Clase 16: Redes Neuronales

Mauricio Castro C.
mcastro@mat.uc.cl

Departamento de Estadística, Pontificia Universidad Católica de Chile

TÓPICOS APLICADOS EN ESTADÍSTICA

Segundo Semestre 2022

Redes Neuronales Multi-Capa

Perceptrones multicapa

- Minsky and Papert (1969): Perceptrons. An introduction to computational geometry.

- Minsky and Papert (1969): Perceptrons. An introduction to computational geometry.

- Las limitaciones de los perceptrones pueden superarse considerando capas + trasnformaciones no lineales.

Perceptrones multicapa

- Minsky and Papert (1969): Perceptrons. An introduction to computational geometry.
- Las limitaciones de los perceptrones pueden superarse considerando capas + trasnformaciones no lineales.
- Estas propuestas no se podían ejecutar debido a las limitaciones computacionales.

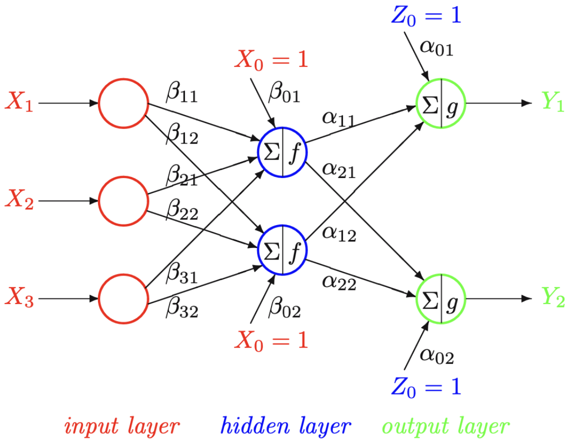
Perceptrones multicapa

- Minsky and Papert (1969): Perceptrons. An introduction to computational geometry.
- Las limitaciones de los perceptrones pueden superarse considerando capas + trasnformaciones no lineales.
- Estas propuestas no se podían ejecutar debido a las limitaciones computacionales.
- Backpropagation algorithm: programación dinámica con dos fases, forward + backward.

Perceptrones multicapa

- Minsky and Papert (1969): Perceptrons. An introduction to computational geometry.
- Las limitaciones de los perceptrones pueden superarse considerando capas + trasnformaciones no lineales.
- Estas propuestas no se podían ejecutar debido a las limitaciones computacionales.
- Backpropagation algorithm: programación dinámica con dos fases, forward + backward.
- Programación dinámica: método de optimización matemática + programación que resuelve un problema grande dividiéndolo en pequeños subproblemas de forma recursiva.

Perceptrones multicapa



- Vector de entrada: $\mathbf{X} = (X_1, \ldots, X_r)^\top$.

Perceptrones multicapa

- Vector de entrada: $\mathbf{X} = (X_1, \ldots, X_r)^\top$.

- Vector de salida: $\mathbf{Y} = (Y_1, \ldots, Y_s)^\top$.

- Vector de entrada: $\mathbf{X} = (X_1, \ldots, X_r)^\top$.
- Vector de salida: $\mathbf{Y} = (Y_1, \ldots, Y_s)^\top$.
- Mapeo: no-lineal.

Perceptrones multicapa

- Vector de entrada: $\mathbf{X} = (X_1, \ldots, X_r)^\top$.
- Vector de salida: $\mathbf{Y} = (Y_1, \ldots, Y_s)^\top$.
- Mapeo: no-lineal.
- La capa oculta y los vectores de salida se llaman nodos o neuronas.

- Vector de entrada: $\mathbf{X} = (X_1, \ldots, X_r)^\top$.

- Vector de salida: $\mathbf{Y} = (Y_1, \ldots, Y_s)^\top$.

- Mapeo: no-lineal.

- La capa oculta y los vectores de salida se llaman nodos o neuronas.

- La figura anterior posee 2 capas (la oculta + output), un vector de entrada de dimensión $r = 3$, un vector de salida de dimensión $s = 2$ y $t = 2$ nodos en la capa oculta.

Perceptrones multicapa

- Regresión múltiple: $s = 1$.

- Regresión múltiple: $s = 1$.

- Regresión multivariada: $s > 1$.

Perceptrones multicapa

- Regresión múltiple: $s = 1$.
- Regresión multivariada: $s > 1$.
- Clasificación binaria: $s = 1$.

- Regresión múltiple: $s = 1$.

- Regresión multivariada: $s > 1$.

- Clasificación binaria: $s = 1$.

- Clasificación multiclase: $s = K - 1$, donde K es el número de clases.

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).
- X_m, $m = 1, 2, \ldots, r$ nodos de entrada.

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).
- X_m, $m = 1, 2, \ldots, r$ nodos de entrada.
- Z_j, $j = 1, \ldots, t$ nodos ocultos.

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).
- X_m, $m = 1, 2, \ldots, r$ nodos de entrada.
- Z_j, $j = 1, \ldots, t$ nodos ocultos.
- Y_k, $k = 1, 2, \ldots, s$ nodos de salida.

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).
- X_m, $m = 1, 2, \ldots, r$ nodos de entrada.
- Z_j, $j = 1, \ldots, t$ nodos ocultos.
- Y_k, $k = 1, 2, \ldots, s$ nodos de salida.
- β_{mj} el peso de la relación entre X_m y Z_j, con sesgo β_{0j}.

Problema de una sola capa oculta

- Suponga una red de dos capas (una oculta + output).
- X_m, $m = 1, 2, \ldots, r$ nodos de entrada.
- Z_j, $j = 1, \ldots, t$ nodos ocultos.
- Y_k, $k = 1, 2, \ldots, s$ nodos de salida.
- β_{mj} el peso de la relación entre X_m y Z_j, con sesgo β_{0j}.
- α_{jk} el peso de la relación entre Z_j y Y_k, con sesgo α_{0k}.

Problema de una sola capa oculta

- $\mathbf{X} = (X_1, \ldots, X_r)^\top$.

Problema de una sola capa oculta

- $\mathbf{X} = (X_1, \ldots, X_r)^\top$.
- $\mathbf{Z} = (Z_1, \ldots, Z_t)^\top$.

Problema de una sola capa oculta

- $\mathbf{X} = (X_1, \ldots, X_r)^\top$.
- $\mathbf{Z} = (Z_1, \ldots, Z_t)^\top$.
- $U_j = \beta_{0j} + \mathbf{X}^\top \boldsymbol{\beta}_j$ y $V_k = \alpha_{0k} + \mathbf{Z}^\top \boldsymbol{\alpha}_k$.

Problema de una sola capa oculta

- $\mathbf{X} = (X_1, \ldots, X_r)^\top$.

- $\mathbf{Z} = (Z_1, \ldots, Z_t)^\top$.

- $U_j = \beta_{0j} + \mathbf{X}^\top \boldsymbol{\beta}_j$ y $V_k = \alpha_{0k} + \mathbf{Z}^\top \boldsymbol{\alpha}_k$.

- Sean $f_j(\cdot)$ y $g_k(\cdot)$ funciones de activación de la capa oculta y la capa de salida, con $j = 1, \ldots, t$ y $k = 1, \ldots, s$. Entonces,

$$\begin{aligned} Z_j &= f_j\left(U_j\right), \quad j = 1, 2, \ldots, t \\ \mu_k(\mathbf{X}) &= g_k\left(V_k\right), \quad k = 1, 2, \ldots, s, \end{aligned}$$

con $\boldsymbol{\beta}_j = (\beta_{1j}, \cdots, \beta_{rj})^\top$ y $\boldsymbol{\alpha}_k = (\alpha_{1k}, \cdots, \alpha_{tk})^\top$.

- Usando notación matricial, tenemos que para el k-ésimo nodo de salida, con $k = 1, \ldots, s$:

- Usando notación matricial, tenemos que para el k-ésimo nodo de salida, con $k = 1, \dots, s$:

$$Y_k = \mu_k(\mathbf{X}) + \epsilon_k,$$

con $\mu_k(\mathbf{X}) = g_k\left(\alpha_{0k} + \sum_{j=1}^{t} \alpha_{jk} f_j\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right)\right).$

Problema de una sola capa oculta

- Usando notación matricial, tenemos que para el k-ésimo nodo de salida, con $k = 1, \ldots, s$:

$$Y_k = \mu_k(\mathbf{X}) + \epsilon_k,$$

con $\mu_k(\mathbf{X}) = g_k \left(\alpha_{0k} + \sum_{j=1}^{t} \alpha_{jk} f_j \left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m \right) \right).$

- $f_j(\cdot)$'s son usualmente logística o tanh (sigmoideas).

Problema de una sola capa oculta

- Usando notación matricial, tenemos que para el k-ésimo nodo de salida, con $k = 1, \ldots, s$:

$$Y_k = \mu_k(\mathbf{X}) + \epsilon_k,$$

con $\mu_k(\mathbf{X}) = g_k\left(\alpha_{0k} + \sum_{j=1}^{t} \alpha_{jk} f_j\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right)\right).$

- $f_j(\cdot)$'s son usualmente logística o tanh (sigmoideas).

- $g_k(\cdot)$'s son usualmente lineales (regresión) o sigmoideas (clasificación).

Problema de una sola capa oculta

- Usando notación matricial, tenemos que para el k-ésimo nodo de salida, con $k = 1, \ldots, s$:

$$Y_k = \mu_k(\mathbf{X}) + \epsilon_k,$$

con $\mu_k(\mathbf{X}) = g_k\left(\alpha_{0k} + \sum_{j=1}^{t} \alpha_{jk} f_j\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right)\right)$.

- $f_j(\cdot)$'s son usualmente logística o tanh (sigmoideas).

- $g_k(\cdot)$'s son usualmente lineales (regresión) o sigmoideas (clasificación).

- ϵ_k es un ruido Gaussiano con media 0 y varianza σ_k.

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.
- Sea $f_j(\cdot) = f(\cdot)$, $\forall j$ *i.e.*, todas iguales y sigmoideas.

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.
- Sea $f_j(\cdot) = f(\cdot)$, $\forall j$ *i.e.*, todas iguales y sigmoideas.
- Entonces, $Y = \mu(\mathbf{X}) + \epsilon$ con

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.
- Sea $f_j(\cdot) = f(\cdot)$, $\forall j$ *i.e.*, todas iguales y sigmoideas.
- Entonces, $Y = \mu(\mathbf{X}) + \epsilon$ con

$$\mu(\mathbf{X}) = \alpha_0 + \sum_{j=1}^{t} \alpha_j f\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right).$$

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.
- Sea $f_j(\cdot) = f(\cdot)$, $\forall j$ *i.e.*, todas iguales y sigmoideas.
- Entonces, $Y = \mu(\mathbf{X}) + \epsilon$ con

$$\mu(\mathbf{X}) = \alpha_0 + \sum_{j=1}^{t} \alpha_j f\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right).$$

- La red anterior equivale al perceptron de una sola capa.

Problema de una sola capa oculta + $s = 1$

- Sea $g(\cdot)$ lineal.
- Sea $f_j(\cdot) = f(\cdot)$, $\forall j$ *i.e.*, todas iguales y sigmoideas.
- Entonces, $Y = \mu(\mathbf{X}) + \epsilon$ con

$$\mu(\mathbf{X}) = \alpha_0 + \sum_{j=1}^{t} \alpha_j f\left(\beta_{0j} + \sum_{m=1}^{r} \beta_{mj} X_m\right).$$

- La red anterior equivale al perceptron de una sola capa.
- Si $f(\cdot)$ y $g(\cdot)$ son lineales, la red es una combinación lineal de inputs.

Arquitectura multicapa

Теорема. *При любом целом* $n \geqslant 2$ *существуют такие определенные на единичном отрезке* $E^1 = [0; 1]$ *непрерывные действительные функции* $\psi^{pq}(x)$, *что каждая определенная на* n*-мерном единичном кубе* E^n *непрерывная действительная функция* $f(x_1, \ldots, x_n)$ *представима в виде*

$$f(x_1, \ldots, x_n) = \sum_{q=1}^{q=2n+1} \chi_q \left[\sum_{p=1}^{n} \psi^{pq}(x_p) \right], \qquad (1)$$

где функции $\chi_q(y)$ *действительны и непрерывны.*

A.N. Kolmogorov: On the representation of continuous functions of many variables by superposition of continuous functions of one variable and addition, Doklady Akademii Nauk SSSR, 1957, Volume 114, Number 5, 953–956

Arquitectura multicapa

- Kolmogorov establece que podrían aproximarse todas las funciones continua a través de una red de dos capas, incluyendo una oculta con un gran número de nodos ocultos, outputs y pesos.

Arquitectura multicapa

- Kolmogorov establece que podrían aproximarse todas las funciones continua a través de una red de dos capas, incluyendo una oculta con un gran número de nodos ocultos, outputs y pesos.

- Mejor la aproximación, más nodos ocultos son requeridos.

Arquitectura multicapa

- Kolmogorov establece que podrían aproximarse todas las funciones continua a través de una red de dos capas, incluyendo una oculta con un gran número de nodos ocultos, outputs y pesos.

- Mejor la aproximación, más nodos ocultos son requeridos.

- El resultado se conoce como el Teorema Universal de Aproximación.

Arquitectura multicapa

- Kolmogorov establece que podrían aproximarse todas las funciones continua a través de una red de dos capas, incluyendo una oculta con un gran número de nodos ocultos, outputs y pesos.

- Mejor la aproximación, más nodos ocultos son requeridos.

- El resultado se conoce como el Teorema Universal de Aproximación.

- El teorema anterior dice que podemos aproximar cualquier función arbitraria con una red neuronal de una capa oculta.

Arquitectura multicapa

- Kolmogorov establece que podrían aproximarse todas las funciones continua a través de una red de dos capas, incluyendo una oculta con un gran número de nodos ocultos, outputs y pesos.
- Mejor la aproximación, más nodos ocultos son requeridos.
- El resultado se conoce como el Teorema Universal de Aproximación.
- El teorema anterior dice que podemos aproximar cualquier función arbitraria con una red neuronal de una capa oculta.
- Sin embargo, no nos dice si esta aproximación es la mejor.

Problerma de más de una capa oculta

- Sea $\boldsymbol{\mu}(\mathbf{X}) = \mathbf{g}\left(\boldsymbol{\alpha}_0 + \mathbf{A}\mathbf{f}\left(\boldsymbol{\beta}_0 + \mathbf{B}\mathbf{X}\right)\right)$.

Problerma de más de una capa oculta

- Sea $\boldsymbol{\mu}(\mathbf{X}) = \mathbf{g}\left(\boldsymbol{\alpha}_0 + \mathbf{A}\mathbf{f}\left(\boldsymbol{\beta}_0 + \mathbf{B}\mathbf{X}\right)\right)$.

- $\mathbf{B} = (\beta_{ij})$ matriz de pesos entre los nodos de entrada y la capa oculta de dimensión $t \times r$.

Problerma de más de una capa oculta

- Sea $\boldsymbol{\mu}(\mathbf{X}) = \mathbf{g}\left(\boldsymbol{\alpha}_0 + \mathbf{A}\mathbf{f}\left(\boldsymbol{\beta}_0 + \mathbf{B}\mathbf{X}\right)\right)$.

- $\mathbf{B} = (\beta_{ij})$ matriz de pesos entre los nodos de entrada y la capa oculta de dimensión $t \times r$.

- $\mathbf{A} = (\alpha_{jk})$ matriz de pesos entre las capa oculta y el output de dimensión $s \times t$.

Problerma de más de una capa oculta

- Sea $\boldsymbol{\mu}(\mathbf{X}) = \mathbf{g}\left(\boldsymbol{\alpha}_0 + \mathbf{A}\mathbf{f}\left(\boldsymbol{\beta}_0 + \mathbf{B}\mathbf{X}\right)\right)$.

- $\mathbf{B} = (\beta_{ij})$ matriz de pesos entre los nodos de entrada y la capa oculta de dimensión $t \times r$.

- $\mathbf{A} = (\alpha_{jk})$ matriz de pesos entre las capa oculta y el output de dimensión $s \times t$.

- $\boldsymbol{\beta}_0 = (\beta_{01}, \cdots, \beta_{0t})^\top$, $\boldsymbol{\alpha}_0 = (\alpha_{01}, \cdots, \alpha_{0s})^\top$, $\mathbf{f} = (f_1, \cdots, f_t)^\top$ y $\mathbf{g} = (g_1, \cdots, g_s)^\top$ (funciones de activación no lineales).

Problerma de más de una capa oculta

- Un caso especial de lo anterior resulta cuando $f_j(\cdot)$ y $g_k(\cdot)$ son funciones identidad.

Problerma de más de una capa oculta

- Un caso especial de lo anterior resulta cuando $f_j(\cdot)$ y $g_k(\cdot)$ son funciones identidad.

- Aqui lo anterior se reduce a $\boldsymbol{\mu}(\mathbf{X}) = \mu + \mathbf{ABX}$ con $\boldsymbol{\mu} = \boldsymbol{\alpha}_0 + \mathbf{A}\boldsymbol{\beta}_0$.

Problerma de más de una capa oculta

- Un caso especial de lo anterior resulta cuando $f_j(\cdot)$ y $g_k(\cdot)$ son funciones identidad.

- Aqui lo anterior se reduce a $\boldsymbol{\mu}(\mathbf{X}) = \mu + \mathbf{ABX}$ con $\boldsymbol{\mu} = \boldsymbol{\alpha}_0 + \mathbf{A}\boldsymbol{\beta}_0$.

- Este modelo es conocido como el modelo de regresión de rango reducido (Izenman, 1975, Journal of Multivariate Analysis, 5:248-264).

Problerma de más de una capa oculta

- Un caso especial de lo anterior resulta cuando $f_j(\cdot)$ y $g_k(\cdot)$ son funciones identidad.

- Aqui lo anterior se reduce a $\boldsymbol{\mu}(\mathbf{X}) = \mu + \mathbf{ABX}$ con $\boldsymbol{\mu} = \boldsymbol{\alpha}_0 + \mathbf{A}\boldsymbol{\beta}_0$.

- Este modelo es conocido como el modelo de regresión de rango reducido (Izenman, 1975, Journal of Multivariate Analysis, 5:248-264).

- La diferencia con los modelos de regresión tradicional es que el modelo de rango reducido impone que el rango de $\mathbf{AB}$ sea menor que $\min\{r, s\}$.

Algoritmo Backpropagation - Fase Forward

- Los inputs ingresan a la red neuronal.

Algoritmo Backpropagation - Fase Forward

- Los inputs ingresan a la red neuronal.

- Dados los pesos actuales, se calculan las todas las componentes de las capas de la red.

Algoritmo Backpropagation - Fase Forward

- Los inputs ingresan a la red neuronal.
- Dados los pesos actuales, se calculan las todas las componentes de las capas de la red.
- El output se compara con el valor real.

Algoritmo Backpropagation - Fase Forward

- Los inputs ingresan a la red neuronal.
- Dados los pesos actuales, se calculan las todas las componentes de las capas de la red.
- El output se compara con el valor real.
- Se calcula la derivada de la función de perdida con respecto del output.

Algoritmo Backpropagation - Fase Forward

- Los inputs ingresan a la red neuronal.
- Dados los pesos actuales, se calculan las todas las componentes de las capas de la red.
- El output se compara con el valor real.
- Se calcula la derivada de la función de perdida con respecto del output.
- Esta derivada debe ser calculada con respecto a los pesos en todas las capas de la red.

Algoritmo Backpropagation - Fase Backward

- Obtener el gradiente de la función de pérdida con respecto de los pesos.

- Obtener el gradiente de la función de pérdida con respecto de los pesos.
- Para esto se necesita usar la regla de la cadena.

Algoritmo Backpropagation - Fase Backward

- Obtener el gradiente de la función de pérdida con respecto de los pesos.
- Para esto se necesita usar la regla de la cadena.
- Los gradientes se obtienen hacia atrás, empezando por el output.

Algoritmo Backpropagation - Fase Backward

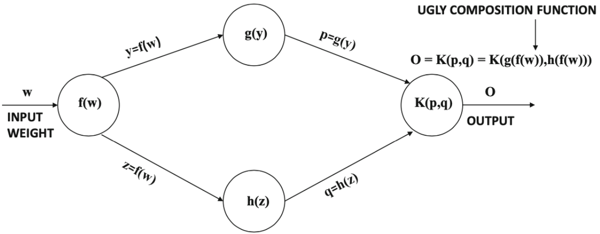
- Obtener el gradiente de la función de pérdida con respecto de los pesos.

- Para esto se necesita usar la regla de la cadena.

- Los gradientes se obtienen hacia atrás, empezando por el output.

- Por esto, la fase se llama Backward.

Algoritmo Backpropagation



$$\begin{aligned}
\frac{\partial o}{\partial w} &= \frac{\partial o}{\partial p}\cdot\frac{\partial p}{\partial w} + \frac{\partial o}{\partial q}\cdot\frac{\partial q}{\partial w} \quad \text{[Multivariable Chain Rule]}\\
&= \frac{\partial o}{\partial p}\cdot\frac{\partial p}{\partial y}\cdot\frac{\partial y}{\partial w} + \frac{\partial o}{\partial q}\cdot\frac{\partial q}{\partial z}\cdot\frac{\partial z}{\partial w} \quad \text{[Univariate Chain Rule]}\\
&= \underbrace{\frac{\partial K(p,q)}{\partial p}\cdot g'(y)\cdot f'(w)}_{\text{First path}} + \underbrace{\frac{\partial K(p,q)}{\partial q}\cdot h'(z)\cdot f'(w)}_{\text{Second path}}
\end{aligned}$$

Algoritmo Backpropagation

- Sea $\mathcal{M}$ el conjunto de los r nodos de entrada, $\mathcal{J}$ el conjunto de los t nodos ocultos y $\mathcal{K}$ es el conjunto de los s nodos output.

Algoritmo Backpropagation

- Sea $\mathcal{M}$ el conjunto de los r nodos de entrada, $\mathcal{J}$ el conjunto de los t nodos ocultos y $\mathcal{K}$ es el conjunto de los s nodos output.

- Sea el i-ésimo vector de entrada de la capa oculta

$$V_{i,k} = \sum_{j \in \mathcal{J}} \alpha_{kj} Z_{i,j} = \alpha_{k0} + \mathbf{Z}_i^\tau \boldsymbol{\alpha}_k, \quad k \in \mathcal{K},$$

con $\mathbf{Z}_i = (Z_{i,1}, \ldots, Z_{i,t})^\top$, $\boldsymbol{\alpha}_k = (\alpha_{k1}, \ldots, \alpha_{kt})^\top$ y $Z_{i,0} = 1$.

Algoritmo Backpropagation

- Sea $\mathcal{M}$ el conjunto de los r nodos de entrada, $\mathcal{J}$ el conjunto de los t nodos ocultos y $\mathcal{K}$ es el conjunto de los s nodos output.

- Sea el i-ésimo vector de entrada de la capa oculta

$$V_{i,k} = \sum_{j \in \mathcal{J}} \alpha_{kj} Z_{i,j} = \alpha_{k0} + \mathbf{Z}_i^{\tau} \boldsymbol{\alpha}_k, \quad k \in \mathcal{K},$$

con $\mathbf{Z}_i = (Z_{i,1}, \ldots, Z_{i,t})^{\top}$, $\boldsymbol{\alpha}_k = (\alpha_{k1}, \ldots, \alpha_{kt})^{\top}$ y $Z_{i,0} = 1$.

- El output aquí se define como $\tilde{Y}_{i,k} = g_k(V_{i,k}), \quad k \in \mathcal{K}$.

Algoritmo Backpropagation

- Sea $\mathcal{M}$ el conjunto de los r nodos de entrada, $\mathcal{J}$ el conjunto de los t nodos ocultos y $\mathcal{K}$ es el conjunto de los s nodos output.

- Sea el i-ésimo vector de entrada de la capa oculta

$$V_{i,k} = \sum_{j \in \mathcal{J}} \alpha_{kj} Z_{i,j} = \alpha_{k0} + \mathbf{Z}_i^{\tau} \boldsymbol{\alpha}_k, \quad k \in \mathcal{K},$$

con $\mathbf{Z}_i = \left(Z_{i,1}, \ldots, Z_{i,t}\right)^{\top}$, $\boldsymbol{\alpha}_k = \left(\alpha_{k1}, \ldots, \alpha_{kt}\right)^{\top}$ y $Z_{i,0} = 1$.

- El output aquí se define como $\tilde{Y}_{i,k} = g_k\left(V_{i,k}\right), \quad k \in \mathcal{K}$.

- La función de activación $g_k(\cdot)$ se asume diferenciable.

- Sea $e_{i,k} = Y_{i,k} - \tilde{Y}_{i,k}$, $k \in \mathcal{K}$ y la función de error

$$E_i = \frac{1}{2} \sum_{k \in \mathcal{K}} e_{i,k}^2 = \frac{1}{2} \sum_{k \in \mathcal{K}} \left(Y_{i,k} - \tilde{Y}_{i,k} \right)^2,$$

con $i = 1, \ldots, n$.

Algoritmo Backpropagation

- Sea $e_{i,k} = Y_{i,k} - \tilde{Y}_{i,k}$, $k \in \mathcal{K}$ y la función de error

$$E_i = \frac{1}{2} \sum_{k \in \mathcal{K}} e_{i,k}^2 = \frac{1}{2} \sum_{k \in \mathcal{K}} \left(Y_{i,k} - \tilde{Y}_{i,k}\right)^2,$$

con $i = 1, \ldots, n$.

- Buscamos minimizar $\mathrm{ESS} = n^{-1} \sum_{i=1}^{n} E_i$.

Algoritmo Backpropagation

- Sea $e_{i,k} = Y_{i,k} - \tilde{Y}_{i,k}$, $k \in \mathcal{K}$ y la función de error

$$E_i = \frac{1}{2}\sum_{k\in\mathcal{K}} e_{i,k}^2 = \frac{1}{2}\sum_{k\in\mathcal{K}}\left(Y_{i,k} - \tilde{Y}_{i,k}\right)^2,$$

  con $i = 1, \ldots, n$.

- Buscamos minimizar $\mathrm{ESS} = n^{-1}\sum_{i=1}^{n} E_i$.

- Actualizamos $\boldsymbol{\alpha}_i$ de acuerdo a: $\boldsymbol{\alpha}_{i+1} = \boldsymbol{\alpha}_i + \Delta\boldsymbol{\alpha}_i$, donde $\Delta\boldsymbol{\alpha}_i = -\eta\frac{\partial E_i}{\partial \boldsymbol{\alpha}_i} = \left(-\eta\frac{\partial E_i}{\partial \alpha_{i,jh}}\right) = (\Delta\alpha_{i,kj})$ y $(\alpha_{i,kj})$ denota el vector de dimensión ts de los pesos de la capa oculta.

- Usando la regla de la cadena,

$$
\begin{aligned}
\frac{\partial E_i}{\partial \alpha_{i,kj}} &= \frac{\partial E_i}{\partial e_{i,k}} \cdot \frac{\partial e_{i,k}}{\partial \widetilde{Y}_{i,k}} \cdot \frac{\partial \tilde{Y}_{i,k}}{\partial V_{i,k}} \cdot \frac{\partial V_{i,k}}{\partial \alpha_{i,kj}} \\
&= e_{i,k} \cdot (-1) \cdot g'_k \left(V_{i,k}\right) \cdot Z_{i,j} \\
&= -e_{i,k} g'_k \left(\alpha_{i,k0} + \mathbf{Z}_i^{\tau} \boldsymbol{\alpha}_{i,k}\right) Z_{i,j}
\end{aligned}
$$

Algoritmo Backpropagation

- Usando la regla de la cadena,

$$\begin{aligned}
\frac{\partial E_i}{\partial \alpha_{i,kj}} &= \frac{\partial E_i}{\partial e_{i,k}} \cdot \frac{\partial e_{i,k}}{\partial \widetilde{Y}_{i,k}} \cdot \frac{\partial \tilde{Y}_{i,k}}{\partial V_{i,k}} \cdot \frac{\partial V_{i,k}}{\partial \alpha_{i,kj}} \\
&= e_{i,k} \cdot (-1) \cdot g_k'\left(V_{i,k}\right) \cdot Z_{i,j} \\
&= -e_{i,k} g_k'\left(\alpha_{i,k0} + \mathbf{Z}_i^{\tau} \alpha_{i,k}\right) Z_{i,j}
\end{aligned}$$

- La actualización de $\alpha_{i,kj}$ usando el descenso por gradiente se puede expresar como

$$\alpha_{i+1,kj} = \alpha_{i,kj} - \eta \frac{\partial E_i}{\partial \alpha_{i,kj}} = \alpha_{i,kj} + \eta \delta_{i,k} Z_{i,j},$$

donde $\delta_{i,k} = -\frac{\partial E_i}{\partial \widetilde{Y}_{i,k}} \cdot \frac{\partial \tilde{Y}_{i,k}}{\partial V_{i,k}} = e_{i,k} g_k'\left(V_{i,k}\right)$.

Algoritmo Backpropagation

- En nuestra notación δ_{ik} es el producto entre e_{ik} y la derivada de $g_k(\cdot)$.

Algoritmo Backpropagation

- En nuestra notación δ_{ik} es el producto entre e_{ik} y la derivada de $g_k(\cdot)$.

- δ_{ik} se conoce también como sensibilidad o gradiente local.

Algoritmo Backpropagation

- En nuestra notación δ_{ik} es el producto entre e_{ik} y la derivada de $g_k(\cdot)$.

- δ_{ik} se conoce también como sensibilidad o gradiente local.

- η es el parámetro de aprendizaje del algoritmo.

Algoritmo Backpropagation

- En nuestra notación δ_{ik} es el producto entre e_{ik} y la derivada de $g_k(\cdot)$.
- δ_{ik} se conoce también como sensibilidad o gradiente local.
- η es el parámetro de aprendizaje del algoritmo.
- La siguiente parte del algoritmo busca la actualización de los pesos que conectan la capa oculta con la capa de entrada.

Algoritmo Backpropagation

- Sea $U_{i,j} = \sum_{m \in \mathcal{M}} \beta_{i,jm} X_{i,m} = \beta_{i,j0} + \mathbf{X}_i^\top \boldsymbol{\beta}_{i,j}$, $j \in \mathcal{J}$, con $\mathbf{X}_i = (X_{i,1}, \cdots, X_{i,r})^\top$, $\boldsymbol{\beta}_{i,j} = (\beta_{i,j1}, \cdots, \beta_{i,jr})^\top$, y $X_{i,0} = 1$.

Algoritmo Backpropagation

- Sea $U_{i,j} = \sum_{m\in\mathcal{M}} \beta_{i,jm} X_{i,m} = \beta_{i,j0} + \mathbf{X}_i^\top \boldsymbol{\beta}_{i,j}$, $j \in \mathcal{J}$, con $\mathbf{X}_i = (X_{i,1}, \cdots, X_{i,r})^\top$, $\boldsymbol{\beta}_{i,j} = (\beta_{i,j1}, \cdots, \beta_{i,jr})^\top$, y $X_{i,0} = 1$.
- El output es $Z_{ij} = f_j(U_{ij})$.

Algoritmo Backpropagation

- Sea $U_{i,j} = \sum_{m \in \mathcal{M}} \beta_{i,jm} X_{i,m} = \beta_{i,j0} + \mathbf{X}_i^\top \boldsymbol{\beta}_{i,j}$, $j \in \mathcal{J}$, con $\mathbf{X}_i = (X_{i,1}, \cdots, X_{i,r})^\top$, $\boldsymbol{\beta}_{i,j} = (\beta_{i,j1}, \cdots, \beta_{i,jr})^\top$, y $X_{i,0} = 1$.

- El output es $Z_{ij} = f_j(U_{ij})$.

- Sea $\boldsymbol{\beta}_i = \left(\boldsymbol{\beta}_{i,1}^\top, \cdots, \boldsymbol{\beta}_{i,t}^\top\right)^\top = (\beta_{i,jm})$ el vector $(r+1)t$ de pesos entre la capa de entrada y la oculta.

Algoritmo Backpropagation

- Sea $U_{i,j} = \sum_{m \in \mathcal{M}} \beta_{i,jm} X_{i,m} = \beta_{i,j0} + \mathbf{X}_i^\top \boldsymbol{\beta}_{i,j}$, $j \in \mathcal{J}$, con $\mathbf{X}_i = (X_{i,1}, \cdots, X_{i,r})^\top$, $\boldsymbol{\beta}_{i,j} = (\beta_{i,j1}, \cdots, \beta_{i,jr})^\top$, y $X_{i,0} = 1$.

- El output es $Z_{ij} = f_j(U_{ij})$.

- Sea $\boldsymbol{\beta}_i = \left(\boldsymbol{\beta}_{i,1}^\top, \cdots, \boldsymbol{\beta}_{i,t}^\top\right)^\top = (\beta_{i,jm})$ el vector $(r+1)t$ de pesos entre la capa de entrada y la oculta.

- La actualización de estos pesos está dada por $\boldsymbol{\beta}_{i+1} = \boldsymbol{\beta}_i + \Delta\boldsymbol{\beta}_i$, donde $\Delta\boldsymbol{\beta}_i = -\eta \frac{\partial E_i}{\partial \boldsymbol{\beta}_i} = \left(-\eta \frac{\partial E_i}{\partial \beta_{i,jm}}\right) = (\Delta\beta_{i,jm})$.

Algoritmo Backpropagation

- Sea $U_{i,j} = \sum_{m\in\mathcal{M}} \beta_{i,jm} X_{i,m} = \beta_{i,j0} + \mathbf{X}_i^\top \boldsymbol{\beta}_{i,j}$, $j \in \mathcal{J}$, con $\mathbf{X}_i = (X_{i,1}, \cdots, X_{i,r})^\top$, $\boldsymbol{\beta}_{i,j} = (\beta_{i,j1}, \cdots, \beta_{i,jr})^\top$, y $X_{i,0} = 1$.
- El output es $Z_{ij} = f_j(U_{ij})$.
- Sea $\boldsymbol{\beta}_i = \left(\boldsymbol{\beta}_{i,1}^\top, \cdots, \boldsymbol{\beta}_{i,t}^\top\right)^\top = (\beta_{i,jm})$ el vector $(r+1)t$ de pesos entre la capa de entrada y la oculta.
- La actualización de estos pesos está dada por $\boldsymbol{\beta}_{i+1} = \boldsymbol{\beta}_i + \Delta\boldsymbol{\beta}_i$, donde $\Delta\boldsymbol{\beta}_i = -\eta\frac{\partial E_i}{\partial \boldsymbol{\beta}_i} = \left(-\eta\frac{\partial E_i}{\partial \beta_{i,jm}}\right) = (\Delta\beta_{i,jm})$.
- Al igual que la parte anterior, se utiliza la regla de la cadena para determinar $\frac{\partial E_i}{\partial \boldsymbol{\beta}_i}$.

Algoritmo Backpropagation

- No hay una demostración formal de la convergencia del algoritmo.

Algoritmo Backpropagation

- No hay una demostración formal de la convergencia del algoritmo.
- El algoritmo es lento y los estimadores muchas veces inestables.

Algoritmo Backpropagation

- No hay una demostración formal de la convergencia del algoritmo.
- El algoritmo es lento y los estimadores muchas veces inestables.
- Existen problemas de identificabilidad en los pesos.

Algoritmo Backpropagation

- No hay una demostración formal de la convergencia del algoritmo.
- El algoritmo es lento y los estimadores muchas veces inestables.
- Existen problemas de identificabilidad en los pesos.
- En la práctica se deja correr el algoritmo hasta que se estabilice.

Algoritmo Backpropagation

- No hay una demostración formal de la convergencia del algoritmo.
- El algoritmo es lento y los estimadores muchas veces inestables.
- Existen problemas de identificabilidad en los pesos.
- En la práctica se deja correr el algoritmo hasta que se estabilice.
- Valores grandes de η aceleran la convergencia.

Consideraciones

- Modos de aprendizaje.

Consideraciones

- Modos de aprendizaje.

- Re-escalamiento.

Consideraciones

- Modos de aprendizaje.
- Re-escalamiento.
- Cantidad de capas ocultas y nodos ocultos.

Consideraciones

- Modos de aprendizaje.
- Re-escalamiento.
- Cantidad de capas ocultas y nodos ocultos.
- Inicialización de los pesos.

Consideraciones

- Modos de aprendizaje.
- Re-escalamiento.
- Cantidad de capas ocultas y nodos ocultos.
- Inicialización de los pesos.
- Poda y sobreajuste.