Inteligência Artificial

Luís A. Alexandre

UBI

Ano lectivo 2018-19

Conteúdo

Introdução

- Vamos estudar um conjunto de técnicas que nos permitem aprender de forma supervisionada a partir de um conjunto de dados.
- Estas abordagens são complementares à árvores de decisão que estudámos na aula anterior.
- Veremos tanto abordagens para classificação como para regressão.

Conteúdo

Regressão Linear

- Vamos ver como fazer **regressão linear**: ajustar uma reta aos dados.
- Podemos escrever a equação de uma reta como

$$y = w_1 x + w_0 \tag{1}$$

onde os parâmetros podem ser colocados num vetor de pesos $\mathbf{w} = [w_0, w_1]$.

- Podemos então reescrever a equação da reta como

$$h_{\mathbf{w}}(x) = w_1 x + w_0 \tag{2}$$

Regressão Linear

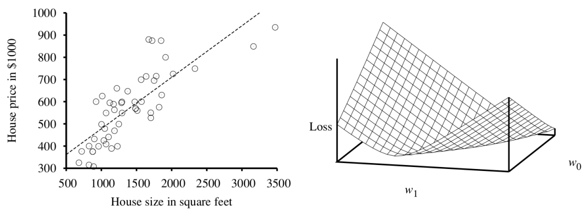


Figura de Russell & Norvig

Regressão Linear

- Para fazermos o ajuste da reta temos que determinar os pesos. Isso faz-se minimizando o **erro empírico** (por vezes chamado loss em inglês):

$$EE(h_{\mathbf{w}}) = \sum_{i=1}^{n} (y_i - h_{\mathbf{w}}(x_i))^2 \tag{3}$$

- Escolhemos então os pesos que minimizam este erro:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}} EE(h_{\mathbf{w}})$$

- Temos que achar as derivadas parciais em ordem às variáveis que estamos a procurar, w_0 e w_1, e igualar a zero:

$$\frac{\partial EE(h_{\mathbf{w}})}{\partial w_0} = 0 \qquad \frac{\partial EE(h_{\mathbf{w}})}{\partial w_1} = 0 \tag{4}$$

Regressão Linear

- A solução é

$$\begin{aligned} w_1 &= \frac{N\sum(x_i y_i) - (\sum x_i)(\sum y_i)}{N(\sum x_i^2) - (\sum x_i)^2} \\ w_0 &= \frac{1}{N}\left(\sum y_i - w_1 \sum x_i\right) \end{aligned} \tag{5}$$

- O que estamos a fazer ao determinar estes pesos é a escolher o mínimo da função mostrada na figura anterior (lado direito): é convexa, logo não tem mínimos locais.
- O exemplo e as equações apresentadas são para o caso unidimensional (regressão univariada), mas podemos trabalhar com dados vetoriais e aí temos a regressão multivariada.

Conteúdo

Regressão Logística

- As funções lineares podem ser usadas não só para regressão mas também para classificação.
- Neste caso vamos criar um **classificador** linear baseado na seguinte função, chamada de logística:

$$logistica(x) = \frac{1}{1 + e^{-x}} \tag{6}$$

- A nossa hipótese será

$$h_{\mathbf{w}}(\mathbf{x}) = logistica(\mathbf{w} \cdot \mathbf{x}) = \frac{1}{1 + e^{-\mathbf{w} \cdot \mathbf{x}}} \tag{7}$$

- Consideramos as entradas como sendo um vetor e não um escalar como no exemplo anterior: agora temos $\mathbf{x}$ e não x.

Descida do gradiente

- Podemos achar os pesos com um processo iterativo: a **descida do gradiente**.
- Na realidade este processo é muito usado também em outros classificadores, quando não é possível obter a solução exata para os valores dos pesos que minimizam o erro empírico.
- Algoritmo:
  $\mathbf{w} \longleftarrow$ valor aleatório no espaço dos pesos
  enquanto não exista convergência fazer:
  para cada w_i em $\mathbf{w}$ fazer:

$$w_i \longleftarrow w_i - \alpha \frac{\partial}{\partial w_i} EE(\mathbf{w}) \tag{8}$$

- Chamamos ao α a **taxa de aprendizagem**.

Descida do gradiente

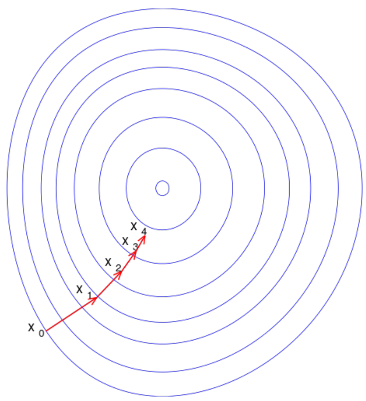


Figura da wikipedia

Regressão Logística

- Para aplicarmos a descida do gradiente no caso da RL para obtermos os pesos necessários, devemos achar o valor do gradiente na expressão (8).
- A expressão da atualização dos pesos fica então a seguinte:

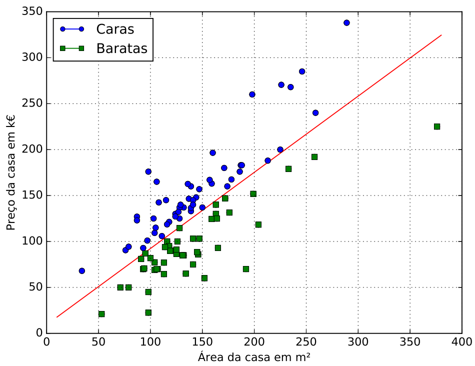
$$w_i \longleftarrow w_i + \alpha(y - h_{\mathbf{w}}(\mathbf{x}))h_{\mathbf{w}}(\mathbf{x})(1 - h_{\mathbf{w}}(\mathbf{x}))x_i \tag{9}$$

Regressão Logística

- Como se pode usar a linha que obtemos na RL para classificar?
- Podemos usar essa linha para separar os dados pertencentes a duas classes: a linha passa a chamar-se **fronteira de decisão**.
- Essa linha separa os pontos de duas classes: os que ficam de um lado e do outro da linha.

Regressão Logística

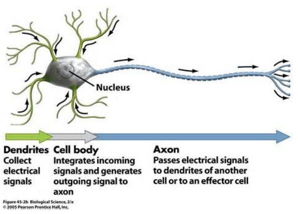
- Exemplo: a linha ajustada aos dados do valor duma casa pode ser agora a fronteira de decisão entre duas classes: as casas baratas e as caras.

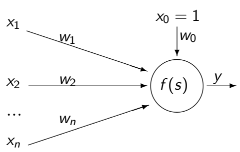


Conteúdo

Redes Neuronais

- Alguns investigadores, inspirados pela única coisa inteligente que conhecemos, o cérebro humano, decidiram criar modelos inspirados no nosso cérebro.
- O primeiro modelo de um neurónio artificial foi proposto em 1943.
- O seu funcionamento é simples: quando os valores das suas entradas excedem um limiar, o neurónio "dispara".





Redes Neuronais

- Do ponto de vista formal, os valores das **entradas**, x_i, num neurónio são multiplicadas por um **peso**, w_i, e somadas para produzirem uma média pesada:

$$s = \sum_{i=0}^{n} x_i w_i \tag{10}$$

- Esta soma é depois passada por uma função não linear, $f(s)$, chamada a **função de ativação**.
- A saída ou **ativação** do neurónio obtém-se com $y = f(s)$.
- Existem muitas propostas para $f(s)$, mas uma muito usada é a logística que vimos atrás (também chamada de **sigmóide**). Neste caso a saída do neurónio obtém-se com

$$y = logistica(s) = \frac{1}{1 + e^{-\sum_{i=0}^{n} x_i w_i}} \tag{11}$$

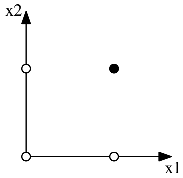
Redes neuronais

- Exemplo: vamos usar um neurónio para implementar a função AND (E lógico) entre duas entradas. A função de ativação que vamos usar é dada por

$$f(x) = \begin{cases} 0, & x \leq 0 \\ 1, & x > 0 \end{cases}$$

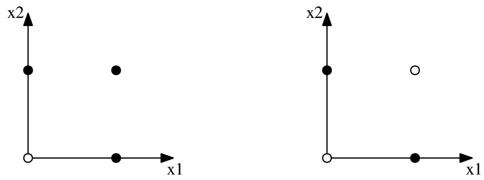
- Podemos representar estes dados num plano, que neste caso representa o espaço de entrada do problema:

x_1	x_2	x_1 AND x_2
0	0	0
0	1	0
1	0	0
1	1	1



Redes neuronais

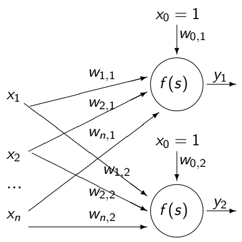
- Como fazer para classificarmos os dados de entrada usando um neurónio?
- Testar os seguintes pesos: $w_0 = -1.5$, $w_1 = 1$, $w_2 = 1$.
- Exercício: OR e XOR.



- Como obter os pesos em geral: usamos a descida do gradiente e a expressão que usamos é a já vista no caso da regressão logística (equação (9)) quando a função de ativação é a sigmóide.

Perceptrão

- Os neurónios estão normalmente organizados em camadas.
- Uma rede neuronal com apenas uma camada de neurónios chama-se um **perceptrão**:



- Neste exemplo temos uma camada com apenas 2 neurónios.
- Para usamos uma rede para classificar dados podemos usar tantos neurónios quantas as classes que queremos processar.

Redes neuronais

- As RNs são muito versáteis e podem ser usadas para todos os tipos de aprendizagem que referimos na aula anterior: não supervisionada, supervisionada (tanto para classificação como para regressão), semi-supervisionada e para aprendizagem por reforço.
- Aqui estudámos apenas a RN mais simples: o perceptrão.
- Vimos que implementa um hiperplano no espaço de entrada (dos dados) podendo apenas resolver problemas que sejam linearmente separáveis.
- Existem muitas mais RNs, todas elas mais potentes que um simples perceptrão, e que hoje em dia são as responsáveis pelos enormes avanços da IA. Para as estudar existem outras UCs em mestrado e doutoramento que focam os detalhes.

Conteúdo

Máquinas de Vetores de Suporte

- As **Máquinas de Vetores de Suporte** (em inglês SVMs) têm algumas propriedades interessantes:
  - constroem uma fronteira de decisão que tem a **margem máxima** em relação aos pontos do conjunto de treino
  - embora implementem um separador linear, conseguem obter fronteiras de decisão complexas construindo esse separador linear num **espaço de maior dimensionalidade que o de entrada**
  - são resistentes ao sobre-ajuste
- Vimos antes que um método como a regressão logística acha uma fronteira de decisão entre 2 classes usando para isso todos os pontos do conjunto de treino. Numa MVS alguns pontos são mais importantes que outros.

Máquinas de Vetores de Suporte

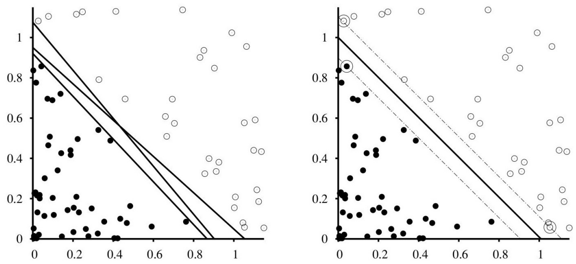


Figura de Russell & Norvig

Máquinas de Vetores de Suporte

- Em vez de minimizarem o erro empírico no conjunto de treino, as MVS tentam minimizar o erro de generalização.
- Como, se não sabemos que pontos serão usados para teste?
- Para o conseguir as MVS escolhem o plano separador que esteja mais afastado dos pontos vistos até ao momento: é o **separador de margem máxima**.
- A **margem** é o dobro da distância entre o separador e o ponto mais próximo (aparece como a largura da zona tracejada na figura anterior).

Máquinas de Vetores de Suporte

- A solução para achar os pesos que permitem construir o plano separador pode ser obtida resolvendo:

$$\arg\max_{\alpha} \sum_j \alpha_j - 0.5 \sum_{j,k} \alpha_j \alpha_k y_j y_k (\mathbf{x}_j \cdot \mathbf{x}_k) \tag{12}$$

  onde $\alpha_j \geq 0$ e $\sum_j \alpha_j y_j = 0$.
- Este é um problema de **otimização quadrática** que pode ser resolvido com software específico: obtemos o vetor α.
- Podemos depois obter os pesos com $\mathbf{w} = \sum_j \alpha_j \mathbf{x}_j$
- Há 3 características importantes na equação (12):
  - é uma expressão **convexa**: tem apenas um máximo global que pode ser encontrado de forma eficiente;
  - os dados só aparecem em produtos de pares de pontos;
  - os pesos α_j associados aos pontos são zero menos nos **vetores de suporte**: os pontos mais perto do separador.

Máquinas de Vetores de Suporte

- Como se consegue usar uma MVS quando os exemplos não são linearmente separáveis?
- Se mapearmos os dados para um espaço de dimensão suficientemente elevada conseguimos separá-los com um hiperplano.
- Em geral, para um problema com N pontos, é sempre possível encontrar uma forma de os separar linearmente num espaço de dimensão $N-1$ ou superior.

Máquinas de Vetores de Suporte

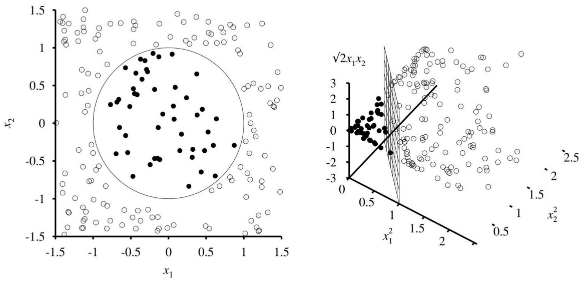


Figura de Russell & Norvig

Leitura recomendada

- Russell e Norvig, sec. 18.6, 18.7, 18.8.