Trabalho 2 de Aprendizado de Máquina

Data de Entrega do Trabalho 2: 11/12/2024

Prof. Dr. Flávio Henrique Teles Vieira

Trabalho 2 de Aprendizado de Máquina

- O trabalho é Individual;
- Não utilizar código da Internet ou de outra fonte para o (s) algoritmo (s) do tema sorteado;
- Não copiar texto ou código do colega;
- Implementação no Python;
- Enviar códigos (.py e .ipynb) em separado do relatório e relatório por email junto com outros arquivos necessários (.csv, .txt, etc)

Trabalho 2 de Aprendizado de Máquina

- Não é necessário inserir teoria do livro no relatório, a não ser para explicar algum resultado;
- Fazer *upload* dos arquivos do trabalho no SIGAA até **23:59**h do dia **11/12/2024**. Sugestão: fazer upload antes;
- Prazo de 1 semana para o envio do trabalho **sem prorrogação**

Trabalho 2 de Machine Learning

- Pode utilizar comandos: para inversão de matriz, transpor vetor ou matriz, média, encontrar autovalores de matrizes;
- Não utilizar kmeans, clusterdata, ksdensity, k-nn, pca;
- Não utilizar códigos prontos para os algoritmos solicitados, por exemplo para k-nn, pca.
- Pode utilizar numpy, matplotlib, random e pandas no Python

Trabalho 2 de Machine Learning

- Pode utilizar dados dos seus trabalhos de pesquisa;
- Caso não tenham em mente que dados utilizar, seguem a seguir sugestões de *links* contendo dados para classificação e regressão;
- Descrever os dados utilizados no relatório

Exemplo de Conjuntos de Dados

- UCI Repository: http://www.ics.uci.edu/~mlearn/MLRepository.html
- UCI KDD Archive: http://kdd.ics.uci.edu/summary.data.application.html
- Delve: http://www.cs.utoronto.ca/~delve/

- No Python: from sklearn.datasets

Exemplo de Conjuntos de Dados

- UCI Repository: http://www.ics.uci.edu/~mlearn/MLRepository.html
- UCI KDD Archive: http://kdd.ics.uci.edu/summary.data.application.html
- Delve: http://www.cs.utoronto.ca/~delve/

- No Python: from sklearn.datasets

Relação Nota-Conceito

Relação Nota-Conceito

9,0 a 10 – A

7,5 a 8,9 – B

6,0 a 7,4 – C

0,0 a 5,9 – D

Relatório

- Descrever os dados utilizados, plotando gráficos sempre que possível;
- Dizer no relatório quais valores (e valores iniciais) foram utilizados para os parâmetros dos algoritmos;
- Descrever configuração ideal encontrada (ex número de *clusters*, ordem do polinômio, número de autovalores, etc)
- Mostrar gráfico de erro (erro quadrático médio, erro quadrático relativo) por iteração e total;
- Comparar desempenho com diferentes ordens e configurações, quando aplicável;
- Mostrar para classificação métricas como: acurácia, precisão, recall, F1-score: pode utilizar códigos/funções prontas

Relatório

- Mostrar superficies/fronteiras de decisão (classificação);
- Mostrar matriz de confusão para problemas de classificação: pode utilizar Código/função pronta;
- Mostrar/plotar função (modelo) de regressão obtido para os dados, quando for o caso;
- Utilizar validação cruzada: Dividir os dados em pelo menos 2 grupos, que podem ser treinamento e teste e mostrar resultados para cada grupo;
- Descrever considerações efetuadas para implementar algoritmos, quando aplicável;
- Ao lidar com 2 algoritmos, comparar os resultados, sempre que possível.

[8, 10, 5, 11, 7, 6, 1, 4, 3, 9, 2]

Sorteio

- Import random

- random.sample(range(1,12), 11)

Sorteio

- 1 AUGUSTO CÉSAR MUNIZ LOPES
- 2 DULCINÉIA GONÇALVES FERREIRA PIRES
- 3 GEAN SILVA DOS SANTOS
- 4 LUCAS DE JESUS BATISTA GONÇALVES
- 5 MARLON PETRI DE LIMA
- 6 NATANAEL AMARAL FALEIRO
- 7 RAFAEL PEREIRA DA SILVA
- 8 RENATO FRANCA DE ALMEIDA
- 9 RICARDO ANDRÉ NAKA
- 10 RICARDO NEVES TAVARES
- 11 VINÍCIUS FARIA COSTA MENDANHA

[8, 10, 5, 11, 7, 6, 1, 4, 3, 9, 2]

Temas

Trabalho Tipo 1: Árvore de Decisão para Classificação

- Utilizar os dados abaixo. Comparar resultados com Gini e Entropia

Tempo	Temperatura	Umidade	Vento	Joga
Chuvoso	71	91	Sim	Não
Ensolarado	69	70	Não	Sim
Ensolarado	80	90	Sim	Não
Nublado	83	86	Não	Sim
Chuvoso	70	96	Não	Sim
Chuvoso	65	70	Sim	Não
Nublado	64	65	Sim	Sim
Nublado	72	90	Sim	Sim
Ensolarado	75	70	Sim	Sim
Chuvoso	68	80	Não	Sim
Nublado	81	75	Não	Sim
Ensolarado	85	85	Não	Não
Ensolarado	72	95	Não	Não
Chuvoso	75	80	Não	Sim

```
GerarArvore(𝒳)
    Se EntropiaNo(𝒳)<θ_I                                   /* eq. 9.3 */
        Criar nó folha rotulado com a classe majoritária em 𝒳
        Retorne
    i ←DividirAtributo(𝒳)
    Para cada ramo de x_1
        Encontre 𝒳_i seguindo o ramo
        GerarArvore(𝒳_i)
DividirAtributo(𝒳)
    EntMin ← MAX
    Para todos os atributos i = 1,...,d
        Se x_i  é discreto com n valores
            Divida 𝒳 em 𝒳_1,...,𝒳_n por x_1
            e ← DividirEntropia(𝒳_1,...,𝒳_n)                /* eq. 9.8 */
            Se e < EntMin        EntMin ← e; melhorF ←i
        Senão /* x_1 é numérico */
            Para todas as possibilidades divida
                Divida 𝒳 em 𝒳_1, 𝒳_2 em x_1
                e ← DividirEntropia(𝒳_1, 𝒳_2)
                Se e < EntMin        EntMin ← e; melhorF ← i
    Retorne melhorF
```

Ganho de Informação

- Ganho de informação mede **redução na entropia** nas partições obtidas de acordo com os valores do atributo

- Ganho de informação alcançado selecionando A para divisão:

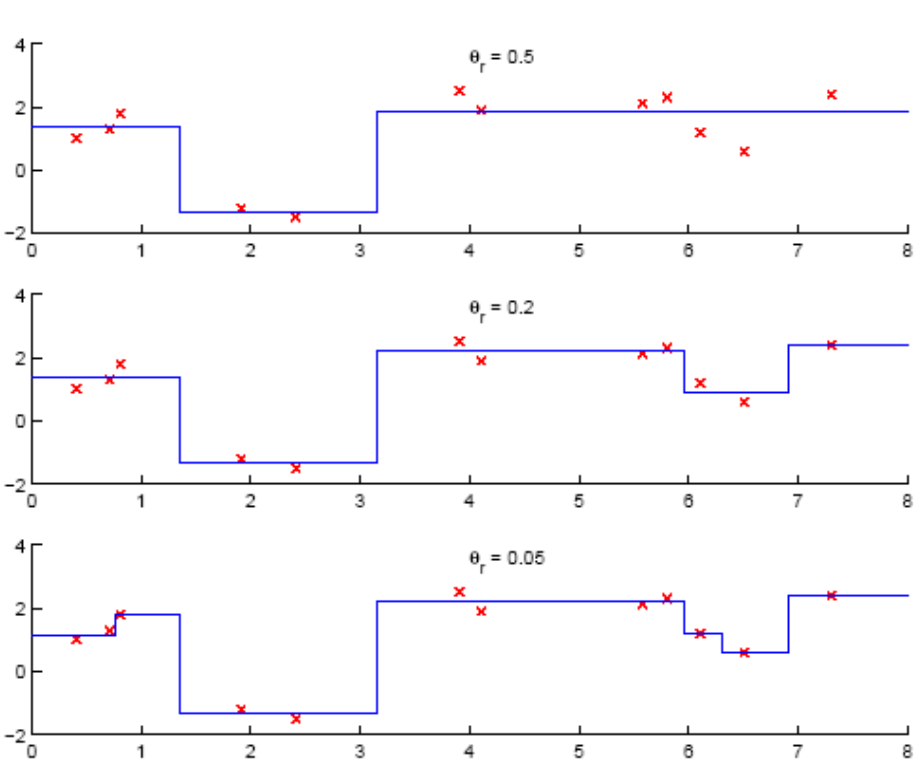
$$IG(A,D) = H(D) - H(A,D)$$

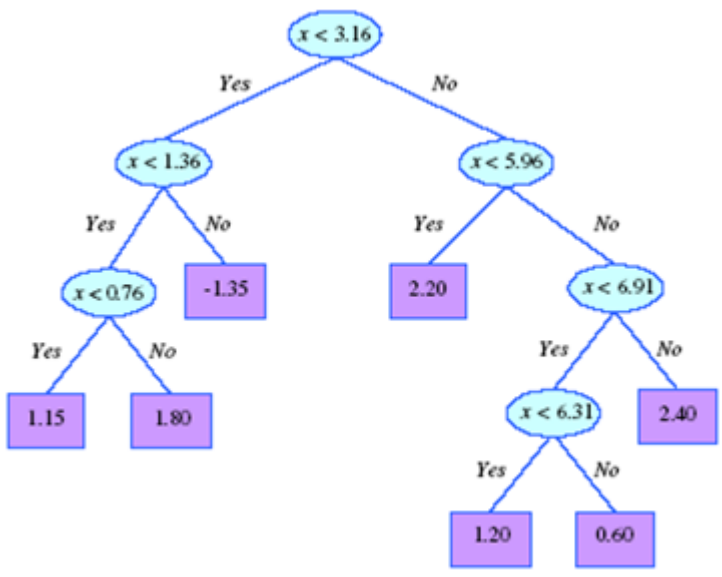
Ganho(*A*) = redução esperada da entropia devido à "classificação" de acordo com o atributo A

$$IG(A,D) \equiv H(D) - \sum_{v \in Valores(A)} \frac{|D_v|}{|D|} H(D_v)$$

Trabalho Tipo 2: Árvore de Decisão para Regressão

- Comparar resultados de erro quadrático médio





Árvores de Regressão

- Semelhante à árvore de classificação: muda a medida de **impureza (Erro quadrático médio)**. Seja X_m, o subconjunto de X que atende condições do nó *m;*
- Erro no nó *m*:

$$b_m(\mathbf{x}) = \begin{cases} 1 & \text{se } \mathbf{x} \in \mathcal{X}_m : \mathbf{x} \text{ alcança o nó } m \\ 0 & \text{senão} \end{cases}$$

$$E_m = \frac{1}{N_m} \sum_t \left(r^t - g_m\right)^2 b_m\left(\mathbf{x}^t\right) \qquad g_m = \frac{\sum_t b_m\left(\mathbf{x}^t\right) r^t}{\sum_t b_m\left(\mathbf{x}^t\right)}$$

- Após a divisão:

$$b_{mj}(\mathbf{x}) = \begin{cases} 1 & \text{se } \mathbf{x} \in \mathcal{X}_{mj} : \mathbf{x} \text{ alcança o nó } m \text{ e o ramo } j \\ 0 & \text{senão} \end{cases}$$

$$E'_m = \frac{1}{N_m} \sum_j \sum_t \left(r^t - g_{mj}\right)^2 b_{mj}\left(\mathbf{x}^t\right) \qquad g_{mj} = \frac{\sum_t b_{mj}\left(\mathbf{x}^t\right) r^t}{\sum_t b_{mj}\left(\mathbf{x}^t\right)}$$

$$N_m = |\mathcal{X}_m| = \sum_t b_m(\mathbf{x}^t)$$

Árvores de Regressão

- A equação $E_m = \frac{1}{N_m} \sum_t (r^t - g_m)^2 b_m(\mathbf{x}^t)$ corresponde à **variância** em *m ;*
- Se o erro é aceitável $E_m < \theta_r$ então um nó folha é criado e armazena o valor de g_m

- Procura-se por divisão que minimize:

$$E'_m = \frac{1}{N_m} \sum_j \sum_t (r^t - g_{mj})^2 b_{mj}(\mathbf{x}^t)$$

- Treinamento da árvore de regressão em comparação à árvore de classificação: troca **entropia** por **erro quadrático médio** e rótulos das **classes por médias**

Trabalho Tipo 3: Aprendizagem Competitiva (On-line Kmeans) + Perceptron para Classificação para K>2 classes

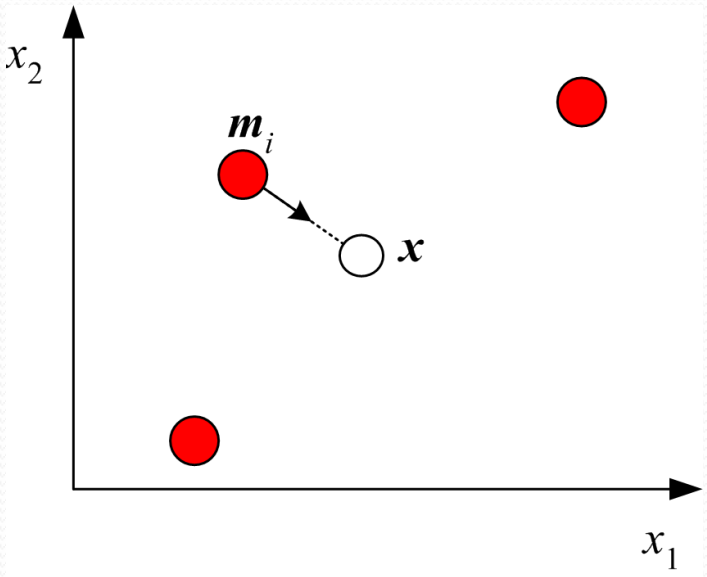
$$E\left(\{\mathbf{m}_i\}_{i=1}^{k}|\mathcal{X}\right)=\sum_t\sum_i b_i^t\left\|\mathbf{x}^t-\mathbf{m}_i\right\|$$

$$b_i^t=\begin{cases}1 & \text{if } \left\|\mathbf{x}^t-\mathbf{m}_i\right\|=\min_l\left\|\mathbf{x}^t-\mathbf{m}_l\right\| \\ 0 & \text{otherwise}\end{cases}$$

$$\text{Batch } k\text{-means}: \mathbf{m}_i=\frac{\sum_t b_i^t\mathbf{x}^t}{\sum_t b_i^t}$$

Online k-means :

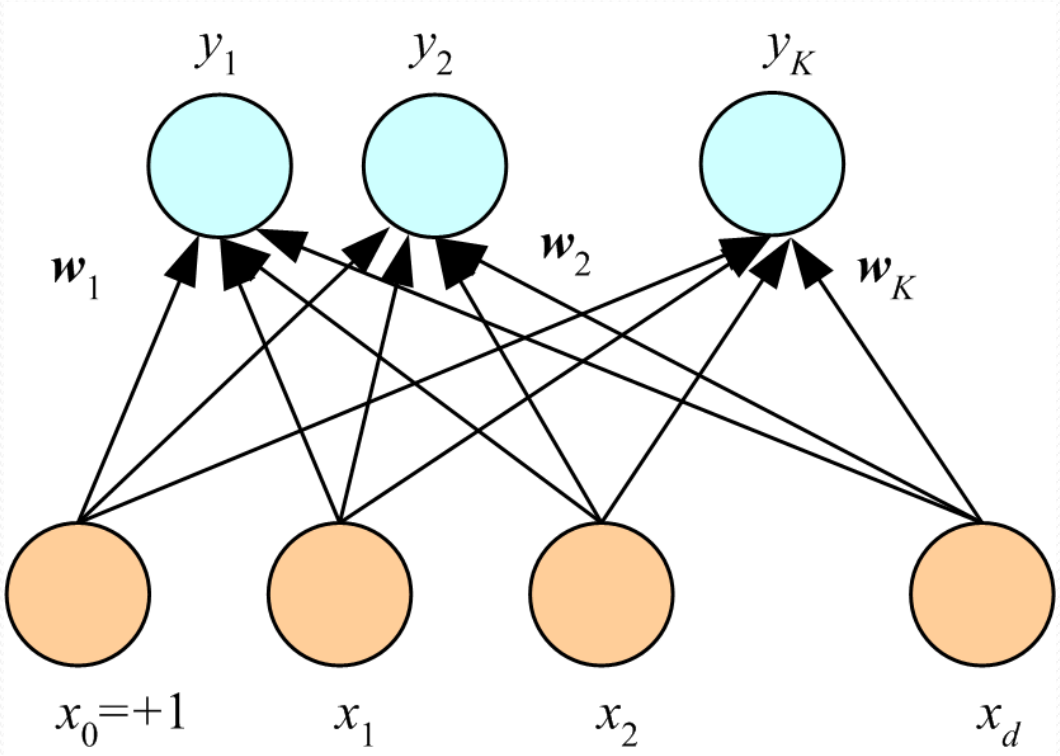
$$\Delta m_{ij}=-\eta\frac{\partial E^t}{\partial m_{ij}}=\eta b_i^t\left(x_j^t-m_{ij}\right)$$



m_i são os centros dos clusters

Trabalho Tipo 3: Aprendizagem Competitiva (On-line Kmeans) + Perceptron para Classificação para K>2 classes

- Com softmax e LMS para treinamento



K Saídas

Regressão:

$$y_i = \sum_{j=1}^{d} w_{ij} x_j + w_{i0} = \mathbf{w}_i^T \mathbf{x}$$

$$\mathbf{y} = \mathbf{W}\mathbf{x}$$

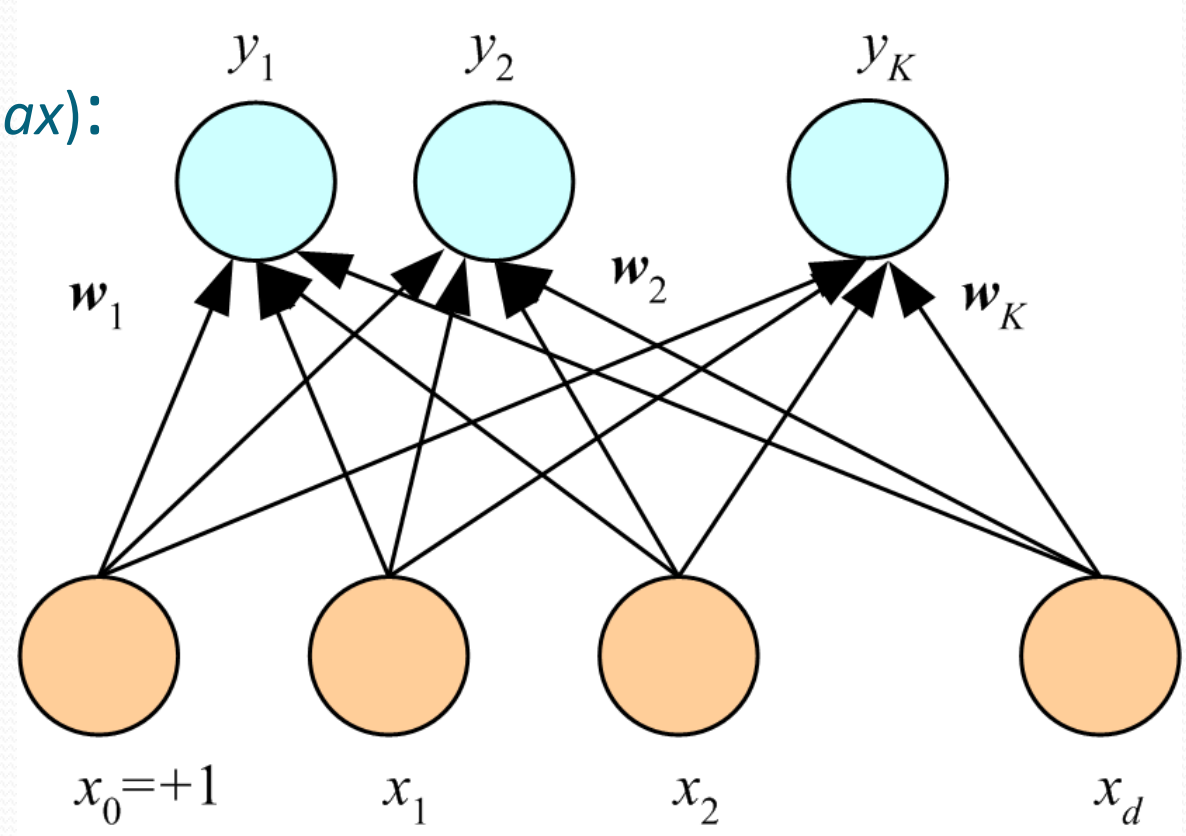
Classificação (com *softmax*):

$$o_i = \mathbf{w}_i^T \mathbf{x}$$

$$y_i = \frac{\exp o_i}{\sum_k \exp o_k}$$

Escolher C_i

se $y_i = \max_k y_k$



W é uma matriz K x (d+1)

Função de Ativação *Softmax*

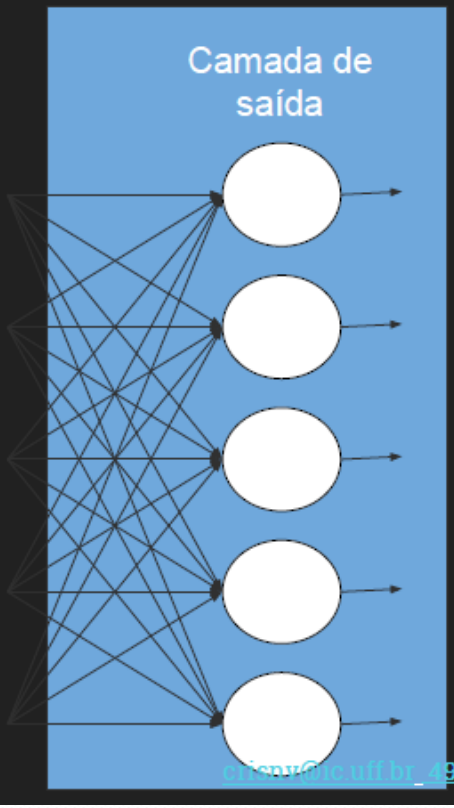
$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{x}^\mathsf{T}\mathbf{w}_j}}{\sum_{k=1}^{K} e^{\mathbf{x}^\mathsf{T}\mathbf{w}_k}}$$

Softmax (normalized exponential function):

Exemplo:

X = [1.0, 2.0, 3.0, 4.0, 5.0]

O = [0.012, 0.032, 0.086, 0.234, 0.636]



Treinamento

- Aprendizagem *Online* (instâncias apresentadas uma por uma) vs *on batch* (toda amostra):
  - Não há necessidade de armazenar toda amostra;
  - Problema pode variar com o tempo;
  - Desgaste e degradação em componentes do sistema.

- Stochastic gradient-descente (Gradiente Descendente Estocástico): atualiza depois de cada instância

- Regra Geral de Atualização (regra LMS):

$$\Delta w_{ij}^t = \eta\left(r_i^t - y_i^t\right)x_j^t$$

$$\text{Atualização} = \textit{TaxadeAprendizagem} \cdot \left(\text{SaídaDesejada} - \textit{Saída Real}\right) \cdot \textit{Entrada}$$

Perceptron: Algoritmo de Treinamento para K>2 classes

For $i = 1, \ldots, K$
  For $j = 0, \ldots, d$
    $w_{ij} \leftarrow \text{rand}(-0.01, 0.01)$
Repeat
  For all $(\mathbf{x}^t, r^t) \in \mathcal{X}$ in random order
    For $i = 1, \ldots, K$
      $o_i \leftarrow 0$
      For $j = 0, \ldots, d$
        $o_i \leftarrow o_i + w_{ij} x_j^t$
    For $i = 1, \ldots, K$
      $y_i \leftarrow \exp(o_i) / \sum_k \exp(o_k)$
    For $i = 1, \ldots, K$
      For $j = 0, \ldots, d$
        $w_{ij} \leftarrow w_{ij} + \eta(r_i^t - y_i) x_j^t$
Until convergence

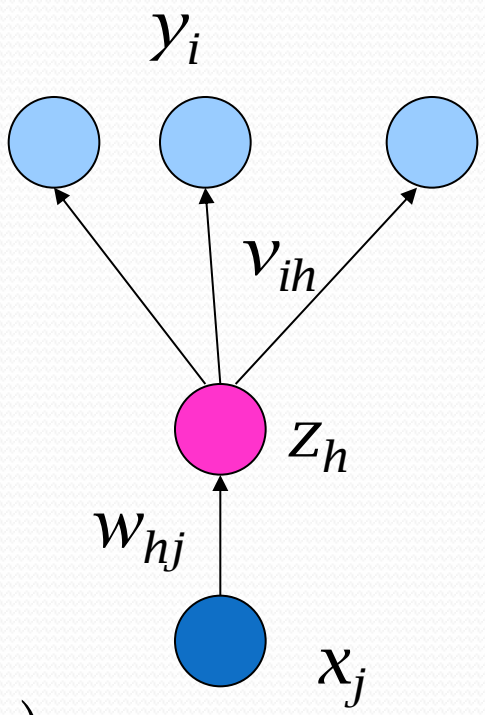
Trabalho Tipo 4: Regressão (MLP com backpropagation) com Múltiplas Saídas

$$E(\mathbf{W},\mathbf{V}\mid\mathcal{X})=\frac{1}{2}\sum_t\sum_i\left(r_i^t-y_i^t\right)^2$$

$$y_i^t=\sum_{h=1}^{H}v_{ih}z_h^t+v_{i0}$$

$$\Delta v_{ih}=\eta\sum_t\left(r_i^t-y_i^t\right)z_h^t$$

$$\Delta w_{hj}=\eta\sum_t\left[\sum_i\left(r_i^t-y_i^t\right)v_{ih}\right]z_h^t\left(1-z_h^t\right)x_j^t$$



Initialize all v_{ih} and w_{hj} to rand$(-0.01, 0.01)$

Repeat

For all $(\boldsymbol{x}^t, r^t) \in \mathcal{X}$ in random order

For $h = 1, \ldots, H$

$z_h \leftarrow \text{sigmoid}(\boldsymbol{w}_h^T \boldsymbol{x}^t)$

For $i = 1, \ldots, K$

$y_i = \boldsymbol{v}_i^T \boldsymbol{z}$

For $i = 1, \ldots, K$

$\Delta \boldsymbol{v}_i = \eta(r_i^t - y_i^t)\boldsymbol{z}$

For $h = 1, \ldots, H$

$\Delta \boldsymbol{w}_h = \eta(\sum_i (r_i^t - y_i^t) v_{ih}) z_h (1 - z_h) \boldsymbol{x}^t$

For $i = 1, \ldots, K$

$\boldsymbol{v}_i \leftarrow \boldsymbol{v}_i + \Delta \boldsymbol{v}_i$

For $h = 1, \ldots, H$

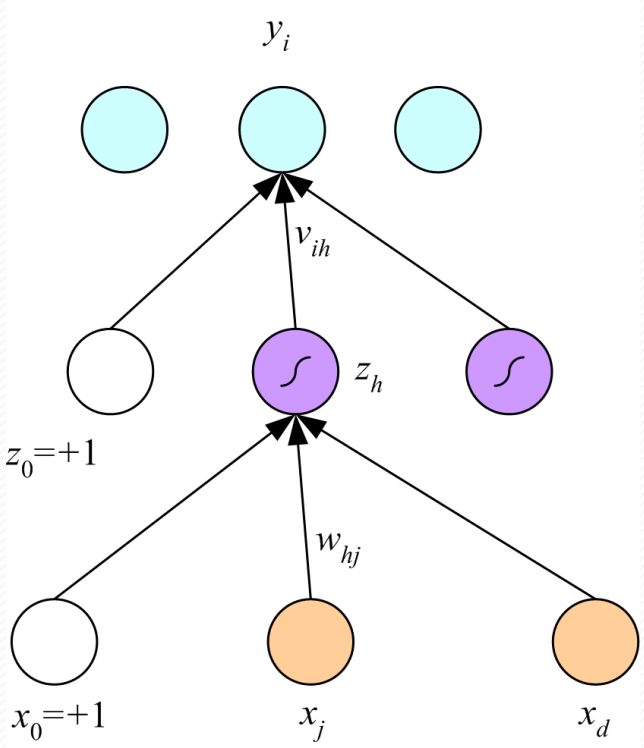
$\boldsymbol{w}_h \leftarrow \boldsymbol{w}_h + \Delta \boldsymbol{w}_h$

Until convergence

Algoritmo de Retropropagação para MLP para Regressão com K saídas

Trabalho Tipo 5: Discriminação com mais de 2 Classes (MLP Backpropagation)

- *MLP aplicada a um problema de Classificação*



K>2 Classes

$$o_i^t = \sum_{h=1}^{H} v_{ih} z_h^t + v_{i0} \qquad y_i^t = \frac{\exp o_i^t}{\sum_k \exp o_k^t} \equiv P\left(C_i \mid \mathbf{x}^t\right)$$

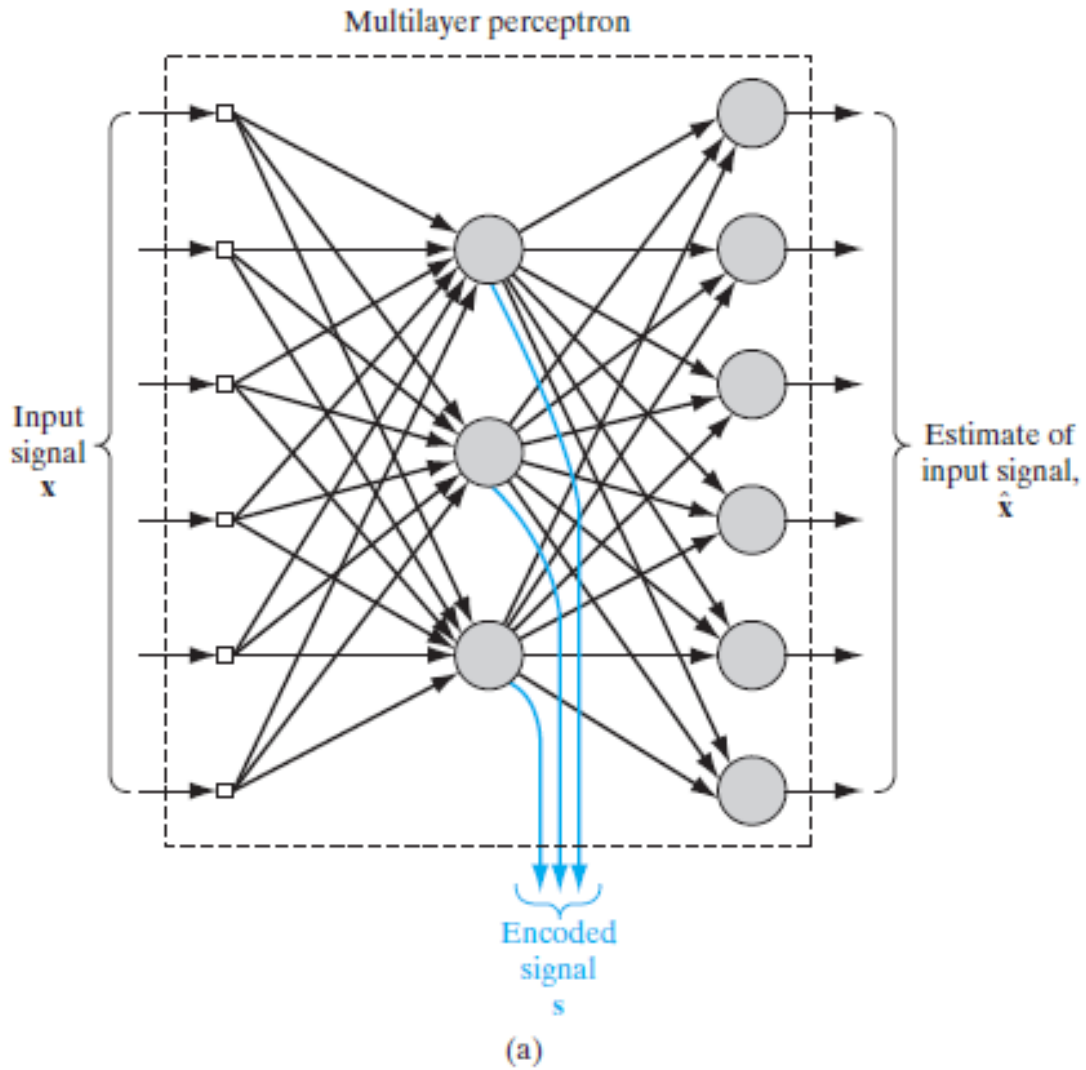
$$E\left(\mathbf{W}, \mathbf{v} \mid \mathcal{X}\right) = -\sum_t \sum_i r_i^t \log y_i^t$$

$$\Delta v_{ih} = \eta \sum_t \left(r_i^t - y_i^t\right) z_h^t$$

$$\Delta w_{hj} = \eta \sum_t \left[\sum_i \left(r_i^t - y_i^t\right) v_{ih} \right] z_h^t \left(1 - z_h^t\right) x_j^t$$

- Uso da função *Softmax*.
- Richard e Lippman (1991): MLP pode estimar probabilidades a posteriori

Trabalho Tipo 6: MLP Autocodificadora



- Bourlard and Kamp 1988 mostraram que MLP autocodificadora com 1 camada oculta implementa PCA.
- Entretanto, pesos dos neurônios ocultos não são autovetores

Trabalho Tipo 7: Implementar o SOM para Classificação

- Malha (Grade) de rede define um espaço de saída discreto;
- A função de vizinhança definida em torno do neurônio vencedor varia com o tempo;
- A taxa de aprendizagem diminui gradualmente com o tempo

Algoritmo SOM

1. inicialize w's (valores diferentes para j=1,2,…,l)
2. Encontrar neurônio vencedor
   $i(x) = \text{argmin} \; || \; x(n) - w_j(n) \; ||$
3. Atualizar os pesos dos neurônios
   $w_j(n+1) = w_j(n) + \eta \, (n) \, h_{j,i(x)}(n) \, [\, x(n) - w_j(n) \,]$
4. Reduzir os vizinhos e η
5. Ir para 2

Modelo de Kohonen

- Grade Bidimensional de Neurônios

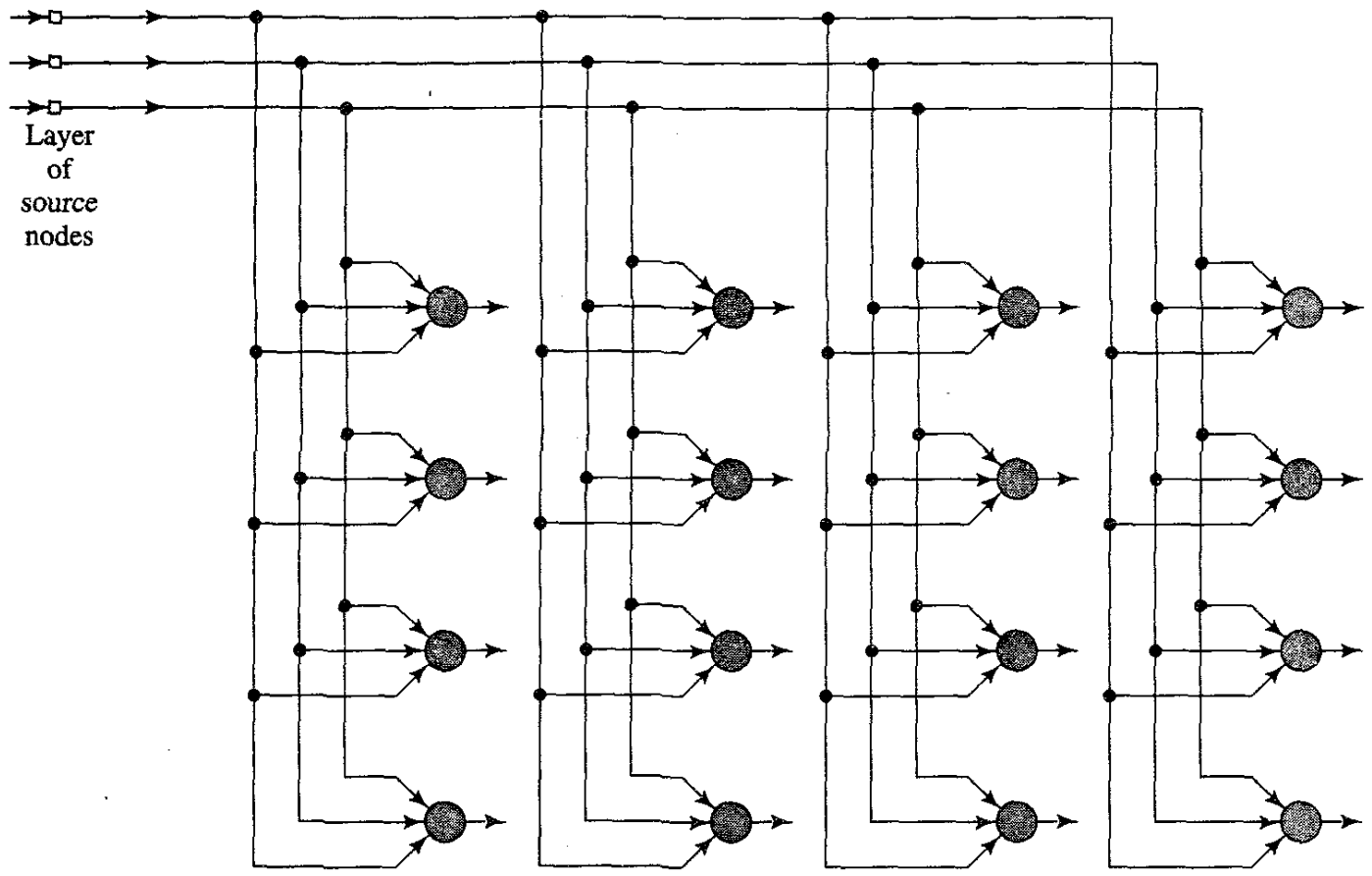


FIGURE 9.2 Two-dimensional lattice of neurons.

Trabalho Tipo 8: RBF aplicada à Regressão. Algoritmo totalmente supervisionado

Erro em batelada:

$$E\left(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}\right) = \frac{1}{2}\sum_t \sum_i \left(r_i^t - y_i^t\right)^2$$

Usando gradiente descendente para os pesos da 2 camada:

$$y_i^t = \sum_{h=1}^{H} w_{ih} p_h^t + w_{i0}$$

$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_i^t\right) p_h^t$$

Atualização dos centros e espalhamento por retropropagação:

$$\Delta m_{hj} = \eta \sum_t \left[\sum_i \left(r_i^t - y_i^t\right) w_{ih}\right] p_h^t \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$

$$\Delta s_h = \eta \sum_t \left[\sum_i \left(r_i^t - y_i^t\right) w_{ih}\right] p_h^t \frac{\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2}{s_h^3}$$

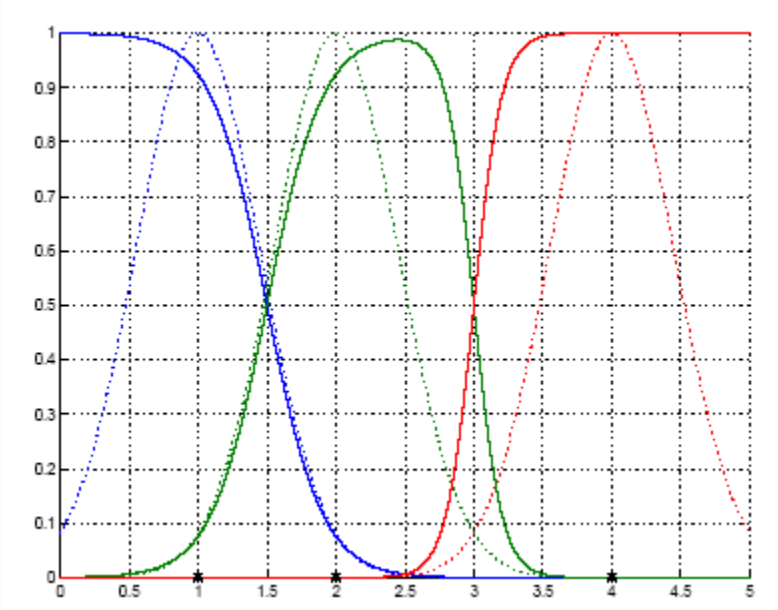
Trabalho Tipo 9: RBF aplicada à Regressão: Funções de Base Normalizadas

Regressão

$$g_h^t = \frac{p_h^t}{\sum_{l=1}^{H} p_l^t}$$

$$= \frac{\exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2 / 2s_h^2\right]}{\sum_l \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_l\right\|^2 / 2s_l^2\right]}$$

$$y_i^t = \sum_{h=1}^{H} w_{ih} g_h^t$$



$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_i^t\right) g_h^t$$

Escolher um algoritmo para calcular espalhamento s_h

$$\Delta m_{hj} = \eta \sum_t \sum_i \left(r_i^t - y_i^t\right)\left(w_{ih} - y_i^t\right) g_h^t \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$

Trabalho Tipo 10: RBF aplicada à Regressão. Funções de Base Competitivas

- Modelo de Misturas: $p(\mathbf{r}^t \mid \mathbf{x}^t) = \sum_{h=1}^{H} p(h \mid \mathbf{x}^t) p(\mathbf{r}^t \mid h, \mathbf{x}^t)$

$$p(h \mid \mathbf{x}^t) = \frac{p(\mathbf{x}^t \mid h) p(h)}{\sum_l p(\mathbf{x}^t \mid l) p(l)}$$

$$g_h^t = \frac{a_h \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_h\right\|^2 / 2s_h^2\right]}{\sum_l a_l \exp\left[-\left\|\mathbf{x}^t - \mathbf{m}_l\right\|^2 / 2s_l^2\right]}$$

Geralmente se ignora a_h

Regressão

$$p\left(\mathbf{r}^t \mid \mathbf{x}^t\right) = \prod_i \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{\left(r_i^t - y_i^t\right)}{2\sigma^2}\right]$$

$$\mathcal{L}\left(\{\mathbf{m}_h, s_h, w_{ih}\}_{i,h} \mid \mathcal{X}\right) = \sum_t \log \sum_h g_h^t \exp\left[-\frac{1}{2}\sum_i \left(r_i^t - y_{ih}^t\right)^2\right]$$

Usando gradiente descendente para maximizar o log da verossimilhança

$y_{ih}^t = w_{ih}$ is the constant fit

$$\Delta w_{ih} = \eta \sum_t \left(r_i^t - y_{ih}^t\right) f_h^t \quad \Delta m_{hj} = \eta \sum_t \left(f_h^t - g_h^t\right) \frac{\left(x_j^t - m_{hj}\right)}{s_h^2}$$

$$f_h^t = \frac{g_h^t \exp\left[-(1/2)\sum_i \left(r_i^t - y_{ih}^t\right)^2\right]}{\sum_l g_l^t \exp\left[-(1/2)\sum_i \left(r_i^t - y_{il}^t\right)^2\right]}$$

$$p(h \mid \mathbf{r}, \mathbf{x}) = \frac{p(h \mid \mathbf{x}) p(\mathbf{r} \mid h, \mathbf{x})}{\sum_l p(l \mid \mathbf{x}) p(\mathbf{r} \mid l, \mathbf{x})} \quad \Rightarrow \quad g_h^t \equiv p(h \mid \mathbf{x}^t)$$

Trabalho Tipo 11: RBF aplicado à Regressão utilizando EM para RBF (EM Supervisionado)

Regressão

- passo-E:

$$f_h^t \equiv p\left(\mathbf{r} \mid h, \mathbf{x}^t\right)$$

- passo-M:

$$\mathbf{m}_h = \frac{\sum_t f_h^t \mathbf{x}^t}{\sum_t f_h^t}$$

$$s_h = \frac{\sum_t f_h^t \left(\mathbf{x}^t - \mathbf{m}_h\right)\left(\mathbf{x}^t - \mathbf{m}_h\right)^T}{\sum_t f_h^t}$$

$$w_{ih} = \frac{\sum_t f_h^t r_i^t}{\sum_t f_h^t}$$