APRENDIZAGEM BASEADA EM INSTÂNCIAS

Prof. Igor da Penha Natal

Aprendizagem baseada em instâncias

- Não se constrói uma descrição explícita e geral da hipótese, somente se armazena os exemplos.
- Cada vez que uma nova instância (exemplo) é questionada, o relacionamento dela com os outros exemplos armazenados é examinado para determinar o valor da função para o novo exemplo.
- Engloba:
  - Vizinho mais próximo (*nearest neighbor*)
  - Regressão localmente ponderada (*locally weighted regression*)
  - Raciocínio baseado em casos (*case-based reasoning*)

Vizinho mais próximo (*Nearest Neighbor)* e Regressão localmente ponderada

- São **abordagens diretas de aproximação** de funções de valores reais ou discretos.
- A aprendizagem destes métodos **consiste em armazenar os dados de treinamento**.
- Quando uma nova instância é encontrada, um conjunto de instâncias similares é recuperado da memória e usado para classificar a nova instância.
- Uma diferença básica destas abordagens é que **elas podem construir uma função de aproximação diferente para cada instância** que deve ser classificada.

Aprendizagem Baseada em casos

- Métodos baseados em instâncias podem usar uma representação simbólica mais complexa para as instâncias.
- Em **aprendizagem baseada em casos** as instâncias são representadas desta forma e o processo de identificar a "vizinhança" é elaborado de acordo.
- O **raciocínio baseado em casos** tem sido aplicado a tarefas como armazenamento e reuso de experiências passadas aplicadas a *help desk*, raciocínio sobre casos legais por se referir a casos anteriores, entre outros.
- Uma desvantagem da abordagem baseada em instâncias é que o custo de classificar um exemplo pode ser alto.
  - Porque quase toda a computação ocorre em tempo de classificação, ao invés de quando temos os exemplos de treino.

K-Vizinhos mais próximos (*K-Nearest Neighbor*)

- É o método mais básico de aprendizagem baseada em instâncias.
- Assume que todas as instâncias correspondem a pontos no espaço n-dimensional $\Re^n$.
- Os vizinhos mais próximos de uma instância são definidos em termos da **DISTÂNCIA EUCLIDIANA**.
- Mais precisamente, seja uma instância arbitrária ***x*** descrita pelo vetor de características ‹$a_1(x)$, $a_2(x)$, ... , $a_n(x)$›, onde $\boldsymbol{a_r(x)}$ denota o valor do r-ésimo atributo da instância ***x***.
- Então a distância entre duas instâncias $\boldsymbol{x_i}$ e $\boldsymbol{x_j}$ é definida como:

$$d(x_i, x_j) \equiv \sqrt{\sum_{r=1}^{n} \left(a_r(x_i) - a_r(x_j)\right)^2}$$

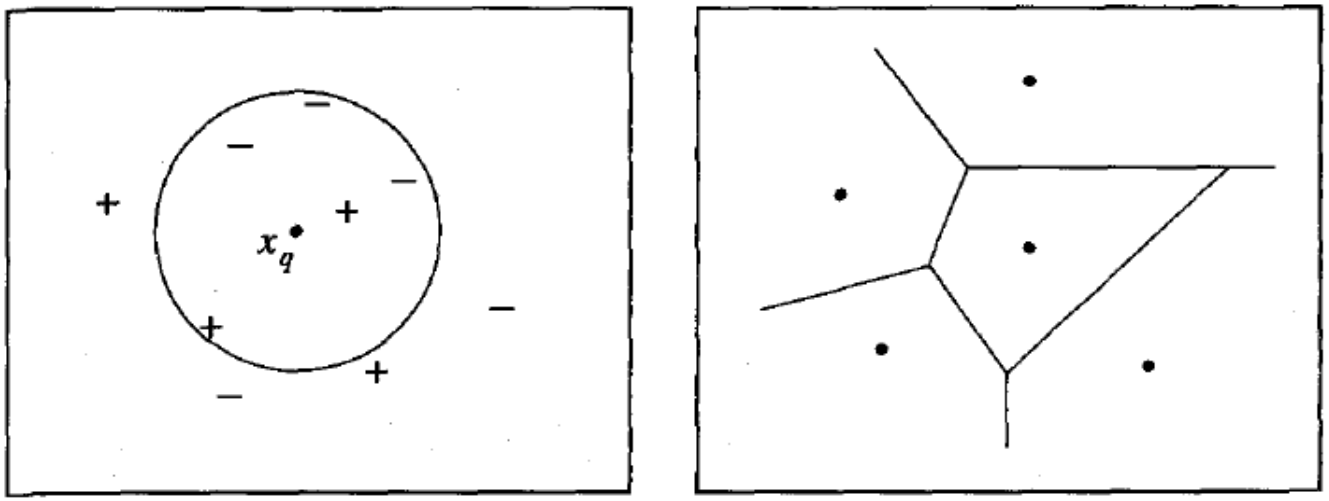
K-Vizinhos mais próximos (*K-Nearest Neighbor*)

- Para uma classificação de valores discretos, temos uma função ***f*** da forma: $f : \Re^n \rightarrow V, \text{ onde } V \text{ é o conjunto finito } \{v_1,...,v_s\}$

- **ALGORITMO DE TREINAMENTO**:
  - Para cada exemplo de treinamento ***‹x, f(x)›***, adicione o exemplo à lista de exemplos de treinamento.

- **ALGORITMO DE CLASSIFICAÇÃO**:
  - Dada uma nova instância x_q para ser classificada:
    - Seja $x_1, ..., x_k$ que denotam as ***k*** instâncias de exemplos de treinamento que estão mais perto de x_q.
    - Retorne o valor mais comum de ***f*** entre estes ***k*** exemplos.

K-Vizinhos mais próximos (*K-Nearest Neighbor*)



- Na esquerda um exemplo da operação do *k-nearest neighbor* para instâncias de duas dimensões e função de classificação binária.
  - O **1-*nearest neighbor*** classifica $\boldsymbol{x_q}$ positivo e *5-nearest neighbor* classifica $\boldsymbol{x_q}$ negativo.
- Na direita um exemplo de como o *1-nearest neighbor* irá associar valores às instâncias mais perto de cada ponto.

K-Vizinhos mais próximos (*K-Nearest Neighbor*)

- Para uma classificação de valores discretos, temos uma função ***f*** da forma:

$$f : \Re^n \rightarrow V, \text{ onde } V \text{ é o conjunto finito } \{v_1,...,v_s\}$$

e o valor da classificação (função de aproximação) se torna:

$\hat{f}(x_q)$

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{arg} \sum_{i=1}^{k} \delta(v, f(x_i))$$

$$\text{onde } \delta(a,b) = 1 \text{ se } a = b \text{ e } \delta(a,b) = 0 \text{ caso contrário.}$$

- Para uma classificação de valores contínuos, temos uma função ***f*** da forma:

$f : \Re^n \rightarrow \Re$

e o valor da classificação se torna:

$\hat{f}(x_q)$

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} f(x_i)}{k}$$

K-Vizinhos mais próximos ponderados pela distância

- Um refinamento óbvio é ponderar a contribuição de cada vizinho ***k*** na classificação de uma instância $\boldsymbol{x_q}$ de acordo com a distância entre eles.
  - Dando maior peso aos exemplos que estão mais perto.
  - Podemos ponderar o voto de cada vizinho de acordo com o inverso do quadrado da distância até $\boldsymbol{x_q}$.

$$\hat{f}(x_q) \leftarrow \underset{v \in V}{\arg} \sum_{i=1}^{k} w_i \delta(v, f(x_i)), \text{no caso discreto e}$$

$$\hat{f}(x_q) \leftarrow \frac{\sum_{i=1}^{k} w_i f(x_i)}{\sum_{i=1}^{k} w_i}, \text{no caso contínuo}$$

$$\text{onde} \quad w_i \equiv \frac{1}{d(x_q, x_i)^2}$$

K-Vizinhos mais próximos ponderados pela distância

- Quando adicionamos o peso da distância, não existe nenhum problema em utilizarmos k = todos os exemplos de treino.
  - Exemplos muito distantes irão influenciar muito pouco a classificação de uma nova instância.
  - A desvantagem é que o classificador será mais lento (requer mais computação).

- Se todos os exemplos de treinamento são considerados para classificar uma nova instância, chamamos o **ALGORITMO DE MÉTODO GLOBAL**.

- Se somente os ***k*** exemplos mais próximos são considerados para classificar uma nova instância, chamamos o **ALGORITMO DE MÉTODO LOCAL**.

K-Vizinhos mais próximos ponderados pela distância - avaliação

- É um método de inferência indutivo altamente efetivo para muitos problemas práticos.
- É robusto para exemplos de treinamento com ruído e bastante efetivo quando são fornecidos conjuntos de treinamento suficientemente grandes.
- **Suposições do algoritmo**: Todos os atributos são importantes para classificar a instância.
  - Emprega a distância euclidiana em todos os eixos (atributos) para calcular a distância entre os exemplos.
  - Diferente de árvores de decisão, por exemplo.
- Abordagem especialmente sensível à praga (*curse*) da dimensionalidade – muitos atributos podem ser irrelevantes.

K-Vizinhos mais próximos ponderados pela distância – abordagens utilizadas

- Para o problema da dimensionalidade:
  - Ponderar cada atributo de forma diferente quando calcular a distância entre duas instâncias.
  - O quanto cada eixo deve pesar na distância pode ser encontrado por validação cruzada. Duas formas:
    - Determinar um vetor de pesos que minimiza o erro da classificação verdadeira.
    - Eliminar completamente os atributos irrelevantes do espaço de instâncias (por validação cruzada por omissão de um).

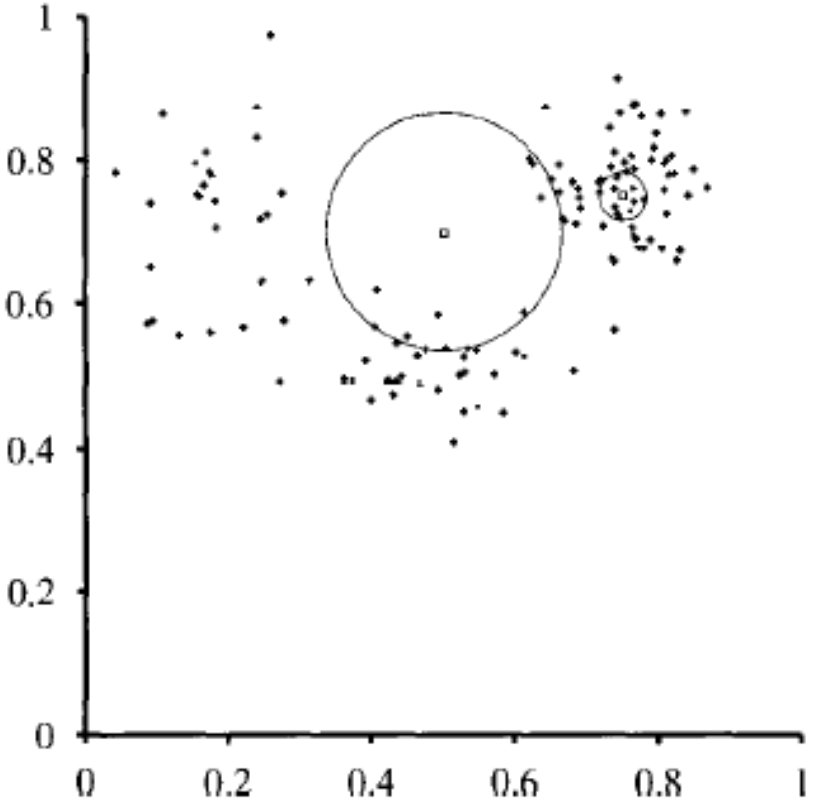
K-Vizinhos mais próximos ponderados pela distância – abordagens utilizadas

- Para reduzir o problema da quantidade de computação exigida para classificar uma nova instância:
  - Pode-se fazer indexação dos exemplos armazenados para identificar os vizinhos mais próximos mais rapidamente.
  - Normalmente se utiliza uma árvore ***Kd*** (*Kd-tree*).

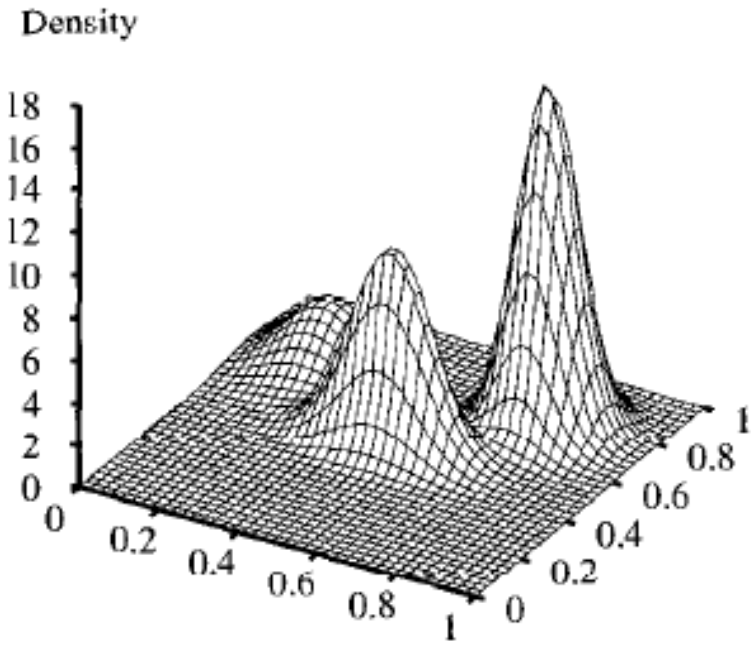
Sobre o valor de *k*

- ***K*** deve ter tamanho suficiente para assegurar uma estimativa significativa.
- Para ***k*** fixo o tamanho da vizinhança varia:
  - Onde os dados são esparsos, a vizinhança é grande.
  - Onde os dados são densos, a vizinhança é pequena.
- Na prática um valor de ***k*** entre 5 e 10 fornece bons resultados na maioria dos conjuntos de dados de baixo número de dimensões.
- Um bom valor de ***k*** também pode ser escolhido com a utilização de validação cruzada.

10 vizinhos mais próximos em uma amostra de 128 pontos

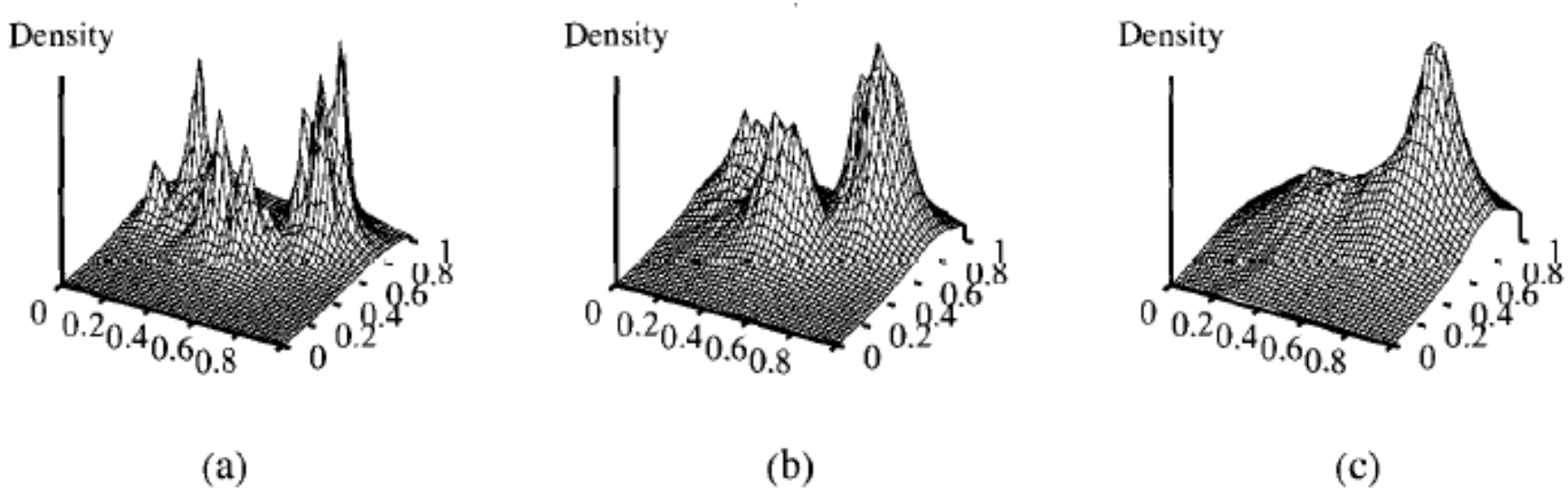


(a)



(b)

Estimativa de densidade dos k-vizinhos mais próximos a partir dos dados da figura anterior



- (a) k=3 – densidade altamente variável
- (b) k=10 – uma boa reconstrução da densidade verdadeira
- (c) k=40 – a vizinhança é muito grande e a estrutura dos dados é perdida

APRENDIZAGEM SUPERVISIONADA POR AGRUPAMENTO

Texto base:

Stuart Russel e Peter Norving - “Inteligência Artificial” - cap 18.

Aprendizagem supervisionada por Agrupamento

- A ideia é selecionar uma coleção inteira ou um **AGRUPAMENTO DE HIPÓTESES** a partir do espaço de hipóteses, e **combinar suas previsões**.
  - Ex.: Poderíamos gerar uma centena de hipóteses diferentes do mesmo conjunto de treinamento e depois fazê-las votar na melhor classificação para um novo exemplo.

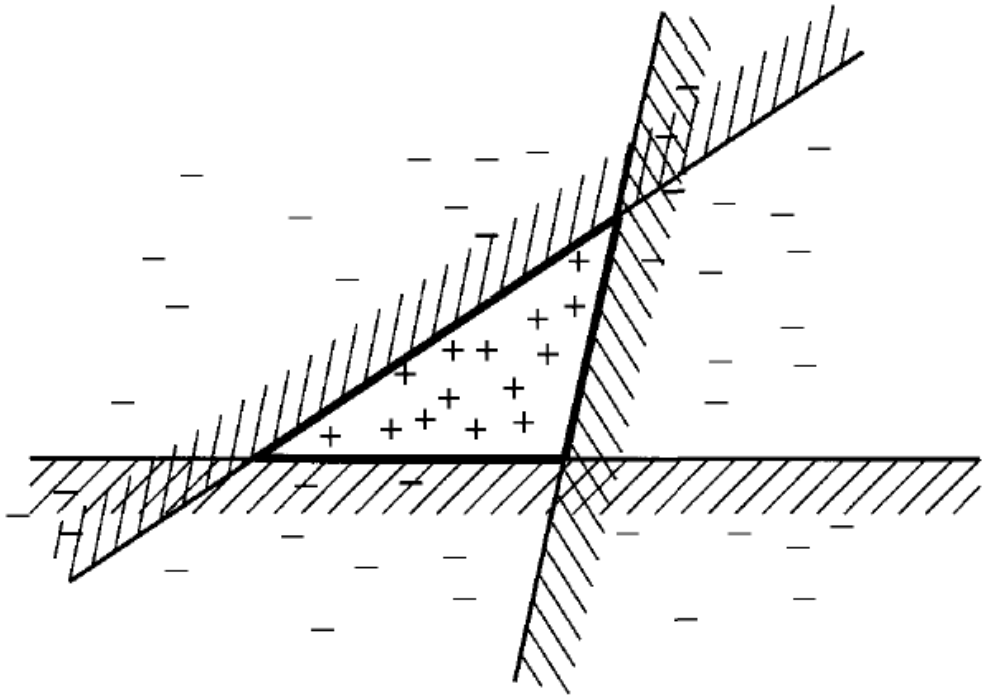
- Considere um conjunto de M=5 hipóteses, para as quais combinamos suas previsões usando votação pela maioria.
  - Para este conjunto classificar de forma errada um novo exemplo, pelo menos 3 das 5 hipóteses tem que classificar o exemplo de forma errada.
  - A ideia é que isso seja muito menos provável do que uma classificação errada por uma única hipótese.

Reduzindo a taxa de erros

- Suponha que cada hipótese h_i no conjunto tem um erro p.
  - A probabilidade de um exemplo escolhido ao acaso ser classificado de forma incorreta por h_i é p.
- Suponha que os erro cometidos por cada hipótese sejam independentes.
  - Isto é pouco provável porque as hipóteses serão iludidas do mesmo modo por quaisquer enganos dos dados de treinamento.
  - Mas se as hipóteses forem pelo menos um pouco diferentes, a correlação entre os seus erros diminui.
- Se p é pequeno, então a probabilidade de ocorrer um grande nº de classificações errada é minúsculo.
  - **Exemplo**: Usar um conjunto de 5 hipóteses reduz a taxa de erro de 1 em 10 para uma taxa menor que 1 em 100.

Conjunto de hipóteses

- Também podemos considerá-lo como uma forma genérica de ampliar o espaço de hipóteses.
  - O próprio conjunto é uma hipótese e o espaço de hipóteses é o conjunto de todos os conjuntos possíveis que podem ser construídos a partir do espaço de hipóteses original.
  - Isto pode resultar em um espaço de hipóteses mais expressivo e, consequentemente, em classes muito mais expressivas de hipóteses.



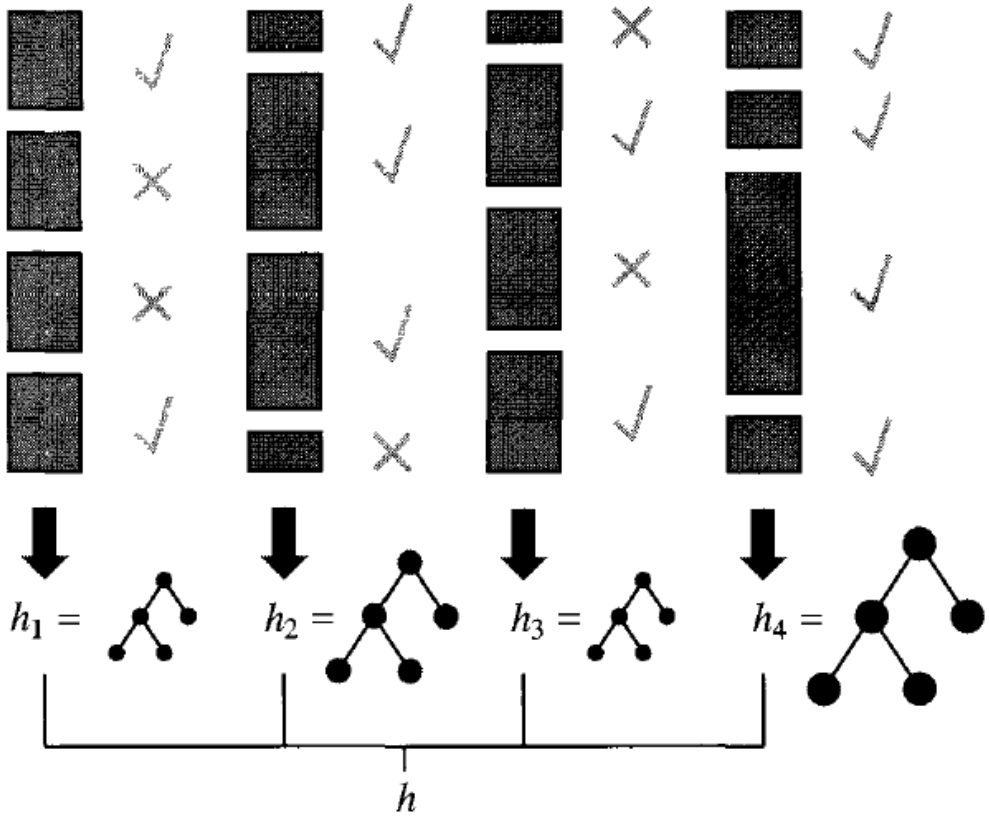
- Três hipóteses de limiar linear, cada uma classifica exemplos positivos no lado não-hachurado e exemplos negativos no lado hachurado.
- Classificamos positivamente qualquer exemplo classificado positivamente pelas 3 hipóteses.
- A região triangular resultante é uma hipótese que não pode ser expressa no espaço de hipóteses original.

Aceleração (*Boosting*)

- É o método de agrupamento mais amplamente utilizado.
- Utiliza o conceito de conjunto de treinamento ponderado.
  - Cada exemplo tem um peso associado $w_j >= 0$.
  - Quanto mais alto o peso de um exemplo, mais alta será a importância associada a ele durante a aprendizagem de uma hipótese.
- A aceleração começa com um conjunto de treinamento normal (com $w_j = 1$ para todos os exemplos) e gera h_1 a partir deste conjunto.
- A hipótese gerada a partir do conjunto de treinamento normal (h_1) classificará alguns exemplos de forma correta e outros de forma errada.

Aceleração (*Boosting*)

- Os exemplos são avaliados de acordo com a resposta de h_1:
  - Os exemplos classificados incorretamente por h_1 tem seus pesos aumentados para que h_2 os classifique melhor e os exemplos classificados corretamente por h_1 tem seus pesos diminuídos.
  - h_2 é gerada a partir deste novo conjunto de treinamento ponderado.
  - O processo continua deste modo até que M hipóteses sejam geradas.
- A hipótese de conjunto final é uma combinação de maioria ponderada de todas as M hipóteses.
  - Cada h é ponderada de acordo com o seu comportamento no conjunto de treinamento.
- Existem muitas variantes da ideia básica de aceleração com diferentes modos de ajustes de pesos e de combinação de hipóteses.



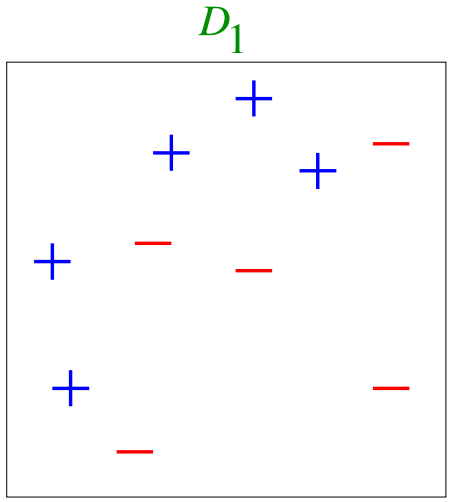
- Funcionamento do algoritmo de aceleração.
- Cada retângulo corresponde a um exemplo. A altura do retângulo corresponde ao peso.
- O tamanho da árvore de decisão indica o peso desta hipótese na hipótese do agrupamento final.

Um algoritmo específico para aceleração

function ADABOOST($examples$, L, M) **returns** a weighted-majority hypothesis
  inputs: $examples$, set of N labelled examples $(x_1, y_1), \ldots, (x_N, y_N)$
    L, a learning algorithm
    M, the number of hypotheses in the ensemble
  local variables: **w**, a vector of N example weights, initially $1/N$
    h, a vector of M hypotheses
    z, a vector of M hypothesis weights

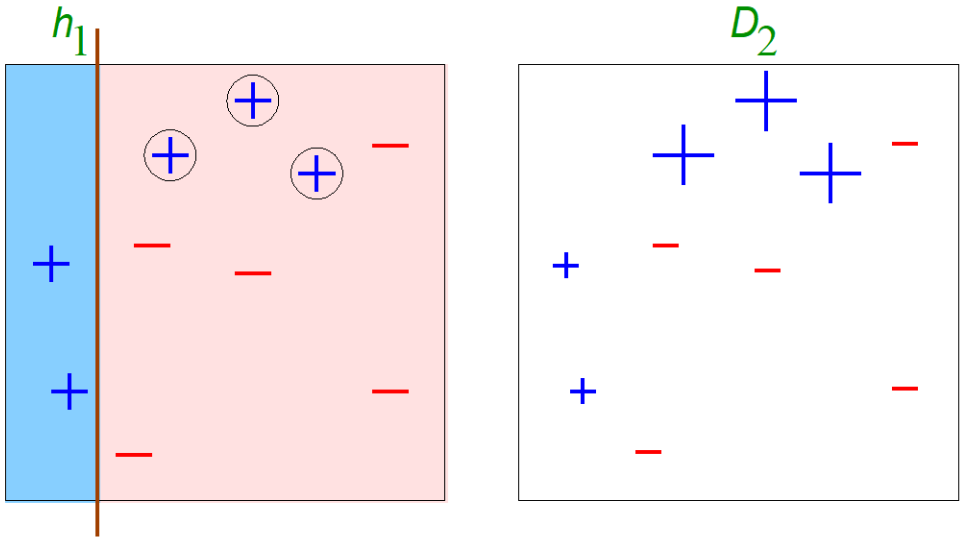
  for $m = 1$ **to** M **do**
    $\mathbf{h}[m] \leftarrow L(examples, \mathbf{w})$
    $error \leftarrow 0$
    for $j = 1$ **to** N **do**
      if $\mathbf{h}[m](x_j) \neq y_j$ **then** $error \leftarrow error + \mathbf{w}[j]$
    for $j = 1$ **to** N **do**
      if $\mathbf{h}[m](x_j) = y_j$ **then** $\mathbf{w}[j] \leftarrow \mathbf{w}[j] \cdot error/(1 - error)$
    $\mathbf{w} \leftarrow$ NORMALIZE(**w**)
    $\mathbf{z}[m] \leftarrow \log(1 - error)/error$
  return WEIGHTED-MAJORITY(**h**, **z**)

Toy Example



weak classifiers = vertical or horizontal half-planes

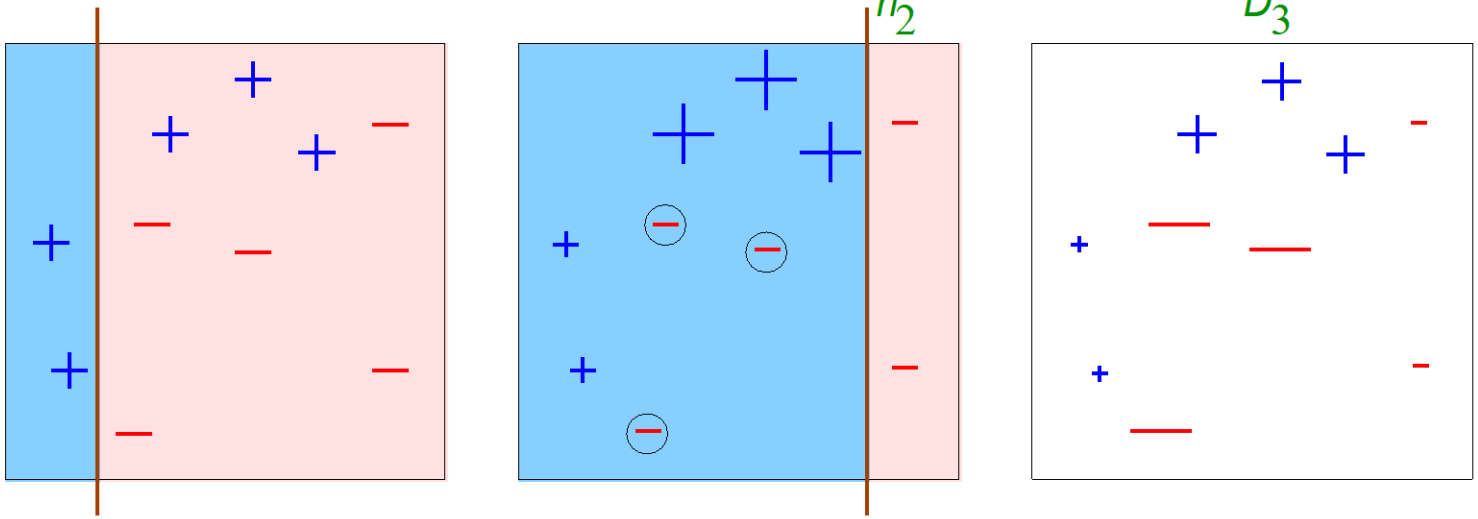
Round 1



$\varepsilon_1 = 0.30$

$\alpha_1 = 0.42$

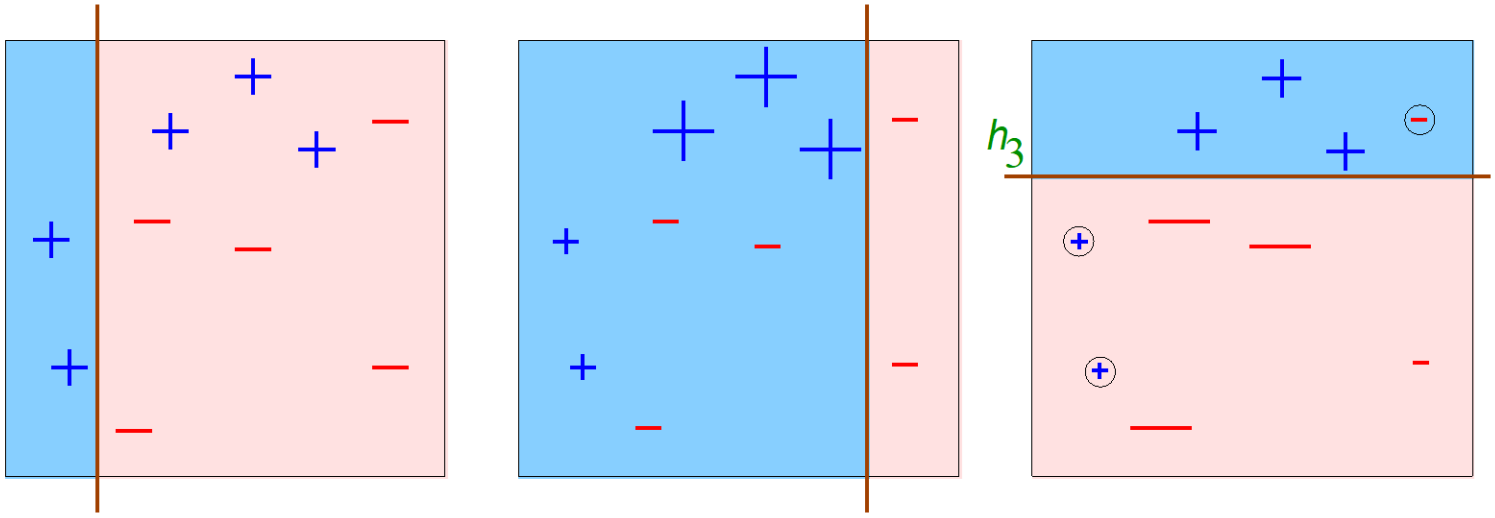
Round 2



$\varepsilon_2 = 0.21$

$\alpha_2 = 0.65$

Round 3

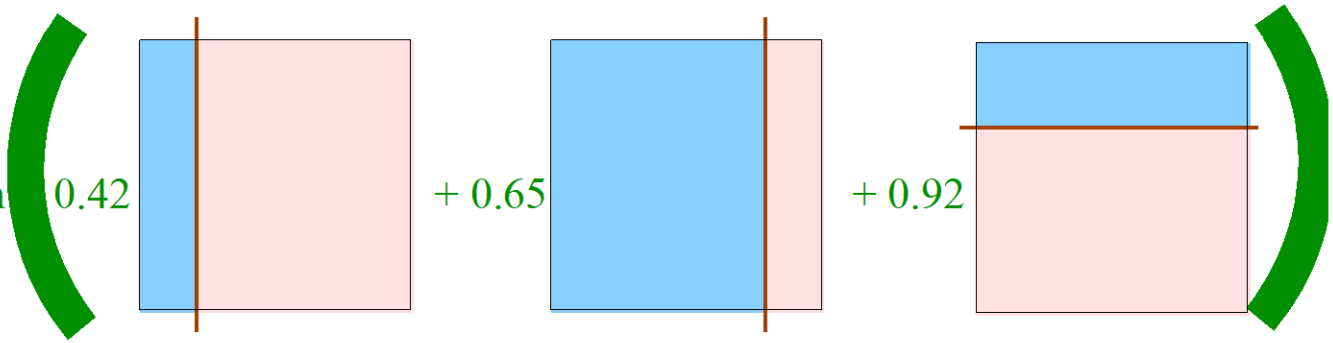


$\varepsilon_3 = 0.14$

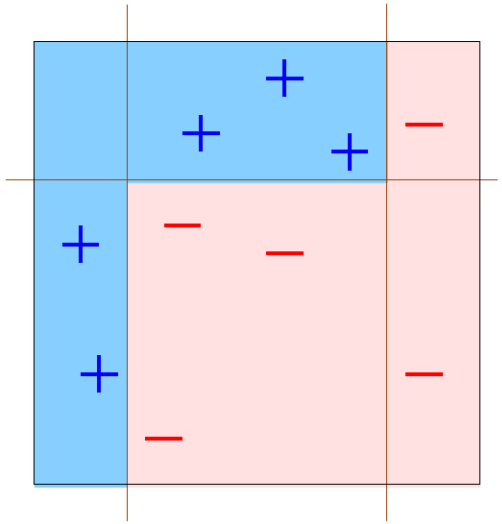
$\alpha_3 = 0.92$

Final Classifier

$H_{\text{final}} = \text{sign}\left(0.42 \right. \quad + 0.65 \quad + 0.92 \left. \right)$



$=$



ADABOOST – Propriedades Importantes

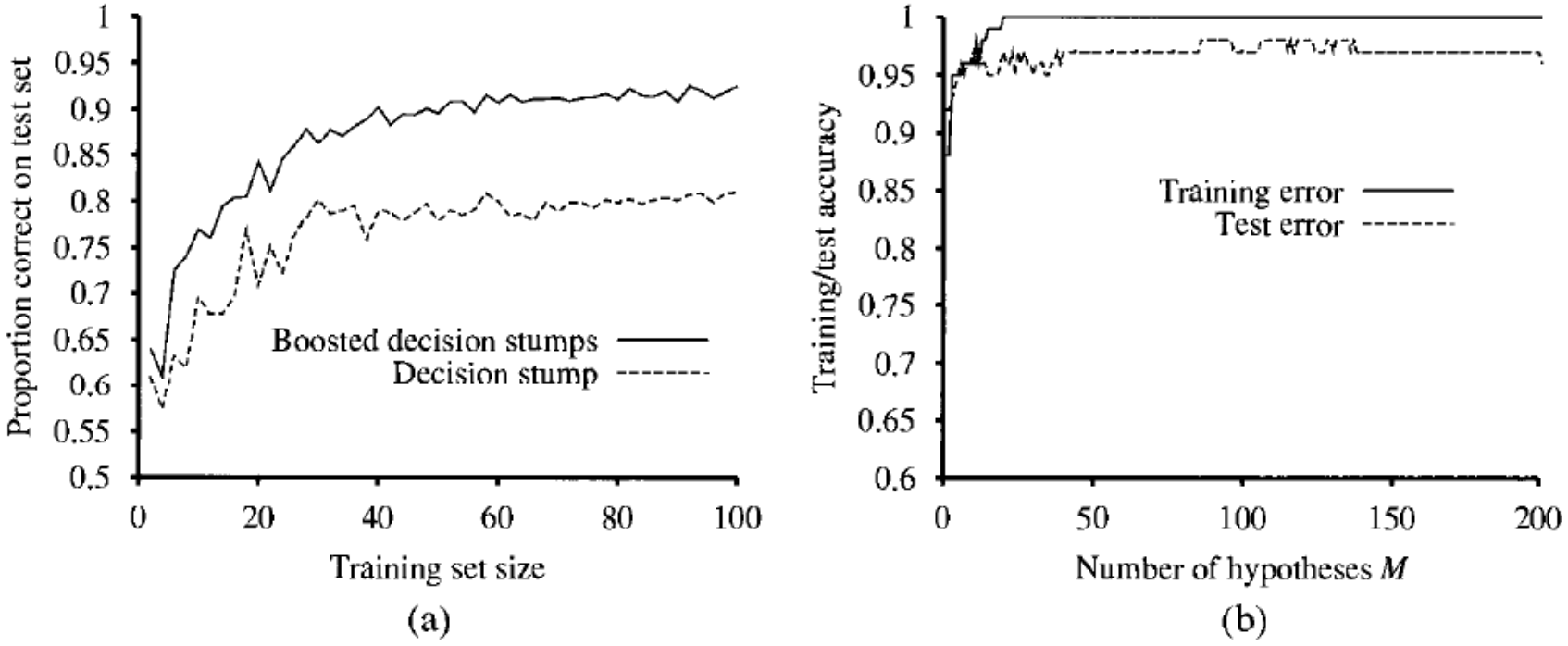
- Se o algoritmo de aprendizagem ***L*** de entrada for um **algoritmo de aprendizagem fraca** (i.e. sempre retorna uma hipótese com erro ponderado sobre o conjunto de treinamento que é ligeiramente melhor do que o palpite aleatório).

- Então o ADABOOST retorna uma hipótese que classifica perfeitamente os dados de treinamento para ***M*** grande o bastante.

- Este resultado é válido independente de quanto o espaço de hipóteses original seja inexpressivo e de quanto é complexa a função que está sendo aprendida.

Aceleração para o exemplo do restaurante

- Com espaço de hipóteses original = classe de cepos de decisão (árvores de decisão com apenas um teste na raiz).



Número de hipóteses X Erro no conjunto de teste

- O erro alcança zero no conjunto de treinamento quando M=20 (uma combinação ponderada pela maioria de 20 cepos de decisão).
- Conforme ***M*** aumenta o erro no conjunto de treinamento continua zero, mas o erro no conjunto de teste diminui muito tempo depois disso (M=137).
- Apesar da lâmina de Ockham nos dizer que não devemos tornar as hipóteses mais complexas do que o necessário, o gráfico nos diz o contrário.
  - Uma explicação é que a inclusão de hipóteses adicionais torna o conjunto mais definido para distinguir os exemplos positivos dos negativos.