Classificação

1. Dado um grupo de instâncias (conjunto de treino) onde cada instância contém uma série de atributos e um deles é a **classe**.
2. Encontre um modelo para o atributo de classe em função dos valores dos demais atributos
3. Para novas instâncias, a classe atribuída deve ser o mais próximo possível do correto
4. Um conjunto de teste deve ser usado para determinar a acurácia do modelo.

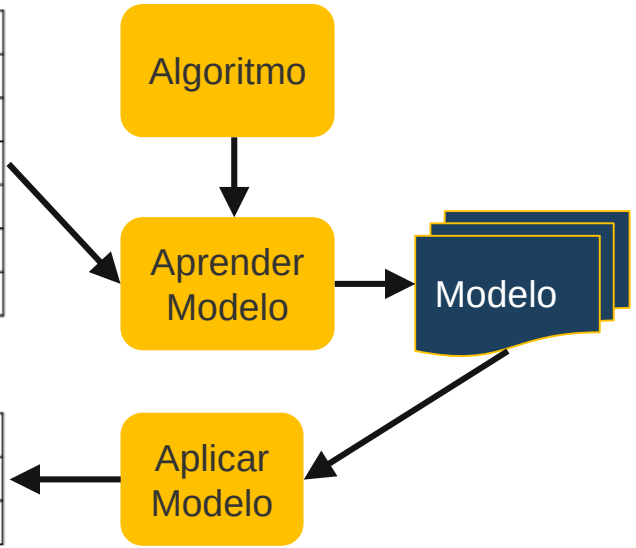
Algoritmos

- Árvore de Decisão
- Naïve Bayes e Redes Bayesianas
- Redes Neurais
- Máquinas de Vetores de Suporte (SVM)
- Regressão Logística

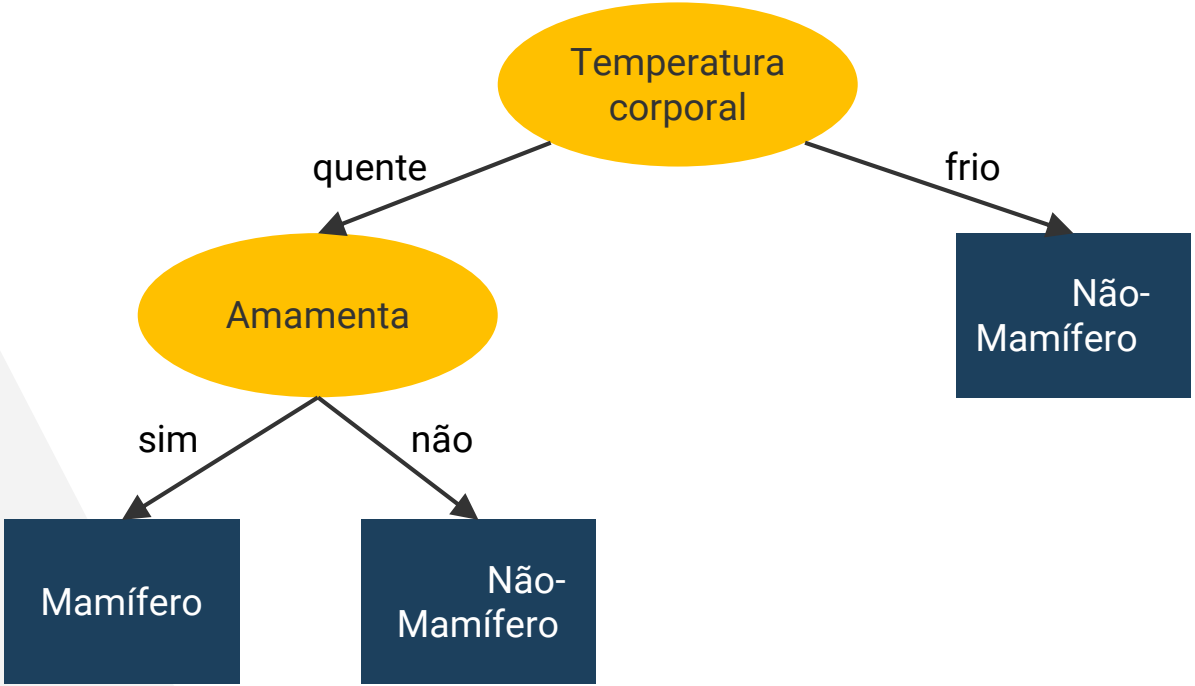
Classificação

ID	Nome	Enjôo	Mancha	Dor	Salário	Diagnóstico
01	Ana	sim	pequena	sim	1000	doente
02	Maria	não	pequena	não	1100	saudável
03	José	sim	grande	não	600	saudável
04	Pedro	não	pequena	sim	2000	doente
05	Paulo	não	grande	sim	1800	saudável
06	Juliana	não	grande	sim	900	doente

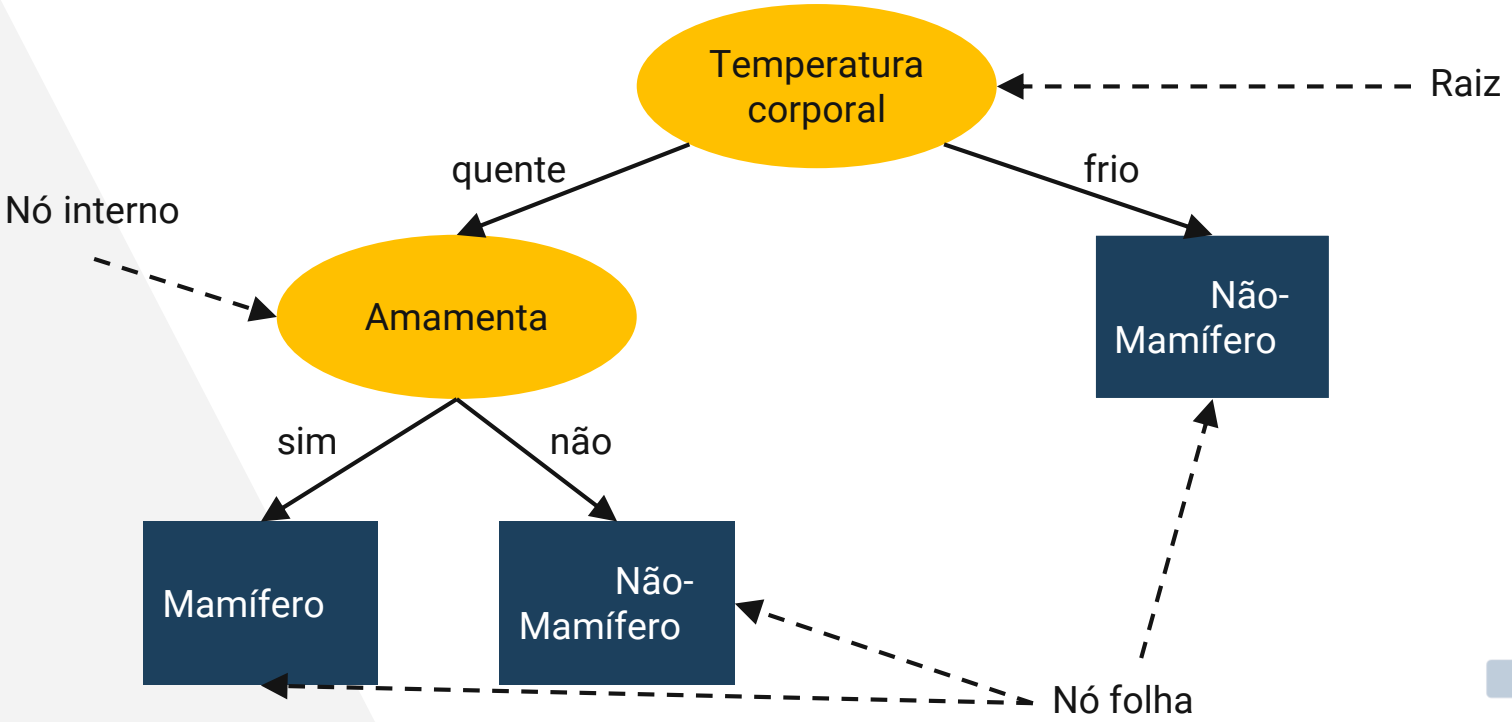
07	Jorge	sim	grande	sim	1900	?
08	Eduarda	não	pequena	não	1700	?
09	Francisco	sim	pequena	sim	950	?



Árvore de Decisão

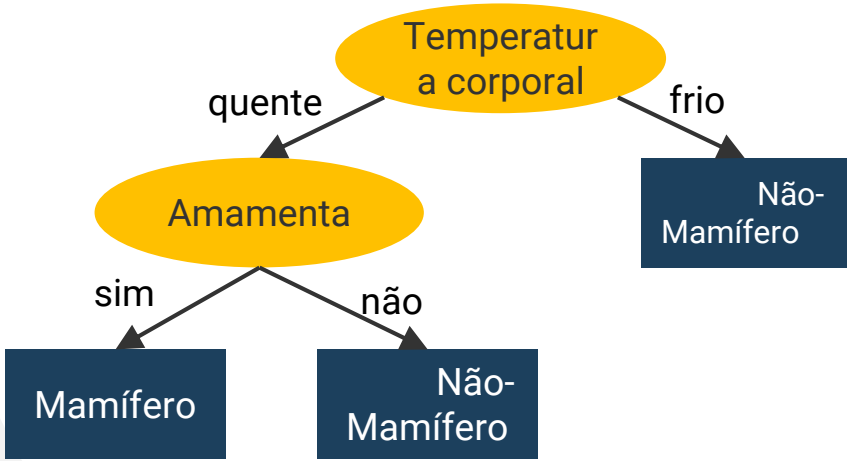


Árvore de Decisão



Árvore de Decisão

Nome	Temperatura Corporal	Amamenta	...	Classe
Flamingo	**Quente**	**Não**		**?**



Árvore de Decisão

- **Algoritmo de Hunt**: D_t = conjunto de treino. y = rótulos de classses

- **Passo 1**: Se todas as instâncias em D_t pertencem à mesma classe y_t, então t é um nó folha rotulado y_t

Árvore de Decisão

- **Passo 2**: Se D_t contém instâncias que pertencem à mais de uma classe, uma condição de teste baseada em um atributo deve ser selecionada para dividir as instâncias em conjuntos menores. Um nó filho deve ser criado para cada saída desta condição e as instâncias em D_t são distribuídas para os nós gerados com base nas saídas. O algoritmo é aplicado recursivamente para cada nó filho.

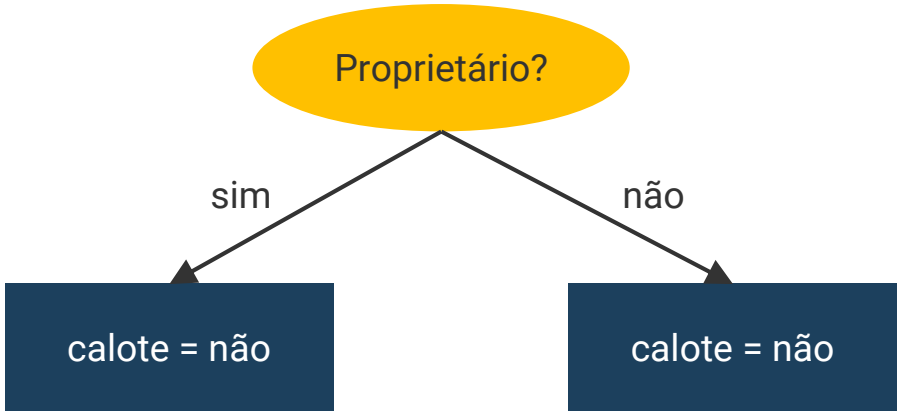
Árvore de Decisão

ID	Proprietário de imóvel	Estado civil	Renda Annual	Calote
1	Sim	Solteiro	125k	Não
2	Não	Casado	100k	Não
3	Não	Solteiro	70k	Não
4	Sim	Casado	120k	Não
5	Não	Divorciado	95k	Sim
6	Não	Casado	60k	Não
7	Sim	Divorciado	220k	Não
8	Não	Solteiro	85k	Sim
9	Não	Casado	75k	Não
10	Não	Solteiro	90k	sim

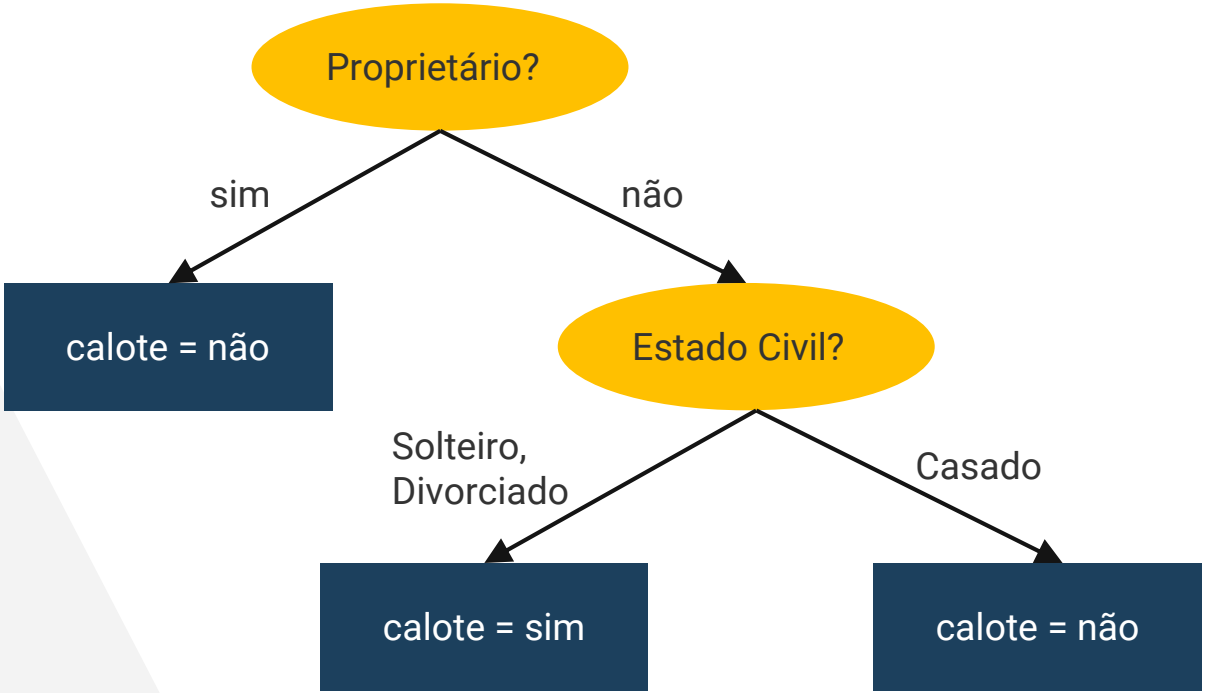
Árvore de Decisão

calote = não

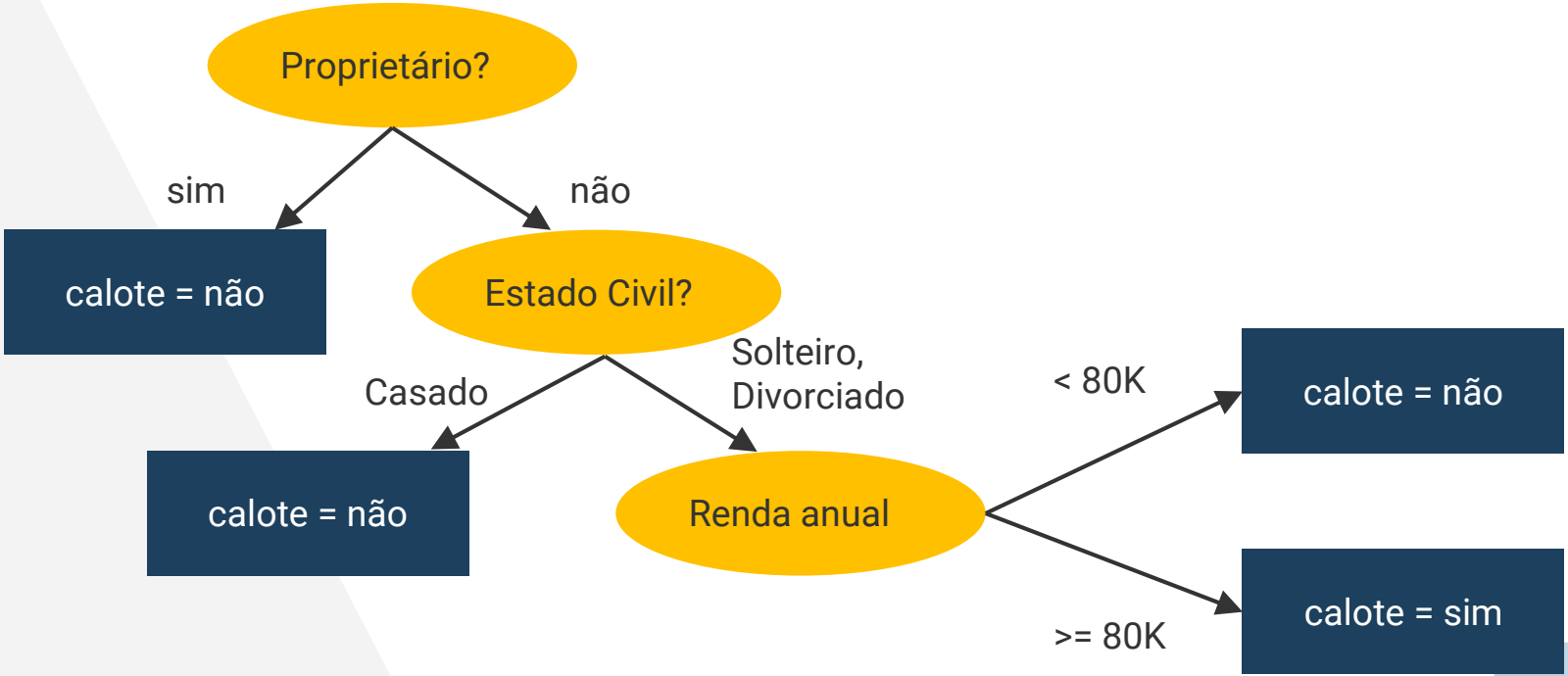
Árvore de Decisão



Árvore de Decisão



Árvore de Decisão



Árvore de Decisão

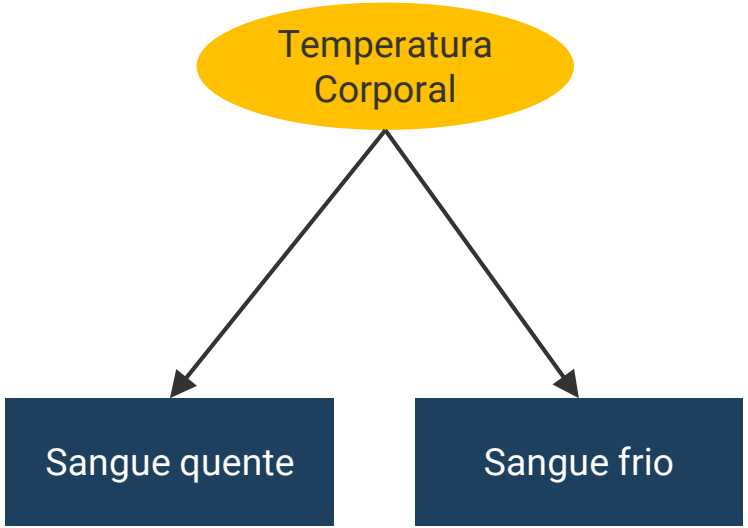
- Pode haver situações onde não é possível fazer a distribuição das instâncias em nós filhos
- Quando todos os valores dos atributos (exceto a classe) forem iguais;
- Quando não houver instâncias que possuem a combinação de atributos necessária para ser atribuídas a um nó.

Árvore de Decisão

1. Como dividir as instâncias de treino?
2. Quando parar de dividir?

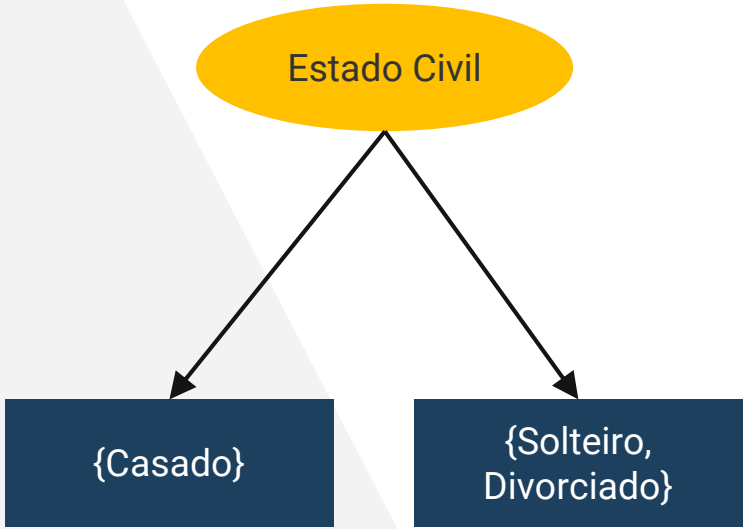
Árvore de Decisão

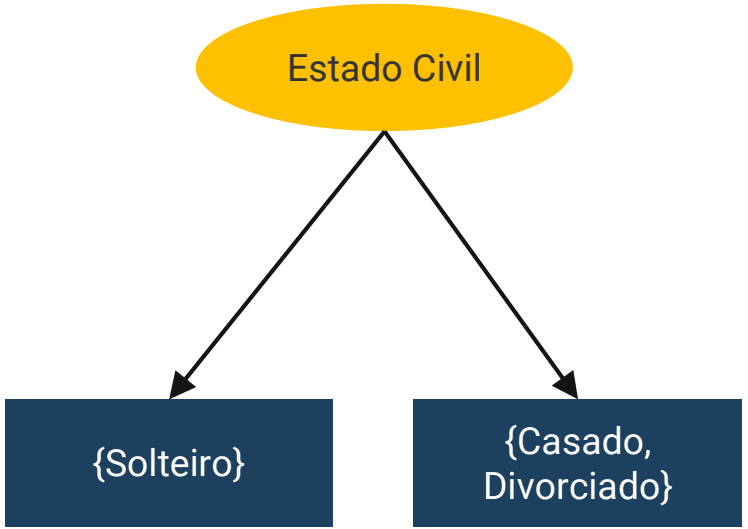
- Atributos binários



Árvore de Decisão

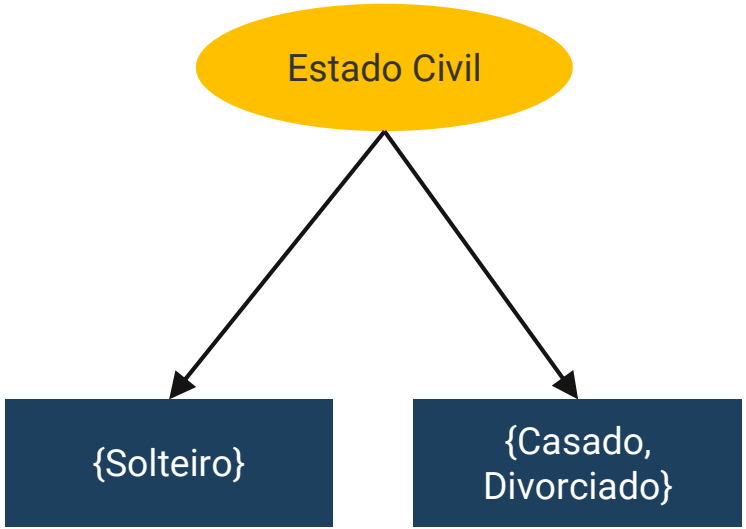
- Atributos categóricos e ordinais





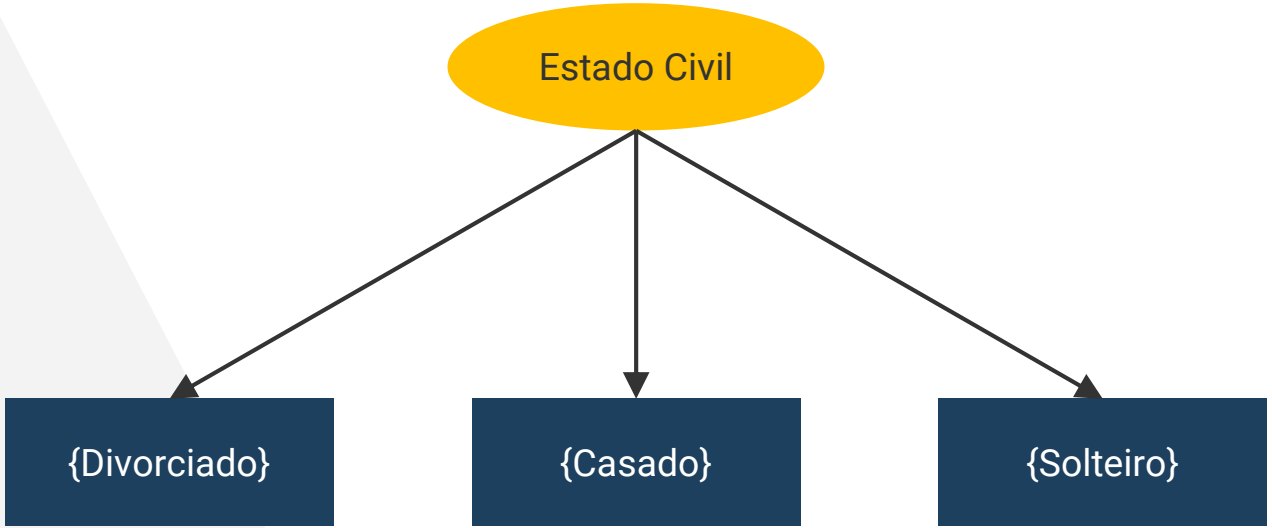
Árvore de Decisão

- Atributos categóricos e ordinais



Árvore de Decisão

- Atributos categóricos e ordinais



Árvore de Decisão

- Atributos categóricos e ordinais
- Tamanho da camiseta = {P, M, G, GG}
- Divisões possíveis?

Árvore de Decisão

- Atributos categóricos e ordinais
- Tamanho da camiseta = {P, M, G, GG}
- Divisões possíveis?
- $2^{k-1}-1$ divisões binárias

Árvore de Decisão

- Atributos contínuos?

Árvore de Decisão

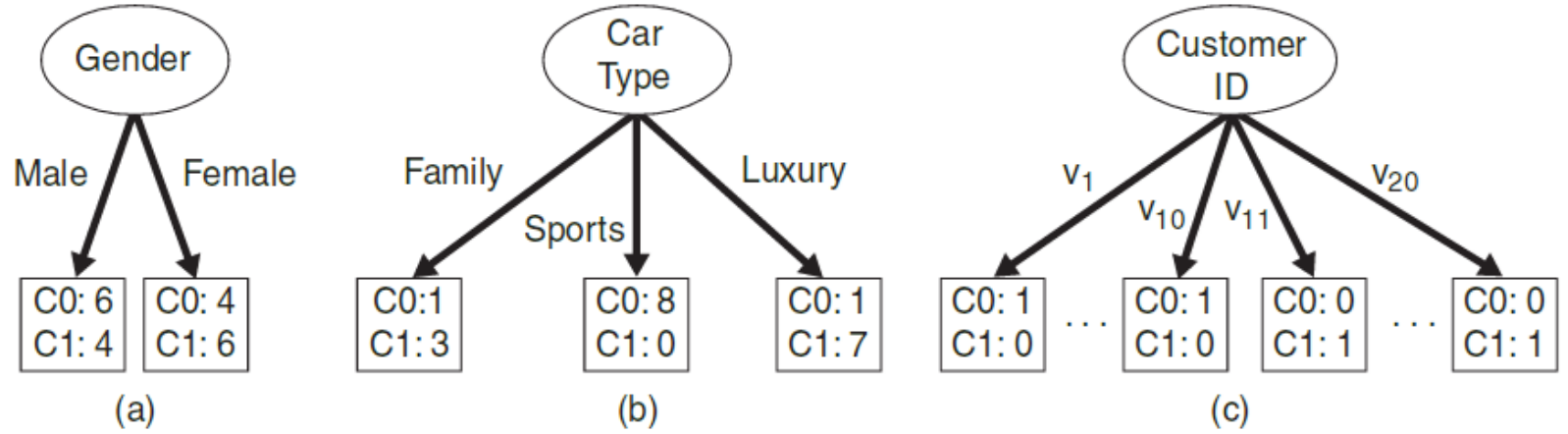
- Atributos contínuos?
- Aplicar discretização

Árvore de Decisão

- Medidas para selecionar a melhor divisão
- Feito com base na distribuição das classes antes e depois de dividir
- **p(i | t)** é a fração de instâncias que pertencem à classe **i** em um nó **t**. Também denominado p_i.
- Em um problema com duas classes a distribuição em qualquer nó pode ser definida como (p0, p1), onde p1 = 1 - p0.

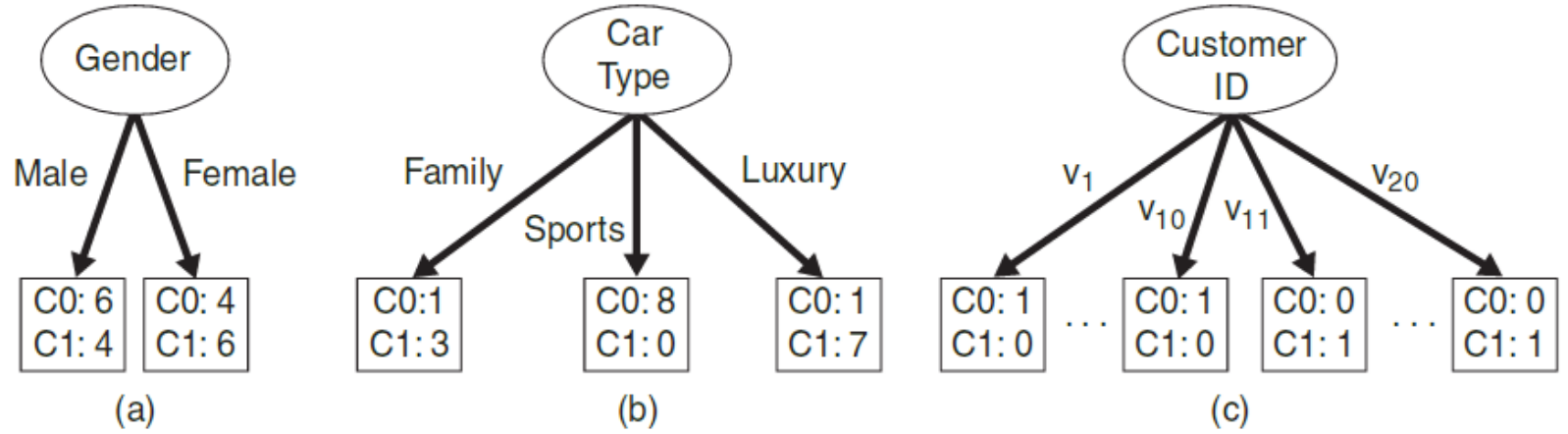
Árvore de Decisão

- (p0, p1), onde p1 = 1 - p0
- Qual é a distribuição de classes **antes** da divisão?



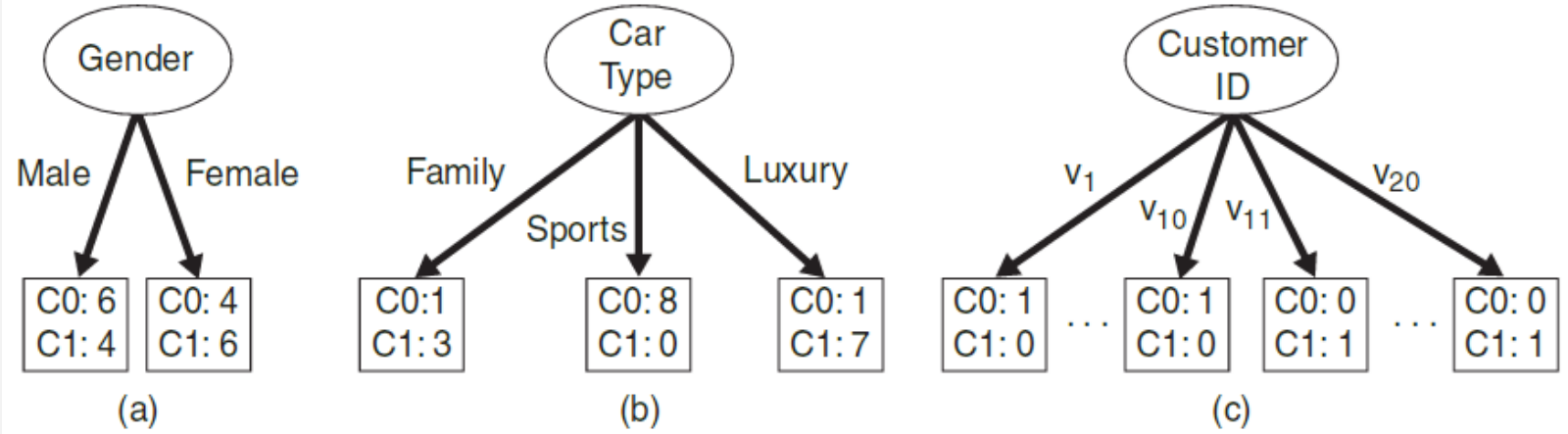
Árvore de Decisão

- (p0, p1), onde p1 = 1 - p0
- Qual é a distribuição de classes **<u>após</u>** a divisão?



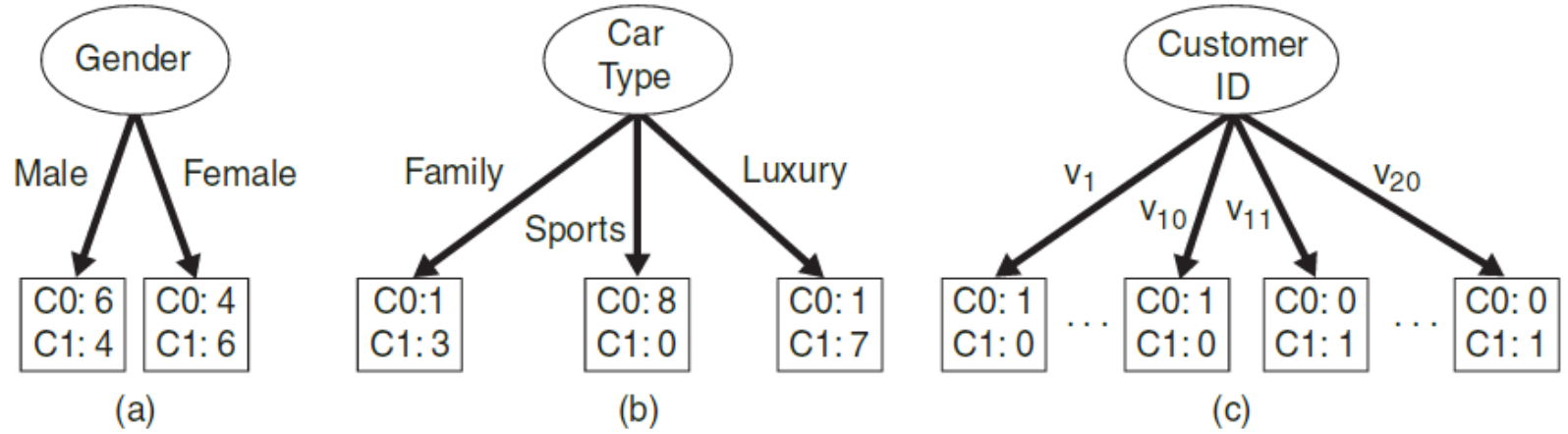
Árvore de Decisão

- Qual destas divisões produz um resultado com nós filhos mais **puros**?



Árvore de Decisão

- Objetivo: **minimizar** a **impureza** com a divisão.

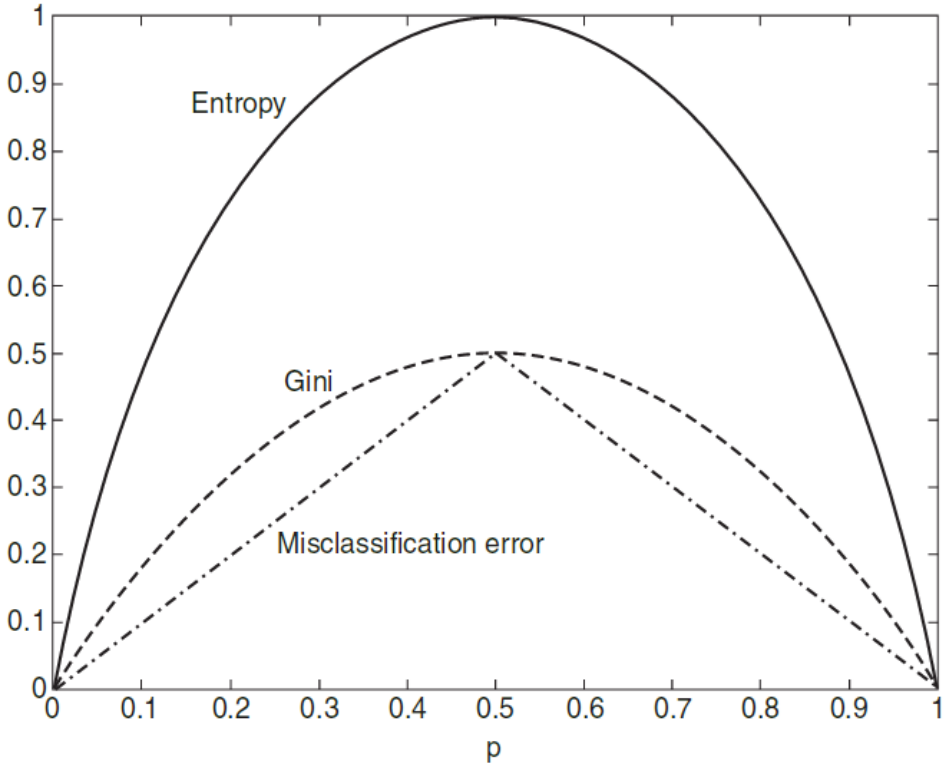


Árvore de Decisão

- Medidas para selecionar a melhor divisão
- Obs.: Considerar 0 * $\log_2 0 = 0$

$$\begin{aligned}
\text{Entropy}(t) &= -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t), \\
\text{Gini}(t) &= 1 - \sum_{i=0}^{c-1} [p(i|t)]^2, \\
\text{Classification error}(t) &= 1 - \max_i [p(i|t)],
\end{aligned}$$

Árvore de Decisão



Árvore de Decisão

Nó N1	Quantidade
Classe 0	**0**
Classe 1	**6**

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)],$$

Gini	
Entropia	
Erro	

Árvore de Decisão

Nó N1	Quantidade
Classe 0	**1**
Classe 1	**5**

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)],$$

Gini	
Entropia	
Erro	

Árvore de Decisão

Nó N1	Quantidade
Classe 0	**3**
Classe 1	**3**

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t),$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\text{Classification error}(t) = 1 - \max_i [p(i|t)],$$

Gini	
Entropia	
Erro	

Árvore de Decisão

- Ganho, Delta, para decidir o quão bom é uma divisão

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

- Onde: I(.) é a impureza de um dado nó;
- N é o total de instâncias no nó pai;
- k é o número de valores de um atributo;
- $N(v_j)$ é o número de instâncias associados com o nó filho, v_j.
- Quanto maior o Delta, melhor a divisão.

Árvore de Decisão

	Nó Pai
Classe 0	**6**
Classe 1	**6**
Gini=0,5	

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

Árvore de Decisão

	Nó Pai
Classe 0	**6**
Classe 1	**6**
Gini=0,5	

Atributo "A"

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

	Nó Filho 1
Classe 0	**4**
Classe 1	**3**
Gini=?	

	Nó Filho 2
Classe 0	**2**
Classe 1	**3**
Gini=?	

Ganho?	

Árvore de Decisão

	Nó Pai
Classe 0	**6**
Classe 1	**6**
Gini=0,5	

Atributo "B"

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

	Nó Filho 1
Classe 0	**1**
Classe 1	**4**
Gini=?	

	Nó Filho 2
Classe 0	**5**
Classe 1	**2**
Gini=?	

Ganho?	

Calcular Gini e Ganho

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2,$$

$$\Delta = I(\text{parent}) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j),$$

	Tipo de carro	
	{Esporte Luxo}	{Famíliar}
C 0	9	1
C1	7	3
Gini		

	Tipo de carro	
	{Esporte}	{Luxo, Famíliar}
C 0	8	2
C1	0	10
Gini		

	Tipo de carro		
	{Familiar}	{Esporte}	{Luxo}
C 0	1	8	1
C1	3	0	7
Gini			

ID	Proprietário de imóvel	Estado civil	Renda Annual	Calote
1	Sim	Solteiro	125k	Não
2	Não	Casado	100k	Não
3	Não	Solteiro	70k	Não
4	Sim	Casado	120k	Não
5	Não	Divorciado	95k	Sim
6	Não	Casado	60k	Não
7	Sim	Divorciado	220k	Não
8	Não	Solteiro	85k	Sim
9	Não	Casado	75k	Não
10	Não	Solteiro	90k	sim

Árvore de Decisão

Class	No	No	No	Yes	Yes	Yes	No	No	No	No
	Annual Income									
Sorted Values →	60	70	75	85	90	95	100	120	125	220
Split Positions →	55	65	72	80	87	92	97	110	122	172	230
	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >	<= >
Yes	0 3	0 3	0 3	0 3	1 2	2 1	3 0	3 0	3 0	3 0	3 0
No	0 7	1 6	2 5	3 4	3 4	3 4	3 4	4 3	5 2	6 1	7 0
Gini	0.420	0.400	0.375	0.343	0.417	0.400	*0.300*	0.343	0.375	0.400	0.420

Árvore de Decisão

- Como evitar que um atributo com alta diversidade de valores interfira de forma negativa?
- Permitir apenas divisões binárias (CART)
- Utilizar razão de ganho (Gain ratio), que penaliza quando há muitas divisões

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times log_2(\frac{|D_j|}{|D|})$$

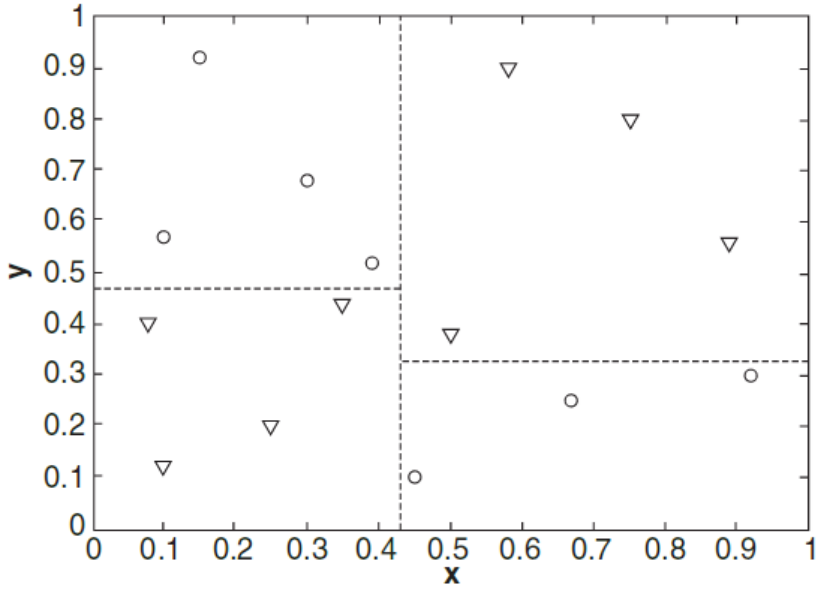
$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

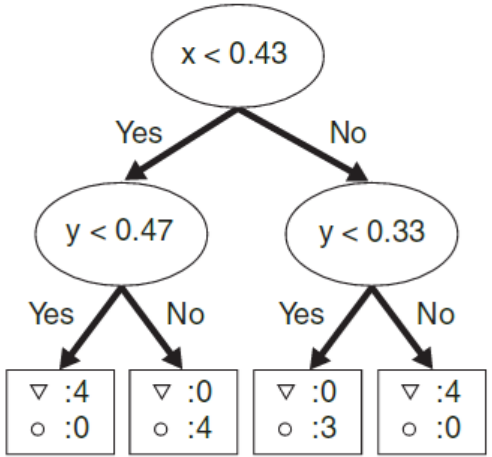
Árvore de Decisão

- Abordagem não-paramétrica
- Encontrar árvore ótima é um problema de alta complexidade
- O processo de construção da árvore é rápido, idem para o teste
- Fáceis de interpretar, inclusive por leigos

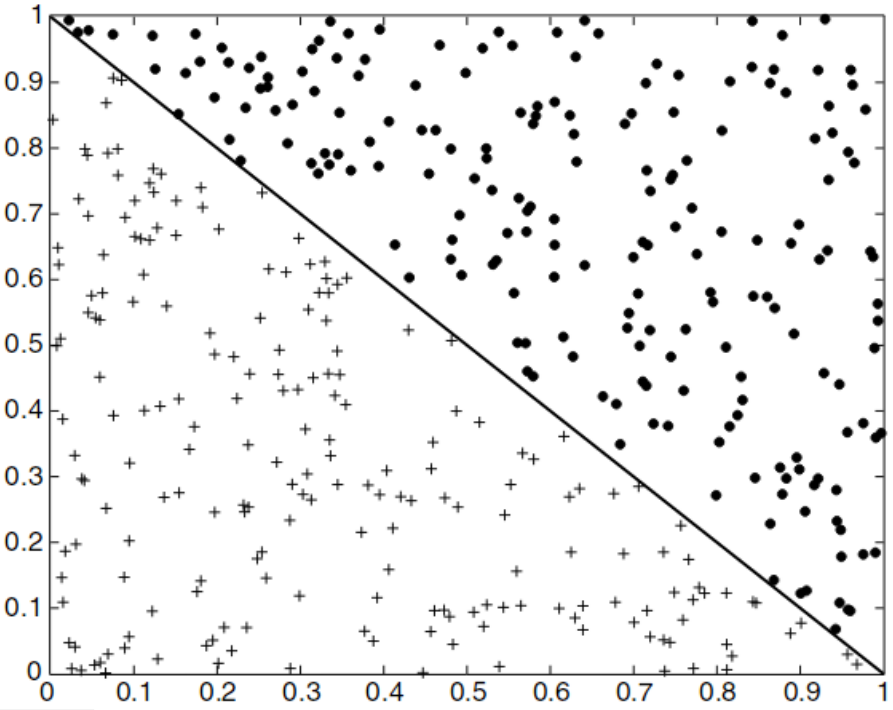
- Robustas à presença de ruídos
- Atributos redundantes não trazem grandes prejuízos para a acurácia
- Podem ter profundidade excessivamente grande
- Pode haver replicação de trechos da árvore (poda)
- Fronteiras de decisão são hiperplanos

Fronteiras de Decisão

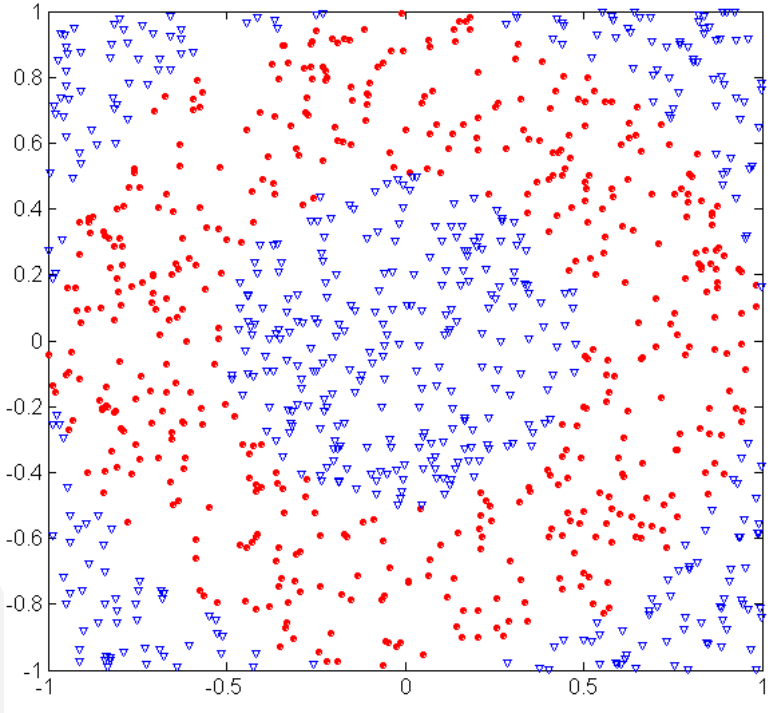




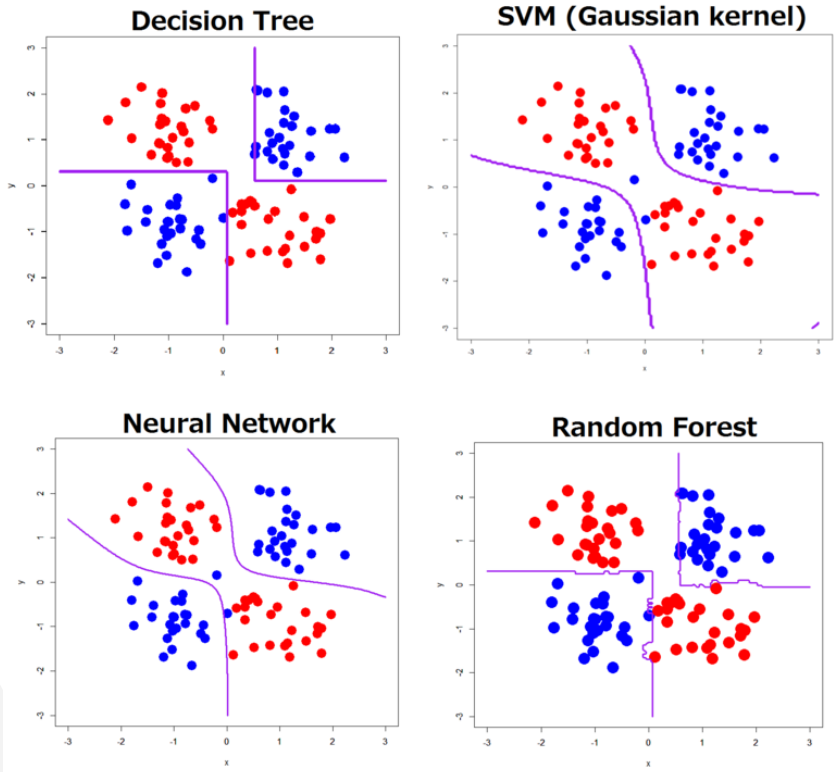
Fronteiras



Fronteiras



Fronteiras


Decision Tree
SVM (Gaussian kernel)
Neural Network
Random Forest

Exercício

Exercício

- Utilize o conjunto de dados “adult” que trata do censo de 1995 nos Estados Unidos.
- A classe é renda anual maior que 50k ou menor.
- Os dados para teste estão no arquivo “**adult.test**”.
- Meça a acurácia. Lide com dados categóricos.

```
from sklearn import tree
clf = tree.DecisionTreeClassifier()
clf = clf.fit(X, Y)
print clf.predict(X_test)
```

Exercício

base.columns = ['age', 'workclass', 'fnlwgt', 'education', 'education-num', 'marital-status', 'occupation', 'relationship', 'race', 'sex', 'capital-gain', 'capital-loss', 'hours-per-week', 'native-country', 'class']

Exercício

```
base['age'] = base['age'].astype(float)
base['workclass'] = base['workclass'].astype('category')
base['fnlwgt'] = base['fnlwgt'].astype(float)
base['education'] = base['education'].astype('category')
base['education-num'] = base['education-num'].astype(float)
base['marital-status'] = base['marital-status'].astype('category')
base['occupation'] = base['occupation'].astype('category')
base['relationship'] = base['relationship'].astype('category')
```

Exercício

```
base['race'] = base['race'].astype('category')
base['sex'] = base['sex'].astype('category')
base['capital-gain'] = base['capital-gain'].astype(float)
base['capital-loss'] = base['capital-loss'].astype(float)
base['hours-per-week'] = base['hours-per-week'].astype(float)
base['native-country'] = base['native-country'].astype('category')
base['class'] = base['class'].astype('category')
```

Exercício

class sklearn.tree.DecisionTreeClassifier(**criterion**='gini', **splitter**='best', **max_depth**=None, **min_samples_split**=2, **min_samples_leaf**=1, **min_weight_fraction_leaf**=0.0, **max_features**=None, **random_state**=None, **max_leaf_nodes**=None, **class_weight**=None, **presort**=False)