

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 6950

# **Klasifikacija uporabom umjetnih neuronskih mreža**

Darijo Brčina

Zagreb, svibanj 2020.

Zagreb, 12. ožujka 2020.

Polje: **2.09 Računarstvo**

## ZAVRŠNI ZADATAK br. 6950

Pristupnik: **Darijo Brčina (0036506587)**  
Studij: Računarstvo  
Modul: Računarska znanost

Zadatak: **Klasifikacija uporabom umjetnih neuronskih mreža**

### Opis zadatka:

Klasifikacijske probleme moguće je rješavati mnoštvom različitih pristupa. Jedan od često korištenih modela su umjetne neuronske mreže. Danas postoji mnoštvo arhitektura umjetnih neuronskih mreža, pri čemu su različite arhitekture pogodne za rješavanje različitih vrsta problema.

U okviru ovog završnog rada potrebno je proučiti unaprijedne višeslojne potpuno povezane neuronske mreže primijenjene na klasifikacijske probleme te algoritme njihova učenja neuronskih mreža koji se temelje na gradijentu funkcije pogreške. Potrebno je napisati programsku implementaciju algoritma učenja, pri čemu treba podržati i učenje kroz minigrupe te vizualizaciju decizijske granice tijekom učenja u slučaju kada su ulazni uzorci iz 2D prostora. Algoritam učenja treba podržati mreže s različitim prijenosnim funkcijama. U okviru rada potrebno je razmotriti i utjecaj različitih prijenosnih funkcija na oblik decizijske granice.

Radu je potrebno priložiti algoritme, izvorne kodove i rezultate uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Zadatak uručen pristupniku: 13. ožujka 2020.

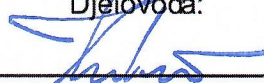
Rok za predaju rada: 12. lipnja 2020.

Mentor:



Doc. dr. sc. Marko Čupić

Djelovoda:



Izv. prof. dr. sc. Tomislav Hrkać

Predsjednik odbora za  
završni rad modula:



Doc. dr. sc. Marko Čupić



# SADRŽAJ

<b>1. Uvod</b>	<b>1</b>
<b>2. Pregled područja</b>	<b>2</b>
2.1. Umjetna inteligencija . . . . .	3
2.1.1. Simboličko učenje . . . . .	3
2.1.2. Strojno učenje . . . . .	3
<b>3. Nadzirano učenje</b>	<b>6</b>
3.1. Učenje modela . . . . .	6
3.2. Problemi nadziranog učenja . . . . .	8
3.2.1. Regresija . . . . .	9
3.2.2. Klasifikacija . . . . .	11
3.3. Algoritmi . . . . .	13
<b>4. Umjetne neuronske mreže</b>	<b>14</b>
4.1. Motivacija . . . . .	14
4.2. Biološki neuron . . . . .	14
4.3. Umjetni neuron . . . . .	15
4.3.1. Povijesni pregled . . . . .	15
4.3.2. Perceptron . . . . .	19
4.4. Općenito o umjetnim neuronskim mrežama . . . . .	22
4.5. Unaprijedna neuronska mreža . . . . .	24
4.5.1. Aktivacijske funkcije . . . . .	28
4.6. Učenje unaprijedne neuronske mreže . . . . .	28
4.6.1. Algoritam propagacije pogreške unatrag . . . . .	28
4.7. Primjena umjetnih neuronskih mreža . . . . .	28
<b>5. Rezultati</b>	<b>29</b>

<b>6. Zaključak</b>	<b>30</b>
<b>Literatura</b>	<b>31</b>

# **1. Uvod**

Uvod rada. Nakon uvoda dolaze poglavlja u kojima se obrađuje tema.

## 2. Pregled područja

Pitate li se ikada što je to inteligencija te čemu nam služi. To pitanje postavljeno je još za vrijeme začetka filozofije kao znanosti kada su se tadašnji filozofi pitali kako i na koji način je ljudsko razmišljanje, učenje i pamćenje ostvareno. Ni dan danas ne postoji jednoznačan odgovor na to pitanje jer ljudski mozak i dalje predstavlja jednu veliku nepoznanicu koja vjerojatno nikada ili ne tako skoro neće biti razriješena. No znanost je podosta napredovala i shodno tomu se razvila želja da se ljudska inteligencija pokuša pretočiti u nekakvu vrstu inteligencije strojeva.

Početak ovakvog razmišljanja datira od 50-tih godina dvadesetog stoljeća kada Alan Turing u članku *Computing Machinery and Intelligence* časopisa *Mind* postavlja pitanje: Mogu li strojevi misliti? (engl. *Can machines think?*) na koje pokušava odgovoriti kroz tzv. igru imitacije (engl. *imitation game*). Sudionici igre su tri igrača: igrač A, igrač B i igrač C gdje su igrači A i B ispitanici a igrač C ispitivač. Cilj igrača C je utvrditi spol ispitanika postavljanjem pitanja, cilj igrača B je pomoći ispitivaču C a cilj igrača A je navesti ispitivača C na pogrešnu identifikaciju. Što će se dogoditi ako stroj uzme mjesto igrača A? Ako broj pogrešaka igrača C bude gotovo jednak u oba slučaja, onda je stroj inteligentan. (Turing, 1950). Ovakav princip se često naziva turingov test (engl. *Turing test*).

Nedugo nakon turingovog eksperimenta, 1956. se održava konferencija u Dartmouthu (Dartmouth workshop, 2020) na inicijativu John McCarthy-a, tadašnjeg mladog profesora matematike na fakultetu u Dartmouthu, koji okuplja oko sebe nekolicinu znanstvenika i prijatelja kako bi pokušali koncepte ljudske inteligencije preslikati u inteligenciju strojeva. Cilj je pokazati kako strojevi koriste jezik, kako zaključuju i stvaraju apstraktne koncepte i kako vremenom postaju sve prilagodljiviji na predložene probleme baš kao i ljudi. Inicijalna ideja je bila da se neki od navedenih problema može dokazati uz manju skupinu dobrih znanstvenika i kroz period od jednog ljeta (McCarthy et al. 1955). To naravno nije bilo moguće. Time se formalno uvodi pojam *umjetna inteligencija*.

## 2.1. Umjetna inteligencija

Kao što je anticipirano ranije, umjetna inteligencija (engl. *artificial intelligence*) je laički rečeno inteligencija strojeva a znanost koja se jednim dijelom bavi proučavanjem umjetne inteligencije jest računarska znanost (engl. *computer science*).

Primjena umjetne inteligencije danas je izrazito rasprostranjena kroz gotovo svaku industriju. Pronalazimo ju u medicini, automobilske industriji, robotici, pa i u sportskoj i industriji igara. Jedan od poznatijih događaja koji prikazuje primjenu umjetne inteligencije dogodio se 2016. kada je računalo naziva *AlphaGo* u igri *Go* uspijelo pobijediti svjetskog prvaka Lee Sedolu rezultatom 4:1 te time ostvario velik uspjeh u svijetu umjetne inteligencije kao i pažnju javnosti (Moyer, 2016).

Umjetnu inteligenciju je dakako potrebno trenirati i učiti pa je tako učenje podijeljeno na dvije veće cjeline:

1. simboličko učenje te
2. strojno učenje.

### 2.1.1. Simboličko učenje

Simbolička umjetna inteligencija (engl. *symbolic artificial intelligence*) je izraz koji definira skup istraživačkih metoda koje se temelje na ljudima lako čitljivim simbolima (engl. *human-readable simbol*) koji modeliraju probleme i logiku. Jedan od najboljih primjera jesu *ekspertni sustavi* koji se temelje na skupu pravila. Pravila su modelirana na sličan način kao i Ako-Onda rečenica (engl. *If-Then statement*) koja je u ljudskoj komunikaciji svakodnevno u upotrebi. Također, razne vrste logika poput propozicijska (engl. *Propositional logic*), često referirana kao Boolova algebra, logika prvog reda (engl. *First order logic*), poznatija kao predikatna logika (engl. *Predicate logic*), neizrazita logika (engl. *Fuzzy logic*) pripadaju upravo simboličkoj umjetnoj inteligenciji. Ovakav način učenja bio je popularan početkom 1950. sve do kraja 1980 (Symbolic artificial intelligence, 2020).

### 2.1.2. Strojno učenje

Strojno učenje (engl. *machine learning, ML*) predstavlja niz metoda i algoritama koji sustavima pružaju stjecanje novog znanja kroz modeliranje obrazaca koje onda kasnije mogu iskoristiti za predviđanje novih podataka ili sličnih (Čupić, 2020). Glavna ideja



je da sustavi uče iz iskustva, empirijski, bez da se programska implementacija mijenja što znatno olakšava manipulaciju istih.

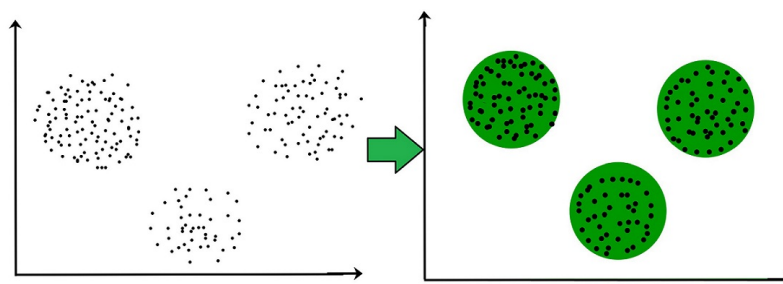
Danas postoji nezgrapno puno podataka koje je moguće i koje je potrebno iskoristiti za učenje pa je cilj konstruirati sustave koji mogu iskoristiti baš te podatke za neka korisna ponašanja poput predviđanja i raspoznavanja raznih uzoraka (Čupić, 2020). Podatci se svrstavaju u dvije grupe: *numerički* i *kategorički* podatci. Numerički podatci su svi oni podatci nad kojima je moguće izvršiti aritmetičke operacije. Npr. unos dnevnih kalorija, broj slobodnih bacanja na košarkaškoj utakmici, isplata plaća i sl. Kategorički podatci se dodatno dijele u dvije podgrupe: nominalni i ordinalni podatci. Nominalni podatci su imenovani podatci koji nisu numerički, tj. podatci nad kojima nisu definirane aritmetičke operacije. Npr. podatci o spolu jedinke, osjećaju raspoloženja poput "tužan" ili "veseo" i sl. Ordinalni podatci također nemaju definiranu aritmetiku, no imaju definiran prirodan poredak i mogu se uspoređivati. Npr. mišljenje jedne osobe može biti "jako zadovoljan" dok druge "zadovoljan" što možemo usporediti i konstruirati prirodan poredak.

Strojno učenje dijelimo na četiri područja:

1. nadzirano učenje,
2. nenadzirano učenje,
3. polu-nadzirano učenje te
4. podržano učenje.

*Nadzirano učenje* je tema ovog rada pa će biti obrađeno detaljnije u narednom poglavlju.

*Nenadzirano učenje* (engl. *unsupervised learning*) je vrsta učenja u kojem skup podataka predstavljaju samo ulazni podatci bez znanja o tome kako bi isti trebali biti tabelirani. Zbog ovakvog pristupa, učenje je i dobilo ime nenadzirano učenje, tj. učenje bez prisustva učitelja (engl. *supervisor*). Najpoznatiji postupak nenadzirano učenja je postupak grupiranja (engl. *clustering*). Cilj grupiranja je na temelju danih podataka pokušati pronaći sve podatke koji imaju slična svojstva te ih zatim grupirati u odvojene razrede. Npr. neka skup ulaznih podataka sustava bude mješavina slika ljudi i slika automobila. Kao konačni rezultat, moraju se stvoriti dva razreda: razred ljudi te razred automobila. Na slici 2.1 prikazan je primjer grupiranja.

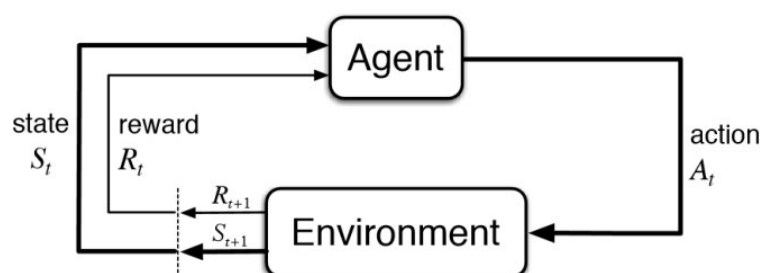


**Slika 2.1:** Primjer grupiranja<sup>1</sup>

Uz grupiranje poznati postupci su i postupci smanjenja dimenzionalnosti (engl. *dimensionality reduction*), postupci otkrivanja stršećih ili novih vrijednosti (engl. *outlier detection*, *novelty detection*) i drugi.

*Polu-nadzirano učenje* (engl. *semi-supervised learning*), kao što samo ime nalaže, je učenje u kojem se isprepliću koncepti nadzirano učenje sa konceptima nenadziranim učenjem. Ono pokazuje dosta veće uspjehe nego navedeni, izvedba je izrazito zahtjevnija (Semi-supervised learning, 2020).

*Podržano učenje* (engl. *reinforcement learning*, *RL*) je vrsta učenja koja se potpuno razlikuje od svih do sada navedenih jer ono ne očekuje nikakve tabelirane ulazne ili izlazne podatke već je glavna ideja optimizacija ponašanja računalnih agenata. Razmatra se interakcija agenta sa okolinom (engl. *environment*) kroz niz akcija za koje isti može biti nagrađen ili kažnjen ovisno o ishodu akcije. Željeni cilj agenta je maksimizirati mogući dobitak nagrade (Reinforcement learning, 2020). Na slici 2.2 prikazan je model podržanog učenja.



**Slika 2.2:** Model podržanog učenja<sup>2</sup>

<sup>1</sup>Preuzeto sa <https://www.geeksforgeeks.org/clustering-in-machine-learning/>.

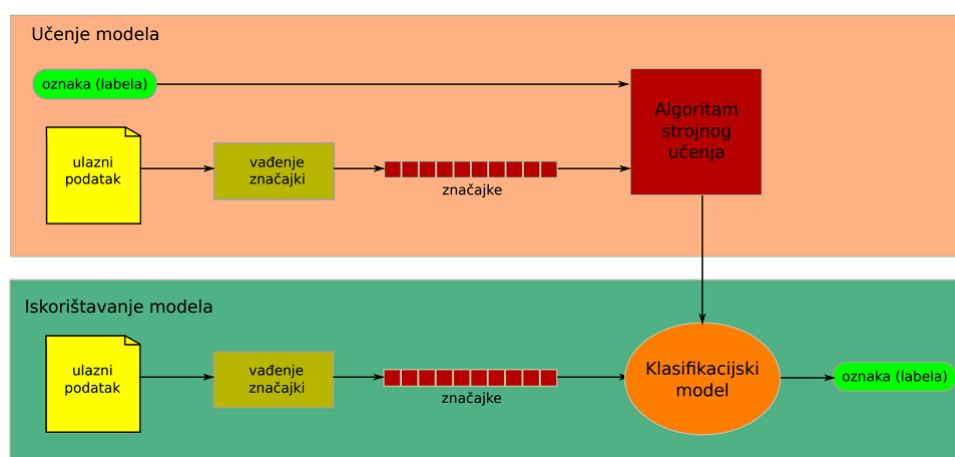
<sup>2</sup>Preuzeto sa <https://www.kdnuggets.com/2018/03/5-things-reinforcement-learning.html/>.

### 3. Nadzirano učenje

Nadzirano učenje (engl. *supervised learning*) je vrsta učenja čiji je cilj sve podatke koji dođu na ulaz sustava preslikati u izlaz koji točno odgovara predanim ulazima. Dakle, prije samog učenja poznati su ulazi, koji se često modeliraju vektorima, te izlazi, koji su pridruženi tim ulazima i predstavljaju jedan podatak, stoga se tijekom učenja u svakom trenutku zna željeni izlaz te se za krive rezultate daje određena povratna informacija (engl. *feedback*) o tome koliko sustav griješi. Upravo zbog navedenog koncepta je učenje i dobilo ime nadzirano učenje jer se proces učenja izvršava uz prisustvo učitelja (engl. *supervisor*). Razlikujemo dvije faze strojnog modela prikazane na slici 3.1:

1. faza učenja modela te
2. faza iskorištavanja modela.

#### 3.1. Učenje modela



Slika 3.1: Model strojnog učenja<sup>1</sup>

<sup>1</sup>Preuzeto iz literature (Čupić, 2020).

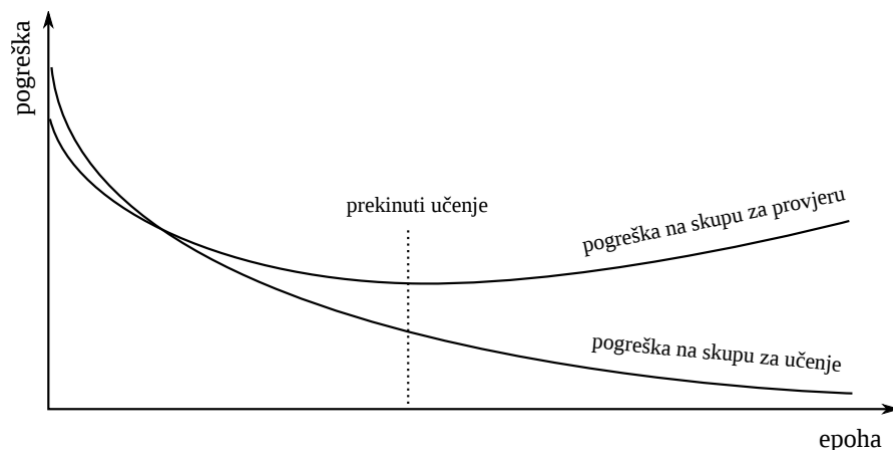
Tijekom faze učenja, modelu se predaje skup ulaznih podataka za učenje (engl. *training set*). Svaki podatak skupa možemo notirati kao  $(\mathbf{x}, y)_i$  što predstavlja  $i$ -ti primjer iz skupa za učenje gdje  $\mathbf{x}$  predstavlja vektor ulaznih podataka, a  $y$  vrijednost pridružena ulaznom vektoru, odnosno oznaku (engl. *label*). Ulazni podatak često može biti kompleksne naravi što znači da ga je potrebno "razbiti" u manje cjeline. Takav postupak se naziva vađenje značajki (engl. *feature extraction*). Rezultat navedenog je kolekcija značajki koja se zatim predaje određenom algoritmu strojnog učenja. Npr. ulazni podaci mogu biti razne gume poput gume automobila, gume bicikla, gume traktora i sl. a značajka bi onda mogla biti volumen gume, oblik gume, boja gume i sl. Algoritam strojnog učenja će na temelju predanih značajki pokušati optimizirati parametre odabranog modela kako bi se isti mogao koristiti u sljedećoj fazi.

Faza iskorištavanja modela predstavlja fazu u kojoj se treba utvrditi radi li naučeni model sa podacima nad kojima nije učio, tj. je li razvio sposobnost generalizacije (engl. *generalization*). Ako model dodijeli točnu oznaku za većinu ulaznih podataka, onda možemo reći da je razvio svojstvo generalizacije. No što ako u velikoj mjeri ne uspije točno povezati izlaze sa ulazima? Ako dođe to takve pojave, kažemo da je model pretreniran (engl. *overfitting*). Do navedene pojave dolazi kada model nauči previše nad skupom podataka za učenje, tj. podatke za učenje nauči savršeno raspoznavati dok za nove podatke ne uspijeva dati željene rezultate. Postoji nekoliko načina kako se može utjecati da model razvije svojstvo generalizacije a spomenut ćemo samo jedan od njih, *unakrsna provjera*.

Unakrsna provjera (engl. *cross-validation*) je postupak koji se koristi tijekom faze učenja modela. Ideja je da se dio podataka iz skupa za učenje razdijeli u novi skup, skup za provjeru (engl. *validation set*). Skup za provjeru često sadrži od 20% do 30% svih podataka iz skupa za učenje (Čupić, 2020). Tijekom faze učenja modela, model će i dalje učiti samo nad skupom za učenje dok ćemo mu povremeno dati i podatke iz skupa za provjeru čisto da se vidi koliko griješi nad njima. Važno je napomenuti da model nad podacima iz skupa za provjeru neće učiti, tj. neće optimizirati parametre već će bit prisiljen da nauči generalizirati. Postavlja se pitanje do kada će faza učenja modela onda trajati i odgovor je vrlo jasan. Trajat će do trenutka kada pogreška nad skupom za provjeru počinje rasti. Dijagram na slici 3.2 prikazuje kako model uči kroz epohe<sup>2</sup>.

---

<sup>2</sup>Epohe je ništa drugo nego jedna iteracija učenja, ali u strojnom učenju se često koristi izraz epoha pa ćemo se držati konvencije.



**Slika 3.2:** Kretanje pogrešaka pri učenju strojnog modela<sup>3</sup>

Također postoji mogućnost da neki od parametara modela utječu na njegovu složenost. Shodno tomu se uvodi još jedan skup, skup za testiranje (engl. *testing set*). Ideja je sljedeća: skup podataka razdijelimo u sva tri do sada navedena skupa i normalno provedemo fazu učenja modela uz malu promjenu. Umjesto da se zadovoljimo samo jednim naučenim modelom, pokušavamo pronaći onaj model koji će najbolje minimizirati pogrešku nad skupom za testiranje (Čupić, 2020).

## 3.2. Problemi nadziranog učenja

Zadatak nadziranog učenja je sljedeći: pretpostavimo da imamo skup za učenje koji se sastoji od  $N$  različitih parova ulaza i izlaza;

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), (\mathbf{x}_3, y_3), \dots, (\mathbf{x}_N, y_N)$$

gdje  $\mathbf{x}_i$  predstavlja  $i$ -ti vektor ulaznih podataka dok  $y_i$   $i$ -ti željeni izlaz koji je generiran nekom nepoznatom funkcijom  $y = f(x)$ .

Potrebno je pronaći takvu funkciju  $h$  koja će najbolje aproksimirati funkciju  $f$ .

Funkcija  $h$  naziva se hipoteza (engl. *hypothesis*) i pripada nekom skupu svih hipoteza  $\mathbf{H}$  koje donekle aproksimiraju zadanu funkciju  $f$ , neke bolje, neke lošije. U ovom trenutku uključujemo skup za testiranje u proces učenja jer je potrebno pronaći onu hipotezu za koju će aproksimacija biti najbolja, tj. ukupna pogreška generirana tijekom učenja modela bit će minimalna. Time će se osigurati da model poprimi svojstvo generalizacije. Često se pokazuje da je funkcija  $f$  stohastička (engl. *stochastic*), što

<sup>3</sup>Preuzeto iz literature (Čupić, 2016).

znači da ne ovisi uvijek samo o varijabli  $x$  pa se mora koristiti i uvjetovana vjerojatnost  $P(Y|x)$  tijekom faze učenja modela (Russell i Norvig, 2009)<sup>4</sup>.

Kao što je rečeno ranije,  $y_i$  predstavlja  $i$ -ti željeni izlaz za  $i$ -ti ulaz te može poprimiti vrijednost broja ili vrijednost iz nekog konačnog skupa podataka. Ako je slučaj da je izlaz broj, radi se o problemu koji se naziva *regresija*, a ako je slučaj da je izlaz vrijednost iz nekog konačnog skupa podataka, onda se radi o problemu koji se naziva *klasifikacija*.

### 3.2.1. Regresija

Regresija (engl. *regression*) je tehnika, tj. algoritam nadziranog učenja koji se koristi kada je potrebno predvidjeti određeni iznos neke funkcije u odnosu na određene parametre. Dakle, dane podatke potrebno je aproksimirati nekom proizvoljnom funkcijom  $h$  koja je često polinomijalna. Postoje različite vrste regresije pa spomenimo neke od njih.

Prije svega, postoji regresija s jednom varijablom (engl. *regression with one variable*) i regresija sa više varijabli (engl. *multivariate regression*). Razlika je očita, no pokažimo to na kratkom primjeru. Recimo da želimo modelirati cijene obiteljskih kuća u okolini Zagreba. Za to su nam potrebni neki parametri na temelju kojih ćemo moći vršiti predikcije kao npr. broj kvadratnih metara ( $m^2$ ), broj spavaćih soba, blizina vrtića i škola, povezanost sa javnim prijevozom i sl. U slučaju regresije s jednom varijablom, za modeliranje bi uzeli samo jedan od navedenih parametara, dok bi u slučaju regresije sa više varijabli uzeli sve ili dio navedenih. Dakle, očigledno je da je regresija s jednom varijablom konkretan slučaj regresije sa više varijabli.

*Linearna regresija* (engl. *linear regression*) je tip regresije koji koristi linearnu krivulju (pravac), tj. polinom prvog stupnja kao aproksimacijsku krivulju.

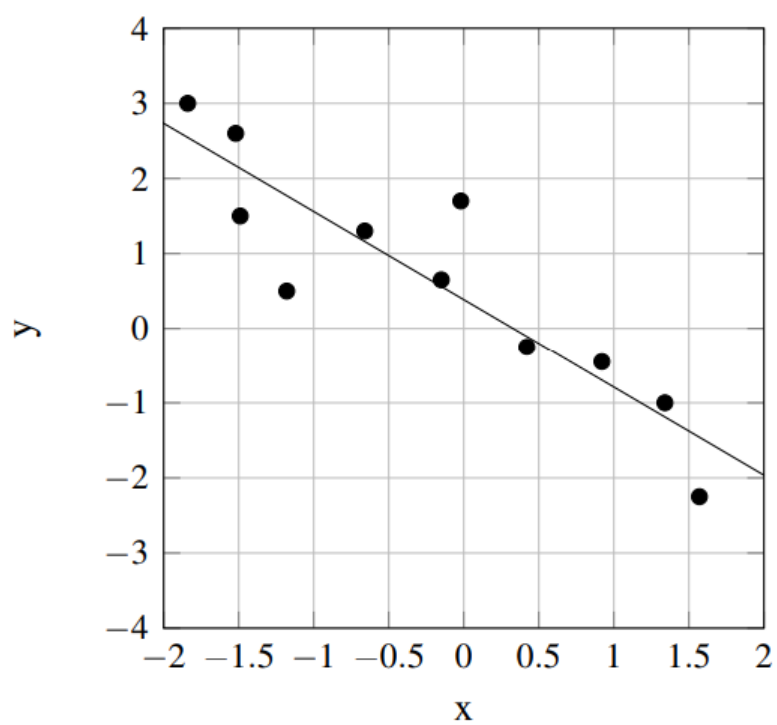
*Polinomijalna regresija* (engl. *polynom regression*) je tip regresije koji koristi polinom reda  $r$  kao aproksimacijsku krivulju.

Na slikama 3.3 i 3.4 su prikazani navedeni tipovi regresija.

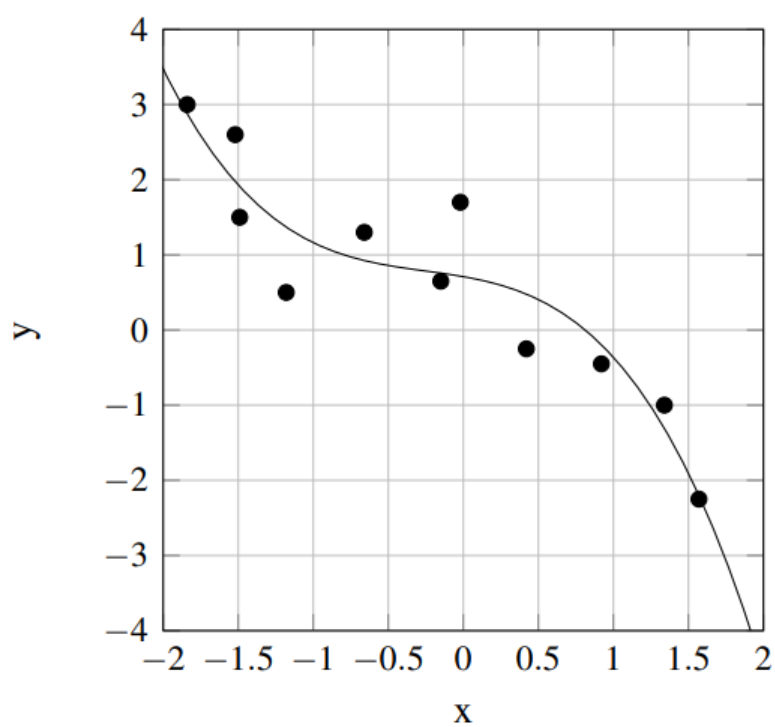
Dakle, regresija je metoda koja se koristi za predviđanje numeričkih ishoda.

---

<sup>4</sup>Iz poglavlja 18.2 Supervised learning.



**Slika 3.3:** Primjer linearne regresije<sup>5</sup>



**Slika 3.4:** Primjer polinomijalne regresije sa kubnim polinomom<sup>6</sup>

<sup>5</sup>Preuzeto iz literature (Čupić, 2020).

<sup>6</sup>Preuzeto iz literature (Čupić, 2020).

### 3.2.2. Klasifikacija

Klasifikacija (engl. *classification*) je tehnika, tj. algoritam nadziranog učenja koji se koristi kada je određen ulazni podatak potrebno smjestiti odnosno klasificirati kao pripadnika određenog razreda. Izlazi tako organiziranog modela često poprimaju diskretne vrijednosti za razliku od regresije i takav model onda nazivamo klasifikator (engl. *classifier*). Razlikujemo nekoliko različitih tipova klasifikatora:

1. binarni klasifikator,
2. višerazredni klasifikator te
3. klasifikator više oznaka.

Binarni klasifikator (engl. *binary classifier*) je tip klasifikatora koji zna odrediti pripada li ulazni podatak jednom od ukupno dva moguća razreda. Npr. sustav kao ulaze prima slike na kojima se nalazi ili mačka ili pas. Jednom naučeni klasifikator bi na temelju predane slike morao moći odrediti što se nalazi na slici, mačka ili pas. Naravno uz uvjet da se preda slika mačke ili psa.

Višerazredni klasifikator (engl. *multi-class classifier*) je tip klasifikatora koji zna odrediti pripada li ulazni podatak jednom od barem tri moguća razreda. Dakle, mora postojati minimalno tri različita razreda kako bi se klasifikator nazivao višerazrednim. U ovom radu ćemo se više fokusirati upravo na navedenom klasifikator koji će naučiti raspoznati tri moguća razreda, no o tome nešto kasnije.

Klasifikator više oznaka (engl. *multi-label classifier*) je tip klasifikatora koji predstavlja generalizaciju višerazrednog klasifikatora. Pogledajmo sljedeći primjer klasifikacije filmova za bolje razumijevanje. Pretpostavimo da film može imati sljedeće oznake: (12+), (16+) i (18+) gdje svaka oznaka predstavlja minimalan broj godina kojih gledaoc filma mora imati. Određivanje oznake za pojedini film je rezultat korištenja višerazrednog klasifikatora jer je potrebno odrediti pripada li film samo jednom od navedene tri oznake. No što ako film poželimo klasificirati pomoću žanrova? Tada nam višerazredni klasifikator neće puno pomoći jer jedan film može imati više oznaka kao npr. "drama" i "komedija" i sl. Tada ćemo koristiti klasifikator više oznaka.

Klasifikaciju, kao i regresiju, možemo podijeliti na *linearnu klasifikaciju* i *nelinearnu klasifikaciju*, odnosno polinomijalnu.

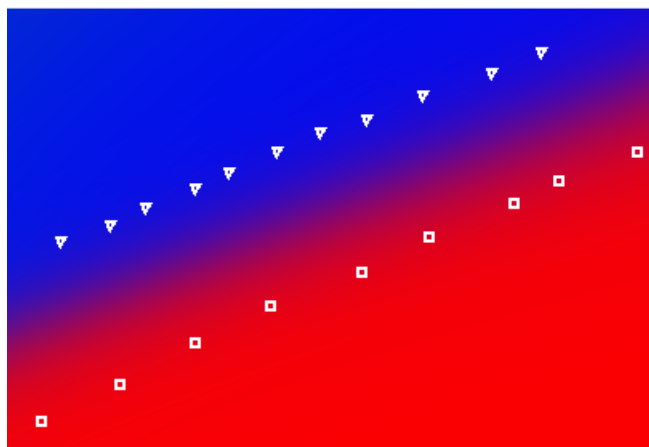
Linearna klasifikacija (engl. *linear classification*) je tip klasifikacije u kojoj se granica između razreda aproksimira pomoću linearna krivulje, tj. pravca.



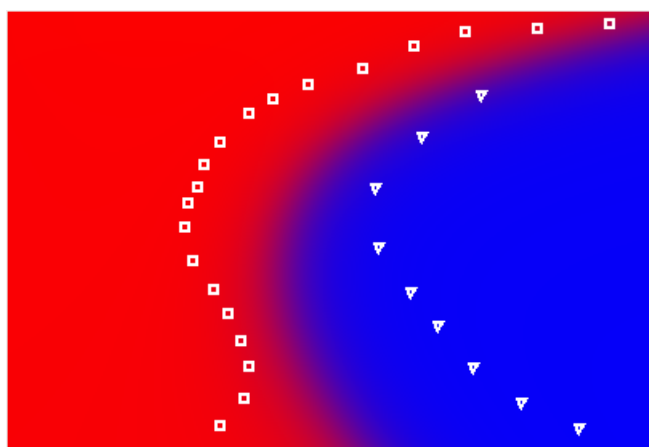
Nelinearna klasifikacija (engl. *non-linear classification*) je tip klasifikacije u kojoj se granica između razreda aproksimira pomoću nelinearne krivulje koja je često polinomijalna.

Ono što je zajedničko u oba slučaja je postojanje granice između razreda koja se u literaturi naziva *decizijska granica* (Čupić, 2016). O oblicima decizijskih granica će biti riječ kasnije.

Na slikama 3.5 i 3.6 su prikazani primjeri klasifikacija.



**Slika 3.5:** Primjer linearne klasifikacije<sup>7</sup>



**Slika 3.6:** Primjer nelinearne klasifikacije<sup>8</sup>

---

<sup>7</sup>Rezultat programske implementacije.

<sup>8</sup>Rezultat programske implementacije.

### 3.3. Algoritmi

U području nadziranog učenja postoji veliki broj algoritama koji uvelike pridonose boljem učenju. Neki od poznatijih su:

- Stroj za podršku vektorima (engl. *support-vector machine, SVM*),
- Stabla odluke (engl. *decision trees*),
- Naivni Bayesov klasifikator (engl. *naive Bayes classifier*),
- Slučajne šume (engl. *random forests*),
- Algoritam k-najbližih susjeda (engl. *k-nearest neighbors algorithm*),
- Umjetna neuronska mreža (engl. *artificial neural network*) i mnogi drugi.

Vidimo da postoji jako velika paleta algoritama od kojih su neki složeniji od drugih, neki daju bolje rezultate od drugih, no niti jedan od algoritama nije u stanju uvijek dati najbolje rezultate za proizvoljan problem nadziranog učenja (Supervised learning, 2020). Zbog navedenog vrijedi tzv. *No free lunch* teorem koji nalaže da je za probleme koji zahtijevaju mnogo računalnih resursa poput pretraživanja prostora stanja i optimizacije parametara složenost<sup>9</sup> u prosjeku jednaka za sve metode (No free lunch theorem, 2020).

U sljedećem poglavlju ćemo se detaljnije baviti algoritmom *umjetna neuronska mreža*. Pogledat ćemo kako je nastao koncept umjetnog neurona, strukturu umjetnog neurona, arhitekturu unaprijedne neuronske mreže te algoritam kojim unaprijedna neuronska mreža uči.

---

<sup>9</sup>Često se ovako definirani problemi nazivaju *NP-teškim* problemima (engl. *Non-deterministic polynomial-time problems*).

## 4. Umjetne neuronske mreže

U drugom poglavlju (*Pregled područja*) dotaknuli smo se razlika između simboličke umjetne inteligencije i strojnog učenja i rekli smo kako simbolička umjetna inteligencija koristi simbolički pristup, tj. sve iskaze pokušava predočiti u mnoštvo pravila pomoću ljudima čitljivim simbolima, dok se strojno učenje bavi raznim algoritmima pomoću kojih se računalni sustav uči. U ovom poglavlju bavit ćemo se algoritmom strojnog učenja koji koristi tzv. konektivistički pristup.

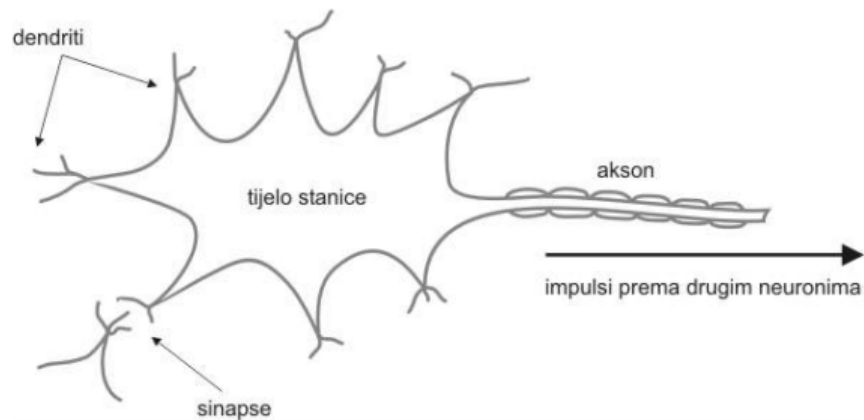
### 4.1. Motivacija

Ideja za konektivizmom (engl. *connectivism*) potaknuta je na temelju strukture ljudskog mozga te zbog izrazite brzine kojom mozak obrađuje podražaje. Danas je poznato da u ljudskom mozgu postoji oko  $10^{11}$  neurona te  $10^{15}$  međusobnih veza što znači da je svaki neuron u prosjeku povezan sa  $10^4$  različitih veza (Čupić et al., 2013). To upravo dokazuje da mozak predstavlja jedan kompleksan i paralelan sustav zbog čega je konektivizam dobio ime te čemu algoritam umjetne neuronske mreže teži. Također, koncept ljudskog mozga nije uzet samo radi svoje izrazite brzine prilikom obrade podražaja, već i zbog činjenice da ga se može učiti na temelju velikog broja podataka koji su često ispunjeni raznim šumovima, što je realna situacija, te ga time potaknuti da razvije svojstvo generalizacije. Područje koje se bavi proučavanjem umjetnih neuronskih mreža naziva se neuro-računarstvo (engl. *neuro-computing*) koje je jedno od grana mekog računarstva (engl. *soft-computing*).

### 4.2. Biološki neuron

Neuroni (engl. *neurons*) su glavne stanice mozga i živčanog sustava. Zadaća im je da primaju podražaje iz okoline, da prenose određene akcije od mozga prema našim mišićima i da vode računa o pretvaranju električnog potencijala između pojedinih neurona u svim koracima procesa vođenja. Neuron je izgrađen od posebne strukture koja

se sastoji od: *tijela stanice* (ili češće *soma*), *dendrita*, *aksona*, i *sinapse*. Struktura biološkog neurona prikazana je na slici 4.1.



**Slika 4.1:** Biološki neuron<sup>1</sup>

Dendriti (engl. *dendrites*) su ulazni dijelovi neurona, tj. strukture koje prihvaćaju impulse od drugih neurona preko veza koje se nazivaju sinapse. Suma svih ulaznih impulsa, tj. potencijala određuje hoće li dotični neuron biti aktiviran ili ne.

Aksoni (engl. *axons*) su izlazni dijelovi neurona, tj. strukture u kojima se generira određeni električni potencijal koji se dalje prosljeđuje sljedećem neuronu preko sinoptičkih veza.

Sinapse (engl. *synapses*) su, kao što je rečeno ranije, veze između dendrita jednog neurona i aksona drugog neurona (Woodruff, 2019).

## 4.3. Umjetni neuron

### 4.3.1. Povijesni pregled

#### MCP neuron

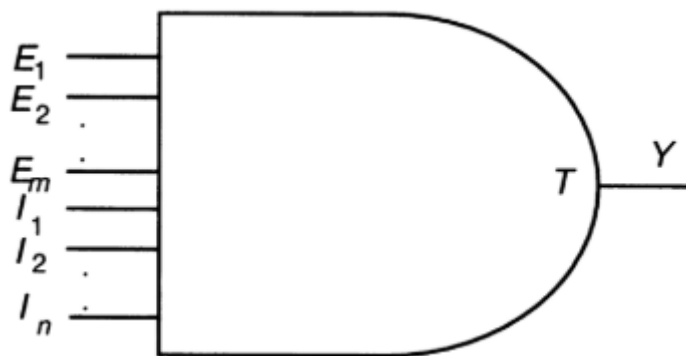
**1943.** dva znanstvenika, Warren S. McCulloch i Walter H. Pitts, u svojem radu *Logical Calculus of Ideas Immanent in Nervous Activity* definiraju strukturu prvog umjetnog neurona koji se u literaturi često naziva kao *MCP* neuron. Na slici 4.2 prikazana je struktura MCP neurona. Dendriti su predstavljeni ulazima označenih slovima

<sup>1</sup>Preuzeto iz literature (Čupić, 2016).

**E** od riječi ekscitacijski<sup>2</sup> (engl. *excitatory*) te **I** od riječi inhibicijski<sup>3</sup> (engl. *inhibitory*), tijelo stanice je predstavljeno slovom **T** od riječi prag (engl. *threshold*) dok je akson predstavljen izlazom koji je označen slovom **Y** bez posebnog značenja.

Ekscitacijski ulazi uzrokuju da neuron postane aktivan, tj. da se neuron "pali" (engl. *fire*), dok inhibicijski ulazi uzrokuju da neuron postane neaktivan, tj. da se neuron "ne pali". Točnije, ako postoji barem jedan inhibicijski ulaz koji je aktivan, tada će neuron sigurno biti neaktivan neovisno o svim ostalim ulazima. Također, ako niti jedan od inhibicijskih ulaza nije aktivan, onda će neuron postati aktivan samo u slučaju kada je suma svih ekscitacijskih ulaza veća ili jednaka pragu,  $T$  (McCulloch i Pitts, 1943). Matematički rečeno (Picton, 2000):

$$Y = \begin{cases} 1, & \sum_{i=1}^n I_i = 0 \text{ i } \sum_{j=1}^m E_j \geq T \\ 0, & \text{inače.} \end{cases} \quad (4.1)$$



**Slika 4.2:** MCP neuron<sup>4</sup>

MCP neuron mogao je modelirati osnovne Booleove logičke operacije kao što su operacija *i* (engl. *and*), operacija *ili* (engl. *or*) i operacija negacije (engl. *not*) pa se može reći da neuron predstavlja logička vrata (engl. *logic gate*) što je često korištena struktura prilikom konstrukcija uređaja poput računala koji mogu rješavati kompleksne logičke izraze. Shodno tomu se razvilo vjerovanje da se ljudski mozak sastoji upravo

<sup>2</sup>ekscitacija - uzbuđenje ili aktivirano stanje zbog određenog podražaja (ekscitacija. Hrvatska enciklopedija, mrežno izdanje. Leksikografski zavod Miroslav Krleža, 2020. Pristupljeno 13. 5. 2020. <<http://www.enciklopedija.hr/Natuknica.aspx?ID=17398>>).

<sup>3</sup>inhibicija - kočenje prijenosa živčanih impulsa (inhibicija. Hrvatska enciklopedija, mrežno izdanje. Leksikografski zavod Miroslav Krleža, 2020. Pristupljeno 13. 5. 2020. <<http://www.enciklopedija.hr/Natuknica.aspx?ID=27450>>).

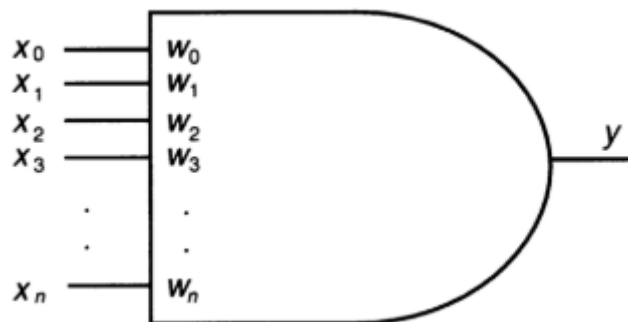
<sup>4</sup>Preuzeto iz literature (Picton, 2000) iz poglavlja 1.4.2 Biologically inspired neural networks.

od mnoštva logičkih vrata, no nisu uzimali u obzir činjenicu da takva konstrukcija ne može biti učena te da nema svojstvo generalizacije (Picton, 2000).

Postoji još jedna bitna stavka oko strukture MCP neurona koja je prešutno korištena, a to su vrijednosti koje poprimaju ulazi i izlaz. Vrijednosti koje mogu poprimiti su iz skupa  $\{0, 1\}$  i niti jedne druge, dok vrijednost praga može biti proizvoljan realan broj. To se implicitno moglo zaključiti iz spomenutih primjena na Boolove logičke operacije te iz formule (4.1).

## ADALINE

**1960.** dva znanstvenika, Bernard Widrow i Marcian E. Hoff, u svojem radu *Adaptive Switching Circuits* na temelju MCP neurona definiraju jednu od prvih neuronskih mreža nazvanu *ADALINE* (engl. *ADaptive LINEar Elements*). Na slici 4.3 prikazana je struktura jednog neurona iz neuronske mreže ADALINE. Dendriti su predstavljeni ulazima označenim slovom  $x$ , tijelo stanice predstavljeno je težinama označenim slovom  $w$ , a akson je predstavljen izlazom označenim slovom  $y$ . Dakle, uočljivo je da postoje neke novine za razliku od MCP neurona. Ulogu ekscitacijskih odnosno inhibicijskih ulaza sada imaju težine koje poprimaju vrijednosti iz realnog skupa brojeva, ulazi i izlaz poprimaju vrijednosti iz skupa  $\{-1, 1\}$  i niti jedne druge, osim za ulaz  $x_0$  čija je vrijednost uvijek 1, dok prag više nije eksplicitno zadan, već se u definiciju implicitno uključuje simbolom  $w_0$  koji predstavlja konstantan pomak (engl. *bias*) (Picton, 2000).



Slika 4.3: ADALINE neuron<sup>5</sup>

Definiranje težina, koje odgovaraju svaka svom ulazu, revolucionarno je otkriće jer se time pokazalo da se tako definirani neuroni mogu učiti što uvelike olakšava izvedbu računalnih sustava jer više nije potrebna kvantiteta da bi sustav funkcionirao,

<sup>5</sup>Preuzeto iz literature (Picton, 2000) iz poglavlja 1.4.2 Biologically inspired neural networks.

već kvalitetna izvedba. Umjesto praćenja koliko je ekscitacijskih odnosno inhibicijskih ulaza aktivno, definirana je težinska suma *net* koja je opisana sljedećom formulom:

$$net = \sum_{i=0}^n w_i * x_i \quad (4.2)$$

Vrijednost *net*-a je onda transformirana u izlaz *y* pomoću nelinearne funkcije koja pozitivne vrijednosti preslikava u 1, dok negativne vrijednosti ili vrijednosti koje su jednake 0 preslikava u -1. Matematička definicija dana je sljedećom formulom (Picton, 2000):

$$y = \begin{cases} 1, & \text{net} > 0 \\ -1, & \text{net} \leq 0 \end{cases} \quad (4.3)$$

Koncept težina definiran je 1949. kada je Donald Hebb u svome dijelu *The Organization of Behavior: A Neuropsychological Theory* pokušao definirati da kada je akson neurona A dovoljno blizu da aktivira neuron B i to ponavlja veći broj puta dolazi do metaboličkih promjena tako da se povećava efikasnost neurona A u aktiviranju neurona B (Hebb, 1949). Često se navedeno pravilo naziva hebbov princip učenja i može se definirati pomoću ADALINE neurona. Pravilo je dano sljedećom formulom (Picton, 2000):

$$w_i = \sum_{p=1}^p x_{ip} * y_p \quad (4.4)$$

Svaka težina  $w_i$  računa se kao težinska suma svih ulaza  $x_i$  sa željenim izlazom  $y$  i to za svaki uzorak po skupu svih uzoraka koji su označeni slovom  $p$ . Ako uzmemo u obzir da ulazi i izlazi poprimaju vrijednosti iz skupa  $\{-1, 1\}$ , onda je očito da iznos težine  $w_i$  raste ako su ulaz za tu težinu  $x_i$  i izlaz uzorka  $y_p$  aktivni, odnosno smanjuje ako su neaktivni. Iako se čini da je pravilo uspješno, pokazuje se da se njime ne uspijeva dobro naučiti mrežu jer se u izračun ne uzimaju stvarni izlazi neurona, već samo željeni izlazi.

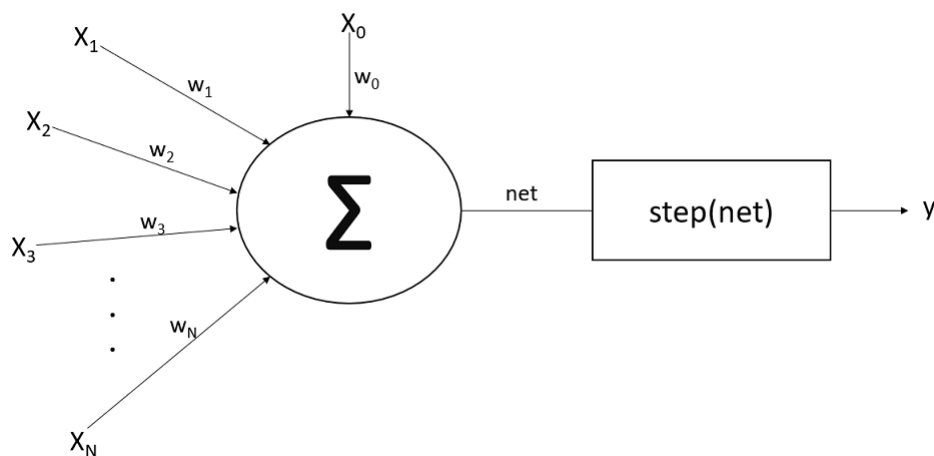
Zbog navedenog nedostatka hebbovog principa, Widrow i Hoff su došli do ideje da bi se težine morale namještati pomoću pogreške (engl. *error*) koja se događa između željenog izlaza i stvarnog izlaza te algoritma gradijentni spust. Navedeni princip nazvan je: *delta pravilo* (engl. *the delta rule*) ili widrow-hoffovo pravilo, a danas je poznato i pod nazivom: pravilo najmanjeg srednjeg kvadrata (engl. *least mean square rule*). Time su uspjeli postići da mreža ADALINE pokazuje jako dobre rezultate tijekom procesa učenja, a to se implicitno potkrepljuje riječju (engl. *ADaptive*) iz imena

što znači prilagodljiv (Picton, 2000). Detaljnije definicije i izvode formula pokazat ćemo u poglavlju gdje se govori o učenju unaprijedne neuronske mreže algoritmom propagacije pogreške unatrag.

Glavni nedostatak ADALINE neuronske mreže je činjenica da se njome uređaji konstruiraju isključivo fizičkim putem što je puno puta nepraktično i skupo za projektirati. U narednim poglavljima reći ćemo nešto o drugačijem izvodu umjetnih neuronskih mreža pomoću računalnih programa koje je puno lakše uklopiti u električne uređaje.

### 4.3.2. Perceptron

**1958.** znanstvenik Frank Rosenblatt u svojem radu *The perceptron: a probabilistic model for information storage and organization in the brain* na temelju MCP neurona i hebbovog principa učenja definira strukturu, perceptron<sup>6</sup>, čija je glavna ideja da se računalnim programom pokuša konstruirati sustav koji će moći učiti na temelju danih podražaja, dakle iz iskustva. U to vrijeme je ta ideja bila suluda i izazvala je mnoge kontroverze među pionirima umjetne inteligencije. The New York Times, dnevne američke novine, te iste godine opisuju perceptron kao zametak (engl. *embryo*) računala koji, jednom kada se kreira i usavrši, bi predstavljao prvi računalni sustav koji bi bio u stanju opažati, prepoznati i identificirati svoju okolinu bez ljudskog osposobljavanja ili kontrole<sup>7</sup>.



**Slika 4.4:** Umjetni neuron perceptron-a<sup>8</sup>

<sup>6</sup>Dolazi od latinske riječi *perceptio* što u prijevodu znači razumijeti (engl. *understand*).

<sup>7</sup>The New York Times, *Electronic 'Brain' Teaches Itself*, stranica 9, srpanj 13, 1958.

<sup>8</sup>Izrađeno pomoću powerpoint-a i na temelju modela umjetnog neurona iz literature (Čupić, 2016).



Struktura perceptrona je, kao što je rečeno ranije, motivirana strukturom MCP neurona. Na slici 4.4 nalazi se prikaz perceptrona koji je predstavljen kao generalizirani MCP neuron gdje je uloga ekscitacijskih, odnosno inhibicijskih ulaza zamijenjena težinama slično kao i kod ADALINE neurona. Novost u strukturi je prikaz funkcije skoka (engl. *step*) koja je implicitno postojala i kod izvornog MCP neurona, no ovdje je navedena eksplicitno jer se takav oblik koristi u literaturi. Također, ovakva struktura se u literaturi često naziva jednoslojni perceptron (engl. *single-layer perceptron*), a kasnije ćemo se susresti i sa višeslojnim perceptronom (engl. *multi-layered perceptron*).

Glavna zamisao koju je Rosenblatt dokumentirao u svom radu bila je izvesti nekakvu vrstu binarnog klasifikatora<sup>9</sup> i definirati algoritam kojim bi se taj isti klasifikator uspio naučiti (Rosenblatt, 1958). Ideja algoritma je da se težine ažuriraju pomoću pogreške koja se događa između željenog izlaza i stvarnog izlaza te se onda pomnoži sa nekom konstantom koju ćemo definirati u nastavku. Pravilo učenja perceptrona dano je algoritmom 1.

---

**Algorithm 1** Pravilo učenja perceptrona<sup>10</sup>

---

1. Ciklički prolazi kroz svih  $N$  uzoraka za učenje, jedan po jedan.
2. Klasificiraj trenutni uzorak.
  1. Ako se klasificira korektno, ne mijenjaj težine i
    1. ako je to  $N$ -ti uzastopni uzorak klasificiran korektno, prekini učenje,
    2. inače prijeđi na sljedeći uzorak.
  2. Ako se ne klasificira korektno, korigiraj težine perceptrona prema sljedećem izrazu:

$$w_i(k+1) \leftarrow w_i(k) + \eta * (d - y) * x_i \quad (4.5)$$


---

U procesu učenja klasifikatora može se uočiti da su spomenuti samo uzorci za učenje, no sada znamo i da postoji još nekoliko mehanizama koji uvelike doprinose da klasifikator razvije svojstvo generalizacije. Neki od tih mehanizama je npr. uporaba skupa za provjeru i skupa za testiranje, ali navedenim algoritmom smo htjeli pokazati elementarni proces učenja pa se time ovdje nismo htjeli zamarati. Ono što nas najviše zanima je interpretacija formule (4.5). Index  $i$  predstavlja  $i$ -tu vrijednost, tj.  $w_i$  je  $i$ -ta težina za  $i$ -ti ulaz što se može vidjeti i na slici 4.8,  $k$  je  $k$ -ta iteracija procesa učenja,  $d$

---

<sup>9</sup>Vidi 3.2.2 Klasifikacija.

<sup>10</sup>Iz literature (Čupić, 2016).

predstavlja željeni (engl. *desired*) izlaz neurona, dok  $y$  predstavlja stvarni izlaz neurona te  $\eta$  (eta) predstavlja parametar koji se zove stopa učenja (engl. *learning rule*). To je pozitivan realan broj malog iznosa (najčešće između 0.001 i 0.5) kojim se regulira kolikom mjerom će se ažurirati trenutna težina neurona. Ako je  $\eta$  jako mali broj, onda će proces učenja biti prespor, tj. težine će se mizerno malo ažurirati. Ako je  $\eta$  jako veliki broj, onda će proces učenja zasigurno divergirati jer će se težine ažurirati povećim brojevima i vrijednosti će "eksplodirati". Navedeni parametar ima jako bitnu ulogu prilikom optimizacije raznih *NP* teških problema koji koriste inačice algoritma gradijentni spust o kojemu će biti nešto riječi kasnije kada se dotaknemo algoritma propagacije pogreške unatrag.

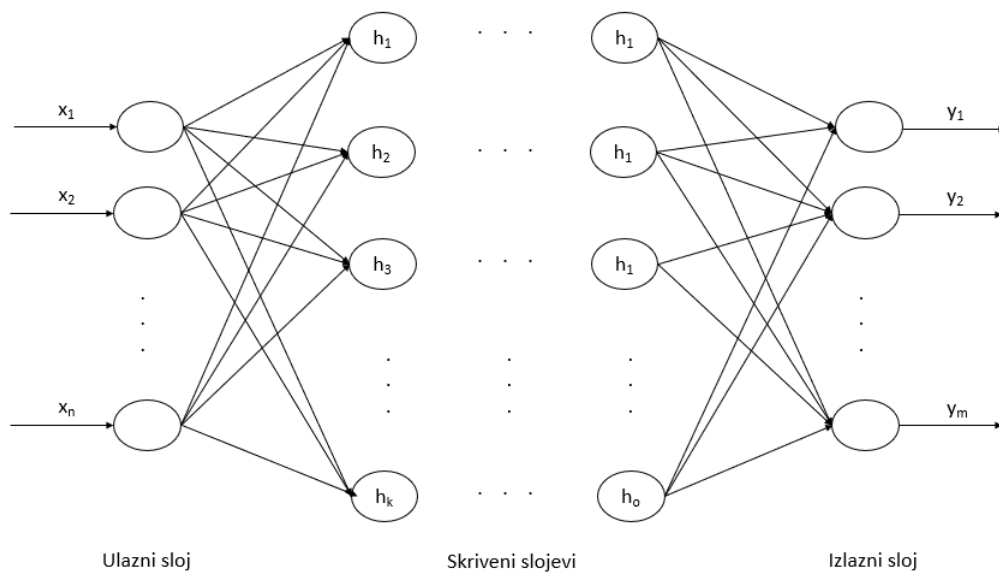
Dakle, Rosenblatt je uspješno uspio iznijeti svoje teze iako su mnogi sumnjali u njih. No, mreže koje su etiketirane kao jednoslojni perceptron imaju jednu veliku manu kao i prethodno navedene ADALINE neuronske mreže, a to je činjenica da uspješno djeluju samo kada je problem linearno interpretabilan, tj. kada su izlazi dotičnih neuronskih mreža linearno odvojivi (engl. *linearly separable*), a ADALINE neuronska mreža to još dodatno potvrđuje i imenom *LINEar Elements*. Ako se kratko vratimo na poglavlje u kojem smo govorili o tipovima klasifikacija, onda možemo vidjeti da smo problem klasifikacije podijelili na linearnu i nelinearnu klasifikaciju. Jednoslojni perceptroni su tipovi linearnih klasifikatora što znači da bi problem na slici 3.5 uspješno uspjeli klasificirati, dok problem na slici 3.6 ne bi jer se radi o nelinearnoj klasifikaciji. Primjeri koji se često vežu uz jednoslojne perceptrone jesu osnovne Booleove logičke operacije i, ili i negacija. Svaka od njih ima linearno interpretabilan izlaz te ih je bilo moguće realizirati. U to vrijeme je to bilo značajno otkriće, kao što je i spomenuto kod opisa primjene ADALINE neuronskih mreža. S druge strane, npr. operacija isključivo ili (XOR) nema linearno interpretabilan izlaz te se ne može realizirati jednoslojnim perceptronom kao takvim, već se mora konstruirati dvoslojni perceptron kako bi realizacija bila korektna <sup>11</sup>. No, ideja o višeslojnom perceptronu je tada bila tek u procesu stvaranja jer pravilo učenja perceptrona koje je predložio Rosenblatt je djelovalo samo na jednoslojne perceptrone. 1969. Marvin L. Minsky i Seymour A. Papert u svojem djelu *Perceptrons* jasno iznose nedostatke perceptrona koji je Rosenblatt definirao i to ponajviše o nemogućnostima rješavanja nelinearno odvojivih problema i učenja višeslojnih perceptrona. Time je i započela prva zima u eri umjetne inteligencije (engl. *1st AI winter*) kada se konektivistički pristup na kratko obustavio te se rodio simbolizam. Konektivizam će biti po strani sve negdje do 1989. kada će se po prvi put defini-

<sup>11</sup>Zanimljivi primjeri mogu se pronaći u skripti Umjetne neuronske mreže (Čupić, 2016).

rati algoritam kojim će višeslojni perceptron moći učiti, a to je algoritam propagacije pogreške unatrag.

## 4.4. Općenito o umjetnim neuronskim mrežama

Do sada smo se upoznali sa modelom umjetnog neurona i prvim modelima neuronskih mreža, ADALINE i perceptron i vidjeli smo razne nedostatke koji isti sadrže. Mali broj neurona može davati dosta uspješne rezultate, no ograničeni su na jako jednostavne i specifične probleme kao npr. konstrukcija jednostavnih Booleovih logičkih operacija što smo spomenuli ranije. Ljudski mozak je evidentno puno kompleksniji od strukture par neurona te se time došlo na ideju povezati veliki broj neurona u jednu veliku strukturu, tj. algoritam koji će biti vjerni prikaz ljudskog mozga, a takav algoritam je nazvan umjetna neuronska mreža. Na slici 4.5 je prikazana tipična građa jedne unaprijedne višeslojne potpuno povezane umjetne neuronske mreže.



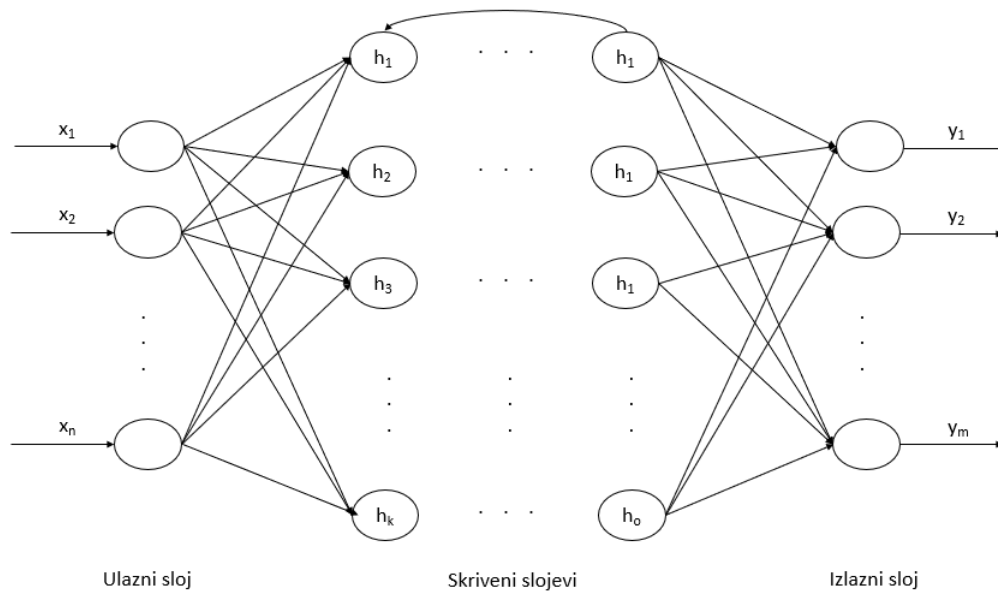
**Slika 4.5:** Tipična građa umjetne neuronske mreže<sup>12</sup>

Dakle, mreža se sastoji od tri različite vrste slojeva: ulazni, skriveni i izlazni sloj. Ulazni sloj je sloj na koji se dovode uzorci za koje želimo da mreža nešto napravi i ulazi su prikazani slovom  $\mathbf{x}$ . Izlazni sloj je sloj na kojemu će se nalaziti izlazi za navedeni uzorak u ulaznom sloju i prikazani su slovom  $\mathbf{y}$ . Skriveni sloj je sloj koji jedini nije vidljiv izvana i u kojemu se sva logika umjetne neuronske mreže događa.

<sup>12</sup>Izrađeno pomoću powerpoint-a i na temelju modela umjetne neuronske mreže iz literature (Čupić et al., 2013).

Neuroni skrivenog sloja označeni su slovom **h** i broj slojeva koji se nalaze u skrivenom sloju može biti proizvoljan. Primjetite koji se indeksi koriste kod enumeracije svakog od slojeva. U ulaznom sloju neuronska mreža očekuje  $n$  ulaza, u izlaznom sloju se očekuje  $m$  izlaza, dok u skrivenom sloju broj neurona može varirati po sloju, stoga su korišteni proizvoljni indeksi označeni slovima **k** i **o**.

U navedenoj strukturi pretpostavili smo da je svaki neuron nekog sloja povezan sa svakim neuronom njemu sljedećeg sloja i to samo tog sloja, osim izlaznog sloja koji je posljednji sloj neuronske mreže. Takve mreže ćemo nazivati slojevitim potpuno povezanim mrežama i njihova je zadaća da za neki fiksni ulaz generiraju neki stabilan izlaz (Čupić et al., 2013). U matematičkom smislu navedena operacija naziva se preslikavanje i to je glavna tema ovog rada kojoj ćemo se uskoro posvetiti detaljnije. Međutim, postoje i različite vrste mreža koje ne moraju biti niti slojevite niti potpuno povezane, a mogu imati i cikluse. Kod takvih mreža neuroni mogu biti povezani i sa neuronima prethodnog sloja te se time postiže da neuronske mreže budu promjenjive kroz vrijeme i da imaju mogućnost čuvanja nekog određenog stanja, tj. možemo reći da razvijaju nekakvu vrstu memorije. Na slici 4.6 je prikazana struktura jedne cikličke umjetne neuronske mreže koja je gotovo identična kao i na slici 4.5, no dodana je jedna povratna veza kojom se cijela ideja mreže mijenja.



**Slika 4.6:** Tipična građa cikličke umjetne neuronske mreže<sup>13</sup>

Često spominjemo sintagme poput struktura ili građa umjetne neuronske mreže,

<sup>13</sup>Izrađeno pomoću powerpoint-a i na temelju modela umjetne neuronske mreže iz literature (Čupić, 2016).

no u raznim literaturama koristi se sintagma arhitektura umjetne neuronske mreže pa ćemo i mi ubuduće koristiti upravo takav opis. Arhitektura umjetne neuronske mreže definira se na sljedeći način: uvijek započinje brojem neurona ulaznog sloja, zatim se definiraju skriveni slojevi i to svaki sloj je predstavljen brojem neurona tog sloja i na kraju se nalazi broj neurona izlaznog sloja. Npr. jedna od arhitektura mogla bi biti  $2 \times 5 \times 4 \times 3$  i nju tumačimo na sljedeći način: mreža se sastoji od četiri sloja; ulaznog, dva skrivena i izlaznog. Ulazni sloj sastavljen je od 2 neurona, prvi skriveni sloj je sastavljen od 5 neurona, drugi skriveni sloj je sastavljen od 4 neurona i izlazni sloj je sastavljen od 3 neurona. Onda bi npr. jednu inačicu rosenblattovog perceptrona mogli definirati kao  $3 \times 1$  jer on ne sadrži niti jedan skriveni sloj. Kasnije ćemo pogledati na koji način različite arhitekture utječu na uspješnost jedne unaprijedne neuronske mreže tijekom problema klasifikacije.

## 4.5. Unaprijedna neuronska mreža

Najčešće korišteni tip umjetne neuronske mreže je unaprijedna višeslojna potpuno povezana umjetna neuronska mreža (engl. *feedforward multilayered fully connected artificial neural network*) čiju smo arhitekturu već upoznali sa slike 4.5. Često se ista u literaturi naziva i samo unaprijedna neuronska mreža zbog lakšeg dokumentiranja rada pa ćemo i mi preuzeti tu konvenciju, a implicitno ćemo znati puno ime i svojstva. Također, unaprijedna neuronska mreža je pravi primjer višeslojnog perceptrona jer se očigledno sastoji od više slojeva te se može učiti danas već nekolicinom uspješnih algoritama.

Glavna zadaća unaprijedne neuronske mreže, kao što i samo ime nalaže, je da ulaze propagira kroz sve slojeve sve do izlaznog pri čemu se izvršava operacija mapiranja između ulaza i izlaza mreže na mnoštvo različitih načina. Također, sve operacije koje ćemo susresti se vrlo efikasno mogu prikazati matričnim računima čime se računalna izvedba implementacije i učenja unaprijedne neuronske mreže čini jako jednostavnim, no ponekad ipak to neće biti tako jednostavno jer postoji nekolicina parametara koje je potrebno "nariktati" različitim metoda koje ponekada i nisu zahvalne.

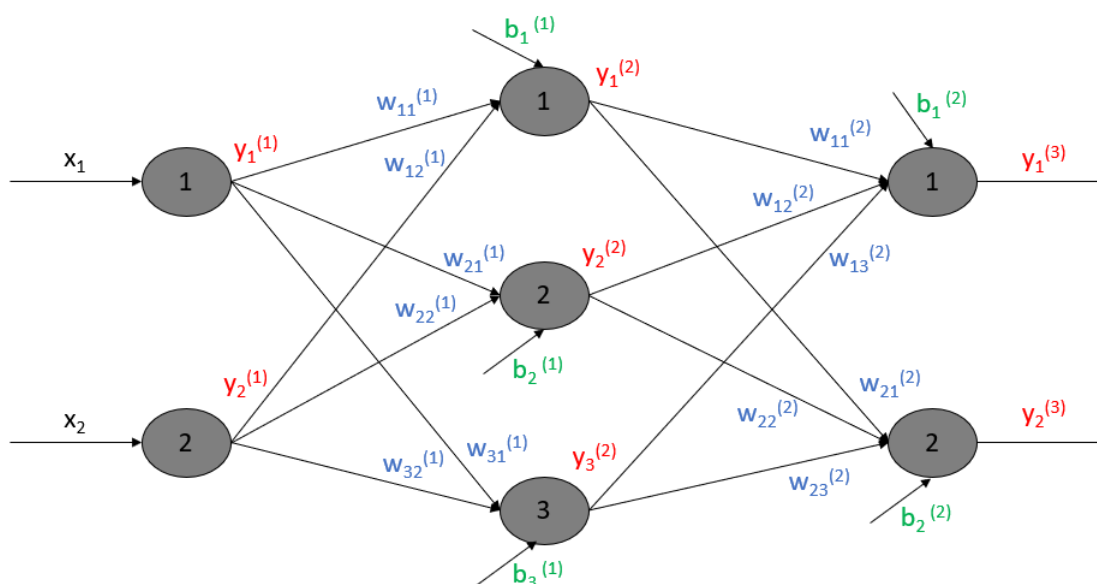
Pogledajmo jednu jednostavnu unaprijednu neuronsku mrežu s arhitekturom  $2 \times 3 \times 2$ . Crnom bojom su označeni ulazi u ulaznom sloju, plavom bojom su označene težine između dva neurona, zelenom bojom je označen bias, a crvenom bojom su označeni izlazi iz neurona pa krenimo redom s notacijama:

$w_{jk}^{(l)}$  : težina između  $k$ -tog neurona  $(l-1)$ -tog sloja i  $j$ -tog neurona  $l$ -tog sloja.

$b_j^{(l)}$  : bias j-tog neurona l-tog sloja.

$y_j^{(l)}$  : izlaz j-tog neurona l-tog sloja.

Prije nego se dotaknemo matematičke podloge, definirajmo ponovno strukturu umjetnog neurona.



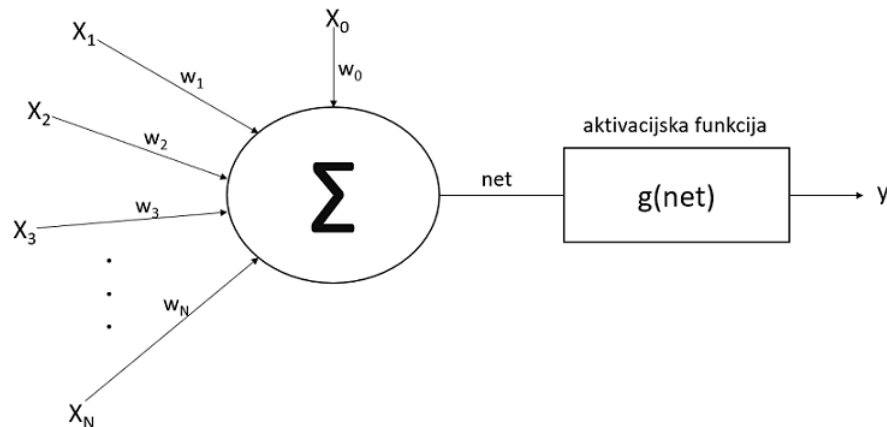
**Slika 4.7:** 2x3x2 arhitektura unaprijedne neuronske mreže<sup>14</sup>

Do ovog trenutka upoznali smo se samo sa dvije vrste neurona, a to su MCP neuron koji se koristio u rosenblattovu modelu perceptron i ADALINE neuron. Pokazali smo i da se tako definirani neuroni koriste samo kada je problem kojeg rješavamo linearno interpretabilan. To se događa zbog funkcije skoka koja se nalazi na izlazu neurona te zbog nje učenje višeslojnog perceptrona nije moguće jer funkcija skoka nije derivabilna te algoritam propagacije pogreške unatrag neće biti moguće provesti, ali o tome ćemo reći nešto detaljnije kada se dotaknemo učenja unaprijedne neuronske mreže. Zbog navedenog ćemo uvesti jednu generalnu strukturu umjetnog neurona koji ne mora koristiti funkciju skoka kao funkciju izlaza. Na slici 4.8 nalazi se ažurirana struktura umjetnog neurona. Vidimo da je struktura gotovo identična već spomenutom neuronu na slici 4.4 osim oznake za funkciju na izlazu. Funkcije koje se nalaze na izlazu neurona nazivaju se aktivacijske funkcije<sup>15</sup> i njihova je zadaća, kao što i samo ime nalaže, aktiviranje, odnosno deaktiviranje neurona ovisno o svojstvima funkcije i

<sup>14</sup>Izrađeno pomoću powerpoint-a i na temelju modela umjetne neuronske mreže iz literature (Nielsen, 2015).

<sup>15</sup>U starijim literaturama češći naziv bio je prijenosna funkcija.

vrijednosti argumenta funkcije, tj. net vrijednosti i takvu funkciju ćemo označavati sa slovom **g**. Dakle, MCP neuron kojeg smo nedavno obrađivali kao aktivacijsku funkciju koristi funkciju skoka. U narednom poglavlju ćemo dati pregled često korištenih aktivacijskih funkcija te njihovih formula kao i derivacija istih, no u ovom trenutku je jedino bitno da smo svjesni da one postoje radi matričnog prikaza unaprijedne neuronske mreže.



**Slika 4.8:** Umjetni neuron<sup>16</sup>

Prisjetimo se formule za izračun net vrijednosti koju smo definirali kada smo pričali o ADALINE neuronu (4.2):

$$net = \sum_{i=0}^n w_i * x_i$$

Primjetite da sumacija u ovom slučaju ide od 0 jer, ako se prisjetite strukture ADALINE neurona, postoji težina  $w_0$  koja predstavlja bias i ulaz  $x_0$  konstante vrijednosti koja iznosi 1. Na slici 4.7 bias pojedinog neurona je eksplicitno naveden slovom **b** te zbog efikasnosti prikaza je izostavljen konstantni ulaz. S obzirom da se ADALINE neuron jedino mogao nalaziti u jednoslojnoj neuronskoj mreži, korišteno je slovo **x** kao ulaz neurona, a u našem slučaju mreža je višeslojna te ćemo ulaze notirati sa slovom **y**. Ako se pitate zašto je tomu tako, onda se dobro pitajte jer je ta činjenica prešućena. Naime, ideja samog ulaznog sloja je da prenese dane ulaze na izlaz neurona te će tako izlazi ulaznog sloja biti jednaki ulazima, odnosno ne postoji nikakvo računanje net vrijednosti jer težine nisu prisutne. Također, svaki neuron osim ulaznih sadrži svoju net vrijednost. Uzevši u obzir navedene promjene i navedene notacije, net vrijednost pojedinog neurona možemo definirati na sljedeći način:

<sup>16</sup>Izrađeno pomoću powerpoint-a i na temelju modela umjetnog neurona iz literature (Čupić, 2016).

$$net_j^{(l)} = \sum_{k=1}^n w_{jk}^{(l)} * y_k^{(l)} + b_j^{(l)} \quad (4.6)$$

gdje  $net_j^{(l)}$  predstavlja net vrijednost  $j$ -tog neurona  $l$ -tog sloja pri čemu  $l$  predstavlja sloj koji nije ulazni, a sumacija se proteže do  $n$  pri čemu  $n$  predstavlja broj neurona u  $l-1$  sloju. Nakon izračuna, net vrijednost je potrebno "provući" kroz aktivacijsku funkciju kako bi se dobio izlaz za dotični neuron. Izračun za izlaz dan je sljedećom formulom:

$$y_j^{(l)} = g\left(\sum_{k=1}^n w_{jk}^{(l)} * y_k^{(l)} + b_j^{(l)}\right) \quad (4.7)$$

Evidentno je da se formula može zapisati i matrično po svakom sloju neuronske mreže izuzev ulaznog. Uvedimo još nekoliko notacija:

$\mathbf{W}^{(l)}$  : matrica svih težina  $l$ -tog sloja gdje su retci težine iz svih neurona  $(l-1)$ -tog sloja u jedan neuron  $l$ -tog sloja.

$\vec{b}^{(l)}$  : vektor svih bias vrijednosti  $l$ -tog sloja.

$\vec{net}^{(l)}$  : vektor svih net vrijednosti  $l$ -tog sloja.

$\vec{y}^{(l)}$  : vektor svih izlaza  $l$ -tog sloja.

Nakon uvedenih notacija, formule možemo prikazati na sljedeći način:

$$\vec{net}^{(l+1)} = \mathbf{W}^{(l)} * \vec{y}^{(l)} + \vec{b}^{(l)} \quad (4.8)$$

$$\vec{y}^{(l+1)} = g(\mathbf{W}^{(l)} * \vec{y}^{(l)} + \vec{b}^{(l)}) \quad (4.9)$$

Radi lakšeg razumijevanja, pokušajmo pokazati kako će izgledati matrica težina, vektor bias-a i vektor izlaza ako promatramo vezu između prvog i drugog sloja, odnosno ulaznog i skrivenog sa slike 4.7.

$$\mathbf{W}^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{12}^{(1)} \\ w_{21}^{(1)} & w_{22}^{(1)} \\ w_{31}^{(1)} & w_{32}^{(1)} \end{bmatrix}, \quad \vec{y}^{(1)} = \begin{bmatrix} y_1^{(1)} & y_2^{(1)} \end{bmatrix}, \quad \vec{b}^{(1)} = \begin{bmatrix} b_1^{(1)} & b_2^{(1)} & b_3^{(1)} \end{bmatrix}$$



#### **4.5.1. Aktivacijske funkcije**

### **4.6. Učenje unaprijedne neuronske mreže**

#### **4.6.1. Algoritam propagacije pogreške unatrag**

### **4.7. Primjena umjetnih neuronskih mreža**

## **5. Rezultati**

## **6. Zaključak**

Zaključak.

# LITERATURA

Dartmouth workshop. Dartmouth workshop — Wikipedia, the free encyclopedia, ožujak 2020. URL [https://en.wikipedia.org/wiki/Dartmouth\\_workshop](https://en.wikipedia.org/wiki/Dartmouth_workshop). [Pristupljeno 10-Maj-2020].

Donald O. Hebb. *The Organization of Behavior*. New York: Wiley, 1949.

Linear regression. Linear regression — Wikipedia, the free encyclopedia, maj 2020. URL [https://en.wikipedia.org/wiki/Linear\\_regression](https://en.wikipedia.org/wiki/Linear_regression). [Pristupljeno 12-Maj-2020].

Warren S. McCulloch i Walter H. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:115–133, 1943.

Christopher Moyer. How Google's AlphaGo Beat a Go World Champion, ožujak 2016.

Michael A. Nielsen. *Neural Networks and Deep Learning*. Determination Press, 2015. URL <http://neuralnetworksanddeeplearning.com/>. [Pristupljeno 10-Maj-2020.].

No free lunch theorem. No free lunch theorem — Wikipedia, the free encyclopedia, maj 2020. URL [https://en.wikipedia.org/wiki/No\\_free\\_lunch\\_theorem](https://en.wikipedia.org/wiki/No_free_lunch_theorem). [Pristupljeno 12-Maj-2020].

Phil Picton. *Neural Networks*. PALGRAVE, u drugom izdanju, 2000.

Reinforcement learning. Reinforcement learning — Wikipedia, the free encyclopedia, maj 2020. URL [https://en.wikipedia.org/wiki/Reinforcement\\_learning](https://en.wikipedia.org/wiki/Reinforcement_learning). [Pristupljeno 11-Maj-2020].

Frank Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, 65:386–408, 1958.

S. Russell i P. Norvig. *Artificial Intelligence - A Modern Approach*. Prentice Hall, u trećem izdanju, prosinac 2009.

Semi-supervised learning. Semi-supervised learning — Wikipedia, the free encyclopedia, travanj 2020. URL [https://en.wikipedia.org/wiki/Semi-supervised\\_learning](https://en.wikipedia.org/wiki/Semi-supervised_learning). [Pristupljeno 11-Maj-2020].

Supervised learning. Supervised learning — Wikipedia, the free encyclopedia, maj 2020. URL [https://en.wikipedia.org/wiki/Supervised\\_learning](https://en.wikipedia.org/wiki/Supervised_learning). [Pristupljeno 12-Maj-2020].

Symbolic artificial intelligence. Symbolic artificial intelligence — Wikipedia, the free encyclopedia, travanj 2020. URL [https://en.wikipedia.org/wiki/Symbolic\\_artificial\\_intelligence](https://en.wikipedia.org/wiki/Symbolic_artificial_intelligence). [Pristupljeno 10-Maj-2020].

Alan Turing. Computing Machinery and Intelligence. *Mind*, stranice 433 – 460, listopad 1950.

Unsupervised learning. Unsupervised learning — Wikipedia, the free encyclopedia, travanj 2020. URL [https://en.wikipedia.org/wiki/Unsupervised\\_learning](https://en.wikipedia.org/wiki/Unsupervised_learning). [Pristupljeno 11-Maj-2020].

Alan Woodruff. What is a neuron?, kolovoz 2019. URL <https://qbi.uq.edu.au/brain/brain-anatomy/what-neuron>. [Pristupljeno 13-Maj-2020].

M. Čupić, B. Dalbelo Bašić, i M. Golub. *Neizrazito, evolucijsko i neuroračunarstvo*, kolovoz 2013. URL <http://java.zemris.fer.hr/nastava/nenr/knjiga-0.1.2013-08-12.pdf>. [Pristupljeno 13-Maj-2020.].

Marko Čupić. *Umjetne neuronske mreže*, svibanj 2016. URL <http://java.zemris.fer.hr/nastava/ui/ann/ann-20180604.pdf>. [Pristupljeno 11-Maj-2020.].

Marko Čupić. *Uvod u strojno učenje*, travanj 2020. URL <http://java.zemris.fer.hr/nastava/ui/ml/ml-20200410.pdf>. [Pristupljeno 10-Maj-2020.].

## **Klasifikacija uporabom umjetnih neuronskih mreža**

### **Sažetak**

Sažetak na hrvatskom jeziku.

**Ključne riječi:** Ključne riječi, odvojene zarezima.

## **Classification Based on Artificial Neural Networks**

### **Abstract**

Abstract.

**Keywords:** Keywords.