Preditiva.ai

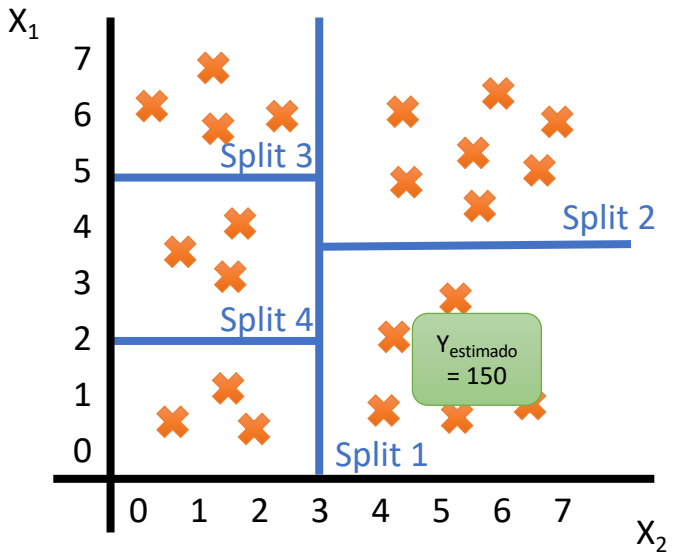
Aprendizado Supervisionado
Árvores de Decisão para Regressão

Árvores de Decisão

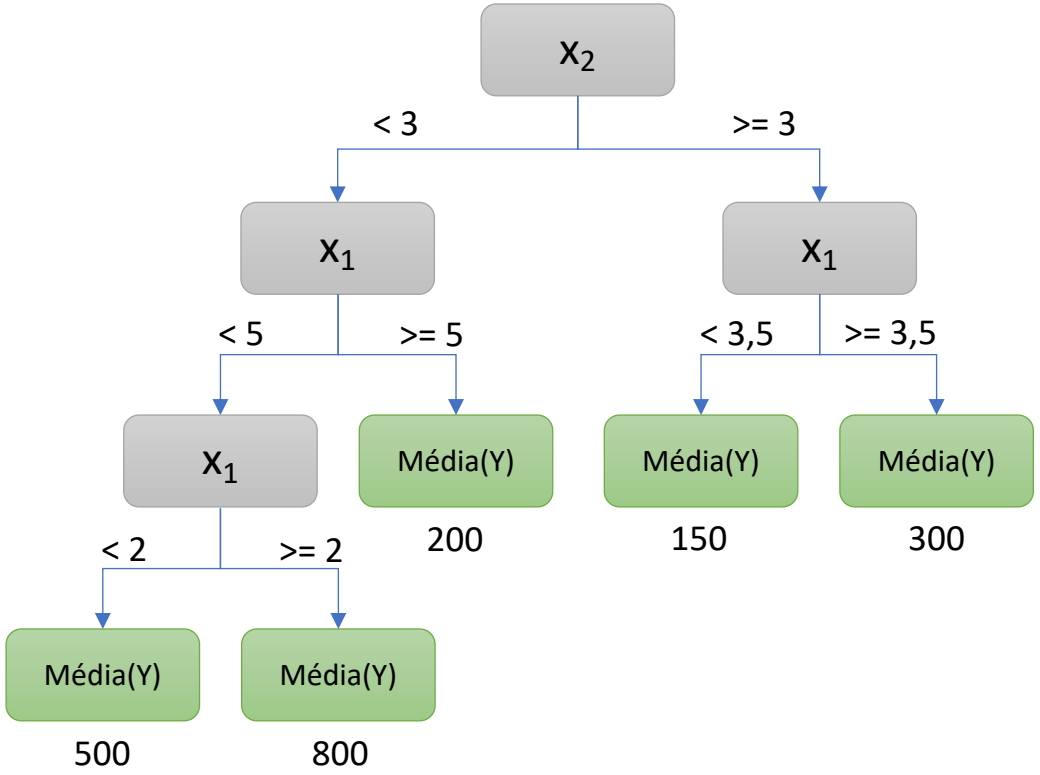
Intuição

Preditiva.ai

Outro tipo de método muito versátil usado para explicar variáveis quantitativas (e qualitativas) é a **Árvore de Decisão**. Neste método, as variáveis explicativas passam por sucessivos **splits** (cortes) de forma a chegarmos em boas estimativas da variável resposta. Veja um exemplo:



Sendo x_1 e x_2 as variáveis explicativas.

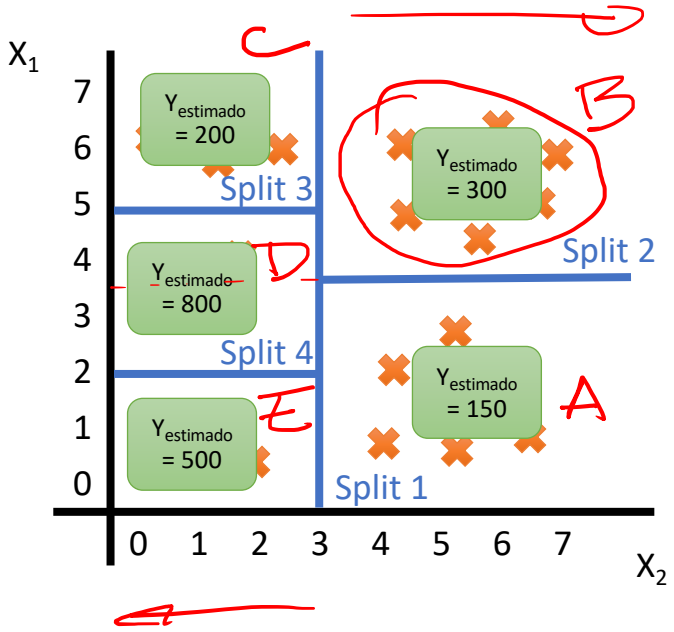


Árvores de Decisão

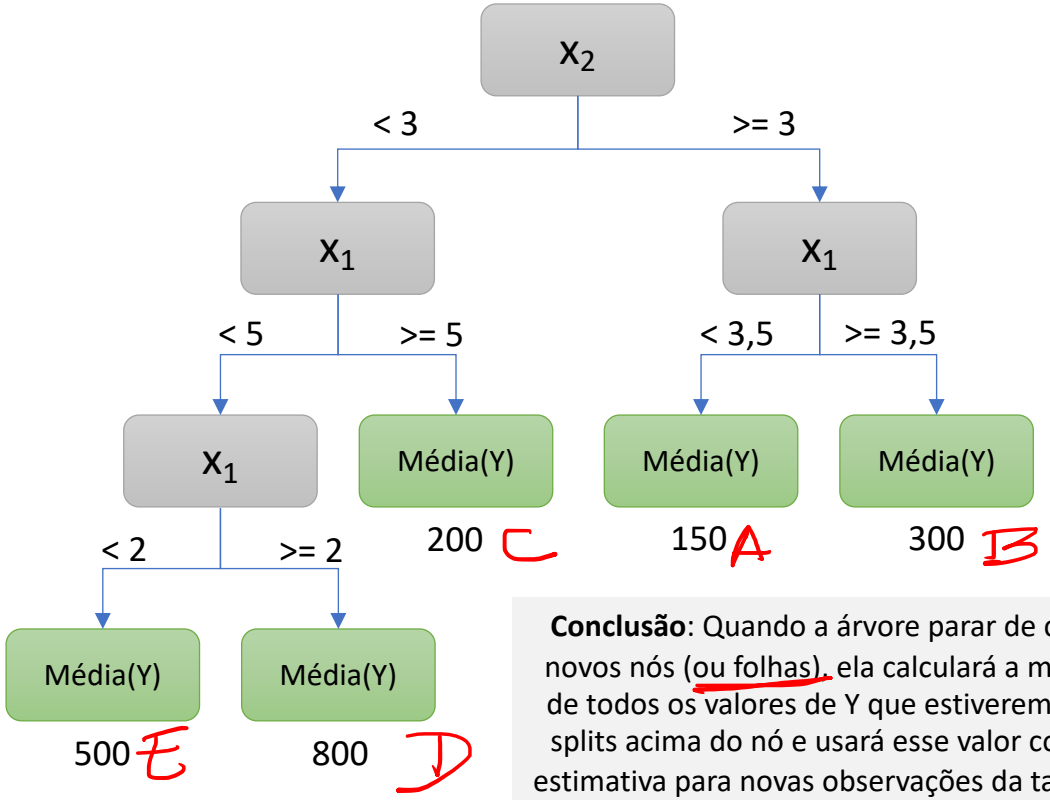
Intuição

Preditiva.ai

Outro tipo de método muito versátil usado para explicar variáveis quantitativas (e qualitativas) é a **Árvore de Decisão**. Neste método, as variáveis explicativas passam por sucessíveis **splits** (cortes) de forma a chegarmos em boas estimativas da variável resposta. Veja um exemplo:



Sendo x_1 e x_2 as variáveis explicativas.



Conclusão: Quando a árvore parar de criar novos nós (ou folhas), ela calculará a média de todos os valores de Y que estiverem nos splits acima do nó e usará esse valor como estimativa para novas observações da tabela.

Árvores de Decisão

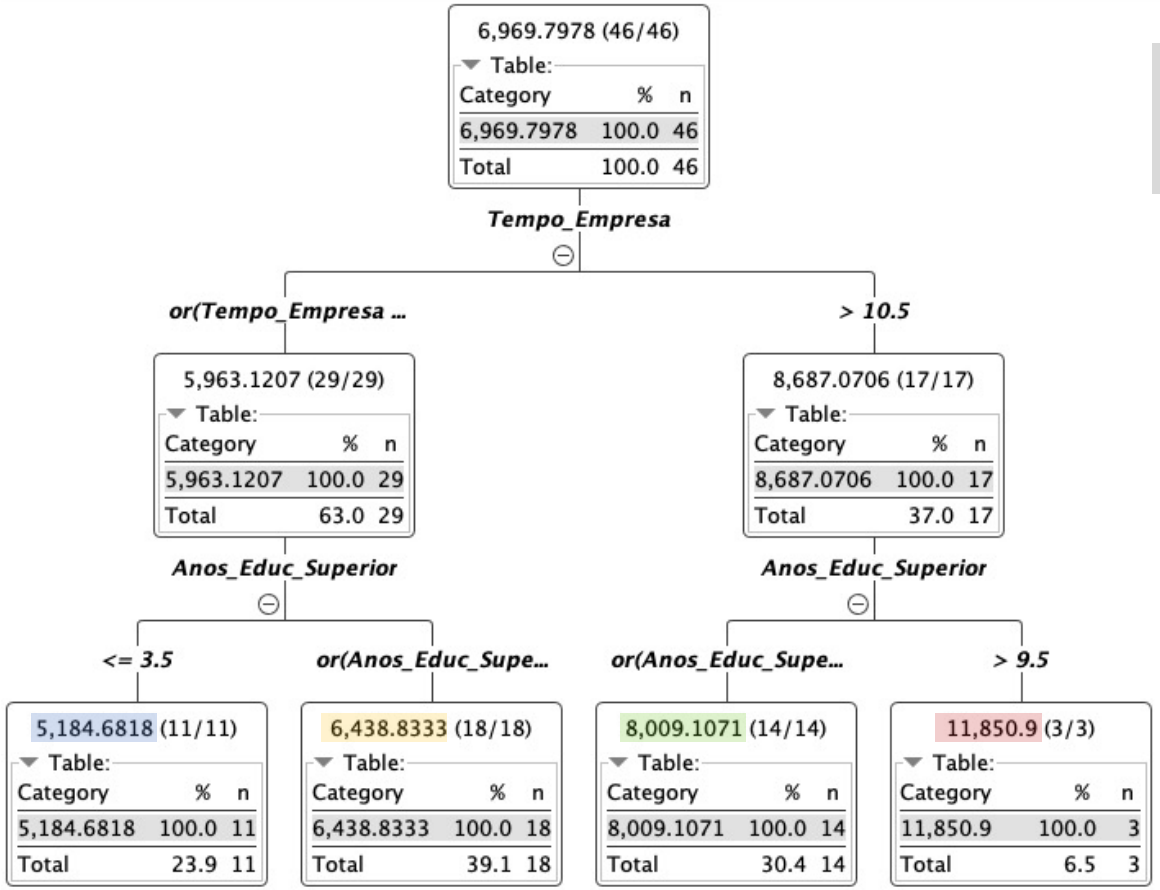
Exemplo

Preditiva.ai

Vejamos um exemplo com a base de funcionários já utilizada anteriormente:

Exemplo dos primeiros 15 funcionários

Salario (Y)	Anos de Educação Superior (x_1)	Tempo na Empresa (x_2)
5.517,4	3	3
6.399,9	4	6
6.206,7	6	3
6.060,6	4	5
6.122,7	2	9
6.955,0	5	9
7.643,0	4	6
6.210,2	2	8
5.761,0	9	15
8.086,9	6	14
6.375,4	4	9
9.568,8	6	20
9.316,0	6	25
6.822,4	9	18
6.570,9	4	19



Salario (Y)	Anos de Educação Superior (x_1)	Tempo na Empresa (x_2)	Salário Estimado pelo modelo
5.517,40	3	3	5.184,68
6.399,90	4	6	6.438,83
6.206,70	6	3	6.438,83
6.060,60	4	5	6.438,83
6.122,70	2	9	5.184,68
6.955,00	5	9	6.438,83
7.643,00	4	6	6.438,83
6.210,20	2	8	5.184,68
5.761,00	9	15	8.009,11
8.086,90	6	14	8.009,11
6.375,40	4	9	6.438,83
9.568,80	6	20	8.009,11
9.316,00	6	25	8.009,11
6.822,40	9	18	8.009,11
6.570,90	4	19	8.009,11

R Quadrado: 0.706

Árvores de Decisão

Algoritmo

Preditiva.ai

Para chegar aos melhores splits, os pacotes geralmente usam um algoritmo chamado **CART** (Classification And Regression Tree), que consiste em minimizar uma função de erros a cada nó, até que algum critério de parada seja alcançado. Veja com mais detalhes abaixo:

Algoritmo CART para Regressão

1. Selecione um par (**x** , **s**) sendo **x** uma das variáveis da base e **s** um corte desta variável;
2. Calcule a função de erro conforme abaixo:

   - $$F\ (x, s) = \frac{qte\ de\ linhas_{nó\ da\ esquerda}}{qte\ de\ linhas\ do\ nó\ superior} * MSE_{nó\ da\ esquerda} + \frac{qte\ de\ linhas_{nó\ da\ direita}}{qte\ de\ linhas\ do\ nó\ superior} * MSE_{nó\ da\ direita}$$

   , sendo MSE o Mean Square Error (Média dos Erros ao Quadrado)

3. Selecione o par (**x** , **s**) que minimize a função de erros acima;
4. Use esse par como Split e repita o processo até não ser possível mais separar a base ou que algum critério de parada seja atingido (ex: profundidade da árvore (qte de splits), qte de nós, qte de linhas por nó etc).

Preditiva.ai

Demonstração

Árvore de Regressão no Python

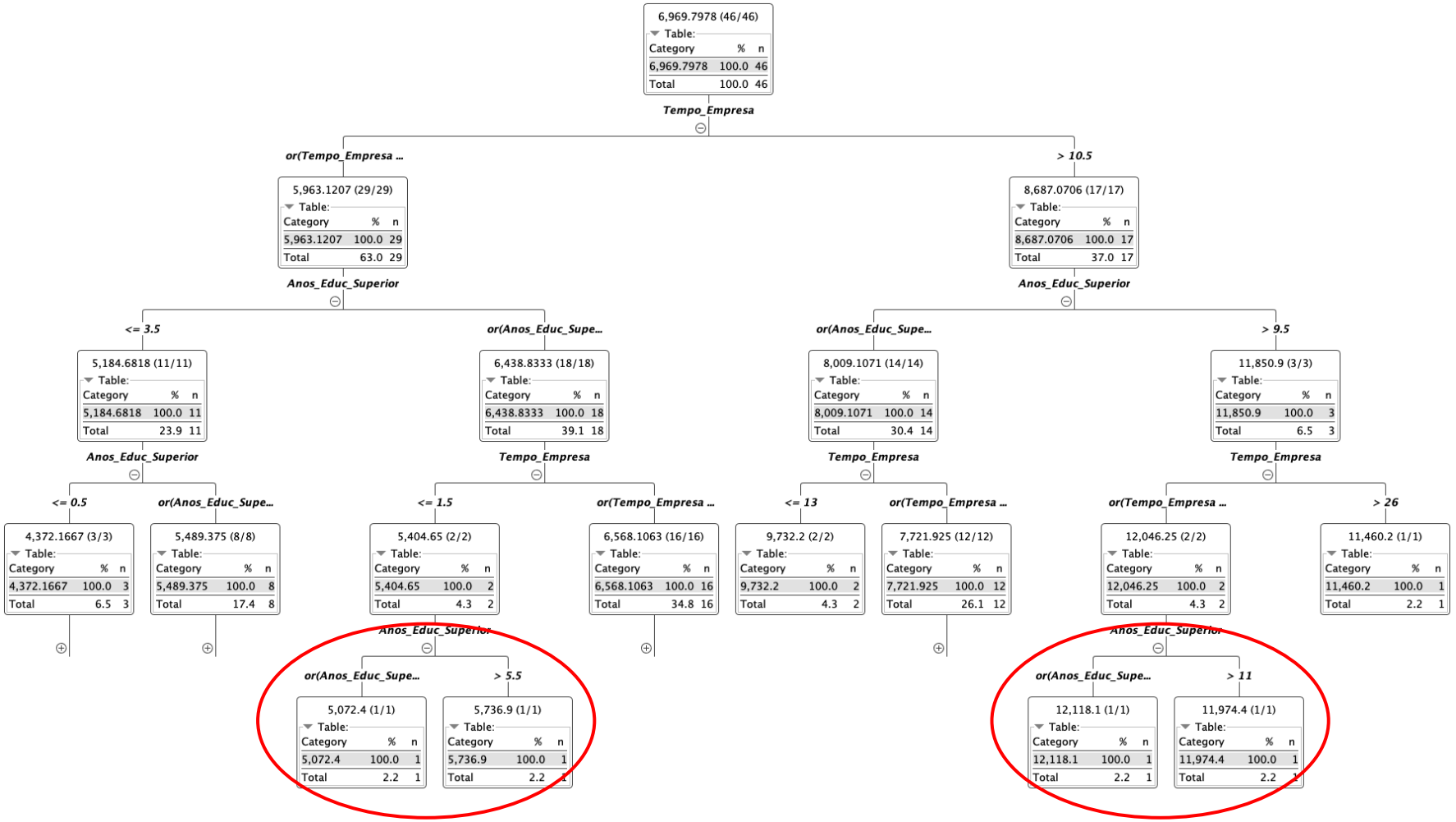
Arquivo: 3_1_Arvores_de_Decisao.ipynb

Árvores de Decisão

Necessidade das podas

Preditiva.ai

Uma característica da árvore de decisão é que ela pode explicar “totalmente” os dados, ou seja, ela “aprende” muito com os dados disponíveis fazendo splits até ficar com apenas uma observação em cada nó. Isso é ruim, pois em geral queremos modelos que “generalizam” seu aprendizado para outras bases. Veja:



R Quadrado: 0.99

Árvores de Decisão

Necessidade das podas

Preditiva.ai

Para prevenir esse comportamento ganancioso “greedy” do algoritmo CART, precisamos realizar “podas”. Ou seja, parametrizamos critérios de parada para que a árvore se mantenha em patameres melhores e úteis em um modelo que generaliza bem para outras bases de dados não utilizadas para seu treinamento. Veja algumas formas de poda:

1. Profundidade máxima: Parâmetro que fixa a quantidade de níveis que a árvore pode chegar.

2. Mínimo de amostras por nó: Parâmetro que fixa a quantidade mínima que um nó deve ter para se fazer um Split.

3. Mínimo de amostras por folha (leaf): Parâmetro que fixa a quantidade mínima que uma folha (ou seja, um nó que não tem outros nós abaixo dele) deve ter para ser criada. No exemplo anterior, se o número mínimo fosse dois, as folhas com apenas uma amostra não seriam criados.

Preditiva.ai

Aprendizado Supervisionado
Árvores de Decisão para Classificação

Árvores de Decisão

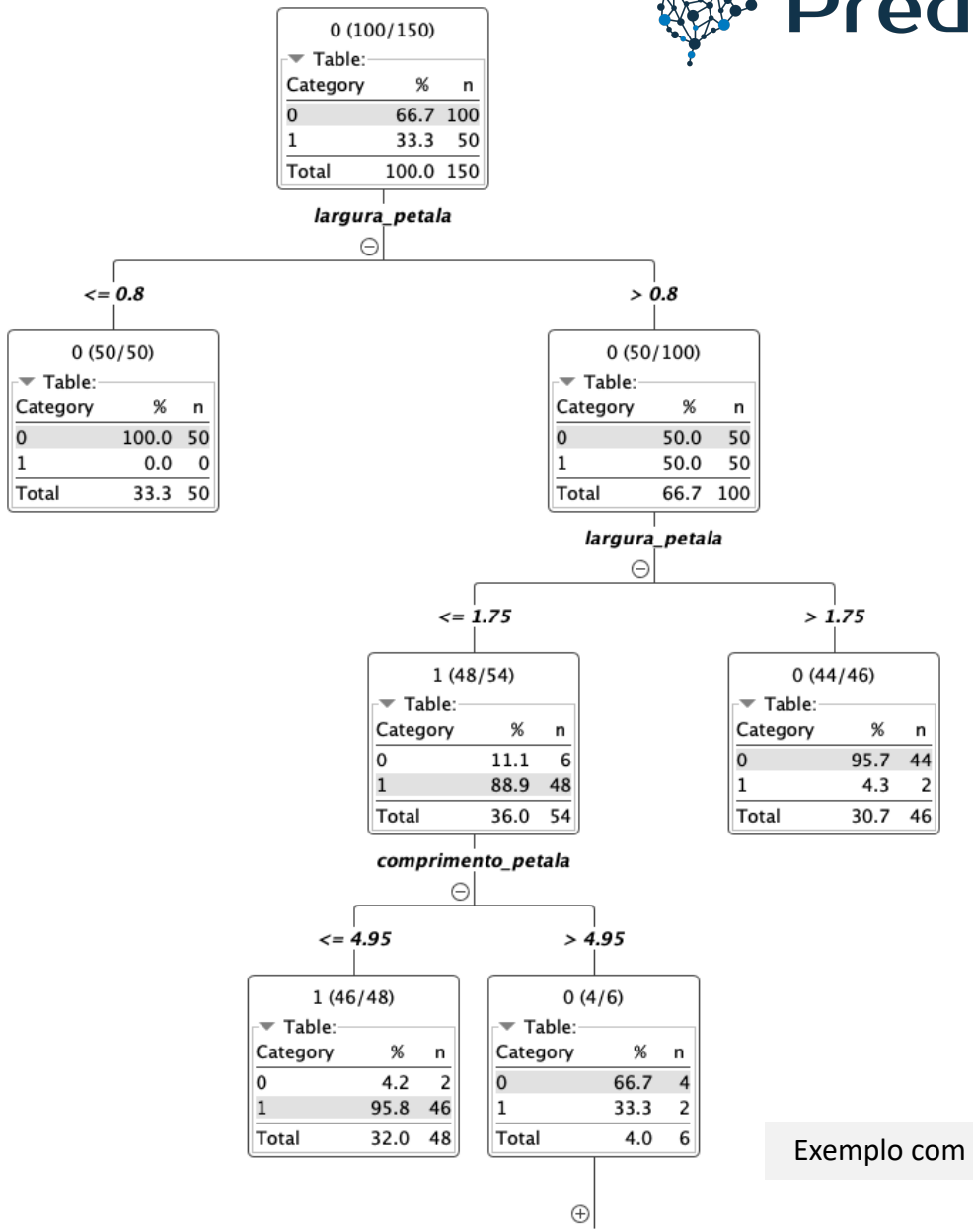
Exemplo para Classificação

Preditiva.ai

Árvores de decisão para problemas de **classificação** seguem o mesmo princípio das árvores para regressão. A cada corte, o algoritmo cria nós que tentam maximizar a separação entre as classes do target.

No exemplo, se **largura_pétala <= 0.8** cm, então nenhuma planta é do tipo orquídea, ou seja, y = 1 foi de 0%.

Por outro lado, se **largura_pétala > 0.8** e **largura_pétala <= 1.75** e **comprimento_pétala <= 4.95** então a planta pode ser classificada como orquídea com 95.8% de probabilidade.



Exemplo com a base **“iris2.csv”**

Árvores de Decisão
Algoritmo

Preditiva.ai

Algoritmo CART para Classificação

1. Selecione um par (x , s) sendo x uma das variáveis da base e s um corte desta variável;
2. Calcule a função de erro conforme abaixo:

   - $F\ (x, s) = \frac{qte\ de\ linhas_{nó\ da\ esquerda}}{qte\ de\ linhas\ do\ nó\ superior} * Impureza_{nó\ da\ esquerda} + \frac{qte\ de\ linhas_{nó\ da\ direita}}{qte\ de\ linhas\ do\ nó\ superior} * Impureza_{nó\ da\ direita}$

3. Selecione o par (x , s) que minimize a função de erros acima;
4. Use esse par como Split e repita o processo até não ser possível mais separar a base ou que algum critério de parada seja atingido (ex: profundidade da árvore (qte de splits), qte de nós, qte de linhas por nó etc).

Como medida de "impureza", podemos usar basicamente as medidas **GINI** e **Entropia**. Como praticamente as duas produzem os mesmos resultados, muitos pacotes acabam escolhendo a medida GINI por ser mais rápida em ser calculada. Veja seu cálculo:

$$Gini = 1 - \sum_{k=1}^{n} {p_{i,k}}^2$$, sendo que $p_{i,k}$ é a razão da classe "**k**" pelo total de observações do nó "**i**".

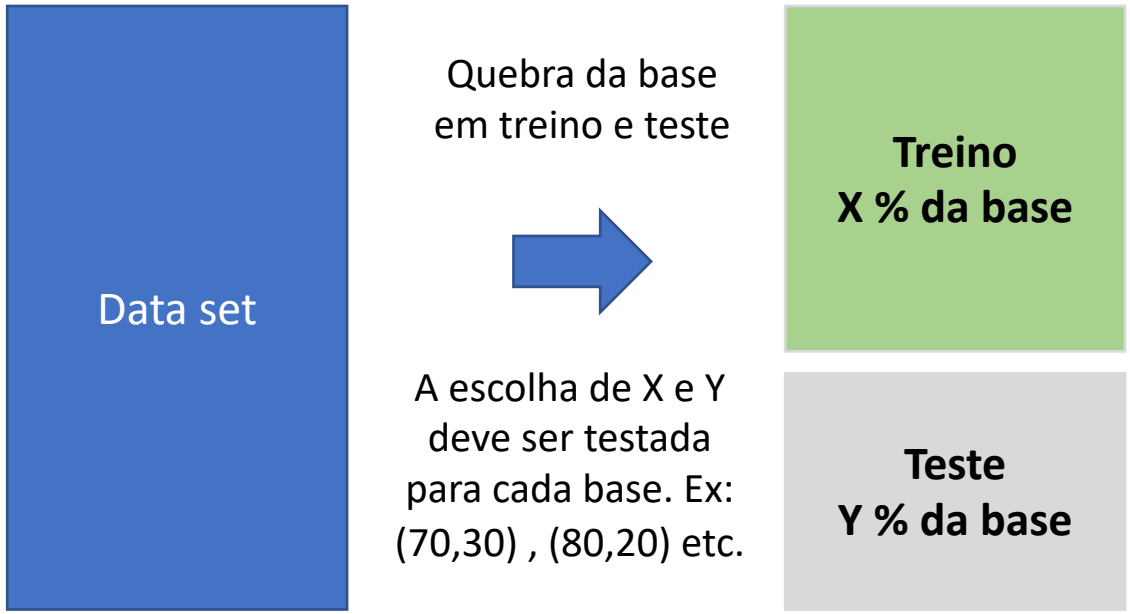
Árvores de Decisão

Modelos que não generalizam

Preditiva.ai

Quando desenvolvemos modelos descritivos de ajuste, estamos mais interessados em explicar os dados atuais, e não prever resultados futuros. Caso esta seja a necessidade, precisamos desenvolver modelos com a capacidade de ter bons resultados não só na base em que foram ajustados/treinados, mas em outras bases em que o modelo nunca observou. Esse conceito é chamado de **validação cruzada** e é essencial em Data Science.

Processo de Validação Cruzada



Usamos a base de treino para ajustar/treinar o modelo. Após isso calculamos alguma medida de performance. Ex: Acurácia.

Usamos a base de teste para predizer/escorar o modelo treinado. Após isso calculamos a mesma medida de performance e comparamos com a medida na base em treino.

Preditiva.ai