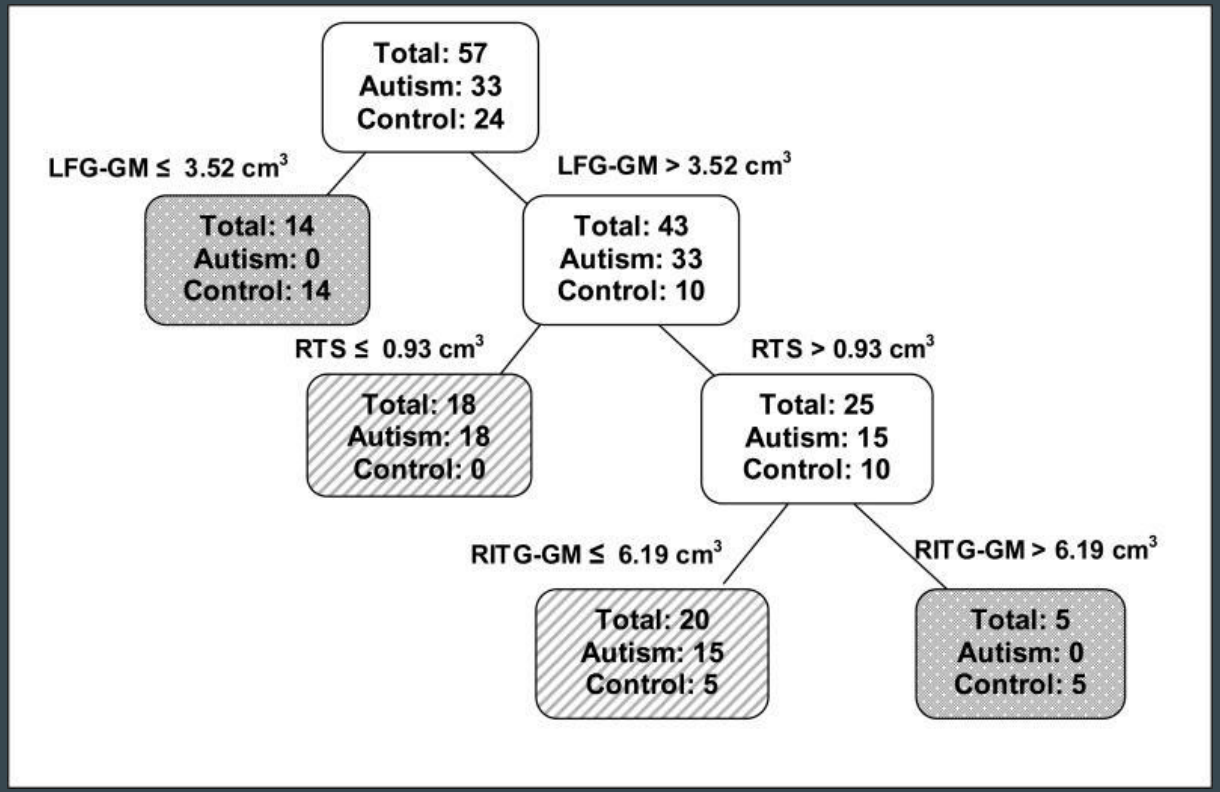
Entretanto, análises volumétricas geralmente não revelam isto.

Neeley et al. (2007) usaram uma árvore para distinguir entre indivíduos com e sem autismo.

Features = variáveis extraídas de imagens de ressonância magnética (MRI).

Árvore mostra que a distinção baseia-se numa relação entre o "volume of the left fusiform gyrus (LFG) gray and white matter, the right temporal stem (RTS) and the right inferior temporal gyrus gray matter (RITG-GM). "

Resultado



LFG-GM: volume of the left fusiform gyrus gray matter

RTS: the right temporal stem

RITG-GM: the right inferior temporal gyrus gray matter.

Outros exemplo:

Keynejad et al. (2019)

Assessment of machine-learning techniques in predicting lithofluid facies logs in hydrocarbon wells

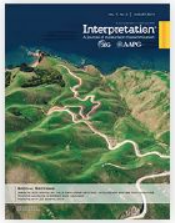
Authors:

Saba Keynejad, Marc L. Sbar, and Roy A. Johnson

https://doi.org/10.1190/INT-2018-0115.1

Abstract

Wireline log interpretation is a well-exercised procedure in the oil and gas industry with all its added value from exploration to production stages. It becomes even more important when it is one of only a few available alternatives to compensate for the lack of core samples in a study of lithologic and fluid variations in a well. Yet, as with other purely expert-oriented interpretational techniques, there is always a considerable risk of subjective or technical errors. We have adopted a hybrid approach that links a machine-learning (ML) algorithm to the log interpretation procedure to solve these problems. We have applied this approach to two different hydrocarbon (HC) fields with the aim of predicting the HC-bearing units in the form of lithofluid facies logs at different well locations. The values of these logs are labels of classes that are separated based on their lithologic and fluid content characteristics. After training different MLs on the designed lithofluid facies logs, we chose a bagged-tree algorithm to predict these logs for the target wells due to its superior performance. This algorithm predicted HC units in an accurate interval (above the HC-fluid contact depth), and it showed a very low false discovery rate. The high-accuracy rate, speed of analysis, and its generalization ability, even in data-deficient cases, accentuate why including ML algorithms can improve the understanding of the subsurface at every phase of the exploration and production process. The proposed approach of using ML algorithms, trained and tuned based on the expert's knowledge of the reservoir, can be modified and applied to future wells in a HC field to significantly minimize the risk of false HC discoveries.



Volume 7
Issue 3
Aug 2019

Pages: 1A-T725

ISSN (print): 2324-8858
ISSN (online): 2324-8866

PUBLICATION DATA

Publisher:Society of Exploration Geophysicists, American Association of Petroleum Geologists

History

Received: 04 Jul 2018
Accepted: 08 Nov 2018
Published: 28 May 2019
Published in print: 01 Aug 2019

CITATION INFORMATION

Saba Keynejad, Marc L. Sbar, and Roy A. Johnson, (2019), "Assessment of machine-learning techniques in predicting lithofluid facies logs in hydrocarbon wells," *Interpretation* 7: SF1-SF13.

https://doi.org/10.1190/INT-2018-0115.1

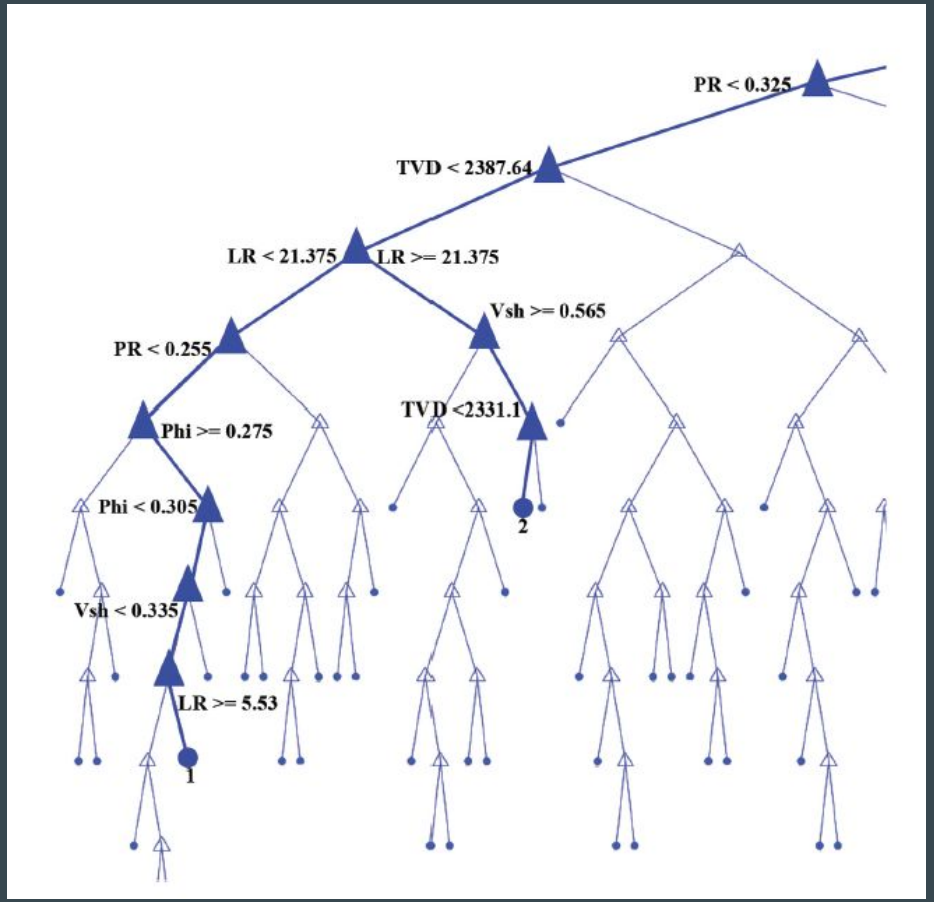
PLAIN-LANGUAGE SUMMARY

Uma árvore do paper

A part of one of the trees. Two observations leading to two example leaves are shown in bold.

Classes are:
(1) gas sand,
(2) shale,
(3) oil sand, and
(4) brine sand.

Features used:
TVD: true vertical depth,
PR: Poisson's ratio,
LR: lambda-rho,
phi: porosity,
VSh: shale volume.

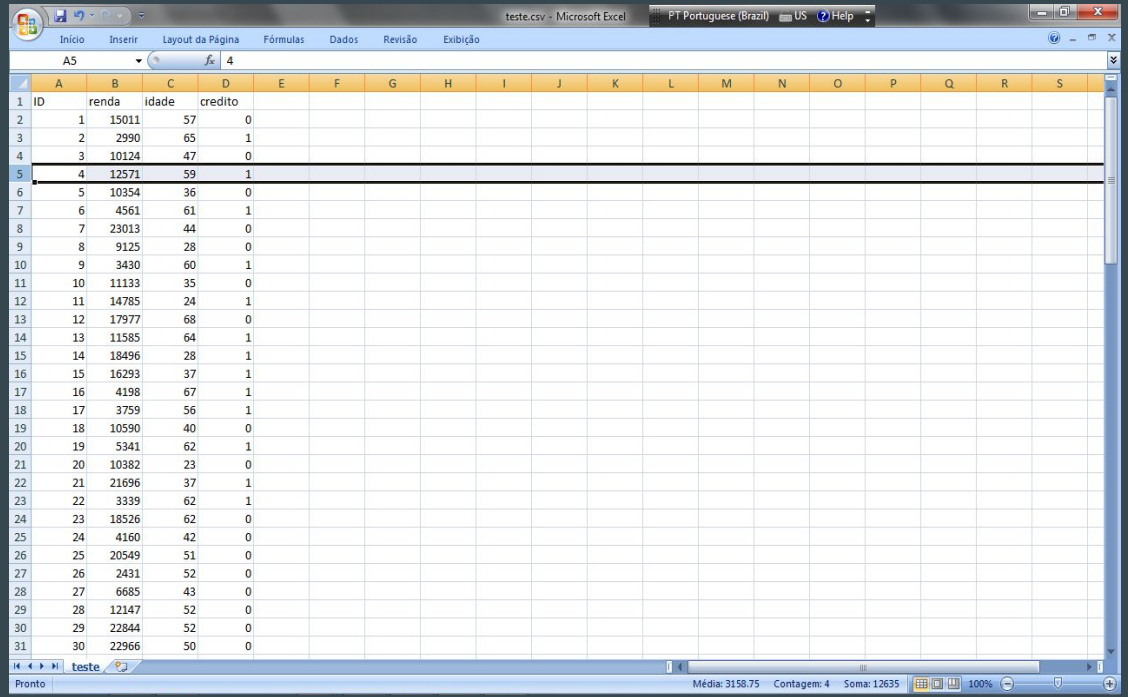


Começando: risco de crédito

Indivíduos que receberam crédito num banco.

Crédito: 0 ou 1
pagou de volta (0) ou não pagou (1) crédito recebido.

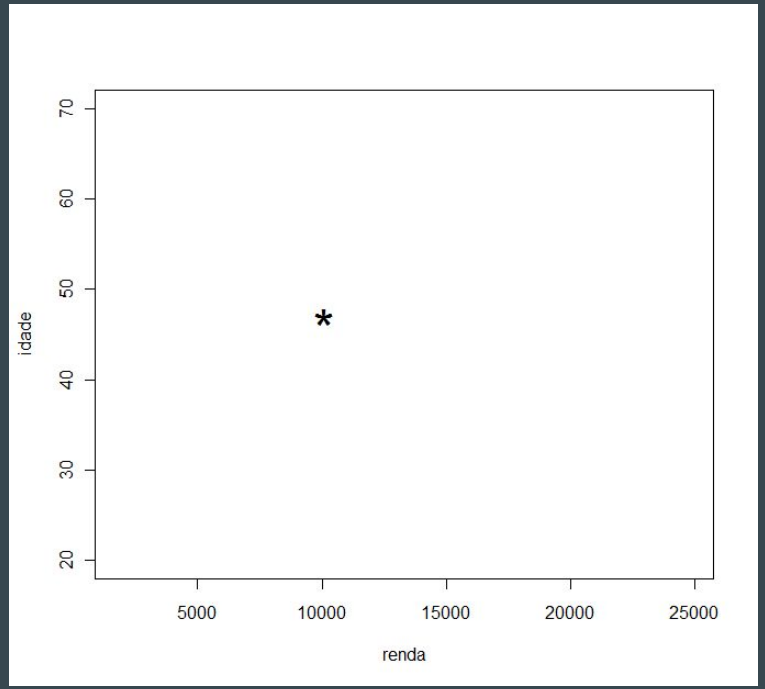
Features: Idade (anos) e renda mensal (em reais)



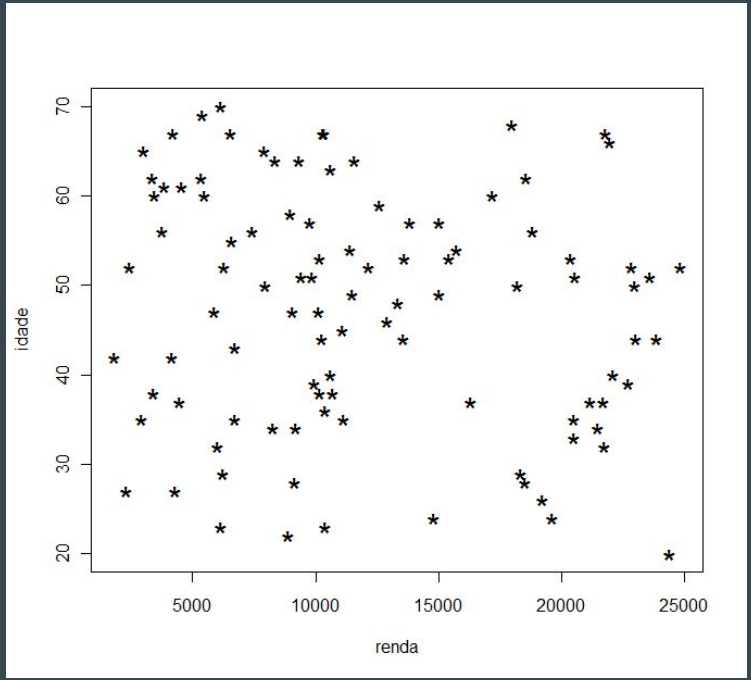
teste.csv - Microsoft Excel

ID	renda	idade	credito
1	15011	57	0
2	2990	65	1
3	10124	47	0
4	12571	59	1
5	10354	36	0
6	4561	61	1
7	23013	44	0
8	9125	28	0
9	3430	60	1
10	11133	35	0
11	14785	24	1
12	17977	68	0
13	11585	64	1
14	18496	28	1
15	16293	37	1
16	4198	67	1
17	3759	56	1
18	10590	40	0
19	5341	62	1
20	10382	23	0
21	21696	37	1
22	3339	62	1
23	18526	62	0
24	4160	42	0
25	20549	51	0
26	2431	52	0
27	6685	43	0
28	12147	52	0
29	22844	52	0
30	22966	50	0

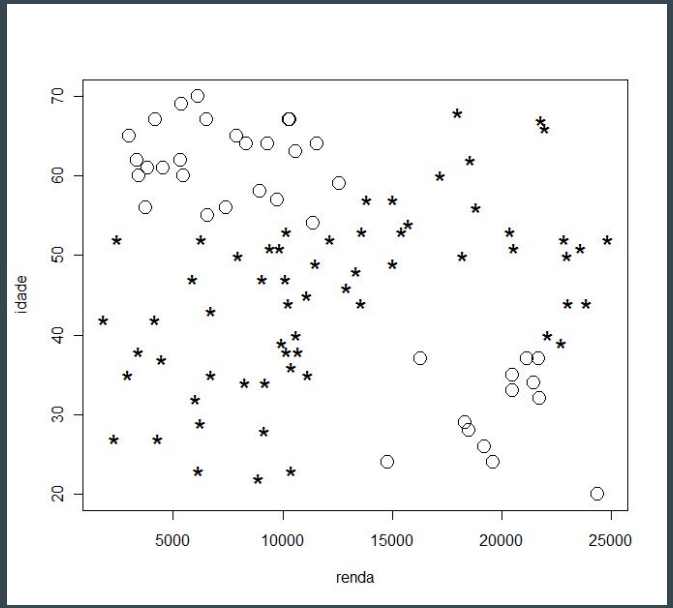
O espaço das variáveis



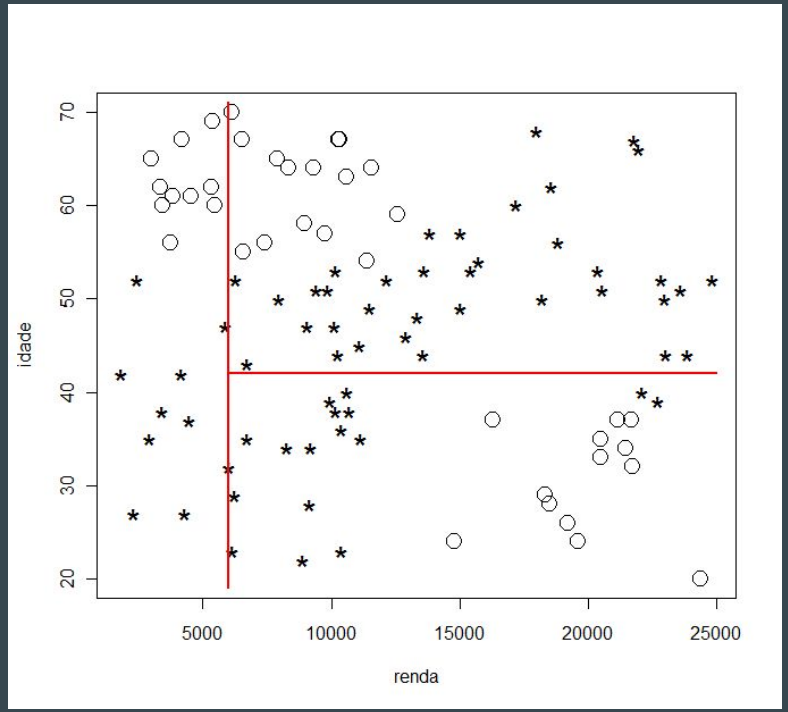
Visualizando a amostra



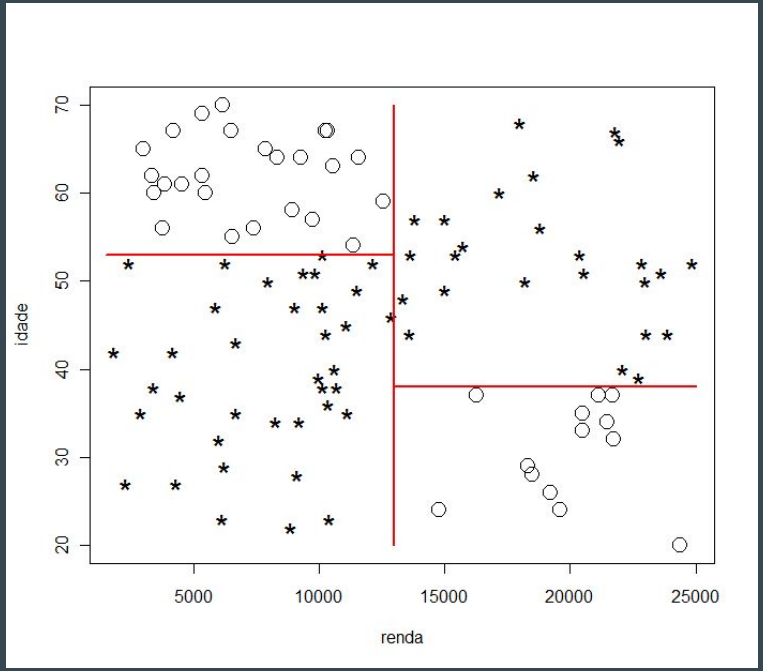
Identificando os bons * e os maus pagadores o



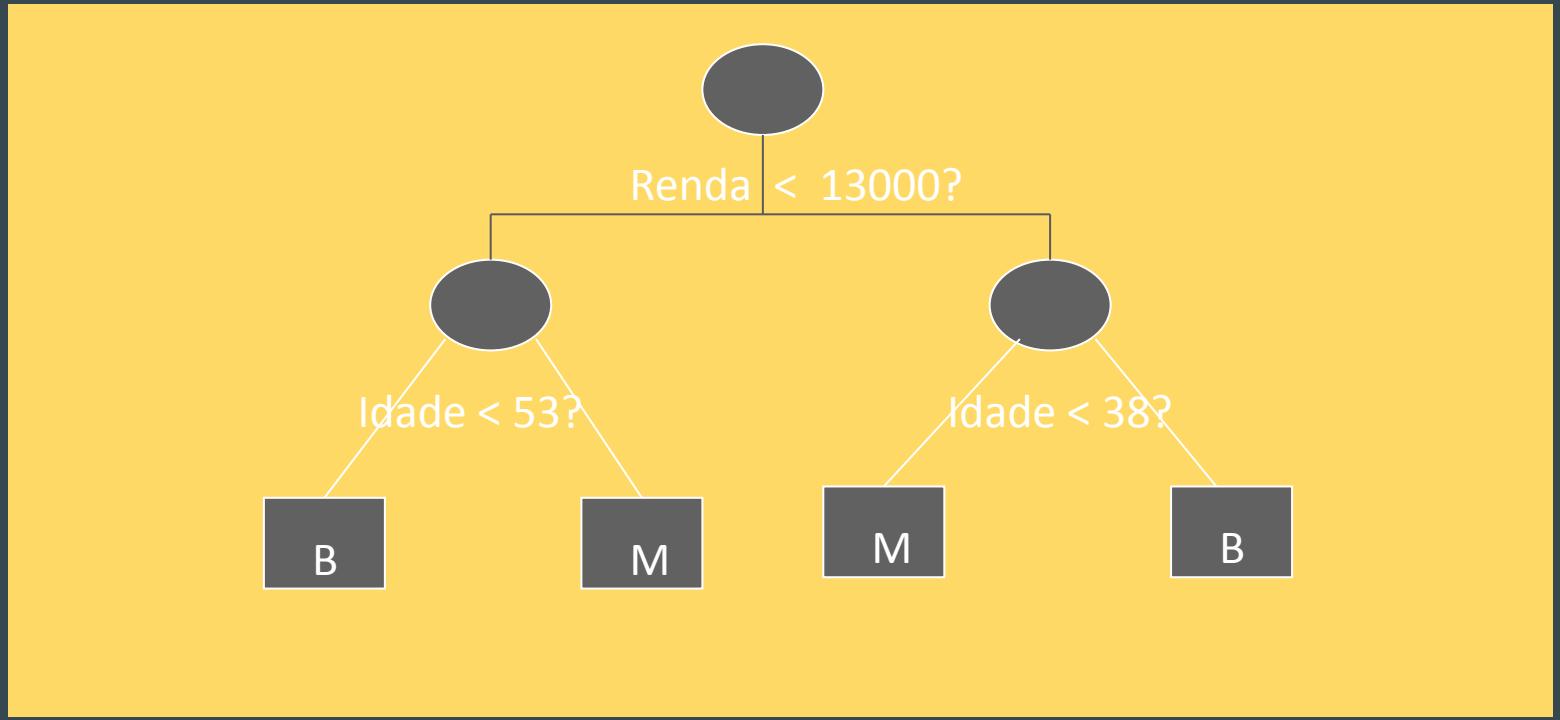
Separando MAL os bons e os maus pagadores



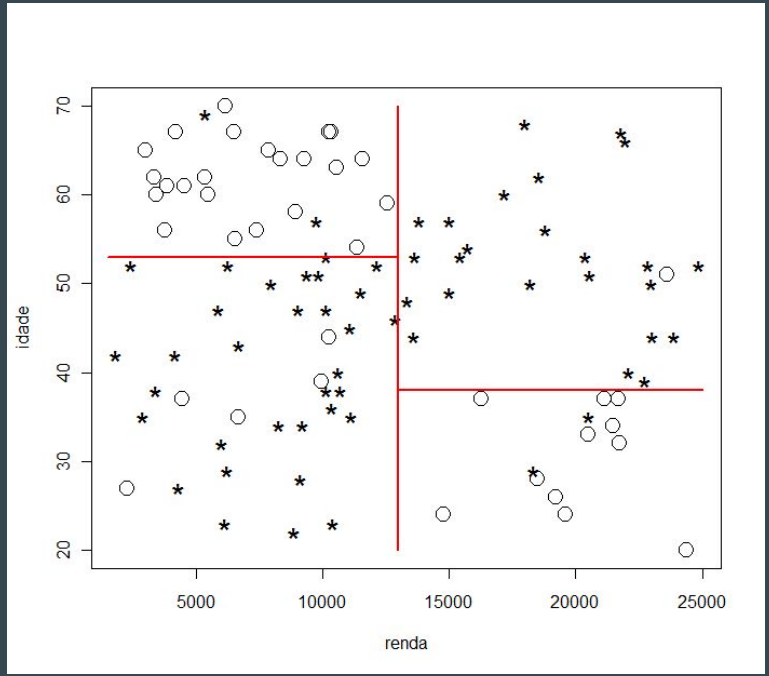
Uma separação muito melhor



Árvore correspondente



Na prática, separação não é perfeita



Impureza no segmento

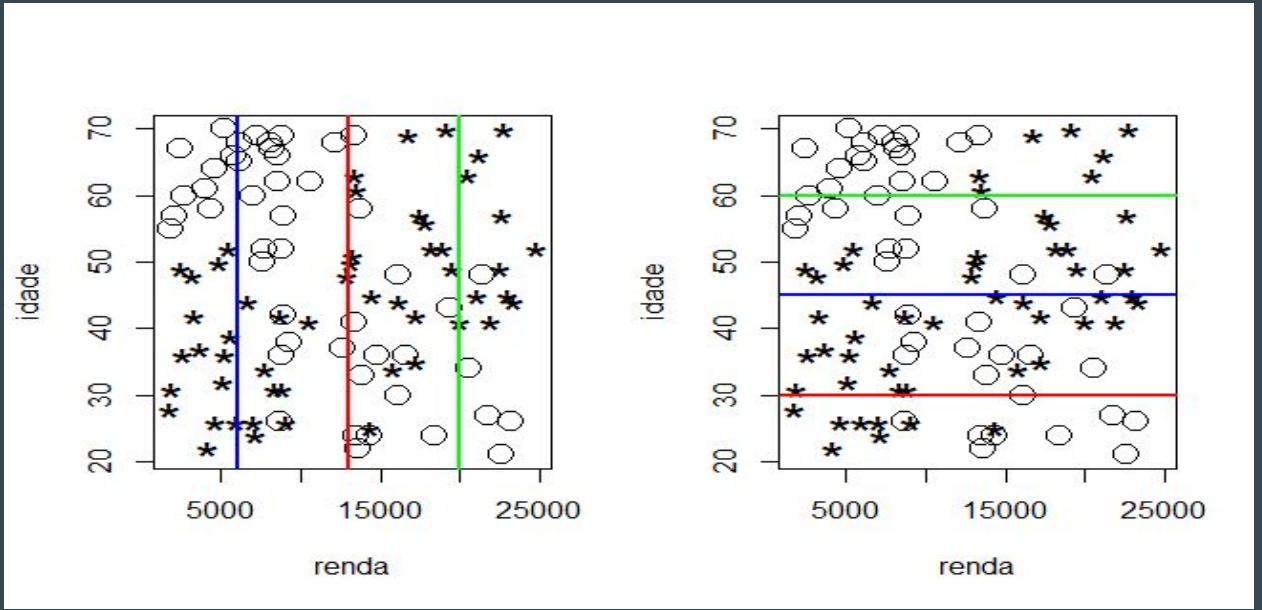
- Idealmente, queremos a separação perfeita:
  - em cada segmento, termos apenas uma classe, bom ou mau.
- Na prática:
  - buscamos a porcentagem de bons próxima de 100% ou de 0% em cada segmento
- Calculamos uma medida de impureza em cada segmento criado:
  - quanto mais distante de 100% e de 0%, mais impuro

Ideal

- Estamos tentando aproximar uma probabilidade condicional complicada
- Seja A = evento pertencer à classe 0
- Queremos P(A | x1= renda, x2=idade)
- Queremos esta probabilidade PARA TODO para de valores (x1, x2)!!!!
- Existem infinitos pares.

- A probabilidade desejada é uma função de x1 e x2:
- P(A | x1= renda, x2=idade) = g(x1, x2)

Como obter a 1ª separação? Espaço de possibilidades



Como obter a 1ª separação?

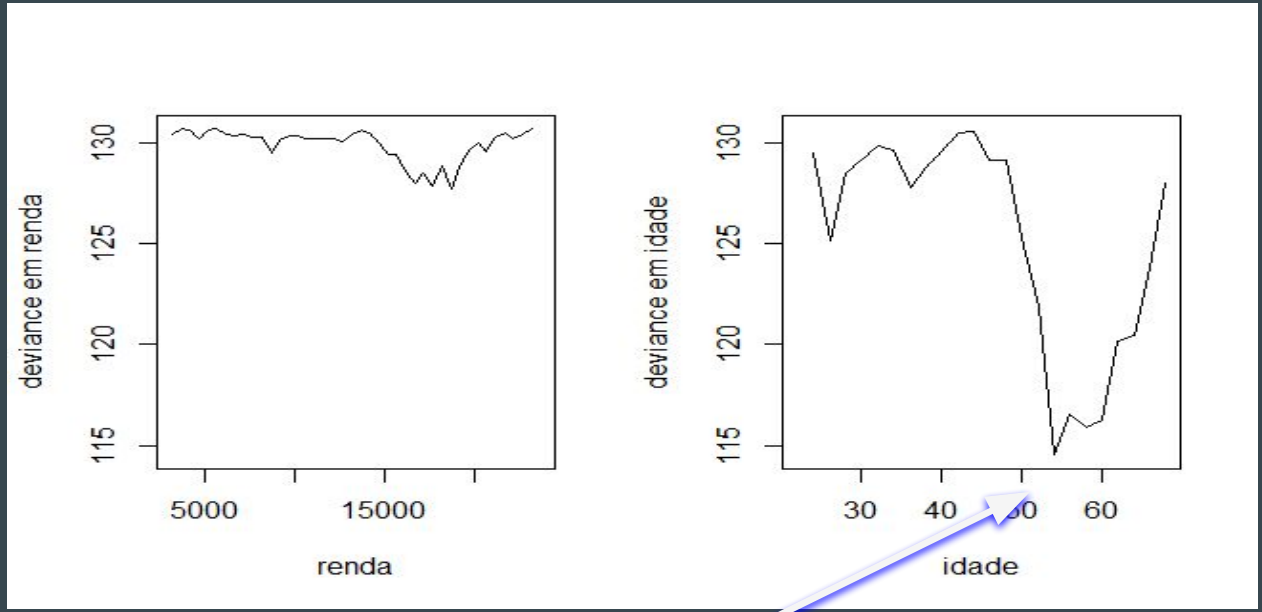
- Percorra o eixo horizontal (renda) parando em cada valor possível
  - Calcule a impureza dos dois segmentos resultantes
  - Escolha aquele valor de renda que produz menor impureza nos segmentos
- Percorra o eixo vertical (idade) parando em cada valor possível.
  - Calcule a impureza dos dois segmentos resultantes

Impureza ao longo de cada eixo

Para cada variável:
  Para cada ponto de corte possível:
    calcule a impureza se fizermos a partição.

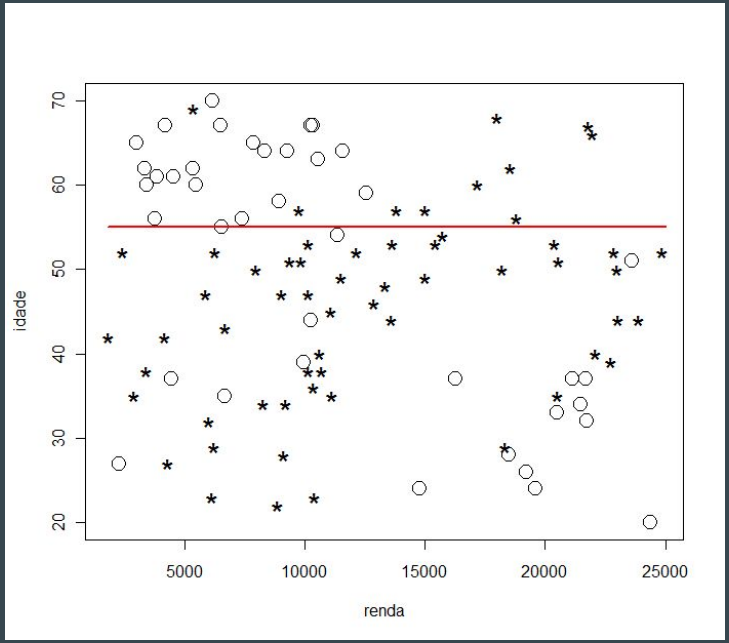
Escolha a variável e o ponto de corte que geram a menor impureza.

Ao lado, a variável idade, no ponto de corte 55 leva à menor impureza.

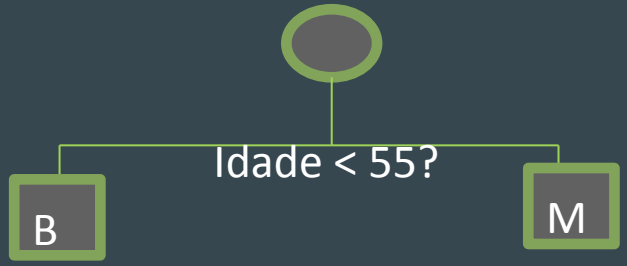


Melhor idade para segmentar

Primeira segmentação



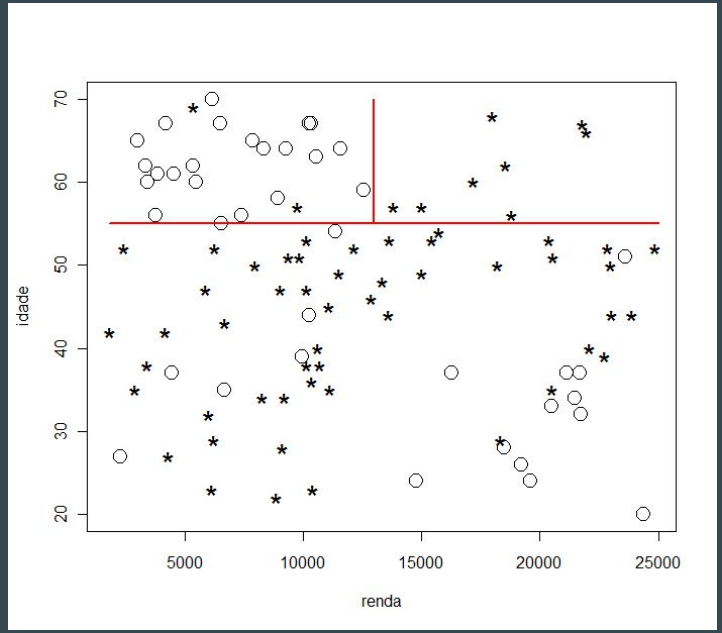
árvore correspondente



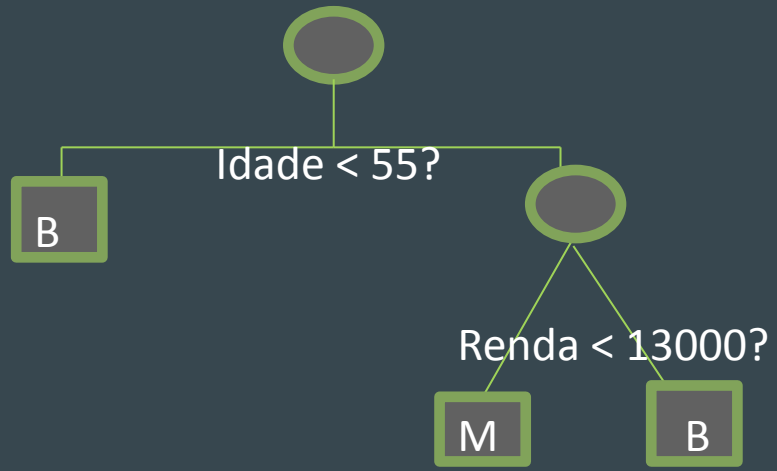
A segunda segmentação

- Existem agora dois segmentos: clientes com idade < 55 anos e aqueles com idade ≥ 55 anos.
- No segmento idade < 55 anos:
  - Escolha a valor de renda que melhor separa os bons e maus pagadores
  - Escolha o valor de idade que melhor separa os bons e maus pagadores
  - Escolha o melhor dentre as duas variáveis
- Faça o mesmo no segmento ≥ 55 anos.
- Escolha segmentar pela segunda vez aquele segmento que separar melhor os grupos.

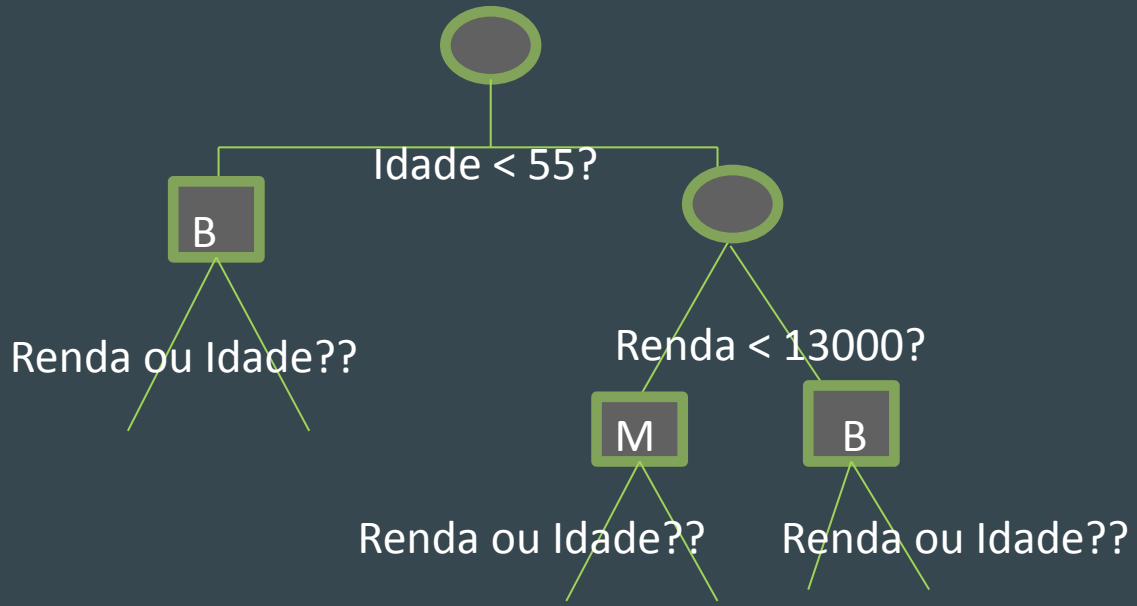
A segunda segmentação



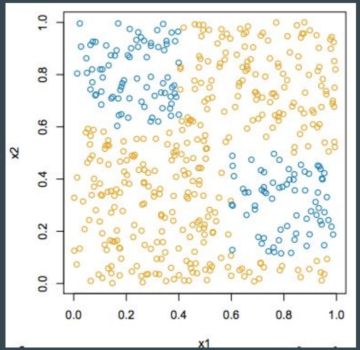
árvore correspondente

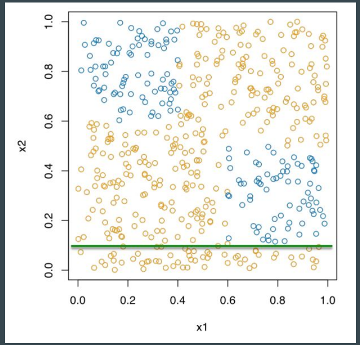


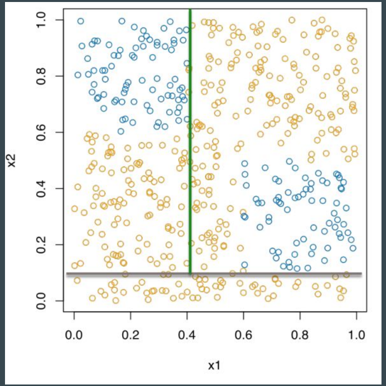
Próximo Passo: Apenas uma segmentação adicional

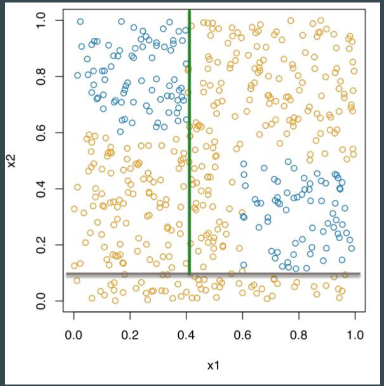


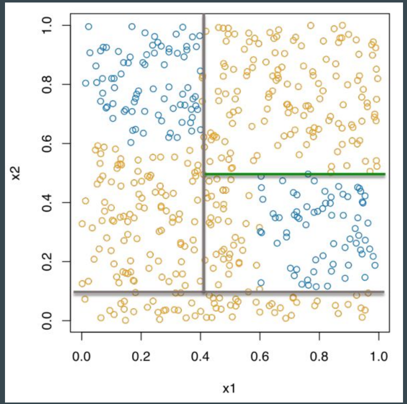
Partição recursiva

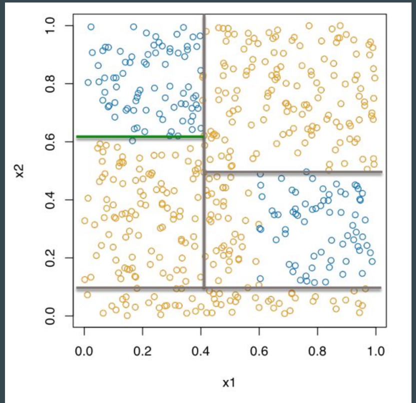


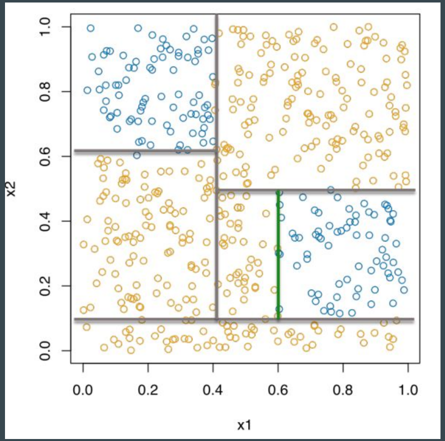












E quando a variável for categórica?

- Algumas vezes a variável não é uma medição numérica como renda ou idade.
- Ela pode ser categórica: cada indivíduo é classificado em duas ou mais categorias.
- Por exemplo: estado civil.
    - Casado, solteiro, divorciado, viúvo
- Como fazer no caso desta variável?

Segmentações possíveis com preditor categórico

- Se tivermos M categorias, existirão $2^{M-1}-1$ possíveis segmentações
- Por exemplo, com 4 estados civis, existem $2^{4-1}-1=7$

casado	Solteiro, divorciado, viúvo
solteiro	casado, divorciado, viúvo
divorciado	Solteiro, casado, viúvo
viúvo	Solteiro, divorciado, casado
Casado, solteiro	divorciado, viúvo
Casado, divorciado	Solteiro, viúvo
Casado, viúvo	Solteiro, divorciado

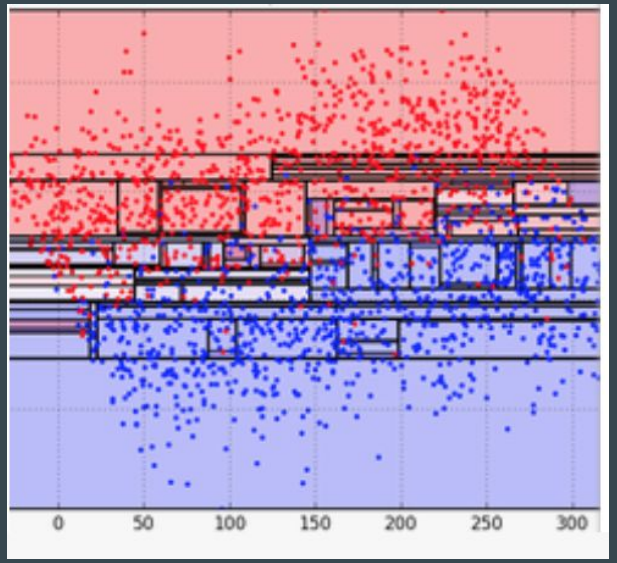
Calculando a impureza

- Para cada possível segmentação categórica:
  - Calcule a redução da impureza ao fazer a segmentação
  - Escolha a segmentação categórica que produz a maior redução de impureza
  - Compare com a redução de impureza que as outras variáveis (numéricas ou categóricas) proporcionam
  - Escolha a variável e a segmentação que produzem a máxima redução de impureza.

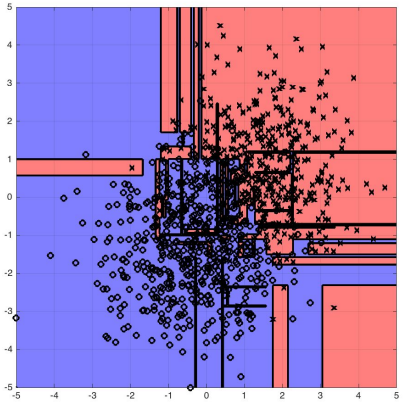
Quando parar a segmentação?

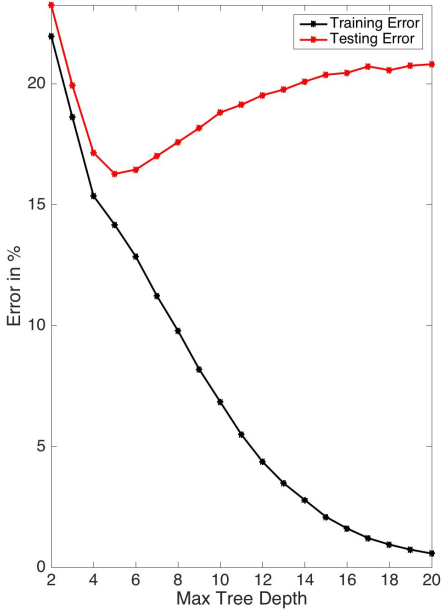
- Prossiga segmentando de forma recursiva.
- Quando parar?
  - Podemos prosseguir até cada folha ser absolutamente pura:
    - → 100% dos dados são de uma única classe.
  - Para obter isto, pode ser que tenhamos que particionar excessivamente.
  - Por exemplo, quando os dados não forem muito facilmente separáveis.
  - Podemos ser obrigados a particionar até que a folha tenha um único ponto (um único caso)
  - Leva a over-fitting -- um dos principais problemas com as árvores.

Exemplo de over-fitting



Training dataset error x test dataset error





Controlando o over-fitting

Existem várias possibilidades.

Mais a frente vamos estudá-las com mais detalhes.

Basicamente, a ideia é não crescer uma árvore profunda demais.

Mais de 2 classes

Mais de duas classes

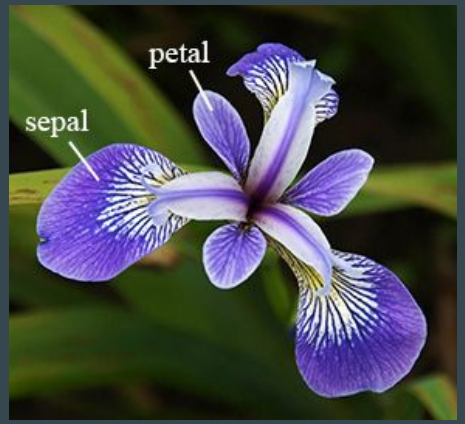
- Variável resposta Y possui mais de duas classes
- O algoritmo é essencialmente o mesmo que para duas classes.
- Vamos ver um exemplo.
- Iris dataset: flores de 3 espécies.

Setosa

Versicolor

Virginica

Iris dataset



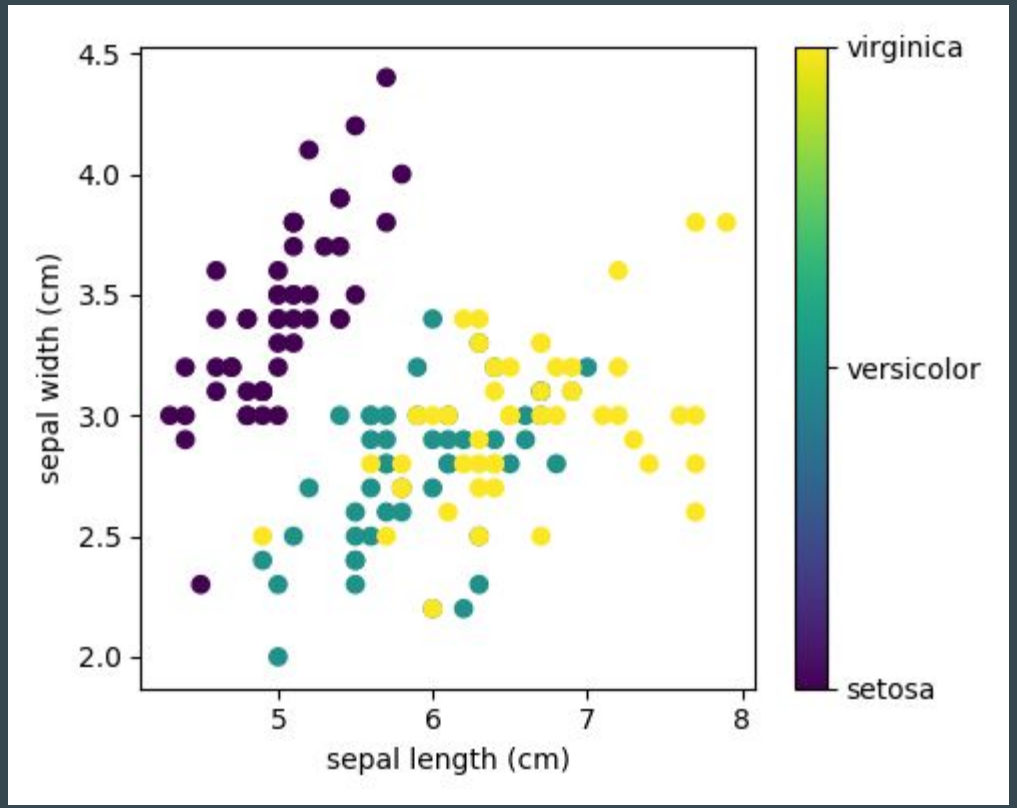
- Três espécies (classes): Setosa, Virginica, Versicolor
- Vários indivíduos de cada classe
- 4 Features em cada indivíduo:
  - comprimento da pétala
  - largura da pétala
  - comprimento da sépala
  - largura da sépala

- OBJETIVO: separar as 3 classes com base nas 4 features

Visualização com 2 features

Setosa é bem separada das demais

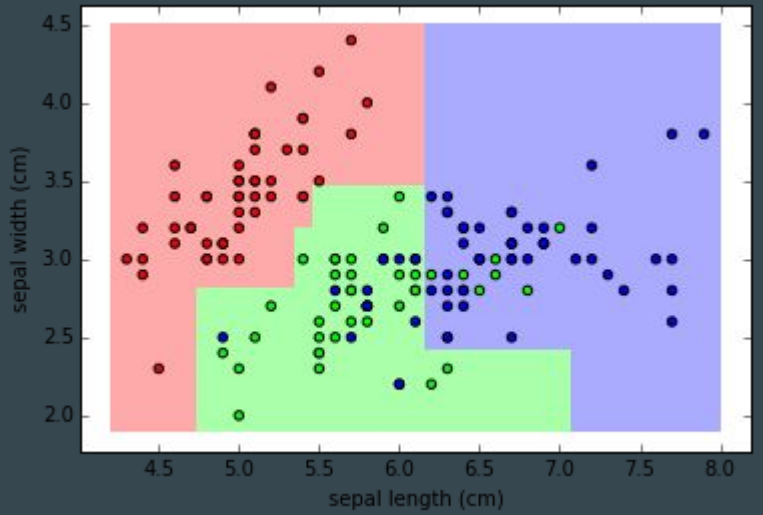
Virginica e Versicolor são um pouco misturadas mas razoavelmente separadas



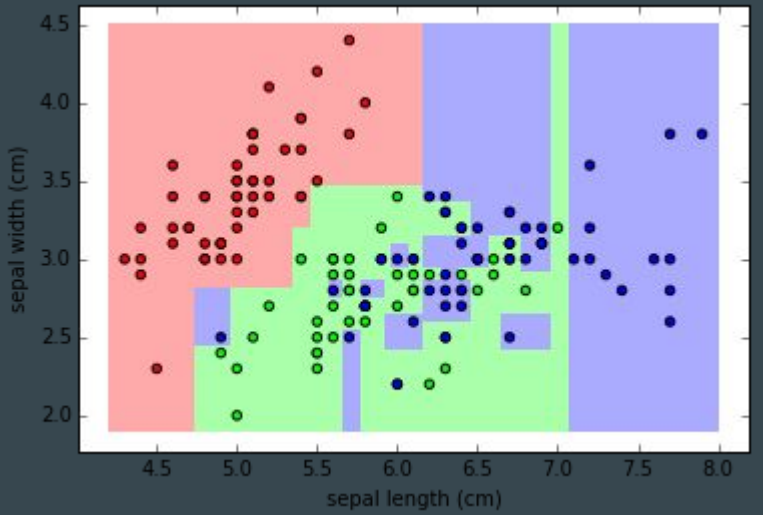
Algoritmo é o mesmo

1. Primeira segmentação:
    a. Para cada feature:
        i. Para cada possível ponto de corte:
            1. Quebre os dados em dois conjuntos:
                a. a direita do ponto de corte
                b. a esquerda do ponto de corte
            2. Calcule a impureza de cada nó resultante e some-as
                a. A impureza será 0 quando o nó tiver apenas uma das 3 classes
                b. Quanto mais misturadas as 3 classes dentro do nó, mais impuro ele é
    b. Escolha a variável e o ponto de corte que minimiza a impureza
2. Repita em cada nó resultante.

Resultado



Profundidade 4



Profundidade 12 (over-fitted)

Refazendo os passos com mais detalhes

- Vamos refazer os passos anteriores fornecendo mais detalhes.
  - Salário alto ou salário baixo?
  - Impureza nos nós: como medir?
  - Como parar de crescer a árvore?
- Vamos usar dados reais mas dataset bem pequeno
- Mais tarde: case study completo

Problema de regressão: Y numérica

Árvore de regressão para prever o log(salário) de um jogador de beisebol (batedor)

Log(salário): classificados em duas categorias:

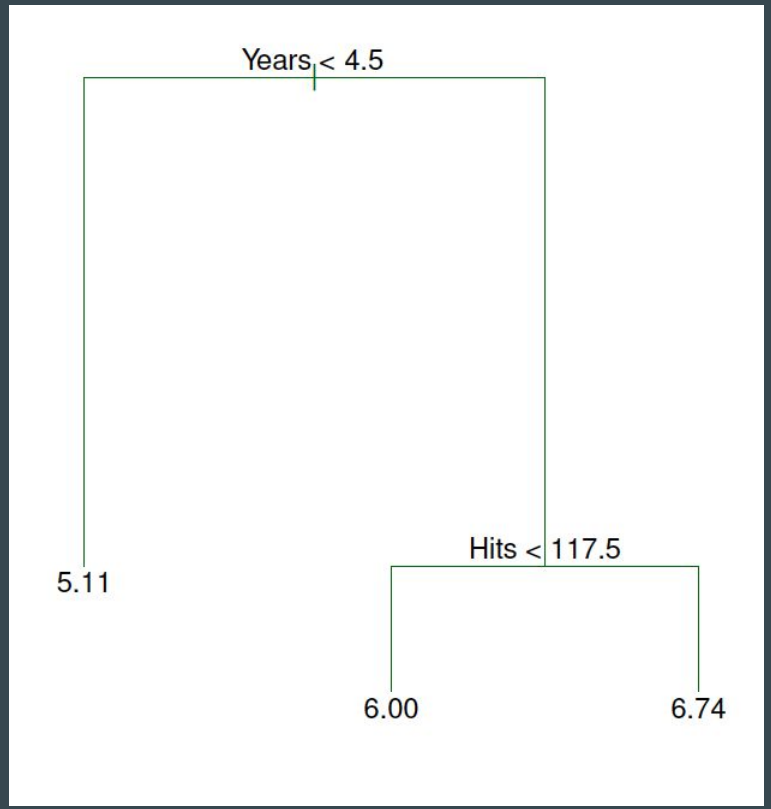
salários altos

salários baixos

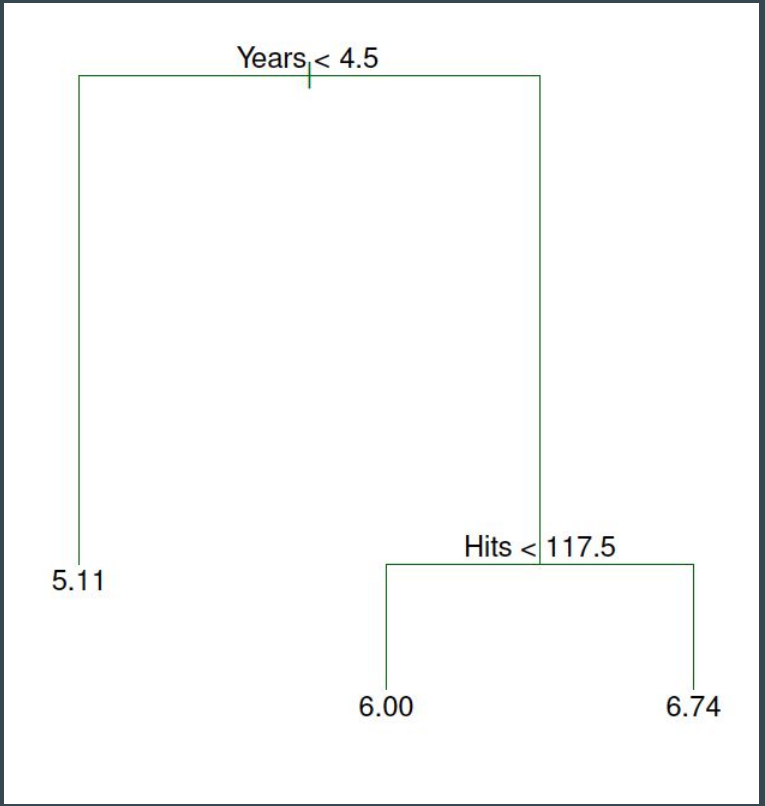
Features: apenas duas

- número de anos que ele jogou nas principais ligas
- no número de acertos (hits) que ele fez na temporada do ano anterior.

Resultado final

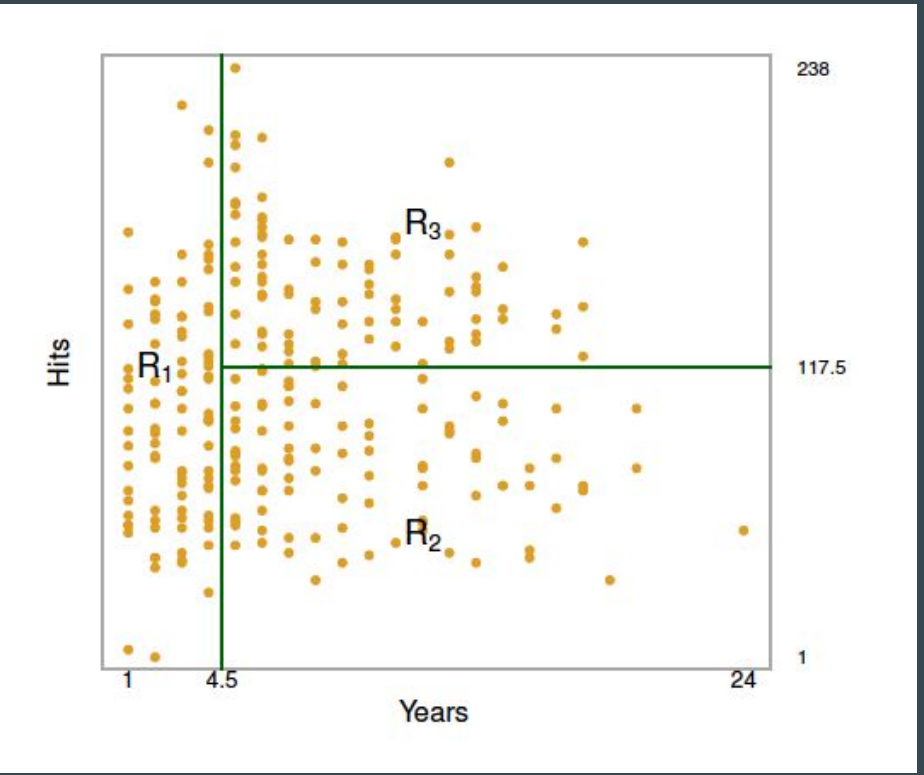


Interpretação



- Anos é o fator mais importante.
- Jogadores com menos experiência ganham salários mais baixos do que jogadores mais experientes.
- Dado que um jogador é menos experiente, o número de hits que ele fez no ano anterior parece ter pouco papel em seu salário.
- Mas entre os mais experientes, o número de hits ocorridos no ano anterior afeta o salário.
- Certamente uma simplificação excessiva, mas fácil de exibir, interpretar e explicar.
- E ATUAR (o que fazer se quiser mudar a situação?)

Outra maneira de visualizar a árvore



Produz uma partição do espaço das variáveis.

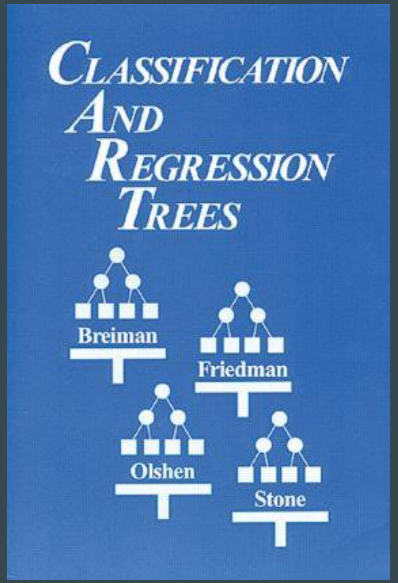
Particiona em três regiões:

$$R_1 = \{X | \text{ Years } < 4.5\}$$

$$R_2 = \{X | \text{ Years } \geq 4.5, \text{ Hits } < 117.5\}$$

$$R_3 = \{X | \text{ Years } \geq 4.5, \text{ Hits } \geq 117.5\}$$

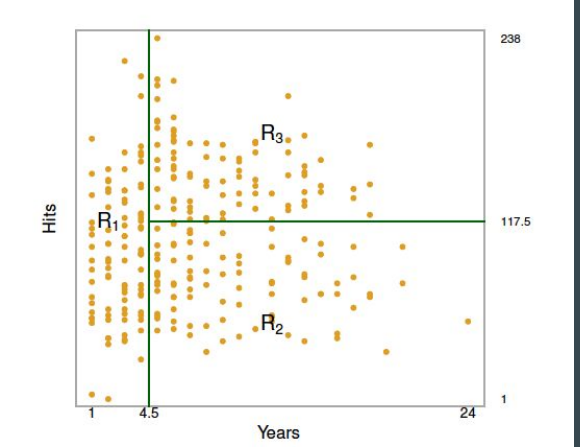
História

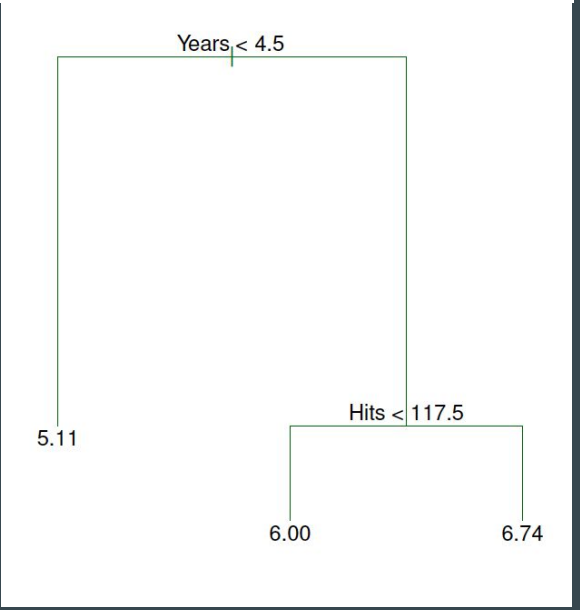


ID3 ou C4.5 - Ross Quinlan (Austrália)

CART - Leo Breiman (UC-Berkeley)

Detalhes





Terminologia

As regiões R1, R2 e R3 : nós terminais (ou folhas)

As árvores são desenhadas de cabeça para baixo, a raiz no alto.

Os pontos ao longo da árvore em que o espaço do preditor é dividido são os nós internos

Os dois nós internos são indicados pelo texto Years < 4.5 e Hits <117.5.

Profundidade da árvore: é o número de nós (splits) desde a raiz até a folha mais distante (mais profunda). Também chamada de altura da árvore.

Detalhes

Dividimos o espaço das features (preditores) $\mathbf{X} = (X_1, X_2, \ldots, X_p)$

em J regiões distintas e não sobrepostas: $R_1, R_2, \ldots, R_J$

Cada observação cai em uma, e só uma, região.

Para cada observação que cai na região Rj, faça a mesma previsão.

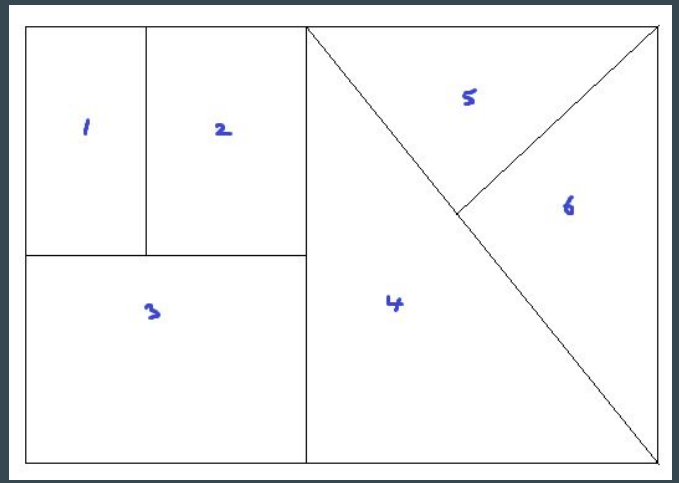
A previsão é:

- problema de classificação: classe mais comum dentro da região no treino

Geometria das regiões

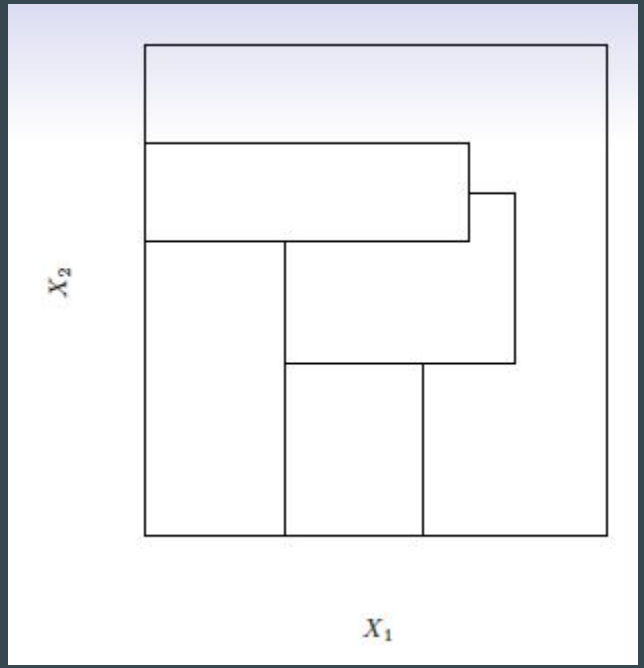
Em teoria, a partição do espaço poderia ser feita usando qualquer forma geométrica.

Na prática, isto é inviável, explosivo computacionalmente



Dividimos em retângulos (ou caixas) p-dimensionais.

Geometria das regiões



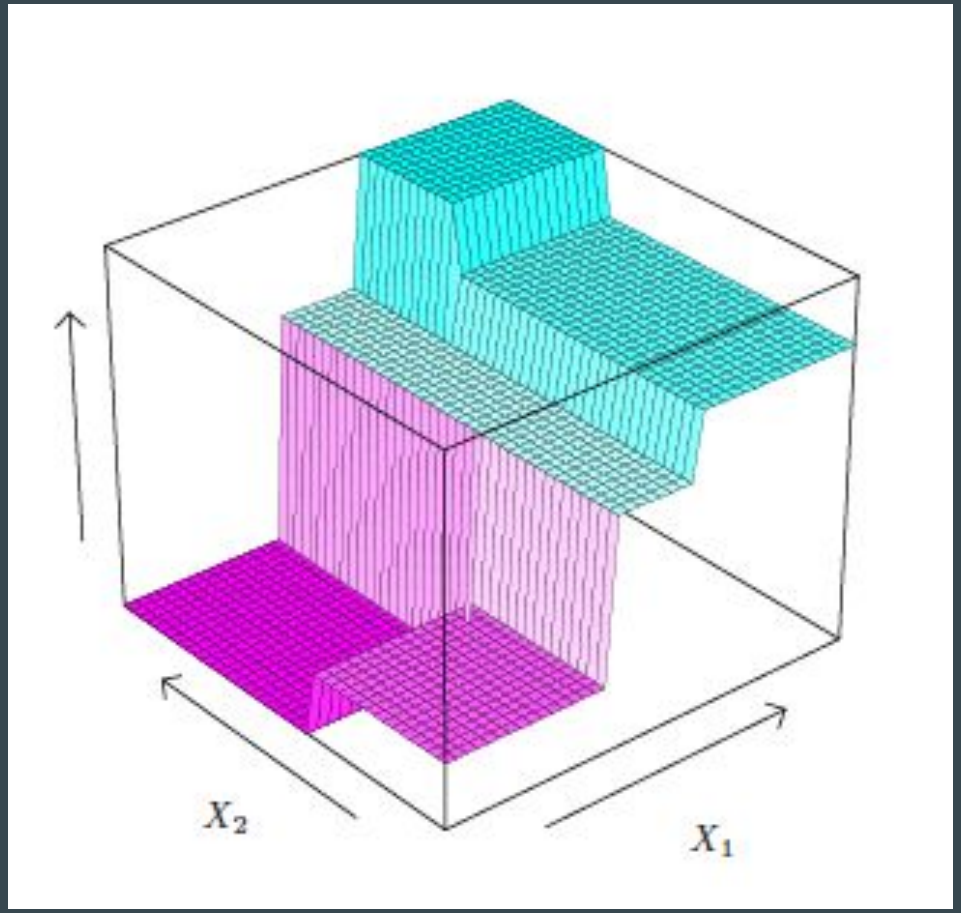
Dividimos em retângulos (ou caixas) p-dimensionais.

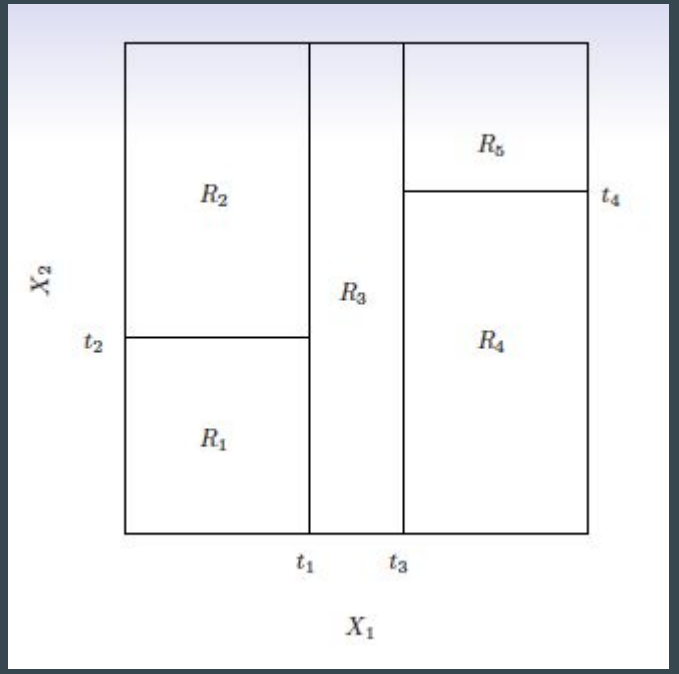
Mesmo sendo apenas caixas, não consideramos todas as possibilidades.

Por exemplo, a configuração ao lado não é pesquisada

Em duas e três dimensões

Apenas divisões assim são consideradas.





Over-fitting

Como evitar over-fitting?

- Várias possibilidades: a melhor delas: usar as florestas aleatórias (curso de machine learning)
- SKLEARN: hiperparâmetros de controle
  - **max_depth : profundidade máxima (default=None → expandir até folhas sejam puras ou tenham menos que min_samples_split.**
  - **min_samples_split: no. mínimo para dividir um nó (default=2)**
  - **min_samples_leaf: no. mínimo na folha (default = 1)**
  - **max_leaf_nodes: no. max de nós (default = None → ilimitado)**
  - **min_impurity_decrease: decréscimo deve ser pelo menos este threshold (default = 0)**
  - **min_impurity_split: só particiona se impureza inicial for pelo menos este threshold (default=0)**

Como evitar over-fitting?

- Regularização: Acrescentar um fator de penalização na função objetivo.
    - Veremos a teoria dessa técnica depois de estudarmos regressão logística
- |T| é o tamanho da árvore (costuma ser o número de folhas). alpha é escolhido por validação cruzada.
- **ccp_alpha: *default=0.0* -** Complexity parameter used for Minimal Cost-Complexity Pruning.
- Em R → competidor de Python em Machine Learning e data analysis
- Testes estatísticos em cada caixa: quebrar ou não?
    - Quebrar apenas se houver evidência estatística de que segmentar produz decréscimo estatisticamente significativo na impureza do nó
    - Teste estatístico é realizado a cada nova possível segmentação.

Partição binária e hierárquica

Esta configuração é obtida escolhendo-se particionar uma caixa já existente em duas sub-caixas.

As duas duas sub-caixas aparecem ao partir ao longo de um único eixo (uma variável) em torno de um ponto de corte.

Como particionar?

Escolha do ponto de corte

Num dado passo, existem J caixas.

Vamos passar a ter J+1 pela partição de UMA das caixas atuais.

Qual caixa e qual o ponto de corte dentro da caixa?

Escolhemos a caixa e o ponto que minimizam uma função objetivo: a impureza.

Impureza

Impureza: Função objetivo para classificação

Quero função que meça a homogeneidade de classes dentro de um nó

Seja $\hat{p}_{mk}$ a proporção da classe k dentro do nó m

1. Classification error rate:

$$E = 1 - \max_k(\hat{p}_{mk})$$

Usando a classe mais frequente para classificar. Não considera a pureza.

1. Gini:

$$G = \sum_{k=1}^{J} \hat{p}_{mk}(1 - \hat{p}_{mk}) = 1 - \sum_{k=1}^{J} \hat{p}_{mk}^2$$

Minimizado quando os $\hat{p}_{mk}$ são próximos de zero ou de 1 (pureza).

Função objetivo para classificação

Seja $\hat{p}_{mk}$ a proporção da classe k dentro do nó m

1. Entropia cruzada:

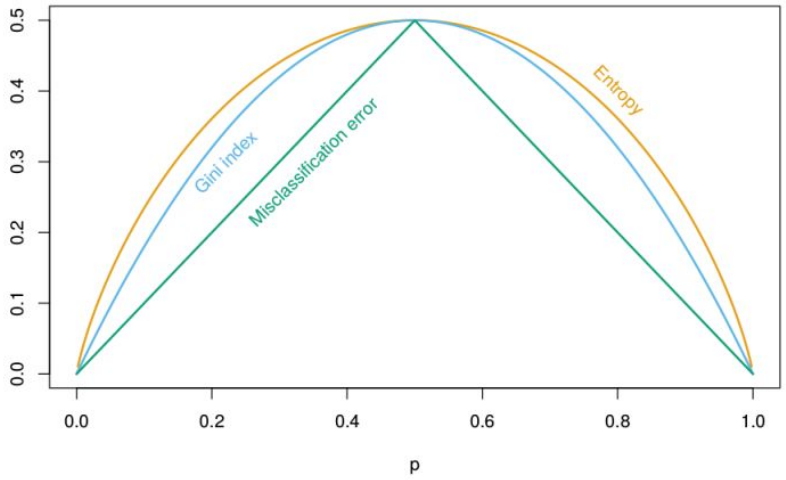
$$G = -\sum_{k=1}^{J} \hat{p}_{mk} \log(\hat{p}_{mk})$$

Minimizada como em Gini (máxima pureza do nó)

Gini e entropia cruzada produzem resultados numericamente similares a maior parte das vezes. Quando não, entropia costuma produzir árvores mais balanceadas (Gini **pode** forçar o isolamento "excessivo" da classe mais frequente

Comparando impurezas

Why not minimize the misclassification error?



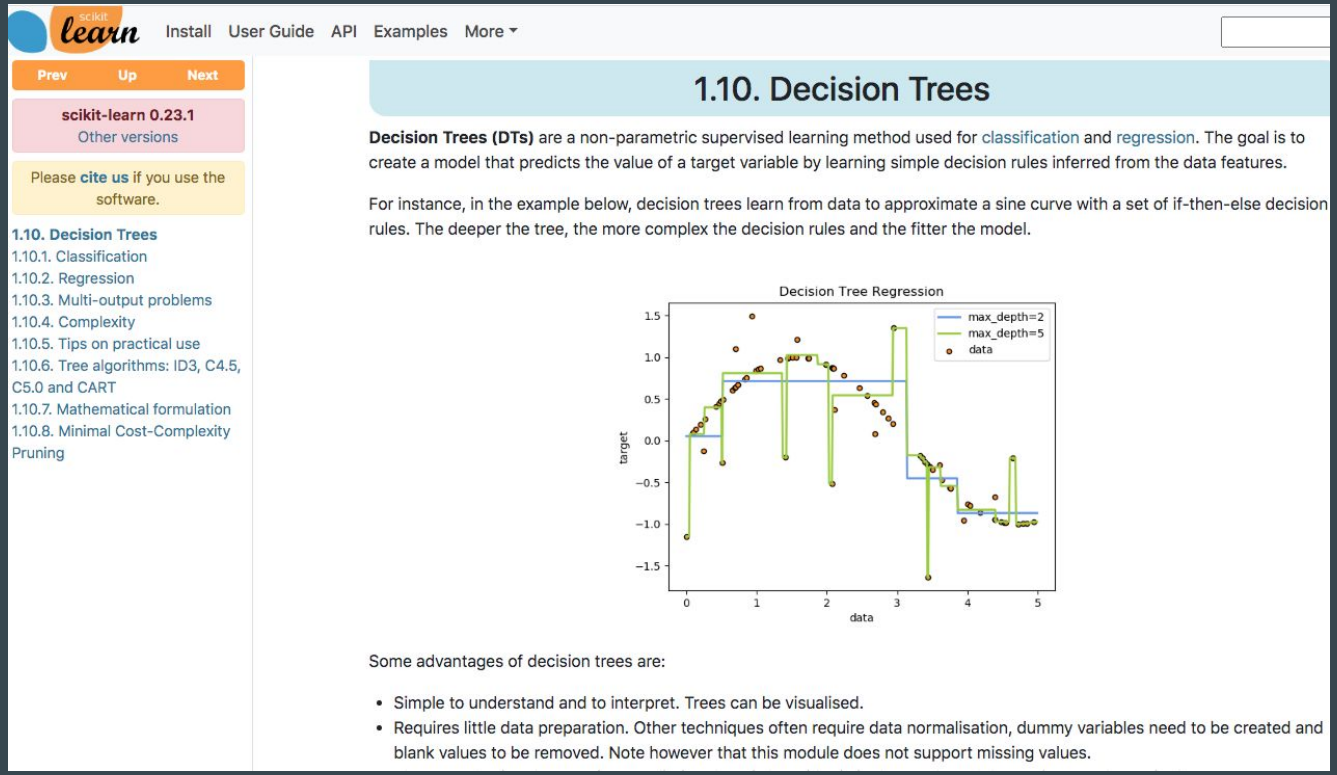
- Misclassification rate is poor at pushing for really pure nodes
- With Gini: going from $\hat{p}_{mk} = 0.8$ to $\hat{p}_{mk} = 0.9$ is better than going from $\hat{p}_{mk} = 0.5$ to $\hat{p}_{mk} = 0.6$
- With Misclassification error, these are considered equal improvements

Classes desbalanceadas

- Árvores criadas usando dados de treino com classes muito desbalanceadas
- Podem submergir as classes mais raras no meio das classes mais comuns
- Solução: dar peso maior para os erros associados com dados da classe rara.
- Em sklearn, use `class_weight`
- Exemplo:
  - Acurácia global = 90% mas detecção da classe 1 (a minoritária) = 45%.
  - class_weight={'0':1, '1': penalizacao_valor}
  - penalizacao_valor deve ser > 1 : por exemplo, 2, 4, etc.
-

SKLEARN

- Ótima documentação: https://scikit-learn.org/stable/modules/tree.html



https://scikit-learn.org/stable/modules/tree.html#mathematical-formulation

1.10.7. Mathematical formulation

Given training vectors $x_i \in R^n$, i=1,..., l and a label vector $y \in R^l$, a decision tree recursively partitions the space such that the samples with the same labels are grouped together.

Let the data at node m be represented by Q. For each candidate split $\theta = (j, t_m)$ consisting of a feature j and threshold t_m, partition the data into $Q_{left}(\theta)$ and $Q_{right}(\theta)$ subsets

$$Q_{left}(\theta) = (x, y)|x_j <= t_m$$
$$Q_{right}(\theta) = Q \setminus Q_{left}(\theta)$$

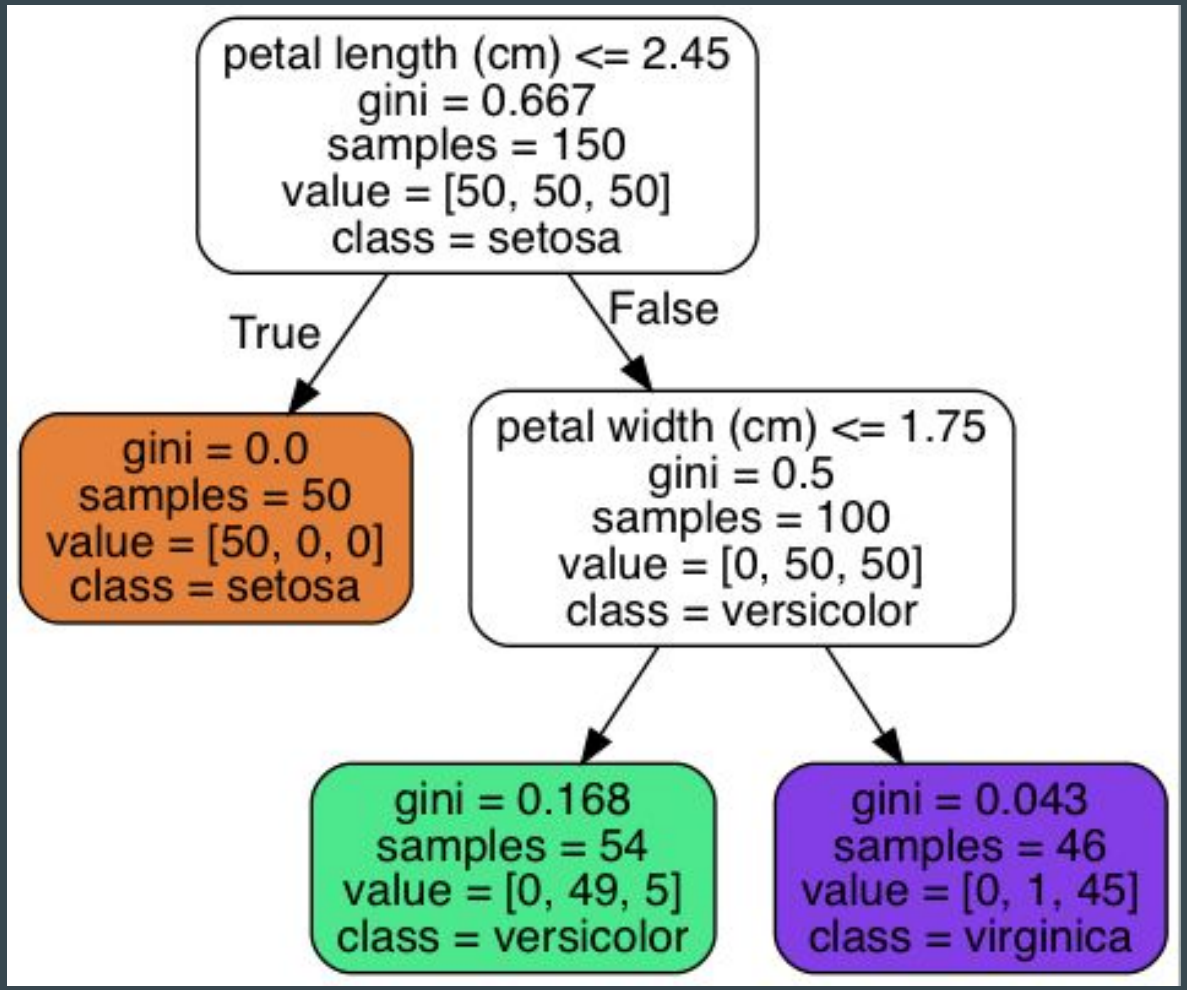
The impurity at m is computed using an impurity function $H()$, the choice of which depends on the task being solved (classification or regression)

$$G(Q, \theta) = \frac{n_{left}}{N_m} H(Q_{left}(\theta)) + \frac{n_{right}}{N_m} H(Q_{right}(\theta))$$

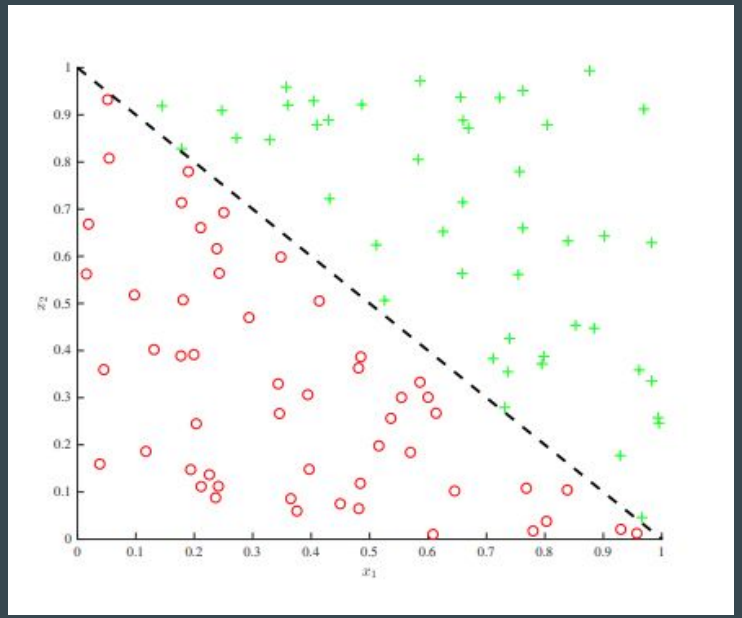
Select the parameters that minimises the impurity

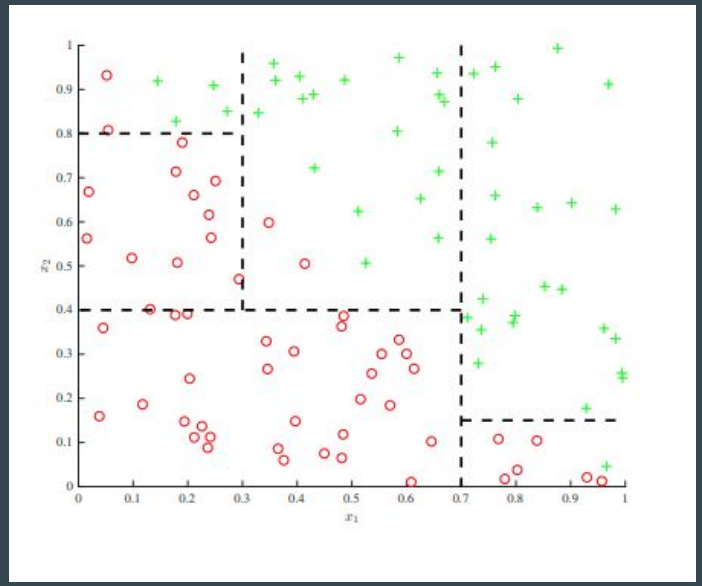
$$\theta^* = \operatorname{argmin}_\theta G(Q, \theta)$$

Recurse for subsets $Q_{left}(\theta^*)$ and $Q_{right}(\theta^*)$ until the maximum allowable depth is reached, $N_m < \min_{samples}$ or $N_m = 1$.


petal length (cm) <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
gini = 0.043
samples = 46
value = [0, 1, 45]
class = virginica

Exemplo de quando árvore é ruim





Regressão logística ou LDA são perfeitos para separar as classes (esquerda).

Árvore precisa de vários nós para representar aproximadamente a reta (direita).

Dicas práticas

- Visualize a árvore. Não use as árvores como caixa-preta.
- Estude a topologia da árvore. Isto ajuda a entender o fenômeno e a alterar o "sistema" para induzir certas respostas.
- Usar profundidade grande ou pequena?
  - Comece com profundidade pequena (de 3 ou 5, por exemplo) como uma profundidade inicial da árvore. Veja se faz sentido. Faça gráficos e scatterplots.
  - Em seguida, aumente a profundidade.
- Não deixe as folhas ficarem com poucos casos → sinal de overfitting.

Resumo: Pros

1. Simples de entender e interpretar.
2. Entendimento visual.
3. Não demanda pré-processamento dos dados.
4. Árvore criada com profundidade k → o custo de predição de um caso novo é k.
5. Preditores podem ser numéricos ou categóricos.
6. Capaz de lidar com problemas de várias saídas.
7. Possível validar um modelo usando testes estatísticos. Isso torna possível explicar a confiabilidade do modelo.
8. Apresenta bom desempenho, mesmo que suas suposições sejam um pouco violadas pelo modelo real a partir do qual os dados foram gerados.

Resumo: Cons

- Sujeitas a over-fitting - solução: usar florestas de árvores (semana que vem) ou podar (prune) a árvore impedindo seu crescimento excessivo
- Podem ser instáveis: pequenas variações nos dados → árvore diferente. Solução: ensemble (florestas ou bagging), semana que vem.
- Certas configurações de dados muito estruturados podem requerer uma árvore muito elaborada (dados de regressão linear, por exemplo).
- Dados com classes muito desbalanceadas: classes raras podem ficar "imersas" em folhas que predizem a classe mais frequente. Solução: dar peso maior às observações da classe mais rara (usar `class_weight` in sklearn)

Árvores de classificação e regressão...

Para ilustrar meu livro, usar qual dessas duas imagens ilustrar as árvores de classificação e regressão? Árvores reais ou obra de Piet Mondrian?