Inteligência Artificial

Aprendendo a partir de Exemplos

Prof. Fabio Augusto Faria

2º semestre 2022

UNIFESP
UNIVERSIDADE FEDERAL DE SÃO PAULO
1933

Tópicos

- Aprendizagem de Máquina
- Aprendizagem a partir de observações
- Tipos de Aprendizagem (feedback)
- Árvores de Decisão
- Avaliação de Desempenho

Aprendizagem de Máquina

- Aprendizagem de máquina é uma linha de pesquisas dentro da área de Inteligência Artificial focada em construir modelos que **aprendam** por meio de exemplos/dados e melhorar a acurácia na aplicação alvo.
- Aprendizagem a partir de observações ou aprendizagem indutiva.
- Mapeia os exemplos (*x) amostrados de uma distribuição desconhecida (***função desconhecida)** para saídas (y)
- Objetivo é encontrar uma hipótese (h) que melhor se ajuste a função desconhecida (f) dentre infinitas hipóteses no espaço de hipóteses (H).

Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo

Espera-se que:

f(x) = h(x)

para diferentes exemplos amostrados

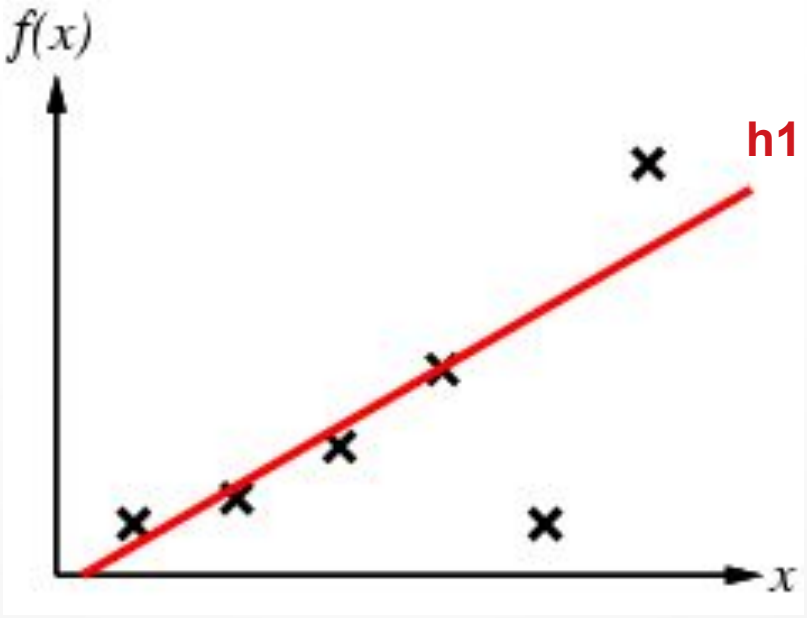
“Melhor Ajuste”

Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo

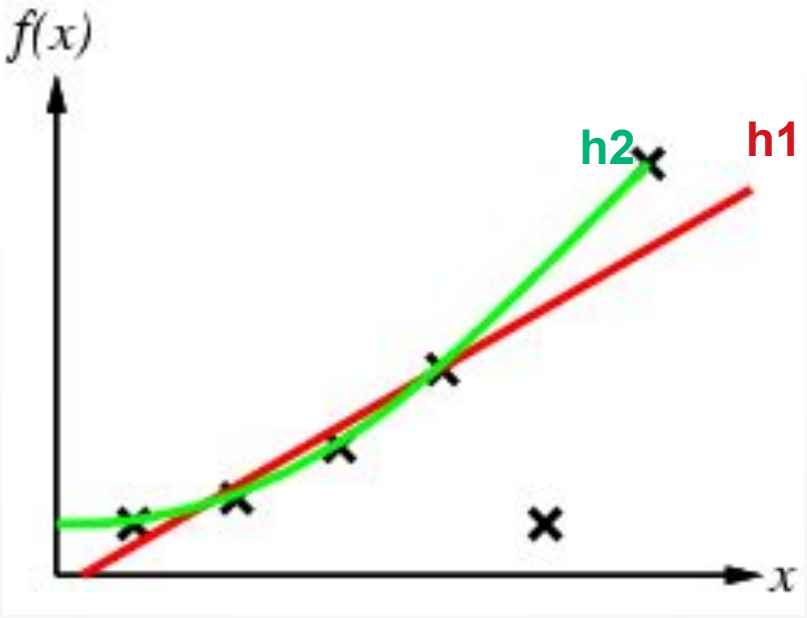
Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo



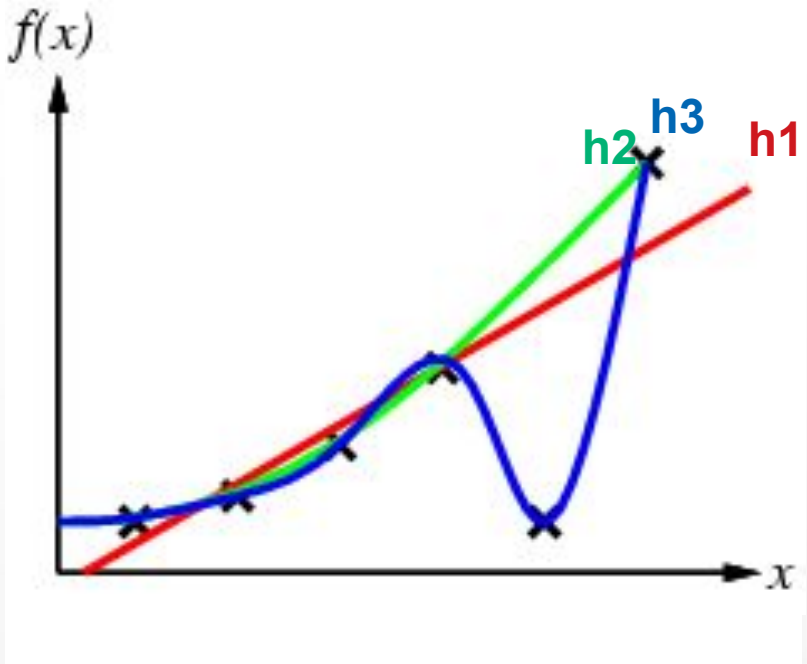
Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo



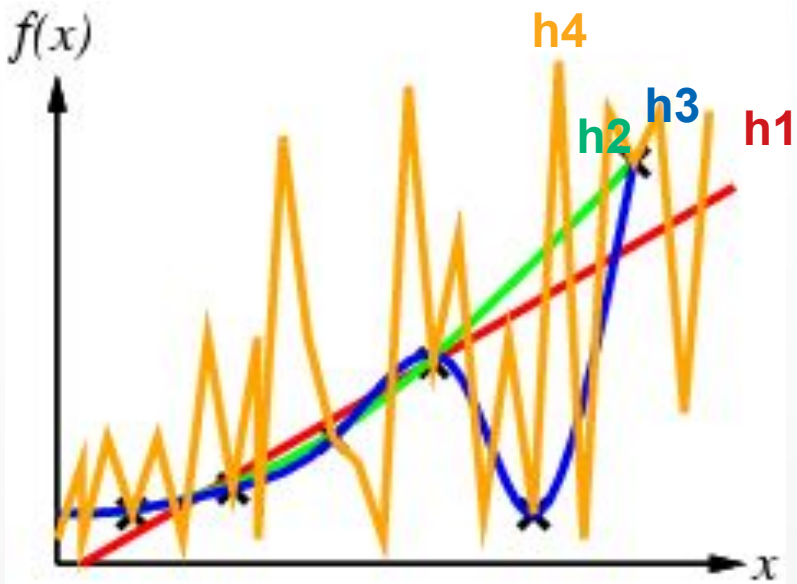
Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo



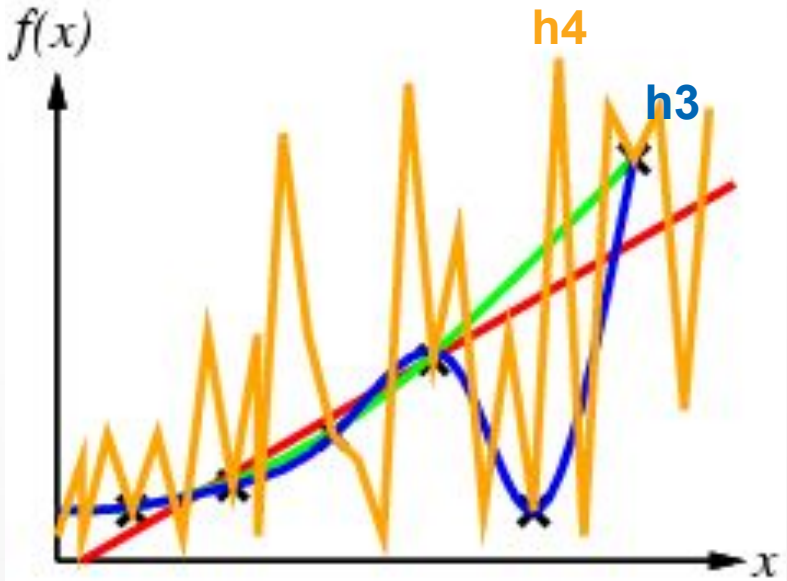
Aprendizagem Indutiva

- Dada uma coleção de exemplos de ***f***, retornar uma função ***h*** que se aproxime de ***f***.
- **x** são exemplos de entrada
- ***h*** é a função de mapeamento buscada
- **f** é a função alvo/objetivo



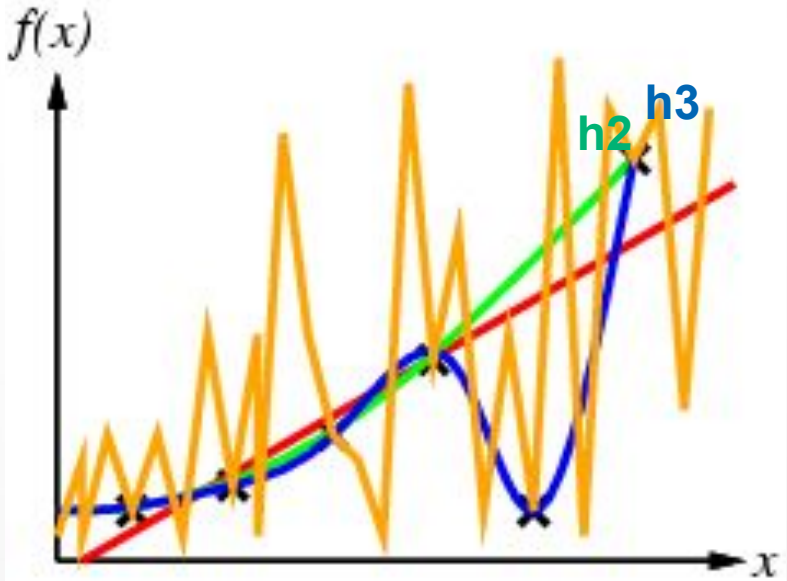
Aprendizagem Indutiva

- Diferente hipótese ***h*** tem um diferente ajuste aos exemplos amostrados
- Uma ***h que "passa" por todos os exemplos é chamada de*** **consistente**
- **h3 e h4**



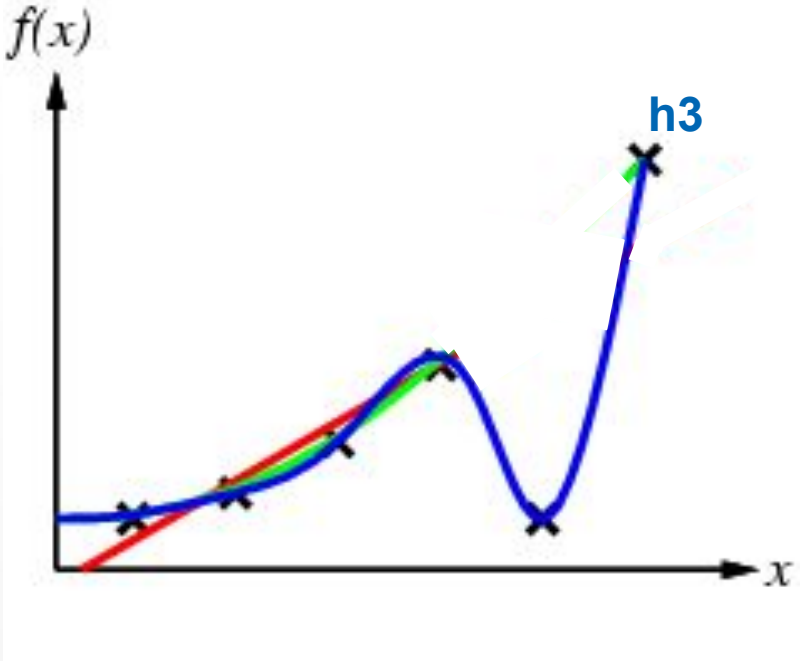
Aprendizagem Indutiva

- Qual é a melhor hipótese h dado uma conjunto de diferentes hipóteses existentes no espaço de hipóteses H?
  - **Lâmina de Ockam**: escolher a hipótese consistente ***<u>mais simples</u>***



Aprendizagem Indutiva

- Qual é a melhor hipótese h dado uma conjunto de diferentes hipóteses existentes no espaço de hipóteses H?
  - **Lâmina de Ockam**: escolher a hipótese consistente ***mais simples***
  - *Hipóteses mais complexas estão mais ajustadas aos dados amostrados e tem menor poder generalização (menos geral).*



Aprendizagem Indutiva

- A possibilidade de encontrar uma hipótese **simples e consistente** depende do espaço de hipótese H fornecido pelo **algoritmo** de aprendizagem utilizado e a busca pelos **parâmetros**.
- Um problema de aprendizagem é **realizável** se o espaço de hipótese contém a **função objetivo**

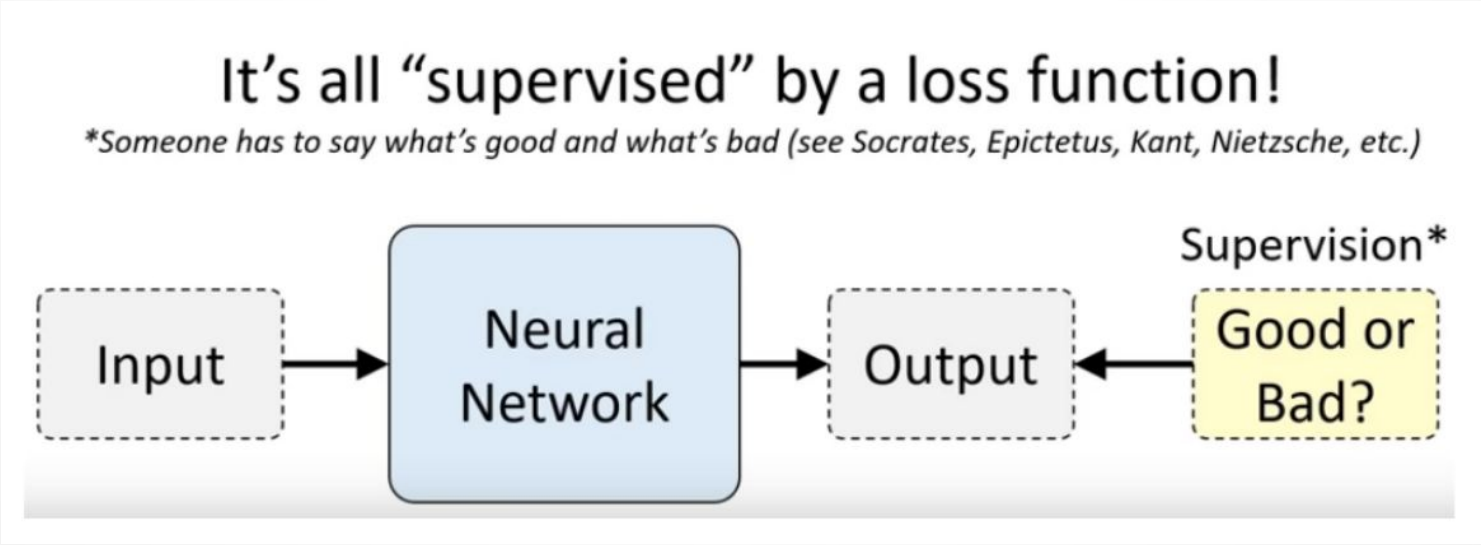
H = { h1, h2, h3, h4 }

f(x) = h3

“PROBLEMA REALIZÁVEL”

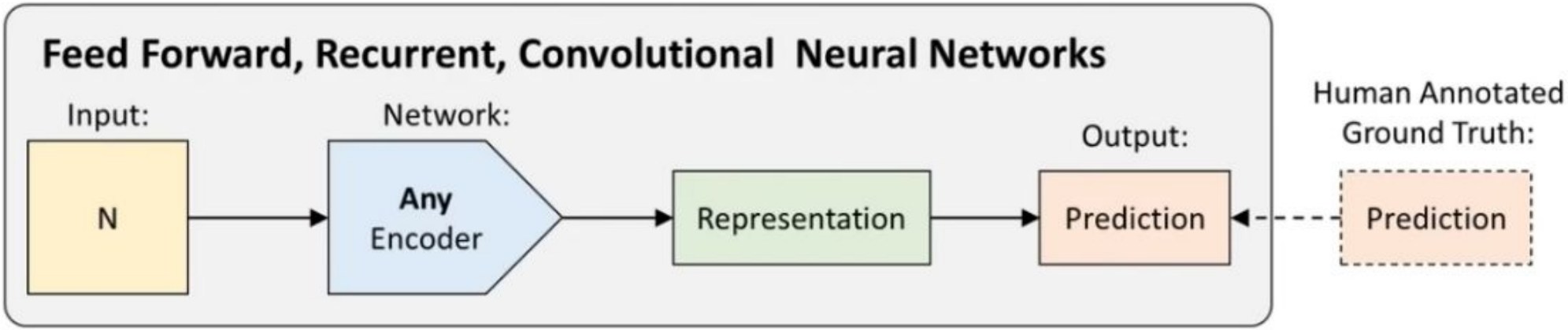
Tipos de Aprendizado

- Supervisionado
- Não Supervisionado
- Semi-Supervisionado
- Por Reforço



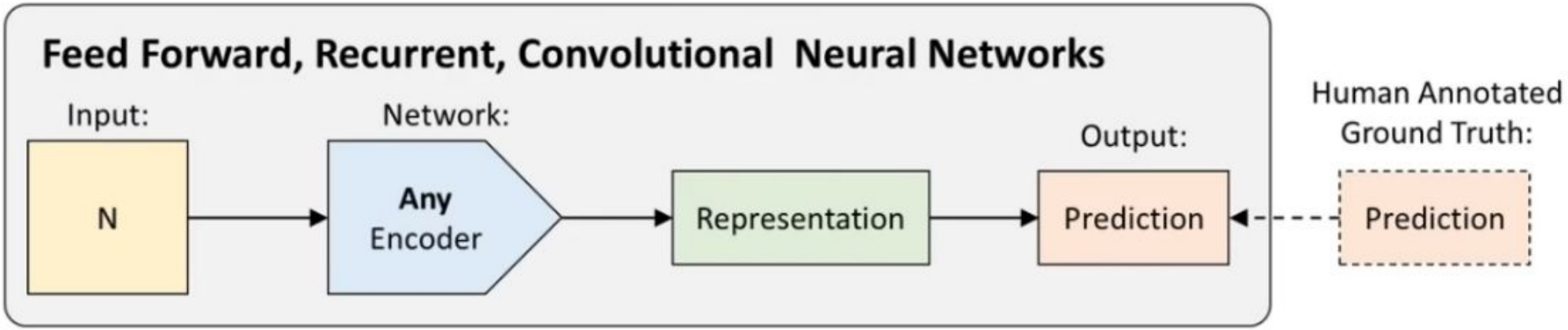
Aprendizado Supersionado

- Ensinado por exemplos amostrados **mapeando** x → y
- x são exemplos de entrada
- y são as saídas (verdades)
- y com valores discreto/categóricos é tarefa de Classificação
- y com valores contínuo é tarefa de Regrassão



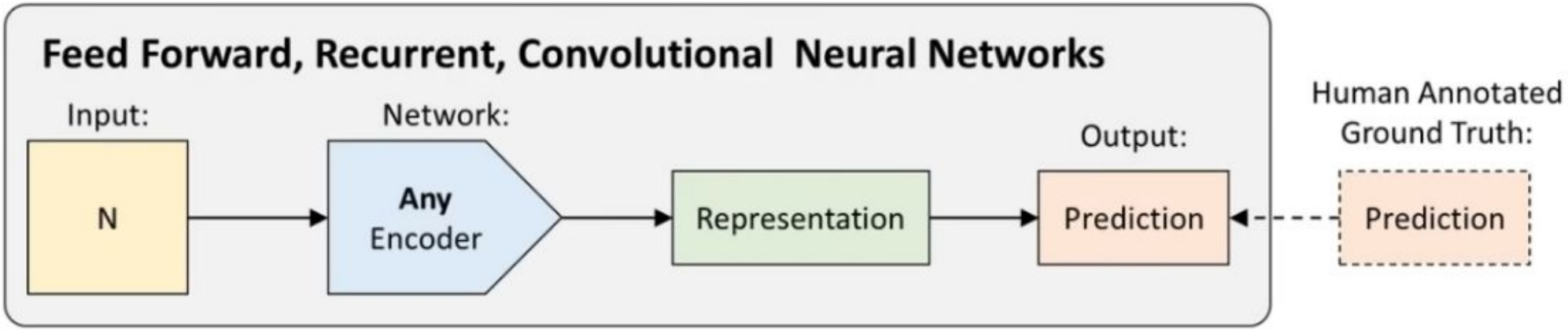
Aprendizado Não-Supersionado

- Ensinado por exemplos amostrados entendendo o comportamento de x, **sem qualquer informação** sobre y
- Tarefa de Agrupamento, Recuperação, Regras de Associação.



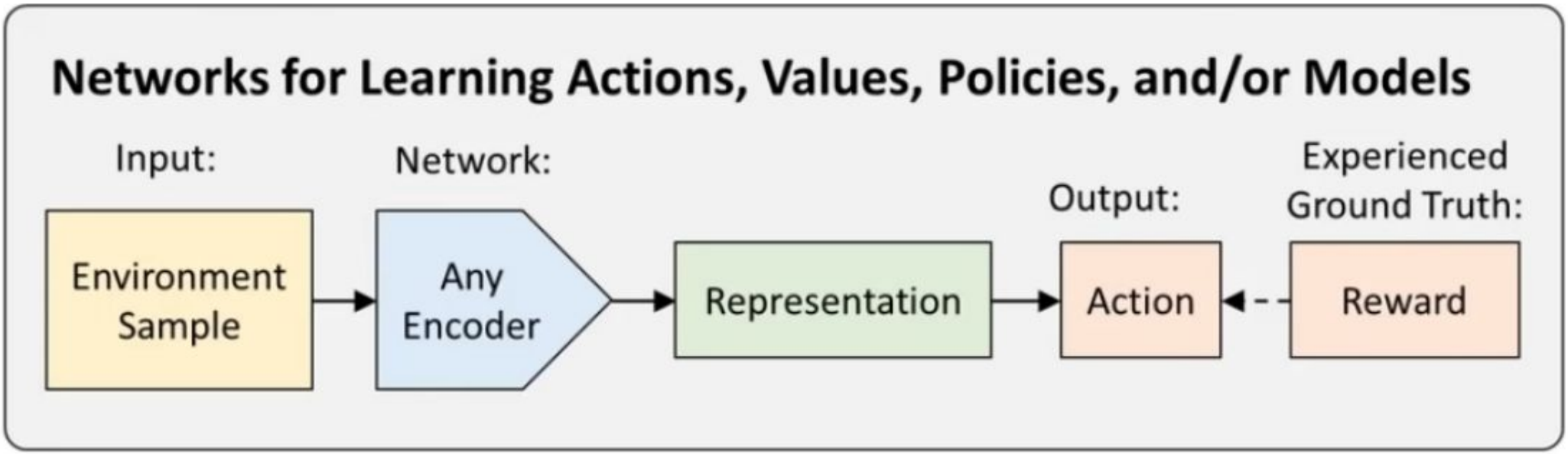
Aprendizado Semi-Supersionado

- Utiliza-se de exemplos x **com e sem** informação sobre o y correspondente
- Exemplos com informação de y é **<<** que exemplos sem informação
- Por exemplo: Aprendizagem Transductiva



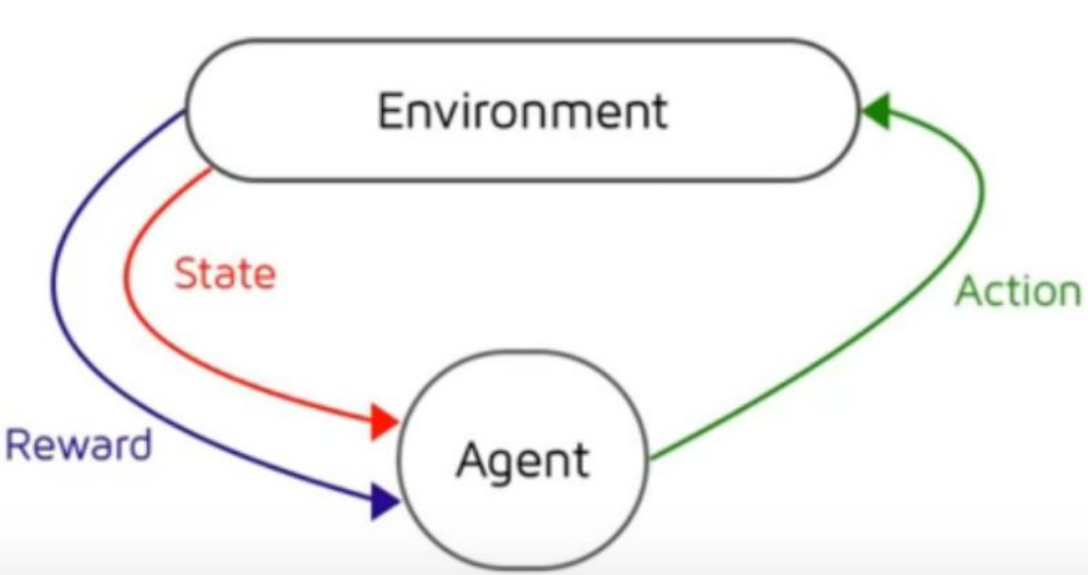
Aprendizado por Reforço

- Ensinado por meio de **experiência**
- Função de Perda recebe como entrada as informações sobre o **efeito** que uma **ação** causou no **ambiente** e calcula a "distância" para o objetivo final.



Aprendizado por Reforço

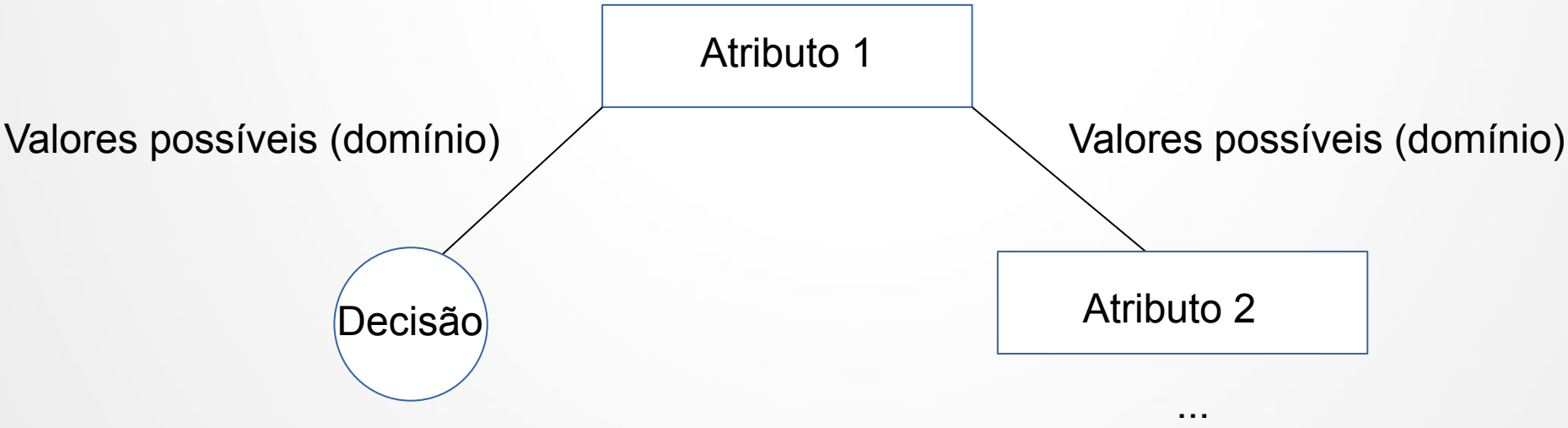
- Em cada passo, o agente:
  1. Executa ação
  2. Observa no estado
  3. Recebe Recompensa/Punição



Árvores de Decisão

- **Indução de árvores de decisão:** um dos algoritmos mais simples e bem sucedidos em aprendizagem de máquina.

- **Árvore de decisão**: toma como entrada um objeto ou situação descritos por um conjunto de **atributos** e retorna uma decisão -- valor de **saída** previsto, dada uma **entrada**.



Árvores de Decisão

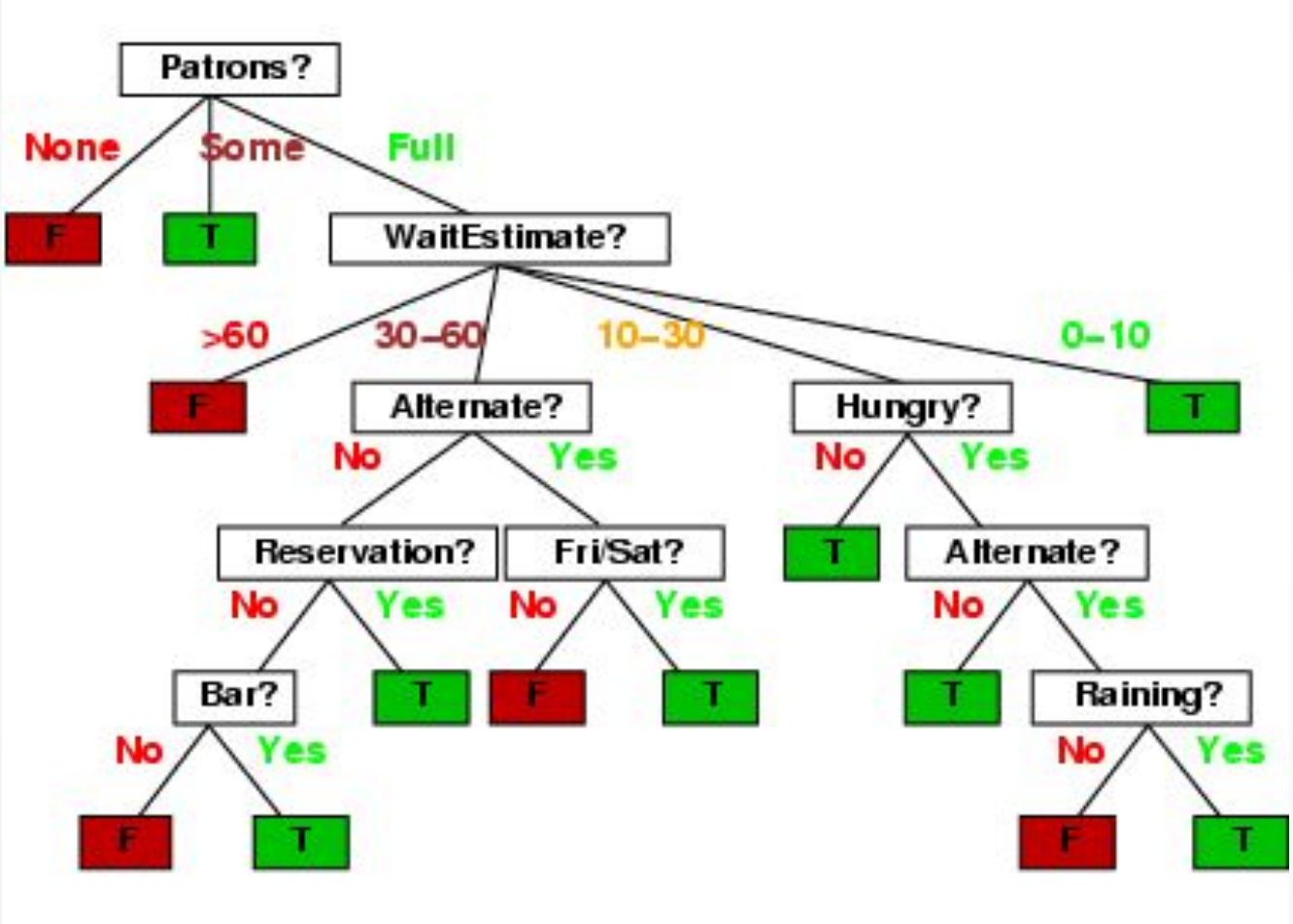
- Exemplo: decidir se devo esperar por uma mesa em um restaurante, dados os atributos:
  - ***Alternate***: há um restaurante alternativo na redondeza?
  - ***Bar***: existe um bar confortável onde se esperar?
  - ***Fri/Sat***: hoje é sexta ou sábado ?
  - ***Hungry***: estou com fome?
  - ***Patrons***: numero de pessoas no restaurante (***None***, ***Some***, ***Full***)
  - ***Price***: faixa de preços ($, $$, $$$)
  - ***Raining***: está a chover?
  - ***Reservation***: temos reserva?
  - ***Type***: tipo do restaurante (***French***, ***Italian***, ***Thai***, ***Burger***)
  - ***WaitEstimate***: tempo de espera estimado (0-10, 10-30, 30-60, >60)

Árvores de Decisão

- Exemplos descritos por valores de atributos (discretos, ou contínuos)
- E.g., exemplos de situações em que o autor do livro não esperará por uma mesa: Classificação/Decisão para cada exemplos é positivo (T) ou negativo (F)

Example	Attributes										Target
	Alt	*Bar*	*Fri*	*Hun*	*Pat*	*Price*	*Rain*	*Res*	*Type*	*Est*	*Wait*
X_1	T	F	F	T	Some	$$$	F	T	French	0–10	T
X_2	T	F	F	T	Full	$	F	F	Thai	30–60	F
X_3	F	T	F	F	Some	$	F	F	Burger	0–10	T
X_4	T	F	T	T	Full	$	F	F	Thai	10–30	T
X_5	T	F	T	F	Full	$$$	F	T	French	>60	F
X_6	F	T	F	T	Some	$$	T	T	Italian	0–10	T
X_7	F	T	F	F	None	$	T	F	Burger	0–10	F
X_8	F	F	F	T	Some	$$	T	T	Thai	0–10	T
X_9	F	T	T	F	Full	$	T	F	Burger	>60	F
X_{10}	T	T	T	T	Full	$$$	F	T	Italian	10–30	F
X_{11}	F	F	F	F	None	$	F	F	Thai	0–10	F
X_{12}	T	T	T	T	Full	$	F	F	Burger	30–60	T

Árvores de Decisão



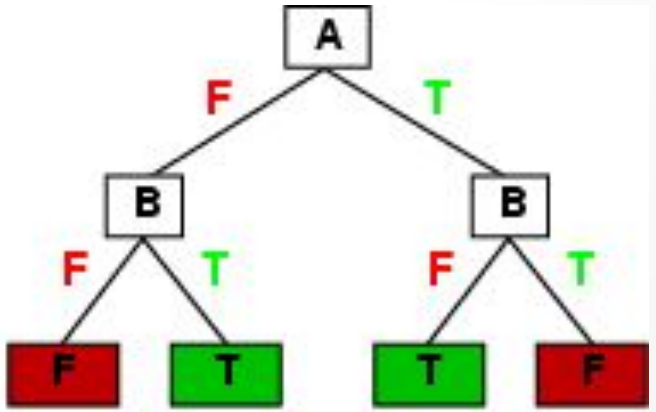
Árvores de Decisão

- Uma árvore de decisão chega a uma decisão executando uma **sequência** de testes.
- Cada **nó interno** da árvore corresponde a um **teste** do valor de uma das propriedades, e as ramificações a partir do nó são identificadas com os valores possíveis do teste.
- Cada **nó de folha** especifica o **valor** a ser retornado se aquela folha for alcançada.

Árvores de Decisão

- Qualquer função booleana pode ser escrita como uma árvore de decisão
- Trivialmente, há uma árvore de decisão consistente para qualquer conjunto de treinamento com um caminho para cada exemplo.
- Isso seria uma representação exponencialmente grande
- Devemos procurar por árvores de decisão mais **compactas**.

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F

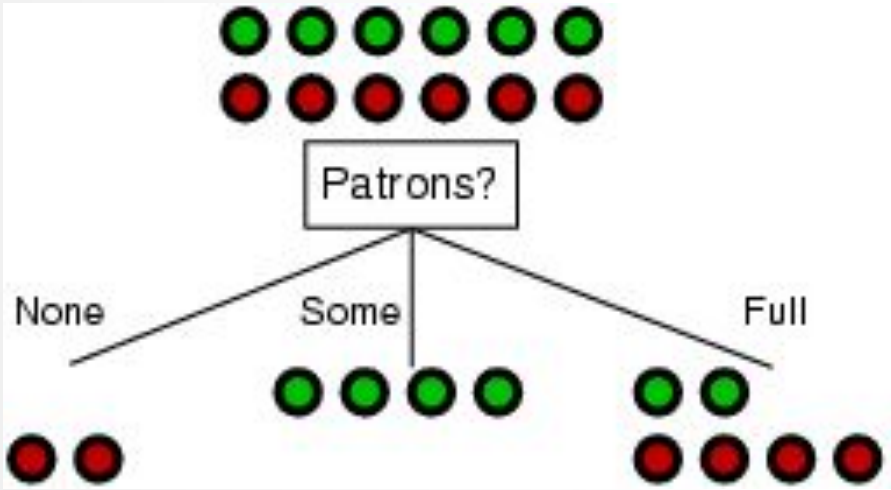


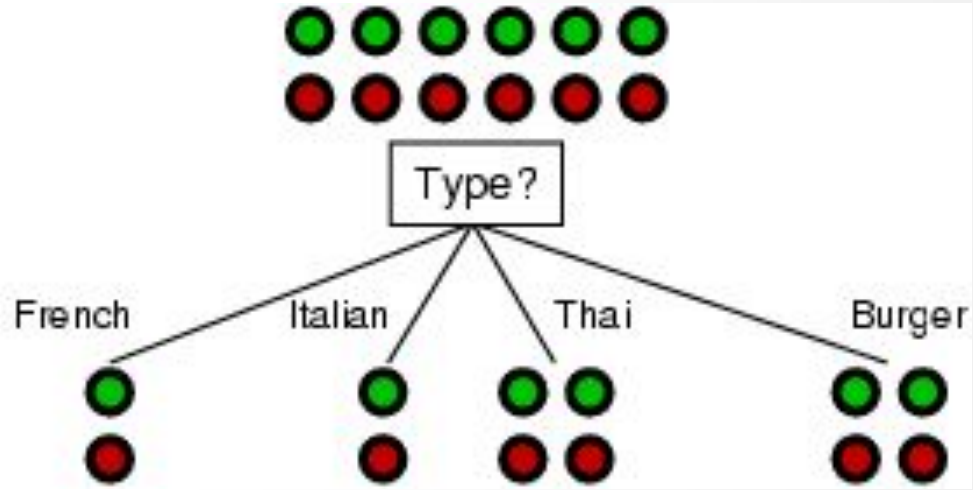
Árvores de Decisão

- Se existem alguns exemplos positivos e alguns negativos, escolha o **melhor** atributo para dividi-los.
- Se todos os exemplos restantes forem positivos (ou todos negativos), então terminamos: podemos responder *Sim* ou *Não.*
- Se não resta nenhum exemplo, nenhum exemplo desse tipo foi observado. Então retorna-se um valor-padrão calculado a partir da classificação de maioria no pai do nó.

Árvores de Decisão

- Idéia: um bom atributo divide os exemplos em subconjuntos que (preferivelmente) são “todos positivos" ou ”todos negativos"





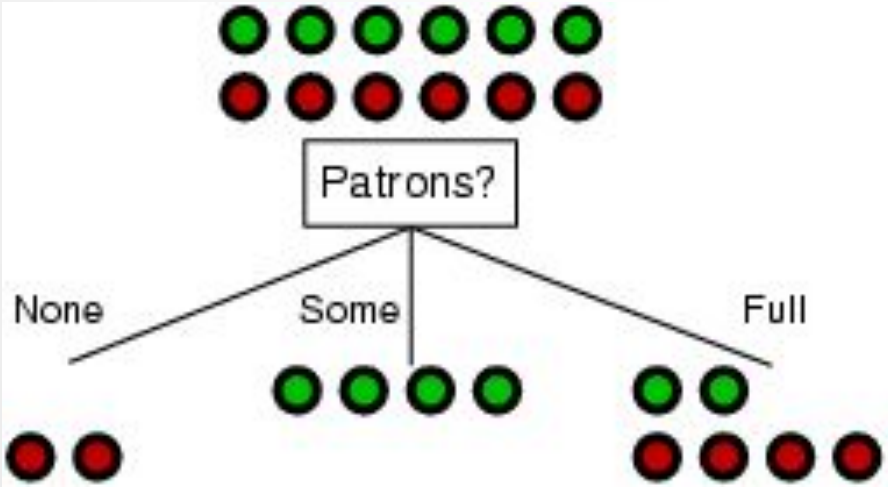
- *Patrons?* é um atributo melhor do que *Type* para ser raiz.

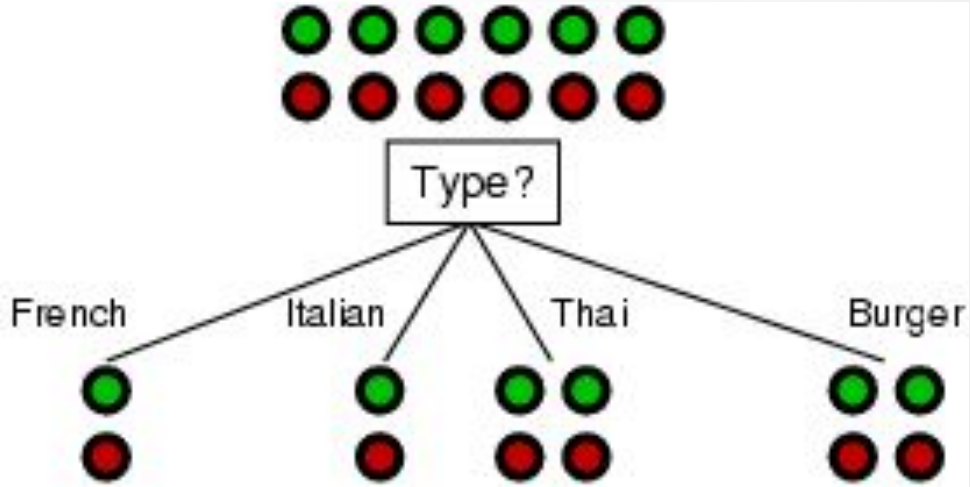
Qual é o melhor atributo?

- A escolha de atributos deve minimizar a profundidade da árvore de decisão
- Escolher um atributo que vá o mais longe possível na classificação exata de exemplos
- Um atributo perfeito divide os exemplos em conjuntos que são todos positivos ou todos negativos
- Solução: medir os atributos a partir da quantidade esperada de informações fornecidas por ele.

Qual é o melhor atributo?

- O atributo “patrons” não é perfeito, mas é o melhor que o atributo “type”
- Precisamos definir uma medida formal de escolha de atributo em “bom” e “ruim”.
- A medida deve ter seu valor máximo quando o atributo for perfeito e seu valor mínimo quando o atributo for inútil.





Ganho de Informação

- Dado um atributo A, podemos medir quanto de informação ainda precisamos *depois* de escolhê-lo;
- Qqr atributo A divide o conjunto de treinamento E em subconjuntos E_1, … , E_v de acordo com seus valores para A, onde A pode ter v valores distintos.
- Cada subconjunto E_i tem p_i exemplos positivos e n_i exemplos negativos;
- Assim seguindo essa ramificação, precisaremos de
  - $I(p_i/(p_i + n_i), n_i/(p_i + n_i))$
- bits de informação para responder a pergunta.

Ganho de Informação

Um exemplo escolhido ao acaso a partir do conjunto de treinamento tem o i-esimo valor para o atributo com probabilidade $(p_i + n_i)/(p+n)$ (“peso do atributo”).

e assim, em média, depois de testar o atributo A, temos:

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

bits de informação para classificar o exemplo...

Ganho de Informação

Um exemplo escolhido ao acaso a partir do conjunto de treinamento tem o i-esimo valor para o atributo com probabilidade $(p_i + n_i)/(p+n)$ ("peso do atributo").

e assim, em média, depois de testar o atributo A, temos:

$$remainder(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p+n} I\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

bits de informação para classificar o exemplo...

"Dentre os exemplos deste atributo, qual é o grau de discernimento dele."

Ganho de Informação

O ganho de informação a partir do teste deste atributo é a diferença entre o *requisito de informações original* e o *novo requisito*:

$$IG(A) = I(\frac{p}{p+n}, \frac{n}{p+n}) - remainder(A)$$

A heurística é então escolher o atributo com o maior valor de ganho de informação (IG).

Ganho de Informação

Para o conjunto de treinamento, $p = n = 6$, $I(6/12, 6/12) = 1$ bit

Considerando os atributos *Patrons* e *Type*, e os outros tb:

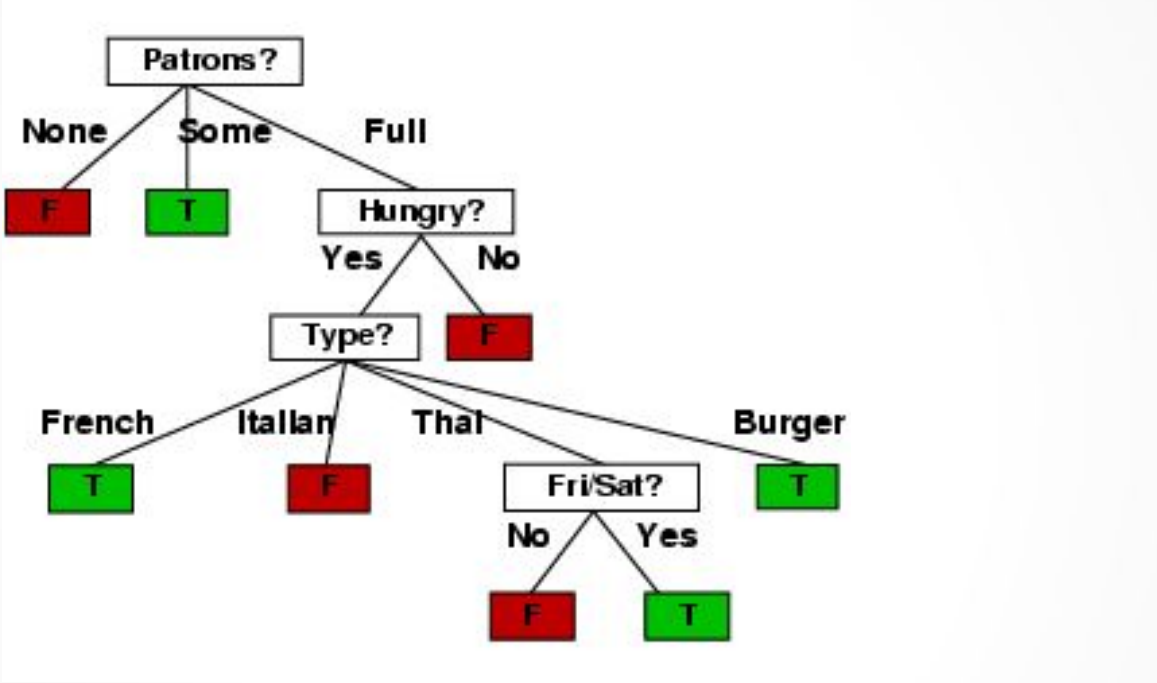
$$IG(Patrons) = 1 - [\frac{2}{12} I(0,1) + \frac{4}{12} I(1,0) + \frac{6}{12} I(\frac{2}{6}, \frac{4}{6})] = .0541 \text{ bits}$$

$$IG(Type) = 1 - [\frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{2}{12} I(\frac{1}{2}, \frac{1}{2}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4}) + \frac{4}{12} I(\frac{2}{4}, \frac{2}{4})] = 0 \text{ bits}$$

Patrons possui o maior valor de IG de todos os atributos e, portanto, é escolhido como raiz da árvore de decisão aprendida pelo algoritmo DTL

Resolvendo o Problema

Árvore de decisão aprendida dos 12 exemplos:

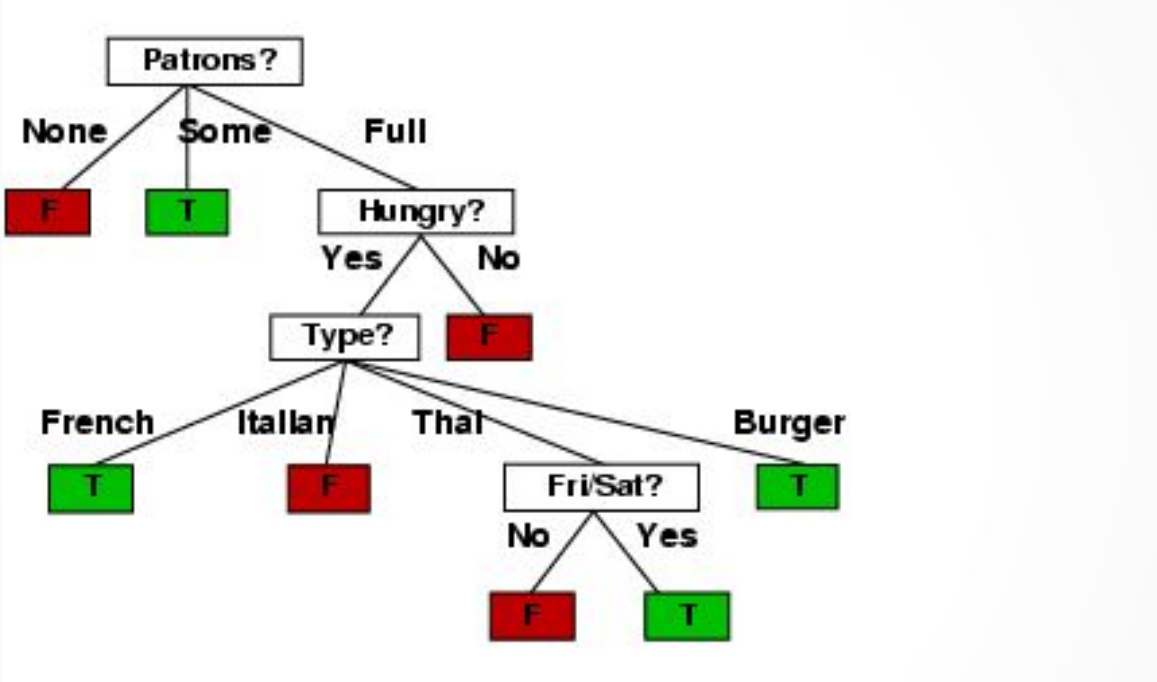


Substancialmente mais simples do que a árvore “verdadeira”;

Não há nenhuma razão para uma solução mais complexa (e.g incluindo *Rain* e *Res*), pois todos os exemplos já foram classificados.

Resolvendo o Problema

Árvore de decisão aprendida dos 12 exemplos:



Com mais exemplos seria possível induzir uma árvore mais semelhante à árvore original;

Esta árvore nunca viu um exemplo de espera 0-10

portanto, pode cometer um engano...

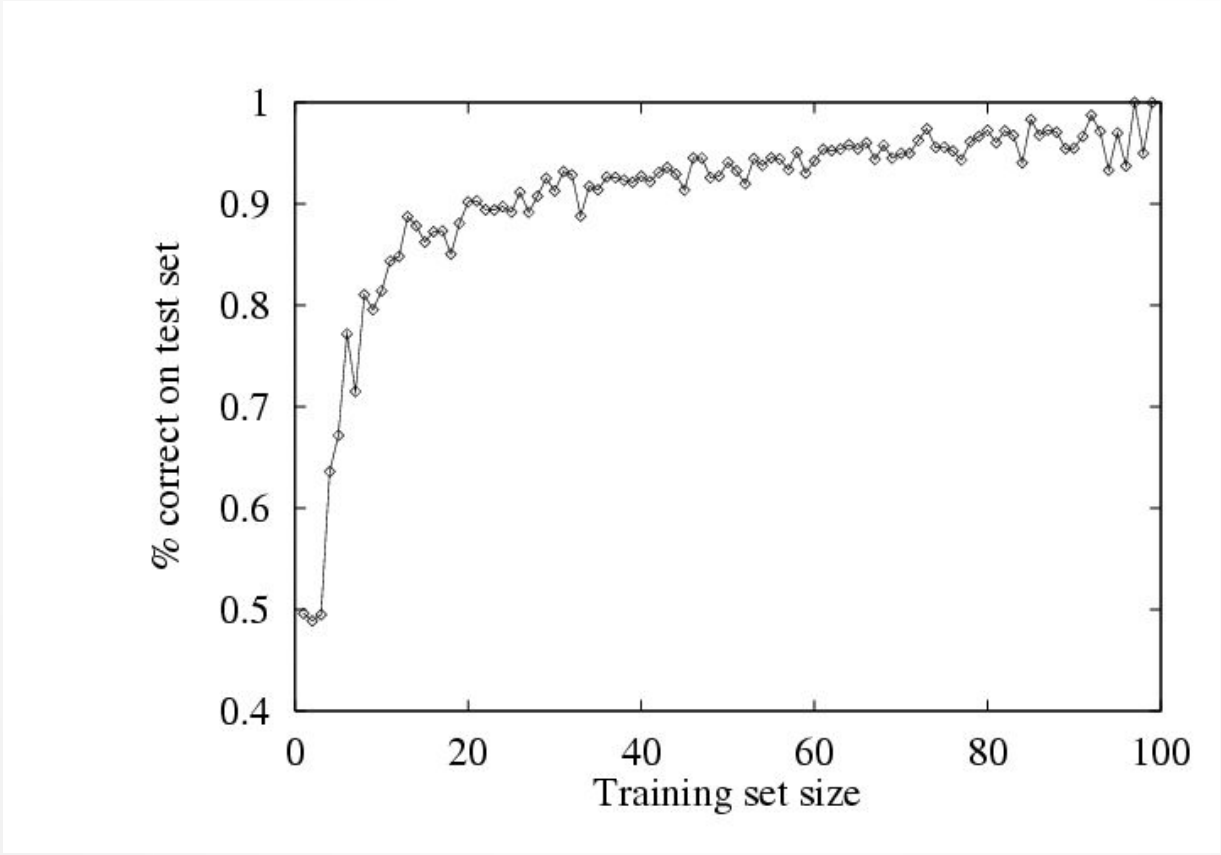
–Como avaliar o desempenho do algoritmo?

Avaliação de Desempenho

- Como sabemos que $h \approx f$?
- Um algoritmo de aprendizagem é bom se produz **hipóteses** que fazem um bom trabalho de previsão de classificações de exemplos não vistos
- **Método de validação:**
  - Coletar um número de exemplos que descreva o comportamento do **Mundo Real**;
  - Dividi-lo em **conjunto de treinamento** e **conjunto de teste;**
  - Aplicar o algoritmo de aprendizagem ao conjunto de treinamento, gerando uma hipótese ***h***
  - Medir a porcentagem de exemplos no conjunto de teste que são corretamente classificados por ***h***
  - Repetir as etapas 1 a 4 para diferentes tamanhos de conjuntos de treinamento e diferentes conjuntos de teste de cada tamanho selecionados aleatoriamente

Avaliação de Desempenho

- Experimente *h* em novos conjuntos de teste.
- Curva de aprendizagem = % de classificações corretas sobre o conjunto de teste como uma função do tamanho do conjunto de treinamento



Exercício – Data entrega 17/10/2021

1. Criar uma árvore de decisão utilizando da base de dados IRIS para classificação de espécies de flores. Siga o tutorial no link abaixo:

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

2. Variar os parâmetros abaixo, observando e reportando cada uma das árvores resultantes.
   a. Criterion = {“gini”, “entropy”}
   b. Splitter = {“best”, “random”}
   c. max_depth = {None,2, 4}
   d. min_samples_split = {2,10}

3. Existe diferença entre as árvores?

Referências

- Peter Norvig e Stuart Russel. Inteligência Artificial. Cap. 18. Seção 3.
- Alguns Slides da Profa. Bianca Zadrozny (IC-UFF)