

Tera

Módulo 3:

Aprendizado de Máquina Supervisionado

Aula 15: Machine Learning com Árvores de Decisão

Let's play a game...

Regras do Jogo

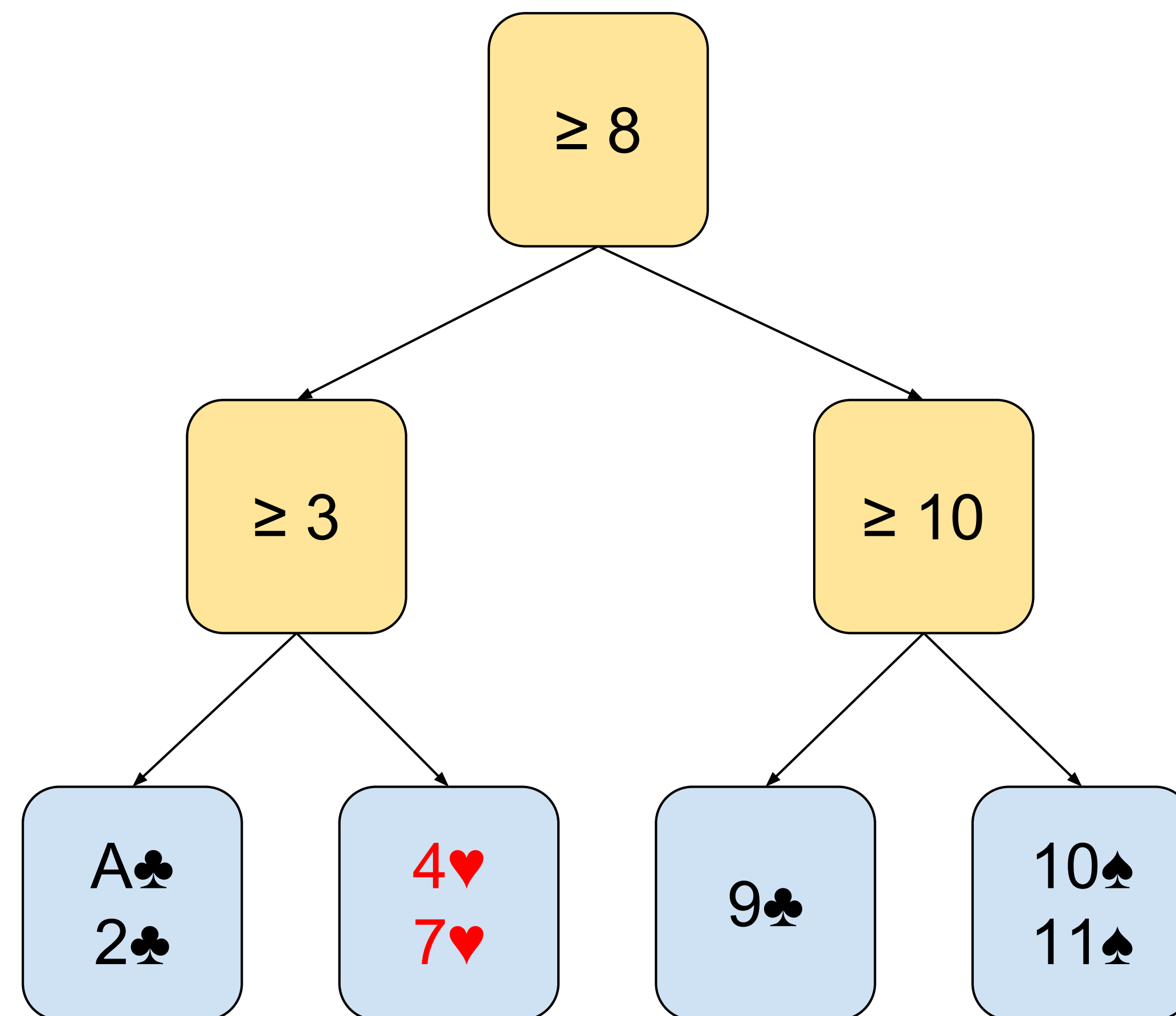
- Vamos dividir a turma em grupos
- Cada grupo recebe 13 cartas de um baralho comum
- Cada grupo deve **construir uma árvore de decisão para classificar suas cartas**
- Cada nó na árvore é simplesmente uma comparação com um valor entre Ás e Rei

Regras do Jogo

- Se a comparação é verdade para uma certa carta, ela vai para a direita do nó, caso contrário, ela vai para esquerda
- O objetivo é construir uma árvore tal que, ao passar todas cartas por ela, **cartas de um mesmo naipe devem estar separadas**
- Use a menor quantidade de nós possível!

Exemplo

- Recebi as cartas:
2♣, 7♥, 11♠, 10♠, A♣, 4♥, 9♣
- Nossa árvore poderia ser:



Let's do it!



Machine Learning?

Aprende com
experiências



Segue regras aprendidas
com dados



Segue regras
programadas



Maça ou Laranja?

- Se você tivesse que implementar um programa que recebe um imagem e retorna se é uma laranja ou maçã?



Maça ou Laranja?

```
if (orange_pxs - red_pxs > 103): return orange
```

```
if (orange_pxs/red_pxs > 2.4): return orange
```

```
if (surface_smoothness > 40%): return apple
```

```
if (green_pxs > orange_pxs): return apple
```

```
if (roundness == super_round): return orange
```

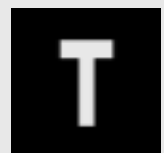
Maça ou Laranja?

- Sempre vai existir uma excessão!
- E se quisermos diferenciar bananas também? Teríamos que começar tudo de novo!
- Precisamos de um algoritmo que descubra e implemente essas regras para nós!
- **Machine Learning** é exatamente o estudo dessa classe de métodos e algoritmos!

Maça ou Laranja?

- Temos algumas informações sobre cada fruta: **features** ou **atributos**
- Qual fruta é: **target** ou **label**
- Cada linha é chamada de um **exemplo** ou **instância**
- Vamos construir nosso primeiro **classificador** com esses dados

Peso	Textura	Cor	Fruta?
150g	rugosa	laranja	laranja
170g	rugosa	verde	laranja
140g	lisa	vermelha	maça
130g	lisa	vermelha	maça
136g	lisa	verde	maça



`<code> ... </code>`

Estratégias de Aprendizado

- O fato que tínhamos dados com nosso target (fruta) durante o treinamento faz com que o problema seja **supervisionado**
- Caso não soubéssemos qual fruta era, ainda assim poderíamos usar estratégias **não-supervisionadas** para agrupar itens semelhantes

Estratégias de Aprendizado

Aprendizado supervisionado: Algoritmos que utilizam dados que incluem a resposta correta Y para aprender a correlação entre X e Y .

Aprendizado não-supervisionado: Algoritmos que somente possuem X e devem explorar padrões relevantes nesses dados para realizar alguma tarefa.

Aprendizado Supervisionado

X

Y

Peso, textura, cor

Fruta

Dados de uma transação bancária

Fraude

Áudio de um cliente

Texto transcrito

Imagem de um cachorro

Raça do cachorro

Perfil do usuário

Anúncio do seu interesse

Aprendizado Não-supervisionado

X**Tarefa**

Artigos da Wikipedia

Agrupar os artigos mais semelhantes

Tweets de seguidores de diferentes
candidatos

Termos mais comuns e característicos de
cada candidato

Clientes de uma instituição financeira

Perfis excepcionais a serem investigados

Chats de atendimento

Palavras com o mesmo significado

Impacto no Mercado

- Supervised Learning
- Deep Learning
- Transfer Learning
- Unsupervised Learning
- Reinforcement Learning

Supervisionado

Classificação: tentamos prever um label (uma categoria)

Regressão: tentamos prever um valor contínuo (um número)

Classificação

X

Y

Imagem

Gato ou cachorro?

Dados de uma transação bancária

Fraude?

Texto do email de um cliente

Problema que o cliente possui

Tweet

Sentimento positivo ou negativo?

Frame da câmera do seu iphone

É o seu rosto?

Regressão

X

Y

Características de um imóvel

Preço avaliado pelo corretor

Histórico de valores de uma ação
e eventos econômicos

Valor máximo no dia seguinte

Perfil de uso de um cliente

Gastos no mês seguinte

Dados da operação de atendimento

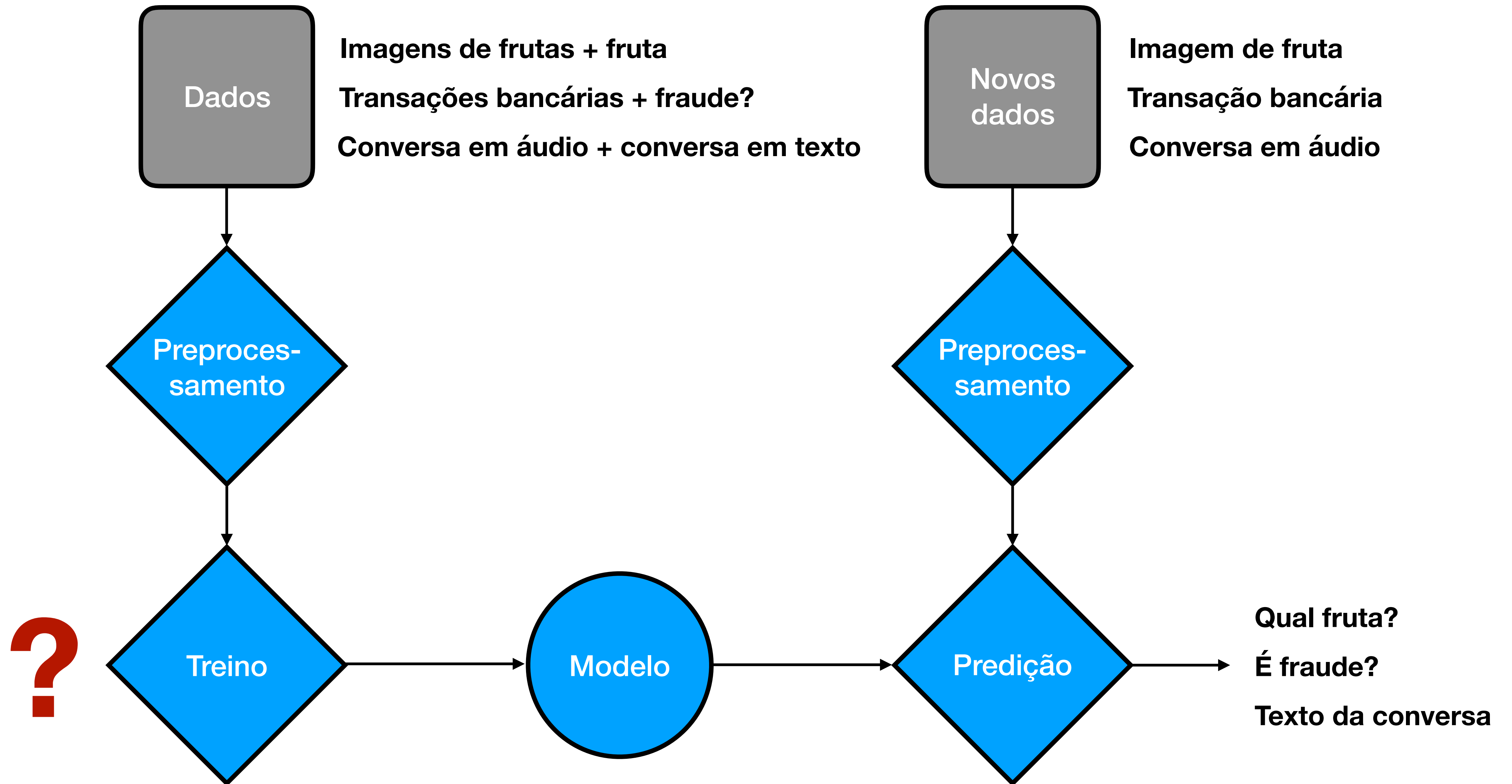
Volume de tickets e chats
no mês seguinte

Altura do pai

Altura do filho

The Machine Learning Process





Árvores de Decisão

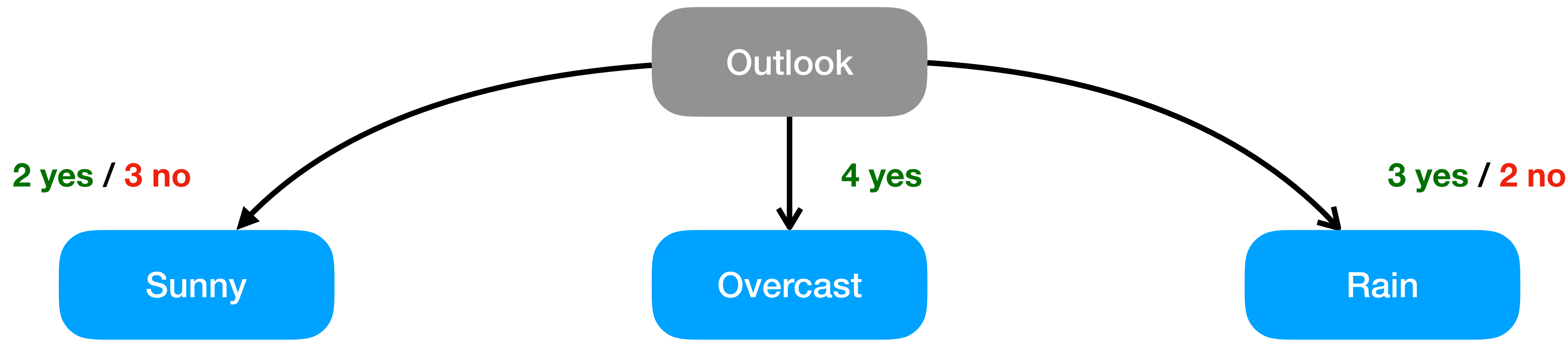
- Usadas amplamente em aplicações reais e competições
- Resolvem problemas não lineares excepcionalmente bem
- Extremamente flexíveis e interpretáveis
- Base para métodos muito poderosos (Gradient Boosting, Random Forests)

Exemplo

Day	Outlook	Humidity	Wind	Playing?
D1	sunny	high	strong	no
D2	sunny	high	weak	no
D3	overcast	high	weak	yes
D4	rain	high	weak	yes
D5	rain	normal	weak	yes
D6	rain	normal	strong	no
D7	overcast	normal	strong	yes
D8	sunny	high	weak	no
D9	sunny	normal	weak	yes
D10	rain	normal	weak	yes
D11	sunny	normal	strong	yes
D12	overcast	high	strong	yes
D13	overcast	normal	weak	yes
D14	rain	high	strong	no

- Queremos prever (e entender) quando alguém vai estar jogando na quadra.
- Temos informações de 14 dias:
 - Sobre o tempo
 - Alguém estava jogando (target)
- Que tal começamos pelo aspecto do tempo (outlook)?

T

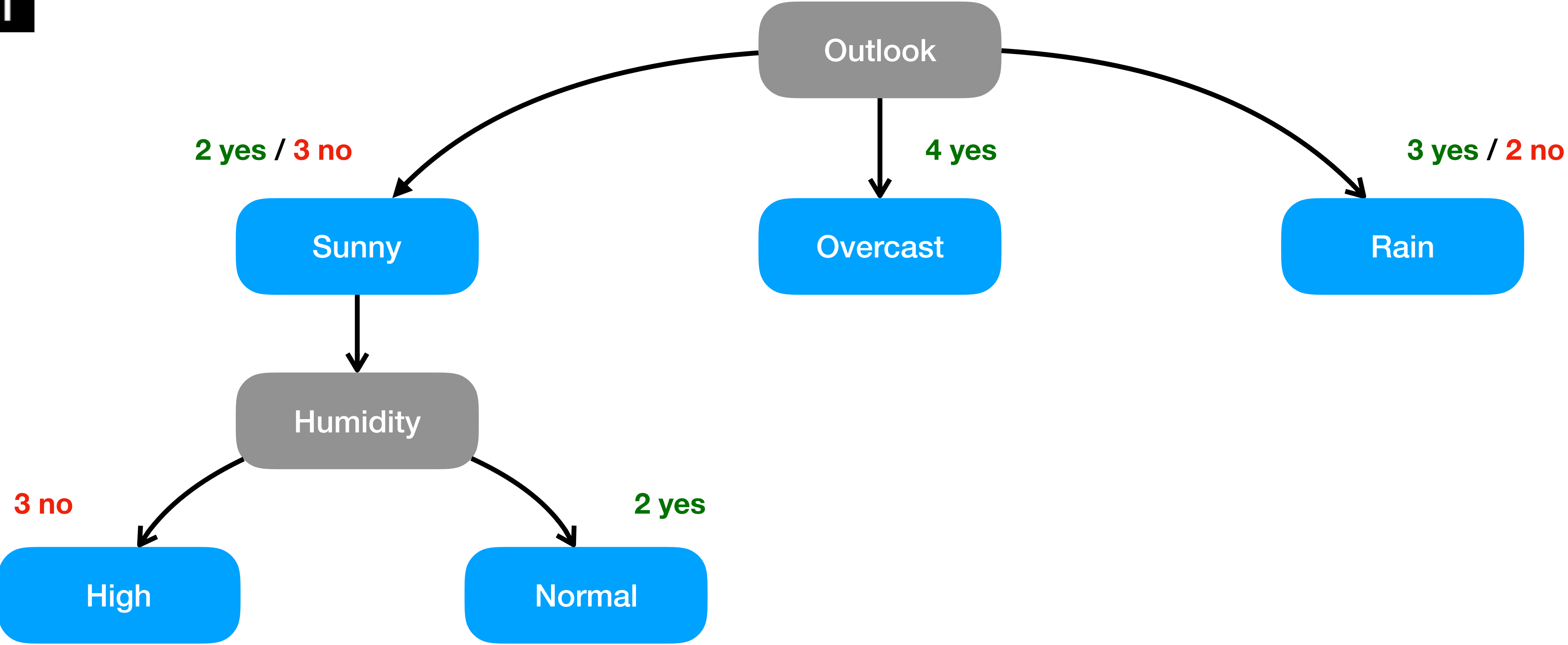


Outlook	Humidity	Wind
sunny	high	strong
sunny	high	weak
sunny	high	weak
sunny	normal	weak
sunny	normal	strong

Outlook	Humidity	Wind
overcast	high	weak
overcast	normal	strong
overcast	high	strong
overcast	normal	weak

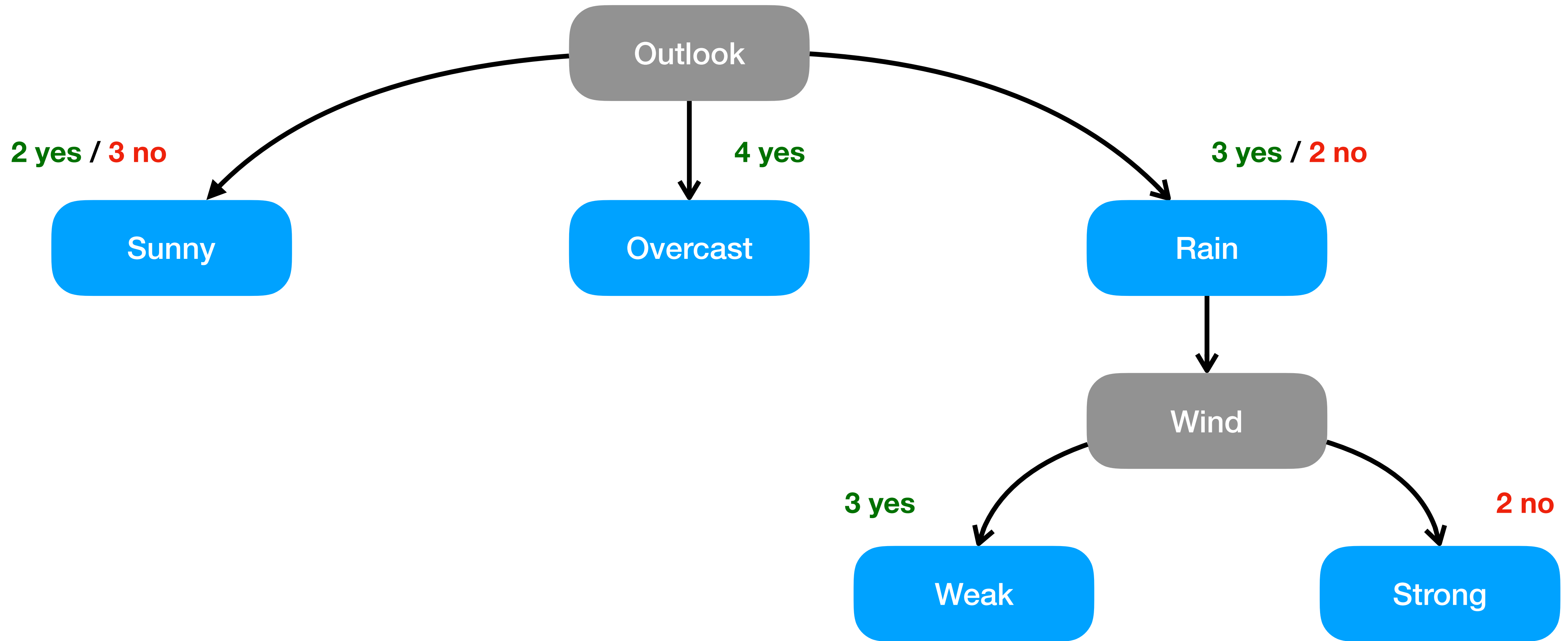
Outlook	Humidity	Wind
rain	high	weak
rain	normal	weak
rain	normal	weak
rain	normal	strong
rain	high	strong

T



Outlook	Humidity	Wind	Outlook	Humidity	Wind
sunny	high	strong	sunny	normal	weak
sunny	high	weak	sunny	normal	strong
sunny	high	weak			

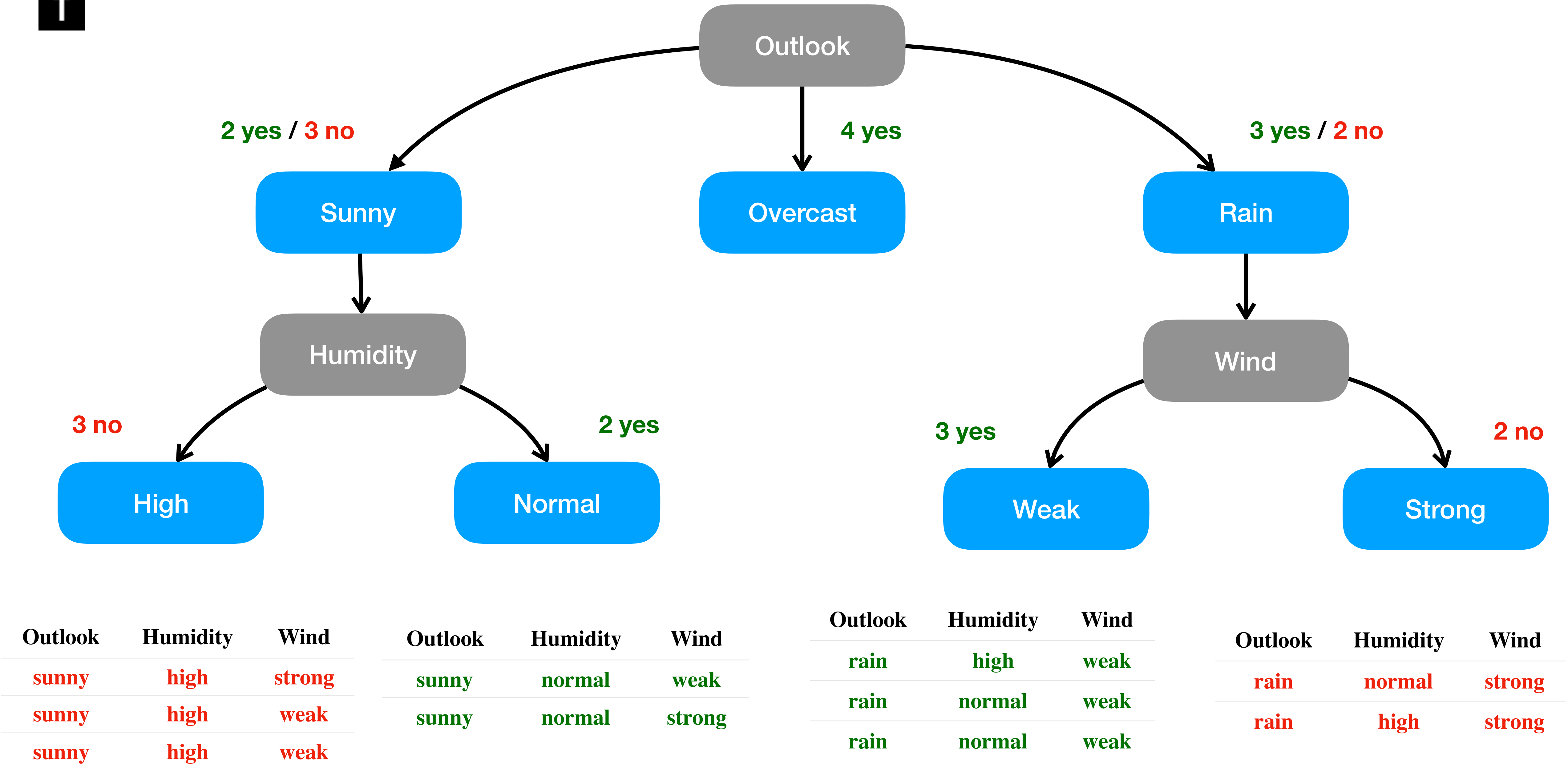
T



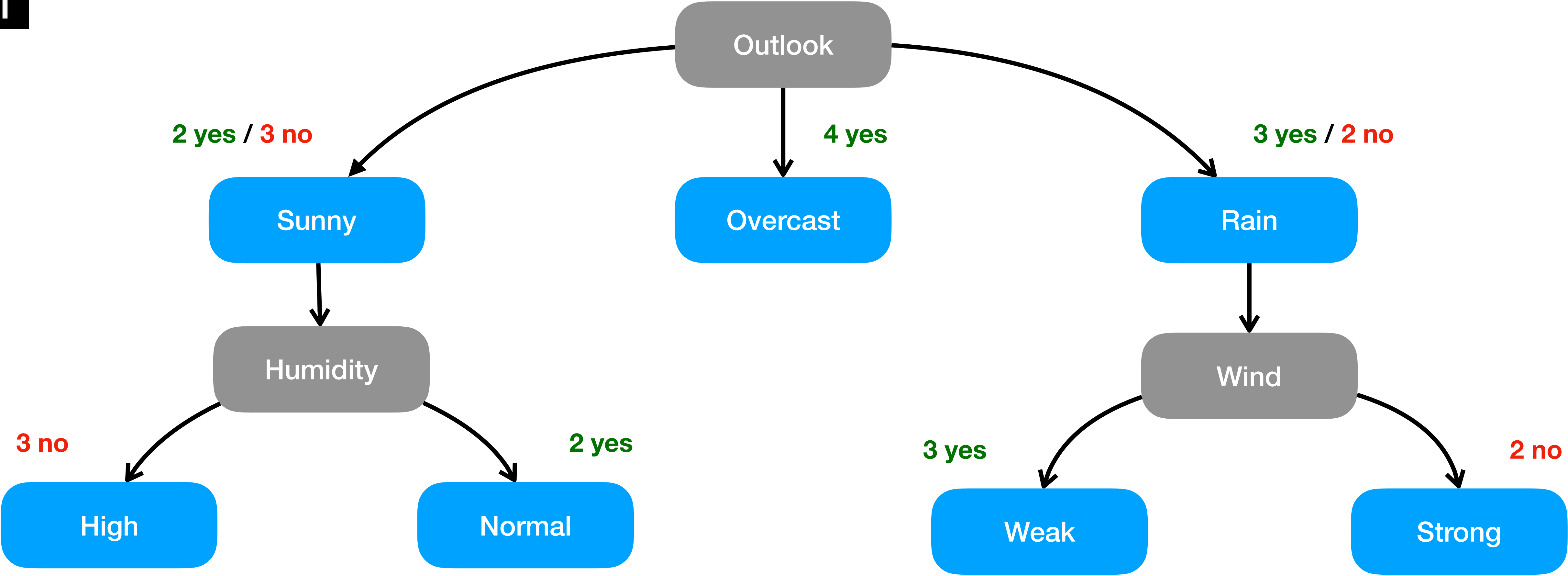
Outlook	Humidity	Wind
rain	high	weak
rain	normal	weak
rain	normal	weak

Outlook	Humidity	Wind
rain	normal	strong
rain	high	strong

T

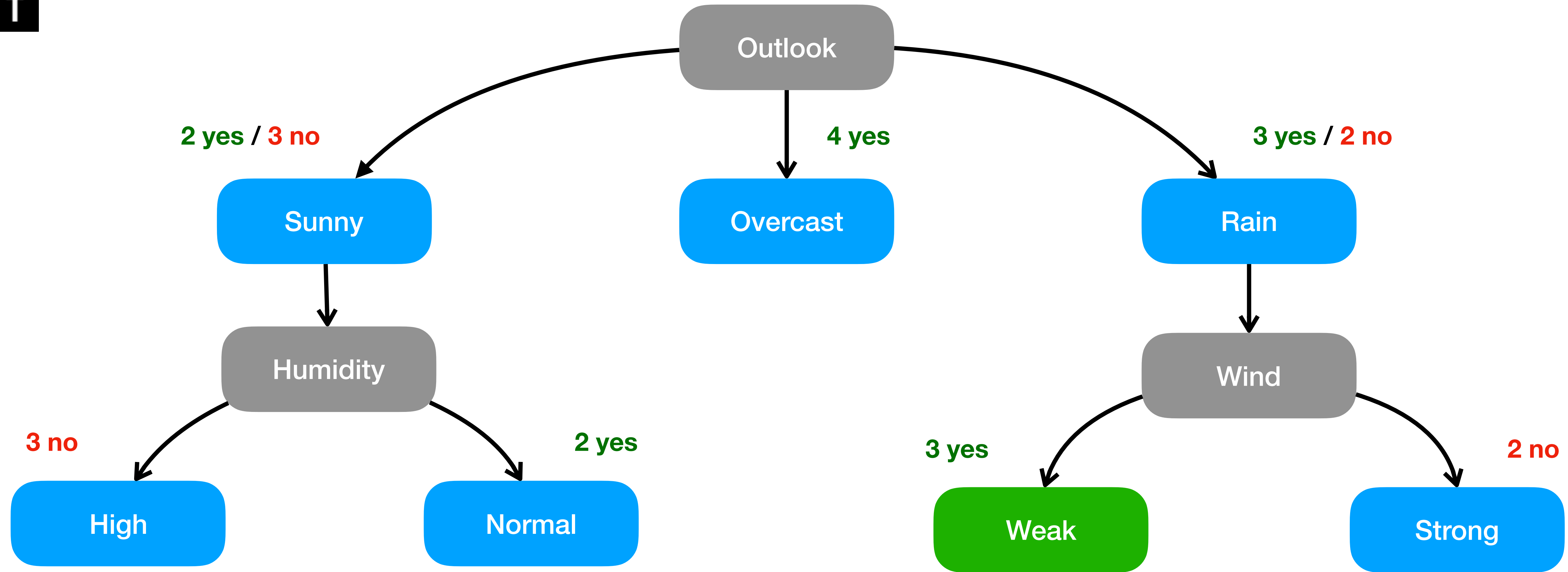


T



Day	Outlook	Humidity	Wind	Playing?
D15	rain	high	weak	???

T



Day	Outlook	Humidity	Wind	Playing?
D15	rain	high	weak	yes

Árvore de Decisão

*Um modelo estatístico **supervisionado** que busca aprender uma **seqüência de regras** estruturadas em uma árvore de modo a maximizar a **separação** entre instâncias de diferentes classes*

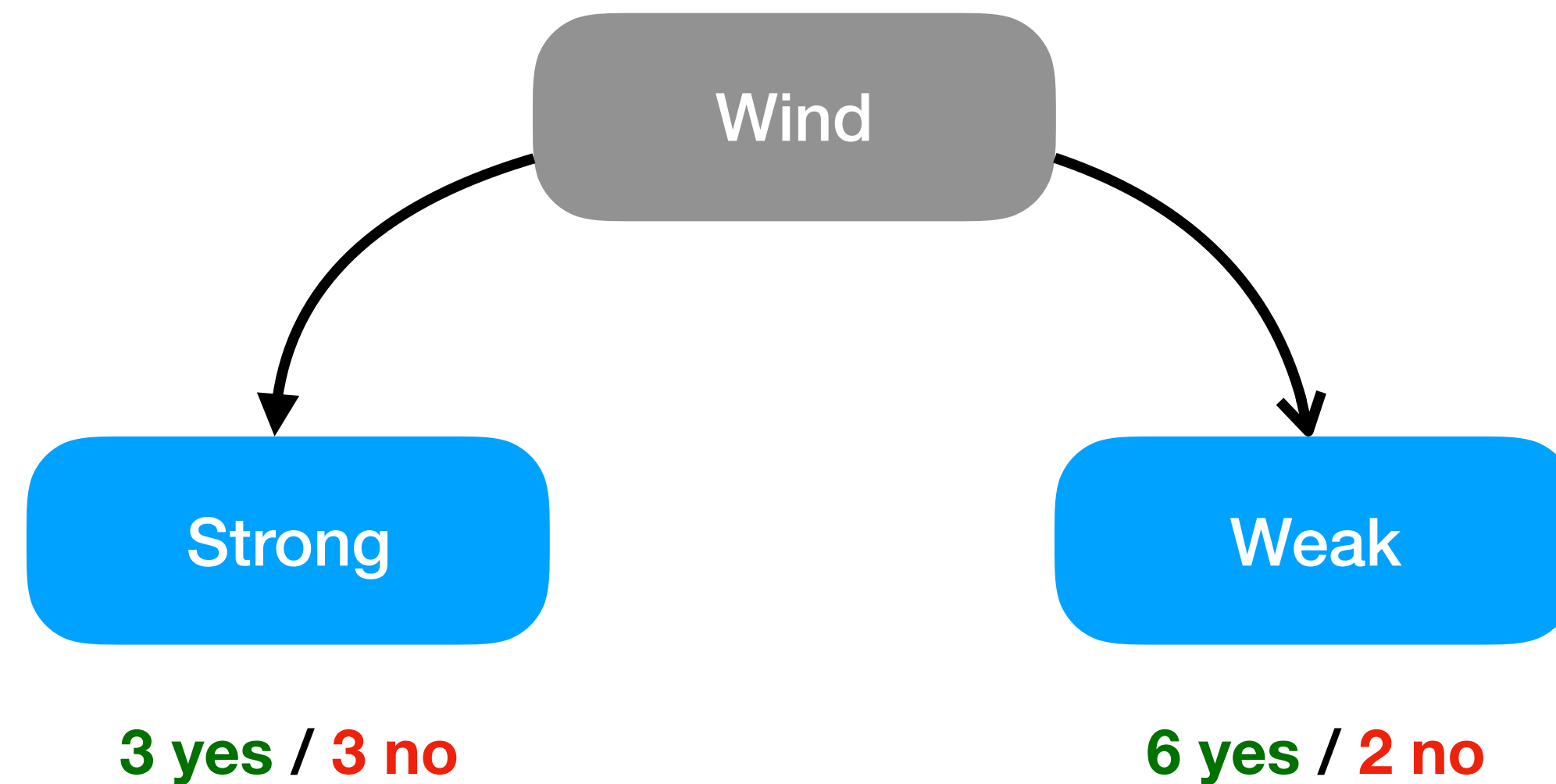
Árvore de Decisão

- Os nós finais são chamados de **folhas** e contém a decisão final em relação a classe (ou probabilidade das classes)
- Os nós com instâncias de uma só classe são chamados de **puro**
- Podem ser usadas para **classificação** e **regressão**
- Operam em atributos **numéricos** e **categóricos***

Escolhendo Atributos

- Escolhemos começar nossa árvore por Outlook
- E se tivéssemos começado por Wind?

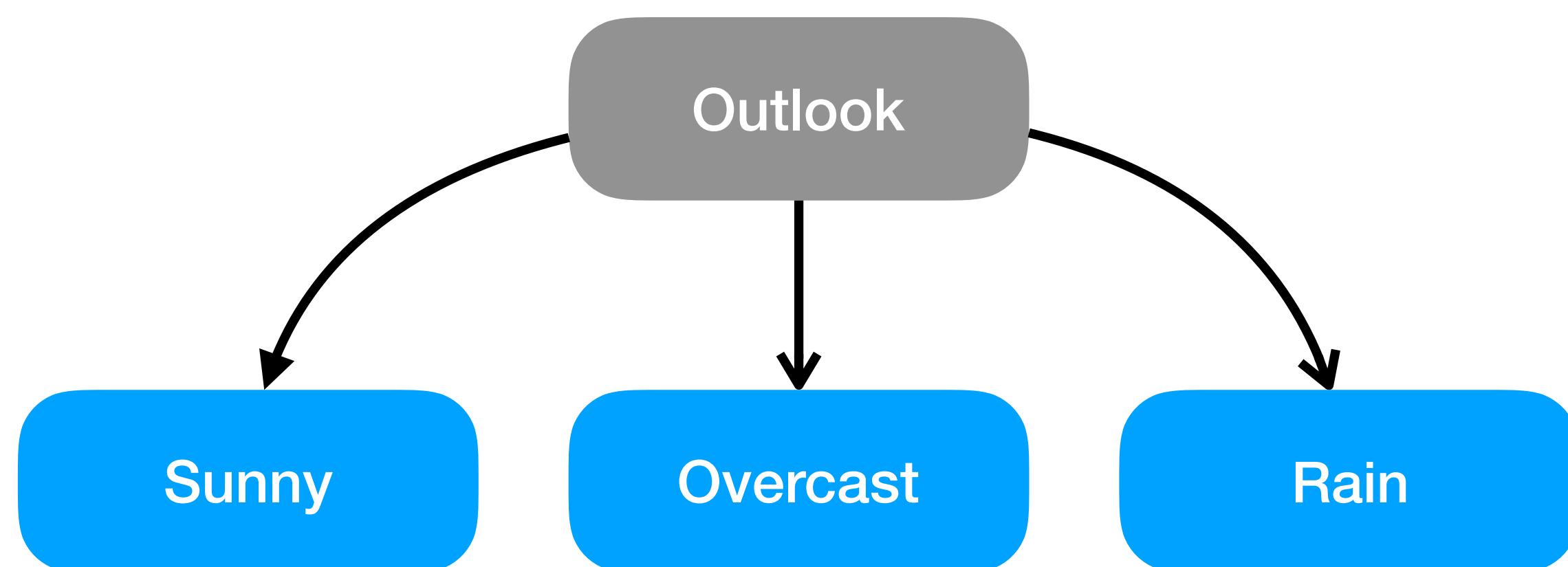
Outlook	Humidity	Wind
sunny	high	strong
rain	normal	strong
overcast	normal	strong
sunny	normal	strong
overcast	high	strong
rain	high	strong



Outlook	Humidity	Wind
sunny	high	weak
overcast	high	weak
rain	high	weak
rain	normal	weak
sunny	high	weak
sunny	normal	weak
rain	normal	weak
overcast	normal	weak

Escolhendo Atributos

- Qual escolha de *split* foi melhor?
- Suponhamos que temos que chutar o resultado logo depois do primeiro split



2 yes / 3 no

4 yes

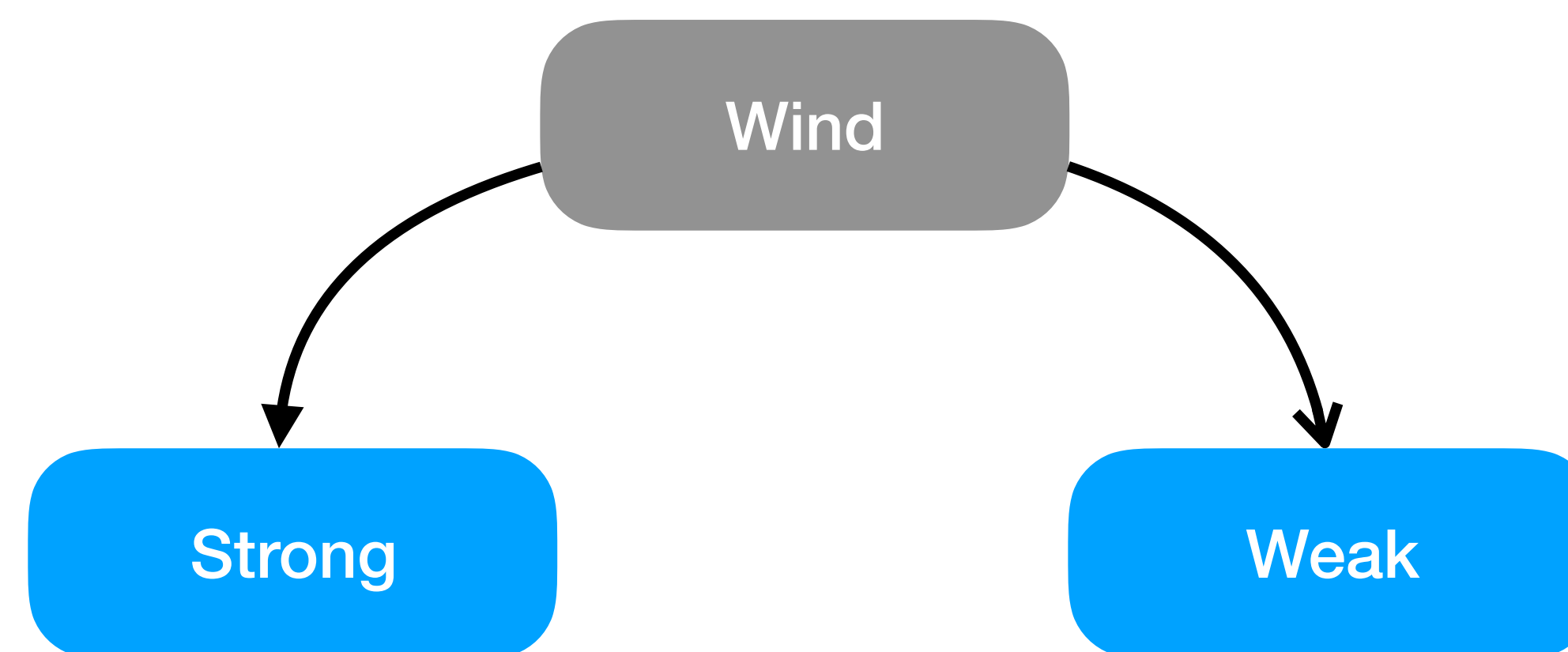
3 yes / 2 no

$$P(\text{no}|\text{sunny}) = 3/5$$

$$P(\text{yes}|\text{overcast}) = 4/4$$

$$P(\text{yes}|\text{rain}) = 3/5$$

$$P(\text{correct}|\text{humidity}) = (0.6 + 1 + 0.6)/3 = 0,7333$$



3 yes / 3 no

6 yes / 2 no

$$P(\text{yes}|\text{sunny}) = 3/6$$

$$P(\text{yes}|\text{weak}) = 6/8$$

$$P(\text{correct}|\text{wind}) = (0.5 + 0.75)/2 = 0,625$$

Escolhendo Atributos

- A probabilidade de acertamos é maior no primeiro split, pois ele discrimina melhor a nossa variável resposta (playing)
- O primeiro split representa um **ganho de informação** maior

Entropia

- Como escolher os melhores splits sistematicamente?
- Vamos usar a entropia dos subconjuntos como métrica de **impureza**:

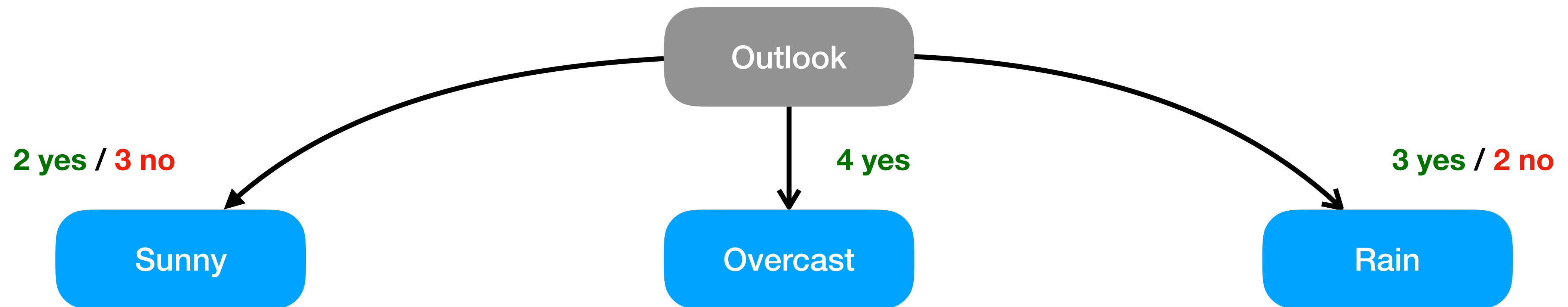
$$H(E) = -P_{yes} \log_2 P_{yes} - P_{no} \log_2 P_{no}$$

$$P_{yes} = \frac{n_{yes}}{n_{yes} + n_{no}}$$

$$P_{no} = \frac{n_{no}}{n_{yes} + n_{no}}$$

Exemplo

$$H(E) = -P_{yes} \log_2 P_{yes} - P_{no} \log_2 P_{no}$$



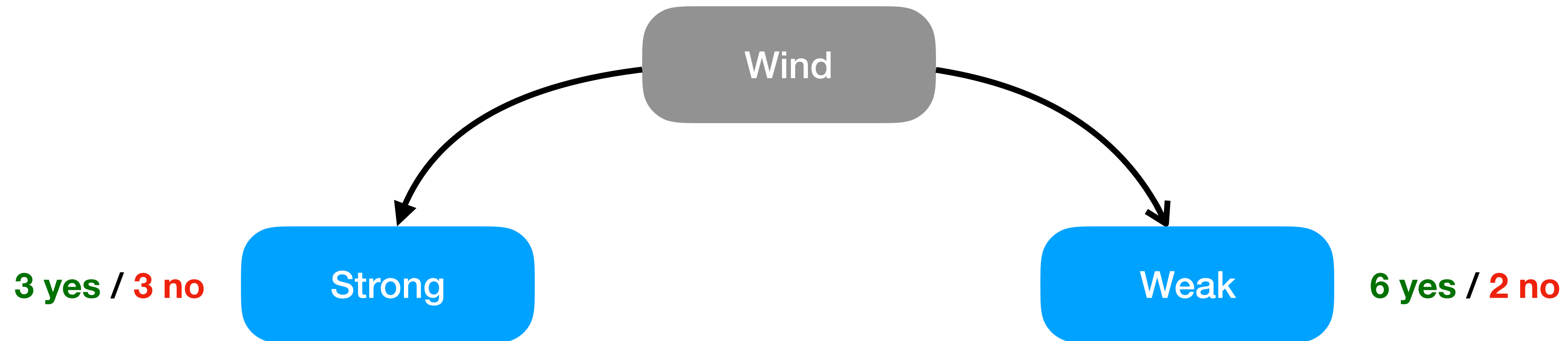
$$\begin{aligned}
 H(E_S) &= -\frac{2}{5} \log_2 \frac{2}{5} - \frac{3}{5} \log_2 \frac{3}{5} \\
 &= -0.4 \log_2 0.4 - 0.6 \log_2 0.6 \\
 &= 0.9705
 \end{aligned}$$

$$\begin{aligned}
 H(E_O) &= -\frac{4}{4} \log_2 \frac{4}{4} - \frac{0}{4} \log_2 \frac{0}{4} \\
 &= -1 \log_2 1 - 0 \log_2 0 \\
 &= 0.0
 \end{aligned}$$

$$\begin{aligned}
 H(E_R) &= -\frac{3}{5} \log_2 \frac{3}{5} - \frac{2}{5} \log_2 \frac{2}{5} \\
 &= -0.6 \log_2 0.6 - 0.4 \log_2 0.4 \\
 &= 0.9705
 \end{aligned}$$

Exemplo

$$H(E) = -P_{yes} \log_2 P_{yes} - P_{no} \log_2 P_{no}$$



$$\begin{aligned} H(E_s) &= -\frac{3}{6} \log_2 \frac{3}{6} - \frac{3}{6} \log_2 \frac{3}{6} \\ &= -0.5 \log_2 0.5 - 0.5 \log_2 0.5 \\ &= 1 \end{aligned}$$

$$\begin{aligned} H(E_w) &= -\frac{6}{8} \log_2 \frac{6}{8} - \frac{2}{8} \log_2 \frac{2}{8} \\ &= -0.75 \log_2 0.75 - 0.25 \log_2 0.25 \\ &= 0.8112 \end{aligned}$$

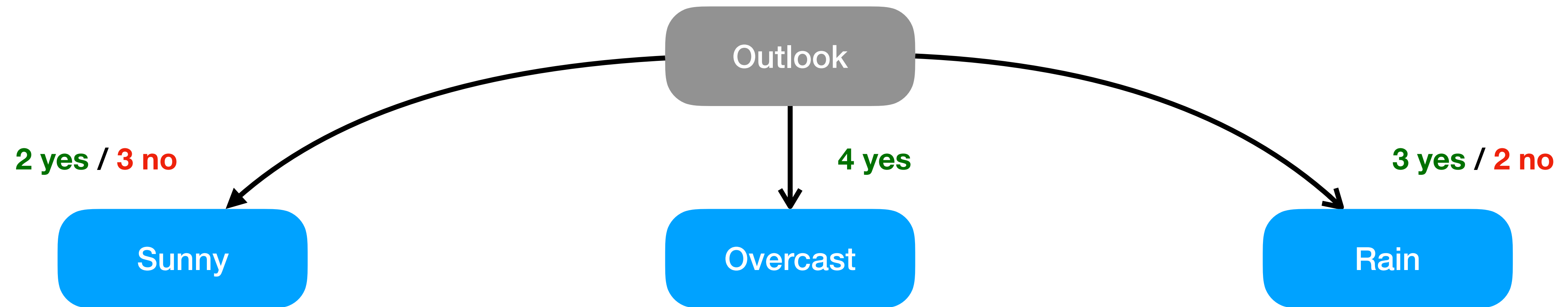
Entropia Esperada

- Ainda resta a pergunta, qual desses splits é objetivamente melhor?
- Considere um split **S** em um atributo com **K** valores diferentes, que divide o conjunto **E**, de tamanho **N**, em subconjuntos **E**₁, **E**₂, ..., **E**_K de tamanhos **N**₁, **N**₂, ..., **N**_K, respectivamente
- Vamos definir a **entropia esperada** após esse split como:

$$EH(S) = \sum_{i=1}^K \frac{N_i}{N} H(E_i)$$

Exemplo

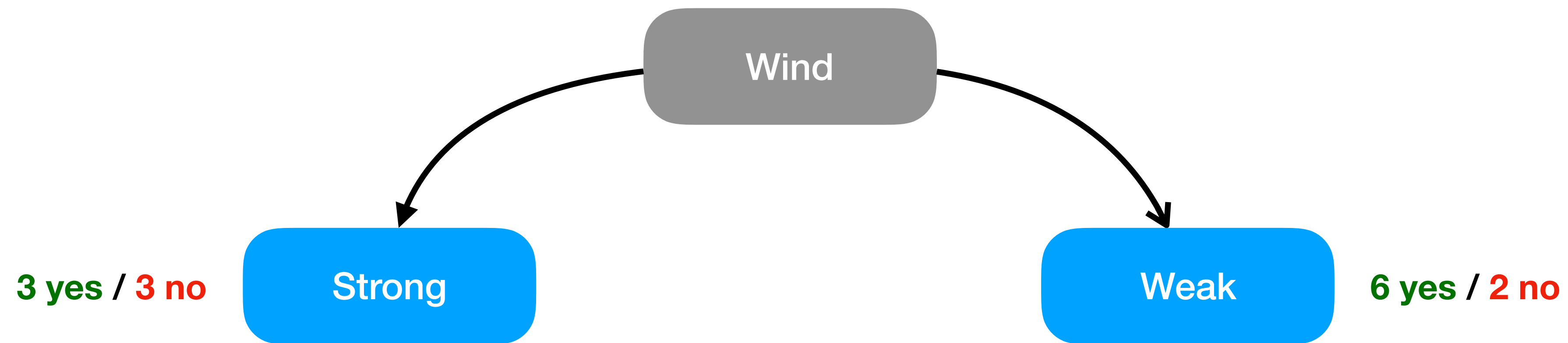
$$EH(S) = \sum_{i=1}^K \frac{N_i}{N} H(E_i)$$



$$\begin{aligned} EH(S_{Outlook}) &= \frac{5}{14} 0.9705 + \frac{4}{14} 0 + \frac{5}{14} 0.9705 \\ &= 0.6932 \end{aligned}$$

Exemplo

$$EH(S) = \sum_{i=1}^K \frac{N_i}{N} H(E_i)$$



$$\begin{aligned} EH(S_{Outlook}) &= \frac{6}{14} 1 + \frac{8}{14} 0.8112 \\ &= 0.8921 \end{aligned}$$

Ganho de Informação

- Queremos que haja **redução de entropia** para haver ganho de informação
- Por fim, definimos o **ganho de informação** de um split **S** como:

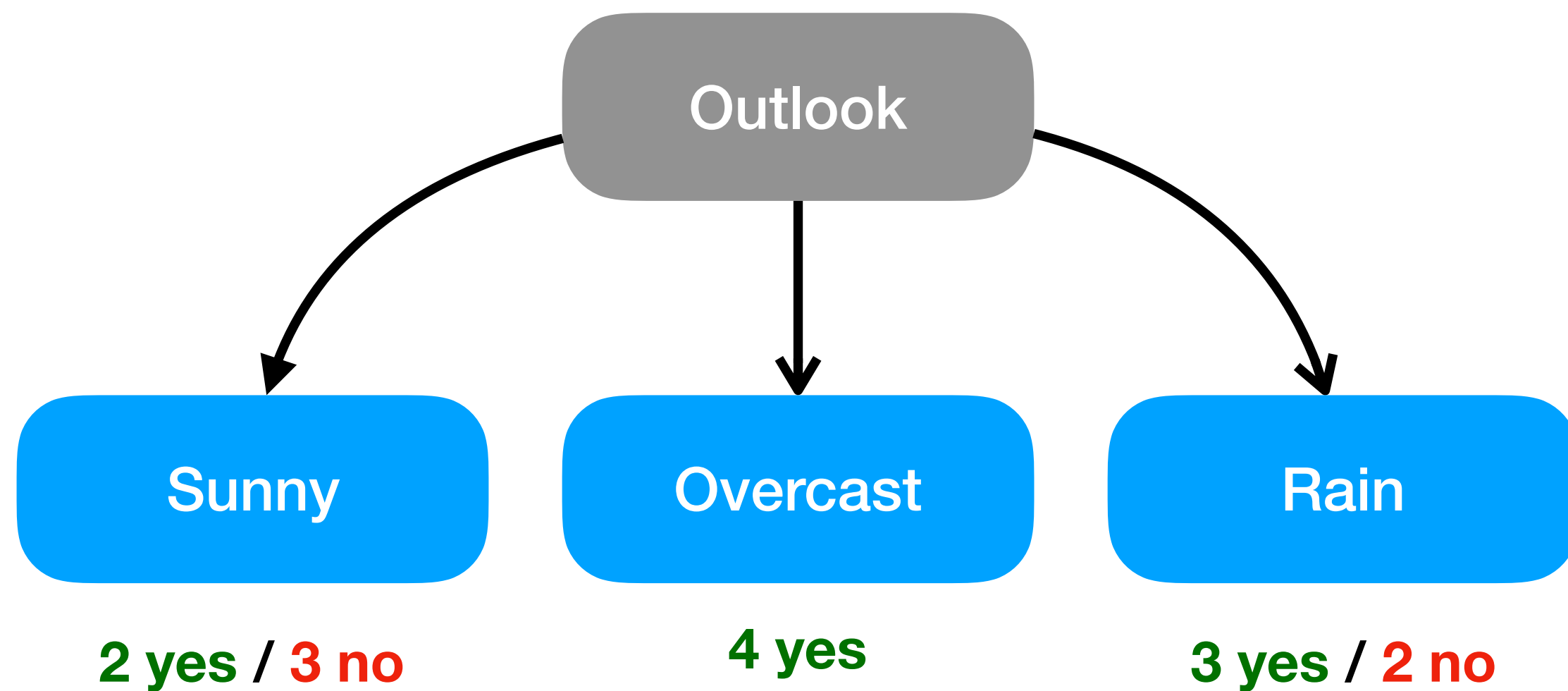
$$I(S) = H(E) - EH(S)$$

$$I(S) = H(E) - \sum_{i=1}^K \frac{N_i}{N} H(E_i)$$

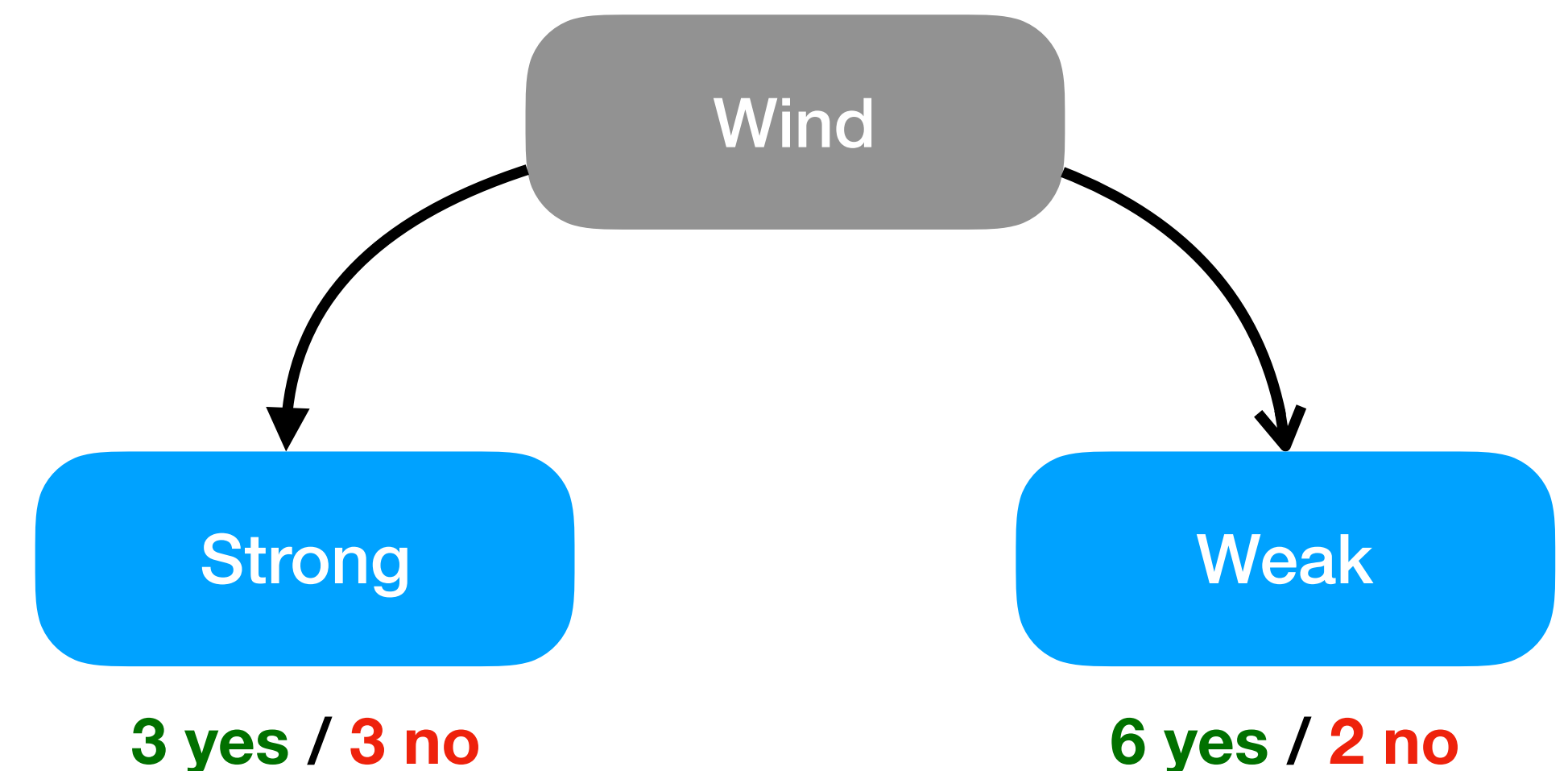
Exemplo

$$I(S) = H(E) - EH(S)$$

$$\begin{aligned} H(E) &= -\frac{9}{14} \log_2 \frac{9}{14} - \frac{5}{14} \log_2 \frac{5}{14} \\ &= -0.643 \log_2 0.643 - 0.357 \log_2 0.357 \\ &= 0.9402 \end{aligned}$$



$$\begin{aligned} I(S_{Outlook}) &= 0.9402 - 0.6932 \\ &= 0.2470 \end{aligned}$$



$$\begin{aligned} I(S_{Wind}) &= 0.9402 - 0.8921 \\ &= 0.0482 \end{aligned}$$

Exemplos

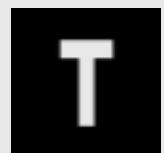
- $H([0,0,0])?$
- $H([1,0,0,1])?$
- $H([1,1,0]) > H([0,0,1])?$
- $H([1,1]) > H([1])?$
- $H([0,1,0]) > H([0,1,0,1,0])?$

Algoritmos

- Buscam o melhor split a todo momento
- Isso é feito achando o atributo e o valor que maximizam o ganho de informação
- Investigamos o caso de atributos **categóricos** (quente, húmido, etc), porém a lógica é a mesma para atributos **contínuos** (temperatura, humidade)
- E como vimos, funciona para **regressão** também

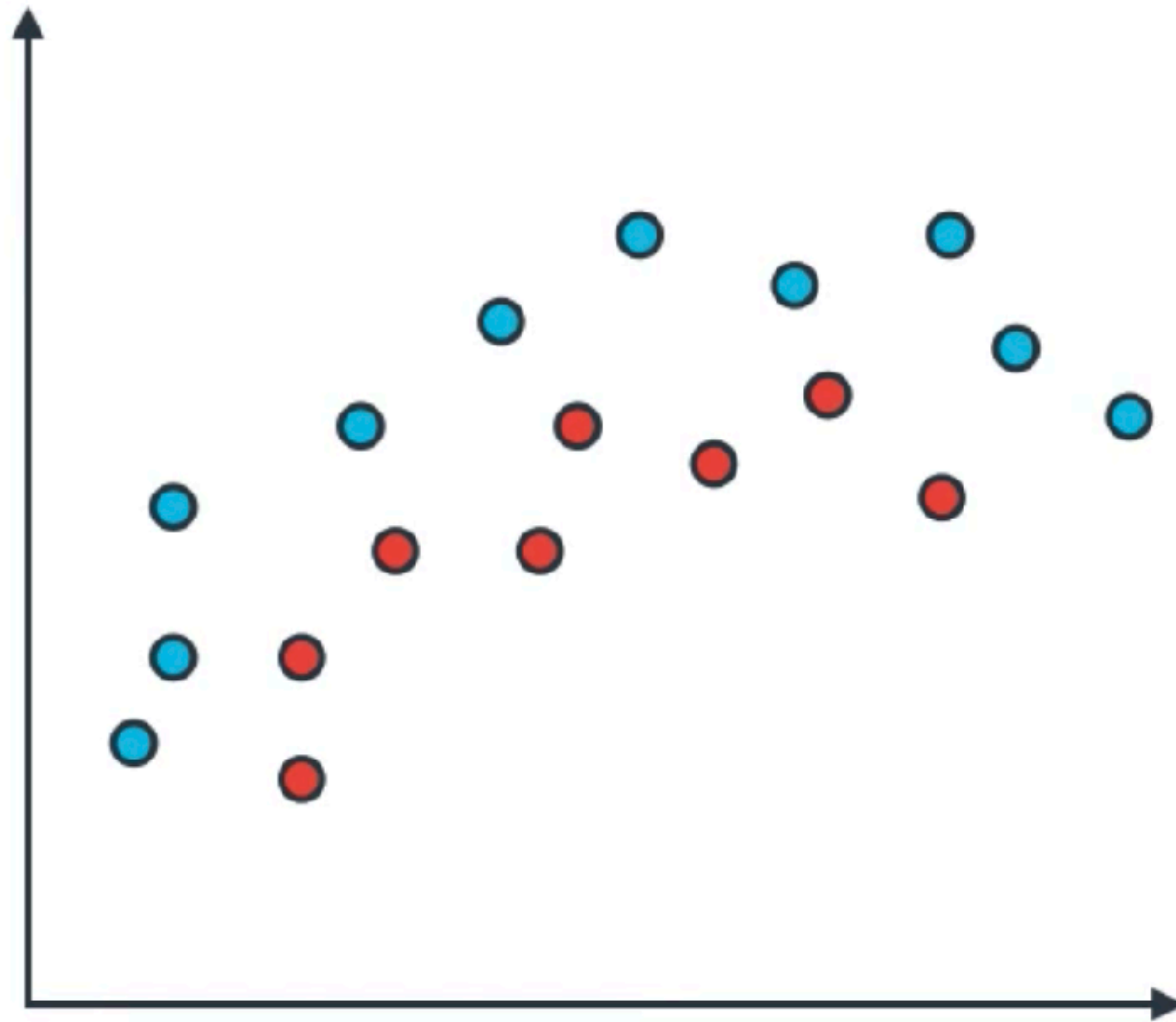
Avaliação

- Vamos supor que temos nossa árvore de decisão treinada
- Como podemos saber se ela está prevendo corretamente novos casos?
- Como avaliar a precisão dos nossos modelos?
- Existem inúmeras métricas e estratégias válidas

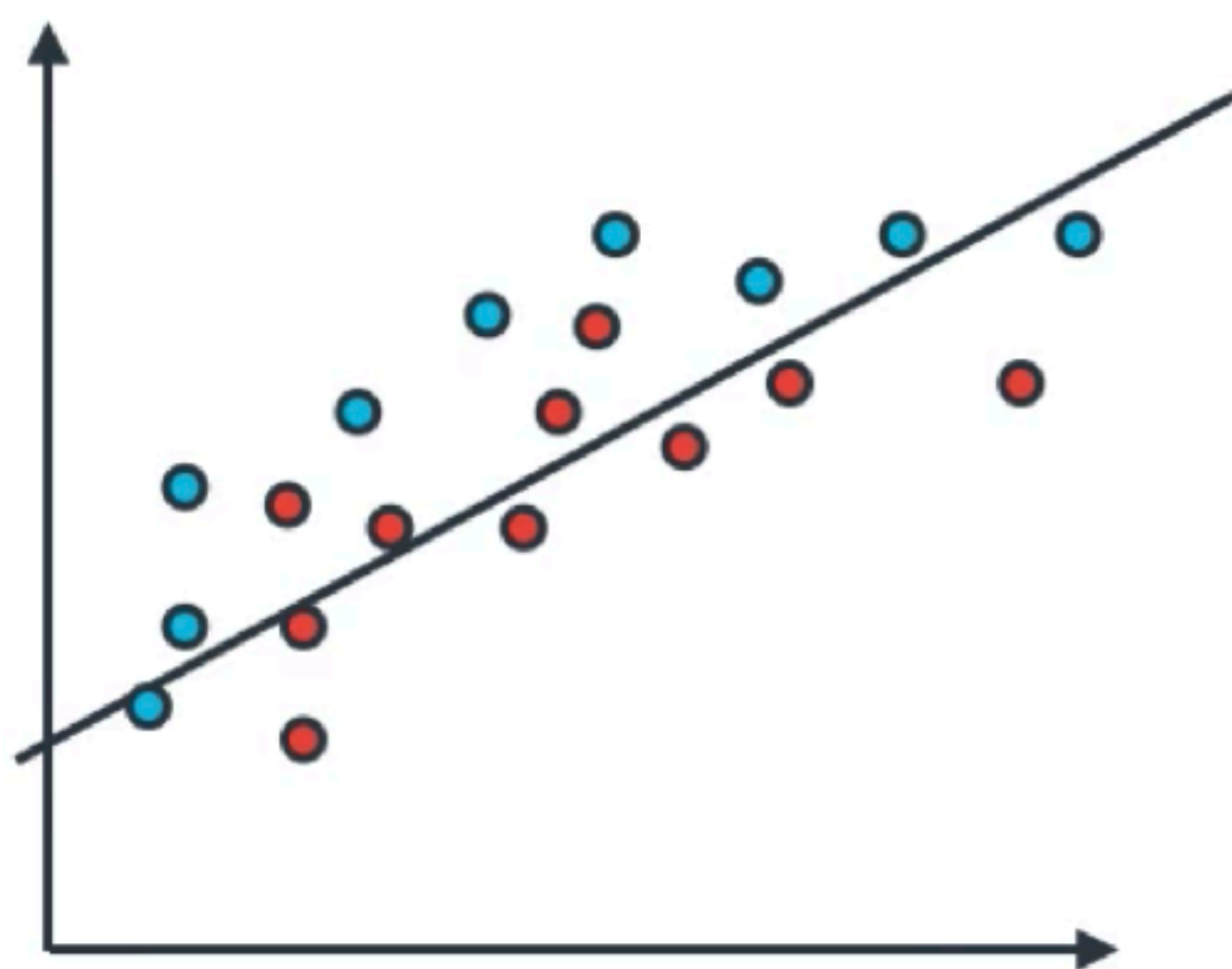


`<code> ... </code>`

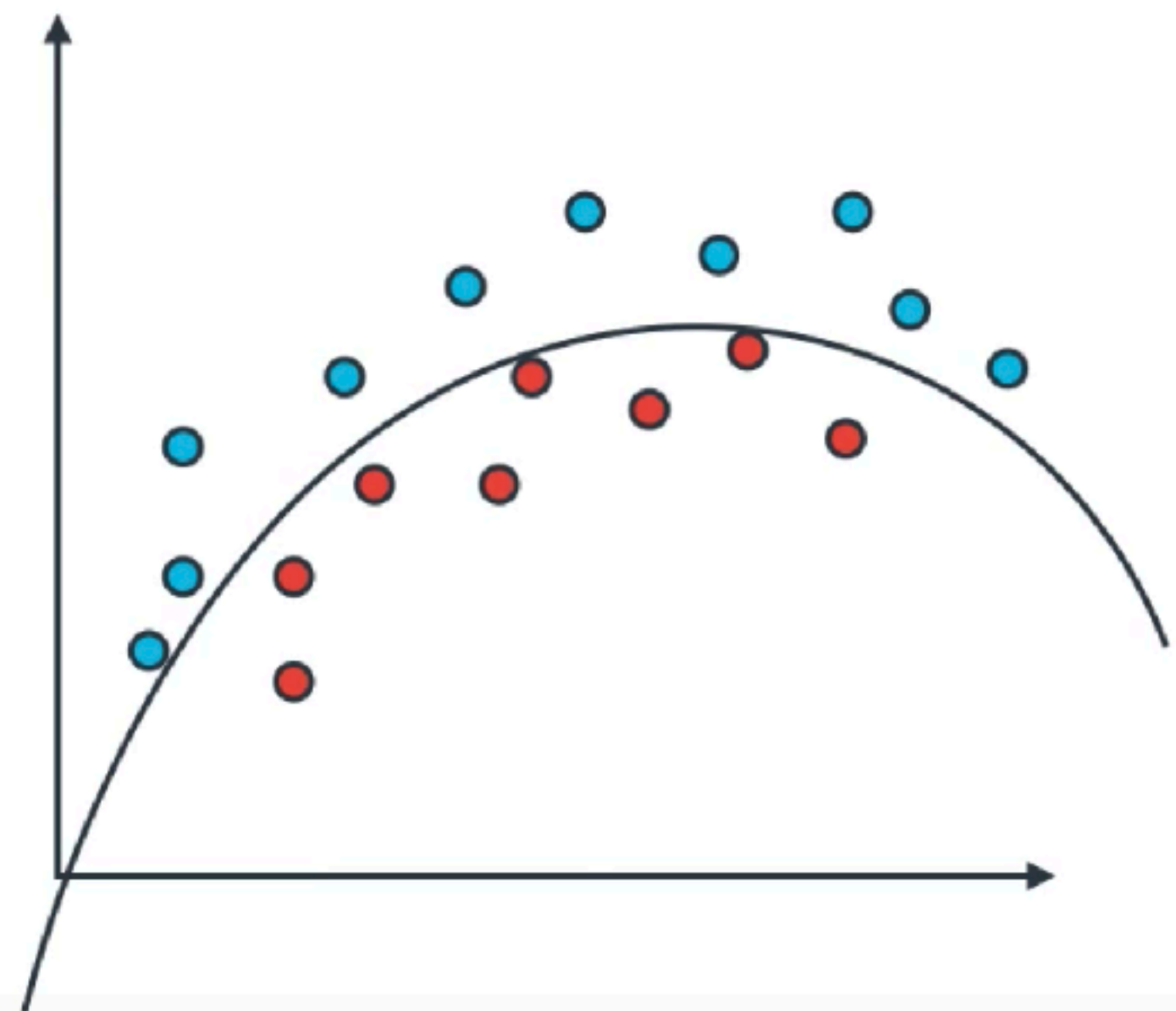
Overfitting vs Underfitting



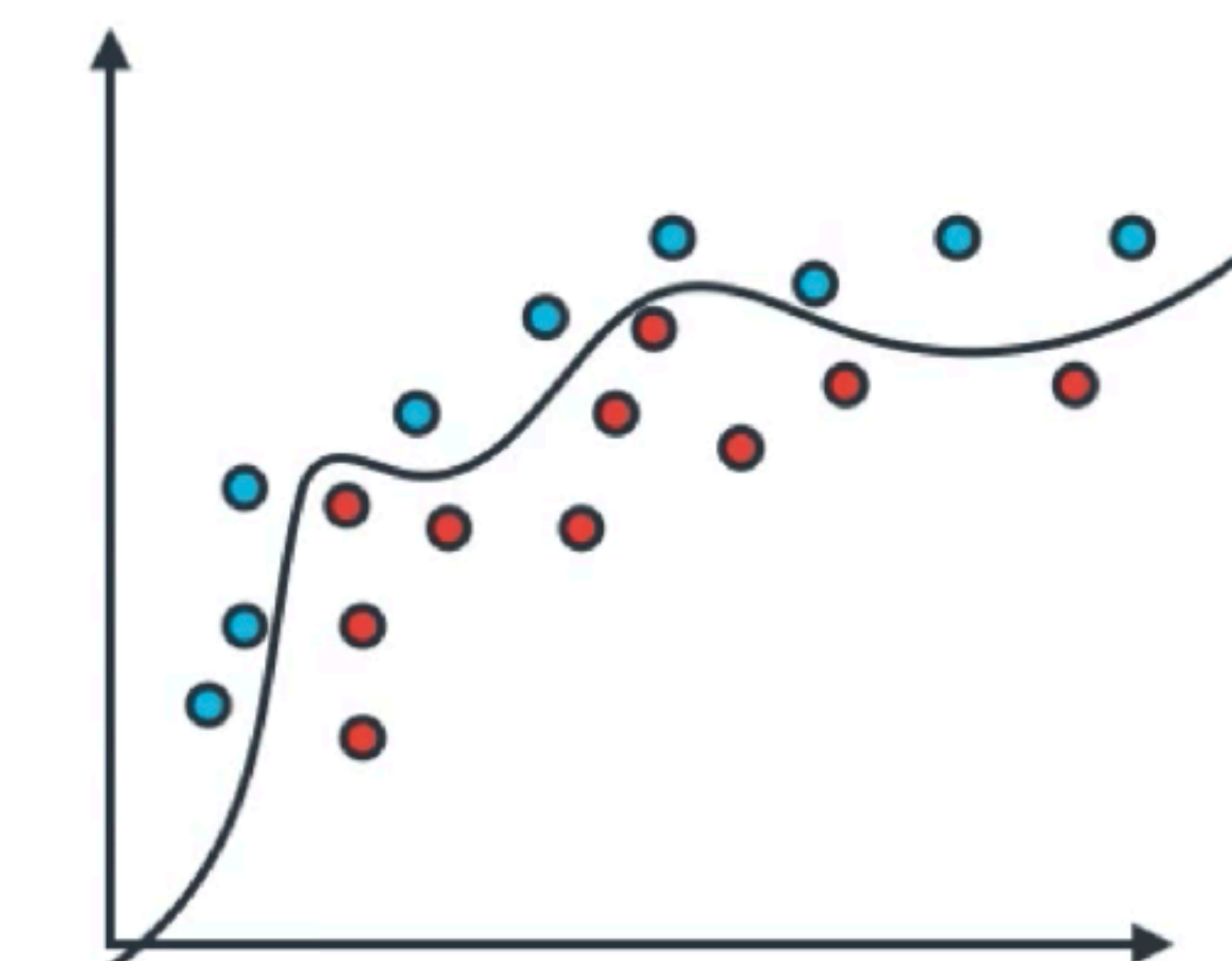
Overfitting vs Underfitting



Underfitted



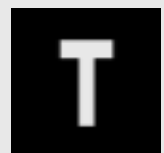
Well fitted



Overfitted

Hiper-parâmetros

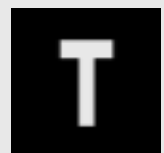
- Devemos encontrar o melhor fit, evitando underfitting e overfitting
- Isso geralmente é feito encontrando **os melhores hiper-parâmetros** para os nossos dados
- Os principais **hiper-parâmetros** que controlam complexidade:
 - *max_depth*: profundidade máxima da árvore
 - *min_samples_leaf*: mínimo de instâncias no nó para que haja um split



`<code> ... </code>`

Análise e Exploração

- Vimos como árvores de decisão podem ser transparente e informativas em relação ao que elas aprendem com os dados
- São muito poderosas para oferecer *insights* para seu negócio, mesmo quando um modelo preditivo não é necessário ou possível de se implementar



`<code> ... </code>`

Regressão

- Árvores de decisão podem também ser usadas para regressão
- Uma ideia possível é categorizar a saída em intervalos discretos e tratar o problema como classificação
- Dois aspectos precisam ser ajustados:
 - Qual métrica de impureza podemos usar para fazer os splits?
 - Que resposta retornamos nos nós folhas?

Regressão

- Ao invés de calcular a redução de entropia de cada subconjunto em um split, podemos calcular a **redução do desvio padrão**:

$$I'(S) = V(E) - EV(S)$$

- Onde $V(E)$ é o desvio padrão de E e $EV(S)$ é o desvio padrão esperado após o split S

$$I'([[0,0],[1,1,1]]) > I'([[0,1],[0,1,1]])$$

$$I'([[20,19],[7,9,6]]) > I'([[20,7],[19,9,6]])$$

`<code> ... </code>`

T

KAGGLE CASE HERE

Vantagens

- Extremamente **interpretáveis** e intuitivas
- **Pouco tratamento dos dados**, pois (em teoria) não exige normalização, remoção de outliers, imputação e one-hot encoding
- Lida com atributos **categóricos e contínuos**
- Predição relativamente **rápida** em $O(\log(n))$
- Possuem uma **seleção de atributos** natural

Desvantagens

- **Instáveis:** pequenas variações nas instâncias podem gerar árvores completamente diferentes
- Propensas a criar modelos muito complexos que não generalizam bem: **overfitting**
- O problema de criar uma árvore ótima é NP-completo, logo os algoritmos frequentemente retornam **árvores sub-ótimas**
- Felizmente, podemos mitigar muitos desses problemas com **métodos de ensembles** que aprenderemos em outra aula!

Conclusões

- Árvores de Decisão são modelos extremamente interessantes por sua versatilidade e interpretabilidade
- Muito cuidado deve ser tomado ao parametrizar o modelo devido ao risco de *overfitting*
- São muito úteis no processo de investigação e exploração, porém não tão poderosas em termos preditivos

Estudos Adicionais

- Post-pruning: removendo nós após a construção da árvore
- Extremely randomized trees
- Stacking
- XGBoost (Gradient Boosting Machine)
- Multi-output classification

DÚVIDAS?!