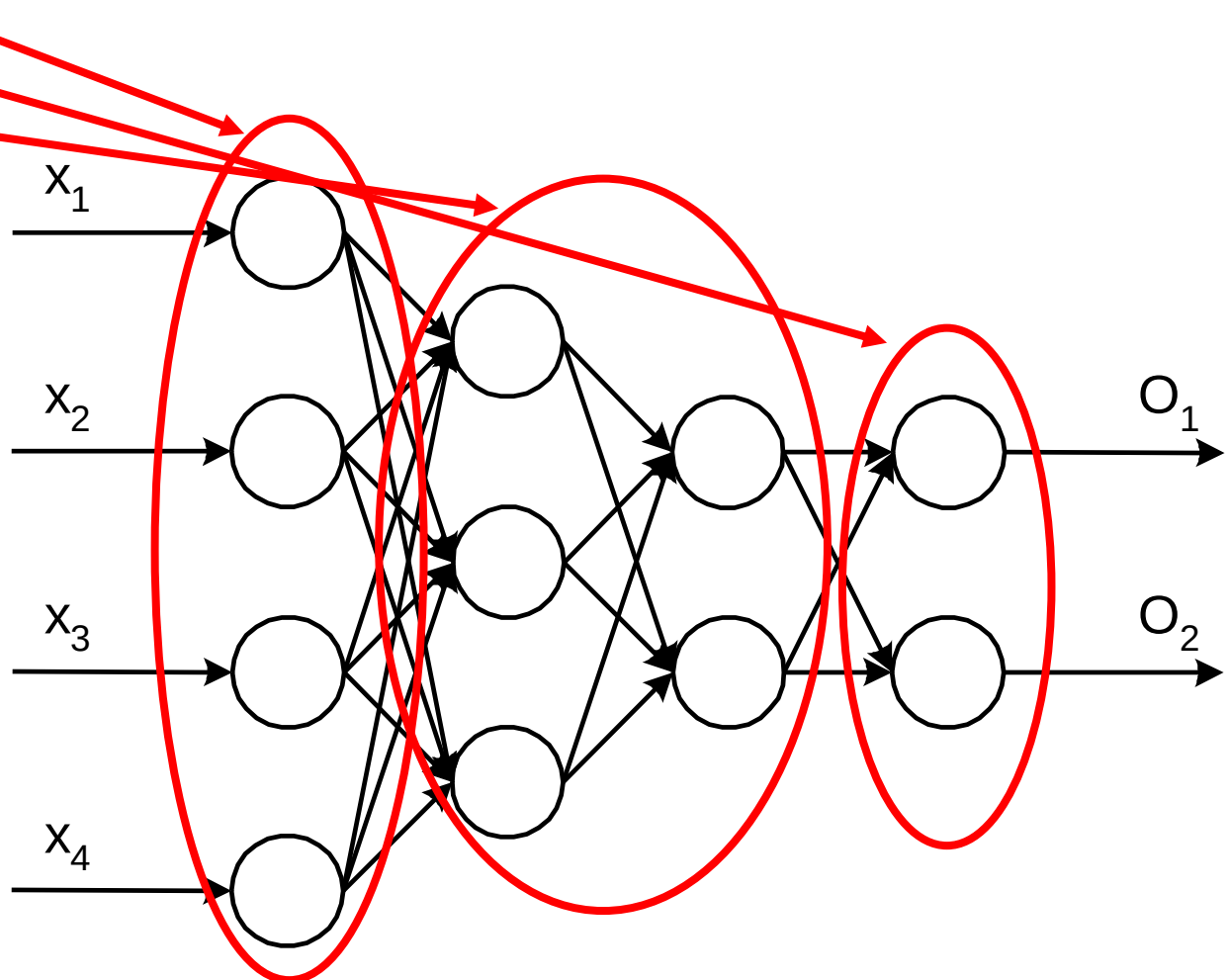
Modeli neuronskih mreža

paradigme,
veštačke neuronske mreže

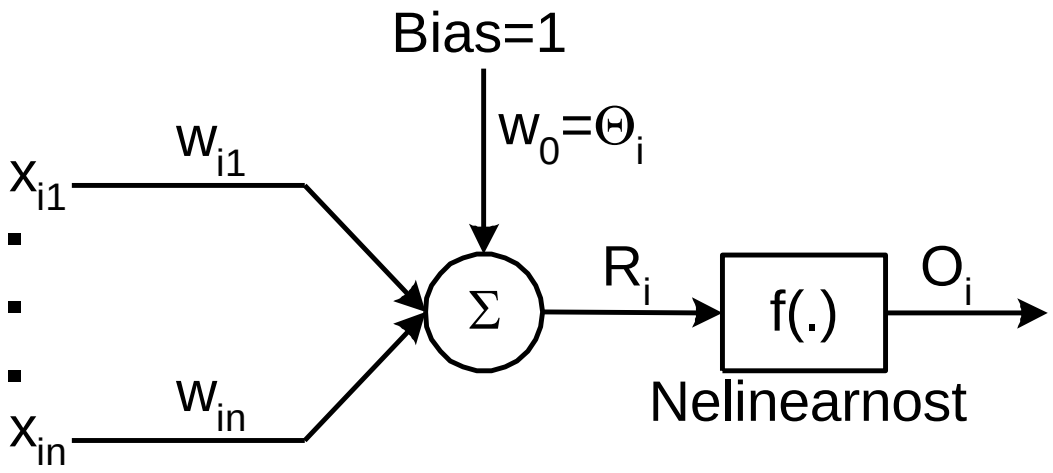
Osnovni pojmovi

- Osnovna podela VNM prema strukturi – broju slojeva:
  - jednoslojne
  - višeslojne
- Prema topologiji, slojevi se nazivaju:
  - ulazni
  - izlazni
  - skriveni

Višeslojna veštačka neuronska mreža, sa prostiranjem signala u napred.



McCulloh – Pitt-ov model



Prvi pokušaj da se modeluje biološki neuron.

Nema obuke, ni adaptacije parametara.

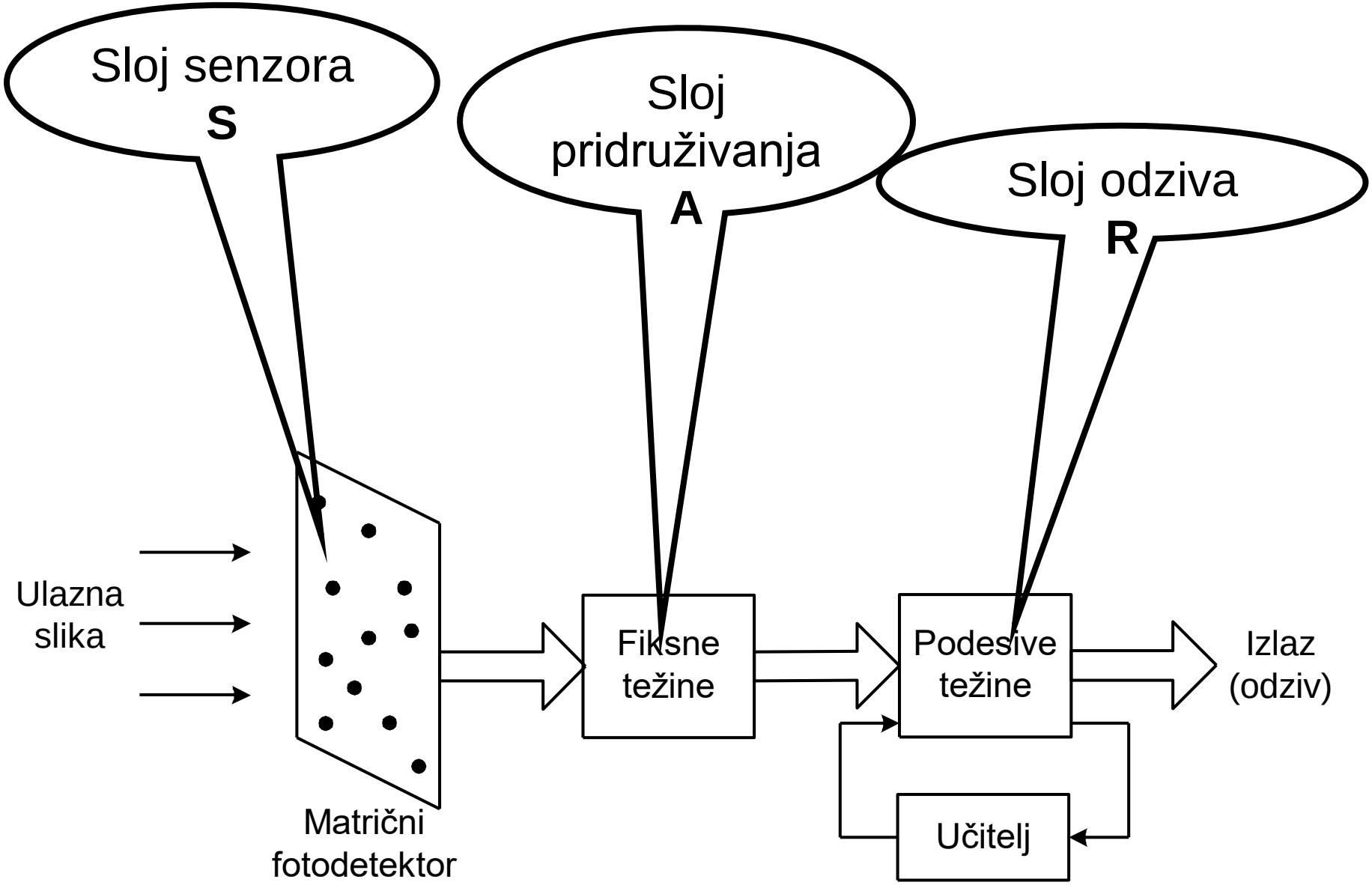
Izlaz je definisan izrazom: $$O_i = f\left(\sum_{i=1}^{n}(x_{ij}w_{ij}) - \Theta_i\right)$$

Perceptron

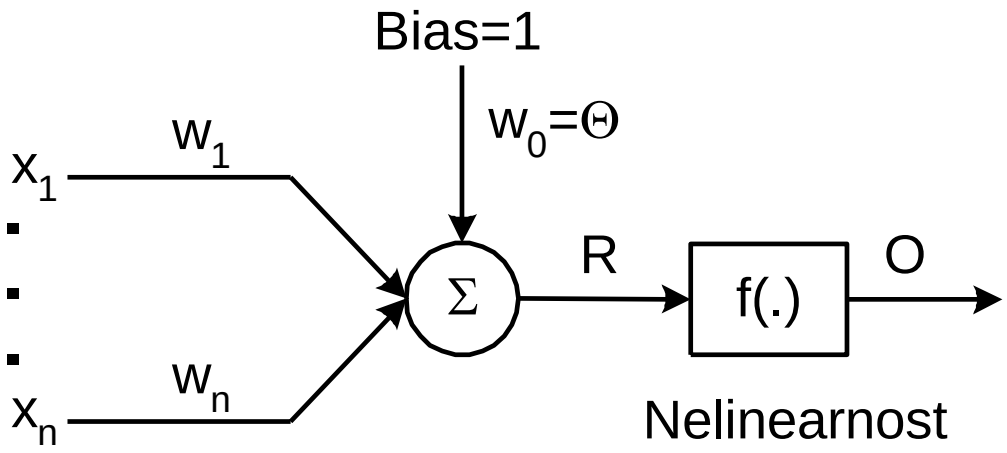
- Naredna generacija veštačkih neuronskih mreža.
- Bazira se na McCulloh-Pitt-ovom modelu uz dodavanje povratne sprege, mogućnosti učenja i adaptacije.
- Njihov nastanak se poklapa sa početkom razvoja adaptivnog upravljanja.
- Model perceptrona Franka Rosenblatta, 1958.

Originalni perceptron

- Model neuronske mreže koji zahteva učenje sa nadzorom.
- Razvijen je kao klasifikator uzoraka koji raspoznaje abstraktne i geometrijske oblike dovedene na optički ulaz.
- Rosenblatt-ov originalni perceptron se sastoji iz tri nivoa:
  - sloj senzora (*S* - sensor)
  - sloj pridruživanja (*A* - association)
  - sloj odziva (*R* - response)


Sloj senzora
S
Sloj pridruživanja
A
Sloj odziva
R
Ulazna slika
Matrični fotodetektor
Fiksne težine
Podesive težine
Izlaz (odziv)
Učitelj

Model pojedinačnog perceptrona



Greška izlaza perceptrona: $\boldsymbol{E} = \boldsymbol{T} - \boldsymbol{O}$

Korekcija težina sinapsi se računa prema izrazu:

$$\Delta \boldsymbol{w} = \mu[\boldsymbol{T} - f(\boldsymbol{w}(k)\boldsymbol{x})]\boldsymbol{x}$$

odnosno:

$$\boldsymbol{w}(k+1) = \boldsymbol{w}(k) + \mu[\boldsymbol{T} - f(\boldsymbol{w}(k)\boldsymbol{x})]\boldsymbol{x}$$

Obuka perceptrona

Pre početka obuke potrebno je izvršiti pripremne korake koji su slični za sve modele neuronskih mreža:

1. Odrediti skup ulaznih vektora {$\boldsymbol{x}$}.
2. Odrediti skup željenih izlaza {$\boldsymbol{T}$}, gde jedan izlazni podatak odgovara jednom ulaznom.
3. Izabrati mali pozitivan broj kao korak učenja (μ) i kriterijum po kome će se taj parametar menjati tokom obuke.
4. Izabrati nelinearnost (aktivacionu funkciju neurona) i ako je potrebno njene parametre.
5. Odrediti kriterijum za završetak obuke (maksimalna dozvoljena greška izlaza, maksimalni broj iteracija, itd.)

Nakon završetka pripreme, prelazi se na obuku:

1. Izabrati inicijalne vrednosti za pragove (Θ_i) i težine (w_{ij}) kao male slučajne vrednosti (najčešće sa intervala [-1,+1]).
2. Dovesti na ulaz uzorak x_p i odgovarajući izlaz T_p, gde je p redni broj tekućeg uzorka.
3. Izračunati tekući izlaz O, prema izrazu:

$$O(k) = f\left[\sum_{j=0}^{n} w_j(k)x_j\right] = f\left[\mathbf{w}^T(k)\mathbf{x}\right]$$

4. Izvršiti adaptaciju težina prema iterativnoj relaciji:

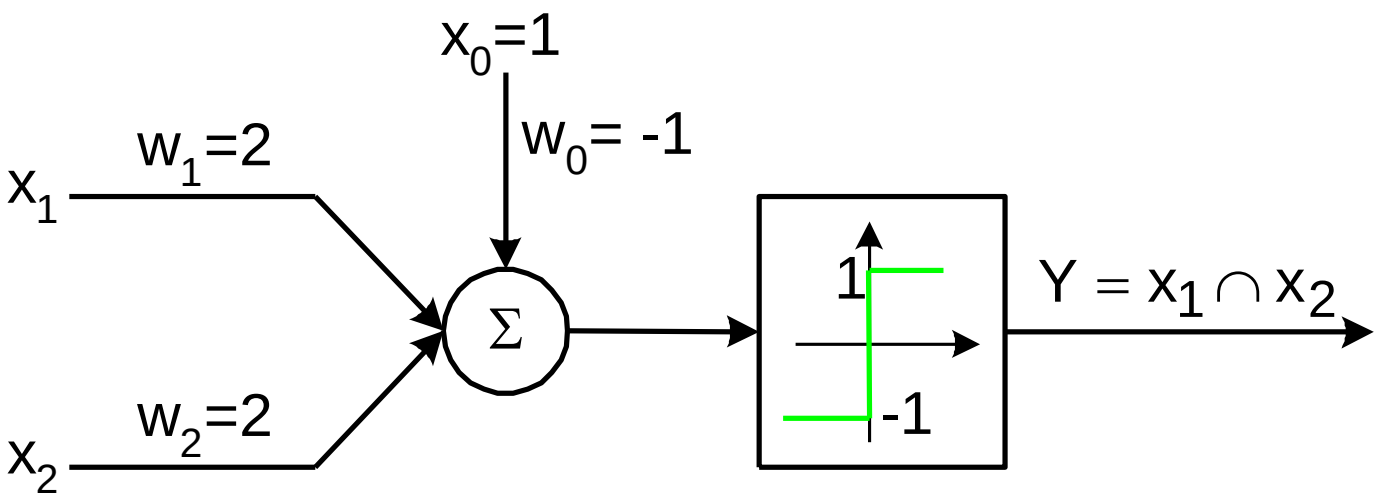
$$w(k+1) = w(k) + \mu[T(k) - f(w(k)x)]x$$

   Kada se težine prestanu menjati dostignut je željeni izlaz.
5. Ponoviti korake 2-4.

Logičke operacije sa jednoslojnim perceptronom

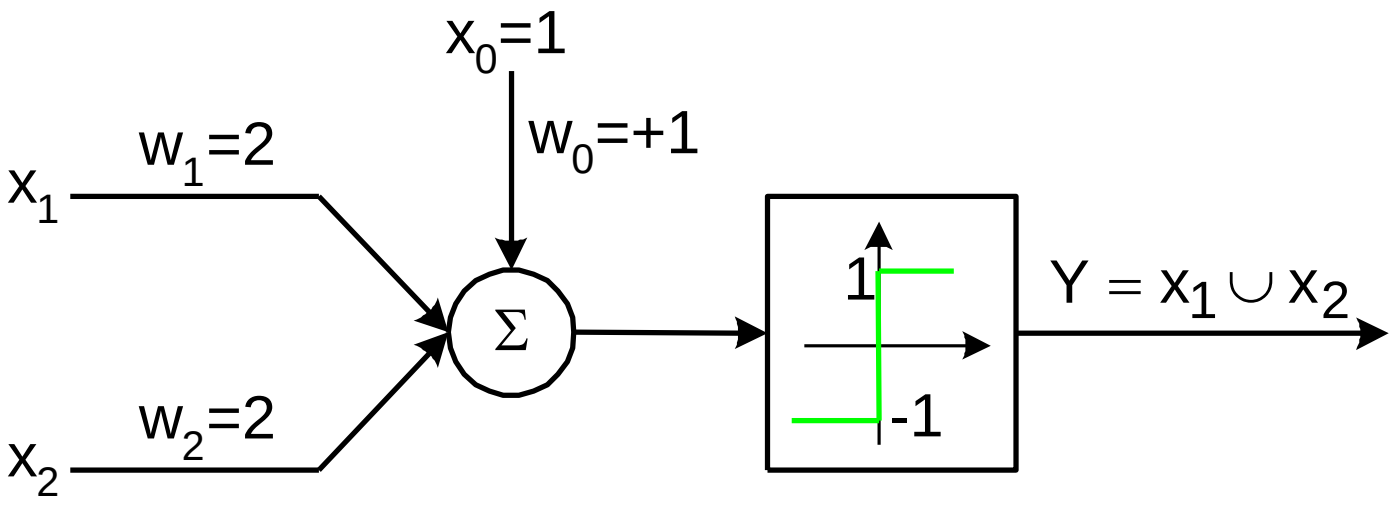
- Jednoslojni perceptron se može smatrati klasifikatorom za rešavanje problema razgraničenja dve klase podataka u okviru nekog uzorka.
- Jednoslojni perceptron je, takođe, i binarna logička jedinica. Isti perceptron, u zavisnosti od vrednosti njegovih parametara, može implementirati različite logičke funkcije, ali ne sve.
  - Perceptron sa dva ulaza može da implementira funkcije I, ILI i NE, ali ne i ekskluzivno-ILI.
  - Za implementaciju višeznačne funkcije *pluralnosti*, potreban je perceptron sa tri ulaza.

Logička operacija “I”



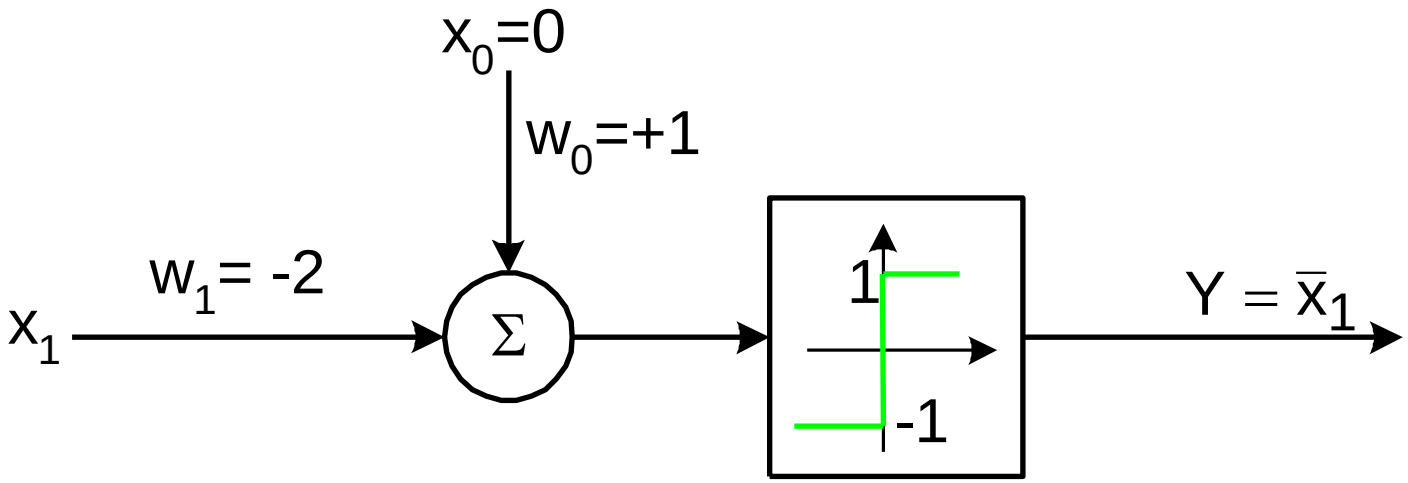
x_1	x_2	Y
-1	-1	-1
-1	+1	-1
+1	-1	-1
+1	+1	+1

Logička operacija “ILI”



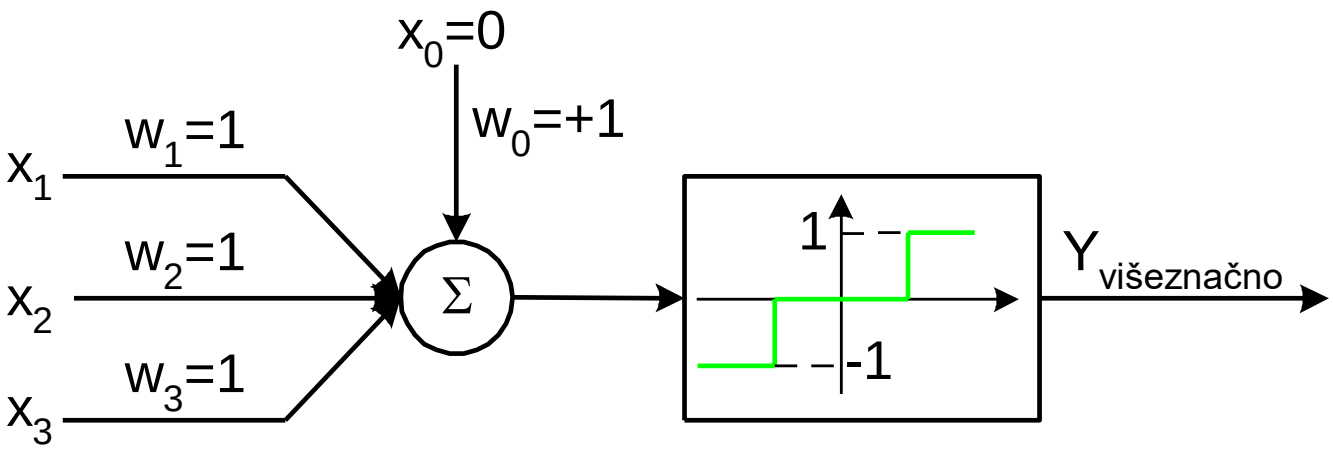
x_1	x_2	Y
-1	-1	-1
-1	+1	+1
+1	-1	+1
+1	+1	+1

Logička operacija “NE”



x_1	Y
-1	+1
+1	-1

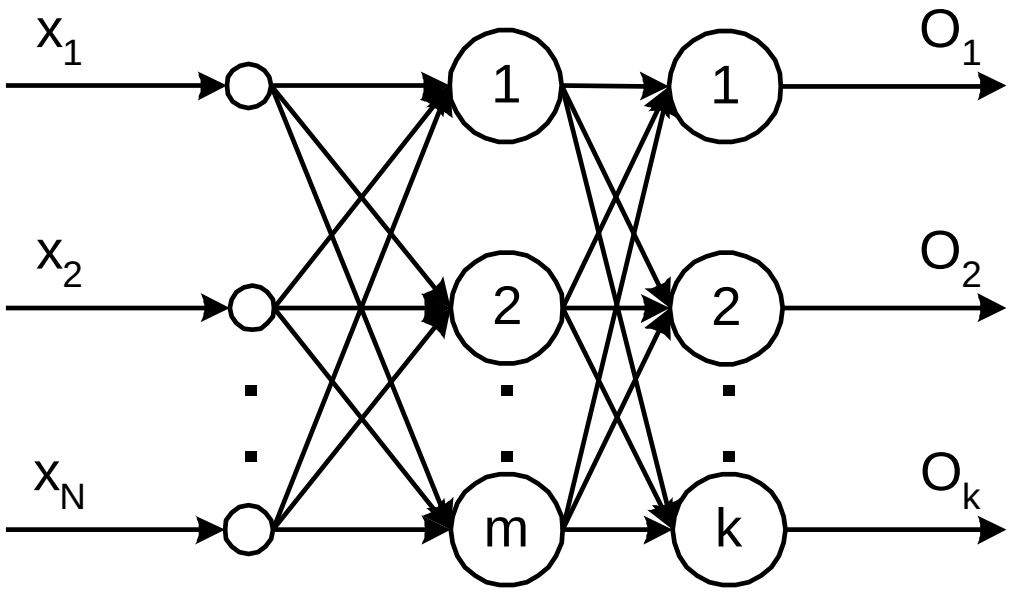
Logička operacija “PLURALNOST”



Bilo koji ulaz			*Y*
0	0	0	0
0	-1	+1	0
0	0	+1	+1
0	+1	+1	+1
+1	+1	+1	+1
-1	+1	+1	+1
0	0	-1	-1
0	-1	-1	-1
-1	-1	-1	-1
+1	-1	-1	-1

Višeslojni perceptron

- Višeslojni perceptron predstavlja mrežu većeg broja običnih perceptrona koji čine hijerarhijsku strukturu sa prostiranjem signala u napred.
- Osnovni perceptroni su organizovani u slojeve, kojih između ulaznog i izlaznog može biti više.
- Svaki perceptron u datoj mreži naziva se neuron.
- Broj skrivenih slojeva i broj neurona na slojevima nije fiksan.
- Svaki sloj može imati različit broj neurona, u zavisnosti od namene.
- Projektant VNM treba da odredi broj slojeva i broj neurona na njima. Za različite primene upotrebljavaju se različite strukture.
- Uobičajeni postupci obuke višeslojnog perceptrona su:
  - Delta pravilo
  - Prostiranje greške u nazad (Back-propagation)



Sigmoidna aktivaciona funkcija: $f_S(R) = \frac{1}{1+e^{-kR}}$

Prvi izvod funkcije: $f_S'(R) = \frac{ke^{-kR}}{\left(1+e^{-kR}\right)^2} = kf_S(R)\left(1-f_S(R)\right)$

Višeslojni perceptron se može koristiti za:

- ✓ implementaciju proizvoljne Bulove logičke funkcije koja vrši podelu prostora uzoraka (dovoljna su dva sloja);
- ✓ klasifikacija uzoraka (dovoljna su dva ili tri sloja);
- ✓ implementacija nelinearnih funkcija (dovoljna su dva ili tri sloja).

U prethodnim primerima primene date su preporuke o broju slojeva VNM za rešavanje određene klase problema.

Što se tiče broja neurona po slojevima, važi sledeće:

- ✓ ne postoji formalan način da se odredi optimalan broj neurona po sloju;
- ✓ taj problem se rešava metodom probe i greške.

Delta pravilo

Delta pravilo se bazira na metodi minimizacije kvadrata greške

Cilj ovog metoda jeste da izrazi razliku između željenog i stvarnog izlaza preko ulaza i težina VNM.

Za početak, na osnovu stvarnog (***O***) i željenog (***T***) izlaza iz VNM se definiše kvadrat greške (***E***), na sledeći način:

$$\mathbf{E} = \frac{1}{2}(\mathbf{T_i} - \mathbf{O_i})^2 = \frac{1}{2}[\mathbf{T_i} - \mathrm{f}(\mathbf{w_i x_i})]^2$$

gde je:
$\boldsymbol{w}_i$ – matrična prezentacija skupa težina *i*-tog neurona;
$\boldsymbol{x}_i$ – vektor ulaza *i*-tog neurona;
$\boldsymbol{O}_i$ – vektor stvarnih izlaza *i*-tog neurona;
$\boldsymbol{T}_i$ – vektor željenih izlaza *i*-tog neurona;
f – aktivaciona funkcija (nelinearnost) *i*-tog neurona.

Vektor gradijenta greške je:

$$\nabla \mathbf{E} = -\left(\mathbf{T_i} - \mathbf{O_i}\right) f'\left(\mathbf{w_i x_i}\right)\mathbf{x_i}$$

Pošto se traži minimum greške, korekcija težina se vrši na osnovu negativnog gradijenta:

$$\Delta \mathbf{w_i} = -\mu \nabla \mathbf{E}$$

gde je μ pozitivna konstanta.

Na osnovu gornja dva izraza može se pisati:

$$\Delta \mathbf{w_i} = \mu\left(\mathbf{T_i} - \mathbf{O_i}\right) f'\left(\mathbf{w_i x_i}\right)\mathbf{x_i}$$

Korigovane vrednosti težina u iteraciji (k+1) su:

$$\mathbf{w_i}(k+1) = \mathbf{w_i}(k) + \mu\left(\mathbf{T_i} - \mathbf{O_i}\right) f'\left(\mathbf{w_i x_i}\right)\mathbf{x_i}$$

Generalizovano Delta pravilo

Ako se kao aktivaciona funkcija neurona usvoji sigmoida, sa parametrom *k*=1, njen prvi izvod se može napisati u obliku:

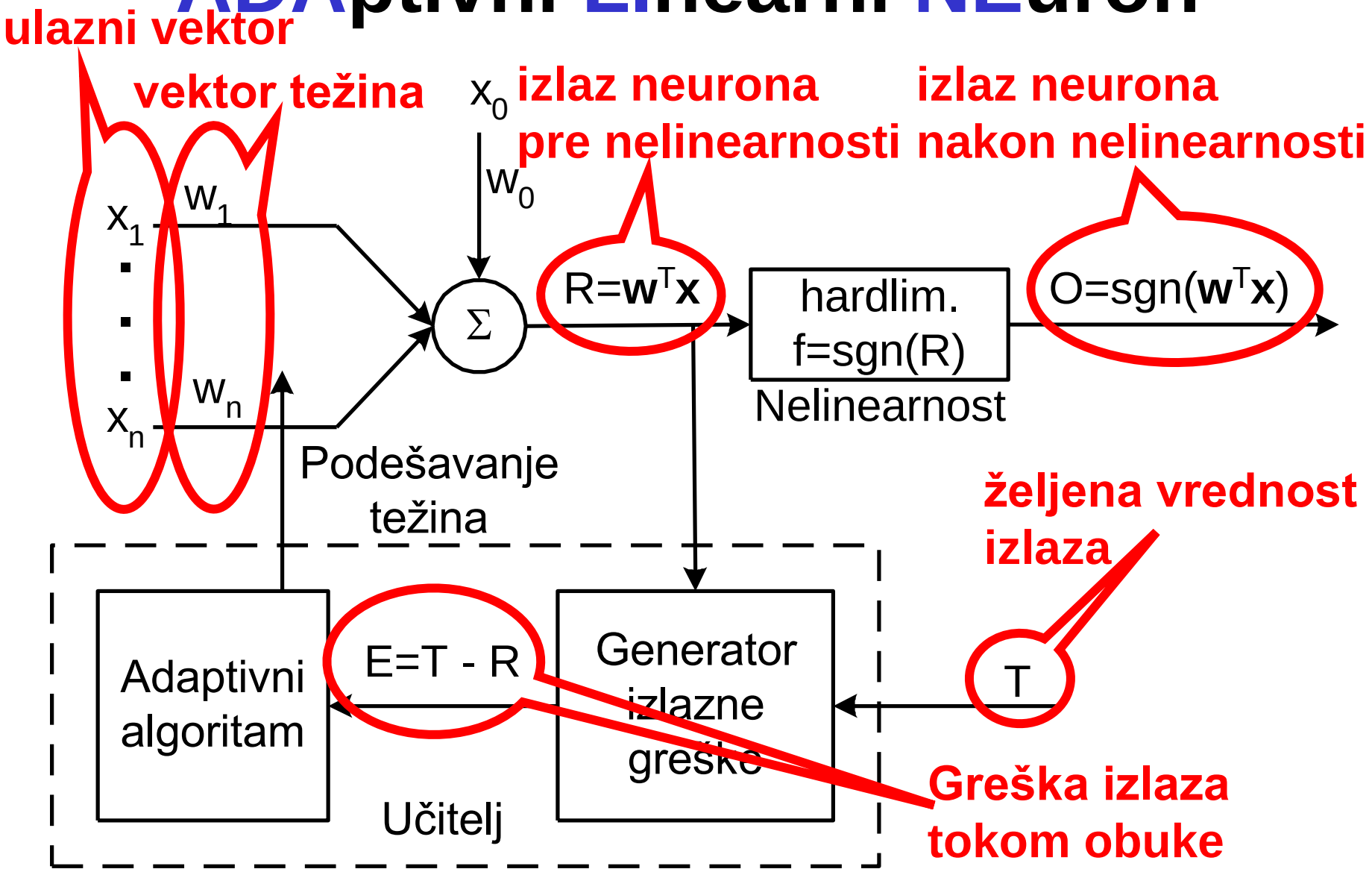
$$f'\left(w_i x_i\right)=\left(O_i-O_i^2\right)$$

Korigovane vrednosti težina u iteraciji (k+1) su sada:

$$w_i(k+1)=w_i(k)+\mu\left(T_i-O_i\right)\left(O_i-O_i^2\right)x_i$$

ADALINE

ADAptivni LInearni NEuron



- Na ulaz ADALINE se dovode dve vrednosti (+1,-1) – bipolarni ulaz.
- Težine sinapsi su pozitivne i negativne (inicijalno male slučajne vrednosti)
- Izlaz iz ADALINE uzima vrednosti iz skupa {+1,-1}.
- Tokom obuke, upoređuju se željena vrednost izlaza (*T*) i stvarna vrednost izlaza neurona (*R*) **pre nelinearnosti**.
- Obuka se bazira na postupku minimizacije srednje kvadratne greške.

Greška izlaza ADALINE: **E** = **T** - **R**

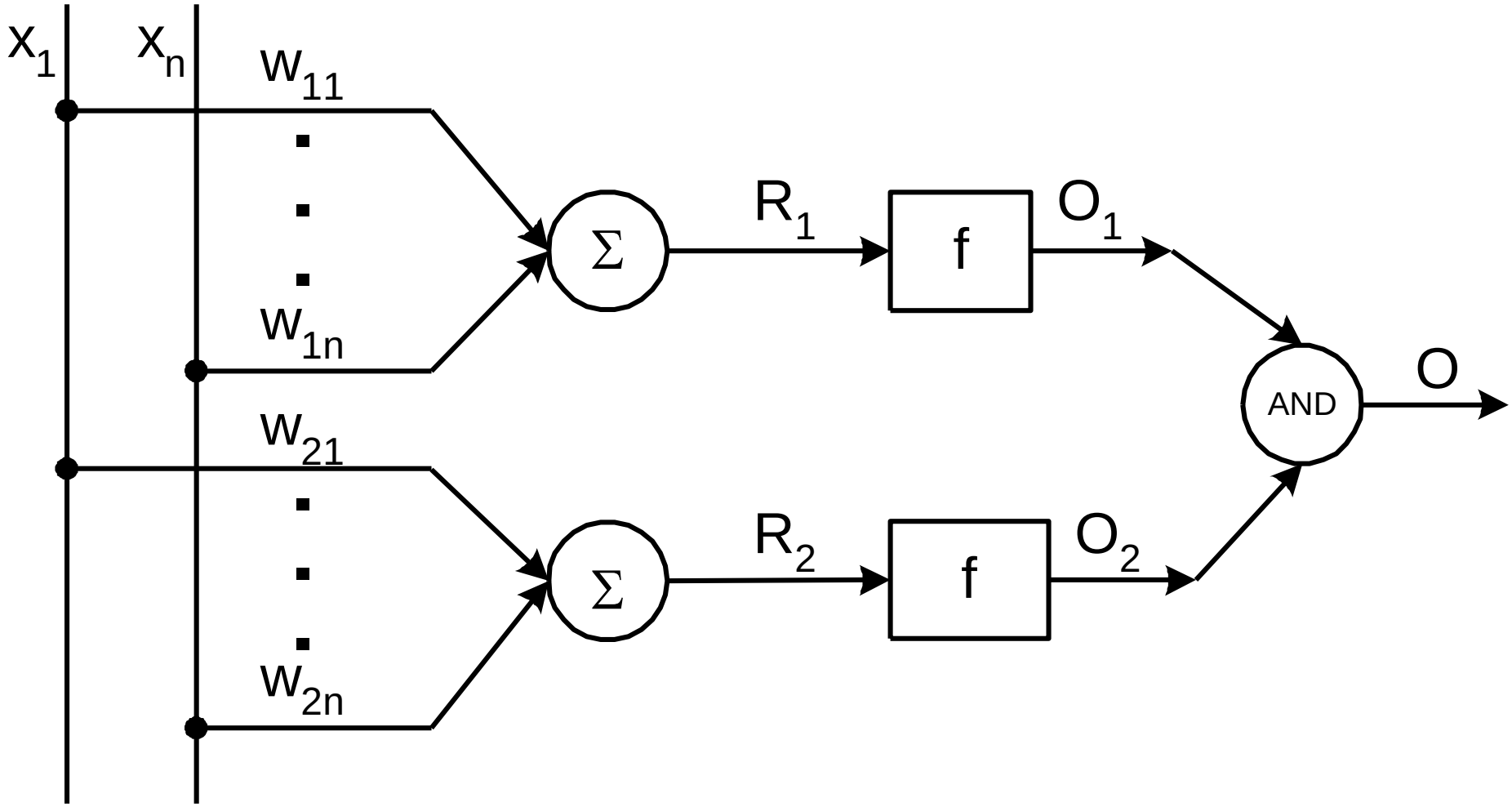
Korekcija težina sinapsi se računa prema izrazu:

$$\Delta w_{ij} = \alpha(\mathbf{T} - \mathbf{R})\mathbf{X} \qquad 0<\alpha<1$$

MADALINE
Many ADALINE

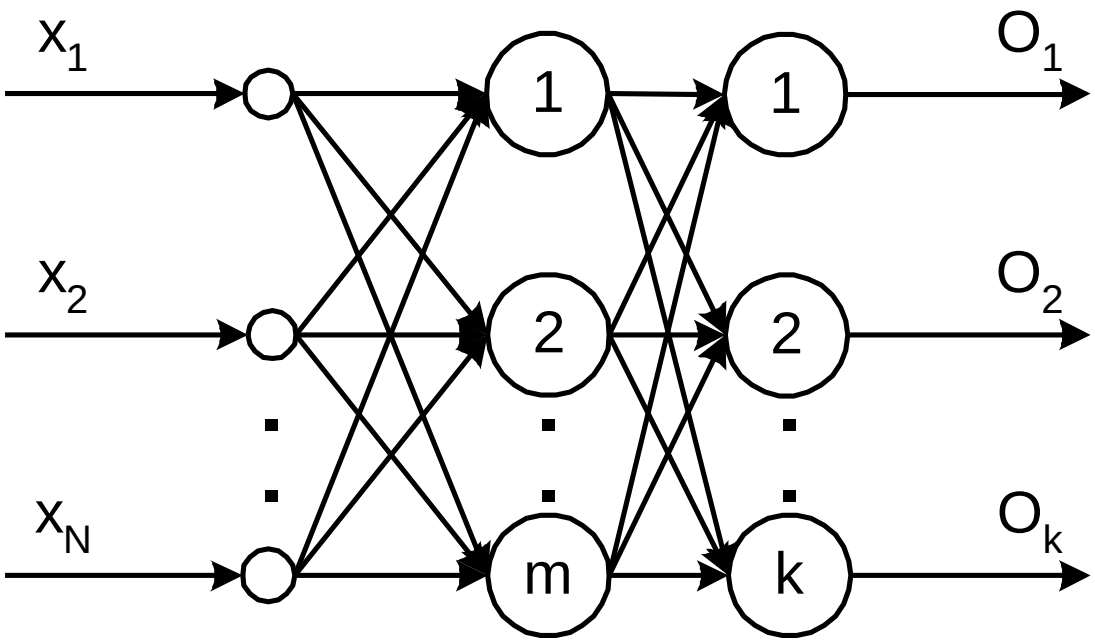
- MADALINE se formira povezivanjem u paralelni rad više ADALINE-a.
- Svakoj od ADALINE-a se na isti ulaz dovodi isti signal.
- MADALINE ima jedan izlaz, koji se formira od izlaza svih ADALINE-a primenom određenih pravila. Ako se primeni pravilo “I”, tada će izlaz iz MADALINE imati vrednost +1(tačno) ako izlazi svih ADALINE imaju vrednost +1(tačno).

Na sledećoj slici je prikazana MADALINE koja se sastoji od dve ADALINE, kojima se na ulaz dovode isti signali, i čiji su izlazi povezani funkcijom “I”.


x1
xn
w11
w1n
w21
w2n
Σ
Σ
R1
R2
f
f
O1
O2
AND
O

Algoritam prostiranja greške u nazad
Back-Propagation (BP) Learning Algorithm

- Paul Werbos, 1974
- D.Rumelhart i D.Parker
- Najčešće korišćeni algoritam obuke za višeslojne VNM sa prostiranjem signala u napred (višeslojni perceptron).
- Pomoću ovog algoritma se vrši mapiranje, odnosno uspostavljanje veze između ulaznih i izlaznih podataka skupa za obuku.
- VNM obučena ovim algoritmom je sposobna da aproksimira funkcije sa visokim stepenom nelinearnosti.



- Veći broj slojeva predstavlja problem tokom obuke.
- Obukom se vrši korekcija težina sinapsi koje povezuju neurone na svim slojevima, pa i skrivenim.
- Korekcija težina se vrši na osnovu greške izlaza neurona.
- Greška na izlaznim neuronima se računa lako, ali kako odrediti grešku izlaza neurona skrivenog sloja?
- Ovaj problem je rešen u okviru BP algoritma.

Obuka VNM pomoću BP algoritma

- Iterativni postupak;
- Jednostavan za primenu na računaru;
- Koriste se dva skupa uzoraka (podataka):

  $\boldsymbol{X}_k$ – skup (matrica) ulaznih uzoraka;

  $\boldsymbol{T}_k$ – skup (matrica) željenih izlaza (odziva)
- Aktuelna vrednost izlaza ($\boldsymbol{O}_k$) zavisi od težina sinapsi svih neurona u mreži.
- Greška izlaza ($\boldsymbol{E}_k$) takođe zavisi od težina sinapsi svih neurona u mreži.
- **Cilj obuke je odrediti skup težina sinapsi VNM za koji će greška izlaza biti minimalna.**
- Brzina i tačnost procesa obuke zavisi od faktora – korak učenja.

Pre početka obuke potrebno je odrediti ili usvojiti:

- **Obučavajući skup** (skup ulaznih i izlaznih podataka);
- Vrednost **koraka učenja**;
- Kriterijum za **prekid algoritma**;
- Način **korekcije težina** sinapsi;
- **Aktivacionu funkciju** (najčešća nelinearnost je sigmoida);
- Inicijalne vrednosti **težina sinapsi** (obično mali slučajni brojevi)

Nakon ovoga počinje proces obuke. **Vektor ulaznih podataka X_k se prenosi kroz VNM od ulaza ka izlazu, a greška E od izlaza ka ulazu.**

Matematička analiza procesa obuke

Posmatra se VNM sa prostiranjem signala u napred i sledećim parametrima:

VNM se sastoji od L slojeva i N_l neurona na sloju l;

w^l_{ij} – težina između i-tog neurona na sloju l+1 i j-tog neurona na sloju l;

$O^l_j(\boldsymbol{x}_p)$ – aktuelni izlaz j-tog neurona na sloju l za p-ti ulazni uzorak (nakon nelinearnosti, aktivacione funkcije);

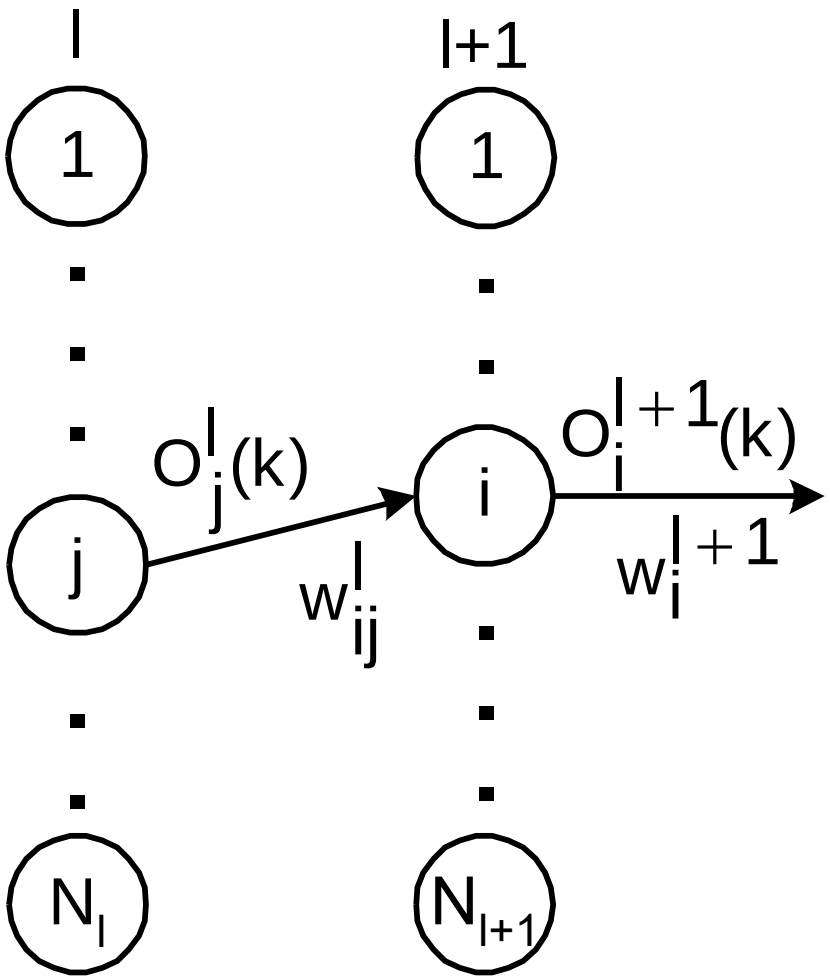
$T^L_j(\boldsymbol{x}_p)$ – željeni izlaz j-tog neurona na sloju L za p-ti ulazni uzorak;

$R^l_j(\boldsymbol{x}_p)$ – aktivacioni izlaz j-tog neurona na sloju l za p-ti ulazni uzorak (pre nelinearnosti, aktivacione funkcije);

$\boldsymbol{P}$ – skup za obuku;

$\boldsymbol{x}_p$ – p-ti obučavajući uzorak, p-ti elemenat skupa za obuku.

U cilju ilustracije BP algoritma posmatra se *i*-ti neuron na sloju *l*+1 koji prima signale od *j*-tog neurona sa sloja *l*, preko težine w^l_{ij}.



Izlaz *i*-tog neurona sa sloja *l*+1, za *k*-ti ulazni obučavajući vektor je opisan izrazom:

$$O_i^{l+1}(k) = f\left(\sum_{j=1}^{N_l} w_{ij}^{l} O_j^{l}(k) - \theta_i^{l+1}\right) = f\left(\sum_{j=1}^{N_l+1} w_{ij}^{l} O_j^{l}(k)\right)$$

Ako se za aktivacionu funkciju usvoji sigmoida, važe izrazi:

$$f(x) = \frac{1}{1+e^{-\beta x}} \qquad f'(x) = \beta f(x)(1-f(x))$$

Ukupna greška (*E*) mreže, za ceo obučavajući skup sa *K* uzoraka se definiše kao suma kvadrata greški svih neurona sa izlaznog sloja *L*:

$$E = \sum_{k=1}^{K} E_k = \sum_{k=1}^{K}\left(\frac{1}{2}\sum_{i=1}^{N_L}\left[T_i(k) - O_i^{L}(k)\right]^2\right)$$

Cilj je odrediti skup svih težina VNM koji minimizira *E*

Pravilo obuke definiše da je promena težina srazmerna negativnom gradijentu greške izlaza:

$$\Delta w_{nm}^{l} \approx -\frac{\partial E_k}{\partial w_{nm}^{l}}$$

Da bi se odredila vrednost prethodnog izraza primenjuje se pravilo ulančavanja izvoda:

$$\frac{\partial E_k}{\partial w_{nm}^{l}} = \frac{\partial E_k}{\partial O_i^{L}(k)} \frac{\partial O_i^{L}(k)}{\partial w_{nm}^{l}}$$

Što se može napisati u obliku:

$$-\frac{\partial E_k}{\partial w_{nm}^{l}} = \sum_{i=1}^{N_L} \left(T_i(k) - O_i^{L}(k) \right) \frac{\partial O_i^{L}(k)}{\partial w_{nm}^{l}}$$

Pošto je pretpostavljena sigmoidna aktivaciona funkcija, za $l=L-1$ (težine izlaznog sloja) može se pisati sledeći izraz:

Na osnovu prethodnog izraza se piše:

$$-\frac{\partial E_k}{\partial w^l_{nm}} = \left(T_n - O^L_n\right)\beta O^L_n\left(1 - O^L_n\right)O^{L-1}_m$$

pa je izraz za korekciju težina sinapsi neurona izlaznog sloja:

$$\Delta w^L_{nm} \equiv \eta\left[\left(T_n - O^L_n\right)O^L_n\left(1 - O^L_n\right)\right]O^{L-1}_m$$

korak učenja

Ako je sada $l \neq L-1$, O^{L-1}_m i dalje zavisi od w^l_{nm}, i zavisnost greške od težina se ponovo može odrediti, primenom pravila ulančavanja:

$$-\frac{\partial E_k}{\partial w^l_{nm}} = \sum_{i=1}^{N_L}\left(T_i - O^L_i\right)f'\left(O^L_i\right)\sum_{j=1}^{N_{L-1}+1} w^{L-1}_{ij}\frac{\partial O^{L-1}_j}{\partial w^l_{nm}}$$

Ako je l=L-2 (poslednji skriveni sloj), prethodna jednačina se može napisti u obliku:

$$-\frac{\partial E_k}{\partial w_{nm}^{l}} = \sum_{i=1}^{N_L}\left(T_i - O_i^L\right) f'\left(O_i^L\right) w_{in}^{L-1} f'\left(O_n^{L-1}\right) O_m^{L-2}$$

odnosno:

$$-\frac{\partial E_k}{\partial w_{nm}^{l}} = f'\left(O_n^{L-1}\right)\left[\sum_{i=1}^{N_L}\left(T_i - O_i^L\right) f'\left(O_i^L\right) w_{in}^{L-1}\right] O_m^{L-2}$$

pa je izraz za korekciju težina sinapsi neurona poslednjeg skrivenog sloja:

$$\Delta w_{nm}^{L-2} = \eta\left[f'\left(O_n^{L-1}\right)\sum_{i=1}^{N_L}\left(T_i - O_i^L\right) f'\left(O_i^L\right) w_{in}^{L-1}\right] O_m^{L-2}$$

Prethodni izraz se može napisati u malo jednostavnijoj formi:

$$\Delta w_{ij}^{l} = \eta \delta_i^{l} O_j^{l-1}$$

Gde je, za neurone izlaznog sloja: $\delta_i^{L} = \left(T_i - O_i^{L} \right) O_i^{L} \left(1 - O_i^{L} \right)$

a za neurone skrivenog sloja: $\delta_i^{l} = \left[\sum_{r=1}^{N_l} \delta_r^{l+1} w_{ri}^{l+1} \right] O_i^{l} \left(1 - O_i^{l} \right)$

Proces određivanja gradijenata i korekcija težina se ponavlja dok funkcija greške *E* ne dostigne svoj minimum.

Na osnovu prethodnih izraza se uočava da faktor δ_i^{l} zavisi od greške određene na sloju *l+1*, zbog čega se ovaj metod i zove metod prostiranja greške u nazad.

Primena BP algoritma

Priprema:

1. Odrediti funkciju VNM (regresija, klasifikacija, prepoznavanje oblika, ...);
2. Formirati skup za obuku (kompletan skup ulazno-izlaznih podataka);
3. Odrediti broj slojeva i broj neurona po svakom sloju;
4. Izabrati aktivacione funkcije (nelinearnosti) po slojevima;
5. Odrediti kriterijum prekida algoritma.

Primena BP algoritma

Primena:

1. Inicijalizacija – izabrati vrednosti svih težina kao male slučajne brojeve;
2. Izabrati par za obuku ($\boldsymbol{x}(k)$,$\boldsymbol{T}(k)$);
3. Izračunati izlaze svih neurona ($O^l_j(k)$) na svim slojevima počevši od ulaznog, preko skrivenih, pa do izlaznog;
4. Izračunati gradijente δ_i i korekcije težina Δw^l_{ij} za svaki ulaz svih neurona, počevši od izlaznog sloja i pomerajući se unazad prema ulaznom;
5. Izvršiti korekciju vrednosti težina;
6. Ponavljati korake 2-5 dok se ne steknu uslovi za završetak algoritma.

Winner-Takes-All algoritam

- Algoritam kompetitivne obuke bez nadzora.
- Pretpostavka je da VNM ima jedan sloj sa *N* neurona, i da svakom neuronu odgovara vektor težina $\boldsymbol{w}_i$.
- Svaki čvor se pobuđuje istim setom ulaznih podataka, vektorom $\boldsymbol{x}_{j(mx1)}$, tako da je odziv *i*-tog neurona

$$O_i = \sum_{j=1}^{m} w_{ij} x_j$$

- Čvor sa "najboljim" odzivom za dati ulazni vektor je pobednik.
- "Najbolji" odziv se određuje prema obrascu:

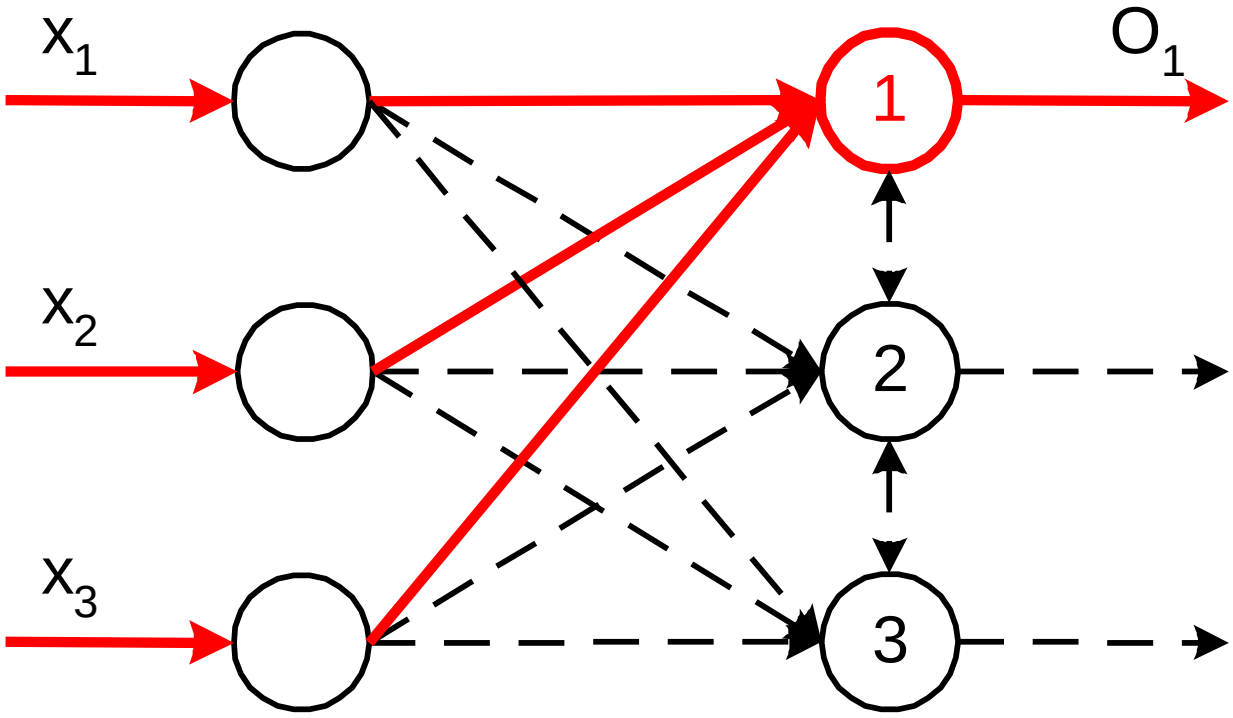
$$O_n = \max_{i=1,2,...,N} \left(O_i \right)$$

Korekcija težina se računa prema izrazu:

$$\Delta \boldsymbol{w}_n = \alpha(k)(\boldsymbol{x}-\boldsymbol{w}_n); \quad 0<\alpha(k)$$

$$\boldsymbol{w}_n(k+1) = \boldsymbol{w}_n(k) + \alpha(k)(\boldsymbol{x}-\boldsymbol{w}_n)$$

Korekcija težina je proporcionalna razlici ($\boldsymbol{x}$-$\boldsymbol{w}_n$), tako da "pobeđuje" onaj vektor $\boldsymbol{w}_n$ koji je najbliži ulaznom vektoru $\boldsymbol{x}$.



Model VNM sa kompetitivnom obukom

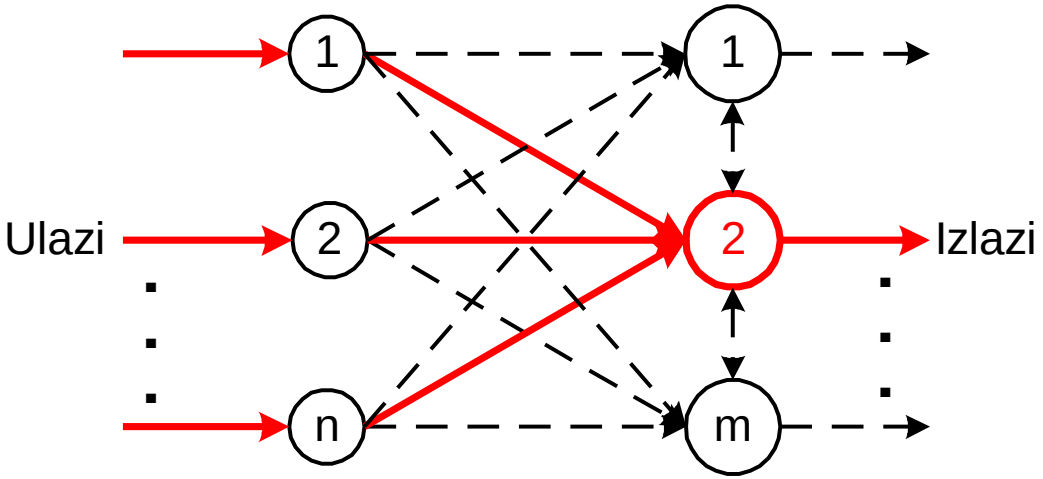
- VNM se sastoji od dva sloja: ulaznog i izlaznog, međusobno povezanih sinapsama sa adaptivnim težinama;
- Ulazni sloj – adaptivni filteri – memorijski nivo;
- Izlazni sloj – filteri maksimalne vrednosti – kodirajući nivo;
- Tokom obuke se metodom winner-take-all odabira izlazna grupa neurona, a zatim se vrši podešavanje njihovih težina.

Pravila i proces obuke

- Neuroni datog sloja su podeljeni u nepreklapajuće klastere;
- Svaki neuron u klasteru utiče na sve ostale. Neuroni u klasteru rade po “winner-takes-all” konceptu;
- Svaki element klastera prima ulazne signale na isti način. Za svaki ulaz se određuje najveći izlaz i “pobednički” neuron. Njegov izlaz je maksimalan (1), dok ostali neuroni u klasteru imaju minimalan izlaz (0);
- Obučava se samo “pobednički” neuron iz klastera;
- Ulazni uzorak x_j je binaran. Aktivni elemenat ima vrednost 1, a neaktivni 0;
- Svaki neuron ima **fiksiranu ukupnu vrednost težina** w_{ij} (**sve su pozitivne**) i one su raspodeljene po ulaznim sinapsama.
- Vrednost **sume svih težina** w_{ij} po neuronu je **jednaka 1**.
- Obuka se može vršiti sa i bez učitelja

Linear Vector Quantization (LVQ)

- Primer klasifikatora
- Obuka sa učiteljem (*supervised learning*)
- VNM ima dva sloja: ulazni i izlazni;
- Tokom obuke se vrši određivanje pripadnosti ulaznih podataka nekom od datih skupova (grupa)
- Koristi se Winner-Takes-All algoritam obuke
- Ovim algoritmom se vrši pomeranje granica grupa ka njihovim optimalnim vrednostima
- Primena je u okviru prepoznavanja štampanih karaktera, konverzije govora u štampani tekst, i sl.



Self-Organizing Map (SOM)

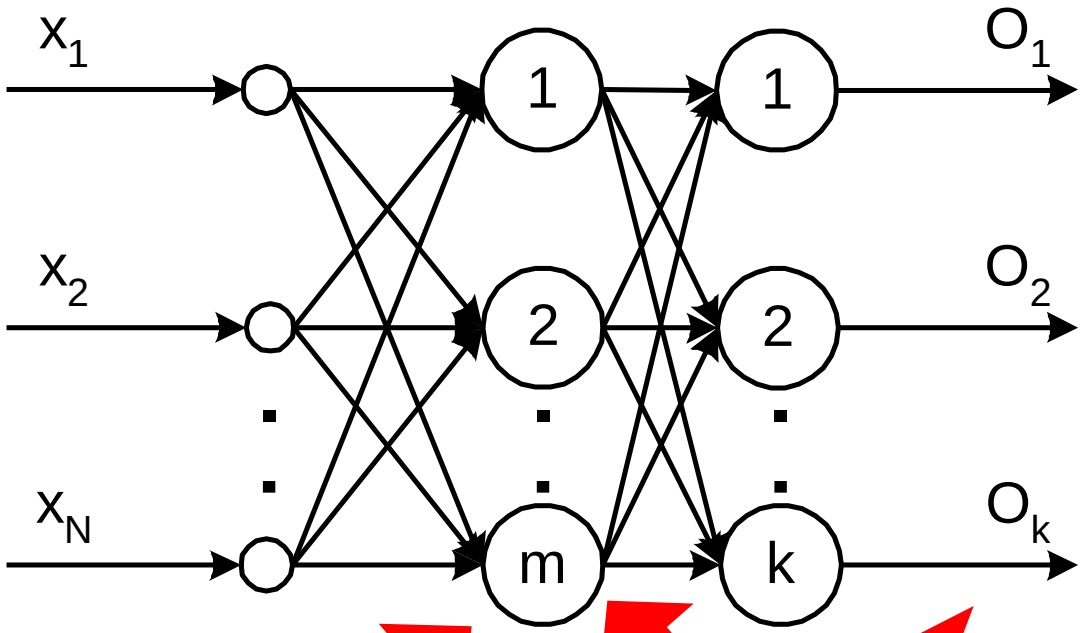
- Ili **Self-Organizing Feature Map (SOFM)**
- Samoorganizujući algoritam za klastering
- Obuka bez učitelja (*unsupervised learning*)
- Proizvodi nisko-dimenzionalnu (tipično dvodimenzionalnu) diskretnu reprezentaciju ulaznih podataka za obuku, koju zovemo **mapa**
- Za razliku od drugih VNM, SOM primenjuju kompetitivnu obuku, a ne obuku minimizacije greške kao kod BP
- Primena: Vizuelizacija nisko-dimenzionalnog prikaza visoko-dimenzionalnih podataka
- Po Finskom profesoru Teuvo Kohonen-u, koji ih je uveo 1980-tih, zovu se i **Kohonenova mapa** ili **Kohonenova mreža**

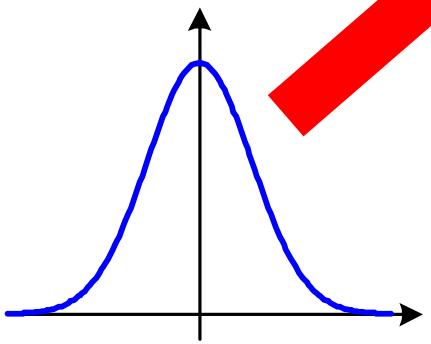
Self-Organizing Map (SOM)

- Za razliku od LVQ algoritma (klasifikacioni problem) ovde se set ulaznih uzoraka deli na unapred nepoznat skup klastera
- Vektor težina je iste dimenzije kao i vektor ulaznih podataka
- Za određivanje neurona koji je najbliži datom uzorku koristi se metod određivanja **udaljenosti između vektora** (pobeđuje neuron čije su težine najsličnije ulaznom vektoru)
- Cilj obuke je da različiti delovi mreže daju sličan odziv na određene oblike ulaznih podataka
- Motivacija je proistekla iz toga kako se vizuelne, auditorne i druge senzorne informacije obrađuju u različitim delovima cerebralnog korteksa ljudskog mozga
- Na kraju obuke svaki izlazni neuron predstavlja jedan klaster

Radial Basis Function (RBF)

- M.J.D.Powel, 1985;
- Problemi klasifikacije i regresije;
- Sastoji se od dva sloja.
- Na izlaznom sloju se formira linearna kombinacija funkcija izračunatih na skrivenom sloju.
- Bazna funkcija (kernel) na skrivenom sloju daje značajan nenulti izlaz u slučaju kada pobudni signal uzima vrednost iz određenog intervala;
- Najčešće pobudne funkcije na skrivenom sloju su sigmoida i Gausijan.





$$u_{1,j} = e^{\left[\frac{(\mathbf{x}-\mathbf{w}_{1,j})(\mathbf{x}-\mathbf{w}_{1,j})}{2\sigma_j^2}\right]}$$

$$O_j = \mathbf{w}_{2,j}^{T}\mathbf{u}_1$$

RBF algoritam obuke

- Početi obuku skrivenog sloja metodom obuke bez nadzora;
- Nastaviti sa obukom izlaznog sloja metodom obuke sa nadzorom;
- Istovremeno primeniti obučavanje sa nadzorom na skriveni i izlazni sloj radi finog podešavanja mreže.

KRAJ