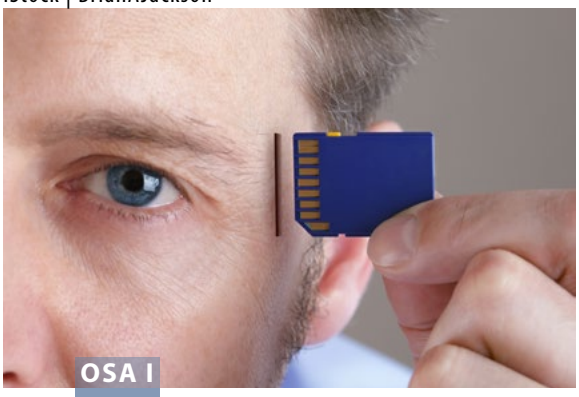
iStock | BrianAJackson



OSA I

TEKOÄLYN KANSSA

Ari Laitala ASUNTOKAUPOILLA

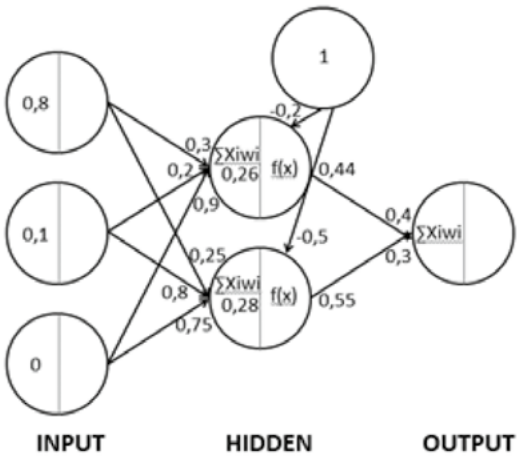
Miten verkko oppii?

Mitä on oppiminen? Kysymys on paitsi laaja ja vaikea, myös relevantti, sillä oppimisprosessista on tullut modernin kilpailuyhteiskunnan avaintoiminto. Yritysten ja kansakuntien välisessä talouskilpailussa menestyminen määrittyy entistä enemmän tehokkaan oppimisen kautta. Loppukädessä kysymys on suuressa määrin yksilöiden osaamisesta. Näin on ollut ainakin toistaiseksi, mutta nyt tätä kaikkea näyttäisi kovasti hämmentävän ilmiö nimeltä koneoppiminen (Machine Learning / ML). Kone-oppiminen on teko-älyn kehittämisen kulmakivi, sillä tekoälykään ei osaa mitään ilman oppimista.

SOPIVAT PAINOT VERKKOON

Käytännössä koneoppiminen pelkistyy neuroverkon laskentaan. Kaikesta hienosta terminologiasta ja pitkälle kehitetyistä tekoälysovelluksista huolimatta kyse on suurelta osin varsin yksinkertaisesta matematiikasta. Tavallisen FF-verkon eteenpäin laskennasta selviää valveutuneempi 9-luokkalainenkin vaivatta. Verkon taaksepäin laskeminen (backpropagation algoritmi) vaatii osittaisderivointia, jolloin korkeakoulun matematiikan peruskurssit ovat tasollisesti sopivaa taustatietoa.

Tässä yhteydessä taaksepäinlaskennan yksityiskohdat sivuutetaan, mutta asia käydään periaatteellisella tasolla läpi, joten koneoppimisesta on mahdollista saada kokonaiskuva.



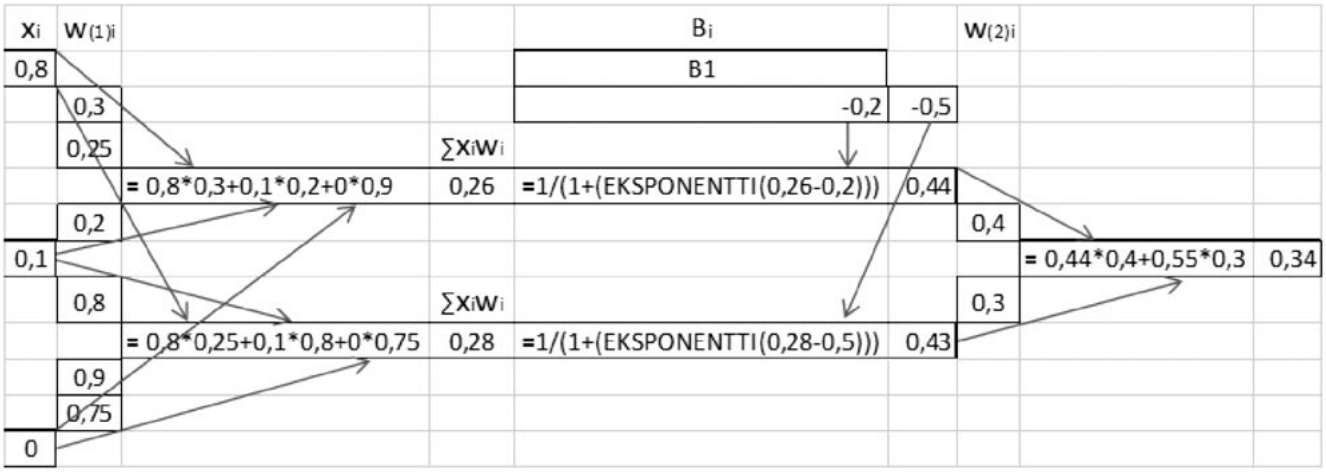
KUVIO 4.

Keinotekoinen FF-verkko, jossa neuronien välisten yhteyksien painoarvot ovat näkyvissä. Samoin näkyvillä ovat piilokerroksen neuronien ulostuloarvot (aktivaatioarvot) sekä ns. bias-neuroni.

Varsinainen laskenta alkaa siten, että kustakin Input kerroksen neuronista on yhteys kuhunkin seuraavan (piilo)kerroksen neuroniin (kuvio 4). Näillä yhteyksillä (*connections*) on kullakin oma painoarvonsa (*weights*). Laskennan lähtötilanteessa yhteyksien välisten painojen suuruus arvotaan, sillä vielä ei ole löydetty menetelmää, jolla painot voitaisiin jo heti alussa valita sopivan suuruisiksi. Jos olisi, varsinaista laskentaa ei edes tarvittaisi, sillä koko verkon laskennassa kyse on vain ja ainoastaan sopivien painoarvojen laskemisesta neuronien välisille yhteyksille. Koneoppiminen on matemaattisessa mielessä siis vain neuronien välisten yhteyksien optimointia siten, että verkolla lasketun mallin antamat tulokset eroavat todellisista arvoista mahdollisimman vähän.

Bias-neuroneita on tyypillisesti yksi kutakin verkon piilokerrosta kohden. Bias-neuronissakaan ei lasketa mitään, sillä sen arvo on aina 1. Bias-neuroni vaikuttaa piilokerroksen neuronien arvojen laskentaan täten vain yhteyksiensä voimakkuuden kautta ($1 * w_b = w_b$).

Jokainen input-kerroksen muuttuja siirtyy (monistuu) seuraavaan neuronikerrokseen siten, että muuttujan arvo kerrotaan kunkin yhteyden painolla. Varsinainen laskenta tapahtuu verkon piilokerroksissa, kussakin varsinaisessa neuronissa erikseen. Esimerkkiverkossamme varsinaista laskentaa suoritetaan siis vain kahdessa neuronissa.



KUVIO 5.

Esimerkkiverkon eteenpäin laskenta Excel-pohjaa hyödyntäen.

Laskennan vaiheita on kaksi. Ensin summataan sinne saapunut informaatio. Tämä tapahtuu kertomalla edellisestä kerroksesta saapuvien muuttujien arvot niiden painoilla. Näin laskettujen kertolaskujen arvot summataan ja saatu summa x viedään funktioon, joka laskee funktion ulostuloarvon $f(x)$. Laskentakerroksen neuroneissa oleva funktio on usein ns. sigmoid-funktio, joka määritellään seuraavasti:

$$f(x) = \frac{1}{1+e^{x+b}}$$

Funktiosta huomataan varsin helposti se, että se saa arvoja välillä]0,1[. Kun x (oikeammin $x + b$) on erittäin suuri, kasvaa myös nimittäjä erittäin suureksi, jolloin koko funktion arvo lähenee nollaa. Kun $x + b$ on erittäin pieni, lähenee jakolaskun tulos ykköstä. e on siis matematiikasta tuttu Neperin luku, lukuarvoltaan noin 2,718…. x on edellä mainittu kertolaskujen summa. Termiä b voidaan nimittää vakiotermiksi, jonka arvoa kuitenkin muutetaan laskennan edetessä (termistä käytetään englannin kielessä nimitystä bias tai treshold). Kyseessä on siis edellä mainittu bias-neuronin vaikutus laskentaan. Aina bias-termiä ei käytetä neuroverkossa lainkaan, eli tältä ja monilta muiltakin osin verkot eroavat toisistaan.

Piilokerroksesta neuronissa laskettu arvo siirtyy jälleen eteenpäin seuraavan kerroksen neuroneihin painokertoimen saattelemana. Meidän esimerkissämme seuraava kerros on jo ulostulokerros (output), jonka neuroneissa ei tyypillisesti enää suoriteta varsinaista laskentaa, vaan kerroksen neuroneissa ainoastaan summataan sinne saapuvat tulokset yhteen.

OHJATTUA OPPIMISTA

Vasta kun verkko on ensimmäinen kerran laskettu eteenpäin, voi verkon varsinainen oppiminen alkaa. Oppiminen voidaan toteuttaa useilla eri tavoilla, mutta perusratkaisu on ohjattu oppiminen *(supervised learning)*. Tämä on kuitenkin mahdollista vain silloin, kun tiedetään, mikä laskennan tuloksen pitäisi olla. Tämän artikkelin esimerkissä tiedossamme on kunkin vertailukaupan hinta. Näin ollen voimme käyttää ohjattua oppimista, jossa laskettua arvoa verrataan output-neuronin tiedossa olevaan arvoon. Jos vertailukaupan hintaa vastaava output-neuronin arvo olisi (tehdyn skaalauksen jälkeen) vaikkapa 0,4, havaittaisiin 0,34 – 0,4 = –0,06 suuruinen virhe. Tällainen tulos ei olisi siinä mielessä yllättävä, että yhteyksien painot on ensimmäisellä kierroksella arvottu.

Kun output-neuronin virhe on laskettu, lähdetään verkkoa laskemaan taaksepäin backpropagation-algoritmilla. Perusajatus on nyt siis se, että virhe lopputuloksessa johtuu virheellisistä arvoista neuronien välisissä painoissa. Virhe olisi siis ollut pienempi, jos verkon painoarvot olisivat olleet sopivammat. Näin ollen virhettä lähdetään korjaamaan muuttamalla neuronien välisten yhteyksien painoja. Ensin muutetaan niitä painoja, jotka johtavat output-kerroksen neuronista piilokerroksen neuroneihin.

Backpropagation-algoritmin avaintoimintona on sigmoid-funktion derivointi. Tässä tapauksessa derivointeja tarvitaan vain kaksi kappaletta, sillä ainoastaan piilokerroksen neuronit ovat varsinaista laskentaa sisältäviä neuroneja.

Viimeiseksi muutetaan niiden painojen arvoja, jotka johtavat (ensimmäisestä) piilokerroksesta takaisin input-kerroksen neuroneihin. Tämän jälkeen ollaan valmiita laskemaan verkko jälleen uudelleen eteenpäin. Seuraava laskentakierros lasketaan tyypillisesti kuitenkin jo seuraavalle vertailukaupalle. Koko aineisto lasketaan kuitenkin hyvin moneen kertaan läpi, eli nyt läpikäytyyn vertailukauppaan voidaan palata esim. satoja tuhansia kertoja.