FIAP

Algoritmos de Regressão

Dheny R. Fernandes

Agenda

1. **Algoritmos de Regressão**
    1. Definição do problema
    2. Principais algoritmos

2. **Linear Regression**
    1. Hipótese
    2. Função de Custo
    3. Gradiente Descendente
    4. Multiple Linear Regression

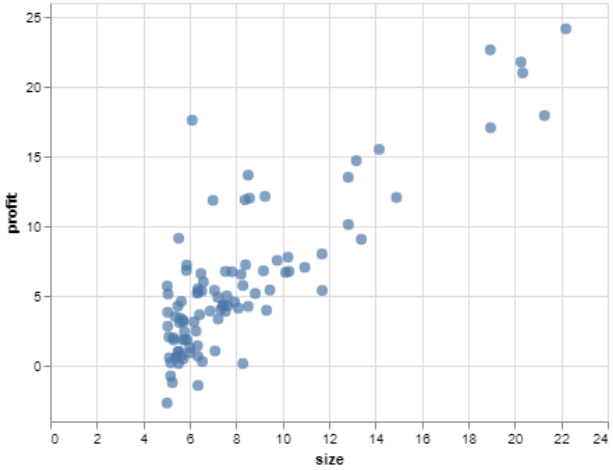
3. **Regression Trees**
    1. Intuição
    2. Exemplo
    3. Como construir uma árvore
        1. RSS
        2. MSE
        3. Overfitting
        4. Múltiplas características
        5. Implementação
    4. Pruning
        1. Implementação

Algoritmos de Regressão

Definição do Problema

Regressão é uma técnica de machine learning em que um modelo prediz como resultado uma variável numérica contínua

Geralmente é usada em finanças, investimentos e outros para encontrar a relação entre uma variável dependente (target) e várias outras variáveis independentes. Exemplos: predizer preço de imóveis, ações, salários, etc.



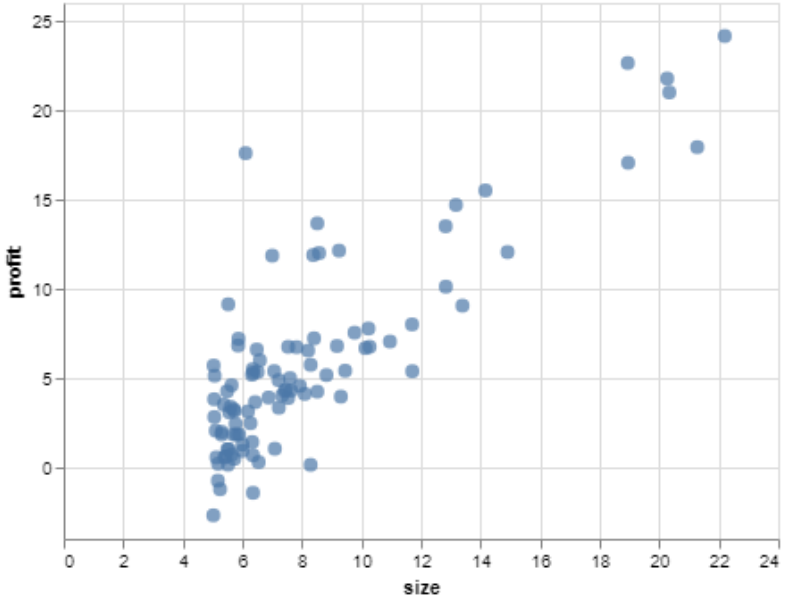
Principais Algoritmos

Existem vários algoritmos de regressão. Aqui estão alguns exemplos:

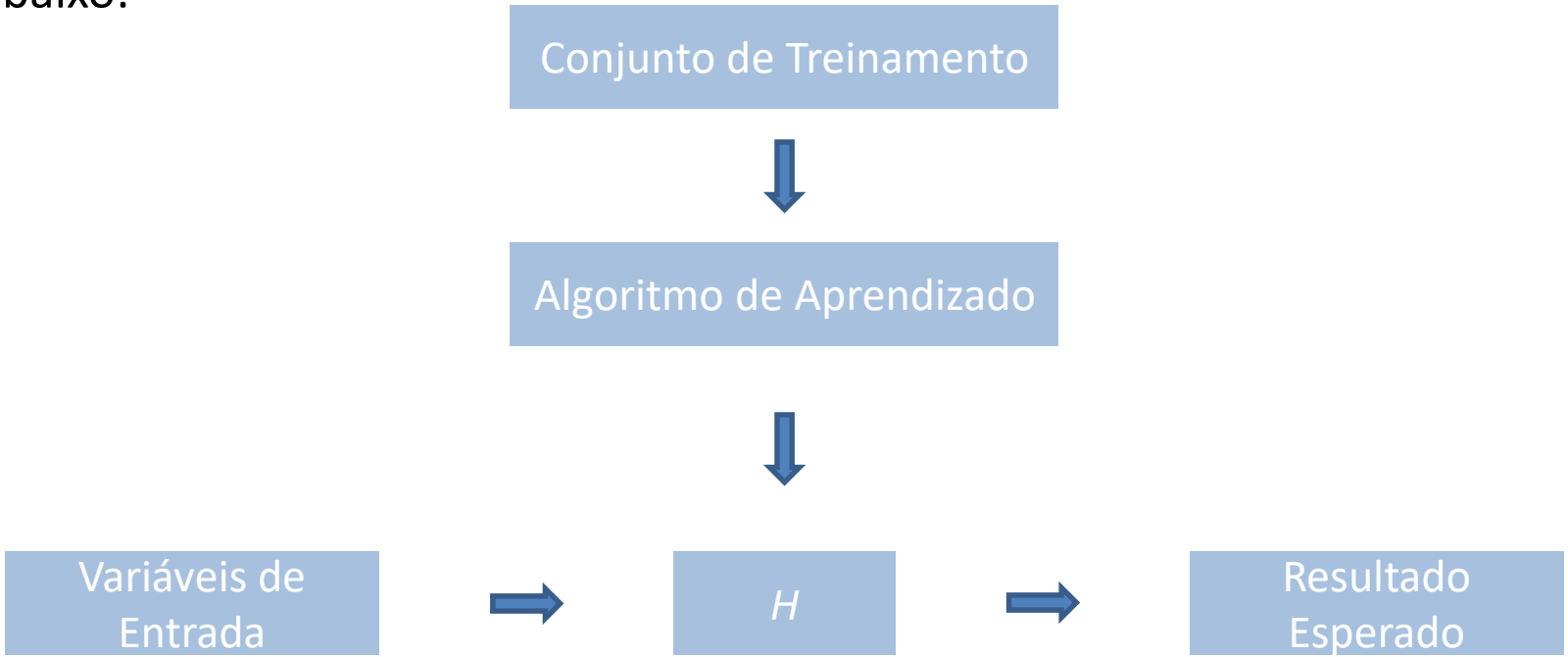
- (Mutiple) Linear Regression
- Regression Trees
- Support Vector Regression
- Random Forest
- SGD – Stochastic Gradient Descent
- KNN – K Nearest Neighbours

Regressão Linear

Suponha que eu deseje criar um algoritmo de machine learning cujo objetivo é estimar se devo ou nao abrir uma filial de minha loja num determinado lugar. Para isso, disponho de um conjunto de dados que relaciona lucro e população

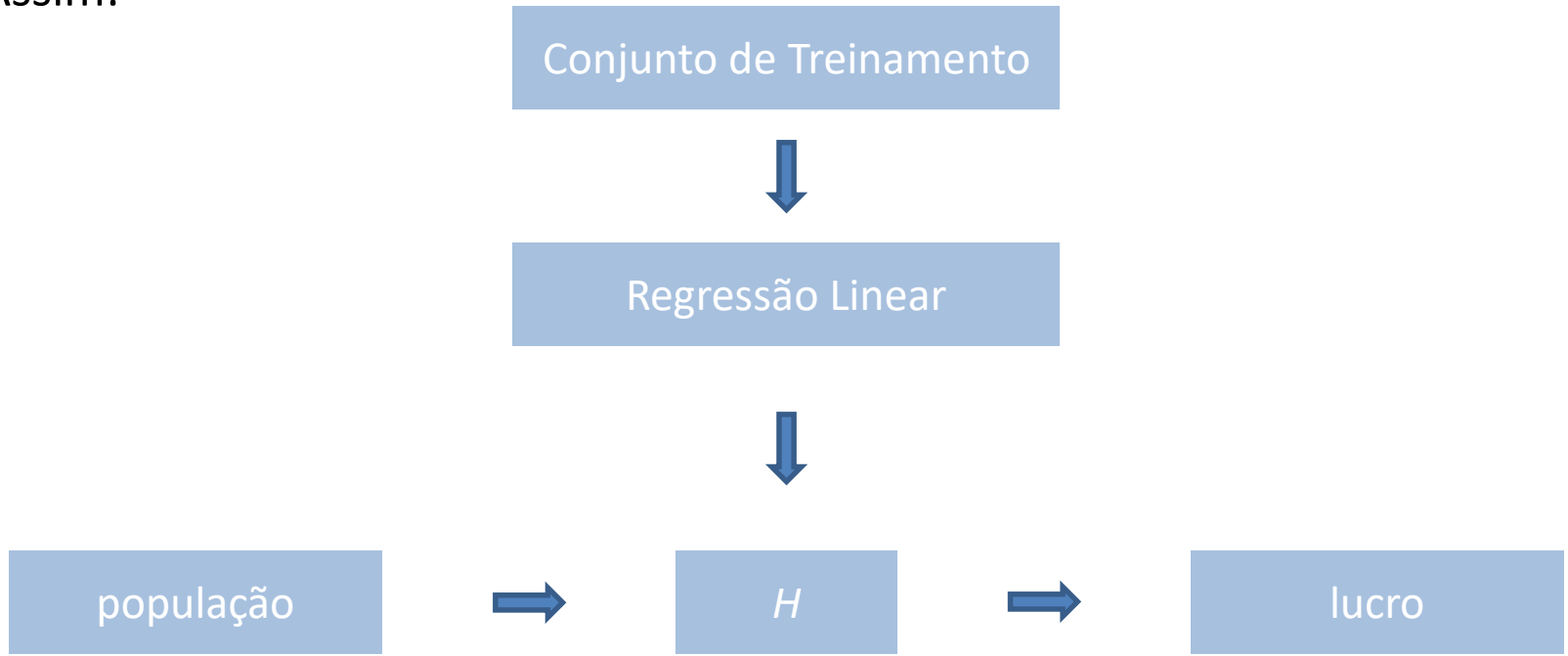


Nosso objetivo é criar um algoritmo que consiga predizer qual o lucro que terei numa cidade a partir da quantidade de pessoas que lá vivem. De modo geral, podemos representar esse objetivo a partir do diagrama abaixo:



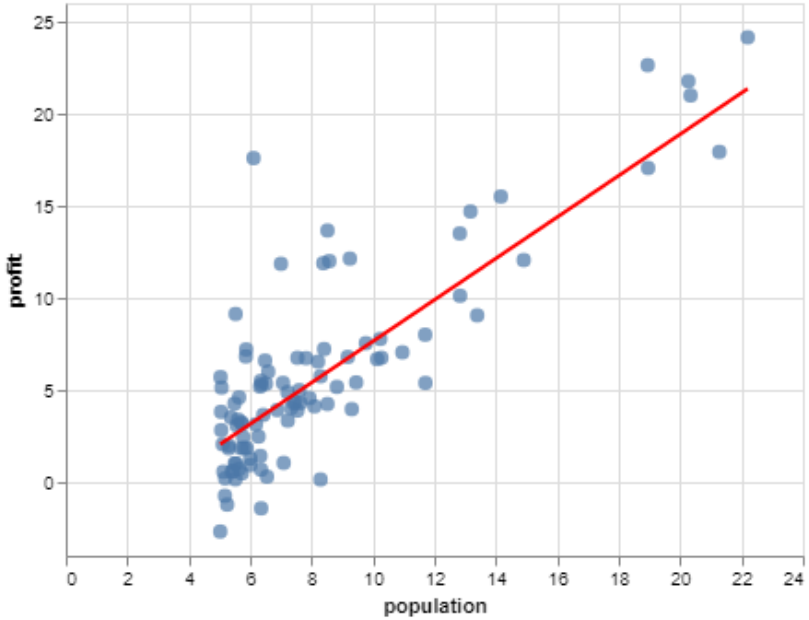
H representa uma hipótese que, a partir de um algoritmo de aprendizado, mapeia as variáveis de entreta para um resultado esperado.

Assim:



Como o próprio nome diz, a Regressão Linear irá ajustar uma reta aos dados a fim de permitir que eu estime o lucro a partir da população.

Desse modo, esperamos como resultado da Regressão Linear, algo do tipo:



Duas perguntas surgem, então:

1. Qual é a hipótese da Regressão Linear que me permite ajustar uma reta aos dados?

2. Dada a minha hipótese, como eu sei que a reta obtida é a melhor possível?

Vamos começar definindo nossa hipótese.

Se eu ajusto uma reta aos meus dados, nada mais óbvio que recorrer à Álgebra Linear e lembrar qual equação dá origem à uma reta:

- $h_\theta(x) = \theta_0 + \theta_1 x$

θ_0 representa o coeficiente linear, ou seja, o valor numérico (ponto) por onde a reta passa no eixo das ordenadas (y).

θ_1, ou coeficiente angular, representa a inclinação da reta em relação ao eixo das abscissas (x).

Vamos analisar alguns exemplos de hipóteses para entender como a reta é criada a partir de diferentes valores de θ_0 e θ_1:

- $h_\theta(x) = 1.5 + 0x$
- $h_\theta(x) = 0.5x$
- $h_\theta(x) = 1 + 0.5x$

Dos exemplos, entendemos que, para determinar uma reta que se ajuste aos dados, precisamos determinar os valores de θ_0 e θ_1.

Há várias maneiras de determinarmos esses valores, mas precisamos de uma que nos permita dizer o quão bom são esses valores.

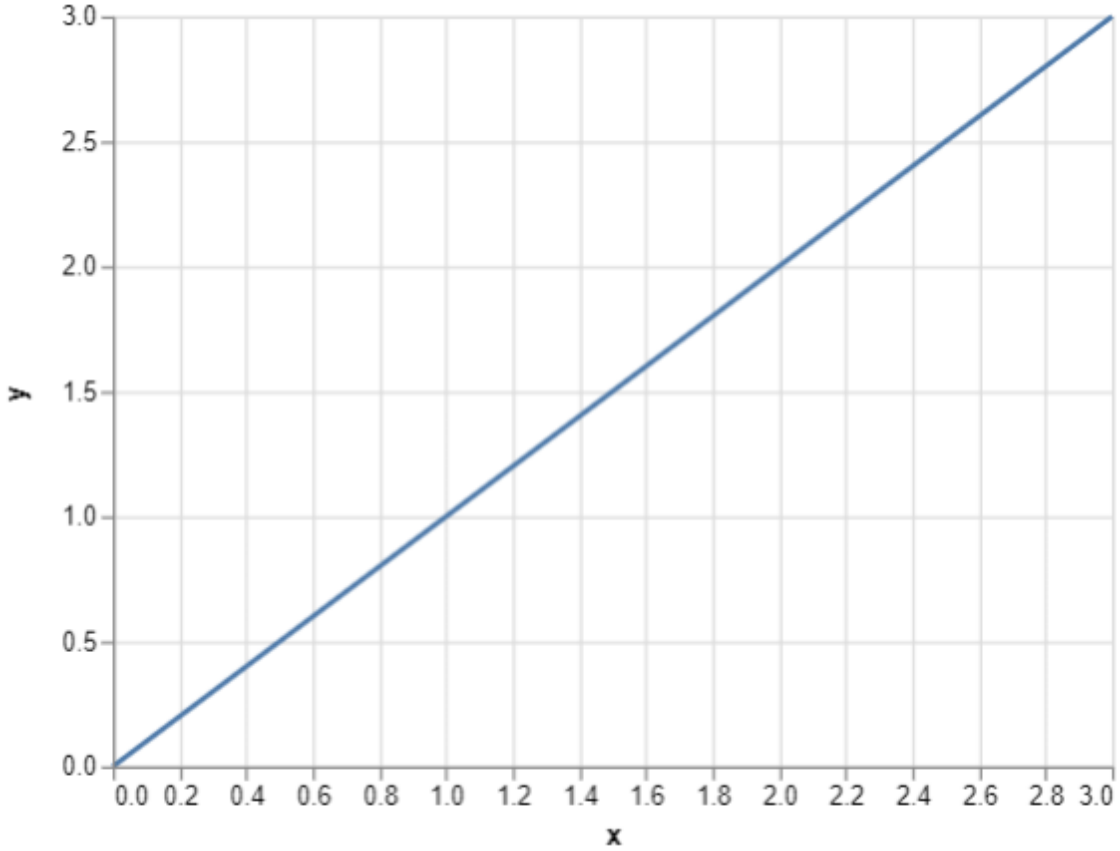
Aqui entra em cena nossa Função de Custo

Considere um conjunto fictício cujas respostas para uma determinada entrada de dados seja $y = [0,1,2,3]$.

Considere, ainda, que minha regressão linear ajustada a esses trouxe como resposta $\gamma = [0,1,2,3]$

Diante disso, qual o erro de predição de meu algoritmo?

Regressão Linear – Função de Custo: Intuição



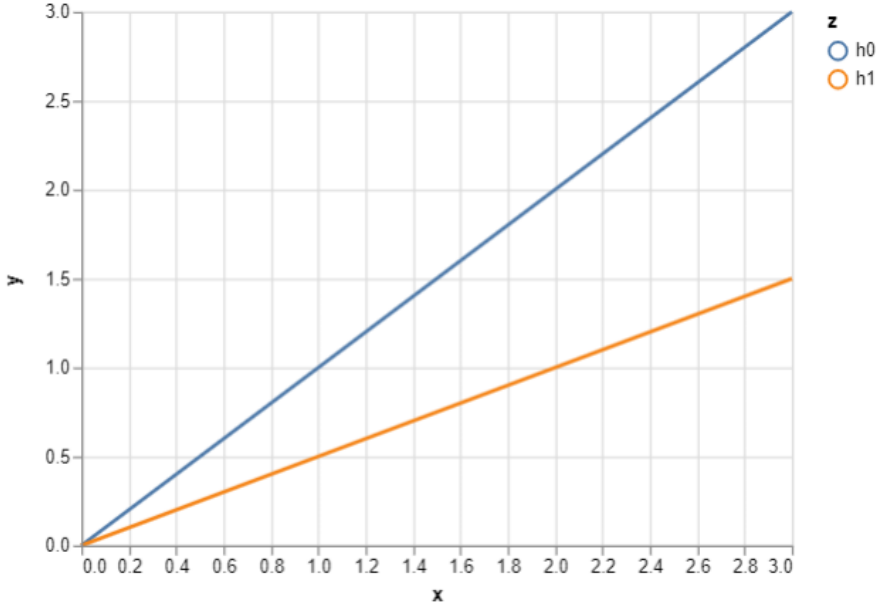
Considere as mesmas respostas $y = [0,1,2,3]$.

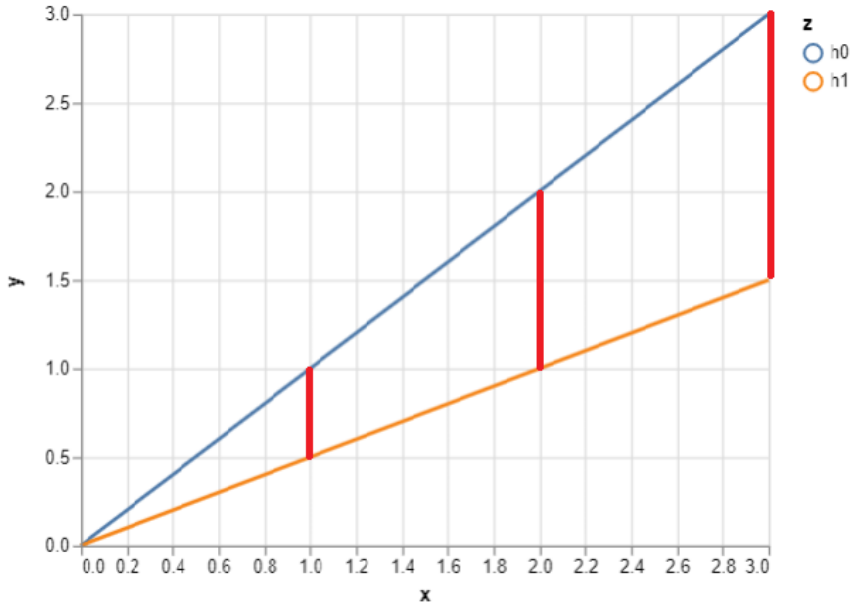
Considere, agora, que minha regressão linear ajustada a esses trouxe como resposta $\gamma = [0,0.5,1,1.5]$

Qual o erro de predição de meu algoritmo agora?

Regressão Linear – Função de Custo: Intuição

O erro pode ser representado pela diferença entre o que foi predito e o que de fato é





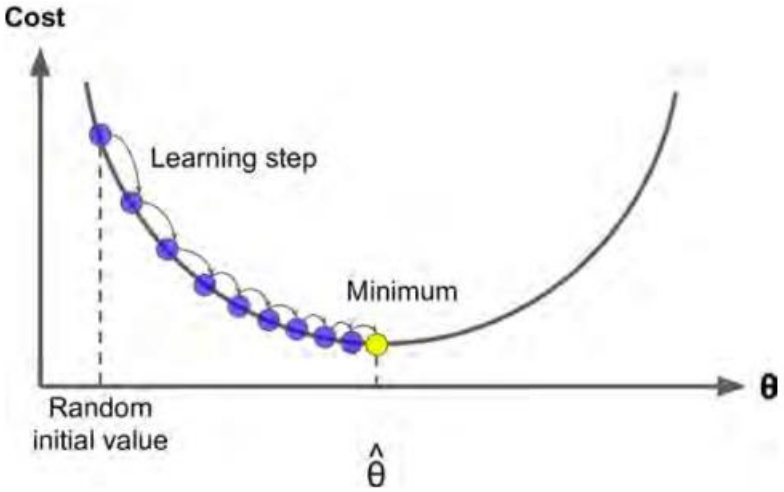
Assim, conseguimos definir nossa função de custo como sendo a diferença entre o valor predito e o valor real.

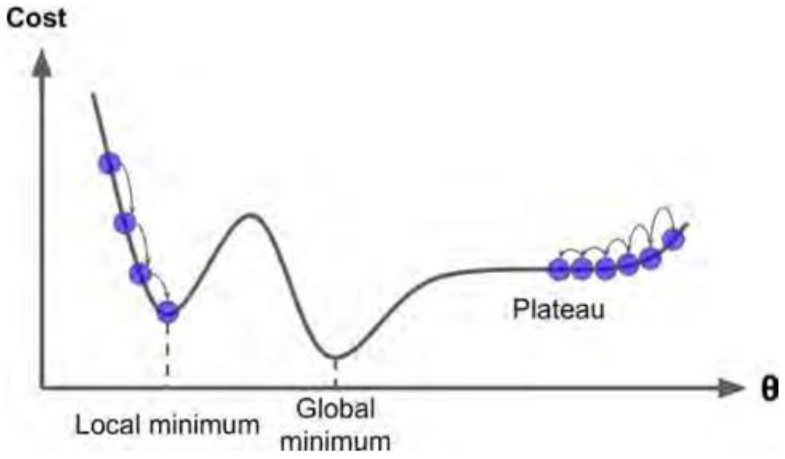
Matematicamente: $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^i) - y^i)^2$

Em que:

- $J(\theta_0, \theta_1)$: função de custo aplicada aos parâmetros θ_0 e θ_1
- m é a quantidade de exemplos de treinamento
- $h_\theta(x^i)$ é o valor predito pela Regressão Linear
- y^i é o valor real
- Elevamos ao quadrado para evitar valores negativos
- Tiramos a média para facilitar a derivada ao calcular o Gradiente Descendente

Por fim, para determinar os valores de θ_0 e θ_1, usaremos o Gradiente Descendente, cuja ideia é, iterativamente, alterar os parâmetros θ_0 e θ_1 até chegar num mínimo, que representará um bom ajuste da reta aos dados.





Para eu caminhar por minha função até atingir um mínimo, eu uso derivadas. Então, eu aplico o gradiente descentende à minha função de custo, obtendo a seguinte equação:

- $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$

α representa o meu learning rate, que simboliza o tamanho do passo que irei dar para caminhar até o mínimo da minha função.

- Se o passo for curto demais, levarei mais iterações para atingir o mínimo;
- Se for longo demais, minha função pode não convergir, ou mesmo divergir.

Posso reescrever a seguinte equação

- $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$

Como:

- $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i)^2$

Resolvendo a derivada para θ_0 *e* θ_1, temos:

- $\theta_0 = \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^i - y^i)$

- $\theta_1 = \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^i - y^i)\, x^i$

Para mais detalhes sobre o processo derivativo, acesse esse link

Resumidamente, para aplicar Regressão Linear a um conjunto de dados, eu faço:

1. Inicio θ_0, θ_1 com algum valore aleaório (geralmente 0)
2. Calculo o gradiente aplicado à função de custo:
   1. $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta \left(x^i \right) - y^i \right)^2$
3. Iterativamente, repito o processo até minha função de custo chegar a um valor mínimo ou até atingir a quantidade de iterações estabelecida

Regressão Linear Multivariada

Até então, usamos apenas a feature população para estimar o lucro e decidir se vamos ou não abrir uma filial de nossa loja nessa cidade.

Na vida real, muitos outros fatores (features) influenciam essa tomada de decisão: renda per capita, localização, logística, etc.

Assim, estamos falando agora de várias variáveis que irão compor nossa hipótese

Desse modo, tratamos θ e x como vetores de $n + 1$ posições, em que n é a quantidade de características.

- $h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 \ldots \theta_n x_n$

Por conveniência, adotamos $x_0 = 1$, assim, podemos multiplicar esses dois vetores para obtermos nossa hipótese

Vejamos o exemplo do dataset consumo_cerveja

Logo, o gradiente descendente aplicado à Regressão linear multivariada é dado por:

- $\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0 \dots \theta_n)$

De modo detalhado, temos:

- $\theta_j = \theta_j - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x^i) - y^i) x_j^i$

Implementação

Regression Trees

Regression Trees

Classification and Regression Trees, ou CART, é um termo usado para se referir aos algoritmos de Árvore de Decisão que podem ser usados para modelar problemas tanto de classificação quanto de regressão.

O Algoritmo CART é o fundamento para importantes algoritmos como *Bagged Decision Trees, Boosted Decision Trees* e *Random Forest*.

Na aula de hoje, focaremos em Regression Trees

Regression Trees - Intuição

Uma Árvore de Decisão usa uma estrutura de árvore para representar um número de possíveis caminhos de decisão e um resultado para cada caminho.

É similar a um jogo de advinha baseado em perguntas:

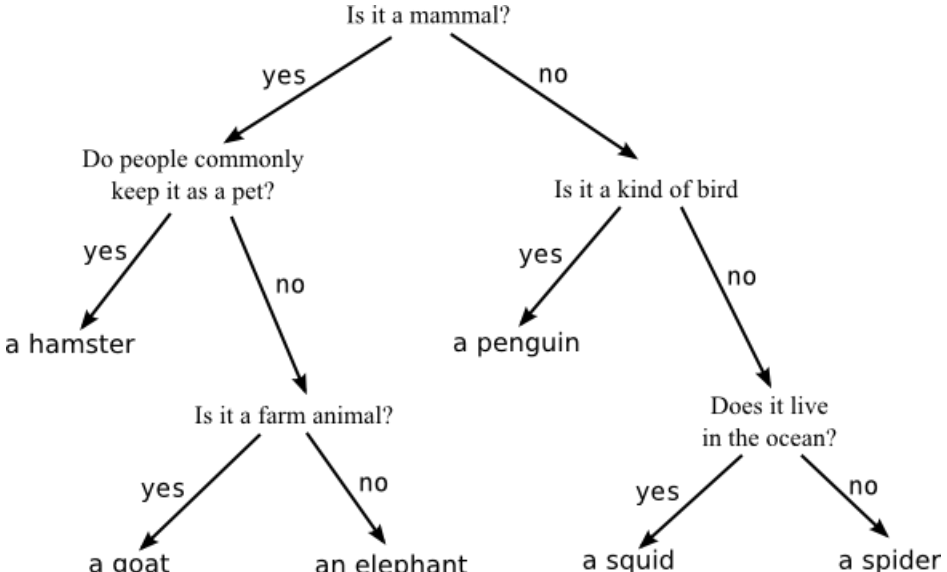
- “Estou pensando em um animal.”
- “É um mamífero?”
- “Sim.”
- “As pessoas costumam tê-lo como um animal de estimação?”
- “Não.”
- “Ele vive numa fazenda?”
- “Não.”
- “É um elefante”
- “Sim

Regression Trees - Intuição

As escolhas correspondem ao seguinte caminho:

- “É mamífero” -> “Não o tem como animal de estimação” -> “Não vive numa fazenda” -> “Elefante”

E podemos representar o jogo (de maneira simplória) usando uma Árvore de Decisão:



Regression Trees - Intuição

Prós:

1. Fácil de entender e interpretar
2. A maneira em que se chega a uma predição é transparente
3. Podem lidar tanto com atributos numéricos quanto categóricos

Contras:

1. Alto custo computacional para encontrar a Árvore ótima
2. Overfitting

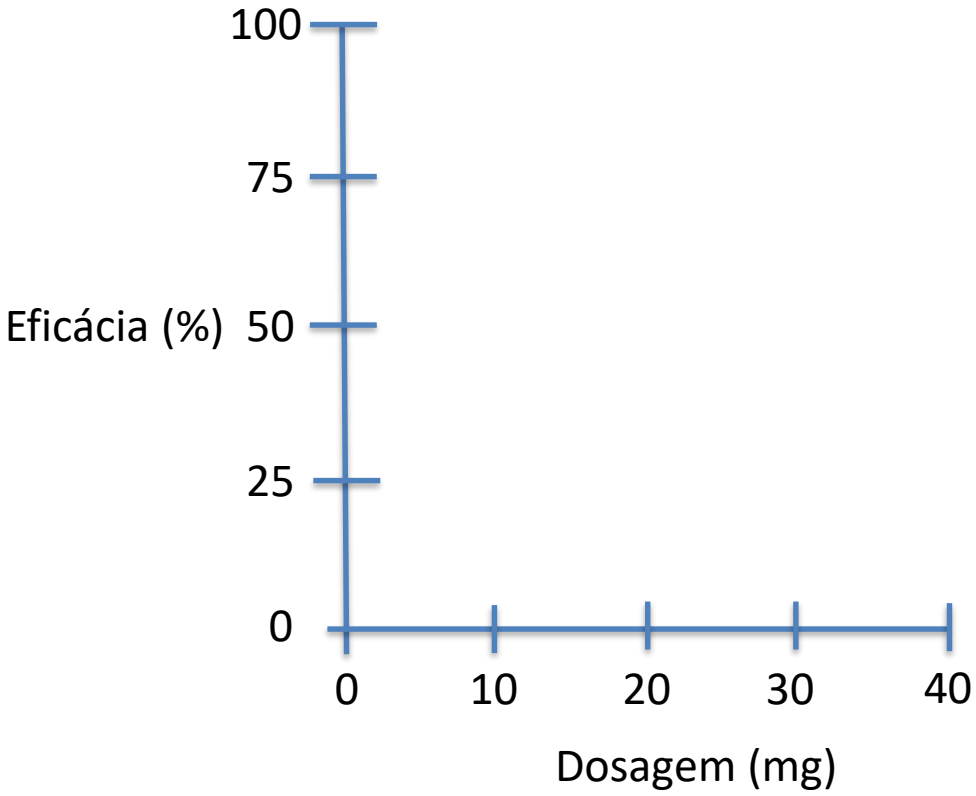
Regression Trees - Exemplo

Vamos entender a construção de uma *Regression Tree* a partir de um exemplo:

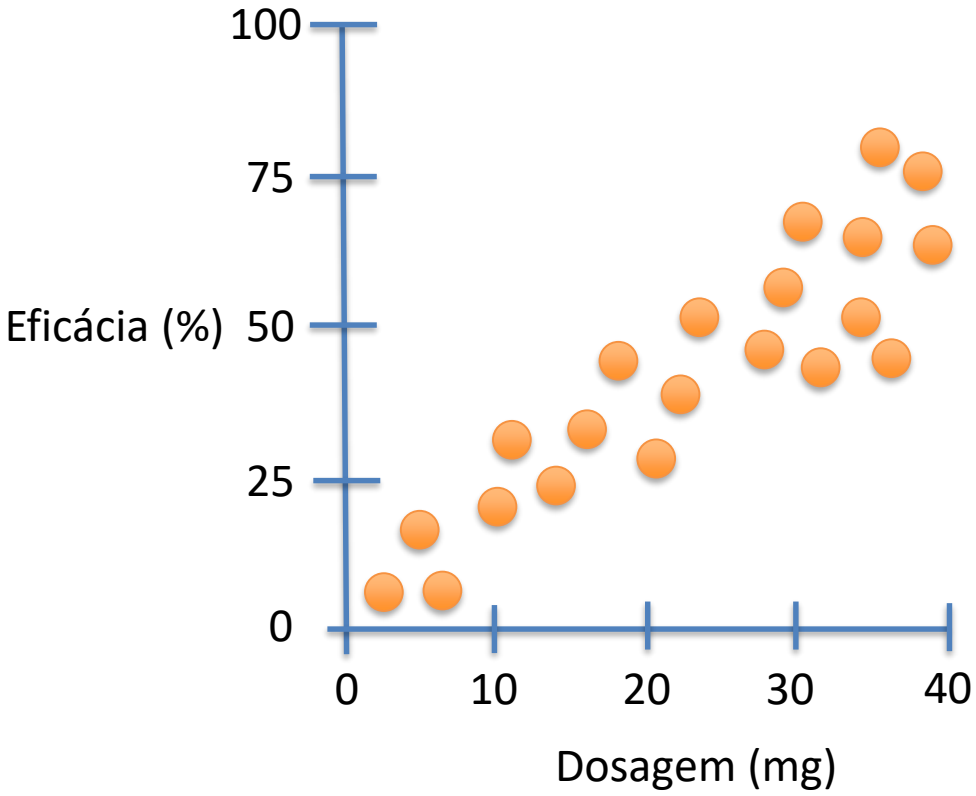
Estamos desenvolvendo um novo medicamento para combater uma gripe comum. Entretanto, ainda não sabemos a dosagem ótima.

Uma maneira de determinar seu valor é dar diferentes dosagens e mensurar o quão eficaz cada uma delas foi.

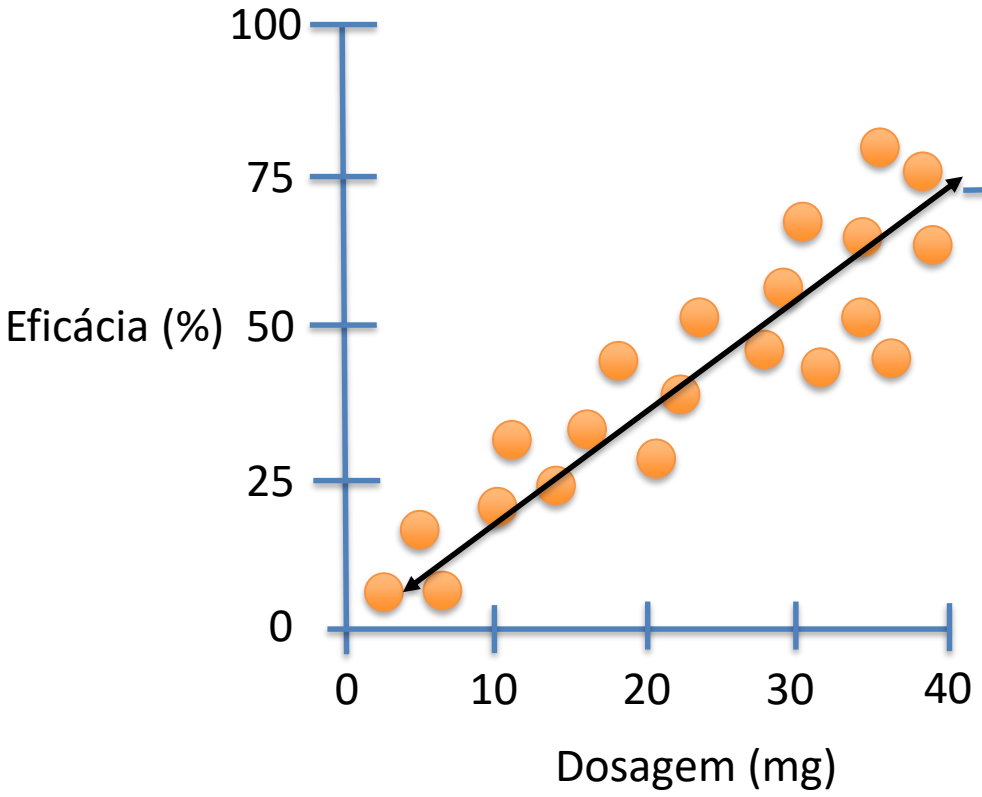
Regression Trees - Exemplo



Regression Trees - Exemplo

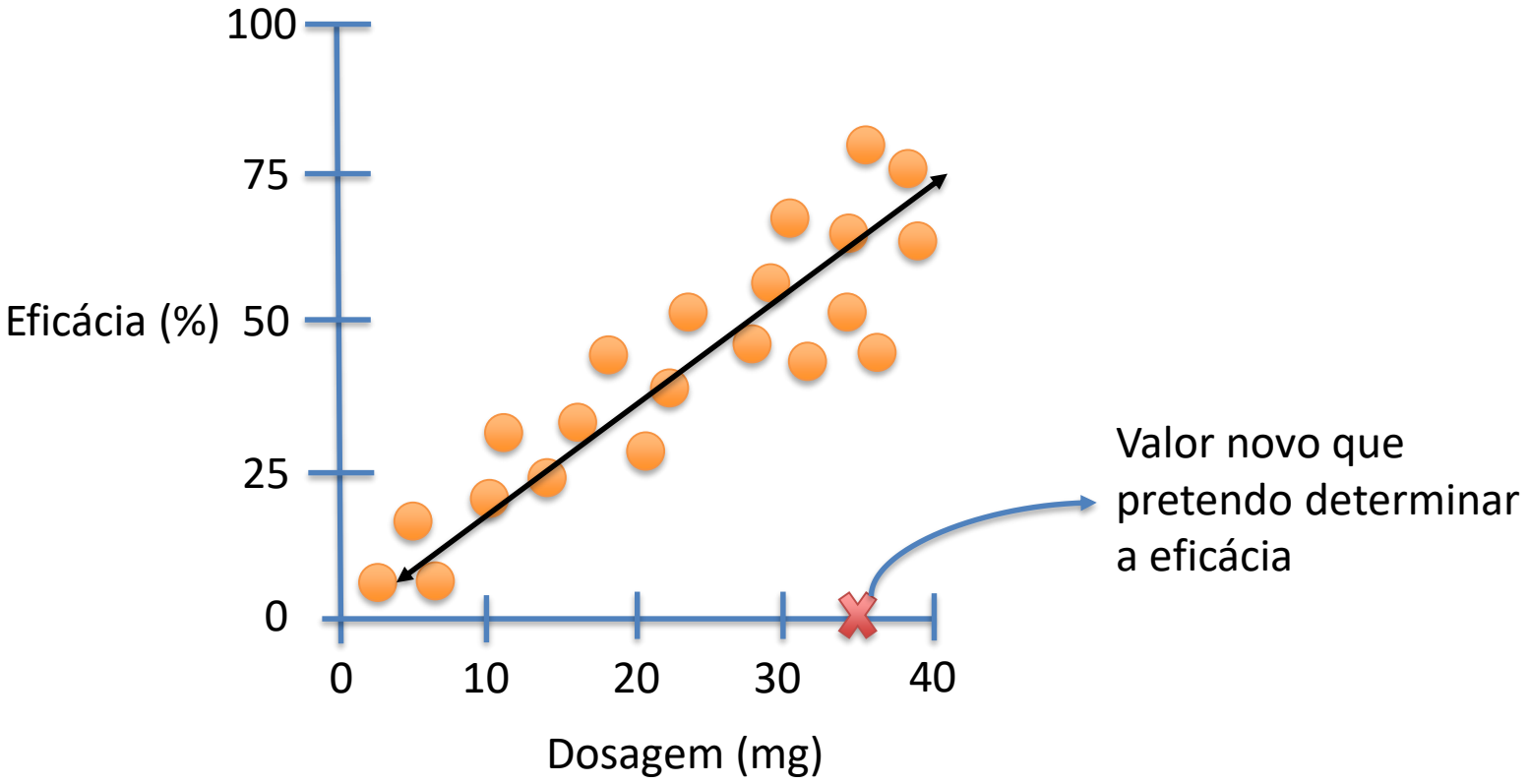


Regression Trees - Exemplo

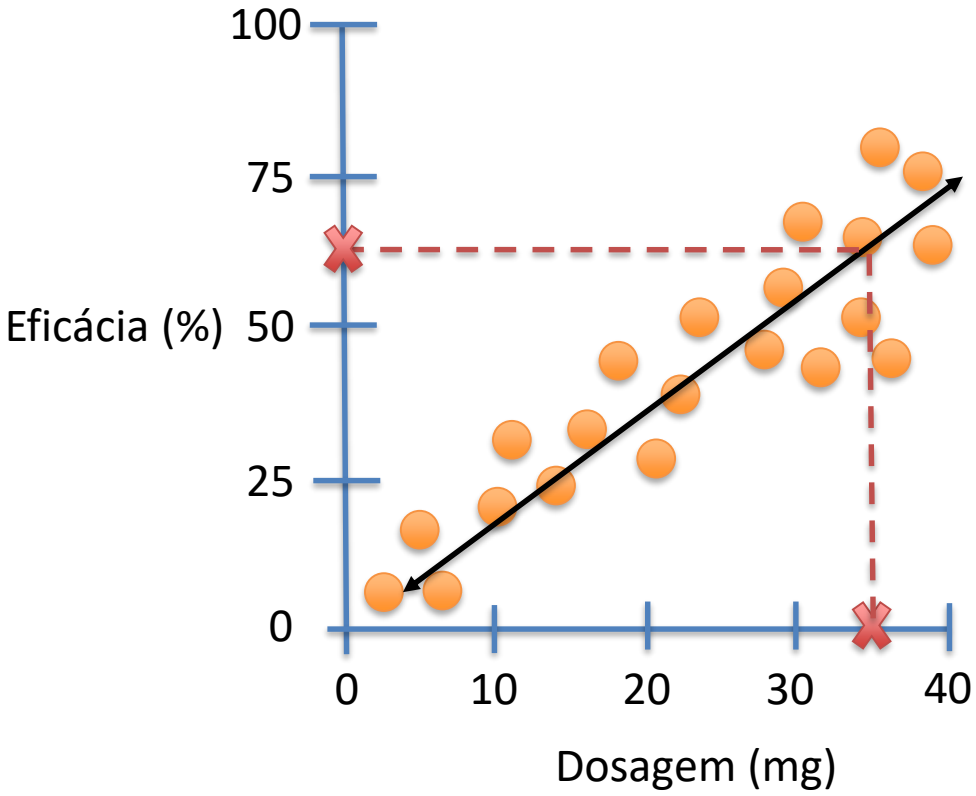


Ajustei uma reta aos dados (fit) para permitir predição de novos valores

Regression Trees - Exemplo

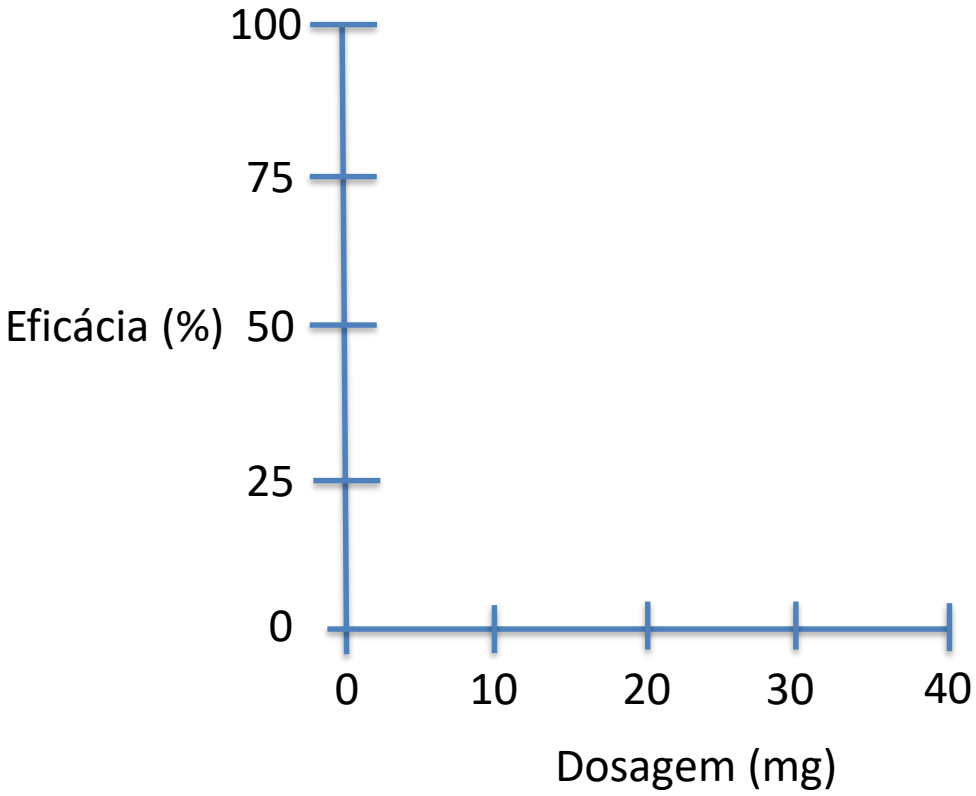


Regression Trees - Exemplo



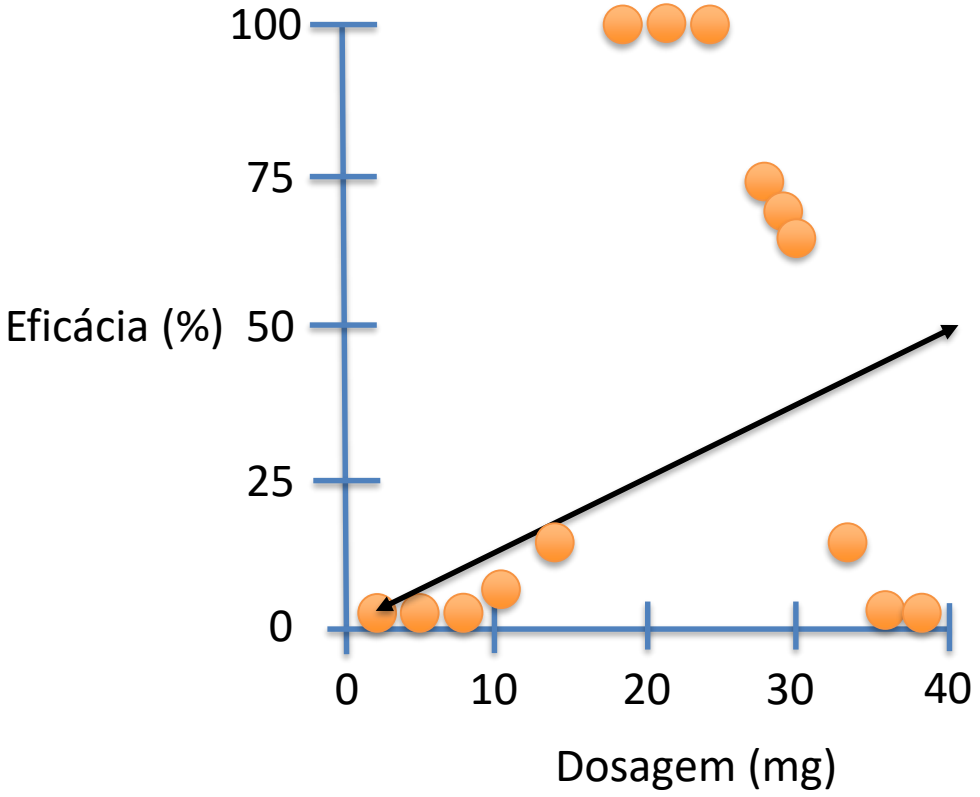
Uma dosagem de 35mg produz um resultado de 68% de eficácia

Regression Trees - Exemplo



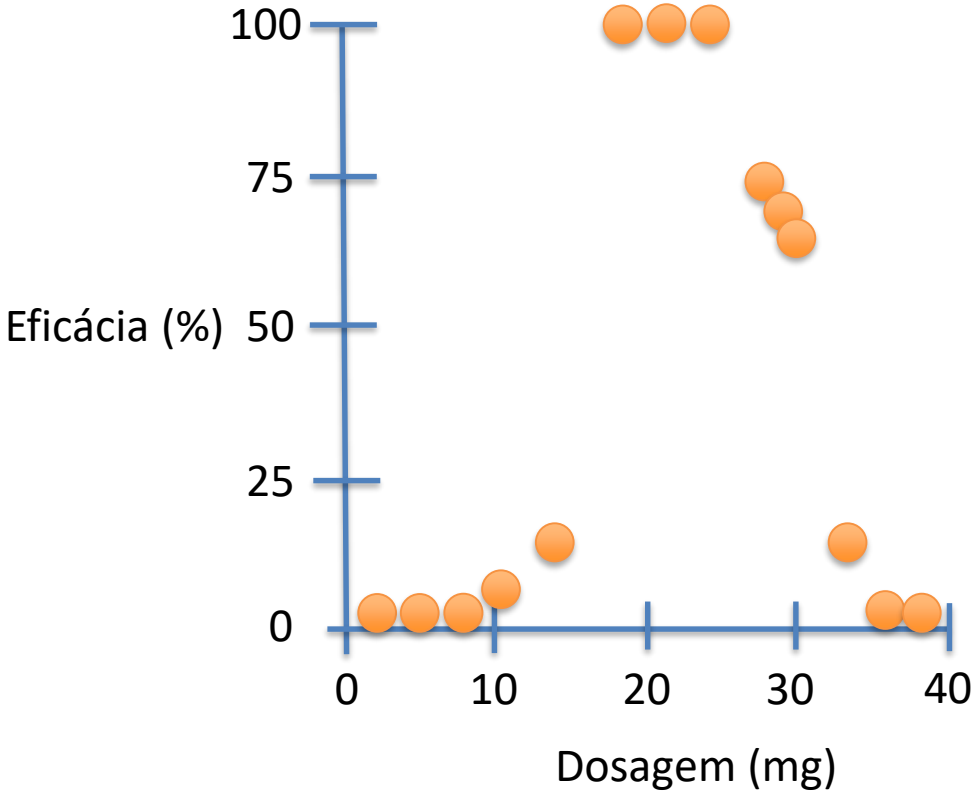
O que acontece quando meus dados possuem esse comportamento?

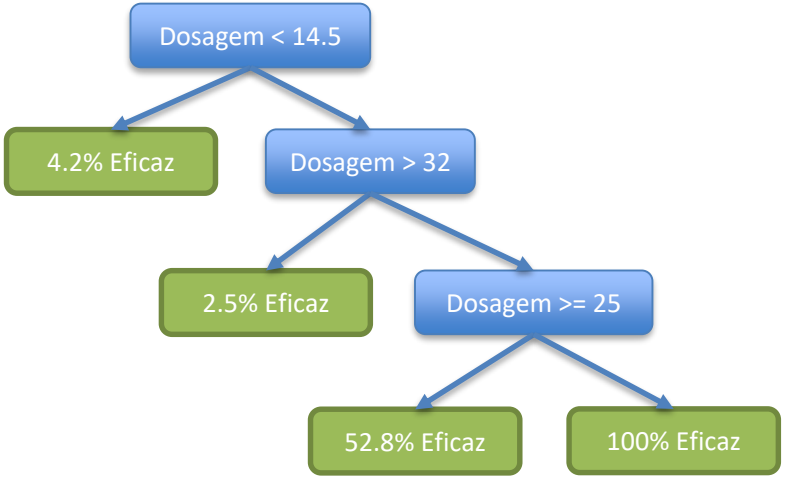
Regression Trees - Exemplo



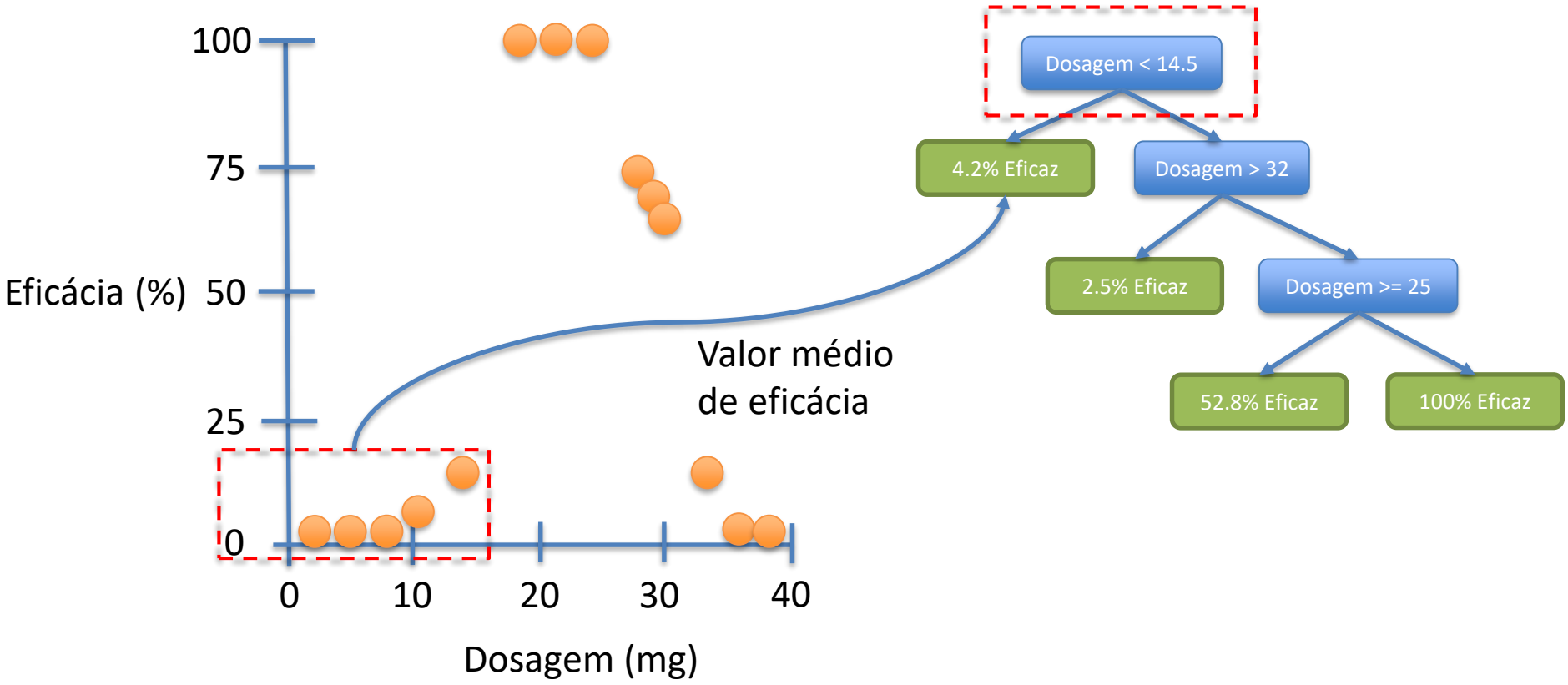
Claramente, uma simples regressão linear não resolve o problema

Regression Trees - Exemplo

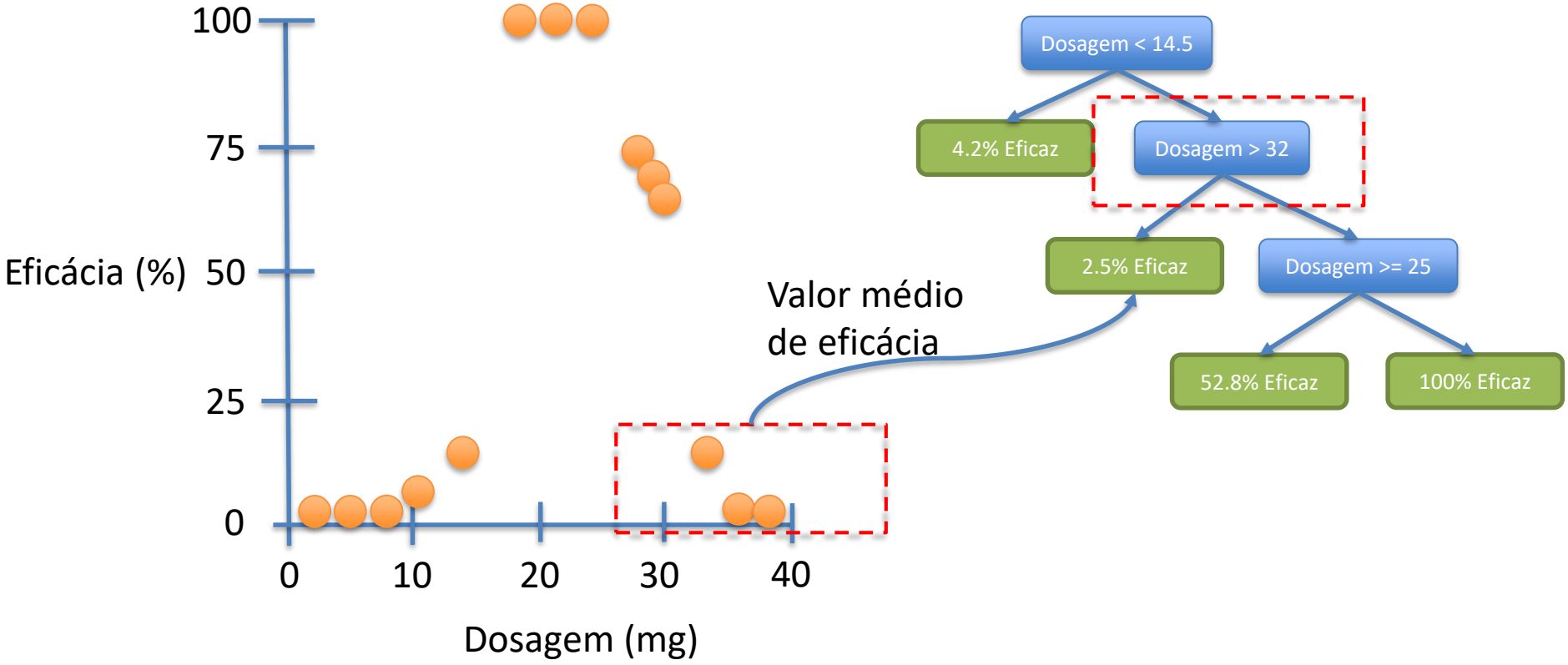




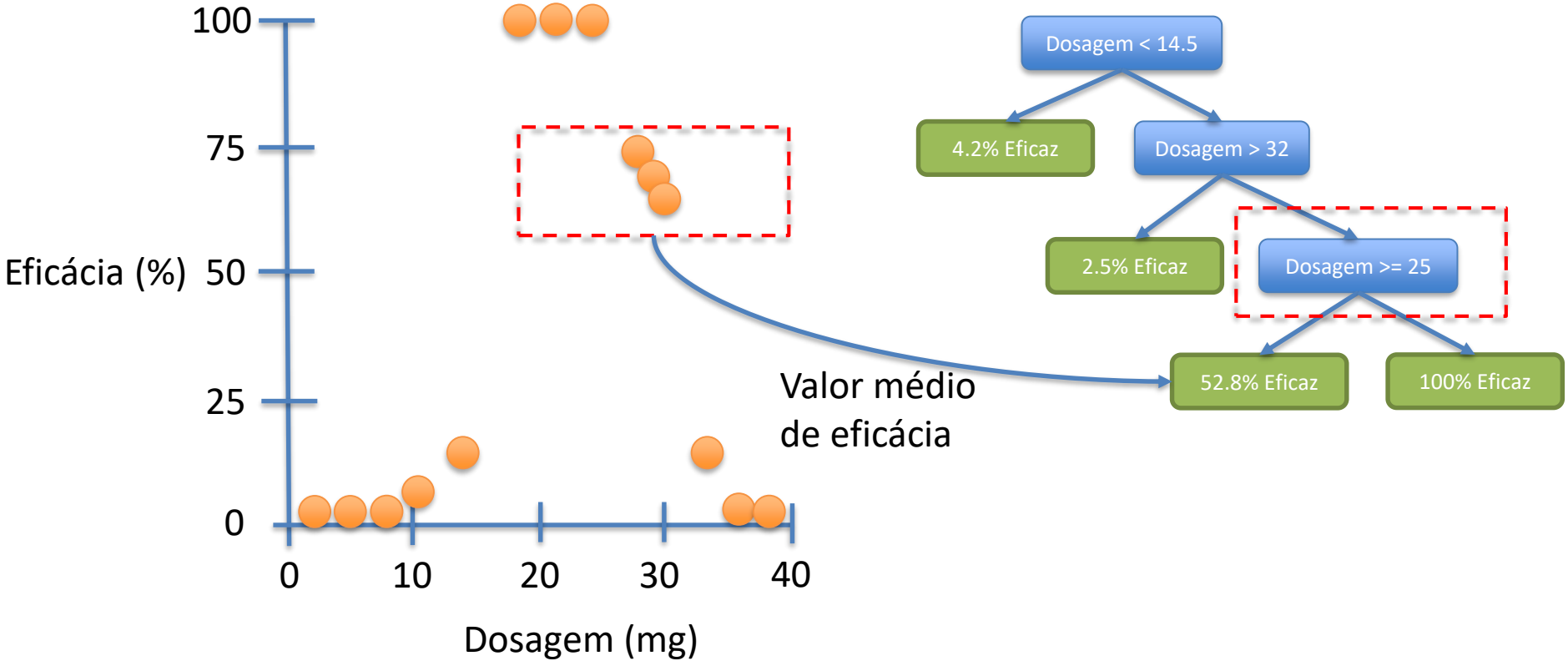
Regression Trees - Exemplo



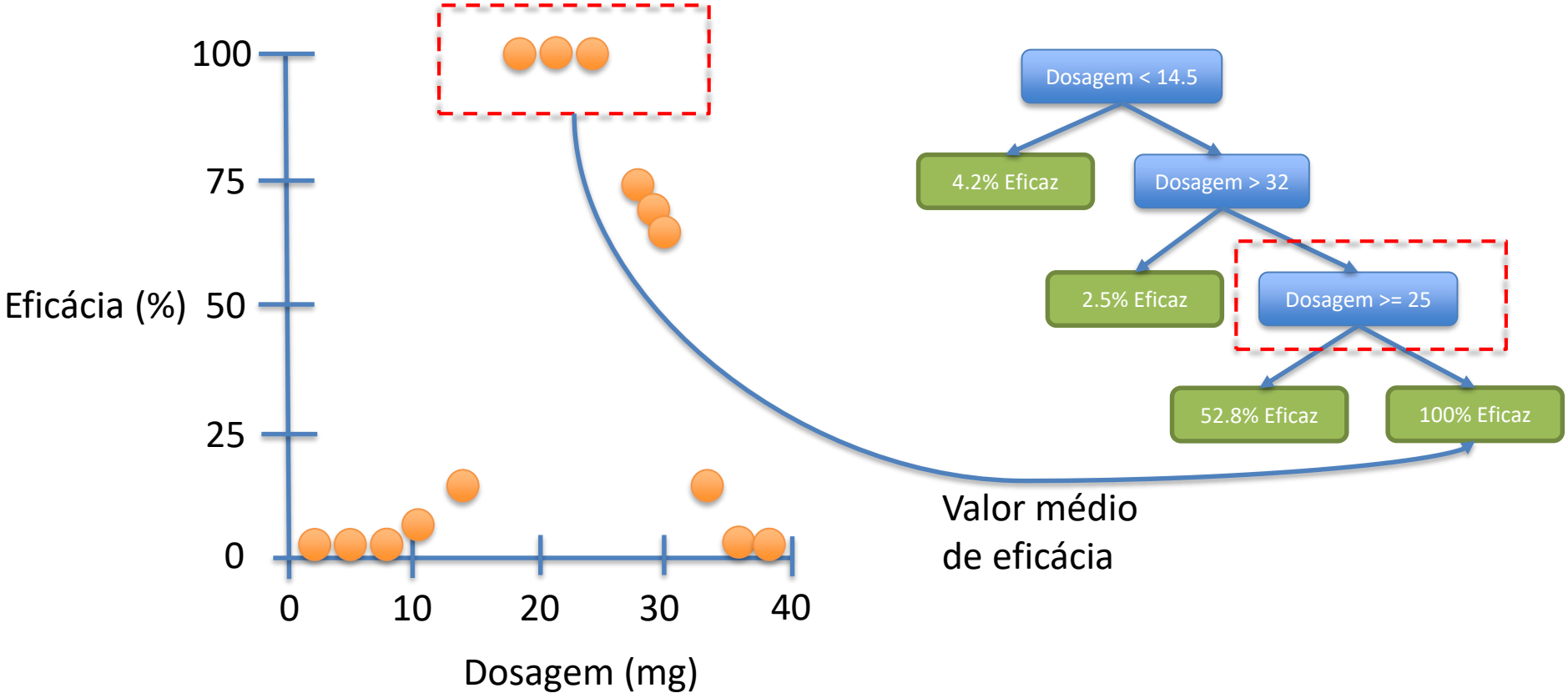
Regression Trees - Exemplo



Regression Trees - Exemplo



Regression Trees - Exemplo



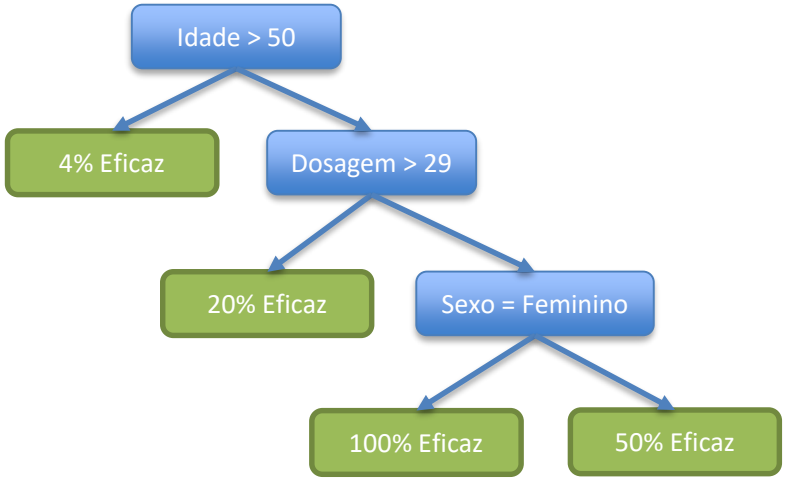
Regression Trees - Exemplo

Entretanto, o ganho de usar *Regression Trees* não está em tentar prever uma variável a partir de apenas uma *feature*. É quando usamos mais de uma feature que seu valor se torna real.

Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Regression Trees - Exemplo

E as Regression Trees lidam muito bem com regressão multivariada.

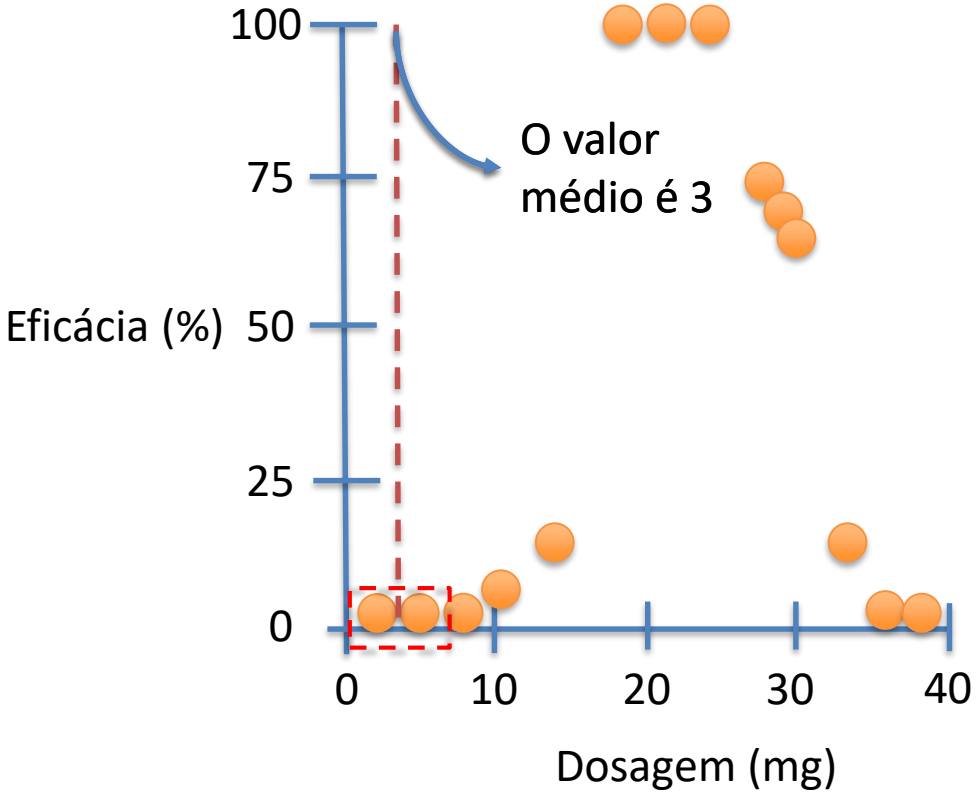


Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Regression Trees – Como constuir uma Regression Tree?

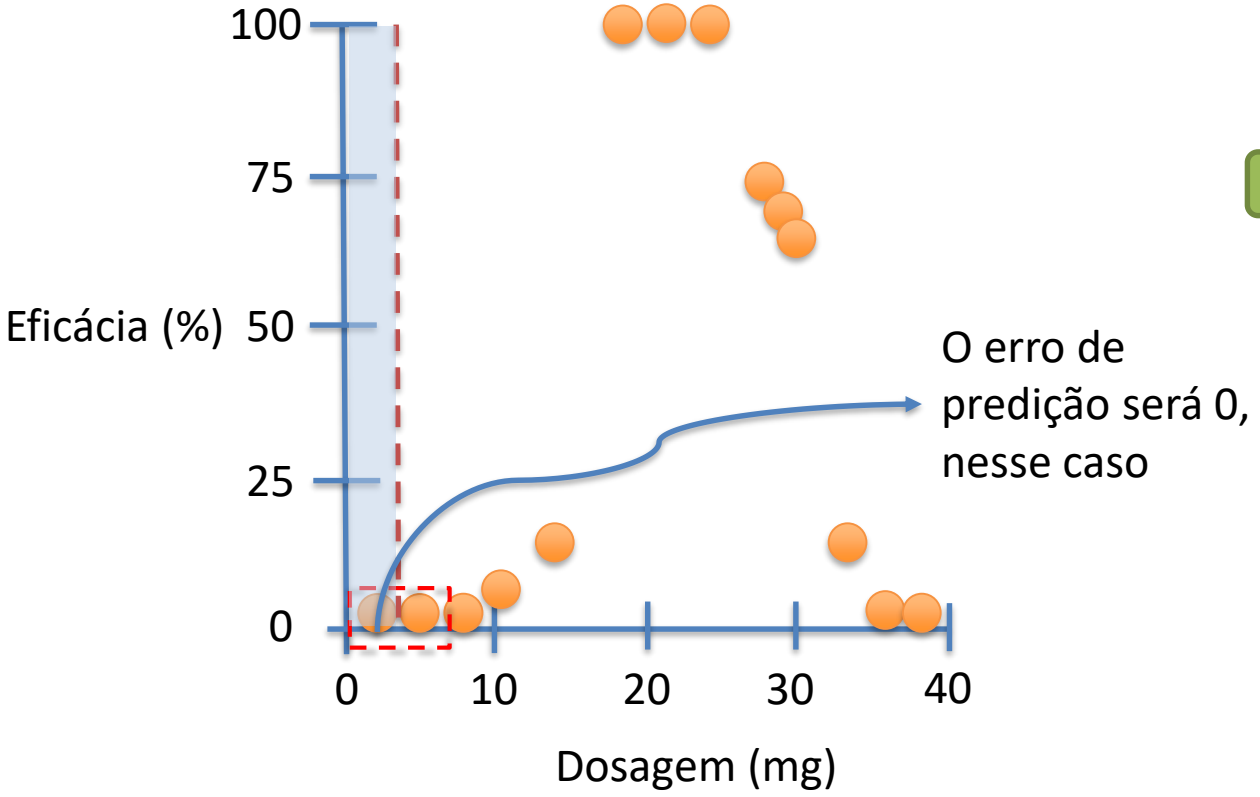
Ok, entendemos a ideia geral por trás das Regression Trees, mas como elas são construídas?

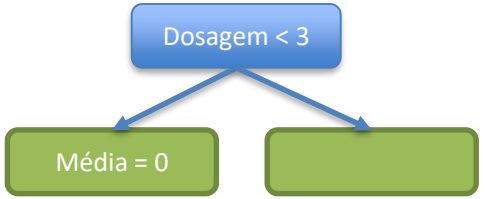
Regression Trees – Como constuir uma Regression Tree?



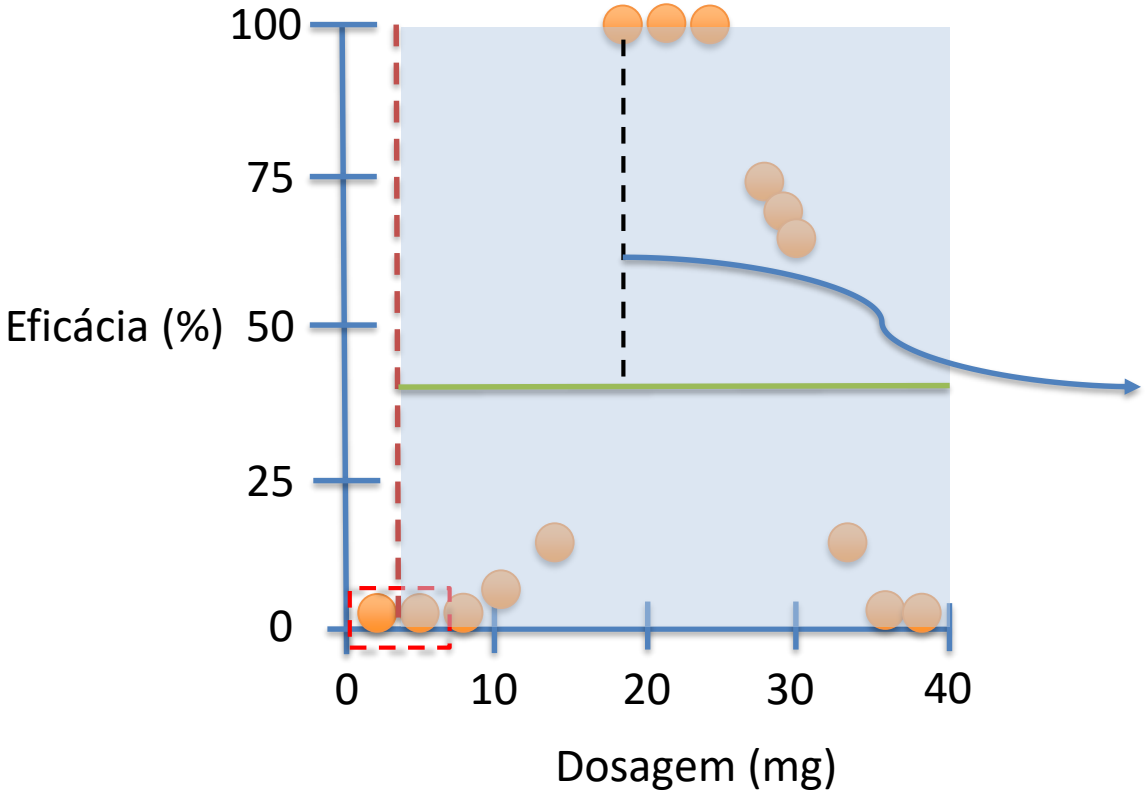
Voltando aos dados, vamos focar nas duas primeiras observações

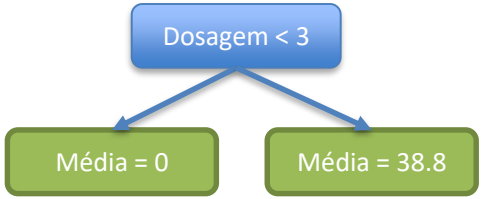
Regression Trees – Como constuir uma Regression Tree?





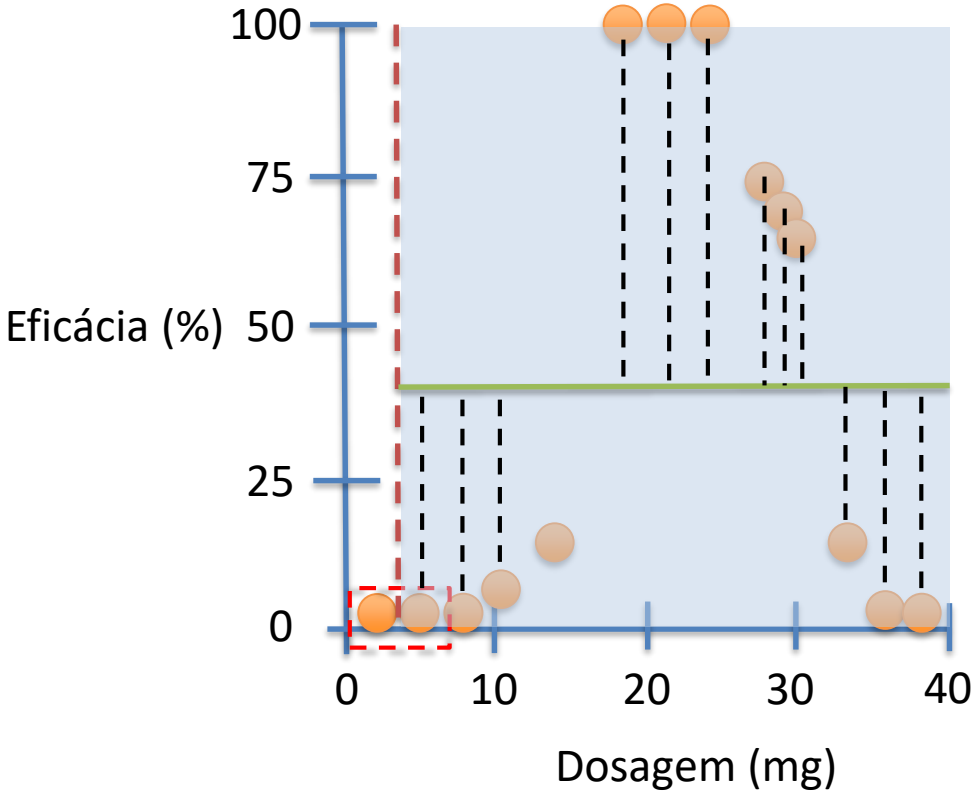
Regression Trees – Como constuir uma Regression Tree?

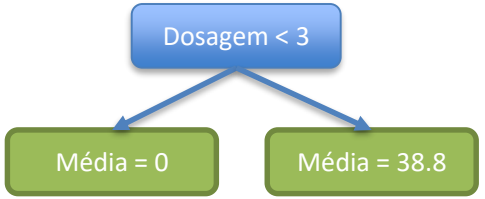




Erro de predição, também denominado ***Residual***

Regression Trees – Como constuir uma Regression Tree?

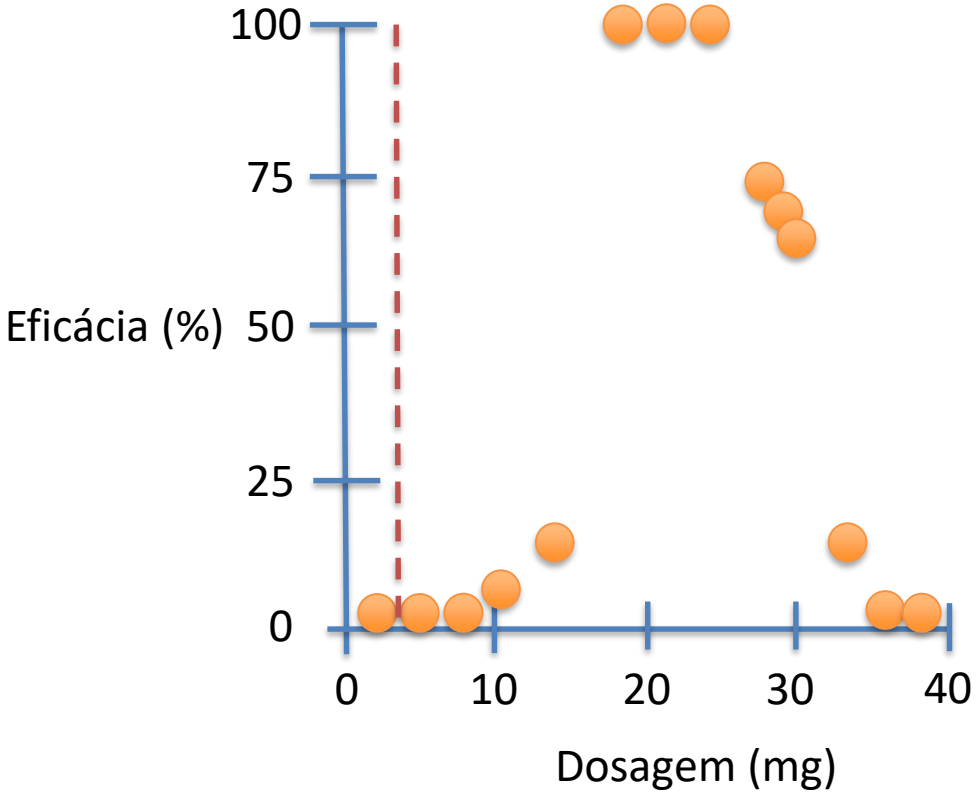




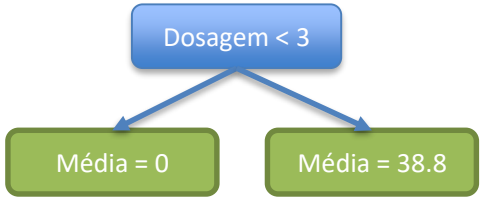
Para cada observação, podemos desenhar os Residuals, que nada mais é que a diferença entre o valor predito e observado

E podemos usar os Residuals para quantificar a qualidade das predições.

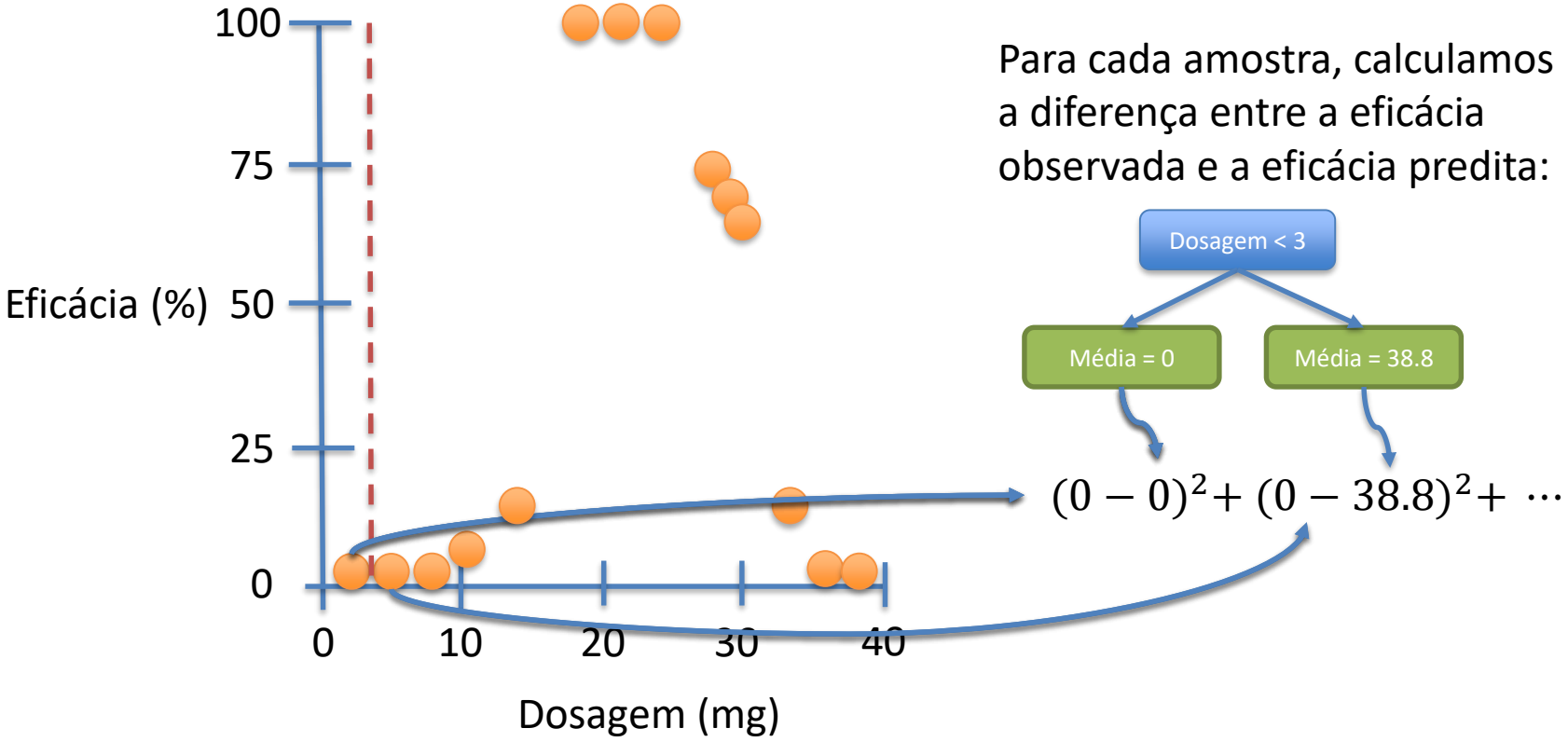
Regression Trees – Como constuir uma Regression Tree?



Para cada amostra, calculamos a diferença entre a eficácia observada e a eficácia predita:



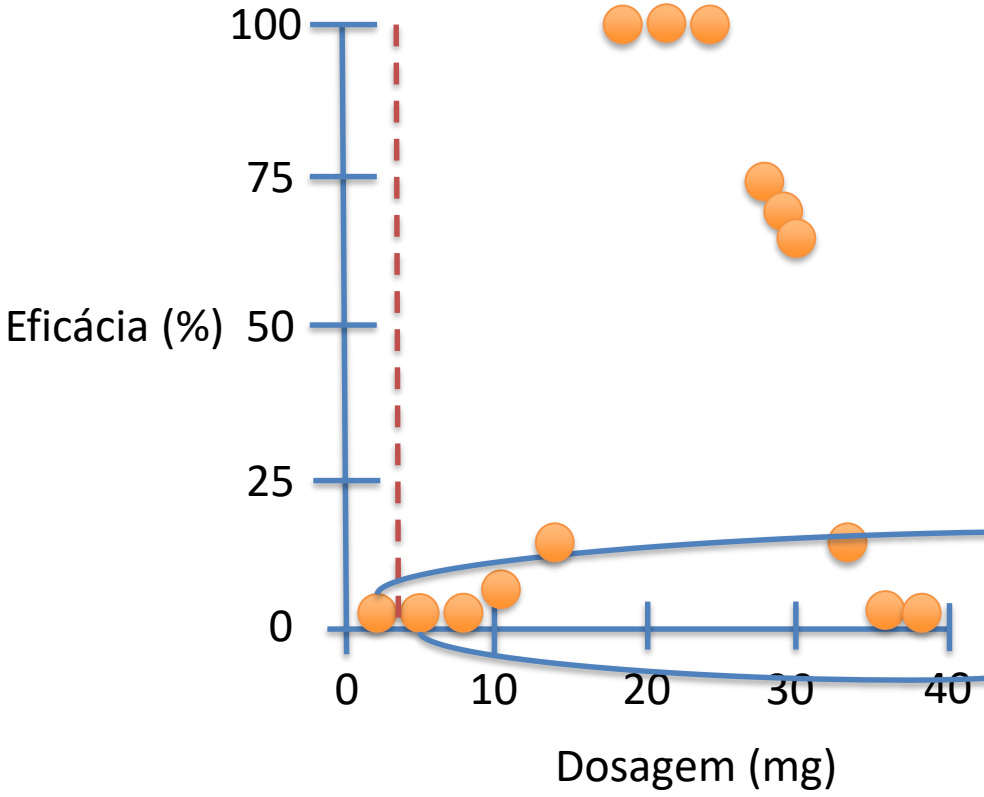
Regression Trees – Como constuir uma Regression Tree?



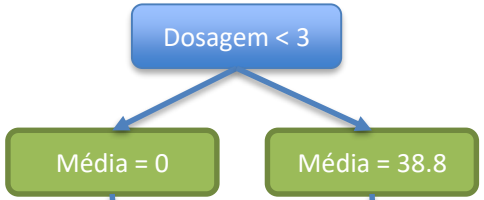
Para cada amostra, calculamos a diferença entre a eficácia observada e a eficácia predita:

$$(0-0)^2+(0-38.8)^2+\cdots$$

Regression Trees – Como constuir uma Regression Tree?



Para cada amostra, calculamos a diferença entre a eficácia observada e a eficácia predita:



$$(0-0)^2+(0-38.8)^2+\cdots=27468.5$$

RSS = Residual Sum of Squares

Regression Trees – RSS

- É uma técnica estatística usada para mensurar a quantidade de variância num conjunto de dados.

- Quanto menor o valor de RSS, melhor o modelo se ajusta aos dados (fit).

- $RSS = \sum_{i=1}^{n} (y^i - f(x)^i)^2$

  - y^i = valor observado para a amostra i
  - $f(x)^i$ = valor predito para a amostra i

Regression Trees – MSE

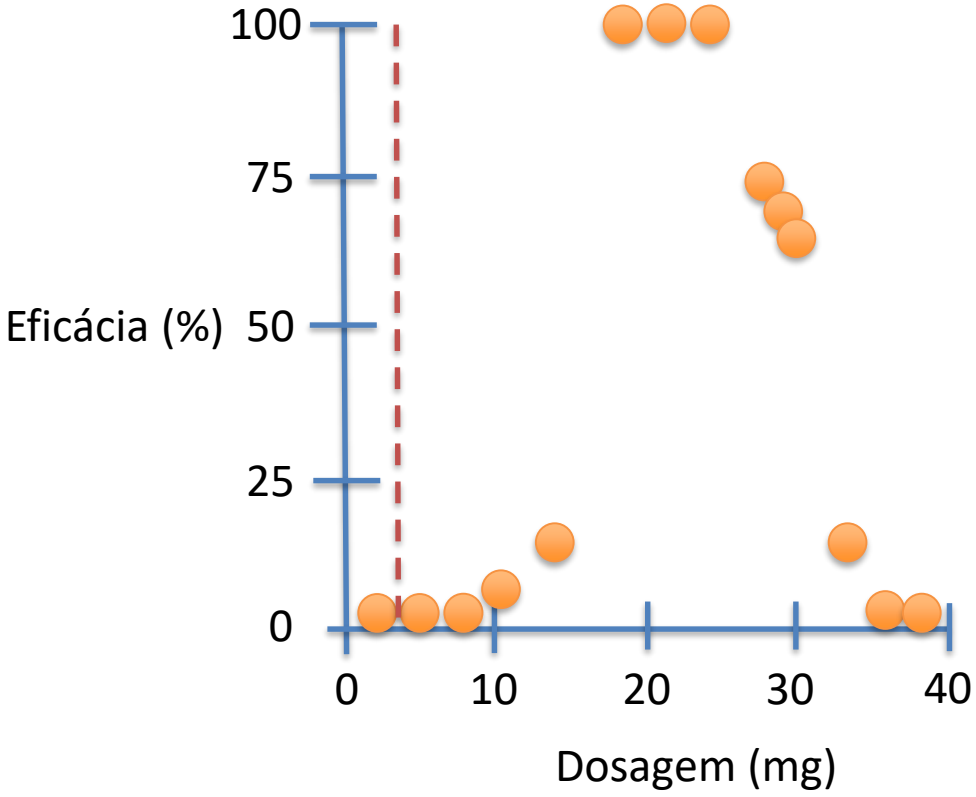
- Uma outra métrica que podemos usar para determinar a qualidade de uma árvore de regressão é o Mean Squared Error (MSE)

- A diferença em relação ao RSS é que ele divide o erro pela quantidade de amostras, oferecendo um valor médio.

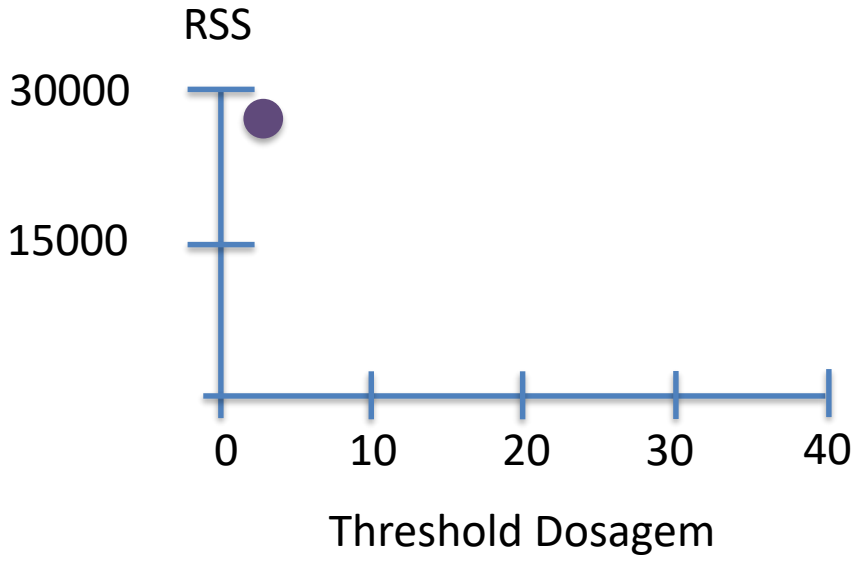
- $MSE = \frac{1}{n}\sum_{i=1}^{n}(y^i - f(x)^i)^2$

  - y^i = valor observado para a amostra i
  - $f(x)^i$ = valor predito para a amostra i

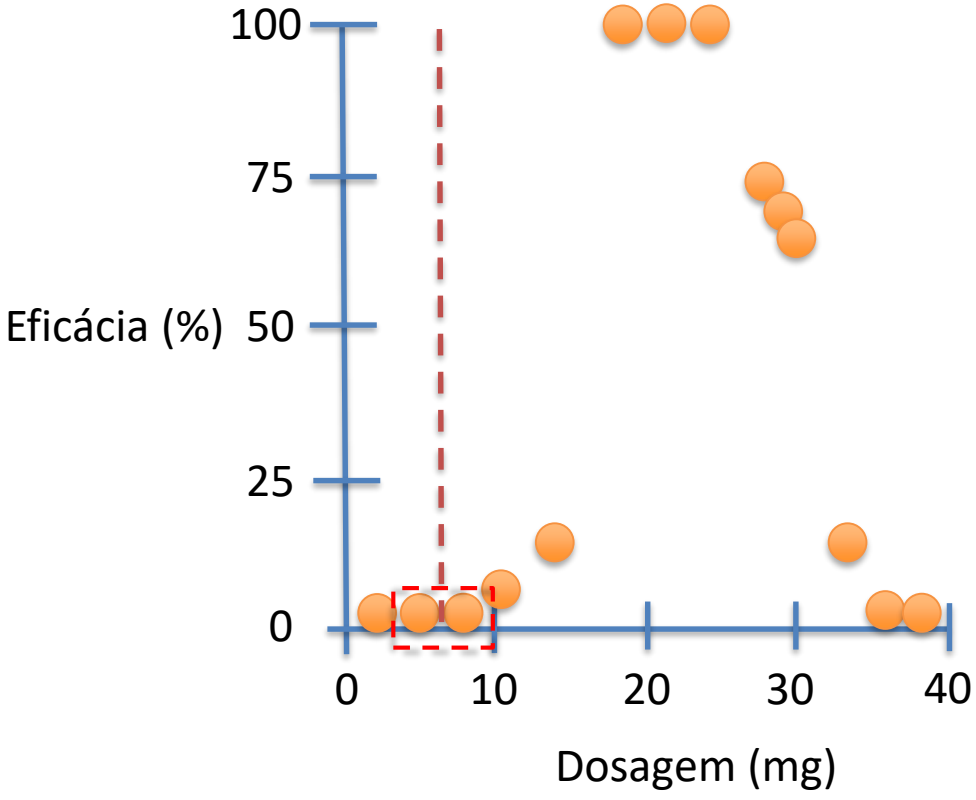
Regression Trees – Como constuir uma Regression Tree?



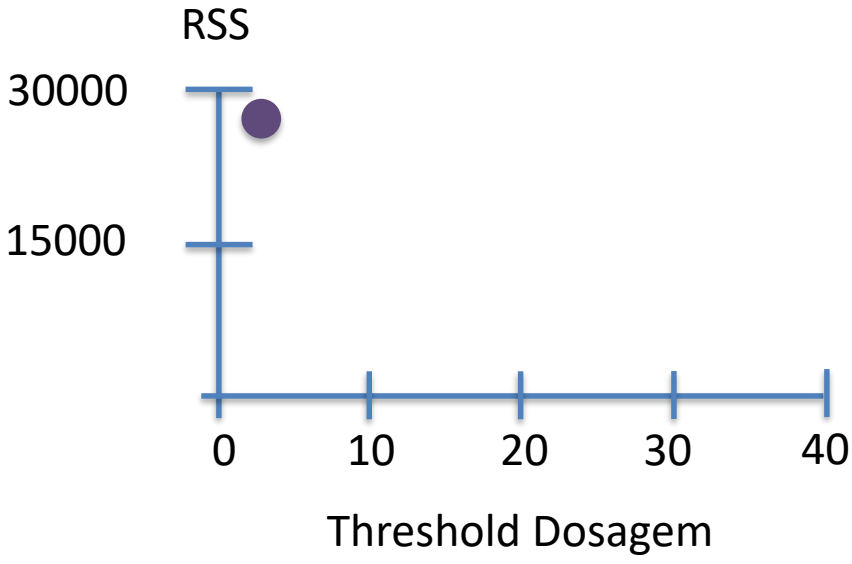
Podemos plotar o valor de RSS de acordo com cada threshold de dosagem para escolher o melhor ponto raiz para construir a árvore



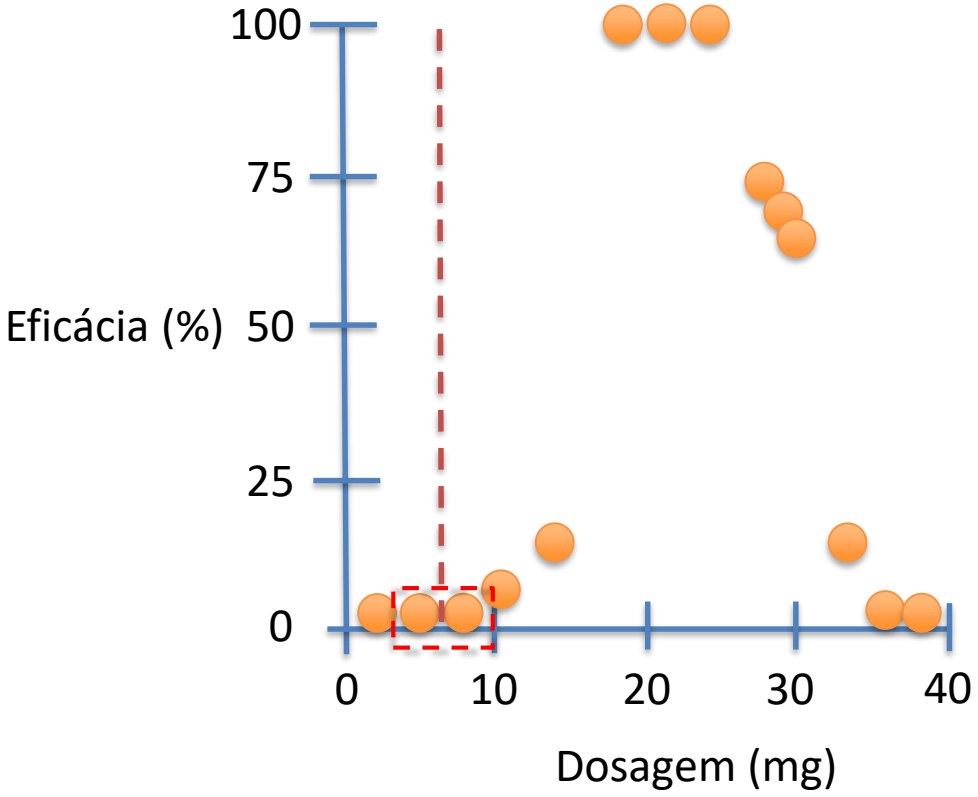
Regression Trees – Como constuir uma Regression Tree?

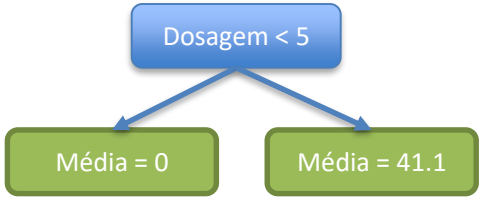


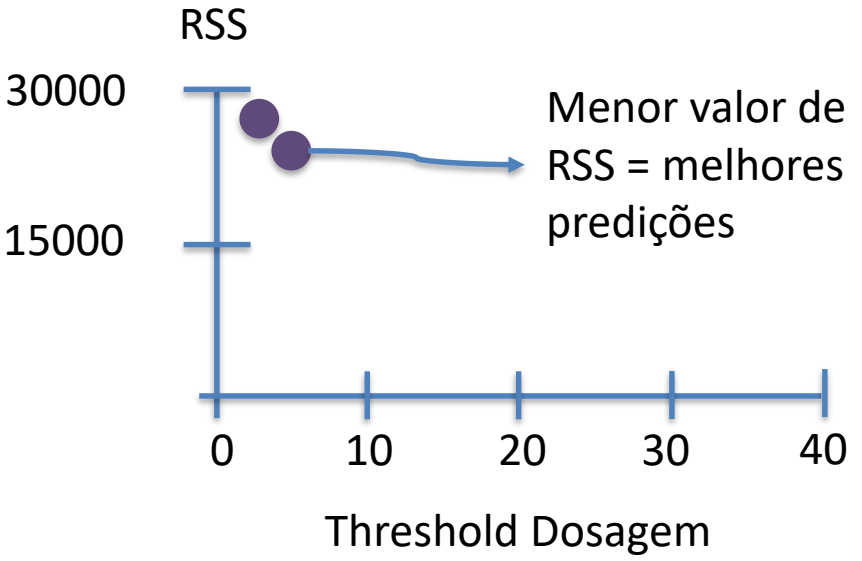
Agora, pegamos o valor médio das duas amostras subsequentes a primeira e criamos um novo threshold



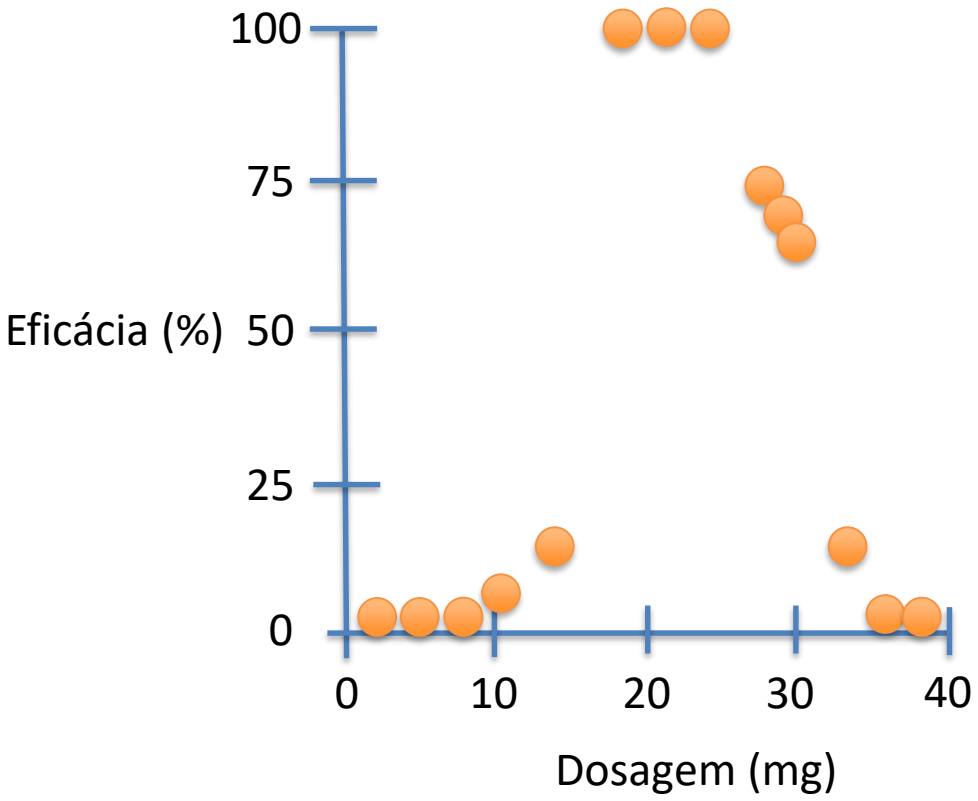
Regression Trees – Como constuir uma Regression Tree?



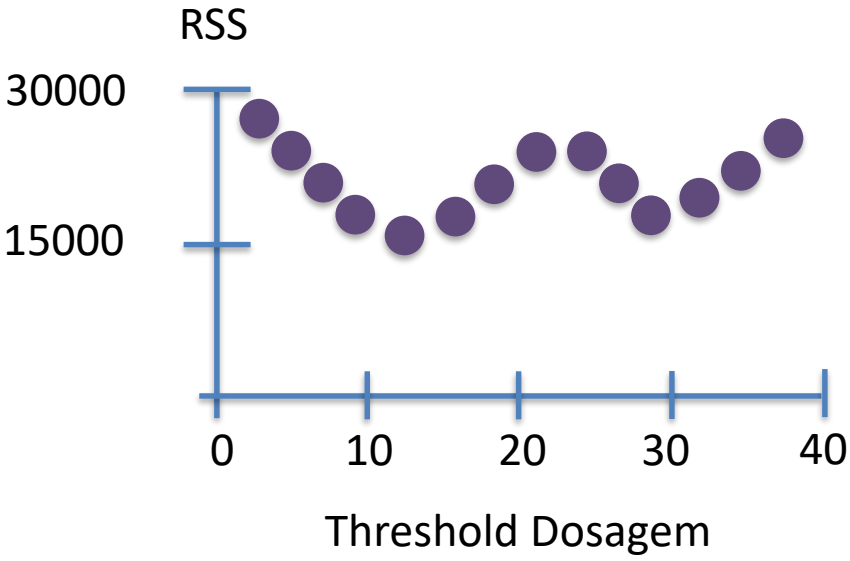




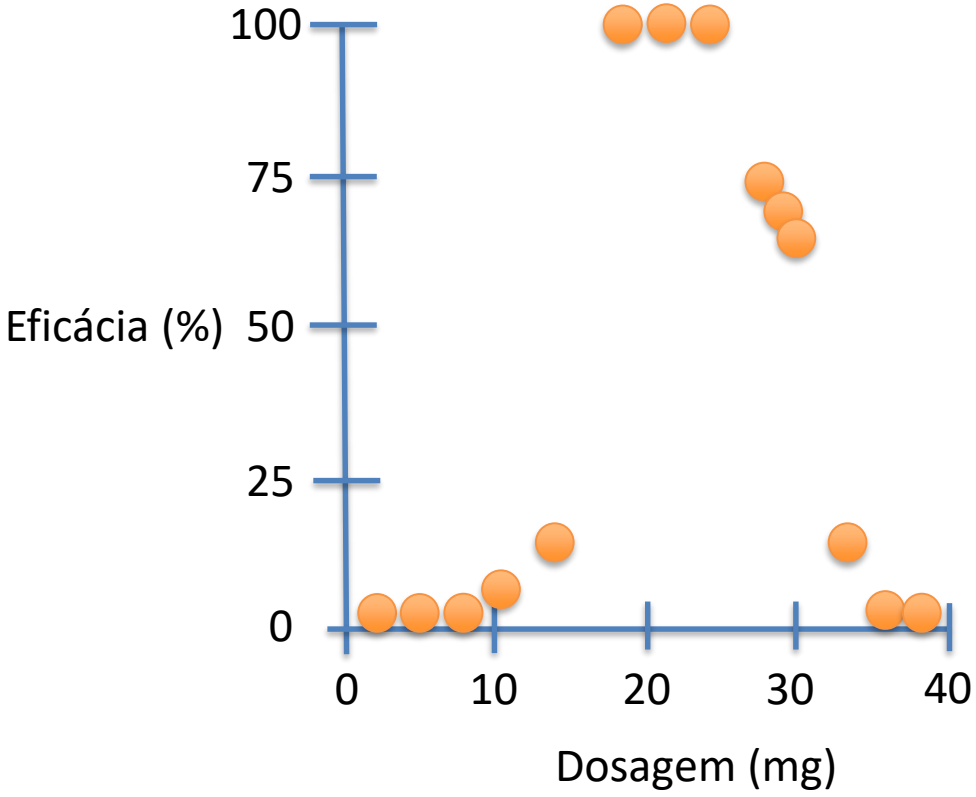
Regression Trees – Como constuir uma Regression Tree?



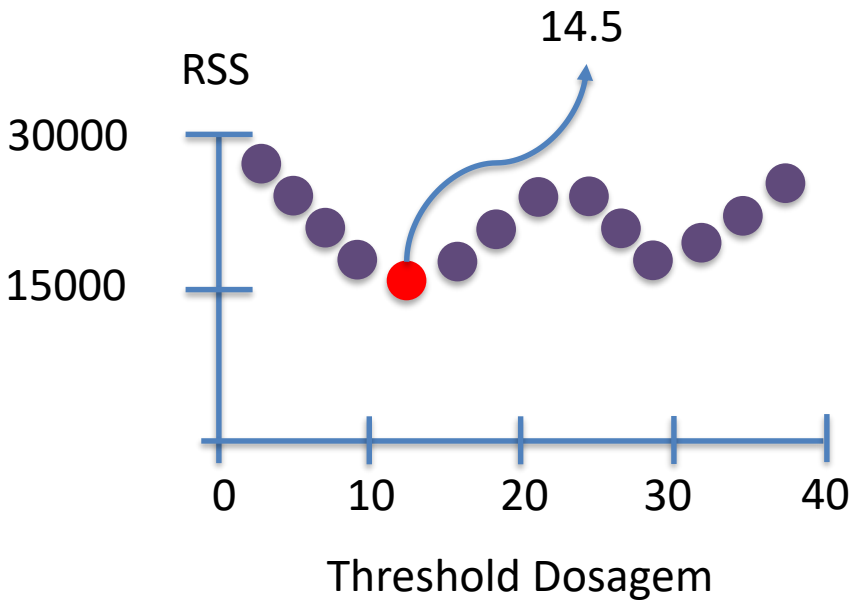
Repetindo o processo para todos os pontos, podemos encontrar o threshold que retorna o menor valor de RSS



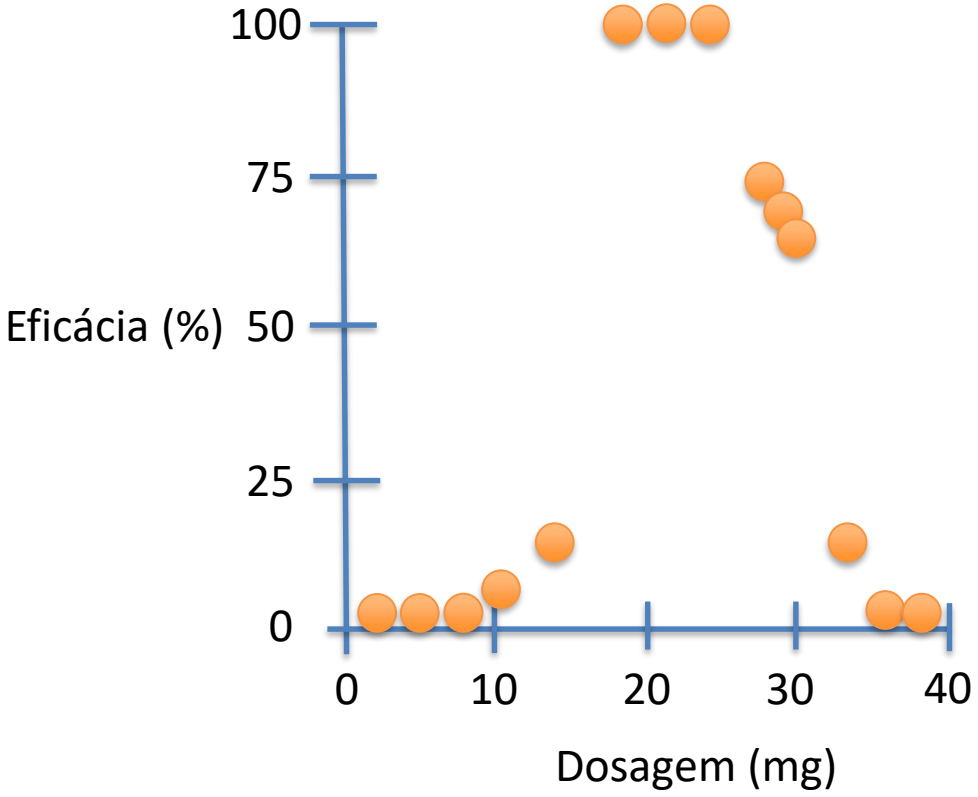
Regression Trees – Como constuir uma Regression Tree?

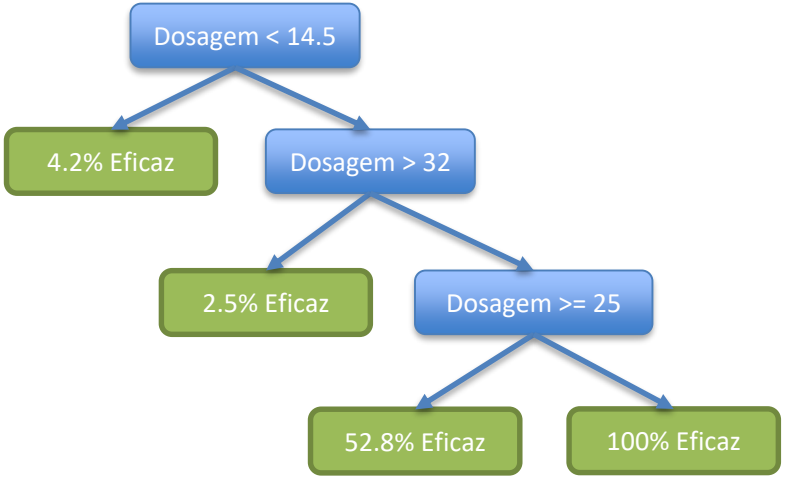


O que nos leva a árvore inicial



Regression Trees - Exemplo



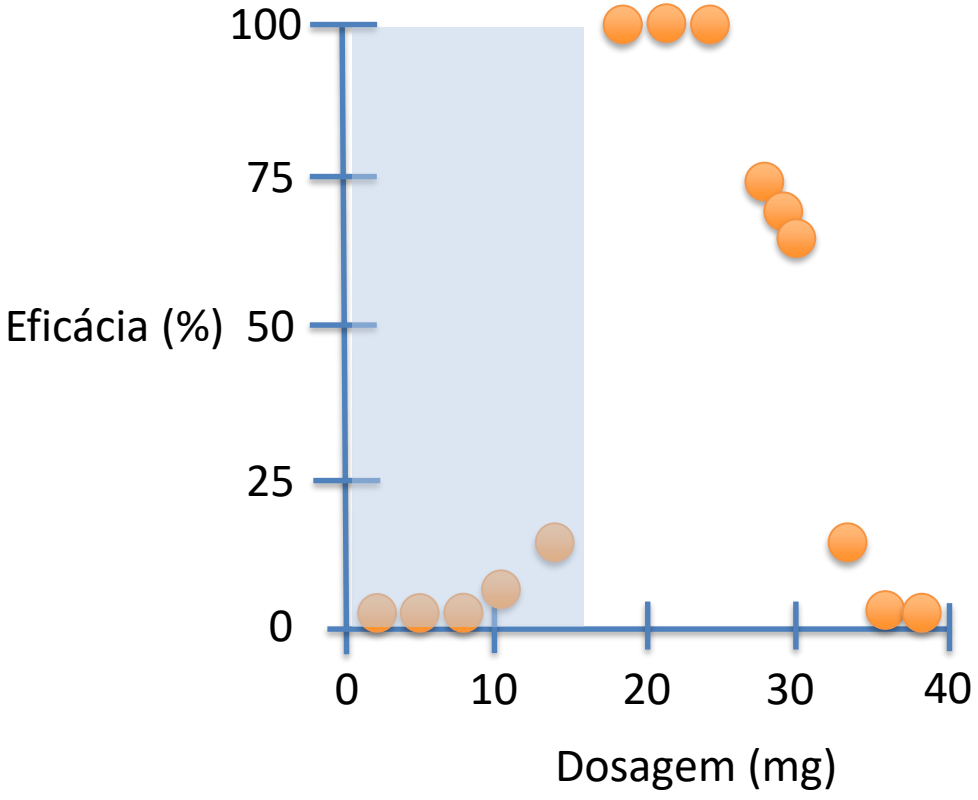


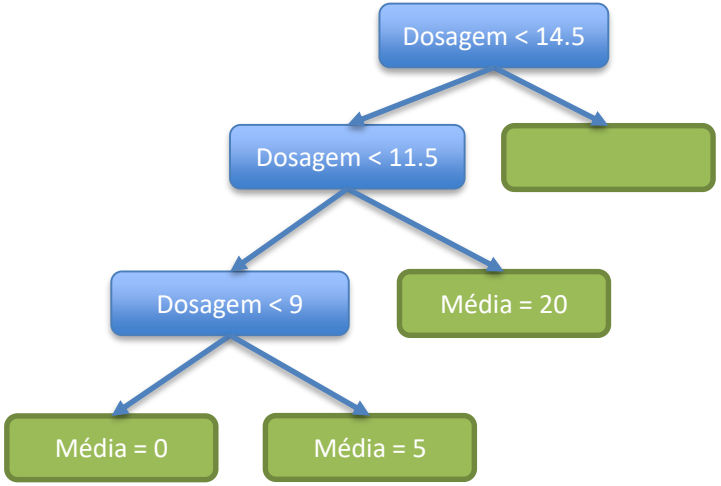
- Em resumo, dividimos os dados em dois grupos encontrando o valor de threshold que nos retorna o menor valor de RSS

- Um ponto importante a se observar ao realizar o split é se há um número mínimo de amostras em cada divisão.

- Focando nas amostras demarcadas, poderíamos criar a seguinte árvore:

Regression Trees - Overfitting





Importante notar que todas as predições são perfeitas. Isso é bom?

- Geralmente, quando existe um fit perfeito no conjunto de dados, há overfitting, isto é, o modelo torna-se especialista no conjunto de treino, mas não sabe generalizar para amostras ainda não vistas

- Bias/Variance tradeoff

- Assim, uma maneira de prevenir esse erro é dividir apenas quando existe mais observações do que um número pré-determinado.

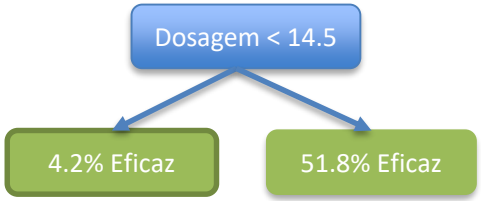
- 20, geralmente, é um bom número para se iniciar, mas é importante testar outros números.

Regression Trees – 2 ou mais Predictors

- Até agora, usamos apenas um preditor (característica) para prever a dosagem.

- Vamos agora entender como construir uma árvore de regressão usando mais de 1 preditor.

Regression Trees - 2 ou mais Predictors

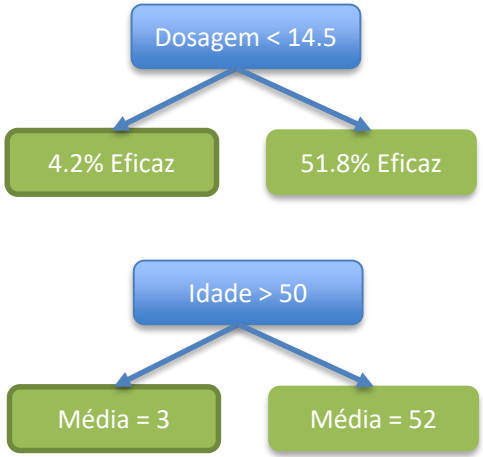


Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Escolhemos, para esse preditor, o valor que retorna o menor RSS

Iniciamos avaliando cada preditor como um candidato a raiz de acordo com o valor de RSS obtido. Repetimos o processo que vimos para construir a árvore para cada preditor.

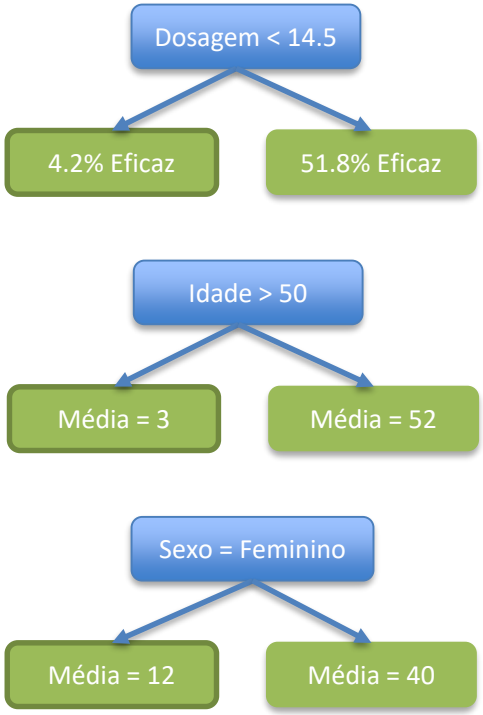
Regression Trees - 2 ou mais Predictors



Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Repetimos o processo para a variável Idade e escolhemos o threshold que retorna o menor valor de RSS

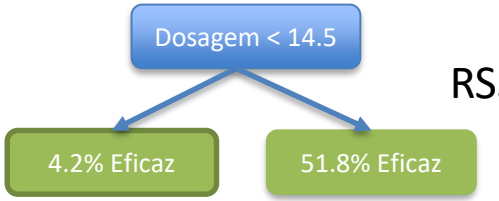
Regression Trees - 2 ou mais Predictors



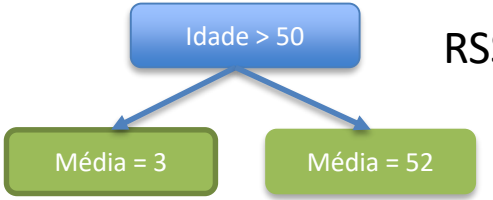
Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Fazemos o mesmo para a variável Sexo

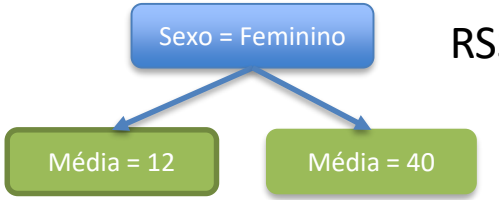
Regression Trees - 2 ou mais Predictors



RSS = 19564



RSS = 12017

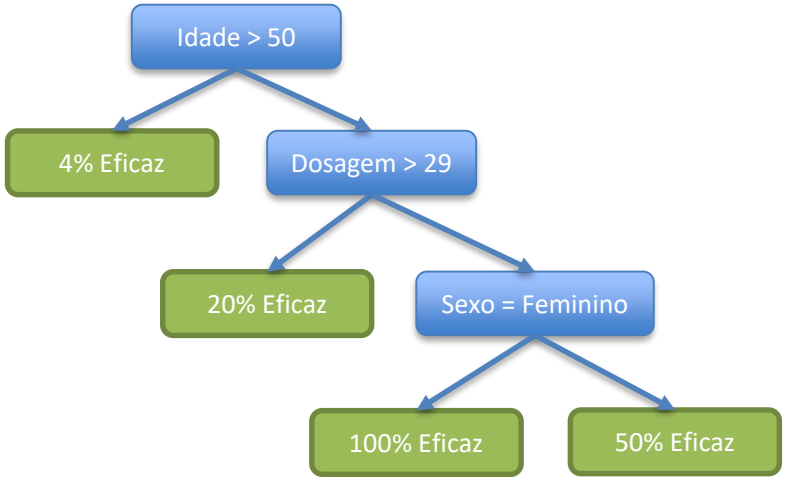


RSS = 20738

Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Agora, comparamos o valor de RSS de cada candidato e escolhemos o menor

Regression Trees - Exemplo



Dosagem	Idade	Sexo	Eficácia
10	25	Feminino	98
20	73	Masculino	0
35	54	Feminino	100
5	12	Masculino	44
...	...	...	...

Agora, tomamos o candidato com menor valor de RSS e usamos como raiz e construímos a árvore levando em conta os valores de RSS, em ordem crescente.

Implementação

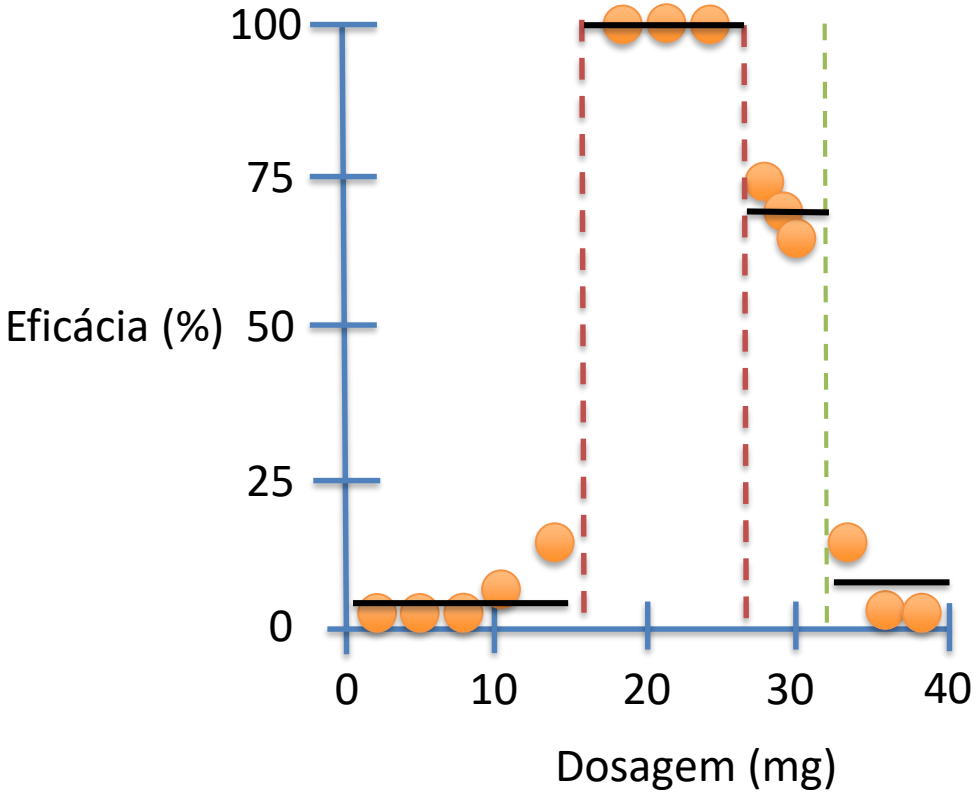
Pruning

Pruning

Overfitting pode ser novamente um problema quando vamos predizer um conjunto de teste. Isso significa que existe um erro muito pequeno na etapa de treinamento e um erro muito grande na etapa de teste.

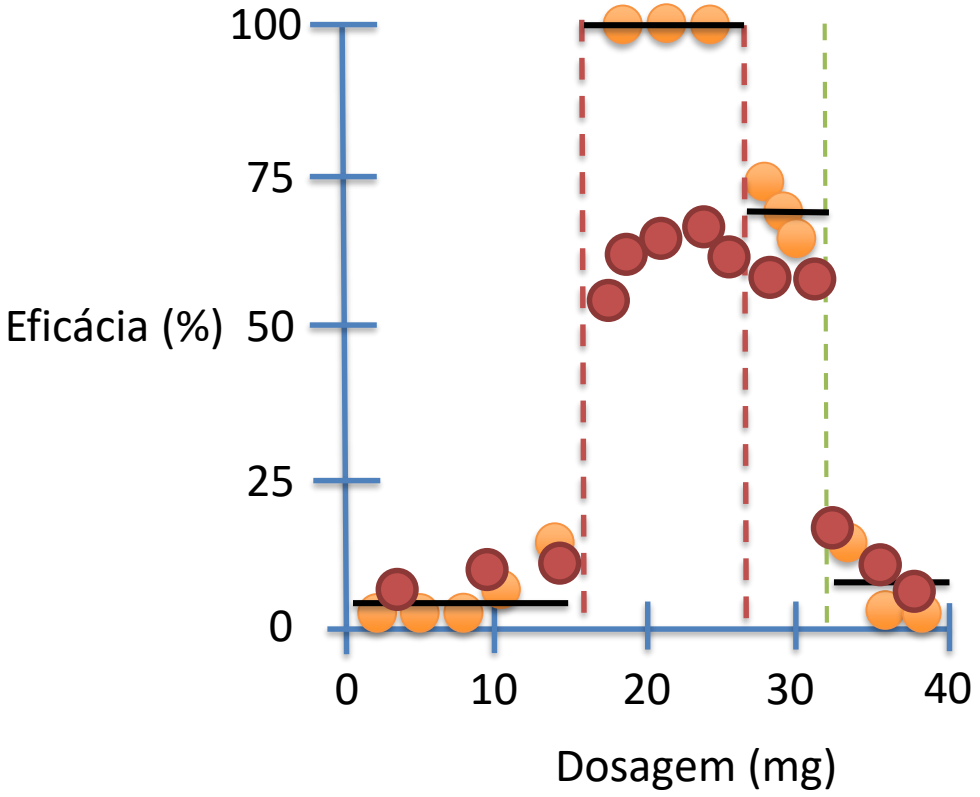
Vamos analisar o conjunto de treino novamente:

Pruning



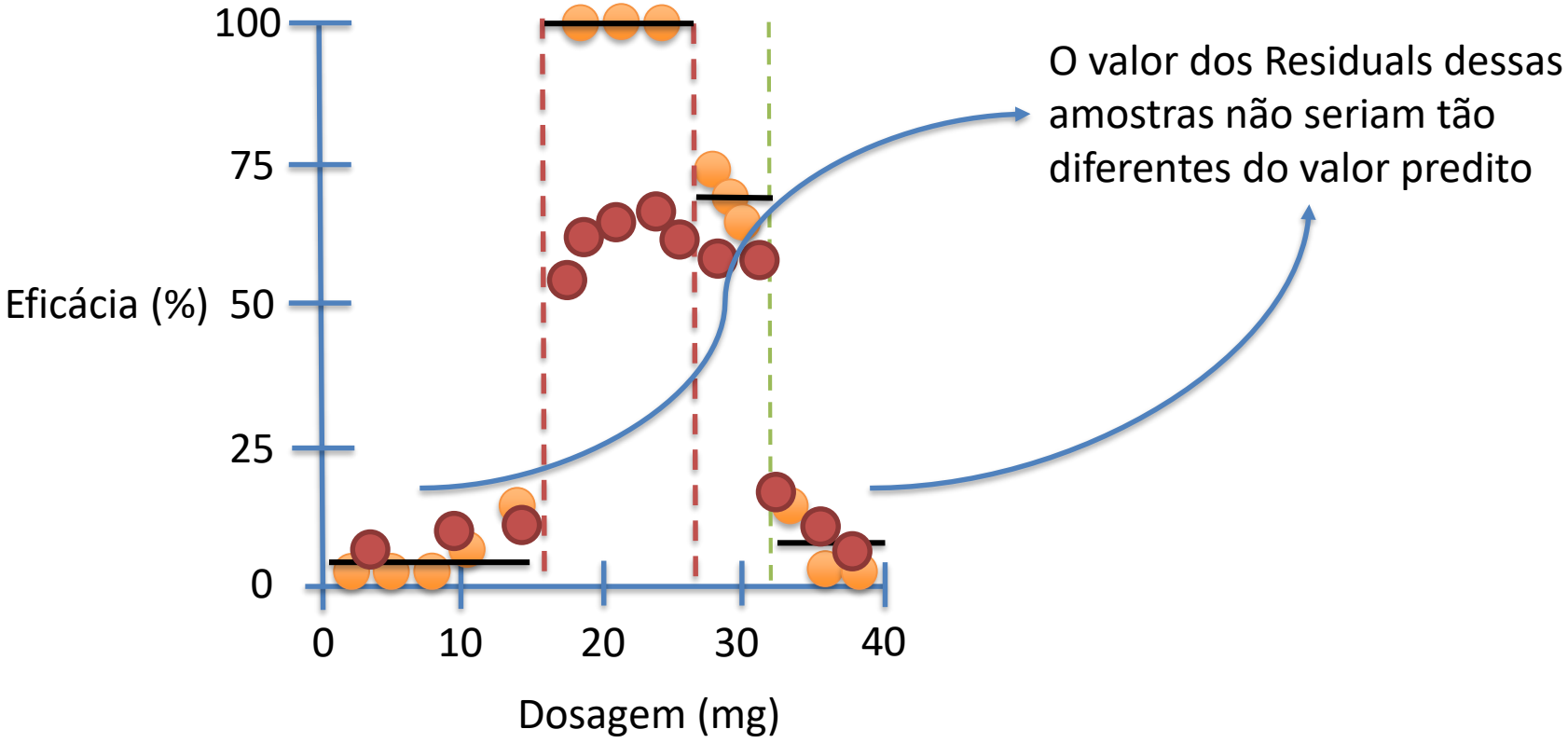
Dados de treinamento com a região que representa a árvore e seus respectivos valores médios de eficácia.

Pruning

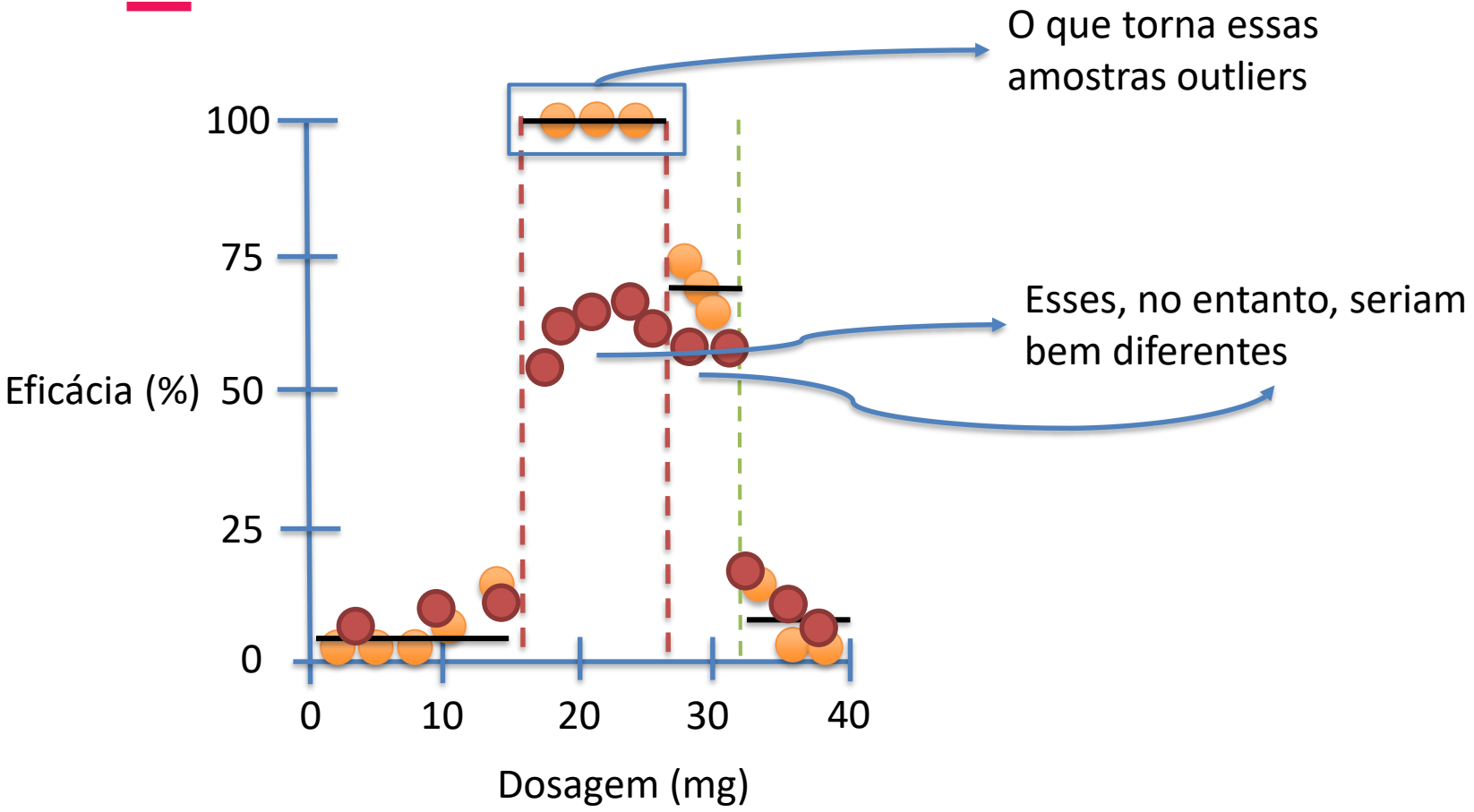


O que aconteceria se esses fossem o conjunto de teste?

Pruning

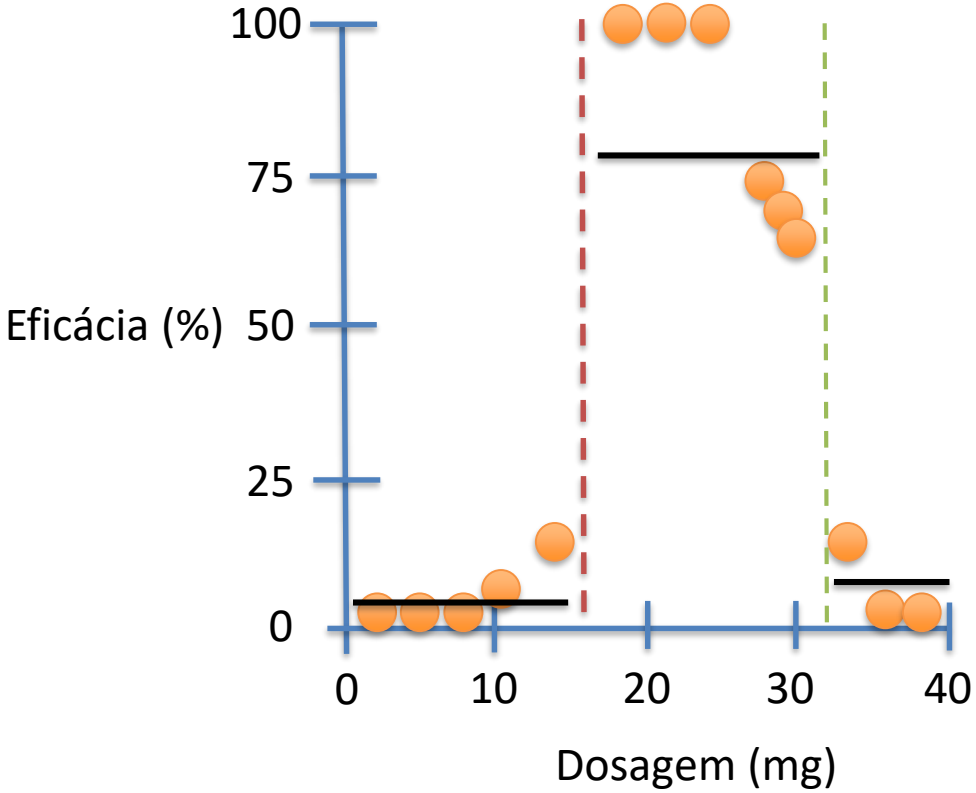

100
75
50
25
0
Eficácia (%)
0
10
20
30
40
Dosagem (mg)
O valor dos Residuals dessas amostras não seriam tão diferentes do valor predito

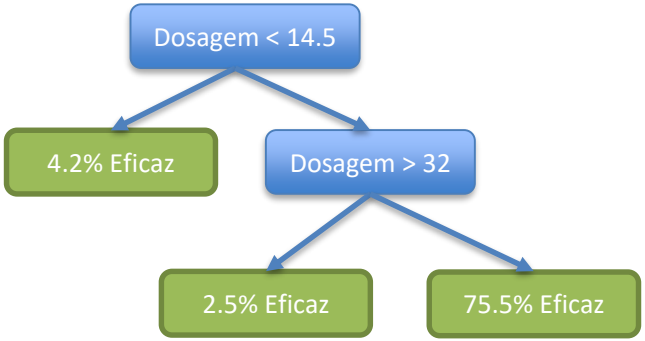
Pruning


O que torna essas amostras outliers
Esses, no entanto, seriam bem diferentes
Eficácia (%)
100
75
50
25
0
0
10
20
30
40
Dosagem (mg)

Pruning

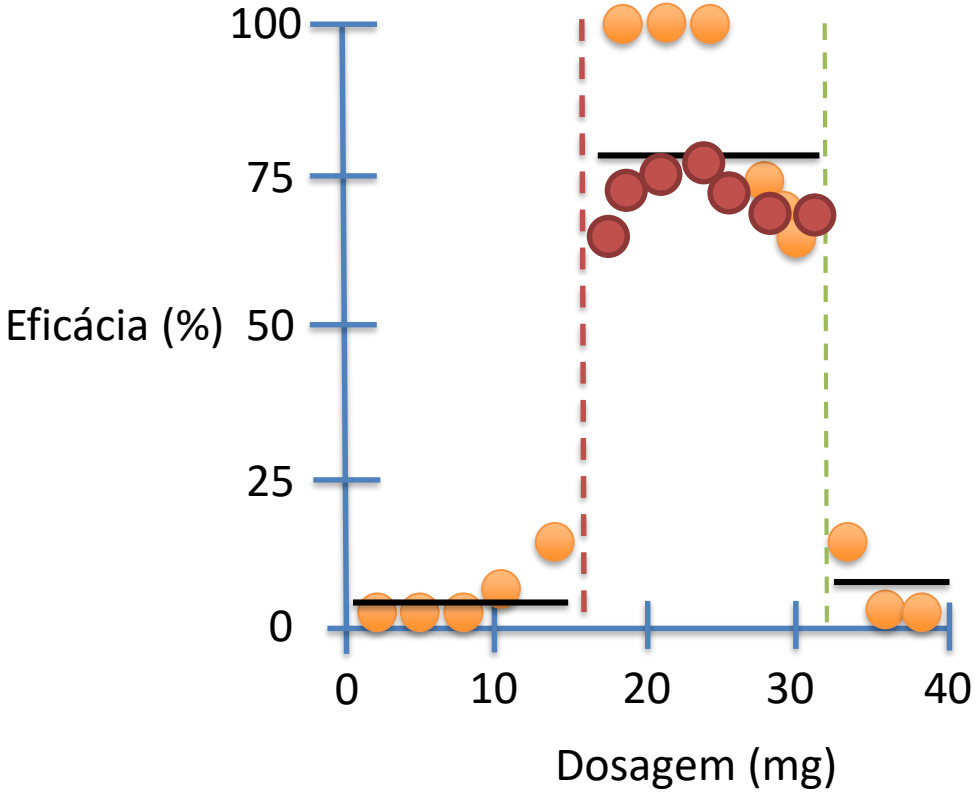
Uma maneira de eliminar overfitting é remover algumas das folhas

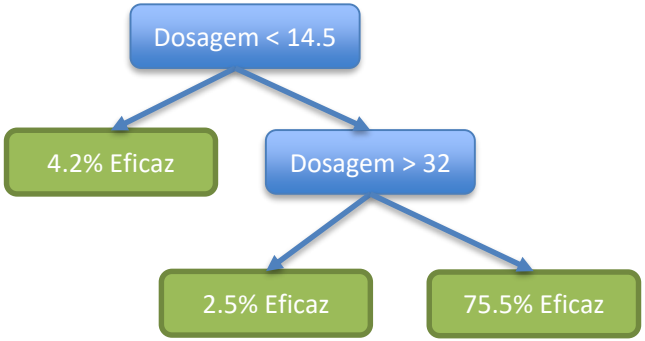




Pruning

Uma maneira de eliminar overfitting é remover algumas das folhas

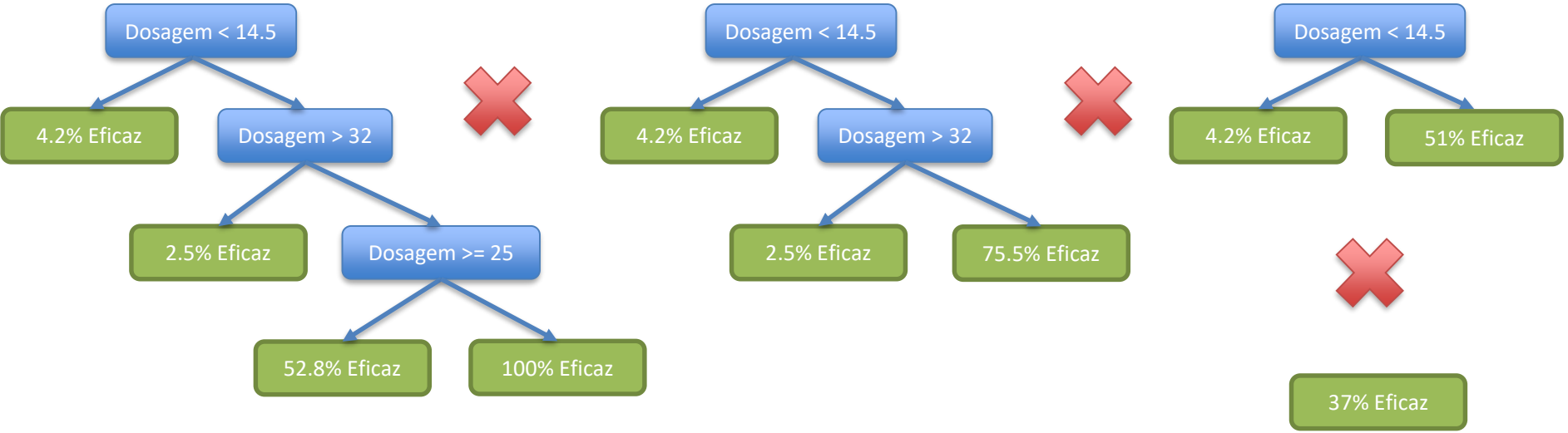




Isso aumenta um pouco os Residuals na etapa de treino, mas essa nova sub-árvore oferece um resultado melhor para os dados de teste

Pruning

Podemos continuar removendo folhas até permanecer apenas a raiz. Assim, a decisão consiste em determinar qual a melhor árvore a ser utilizada



Pruning

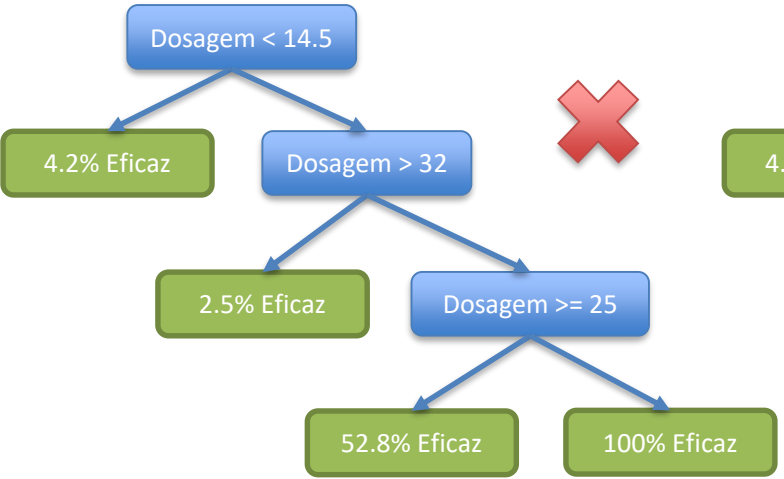
- Para responder essa questão, usaremos Cost Complexity Pruning, um algoritmo parametrizado por $\alpha \geq 0$, conhecido como parâmetro de complexidade, que é usado para mensurar $R_\alpha(T)$ (cost complexity) para uma dada árvore T.

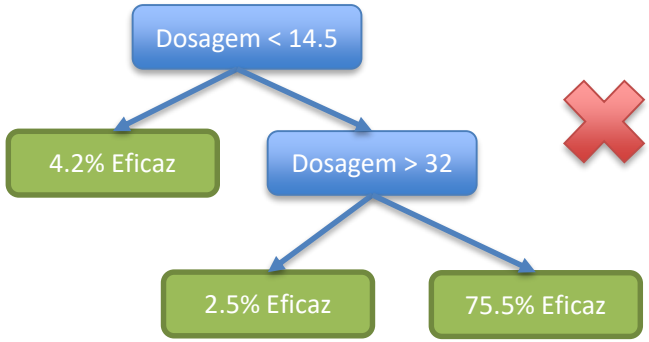
- $R_\alpha(T) = R(T) + \alpha\left|\tilde{T}\right|$

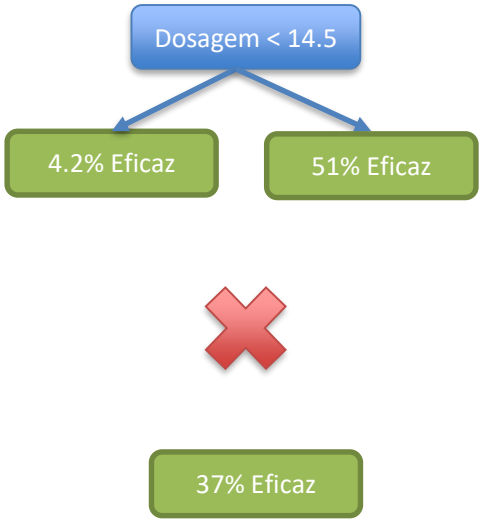
- Em que, $\left|\tilde{T}\right|$ é o número de nós terminais (folhas) e $R(T)$ é a taxa de erro de predição total dos nós terminais (RSS, em nosso caso)

Pruning

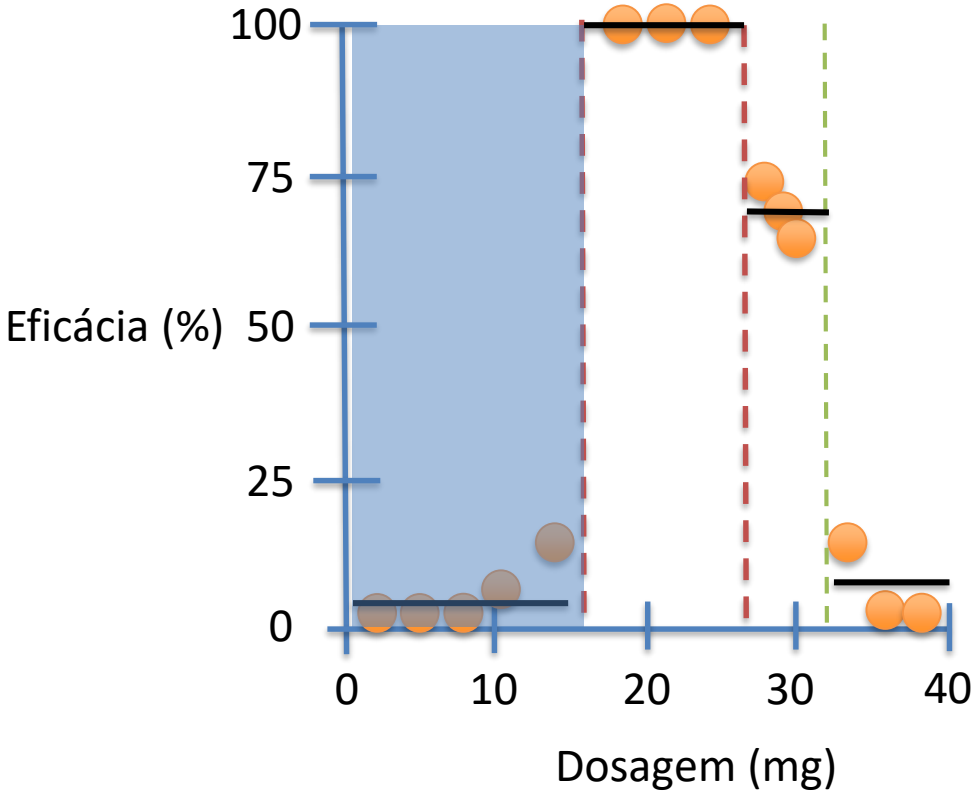
O primeiro passo é calcular o RSS de cada árvore:

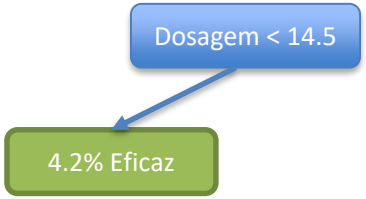






Pruning





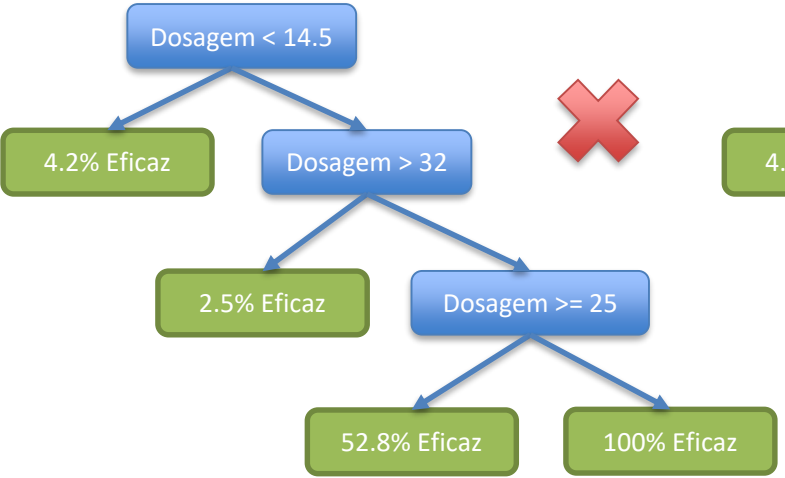
$$(0-4.2)^2+(0-4.2)^2+(0-4.2)^2+(5-4.2)^2+(20-4.2)^2=303.2$$

Depois, eu repito o processo para cada parte e somo o resultado. Em seguida, o processo é feito para cada árvore.

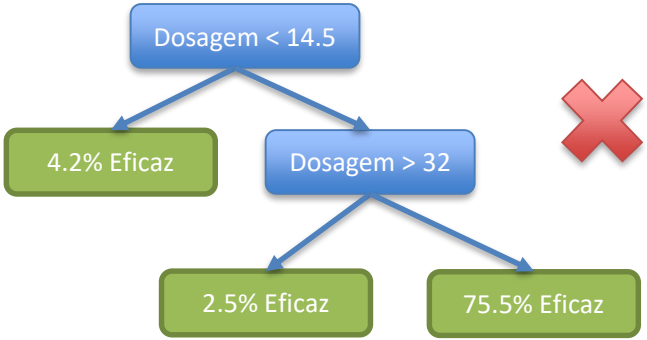
Pruning

O primeiro passo é calcular o RSS de cada árvore:

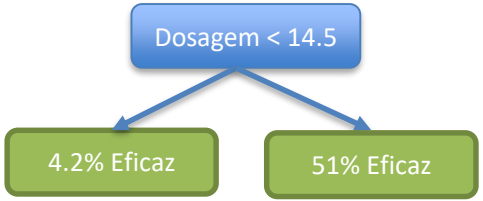
RSS = 504.6



RSS = 5215.8



RSS = 19156.7

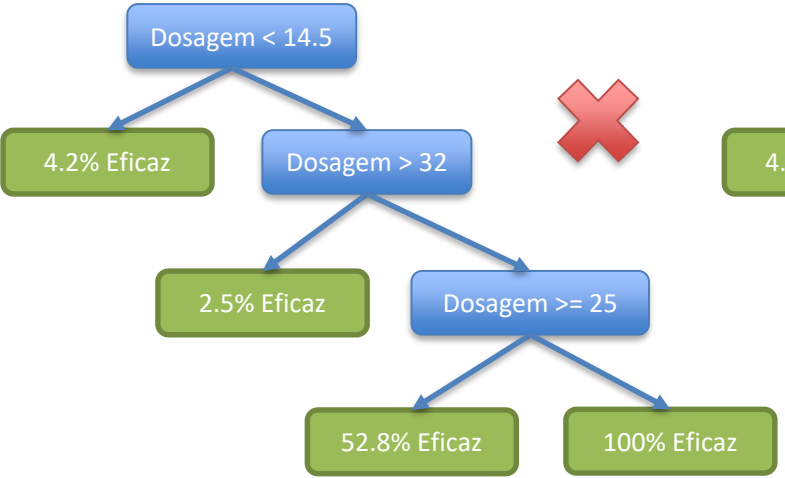


RSS = 28563.4

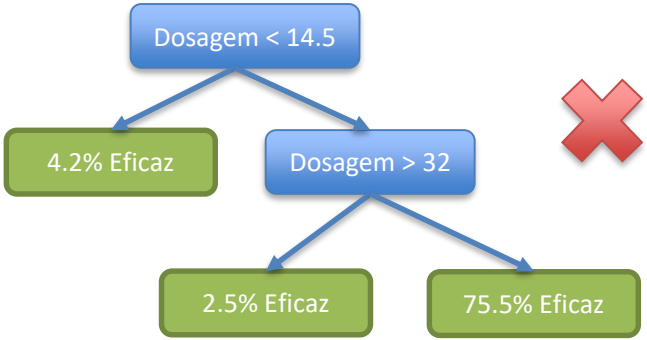
Pruning

Agora, calculamos o parâmetro $\alpha|\tilde{T}|$ para determinar o Score de cada árvore ($R_\alpha(T)$)

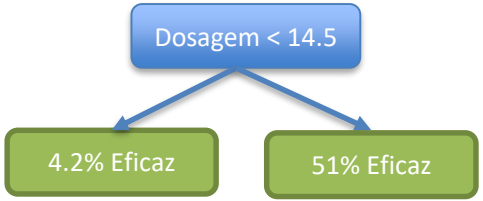
RSS = 504.6



RSS = 5215.8



RSS = 19156.7

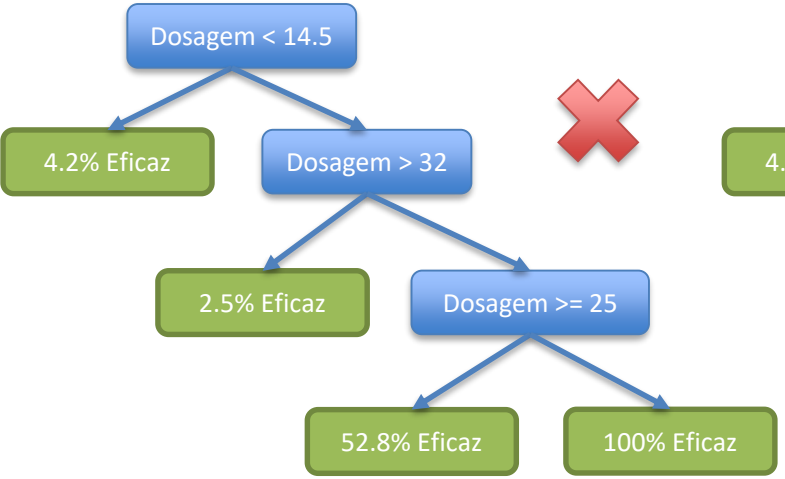


RSS = 28563.4

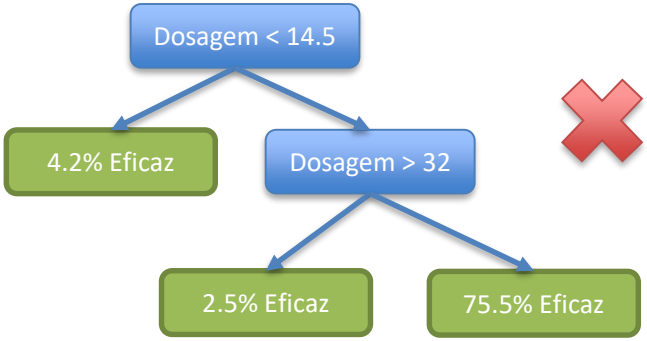
Pruning

O parametro α serve para ponderar a diferença no número de folhas entre as árvores. Podemos usar cross-validation ou separar o dataset em conjunto de treino e teste para determinar o valor de α que maximiza a acurácia no conjunto de teste

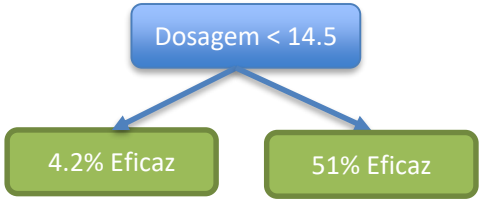
RSS = 504.6



RSS = 5215.8



RSS = 19156.7

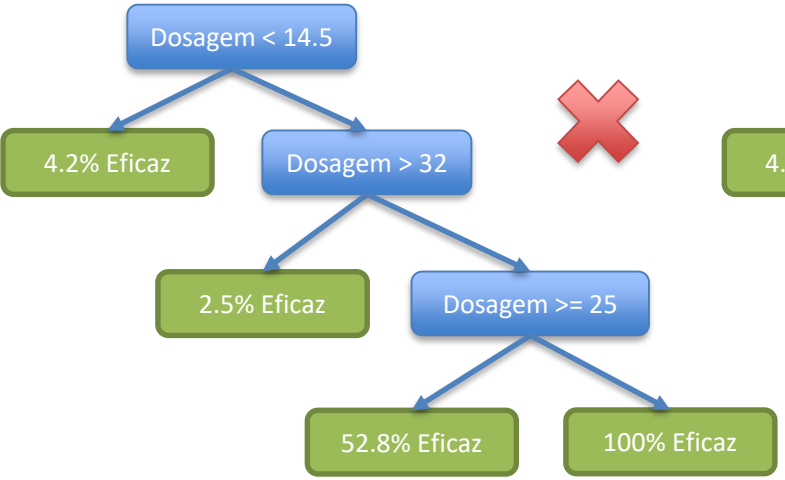


RSS = 28563.4

Pruning

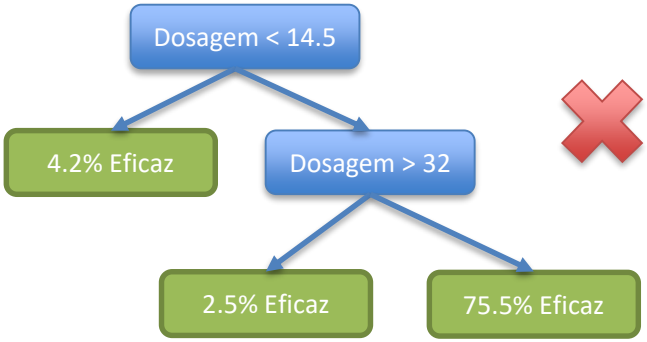
A título de exemplo, tomemos $\alpha = 10000$

RSS = 504.6



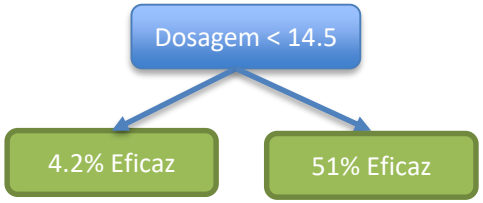
$$R_\alpha(T) = R(T) + \alpha|\tilde{T}| = 504.6 + 10000 \times 4 = 40504.6$$

RSS = 5215.8



$$R_\alpha(T) = 34898.2$$

RSS = 19156.7



$$R_\alpha(T) = 39052.2$$

RSS = 28563.4

$$R_\alpha(T) = 37443.9$$

Pruning

- Podemos entender que $\boldsymbol{R_\alpha(T)}$ depende do valor de α. Assim, é importante métodos que escolham seu valor de maneira adequada.

- O código no jupyter notebook nos fornece uma ideia de como fazer isso.

Implementação

Obrigado!

profdheny.fernandes@fiap.com.br

/dhenyfernandes


FIAP