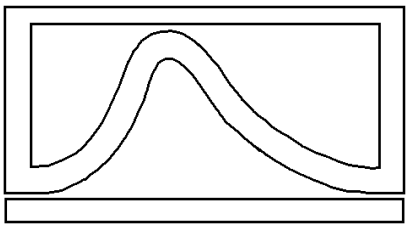
98



SUOMEN AKTUAARIYHDISTYS
THE ACTUARIAL SOCIETY OF FINLAND

WORKING PAPERS ISSN 0781- 4410

SUOMEN AKTUAARIYHDISTYS
The Actuarial Society of Finland

98
Tähtinen, Sami

Neuroverkkolaskenta ja sen soveltaminen vakuutusdataan

(2009)

Neuroverkkolaskenta ja sen soveltaminen vakuutusdataan

Sami Tähtinen

2009

Summary

This paper presents an introduction to neural networks and how they can be used for the data mining of actuarial data. Neural networks will probably develop more in the future. The emergence of more powerful personal computers, real applications of neural networks with real data and new theory for neural networks will make them more serious options for data mining for actuaries. One aim of this paper is also to stimulate interest in further reading of neural networks and other data mining methods.

The chapters can be divided in two main sections. In chapters 2 and 3 the development of neural networks and their basic definitions are discussed. Multilayer perceptron networks are described more in detail. Some other neural network architectures are also listed and briefly described. In chapter 3 some possibilities to visualize and interpret neural network results are covered.

The second part of the paper (chapter 4) illustrates how neural networks could be used for data mining of actuarial data. Several examples are presented. The application of neural networks with insurance data is not limited to only those examples presented. The aim of these examples is to show how neural networks can work in several different tasks and present an overview of neural network applications for actuarial data. The examples also illustrate how neural networks are very flexible and easy to use. The benefits of neural network modelling are highlighted. Besides the benefits, some problems of neural networks are also addressed.

Chapter 5 presents my conclusion on application of neural networks in modelling insurance data.

Chapter 6 lists some terms used in articles on neural networks.

Sisältö

Johdanto

Käsittelen harjoitustyössäni neuroverkkoja ja niiden soveltamista vakuutusalalla tiedon analysointiin. Viimeisen vuosikymmenen aikana erilaiset tilastolliset oppimisalgoritmit ovat saaneet paljon huomiota korkeakoulujen tutkimuksessa, liike-elämässä ja teollisuudessa. Niitä on sovellettu menestyksellä sellaisten tilastollisten prosessien ennustamiseen, joissa on suuri määrä mahdollisia selittäviä tekijöitä. Neuroverkkojen muodostamien mallien teho perustuu siihen, että verkot voivat esittää ei-lineaarisia, hyvin mutkikkaita selittävien tekijöiden yhteyksiä. Lisäksi keinotekoisilla neuroverkoilla on se etu, että eri muuttujien välisten yhteyksien funktiomuotoa ei tarvitse etukäteen valita.

Neuroverkoista on kirjoitettu suuri määrä kirjoja, ja esimerkiksi internetistä löytyy Googlesta hakusanoilla "neural network" noin 12 miljoonaa osumaa. Työni perustuu muutamaan neuroverkkoja melko perustasolla käsittelevään kirjaan, neurolaskennan soveltamista vakuutusalalla käsittelevään kirjaan [10] sekä joihinkin löytämiini artikkeleihin. Laskennallisissa esimerkeissä käytän vapaasti käytettävissä olevaa Joone-ohjelmistoa ja Databionicsin "ESOM"-verkkoja. Neurolaskennasta löytyy runsaasti tietoa yleisellä ja johdantotasolla, mutta tietoa todellisista sovelluksista tai neuroverkkojen käyttämisestä juuri vakuutusalalla ei ole kovin helppoa löytää.

Työn alussa luvussa 2 esitän neuroverkkojen teoriaa ja niiden yleisiä ominaisuuksia. Neuroverkkoja on esimerkiksi kritisoitu niiden "musta laatikko" –ominaisuuden vuoksi. Siksi esittelen luvussa 3 mahdollisuuksia neurolaskennan tulosten arvioimiseksi ja varmistamiseksi. Luvussa 4 kerron mahdollisuuksista soveltaa neurolaskentaa vakuutusdataan ja esitän joitakin tekemiäni pieniä esimerkkejä yritysajoneuvovakuutuksiin liittyen. Työn lopussa esitän johtopäätöksiäni neurolaskennan soveltamisesta vakuutusalalla tiedon analysointiin sekä neurolaskennan sanastoa.

Luku 1

Symbolit ja lyhenteet

Tärkeimmät symbolit

a_j, b_j reaaliarvoinen parametri

d_i haluttu i:s ulostulo

e_j virhevektorin j:s alkio

$\mathbf{e}$ virhevektori

E kokonaisvirhe

$E\{x\}$ muuttujan x odotusarvo

f, g funktioita

i, j, k, l, m, n indeksejä

N datarivien lukumäärä

o_j^l kerroksen l neuronin j ulostulo

p mallin sisääntulojen lukumäärä

s_j^l kerroksen l neuronin j aktivaatiofunktiolle syötettävä summa

w_{ij}^l kerroksen l neuronin j ja neuronin i välinen painokerroin

$\mathbf{x}$ sisääntulovektori

$\mathbf{y}$ ulostulovektori

η, α opetuskerroin

∂ osittaisderivaattaoperaattori

∇ gradienttiestimaatti

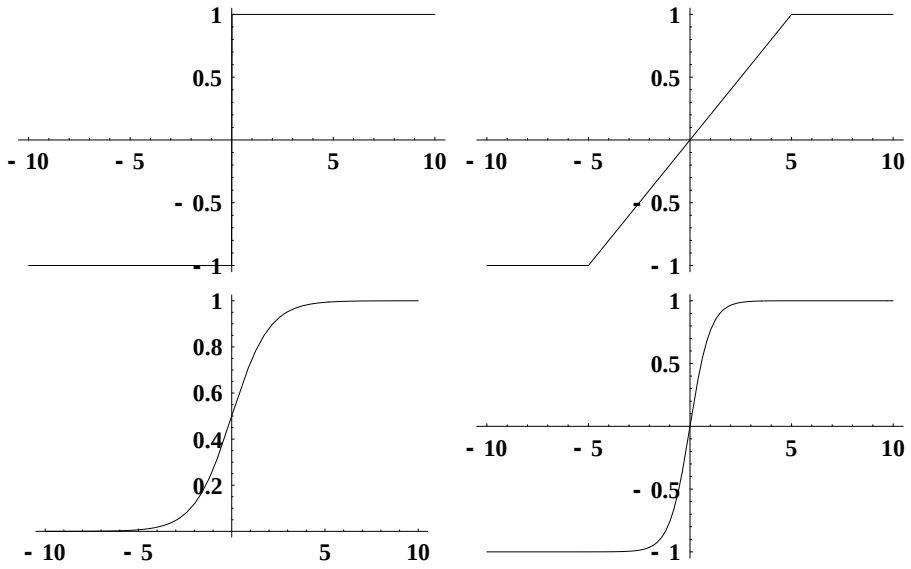
Luku 2

Neuroverkot

2.1 Keinotekoisten neuroverkkojen biologinen perusta

Neurolaskenta on yleisnimi kaikelle ongelmanratkaisulle, joka merkittäviltä osin perustuu neuroverkkojen käyttöön. Keinotekoiset neuroverkot pyrkivät karkealla tasolla jäljittelemään ihmisaivojen toimintaa. Aivot rakentuvat neuronien muodostamista massiivisesti rinnakkaisista rakenteista. Hermosolujen eli neuronien ominaisuudet yksin ja yhdessä muiden hermosolujen kanssa ovat olleet tekniikan esikuvana. Aivojen hermosolut ovat hitaita (10^{-3} s) verrattuna tietokoneiden laskentanopeuteen. Toisaalta aivoissa on erittäin paljon rinnakkaisia, samanaikaisesti toimivia hermosoluja. Hermosoluja arvioidaan olevan noin 10^{11} jokaisella ihmisellä. Lisäksi jokainen niistä on yhteydessä keskimäärin noin 10 000 muuhun hermosoluun. Hermosolut saavat viestejä sähköimpulssien muodossa muilta hermosoluilta dendriittejä eli tuojahaarakkeita pitkin. Käsiteltyään tiedot hermosolu lähettää viestin muille hermosoluille aksonia eli viejähaaraketta pitkin. Hermosolu käsittelee muilta soluilta saamaansa tietoa karkeasti yksinkertaistettuna ikään kuin porras- eli kynnysfunktion tapaan. Kuvassa 2.1 on esimerkkejä yksinkertaisista ja sigmoid-tyyppisistä kynnysfunktioista. Tarkemmin sanottuna, hermosolu lähettää sähköimpulssin eteenpäin, jos sen vastaanottamat impulssit ylittävät jonkin tietyn raja-arvon.

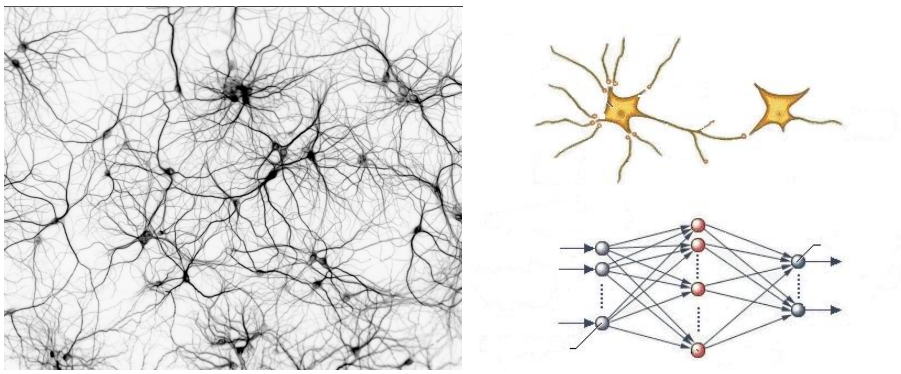
Aivojen muisti ja älykkyys rakentuvat eri hermosolujen välisiin kytkentöihin ja niiden voimakkuuteen. Keinotekoisissa neuroverkoissa aivojen rakennetta yritetään jäljitellä ja verkkoja ns. opetetaan muuttamalla neuronien eli verkon osasten välisiä painokertoimia. Neuroverkot kykenevät tallettamaan tietoa neuronien välisten painokertoimien avulla. Ne voivat oppia niille syötetystä opetusaineistosta erityisen opetusalgoritmin avulla monimutkaisia riippuvuuksia ja funktioita ja niillä on myös kyky yleistää oppimaansa uusiin havaintoihin. Neuroverkkojen oppiminen ei ole herkkää opetusaineistossa olevia puutteellisuuksia ja virheitä kohtaan. Toisaalta datan laatu ja määrä vaikuttavat ratkaisevasti neuroverkon tuloksiin, koska neuroverkko rakentuu suureksi osaksi suoraan datan pohjalta. Tietokoneiden laskentanopeus on suuri, mutta keinotekoisten neuronien lukumäärä vain pieni murto-osa aivojen neuronien lukumäärästä. Keinotekoi-



Kuva 2.1: Yksinkertaiset kynnysfunktiot a) askelfunktio ja b) paloittain lineaarinen funktio sekä sigmoid-tyyppiset funktiot c) logistinen (eli sigmoid-) funktio ja d) hyperbolinen tangentti –funktio

silla neuroverkoilla pyritäänkin ratkaisemaan lähinnä laskennallista älykkyyttä sisältäviä tehtäviä. Vakuutusalalla on hyvät edellytykset hyödyntää uusia neurolaskentamenetelmiä, koska vakuutusyhtiöille kertyy suuret määrät tilastoaineistoa eri vakuutuksista ja vakuutusasiakkaista. Lisäksi eri tekijöiden väliset yhteydet tilastoaineistoissa voivat olla monimutkaisia eikä niitä vältämättä edes tunneta.

Ohessa on kuva joukosta hermosoluja liittyneinä toisiinsa sekä rakennekuva hermosolusta ja keinotekoisesta neuronien verkosta.



Kuva 2.2: Joukko aivojen neuroneja toisiinsa liittyneinä; Oikean neuronin ja keinotekoisen neuronin vastaavuudet

Nykyiset neuroverkkojen mallit koostuvat joukosta elementtejä, formaaleja neu-

roneita, joilla on kullakin oma aktivaatiotasonsa. Elementtien välillä on yhteyksiä, joiden kautta aktiiviset elementit voivat aktivoida tai deaktivoida toisia elementtejä kuten aivoissakin tapahtuu hermosolujen välillä. Kullakin neuronien välisellä yhteydellä on oma yksilöllinen voimakkuutensa, joka määrittää, kuinka paljon neuroni vaikuttaa toiseen. Muodostuva verkko on dynaaminen järjestelmä, jolle annettu syöte saa aikaan aktivaation leviämisen koko verkossa. Neuroverkkojen verkkorakenteen ja aktivaatioperiaatteen lisäksi kolmas tärkeä piirre on oppimisperiaate eli se kuinka elementtien välisten yhteyksien voimakkuus muuttuu, kun neuroverkko oppii lisää.

Eräs keskeisimmistä neuroverkkojen ominaisuuksista on se, että niitä ei ohjelmoida eikä niihin kirjoiteta tietämystä sääntöinä vaan ne järjestäytyvät annetun datan perusteella itsenäisesti määriteltyä oppimismenetelmää käyttäen. Verkon sisältämä tietämys ei siis ole eksplisiittisessä symbolisessa tai sääntömuodossa. Puhutaan alisymbolisesta tasosta: verkon neuronien välisten yhteyksien voimakkuudet vaikuttavat lopputulokseen tavalla, jota voi olla hyvinkin vaikea ennakoida. Neuroverkkopäättelyä on rinnastettu myös inhimilliseen intuitiiviseen päättelyyn.

Neuroverkot kykenevät tekemään yleistyksiä annetuista esimerkeistä ja ne pystyvät tekemään hyviä mallinnuksia myös tiettyyn asteeseen asti ristiriitaisesta ja puuttellisesta tiedosta. Neuroverkot ovat eräänlaisia tilastollisia menetelmiä. Suurin ero on siinä, että neuroverkoissa eri tekijöiden välisen yhteyden luonteesta ei tarvita mitään etukäteistietoa eikä jakaumaoletuksia.

Termillä neuroverkko viitataan joukkoon melko erilaisiakin malleja, joilla kullakin on omanlaisensa verkkorakenteensa, aktivaatio- ja oppimisperiaatteensa. Yksi luokitteluperuste on se, onko oppiminen ohjattua vai ohjaamatonta. Ohjatussa lähestymistavassa määritellään oppimisvaiheessa sekä syöte että toivottu tulos. Ohjaamattomat opetusmenetelmät muodostavat kuvauksen syötteen ominaisuuksista ilman määrättyjä tuloksia. Verkot voidaan jakaa myös kolmeen päätyyppin toimintaperiaatteen mukaan

- Signaalinsiirtokuvaukset
- Tilansiirtokuvaukset
- Kilpailuoppiminen ja itseorganisaatio.

Tässä työssä keskitytään lähinnä MLP- eli Monikerros-perceptron -verkkoon (Multilayer perceptron network) ja SOM-itseorganisaatiokarttaan (Self-organizing map). Artikkelista [13] neuroverkkojen sovelluksista liiketoiminnassa selviää, että noin 90% käytännön neuroverkkosovelluksista on tehty MLP-neuroverkkojen avulla. Artikkelissa [15] tuo arvio oli jopa 95%. Valtaosassa näistäkin on käytetty ns. Backpropagation-opetusalgoritmia, josta myös tässä työssä kerron tarkemmin. Lisäksi joitakin muita, ainakin tärkeimpiä verkkomalleja selostetaan vaikkakaan ei yhtä tarkasti kuin MLP-verkkoa ja Backpropagation–algoritmia.

2.2 Historia

2.2.1 Neuroverkkojen alku 1940–luvulla

Tutkijat monelta eri tieteen alalta ovat tutkineet keinotekoisten neuroverkkojen teoriaa. Lähtökohtana on ollut tieto ihmisen hermojärjestelmän ominaisuuksista, mutta nykyään neuroverkkojen kehitys ja tutkimus ovat suureksi osaksi jo kaukana neurobiologiasta. Teoria on kehittynyt erilaisten ideoiden ja niiden toteutusten kautta. Kehitys on tapahtunut ennemminkin sykäyksittäin kuin tasaisesti. Tällä hetkellä neuroverkkojen kehityksessä on meneillään toinen kukoistuskausi. Ensimmäinen kausi kesti vuodesta 1943 aina 1960-luvun loppuun. Tämä toinen kausi on alkanut 1970-luvun loppupuolella.

Keinotekoisten neuroverkkojen taustalla olevat oppimisen, ehdollistamisen, neurofysiologian, ym. teoriat kehittyivät 1800- ja 1900-lukujen vaihteessa sellaisten tutkijoiden kuten Hermann von Helmholtz, Ernst Mach ja Ivan Pavlov toimesta. Hermosolujen eli neuronien toiminnalle ei tuolloin kuitenkaan ollut vielä matemaattisia malleja.

Neuroverkkojen teorian varsinaisena alkuna voidaan pitää fysiologian tutkijoiden Warren McCulloch ja Walter Pitts tutkimusta vuodelta 1943, jossa he esittelivät aivojen toiminnan logiikkaa ja osoittivat että keinotekoisten neuronien verkko voi laskea minkä tahansa aritmeettisen tai loogisen funktion syöttötiedoilleen. Tässä teoksessa esiteltiin myös ensimmäisen kerran keinotekoisen neuronin määritelmä. McCullogh oli etsinyt jo 1920-luvulta asti pienintä psyykkisten tapahtumien yksikköä eli "psychonia". Neuronin mallilla, jonka he tekivät, oli kaksi syötettä ja yksi tulos. Syötteillä oli sama paino ja tulos oli binäärinen joko 0 tai 1. Nykyään McCulloghin ja Pittsin neuronin mallia kutsutaan loogiseksi piiriksi. Tämän jälkeen vuonna 1949 Donald Hebb teki oletuksen, että ehdollistuminen (klassisena esimerkkinä Pavlovin koirat) johtui yksittäisten hermosolujen ominaisuuksista ja teki oletuksen myös hermosolujen oppimismekanismista. Hebb kehitti ensimmäisen neuroverkkojen opetusalgoritmin vuonna 1949.

2.2.2 1950–luvun yli-innostuminen

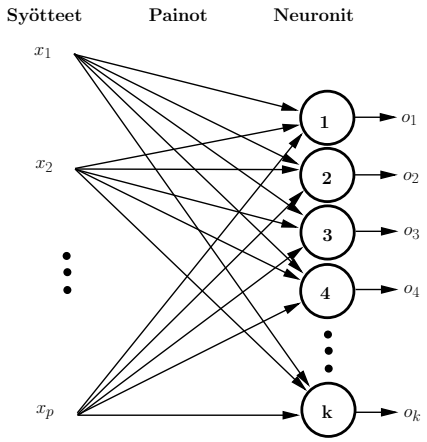
Neuroverkkoja laskennallisina rakenteina tutkivat 50- ja 60-lukujen taitteessa mm. Rosenblatt, Widrow ja Hoff. Ensimmäinen neuroverkkojen käytännön sovellus saatiin aikaiseksi 1950-luvun lopulla, jolloin Rosenblatt kehitti McCulloghPitts -neuronia tehokkaamman neuroverkko-mallin ja verkon opetusmenetelmän. Rosenblat esitti vuonna 1958 neuronille uuden mallin "perceptron". Hän yhdisti McCullogh-Pittsin yksittäisiä neuroneita yhteen ja muutti niiden välisten yhteyksien painoja yritys-erehdys–menetelmällä. Painokertoimien muuttamisella hän yritti saada neuronien verkon oppimaan. Painokerroin on se luku, jolla toisen neuronin antama tulos kerrotaan ennen kuin se tulee seuraavan neuronin käsiteltäväksi.

McCulloghin ja Pittsin neuronin malli kuvaa todellista aivojen neuronia paremmin kuin Rosenblattin neuronimalli, mutta perceptronista tuli nykyisten neuroverkkojen perusta josta kehitys jatkui. Rosenblatt sovelsi kollegoineen neuroverkkoa hahmontunnistukseen. Nämä aikaiset menestykset innoittivat paljon

lisää neuroverkkojen tutkimusta. Valitettavasti myöhemmin osoitettiin, että Rosenblattin neuroverkko pystyi ratkaisemaan vain hyvin rajallisen luokan ongelmia.

Rosenblattin neuroverkko ja verkon opetus painokertoimia muuttamalla toimi hyvin pienissä esimerkeissä. Se aiheutti tiedeyhteisössä euforian aallon, ja Rosenblatt sai puhua yliopistoilla täysinäisille saleille.

Suurin piirtein samaan aikaan Bernard Widrow ja Ted Hoff esittelivät uuden matemaattisen neuroverkon opetusalgoritmin, jolla neuronien välisten yhteyksien painoja muutetaan, sekä ensimmäisen varsinaisen yksikerroksisen neuroverkon (kuva 2.3). He käyttivät algoritmia lineaarisen neuroverkon opetukseen. Heidän neuroverkkonsa nimi oli ADALINE. ADALINE oli samankaltainen ja samantehoinen kuin Rosenblattin verkko. Menetelmässä käytetään apuna gradienttia ja neliövirheen minimointia. Widrow-Hoff -opetusalgoritmi on yhä nykyään käytössä ja se tunnetaan nimellä LMS tai "Least Mean Squares". Lineaarinen neuroverkkomalli sopii tiettyihin ongelmiin erinomaisesti, mutta useimmissa käytännön ongelmissa huonommin.



Kuva 2.3: Yksikerroksinen neuroverkko ADALINE. Ensimmäistä, sisääntulokerrosta ei ole tapana laskea kerrosten lukumäärään.

2.2.3 1960–luku: Yksinkertaisten neuroverkkojen rajoitteet

1960-luvulla yksikerroksisia neuroverkkoja käytettiin hahmontunnistuksessa, sääennusteissa ja säätötekniikassa. Tämä oli neurolaskennassa nopean kehityksen kautta. Tutkijoiden Hertz, Krogh ja Palmer lisäpanos nopeutti neuroverkkojen kehittelyä, kunnes tutkijoiden Marvin Minsky ja Seymour Papert esittämät epäilyt neuroverkkoalgoritmien toimivuudesta lähes pysäyttivät kehityksen. Valitettavasti sekä Rosenblattin että Widrowin neuroverkot kärsivät samoista ilmeisistä rajoituksista, joita Minsky ja Papert laajasti kritisoivat kirjassaan vuonna 1969. He näyttivät, että perceptronit eivät voi luokitella edes joitakin hyvin yksinkertaisia muotoja. Perceptronit eivät osaa erottaa esimerksi binaari-

muotoja (0,0), (1,1) muodoista (1,0), (0,1). Toisin sanoen perceptronit eivät selviäisi XOR- eli Ehdoton tai –operaatiosta. He myös osoittivat kuinka neuroverkon opetuksen vaatima aika kasvoi erittäin nopeasti, kun neuronien määrää lisättiin. Yksikerroksisia neuroverkkoja käytettäessä saadaan syötteen ja vasteen välille vain lineaarinen riippuvuus. Tämä on monissa sovelluksissa liian rajoittava tekijä. Rosenblatt ja Widrow olivat tietoisia näistä puutteista ja pakotetut esittämään uusia verkkoja, joissa ei olisi kyseisiä rajoituksia. Jo 1960-luvulla ymmärrettiin, että lisäämällä useita peräkkäisiä kerroksia rajoite voitiin poistaa. Neuroverkkoja tutkivat eivät kuitenkaan onnistuneet muokkaamaan opetusalgoritmeja uusille, monimutkaisemmille verkoille sopiviksi.

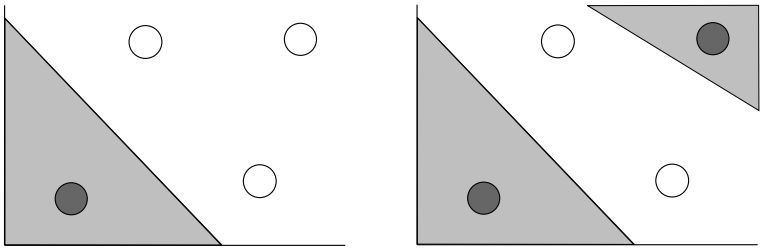
Perceptroniin yhdistettiin toisinaan esikäsittelykerros (preprocessing layer), jonka kertoimet ja yhteydet sisääntuloihin oli valittu satunnaisesti. Joskus tällainen muutettu neuroverkko oppi nopeasti tehtävän, vaikka se ennen esikäsittelykerroksen lisäämistä oli ollut mahdotonta tai erittäin hidasta. Esikäsittelykerroksen painokertoimien määrittelemiseksi ei kuitenkaan ollut mitään muuta menetelmää kuin yrityksen ja erehdyksen kautta.

Minskyn ja Papertin arvovalta oli suuri ja monet tutkijat uskoivat, että neuroverkkojen tutkimus oli tullut päätepisteeseen. Moni lopetti tutkimisen myös tehokkaiden tietokoneiden puuttumisen vuoksi. Neuroverkkojen tutkimus lähes pysähtyi. Samalla myös tutkimusrahoitus siirrettiin muille aloille. Tähän vaikutti myös pettymys, koska keinotekoiselle älylle ja ajatteleville tietokoneille ja roboteille oli 1950- ja 1960-luvuilla oli asetettu suuria odotuksia.

2.2.4 Lineaarisesti erottuvat funktiot

Kun neuroverkosssa on vain sisääntulo- ja ulostulokerrokset kuten kuvassa 2.3, pystyy se mallintamaan vain lineaarisesti erottuvia funktioita. Lineaarisesti erottuva neuroverkoista puhuttaessa tarkoittaa tilannetta, jossa sisääntulovektorit voidaan erottaa toisistaan yhdellä suoralla hypertasolla. Kun sisääntuloja on kaksi, pisteet toisistaan erottava taso on suora viiva. Kolmen sisääntulon tapauksessa erottaja on 3-ulotteinen suora taso. Useampiulotteiset tapaukset toimivat vastaavasti.

Kuvassa 2.4 on esimerkki 2-ulotteisesta tapauksesta.



Kuva 2.4: Lineaarisesti erottuvat ja erottumattomat 2-ulotteiset vektorit

Neuronilla, jolla on n kappaletta binaarisia sisääntuloja, voi olla 2^n kappaletta erilaista sisääntulovektoria. Kukin sisääntulovektori voi tuottaa joko tuloksen 1 tai 0. Ts. n binaarisesta sisääntulosta voi muodostaa 2^{2^n} binaarista funktiota.

Lineaarisesti erottuvien binaaristen funktioiden lukumäärä on

$$s_n = 2\sum_{i=0}^{n} \binom{2^n - 1}{i},$$

missä n on funktion parametrien lukumäärä.

Ohessa funktioiden lukumäärät on esitetty vielä taulukossa, $1 \leq n \leq 6$:

n	2^{2^n}	s_n
1	4	4
2	16	14
3	256	104
4	65536	1882
5	$4,3 \cdot 10^9$	94572
6	$1,8 \cdot 10^{19}$	5028134

Luvuista nähdään, että jo pienillä arvoilla n lineaarisesti erottuvien funktioiden osuus kaikista binaarisista funktioista tulee häviävän pieneksi. Tästä syystä yksikerroksiset (ts. vain sisääntulo- ja ulostulokerros) neuroverkot soveltuvat vain melko yksinkertaisten binaaristen funktioiden mallintamiseen ja käytännössä vain erikoistapauksissa.

2.2.5 1970-luvulta nykypäivään

Arvokkaita tuloksia saavutettiin kuitenkin myös 1970-luvulla. Vain harvat, mutta erittäin terävät tutkijat jatkoivat neuroverkkojen tutkimusta. Vuonna 1972 Teuvo Kohonen ja James Anderson toisistaan erillään keksivät uuden neuroverkkomallin, joka pystyi muistamaan. Myös Stephen Grossberg oli erittäin tuottelias tutkiessaan kyseistä uutta neuroverkkomallia. 1970-luvulta lähtien on Teknillisessä korkeakoulussa kehitelty itseorganisaatiokarttoja (SOM-verkot), joiden oppiminen perustuu ohjaamattomalle oppimiselle. Tutkimusryhmän johtajana on toiminut Teuvo Kohonen. Itseorganisaatiokartoille ei tarvita niin yksityiskohtaisia opetusalgoritmeja kuin monikerroksisissa neuroverkoissa. SOM-verkkoja käytetään informaation luokittelussa, kun ulostuloa eli vastetta ei tunneta. Esimerkiksi eri hahmot luokitellaan niin, että toisiaan muistuttavat hahmot kuuluvat samaan luokkaan.

Neuroverkkojen tutkimus alkoi taas todella, kun toisaalta tietokoneiden tehokkuus kasvoi ja kun Paul Werbos keksi 1974 ensimmäisenä (error) backpropagation –algoritmin neuroverkon opettamista varten. Hän tosin ei määritellyt algoritmia riittävän hyvin. Myöhemmin sen keksi uudelleen Parker (1985) ja LeCun (1985) toisistaan tietämättä sekä vielä Rumelhart ja McClelland (1986). Backpropagation on yleistys Widrow-Hoff LMS -algoritmista ja sen avulla voidaan opettaa verkkoja, joissa on neuroneja useassa tasossa. Menetelmässä neuronien välisiä painoja muutetaan verkon antaman tuloksen ja tiedetyn oikean tuloksen välisen virheen perusteella. Menetelmän nimen mukaisesti painoja muutetaan takaperin alkaen viimeisestä neuronien tasosta eli ulostulokerroksesta edeten kohti sisääntulokerrosta. Backpropagation-opetusalgoritmia käsitellään tarkemmin myöhemmin tässä työssä.

Tutkijat John Hopfield (1982) sekä Grossberg ja Cohen (1983) analysoivat neuroverkkojen dynamiikkaa ottamalla käyttöön tehokkaita, Lyapunovin funktioihin perustuvia menetelmiä. Heidän työnsä näytti kuinka oppivan neuroverkon voitiin ajatella olevan matkalla alas "energiamäkeä" kohti minimiä. Erityisesti Hopfield osoitti kuinka on mahdollista muodostaa neuroverkko (ns. Hopfieldin verkko), jolla on haluttu minimikohta. Sellainen verkko johtaa assosiatiiviseen muistiin, sillä lähtemällä osittainkin oikeasta ratkaisusta päästään lopputulokseen (minimikohtaan) verrattain nopeasti.

Neuroverkkojen energiafunktio johti pian ensimmäisen kerran fysiikan menetelmien soveltamiseen neuroverkkojen ominaisuuksien tutkimisessa. Esimerkiksi Amit osoitti vuonna 1989, että neuroverkko voi muistaa 0,14N hahmoa missä N on neuroverkon neuronien lukumäärä. Samaan arvioon oli Hopfield (1982) päätynyt kokeellisesti aikaisemmin. Gardner (1988) otti ensimmäisenä käyttöön käsitteen neuroverkkojen avaruus ja sitä on edelleen kehitellyt Amari (1991) differentiaaligeometrian uusien saavutusten avulla. Tutkijat Goombes ja Taylor (1993) ovat tutkineet neuroverkkojen muistikapasiteetin rajoja ja he ovat myös esittäneet miten näistä rajoista voidaan päästä niin, että neuroverkon muistikasitetti on koko N.

Yksi merkittävistä merkkipaaluista oli myös Boltzmanin koneen keksiminen (Hinton ja Sejnowski 1983). Boltzmanin konetta voidaan pitää Hopfieldin verkon laajennuksena. Verkon nimitys liittyy Boltzmannin todennäköisyysjakaumaan. Tämän verkon opetusalgoritmi on kuitenkin niin hidas, että se on estänyt monien käyttökelpoisten sovellusten tekemisen.

Vielä on kuitenkin ratkaisematta Minskyn ja Papertin esittämä kysymys neuroverkkojen opetusaikojen nopeasta kasvamisesta, kun muuttujien määrä nousee opetusaineistossa. Papert ja Minsky kirjoittivat vuonna 1989 uudestaan aiheesta ja varoittivat että neuroverkkojen teoria on rakennettu juoksuhiekalle ja että kaikki perustuu leikkikokoisiin aineistoihin. He epäilivät, että neuroverkot eivät enää toimi, kun opetusaineiston määrä on nostettu realistiselle tasolle. Tätä varoitusta ei tietenkään voida ohittaa kevyesti. Ongelmasta aiotaan selvitä lisäämällä neuroverkkojen teorian ymmärrystä ja kehittämällä niitä edelleen sekä lisäämällä tietokoneiden ja muiden neuroverkkosovellusten raakaa laskentavoimaa. Tämä tehokkuusongelma on nykykyisten tutkijoiden mielessä jatkuvasti.

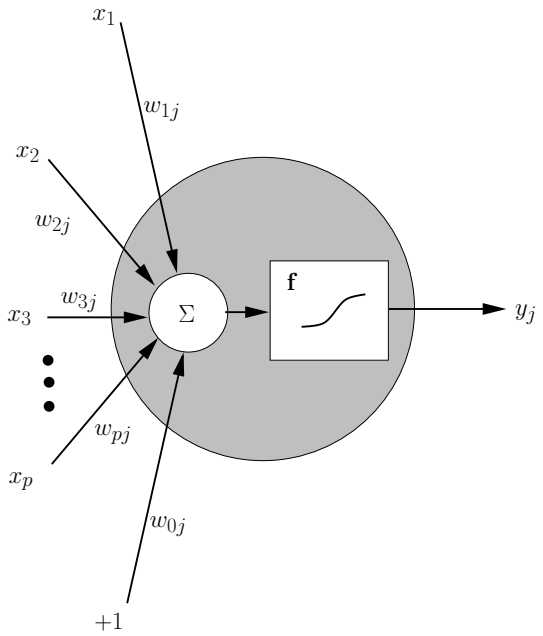
Erilaisia kaupallisia ja ilmaisia neuroverkkosovelluksia on julkaistu paljon 1990-luvulta lähtien. Neuroverkko-ohjelman tekeminen ei ole kauhean vaikeaa ohjelmointitaitoiselle. Tässä työssä käytän esimerkeissä ilmaisia Joone- ja ESOM-ohjelmistoja. Ne riittävät neuroverkkoihin tutustumiseen ja kokeilemiseen. Vakavampaan käyttöön sopivampia ovat esimerkiksi kaupalliset Matlab- ja SAS-ohjelmistojen neuroverkot.

1990-luvulta lähtien on varsinkin kirjallisuus neuroverkoista lisääntynyt paljon. Myös käytännön sovelluksia on tehty. Ymmärrettävästi yritykset eivät yleensä ole valmiita paljastamaan yksityiskohtia neurolaskentasovelluksistaan. Lisäksi julkaistuissa esimerkeissä on usein käytetty hyvin pieniä aineistoja ja neuroverkon rakennetta tai tuloksia ei selosteta tarpeeksi. Myös neuroverkkoihin liityvän matematiikan kehittely on vielä kesken.

2.3 Neuroverkkojen peruselementit

2.3.1 Keinotekoinen neuroni

Neuroverkko koostuu tyypillisesti joukosta yksinkertaisia rinnakkain toimivia laskentayksiköitä, joita kutsutaan (keinotekoisiksi) neuroneiksi. Ne koostuvat kolmesta peruselementistä: synapsit (liitokset muihin neuroneihin), summainyksikkö ja aktivaatiofunktio. Nämä neuronit ovat neuroverkkojen tärkeimpiä rakennusosia. Neuronin kaavio on esitetty kuvassa 2.5. Neuronit on järjestetty neuroverkossa kerroksiksi siten, että sisääntulovektorin arvot syötetään ensimmäiselle kerrokselle, ensimmäisen kerroksen tulokset toiselle kerrokselle ja niin edelleen ulostulokerrokseen (viimeinen) saakka. Ensimmäisen kerroksen signaalit tulevat siis verkon ulkopuolelta ja viimeisen kerroksen tulos tulee verkosta ulos. Niitä kerroksia, jotka ovat sisääntulojen ja ulostulokerroksen välissä, kutsutaan piilokerroksiksi (hidden layer). Neuroverkko, jossa on yksi piilokerros voi approksimoida mitä tahansa jatkuvaa funktiota, mutta kerroksen tarvittavien neuronien määrä voi olla suuri. Verkko, jossa on kaksi piilokerrosta, voi approksimoida mitä tahansa funktiota. Tätä ominaisuutta sanotaan neuroverkkojen *universaali approksimaattori* –ominaisuudeksi.



Kuva 2.5: (Keinotekoinen) neuroni. Sisääntulojen x_i painokertoimilla w_{ij} painotettu summa lisättynä kynnysarvolla w_{0j} syötetään aktivaatiofunktiolle, jonka tuloksena saadaan neuronin ulostulo y_j.

Neuroni j saa syötteenään signaalit x_i. Kukin sisääntuleva signaali kerrotaan sitä vastaavalla painokertoimella w_{ij}, joka kuvaa hermosolun synapsikytkennän voimakkuutta. Painokerroin voi olla joko negatiivinen, jolloin sillä on neuronin ulostuloa vaimentava vaikutus, tai positiivinen, jolloin sillä on neuronin ulostuloa voimistava vaikutus. Painokertoimilla kerrotut signaalit summataan ja summaan lisätään vielä ns. bias-termi eli kynnysarvo w_{0j}. Kynnysarvo mahdollistaa neuronin ulostulon poikkeamisen nollasta neuronin sisääntulojen painotetun summan ollessa nolla. Saatu summa s_j kynnystetään aktivaatiofunktiolla f. Aktivaatiofunktio jäljittelee hermosolun aktivaatiopotentiaalin toimintaa. Se rajaa neuronin ulostulon arvon yleisimmin välille $[-1, 1]$ tai $[0, 1]$. Aktivaatiofunktio voi sisältää epälineaarisuuden, mikä on neuroverkkojen tärkeä ominaisuus. Muutetut signaalit syötetään edelleen seuraavan tason neuroneille. Yleensä neuronit ovat täysin kytkettyjä kerrosten välillä eli jokainen neuroni kerroksessa l on kytketty jokaiseen neuroniin seuraavassa kerroksessa $l + 1$. Jokaisessa kerroksessa on useimmiten eri määrä neuroneita.

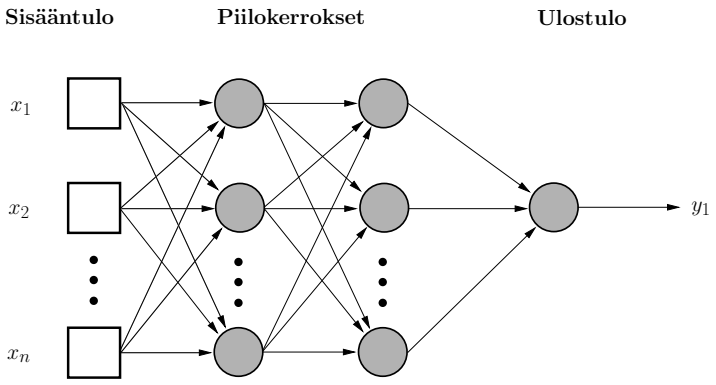
2.3.2 Monikerros-perceptron - eli MPL-neuroverkko

Monikerros-perceptron–verkko on paljon käytetty neuroverkkomalli. Se on yksisuuntainen ja siinä on sisääntulo-, ainakin yksi piilo- ja viimeiseksi ulostulokerros. Yksisuuntaiset monikerroksiset neuroverkot koostuvat verkkorakenteista, joissa neuronit on järjestetty kerroksiin ja kytkennät kerrosten välillä kulkevat yhdensuuntaisesti eteenpäin, ei koskaan taaksepäin. Saman kerroksen neuronien välillä ei ole kytkentöjä. Kaikki tällaiset neuroverkot ovat myös staattisia, mikä tarkoittaa tässä neuroverkon ulostulon olevan riippuvainen vain hetkellisistä sisääntulojen arvoista. Tämä tarkoittaa käytännössä sitä, että neuroverkolla ei ole muistia, johon voitaisiin tallettaa ajallista konteksti-informaatiota. Staattisen neuroverkon sisääntulojen tulee siis sisältää kaikki oleellinen informaatio, jonka perusteella ulostuloa mallinnetaan. Lukuisia erilaisia staattisia neuroverkkoja, jotka hyödyntävät edellisessä luvussa esitettyä keinotekoista neuronimallia, on esitetty. Tunnetuimpia ja käytetyimpiä ovat MLP-verkko (Multilayer Perceptron), kaskadikorrelaatio-verkko CC (Cascade Correlation network) ja radiaalikantafunktio-verkko RBF (Radial Basis Function network). RBF-verkon aktivaatiofunktiot eroavat edellä esitetyistä. Tässä työssä keskitytään lähinnä MLP-verkkoihin ja niiden mahdollisuuksiin ja soveltamiseen. Itseorganisaatiokartat SOM (Self-organizing map) ovat myös neuroverkkoja, mutta ne eroavat paljon MLP-verkoista. SOM-verkkoja selostetaan luvussa, jossa kerrotaan muista neuroverkkomalleista.

Ohjattu neuroverkko, jota kutsutaan monikerroksiseksi verkoksi (MLP), muodostuu neuronien kerroksista. Kuvassa 2.6 on esimerkki 3-kerroksisesta verkosta. Verkon rakenteesta on nähtävissä neuronien kerroksisuus ja signaalien kulku yhdensuuntaisesti sisääntulosta ulostuloon. Sisääntulokerroksen kautta neuroverkolle syötetään syötevektori $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$. Vektori syötetään kaikille ensimmäisen piilokerroksen neuroneille, joista saadaan niiden ulostulot y_j. Yleensä saman kerroksen kaikissa neuroneissa käytetään samantyyppistä aktivaatiofunktiota. Ensimmäisen piilokerroksen neuronien ulostulot syötetään edelleen seuraavan piilokerroksen kaikille neuroneille. Ulostuloneuronit saavat lopulta sisääntulonsa sitä edeltävän kerroksen neuronien ulostulosta. MLP-verkossa

voi olla yksi tai useampia piilokerroksia. Ulostulokerros sisältää jokaista haluttua ulostulon muuttujaa kohti yhden ulostuloneuronin. Verkon ulostulona saadaan vektori tai skalaari $\mathbf{y}$ vastaten jokaista sisääntulon vektoria $\mathbf{x}$. MLP-verkolle käytettävät merkinnät on koottu oheiseen taulukkoon.

w_{ij}^k	Kerroin kerroksen $k-1$ neuronin i ja kerroksen k neuronin j välillä
w_{0j}^k	Kynnysarvo neuronille j kerroksessa k
$s_j^k = w_{0j}^k + \sum_i w_{ij}^k o_i^{k-1}$	Aktivaatiofunktiolle syötettävä summa kerroksen k neuronissa j
$o_j^k = f(s_j^k)$	Kerroksen k neuronin j ulostulo. Funktio f on yleensä sama kaikissa kerroksen neuroneissa, mutta ei aina.
$x_i = o_i^0$	Verkon sisääntulovektorin $\mathbf{x}$ i:s alkio
y_i	Verkon ulostulovektorin $\mathbf{y}$ i:s alkio



Kuva 2.6: MLP-verkko. Verkolla on sisääntulokerros (input layer), yksi tai useampi piilokerros (hidden layer) ja ulostulokerros (output layer).

MLP-verkko muodostaa kuvauksen $\mathbf{y} = F(\mathbf{w}, \mathbf{x})$ sisääntulon $\mathbf{x}$ ja ulostulon $\mathbf{y}$ välillä painokertoimien $\mathbf{w}$ avulla. Tämä kuvaus on valitulla neuroverkkorakenteella ja kiinteillä painokertoimien arvoilla staattinen, epälineaarinen funktio. Jos käytetyssä MLP-verkossa on vain yksi piilokerros, voidaan sen muodostama funktio kuvata seuraavasti

$$y_k = f(w_{0k}^2 + \sum_{j=1}^{q} w_{jk}^2 \cdot f(w_{0j}^1 + \sum_{i=1}^{p} w_{ij}^1 x_i)),$$

missä y_k on verkon k:nnen ulostuloneuronin tulos, w_{jk}^2 ovat ulostulokerroksen neuronien painokertoimet, w_{ij}^1 ovat piilokerroksen neuronien painokertoimet ja

x_i ovat verkon sisääntulot. Tässä verkon sisääntulojen lukumäärä on p ja piilokerroksen neuronien lukumäärä q. Painokertoimien yläindeksit kuvaavat verkon kerrosta.

Ulostulossa käytetään usein myös lineaarista aktivaatiofunktiota, jolloin ulostulo muodostuu piilokerroksen neuronien painotettuna summana.

Monikerros-perceptron –verkossa syötetiedoista tehdään siis ensin lineaarisia kombinaatioita:

$$s_i = w_{i,0} + \sum_{j=1}^{n} w_{i,j} x_j,$$

missä x_j on j. selittävän muuttujan saama arvo ja $w_{i,0}$ ja $w_{i,j}$ ovat reaalivakioita. Kuten myöhemmin nähdään, nämä kertoimet w ovat verkon muutettavia kertoimia ja niiden arvot määritellään sopivaa parametrien estimointimenetelmää käyttäen. Parametrien estimointimenetelmää kutsutaan verkon opetusalgoritmiksi. Verkon oppiminen ja muisti ovat sen parametreissa. Kukin lineaarikombinaatio s_i esittää projektiota annettuun suuntaan selittävien muuttujien muodostamassa avaruudessa. Jokainen kombinaatio sisältää informaatiota monesta muuttujasta.

Seuraava vaihe neuroverkossa on ei-lineaarisen aktivaatiofunktion soveltaminen kullekin edellä saadulle arvolle s_i. Tuloksena saadaan kunkin neuronin ulostulot

$$y_j = f(s_j).$$

Yksinkertaisin aktivaatiofunktion muoto on tavallinen porrasfunktio (alaindeksi p tarkoittaa porrasta)

$$\gamma_p(u) = \begin{cases} 1 & \text{jos } u \geq \theta \\ 0 & \text{jos } u < \theta. \end{cases} \tag{2.1}$$

Tämän funktion huono puoli on siinä, että se ei ole differentioituva eikä sitä voida käyttää monissa sovelluksissa sen vuoksi.

Yleisimmät muutoksessa käytettävät kynnysfunktiot ovat hyperbolinen tangenttifunktio

$$\begin{aligned} y_i &= \tanh(s_i) \\ &= \frac{e^{s_i} - e^{-s_i}}{e^{s_i} + e^{-s_i}} \end{aligned} \tag{2.2}$$

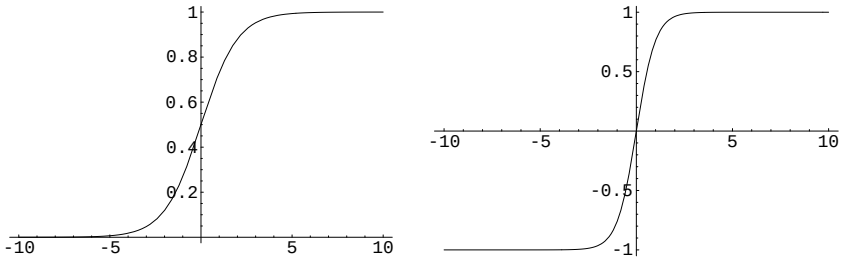
sekä sigmoid- eli logistinen funktio

$$y_i = \frac{1}{1 + e^{-s_i}}. \tag{2.3}$$

missä y_i on i. neuronin ulostuloarvo. Neuroverkkojen toiminnalle ja mahdollisuudelle esittää muuttujien korkea-asteisia yhteyksiä on tärkeää, että muutoksessa käytettävä funktio on ei-lineaarinen.

Mahdollisia, sopivia aktivaatiofunktioita on oikeastaan ääretön määrä, koska ainoat välttämättömät vaatimukset ovat 1) rajattu välille $[0, 1]$, 2) monotonisesti kasvava ja 3) differentioituva.

Kuvassa 2.7 nämä kaksi funktiota on kuvattu väliltä $[-10, 10]$. Kannattaa huomata, että funktiot muistuttavat toisiaan paljon. Ainoa suurempi ero on funktion tulosarvojen vaihteluväleissä: sigmoidifunktion tulokset on rajattu välille $(0, 1)$ ja hyperbolisen tangenttifunktion välille $(-1, 1)$. Näistä funktioiden kuvaajista on helposti nähtävissä myös kynnysparametrin merkitys. Kynnysparametrin avulla kynnyksen paikkaa voidaan siirtää x-akselin suunnassa.



Kuva 2.7: Sigmoidi- ja hyperbolinen tangenttifunktio

Viimeisessä vaiheessa kaikkien neuronien ulostulot yhdistetään yleensä lineaarisesti yhdeksi koko neuroverkon ulostuloksi

$$o(\mathbf{x}) = w_0^o + \sum_{i=1}^{n_h} w_i^o o_i,$$

missä $o(\mathbf{x})$ on neuroverkon ulostulo, n_h on neuronien lukumäärä ulostulokerrosta edeltävässä piilokerroksessa ja w_0^o ja w_i^o ovat reaalivakioita. Yläindeksi o tarkoittaa ulostulokerrosta. Ulostulokerroksen aktivaatiofunktiona voi olla muukin kuin lineaarifunktio. Tehtävän luonteen vuoksi joskus voidaan haluta verkon ulostulon olevan aina esimerkiksi välillä $[0, 1]$, jolloin aktivaatiofunktioksi sopisi paremmin hyperbolinen tangentti– tai logistinen funktio. Neuroverkon syötetiedot (n kappaletta) on esitetty vektorina $\mathbf{x}$.

Koko neuroverkon antama ulostulo voidaan esittää yhdellä lausekkeella, kun piilokerroksen kynnysfunktiona on hyperbolinen tangentti –funktio ja ulostulokerroksessa lineaarinen funktio

$$o(\mathbf{x}) = w_0^o + \sum_{i=1}^{n_h} w_i^o \tanh(w_{i0} + \sum_{j=1}^{n} w_{ij} x_j).$$

Tästä lausekkeestakin voi nähdä, että estimoitavia parametreja tulee helposti suuri määrä kun selittävien tekijöiden määrä nousee.

Sisääntulo-, yhden piilo- ja ulostulokerroksen muodostama 1-1-1 MLP-verkko voi mallintaa mitä tahansa suljetulla välillä määriteltyä jatkuvaa funktiota mielivaltaisella tarkkuudella. Teoria ei kuitenkaan kerro sitä, mikä sisääntulojen lukumäärä ja piilokerroksen neuronien lukumäärä olisi paras mahdollinen kussakin sovelluksessa. Todistus perustuu *Kolmogorovin teoreemaan* (1957):

Mikä tahansa jatkuva reaaliarvoinen funktio $f: \ x \mapsto \Re$, missä $x \in \Re^n$ voidaan esittää yhden muuttujan funktioiden lineaarikombinaationa

$$f(x_1, x_2, \ldots, x_n) = \sum_{k=1}^{2n+1} g_k(\sum_{i=1}^{n} \eta_{ki}(x_i)),$$

missä g_k ja $\eta_{k,i}$ ovat jatkuvia funktioita. Funktiot g_k ovat lisäksi f:stä riippuvia ja funktiot $\eta_{k,i}$ kasvavia funktioita. Yleensä funktiot g_k ja η_{ki} ovat erittäin monimutkaisia, epätasaisia, jne.

Kolmogorovin teoreemaa on myöhemmin paranneltu ja vuonna 1980 osoitettiin MLP-verkkojen universaali approksimaattori -ominaisuus. Universaali approksimaattori -ominaisuudesta on seuraavassa selostettu hieman lisää.

2.3.3 Universaali approksimaatio-ominaisuus

Neuroverkkoa voidaan pitää funktiona $T: \mathbb{R}^n \to \mathbb{R}^m$. Kääntäen voidaan kysyä, millaiset funktiot voidaan esittää neuroverkkona. Ilmenee, että teoriassa neuroverkolla voi approksimoida mitä tahansa jatkuvaa reaalifunktiota. Tästä johtuu, että neuroverkoilla sanotaan olevan ns. universaali approksimaatio –ominaisuus. Todistusta varten otetaan ensin käyttöön seuraava lause, jonka todistusta ei tässä esitetä.

Kolmogorov-Sprecherin lause. Jokaista $n \geq 2$ kohti on olemassa aidosti kasvava reaalifunktio ψ, jolla on seuraava ominaisuus:
Jokaista δ kohti on olemassa sellaiset $0 < q \leq \delta,\ q \in \mathbb{Q}$ ja $\lambda_{ij} \in \mathbb{R}$, että jokainen jatkuva funktio $f: [0,1]^n \mapsto \mathbb{R}$ voidaan esittää muodossa

$$f(x_1, \ldots, x_n) = \sum_{j=0}^{2n} g(\sum_{i=1}^{n} \lambda_{ij} \psi(x_i + qj)),$$

missä g on jatkuva reaalifunktio, joka riippuu funktioista f ja ψ sekä vakiosta q. Funktio ψ on kasvava reaalifunktio, joka riippuu n:stä, mutta ei f:stä.

Kolmogorov-Sprecherin lauseen avulla voidaan todistaa seuraava lause

Lause. Jos $T: [a_i, b_1]^n \mapsto \mathbb{R}^m$ on jatkuva, niin se voidaan esittää 2-tasoisena neuroverkkona (sisääntulo, yksi piilo- ja ulostulokerros), missä piilokerroksessa on $2n + 1$ neuronia.

Todistus. Sovelletaan edellistä lausetta T:n jokaiseen komponenttifunktion $T_k : [a_1, b_1]^n \mapsto \mathbb{R}$. Muodostetaan verkko, jossa piilokerroksessa on $2n + 1$ neuronia, joista jokainen suorittaa seuraavan operaation: aluksi $x_i \in [a_i, b_i]$ skaalataan välille $[0, 1]$ eli $y_i = (x_i - a_i)/(b_i - a_i)$ ja sen jälkeen piilokerroksen j. yksikön ulostuloksi asetetaan $o_j = \sum_{i=1}^{n} \lambda_{ij} \psi(y_i + qj)$ ja ulostulokerroksen neuroni suorittaa operaation $z_k = \sum_{j=0}^{2n} g_k(o_j)$.

Edellisellä lauseella on vain teoreettista merkitystä, sillä funktiot ψ ja g ovat usein monimutkaisia ja epätasaisia ja niiden numeerinen käsittely on hankalaa. Mitä voidaan tämän perusteella sanoa tavallisesta MLP-verkosta, jossa aktivaatiofunktiona on esimerkiksi $f(x) = 1/(1 + exp(-x))$? Tällöin verkon aktivaatiofunktio on (aina) jatkuvasti derivoituva, joten verkko ei voi esittää mielivaltaista jatkuvaa funktiota. Kuitenkin tällöinkin on voimassa seuraa tulos.

Neuroverkon esityslause.
Olkoon ϕ aidosti kasvava reaalifunktio jolle $\lim_{x \to -\infty} \phi(x) = 0$ ja $\lim_{x \to \infty} \phi(x) = 1$, $K \subset \mathbb{R}^n$ kompakti ja $f: K \mapsto \mathbb{R}$ jatkuva. Tällöin jokaista $\epsilon > 0$ kohti on olemassa $N \in \mathbb{N}$ ja sellaiset $c_i,\ \theta_i,\ w_{ij} \in \mathbb{R}$, että

$$\max_{x \in K} |f(x) - g(x)| < \epsilon,$$

missä

$$g(x_1, \ldots, x_n) = \sum_{i=1}^{N} c_i \phi(\sum_{j=1}^{n} w_{ij} x_j - \theta_i).$$

Funktioksi ϕ kelpaa esimerkiksi sigmoidi-funktio $\phi(z) = 1/(1 + e^{-z})$. Siten jokaista jatkuvaa funktiota voidaan approksimoida mielivaltaisen tarkasti 2-kerroksisella MLP:llä.

Tämän lauseen todistus ohitetaan. Sen voi kuitenkin löytää esimerkiksi K. I. Funashin artikkelista *On the approximate realization of continuous mappings by neural networks*, Neural Networks 2 (1989), sivut 183-192.

Soveltamalla edellistä lausetta erikseen jokaiseen komponenttifunktioon näemme, että jokaista jatkuvaa funktiota $T : \mathbb{R}^n \to \mathbb{R}^m$ voidaan approksimoida mielivaltaisen tarkasti MLP:llä. Siten 2-kerroksiset neuroverkot ovat universaaleja approksimaattoreita. Vaikka annetun tehtävän suorittamiseen approksimatiivisesti aina löytyy 2-kerroksinen neuroverkko, niin se ei välttämättä ole paras mahdollinen neuroverkkomalli käytännön kannalta. Yleensä "paras" riippuu annetusta tehtävästä ja siitä mitä käytännössä tavoitellaan. Kirjallisuus ja teoria ovat yleensä vaiti näistä puolista. Universaali approksimaatio-ominaisuus myös katoaa, jos laskuja ei voida suorittaa tarkasti vaan vain jollakin äärellisellä tarkkuudella.

2.4 Neuroverkon toiminta

Neuroverkot voidaan jakaa kolmeen luokkaan opetusmenetelmän mukaan. Opetusmenetelmiä ovat ohjattu (supervised), vahvistava (reinforcement) ja ohjaamaton (unsupervised) opetus. Ero näiden välillä on se, että verkon opettamisessa ohjatun verkon syöttötiedot sekä tavoiteltu tulos tiedetään. Tavoitteena on tällöin löytää syötetietojen ja tulosten välinen yhteys. Ohjaamattomassa verkossa vain syötetiedot ovat tiedossa. Tällöin tavoitteena on saada selville ominaisuuksia opetusaineistosta, joiden mukaan tiedot voidaan sitten luokitella. Vahvistavassa opetuksessa ei ole välttämättä tiedossa tarkkaa haluttua tulosta, vaan ainoastaan tieto siitä kuinka hyvä verkon tuottama tulos on. Verkon tarve muuttaa toimintaansa suhteessa esimerkkitapaukseen on sitä suurempi, mitä virheellisemmäksi verkon tuottama tulos arvioidaan. Ts. verkko oppii vain virheistään. Esimerkiksi joidenkin pelien oppimiseen tämä opetusmenetelmä voisi olla luonnollinen: pelissä ei ehkä ole yksiselitteistä oikeaa vastausta eli siirtoa tietyssä tilanteessa, mutta siirtojen hyvyyttä voidaan arvioida.

Ennen neuroverkon opettamista neuronien liitoksille annetaan sattumanvaraiset painokertoimet ja vakiot. Seuraavassa vaiheessa käyttämällä opetusaineiston yhtä havaintoa lasketaan eri neuronien tulokset ja koko verkon lopputulos. Jos verkon lopputulos on sama kuin havainnon tiedetty vastaus, kertoimia ei tarvitse muuttaa. Muussa tapauksessa painokertoimia muutetaan opetusalgoritmin mukaisesti. Sen jälkeen verkon tulos lasketaan uudelleen jollakin toisella havainnolla. Tätä prosessia jatketaan kunnes saavutetaan haluttu tarkkuus tai jokin muu pysäytyssääntö toimii.

Jos tieto kulkee verkossa vain eteenpäin seuraaviin neuroneihin, verkkoa sanotaan myötäkytketyksi tai eteenpäin-syöttäväksi (feed-forward) verkoksi. Jos

neuroverkko pyrkii tarkentamaan toimintaansa virheellisten tulosten avulla, sillä sanotaan olevan backpropagation -ominaisuus.

Neuroverkon mallinnuksessa jokaiselle sovellukselle on erikseeen etsittävä

- Oikea määrä neuroneita
- Oikea määrä kerroksia
- Sopiva opetusalgoritmi ja siihen liittyvät aloitusparametrit

Opetusaineiston määrä ja ominaisuudet sekä ongelman kompleksisuus vaikuttavat näihin asioihin. Seuraavassa kappaleessa esitetään numeerinen esimerkki neuroverkosta. Esimerkissä lopulliset painokertoimet voidaan laskea helposti, joten kovin realistinen esimerkki se ei ole.

2.4.1 Esimerkki

Seuraavaksi esitetään pieni ja aika teoreettinen esimerkki neuroverkosta. Esimerkissä verkon painokertoimet saadaan analyyttisesti. Yleensä tilanne ei ole tällainen, vaan painokertoimet saadaan verkon opetusalgoritmin avulla numeerisesti. Itse asiassa neuroverkon opetusalgoritmi on vähintään yhtä tärkeä tai todennäköisesti tärkeämpi tekijä neuroverkossa kuin itse verkon rakenne tai siinä käytetyt funktiot.

Oletetaan, että neuroverkolla halutaan mallintaa seuraavanlaista yksinkertaista funktiota

$$f(x) = c, \text{ kun } a \leq x \leq b, \text{ ja } f(x) = 0 \text{ muulloin.}$$

Ensin pitää päättää neuroverkon rakenne. Nyt on luonnollista valita sisääntuloksi vain x ja ulostulojen lukumääräksi myös 1 eli $z = f(x)$. Piilokerroksen neuronien lukumääräksi riittää 2. Valitaan aktivaatiofunktioiksi piilokerroksen neuroneissa sigmoidi- ja ulostuloneuronissa lineaarinen funktio. Kuvassa 2.8 on esitetty tällainen neuroverkko ja sen painokertoimet.

Jotta tarvittavat painokertoimet saataisiin selville, kirjoitetaan $f(x)$ tavallisen porrasfunktion γ_p (2.1) avulla ($\theta = 0$):

$$\begin{aligned} f(x) &= c\{\gamma_p(x-a) - \gamma_p(x-b)\} \\ &= c\gamma_p(x-a) - c\gamma_p(x-b). \end{aligned} \tag{2.4}$$

Sigmoidifunktio $\gamma(u)$ (2.3) voi approksimoida porrasfunktiota mielivaltaisen tarkasti

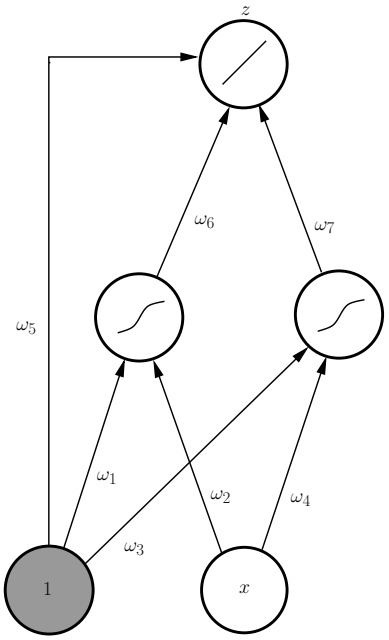
$$\gamma(ku) \to \gamma_p(u) \text{ kun } k \to \infty. \tag{2.5}$$

Nyt lauseke (2.4) voidaan kirjoittaa muodossa

$$f(x) = c\gamma(k(x-a)) - c\gamma(k(x-b)), \text{ kun } k \text{ on suuri.} \tag{2.6}$$

Toisaalta neuroverkon ulostulo voidaan laskea painokertoimien ja aktivaatiofunktioiden avulla seuraavasti

$$z = \omega_5 + \omega_6\gamma(\omega_1 + \omega_2 x) + \omega_7\gamma(\omega_3 + \omega_4 x). \tag{2.7}$$



Kuva 2.8: Esimerkin neuroverkko ja painokertoimet.

Näistä lausekkeista (2.6) ja (2.7) voidaan ratkaista kaksi mahdollista painokertoimien arvojen joukkoa

Painokerroin	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7
Ratkaisu 1	$-kb$	k	$-ka$	k	0	$-c$	c
Ratkaisu 2	$-ka$	k	$-kb$	k	0	c	$-c$

Jo tästä esimerkistä nähdään, että neuroverkko yhdellä piilokerroksella voi approksimoida mielivaltaisella tarkkuudella mitä tahansa yhden syötteen funktiota kunhan piilokerroksessa on riittävästi neuroneita. Näin on, koska funktioita voidaan approksimoida paikallisilla "hypähdyksillä". Käytännässä ei kuitenkaan voida tietää kuinka monta neuronia tarvittaisiin ja halutun tarkkuuden saavuttaminen voi vaatia reilusti liian suuren määrän neuroneita. Toisaalta reaalimaailman ongelmissa funktiot ovat yleensä sillä tavalla hyvin käyttäytyviä, että jopa jo muutama piilokerroksen neuroni voi riittää.

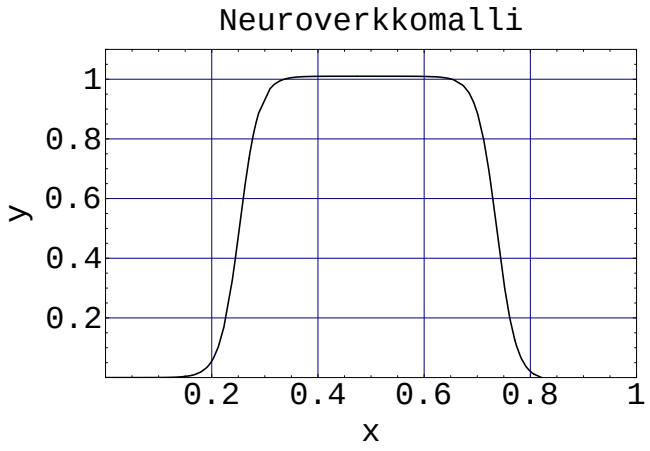
Seuraavassa on esimerkki ratkaistu neuroverkon avulla numeerisesti. Neuroverkon opetusaineistoa varten valittiin satunnaisesti 200 lukua väliltä $[0, 1]$. Estimoitavan funktion parametreiksi valittiin

$$a = 0,25,\ b = 0,75 \text{ ja } c = 1,00.$$

Kertoimiksi tuli seuraavat:

Painokerroin	ω_1	ω_2	ω_3	ω_4	ω_5	ω_6	ω_7
	-13,77	54,59	-40,39	54,77	0	1,01	-1,02

Neuroverkon painokertoimet vastaavat Ratkaisun 2 kertoimia. Kuvassa 2.9 on kuvattu neuroverkon antama malli 200 opetusaineiston pisteessä.



Kuva 2.9: Neuroverkon muodostama malli.

2.5 Muita neuroverkkoja kuin MLP-verkot

Erilaisia neuroverkkomalleja ja opetusalgoritmeja ja niiden muunnelmia on suuri määrä. Jotkut verkot on tarkoitettu vain johonkin tiettyyn tarkoitukseen, toiset sopivat melkein minkä tahansa regressio-, ryhmittely-, hahmon tunnistus-, jne. ongelman ratkaisemiseen ainakin periaatteessa. Erilaiset neuroverkkomallit voidaan erotella seuraavilla ominaisuuksilla:

- Neuronien välisten yhteyksien topologia
- Opetusmenetelmä
- Opetusalgoritmi

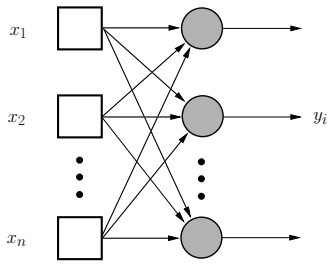
Yhteyksien topologia määrää miten neuronit ovat yhdistyneet toisiinsa. Opetusmenetelmä kertoo sen miten verkkoa opetetaan, ohjatusti vai ohjaamattomasti. Opetusalgoritmi määrää sen, miten virhettä opetuksen aikana mitataan ja miten neuronien välisiä painokertoimia päivitetään.

Neuroverkot voidaan jakaa karkeasti kolmeen päälajiin: monikerros- eli MLP-neuroverkot, radiaaliset kantafunktio -neuroverkot ja itseorganisoituvat verkot.

2.5.1 Perceptron ja Adaline

Yksinkertaisin ohjatusti oppiva neuroverkko on sellainen, jossa on sisääntuloja ulostulokerros mutta ei välissä olevaa piilokerrosta. Rosenblattin 1958 kehittelemä perceptron (kuva 2.10) oli ensimmäinen tällainen oppiva neuroverkko. Se on sukua Widrow'n v. 1960 esittelemälle Adaline-neuronille. Adalinen ja

perceptronin ero on siinä, että jälkimmäinen olettaa syötteiden olevan binaarilukuja ja binarisoi myös oman ulostulonsa kynnystämällä, kun taas Adaline on lineaarinen neuroni joka sallii myös jatkuva-arvoiset syötteet. Näistä neuroneista koostuva yksitasoinen verkko edustaa yksinkertaisinta ohjatun oppimisen lajia, missä verkko opetetaan vastaamaan sisääntuloihin halutulla tavalla. Vain ulostulokerroksen painokertoimia voidaan muuttaa.



Kuva 2.10: Perceptron-verkko.

Perceptron–verkon hyviä puolia on helppokäyttöisyys. Sen saa kohtuullisesti toimimaan varsin vähillä tiedoilla ja se ei ole kovin herkkä oppimaan opetusaineistoa ulkoa. Menetelmän ongelmana on ilmaisukyky. Perceptron ei näet pysty tekemään epälineaarisia kuvauksia sisääntuloista. Tätä pidettiin aikanaan vakavana puutteena ja USA:ssa se johti neurolaskennan lähes täydelliseen hylkäämiseen aina 1980-luvun puoliväliin asti. On kuitenkin monia tehtäviä, jotka ovat luonteeltaan lineaarisia. Tällöin perceptron voi olla sopiva menetelmä niiden ratkaisemiseen. Perceptron-verkon hyvä puoli on myös, että teorian mukaan se aina löytää oikean ratkaisun painokertoimille edellyttäen että ratkaisu on olemassa. Alussa painokertoimet kannattaa valita pieniksi satunnaisluvuiksi $[-0,1; 0,1]$, jotta algoritmi konvergoisi tasaisesti ratkaisuun, mutta ratkaisu kyllä löytyy joka tapauksessa, jos se vain on olemassa.

Perceptronin opetusalgoritmi

Opetusalgoritmi voi muuttaa minkä tahansa painokertoimen w_{ij} positiivisesta negatiiviseksi tai päinvastoin riippumatta siitä mihin edellisen kerroksen neuroniin painokerroin liittyy. Aivoissa hermosolut ovat aina joko kiihdyttäviä (excitatory) tai vaimentavia (inhibitory) (ns. *Dalen laki*). Tällä rajoituksella ei ole merkitystä keinotekoisille neuroneille ja rajoituksen ottaminen mukaan neuroverkkomalliin vain vähentäisi mallin joustavuutta.

Seuraavassa on esitetty perceptronin opetusalgoritmi pääpiirteissään:

- Valitse satunnainen sisääntulovektori x.
- Laske ulostulot neuroneille $o_i = \sum_j x_j w_{ij}$.
- Laske halutun tuloksen d ja neuroverkon laskeman tuloksen y_i avulla virhetermi $e_i = (d_i - y_i)$.

- Korjaa painoja $w_{ij} = w_{ij} + \alpha \cdot e_i \cdot x_j$, missä $< \alpha \leq 1$

Opetusnopeus $0 \leq \alpha \leq 1$ oli alkuperäisessä ehdotetussa algoritmissa $\alpha = 1$. Samalla tavoin kuin aloituspainokertoimien kohdalla, verkko löytää aina ratkaisun opetusnopeuskertoimen arvosta huolimatta (kunhan se on välillä $[0, 1]$). Vastaava epäherkkyys aloituspainokertoimien ja opetusnopeudelle ei päde yleensä missään muussa neuroverkkomallissa!

2.5.2 Takaisinkytketyt neuroverkot

Ihmisaivojen hermosolut muodostavat takaisinkytketyn verkon. Jonkin tietyn hermosolun aktivoituminen voi edelleen aktivoida muita hermosoluja, jotka lähettävät signaalin ensimmäiseen hermosoluun. Takaisinkytketyssä neuroverkossa (recurrent network) neuronin b ulostulo hetkellä t voi olla neuronin a sisääntulo hetkellä $t + x$. Neuronit a ja b voivat olla samalla tai eri neuronien tasolla ja ne voivat olla sama neuronikin ($a = b$). Takaisinkytketyt neuroverkot voivat olla erityisen käyttökelpoisia aikasarjamallinnuksessa, koska takaisinkytkennät voivat säilyttää edellisten aika-askelien informaatiota ja sitten syöttää se takaisin aiempiin neuronikerroksiin. Tavallisissa neuroverkoissa voidaan käyttää sisääntulokerroksessa kiinteätä ikkunaa aikaisempien aika-askelien tietoihin, esim. t, $t - 1$, $t - 2, \ldots, t - k$. Mitään hetkeä $t - k$ aikaisempaa tietoa ei verkossa voida käyttää yhdessä hetkien t, $t - 1$, $t - 2, \ldots, t - k$ tietojen kanssa. Takaisinkytketyt verkot ovat usein yllättävän kompakteja ja selkeitä. Oletetaan esimerkiksi, että sisääntuloja on M kappaletta ja jokaisesta halutaan käyttää kunakin hetkenä N kappaletta viimeisintä tietoa. Tavallisessa verkossa tarvitaan tällöin $M \cdot N$ sisääntuloa, ja silloin myös neuroverkon painokertoimien kokonaislukumäärästä voi tulla liian suuri. Takaisinkytketyssä verkossa ehkä selvitään vain M sisääntulolla. Neuronien takaisinkytkennät toimivat verkon muistina aiempien hetkien tiedoille.

Elman-verkot ovat esimerkki yksinkertaisista takaisinkytketyistä neuroverkoista. Niissä on kolme tavanomaista neuroverkon kerrosta sekä lisäksi vielä yksi kontekstineuronien (context nodes) kerros. Nämä kontekstineuronit sisältävät piilokerrosten neuronien vanhat arvot. Piilokerroksen neuronien ulostulo riippuu siis nykyisistä neuronin sisääntuloista ja neuronin edellisestä ulostulosta. Kontekstineuronien kerroksia voi tietysti olla enemmänkin. Aika tällaisessa verkossa on määritelty ennemminkin implisiittisesti kuin eksplisiittisesti.

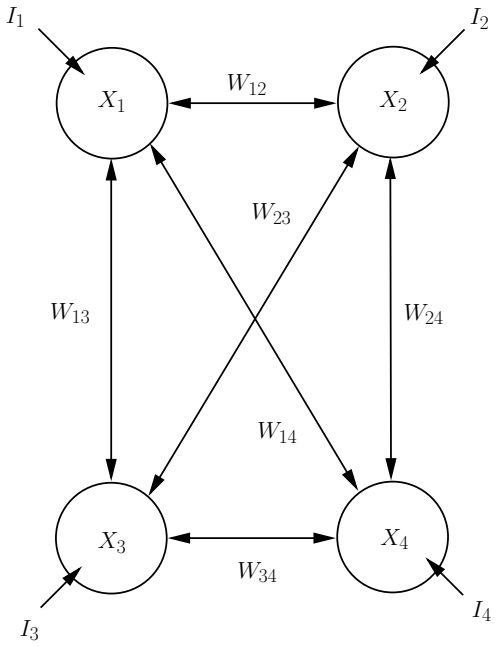
Elman-verkkoja voidaan opettaa monella algoritmilla. Error backpropagation –algoritmin tai jonkin muun gradienttimenetelmän käyttäminen voi olla aikaavievää. Sen vuoksi on kehitelty uusia menetelmiä, jotka perustuvat biologiasta ideansa saaviin algoritmeihin (geneettiset algoritmi, ns. parveilu (particle swarm) -algoritmit, yms.).

2.5.3 Hopfield-neuroverkot ja Boltzmannin kone

Neuroverkkotutkimuksen 1980-luvulla tapahtuneen uuden nousun yhtenä käynnistäjänä oli John J. Hopfieldin assosiaatiomuistia koskeva julkaisu. Assosiaatiomuisti muistuttaa ihmisen muistia, jossa tietyt asiat yhdistyvät ja saavat aikaan mielleyhtymän eli assosiaation. Assosiaatiomuisti esimerkiksi voi tunnistaa

epäselvän kuvan ja löytää oikean alkuperäisen kuvan "muististaan". Hopfieldin tarkastelema verkko pyrittiin saamaan toimimaan assosiaatiomuistina valitsemalla neuronien väliset kytkennät siten, että verkon stabiilit tilat vastaisivat muistiin talletettavia hahmoja. Jos tällaisen verkon alkutilana on esimerkiksi hieman virheellinen tai puutteellinen versio muistiin talletetusta hahmosta ja tämä alkutila on riittävän lähellä talletettua virheetöntä hahmoa, niin verkko dynamiikkansa ohjaamana konvergoi stabiiliin lopputilaan, joka on juuri haluttu virheetön hahmo. Tällainen verkko on toimintaperiaatteeltaan tilansiirtokuvaus.

Hopfield-neuroverkot ovat täysin erilaisia verrattuna MLP-verkkoihin niin toiminnan, rakenteen kuin perusperiaattensakin osalta. Hopfield-verkoissa ei ole neuroneita järjestetty kerroksiin ja verkon painokertoimet eivät muutu. Verkon muodostaa N kappaletta täysin toisiinsa kytkettyä neuronia, kuten oheisessa kuvassa 2.11.



Kuva 2.11: Hopfield-neuroverkon rakenne (N=4).

Hopfield-verkossa painokertoimet w_{ij} ovat kiinteitä ja symmetrisiä ($w_{ij} = w_{ji}$). Jokainen neuroni on jossakin tilassa x_i, joka on rajoitettu välille $[0, 1]$. Painokertoimien sijasta neuronien tilaa päivitetään niin, että verkon energiafunktio minimoituu. Energiafunktion (paikallinen) minimikohta vastaa verkon stabiilia tilaa.

Tällaisen assosiaatiomuistin kapasiteetti on melko rajoitettu. Suurimman mah-

dollisen talletettavien hahmojen lukumäärän arvioidaan olevan arvojen $0,1N$ ja $0,2N$ välissä. Tehokas yläraja talletettavien hahmojen lukumäärälle m on nykytietämyksen mukaan $m \leq 0,5 \cdot n/\log n$, missä n on neuronien lukumäärä. Jos verkkoon talletetaan liikaa hahmoja, alkavat ensin opetetut "unohtua". Toinen ongelma on stabiilien tilojen joukossa olevat muut kuin halutut tilat, ns. harhatilat (spurious states), joihin verkko myös voi konvergoida.

Boltzmannin koneen (Boltzmann machine) perusrakenne on sama kuin symmetrisillä painokertoimilla varustetulla Hopfieldin verkolla. Neuronit ovat nyt kuitenkin stokastisia. Neuroni i saa arvot ± 1 vain tietyillä todennäköisyyksillä. Stokastisuuden lisäksi toinen ero Hopfieldin verkkoon on se, että painot w_{ij} opitaan. Neuronit jaetaan näkyviin ja piiloyksiköihin, ja näistä näkyvät yksiköt voidaan vielä jakaa sisääntulo- ja ulostuloyksiköihin. Opetuksen tavoitteena on määrätä painot w_{ij} siten, että kun verkko on tasapainotilassa, niin näkyvien neuronien hetkittäisillä tiloilla on haluttu todennäköisyysjakauma.

Boltzmannin koneen opetusalgoritmi on laskennallisesti erittäin raskas. Laskennan raskauden vuoksi Boltzmannin koneelle ei ole tehty käytännön sovelluksia.

2.5.4 Itseorganisoituvat verkot SOM

Itseorganisoituva verkko SOM (Self-Organizing Map) on ohjaamattomaan oppimiseen perustuva neuroverkkomalli. Se ryhmittelee dataa siten, että sen kuvausavaruudessa samantyyppiset data-alkiot ovat lähellä toisiaan ja erityyppiset kaukana toisistaan. SOM muuttaa tilastolliset yhteydet aineiston muuttujien välillä yksinkertaisiksi geometrisiksi suhteiksi, joita voidaan havainnollistaa esimerkiksi kaksiulotteisen kartan avulla. Teuvo Kohonen kehitti itseorganisoituvat verkot 1980-luvulla. Siksi niitä kutsutaan englanniksi myös nimellä "Kohonen network".

Itseorganisoituva kartta on yleensä yksi- tai kaksiulotteisen hilan muotoinen neuroverkko, jossa jokainen karttayksikkö edustaa yhtä neuronia. Kartalle jakautuneet neuronit ovat vuorovaikutuksessa lähellään sijaitsevien neuronien kanssa.

SOM-verkko oppii kilpailemalla. Sisääntulo-data tuodaan samanlaisena kaikille neuroneille, ja ulostuloneuroneista vain yksi saa arvon 1, joka on voittaja. Opetusvaiheessa tähän voittajaan ja sen lähellä oleviin neuroneihin liittyviä painokertoimia muutetaan. Kun verkolle tuodaan iso joukko erilaisia sisääntulovektoreita, oppii verkko sijoittamaan toisiaan muistuttavat vektorit vierekkäisiin neuroneihin. Yleensä neuronit siis sijoitetaan kaksiulotteiseen matriisiin, jolloin neuronien voidaan katsoa muodostavan kartan. Kun verkolle oppimisvaiheen jälkeen tuodaan uusia sisääntulovektoreita, aktivoi jokainen vektori vain yhden ulostuloneuronin, ns. voittajaneuronin. Toisiaan muistuttavat sisääntulot aktivoivat lähellä toisiaan kartalla sijaitsevia neuroneita, jolloin sanotaan samanlaisten piirteiden sijoittuvan kartalla samoille alueille. Verkko on tällä tavalla muodostanut piirrekartan. SOM-verkot yleensä erottelevat eri havainnot sitä paremmin, mitä enemmän verkossa on neuroneita suhteessa opittaviin havaintoihin. Tässäkin tulee taas esille kysymys sopivasta tasapainosta neuronien lukumäärän ja verkon yleistyskyvyn välillä, mutta SOM-verkoissa ylioppiminen ei ole ongelma.

SOM-verkon käyttötavat voidaan jakaa käytännössä kahteen erilaiseen. Ensimmäisessä verkon mallivektorien eli neuronien lukumäärä k on aika alhainen (ns. k cluster SOM). Verkossa on siis tavoitteena jakaa havainnot k ryhmään. Tätä tapaa käyttää SOM-verkkoja on kritisoitu, koska tällöin SOM-verkko hävittää joukon hyviä ominaisuuksiaan ja ei enää eroa paljon muista tilastollisista luokittelualgoritmeista. Ilmeisesti juuri tämän vuoksi itseorganisatioverkkojen kykyä järjestää dataa vähätellään kirjallisuudessa. Usein verkossa on vain muutamia tai korkeintaan kymmeniä neuroneita ja verkon tavoite on luokitella ne oikein noille neuroneille. Silloin itseorganisaatioverkko toimii suurin piirtein samalla tehokkuudella kuin muut tilastotieteelliset luokittelualgoritmit, kuten "k-Means". Muutama neuroni on helppo nimetä ja silloin havainto, joka kuvautuu ko. neuronille, kuuluu kyseiseen neuronin luokkaan.

Mielenkiintoista itseorganisaatioverkkojen käytöstä tulee, kun neuroneita on vähintään tuhansia. Tämä toinen tapa käyttää SOM-verkkoja sopii paremmin moniulotteisen havaintoaineiston visualisointiin. Moniulotteinen aineisto on usein harvaa ja siinä on paljon binaarisia muuttujia. Se, että havaintoaineistoa ei voida jakaa lineaarisesti erottuviin luokkiin ei haittaa toisin kuin muilla menetelmillä. SOM-verkon avulla havainnot voidaan jakaa intuitiivisesti ymmärrettäviin ryhmiin ja visualisoida tuloksia. Tässä tapauksessa verkossa on hyvä olla mallivektoreita tuhansia. Lisäksi mallivektoreita pitäisi olla eri määrä riveillä kuin sarakkeissa. Visualisointiin voidaan käyttää esimerkiksi U-matriisia, josta vähän lisää Neuroverkkolaskennan tulosten arviointia –luvussa.

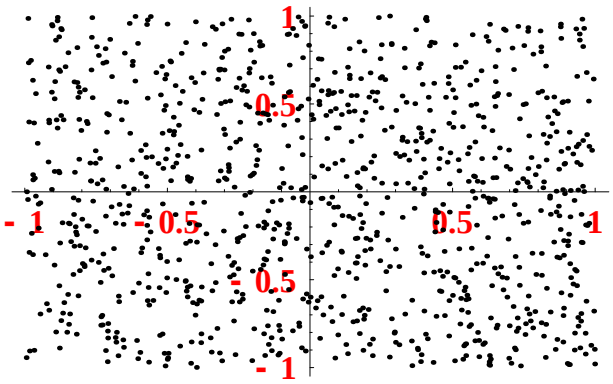
Neuroneita voi olla jopa enemmän kuin havaintoja opetusaineistossa. Opetuksen jälkeen verkon neuronien kertoimet eivät kerro suoraan mitään, mutta kun käytetään verkon visualisointikeinoja kuten U-matriiseja, voidaan erilaisia havaintojen luokkia määritellä intuitiivisesti. Eroja tilastollisiin luokittelualgortimeihin on myös, että SOM-verkossa jossa on paljon neuroneita ei saatavien luokkien lukumäärää ole kiinnitetty: luokkien lukumäärä selviää kuin itsestään, kun tuloksia visualisoidaan. Lisäksi SOM-verkot pystyvät luokittelemaan myös ei-lineaarisesti erottuvia ryhmiä.

Esimerkki SOM-verkon toiminnasta

Itseorganisaatioverkot oppivat havaintoaineiston tilastollisia piirteitä. Opitut painokertoimet approksimoivat sisääntulovektoreiden tiheysfunktioita. Ihan tarkasti näin ei kuitenkaan ole, mutta käytännössä se pätee. Tätä SOM-verkon ominaisuutta on havainnollistettu seuraavassa yksinkertaisessa esimerkissä.

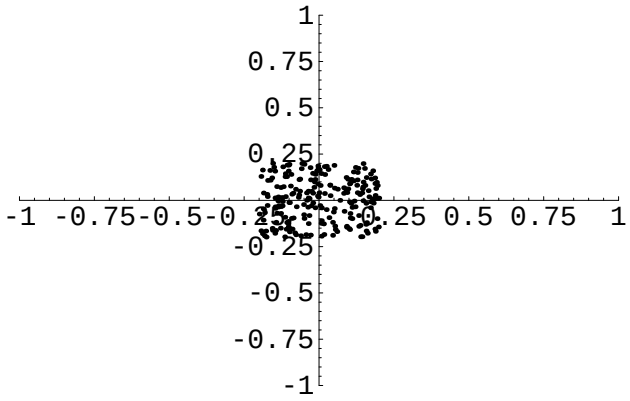
Esimerkin havaintodata on 2-ulotteisesta tasaisesta jakaumasta. Kuvassa 2.12 on havaintodatan 1000 ensimmäistä havaintoa. Kaikkiaan parin x, y havaintoja on simuloitu esimerkkiä varten 25 000 kappaletta.

Joone-ohjelmassa muodostetaan sitten itseorganisaatioverkko, jossa ulostulokerroksessa on $15 \cdot 15$ eli yhteensä 225 neuronia. Jokaiseen neuroniin liittyy kaksi painokerrointa, koska havainnotkin ovat 2-ulotteisia vektoreita. Näin ollen painokertoimia on 450 kappaletta. Nyrkkisääntönä havaintoja näin suurelle verkolle tarvittaisiin 12 500 kappaletta. Havaintoaineistoa on kaksikertainen määrä. Seuraavista kuvista nähdään miten painokertoimet alkavat (eksponentiaalisesti) konvergoitua lopullisiin arvoihinsa. Ensimmäisessä kuvassa on painokertoimet alussa, jolloin niiden arvot ovat pieniä satunnaislukuja. Seuraavissa kuvissa

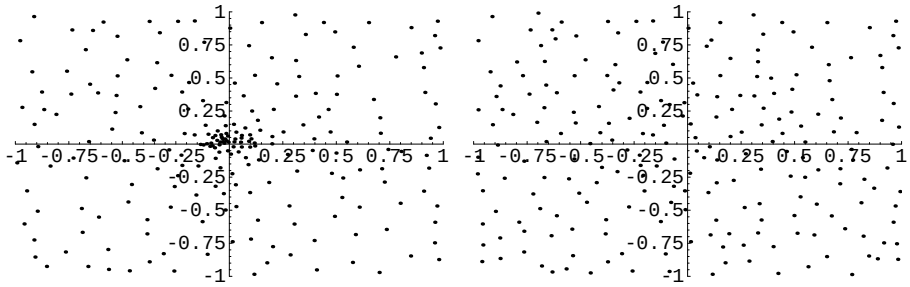


Kuva 2.12: Esimerkin 1000 havaintoa.

on esitetty painokertoimet aina kun havaintoaineisto on käyty 1, 2, 3 tai 10 kertaa lävitse.

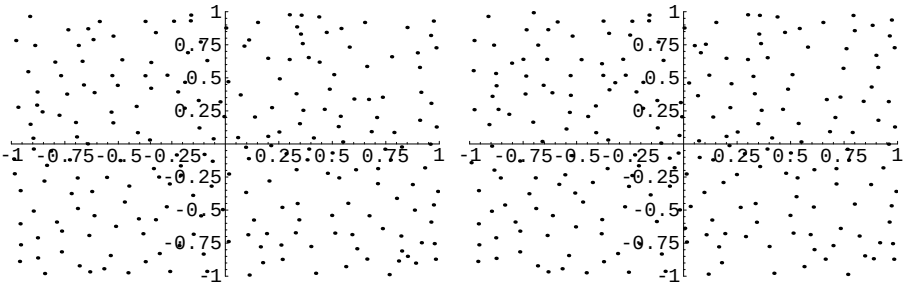


Kuva 2.13: Satunnaiset painokerroinparit alussa 225 kpl.



Kuva 2.14: Painokertoimet 1 ja 2 opetuskierroksen jälkeen.

Kun koko havaintoaineisto on käyty kaksi kertaa lävitse ovat painokertoimien arvot hakeutuneet jo aika tasaisesti koko alueelle. Eli valmiin SOM-verkon neuronit ovat jakautuneet samalla tavalla tasaisesti kuin havainnotkin opetusaineis-



Kuva 2.15: Painokertoimet 3 ja 10 opetuskierroksen jälkeen.

tossa. Liitteessä työn lopussa on kuva Joone-ohjelmassa muodostetusta SOM-verkon kaaviosta. Verkon muodostaminen ohjelmalla tapahtuu lähinnä vetämällä ohjelman valikosta erilaisia osasia ja yhdistämällä niitä. Suurenkin verkon pystyy muodostamaan helposti.

SOM-verkon opetusalgoritmi

Seuraavassa SOM-verkon perusopetusmenetelmä esitetään tarkemmin. Voittajaneuronin löytämiseksi on SOM:n opetuksessa käytetty useita eri etäisyyden mittareita. Tässä käytetään euklidista etäisyyttä. Valitaan seuraavat merkinnät:

- Opetusesimerkkiin i liittyvän piirteen j arvo on x_j^i.
- Neuronilla m_k on jokaista piirrettä j kohti opittava painokerroin w_j^k.
- Opetusesimerkkien lukumäärä on N ja piirteiden lukumäärä eli opetusvektoreiden pituus on P.

Algoritmi toimii seuraavasti

- Alustetaan painokertoimet w_j^k satunnaisluvuiksi.
- Lasketaan kaikkien opetusvektoreiden i etäisyydet kaikista SOM-verkon neuroneista m_k. Siis $y_i = \sqrt{\sum_{j=1}^{P}(x_j^i - w_j^k)^2}$.
- Määritetään voittajaneuronit. Jokaiselle opetusvektorille i etsitään voittajaneuroni m_w, joka on se jonka etäisyys ko. opetusvektorista oli pienin. Voittajia on siis yhtä monta kuin oli opetusvektoreitakin. Painokertoimia w_j^k muutetaan määrittämällä ensin kunkin voittajaneuronin ympäristöstä naapuristoalue. Naapuristoalueeseen kuuluvat kaikki voittajaneuronin lähellä olevat muut neuronit. Kaikkien naapuristoalueen neuronien painokertoimia muutetaan. Alussa naapuriston koko on pienempi kuin neuronien lukumäärä, mutta suurempi kuin puolet siitä. Opetuksen edetessä naapuristoalueen säde pienenee.

  Voittaja- ja naapuristoalueeen neuronien painokertoimia muutetaan kaavalla

$$w_j^k(t+1) = w_j^k(t) + \alpha \cdot (x_j^i - w_j^k(t)),$$

missä w_j^k on neuroniin k liittyvä piirteen j painokerroin, x_j^i on opetusvektorin i piirteen j arvo, α on oppimisnopeuskerroin ja t on kierrosten järjestysluku.

Kerroin α on aluksi lähellä arvoa 1 ja pienenee sitten kohti nollaa opetuksen kestäessä. Samalla oppimisen edetessä naapuriston kokoa pienennetään siten, että pienimmillään muutetaan enää vain voittajaneuronin painokertoimia.

- Jollei opetuksen lopetuskriteerit täyty, pienennetään naapuriston kokoa ja suoritetaan edelliset toimenpiteet uudestaan. Opetus lopetetaan, jos opetuskierroksia on tehty etukäteen valittu maksimimäärä tai keskimääräinen virhe ei enää pienene.

2.5.5 Muita neuroverkkomalleja

Muista neuroverkkomalleista kannattaa mainita seuraavat

- Oppiva vektorikvantisaatio LQV, joka on SOM-verkon lisäksi toinen merkittävä Kohosen esittämä neuroverkkomalli.
- Radiaaliset kantafunktiot (Radial Basis Function RBF), joiden verkolla on yleensä yksisuuntaisen, 3-kerroksisen neuroverkon rakenne. RBF:stä on tullut viime aikoina suuren kiinnostuksen kohde.
- Tukivektorikone (Support Vector Machine SVM), joka on 1990-luvulla kehitetty lineaarinen luokitin. Se soveltuu erityisesti luokitteluun ja käyränsovitustehtävään. Tukivektorikoneella on usein parempi yleistyskyky luokittelutehtävissä kuin perinteisellä MLP-verkolla.

2.6 Lyhyesti hybridimenetelmistä

Käytännön sovellusten kannalta yhdellä ns. soft computing –menetelmällä (neuroverkot, sumeat joukot, geneettiset algoritmit, parveilualgoritmit, jne) ei usein saada riittävän hyviä tuloksia, mutta erilaisilla menetelmien yhdistelmillä tuloksia voidaan parantaa. Nämä soft computing –menetelmät ovat usein saaneet inspiraationsa biologisista prosesseista ja luonnon ongelmanratkaisumenetelmistä (vrt. muurahaiset, lintuparvet, geenit, jne). Neuroverkkoja on yhdistetty ainakin seuraavien muiden menetelmien kanssa:

- Sumea laskenta → FNN eli Neurosumea laskenta
- Geneettiset algoritmit → GANN eli Neuro-geneettinen laskenta

Sumeat luvut ja laskenta liittyvät havaintoaineiston arvojen ja neuroverkon painokertoimien esittämiseen. Geneettisten algoritmien avulla neuroverkko-ehdotuksia voidaan yhdistellä, jolloin uusi neuroverkkosukupolvi on aina jotenkin parempi kuin vanha. Muut mainitut menetelmät ovat lähinnä erilaisia optimointimenetelmiä, joita käytetään painokertoimien opettamisessa.

2.6.1 Neurosumea laskenta

Esimerkiksi sumean laskennan ja neuroverkkojen yhdistäminen eli ns. neurosumea laskenta on tullut viimeisen kymmenen vuoden aikana suosituksi. Oheinen neurosumean laskennan lyhyt kuvaus on luentomonisteesta [5]. Aihepiiri on pirstoutunut moniin erikoistapauksiin ja sovellusalueisiin. Myös yhtenäinen matemaattinen perusta ei ole vielä vakiintunut. Yleensä ottaen neurosumeilla systeemeillä voidaan tarkoittaa seuraavia eri asioita:

- Sumeita menetelmiä käytetään neuroverkon oppimiskyvyn parantamiseen. Sumeita sääntöjä voidaan käyttää oppimisnopeuden parantamiseen tai verkko toimii sumeilla syötetiedoilla.
- Neuroverkko ja sumea järjestelmä toimivat saman tehtävän parissa, mutta eivät vaikuta toistensa toimintaan. Tavallisesti neuroverkko toimii datan esi- tai jälkikäsittelijänä sumeassa systeemissä.
- Neuroverkkoa käytetään sumean systeemin parametrien viritykseen, kuten sumeiden sääntöjen tai sumeiden joukkojen generoimiseen annetusta datasta. Virityksen jälkeen sumea systeemi toimii ilman neuroverkkoa.
- Neuroverkko ja sumea järjestelmä yhdistetään neurosumeaksi systeemiksi. Tämä systeemi voidaan tulkita joko neuroverkoksi, jonka parametrit ovat sumeita, tai rinnakkaislaskentaa varten tehdyksi sumeaksi systeemiksi.

Neurosumeiden verkkojen ajatuksena on sumentaa neuroverkon parametrit. Se voidaan voidaan tehdä jollakin seuraavista kolmesta tavasta.

- Verkossa on reaalinen sisääntulo ja painokertoimet ovat sumeat.
- Verkossa on sumennettu sisääntulo ja painokertoimet ovat reaaliset.
- Sekä sisääntulot että painokertoimet ovat sumeita.

2.6.2 Neuro-geneettinen laskenta

Neuroverkot voivat oppia monia eilaisia laskenta-, luokittelu- yms tehtäviä opetusaineistosta. Oikeanlaisen aineiston lisäksi ongelmana on kuitenkin ongelmalle sopivan neuroverkkorakenteen löytäminen. Erilaisia neuroverkkoja on suuri määrä ja usein neuroverkon suunnittelu riippuu paljon sovelluksesta mihin sitä aiotaan käyttää. Mitään yleistä, kaikkiin tehtäviin sopivaa neuroverkkoa ei ole nyt olemassa. On olemassa vain yleisiä ohjeita ja kokemuksia siitä, mihin mikäkin neuroverkko soveltuu hyvin.

Neuroverkkoja ja geneettisiä algoritmeja on yhdistetty, jotta saataisiin jokin laskennallinen tapa optimoida neuroverkon parametreja kuten piilokerroksen neuronien lukumääriä tai opetuskertoimia. Neuro-geneettisellä laskennalla on raportoitu saataneen vastaavanlaisia tuloksia kuin perinteisillä menetelmillä.

Neuro-geneettisissä algoritmeissa neuroverkon parametrit koodataan lukujen vektoreiksi, joita kutsutaan kromosomeiksi. Algoritmin alussa muodostetaan

suuri joukko kromosomeja eli erilaisia neuroverkon parametrien vektoreita. Kromosomeista valitaan jatkoon ne, jotka tuottavat parhaimmat neuroverkot. Geneettisissä algoritmeissa kromosomien eli ratkaisujen joukkoa yritetään parantaa myös mutaatioiden ja rekombinaatioiden avulla. Neuro-geneettiset algoritmit löytävät yleensä nopeasti lähelle optimaalista ratkaisua, mutta lopullinen konvergoituminen siihen kestää kauan. Käytännön kannalta ongelma on myös, että laskenta vaatii paljon muistia ja laskentatehoa. Lisäksi myös neurogeneettisessä algoritmissa alkuparametrien kuten mutaatiotahdin ja kromosomien hyvyysarvio-funktion määrittelyllä on suuri merkitys algoritmin toiminnalle. Kirjassa [9] on esitetty erilaisia neuroverkkojen ja sumean laskennan ja/tai geneettisen algoritmien yhdistelmiä esimerkkien kanssa.

2.7 Neuroverkon opetus

Neuroverkkojen huomattavin piirre on niiden kyky oppia opetusaineiston avulla. Edellä esitettiin, että tiettyjen neuroverkkojen on todistettu olevan universaaleja funktioapproksimaattoreita. Opetusalgoritmit mahdollistavat halutun funktion etsinnän pelkästään opetusaineiston perusteella. Tämä tuo huomattavan helpotuksen epälineaaristen mallien valintaan ja estimointiin. Neuroverkkojen opetus ei kuitenkaan aina ole suoraviivaista, eikä hyviä tuloksia useinkaan saada ensimmäisellä yrityksellä. Samoin neuroverkkomallien valintaan ei ole olemassa yleispätevää menetelmää.

Neuroverkkojen opetus voidaan jakaa ohjattuun opetukseen ja ohjaamattomaan opetukseen. Ohjatussa opetuksessa mallinnettava data jaetaan yleensä opetusaineistoon ja testiaineistoon sekä usein vielä kolmanteen, validointiaineistoon. Opetusaineisto koostuu neuroverkolle esitettävistä vektoripareista, jotka sisältävät sisääntulovektorit ja niitä vastaavat halutut ulostulovektorit

$$\{(\mathbf{x_1}, \mathbf{d_1}), \ldots, (\mathbf{x_N}, \mathbf{d_N})\}.$$

Neuroverkolla pyritään approksimoimaan näiden vektorien muodostamaa kuvausta. Opetusalgoritmi esittää opetusaineiston vektoreita neuroverkolle ja lukee neuroverkon ulostulon y. Annetulle sisääntulovektorille x saadaan sitä vastaava virhe verkon ulostulon ja halutun ulostulon erotuksena

$$\mathbf{e} = \mathbf{y} - \mathbf{d}.$$

Koko opetusaineistolle saatavana virhemittana voidaan käyttää virheiden neliösummaa

$$E = \sum_{p=1}^{N} \mathbf{e}_p^T \mathbf{e}_p.$$

Opetuksen tavoite on löytää neuroverkon painokertoimet $\mathbf{w}$, jotka minimoivat virheen E verkon rakenteen asettamissa rajoissa. Neuroverkon opetus on näin ollen optimointiongelma. Jos verkossa on n kpl painokertoimia, virhepinta (*error surface*) on $n+1$-dimensioisessa avaruudessa. Opetusalgoritmit helposti takertuvat paikallisiin minimikohtiin, mutta on olemassa monia keinoja yrittää

välttää sitä. Tavallisin keino on käyttää opetusnopeus-kerrointa, joka opetuksen alussa on iso ja sitten vähitellen pienenee. Valitettavasti parhaan opetusnopeuden valinnalle ei ole yleissääntöä. Opetuksen lopputulos riippuu usein myös paljon painokertoimien arvoista alussa, ts. mistä kohtaa virhefunktion pintaa aloitetaan optimipainojen etsintä. Opetuksessa käytetään usein myös opetuksen momenttikerrointa, jolla aikaaansaadaan se että korjaus painokertoimiin ja optimiarvojen etsintäsuunta riippuu tietyssä määrin edellisistä painokertoimien korjauksista $\Delta w_{ij}(t)$:

$$\Delta w_{ij}(t+1) = \lambda \Delta w_{ij}(t) - (1-\lambda)\alpha(\frac{\partial E}{\partial w_{ij}(t)}),$$

missä λ on momenttikerroin ja α on opetusnopeus. Kun monien painokertoimien muutoksen merkki on sama, voidaan momenttikerrointa suurentaa. Kun etsintä tulee alueelle, jolla painokertoimien muutokset ovat erimerkkisiä, kerrointa pienennetään. Näin opetusprosessi nopeutuu alueilla, jolla virhefunktio muuttuu tasaisesti.

Backpropagation-algoritmia voi verrata hyppykepillä hyppelyyn virhepinnalla: jos hypyt ovat liian suuria, voidaan optimaalisen kohdan yli hyppiä monta kertaa ja oppiminen on hidasta. Jos hypyt ovat liian lyhyitä, oppiminen on hidasta ja voi helposti juuttua johonkin paikalliseen minimikohtaan. Painokertoimien alkuarvoilla on myös suuri merkitys oppimisen kannalta. Siksi opetus pitäisi suorittaa useampia kertoja erilaisilla painokertoimien alkuarvoilla.

Menetelmiä ja algoritmeja neuroverkon parametrien optimointiin on esitetty useita. Itse asiassa niitä esitetään joka vuosi monta uutta. Selvästi parasta, monenlaisiin sovelluksiin sopivaa menetelmää ei vielä ole välttämättä löydetty. MLP-verkon opetukseen käytetään yleisimmin opetusalgoritmeja, jotka perustuvat painokertoimien suhteen lasketun virheen gradientin estimointiin. Verkon painokertoimia korjataan tällöin määrällä

$$\Delta \mathbf{w} = -\alpha \hat{\nabla},$$

missä $\hat{\nabla}$ on hetkellinen virheen gradientti vastaten senhetkistä sisääntulovektoria ja α on opetuskerroin. Merkintä $\hat{\nabla}$ on opetusalgoritmin laskema estimaatti kokonaisvirheen gradientille painokertoimien suhteen

$$\nabla = \frac{\partial E}{\partial \mathbf{w}} = \sum_{p=1}^{N} \frac{\partial \mathbf{e}_p^T \mathbf{e}_p}{\partial \mathbf{w}}.$$

Opetusta jatketaan läpikäyden opetusaineistoa ja korjaten painokertoimia kunnes kokonaisvirhe on riittävän pieni tai kunnes se ei enää pienene. Tämän jälkeen neuroverkon mallinnuskyky yleensä testataan testiaineistolla, jota ei ole käytetty verkon opetuksessa.

Neuroverkkojen epälineaarisilla muutosfunktioilla on suuri merkitys verkon tehokkuudelle mallintaa muuttujien välisiä yhteyksiä. Huono puoli epälineaarisissa funktioissa on se, että neuroverkkojen estimoitaville parametrien ($\mathbf{w}$) optimaalisille arvoille ei voi löytää analyyttista ratkaisua.

Neuroverkon liitoksien painokertoimet ja kynnysarvot toimivat verkon muistina. Neuroverkon ohjatussa opetuksessa ne pyritään määrittelemään niin, että

neuroverkon antama tulos olisi mahdollisimman lähellä opetusaineiston tiedettyä "oikeaa" vastetta. Opetusaineiston jokainen havainto muodostuu kahdesta osasta. Ensimmäisessä osassa on selittävien muuttujien saamat arvot $\mathbf{x}$ ja toisessa osassa on havaittu tulos $\mathbf{y}$. Vakuutusesimerkissä yksittäinen havainto voisi muodostua vakuutustiedoista (kaikki oleelliset riskitekijät) $\mathbf{x}$) ja ko. vakuutukselle sattuneesta vahinkomäärästä. Toinen tapa verkon opetuksessa on itseohjautuva oppiminen. Siinä havainnoissa ei ole tiettyä tulosta, joka verkon pitäisi pystyä ennustamaan vaan verkko oppii syöttötietojen perusteella esimerkiksi luokittelemaan havainnot sisäisesti yhtenäisiin ryhmiin.

Molemmissa tapauksissa verkon parametrit (painokertoimet ja kynnysarvot) muokkautuvat opetusaineiston perusteella. Opetus voi tapahtua asteittain (incremental training), jolloin painoja ja kynnysarvoja muokataan joka kerta, kun uusi opetusesimerkki syötetään verkolle tai ryhmissä (batch training), jolloin parametreja muokataan vasta kun kaikki opetusesimerkit on käyty läpi.

Ennen kuin verkkoa voidaan alkaa opettaa on verkon eri kertoimille annettava jotkin lähtöarvot. Arvot asetetaan yleensä pieniksi satunnaisiksi luvuiksi. On olemassa myös teoria, joka antaa suuntaviivoja sille millainen satunnaisarvojen jakauman olisi hyvä olla.

Esimerkki yksinkertaisesta verkon opetusalgoritmista on Delta-sääntö, jossa kerrotaan verkon opetusnopeuskerroin, virhesignaali ja liitoksen painokerroin. Verkon opetusnopeuskertoimen valinta on tärkeää. Jos kerroin on liian suuri, virhettä ei ehkä saada pienenemään ollenkaan. Jos kerroin on liian pieni, verkko voi oppia erittäin hitaasti tai se ehkä pysähtyy johonkin paikalliseen optimiin. Verkon opetuksessa ensin lasketaan neuroverkon tulos $y(\mathbf{x})$ annetulle havainnolle. Tuloksen ja tiedetyn havaitun tuloksen avulla lasketaan virhe $\mathbf{e} = y(\mathbf{x}) - \mathbf{d}$. Yleensä minimoitavana kriteerina käytetään virheen neliötä eli $q = \mathbf{e}^2 = (y(\mathbf{x}) - \mathbf{d})^2$. Seuraavaksi kaikille verkon parametreille ($\mathbf{w}$) lasketaan differentiaalilaskennan ketjusäännöllä derivaatat kriteerin q suhteen. Näin saatuja derivaattojen arvoja käytetään sitten muokkaamaan kunkin parametrin arvoa sillä tavalla, että jos sama havainto syötettäisiin verkolle uudestaan olisi neuroverkon antama tulos lähempänä oikeaa arvoa kuin aikaisemmin. Tämä prosessi tehdään jokaiselle opetusaineiston havainnolle. Yleensä koko opetusaineisto käydään useampi kerta lävitse. Opetusta jatketaan kunnes verkon antamat tulokset ovat riittävän lähellä opetusaineiston oikeita arvoja. Erityisesti vakuutuksissa neuroverkkojen opetusaineistot voivat olla todella suuria, esimerkiksi miljoonia rivejä. Sellaisen aineiston läpikäyminen havainto havainnolta ja yleensä vielä useamman kerran, vaatii paljon laskenta-aikaa, helposti muutamankin tunnin. Toisaalta neuroverkon opetus tarvitsee tehdä vain kerran.

Neuroverkon ylioppimisella tarkoitetaan tilannetta, jossa verkko oppii opetusaineiston erittäin hyvin, mutta ei pysty yleistämään tuloksia testiaineistoon. Syynä voi olla liian suuri verkko tai liian monta opetuskierrosta. Suuresta opetusaineiston määrästä on aina hyötyä, koska silloin suurempi osa aineistosta voidaan säästää verkon testausta tai validointia varten.

Neuroverkon opetukseen liittyy monia asioita, joihin teoria ei useinkaan anna selvää ohjetta, kuten piilokerroksen neuronien lukumäärä, opetusnopeus, kertoimien alkuarvot ja sigmoid-funktion jyrkkyys. Niille täytyy yleensä löytää sopivimmat arvot yrityksen ja erehdyksen kautta. Algoritmi voi löytää globaalin

minimin sijasta myös paikallisen minimin. Myös siksi erilaisia neuroverkkorakenteita ja erilaisia lähtöarvoja on kokeiltava.

2.7.1 Error backpropagation –opetusalgoritmi

Monikerroksisten neuroverkkojen opettamiseen käytetään usein error backpropagation –algoritmia. Sen esittelivät ensimmäisinä Rumelhart yhdessä Hintonin ja Williamsin kanssa sekä McClelland vuonna 1986. Siihen asti oli pidetty mahdottomana opettaa monikerroksisia verkkoja. Jälkeenpäin havaittiin, että Parker (1982) ja Werbar (1974) olivat esittäneet jo aikaisemmin saman algoritmin. Algoritmi koostuu kahdesta vaiheesta: etenevä kierros (forward pass) ja palaava kierros (backward pass) koko verkon läpi. Etenevässä kierroksessa sisääntulo kulkeutuu neuronien kerros kerrokselta ulostuloneuroniin saakka, josta saadaan verkon lopullinen ulostulo. Palaavan kierroksen aikana muodostetaan virhesignaali, joka saadaan vähentämällä verkon antamasta tuloksesta haluttu vastaus. Tätä virhesignaalia kuljetetaan kerros kerrokselta takaisinpäin verkossa muuttaen samalla verkon painoja siten, että verkon ulostulo lähenee haluttua.

Tarkastellaan ensin yhtä myötäkytketyn neuroverkon neuronia (joko piilo- tai ulostulokerroksessa), jonka aktivaatiofunktio on γ. Neuroni on esitelty kuvassa 2.16. Neuronin ulostuloa merkitään symbolilla h_j ja neuronin kokonaissisääntuloa net_j. Määritelmän mukaan

$$h_j \equiv \gamma(net_j), \tag{2.8}$$

$$\text{missä } net_j \equiv \sum_k h_k w_{kj}. \tag{2.9}$$

Summa net_j on laskettu kaikista neuronia j edeltävien neuronien ulostuloista. Olkoon neuroni i yksi niistä.

Derivaattaoperaattorin ketjutussäännön mukaan on

$$\frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial net_j} \frac{\partial net_j}{\partial w_{ij}}.$$

Koska mikään termeistä summassa (2.8), joilla $k \neq i$, ei riipu w_{ij}:sta, saadaan

$$\frac{\partial net_j}{\partial w_{ij}} = h_i.$$

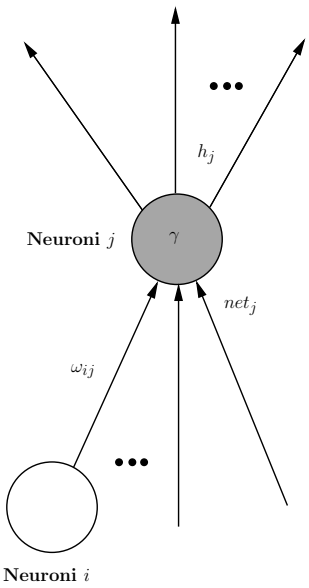
Määritellään vielä

$$\partial_j = \frac{\partial E}{\partial net_j},$$

jolloin saadaan derivaatalle esitysmuoto

$$\frac{\partial E}{\partial w_{ij}} = \partial_j \cdot h_i. \tag{2.10}$$

Pian nähdään, että yllä oleva kaava on backpropagation-algoritmissa tärkeässä asemassa. Arvot ∂_j voidaan laskea rekursiivisesti neuroverkon ulostuloista takaisinpäin sisääntuloja kohti.



Kuva 2.16: Neuroverkon yksi neuroni.

"Käytännössä" backpropagation-algoritmissa muodostetaan aktivaatiofunktion avulla gradientti verkon painokerroinavaruuteen ja painoja muutetaan gradientin vastakkaiseen suuntaan. Täten aktivaatiofunktion on oltava jatkuva ja kaikkialla derivoituva. Mikäli ulostulokerroksessa käytetään sigmoidfunktioita aktivaatiofunktioina, verkon ulostulot rajoittuvat pienelle alueelle. Jos ulostulokerroksessa käytetään lineaarisia aktivaatiofunktioita, verkon ulostulot voivat saada mitä tahansa reaalilukuarvoja.

Backpropagation-algoritmi perustuu siis virheen gradientin estimointiin ja neuroverkon parametrien korjaamiseen sen avulla. Algoritmin nimi johtuu sen tavasta syöttää virhetermejä taaksepäin neuroverkon ulostulosta sisääntuloa kohti. Algoritmi on varsin yleinen ja soveltuu monien erilaisten neuroverkkojen opetukseen.

Johdetaan seuraavaksi kaavat derivaatoille ∂_k (ulostuloneuroni) ja ∂_j (piilokerroksen neuroni). Tarkastellaan neuroverkon pientä osaa, jota kuvataan kuvassa 2.17.

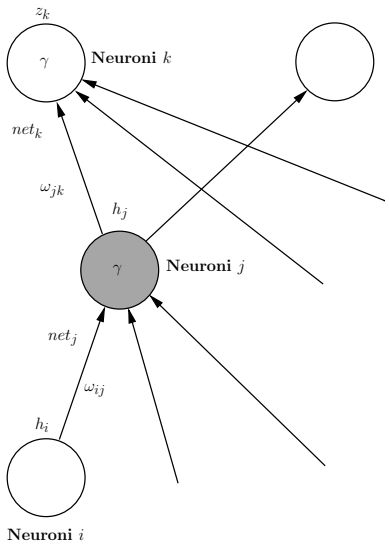
Kun verkolle esitetään yksi opetusaineiston havainto, on virhe

$$E = \frac{1}{2}\sum_{l=1}^{m}(y_l - d_l)^2,$$

missä l on ulostulojen lukumäärä.

Voidaan kirjoittaa

$$\partial_k = \frac{\partial E}{\partial net_k} = (\frac{\partial E}{\partial y_k})(\frac{\partial y_k}{\partial net_k}). \tag{2.11}$$



Kuva 2.17: Pieni osa neuroverkosta.

Koska $y_k = \gamma(net_k)$, on

$$\frac{\partial y_k}{\partial net_k} = \gamma^{'}(net_k). \tag{2.12}$$

Edelleen virheen E lausekkeesta saadaan $\frac{\partial E}{\partial y_k} = (y_k - d_k)$. Tästä yhtälöstä ja yhtälöistä (2.11), (2.12) ja (2.10) saadaan

$$\partial_k = (y_k - d_k)\gamma^{'}(net_k) \text{ ja} \tag{2.13}$$

$$\frac{\partial E}{\partial \omega_{jk}} = \partial_k h_j. \tag{2.14}$$

Kaavat pätevät kaikille painoille, jotka liittyvät ulostulokerrokseen. Termit h_j ovat ulostulokerroksen neuroniin tuleva sisääntulo piilokerroksen neuronista, kynnysparametrin arvo $(= 1)$ tai se voi olla myös neuroverkon sisääntulo x_j.

Etsitään seuraavaksi kaava derivaatalle

$$\partial_j \equiv \frac{\partial E}{\partial net_j} = \sum_l (\frac{\partial E}{\partial net_l})(\frac{\partial net_l}{\partial net_j}). \tag{2.15}$$

Summassa (2.15) indeksi l käy läpi kaikki neuronin j välittömät edeltäjät. Näin ollen,

$$\partial_j = \sum_l \partial_l (\frac{\partial net_l}{\partial net_j}). \tag{2.16}$$

Määritelmän mukaan on

$$net_l = \sum_s \omega_{sl}\gamma(net_s).$$

Tästä määritelmästä nähdään, että

$$\frac{\partial net_l}{\partial net_j} = \omega_{jl}\gamma^{'}(net_j) \tag{2.17}$$

Yhdistämällä tulokset (2.16) ja (2.17) saadaan

$$\partial_j = \sum_l \partial_l \omega_{jl}\gamma^{'}(net_j) \tag{2.18}$$

$$\partial_j = (\sum_l \partial_l \omega_{jl})\gamma^{'}(net_j) \tag{2.19}$$

$$\frac{\partial E}{\partial \omega_{ij}} = \partial_j h_i. \tag{2.20}$$

Keskimmäisessä yhtälössä ∂_j lasketaan käyttämällä arvoja ∂_l, eli arvoja neuronia j välittömästi seuraavista neuroneista. Toisin sanoen, arvot ∂ syötetään verkossa taaksepäin lähtien ulostulokerroksesta.

Yhteenvetona

- Kaava (2.10) on voimassa *kaikille* neuroverkon painoille.
- Kaavat (2.13) ovat voimassa kaikille ulostulokerroksen neuroneille.
- Kaavat (2.18) ovat voimassa kaikille piilokerroksen neuroneille, kun indeksi l käy läpi kaikki neuronin välittömät seuraajat.

Yllä olevat tulokset on helppo laajentaa koskemaan useampaa opetusaineiston havaintoa kuin vain yhtä. Yksittäisten havaintojen tulokset lasketaan yksinkertaisesti yhteen ja painoja muutetaan sitten.

Backpropagation-algoritmi korjaa verkon painokertoimia joko jokaisen esitetyn sisääntulovektorin jälkeen tai vasta kun koko opetusaineisto on käyty läpi (ns. yksi epookki). Jälkimmäisessä tapauksessa saadaan tarkempi estimaatti kokonaisvirheen gradientille. Alla on esitetty joka sisääntulovektorilla painokertoimia korjaava algoritmi.

1. Alustetaan valitun neuroverkon painokertoimet ja kynnysarvot pienillä satunnaisluvuilla väliltä $[-1, 1]$.
2. Valitaan opetusaineistosta seuraava sisääntulovektori $\mathbf{x}$ ja sitä vastaava haluttu ulostulo $\mathbf{d}$.
3. Lasketaan verkon ulostulo $\mathbf{y}$ ja sitä vastaava virhe $\mathbf{e} = \mathbf{y} - \mathbf{d}$.
4. Lasketaan korjauskertoimet painokertoimille

   $$\Delta w_{ij}^l = -\mu \partial_j^l h_i^{l-1},$$

   missä μ on opetuskerroin ja osittaisderivaatta-termit ∂_j^l lasketaan rekursiivisesti lähtien verkon ulostulokerroksesta ja edeten kohti sisääntuloa. Ulostulokerrokselle L saadaan

   $$\partial_j^L = \frac{\partial \mathbf{e}^T\mathbf{e}}{\partial net_j^L} = \frac{\partial \mathbf{e}^T\mathbf{e}}{\partial y_j} f^{'}(net_j^L) = -2e_j f^{'}(net_j^L),$$

missä $f^{'}$ on aktivaatiofunktion derivaatta ja e_j on virhevektorin e j:s alkio. Vastaavasti verkon kerrokselle l saadaan

$$\partial_j^l = \frac{\partial \mathbf{e}^T \mathbf{e}}{\partial net_j^l} = f^{'}(net_j^l) \sum_k \partial_k^{l+1} w_{jk}^{l+1},$$

missä k käy läpi kaikki neuronit, joihin neuroni j syöttää ulostulonsa.

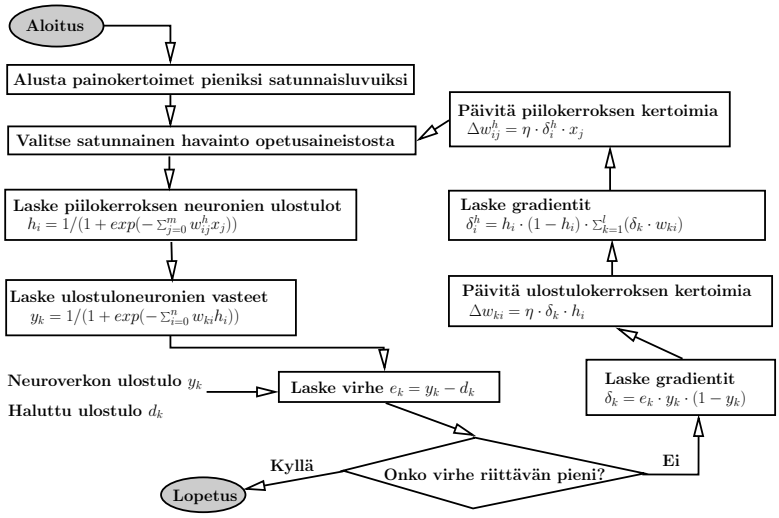
5. Korjataan painokertoimia summaamalla niihin lasketut korjaustermit

$$w_{ij}^l(n+1) = w_{ij}^l(n) + \Delta w_{ij}^l.$$

Painokertoimien korjaus voidaan tehdä joko jokaisen opetusesimerkin jälkeen tai vasta sitten kun kaikki opetusesimerkit on käyty läpi ja korjaustermit on laskettu yhteen yhdeksi korjaustermiksi.

6. Toistetaan kohtia 2-5 kaikilla opetusaineiston sisääntulovektoreilla, kunnes saavutetaan haluttu kokonaisvirhetaso E verkon ulostulossa tai kunnes virhe ei enää pienene riittävästi.

Oheisessa kuvassa 2.18 Backpropagation-algoritmi on vielä esitetty kaaviomuodossa.



Kuva 2.18: Backpropagation-algoritmin vaiheet, kun aktivaatiofunktiona on sigmoidi-funktio.

Neuroverkkojen opetuksessa on huomioitava monia seikkoja. Opetus- ja testiaineiston esikäsittelyt ja valinta, selittävien muuttujien valinta sekä verkon rakenteen määrääminen ovat näistä tärkeimmät. Muita neuroverkkojen opetukseen

liittyviä aihepiirejä ovat mm. opetusalgoritmin parametrien ja neuroverkon parametrien alkuarvojen päättäminen, verkon yleistyskyvyn testaaminen, verkon konvergoituminen paikalliseen tai globaaliin minimiin, virhemittana käytetyn funktion valinta, verkon painokertoimien arvojen rajoittaminen opetuksen aikana (weight decay) ja verkon kytkentöjen karsiminen opetuksen jälkeen (pruning).

Jos MLP-verkkoa opetetaan pitkään backpropagation-algoritmilla, on suuri todennäköisyys sille, että verkko ylioppii ja sen yleistyskyky alkaa opetuksen edetessä vain heikentyä. Tällöin verkko on alkanut oppia/mallintaa aineistossa olevaa kohinaa ja virheitä. Yksi tapa estää tämä on käyttää testiaineistoa. Testiaineisto on erotettu alkuperäisestä opetusaineistosta ja sitä ei käytetä painokertoimien määrittämiseen. Vähän väliä verkon yleistyskykyä testataan testiaineistolla. Kun verkon virhe lopettaa pienenemisen ja alkaa taas nousta testiaineistolla, verkon lopetus keskeytetään. Jos opetusaineistoa on tarpeeksi, siitä voidaan erottaa vielä kolmas osajoukko lopullista testausta varten eli ns. validointiaineisto. Tällä aineistolla voidaan esimerkiksi verrata eri opetuskertojen muodostamia verkkoja keskenään ja valita niistä paras.

2.8 Neuroverkon käyttöön liittyviä näkökohtia

Seuraavassa on lyhyesti kirjoitettu neuroverkon soveltamisen vaiheita ja kysymyksiä, jotka ovat yleisiä ja tulevat aina jossakin muodossa vastaan.

A. Opetusaineisto

Jotta neuroverkkoa voitaisiin käyttää, täytyy ensin kerätä opetusaineisto, siis parit syötteet-vaste. Aineiston täytyy riittävästi kuvata ongelmaa ja sitä täytyy monessa tapauksessa olla paljon. Yleensä opetusaineistoa täytyy lisäksi hieman muokata, jotta neuroverkko pystyy käsittelemään sitä, oppiminen on nopeaa ja tehokasta eikä juutu paikallisiin minimeihin, tai että erilaiset havainnot kuvastavat oikeassa suhteessa niiden esiintymistä. Neuroverkko voi oppia vain sellaiset havainnot, joita opetusaineistossa on, ja interpoloimaan näiden havaintojen väliltä. Toisaalta ylimääräinen tieto, jota neuroverkko ei lopulta tarvitse tai käytä tai jolla ei ole merkitystä ratkaistavan ongelman kannalta, hidastaa oppimista tai voi johtaa prosessin kokonaan väärään suuntaan.

Opetusaineisto on järkevää myös käsitellä etukäteen paremmin neuroverkolle sopivaksi. Kaikkien tietojen pitää ensinnäkin olla koodattuina numeerisiksi. Sisääntulot ja halutut neuroverkon ulostulot on skaalattava kyseiselle neuroverkolle sopivaan vaihteluväliin. Skaalaamisesta voi tulla yllättäviä ongelmia, jos ei ole tarkkana. Skaalaus on tehtävä täsmälleen samalla tavalla sekä opetus-, testi- että validointidatalle. Skaalauksessa pitää käyttää samaa muuttujien vaihteluväliä, johon kaikki aineistot varmasti sopivat. Jos skaalaus tehdään eri tavalla, skaalautuu esimerkiksi arvo 100 erilaiseksi eri aineistoissa. Muutenkin selvästi poikkeavia kannattaa poistaa aineistosta, koska jos niitä on, muut arvot skaalautuvat lähelle nollaa.

B. Neuroverkon rakenteen valinta

Yleensä sisään- ja ulostulot on helppo määrätä eri sovelluksissa. Kokonaan toinen ongelma on piilokerrosten ja piiloneuronien lukumäärien päättäminen. Lu-

kumäärien päättämistä varten ei ole olemassa yleisesti toimivaa menetelmää, vaan kerrosten ja neuronien lukumäärät riippuvat paljon sovellusalasta ja se täytyy usein tehdä yrityksen ja erehdyksen kautta. Erilaiset neuroverkkotyypit toimivat käytännössä usein yhtä tehokkaasti.

Piiloyksiköiden eli neuronien lukumäärän valinta

Neuronien lukumäärä n_h on erittäin tärkeä neuroverkon toiminnan kannalta. Jos n_h on liian pieni, neuroverkossa on liian vähän parametreja eikä se pysty kuvaamaan kaikkia oleellisia muuttujien välisiä yhteyksiä. Tällöin neuroverkon ennustuskykykin on huono. Toisaalta jos n_h on liian iso, neuroverkon opetus ensinnäkin kestää kauemmin ja mikä pahinta, verkko alkaa mallintaa havainnoissa ollutta satunnaista häiriötä. Seurauksena on ylioppiminen ja se, että verkko pystyy hyvin tekemään ennusteet opetusaineiston havainnoille, mutta sen ulkopuolisten havaintojen ennustaminen huononee.

Yleistäen voidaan arvioida, että jokainen neuroverkon painokerroin vie yhden vapausasteen. Esimerkiksi MLP-verkko 20-10-1, jossa kaikki neuronit ovat aina kytketty kaikkiin seuraavan kerroksen neuroneihin tarvitsee $20 \cdot 10 + 10 + 10 \cdot 1 + 1 = 221$ painokerrointa. Yleissääntönä voidaan pitää vaatimusta, että jokaista painokerrointa varten tarvitaan ainakin 5-10 havaintoa. Esimerkin neuroverkon opetusta varten tarvitaan siis suhteellisen suuri opetusaineisto eli noin 1100-2200 riviä. Vakuutusalalla tällainen havaintomäärä on monissa tapauksissa vielä pieni, mutta verkon opetus kuitenkin alkaa olla hidastua aineiston kasvaessa. Toinen, sisääntulojen lukumäärään liittyvä sääntö on: piilokerroksen neuronien lukumäärän on hyvä olla enintään puolet sisääntulojen lukumäärästä.

Joskus painokertoimien lukumäärää pystyy pienentämään ottamalla neuroverkkoon vielä toisen piilokerroksen ilman että verkon teho vähenisi. Esimerkiksi MLP-verkossa 20-5-5-1 tarvittavien painokertoimien lukumäärä putoaa $20 \cdot 5 + 5 + 5 \cdot 5 + 5 + 5 \cdot 1 + 1 = 141$ kappaleeseen. Teoriassa MLP-verkossa kerroksen kaikkien neuronien ei tarvitse olla kytkettynä kaikkiin seuraavaan kerroksen neuroneihin. Lisäksi neuroni voi olla kytkettynä neuroniin, joka on kauempana kuin heti seuraavassa neuronien kerroksessa. Myös tällaisia verkkoja voidaan opettaa normaalilla backpropagation-algoritmilla. Puuttuvat neuronien väliset yhteydet tai suorat yhteydet kauemmaksi tietysti pienentää lopullista painokertoimien lukumäärää. Jokaisessa kokeilemassani ohjelmistossa kaikki kerroksen neuronit on yhdistetty kaikkiin seuraavan kerroksen neuroneihin.

C. Opetusalgoritmin valinta

Backpropagation-algoritmi on paljon käytetty verkon opetusmenetelmä. Se sopii moniin erilaisiin sovelluksiin. Backpropagation-algoritmin kohdalla on valittava opetusnopeus α. Muitakin kehittyneempiä menetelmiä on olemassa, kuten konjugaattigradientti-menetelmä tai Kalman filter -menetelmä. Edellisen vaiheen eli neuroverkon rakenteen valinta vaikuttaa paljon siihen, mitä vaihtoehtoja opetusalgoritmiksi on.

D. Painokertoimien alustus

Neuroverkon painokertoimet yleensä alustetaan pieniksi satunnaisluvuiksi. Jos kaikilla kertoimilla olisi samat alkuarvot, monet opetuksessa käytettävät osittaisderivaatat olisivat yhtäsuuria ja ne voisivat pysyä sellaisina koko opetuksen ajan. Kertoimien yhtäsuuruus alussa hidastaa oppimista. Jos painokertoimet

ovat suuria, on todennäköistä että monien piilokerroksen neuronien aktivaatiofunktiot jämähtävät funktion tasaiselle osalle, jossa derivaatat ovat lähellä arvoa 0. Tällainenkin tilanne hidastaa opetusta.

E. Virhefunktion valinta

Neuroverkon kykyä mallintaa aineistoa voidaan mitata monilla erilaisilla virhefunktioilla. Tavallisimpia ovat mm. virheiden neliösumma tai keskineliövirhe (MSE)

$$MSE = \frac{\sum_{i=1}^{n} \|y_i - t_i\|^2}{n}.$$

MSE-virhefunktion käyttäminen voi johtaa kuitenkin helposti verkon huonoon yleistämiskykyyn, koska yksi tapa pienentää MSE-arvoa on lisätä neuronien lukumäärää liian suureksi. Helpotus tähän on esimerkiksi rangaistusparametrin käyttäminen: virhefunktion arvoon lisätään sitä suurempi rangaistus, mitä enemmän verkossa on neuroneita/painokertoimia tai mitä suurempia ne ovat:

$$Virhe = \alpha \cdot MSE + (1 - \alpha) \cdot MSW,$$

missä MSW on ko. rangaistustermi, esimerkiksi verkon painokertoimien neliösumma. Arvot α ja $1 - \alpha$ kuvaavat virhefunktion osien tärkeyttä. Rangaistustermi vähentää isojen painokertoimien käyttöä neuroverkossa ja tasoittaa siten verkon antamia tuloksia. Muita, esimerkiksi tilastotieteestä tuttuja virhefunktioita käytetään usein myös neuroverkoille, mutta sopivan funktion valinta riippuu enemmän itse sovelluksesta, esimerkiksi osakekurssien ennustamisessa ei neliövirhesumman minimointi ole niin tärkeää kuin hintamuutosten suunnan oikea ennustaminen.

F. Opetusnopeuskertoimen valinta

Sopivan opetusnopeuskertoimen α valinta voi olla hankalaa, kuten alla oleva esimerkki näyttää. Oletetaan, että yksinkertainen virhefunktio olisi muotoa

$$E = 20\omega_1^2 + \omega_2^2.$$

Kuvassa 2.19 on kuvattu virhefunktion muodostama pinta. Yleensä virhefunktiossa on muuttujina satoja tai jopa tuhansia painokertoimia muuttujina.

Tämän virhefunktion minimikohta on $(\omega_1, \omega_2) = (0, 0)$. Kokeillaan seuraavaksi kuinka nopeasti opetusalgoritmi konvergoi minimiin erilaisilla opetusnopeuskertoimilla α. Valitaan lopetuskriteeriksi $\sqrt{E} < 10^{-6}$.

Virhefunktion osittaisderivaatat ovat $\frac{\partial E}{\partial \omega_1} = 40\omega_1$ ja $\frac{\partial E}{\partial \omega_2} = 2\omega_2$. Kerrointa ω_1 päivitetään siis seuraavan kaavan avulla:

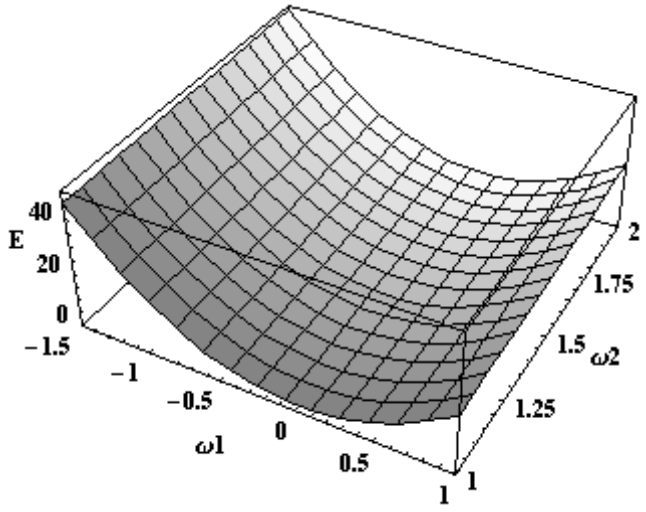
$$\omega_1(t+1) = \omega_1(t) - \alpha \frac{\partial E}{\partial \omega_1(t)}$$

eli

$$\omega_1(t+1) = \omega_1(t) \cdot (1 - 40\alpha).$$

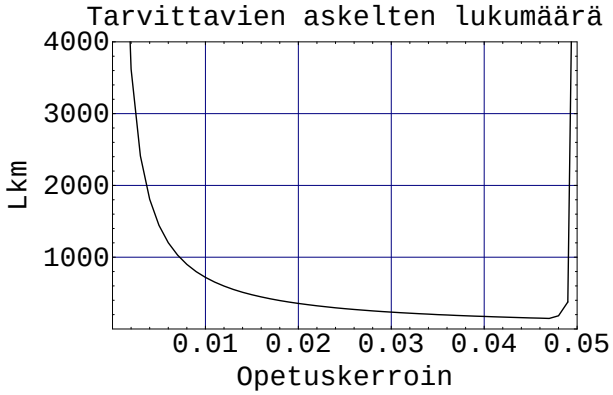
Vastaavasti kertoimen ω_2 päivityskaava on

$$\omega_2(t+1) = \omega_2(t) \cdot (1 - 2\alpha).$$



Kuva 2.19: Yksinkertaistettu esimerkki virhefunktiosta.

Kaaviossa 2.8 on virhefunktion konvergoitumiseen vaadittava askelten lukumäärä erilaisilla opetusnopeuskertoimen α arvoilla. Kaaviosta näkee hyvin kuinka opetusnopeuskerroin voi nopeuttaa oppimista, mutta jos opetusnopeus on liian suuri voi oppiminen estyä kokonaan.



G. Verkon kapasiteetti ja opetusaineisto

Neuroverkon kapasiteetti on suhteessa verkon painokertoimien määrään. Mitä enemmän neuroneita verkossa on, sitä helpompi sen on oppia opetusaineisto. Kun tavoitteena on hyvä verkon yleistyskyky, ei kapasiteetti saisi olla niin suuri että opetusaineiston ulkoaoppiminen käy mahdolliseksi. Koska yleistysominaisuudet vaihtelevat paljon tutkittavan ongelman ja valittujen neuroverkkorakenteiden mukaan, ei yleistä sääntöä verkon koolle suhteessa opetusaineiston määrään voida antaa. Eräitä joskus käytettyjä yleissääntöjä ovat:

- Koko verkon painokertoimien määrä ei ainakaan saisi ylittää sisääntulovektorin kokoa kerrottuna vektorien lukumäärällä. Käytännössä kertoi-

mien lukumäärän pitäisi olla selvästi ko. lukua pienempi.

- Monikerrosverkoissa tulee piilokerrosten neuronien määrän olla opetusaineiston kokoa (rivien lukumäärä) selvästi pienempi.
- Jos tehtävä vaatii kaarevia päätöspintoja, on monikerrosverkoissa piilokerrosten neuronien määrän oltava noin kolme kertaa sisääntulovektorin koko.
- Analogiana voi myös käyttää algebrasta tuttua yhtälöryhmän ratkaisemista. missä yhtä muuttujaa (painokerroin) kohti tarvitaan muista yhtälöistä lineaarisesti riippumaton skalaariyhtälö. Neurolaskennassa tämä tarkoittaa, että opetusaineiston koko kerrottuna ulostulojen määrällä on oltava suurempi kuin painojen määrä.

Yllä olevat säännöt tuntuvat liian väljiltä. Edelleen käytännöllisimpänä sääntönä voinee pitää vaatimusta 5-10 havaintoa painokerrointa kohti.

2.9 Neuroverkkojen edut ja haitat

Neuroverkkojen etuja ja heikkouksia voidaan koota esimerkiksi seuraavanlaiseen taulukkoon.

Edut:

- Neuroverkot ovat joustavia. Niiden rakentaminen on helpompaa ja vähemmän sovelluksen yksityiskohtaista osaamista vaativaa kuin perinteisten menetelmien käyttö. Niillä on myös hyvä siirrettävyys tehtävästä toiseen.
- Virheiden sietokyky. Neuroverkot selviävät muita menetelmiä paremmin epätäydellisistä, puuttuvista tai epätarkoista tiedoista. Esimerkiksi asiantuntijajärjestelmissä epävarmuuden hallinta on vaikeaa.
- Neurolaskenta muodostaa itse omat sääntönsä.
- Neurolaskenta on yleistävää.
- Valmiin, opetetun neuroverkon hyväksikäyttö on laskennallisesti erittäin tehokasta.
- Tarkkuus: Neuroverkot pystyvät approksimoimaan erittäin monimutkaisia, ei-lineaarisia funktioita. Esimerkiksi talouden tilaa kuvaava korkotaso voi eri tilanteissa vaikuttaa eri tavalla muiden muuttujien suhteen, jolloin lineaarinen malli ei sovellu.
- Etukäteisoletuksia havaintoaineistosta, funktioiden muodosta, tms. ei tarvita.
- Neurolaskenta soveltuu hyvin suurten tietomassojen käsittelyyn.
- Helppo ylläpito: neuroverkot on helppo päivittää uudella tiedolla ja uusien muuttujienkin lisääminen on helppoa.

- Neuroverkko voi päihittää monta muuta tilastollista menetelmää, koska sillä ei ole montaa rajoitusta. Esimerkiksi muuttujien jakaumaoletuksia ei neuroverkoilla tarvita.
- Neuroverkon piiloneuronit toimivat kuin eräänlaiset apumuuttujat.
- Neuroverkot voivat hyödyntää rinnakkaislaskentaa. Tietokoneohjelmana toteutetulle neuroverkolle tällä ei ole merkitystä, mutta fyysisenä laitteistona toteutettuna rinnakkaisuudella voi olla suuri merkitys.
- Neuroverkko voidaan ohjelmoida niin, että ihmisen tarve sen ohjaamiseen on pieni.
- Neuroverkot sopivat usein ongelmiin, joihin aiemmat tilastolliset menetelmät eivät ole sopineet.
- Neuroverkot on liitettävissä olemassaoleviin järjestelmiin, kuten jonkin teollisuusprosessin ohjausjärjestelmään.

Haitat:

- Ns. "black-box" –ominaisuus. Neuroverkkojen painokertoimia voi sovelluksesta riippuen olla satoja. Niiden ja eri sisääntulomuuttujien välisten yhteyksien ja yhteyksien voimakkuuden ymmärtäminen voi olla vaikeaa.
- Sopivan neuroverkon rakenteen määritteleminen vaatii usein yritys-erehdys –menetelmää. Ei ole olemassa yleispätevää menetelmää määrätä optimaalisin neuroverkon rakenne.
- Neuroverkon kertoimien laskeminen vaatii paljon havaintoja.
- Ylioppiminen: Jos neuroverkkoon sovitetaan liian monta kerrointa, sen yleistyskyky heikkenee. Silloin neuroverkko on voinut oppia havaintoaineiston satunnaisheilahtelun.
- Neuroverkon toimintakyky riippuu suureksi osaksi opetusaineiston määrästä ja laadusta, koska se on muodostettu suoraan opettamalla mittauksista ja havainnoista.
- Neuroverkot voivat löytää paikallisen optimin globaalin optimin sijasta.
- Neuroverkon tuloksille ei ole luottamusvälien määräämismenetelmää eikä hypoteesien testausta.
- Neuroverkkojen teoria kehittyy edelleen kovaa vauhtia ja reaalimaailman sovelluksia on vielä toistaiseksi vähän.
- Ongelman koodaaminen numeeriseen vektorimuotoon voi olla joskus hankalaa.

Luku 3

Neuroverkkolaskennan tulosten arviointi

Neuroverkon opetuksessa on hyvä käyttää kolmea erillistä opetusaineiston osaa: 1) itse neuroverkon opetusaineisto sekä 2) testausaineisto ja 3) validointiaineisto. Monissa raportoiduissa käytännön sovelluksissa on käytetty vain varsinaista opetusaineistoa. Sellaisisssa tapauksissa ei ole varmaa tietoa miten neuroverkko onnistuu yleistämisessä. Vaikka verkon tuottama virhe (opetusaineistossa) on pieni, voi olla että neuroverkko on oppinut aineiston satunnaisvirheet eikä pysty yleistämään uuteen aineistoon. Testausaineistoa käytetään opetuksen aikana lopettamaan opetus siinä vaiheessa, kun verkko on oppinut tehtävästä mahdollisimman paljon eikä ole alkanut vielä "muistaa ulkoa" havaintoja ja havaintovirheitä.

Miksi sitten olisi hyvä käyttää vielä kolmatta aineistoa opetuksessa? Syy on, että vaikka testausaineistoa ei käytetty painokertoimien tms. muuttamiseen, sitä käytettiin kuitenkin sen määrittelemiseen milloin opetus on lopetettava. Oikea testi neuroverkon tehosta saadaan käyttämällä aineiston osaa, joka ei ole ollut millään muotoa mukana opetuksessa eli validointidataa. Kun (saman ongelman) eri neuroverkkoja vertaillaan, on lähestymistapa empirinen: hyvät mallit erottuvat huonoista paremmilla ennustuksillaan uusilla havainnoilla. Mallien vertailua ja valintaa voi tässä ohjata joko 1) mallinnusteholla eli tulosten tarkkuudella tai 2) operationaalisella teholla eli tulosten perusteella tehtyjen päätöksen paremmuudella. Joskus edellä mainitut kaksi tapaa eroavat toisistaan. Ne johtavat myös erilaisiin neuroverkon testeihin. Operationaalinen tarkastelutapa antaa yleensä tulokseksi konservatiivisempia tai varovaisempia päätöksiä (mallit ovat yksinkertaisempia).

Yleensä neuroverkon mahdollinen ylioppiminen on vakavampi ongelma kuin esimerkiksi paikalliseen minimikohtaan juuttuminen painokertoimien opetuksessa. Joissakin tapauksissa verkon rakenteesta tai opetusaineistosta johtuen verkko voi alkaa ylioppia jo parin, kolmen opetuskierroksen (epookin) jälkeen. Tällaisessa tapauksessa verkko tuskin löytää paikallista saati globaalia minimiä. Paikalliseen minimiin päätyminen voidaan estää tekemällä opetus monta kertaa eri kertoimien lähtöarvoilla ja erilaisilla opetuskertoimilla. Ylioppiminen-

kaan ei ole ongelma, jos opetusaineistoa on riittävästi ja neuroverkkoon ei lisätä ylimäärää neuroneita piilokerroksessa.

Monet muita tilastollisia menetelmiä käyttämään tottuneet soveltajat, esimerkiksi tilastotieteilijät, aktuaarit ja vakuutusanalyytikot, eivät ehkä ole kovin innostuneita käyttämään neuroverkkoja tiedon analysoinnissa niiden tulosten arviointiin liittyvien puutteiden vuoksi. Yksi tällainen puute liittyy käsitykseen neuroverkkojen "black box"-luonteesta: data menee sisälle ja tulokset tulevat ulos, mutta miten tulokset saatiin pysyy mysteerinä. Neuroverkon antamat tulokset paljastavat vain vähän selitettävän ja selittävien muuttujien välisestä yhteydestä. Siksi neuroverkkokokeilun tuloksia voi olla vaikeaa esitellä muille. Neuroverkkoa ei voi pilkkoa muutamaksi yksinkertaiseksi funktioksi, joita olisi helppoa tulkita. Samalla neuroverkosta ei ole mahdollista saada täyttä hyötyä mallinnettavan ilmiön ymmärtämisessä.

Seuraavaksi esitellään joitakin neuroverkon tulosten arviointiin ja mallin selittämiseen liittyviä menetelmiä.

3.1 Luokittelevat neuroverkot

Luokittelevien neuroverkkojen tehokkuuden arviointiin on aika yksinkertaisia tapoja. Havaintoja k eri luokkaan lajittelevan neuroverkon virhe $E = \sum_i^k (y_i - d_i)^2$ tuskin tulee nollaksi opetuksessa. Esimerkiksi sigmoid-funktio ei saa millään rajallisella syötteellä arvoa 1 tai 0. Lisäksi verkon rakenne asettaa rajan kuinka tarkkaan edes periaatteessa verkko voi oppia tehtävän. Useimmiten käytännön sovellukset, ja etenkin talouteen liittyvät, ongelmat ovat erittäin monimutkaisia. Talouteen liittyvät aikasarjat sisältävät niin paljon "kohinaa", että teoreettisimmat tutkijat yhä uskovat huolimatta näennäisestä säännönmukaisesta kehityksestä lyhyillä aikaväleillä, että esimerkiksi rahoitusmarkkinoiden muutokset ovat pohjimmiltaan ennustamattomia ja liikkuvat satunnaisesti (tehokkaiden markkinoiden hypoteesin vahva muoto).

Virheiden neliösumma ei välttämättä ole edes paras mittari luokittelevan neuroverkon tehokkuudelle. Yksinkertaisin luokitteleva verkko sisältää vain yhden ulostulon, jonka arvo on 1 *positiivisille tapauksille* ja 0 *negatiivisille tapauksille.* Oletetaan, että verkko on opetettu ja ulostuloneuronin arvon tulkinta olisi luonnollinen: $0 \leq$ arvo $\leq 0,5$ merkitsee luokkaa 0 (negatiivinen) ja $0,5 \leq$ arvo ≤ 1 merkitsee luokkaa 1 (positiivinen). Arvoja 0 tai 1 ei sigmoid-funktio voi saada äärellisellä arvolla.

Kun MLP-verkossa käytetään aktivaatiofunktiona sigmoid-funktiota, ovat ulostulot jatkuvia ja rajoitettu välille $(0, 1)$. Ulostuloa $y(x)$ voidaan siis tulkita todennäköisyydeksi $p(t = 1|x)$ eli havainnon x:n todennäköisyydeksi kuulua luokkaan $t = 1$. Verkosta saadaan havaintojen luokittelija, kun asetetaan raja-arvo jota suuremmat arvot kuuluvat luokkaan $t = 1$ ja pienemmät arvot kuuluvat luokkaan $t = 0$. Rajaa kutsutaan luokittelun kynnysarvoksi (classification threshold). Optimaalinen arvo tälle rajalle saadaan ottamalla huomioon kustannukset vääristä luokitteluista. Optimaalinen luokittelija toteuttaa ehdon

$$\arg \min_{j \in \{0,1\}} \sum_{k=0}^{1} p(k|x) L_{j,k}(x),$$

missä $p(k|x)$ on luokan k ehdollinen todennäköisyys ehdolla x ja $L_{j,k}(x)$ on kustannus kun luokkaan k kuuluva havainto luokitellaan luokkaan j havaintovektorin ollessa x. Kyseessä on kustannus, kun $L_{j,k}(x) > 0$ ja palkkio kun $L_{j,k}(x) < 0$. Luokittelun $t = 1$ raja-arvoksi saadaan yo. kaavasta arvo

$$p(t = 1|x) = \frac{L_{1,0}(x) - L_{0,0}(x)}{L_{0,1}(x) - L_{1,1}(x) + L_{1,0}(x) - L_{0,0}(x)}, \tag{3.1}$$

jos oletetaan että väärinluokittelun kustannus on aina suurempi kuin palkkio oikeasta luokittelusta. Käytännössä kustannukset $L_{j,k}$ ovat yleensä riippumattomia havainnoista x. Silloin raja-arvoksi saadaan laskettua helposti kiinteä arvo yllä olevasta kaavasta.

Seuraavassa on vielä pari muuta yksinkertaista luokittelun mittaria.

Oikein luokittelu –aste% (success rate) määritellään seuraavasti

$$\frac{\textit{oikein luokiteltujen lkm}}{\textit{kaikkien tapausten lkm}} \times 100\%.$$

Oletetaan, että oikein-luokittelu-% olisi 95%. Tulos ei välttämättä ole kovin hyvä, jos luokkien esiintymistiheydet ovat erilaiset havaintoaineistossa. Esimerkiksi aineistossa voisi olla 95% tapauksia 1 ja neuroverkko luokittelee kaikki tapaukset aina luokkaan 1.

Luokittelun herkkyys (sensitivity): Oikeiden positiivisten tulosten suhteellinen osuus kaikista oikeasti positiivisista tapauksista.

Luokittelun tarkkuus (specificity): Oikeiden negatiivisten tulosten suhteellinen osuus kaikista negatiivisista tapauksista.

Tarkkuus ja herkkyys saavat arvoja väliltä $[0, 1]$. Luokittelun herkkyyttä ja tarkkuutta käytetään esimerkiksi ROC-käyrissä. Yksi suosittu tapa arvioida luokittelevien neuroverkkojen tai muidenkin luokittelevien algoritmien tehoa ovat toimintaominaiskäyrät eli lyhyesti ROC-käyrät (Receiver operating characteristic curves). Luokittelun kynnysarvo vaikuttaa luokittelun herkkyyteen ja tarkkuuteen. Tutkimalla luokittelun luokittelykykyä eri kynnysarvoilla voidaan piirtää kaksiulotteinen kuvaaja herkkyys vs. 1-tarkkuus.

ROC-käyrä on 2-ulotteinen kaavio, jossa erilaisilla luokittelun kynnysarvoilla verrataan oikein tiettyyn luokkaan luokiteltujen tapausten osuutta väärin ko. luokkaan luokiteltujen tapausten osuuteen, ts. y-akselilla ($\frac{N_{1,1}}{N_{1,1}+N_{0,1}}$) ja x-akselilla ($\frac{N_{1,0}}{N_{1,0}+N_{0,0}}$)). Jos ROC-käyrä kulkee suoraviivaisesti pisteiden (0,0) ja (1,1) kautta, vastaa malli kolikonheittoa. Jos käyrän alle jäävä pinta-ala on alle 0,5, on malli huonompi kuin kolikonheitto. Mitä suurempi on käyrän alapuolelle jäävä pinta-ala, sitä parempi luokittelija on. Käytännössä kun luokittelijaa on tarkoitus käyttää jonkin aineiston luokitteluun, on luokittelun kynnysarvo kuitenkin kiinnitettävä johonkin tiettyyn arvoon.

3.2 Jatkuva-arvoiset, ennustavat neuroverkot

Ennustavien neuroverkkojen lopullista tehokkuutta mitataan yleensä samalla virhemitalla, kuin mitä verkon opetuksessa käytettiin. Yleisin käytetty menetelmä on keskimääräinen virheen neliösumma eli MSE.

3.3 Neuroverkkomallin tulkitseminen

Opetettua neuroverkkoa voidaan tulkita ainakin kahdesta eri näkökohdasta: eri muuttujien merkitys mallissa ja selitettävän ja selittävien muuttujien välisten yhteyksien havainnollistaminen.

Yksi tapa arvioida muuttujien merkitystä selitettävän muuttujan ennustamisessa on tutkia opetetun neuroverkon kertoimia sisääntulo- ja piilokerrosten välissä. Ne muuttujat, joihin liittyvät kertoimet ovat lähinnä arvoa 0, tulkitaan vähämerkityksellisimmiksi. Tähän menetelmään liittyy kuitenkin joitakin kysymyksiä. Ensinnäkin vain sellaiset muuttujat, joihin liittyvät kertoimet ovat kaikki lähellä nollaa, voidaan tulkita vähämerkityksellisiksi. Toiseksi, muuttuja voi olla tarpeellinen mallissa vaikka siihen liittyvät kertoimet ovatkin pieniä. Joissakin raportoiduissa tapauksissa on tarkasteltu muuttujiin liittyvien painokertoimien summia. Tällä summalla ei ole kuitenkaan mitään tilastollista selitystä, jota voisi käyttää muuttujien merkityksen arvioimiseen. Yleensä tätä menetelmää käytetään vain muuttujien poistamiseen neuroverkkomallista. Näin verkko (ja samalla kertoimien määrä) saadaan pienemmäksi ja opetus tehokkaammaksi. Samalla kuitenkin verkossa pidetään todennäköisesti parhaimmat selittävät muuttujat mukana. Tämän menetelmän etu on, että se on helppo tehdä kun neuroverkko on opetettu.

Yksittäisten sisääntulojen ja ulostulojen välisen yhteyden voimakkuutta voidaan arvioida myös seuraavalla mittarilla RS_{ji}

$$RS_{ji} = \frac{\sum_{k=0}^{n}(w_{ki} \cdot w_{jk})}{\sum_{i=0}^{m}|\sum_{k=0}^{n}(w_{ki} \cdot w_{jk})|},$$

missä W_{ki} on painokerroin piiloneuronin k ja sisääntuloneuronin i välissä ja W_{jk} on painokerroin ulostuloneuronin j ja piiloneuronin k välissä. Kaava yksinkertaistuu vielä hieman, jos ulostuloja on vain yksi. Silloin RS_{ji} saadaan kaavasta

$$RS_i = \frac{\sum_{k=0}^{n}(w_{ki} \cdot w_k)}{\sum_{i=0}^{m}|\sum_{k=0}^{n}(w_{ki} \cdot w_k)|}$$

Sisääntulon merkitys ulostulolle on sitä suurempi mitä suuremman arvon RS_{ji} saa.

Vielä yksi tapa muuttujien merkityksen arvoimiseen on ns. herkkyysanalyysi. Herkkyys on määritelty mittana siitä, miten neuroverkon virhe kasvaa kun muuttuja poistetaan neuroverkkomallista. Yleensä neuroverkkoa ei opeteta uudelleen aina kun yksi muuttuja on poistettu, koska se vaatisi tietysti muuttujien lukumäärää vastaavasti lisää laskenta-aikaa. Sen sijaan käytetään yleensä seuraavanlaista menettelytapaa:

- Kiinnitä tarkasteltavan muuttujan sisääntulot vakioarvoksi (esimerkiksi muuttujan keskiarvo tai mediaani).
- Laske neuroverkon tulokset testiaineistolle, kun em. muuttujan arvo on kiinnitetty.
- Laske virheen keskineliösumma testiaineistolle ja vertaa sitä täyden mallin antamaan virheen keskineliösummaan.

- Toista edelliset vaiheet kaikille muuttujille vuorollaan. Muuttujan herkkyys määritellään suhteelliseksi virheen pienemiseksi muuttujan sisältävässä mallissa verrattuna malliin, josta muuttuja oli poistettu.
- Muuttujat voidaan järjestää niiden merkityksen perusteella.

Oikeastaan verkko olisi opetettava aina uudestaan, kun jokin muuttuja on poistettu. Jokin toinen muuttuja ja jäljelläolevien muuttujien väliset yhteydet voisivat riittävästi korvata poistetun muuttujan. Siinä tapauksessa poistettu muuttuja ei olisikaan ollut tarpeellinen mallissa. Muuttujan sisääntulojen kiinnittäminen johonkin tiettyyn arvoon voi antaa kuitenkin toisenlaisen tulkinnan. Toisaalta, verkon opettaminen aina uudestaan vaatisi aikaa ja aina on mahdollista, että eri opetuskertojen tarkkuus olisi erilainen (esim. juuttuminen paikalliseen minimiin, tms.)

Muuttujan herkkyysmittari voidaan siis laskea esimerkiksi suhteena kahdesta virheestä: mallin, josta muuttuja on poistettu, neliövirhesumma ja kokonaisen mallin neliövirhesumma. Mitä suuremman arvon mittari saa sitä suurempi merkitys tarkastellulla muuttujalla on. Suhdeluvun pitäisi olla aina vähintään 1. Jos se on alle 1, on kokonaisen mallin virhe liian suuri eli verkon painokertoimet ovat osuneet johonkin paikalliseen minimiin.

Muuttujien järjestäminen niiden tärkeyden mukaan ei kerro vielä muuttujien tärkeydestä mallissa. Seuraavassa on yksi mahdollinen tapa valita neuroverkkoon tarpeelliset muuttujat ja pitää neuroverkko vielä mahdollisimman pienenä. Tämäkin menetelmä vaatii paljon laskenta-aikaa ja käyttäjän vaikutusta.

- Ensimmäiseen neuroverkkoon valitaan kaikki tai ainakin varmasti riittävä otos selittävistä muuttujista sisääntuloiksi. Tässä vaaditaan jo mahdollisesti käyttäjän vaikutusta, kun muuttujat valitaan.
- Neuroverkko opetaan normaalisti ja neuroverkon selitysaste testiaineistolle lasketaan.
- Mallin muuttujat järjestetään niiden herkkyyden perusteella ja seuraavaan neuroverkkoon otetaan yksi vähemmän muuttujia. Mukaan otetaan muuttujat tärkeimmästä alkaen.
- Kahta edellistä vaihetta toistetaan kunnes verkossa on jäljellä enää vain yksi muuttuja.
- Tuloksena saadaan vektori muuttujien lukumääristä ja selitysasteista. Jollakin muuttujien lukumäärällä selitysaste on riittävän hyvä tai sitten esimerkiksi selitysaste ei juuri enää muutu vaikka muuttujia lisättäisiin.

3.4 Neuroverkon tulosten visualisointi

Jonkin ilmiön matemaattista mallia voi yrittää havainnollistaa kuvaamalla mallin saamat arvot vuorotellen kaaviossa kunkin selittävän muuttujan suhteen. Ongelma on, että neuroverkossa on mukana paljon selittävien muuttujien yhteisvaikutuksia. Jonkin muuttujan vaikutus mallissa ei riipu siis ainoastaan sen

itsensä saamasta arvosta vaan myös muiden muuttujien samaan aikaan saamista arvoista. Tällaiset kaaviot eivät yleensä kerro paljoakaan muuttujien vaikutuksesta mallinnettavaan muuttujaan, koska samalla selittävän muuttujan arvolla mallin tulos voi olla aivan erilainen riippuen muiden muuttujien arvoista.

Seuraavassa on yksi mahdollinen tapa yrittää kuvata muuttujan merkitystä neuroverkossa kaavion avulla.

- Kiinnitä kaikki muut kuin tarkasteltava muuttuja joiksikin vakioiksi. Sopivia ehdokkaita vakioiksi ovat muututjien keskiarvot ja mediaanit.
- Laske neuroverkon saamat arvot kaikilla aineiston havainnoilla. Vaihtoehtoisesti voi valita vain jonkin edustavan joukon tarkasteltavan muuttujan arvoja. Muiden muuttujien arvot pysyvät samoina havainnoista riippumatta.
- Piirrä neuroverkon tulokset kaavioon tarkasteltavan muuttujan arvojen suhteen.
- Suositeltavaa, mutta ei pakollista, on skaalata neuroverkon tulos välille $[0, 1]$.

Tällä tavalla neuroverkon tulosten ja kaikkien selittävien muuttujien yksi kerrallaan välistä yhteyttä voi tutkia kaavioissa. Antamalla kahden selittävän muuttujan vaihdella saadaan aikaiseksi 3-ulotteinen kuvaaja. Useampiulotteisia kuvaajia on hankalampi kuvata kaavioissa. Jos toinen tarkasteltava muuttuja saa vain muutamia (esimerkiksi 2-4) arvoja, on ehkä mahdollista piirtää vastaava lukumäärä erillisiä kuvaajia joissa kussakin on ko. muuttuja kiinnitetty ja vain toinen vaihtelee.

Nämä esimerkit vain osoittivat, että usein neuroverkkoa voi visualisoida samoilla tavoilla kuin muitakin menetelmiä. Herkkyysanalyysi kertoo eri muuttujien merkityksestä tuloksiin ja visualisointi kertoo muuttujien ja selitettävän muuttujan välisestä yhteydestä.

3.5 Itseorganisoituvan kartan visualisointi

Seuraavassa kerrotaan miten SOM-verkon tuloksia voidaan visualisoida ja tulkita, kun kartassa on suuri määrä mallivektoreita.

Itseorganisoituva kartta kuvaa havaintovektorit kartan neuroneille, joita kutsutaan usein myös mallivektoreiksi. Neuronit on yleensä sijoitettu 1- tai 2-ulotteiseen ruudukkoon. Havaintojen kuvautuminen neuroneille ei vielä kuitenkaan kerro paljon aineiston luokista ja rakenteesta.

Kartan visualisointi voidaan tehdä esimerkiksi U-matriiseja (Unified distance matrix) käyttäen. Neuronien tiheys ja lähekkäisyys valmiissa SOM-kartassa kuvaa myös havaintoaineiston vektoreiden tiheyttä ja lähekkäisyyttä. U-matriisiin on laskettu kartan neuronien ja niiden naapureiden väliset (yleensä euklidiset) etäisyydet. Tarkemmin sanottuna kullekin kartan neuronille on laskettu etäisyyksien summa sen naapureihin skaalattuna pisimmällä noista etäisyyksistä. Etäisyydet voidaan sitten esittää vaikka normaalin maastokartan väreissä. Tällöin

sininen (vesi) ja ruskea väri kuvaavat yksiköiden lyhyitä ja valkoiset pitkiä välejä. Vaaleat alueet ovat siis ikään kuin vuoristoja, jotka erottavat ryhmiä (tasankoja, laaksoja, yms). Neuronien tiheys ja lähekkäisyys valmiissa kartassa kuvaa myös havaintoaineiston vektoreiden tiheyttä ja lähekkäisyyttä. Jos SOM-verkko ei "tiedä" miten luokitella jokin tietty havainto, havainto kuvautuu kartassa jollekin alueelle, jolla on vain vähän muita havaintoja. Se ei siis väkisin anna havainnolle jotakin luokkaa. Tällaiset havainnot voivat olla jotenkin erilaisia, virheellisiä havaintoja tai ne voivat olla merkki siitä, että ehkä jokin oleellinen muuttuja puuttuu mallista.

Esimerkkejä SOM-verkon visualisoinneista on kuvissa 4.11 sivulla 74.

U-matriisien visualisoinneilla on mm. seuraavia ominaisuuksia:

- Aineiston lähekkäiset havainnot ovat myös kartalla lähekkäin.
- Osumat (bestmatch), jotka ovat toistensa lähinaapureita aineistossa, sijaitsevat kartan samassa laaksossa.
- Kartan "vuoret" ja "ylängöt" erottavat ei havaintojen luokkia.
- Poikkeavat havainnot näkyvät kartassa yksittäisinä pisteinä, joita vuoret ympäröi.

Luku 4

Mahdollisia sovelluksia vakuutusalalla

Vakuutusalalla on paljon mahdollisuuksia neuroverkkojen soveltamiselle. Esimerkkejä sovellusalueista ovat korvausvastuut, kuolevuusennusteet, vakuutusten hinnoittelu ja luokittelu, vakuutettavien riskien valinta, vararikon ennustaminen ja sijoitussuunnitelmat. Tässä luvussa esittelen joitakin esimerkkejä neuroverkkojen soveltamisesta vakuutusdataan. Selostan myös hieman vakuuttamisen erityispiirteitä neuroverkon sovellusalueena ja myös kuinka paljon tällaisia sovelluksia on jo olemassa. Jotkin sovellukset eivät ole merkityksellisiä vain vakuutusalalla vaan myös esimerkiksi rahoituksessa, pankkitoiminnassa tai tavaroiden tai palvelusten myynnissä. Näilläkin neuroverkkojen sovelluksilla on kuitenkin vakuuttamiseen liittyviä erikoispiirteitä. Joitakin toisia sovelluksia taas ei käytetä muilla kuin vakuuttamisen alalla, kuten korvausvastuiden estimointi.

Vakuutusyhtiöillä, jotka kykenevät tarjoamaan asiakkaille uusia, innovatiivisia tuotteita ja palveluja, on mahdollisuudet kasvuun ja tasaiseen kannattavaan kehitykseen. Neuroverkot voivat olla jollakin tavalla apuna esimerkiksi seuraavissa vakuutusyhtiön prosesseissa:

- keskivahingon ennustaminen
- vakuutusten hinnoittelu
- uusien asiakkaiden hankinta
- nykyisten asiakkaiden pitäminen
- uusien vakuutustuotteiden kehittäminen
- raportointi
- epäselvien vahinkojen erottelu kaikista vahingoista
- markkinointi
- varausten laskeminen

- vakavaraisuuden tutkiminen

Pääpaino neurolaskennan soveltamisessa on prosessissa. Pelkkä neurolaskentasovellus ei riitä, vaan soveltajalla täytyy olla tarpeeksi tietämystä vakuutusalasta ja neurolaskennasta. Neurolaskenta on tyypillisesti jatkuva prosessi, jossa on monia vaiheita jotka voivat toistua monta kertaa. Tämän prosessin päävaiheet yleisesti ovat:

- Tutkittavan ongelman ymmärtäminen
- Tilastoaineiston ymmärtäminen
- Tilastoaineiston valmistelu
- Mallintaminen (sisältää neuroverkkojen opetus- ja muiden tiedonlouhinta-algoritmien soveltamisen)
- Tulosten arviointi
- Mallin käyttäminen (ennustamiseen, luokitteluun, jne)

Itse neuroverkot voidaan nähdä muiden, tutumpien laskentamallien kuten lineaariregression yleistyksinä tai laajennuksina. Neuroverkkojen avulla voidaan selvitä monista vakuustusalan aineistoon liittyvistä haaveista, joista kolme tärkeintä ovat varmasti

- ei-lineaariset funktiot
- korrelaatiot datassa
- eri tekijöiden väliset monimutkaiset yhteydet

Neuroverkkojen soveltamisesta juuri vakuutusalalla ei löydy helposti käytännön tai pintaa syvemmälle menevää materiaalia. Yeo ja Smith ovat selostaneet eräässä artikkelissa [16] yhtä neurolaskennan projektia autovakuutuksissa. Artikkelista voi lukea rivien välistä, kuinka vakuutusyhtiön johto on ollut alussa innokas soveltamaan neuroverkkoja mutta projektin hiljalleen edetessä ajankohtaisemmat asiat ja ajan kuluminen ovat sitten saaneet innostuksen hiipumaan. Neurolaskenta ei ole mikään taikatemppu eikä sitä voi soveltaa ilman riittävää tietämystä. Suurimman osan ajasta niin kuin yleensäkin tilastotutkimuksissa vie havaintoaineiston kerääminen ja muokkaaminen.

Yeon ja Smithin artikkelissa esitellyssä projektissa päädyttiin lopulta kolmen vuoden jälkeen seuraavaan neuroverkkojen soveltamisen malliin. Kokonaistavoitteena oli maksujen määrääminen niin, että toiminta olisi kannattavaa. Vakuutusmaksujen on siis katettava korvauskulut. Lisäksi vakuutusmaksut eivät saisi olla niin korkeat, että ne vähentäisivät liikaa markkinaosuutta. Projektissa vakuutukset oli luokiteltu niiden riskillisyyden mukaan. Lisäksi toisella neuroverkkomallilla oli tutkittu vakuutuksien hintaherkkyyttä. Mallien tulokset oli sitten yhdistetty optimoimalla kannattavuus, kun vakuutusyhtiön johto antoi rajat kannan koon muutoksille. Mallista puuttuu vielä kilpailijoiden toimenpiteiden vaikutukset. Näin ollen se ei ole kovin luotettava suurissa hinnoittelumuutoksissa. Toisaalta tuloksena oli parempi vakuutusmaksumuutosten jako eri vakuutusryhmille.

4.1 Keskivahinko

Seuraava esimerkki näyttää miten neuroverkkoja voidaan käyttää epälineaaristen funktioiden approksimointiin.

Kun keskivahinkoa varten tehdään ennusteita, mallissa käytetään joskus potenssifunktiota:

$$\text{Keskivahinko}_t = \text{Keskivahinko}_{t_0} \cdot I^{t-t_0} \cdot e.$$

Yllä olevassa kaavassa I on inflaatiokerroin ($1 +$ inflaatioprosentti), t_0 on tunnetttu vertailuajankohta ja e on satunnainen virhetermi. Jos yllä olevan yhtälön molemmille puolille tehdään logaritmimuunnokset, saadaan

$$\ln(\text{Keskivahinko}_t) = ln(\text{Keskivahinko}_{t_0}) + (t - t_0) \cdot \ln(I) + \ln(e),$$

mikä on lineaarinen termin $(t - t_0)$ suhteen.

Käytetään seuraavissa simuloinneissa hieman yleistetympää mallia, jossa keskivahingon kasvu on nopeampaa

$$\text{Keskivahinko}_t = \text{Keskivahinko}_{t_0} \cdot I^{(t-t_0)^p} e.$$

Valitaan $t_0 = 0$, $\text{Keskivahinko}_0 = 10000$, $p = 1, 2$ ja

$$\begin{aligned} Y &= 10000 \cdot I^{1,2} \\ I &\sim N(1,05; 0,005) \end{aligned}$$

I kuvastaa nyt satunnaista vuosi-inflaatiota.

Todellisuudessa keskivahingon koko vaihtelee enemmän kuin tässä keinotekoisessa esimerkissä. Malli ei muutenkaan tunnu kovin hyvältä, koska vaihtelu viereisinä kvartaaleina on aluksi pientä mutta suurta simuloinnin viimeisinä kvartaaleina.

Simuloidaan 40 vuosineljänneksen keskivahingon koot. Ne on esitty kuvassa 4.1 yhdessä mallin odotusarvojen kanssa (yhtenäinen viiva).

Seuraavaksi opetetaan simuloidut keskivahingot yksinkertaiselle neuroverkolle, jossa aktivaatiofunktioina käytetään logistista funktiota sekä piilo- että ulostulokerroksessa. Monissa käytännön sovelluksissa ulostulokerroksen neuroneille käytetään lineaarista aktivaatiofunktiota.

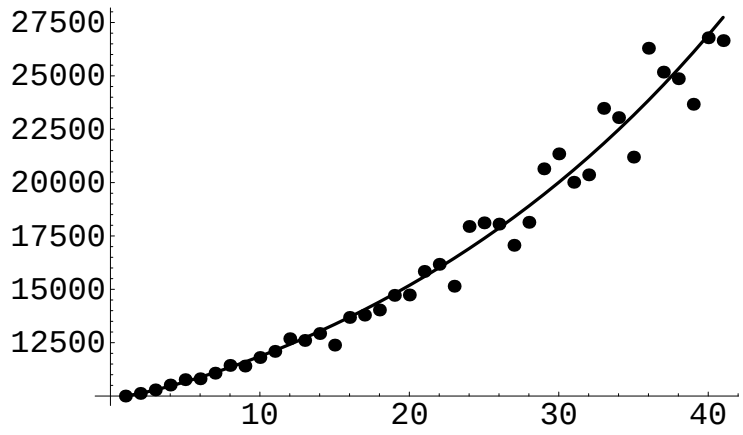
Vertailun vuoksi sovitetaan havaintoaineistoon myös regressiokäyrä.

Neuroverkossa on tässä tapauksessa vain yksi sisääntuloneuroni (aika, ts. kvartaali), yksi piilokerroksen neuroni ja yksi ulostuloneuroni (keskivahingon ennuste).

$$f(X) = \frac{1}{1 + e^{-X}}.$$

Tässä yksinkertaisessa tapauksessa piilokerroksen neuronin arvo siis saadaan kaavasta

$$h = f(t; w_0, w_1) = f(w_0 + w_1 * t) = \frac{1}{1 + e^{-(w_0 + w_1 * t)}}.$$



Kuva 4.1: Simuloidut keskivahingot kvartaaleittain ja odotusarvojen käyrä.

Vastaavasti lopullinen ulostulo saadaan kaavasta

$$o = f(h; w_2, w_3) = \frac{1}{1 + e^{-(w_2 + w_3 * h}} = \frac{1}{1 + e^{-(w_2 + w_3 \cdot \frac{1}{1+e^{-(w_0 + w_1 * t)}})}}.$$

Aineisto (sisään- ja ulostulot) pitää normalisoida ennen neuroverkon opetusta. Se voidaan tehdä monella tavalla, mutta tavallisin ja turvallisin tapa on vähentää jokaisesta havainnosta pienin arvo (opetusaineistossa) ja jakaa vielä havaintojen vaihteluvälillä (ts. maksimi-minimi). Neuroverkon ulostulo muutetaan takaisin kertomalla normalisoinnissa käytetyllä vaihteluvälillä ja lisäämällä sitten vielä vähennetty vakio.

Neuroverkon muodostamista Joone-ohjelmalla varten talletetaan simuloidut keskivahingot Excelistä csv-tiedostomuodossa. Liitteessä työn lopussa on esitetty kuva 7.2 Joone-ohjelmalla muodostetusta verkosta. Kvartaalit 0-40 on normalisoitu valmiiksi opetusaineistossa välille $[0, 1]$. Myös keskivahinko-havainnot on normalisoitu samalle välille. Näin verkko voi suoraan lukea havainnot, eikä mitään ohjelman tekemää normalisointia enää tarvita.

Neuroverkon opetus menee läpi nopeasti. Opetuksessa ei käytetty testaus- tai validointijoukkoa, koska havaintoja oli vain 41 kpl. Se ei ole ongelma, koska verkko ei voi oppia havaintoja ulkoa sillä piilokerroksessa on vain yksi neuroni eli yksinkertaisempaa verkkoakaan ei voi olla.

Neuroverkolle tuli seuraavanlaiset kertoimet

Piilokerros:

$$\left\{ \begin{array}{rcl} w_0 & = & 3,096002 \\ w_1 & = & -3,69545 \end{array} \right)$$

ja ulostulokerros:

$$\left\{ \begin{array}{rcl} w_0 & = & 5,811116 \\ w_1 & = & -8,75064. \end{array} \right)$$

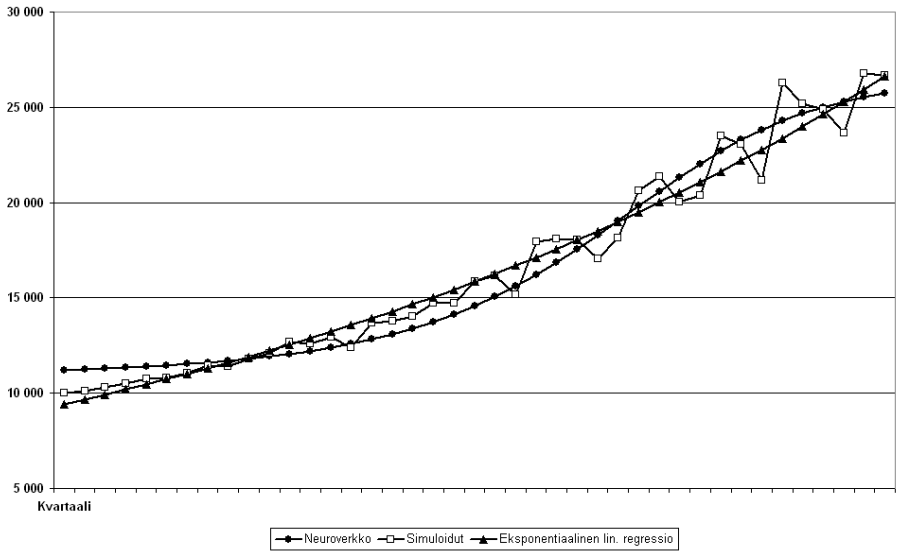
Kun sitten tehdään lineaarinen regressio logaritmi-muunnetuille keskivahingoille, saadaan kertoimiksi $b_0 = 9,15191$ ja $1,03714$. Keskivahingon malli on siis nyt $e^{b_0+b_1*Kvartaali}$.

Kuvassa 4.2 kummankin mallin kuvaajat sekä simuloidut keskivahingot on esitetty yhdessä. Kuvasta näkyy, että neuroverkon antama malli on ihan yhtä hyvä kuin eksponentiaalifunktion antama. Kannattaa huomata, että simuloinnissa käytettiin eksponentiaalifunktiota ja neuroverkko oli yksinkertaisin mahdollinen. Neuroverkko ei kuitenkaan ole suositeltava tällaiseen ongelmaan, jossa on vain yksi selittävä tekijä. Muut menetelmät ovat yleensä parempia ainakin jossakin suhteessa. Näin on varsinkin, jos on olemassa ennakkotieto selittävän ja selitettävän muuttujan välisestä funktiosta.

Neuroverkon ja regressiomallin selitysasteet olivat $0,962$ ja $0,977$ eli samaa tasoa. Neuroverkoille ei ole olemassa varsinaisia tilastollisia testejä. Edellä olleet selitysasteet on laskettu kaavalla

$$r^2 = 1 - \frac{SS_{Error}}{SS_{Total}} = 1 - \frac{\sum(d_i - y_i)^2}{\sum(d_i - \bar{d})^2},$$

missä d_i on tavoiteltu tulos, y_i on mallin antama tulos ja $\bar{d}$ on keskiarvo arvoista d_i.

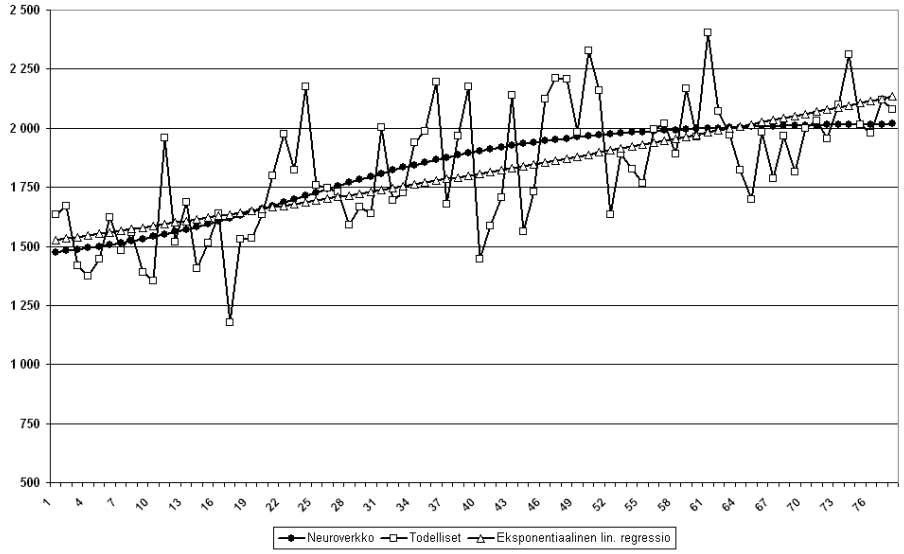


Kuva 4.2: Simuloidut keskivahingot ja mallien tulokset.

Kun sitten samaa kokeiltiin oikealla datalla yritysajoneuvojen kaskovakuutuksista, saatiin yksinkertaisen neuroverkon mallin selitysasteeksi $0,456$ ja regressiomallin selitysasteeksi $0,436$. Nyt selitysasteet ovat selvästi alemmat, mutta se oli odotettavissakin ja taas mallien selitysasteet ovat samaa luokkaa. Kaaviossa 4.3 on todelliset keskivahingot ja sovitetut mallit. Neuroverkolle käytettiin samaa mallia kuin edellä simuloitujen keskivahinkojen tapauksessa.

Neuroverkon malli kuvaa paremmin miten keskivahinko on kehittynyt viime vuosina. Regressiomalli puolestaan arvioi keskimääräisen muutoksen käytössäolevista

tiedoista ja ennustaa jatkokehityksen olevan samanlaista kuin historiassa keskimäärin. Koska näissä kahden mallin tuloksissa on eroa, voisi yksi selitys olla että vahinkoinflaatiossa on tapahtunut jonkinlainen muutos aiempaan historiaan verrattuna ja ilmeisesti muutos on, että vahinkoinflaatio on ainakin ko. vakuutuskannan osassa hieman tasaantumassa. Siihen voi vaikuttaa moni asia: muutokset vakuutuksissa, sääolosuhteissa, liikenneympäristössä, jne.



Kuva 4.3: Todelliset keskivahingot ja mallien tulokset.

Seuraavaksi kokeillaan miten neuroverkolla onnistuu kvartaalitasoisen keskivahingon mallintaminen. Pohjana on sama data kuin edellä, mutta kvartaaleittain esitettynä. Neuroverkkomalliin valitaan nyt sisääntuloiksi kvartaalin numero ja neljä viimeisintä havaintoa. Neuroverkon ulostuloksi valitaan seuraavan kvartaalin keskivahinko eli sitä yritetään ennustaa.

Neuroverkon kertoimiksi tulee seuraavanlaiset

Piilokerros:

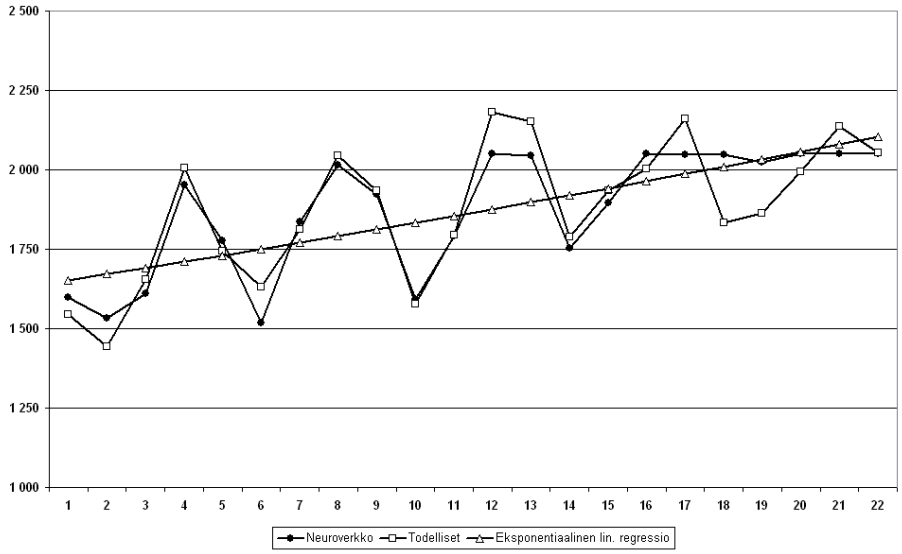
$$\left\{ \begin{array}{lcl} w_0 & = & -1,4004 \\ w_1 & = & 12,1251 \\ w_2 & = & 2,6774 \\ w_3 & = & -9,7358 \\ w_4 & = & 2,3321 \\ w_5 & = & -0,0366 \end{array} \right)$$

Kerroin w_0 on kynnysarvo ja muut painokertoimet liittyvät kvartaalinumeroon (kerroin w_1) ja keskivahinkojen havaintoihin $t-1,\ t-2,\ t-3$ ja $t-4$ (kertoimet $w_2 - w_5$).

ja ulostulokerros:

$$\left\{ \begin{array}{lcl} w_0 & = & -2,24659 \\ w_1 & = & -3,777412. \end{array} \right)$$

Neuroverkkomallin selitysaste on $0,833$. Vastaavan eksponentiaalisen regressiokäyrän selitysaste on $0,406$. Nyt malleille on tullut jo eroa. Eroa havainnollistaa myös todellisten keskivahinkojen ja vastaavien keskivahinkoennusteiden kuvaajat kuvassa 4.4.



Kuva 4.4: Keskivahingot kvartaaleittain - todelliset ja ennustetut.

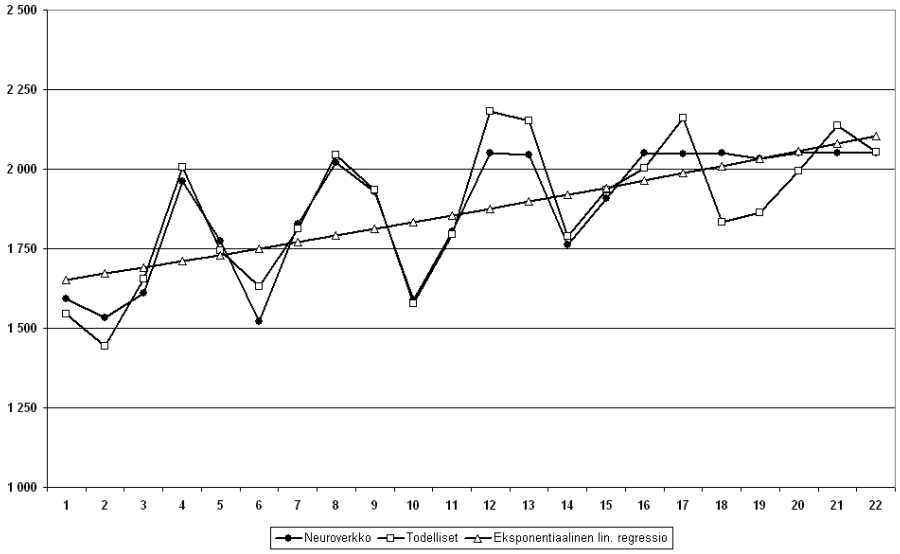
Kuvasta näkyy, että pari kolme ensimmäistä vuotta neuroverkon ennusteet osuvat hyvin, mutta kahtena viimeisenä vuotena ennusteet eivät enää osu. Todellisten keskivahinkojenkin kuvaaja muuttuu samaan aikaan. Painokertoimien arvoista voisi päätellä eri sisääntulojen merkitystä ennusteelle. Itseisarvoltaan suurin painokerroin on kvartaalilla. Toisin sanoen, ajan kuluessa vahinkoinflaatio kohottaa keskivahinkoa. Vuoden takainen $(t-4)$ keskivahinko näyttäisi olevan vähämerkityksisin.

Muodostetaan vielä vertailun vuoksi kaksi neuroverkkoa: Malli2 ilman vuoden takaista havaintoa, ja Malli3 ilman vuoden takaista havaintoa ja ilman kvartaalia.

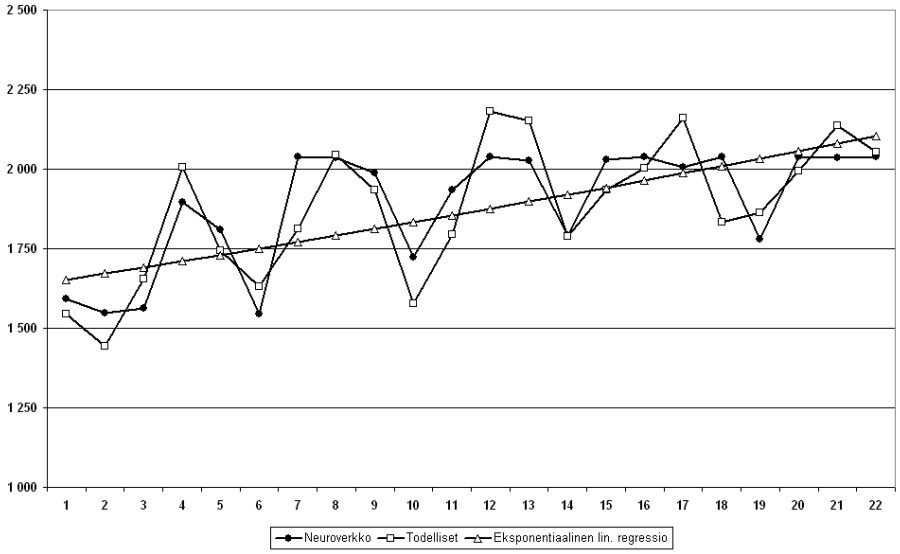
Selitysasteet: Malli $2 = 0,834$ Malli $3 = 0,721$

Kuvissa 4.5 ja 4.6 on mallien tulokset esitetty graafisesti. Selvästi mallin laatu huononee, kun kvartaali (vastaa ajan kulumista) jätetään pois. Toisaalta vuoden takainen havainto - eli vastaavalta vuosikvartaalilta mutta edelliseltä vuodelta kuin ennustettava arvo - ei paranna ennustetta. Ensimmäisen mallin ja Malli2:n välinen ero on todella pieni, eli vuoden takaisesta havainnosta ei ole hyötyä.

Testataan eri sisääntulojen merkitys poistamalla mallista aina yksi niistä kerrallaan ja tarkastelemalla selitysasteita.



Kuva 4.5: Keskivahinkojen ennusteet Malli2



Kuva 4.6: Keskivahinkojen ennusteet Malli3

Poistettu sisääntulo	Mallin selitysaste
Kvartaali	0,793
Keskivahinko $t-1$	0,818
Keskivahinko $t-2$	0,724
Keskivahinko $t-3$	0,826
Keskivahinko $t-4$	0,834

Selitysaste-tarkastelun perusteella eri sisääntulot voidaan laittaa tärkeysjärjestykseen: keskivahinko $t-2$, kvartaali, keskivahinko $t-1$, keskivahinko $t-3$ ja keskivahinko $t-4$. Samansuuntaisia tuloksia saadaan myös tarkastelemalla painokertoimien arvoja. Kvartaalilla ja keskivahingolla $t-3$ on itseisarvoiltaan suurimmat arvot ja keskivahingolla $t-4$ kerroin on lähellä arvoa 0.

4.2 Korvausten ennustaminen ja varaaminen

Seuraava esimerkki neuroverkkojen mahdollisesta soveltamisesta vakuutuksissa on keskivahinko-esimerkkiä monimutkaisempi. Perinteisesti tulevia korvauksia on arvioitu erilaisilla Chain ladder -tyyppisillä menetelmillä. Chain ladder -menetelmästä on hyvin lyhyesti seuraavassa kappaleessa. Menetelmässä vahinkovuosi (tai -kvartaali tms) ja korvausvuosi ovat merkityksellisimpiä tekijöitä. Neuroverkkomallissa tulee helposti houkutus valita mukaan myös muita tekijöitä. Seuraavassa esimerkissä on vahinko- ja korvaushetkien lisäksi valittu selittäviksi tekijöiksi kuluttajahintaindeksi, vakuutuskannan eri ajoneuvotyyppien osuudet vahinkohetkellä, korvausviive kuukausina ja vuosikvartaaleina sekä koko vakuutuskannan ajoneuvojen lukumäärä. Vahinko- ja korvaushetkistä on tiedossa vuosi, kvartaali ja kuukausi. Aineistona on käytetty yritysten kaskovakuutusten vahinkoja.

Chain ladder -menetelmä

Merkitään C_{ij} vuoden i vahingoista kehitysvuonna j maksettua korvausta. Silloin

$$D_{ij} = \sum_{k=1}^{j} C_{ik}$$

on vuoden i vahingoista kehitysvuoteen j mennessä kumuloitunut maksettujen korvausten summa.

Maksettujen korvausten kertymistä kuvaavat Chain-ladder–kertoimet

$$\lambda_{ij} = \frac{D_{i,\ j+1}}{D_{ij}}.$$

Kertoimien odotus- ja keskiarvot voidaan laskea esimerkiksi seuraavilla kaavoilla

$$\lambda_j \quad = \quad \frac{\sum_{i=1}^{n-j} D_{i,\ j+1}}{\sum_{i=1}^{n-j} D_{ij}} \quad \text{ja}$$

$$\lambda_j = \frac{\sum_{i=1}^{n-j} \lambda_{ij}}{n-j}.$$

Näiden kertoimien avulla voidaan arvioida lopullista korvausmenoa (kehitysvuoden u lopussa):

$$D_{iu} = D_{ij} \prod_{k=j}^{u} \lambda_k.$$

Neuroverkkomalli

Kun neuroverkkoja sovelletaan jäljellä olevien korvausten estimoimiseen ja ennustamiseen, käytetään kirjallisuuden mukaan datana yleensä vuosittaisia tms. maksettuja korvausten lisäyksiä, ei kumuloituja korvaussummia. Maksetut korvausten lisäykset yleensä ensin kasvavat, mutta ennemmin tai myöhemmin ne pienenevät ja lopulta loppuvat kokonaan, kun kaikki vahingot on korvattu loppuun. Myös seuraavassa esimerkissä käytettiin ensin maksettujen korvausten lisäyksiä.

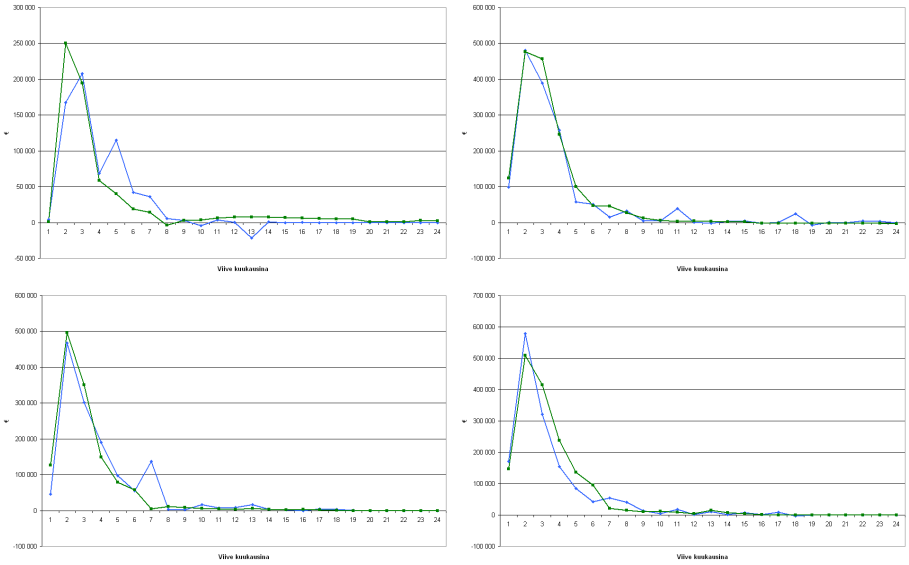
Autovakuutusten korvaukset maksetaan yleensä loppuun nopeasti. Silloin tällöin vahinko- ja korvauksen maksuhetkellä on kuitenkin pitempi väli. Näin ollen korvausaineisto on aika harva kun viive vahingon ja korvauksen välillä on suurempi. Jos esimerkiksi jollekin tietylle vahinkokuukausi-korvauskuukausi –parilla ei ole yhtään maksettuja korvauksia (ts. havainto puuttuu), on neuroverkon opetusaineistoa täydennettävä havainnolla jossa maksetut korvaukset ovat 0 euroa. Muussa tapauksessa neuroverkkoa opetetaan vain sellaisilla tapauksilla, joissa ko. olevalla vahinkokuukausi-korvauskuukausi –pareilla on jotain maksettuja korvauksia tai korvausten vähennyksiä. Kun korvausviive on luokkaa 24 kuukautta, alkavat korvauslisäykset olla jo harvinaisia autovakuutuksesa.

Neuroverkkomallia varten muodostettiin ensin opetusaineisto, jossa olivat vahinkokuukauden yhteenlasketut korvaukset korvauksen maksamiskuukausittain. Lisäksi havaintoihin oli liitetty vahinkokuukauteen liittyviä muita selittäviä tekijöitä, kuten ajoneuvojen lukumääriä, indeksejä, jne.

Muodostettu neuroverkkomalli oppi aineiston hyvin. Oheisissa kuvissa 4.7 on esimerkkejä todellisista ja mallinnetuista korvausten kehittymisistä. Neuroverkon selitysaste oli 0,883 eli aika hyvä. Mallinnettujen korvausten yhteismäärä 2000-2007 oli noin 2% pienempi kuin aineiston todelliset korvaukset.

Seuraavassa esimerkissä neuroverkon mallia on paranneltu niin, että se soveltuisi paremmin ennustamiseen ja lisäksi ennustettavana muuttujana on käytetty kumuloituja korvauksia. Joka kuukausi on selvillä se, kuinka paljon tähän hetkeen mennessä on maksettu korvauksia. Neuroverkkomallin pitäisi ennustaa seuraavan kuukauden maksetut korvaukset. Näin saatua ennustetta voidaan käyttää taas sitä seuraavan kuukauden korvausten ennustamiseen, jne.

Neuroverkon sisääntuloina oli tässä uudestaan muodostetussa mallissa vahinkovuosi ja -kuukausi, korvauksen maksamisvuosi ja -kuukausi, korvauksen maksamisviive kuukausissa, edellisen kuukauden kumuloidut korvaukset, hintaindeksi, eri ajoneuvotyyppien osuudet vakuutuskannassa ja kannan ajoneuvojen lukumäärä. Yhteensä sisääntuloja oli 26 kappaletta ja ulostuloja siis yksi eli



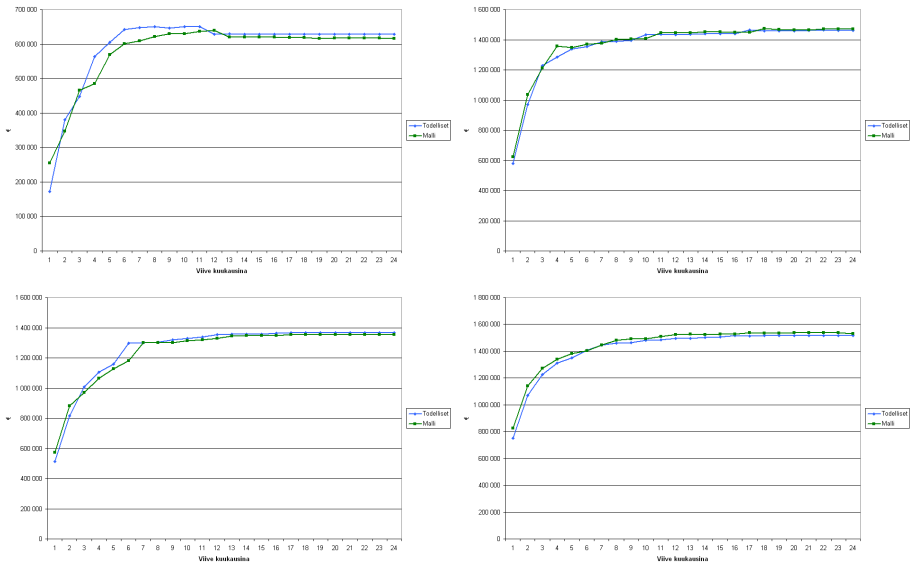
Kuva 4.7: Esimerkkejä todellisista ja mallinnetuista kuukauden maksetuista korvauksista.

seuraavan kuukauden maksettujen korvausten ennuste. Piilokerroksessa oli 16 neuronia, joka ehkä oli hieman liikaakin. Opetetun neuroverkkomallin selitysaste R^2 oli 0,989 eli erittäin korkea. Kuvassa 4.8 on esimerkkejä todellisista ja mallinnetuista maksetuista, kumuloiduista korvauksista. Kannattaa huomata, että kuvaajat alkavat kohdasta viive = 1 kuukausi. Tämä johtuu siitä, että neuroverkko käyttää yhtenä sisääntulona aina edellisen kuukauden kumuloidut korvaukset –tietoa. Kuvaajat esittävät samojen vahinkovuosien ja -kuukausien vahinkojen kehittymistä kun kuvassa 4.7.

Opetettua neuroverkkoa käytettiin sitten korvausten ennustamiseen uudella datalla, joka ei ollut mukana opetusvaihessa. Ainakin tällä kerralla neuroverkko arvioi kumuloitujen korvausten kasvavan alussa nopeammin ja suuremmilla viivekuukausilla vähemmän kuin todellisuudessa tapahtui. Suhteellinen ero kumuloiduissa korvauksissa oli suurin vahinkokuukautta seuraavassa kuukaudessa, eli ensimmäisessä ennusteessa. Samalla euromääräiset erot olivat silloin kuitenkin pieniä.

Neuroverkkojen käyttäminen tulevien korvausten ennustamiseen toimii hyvin. Menetelmä voisi sopia esimerkiksi jonkinlaiseen ennustamiseen, suunnitteluun ja raportointiin. Malli on helppo muodostaa neuroverkko-ohjelmassa ja opetuksen tuloksia voi hyödyntää esimerkiksi taulukkolaskentaohjelmassa.

Malliin valittavat sisääntulot määräytyvät saatavilla olevan datan ja oman kokemuksen tai näkemyksen perusteella. Ulostulona on kirjallisuudessa pidetty maksettujen korvausten lisäystä parhaimpana vaihtoehtona, mutta kumuloidut korvaukset toimivat ainakin tässä esimerkissä vielä paremmin. Mallin suunnittelussa ei sisääntulojen ja ulostulojen määrittelyn lisäksi ole paljon päätettävää: piilokerrosten ja piilokerroksen neuronien lukumäärä ja eri opetuskertoimet. Tässä esimerkissä ei erilaisia piilokerrosten tai piilokerrosten neuronien lukumääriä ko-



Kuva 4.8: Esimerkkejä todellisista ja mallinnetuista kumuloiduista vahinkokuukauden korvauksista.

keiltu. Opetus vain tehtiin useamman kerran ja kertoimista valittiin parhaiten dataan sopiva. Yllättävästi jälkimmäisessä esimerkissä eri opetuskerroilla kertoimista tuli joka kerralla aika samanlaisia. Neuronien kertoimista ei kuitenkaan voi sanoa kuinka lähellä ne toisiaan todella ovat.

4.3 Vakuutusten hinnoittelu

Vakuutusten hinnoittelu on oleellinen ongelma vakuutusyhtiössä. Tavoitteena on etsiä sellainen maksutaso, joka vastaa riskiä. Tärkeä kysymys tässä on ”Mitkä tekijät tai riskin ominaisuudet ovat merkityksellisiä, kun ennustetaan vahingon sattumisen todennäköisyyttä tai vahingon määrää?”. Esimerkiksi auto- ja liikennevakuutuksessa on vahingon todennäköisyydellä ja suuren kaupungin tai taajaman läheisyydellä positiivinen korrelaatio. Asiakkaan postinumero voisi siten vaikuttaa hänen vakuutusmaksuunsa. Vaikka monet vakuutusmaksuun vaikuttavista tekijöistä ovat itsestään selviä, on paljon muita maksuun vaikuttavia tekijöitä joita ei näe suoraan ja varsinkin eri tekijöiden välisiä yhteyksiä. Uusilla tilastollisilla menetelmillä, kuten juuri neuroverkoilla, mahdollisten riskitekijöiden (yhteis)vaikutusta voidaan tarkastella, jolloin tariffit voidaan kehittää paremmin riskiä vastaaviksi.

Havaintoaineistoja voidaan jakaa sisäisesti homogeenisiin ryhmiin, jotka toisaalta eroavat toisistaan jossakin mielessä, mutta samalla ryhmien sisällä eroavaisuudet ovat mahdollisimman pieniä. Jokaista ryhmää erikseen voidaan sitten analysoida ja mallintaa. Usein muodostuvat mallit eroavat selvästi ryhmien kesken ja tulkinta on helpompaa ja tarkempaa kuin jos yritettäisiin mallintaa koko joukkoa yhdellä kerralla. Vakuutusyhtiö voi esimerkiksi huomata, että 18-20 -

vuotiaiden liikennevakuutusasiakkaiden joukossa on ryhmä, jolla vahinkotiheys on selvästi muuta 18-20 -vuotiaiden ryhmää pienempi.

Vakuutuksen hinnoittelun lähtökohtana on riskimaksu eli vahingon odotusarvo vakuutuksesta. Tuote, jota vakuutusyhtiö myy, on korvaus asiakkaalle vahingon sattuessa. Lopullinen maksu saadaan, kun riskimaksuun lisätään hoitokulu-, varmuus- ja muut kuormitukset. Neuroverkkoja voidaan käyttää hyväksi riskimaksujen määrittelemisessä. Hoitokulut taas riippuvat vakuutusyhtiön kulurakenteesta, mutta neuroverkkojen avulla asiakkaita voidaan esimerkiksi ryhmitellä asiakaskäyntien ja -yhteydenottojen, vahinkojen ja vakuutuskäsittelyn kustannusten yms. mukaan. Tällaisen ryhmittelyn avulla kulukuormitus voitaisiin sitten jakaa mahdollisimman oikeudenmukaisesti eri asiakkaille.

Riskimaksun määrittelemisessä on kaksi lähestymistapaa: määritellään erikseen odotusarvot vahinkotiheydelle ja korvauksen koolle jolloin riskimaksu on näiden kahden odotusarvon tulo tai toinen vaihtoehto on määritellä suoraan riskimaksu.

Neuroverkon voi periaatteessa opettaa estimoimaan keskivahinkoa tai vahinkotiheyttä vakuutusten tietojen pohjalta: alue, sylinteritilavuus, paino, ikä, asiakkaan segmentti, jne. Tiedot voivat olla binaarisia, jatkuva-arvoisia tai kategorisia. Kategoriset muuttujat pitää tosin koodata usemmaksi binaariseksi muuttujaksi. Esimerkiksi tilastollinen, aktuaareille käytännössä paljon neuroverkkoja tutumpi GLM-algoritmi taas vaatii, että muuttujat ovat kategorisia. Toisin sanoen, jatkuva-arvoisten muuttujien vaihteluvälit pitää jakaa jollakin tavalla luokkiin.

Normaalin, monikerros-perceptron–verkon (MLP) käyttäminen keskivahingon tai vahinkotiheyden mallintamiseen onnistuu samalla tavalla kuin aikaisemmassa esimerkissä keskivahingon ennustamisesta. Opetusdata voisi olla joko vakuutus- (vahinkokohtaista) tai ryhmiteltyä. Ryhmitellyn ja summatun datan tapauksessa voi tulla ongelmaksi joidenkin muuttujien jatkuvien arvojen jakaminen luokiksi. Ilmeisesti kumpaakin lähestymistapaa olisi kokeiltava. Tarvittavia sisääntuloja tietysti on useampia ja verkon ulostulona on joko keskivahinko tai vahinkotiheys. Neuroverkon avulla on tällä tavalla mahdollista hinnoitella vakuutuksia, mutta huonoja puolia tässä lähestymistavassa on ainakin seuraavat:

- Neuroverkon "black-box" -ominaisuus: Eri tekijöiden vaikutusta hintaan voi olla vaikea ymmärtää.
- Eri tekijöiden yhteisvaikutusten huomioiminen voi antaa riskiopillisesti oikeamman hinnan, mutta se voi olla muusta syystä (kuten markkinoiden vaatimukset) vaikea perustella. On tutkittu, että ihminen voi ottaa huomioon yhtä aikaa viisi eri tekijää. Neuroverkot voivat huomioida samanaikaisesti kymmenien tai satojen tekijöiden yhteisvaikutuksen. Jotkin vaikutuksista voivat tuntua epäloogisilta ja niitä voi olla mahdotonta perustella asiakkaalle.
- Neuroverkon yli-opettamisen–vaara: Opetettu neuroverkko sopii hyvin opetukseen käytettyyn dataan, mutta se toimii huonosti uusien vakuutusten hinnoittelussa.

Esimerkiksi neuroverkon yliopettamisen voi välttää verkon oikealla suunnittelulla, riittävällä opetusaineistolla ja testaamisella. Neuroverkoissa on hyviäkin puolia kuten muuttujien väliset yhteydet ja vaikutus hintaan muodostuvat opetusvaiheessa automaattisesti eli mitään ennakkotietoa funktiomuodosta tms. ei tarvita, neuroverkot voivat tehdä aineistosta tarkempia malleja ja muuttujien vaikutukset hintaan voivat olla myös epälineaarisia. Monet tilastolliset algoritmit pohjautuvat vaikutusten lineaarisuuteen. Vaikka muuttujille olisikin tehty joku ei-lineaarinen muutos, se yleensä vaatii ennakkotietoa siitä minkälainen funktiomuutos tarvitaan.

Samat hankaluudet (ja edut) kuin hinnoittelussa MLP-verkon avulla koskevat myös SOM-verkkoja. SOM-verkon voi opettaa luokittelemaan vakuutukset ja jokaiseen luokkaan voisi yhdistää määrätyn maksun.

Käytännöllisin tapa hyödyntää neuroverkkoja vakuutusten hinnoittelussa voisi kuitenkin olla niiden käyttäminen tariffiluokkien määrittelyssä. SOM-verkkoja voi käyttää tariffitekijöiden koko jatkuvan vaihteluvälin jakamiseen erillisiin luokkiin. SOM-algoritmin paljon kehuttu ominaisuus säilyttää havaintojen topologia myös lopullisessa verkossa on ihanteellinen vakuutusten hinnoittelun kannalta. Jos SOM-verkko opetetaan luokittelemaan esimerkiksi ajoneuvojen ikä, se tyypillisesti luokittelee iät niin että ne muodostavat jatkuvia välejä (esimerkiksi -1-0 vuotta, 1-5 vuotta, 6-15 vuotta, jne).

Esimerkki tariffimuuttujan "ikä" jakamisesta tariffiluokkiin

Seuraavassa esimerkissä SOM-verkon avulla pyritään jakamaan yhden tariffitekijän pieniä perusluokkia isommiksi tariffiluokiksi. Tariffiluokkien pitää muodostua ikä-muuttujan jatkuvista väleistä, esimerkiksi 0-2 vuotta, 3-5 vuotta, jne. SOM-verkoilla on tähän sopiva ominaisuus, että opetusaineiston lähekkäiset havainnot kuvautuvat valmiissa SOM-kartassakin läheisiin neuroneihin. Tariffiluokkien muodostamista voidaan pitää ryhmittelytehtävänä, johon SOM-verkot sopivat hyvin.

Esimerkin aineisto muodostuu ajoneuvojen ikien muodostamista perusluokista. Ikä on määritelty vuoden tarkkuudella. Jokaisesta ikä-perusluokasta on tiedossa kyseisten ajoneuvojen vahinkotiheys ja osuus kannasta tai tarkemmin sanottuna osuus ajoneuvovuosista. Yhden vuoden verran voimassa ollut autovakuutus yhdelle autolle muodostaa yhden ajoneuvovuoden. Osuuden tilalla voitaisiin käyttää myös suoraan ajoneuvovuosia. Tavoitteena esimerkissä on jakaa ikämuuttujan perusluokat vahinkotiheyden mukaan tariffiluokkiin. Koska ajoneuvojen määrät eri perusluokissa ovat hyvinkin erilaisia, ei perusluokkien vahinkotiheyksiä voida verrata suoraan tavallisilla samankaltaisuus- ja erilaisuusmitoilla. Pienimpien luokkien vahinkotiheydet ovat melko sattumanvaraisia ja niitä ei voi pitää kovin luotettavina. Niinpä aktuaarikirjallisuudessa on esitetty ryhmittelymenetelmiä, joilla huomioidaan myös perusluokkien erilaiset riskimäärät (esimerkissä ajoneuvovuodet). Esimerkiksi K. Loimaranta, J. Jacobsson ja H. Lonka ovat esittäneet tällaisen menetelmän ([7]).

Taulukossa 4.1 on tietyn kaskovakuutusten ryhmän ajoneuvovuosien määrä (prosenttiosuudet) ja vahinkotiheyksiä ajoneuvon iän mukaan jaettuna. Taulukossa on iät 0-28 vuotta, mutta SOM-verkon opetuksessa käytettiin kaikkia ajoneu-

von ikiä 0-50 vuotta. Korkeimmat iät ovat todennäköisesti vakuutusten tiedoissa olevia virheitä.

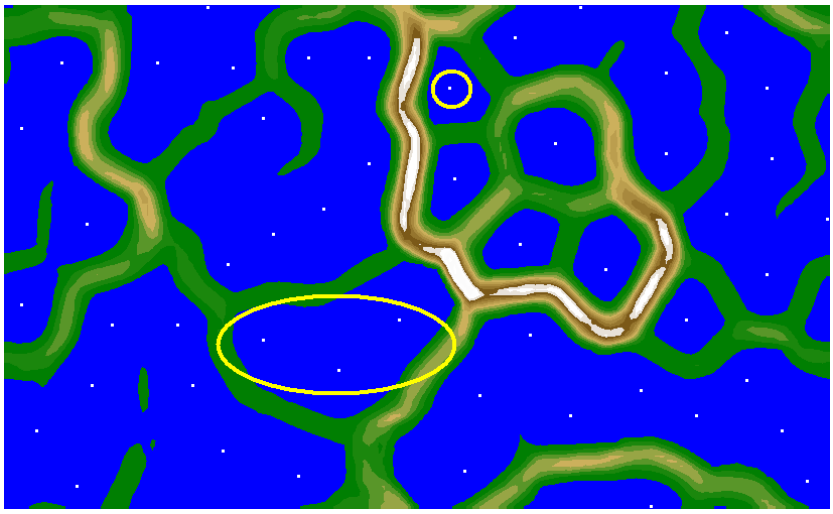
Taulukko 4.1: Vahinkotiheydet ja osuus kannassa ajoneuvojen ikien mukaan.

Ikä	Vahinkotiheys %	Osuus kannasta %
0	29	9
1	24	20
2	33	17
3	35	13
4	29	8
5	26	5
6	25	4
7	23	4
8	23	3
9	17	3
10	16	2
11	16	2
12	17	1
13	16	1
14	15	1
15	10	1
16	8	1
17	11	1
18	6	1
19	5	1
20	4	0
21	5	0
22	6	0
23	2	0
24	0	0
25	3	0
26	0	0
27	6	0
28	0	0
...	...	...

Perusluokkien ryhmittelyn voisi joissakin yksinkertaisissa tapauksissa tehdä yhdellä SOM-verkolla, jossa sisääntuloina olisivat ikä ja vahinkotiheys. Ongelma tulee siitä, että joissakin perusluokissa on hyvin vähän ajoneuvoja. SOM-verkossa käytettävän etäisyysmitan pitäisi silloin olla sellainen, että kun osuus kannasta on pieni, on myös vahinkotiheydellä pieni merkitys etäisyysmitassa (ja iällä suuri merkitys). Näin siksi, että pienten kannanosien vahinkotiheydet aineistossa voivat erota oikeista vahinkotiheyksistä paljonkin. Jos ei voida määritellä uutta etäisyysmittaa, olisi perusluokkia yhdisteltävä niin että ne olisivat yhtä suuria. Joissakin tapauksista se voisi olla helppo tehtävä, monessa muussa tapauksessa luokkien rajojen määrääminen voisi olla mielivaltaista.

Tässä esimerkissä otetaan kuitenkin erilainen lähestymistapa. Ikä-muuttujan

merkitystä tariffiluokkien määrittelyssä painotetaan muodostamalla luokat muodostava SOM-verkko kahdessa vaiheessa. Samalla yritetään varmistaa, että a) iät tulevat jaetuiksi jatkuviin väleihin ja että b) pienimpien luokkien vahinkotiheydet eivät saa liian suurta merkitystä. Ensimmäisessä vaiheessa muodostetaan kaksi erillistä SOM-verkkoa. Ensimmäisessä verkossa sisääntuloina ovat ajoneuvon ikä ja ko. ikäisten ajoneuvojen suhteellinen osuus kannassa. Toisessa verkossa sisääntuloina ovat perusluokkien vahinkotiheys ja osuus kannassa. Näin jako tariffiluokkiin tehdään ensimmäisessä vaiheessa kahdella tavalla: iän ja vahinkotiheyden mukaan. Suhteellista osuutta kannasta tarvitaan aiemmin mainitusta syystä. Muodostettavien luokkien lukumääräksi tuli 11 molemmassa SOM-verkossa. SOM-verkkojen tuloksia tulkittiin U-matriisien avulla. Kuvassa 4.9 on U-matriisikuva SOM-verkosta, jossa sisääntuloina olivat ikä ja osuus kannasta. Luokkien lukumäärät päätetään vasta kun SOM-verkon tulokset visualisoidaan. Kuvaan on merkitty esimerkkinä ryhmät 41-43 vuotta vanhat sekä 4 vuoden vanhat autot. Kannattaa huomata, että U-matriisien kuvat ovat rajattomia eli esimerkiksi alareunan jälkeen ne jatkuvat taas kuvan yläreunasta. U-matriisissa aineiston havainnot ovat kuvautuneet valkoisella pisteellä merkittyihin ns. bestmatch- eli voittajaneuroneihin. Voittajaneuroni on se verkon neuroni, jonka etäisyys aineiston havainnosta on pienin. Yhteen voittajaneuroniin kuvautuu yleensä useampi havainto. Maa-alue (vihreä väri) ympäröi samaan luokkaan kuuluvia neuroneita. Kuvan tulkinnassa pitää ottaa huomioon, että esimerkiksi oikean reunan katkennut sininen alue jatkuu kuvan vasemmasta reunasta.



Kuva 4.9: Visualisoitu U-matriisi ajoneuvon ikä –muuttujan tapauksessa. Aineiston havainnot ovat kuvautuneet valkoisella merkittyihin SOM-kartan neuroneihin. Esimerkkinä on rajattu viivoilla ryhmät ”41-43 vuotta vanhat” ja ”4 vuotta vanhat”.

Ensimmäisen vaiheen SOM-verkkojen tuloksena saatiin jokaiselle ajoneuvon ikä -perusluokalle ikä- ja vahinkotiheysluokat taulukoiden 4.2 ja 4.3 mukaisesti. Vahinkotiheysluokittelussa kahdessa viimeisessä luokassa iät menevät lomittain.

Taulukko 4.2: Perusluokkien jako iän ja suhteellisen osuuden mukaan.

Luokka	Ajoneuvon ikä	Osuus kannasta %	Vahinkotiheys keskimäärin %
1	0	8,8	29
2	1	19,5	24
3	2	17,4	33
4	3	13,1	35
5	4	7,9	29
6	5-9	19,7	23
7	10-19	11,4	13
8	20	0,5	4
9	21-40	1,5	3
10	41-43	0,1	0
11	44-50	0,2	0

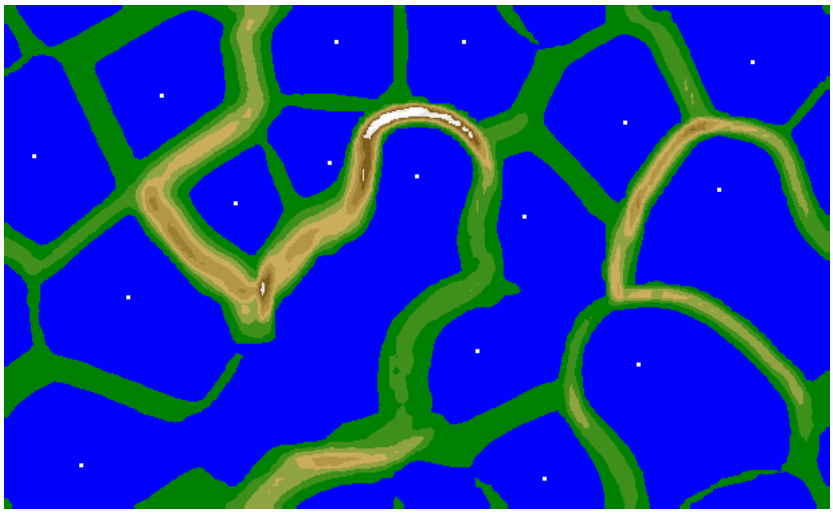
Taulukko 4.3: Perusluokkien jako vahinkotiheyden ja suhteellisen osuuden mukaan.

Luokka	Ajoneuvon ikä	Osuus kannasta %	Vahinkotiheys keskimäärin %
1	0	8,8	29
2	1	19,5	24
3	2	17,4	33
4	3	13,1	35
5	4	7,9	29
6	5-6	9,5	25
7	7	4,0	23
8	8	3,4	23
9	9-14	9,4	16
10	15, 17	2,0	11
11	16, 18-50	5,0	5

Seuraavassa vaiheessa kaksi edellä saatua luokittelua yhdistetään.

Toisessa vaiheessa muodostetaan kolmas SOM-verkko, jonka sisääntuloina ovat ensimmäisen vaiheen verkkojen antamat ikä- ja vahinkotiheysluokat. SOM-verkon ominaisuus topologian säilyttämisestä käy ilmi tuloksista. SOM-verkot ovat luokitelleet aineiston ajoneuvon iän jatkuviin luokkiin ja toisiaan lähellä olevat havainnot ovat lopullisessa mallissa lähellä toisiaan. Aineisto on luokiteltu lopulta 9 luokkaan (kuva 4.10). Kuvassa on merkitty valkoisilla pisteillä voittajaneuronit.

Taulukossa 4.4 on esitetty lopulliset ajoneuvojen ikä –luokat sekä luokkien keskimääräiset vahinkotiheydet ja osuudet kannasta.



Kuva 4.10: U-matriisi. Ikä-tariffimuuttujan lopulliset tariffiluokat.

Taulukko 4.4: Lopulliset ikä–muuttujan tariffiluokat

Luokka	Ajoneuvon ikä	Osuus kannasta %	Vahinkotiheys %
1	0	8,8	29
2	1	19,5	24
3	2-4	38,4	33
4	5-6	9,5	25
5	7-8	7,3	23
6	9	2,8	17
7	10-19	11,4	13
8	20-43	2,0	3
9	44-50	0,2	0

4.4 Asiakkaiden hankinta

Toinen tärkeä tehtävä on asiakkaiden hankinta. Perinteinen tapa on lisätä panostuksia mainontaan ja jakelukanaviin. Myös vakuutusten hintaa voidaan laskea enemmän tai vähemmän kaikilla mahdollisilla asiakkailla. Neuroverkkojen mallien avuilla panostukset voidaan suunnata juuri niille asiakkaille ja alueille, joista saadaan mahdollisimman suuri teho. Myös asiakaskannattavuudessa voi olla suuria eroja eri asiakasryhmien välillä. Ei ole samantekevää millaiset asiakkaat yrityksen vakuutuksia ostavat.

Sen sijaan, että mainonta kohdistettaisiin kaikille tietyt ehdot täyttäville asiakkaille, voidaan mainonnan tehoja parantaa kohdistamalla lisää resursseja oikeille asiakkaille. Valittua asiakassegmenttiä voidaan rajata esimerkiksi ottamalla mukaan vain ne, jotka todennäköisimmin ottaisivat vakuutuksen. Tätä ryhmää voidaan edelleen rajata ottamalla mukaan kamppanjaan vain ne asiakkaat, jotka olisivat uskollisimpia ja joiden asiakkuus todennäköisesti kestäisi esimerkiksi vähintään 5 vuotta.

Panostukset ja markkinointikeinot eri asiakasluokissa voitaisiin optimoida niin, että saatava hyöty rajallisista resursseista olisi mahdollisimman hyvä.

4.5 Asiakkaiden pitäminen ja asiakkaiden segmentointi

Asiakkaiden pitäminen yrityksen asiakkaina tulevaisuudessakin liittyy asiakkaiden hankintaan. Kannattavista asiakkaista kannattaa pitää kiinni ainakin tiettyyn pisteeseen asti. Asiakkaiden uushankinnan kulut ovat joskus suuria ja kaikista uusista asiakkaista ei lopulta saada läheskään kannattavia. Kokemukset ovat osoittaneet, että asiakas jolla on kaksi vakuutusta pysyy asiakkaana todennäköisemmin kuin jos hänellä olisi vain yksi vakuutus. Samalla tavalla asiakas, jolla on kolme vakuutusta, pysyy todennäköisemmin tulevaisuudessakin asiakkaana kuin jos hänellä olisi vakuutuksia vähemmän kuin kolme. Paketoimalla useita vakuutuksia yhteen ja esimerkiksi antamalla siitä jotain etuja (alennuksia yms) asiakkaalle, voidaan asiakasuskollisuutta parantaa.

Tietojen analysointi on tehtävä asiakastasolla. Pelkät summatason luvut eri asiakassegmenteistä eivät riitä.

Kysymyksiä johon tällaiset analyysit voivat vastata ovat esimerkiksi:

- Mitkä vakuutukset asiakkaat olisivat valmiita hankkimaan yhdessä kimpussa?
- Miten koko asiakasjoukko pitäisi segmentoida?
- Minkälaisia uusia vakuutuksia ja mitä hintoja kannattaisi tarjota jollekin tietylle asiakasryhmälle?
- Miten eri asiakassegmentit eroavat kannattavuudeltaan ja pitämiltään vakuutuksiltaan?
- Kuinka moni uusi autovakuutusasiakas hankkii vuoden sisällä myös kotivakuutuksen?

Asiakassegmenttien analysoinnilla voidaan myös kohdistaa huomio kannattaviin asiakkaisiin, jotka ovat todennäköisimmin vaihtamassa vakuutusyhtiötä. Heille voitaisiin tehdä enemmän räätälöityjä ratkaisuja. Tai ehkä samalla huomataan uusia vakuutustarpeita, joita voidaan käyttää sekä asiakkaiden pitämisessä että uusien kannattavien asiakkaiden hankkimisessa. Logistista regressioanalyysia on perinteisesti käytetty löytämään ne asiakasryhmät, joilla on suurin todennäköisyys vaihtaa vakuutusyhtiötä. Neuroverkot ottavat enemmän huomioon eri tekijöiden välisiä yhteyksiä ja yhteydet voivat olla myös epälineaarisia luonteeltaan.

Pienet yritykset voivat usein käsitellä asiakkaitaan enemmän yksilöllisesti kuin suuret yritykset. Kun yrityksen koko kasvaa, markkinointiosasto alkaa helposti pohtia enemmän vakuutusten yms. kehittelyä sen sijaan että asiakassuhteita kehitettäisiin ja ylläpidettäisiin. Sen sijaan, että mietittäisiin miten nykyisten

asiakkaiden tarpeita voitaisiin palvella paremmin, on suuri houkutus yrittää kehitellä uusia massatuotteita uusille asiakkaille. Vakuutusmarkkinat ovat jo melko vakiintuneet ja kokonaan uusia markkina-alueita voi olla vaikea löytää. Silloin asiakkaiden hankkiminen ja menettäminen on vakuutusyritysten nollasummapeliä. Näin ollen yleisen, kohdistamattoman mainonnon teho jää vähäiseksi. Erilaisilla uusilla analyyseille voidaan mainontaa kohdistaa haluttuihin, pysyviin asiakkaisiin tai heille voidaan muodostaa vaikka uskollisuusohjelmia. Kaiken kaikkiaan yleisen mainonnan sijasta on mainontaa ja muuta viestintää kohdistettava tarkemmin asiakkaisiin.

Neuroverkkoanalyyseillä voidaan tutkia mitä asiakkaat haluavat ja millaisia vakuutustarpeita heillä voi olla. Erilaisten tuotepakettien muodostaminen on myös mahdollista analyysien perusteella. Tarkkarajaisesta vakuutustuotenäkökulmasta ollaan siirtymässä asiakasnäkökulmaan.

Nykyinen tietotekniikka mahdollistaa suuren tietomäärän keräämisen asiakkaista: mitä vakuutustuotteita kullakin on, kuinka kauan asiakkuus on kestänyt, vahinkojen määrä, riskien maantieteellinen sijainti ja paljon muuta. Tähän tietoon voidaan lisätä rinnalle myös muita mittareita, kuten kunnan autovarkauksien lukumäärä, asukas- tai liikennetiheys, joilla voi olla käyttöä tehtäessä vakuutustariffeja.

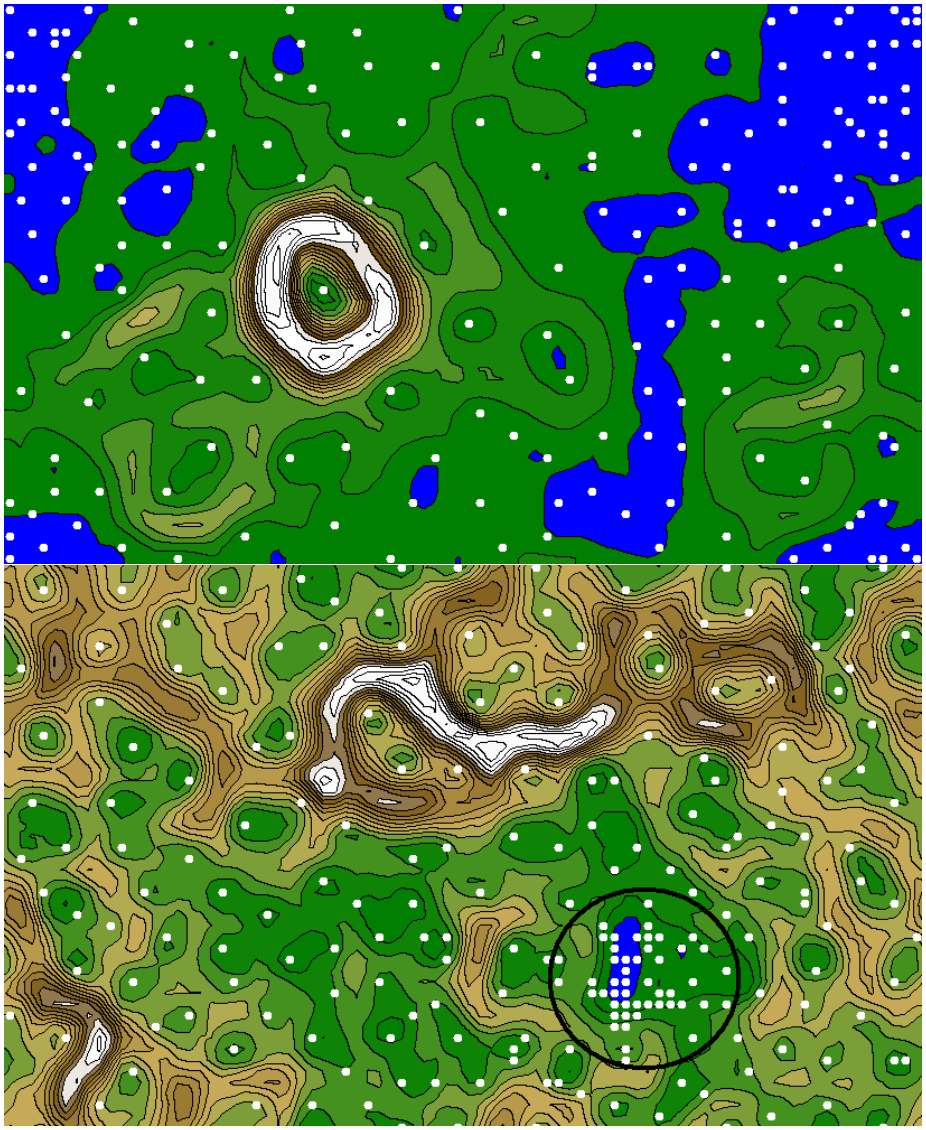
Vakuutuskantaa voidaan analysoida myös vakuutusmyyjien kautta: millaista kantaa kullakin on, miten kannat eroavat toisistaan ja onko jossain esimerkiksi lisämyynnin mahdollisuuksia.

Kuvissa 4.11 yhteyshenkilöitä on jaettu luokkiin SOM-kartan avulla. Jokaisesta yhteyshenkilön kannasta on aineistossa ollut mukana esimerkiksi erityyppisten ajoneuvojen lukumääriä auto- tai liikennevakuutuksissa ja asiakkaiden lukumäärät eri asiakassegmenteissä. Yhteensä muuttujia oli 107 jokaista yhteyshenkilöä kohti.

Ensimmäisestä kuvasta erottuu selvästi yksi yhteyshenkilönumero "valkoisten vuorten" ympäröimänä. Se poikkeaa muista, koska se ei liity yhteen yhteyshenkilöön vaan on suuremman joukon yhteinen yhteyshenkilönumero. Seuraavassa kuvassa SOM-verkko on opetettu uudelleen ilman kyseistä yhteyshenkilönumeroa. Kuvasta erottuu yksi suurempi ryhmä keskenään jossakin mielessä samanlaisia yhteyshenkilöitä. Suuri osa kyseisestä ryhmästä kuuluu samaan yhteyshenkilöiden organisaatioon, mutta ei kaikki. Kartan tulkinnasta tulee mielenkiintoista, kun alkaa pohtia miksi joku yhteyshenkilö kuuluu kartalle toiseen ryhmään kuin organisaation perusteella voisi kuvitella, jne. Karttaa ja yhteyshenkilöiden paikkaa kartalla tutkimalla löytyy esimerkiksi yhteyshenkilöitä joilla on selvästi tietynlainen vakuutuskanta, sellaiset yhteyshenkilöt joilla on hyvin pieni autovakuutuskanta tai vain tietynmuotoisia vakuutuksia.

4.6 Vakuuttaminen ja riskinvalinta

Vakuutusmarkkinat muuttuvat ajan myötä ja niin muuttuvat myös asiakkaiden tarvitsemat vakuutukset. On erittäin tärkeää, että vakuutusyhtiö seuraa ja huomaa muutokset asiakkaiden tarpeissa, jotta vakuutuksia voidaan edelleen myydä.



Kuva 4.11: a) Yksi erottuva havainto, b) Sama aineisto ilman erottuvaa havaintoa

Riskinvalinta (underwriting) on prosessi, jossa vakuuttaja määrittelee mitä riskejä se ottaa kannettavakseen, riskimäärät ja ehdot. Tavoitteena on saada vakuutuskanta, jossa on keskimäärin turvallinen mutta kannattava riski- ja vakuutusmaksutaso. Esimerkiksi Vaughn *et al.* [12] käyttivät MLP-verkkoa henkivakuutusta hakevien asiakkaiden luokitteluun standardi- ja ei-standardi – riskeihin. Standardi-riskejä voi vakuuttaa kuka tahansa vakuutusmyyjä, mutta ei-standardien riskien vakuuttamiseen vaaditaan tarkempaa tietoa kohteesta ja asiantuntijan kokemusta riskeistä ja ehkä myös enemmän valtuuksia.

Yhtiö voi analysoida mitkä ovat kannattavimmat vakuutustuotteet ja mille asiakasryhmille, ja sitten antaa niille etusija markkinoinnissa. Toisaalta yleensä ei ole olemassa sellaista asiakasta, jolle ei voitaisi myydä vakuutusta sopivilla hinnoilla ja ehdoilla. Ongelma voi tulla siitä, että yhtiö ei onnistu myymään vakuutusta oikealla hinnalla oikealle asiakkaalle oikeaan aikaan. Esimerkiksi kannattavin asiakassegmentti voi olla hieman suuremman vahinkoriskin omaavat asiakkaat, mutta joille myytävien vakuutusten hinnatkin ovat korkeammat. Tai se voi olla jokin ryhmä, joka todennäköisesti samalla hankkii myös joukon muita vakuutuksia. Tai jotkin asiakkaat voivat olla selvästi muita kannattavimpia kun seurataan kokonaista monta vuotta kestävää asiakkuutta. Kaikki nämä erilaiset näkökulmat vaativat erilaista analyysia ja erilaisia kannattavuusmittareita.

Vakuutusyhtiö voi olla myös kiinnostunut seuraavanlaisista tuotteisiin, asiakkaisiin ja kannattavuuteen liittyvistä kysymyksistä:

- Mitkä ovat uusista mahdollisista vakuutustuotteista kannattavimmat?
- Mille vakuutuksille olisi suurimmat markkinat?
- Mikä uusi vakuutustuote olisi markkinoille helpoin omaksua?

Kun tietyt uudet tuotteet on arvioitu kannattavimmiksi, niiden tuontia markkinoille voidaan analyysien avulla priorisoida. Tämä järjestykseen asettaminen voi perustua monelle eri mittarille, kuten odotettavissa oleva kannattavuus, odotettavissa olevat uudet asiakkaat ja/tai odotettavissa oleva läpimeno markkinoille. Neuroverkkoja voidaan käyttää hyväksi, kun eri asiakassegmentit erotellaan ja niiden käyttäytymistä ennustetaan, lasketaan arvioita markkinointikampanjoiden vaikutuksista ja lopulta verrataan erilaisia strategioita.

4.7 Vakuutusvilpin havaitseminen

Suurempi tai pienempi vakuutusvilppi on ollut aina vakuutusyhtiöiden vaivana. Tehokas vakutuusvilpin vähentäminen usein johtaa selvään kannattavuuden parantamiseen. Lisäksi asiakkaiden kohtelu ja vakuutusten hinnoittelu tulevat oikeudenmukaisemmiksi. USAssa arvioidaan vakuutusvilpin osuuden koko korvausmenosta olevan noin 10%. Yhtiöt, jotka onnistuvat välttämään vakuutusvilpin, voivat alentaa korvauskustannuksia ja tarjota kilpailukykyisempiä vakuutusten hintoja. Vakuutusvilpit eivät tyypillisesti ole kaikkein suurimpia vahinkoja, koska yleisesti tiedetään että isompia vahinkoja tarkastellaan tarkemmin. Vakuutusvilppiä epäillessä/etsiessä on haettava erikoisia yhteyksiä, epäselvyyksiä ja poikkeavuuksia havaintoaineistosta.

Vakuutusyhtiöille kertyy suuret määrät tietoa sattuneista vahingoista. Neuroverkkosovelluksia vakuutusvilpin havaitsemiseksi on raportoitu paljon, mutta sovellusten yksityiskohdat pidetään useimmiten piilossa. Syitä siihen on tietysti paljon. Jos vakuutusvilppiä haluttaisiin tutkia neuroverkoilla, olisi käytännössä kaksi lähestymistapaa. Saatavilla oleva aineisto voi rajata kumpaa tapaa olisi käytettävä. Ensimmäisessä tavassa käytössä olisi joukko vahinkoon liittyviä tietoja ja jostakin (vahingontarkastajalta, korvauskäsittelijältä) saatu tieto vakuutusvilpistä ko. vahingossa. Tässä tavassa neuroverkko opetettaisiin ohjatusti tunnistamaan vilpilliset vahingot tai liittämään vahinkoon todennäköisyyden vilpistä. Toisessa tavassa neuroverkko ryhmittelisi vahingot keskenään samankaltaisiin. Poikkeavat vahingot valittaisiin sitten lähempään tarkasteluun: Mikä niissä on erikoista ja miksi ne erottuvat massasta? Kummassakin tavassa neuroverkko on vain apuväline, joka seuloo koko vahinkomäärästä ne vahingot joita on tutkittava tarkemmin. Käytännössä asiantuntijoiden apua tarvitaan jo siinä vaiheessa, kun kysytään mitkä muuttujat ovat merkityksellisimpiä ongelman ratkaisuun. Turhat muuttujat neuroverkossa lisäävät vain kertoimien lukumäärää. Tietyn pisteen jälkeen merkityksellistenkin muuttujien lisääminen neuroverkkoon alkaa huonontaa saatavaa mallia. Yleensä havaintoaineisto on lisäksi harvaa eli havainnoissa on puuttuvia arvoja monien muuttujien kohdalla. Yhä uusien muuttujien sisällyttäminen malliin vaatii havaintomäärien huomattavaa kasvattamista. Muussa tapauksessa havaintoaineiston tuottama malli voi olla yhä kauempana todellisesta mallista.

Viaenen *et al* artikkelissa ([14]) käytettiin seuraavanlaisia binaarisia sisääntuloja neuroverkolle, jonka piti arvioida vahingon vilpillisyyttä:

- Ei poliisiraporttia onnettomuuspaikalta
- Vain yksi ajoneuvo osallisena
- Ei kunnollista selitystä onnettomuudelle
- Onnettomuuden syyllisellä vanha, vähäarvoinen auto
- Vuokra-auto osallisena onnettomuudessa
- Omaisuusvahinko ei yhteensopiva onnettomuuden kanssa
- Vain hyvin pieni törmäysvahinko
- Syyllisen auto pysähtynyt nopeasti
- Vahingon kärsineen ja syyllisen kertomukset eroavat
- Vakuutettu oli järkyttynyt, kielsi syyllisyyden
- Vakuutetulla aiempaa vahinkohistoriaa
- Vakuutettu asuu toisessa osavaltiossa
- Ajoneuvossa oli kolme tai enemmän ihmisiä
- Henkilövahinko muodostui vain kivusta ja ruhjeista
- Henkilövahingoista ei objektiivista todistusta

- Poliisiraportin mukaan ei henkilövahinkoja
- Ensiapua ei tarvittu onnettomuudessa
- Onnettomuuden jälkeistä hoitoa tarvittu vasta viiveen jälkeen
- Epätavallinen vahinko ko. onnettomuustyyppiin
- Aikaisempaa henkilövahinkohistoriaa
- Myönsi heti syyllisyytensä onnettomuuteen
- Vakuutettuun oli vaikeaa ottaa yhteyttä/yhteistyöhaluton
- Onnettomuus sattui nopeasti vakuutuksen voimaantulon jälkeen
- Vahingonkärsinyt syyllisen perheenjäsenen
- Vahingonkärsinyt aloittanut hiljattain työnteon

Yllä olevat sisääntulot olivat vahinkokäsittelijöiden ns. punaisia lippuja, eli mahdollisia vakuutusvilpin merkkejä. Näiden lisäksi neuroverkkoon oli otettu mukaan vielä muita muuttujia, kuten

- Ikä
- Viive päivissä vakuutuksen ottamisesta onnettomuuteen
- Viive päivissä vahingon ilmoittamisessa
- Viive onnettomuuden ja ensimmäisen lääkärissäkäynnin välillä
- Oliko edellisen kohdan viive 0 päivää? (binaarinen muuttuja)
- Viive onnettomuuden ja toisella lääkärillä käynnin välillä
- Oliko edellisen kohdan viive 0 päivää? (binaarinen muuttuja)
- Ambulanssikulut dollareissa
- Oliko ambulanssikuluja? (binaarinen muuttuja)
- Vahingonkärsinyt osittain työkyvytön
- Vahingonkärsinyt täysin työkyvytön
- Vahingonkärsinyttä edustaa asianajaja

Jatkuva-arvoiset muuttujat Viaenen *et al.* esimerkissä diskretisoidaan. Toisin sanoen, esimerkiksi ikä-muuttujan koko vaihteluväli on jaettu eripituisiin väleihin ja muuttujasta on muodostettu yhtä monta uutta binaarista muuttujaa. Jatkuva-arvoisten muuttujien diskretisointi muodostaa aina enemmän tai vähemmän uusia muuttujia, joista aina vain yksi saa arvon 1 muiden saadessa arvon 0. Asiantuntijoiden tehtävä oli käydä vahingot yksi kerrallaan lävitse ja liittää vahinkohin heidän arvionsa vakuutusvilpistä arvoasteikolla 0-10. Arvo 0 merkitsee "Ei vilppiä" ja suuremmat arvot aina vain todennäköisempää vakuutusvilppiä.

Valitettavasti artikkelissa ei tarkemmin selostettu esim. ROC-käyrien tai muulla tavalla neuroverkon tehokkuutta ennustamaan vilpillisiä vahinkoja. Joka tapauksessa yllä oleva muuttujalista voisi olla hyvä lähtökohta itse tehdylle neuroverkkosovellukselle.

4.8 Vakuutusten hinta- ym. herkkyys

Vakuutusyhtiöiden välinen kilpailu on vapautunut ja lisääntynyt viime vuosina. Yhtiöt pyrkivät kasvattamaan markkinaosuuttaan ja olemaan mahdollisimman kannattavia. Nämä kaksi tavoitetta ovat usein ristiriidassa keskenään. Vakuutusten hinta on tärkeä tekijä näiden kahden tavoitteen yhteensovittamisessa.

Vakuutusmaksujen pitäisi olla oikeudenmukaisia siinä mielessä, että vakuutuskannan eri osissa vakuutusmaksujen pitäisi kattaa vakuutuksista aiheutuneet korvaukset. Toisaalta maksut eivät saa olla niin korkeita, että esimerkiksi kannattavimmat asiakkaat lähtevät muiden yritysten asiakkaiksi ja vakuutuskantaan jäävät vain kannattamattomimmat.

Yksi ratkaisuehdotus markkinaosuus- ja kannattavuustavoitteiden yhteensovittamisessa voisi olla kannan jako ryhmiin kannattavuuden ja erilaisten asiakasta tai vakuutusta koskevien tekijöiden mukaan. Tällaisesta jaosta voitaisiin löytää vaikkapa ryhmiä, jotka ovat erittäin kannattavia ja toisia jotka ovat selvästi kannattamattomia. Jos sitten näiden ryhmien herkkyyttä vakuutusten hinnalle tutkittaisiin, voitaisiin edelleen huomata vaikka että kannattavan ryhmän hintaherkkyys on pieni. Siinä tapauksessa vakuutusten hinnan laskemisella tälle ryhmälle ei olisi suurta merkitystä. Joka tapauksessa tieto alennusvarasta ja sen mahdollisesta vaikutuksesta olisi erittäin hyödyllinen. Kannattamattoman ryhmän hintaherkkyys taas voisi olla suuri, jolloin pienikin hintakorotus aiheuttaa liikettä asiakkaissa. Nämä asiakkaat ovat ehkä kokeneet jo aikaisempina vuosina bonusten menetyksiä ja muita hinnankorotuksia ja ovat siksi herkkiä lisäkorotuksille. Tällöin suurten korotusten sijasta ehkä kannattaisi tehdä pienempiä hintakorotuksia ja muita kannattavuustoimenpiteitä, kuten nostaa omavastuita tai tehdä asiakkaan kanssa riskintorjuntatyötä (sprinklerisysteemit, kuljettajakoulutus, toiminta onnettomuustilanteissa, jne).

Seuraavassa esimerkissä neuroverkkoa (MLP) on käytetty vakuutusten hintaherkkyyden mallintamiseen autovakuutuskannassa. Analysoitavaksi on valittu tietty osa vakuutuskantaa, jonka kokonaiskannattavuus ja tarvittavien kannattavuustoimenpiteiden taso tiedetään. Tavoitteena kokonaisprojektissa voisi olla hienontaa toimenpiteet eri vakuutuksille mahdollisimman oikeudenmukaisesti huomioiden markkinaosuustavoitteet. Koko herkkyysanalyysin tekeminen koko vakuutus- tai asiakaskannalle olisi suuri työ ja vaatisi kokonaisen oman projektinsa. Käytännössä neuroverkkojen rakenne ja muuttujat pitäisi valita yritys–erehdys –periaatteella, verkkoja olisi opetettava lähtien useamman kerran erilaisilla painokertoimien lähtöarvoilla, erilaisia opetusparametreja olisi kokeiltava, jne. Kaikista tärkeintä olisi kuitenkin mahdollisimman hyvän opetusaineiston hankkiminen ja malleihin otettavien muuttujien valinta. Muuttujien valinta tehdään joko etukäteen tai neuroverkon opetuksen jälkeen. Kun vähemmän merkityksellisiä muuttujia on poistettu, on neuroverkkoa taas opetettava uudestaan yhä uudestaan erilaisilla kertoimien lähtöarvoilla ja opetusparametreilla.

Koko kannalle ei todennäköisesti saada muodostettua yhtä, kaiken kattavaa hyvää mallia. Sen vuoksi kanta on jaettava osiin. Vakuutuskanta on ensin jaettu kannattavuuden ja muiden tekijöiden suhteen ryhmiin, joiden kunkin sisällä asiakkaat ja vakuutukset ovat toistensa kaltaisia. Sen jälkeen neuroverkko voidaan opettaa ennustamaan herkkyyttä vakuutusten hintamuutoksille kussakin eri ryhmässä. Tietoja kannattavuudesta ja hintaherkkyydestä voidaan sitten

käyttää esimerkiksi tavoitteen asetteluun vakuutuskannan eri osissa: esimerkiksi korvaussuhdetavoite, kannattavuustavoite euroissa tai markkinaosuus. Tavoitteet voivat myös olla erilaiset kannan eri osissa. Hinta- yms. muutoksia ei ole käytännöllistä tehdä liian asiakaskohtaisesti. Neuroverkkojen malleja on tarkoitus käyttää eri vaihtoetojen testaamisessa: millaiset muutokset todennäköisesti menisivät läpi ja millaisia kannattavuus- ja markkinaosuusvaikutuksia niillä olisi.

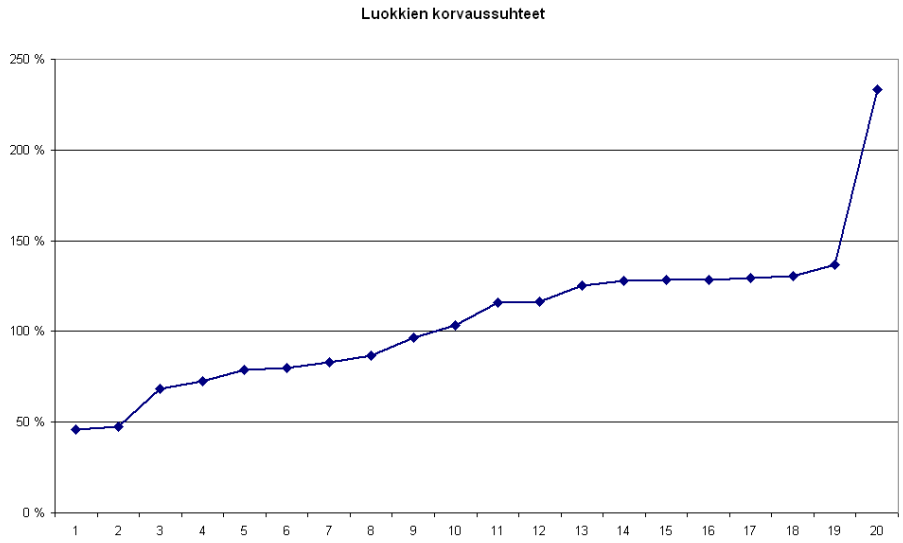
Kannan ryhmittely keskenään toistensa kaltaisiin asiakkaisiin ja vakuutuksiin on tehty itseorganisaatiokartan SOM avulla. Muuttujat tässä vaiheessa olivat seuraavat:

- Asiakkaan toimiala
- Asiakkaan kannattavuusluokka
- Vakuutuksen voimassaolovuodet
- Asiakkaan koko
- Asiakkaan ajoneuvojen lukumäärät ajoneuvotyypeittäin

SOM-algoritmi jakoi asiakaskannan 20 luokkaan. Oheisessa kaaviossa 4.12 on esitetty luokkien korvaussuhteet (keskim. = 100%). Varsinkin ääripäiden luokat SOM on onnistunut erottamaan. Kaikista kannattamattomin luokka on sattumalta myös selvästi pienin ajoneuvojen määrällä mitattuna. Tällä kerralla luokkia yhdistetään manuaalisesti kolmeksi suuremmaksi luokaksi, joille sitten muodostetaan hintaherkkyyden malli MLP-verkolla. Kukin suurempi luokka sisältää noin kolmanneksen kannasta. Oikeasti luokkia pitäisi yhdistää sen mukaan mitkä luokat ovat lähimpänä toisiaan. Siinä voitaisiin taas käyttää esimerkiksi U-matriiseja apuna. Nyt luokkien yhdistely tehdään korvaussuhteiden ja kannan osuuksien mukaan. Todennäköisesti luokkien oikea yhdistely parantaisi myös tuloksia, mutta analyysin harjoittelun kannalta sillä ei ole suurta merkitystä.

Seuraavaksi kullekin muodostetulle kolmelle suurelle asiakasluokalle sovitetaan erikseen neuroverkkomalli hintaherkkyyttä varten. Tätä vaihetta varten otettiin mukaan edellisen vaiheen muuttujien lisäksi seuraavat muuttujat:

- Autojen lukumäärä
- Maksut yhteensä
- Keskimääräinen maksu
- Maksumuutos euroina
- Vakuutusmaksun muutos prosentteina
- Bonus
- Uusi bonus
- Vahinkojen lukumäärä vakuutuskaudella



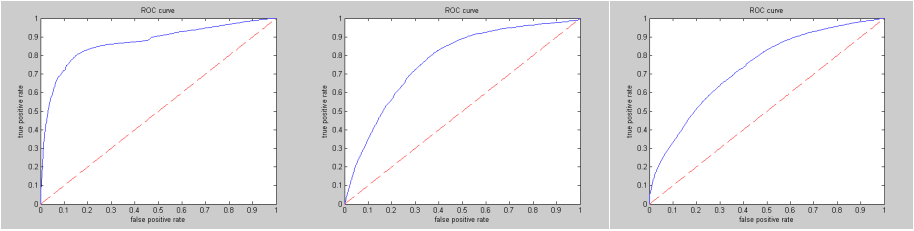
Kuva 4.12: SOM-verkon muodostamat asiakasluokat

Muuttujien kokonaismäärä oli 35, joista viimeinen oli opetettava kohdemuuttuja eli tieto siitä uudistuuko sopimus vai ei. Koko havaintoaineisto jaettiin siis kolmeen luokkaan. Kunkin luokan aineisto jaettiin edelleen opetus- ja validointiaineistoksi. Opetusaineistoon valittiin satunnaisesti samat määrät uudistuneita ja ei-uudistuneita sopimuksia. Tavoitteena opetuksessa on, että neuroverkko oppisi mahdollisimman hyvin löytämään ne sopimukset joissa on suuri todennäköisyys, että sopimus ei uudistu. Suurin osa sopimuksista kuitenkin uudistuu. Jos opetusaineiston havaintoja ei edellä kerrotulla tavalla tasapainotettaisi, neuroverkko voisi helposti oppia liian usein ennustamaan sopimuksen uudistuvan. Vaikka se ennustaisi kaikkien sopimusten uudistuvan, olisi ennusteiden osumisprosentti aika hyvä. Validointiaineistoon otettiin mukaan havaintoja, jotka eivät olleet mukana opetusaineistossa. Uudistuneiden ja ei-uudistuneiden suhde pidettiin samana kuin alkuperäisessä aineistossa.

Opetusaineistojen rivimäärät luokille 1-3 olivat vastaavasti 14 000, 16 000 ja 16 000 ja neuroverkkoon otettiin 20 neuronia ensimmäiseen piilokerrokseen ja toiseen 8 neuronia. Neuronien lukumäärä oli varmasti ylärajoilla, mutta yleissäännön mukaan (5-10 havaintoa kerrointa kohti) ainakin havaintoja opetusaineistossa oli tarpeeksi. Ensimmäisen neuroverkon opetus onnistui heti hyvin, mutta seuraavat kaksi olivat hankalampia opettaa. Ensimmäisen neuroverkon opetuksesta saatiin vertailukohta sille, kuinka suuri opetetun verkon virheen pitäisi suurin piirtein olla. Mutta ilman onnistumista ensimmäisen verkon opetuksessa ei olisi mitenkään voinut tietää mikä virhe oli vielä suuri ja mikä pieni. Opetuksessa voi helposti tulla houkutus valita suuri opetusnopeuden kerroin ja vastaavasti pienempi momenttikerroin. Tämän tehtävän neuroverkoille hyvät kertoimet olivat päinvastoin hyvin pienet opetusnopeudelle (luokkaa 0,01-0,02) ja suuri momentille (noin 0,8-0,9). Jos opetusnopeus on suuri ja momentti pieni, neuroverkon opetus takertuu helposti paikalliseen minimiin. Toisaalta pienellä

opetusnopeudella opetus voi kestää todella kauan, eikä se siltikään välttämättä johda globaaliin minimiin.

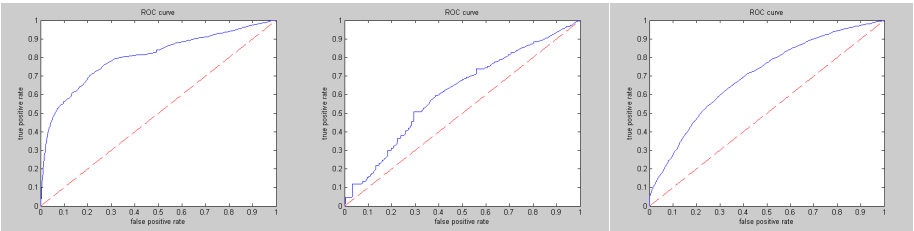
Kuvissa 4.13 on esitetty ROC-käyrät neuroverkkojen opetusdatalla antamilla tuloksilla. ROC-käyrät on määritelty kappaleessa "Neurolaskennan tulosten arviointi".



Kuva 4.13: ROC-käyrät – Neuroverkkojen 1-3 opetusaineistolla

Luokkaan 1 liittyvä ROC-käyrä osoittaa, että neuroverkko löytää hyvin vakuutuksista ne, jotka ovat vaarassa ei-uudistua. Luokkiin 2 ja 3 liittyvät toimintaominaiskäyrät eivät ole yhtä hyviä ja kuvissa on kuitenkin kyseessä neuroverkkojen opetusdatalla saadut luokittelut.

Kuvissa 4.14 on varsinaiset vastaavat toimintaominaiskäyrät, jotka on tehty opetusvaiheessa käyttämättömällä validointidatalla. Validointidatan ROC-käyrien ei pitäisi näyttää kovin paljon huonommalta kuin opetusdatan käyrät. Jos näin on, on opetuksessa tapahtunut joko ylioppimista tai sitten se johtuu käytetystä datasta. Yleensä ylioppiminen ei ole ongelma, vaan se että opetus pysähtyy paikalliseen minimiin.



Kuva 4.14: ROC-käyrät – Neuroverkkojen 1-3 validointiaineistolla

Jälleen luokan 1 neuroverkkomalli näyttää toimivan hyvin. Luokkien 2 ja 3 kohdalla luokittelu ei toimi yhtä hyvin. Ilmeisesti opetuksessa ei ole vielä löytynyt oikeata minimikohtaa. Verkkojen virhe on samalla tasolla kuin verkon 1 opetuksessa ennen kuin verkon 1 virhe yhtäkkiä puolittui opetuksen aikana. Luokan 2 huonolta näyttävä ROC-käyrä on myös varoittava esimerkki. Suurin syy käyrän muotoon on se, että validointiaineistossa oli joitakin erittäin suuria muuttujien arvoja. Kun Joone-ohjelma sitten normalisoi automaattisesti aineiston, se käytti ko. validointiaineistosta laskettuja muuttujien maksimeja, minimejä ja vaihteluvälejä. Toisin sanoen, normalisointi tuotti paljon arvoja jotka olivat lähellä arvoa 0. Opetusaineiston tapauksessa kyseiset arvot olisivat saaneet nollasta selvästi poikkeavia arvoja. Validointiaineisto olisi siis pitänyt normalisoida samoilla parametreilla kuin opetusaineistokin. Silloin suurimmalle osalle validoin-

tiaineistoa olisi saatu kunnollisia tuloksia ja vain selvästi poikkeavien havaintojen luokittelut olisivat olleet mitä sattuu.

Taulukossa 4.5 on esitetty esimerkkinä Luokan 1 neuroverkon tulokset validointidatalla. Jotta tällaisen taulukon voisi tehdä, täytyy ensin kiinnittää luokittelun kynnysarvo esimerkiksi kaavan (3.1) avulla. Taulukko on laadittu kynnysarvoa 0,4 vastaavasti.

Ennuste	Oikea tulos: Uudistunut	Ei uudistunut
Uudistuu	4 234	654
Ei uudistu	1 180	1 604

Taulukko 4.5: Luokan 1 tulokset neuroverkolla

Neuroverkon tehtävä oli tunnistaa vakuutuksista ne vakuutukset, jotka eivät uudistu. Vakuutuksilla on vain kaksi vaihtoehtoa, tosin sanoen muut vakuutukset siis uudistuvat. Taulukosta nähdään esimerkiksi, että ei-uudistuvista vakuutuksista neuroverkko (kynnysarvo=0,4) tunnisti 1 604 kappaletta ja tunnistamatta jäi 654 kappaletta. Neuroverkko sai siis 71% ei-uudistuvista vakuutuksista kiinni. Toisaalta niistä vakuutuksista, jotka verkko ennusti ei-uudistuvan 58% ei uudistunut. Taulukon luvut ja prosentit muuttuvat sen mukaan miten eri kustannuksia vääristä ja hyötyjä oikeista luokitteluista painotetaan.

Opetettuja neuroverkkoja voi käyttää ennustamaan kuinka moni vakuutus ei uudistu, kun maksukorotukset ovat tietyt kiinnitetyt. Niillä voidaan esimerkiksi kokeilla yksittäisiä tapauksia tai simuloida suuri määrä vakuutuksia, jotka sitten syötetään neuroverkolle. Tuloksista voidaan arvioida vakuutusten uudistumisprosenttia sekä kappale- että euromääräisesti, uudistumisen merkitystä eri ajoneuvolajeittain, yms.

4.9 Muita

Neuroverkkoja koskevissa artikkeleissa ja kirjallisuudessa on esitetty, että neuroverkkoja voitaisiin käyttää myös esimerkiksi seuraavissa vakuutusalaan liittyvissä tehtävissä:

- Sijoitukset
- Jälleenvakuutus
- CRM
- Vakavaraisuuden valvonta

CRM eli asiakkuudenhallinta sekä sijoitusten hallinta liittyvät muihinkin aloihin kuin vain vakuutusalaan. Neuroverkkojen avulla yrityksen asiakkaita voidaan luokitella esimerkiksi erilaisia kampanjoita varten. Neurolaskennan käyttämisestä CRM:ssä on suhteellisen helppoa löytää tietoa ainakin yleisellä tasolla.

Myös sijoitusovelluksista, joissa on käytetty neurolaskentaa, voi löytää paljon aineistoa. Osakekurssien mallintaminen neuroverkkojen avulla on melko suosittu neuroverkkojen sovellusala, mutta mikään mullistava tekniikka se ei ole.

Jälleenvakuutus on ainakin mainittu mahdollisena neuroverkkojen sovellusalana. Neuroverkkoja voidaan käyttää suurten vahinkojen ennustamiseen siinä missä pienienkin. Ongelma tulisi varmaan riittävän suuren havaintoaineiston saamisesta.

Vakavaraisuuden valvonnan neuroverkkosovelluksia on myös tutkittu. Tutkimuksissa on lähinnä vertailtu perinteisiä erilaisiin suhdelukuihin perustuvia arviointimenetelmiä neuroverkkosovelluksiin, joissa käytetään samojen suhdelukujen lisäksi joitakin muita lisämuuttujia. Tulokset ovat olleet aika samanlaisia sekä vanhoilla menetelmillä että uusilla neuroverkkosovelluksilla.

Luku 5

Johtopäätökset

Neuroverkot tarjoavat mielenkiintoisia mahdollisuuksia vakuutusalan aineistojen analysoimiseen. Niitä voi käyttää ennustamiseen, luokitteluun ja vaikka havaintoaineiston muokkaamiseen muita analyyseja varten. Sama neuroverkkomalli voi sopia monenlaisiin tehtäviin. Monikerros-perceptron –verkko on ylivoimaisesti käytetyin malli varmaan juuri sen vuoksi, että monet laskennalliset ongelmat on helppoa muokata sille sopiviksi. MLP-verkkojen soveltamiseen tarkoitetut ohjelmat ovat myös pisimmälle kehitettyjä.

Neuroverkkojen negatiivisia puolia on ehkä liikaakin korostettu. "Black box"–ominaisuus on suuri ongelma, mutta opetetun neuroverkon kertoimien tutkiminen ja tulosten visualisointi auttavat paljon. Neuroverkon ratkaisun tutkiminen lisää tietoa myös pohjalla olevasta mallista. Usein mainitaan myös neuroverkkojen ylioppimisen vaara. Käytännössä se ei kuitenkaan ole kovin vakava ongelma. Neuroverkon kokoa rajoittamalla ja opetetun verkon testaaminen validointidatalla auttavat estämään mahdolliset ylioppimiset. Myös havaintoaineistoa on oltava paljon, usein enemmän kuin muissa parinteisimmillä menetelmillä. Perinteiset menetelmät voivat antaa järkeviä tuloksia vähälläkin aineistolla, mutta esimerkiksi kertoimien luottamusvälit ovat suuret. Pienellä opetusaineistolla ei pysty opettamaan neuroverkkoa, josta olisi mitään hyötyä käytännössä.

Parhaiten neuroverkkojen soveltamista oppii kokeilemalla niitä todelliseen aineistoon. Opetusaineiston laadun ja määrän merkitys on suuri. Virheellisestä aineistosta mikään analysointitapa ei löydä oikeaa mallia. Huonolaatuinen aineisto voi hidastaa tai estää kokonaan neuroverkkoa oppimasta. Pieni aineiston määrä asettaa rajat mallin painokertoimien määrälle. Yleissääntöä 5-10 havaintoa painokerrointa kohti tuntui olevan riittävän hyvä ja helposti muistettava ohje. Itse neuroverkon opetus on usein arpapeliä. Opetusvaiheessa on kokeiltava monia eri lähtöpisteitä ja erilaisia opetuskertoimia. Käytännön ongelmissa on varmaan parasta aloittaa heti pienillä opetusnopeuskertoimilla ja melko suurilla momenttikertoimilla.

Olen käyttänyt työssä ilmaista Joone-nimistä neuroverkko-ohjelmaa ja Databiosonics'in ESOM-verkkoja. ESOM:ssa itseorganisaatioverkon tuloksia visualisoidaan U-matriisien avulla. Kuvat ovat joissakin tapauksissa hyödyllisiä ja ainakin ne ovat intuitiivisesti selviä muillekin jotka eivät välttämättä tiedä SOM-verkoista mitään. Kumpikin ohjelma sopii hyvin neuroverkkojen kokeilemiseen.

Ne ovat helppoja käyttää, niissä on joitakin käytännöllisiä piirteitä ja ne toimivat suhteellisen tehokkaasti. Ohjelmien huono puoli on, että niissä ei ole toteutettu kuin joitakin erilaisia neuroverkkomalleja ja esimerkiksi Joone-ohjelman kehitys on pysähtynyt kokonaan. Kaupallisista sovelluksista lukemani perusteella päätyisin varmaan Matlab-ohjelmistoon.

Vakuutusalalla neuroverkkoja voisi soveltaa raportoinnissa ja erilaisten ennusteiden laatimisessa. Niitä voisi käyttää myös päätöksenteon apuna. Mutta esimerkiksi suora hinnoittelu neuroverkkojen avulla on selvästi liian kunnianhimoinen tavoite. Myös kaikenlainen mallintaminen, jossa mallilla täytyy olla selvät, kirjalliset perusteet, ei oikein sovi neuroverkoille. Neuroverkon oikeastaan määrittää vain verkon topologia ja käytetty opetusaineisto. Joka tapauksessa neuroverkkojen kokeileminen erilaisissa käytännön ongelmissa lisää tietoa itse ilmiöstä ja siitä olevasta havaintoaineistosta. Lisäksi jos neuroverkko pystyy oppimaan jonkinlaisen selvästi toimivan ja validointidatalla testatun mallin, on ilmeisestikin jonkinlainen selitysmalli olemassa.

Merkityksellisin ero neuroverkkojen ja tilastollisten menetelmien välillä on, että neuroverkkoja varten ei tarvita etukäteen oletuksia havaintojen jakaumista tai muista ominaisuuksista. Siksi niiden soveltaminen monissa käytännön tehtävissä on houkuttelevaa. Neuroverkkojen mallit ovat myös luonnostaan epälineaarisia ja voivat mallintaa monimutkaisia muuttujien välisiä yhteyksiä. Neurolaskennan ajatusmalli on lähellä tilastointia, jossa ongelman ratkaisu tehdään esimerkeistä opettamalla. Tällöin saattaa olla vaikea selvittää syitä, joihin neuromallin antama vastaus perustuu. Viime aikoina on ollut voimassa suuntaus yhdistää tietämys ongelmasta ja mitattu tieto. Tällöin puhutaan neuro-sumeista järjestelmistä tai neuro-hybridi –malleista. Niissä määrätään joukko tulkittavissa olevia sääntöjä, joiden yhteisvaikutus määrää päättelyn tuloksen. Neurolaskennan osuus on virittää sääntöjen yhteisvaikutus vastaamaan havaintoaineistoa.

Neuroverkkojen viimeaikainen kehitys on ollut mahdollista osaksi tehokkaampien tietokoneiden ansiosta. Vielä tärkeämpää on ollut teorian kehittyminen. Esimerkiksi neuroverkkojen teorian tarkastelu dynaamisten järjestelmien teorian pohjalta tuo lähivuosina todennäköisesti merkittäviä tuloksia. Tavallisimmat neuroverkkomallit ovat jo vakiinnuttaneet asemansa kaupallisissakin sovelluksissa. Parhaat neuroverkkomallit tuskin jäävät nykyiselle tasolle, vaan tehokkaampia löydetään yhä lisää. Voi olla, että odottamassa on jopa suurempi käännekohta neurolaskennan alalla.

Luku 6

Sanasto

- *Aktivaatio (activation).* Ks. Ulostulo
- *Aktivaatiofunktio (activation function).* Funktio, joka määrää miten laskennallisen perusyksikön, neuronin, ulostulo määräytyy sisääntulojen ja painojen perusteella. Aktivaatiofunktio liittyy läheisesti käsitteeseen kynnysfunktio. Esimerkkejä kynnysfunktioista on askel-, merkki-, lineaarinen ja sigmoidifunktiot. Aktivaatiofunktio muuntaa sisääntulonsa kynnysfunktion avulla ulostulokseen.
- *Aktivaatiofunktio (treshold function).* Funktio, joka määrää miten laskennallisen perusyksikön eli neuronin ulostulo määräytyy solun syötteiden eli sisääntulojen ja painojen perusteella. Ks. Ulostulo, Sisääntulo, Painokertoimet.
- *Aksoni (axon).* Hermosolun viejähaarake aivoissa, joka toimii johtimena siirtäen neuronin signaalin toisiin neuroneihin. Aivojen aksonit voivat olla jopa metrin pituisia.
- *Assosiatiivinen muisti (associative memory).* Neuroverkkomalli muistille, jossa tallenetun informaation A haku tapahtuu opetusvaiheessa määrätyn informaation B avulla. B:tä kutsutaan usein hakuavaimeksi ja tavanomaisista muisteista poiketen B voi olla vääristynyt tai puutteellinen. Mikäli B on A:n osa tai vääristymä, niin muistia kutsutaan autoassosiatiiviseksi. Itse informaatio on usein koodattu tavallisista muisteista poiketen hajautetusti kaikkiin neuroverkon painoihin esim. korrelaatiomatriisin avulla.
- *Autoassosiatiivinen verkko.* Jos kaikilla opetusesimerkeillä haluttu ulostulo on samanlainen kuin sisääntulo, sanotaan verkkoa autoassosiatiiviseksi.
- *Backpropagation-algoritmi, BP (backpropagation).* Neuroverkon opetusalgoritmi, jolla voidaan ohjatusti opettaa monikerroksista epälineaarista perceptron-verkkoa. Algoritmi toimii kahdessa vaiheessa: ensin neuroverkolle annetaan uudet syötearvot ja neuroverkon tulos lasketaan. Tulosta verrataan haluttuun ulostuloon ja virhe lasketaan. Sitten virhettä kuljetaan verkossa takaperin sisääntulokerrosta kohti. Samalla painokertoimien arvoja korjataan niin, että seuraavalla kerralla virhe pienenisi. Jos-

sakin lähteissä on ehdotettu algoritmille suomenkielistä nimeä vastavirtaagoritmi. Ks. monikerros-perceptron.

- *Boltzmannin kone (Boltzmann machine).* Stokastiseen aktivaatiofunktioon perustuva oppiva neuroverkkomalli, jossa neuronien ulostulot ovat kaksiarvoisia.

- *Delta-sääntö (Delta rule).* Perusmenetelmä, jolla perceptronin painokertoimia säädetään opetuksen aikana. Painokertoimen muutos on verrannollinen ulostulon virheeseen: $\Delta w_{ij} = \eta \cdot (d_j - y_j$, missä d_j on haluttu ulostulo ja y_j on verkon toteutunut ulostulo.

- *Dendriitti (dendrite).* Hermosolun tuojahaarake eli osa neuronista, joka haarautuu puumaisesti yhdistyen toisista neuroneista lähteviin aksoneihin.

- *Estävä/vaimentava kytkent (Inhibitory connection).* Neuronin ulostuloa heikentävä kytkentä, joka on yleensä toteutettu neuroverkossa negatiivisen painokertoimen avulla.

- *Hahmo (pattern).* Rinnakkainen joukko lukuja, tyypillisesti verkon samanaikaisten sisääntulojen tai ulostulojen joukko, joihin on koodattuna verkon käsittelemä informaatio. Sisääntulohahmo (input pattern) esittää esimerkiksi jotain luokiteltavaa kohdetta ja ulostulohahmo (output pattern) neuroverkon kohteelle määräämää luokkaa.

- *Hahmontunnistus (pattern recognition).* Menetelmät, joilla kohteista kuten kuvista, puheesta, tms. tunnistetaan joitakin mielekkäitä olioluokkia kuten esim. kirjaimia.

- *Haluttu ulostulo (desired output, target output).* Opetusjoukkoon kuuluva ulostulo, joka verkon tai yksittäisen neuronin pitäisi antaa kun sille annetaan tietty opetusjoukkoon kuuluva sisääntulo. Ks. Opetusjoukko.

- *Hebb'in sääntö (Hebb's rule).* Hebb'in sääntö on yksinkertaisin keinotekoisten neuronien opetusmenetelmä. Tässä menetelmässä neuronien painokertoimia muutetaan kaavalla $\Delta w_{ij} = \eta \cdot x_i \cdot y_j$, missä η on opetusnopeuskerroin, x_i neuronin i sisääntulo ja y_j neuronin j ulostulo. Menetelmä voidaan perustella biologisen neuronin toiminnalla seuraavasti: kahden neuronin välinen synaptinen liitos välittää sisääntuloneuronin aksonin aktiivisuustilan ulostuloneuronin dendriitin aktiivisuudeksi. Tämä yhtäaikainen aktiivisuus aiheuttaa toisessa tai molemmissa neuroneissa metabolisia muutoksia niin, että ensimmäisen neuronin aktiivisuus kasvattaa jatkuvasti toisen neuronin aktiivisuutta. Ts. neuronien välinen yhteys voimistuu. Menetelmän puute on se, että painokerroin w_{ij} voi kasvaa rajattomasti. Hebb'in parannetussa menetelmässä painokertoimia muutetaan kaavalla $\Delta w_{ij} = \alpha \cdot (x_i - \beta_j \cdot w_{ij}$. Kaavasta on siis jätetty pois ulostulo y_j ja lisätty uonohtamisnopeuskerroin β_j. Kerroin α on ajasta riippuva kerroin, joka kuvaa neuronin ympäristön aktiivisuuden vaikutusta oppimiseen eli se kuvaa neuronin ulkopuolisten tekijöiden vaikutusta. Kertoimen β avulla estetään painokertoimen rajaton kasvu eli se kuvaa neuronin sisäisten tekijöiden vaikutusta. Ulostulon y_j poistaminen kaavasta perustuu havaintoon, että neuronin ulostulon aktivoituminen riippuu koko ympäristön aktiivisuudesta. Unohtamisnopeuskerroin β_j voidaan

kuvata kaavalla $\beta_j = \sum_{r=1}^{N} w_{ir} \cdot x_i$, missä indeksi r juoksee yli kaikkien läheisten synapsien. Naapurisynapsien aiheuttama unohtamisvaikutus perustuu ajatukseen, jonka mukaan solun tietyn haaran synapsit käyttävät samoja molekyyli- ja energiaresursseja, jolloin aktiiviset naapurisynapsit pienentävät vähemmän aktiivisen synapsin signaalia.

- *Heteroassosiatiivinen verkko.* Jos haluttu ulostulo on erilainen kuin sisääntulo, sanotaan verkkoa heteroassosiatiivikseksi.

- *Itseorganisoituva kartta (Self-organizing map).* Tunetuin ohjaamattomaan oppimiseen perustuvista neuroverkkomalleista, jota voi käyttää sisääntulojen ryhmittelyyn ja visualisointiin.

- *Epookki (epoch).* Yksi kierros, jossa koko opetusaineisto on käyty läpi.

- *Kilpailuoppiminen (competitive learning).* Eräs ohjaamattoman oppimisen muoto. Neuronit kilpailevat keskenään siitä mikä yksi ainoa niistä vastaisi tiettyyn sisääntuloon. Sitä neuronia, joka tämän kilpailun voittaa kutsutaan „winner-takes-all" -neuroniksi. Ks. Oppiminen.

- *Konnektionismi (connectionism).* Neuroverkoilla toteutettu tekoälyn muoto, jossa symbolisten esitysten väliset relaatiot ovat koodattuina neuroverkon painokertoimiin.

- *Kynnysparametri (treshold parameter, bias, offset).* Perceptron-neuronin parametri, jolla se voi siirtää sigmoidisen aktivaatiofunktionsa toimintapistettä.

- *Kytkentä (connection).* Liitäntä kahden neuronin tai muunlaisen laskennallisen elementin välillä. Ks. Painokertoimet.

- *LVQ.* Lyhenne sanoista Learning Vector Quantizer. Ks. Oppiva vektorikvantisaatio.

- *MLP.* Lyhenne sanoista Multi-layer Perceptron. Ks. Monikerros-perceptron.

- *Monikerros-perceptron (Multi-layer perceptron).* Useista kerroksista kostuva neuroverkko, jossa kukin kerros koostuu joukosta rinnaksisia perceptroneja. Yleisimmin käytetään kaksikerros-perceptronia, jossa on sisääntulo-, piilo ja ulostulokerros. Sisääntulokerrosta ei lasketa omaksi kerroksekseen.

- *Myötäkytkentä (feedforward connection).* Myötäkytketyssä neuroverkossa neuronien ulostulot ovat aina seuraavan kerroksen neuronien sisääntuloja. Neuronit saavat aina kaikki sisääntulonsa vain edeltävistä kerroksista.

- *Neurolaskenta.* Yleisnimi kaikelle ongelmanratkaisulle, joka merkittäviltä osin perustuu neuroverkkojen käyttöön. Sovelluksissa neurolaskentaa käytetään tyypillisesti rajattuihin tehtäviin osana laajempaa järjestelmää.

- *Neuroni.* Hermosolu, joka koostuu solurungosta, aksonista ja dendriiteistä sekä niihin liittyvistä synapseista. Neuroverkkomalleissa keinotekoinen neuroni (artificial neuron) on laskennallinen verkon perusyksikkö, joka kokoaa sisääntulot, laskee nidien perusteella ulostulonsa ja lähettää sen verkossa eteenpäin.

- *Neuroverkon rakenne eli arkkitehtuuri (architecture).* Yksittäisen neuroverkon kytkentäkaavio, esim monikerros-perceptron, tai neuroverkoista koostuvan systeemin rakennekuvaus.

- *Ohjaamaton oppiminen (unsupervised learning).* Ks. Oppiminen.

- *Ohjattu oppiminen (supervised learning).* Ks. Oppiminen.

- *Opetus (training).* Neuroverkkomallin parametrien viritys opetusaineiston avulla. Ks. Oppiminen.

- *Opetusaineisto (training set).* Neuroverkon opetusta varten kerätty joukko sisääntuloja ja haluttuja ulostuloja.

- *Opetusalgoritmi.* Opetusalgoritmi tarkoittaa joukkoa peräkkäisiä operaatioita verkon painokertoimien määräämiseksi, joita toistetaan verkon opetuksen ajan. Nämä operaatiot määräävät miten oppiminen neuroverkossa tapahtuu. Eräät algoritmit ovat tätä yksinkertaistettua mallia mutkikkaampia. Usein opetusalgoritmi on sidoksissa tiettyy verkkorakenteeseen, jolloin koko menetelmää voidaan kutsua opetusalgoritmin mukaan.

- *Opetuskierros (learning cycle, epoch).* Opetusjakso, jonka aikana koko opetusaineisto on kertaalleen käyty läpi opetuksessa. Neuroverkon opettaminen voi vaatia useita opetuskierroksia.

- *Oppiminen (learning).* Vaihe, jossa neuroverkko muuttaa painokertoimiaan saamiensa sisääntulojensa perusteella. Oppimisen lajeja ovat: 1) Ohjattu oppiminen, jossa verkko opetetaan vastaamaan annettuihin sisääntuloihin annetuilla ulostuloilla. Opetusaineisto koostuu joukosta sisääntuloja ja niitä vastaavia haluttuja ulostuloja. 2) Ohjaamaton oppiminen, jossa verkko pyrkii mukautumaan tietyn kriteerin mukaisesti annettuihin sisääntulohin. Opetusaineistossa ei ole sisääntuloihin liittyviä haluttuja vasteita. 3) Vahvistava oppiminen (reinforcement), joka on kahden edellisen opetustavan välimuoto. Verkon antamasta tuloksesta tarkistetaan vain vastaako se hyvin vai huonosti haluttua tulosta. Tämä oppimistapa perustuu Hebb'in sääntöön.

- *Opetusnopeus (learning rate).* Opetusalgoritmien tärkeä parametri $0 \leq \eta \leq 1$, joka määrää kuinka suuri muutos kullakin askeleella tehdään painokertomien arvoihin.

- *Oppiva vektorikvantisaatio (learning vector quantizer, LVQ.* Itseorganisoituvan kartan versio, jossa karttaa opetetaan lopuksi käyttäen ohjattua opetusta.

- *Painokertoimet (weights).* Neuronien välisten kytkentöjen voimakkuudet, jotka yleensä toteutetaan jatkuva-arvoisina lukuina. Neuroverkoissa painoilla on keskeinen osuus verkkojen suorittaman informaatiokäsittelytehtävän kannalta. Opetuksen tulos koodautuu vähitellen painokertoimiin.

- *Perceptron.* Frank Rosenblattin ehdottama yksinkertaisin neuroverkon muoto. Perceprtonin muodostaa yksi ainoa neuroni, jossa on muutettavat painokertoimet ja hard limiter -aktivaatiofunktio. Hard limiter -funktio on

askel- ja merkkifunktion yhdistelmä. Merkkifunktio antaa arvon +1, jos syöte on positiivinen ja arvon −1, jos syöte on negatiivinen.

- *Piilokerros (hidden layer).* Monikerroksisen neuroverkon kerros sisääntuloja ulostulokerrosten välissä, johon ei ole suoria kytkentöjä verkolle annettavista sisääntuloista ja josta ei lähde suoraan neuroverkon ulostuloja. Piilokerroksen oppimia esityksiä kutsutaan verkon sisäisiksi esityksiksi (internal representations).

- *Radiaalikantafunktioverkko (radial basis function network, RBF network.* Yhdestä piilokerroksesta ja ulostulokerroksesta koostuva verkko, jossa piilokerroksen neuronien aktivaatiofunktiot ovat erityistä radiaalikantafunktiotyyppiä.

- *Sigmoidifunktio (sigmoid function).* Loivan S-kirjaimen muotoinen käyrä, jota sovelletaan usein neuronien aktivaatiofunktioissa. Sigmoidi on rajoitettu sekä alhaalta että ylhäältä ja on monotonisesti kasvava. Tyypillinen sigmoidifunktio on hyperbolinen tangentti, joka saa arvoja vain −1 ja +1 väliltä.

- *Signaalinsiirtokuvaus-neuroverkko (signal transfer mapping).* Kaikkein yleisimmin esiintyvä neuroverkkomalli, jossa sisääntulona tulleita signaaleja siirretään verkossa etenpäin kunnes ulostulokerroksesta saadaan lopulliset ulostulot. Yleisimmin esiintyvä signaalinsiirtoa suorittava verkkomalli on *yksisuuntainen kerrosverkko (feedforward layered network).*

- *Sisääntulo (input).* Neuroniin tai neuroverkon ensimmäiseen kerrokseen tulevat lukuarvot, esim. mittausarvot tai edellisen kerroksen neuronin ulostulo joka tulee seuraavaan neuroniin.

- *Sisääntulokerros (input layer).* Neuroverkon sisääntulojen muodostama kuvitteellinen kerros. Kyseessä eivät ole neuronit vaan pelkät haaroittimet, jotka vievät kunkin sisääntuloarvon jokaiseen ensimmäisen kerroksen neuroniin. Tätä neuroverkon ensimmäistä kerrosta ei lasketa verkon varsinaisten kerrosten lukumäärään.

- *Synapsi (synapse).* Kahden hermosolun liittymäkohta. Keinotekoisissa neuroverkoissa neuronien väliseen yhteyteen liittyy painokerroin, joka joko vahvistaa tai heikentää signaalia.

- *Takaisinkytkentä (feedback connection, recurrent connection).* Takaisinkytketyssä neuroverkossa ainakin jonkin neuronin ulostuloa käytetään jonkin edellisen kerroksen neuronin sisääntulona.

- *Testiaineisto (test set).* Opetusjoukosta erillinen joukko tyypillisiä sisääntuloja ja ulostuloja, joita käytetään opetetun neuroverkon suorituskyvyn arviointiin. Ks. Yleistys

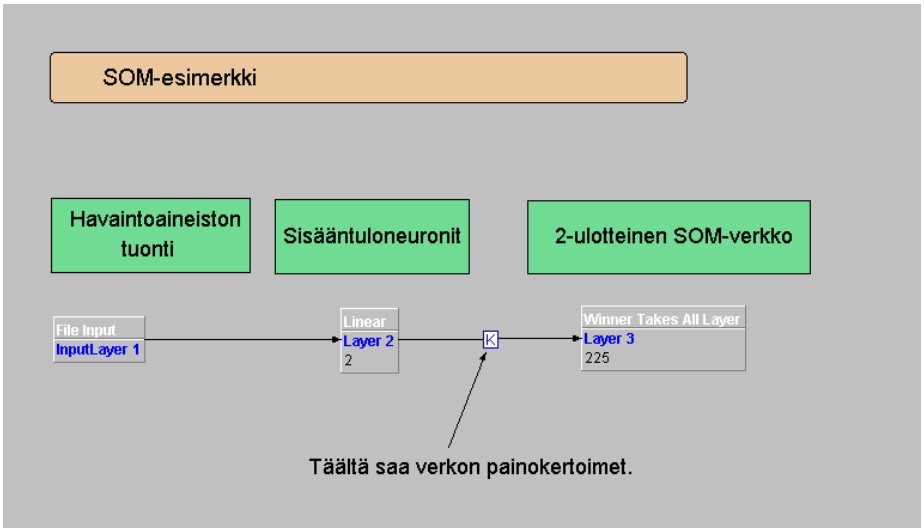
- *Tilansiirtokuvaus-neuroverkko (state transfer mapping).* Neuroverkkomalli, joka on voimakkaasti takaisinkytketty ja epälineaarinen ja jolla on runsaasti stabiileja tiloja. Tällainen verkko pyritään saamaan toimimaan assosiaatiomuistina valitsemalla neuronien kertoimet siten, että verkon stabiilit tilat vastaisivat muistiin talletettavia *hahmoja (pattern)*, siis havaintoja tms.

- *Ulostulo (output).* Neuronin tai neuroverkon antama tulos, joka ilmoitetaan yleensä jatkuva-arvoisena lukuna tai binaarimuuttujana.
- *Ulostulokerros (output layer).* Neuroverkon kerros, jonka kaikki neuronit ovat ulostuloneuroneita.
- *Vahvistava kytkentä (excitatory connection).* Neuronin vastetta vahvistava kytkentä, joka yleensä on toteutettu positiivisen painokertoimen avulla.
- *Vaimentava kytkentä (inhibitory connection.* Neuronin ulostuloa heikentävä kytkentä, joka on yleensä toteutettu negatiivisen painokertoimen avulla.
- *Virhefunktio (Error function, cost function).* Neuroverkon oppimisen tai järjestymisen tilaa mittaava funktio, jonka minimointiin opetusalgortimit pyrkivät.
- *Yksisuuntainen kerrosverkko (feedforwar layered network).* Neuroverkko, jossa signaalit kulkevat aina vain eteenpäin sisääntulokerroksesta kohti ulostulokerrosta, eikä signaali missään vaiheessa palaa edellisen tai saman kerroksen toiseen neuroniin.
- *Yleistys (generalization).* Neuroverkon kyky vastata järkevästi sisääntuloihin, jotka eivät sisälly opetusaineistoon. Tätä testataan erillisellä testausaineistolla.
- *Ylioppiminen (overtraining, overfitting).* Ohjatussa opetuksessa tapahtuva virhe, jossa neuroverkon painojen (vapaiden parametrien) lukumäärän ja opetusaineiston koon välillä vallitsee epäsuhta. Opetusjoukko on niin pieni painojen määrään nähden, että verkko pystyy oppimaan opetusjoukon lähes ulkoa jolloin verkon yleistyskyky huononee. Käytännöllisin tapa estää tätä tapahtumasta on valita pienin mahdollinen määrä piilokerroksen neuroneita, joka viel kuitenkin riittä antamaan hyviä tuloksia.

Luku 7

Liite A. Neuroverkkojen kaaviot Joone-ohjelmassa

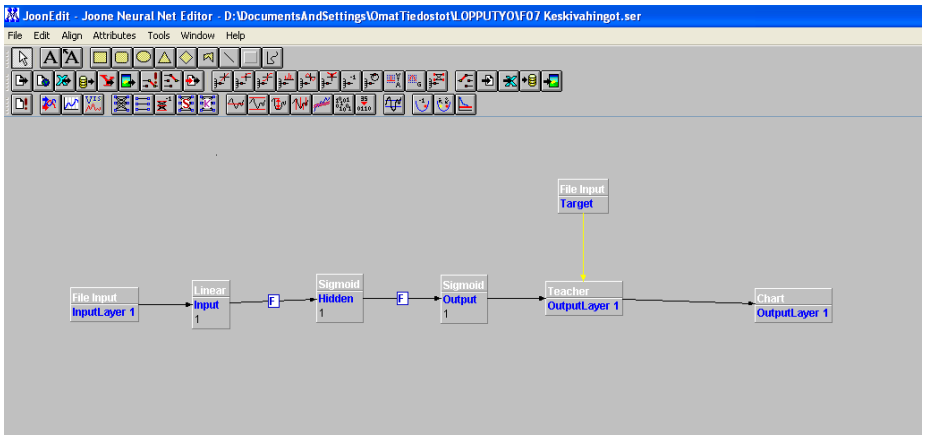
Oheisessa kuvassa (7.1) on SOM-verkon esimerkkiä varten tehty verkkomalli Joone-ohjelmassa.



Kuva 7.1: Itseorganisaatioverkon muodostaminen Joone-ohjelmassa.

Kuvassa (7.2) on keskivahinko-esimerkkiä varten muodostettu verkkomalli Joone-ohjelmassa. Kuvan yläosassa näkyvät verkkojen osat, joita voi käyttää verkkojen muodostamisessa. Verkon ensimmäisessä laatikossa on määritelty, missä havaintoaineisto on. Havainnot luetaan sisääntulokerrokseen toisessa laatikossa. Kolmas laatikko kuvastaa piilokerrosta. Siinä on vain yksi piiloneuroni, jolla on logistinen funktio aktivaatiofunktionaan. Seuraavaksi havainnot kulkevat ulostulokerrokseen, jossa myös on logistinen aktivaatiofunktio. Kirjain F laatikoiden välisissä kytköksissä tarkoittaa "Full connection", eli että kaikki edellisen kerroksen neuronit on yhdistetty kaikkiin seuraavan kerroksen neuroneihin. Tässä

esimerkissä neuroneita on piilo- ja ulostulokerroksissa kummassakin vain yksi. Viides laatikko vasemmalta opettaa neuroverkolle havainnot. Se tarvitsee myös halutut ulostulot, jotka saadaan Teacher-laatikon yläpuolella olevasta laatikosta. Tässä aineisto on itse asiassa sama kuin sisääntulossa, mutta nyt aineistosta ei luetakaan kvartaalin numeroa vaan vain vastaava keskivahinkohavainto. Kaikkein oikeanpuoleisin laatikko piirtää kuvaajan keskivirheen kehittymisestä opetuksen aikana.



Kuva 7.2: Neuroverkko keskivahinkojen mallia varten Joonessa.

Kirjallisuutta

[1] Hervé Abdi, Dominique Valentin, Betty Edelman. *Neural networks.* Sage publications, 1999.

[2] Databionic ESOM tools. http://databionic-esom.sourceforge.net/project-info.html

[3] Саймон Хайкин. *Нейронные сети.* Издательский дом Вильямс, 2006.

[4] Joone - Java Object Oriented Neural Engine. http://www.jooneworld.com

[5] O. Kaleva. *Mathematical foundations of Soft computing.* Luentomoniste

[6] P. Koikkalainen, E. Oja. *Suppea neuroverkkosanasto.*

[7] K. Loimaranta, J. Jacobsson, H. Lonka. *On the use of mixture models in clustering multivariate frequency data.* Transactions of the 21st International Congress of Actuaries, 2, 147-161, 1980.

[8] C. Mano, E. Rasa. *A discussion of modeling techniques for personal lines pricing.* Trans 27th ICA 2002.

[9] S. Rajasekaran, G. A. Vijayalakshmi Pai. *Neural networks, Fuzzy logic, and Genetic algorithms.* Prentice-Hall of India Private Limited, 2003.

S. Gayle.

[10] A. F. Shapiro, L. C. Jain. *Intelligent and other computational techniques in insurance – theory and applications.* World scientific publishing Company, 2003.

[11] E. Jurva (Taipale Engineering Oy). *Neurolaskennan mahdollisuudet.* Tekes-raportti 43/1994.

[12] M. L. Vaughn, E. Ong, S. J. Cavill, *Interpretation and knowledge discovery from a multilayer perceptron network that performs whole life assurance risk assessment.* Neural computing and applications 6, 201-213.

[13] A. Vellido, P. J. G. Lisboa, J. Vaughan. *Neural networks in business: a survey of applications 1992-1998.* Expert Systems with Applications, 17, 51-70, 1999.

[14] S. Viaene, R. A. Derrig, G. Depene. *Illustrating the Explicative Capabilities of Bayesian Learning Neural Networks for Auto Claim Fraud Detection.* Intelligent and Other Computational Techniques in Insurance, s. 365–399 World Scientific Publishing Co. Pte. Ltd., 2003

[15] B. K. Wong, T. A. Bodnovich, Y. Selvi. *Neural network applications in business: a review and analysis of the literature (1988-1995).* Decision support systems,19, 301-320, 1997.

[16] Ai Cheo Yeo, Kate A. Smith. *Implementing a data mining solution for an Automobile insurance company: Roconciling theoretical benefits with practical considerations.* Idea grouping publishing, 2003.