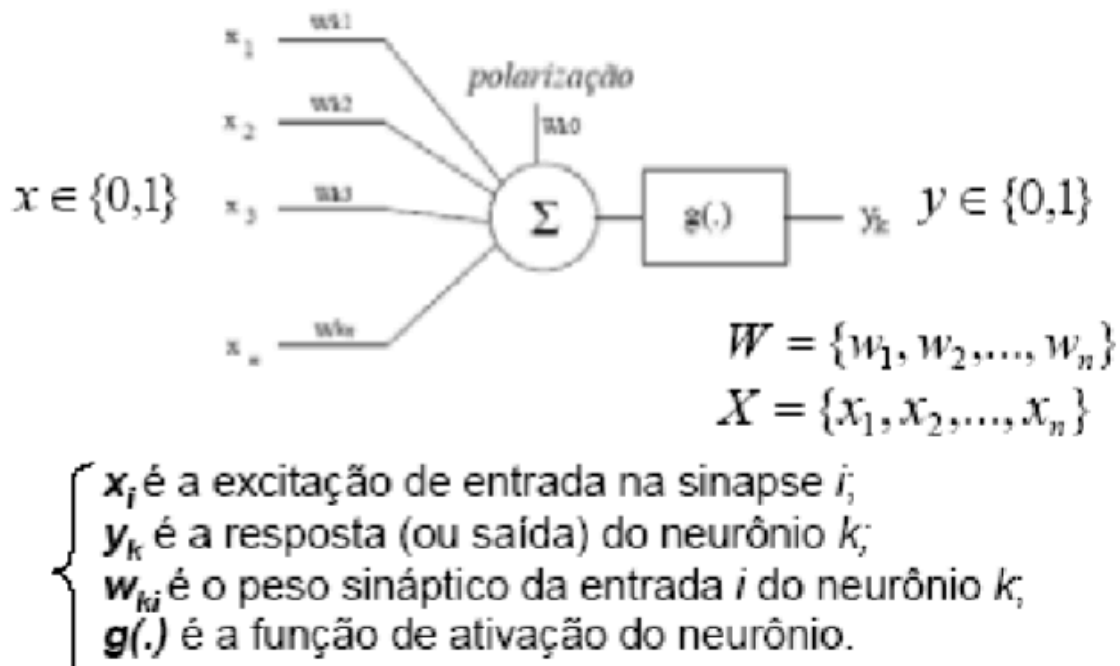


INTELIGÊNCIA ARTIFICIAL

Redes Neurais - 06/10

Estrutura do neurônio de McCulloch-Pitts:



A entrada é um conjunto de informações de um objeto analisado. O caso explicado na sala de aula foi de uma pessoa, então:

$x_1 = \text{Gênero} \cdot W_g$

$x_2 = \text{Escolaridade} \cdot W_e$

$x_3 = \text{Idade} \cdot W_i$

$x_4 = \text{Nacionalidade} \cdot W_n$

$x_{bias} = \text{Bias} \cdot W_{bias}$

Peso sináptico é o peso atrelado a cada "sinapse" da entrada.

- O peso diz a importância dos pesos para o resultado final de uma análise.

O somatório do neurônio é o **somatório do produto**.

- Soma o peso atrelado a entrada x com o valor da entrada de fato.

Existem uma entrada padrão chamada de polarização (BIAS) que, normalmente vai ser 1, mas o seu peso que vai fazer a diferença, ou seja:

- Se tivermos 10 entradas, teremos 11 informações → 10 entradas + 1 polarização.

A saída dessa análise é binária, logo:

- Temos uma função de ativação após o somatório so neurônio ser calculado.
- Essa função auxiliar ativa um neurônio com base em uma condição. Por exemplo:
 - Se o somatório for > que 0 a saída é 1.
 - Se o somatório for < que 0 a saída é 0.

Perceptron:

Consirando a função AND:

- Ela é LINEARMENTE separável.
- Considerando que o peso de X é 0,2, de Y é -0,4 e do bias (valor 1) é 0,3:

X	Y	X ^ Y	Σ	f
0	0	0	0,3	1
0	1	0		
1	0	0		
1	1	1		

$X \wedge Y \rightarrow$ É a saída esperada.

$\Sigma \rightarrow$ É o somatório do produtório - entradas vezes os pesos.

f → Saída encontrada

- Cálculos feitos:
 - Para Σ : $X \cdot W_x + Y \cdot W_y + \text{Bias} \cdot W_{\text{bias}}$
 - No primeiro caso: $0 \cdot 0,2 + 0 \cdot -0,4 + 1 \cdot 0,3 = 0,3$.
 - No caso acima, pela função auxiliar - por ser maior que 0 - o resultado f = 1. O que está **ERRADO**.

- Após o calculo acima, utilizamos a seguinte fórmula para ajustar esse erro:
 - $W_{t+1} = W_t + t_a \cdot \text{erro} \cdot \text{entrada}$
 - $W_{t+1} \rightarrow$ Peso novo, após o ajuste.
 - $W_t \rightarrow$ Peso atual, sem o ajuste.
 - $t_a \rightarrow$ Taxa de aprendizado (Será bem pequena para que o ajuste aconteça devagar)
 - erro \rightarrow Saída esperada MENOS Saída encontrada.
 - entrada \rightarrow Entrada que estamos ajustando.
 - $W_{f1} \rightarrow$ primeira saída $\rightarrow W_{f1} = 0,2 + 0,3 \cdot (0 - 1) \cdot 0$
 - $W_{f1} = 0,2 + 0 \rightarrow 0,2$
 - Atualizamos o peso de X:
 - Repetimos esse passo a passo para Y e para o BIAS.
 - Atualizamos eles, se necessário.
- Em entradas que podem variar entre muitos valores - que são altos, por exemplo idade -, devemos fazer um "ajuste" nessas entradas para que os cálculos sejam feitos com valores na mesma escala. Isso busca evitar que uma entrada roube no resultado.

Redes Neurais - 08/10

XOR | XNOR \rightarrow Não podem ser resolvidas por um perceptron simples, pois não são lineares.

Exercícios

Dado um perceptron simples de duas entradas e um bias, cujos pesos são $w_1 = 0,5$, $w_2 = 0,4$ e $w_0 = -0,3$, respectivamente, assinalar a resposta correta:

- (a) o perceptron realiza a função NOR
- (b) o perceptron realiza a função AND
- (c) o perceptron realiza a função OR
- (d) o perceptron realiza a função XOR
- (e) nenhuma das alternativas

R: (c) o perceptron realiza a função OR

X	Y	$X \wedge Y$	Σ	f		O limiar é: $> 0 = 1 \mid < 0 = 0$
0	0	0	-0,3	0	→	$0 \times 0,5 + 0 \times 0,4 + 1 \times -0,3 = -0,3 \rightarrow 0$
0	1	0	0,1	1	→	$0 \times 0,5 + 1 \times 0,4 + 1 \times -0,3 = 0,1 \rightarrow 1$
1	0	0	0,2	1	→	$1 \times 0,5 + 0 \times 0,4 + 1 \times -0,3 = 0,2 \rightarrow 1$
1	1	1	0,6	1	→	$1 \times 0,5 + 1 \times 0,4 + 1 \times -0,3 = 0,6 \rightarrow 1$

MLP - Multilayer Perceptron

Algoritmo Backpropagation:

- Fase 1 - Propagação (fase forward):
 - Os perceptrons recebem cada entrada e realizam o cálculo do **somatório do produto** de suas entradas e respectivos pesos - **NÃO** podemos esquecer do BIAS.
 - A função auxiliar **NÃO É MAIS A MESMA** → Agora é utilizado uma função **NÃO** linear.
 - As saídas de cada perceptron são passadas para os próximos perceptrons e o processo se repete até a camada da saída final, onde encontramos a resposta e onde o erro é calculado.
- Fase 2 - Retropropagação (fase backward):
 - Ao encontrar um erro, precisamos estimar a contribuição de cada perceptron para esse resultado.

- Começamos a estimar na camada de saída.
 - Se for trivial fazemos: **derivada(função de ativação) * (desejado - real)**.
 - Se não for trivial, utilizamos a solução do próximo item.
- Essa é a fórmula mais usual que utilizamos para encontrar esse erro:

Ao invés de usar a diferença absoluta, pode-se utilizar também o **erro médio quadrático (Mean Square Error – MSE) ou Root Mean Square Error – RMSE)**

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y}_i)^2 \qquad \text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (S_i - O_i)^2}$$

Técnicas de Agrupamento - 27/10

Aprendizado Indutivo → Cria uma função e com base nela classifica os testes.

- Descritivo:
 - Associação.
 - Agrupamento.
- Preditivo:
 - Classificação.
 - Regressão.
 - O Random Forest, Backpropagation e Cart resolvem esse problema.

Tipos de agrupamento → Agrupamentos são formados com base em similaridades

- Particional:
 - K-Means → Vamos ver o que cada indivíduo está em um único grupo.
 - Muito bom quando temos uma clara divisão de grupos.

- Complexidade: $O(n*k*i*d)$
 - N = Número de instâncias.
 - K = Número de clusters (grupos).
 - I = Número de iterações.
 - D = Número de atributos.
- Existe o **nebuloso**, onde um indivíduo do teste pode participar de diversos grupos.
- Distâncias intra-clusters são minimizadas, já as entre clusters são maximizadas.
 - Distância de Manhattan.
- 1a interação:
 - Chuta os Ks (instâncias).
 - Calcula a distância entre cada ponto com os Ks escolhidos (centróides).
- 2a interação:
 - Calcula a média entre as distâncias das instâncias do grupo para cada atributo. Os centróides antigos participam do cálculo dessa média.
 - Marca os novos centróides como + em cada grupo.
 - Calcula a distância entre todas as instâncias aos novos centróides, criando um **NOVO** grupo.
- 3a interação e seguintes:
 - Refaz o passo 2.
 - O critério de parada é quando X% que estão mudando de grupo. "Até que somente 1% dos objetos mudam de clusters", onde $X = 1$.
- Vamos pensar que utilizamos o K-Means e encontramos 3 grupos:
 - De fato essa base tem 3 grupos?
 - Não! Afinal, se ele encontrou 3, é porque foi definido $K = 3$.

- Uma solução é utilizar esse algoritmo diversas vezes com diversos Ks, calculando a qualidade de cada uma dessas "respostas" e utilizando a melhor.
- Como posso dar um nome ao grupo?
 - Roda uma árvore, gerando regras que ajudam a dar uma ideia sobre o nome do grupo.
- Hierárquico:
- Por densidade:
 - DB Scan.
- Baseado em modelo:
 - SOM.

12/11

Dados inconsistentes → São aqueles dados que possuem valores conflitantes em seus atributos.

- Duas instâncias com todos atributos iguais, mas com o resultado diferente.
 - Exemplo: Dois pacientes com todos os seus sintomas iguais, mas com doenças diagnosticadas diferentes.
- Quanto mais reduzimos a quantidade de atributos, maior a chance de termos dados inconsistentes.
- Retiramos todos os dados inconsistentes. Se temos duas instâncias em conflito, retiramos as duas.

Dados redundantes → São aqueles dados que possuem valores semelhantes em seus atributos.

- Uma instância é redundante quando ela é muito semelhante a uma outra instância do mesmo conjunto de dados.
- Podemos se referir a **instâncias** ou **atributos**.
 - Dados redundantes em relação ao atributo X.
 - Dados redundantes em relação a instâncias completas.

- Exemplo: Pacientes com todos ou quase todos sintomas iguais, com diagnósticos iguais.
- Retiramos todos os dados redundantes, exceto uma instância deles.
- A redundância de um atributo está relacionada à sua **correlação** com um ou mais atributos do conjunto de dados.
 - Dois atributos estão correlacionados quando apresentam um perfil de variação semelhante.
 - Exemplo: Alunos que tem mais **horas de estudo** possuem maiores **notas**. (Atributos de entrada!)
 - Se a correlação ocorrer entre um **atributo de entrada** e um **atributo de saída**, o atributo de entrada terá uma **grande** influência na predição do valor do atributo de classe (saída).
 - Exemplo: Uma instância com o atributo de entrada "quebrou regra do tipo" está **muito** correlacionada ao atributo de saída "quebrou regra?". Afinal, ter um valor positivo no de entrada, diz 100% sobre o valor do de saída.
- **Dados ausentes** → O ideal é **NÃO** ter dado ausente, mas na vida real **quase sempre** temos eles.
 - **Próxima aula.**