TYYPIN 1 DIABETEKSEN MATEMAATTISISTA ENNUSTAMISMENETELMISTÄ

Perttu Markula

Pro gradu -tutkielma
Helmikuu 2005

UNIVERSITY OF TURKU
DEPARTMENT OF MATHEMATICS
FIN-20014 TURKU
FINLAND

TURUN YLIOPISTO
Sovelletun matematiikan laitos

MARKULA, PERTTU: Tyypin 1 diabeteksen matemaattisista ennustamismenetelmistä
Pro gradu -tutkielma, 52 s., 27 liites.
Sovellettu matematiikka
Tammikuu 2005

Tutkielmassa käsitellään matemaattisia ennustamismenetelmiä, jotka soveltuvat tyypin 1 diabeteksen ennustamiseen.

Aluksi esitellään menetelmiä, jotka soveltuvat puuttuvia havaintoja sisältävien aineistojen paikkaamiseen. Paikattua aineistoa on mahdollista analysoida useilla tavallisilla tilastollisilla menetelmillä, jotka sopivat täydellisiin aineistoihin. Seuraavaksi pyritään mallintamaan aineistoa semiparametrisilla komponenttimalleilla (eng. mixture model), jolloin mallin muotoa ei ole tiukasti etukäteen rajoitettu. Sen jälkeen sovelletaan kolmea luokittelevaa ennustajaa: logistista regressiomallia, eteenpäinsyöttävää yhden piilotason neuroverkkoa ja SVM-menetelmää (eng. support vector machine).

Esiteltäviä menetelmiä on sovellettu todelliseen aineistoon, joka on kerätty Turun yliopistossa käynnissä olevassa tutkimusprojektissa. Projektin tavoitteena on oppia ennustamaan ja ehkäisemään tyypin 1 diabetesta (Type 1 diabetes prediction and prevention project, lyh. DIPP-projekti). Erityisesti projektissa on pyritty löytämään uusia tuntemattomia taudinaiheuttajia. Tässä tutkielmassa paneudutaan sen sijaan kerätyn havaintoaineiston matemaattisiin analysointimenetelmiin.

Parhaat ennusteet saatiin perinteisellä logistisella regressiomallilla. Tutkielmassa kuitenkin todetaan, että tulevaisuudessa on mahdollista löytää parempia ennustajia parantamalla muita edellä mainittuja menetelmiä. Erityisesti SVM-menetelmä ansaitsisi lisähuomiota, sillä tässä tutkielmassa sitä sovellettiin vain kaikkein yksinkertaisimmassa muodossa.

Asiasanat: matemaattiset ennustamismenetelmät

Sisältö

1 Johdanto

Tyypin 1 diabetekseen[1] ei ole olemassa parannuskeinoa eikä sairauden puhkeamisen syitä vielä tiedetä. Monia epäiltyjä taudinaiheuttajia on kuitenkin tutkittu. Diabeteksen ennustaminen on tärkeää, jotta voidaan kehittää ennaltaehkäiseviä hoitoja. Turun yliopistossa toimii DIPP-projekti, joka tähtää juuri diabeteksen ennustamis- ja ehkäisymenetelmien kehittämiseen.[4]

Diabetesta voidaan pitää Suomessa kansantautina, sillä missään muussa maassa se ei ole näin yleinen, jopa seitsemän lasta tuhannesta sairastuu lapsuusikänsä aikana. Suomessa sairastuneita on yhteensä noin 40000. Huomionarvoista on myös se, että sairauden ilmaantuvuus on noussut Suomessa viimeisen 50 vuoden aikana yli kahden prosentin vuosivauhtia. Diabetekseen voi sairastua myös aikuisena, kolmannes diabeetikoista on sairastunut 25 ikävuoden jälkeen.[4, 3]

On löydetty monia asioita, joiden epäillään, tai on havaittu, olevan yhteydessä sairauden kehittymiseen, esimerkiksi tietyt geenit, varhainen insuliinivaste, verestä mitatut diabetesvasta-aineet, virusinfektiot, ruokavalio. Yleensä diabeteksen mahdollisia syitä lienee tutkittu erillisinä. Kuitenkin, koska tutkittavia ilmiöitä on useita, diabeteksen ennustamiseksi DIPP-projektissa kerätty havaintoaineisto sisältää paljon muuttujia. Myös tutkittuja henkilöitä on paljon, koska diabeetikkoja halutaan tutkia jo ennen diabeteksen puhkeamista. Tällöinhän tutkimukseen tulee mukaan pääasiassa henkilöitä, jotka eivät koskaan sairastu. Havaintoaineiston laajuus on nostanut esiin ajatuksen etsiä ja soveltaa nykyaikaisempia analyysimenetelmiä, sillä on mahdollista, että niitä voidaan käyttää ennustamiseen paremmin kuin tähän asti sovellettuja perinteisiä tilastollisia malleja.[4]

Havaintoaineiston ominaispiirteitä muuttujien ja havaintojen suuren lukumäärän lisäksi ovat muuttujien jakaumien epänormaalisuus ja puuttuvat havainnot. Aluksi (luvuissa 4-5) esitellään kaksi tapaa suhtautua puuttuviin

[1] Tyypin 1 diabetesta kutsutaan myös nuoruustyypin diabetekseksi. Tässä tutkielmassa kirjoitetaan jatkossa pelkästään diabetes, mutta sillä tarkoitetaan nimenomaan tyypin 1 diabetesta.

havaintoihin: voidaan paikata havaintoaineistoa tai rakentaa malleja, jotka sallivat puuttuvat arvot.

Toiseksi (luvuissa 6-7) esitellään ratkaisuja epänormaalisuusongelmaan. Monet perinteiset tilastotieteen menetelmät soveltuvat vain normaalijakaumaa noudattaville muuttujille. Diabetekseen liittyvät muuttujat ovat kuitenkin niin epänormaaleja, että on harkittava järjestyslukuihin siirtymistä, muunnoksia (esim. logaritmimuunnos) tai muunlaisia malleja. Esitellään komponenttimallit ja EM-algoritmi.

Kolmanneksi (luvussa 9) sovelletaan logistista regressiota sairastumisen ennustamiseen.

Neljänneksi (luvussa 10) tutkielmassa käsitellään neuroverkkoja, jotka pystyvät approksimoimaan epälineaarisia funktioita. Yritetään rakentaa neuroverkko havaintoaineistoille, jossa on puuttuvia havaintoja.

Viidenneksi (luvussa 11) esitellään lyhyesti SVM-menetelmä, jotka voi luokitella kuuluvaksi ns. tiedonlouhinta -menetelmiin.

2 Havaintoaineisto

DIPP-projektin seurannassa on ollut mukana vain kohtalaisen tai korkean geneettisen sairastumisriskin omaavia lapsia. Lapset käyvät antamassa verinäytteen kuuden kuukauden väliajoin. Jos lapsella havaitaan saarekesoluvasta-aineen (lyh. ica) olevan koholla, niin käyntiväli lyhennetään kolmeen kuukauteeen. Tällöin verinäytteestä aletaan määrittää myös muita vasta-aineita (iaa, ia2, gad). Insuliinivasta-aineen (iaa) määritysmenetelmä on muuttunut tiettynä ajankohtana toiseksi (iaav tarkoittaa vanhaa menetelmää). Jos muuttujien nimien perässä esiintyy ylimääräisiä numeroita, niin yksi viittaa tällöin lapsen viimeiseen tutkimuskäyntiin ja kaksi toiseksi viimeiseen tutkimuskäyntiin.

Ennen sairauden puhkeamista (yhteensä n.50 on sairastunut) lapset ovat ns. vasta-aineposititiivisia eli viimeksi otetuissa verinäytteissä (3-6kk ennen diagnoosia) on havaittavissa vasta-aineiden korkeita pitoisuuksia. Näin on

poikkeuksetta tähän asti käynyt. Toisilla sairastuneista vasta-aineita on ollut koholla jo vuosia, toisilla sairaus voi puhjeta nopeasti, jopa muutamassa kuukaudessa.

Monet vasta-ainepositiiviset eivät koskaan sairastu. Tarkemman kuvan vasta-ainepositiivisen lapsen tilanteesta antaa sokerirasitustesti, johon kaikki vasta-ainepositiiviset kutsutaan. Testi uusitaan 3-6 kuukauden välein. Ennen sairastumista sokerirasitustestissä havaitaan useimmiten varhaisen insuliinivasteen (fpir) heikentymistä. Tällöin haima ei pysty enää pysty normaalilla nopeudella tuottamaan vereen insuliinia, kun vereen ilmaantuu sokeria. Testissä annetaan glukoosia suonensisäisesti, jonka jälkeen otetaan verinäytteet tietyillä ajan hetkillä. Varhainen insuliinivaste on insuliinipitoisuuksien summa yhden ja kolmen minuutin kohdalla otetuista näytteistä. Ennen rasitustestiä lapsilta mitataan verensokeri. Jos se on selvästi koholla ennen rasitustestiä, niin tehdään diabetesdiagnoosi ja lapsi lähetetään hoitoon. Yleensä diagnoosi tehdään kuitenkin tutkimusprojektin ulkopuolella, kun perheet hakeutuvat lääkärin vastaanotolle oireiden takia. Sokerirasitustestin raskauden ja kalleuden takia henkilöitä on pyydetty tutkimukseen vain, jos heillä on havaittu tiettyjä vasta-aineita veressä.

Jos muuttujien nimet alkavat log-kirjaimilla, niin muuttujille on tehty logaritmimuunnos.

DIPP-projektissa on kerätty kosolti muutakin tietoa tutkittavista, mutta edellä kuvatut ovat keskeisimpiä ja tässä tutkielmassa rajoitutaan niihin.

3 Tutkielmassa käytetyt merkinnät

Vektorit ovat aina pystyvektoreita. Selittävien muuttujien muodostama satunnaisvektori on X. Vektorin X yksittäinen komponentti on X_j. Satunnaisvektorissa X on yhteensä J komponenttia: $X = (X_1, X_2, \ldots, X_J)^T$. Selitettävien muuttujien vektoria merkitään kirjaimella $Y = (Y_1, Y_2, \ldots, Y_H)^T$, ja vektorin yhtä komponenttia Y_h.

Satunnaisvektorien havaittuja arvoja merkitään x_i ja y_i. Havaintojen lu-

kumäärä olkoon N, jolloin havainnoista voidaan muodostaa $N \times J$-matriisi $\mathbf{X}$ ja $N \times H$-matriisi $\mathbf{Y}$. Tällöin matriisin X i :s rivi on x_i^T. Matriisit siis lihavoidaan, vektoreita ei.

Selitettävää vektoria Y pyritään ennustamaan, merkitään ennustetta $\hat{Y}$:lla. Ennustamiseen käytettävää funktiota merkitään $\hat{Y} = f(X)$. Satunnaisvektorin tiheysfunktio on $p(X)$ ja todennäköisyys, että satunnaismuuttuja saa tietyn arvon, on $P(X = x)$.

4 Puuttuvat havainnot

Tilastollinen havaintoaineisto muodostuu taulukosta, jossa rivit edustavat havaintoja (esim. ihmisiä) ja sarakkeet muuttujia (esim. pituus). Taulukon soluissa olevat tiedot ovat reaalilukuja, jotka voivat esittää jatkuvia muuttujia (esim. pituus) tai kategorisia muuttujia (esim. sukupuoli voi olla 0 tai 1).

Puuttuvat havainnot tällaisessa havaintoaineistossa ovat ongelmallisia, eihän puuttuvalla luvulla voida laskea mitään. On hyvin tavallista, että havaintoaineistoa ei saada kerättyä täydellisesti. Esimerkiksi kyselyyn vastaaja saattaa helposti jättää jonkin kohdan tyhjäksi tai lapsi kieltäytyä verinäytteen otosta. Tällöin on harkittava seuraavia vaihtoehtoja [11, s. 6]:

1) Jos puuttuvia lukuja on hyvin vähän, niin ko. rivit voidaan jättää analyysistä pois.

2) Täytetään puuttuvat tiedot joillain luvuilla (paikkaus l. imputointi).

3) Painotetaan enemmän sellaisia havaintoja joiden parissa puuttuminen on tavallisempaa. Esim. jos rikkaat jättäisivät useammin kertomatta tulonsa, niin väestön keskituloja arvioitaessa rikkailta kerättyjä havaintoja painotettaisiin enemmän kuin muiden.

4) Käytetään malleja, jotka on rakennettu ottamaan huomioon puuttuvat mittaustulokset.

Puuttuviin havaintoihin johtavien syiden pohdinta on havaintoaineiston analyysointimenetelmien valinnan ja tulosten tulkinnan kannalta oleellista.

Otanta voi olla syynä puuttuviin havaintoihin, mutta tällainen syyhän on täysin kontrolloitavissa. Ajatellaankin nyt, että puuttuminen ei johdu otantaasetelmasta. Olkoon kaksi satunnaismuuttujaa X_1 ja X_2. Jos muuttujasta X_1 on saatu mitattua kaikki N havaintoa, mutta muuttujasta X_2 vain $u < N$ havaintoa, niin puuttumisen syyt voidaan luokitella seuraavalla tavalla. Puuttumisen todennäköisyys joko

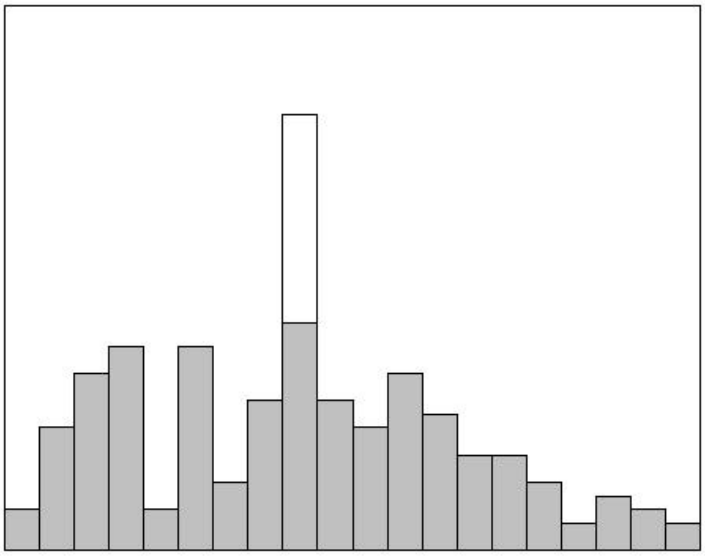
1. riippuu ainakin X_2:sta ja mahdollisesti myös X_1:stä (PEOS) tai
2. ei riipu X_2:sta mutta riippuu X_1:stä (POS) tai
3. ei riipu X_2:sta eikä X_1:stä (POTS).

Kolmannessa tapauksessa sanotaan, että puuttuminen on täysin satunnaista (POTS). Tällöin havaitut X_2:n arvot ovat u:n kokoinen satunnaisotos N:n kokoisesta otoksesta. Toisessa tapauksessa puuttuminen ei ole täysin satunnaista (POS), vaan havaitut X_2:n arvot ovat satunnaisotos X_1:n määräämissä luokissa. Jos puuttumisen todennäköisyys riippuu X_2:stä, niin on kaikkein vaikein tilanne, tällöin puuttuminen ei ole mitenkään satunnaista (PEOS). Valitettavasti tällainen tilanne, jossa puuttumisen syyt ovat ohittamattomia, on hyvin yleinen. [11, s. 14]

4.1 Yksiarvopaikkaus

Havaintoaineiston puuttuvien lukujen paikkaus on houkutteleva tapa käsitellä epätäydellistä aineistoa, koska paikkauksen jälkeen voisi ajatella unohtavansa aineistossa olleet puutteet ja tehdä analyysit ja päätelmät kuten täydellisellekin aineistolle. Tämän vuoksi menettely on vaarallinen, koska paikkauksen seurauksena havaintoaineistoon sovitettujen tilastollisten mallien parametrien estimaatit voivat olla hyvin harhaisia. Tässä luvussa esitellään erilaisia paikkausmenetelmiä kirjan [11, s. 60-61] mukaan.

a) Puuttuva havainto voidaan korvata ko. muuttujan havaittujen arvojen keskiarvolla. Näin paikatusta aineistosta lasketut varianssit ovat todellisia



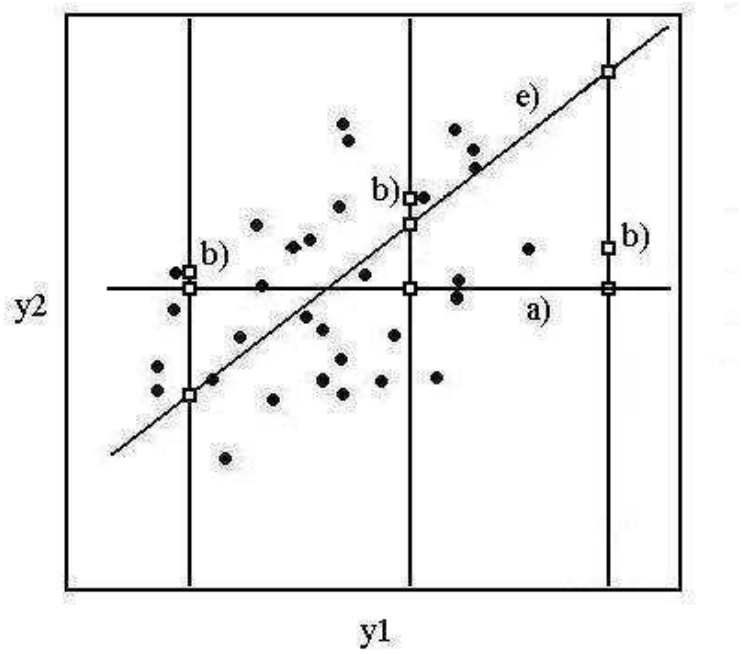
Kuva 1: Keskiarvopaikkaus. Histogrammi kuvaa keinotekoisen muuttujan jakaumaa. Tummat palkit ovat havaittuja arvoja ja tyhjä palkki kuvaa puuttuvia havaintoja, jotka on paikattu jakauman keskiarvolla. Koska kaikki puuttuvat arvot eivät todellisuudessa sijaitse jakauman keskellä, keskiarvopaikkaus aliarvioi muuttujan varianssia.

variansseja pienemmät, koska paikkaukseen käytetyt arvot otettiin jakaumien keskeltä (ks. kuva 1). Myös muuttujien välisiä kovariansseja aliarvioidaan.

b) Hot deck -menetelmässä valitaan otoksesta havainto, joka muistuttaa eniten paikattavaa havaintoa. Puuttuvan arvon tilalle sijoitetaan valitun havainnon vastaava arvo.

c) Otantatutkimuksissa voidaan puuttuvat havainnot korvata sellaisilla havainnoilla, jotka eivät alunperin tulleet valituiksi otokseen. Esimerkiksi, jos jokin kotitalous ei vastannut kyselyyn, niin saatetaan valita jokin toinen kotitalous samasta korttelista. Tällöin tulee muistaa, että puuttuvien havaintojen ja paikattujen arvojen välillä voi olla systemaattista eroa.

d) Cold deck -menetelmässä puuttuvien havaintojen tilalla käytetään jotain vakiota, joka on saatu jostain tutkimuksen ulkopuolelta, esimerkiksi toisesta samantapaisesta tutkimuksesta. Mitään tyydyttävää teoriaa näin paikatun aineiston analysoimiseksi ei ole olemassa.



Kuva 2: Buckin menetelmä. Muuttujat y1 ja y2 ovat keinotekoisia. Mustat pisteet kuvaavat täydellisiä havaintoja. Kolmen havainnon tapauksessa on saatu mitattua vain muuttujan y1 arvo ja muuttujan y2 arvo puuttuu. Näitä kolmea havaintoa kuvataan pystysuorilla suorilla. Neliöt ovat paikattuja arvoja. Buckin menetelmä paikkaa puuttuvat havainnot kuvaan piirretyn regressiosuoran e) antamilla ennusteilla. Kuvaan on piirretty myös muuttujan y2 keskiarvosuora a) ja keskiarvopaikatut arvot sekä hot deck -menetelmällä paikatut arvot b).

e) Buckin menetelmässä lasketaan ensin aineiston täydellisestä osasta estimaatit keskiarvovektorille ja kovarianssimatriisille. Jos aineisto on jakautunut multinormaalisti, niin estimaattien avulla voidaan laskea arvot puuttuville havainnoille lineaarisella regressiolla, havainto kerrallaan. Kun regressiomalliin sijoitetaan muuttujien tilalle havaitut arvot, voidaan laskea ennusteet puuttuville arvoille. Buckin menetelmän huonoja puolia on, että sekin aliarvioi variansseja ja kovariansseja, olettaa muuttujien riippuvuuden olevan lineaarista ja antaa ennusteita, jotka ovat suurempia (tai pienempiä) kuin mitkään havaitut arvot (ks. kuva 2).

f) Buckin menetelmää voidaan yrittää parantaa lisäämällä ennusteeseen

satunnaisuutta. Valitaan satunnaisen termin jakauma normaaliksi keskiarvolla nolla ja varianssilla, joka on yhtä suuri kuin regressiomallin jäännösvarianssi. Tällainen menettely kasvattaa varianssien ja kovarianssien estimaatteja.

4.2 Moniarvopaikkaus

Moniarvopaikkausta sanotaan myös imputoinniksi (eng. multiple imputation). Se on teoreettisesti hyväksyttävissä oleva tapa paikata aineistoa. Kaikissa edellisissä menetelmissä puuttuva arvo paikattiin vain yhdellä luvulla. Näiden menetelmien yleinen ongelma oli varianssien aliarvioiminen. Moniarvopaikkauksessa tämä ongelma pyritään välttämään korvaamalla tyhjät kohdat aineistossa vektorilla, jossa on $M \geq 2$ arvoa. Sen jälkeen voidaan muodostaa M eri havaintoaineistoa ja analysoida näistä jokainen tavallisilla täydelliseen ainestoon sopivilla menetelmillä. Jos nuo M eri paikkausta ovat toistettuja satunnaisia poimintoja yhdestä mallista, niin myös eri tulokset voidaan yhdistää. Yhdistetyistä tuloksista voidaan tehdä todenmukaisia päätelmiä, jos tuo puuttumismekanismin malli on oikein valittu. Menetelmän edut ovat selkeät: se antaa luotettavampia tuloksia kuin aiemmin esitellyt menetelmät. Ainoa huono puoli on paikkaamisen työläys, mutta tämäkin vaikeus on voitettu tietotekniikan kehityksen myötä.

4.2.1 Arvojen generointi

Paikkauksessa käytettävät arvot saadaan periaatteessa bayesiläisestä ennustejakaumasta:

$$p(Y_{\text{puu}}|Y_{\text{hav}}) = \int p(Y_{\text{puu}}|Y_{\text{hav}}, \theta)p(\theta|Y_{\text{hav}})d\theta, \tag{1}$$

missä θ on parametrivektori. Jakaumaa ei pystytä kuitenkaan laskemaan eksplisiittisesti, vaan on käytettävä iteratiivisia menetelmiä. Tähän soveltuvia ovat ns. Markovin ketju Monte Carlo -menetelmät: Gibbsin otanta, aineiston täydennys (eng. data augmentation) ja Metropolis-Hastings-algoritmi. Näissä menetelmissä ajatuksena on muodostaa satunnaisvektorien jono

$\{X^{(1)}, X^{(2)}, \ldots, X^{(t)}, \ldots\}$, jossa jokainen muuttuja riippuu jollain tavalla edellisestä muuttujista ja jossa $X^{(t)}$ lähestyy stationaarista jakaumaa X, kun t lähestyy ääretöntä. Näin ei tarvitse generoida satunnaislukuja suoraan jakaumasta X, vaan voi poimia lukuja muodostetusta jonosta. Jonon luvut ovat likimain jakautuneet jakauman X mukaan, kun t on riittävän suuri. On kuitenkin muistettava, että useita satunnaislukuja generoitaessa ei voida ottaa jonon peräkkäisiä lukuja, sillä nehän riippuvat toisistaan. Gibbsin otannassa satunnaisvektorien jonossa seuraavana $(t+1)$ olevan satunnaisvektorin komponenttien arvot saadaan jonon edellisestä satunnaivsektorista poiminnoilla ehdollisista jakaumista:

$$\begin{aligned} X_1^{(t+1)} &\sim p(X_1|X_2^{(t)}, X_3^{(t)}, \ldots, X_J^{(t)}) \\ X_2^{(t+1)} &\sim p(X_2|X_1^{(t+1)}, X_3^{(t)}, \ldots, X_J^{(t)}) \\ &\vdots \\ X_J^{(t+1)} &\sim p(X_J|X_1^{(t+1)}, X_2^{(t+1)}, \ldots, X_{J-1}^{(t+1)}). \end{aligned} \tag{2}$$

Tätä voidaan soveltaa puuttuvien arvojen laskemiseen valitsemalla $J = 2$, $X_1 = Y_{puu}$ ja $X_2 = \theta$. Tällöin jonossa seuraavana (t+1) olevan vektorin arvo saadaan kahdella poiminnalla ehdollisista jakaumista:

$$\begin{aligned} Y_{puu}^{(t+1)} &\sim p(Y_{puu}|Y_{hav}, \theta^{(t)}) \\ \theta^{(t+1)} &\sim p(\theta|Y_{hav}, Y_{puu}^{(t+1)}). \end{aligned} \tag{3}$$

[13, s. 68-78]

4.2.2 Aineiston analysointi

Esitellään seuraavaksi M:n eri analyysin tulosten yhdistäminen. Oletetaan, että aineisto on paikattu ja analysoitu M kertaa jollakin täydelliselle aineistolle sopivalla menetelmällä. Oletetaan myös, että $\hat{\theta}_m$ ja $\hat{W}_m$, missä $m = 1, \ldots, M$, ovat toistetuista paikkauksista saaduista aineistoista lasketut estimaatit ja niiden varianssit. Tällöin yhdistetty estimaatti on

$$\bar{\theta}_M = \sum_{m=1}^{M} \frac{\hat{\theta}_l}{M}. \tag{4}$$

Estimaatin vaihtelulla on kaksi komponenttia: paikkauksen sisäisistä variansseista laskettu keskiarvo

$$\bar{W}_M = \sum_{m=1}^{M} \frac{\hat{W}_l}{M} \tag{5}$$

ja estimaattien varianssi eri paikkausten välillä

$$B_M = \frac{\sum_{m=1}^{M} (\hat{\theta}_m - \bar{\theta}_M)^2}{M-1}. \tag{6}$$

Kokonaisvarianssi estimaatille $\bar{\theta}_M$ on

$$T_M = \bar{W}_M + (1 + M^{-1})B_M. \tag{7}$$

θ voi olla vektori, jolloin $\theta^2 = \theta^T\theta$. Jos θ on kuitenkin skalaari, niin $\frac{\theta - \bar{\theta}_M}{\sqrt{T_M}}$ noudattaa t-jakaumaa, jonka vapausaste on $v = (M-1)[1 + \frac{1}{M+1}\frac{\bar{W}_m}{B_M}]^2$.[11, s. 257]

4.3 Sovellus: moniarvopaikkaus vasta-aineille

DIPP-projektissa on tutkittu sokerirasitustestillä haiman kykyä tuottaa nopeasti insuliinia verenkiertoon, kun verensokeripitoisuus nousee. Otetaan tässä luvussa aineistoksi Turussa tutkittujen lasten kahden viimeisimmän mittauskerran varhainen insuliinivaste. Vain sellaisten lasten, jotka ovat käyneet vähintään kerran sokerirasituksessa (145 kpl), mittaustulokset otetaan nyt mukaan tarkasteluun. Varhaiselle insuliinivasteelle tehdään logaritmimuunnos, jotta se olisi paremmin normaalijakaumaoletuksen mukainen.

Ensin tutkitaan kahden paikkausmenetelmän kykyä paikata aineistoa. Otetaan täydellinen aineisto eli poistetaan ne lapset aineistosta, jotka ovat käyneet vain yhden kerran sokerirasituksessa. Jäljelle jää 90 lasta. Poistetaan tietoja satunnaisesti niin, että poistamisen todennäköisyys on p. Sen jälkeen tehdään kaksi aineistoa: keskiarvopaikattu ja moniarvopaikattu. Moniarvopaikkaus tehdään kuitenkin nimensä vastaisesti niin, että käytetään vain yhtä arvoa paikkaamiseen. Paikatuista aineistoista lasketaan todellisten arvojen ja paikattujen arvojen erojen neliöiden summat. Neliösummia käytetään paikkausvirheen arvioimiseen. Se menetelmä, jonka paikkausvirheen

neliösumma on pienempi on parempi. Myös muita virhefunktioita voisi käyttää, mutta neliösumma on tavallisin. Tulosten tarkkuuden vuoksi havaintojen satunnainen poisto ja paikkaukset tehdään 100 kertaa ja lasketaan neliösummien keskiarvot. Osoittautuu, että moniarvopaikkauksessa paikkausvirheiden neliösumma on pienempi kuin keskiarvopaikkauksessa (ks. taulukko 1). Satunnaiset poistot tehtiin todennäköisyyksillä 0,05, 0,1, 0,2 ja 0,3. Moniarvopaikkauksessa käytetty arvojen generointimenetelmä on Gibbsin otanta, joka on esitetty sivulla 9. Käytetty ohjelmisto olettaa, että aineiston muuttujat ovat multinormaalisti jakautuneet.

Käytetään koko aineistoa (145 lasta), joka on moniarvopaikattu. Tarkastellaan viimeistä edellisen mittauskerran varhaisen insuliinivasteen kykyä selittää viimeisen käynnin mittaustulosta. Selityskykyä arvioidaan regressiomallilla, joka on sovitettu aineistoon käyttämällä pienimmän neliösumman menetelmää. Mallin mukainen havaittu merkitsevyystaso (l. p-arvo) on pienempi kuin 0,01, joten edellinen mittaustulos selittää seuraavaa tilastollisesti merkitsevästi (yleensä rajana pidettään p-arvoa 0,05). Vertailun vuoksi sovitetaan regressiomalli myös täydelliseen aineistoon (90 lasta). Tällöin havaittu merkitsevyystaso on myös pienempi kuin 0,01. Eli tässä aineistossa moniarvopaikkaus antaa samansuuntaisen tuloksen kuin täydellisen mutta suppeamman aineiston analysointi.

Vaikka moniarvopaikkaus osoitetusti paikkaa aineistoa paremmin kuin keskiarvopaikkaus, niin tässä ei pystytty osoittamaan moniarvopaikkauksen hyödyllisyyttä. Tämä johtui aineiston ominaispiirteistä ja ohjelmistojen puutteista. Aineistossa oli vain kaksi muuttujaa. Olisi ollut parempi käyttää aineistoa, jossa on useampia muuttujia, sillä moniarvopaikkauksen hyöty havaitaan, kun puuttuvia arvoja on eri lapsilla eri muuttujissa. Tällöin täydellisen aineiston koko saattaa pudota huomattavasti, jolloin regressioanalyysin tapaiset täydellisen aineiston analysointimenetelmät saattavat antaa huonompia tuloksia. Parempaa aineistoa ei ollut mahdollista saada vastaainetuloksista, koska ne eivät noudata tarpeeksi hyvin normaalijakaumaa. Kolmanneksi viimeisen sokerirasitustestin tulostakaan ei otettu mukaan, kos-

puuttumisen todennäköisyys	keskiarvopaikkausen neliösummat	moniarvopaikkauksen neliösummat
0,05	6,77	1,93
0,1	13,1	3,65
0,2	26,6	6,76
0,3	40,5	8,90

Taulukko 1: Paikkausmenetelmien vertailu. Aineistossa oli 90 havaintoa ja kaksi muuttujaa: viimeisen ja viimeistä edellisen tutkimuskäynnin varhaiset insuliinivasteet. Havainnot poistettiin satunnaisesti tietyllä todennäköisyydellä. Keskiarvopaikkauksessa puuttuvat kohdat paikattiin muuttujan havaituista arvoista lasketulla keskiarvolla. Moniarvopaikkauksessa käytettiin Gibbsin otantaa ja jokainen puuttuva tieto paikattiin vain kerran.

ka sitä ei ole olemassa yhdeltäkään lapselta, joka ei olisi ollut toiseksi viimeisessä sokerirasitusmittauksessa. On toivottavaa, että tulevaisuudessa ohjelmistot kehittyvät sellaisiksi, että ne toimivat myös sellaisten muuttujien kanssa, jotka eivät noudata normaalijakaumaa. Tällöin moniarvopaikkaus voisi osoittautua hyödylliseksi vasta-ainemuuttujien analysoinnissa.

5 Uskottavuusfunktio ja puuttuvat havainnot

Merkitään havaintoaineistoa $X = (X_{\text{hav}}, X_{\text{puu}})$, missä X_{hav} on aineiston havaitut arvot ja X_{puu} puuttuvat arvot. Havaintoaineiston tilastollinen malli on $p(X|\theta)$, missä θ on parametrivektori. Parametrien määrittämistä havaintoaineiston perusteella sanotaan tilastollisen mallin sovittamiseksi. Marginaalijakauma havaituille arvoille saadaan integroimalla yli puuttuvien arvojen:

$$p(X_{\text{hav}}|\theta) = \int p(X_{\text{hav}}, X_{\text{puu}}|\theta) dX_{\text{puu}}. \tag{8}$$

Parametrin uskottavuus on mikä tahansa θ:n funktio, joka on suoraan verrannollinen tiheysfunktioon $p(X_{\text{hav}}|\theta)$:

$$L(\theta|X_{\text{hav}}) \propto p(X_{\text{hav}}|\theta), \tag{9}$$

kun oletetetaan, että puuttuvat havainnot ovat satunnaisia (POS tai POTS, katso s.5). Tämä malli ei näin ollen riipu puuttuvista havainnoista. [11, s. 89]

Malliin voidaan ottaa mukaan indikaattorimatriisi R, joka ajatellaan satunnaismuuttujaksi. Se saa arvoja yksi ja nolla sen mukaan onko arvo havaittu vai puuttuva. Yhteisjakauma voidaan kirjoittaa havaintoaineiston X jakauman ja indikaattorimatriisin R ehdollisen jakauman tulona

$$p(X, R|\theta, \psi) = p(X|\theta)p(R|X, \psi). \tag{10}$$

Tulon jälkimmäistä termiä sanotaan puuttumismekanismin jakaumaksi. Joskus puuttumismekanismi on tunnettu eikä vektoria ψ tarvita. Oikeastaan havaintoaineisto koostuu muuttujien (X_{hav}, R) arvoista. Tämän jakauman tiheysfunktio saadaan integroimalla:

$$\begin{aligned} p(X_{\text{hav}}, R|\theta, \psi) &= \int p(X_{\text{hav}}, X_{\text{puu}}, R|\theta, \psi) dX_{\text{puu}} \\ &= \int p(X_{\text{hav}}, X_{\text{puu}}|\theta) p(R|X_{\text{hav}}, X_{\text{puu}}, \psi) dX_{\text{puu}}. \end{aligned} \tag{11}$$

Mikä tahansa funktio, joka on edelliseen tiheysfunktioon suoraan verrannollinen, on parametrivektorin (θ, ψ) uskottavuus:

$$L(\theta, \psi|X_{\text{hav}}, R) \propto p(X_{\text{hav}}, R|\theta, \psi). \tag{12}$$

Tämän uskottavuusfunktion käyttö ei oleta, että puuttuvat arvot ovat satunnaisia, vaan ottaa huomioon puuttumismekanismin. [11, s. 89-90]

Puuttumismekanismin ollessa riippumaton puuttuvista havainnoista eli

$$p(R|X_{\text{hav}}, X_{\text{puu}}, \psi) = p(R|X_{\text{hav}}, \psi) \tag{13}$$

saadaan kaavan (11) avulla

$$\begin{aligned} p(X_{\text{hav}}, R|\theta, \psi) &= p(R|X_{\text{hav}}, \psi) \int p(X_{\text{hav}}, X_{\text{puu}}|\theta) dX_{\text{puu}} \\ &= p(R|X_{\text{hav}}, \psi) p(X_{\text{hav}}|\theta), \end{aligned} \tag{14}$$

jolloin parametrivektori (θ, ψ) voidaan jakaa komponentteihinsa. Ehto (14) on yhtäpitävä sen kanssa, että puuttuminen on satunnaista (POS tai POTS),

ja sen ollessa voimassa voidaan parametrivektoriin θ liittyvä uskottavuusperusteinen päättely tehdä yksinkertaisemman uskottavuusfunktion (8) avulla. [11, s. 90]

Funktio $L_{\text{puu}}(\theta|X_{\text{puu}})$ ei ole uskottavuusfunktio, koska argumentti sisältää satunnaismuuttujan X_{puu}, kun taas funktiossa $L(\theta|X_{\text{hav}})$ argumentti on tunnettu havaittu osa aineistosta. Satunnaisuus (POS tai POTS) on siis sekä riittävä että välttämätön ehto uskottavuusperusteiselle päättelylle funktiosta $L(\theta|X_{\text{hav}})$. Jos ehto PEOS täyttyy, niin on käytettävä funktiota (12).

5.1 Suurimman uskottavuuden estimointi eksponenttijakaumalle

Havaintojen ollessa riippumattomia ja noudattaessa yksiulotteista eksponenttijakaumaa, jonka odotusarvo on θ, on havaintojen jakauma

$$p(X|\theta) = \prod_{i=1}^{N} p(x_i|\theta) = \theta^{-N} \exp\left(-\sum_{i=1}^{N} \frac{x_i}{\theta} \right). \tag{15}$$

Jos havainnot tehdään todennäköisyydellä ψ ja niitä tehdään vain $u < N$ kappaletta, niin indikaattorimatriisi R noudattaa Bernoullijakaumaa

$$p(R|X, \psi) = \psi^u (1 - \psi)^{N-u} \tag{16}$$

ja yhteisjakauma on

$$p(X_{\text{hav}}, R|\theta, \psi) = \psi^u (1 - \psi)^{N-u} \theta^{-u} \exp\left(-\sum_{i=1}^{u} \frac{x_i}{\theta} \right). \tag{17}$$

Koska parametrit θ ja ψ ovat erillisiä, niin päätelmät parametrista θ voidaan tehdä jättämällä jakaumasta ψ:tä sisältävä osa pois. Tällöin uskottavuusfunktion $L(\theta|X_{\text{hav}})$ derivaatan nollakohta eli suurimman uskottavuuden estimaatti parametrille θ on $\hat{\theta} = \sum_{i=1}^{u} x_i/u$. [11, s. 91]

6 EM-algoritmi

EM-algoritmia voidaan käyttää puuttuvia havaintoja sisältävän aineiston analysointiin. Esitellään EM-algoritmi ensin yleisessä muodossa ja myöhem-

min luvussa 7 komponenttimallin parametrien estimoimiseksi.

Oletetaan tässä luvussa, että puuttuvat havainnot ovat satunnaisia (POS tai POTS, katso s.5) ja että tavoitteena on etsiä uskottavuusfunktion

$$L(\theta|X_{\text{hav}}) = \int p(X_{\text{hav}}, X_{\text{puu}}|\theta)dX_{\text{puu}} \tag{18}$$

maksimi.

Puuttuvia kohtia sisältävää havaintoaineistoa voisi ajatella käsiteltävän seuraavaan tapaan [2]:

1. Korvaa puuttuvat arvot estimoiduilla arvoilla.
2. Estimoi mallin parametrit.
3. Estimoi uudelleen puuttuvat arvot olettaen, että uudet parametrien estimaatit ovat oikein.
4. Estimoi mallin parametrit uudelleen. Siirry kohtaan yksi kunnes algoritmi suppenee.

Tämä on EM-algoritmi [2] siinä tapauksessa, jos logaritminen uskottavuusfunktio $l(\theta|X) = \ln(L(\theta|X))$ on lineaarinen puuttuvien arvojen X_{puu} suhteen.

Täydellisen aineiston tiheysfunktio voidaan jakaa osiin

$$p(X|\theta) = p(X_{\text{hav}}|\theta)p(X_{\text{puu}}|X_{\text{hav}}, \theta). \tag{19}$$

Ottamalla puolittain logaritmi päädytään maksimoitavana olevan uskottavuusfunktion logaritmiin

$$l(\theta|X_{\text{hav}}) = l(\theta|X) - \ln p(X_{\text{puu}}|X_{\text{hav}}, \theta). \tag{20}$$

Laskemalla puolittain odotusarvo puuttuville havainnoille X_{puu} ehdolla havaitut arvot X_{hav} ja parametrin estimaatti $\theta^{(t)}$ iteraation hetkellä t, saadaan

$$l(\theta|X_{\text{hav}}) = Q(\theta|\theta^{(t)}) - H(\theta|\theta^{(t)}), \tag{21}$$

[2] Algoritmin nimi tulee englannin kielen sanoista expectation ja maximization, joista ensimmäinen viittaa puuttuvien arvojen estimoimiseen ja toinen mallin parametrien estimoimiseen.

missä

$$Q(\theta|\theta^{(t)}) = \int [l(\theta|X_{\text{hav}}, X_{\text{puu}})] P(X_{\text{puu}}|X_{\text{hav}}, \theta^{(t)}) dX_{\text{puu}} \tag{22}$$

ja

$$H(\theta|\theta^{(t)}) = \int [\ln p(X_{\text{puu}}|X_{\text{hav}}, \theta)] p(X_{\text{puu}}|X_{\text{hav}}, \theta^{(t)}) dX_{\text{puu}}. \tag{23}$$

Ajatellaan iteraatiota, jossa parametrin estimaatti on edellisen iteraatiokierroksen estimaatin funktio: $\theta^{(t+1)} = M(\theta^{(t)})$. Logaritmisen uskottavuuden muutos on

$$l(\theta^{(t+1)}|X_{\text{hav}}) - l(\theta^{(t)}|X_{\text{hav}}) = [Q(\theta|\theta^{(t+1)}) - Q(\theta|\theta^{(t)})] - [H(\theta|\theta^{(t+1)}) - H(\theta|\theta^{(t)})]. \tag{24}$$

Jensenin epäyhtälön avulla voidaan osoittaa, että H-funktioiden ero on aina negatiivinen. EM-algoritmi valitsee parametrin $\theta^{(t+1)}$ funktion $Q(\theta|\theta^{(t)})$ minimikohdasta, joten uskottavuus kasvaa jokaisella iteraation askeleella. Yleistetty EM-algoritmi valitsee uuden parametrin arvon yksinkertaisemmin ehdosta

$$Q(\theta^{(t+1)}|\theta^{(t)}) \leq Q(\theta^{(t)}|\theta^{(t)}). \tag{25}$$

[11, s. 134-135]

6.1 EM-algoritmi eksponentiaalisen perheen jakaumille

Tunnusluku (eng. statistic) T on funktio, jonka arvo $T(X)$ voidaan laskea havainnoista X tietämättä mallin parametrin θ arvoa. Tunnusluku T on tyhjentävä (eng. sufficient), jos sen havaittu arvo riittää määrittämään uskottavuuden $L(\theta|X)$. Tällöin mallin tiheysfunktio on muotoa

$$P(X|\theta) = c(X)h(T(X|\theta)), \tag{26}$$

missä funktio $c(X)$ on vakio θ:n suhteen. Todennäköisyys, että tunnusluku T saa arvon t, on

$$P(T = t|\theta) = \sum_{\forall X:T(X)=t} P(X|\theta) = h(t|\theta) \sum_{T(X)=t} c(X) = h(t|\theta)d(t), \tag{27}$$

missä $d(t)$ ei ole X:n funktio. Ehdollinen todennäköisyys

$$P(X|T(X) = t) = \frac{c(X)h(T(X|\theta))}{h(t|\theta)d(t)} = \frac{c(X)}{d(t)} \tag{28}$$

ei ole θ:n funktio. Tämä voidaan myös ottaa tyhjentävän tunnusluvunn määritelmäksi: tunnusluku T on tyhjentävä silloin ja vain silloin kun kaikki θ:a koskevat päätelmät voidaan tehdä T:n avulla.[10, s. 48]

Havaintoaineiston jakauma on säännöllisestä eksponentiaalisesta perheestä, jos se on muotoa

$$p(X|\theta) = b(X)\exp(S(X)\theta)/a(\theta), \tag{29}$$

missä $S(X)$ on tyhjentävien tunnuslukujen muodostama vektori, a ja b joitain funktioita. Hyvin monet tilastolliset mallit ovat eksponentiaalisesta perheestä, esimerkiksi normaalijakauma. EM-algoritmin E-askel on tunnuslukujen estimoimista kaavalla

$$s^{(t+1)} = E(s(X)|X_{\text{hav}}, \theta^{(t)}) \tag{30}$$

ja M-askel uuden parametrivektoriestimaatin θ^{t+1} laskemista uskottavuusyhtälöstä

$$E(s(X)|\theta) = s^{(t)}, \tag{31}$$

jossa $S(X)$ on korvattu $s^{(t)}$:llä: Tästä yhtälöstä voidaan usein ratkaista θ eksplisiittisesti, mutta jos ei, niin ainakin numeerinen ratkaisu on mahdollista. Molemmissa tapauksissa laskennalliset ongelmat jäävät E-askeleeseen.[11, s. 138-139] [5, update vol 1 s. 129]

7 Komponenttimallit

Parametriset tilastolliset mallit (esim. normaalijakauma) voivat sopia huonosti havaintoaineistoon, koska ne olettavat havaintoaineiston noudattavan tietyn muotoista tiheysfunktiota, joka voi poiketa huomattavasti todellisesta jakaumasta. Epäparametriset mallit sallivat yleisemmän tiheysfunktion

muodon, mutta ovat laskennallisesti ongelmallisia, koska mallin muuttujien lukumäärä kasvaa havaintoaineiston kasvaessa samalla nopeudella kuin havaintoaineisto itse.[1, s. 59-60]

Semiparametriset mallit sallivat parametrisia yleisemmän muodon tiheysfunktiolle. Kuitenkaan mallin muuttujien lukumäärä ei kasva havaintoaineiston mukana, vaan todellisen jakauman monimutkaisuuden mukaan. Toisaalta semiparametristen mallien estimoiminen on hidasta. Keskitytään nyt semiparametristen mallien joukossa tietynlaisiin malleihin, komponenttimalleihin (eng. mixture models):

$$p(x) = \sum_{k=1}^{K} p(x|k)P(k), \tag{32}$$

missä (priori)todennäköisyydet $P(k)$ toteuttavat ehdot $\sum_{k=1}^{K} P(k) = 1$ ja $0 \leq P(k) \leq 1 \ \forall k$ sekä komponenttien tiheysfunktiot ehdon

$$\int p(x|k)dx = 1 \ \forall k. \tag{33}$$

Jos havaintoaineistoon sovitetaan tällainen malli, jossa parametri K on riittävän suuri, niin mitä tahansa jatkuvaa jakaumaa voidaan approksimoida hyvin tarkasti. Ei ole tiedossa mikä k:s luokka on generoinut kunkin havainnon, mutta Bayesin kaavalla saadaan laskettua (posteriori)todennäköisyys sille

$$p(k|x) = \frac{p(x|k)P(k)}{p(x)}, \tag{34}$$

että k:s luokka generoi havainnon x. Komponenttimallin logaritminen uskottavuus riippumattomille havainnoille on

$$\ln p(x) = \ln \prod_{i=1}^{N} p(x_i) = \sum_{i=1}^{N} \ln p(x_i) = \sum_{i=1}^{N} \ln \sum_{k=1}^{K} p(x_i|k)P(k). \tag{35}$$

Uskottavuusfunktion maksimi sijaitsee joko parametrialueen reunalla tai parametrien suhteen laskettujen derivaattojen nollakohdissa. Mallin parametreja ovat (priori)todennäköisyydet $P(k)$ ja tiheysfunktioiden $p(x|k)$ parametrit. Normaalijakaumankin tapauksessa derivoiminen johtaa hankaliin yhtälöihin, joten on etsittävä muita menetelmiä funktion maksimin etsimiseen.[1, s. 60-63]

7.1 EM-algoritmi komponenttimalleille

Luvussa 7 esitetty komponenttimallin logaritminen uskottavuusfunktio voidaan kirjoittaa indikaattorimuuttujan z_i avulla. Indikaattorimuuttuja saa arvon k, jos k:s komponentti on generoinut havainnon x_i. Kiinnitetyllä i:n arvolla deltafunktio δ_{kz_i} saa arvon yksi vain yhdellä k:n arvolla ja muulloin arvon nolla. Niinpä summa koostuu vain yhdestä yhteenlaskettavasta, joten logaritmi voidaan ottaa ennen yhteenlaskua. Näin saadaan

$$l(\theta|X,Z) = \sum_{i=1}^{N}\sum_{k=1}^{K} \delta_{kz_i} \ln[p(x_i|k,\theta)P(k,\theta)]. \tag{36}$$

EM-algoritmin E-askel on yhtälön (36) odotusarvon laskemista:

$$\begin{aligned} Q(\theta|\theta^{(t)}) &= E[l(\theta|X,Z)|X,\theta^{(t)}] \\ &= \sum_{z_1=1}^{K}\cdots\sum_{z_N=1}^{K} l(\theta|X,Z)\prod_{i'=1}^{N} P(z_{i'}|x_{i'}) \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K}\Big(\sum_{z_1=1}^{K}\cdots\sum_{z_N=1}^{K}\Big)\delta_{kz_i}\prod_{i'=1}^{N} P(z_{i'}|x_{i'})\ln[p^u(x_i|k)P^u(k)] \\ &= \sum_{i=1}^{N}\sum_{k=1}^{K} P^v(k|x_i)\ln[p^u(x_i|k)P^u(k)] \end{aligned} \tag{37}$$

ja M-askel parametrin $\theta^{(t+1)}$ valitsemista siten, että Q-funktio minimoituu. [1, s. 65-72]

7.2 Sovellus: yhden vasta-aineen malli

Tarkastellessani diabetesvasta-aineiden jakaumia histogrammien avulla ajattelin, että tämä jakauma ei ole peräisin yhdestä muuttujasta. Tuli mieleeni, että taustalla on piilomuuttuja, joka selittää merkittävästi vasta-aineiden arvoja. Kyseessä ei ehkä kuitenkaan ole kahden normaalijakauman yhdistelmä. Voisi pikemminkin ajatella, että terveillä vasta-aineet noudattavat eksponenttijakaumaa

$$p^u(x_i|1) = \frac{1}{\theta}\, e^{-x_i/\theta}, \tag{38}$$

jonka odotusarvo on θ ja varianssi θ sekä sairastumassa olevilla normaalijakaumaa

$$p^u(x_i|2) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x_i-\mu)^2}{2\sigma^2}}, \tag{39}$$

jonka odotusarvo on μ ja varianssi σ^2.

Tällaisen komponenttimallin Q-funktio on

$$\begin{aligned} Q = -\sum_{i=1}^{N} \{ & P^v(1|x_i)[\ln P^u(1) + \ln p^u(x_i|1)] \\ & + P^v(2|x_i)[\ln P^u(2) + \ln p^u(x_i|2)]\} \\ = -\sum_{i=1}^{N} \{ & P^v(1|x_i)[\ln P^u(1) - \frac{x_i}{\theta^u} - \ln \theta^u] \\ & + P^v(2|x_i)[\ln P^u(2) - \ln \sigma - \frac{(x_i-\mu^u)^2}{2\sigma^{u2}}]\} + \text{vakio}. \end{aligned} \tag{40}$$

Piilomuuttuja saa siis arvoja yksi (terve) ja kaksi (sairastumassa). Näiden summa on $P(1) + P(2) = 1$. Koska funktion Q minimointia rajoittaa tämä ehto, minimoidaankin funktiota $Q + \lambda(P^u(1) + P^u(2) - 1)$. Tämä on nk. Lagrangen menetelmä. Jos funktio derivoidaan $P^u(k)$:n suhteen ja merkitään yhtä suureksi kuin nolla, niin saadaan $\sum_{i=1}^{N} \frac{P^v(k|x_i)}{P^u(k)} = \lambda$. Kerrotaan puolittain nimittäjällä $P^u(k)$ ja summataan sitten puolittain yli k:n. Näin saadaan $\lambda = N$.

Derivoidaan nyt minimoitava funktio vuorotellen eri parametrien suhteen ja etsitään derivaatan nollakohta. Tulokseksi saadaan seuraavat neljä yhtälöä:

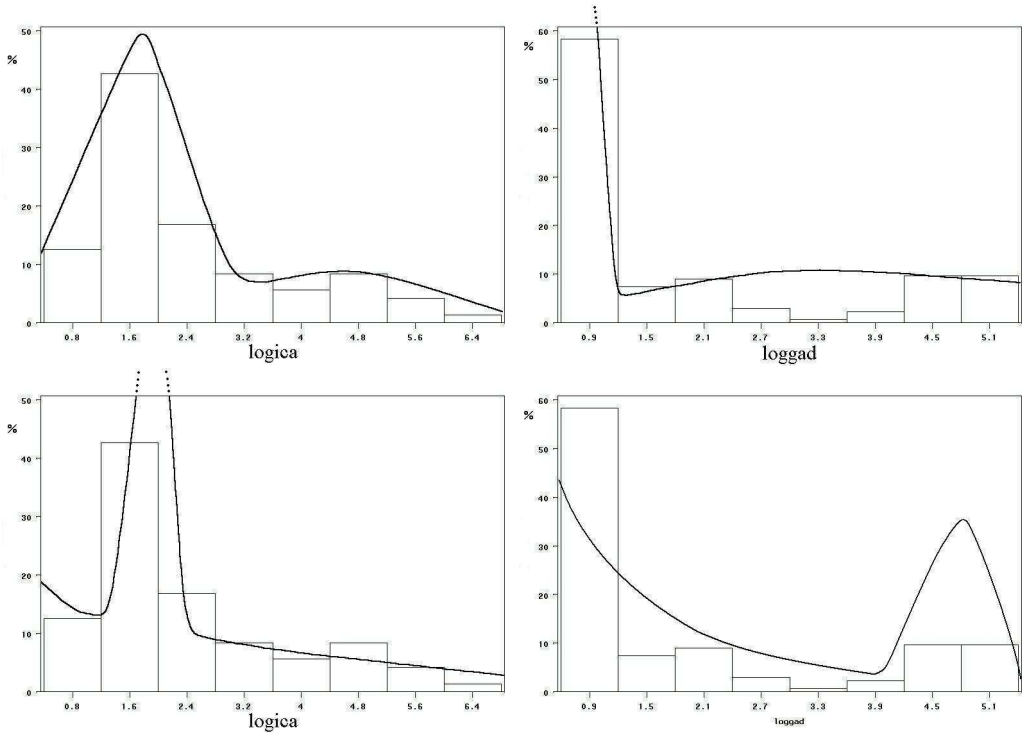
$$\begin{aligned} P^u(k) &= \frac{1}{N}\sum_{i=1}^{N} P^v(k|x_i) \\ \mu^u &= \frac{\sum_{i=1}^{N} P^v(2|x_i)x_i}{\sum_{i=1}^{N} P^v(2|x_i)} \\ (\sigma^u)^2 &= \frac{\sum_{i=1}^{N} P^v(2|x_i)(x_i-\mu^u)^2}{\sum_{i=1}^{N} P^v(2|x_i)} \\ \theta^u &= \frac{\sum_{i=1}^{N} P^v(1|x_i)x_i}{\sum_{i=1}^{N} P^v(1|x_i)}, \end{aligned} \tag{41}$$

yksi jokaiselle parametrille

Edellisillä kaavoilla lasketaan parametreille uudet estimaatit havaintoaineistosta ja todennäköisyyksistä $P^u(k|x_i)$. Tämä on EM-algoritmin M-askel. Kahden M-askeleen välissä on kuitenkin laskettava uudet estimaatit todennäköisyyksille

$$p^u(k|x_i) = \frac{p^u(x_i|k)P^u(k)}{p^u(x_i)} = \frac{p^u(x_i|k)P^u(k)}{p^u(x_i|1)P^u(1) + p^u(x_i|2)P^u(2)}. \qquad (42)$$

Tämä on puolestaan EM-algoritmin E-askel.



Kuva 3: Sokerirasituksessa käyneiden lasten logaritmimuunnettujen ica ja gad vasta-aineiden komponenttimallit. Ylemmissä kuvissa on käytetty kahden normaalijakauman yhdelmää ja alemmissa kuvissa exponenttijakauman ja normaalijakauman yhdelmää.

Kuvassa 3 on kahden eri logaritmimuunnetun vasta-aineen (logica ja loggad) histogrammit ja niihin sovitetut mallit. Malleina on käytetty sekä edellä johdettua normaalijakauman ja eksponenttijakauman yhdelmää että kahden normaalijakauman yhdelmää. Ensin mainittua mallia käyttäen sovitet-

taessa loggad-vasta-aineen havaintoihin, saadaan parametrien estimaateiksi $P(1) = 0,76$, $\mu = 4,77$, $\sigma^2 = 0,08$ ja $\theta = 1,29$. Näin ollen tilastollinen malli on

$$p(\text{loggad}) = 0,59\exp(-0,77\text{loggad}) + 0,33\exp(-6,25(\text{loggad} - 4,77)^2).$$

Logica-vasta-aineen kohdalla parametrien estimaatit ovat $P(1) = 0,54$, $\mu = 1,88$, $\sigma^2 = 0,04$ ja $\theta = 2,93$. Malli on niiden avulla laskettuna

$$p(\text{logica}) = 0,18\exp(-0,34\text{logica}) + 0,92\exp(-11,9(\text{logica} - 1,88)^2).$$

Kahden normaalijakauman mallia käyttäen parametrien estimaatit ovat loggad-muuttujalle $P(1) = 0,52$, $\mu_1 = 0,83$, $\sigma_1^2 = 0,018$, $\mu_2 = 3,29$ ja $\sigma_2^2 = 2,16$. Malli on silloin

$$p(\text{loggad}) = 1,54\exp(-27,8(\text{loggad}-0,83)^2)+0,13\exp(-0,23(\text{loggad}-3,29)^2).$$

Logica-muuttujalle parametrien estimaatit ovat puolestaan $P(1) = 0,74$, $\mu_1 = 1,80$, $\sigma_1^2 = 0,37$, $\mu_2 = 4,45$ ja $\sigma_2^2 = 1,02$, jolloin malli on

$$p(\text{logica}) = 0,49\exp(-1,35(\text{logica}-1,80)^2)+0,10\exp(-0,49(\text{logica}-4,55)^2).$$

Kuvien perusteella näyttää siltä, että logica-muuttujaan sopii paremmin kahden normaalijakauman yhdelmä kuin eksponenttijakauman ja normaalijakauman yhdelmä. loggad-muuttujaan ei sovi kahden normaalijakauman malli, koska malli ei pysty huomioimaan arvojen $2,4$ ja $4,2$ välillä olevia matalia frekvenssejä. Eksponentti- ja normaalijakauman yhdelmä pystyy mainitulla välillä parempaan, mutta etenkin suurilla arvoilla malli liioittelee frekvenssiä.

Kuvassa 3 olevat mallit sovitettiin aluksi käyttämällä alkuarvoja $P(1) = 0,5$ ja $P(2) = 0,5$. Huonojen tulosten takia niitä kuitenkin jouduttiin muuttamaan kahdessa tapauksessa neljästä. Tämä kertoo siitä, että EM-algoritmi ei ole globaali.

7.3 Globaalit EM-algoritmit

Edellä kappaleessa 7.1 esitetyssä EM-algoritmissa on kaksi heikkoa kohtaa:

1. EM-algoritmi on lokaalinen optimointimenetelmä eikä siis uskottavuusfunktion globaalin maksimin löytämisestä ole takeita. Alkuarvoista riippuu mihin lokaaleista maksimikohdista algoritmi päätyy.

2. Komponenttien paras lukumäärä ei yleensä ole etukäteen tiedossa.

Näistä syistä on kehitetty ahne EM-algoritmi. Se lisää komponenttien lukumäärää yksi kerrallaan ja valitsee uuden komponentin keskipisteen havaintojen joukosta. Algoritmi valitsee keskipisteeksi sen havainnon, johon liittyvä uskottavuus on suurin. Ahne EM-algoritmi on globaali menetelmä, koska se ei käytä vain yhtä alkuarvovektoria uuden komponentin valinnassa. Algoritmi kuitenkin rajoittuu tilanteeseen, jossa komponentit noudattavat normaalijakaumaa.

7.3.1 Ahne algoritmi

Ahneessa algoritmissa mallinnus aloitetaan yhdellä komponentilla ($K = 1$). Sen jälkeen niiden lukumäärää lisätään yhdellä iteratiivisesti, kunnes mallissa on komponentteja ennalta päätetty lukumäärä. Rajoitutaan tilanteeseen, jossa komponentit noudattavat normaalijakaumia

$$p(x|k) = \frac{1}{(2\pi)^{J/2}|\mathbf{S}_k|^{-1/2}} \exp[-\frac{1}{2}(x - m_k)^T \mathbf{S}_k^{-1}(x - m_k)].$$

1. Määrätään algoritmin alkuarvot. Lasketaan havaintoaineistosta odotusarvot $m = E(\mathbf{X})$ ja kovarianssit $\mathbf{S} = Cov(\mathbf{X})$. Lasketaan β siten, että se on puolet matriisin $\mathbf{S}$ suurimmasta singulaariarvosta. Lasketaan σ kaavasta

$$\sigma = \beta[\frac{4}{(J+2)N}]^{1/(J+4)},$$

missä J on havaintoaineiston ulottuvuuksien lukumäärä. Asetetaan $N \times N$- matriisin $\mathbf{G}$ arvoiksi luvut, jotka saadaan kaavasta

$$\mathbf{G}_{ij} = (2\pi\sigma^2)^{-J/2} \exp(-0.5 \left\| x_i - x_j \right\|^2 / \sigma^2).$$

Asetetaan komponenttien lukumääräksi $K = 1$.

2. Suoritetaan EM-algoritmi K-komponenttiselle mallille. Suppenemiskriteeri olkoon $|L_K^t/L_K^{t-1} - 1| < 10^{-6}$. Lasketaan siis parametrit iteratiivisesti seuraavilla kaavoilla

$$P(k|x_i) = \frac{P(k)p(x_i|k, m_k, \mathbf{S}_k)}{p_k(x_i)} \tag{43}$$

$$P(k) = \frac{1}{N}\sum_{i=1}^{N} P(k|x_i) \tag{44}$$

$$m_k = \frac{\sum_{i=1}^{N} P(k|x_i)x_i}{\sum_{i=1}^{n} P(k|x_i)} \tag{45}$$

$$\mathbf{S}_k = \frac{\sum_{i=1}^{N} P(k|x_i)(x_i - m_k)(x_i - m_k)^T}{\sum_{i=1}^{N} P(k|x_i)}, \tag{46}$$

kaikille komponenteille $k = 1, 2, \ldots, K$.

3. Jos komponenttien lukumäärä K saavuttaa ennalta määrätyn maksiminsa, niin pysäytä algoritmi. On myös mahdollista käyttää muita pysähtymiskriteereitä, esim. Akaiken informaatiokriteeriä.

4. Etsitään optimaalinen uusi komponentti valitsemalla kaikista havainnoista x_j se, joka maksimoi uskottavuusfunktion

$$L_{K+1} = \sum_{i=1}^{N} \ln p_{k+1}(x_i) = \sum_{i=1}^{N} \ln[(1-a)p_k(x_i) + a\phi(x_i;\theta)]$$

approksimaation

$$\hat{L}_{K+1} = \sum_{i=1}^{N} \ln \frac{p_K(x_i) + \phi(x_i|\theta)}{2} + \frac{1}{2}\frac{[\sum_{i=1}^{N} \delta(x_i|\theta)]^2}{\sum_{i=1}^{N} \delta^2(x_i|\theta)},$$

missä

$$\delta(x, \theta) = \frac{p_k(x) - \phi(x|\theta)}{p_k(x) - \phi(x|\theta)}$$

ja lukujen $\phi(x_i|x_j, \sigma^2 I)$ sijasta käytetään matriisin $\mathbf{G}$ arvoja $\mathbf{G}_{ij}$. Asetetaan kandidaateista x_j optimaalisimmaksi havaittu luvuksi m.

5. Asetetaan osittaisen EM-algoritmin alkuarvoiksi m, $\mathbf{S} = \sigma^2 I$ ja $\hat{a} = \frac{1}{2} - \frac{1}{2}\frac{\sum_{i=1}^{N} \delta(x_i|\theta)}{\sum_{i=1}^{N} \delta^2(x_i|\theta)}$

6. Sovelletaan osittaista EM-algoritmia:

$$P(k+1|x_i) = \frac{a\phi(x_i; m, \mathbf{S})}{(1-a)p_k(x_i) + a\phi(x_i; m, \mathbf{S})} \quad (47)$$

$$a = \frac{1}{N}\sum_{i=1}^{N} P(k+1|x_i) \quad (48)$$

$$m = \frac{\sum_{i=1}^{N} P(k+1|x_i)x_i}{\sum_{i=1}^{N} P(k+1|x_i)} \quad (49)$$

$$\mathbf{S} = \frac{\sum_{i=1}^{N} P(k+1|x_i)(x_i-m)(x_i-m)^T}{\sum_{i=1}^{N} P(k+1|x_i)} \quad (50)$$

   kunnes algoritmi suppenee kohdassa 5 esitetyn kriteerin mukaan.

7. Jos uskottavuus pienenee $L_{k+1} \leq L_k$, niin pysäytä algoritmi. Muussa tapauksessa siirry kohtaan 2.

[17]

7.3.2 Tehokkaampi ahne algoritmi

Edellä esitetyllä ahneella EM-algoritmilla on ainakin seuraavat heikkoudet:

1. On liian hidasta ottaa uuden komponentin keskipisteen kandidaateiksi kaikki havaintoaineiston hahmot.

2. Menetelmä käyttää samaa vakiokovarianssimatriisia, joten tehdään (kovarianssimatriisin derivaataltaan) liian pieniä komponentteja sellaisille alueille, joissa on paljon havaintoja. Kandidaattien joukossa ei ole suuria komponentteja, jotka antaisivat parempia tuloksia.

Onkin kehitetty ahneesta algoritmista tehokkaampi versio, joka etsii ratkaisua mainittuihin kahteen ongelmaan. Näin saavutetaan nopeus $O(NK^2)$ nopeuden $O(N^2)$ sijaan. [16]

Uuden komponentin sijaintia etsittäessä kohdassa (4) ei käydäkään läpi kaikkia havaintoja, vaan jaetaan kohdassa (2) laskettujen posterioritodennäköisyyksien avulla havaintoaineisto X erillisiin joukkoihin

$$A_i = \left\{ x \in X : i = \arg\max_k \{P(k|x)\} \right\},$$

$i = 1, \ldots, k$. Sen jälkeen valitaan jokaisesta joukosta A_i satunnaisesti kaksi havaintoa ja jaetaan jokaisen joukon A_i havainnot kahteen osaan sen mukaan kumpaa kahdesta havainnosta ne ovat lähempänä. Lasketaan uusista joukoista A_{i1} ja A_{i2} keskiarvot ja kovarianssit, joita käytetään komponenttikandidaatteina. Toistamalla joukkojen A_i puolittaminen saadaan lisää kandidaatteja. Kandidaatien etsimisen jälkeen tehostetaan myös osittaista EM-algoritmia (ks. ahneen algoritmin kohtaa 6) niin, että käytetään parametrien a, m ja S laskemisessa vain kandidaatteja kaikkien havaintojen sijaan. [16]

Tehokkaan ahneen algoritmin on osoitettu olevan tavallista EM-algoritmia ja ahnetta EM-algoritmia parempi keinotekoisessa havaintoaineistossa. Tehokas ahne EM-algoritmi löysi komponenttimalleja, joiden uskottavuus oli suurempi kuin muilla algoritmeilla löydettyjen mallien uskottavuus. Tehokas ahne EM-algoritmi toimi muihin verrattuna sitä paremmin mitä separoituneempia komponentit aineistoissa olivat ja mitä enemmän aineistossa oli komponentteja ja muuttujia.

Valitettavasti kirjallisuudessa ei ole esitetty algoritmia, joka vastaavalla tavalla parantaisi tavallista EM-algoritmia, mutta samalla sallisi käytettävän muuttujia, jotka eivät noudata normaalijakaumaa. Tehokas ahne EM-algoritmi ei myöskään salli puuttuvia havaintoja. Tavallinen EM-algoritmi sen sijaan voidaan muuntaa sellaiseksi, että se sallii puuttuvat havainnot [6].

7.4 Yhteistiheysfunktio

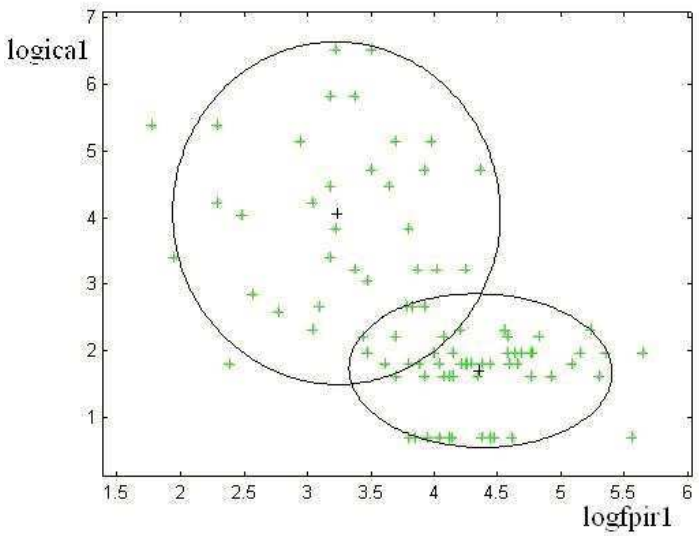
Aiemmin tässä luvussa approksimoitiin muuttujien yhteisjakaumaa komponenttimallilla. Kun mallin parametrit on estimoitu, voidaan yrittää ennustaa jotakin muuttujista muiden avulla [6]:

1. Estimoidaan koko havaintoaineiston (mukaan lukien tulostusmuuttujat) yhteisjakaumaa $p(X, Y)$ komponenttimallilla.

2. Lasketaan yhteisjakaumasta ehdollinen jakauma $p(Y|X) = \frac{p(X,Y)}{p(X)} = \frac{p(X,Y)}{\int_Y p(X,Y)}$ ja siitä ehdollinen odotusarvo $E(Y|X) = \int yp(Y|X)dy$.

7.5 Sovellus: mallinnetaan diabetesaineiston yhteistiheyttä

Sovitetaan kahden muuttujan (logfpir ja logica) aineistoon komponenttimalli, jonka komponentit ovat normaalijakaumia. Käytetään mallin estimoimiseen luvun 7.3.2 tehokasta ahnetta EM-algoritmia, joka on globaali optimointimenetelmä. Ahne algoritmi sovittaa aineistoon ensin yhden komponentin mallin ja lisää komponenttien lukumäärää iteratiivisesti yhdellä. Tässä aineistossa suurin uskottavuus on kaksikomponenttisella mallilla (ks. kuva 4), kun algoritmi pysäytetään komponenttien lukumäärän kasvaessa neljään.



Kuva 4: Komponenttimalli viimeisen mittauskerran varhaisen insuliinivasteen (fpir1) ja ica-vasta-aineen (ica1) yhteisjakaumalle. Molemmille muuttujille on tehty logaritmimuunnos. Plus-merkit ovat havaintoja. Pallot ovat komponenttien keskiarvoja. Jokaista komponenttia kohti on piirretty ellipsi, jonka akselit ovat kovarianssimatriisin ominaisvektorit ja säteet kaksi kertaa ominaisarvojen suuruiset.

Tähän aineistoon onnistuttiin siis sovittamaan komponenttimalli, joka mallintaa yhteistiheyttä paremmin kuin normaalijakauma.

Mainittu komponenttimalli olisi löydetty myös tavallisella EM-algoritmilla, jos alkuarvot olisi valittu K-means algoritmilla. K-means algoritmissa vali-

taan ensin K satunnaista pistettä luokkakeskiarvoiksi, jonka jälkeen kaikki havainnot luokitellaan luokkiin sen mukaan mitä luokkakeskiarvoa lähimpänä ne ovat. Luokittelun jälkeen luokkien sisällä lasketaan keskiarvot uudelleen. Jos uudet keskiarvot eroavat vanhoista, niin käytetään uusia. Luokittelua ja keskiarvojen laskemista jatketaan, kunnes keskiarvot eivät enää huomattavasti muutu.

Vertaillaan tehokasta ahnetta EM-algoritmia ja tavallista EM-algoritmia. Tehokkaassa ahneessa algoritmissa käytettävien kandidaattien lukumääräksi valitaan kaksi. Menetelmien löytöämien mallien uskottavuuksia verrataan aineistossa, jossa on kaksi muuttujaa (ks. kuva 2) ja 87 havaintoa. Tulosten mukaan tässä aineistossa ei löydetty mainittavia eroja algoritmien välille. Algoritmit ovat nopeudeltaan likimain yhtäsuuria: tavallisen EM-algoritmin suoritus kesti n. $0,18$s ja tehokkaan ahneen EM-algoritmin suoritus kesti n. $0,20$s.

komponenttien lukumäärä K	tavallinen EM K-means alkuarvoilla	tehokas ahne EM 2:lla kandidaatilla
1	-2,8	-2,9
2	-2,6	-2,7
3	-3,0	-2,8

Taulukko 2: EM-algoritmin ja tehokkaan EM-algoritmin löytämien komponenttimallien uskottavuuksien vertailu diabetesaineistossa. Muuttujina olivat muuttujat logica1 ja logfpir1. Laskut toistettiin 10 kertaa. Taulukkoon on kirjattu logaritmisten uskottavuuksien keskiarvot. Komponenttien lukumäärä oli 1-3. Tehokas ahne algoritmi suoritettiin vain kolmelle komponentille, koska se antaa samalla tulokset 1-2 komponentin malleille. Tehokkaassa ahneessa algoritmissa käytettiin kandidaattien lukumäärän kahta.

8 Ennustamisesta

Mallintamista käsittelevässä kirjallisuudessa käytetty terminologia tuntuu vaihtelevan tieteenaloittain, tai ainakin se vaihtelee kirjoittajasta riippuen.

Tietojenkäsittelijät kirjoittavat usein syöte- ja tulostusmuuttujista (eng. input/output), kun taas tilastotieteilijät kirjoittavat vastaavassa yhteydessä riippumattomista ja riippuvista muuttujista (eng. independent/dependent) tai selittävistä ja selitettävistä muuttujista. Kirjallisuudessa esiintyy myös termi tavoite- eli kohdemuuttuja (eng. target). Matemaatikko saattaa kirjoittaa mallin sovittamisesta (eng. model fitting), tietojenkäsittelijä mallin opettamisesta (eng. training) ja tilastotieteilijä mallin estimoimisesta (eng. estimation). Kuitenkin he kaikki tarkoittavat yhtä ja samaa asiaa: mallia sovitettaessa havaintoaineistoon mallia opetetaan havainnoilla ja mallin parametreja estimoidaan havaintojen avulla. Terminologia lienee vaihtelevaa, koska eri tieteenalat ovat perehtyneet erilaisiin malleihin, omien vahvuuksiensa mukaan. Matemaatikot tuntevat hyvin esim. differentiaaliyhtälöt. Tietojenkäsittelijät ovat parhaiten perillä esim. neuroverkoista ja tilastotieteilijöillä puolestaan on omat tilastolliset mallinsa, esim. lineaariset mallit.

Tähän mennessä tutkielmassa on käsitelty tilastotieteen parissa kehitettyjä malleja. Tutkielman loppuosassa käsitellään neuroverkkoja ja svm-menetelmiä, jotka ovat peräisin lähinnä tietojenkäsittelytieteestä.

Yleistettävyydellä tarkoitetaan mallin ennustamiskykyä sellaisen aineiston kohdalla, jota ei ole käytetty mallin opettamisen yhteydessä. Periaatteelliselta kannalta tutkimustulosten yleistettävyys on DIPP-projektissa vaikea, haastava ja mielenkiintoinen ongelma. Mielenkiintoista yleistäminen on, koska siihen DIPP-projekti nimenomaan pyrkii eli halutaan tuottaa uutta tieteellistä tietoa, joka pätee muihinkin ihmisiin kuin tutkittuihin. Vaikeaksi yleistämisen tekee se, ettei tutkittavia ole oikeastaan valittu satunnaisotannalla, vaan käytännön syistä on päätetty tutkia kaikki Turussa, Tampereella ja Oulussa syntyvät lapset. Näin ollen perinteisesti käytetyt tilastolliset mallit ovat lähtökohtaisesti ottaen tavallaan heikosti perusteltavissa oleva valinta mallinnusvälineiksi. Haastavaksi yleistämistä voisi sanoa siksi, että se vaikeudestaan huolimatta se on mahdollista. Sillä on selvää, että uudet tulokset ainakin jollain tarkkuudella pätevät myös muihin suomalaisiin kuin tutkittaviin. Vaikka tuntuukin, että yleistettävyys on lähinnä teoreettinen ongelma

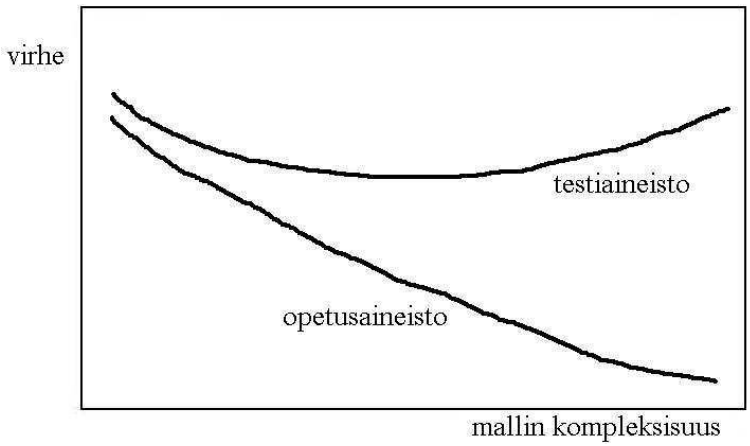
DIPP-projektissa, olisi jokaisen lääkäritutkijankin hyvä tiedostaa se ja pitää mielessä, ainakin sovellettaessa tilastollisia malleja, jotka olettavat, että havainnot ovat toisistaan riippumattomia ja peräisin satunnaisotoksesta.

Tilastollisten mallien parissa yleistettävyydestä pystytään oletusten nojalla tekemään teoreettisia ja laskennallisia päätelmiä, mutta neuroverkkojen tapauksessa yleistettävyyttä on tutkittava testiaineiston avulla. Eli osaa havaintoaineistosta ei käytetä neuroverkon opettamiseen, vaan sen avulla tutkitaan neuroverkon kykyä ennustaa testijoukossa.

Neuroverkoilla ja tilastollisilla malleilla onkin paljon yhteistä, kuten Bishop[1] painokkaasti huomauttaa. Varsinkin neuroverkkoihin tutustumista suunnittelevien olisi syytä paneutua hieman myös tilastotieteen perusteisiin.

Havaintoaineistoa voidaan mallintaa aina vain tarkemmalla mallilla ja näin opetusvirhe pienenee. Kuitenkin mallin yleistämiskykyä ei voida arvioida opetusvirheen avulla, vaan yleistämistä on arvioitava erillisellä aineistolla. Erillistä aineistoa kutsutaan testiaineistoksi ja siitä laskettua virhettä testivirheeksi. Testivirhekin kyllä aluksi pienenee, jos opetusaineistoon sovitettavaa mallia tarkennetaan. Jossain vaiheessa mallia tarkennettaessa testivirhe alkaa kuitenkin kasvaa, koska mallia *ylisovitetaan* (eng. over-fitting) opetusaineistoon. Tällöin (ks. kuva 5) malli oppii mukailemaan opetusaineistossa olevia kokonaisuuden kannalta epäoleellisia yksityiskohtia, ja testiaineiston kohdalla malli ennustaakin sitten huonosti, koska nämä yksityiskohdat puuttuvat testiaineistosta.[9, s. 193-194]

Havaintoaineisto on parasta jakaa satunnaisesti kolmeen osaan: opetusaineistoon, validointiaineistoon ja testiaineistoon, jos haviantoja vain on saatavilla tarpeeksi. Opetusaineistoa käytetään mallin sovittamiseen. Validointiaineistoa käytetään ennustevirheen laskemiseen mallien vertailemisessa ja sopivimman mallin valinnassa. Testiaineistoa käytetään testivirheen eli yleistämiskyvyn arvioimiseen. Mikäli havaintoja on niukalti, ei tällaiseen jakoon ehkä ole varaa. Tällöin yleisesti käytetty lähestymistapa on ristiinvalidointi.[9, s. 196, 214]



Kuva 5: Sovitettaessa opetusaineistoon mallia opetusaineistosta laskettu virhe pienenee. Sitä vastoin testiaineistosta laskettava ennustevirhe alkaa kasvaa siinä vaiheessa, kun malli alkaa olla opetusaineistoon ylisopiva.

8.1 Ristiinvalidointi

Ristiinvalidointia voidaan käyttää sekä mallin valinnassa että ennustamisvirheen arvioinnissa.

Ennustamisvirheen arvioinnissa havaintoaineisto jaetaan satunnaisesti O:hon (likimain yhtäsuureen) osaan. Kokemusten perusteella on huomattu, että $O = 5, \ldots, 10$ on usein hyvä valinta. Sen jälkeen jätetään vuorotellen yksi osista testiaineistoksi, sovitetaan malli muiden osien muodostamaan opetusaineistoon ja lasketaan testivirhe testiaineiston avulla. Olkoon $\kappa : 1, \ldots, N \mapsto 1, \ldots, O$ indeksimuuttuja, joka kertoo mihin osaan kukin havainnoista on jaettu. Merkitään sovitettuja malleja $f^{-o}(x)$, jonka estimoimiseen siis ei ole käytetty osaa o. Tällöin ennustevirheen ristiinvalidoitu estimaatti on

$$\mathrm{RV} = \frac{1}{N} \sum_{i=1}^{N} v(y_i, f^{-\kappa(i)}(x_i)),$$

missä v on virhefunktio.[9, s. 214-215]

Mallin valinnassa *opetus*aineisto jaetaan satunnaisesti O:hon osaan. Sen jälkeen jätetään vuorotellen yksi osista validointiaineistoksi, sovitetaan malli muiden osien muodostamaan opetusaineistoon ja lasketaan ennustevirhe

validointiaineiston avulla. Merkitään sovitettuja malleja $f^{-o}(x, \alpha)$, missä parametri α säätelee mallin kompleksisuutta. Tällöin validointiaineistosta lasketun ennustevirheen ristiinvalidoitu estimaatti on

$$\mathrm{RV}(\alpha) = \frac{1}{N_{vt}} \sum_{i=1}^{N_{vt}} v(y_i, f^{-\kappa(i)}(x_i, \alpha)),$$

missä v on virhefunktio ja N_{vt} on validointi ja opetusaineistojen havaintojen lukumäärä yhteensä. Jos vertailtavat mallit eivät ole samaa tyyppiä, niin mallin valintaan ei käytetä parametria α. Mutta jos mallit ovat samaa tyyppiä, niin $\mathrm{RV}(\alpha)$ antaa validointivirhekäyrän estimaatin. Valitaan tällöin malliksi $f(x, \hat{\alpha})$, missä $\hat{\alpha}$ on käyrän $\mathrm{RV}(\alpha)$ minimikohta.[9, s. 214]

Ristiinvalidointia voidaan käyttää yhtäaikaa sisäkkäisesti sekä mallin valintaan että ennustamisvirheen arviointiin. Tällöin ennustevirhe lasketaan O kertaa ja validointivirhe O^2 kertaa, jos oletetaan, että molemmissa ristiivalidoinneissa osien lukumäärä on O.

8.2 Herkkyys ja tarkkuus

Herkkyys kertoo kuinka suuren osuuden oikeasti sairastuneista malli osasi ennustaa sairaiksi. Tarkkuus kertoo kuinka suuren osuuden oikeasti terveistä malli osasi ennustaa terveiksi. Herkkyyttä kutsutaan myös sensitiivisyydeksi ja tarkkuutta spesifisyydeksi. Nelikenttä (ks. taulukko 3) havainnollistaa tilannetta. Tässä tutkielmassa käytetään ristiinvalidoitua herkkyyttä ja tark-

	mallin mukaan sairas	mallin mukaan terve
oikeasti sairas	a	b
oikeasti terve	c	d

Taulukko 3: Herkkyys on $\frac{a}{a+b}$. Tarkkuus on $\frac{d}{c+d}$.

kuutta ennustamiskyvyn arvioimiseen.

Sairastuneilla sairas=1 ja terveillä sairas=0. Jos malli antaa ennusteen, joka on yli 0,5, niin malli ennustaa lapsen sairastuneen. Jos ennuste on alle 0,5, niin malli ennustaa lapsen pysyneen terveenä.

9 Logistinen regressio

Perinteisistä tilastollisista menetelmistä kaksiluokkaisen muuttujan ennustamiseen soveltunee parhaiten logistinen regressio. Oletetaan, että selitettävä muuttuja Y saa arvon yksi, jos henkilö on sairastunut, ja muulloin arvon nolla. Logistisessa regressiossa sairastumisen todennäköisyyttä pyritään selittämään useilla muilla muuttujilla, jotka voivat olla sekä jatkuvia että kategorisia. Ongelmana on se, että mallissa oletetaan selittäjien olevan toisistaan riippumattomia. Yleensähän näin ei ole. Jos selittäjät ovat toisistaan riippuvia, niin havaitut merkitsevyystasot[3] ovat todellisuudessa pienempiä kuin mallista lasketuina. Näin muuttujien merkitystä saatetaan aliarvioida. Havaittujen merkitsevyystasojen perusteellahan on tapana tehdä tilastollisia päätelmiä eri muuttujien välisistä yhteyksistä, eli onko yhteyttä vai ei. Nyt tämä ei ole varsinainen kiinnostuksen kohde, koska pyritään löytämään mahdollisimman hyvä malli ennustamista ajatellen. Voimme siksi käyttää korreloivia selittäjiä samassa mallissa.

Logistinen regressiomalli on muotoa

$$\text{logit}(P(Y=1)) = \log\left(\frac{P(Y=1)}{P(Y=0)}\right) = \alpha + X^T\beta, \tag{51}$$

missä α on vakioparametri, β on selittäjiin liittyvä parametrivektori. Malli estimoidaan iteratiivisella painotetulla pienimmän neliösumman menetelmällä (eng. iterative reweighted least squares).

Edellä kuvailtu logistinen regressiomalli voidaan tehdä myös askeltavalla menetelmällä. Takenevasti askeltavassa menetelmässä kaikki muuttujat laitetaan aluksi malliin, ja iteratiivisesti poistetaan aina se, jonka poistaminen vähiten huonontaa mallia, jos malli ei huonone tilastollisesti merkitsevästi. Jäljelle jäävät ne muuttujat, joiden poistaminen mallista huonontaisi mallia. Taaksepäin askeltavalla mallilla voidaan poistaa ylimääräiset selittäjät, mikä yksinkertaistaa mallia.

Logistinen regressiomallissa siis yritetään mallintaa vedonlyöntisuhteen (eng. odds, suom. myös mahdollisuus, suosioluku, vedonlyöntikerroin) loga-

[3] arkikielessä puhutaan usein p-arvosta

ritmia. Kun halutaan arvioida jonkin selittäjän vaikutusta vedonlyöntisuhteeseen, lasketaan ristitulosuhde (eng. odds ratio, suom. myös vedonlyöntisuhteiden osamäärä). Kaksiluokkaisen selittäjän tilanteessa ristitulosuhde lasketaan nimensä mukaisesti selitettävän muuttujan ja selittävän muuttujan välisestä nelikentästä. Esimerkiksi, jos sairastumista diabetekseen selitetään sukupuolella, niin nelikenttä on seuraavan näköinen.

	poika (=1)	tyttö (=0)
sairastunut (=1)	a	b
terve (=0)	c	d

Nelikentästä saadaan ristitulosuhde laskemalla poikien ja tyttöjen vedonlyöntisuhteiden osamäärä $\frac{a}{b}/\frac{c}{d} = \frac{ad}{bc}$. Myös jatkuvan selittäjän tilanteessa ristitulosuhde saadaan kahden vedonlyöntisuhteiden osamäärästä. Osoittajan vedonlyöntisuhde on laskettu tilanteessa, jossa ko. selittäjän arvo on keskimäärin yhden mittayksikön verran suurentunut ja muiden selittäjien arvot ovat samat verrattuna nimittäjän vedonlyöntisuhteen laskemisessa käytettyihin arvoihin.

9.1 Sovellus: ennustetaan sairastumista logistisella regressiolla

Sovitetaan diabetesaineistoon on logistinen regressiomalli, jossa selittäjinä on kahden viimeisen käynnin sokerirasitustulokset ja viimeisen käynnin vastaainetulokset. Selittäjien riippumattomuusoletus ei täyty ja mallissa saattaa olla selittäjiä, jotka eivät paranna mallia, joten käytetään taaksepäin askeltavaa mallia. Lasketut tulokset taulukossa 4 kertovat, että kun kaikki selittäjät ovat mukana mallissa, vain fpir1 selittää mallia tilastollisesti merkitsevästi. On kuitenkin mahdollista, että lopulliseen malliin jää enemmän kuin yksi tilastollisesti merkitsevä selittäjä, koska havaitut merkitsevyystasot muuttuvat jokaisella askeleella. Askeltavasti selittäjiä poistettaessa malliin jäi kuitenkin tällä kertaa vain vakiotermi ja fpir1, kuten taulukossa 5 on esitetty.

Muuttujaan fpir1 liittyvä ristitulosuhde on 0,856. Tämä tulkitaan niin, että sairastumisen vedonlyöntisuhde on pienempi lapsella, jolla on suurempi fpir1-tulos, kuin muilla. Tarkemmin sanottuna, kun fpir1 arvo kasvaa yhdellä mittayksiköllä, sairastumisen vedonlyöntisuhde pienenee 0,856-kertaiseksi.

selittäjä	parametrin estimaatti	keskivirhe	h.m.t.	ristitulosuhde	95%:n luottamusväli
vakio	7,652	2,871	0,008		
fpir1	$-0,190$	0,085	0,024	0,83	(0,70; 0,98)
fpir2	0,042	0,063	0,507	0,96	(0,85; 1,09)
ica1	$-0,004$	0,005	0,452	1,00	(0,99; 1,01)
ia2_1	0,0001	0,010	0,925	1,00	(0,98; 1,02)
gad1	$-0,022$	0,014	0,110	0,98	(0,95; 1,01)

Taulukko 4: Logistinen regressiomalli diabeteksen ennustamiseksi tutkimuslasten kahden viimeisen käynnin sokerirasitustuloksilla ja viimeisen käynnin vasta-ainetuloksilla. Taulukossa on muuttujan nimi, muuttujaan liittyvän parametrin estimaatti, keskivirhe ja havaittu merkitsevyystaso (h.m.t.) sekä ristitulosuhde ja sen 95%:n luottamusväli.

selittäjä	parametrin estimaatti	keskivirhe	h.m.t.	ristitulosuhde	95%:n luottamusväli
vakio	3,658	1,264	0,004		
fpir1	$-0,149$	0,044	0,001	0,862	(0,791; 0,939)

Taulukko 5: Taaksepäin askeltava logistinen regressiomalli. Sarakkeiden merkitykset ovat samat kuin taulukossa 4

Edellä löydetty yhden selittäjän logistinen regressiomalli ei vielä kerro kuinka hyvin sillä voidaan ennustaa. Se kertoo vain, että viidestä selittäjästä yksi oli tilastollisesti merkitsevä. Ennustamistarkoitusta varten havaintoaineisto olisi jaettava opetusaineistoon ja testiaineistoon ja tarkasteltava opetusaineistolla opetetun mallin ennustamiskykyä testiaineistossa. Aineiston jakoa ei ole tapana tehdä tilastollisissa analyyseissä, koska tavoitteena ei ole

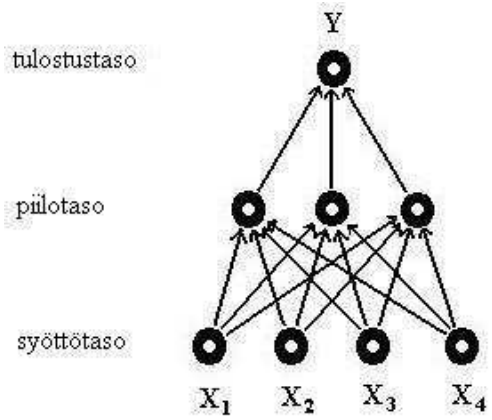
ennustaminen vaan yhteyksien löytäminen tutkimusaineistosta. Toteutetaan jako ristiivalidoimalla (ks. luku 8.1). Jaetaan ristiinvalidointia varten havaintoaineisto satunnaisesti viiteen likimain yhtäsuureen osaan. Olkoon jokainen viidestä osasta vuorotellen opetusaineisto, ja loput havainnot muodostakoot testiaineiston. Opetusaineistoon sovitetaan takeneva logistinen regressiomalli ja lasketaan löydetyn mallin herkkyys ja tarkkuus testiaineistossa. Takenevan logistisen regressiomallin herkkyys on 77% ja tarkkuus on 98%. Todellisuudessa sairastuneista malli olisi siis löytänyt 77 prosenttia. Terveinä pysyneistä malli olisi luokitellut 2 prosenttia sairaiksi. Aineistossa oli 14 sairastuvaa ja 76 terveenä pysyvää lasta. Malli olisi löytänyt noin 11 ($0,77 \times 14 = 10,8$) oikeasti sairastuvaa ja ehkä yhden ($0,02 \times 76 = 1,52$), joka saapuisi seuraavalle tutkimuskäynnille terveenä, vaikka hänen olisi ennustettu sairastuvan. Herkkyys jää tarkkuutta huonommaksi juuri sen takia, että monet sairastuvat vasta myöhemmin.

10 Neuroverkot

Luvussa 7 yritettiin mallintaa koko aineiston yhteistiheysjakaumaa ja ennustaa sen avulla jonkin muuttujan arvoja muiden muuttujien avulla. Luvussa 9 taas tehtiin oletuksia ennustamiseen käytettävän funktion muodosta eli käytettiin yhtä perinteisistä parametrisista tilastollisista malleista.

Neuroverkko on funktio, joka liittää ennustamisen perusteena oleviin syötemuuttujiin ennusteen. Jokaiselle tulevalle havainnolle voidaan siis neuroverkon avulla laskea ennuste. Neuroverkon parametrit eli painot estimoidaan jo havaittujen tapahtumien eli havaintoaineiston avulla, jossa siis myös tulostusmuuttujan arvo tunnetaan. Estimointia sanotaan tavallisesti neuroverkon opettamiseksi. Opettamisen jälkeen neuroverkkoa voidaan käyttää ennustamiseen. Kun saadaan uusi havainto, josta on mitattu vain syötemuuttujat, neuroverkko antaa ennustemuuttujien arvot.[1]

Neuroverkot eroavat muista funktioista rakenteellisesti. Tässä tutkielmassa rajoitutaan eteenpäin syöttäviin neuroverkkoihin, joilla on syöttötaso, pii-



Kuva 6: Eteenpäin syöttävä yhden piilotason neuroverkko, jossa pallot kuvaavat solmuja ja nuolet painoja.

lotaso ja yksisolmuinen tulostustaso (ks. kuva 6). Jokaisen solmun arvo saadaan edellisen tason muuttujien lineaarikombinaatiosta aktivointifunktiolla

$$\mathrm{af}(\sum(w_j \times \mathrm{solmu}_j). \tag{52}$$

Aktivointifunktio saa aikaan neuroverkon epälineaarisuuden. Useimmiten piilotason aktivointifunktioksi valitaan logistinen funktio eli sigmoidi-funktio

$$\mathrm{logistic}(x) = (1 - \exp(-x))^{-1} \tag{53}$$

tai hyperbolinen tangentti

$$\tanh(x) = (\exp(x) - \exp(-x))/(\exp(x) + \exp(-x)). \tag{54}$$

Edellinen saa arvoikseen lukuja väliltä $[0, 1]$ ja jälkimmäinen väliltä $[-1, 1]$. Kokemusten mukaan neuroverkon opettaminen on tehokkaampaa, jos piilotason aktivointifunktioksi valitaan hyperbolinen tangentti. Jos ennustettava muuttuja on jatkuva, niin tulostustason aktivointifunktioksi valitaan yleensä lineaarinen funktio $f(x) = x$. Jos taas tulostusmuuttuja on kaksiluokkainen, niin se koodataan luvuiksi 0 ja 1. Tällöin aktivointifuktioksi sopii hyvin logistinen funktio.[1]

10.1 Opettaminen

Neuroverkkoja voidaan käyttää niin, että niitä opetetaan aina heti, kun uusi havainto saadaan. Havaintoja ei tallenneta, vaan edellisen tiedot katoavat ennen seuraavaa havaintoa (eng. on-line training). Tässä keskitytään kuitenkin tilanteeseen, jossa opetukseen on käytettävissä kaikki havainnot yhtäaikaa (eng. off-line training).

Neuroverkkoa voidaan opettaa joko niin, että käytetään yhtä havaintoa kerrallaan (eng. incremental/sequential training) painojen korjaamiseen tai niin, että käytetään kaikkia havaintoja painojen korjaamiseen (eng. batch training).

Neuroverkon parametrien optimoimiseen opetusaineiston avulla on kehitetty paljon erilaisia menetelmiä. Kuuluisin niistä on nk. takaisinjäljitys (eng. backpropagation). Siinä painoja muutetaan tulostustasolta alkaen ja edeten syöttötasolle asti. Muita optimointimenetelmiä on paljon, niistä keskeisimpiä on esitelty kirjassa [1, s. 253-290].

Tavallisesti neuroverkot eivät hyväksy puuttuvia havaintoja, koska tällöin funktion arvoja ei pystytä laskemaan. Seuraavassa kappaleessa esitellään menetelmä, jolla tämä ongelma voidaan kiertää.

10.2 Ennustaminen puuttuvilla havainnoilla

Neuroverkkomenetelmät on tavallisesti laadittu aineistoille, joissa ei ole puuttuvia tietoja. Tässä luvussa esitellään algoritmi, joka toimii myös epätäydellisen aineiston tapauksessa. Algoritmi soveltuu käytettäväksi mielivaltaisten eteenpäin syöttävien neuroverkkojen kanssa. Menetelmässä sovelletaan *Parzen-ikkunoita*, joita tavallisesti käytetään tiheysfunktioiden approksimoimiseen. Parzen-ikkuna on normaalijakauma, jonka keskipisteessä on havainto ja keskihajonta on ennalta valittu vakio. Tiheysfunktiota on voitu approksimoida mielivaltaisessa pisteessä Parzen-ikkunoiden keskiarvolla. [14]

Oletetaan, että neuroverkko $\mathrm{nv}(X)$ on opetettu ennustamaan Y:n odotusarvoa $E(Y|X)$. Jos syötettävät muuttujat X voivat saada puuttuvia ar-

voja, niin niitä ei voida suoraan syöttää neuroverkolle. Tällöin voidaan kuitenkin ennustaa seuraavalla kaavalla:

$$\hat{Y} = E(Y|X_{\text{hav}}) \approx \frac{\sum_{n=1}^{N} \alpha_n \text{g}(X_{\text{hav}}|x_{\text{hav,n}}, \sigma)}{\sum_{n=1}^{N} \text{g}(X_{\text{hav}}|x_{\text{hav,n}}, \sigma)}, \tag{55}$$

missä $\alpha_n = \text{nv}(X_{\text{hav}}, X_{\text{puu,n}})$ on ennuste, joka saadaan, kun tuntemattomien syöttöarvojen tilalle sijoitetaan vastaavat opetusaineiston komponentit, ja missä $\text{g}(X_{\text{hav}}|x_{\text{hav,n}}, \sigma)$ on moniulotteinen normaalijakauma, jonka keskiarvona on syöttövektorin tunnettuja komponentteja vastaavat havaintoaineiston komponentit ja keskihajontana σ, ja joka on projisoitu syöttövektorin tunnetuille ulottuvuuksille jättämällä tuntemattomat pois normaalijakauman tiheysfunktiosta. [14]

10.3 Opettaminen puuttuvilla havainnoilla

Takaisinjäljitys menettelyä voidaan muuttaa puuttuvia arvoja sisältäville havainnoille sopivaksi, kun approksimoidaan painojen muutosta

$$\Delta w_j \propto (y_n - \text{nv}_w(x_n)) \frac{\partial \text{nv}_w(x_n)}{\partial w_j} \tag{56}$$

painotetuilla keskiarvoilla (samaan tapaan kuin ennustettaessakin tehtiin):

$$\Delta w_j \propto \frac{\sum_l \alpha_l \, \text{g}(y_n|\text{nv}_w(x_{\text{hav,n}}, x_{\text{puu,l}}), \sigma_y) \, \text{g}(x_{\text{hav,n}}|x_{\text{hav,l}}, \sigma)}{\sum_l \text{g}(y_n|\text{nv}_w(x_{\text{hav,n}}, x_{\text{puu,l}}), \sigma_y) \, \text{g}(x_{\text{hav,n}}|x_{\text{hav,l}}, \sigma))}, \tag{57}$$

missä

$$\alpha_l = (y_n - \text{nv}_w(x_{\text{hav,n}}, x_{\text{puu,l}})) \frac{\partial \text{nv}_w(x_{\text{hav,n}}, x_{\text{puu,l}})}{\partial w_j} \tag{58}$$

ja l käy läpi kaikki opetusaineiston havainnot. [14]

10.4 Sovellus: ennustetaan sairastumista neuroverkolla

Ennustetaan sokerirasituksessa käyneiden lasten sairastumista kahden viimeisen sokerirasitustutkimuksen tuloksella (logfpir1 ja logfpir2) ja viimeisen tutkimuskäynnin verikokeen vasta-ainepitoisuuksilla (logica1, logia2_1, loggad1). Kaikille muuttujille on tehty logaritmimuunnos ja standardointi vähentämällä keskiarvo ja jakamalla keskihajonnalla. Aineisto on sama, jota

käytettiin luvussa 9.1 logistisen regressiomallin kanssa. Osa syötemuuttujien arvoista puuttuu (ks. taulukko 6). Logistinen regressiomalli käytti laskuissa vain täydellisiä havaintoja. Neuroverkko kuitenkin sallii puuttuvat havainnot. Neuroverkkona on kuvan 6 kaltainen neuroverkko, jossa on yksi syöttötaso, yksi piilotaso ja yksi tulossolmu. Aktivointifunktiona piilotasolla on hyperbolinen tangentti ja tulostustasolla sigmoidifunktio. Neuroverkon painot opetetaan takaisinjäljittämällä virhefunktion derivaatan suuntaan gradienttimenetelmällä (eng. gradient descent). Virhefunktiona käytetään ristikkäisentropiaa (eng. cross entropy). Piilosolmujen lukumäärän valinnassa käytetään ristiinvalidoitua ristikkäisentropian estimaattia. Myös herkkyys ja tarkkuus lasketaan ristiinvalidoimalla, joten ristiinvalidointi tehdään sisäkkäisesti kahteen kertaan (ks. luku 8.1).

muuttuja	puuttuvien lkm
logfpir1	0
logfpir2	0
logica1	1
logia2_1	2
loggad1	16

Taulukko 6: Puuttuvien arvojen lukumäärät muuttujittain. Tutkittuja henkilöitä oli yhteensä 90.

Neuroverkolla on parametreja, joiden arvot on määrättävä ennen opettamista. Parametreja ovat piilosolmujen lukumäärän lisäksi: normaalijakaumien keskihajonnat, painojen opetettamisessa otettavien askelten pituus ja opetusaineiston läpikäymisten lukumäärä. Viimeinen on valittu kokeilemalla niin, että algoritmi suppenee, muttei tee juurikaan ylimääräistä työtä. Painojen muutosten suuruus pienenee kääntäen tehtyjen iteraatioiden lukumäärään nähden. Normaalijakaumien keskihajonnat on valittu sopiviksi arvaamalla. Lisäksi huomasin, että parametrien arvoiksi sopivat samat luvut kuin mitä algoritmin keksijät [14] ovat omassa sovelluksessaan käyttäneet, vaikka

aineistot ovatkin erilaiset. Tämä johtunee siitä, että selittäjät on logaritmimuunnettu ja standardoitu.

Piilosolmujen lukumäärä määrää mallin monimutkaisuuden asteen (vastaavaan tapaan kuin polynomin aste määrää polynomin monimutkaisuutta). Monimutkaisempi malli sopii opetusaineistoon aina paremmin kuin yksinkertaisempi malli. Mallia ei kuitenkaan voida monimutkaistaa loputtomasti yleistämiskyvyn kärsimättä. Tästä syystä on käytetty ristiinvalidointia, jolloin voidaan arvioida yleistämiskykyä keskimääräisen validointivirheen avulla. Piilotason solmujen lukumääräksi on valittu niin, että validointivirhe minimoituu.

Parametrien määräämisen jälkeen on siis malleista valittu sopivin. Sen jälkeen voidaan laskea testiaineistolla herkkyys ja tarkkuus, kun ajatellaan, että malli ennustaa sairastumista, jos ja vain jos output-muuttuja on yli 0,5. Tämä toistetaan jokaisella ristiinvalidoinnin aineistolla, jolloin saadaan herkkyyden ja tarkkuuden ristiinvalidoidut estimaatit.

DIPP-projektissa on tutkittu neljää vasta-ainetta, joista iaa:n määritysmenetelmä on muuttunut kerran. Eri menetelmillä kerätyt iaa-tiedot on tallennettu eri muuttujiin (iaau ja iaav). Tässä tutkielmassa käytettävässä osaaineistossa on iaa:n määrityksessä käytetty aina vain toista menetelmistä. Uutta ja vanhaa menetelmää ei ole yhtäaikaa käytetty minkään verinäytteen kohdalla. Toisen muuttujan tieto puuttuu aina. Näin ollen molempia muuttujia ei voida ottaa neuroverkkomallinnukseen mukaan, koska puuttuvia tietoja paikataan vain täydellisillä havainnoilla. Jotta aineisto olisi sama kuin logistisen mallin yhteydessä käytetty, on molemmat iaa-muuttujat jätetty pois.

Esitetyn menettelyn tuloksena saatiin herkkyydeksi 75% ja tarkkuudeksi 95%. Todellisuudessa sairastuneista malli olisi siis löytänyt 75 prosenttia. Terveinä pysyneistä malli olisi luokitellut 5 prosenttia sairaiksi. Aineistossa oli 14 sairastuvaa ja 76 terveenä pysyvää lasta. Malli olisi löytänyt sairastuvista noin 11 $(0,75 \times 14 = 10,5)$ ja vääriä sairastuvia se olisi löytänyt ehkä noin neljä $(0,05 \times 76 = 3,8)$. Neuroverkon ennustamiskyky osoittautui siis suunnilleen samantasoiseksi kuin logistisen regressiomallin ennustamiskyky.

11 SVM-menetelmät

Perinteisesti tilastotieteessä on tapana yrittää havaintoaineiston perusteella estimoida ennalta päätetyn muotoisen funktion parametreja. Kuitenkaan etukäteen ei voida tietää minkä muotoinen malli havaintoihin sopii. Funktion muodon tulisikin määräytyä oppimisprosessissa havaintoaineiston perusteella, aivan kuten parametrien arvotkin määräytyvät. Tähän ongelmaan on klassisessa tilastotieteessä pyritty vastaamaan kehittämällä epäparametrisia menetelmiä, mutta pääasiassa tilastotieteellisten menetelmien käyttäjän on turvauduttava parametrisiin menetelmiin, joissa mallin muoto on etukäteen päätetty. [15, s. 10]

Keskeisessä roolissa on ollut myös tulosten asymptoottinen ominaisuus: tulokset pätevät periaatteessa vain, kun havaintojen lukumäärä lähestyy ääretöntä. Myös epäparametriset menetelmät nojaavat tähän oletukseen. Näinhän ei kuitenkaan koskaan käytännössä ole, vaan havaintojen lukumäärä on *äärellinen*. [15, s. 27]

Ratkaistaessa oppimisongelmaa ei kannata välivaiheena ratkaista lopullista ongelmaa vaikeampaa ongelmaa. Jos ensin estimoidaan havintoaineistoon sopiva funktio ja vasta siitä lasketaan ennusteet tietyissä (testiaineiston) pisteissä, niin oikeastaan yritetään oppia vaikeimman kautta. Helpompaa olisi estimoida funktiota vain annetuissa pisteissä. [15, s. 30]

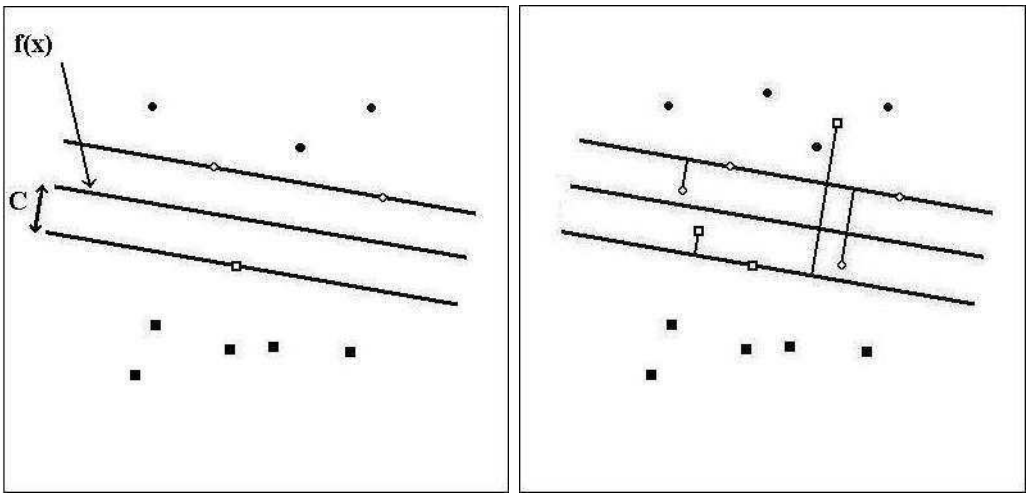
Tiheysfunktion $p(X, Y)$ avulla ennustaminen $E(Y|X)$ ei ole järkevää, sillä siinä estimoidaan koko funktio. Lisäksi monimutkaisten tiheysfunktioiden estimoimiseen kehitetyt menetelmät, kuten komponenttimallit tai Parzen ikkunat, pystyvät approksimoimaan tiheysfunktiota vain, jos havaintojen lukumäärä on riittävän suuri eli lähestyy ääretöntä. [15, s. 30]

Näihin ongelmiin on haettu vastausta vuosikymmenien ajan. Teoreettista tietoa on karttunut aina 1960-luvulta asti, kunnes 1990-luvulla keksittiin SVM-menetelmät (eng. support vector machines), joiden avulla edellä mainitut ongelmat on ehkä mahdollista voittaa. Nämä menetelmät ovat käytössä nykyaikaisessa suurten tietomassojen louhinnassa (eng. data mining). Ensin esitellään SV-luokittelijat, jotka luokittelevat aineiston optimaalisel-

la hypertasolla. Tämä luokittelu yleistetään SVM-menetelmäksi korvaamalla SV-luokittelijassa käytetty sisätulo ydinfunktiolla, jolloin luokittelu muuttuu lineaarisesta epälineaariseksi. [15, s. 12-14]

11.1 SV-luokittelija

Oletetaan, että selittävät muuttujat saavat arvoikseen reaalilukuja ja että on vain yksi selitettävä muuttuja, joka saa arvoja 1 ja -1 sen mukaan kumpaan luokkaan havainto kuuluu. SVM-menetelmiä voidaan laatia myös regressiotilanteisiin, joissa ennustettava muuttuja on reaaliluku. Rajoitutaan tässä kuitenkin kaksiluokkaiseen vasteeseen, koska se saattaisi sopia hyvin diabetestutkimuksen aineistoon, jossa halutaan ennustaa kuka tutkimushenkilöistä sairastuu ($y_i = 1$) ja kuka ei ($y_i = -1$).



Kuva 7: SV-luokittelija kaksiulotteisessa aineistossa. Sairastuneet on merkitty neliöillä ja terveet ympyröillä. Vasemmalla havaintoaineisto on lineaarisesti separoituva, oikealla ei. Suora $f(x)$ on luokitteluraja, $C = 1/\|\beta\|$ on puolet optimaalisesta marginaalista. Tukivektorit on merkitty valkoisella sisuksella. Tukivektorien etäisyydet marginaalista (ξ_i) on piirretty viivoilla.

Määritellään hypertasoksi

$$\{x : f(x) = x^T\beta + \beta_0 = 0\}, \tag{59}$$

missä β on yksikkövektori: $\|\beta\| = 1$. Hypertason mukainen luokittelusääntö on $y(x) = \text{sign}[x^T\beta + \beta_0]$, missä funktio sign saa arvon -1, jos argumentti on negatiivinen ja muulloin arvon 1. Pisteen x etäisyys tasosta saadaan funktiosta $f(x)$. Jos luokat ovat separoituvia, niin voidaan myös valita tasoista optimaalisin. Toisin sanoen, jos on mahdollista löytää taso $f(x)$, joka jakaa havainnot täysin oikein kahteen luokkaan: $y_i f(x_i) > 0 \ \forall i$, niin voidaan valita tasoista se, joka muodostaa suurimman marginaalin (ks. kuva 7) kahden luokan pisteiden välille:

$$\max_{\beta,\beta_0,\|\beta\|=1} \{C : y_i(x_i^T\beta + \beta_0) \geq C \ \forall i = 1, \ldots, N\}. \tag{60}$$

Optimointitehtävä voidaan kirjoittaa myös mukavampaan muotoon

$$\min_{\beta,\beta_0} \{\|\beta\| : y_i(x_i^T\beta + \beta_0) \geq 1 \ \forall i = 1, \ldots, N\}. \tag{61}$$

[9, s. 108-9]

Yleistetään luokittelija tilanteeseen, jossa luokat eivät ole separoituvia, vaan sijaitsevat osin limittäin. Muutetaan tehtävässä (60) ollutta epäyhtälön alarajaa C:stä $C(1 - \xi_i)$:ksi, missä $\forall i : \xi_i \geq 0, \sum_{i=1}^N \xi_i \leq \gamma$. Uusi merkintä ξ_i tarkoittaa suhteellista etäisyyttä, jonka verran havainnot poikkevat luokkansa puoleisesta marginaalista. Luokittelija tekee virheen, kun $\xi_i > 1$, joten summaa $\sum \xi_i$ rajoittava vakio γ määrää kuinka suureksi väärin luokiteltujen tapausten lukumäärä voi kasvaa. Kun siis havainnot eivät ole separoituvia, on pakko sallia luokitteluvirheet optimointitehtävässä:

$$\min_{\beta,\beta_0,\xi_i} \{\|\beta\| : y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \ \xi_i \geq 0, \ \sum \xi_i \leq \gamma, \ \forall i\}. \tag{62}$$

Tämä on konveksi optimointitehtävä, koska minimoitava funktio on toista astetta (etäisyysnormi) ja tehtävällä on lineaarisia epäyhtälörajoituksia. Ratkaistaan tehtävä Lagrangen menetelmällä. Sitä varten on mukavampi muotoilla tehtävä uudestaan:

$$\min_{\beta,\beta_0,\xi_i} \{\frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^N \xi_i : \forall i \ y_i(x_i^T\beta + \beta_0) \geq 1 - \xi_i, \ \xi_i \geq 0\}. \tag{63}$$

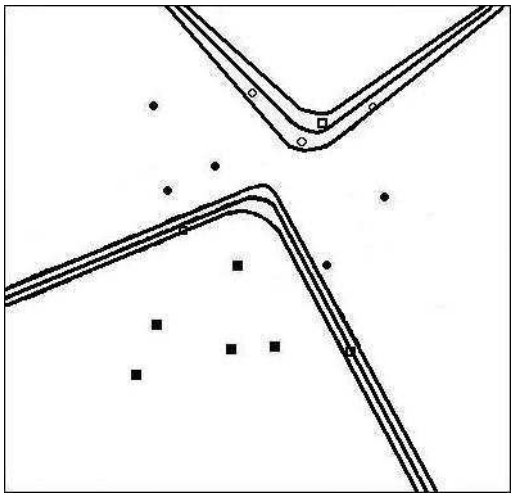
Optimoinnin tulos on muotoa

$$\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i x_i, \tag{64}$$

missä kertoimet $0 \leq \hat{\alpha}_i \leq \gamma$ ovat suurempia kuin nolla vain niille havainnoille i, joille epäyhtälössä $y_i(x_i^T\beta + \beta_0) - (1 - \xi_i) \geq 0$ toteutuu yhtäsuuruus. Muille havainnoille kertoimet $\hat{\alpha}_i$ ovat nollia. Niitä havaintoja i, joille $\hat{\alpha}_i > 0$, sanotaan *tukivektoreiksi* (eng. support vector), koska hypertason parametrit $(\hat{\beta})$ voidaan laskea käyttämällä vain näitä havaintoja.[9, s. 371-5]

Kuvassa 7, kuten yleisestikin, kaikki luokittelumarginaaliin tai kokonaan marginaalin väärälle puolelle jääneet havainnot ovat tukivektoreita. Sen sijaan marginaalin oikealla puolella olevat havainnot eivät ole tukivektoreita.

11.2 SVM-menetelmä



Kuva 8: SVM, jossa ydinfunktio on toisen asteen polynomi. Sairastuneita on merkitty neliöillä ja terveitä ympyröillä. Tukivektorit on merkitty valkoisella sisuksella. Aineisto on keintotekoinen.

Edellä esitelty SV-luokittelija etsii selittäjämuuttujien avaruudessa hypertason, joka optimaalisesti luokittelee havaintoaineiston kahteen osaan. Lineaarinen luokittelu voidaan yleistää kannan laajennuksella, jolloin on mah-

dollista luokitella aineisto käyttäen epälineaarisia rajoja. Näin luokittelu saadaan tarkemmaksi, koska on mahdollista ottaa huomioon havaintoaineiston epälineaariset piirteet. SVM-menetelmässä valitaan kantafunktiot $h_m(x)$, $m = 1, \ldots, M$ ja sovitetaan SV-luokittelija käyttämällä alkuperäisten havaintojen sijaan niiden muunnoksia $h(x_i) = (h_1(x_i), h_2(x_i), \ldots, h_M(x_i))^T$, $i = 1, \ldots, N$. Yhtälöstä (64) nähdään, että optimoitu parametrivektori on $\hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i y_i h(x_i)$. Näin saadaan epälineaarinen luokittelija

$$y(x) = \text{sign}[h(x)^T \hat{\beta} + \beta_0] \tag{65}$$

$$= \text{sign}[\sum_{i=1}^{N} \alpha_i y_i \langle h(x), h(x_i) \rangle + \beta_0], \tag{66}$$

jossa esiintyvää sisätuloa kutsutaan myös ydinfunktioksi (eng. kernel function) ja merkitään

$$K(x, x') := \langle h(x), h(x') \rangle . \tag{67}$$

SVM-menetelmää käytettäessä ei oikeastaan tarvitsekaan määrätä muunnosta $h(x)$, vaan riittää että määrää ydinfunktion K, joka antaa sisätulot muunnetussa avaruudessa. Ydinfunktion valinta ei silti ole mikään helppo tehtävä, itseasiassa se onkin oleellisin ongelma. Tässä tutkielmassa siihen ei kuitenkaan paneuduta, koska tarkoituksena oli vain lyhyesti esitellä SVM-menetelmä. Mainitaan kuitenkin, että tyypillisimpiä ydinfunktioita ovat d:nnen asteen polynomifunktio $(1+\langle x, x' \rangle)^d$, exponenttifunktio $\exp(-\|x-x'\|^2/c$ ja hyperbolinen tangenttifunktio $\tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$. Ydinfunktion lisäksi summaa $\sum \xi_i$ rajoittava vakio γ pitäisi osata valita sopivaksi. Esimerkin vuoksi kuvaan 8 on piirretty keinotekoinen SVM-menetelmän luokittelema aineisto.

11.3 Sovellus: SV-luokittelija diabetesaineistossa

Sovelletaan SV-luokittelijaa aineistoon, jossa on mukana samojen lasten määritystuloksia kuin logistisen regression ja neuroverkkojen sovelluksissa. Muuttujista on kuitenkin poikkeavasti valittu kaksi viimeistä varhaista insuliinivastearvoa. Kuvasta (9) havaitaan, että suurin osa (13/19) lapsista, joilla

molemmilla kerroilla on havaittu alentunut varhainen insuliinivaste, ovat sairastuneet. Useissa tapauksissa (6/19) lapsi ei ainakaan vielä ole sairastunut, vaikka insuliinivaste on ollut alhainen jo kaksi kertaa. Voi kuitenkin olla, että näin käy myöhemmin. Yksi lapsista on sairastunut, vaikka kaksi viimeistä mittausta kertovat ensivaiheen insuliinvasteiden olleen normaaleja. Sairastuminen lienee ollut niin nopeaa (alle 3kk), että lapsi ei ole ehtinyt käydä seuraavassa sokerirasituksessa. Yhdellätoista terveellä lapsella toinen insuliinivasteista on ollut alentunut ja toinen normaali.

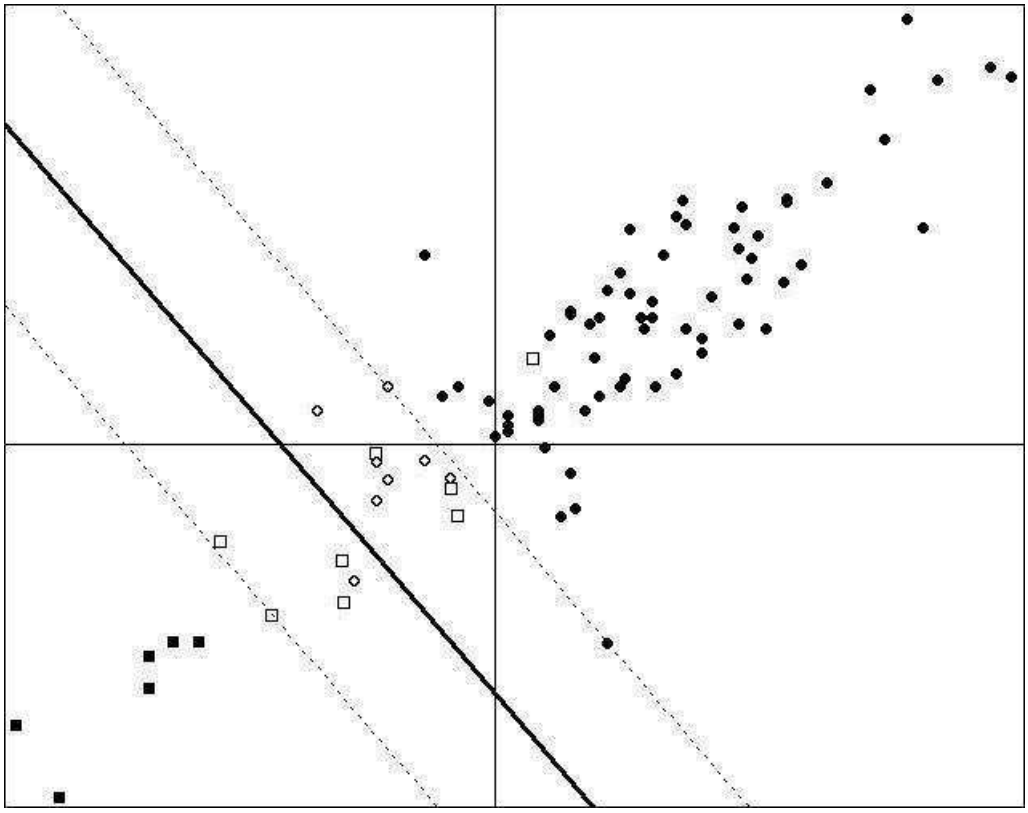
SV-luokittelijan ennustamiskykyä arvioitiin ristiinvalidoidulla herkkyydellä ja tarkkuudella. Ristiinvalidoinnissa aineisto jaettiin satunnaisesti viiteen osaan. Jokainen osista oli vuorollaan testiainestona muiden osien muodostaessa opetusaineiston. Herkkyydeksi saatiin 69% ja tarkkuudeksi 92%. Kuvan perusteella on vaikeaa ajatella, että jokin toinen ydinfunktio sopisi tilanteeseen paremmin kuin käytössä ollut tavallinen sisätulo. Tukivektorien etäisyyksien (marginaalista) summan ylärajaksi valittiin $\gamma = 2$, koska se tuntui antavan paremman tuloksen kuin esim. $\gamma = 1$ tai $\gamma = 5$.

SV-luokittelijaa yritettiin soveltaa myös samoille muuttujille kuin neuroverkkojen ja logistisen regressiomallin yhteydessä. 5-ulotteiseen aineistoon sovitettiin hypertaso, mutta tulokset jäivät hyvin heikoiksi. Herkkyys ja tarkkuus olivat alle 50 prosenttia. Voi olla, että jos hypertason sijaan käytettäisiin epälineaarista SVM-menetelmää, niin tulokset paranisivat. Sopivan ydinfunktion löytäminen jääköön kuitenkin muiden tehtäväksi.

12 Lopuksi

Tutkielmassa sovelletut menetelmät eivät tarjonneet suuria yllätyksiä. Mikään menetelmistä ei ollut paha pettymys, eikä niillä toisaalta saavutettu odottamattoman hyviä tuloksiakaan. Tehdään lopuksi muutamia huomautuksia.

Moniarvopaikkauksen käyttö on viime vuosina yleistynyt, ja siksikin sen esitteleminen tuntui tarpeelliselta. Ohjelmia ajettaessa tuli ilmi, että jos ajon



Kuva 9: SV-luokittelija diabetesaineistossa. Vaaka-akselilla on viimeisen mittauskerran insuliinivaste ja pysty-akselilla toiseksi viimeisen mittauskerran insuliinivaste. Alentuneen insuliinivasteen rajat (38 mU/l) on piirretty kuvaan pysty- ja vaakaviivoina. Molemmille muuttujille on tehty logaritmimuunnos. Sairastuneet (14 kpl) on merkitty neliöillä ja terveet (73 kpl) ympyröillä. Tukivektorit on merkitty valkoisella sisuksella. Ydinfunktio on tavallinen sisätulo, jolloin $h(x) = x$. Vakioksi γ valittiin 2. Ohjelmistona käytettiin Matlabilla tehtyjä valmiita ohjelmia [8]. Liitteessä on omatekoinen ohjelma, jolla tämä aineisto on analysoitu.

tekee useita kertoja, niin saa aina hieman erilaisia vastauksia. Moniarvopaikkaus on satunnaismenetelmä, joten tämä on jossain määrin odotettavaakin. Osittain syynä lienee myös se, että vasta-ainemuuttujat eivät noudata normaalijakaumaa. Ohjelma on laadittu normaalijakaumaoletukseen nojautuen. Tästä huolimatta moniarvopaikkauksella voi tulevaisuudessa olla käyttöä diabeteksen ennustamisessa.

Komponenttimallien ja neuroverkkojen käyttö on perusteltua juuri siksi, että muuttujat eivät noudata normaalijakaumaa. Komponenttimalli estimoitiin tutkielmassa EM-algoritmilla, joka ei salli puuttuvia havaintoja. On olemassa algoritmi, joka sallii puuttuvat havainnot.[6, 7] Se ei kuitenkaan ole globaali algoritmi. Olisi mielenkiintoista yrittää laajentaa tutkielmassa esitettyjä globaaleja algoritmeja niin, että sallisivat puuttuvat havainnot. Neuroverkko tuntui soveltuvan aineistoon paremmin kuin komponenttimallit. Komponenttimalleilla onnistuttiin kuitenkin eräiden muuttujien tapauksessa samaan uskottavampi malli kuin pelkällä parametrisella mallilla. Tarkemmin sanottuna kahden normaalijakauman yhdelmä oli parempi kuin pelkkä normaalijakauma mallintamaan tarkasteltua kaksiulotteista aineistoa. Aineiston muuttujat tosin valittiin sopivasti.

Logistisella regressiolla saavutettiin paremmat ennusteet kuin neurvoverkolla tai SV-luokittelijalla. Neuroverkko kykeni liki yhtä hyvään ennusteeseen kuin logistinen regressiomalli. Neuroverkkojen soveltamisessa on kuitenkin kohtia, joissa saattaisi olla parantamisen varaa. Useiden parametrien arvot on valittu sen tarkemmin perustelematta. Jos neuroverkkoennustajaa vielä parannettaisiin, niin saattaa olla että päästäisiin parempiin tuloksiin kuin logistisella regressiolla. Tässä tutkielmassa uudenaikaisemmilla menetelmillä ei kuitenkaan kyetty ennustamaan paremmin kuin perinteisillä tilastollisilla menetelmilläkään.

SVM-menetelmien soveltaminen jäi tutkielmassa vielä aivan alkutekijöihinsä, koska käytetiin vain SV-luokittelijaa ja rajoituttiin kahteen muuttujaan (poikkeavasti neuroverkkoihin ja logistiseen regressiomalliin nähden). On mahdollista, että SVMmenetelmien avulla aineistosta löydettäisiin nykyistä parempia tapoja ennustaa diabetesta. Siinä ongelmana on sopivan ydinfunktion löytäminen.

Toivottavaa on, että diabetes pystyttäisiin ennustamaan jo ennen kuin varhainen insuliinivaste heikkenee. Jatkossa voisikin asettaa tavoitteeksi löytää muuttujia ja ennustamismenetelmiä, joilla voisi ennustaa varhaista insuliinivastetta. Tällöin ennustettava muuttuja voisi olla joko kaksiluokkainen

(alentunut varhainen insuliinivaste vs normaali) kuten tässä tutkielmassa tai sitten jatkuva muuttuja.

On olemassa myös muita diabeteksen puhkeamiseen vaikuttavia tekijöitä, kuten perinnölliset tekijät ja virustartunnat. Nämä on saatava havaintoaineistoon ja malleihin mukaan, jos halutaan saada mahdollisimman tarkkoja ennusteita. Mainitut tekijät eivät ole jatkuvia muuttujia vaan kategorisia muuttujia. Tämä vaatinee hieman erilaisia tai muokattuja ennustamismenetelmiä. Tähän tutkielmaan perinnöllisiä tekijöitä ja virustartunta tietoja ei otettu tiedon keruuseen liittyvistä syistä.

Kirjallisuutta

[1] Bishop C.M.: Neural networks in pattern recognition. Oxford univerity press, 1995.

[2] Dempster A.P., Laird N.M., Rubin D.B: Maximum likelihood estimation from incomplete data via the EM algorithm. Journal of the royal statistical society B39 (1977), 1-38.

[3] Diabetesliiton www-sivusto. <http://www.diabetes.fi/diabtiet/perus/>. 5.12.2003.

[4] DIPP-projektin www-sivusto. <http://www.utu.fi/research/dipp/>. 26.5.2004.

[5] Encyclopedia of statistical sciences (toim. Kotz S. et al). Wiley, 1982-1999.

[6] Ghahramani Z., Jordan M.I.: Learning from incomplete data. MIT Center for Biological and Computational Learning Technical Report 108, Cambridge, MA, USA, 1994.

[7] Ghahramani Z., Jordan M.I.: Supervised learning from incomplete data via an EM approach. Advances in neural information processing systems 6 (toim. J.D. Cowan, G. Tesauro, J. Alspector), 120-127. Morgan Kaufmann, San Francisco CA, 1994.

[8] Gunn S. R.: Matlab svm toolbox. Saatavilla www-sivulta <http://www.isis.ecs.soton.ac.uk/resources/svminfo/>. 26.8.2004

[9] Hastie T.J., Tibshirani R., Friedman J.: The elements of statistical learning: data mining, inference and prediction. Springer, New York, 2001.

[10] Kalbfleisch J.G.: Probability and statistical inference II. Springer, New York, 1979.

[11] Little R.J.A., Rubin D.B.: Statistical analysis with missing data. Wiley, New York, 1981.

[12] Sarle W.S.: Prediction with missing inputs. JCIS '98 Proceedings vol II, Research Triengle Park NC, 1998 (toim. Wang P.P.), 399-402.

[13] Schafer J.L.: Analysis of incomplete multivariate data. Chapman, New York, 1997.

[14] Tresp V., Neuneier R. ja Ahmad S.: Efficient methods for dealing with missing data in supervised learning. Advances in neural information processing systems 7 (toim. G. Tesauro, D.S. Touretzky, T.K. Leen), 689-696. MIT press, Cambridge MA, 1995.

[15] Vapnik V.N.:The nature of statistical learning theory. Springer, New York, 1999.

[16] Verbeek J.J., Vlassis N., Kröse B.: Efficient greedy learning of gaussian mixture models. Neural computation 15(2) (2003), 469-485.

[17] Vlassis N., Likas A.: A greedy EM algorithm for gaussian mixture learning. Neural processing letters 15 (2002), 77-87.

A Ohjelmat

Tässä osassa esitetään ohjelmakoodit, joilla sovellukset on tehty. Ohjelmat on tehty kahdella kaupallisella ohjelmistolla:

- The SAS system for Windows v8.2
- MatLab v5.2

SVM-menetelmiä käytettiin MatLab:iin liitettävän lisäohjelmiston avulla. Ohjelmiston on tehnyt S.R. Gunn. Ohjelmiston voi ladata osoitteesta http://www.isis.ecs.soton.ac.uk/resources/svminfo/

Paranneltuja komponenttimalleja tehtiin samoin MatLabiin liitettävän lisäohjelmiston avulla. Sen on tehnyt J.J. Verbeek. Ohjelmiston voi hakea osoitteesta http://carol.science.uva.nl/~jverbeek/software/index_en.html

A.1 Moniarvopaikkaus luvussa 4.3

```
data fpir2viim;
  set gradu.fpir2viim;
  if logfpir2>.;
  keep logfpir1 logfpir2 id;
run;

proc sort data=fpir2viim;by id;run;

proc means n nmiss data=fpir2viim;run;

proc univariate data=fpir2viim;
  histogram;
run;

/* Määrätään toistojen lukumäärä ja poistettavien
   teoreettinen osuus */

%global iter;%let iter=100;
%global prosentti;%let prosentti=5;

data tulokset;
  ka=0;
  imp=0;
run;
```

```
/* kierros-makro poistaa satunnaisesti p-prosenttia aineiston
   tiedoista, tekee keskiarvopaikkausen ja moniarvopaikkauksen
   ja laskee molemmille menetelmille todellisten ja paikattujen
   arvojen erotusten neliösummat. */

%macro kierros;

proc iml;

p=&prosentti;
pros=p/100;

edit fpir2viim;
  read all var{logfpir1 logfpir2 id} into fpir2viim;
  read all var{logfpir1} into logfpir1;
  read all var{logfpir2} into logfpir2;
close fpir2viim;

fpir2viim2=fpir2viim;

lkm=nrow(fpir2viim);

do i=1 to lkm;
  if ranuni(0)<pros then fpir2viim2[i,1]=.;
  if ranuni(0)<pros then fpir2viim2[i,2]=.;
  if ranuni(0)<pros then fpir2viim2[i,3]=.;
end;

logfpir1_2=fpir2viim2[,1];
logfpir2_2=fpir2viim2[,2];

create fpir2viim2 var{id logfpir1_2 logfpir2_2};
  append;
close fpir2viim2;

quit;

proc summary data=fpir2viim2;
  var logfpir1_2 logfpir2_2;
  output out=out mean=logfpir1_mean logfpir2_mean;
run;

data fpir2viim3;
  merge fpir2viim fpir2viim2;
  _type_=0;
  keep logfpir1 logfpir2
```

```
       logfpir1_2 logfpir2_2
       _type_;
run;

data fpir2viim3;
  merge fpir2viim3 out;
  by _type_;
  drop _type_ _freq_;
run;

/*keskiarvopaikkaus*/

data fpir2viim3;
  set fpir2viim3;
  logfpir1_ka=logfpir1;
  logfpir2_ka=logfpir2;
  if logfpir1_2=. then logfpir1_ka=logfpir1_mean;
  if logfpir2_2=. then logfpir2_ka=logfpir2_mean;
run;

/*imputointi*/

data fpir2viim3;
  set fpir2viim3;
  logfpir1_imp=logfpir1_2;
  logfpir2_imp=logfpir2_2;
run;

proc mi data=fpir2viim3 out=fpir2viim3 nimpute=1 noprint;
  var logfpir1_imp logfpir2_imp;
run;

data fpir2viim3;
  set fpir2viim3;
  drop _imputation_;
run;

/*erotusten neliöitten summat*/

data fpir2viim3;
  set fpir2viim3;
  logfpir1_ka_nelio  = (logfpir1_ka - logfpir1)**2;
  logfpir2_ka_nelio  = (logfpir2_ka - logfpir2)**2;
  logfpir1_imp_nelio = (logfpir1_imp - logfpir1)**2;
  logfpir2_imp_nelio = (logfpir2_imp - logfpir2)**2;
run;
```

```
proc summary data=fpir2viim3;
  var logfpir1_ka_nelio logfpir2_ka_nelio
      logfpir1_imp_nelio logfpir2_imp_nelio;
  output out = out
         sum = logfpir1_ka_nelio logfpir2_ka_nelio
               logfpir1_imp_nelio logfpir2_imp_nelio;
run;

data out;
  set out;
  ka =  logfpir1_ka_nelio + logfpir2_ka_nelio;
  imp = logfpir1_imp_nelio + logfpir2_imp_nelio;
  keep ka imp;
run;

/* lasketaan eri iteraatiokierrosten neliösummien summa */

proc iml;

edit out;
  read all var{ka imp} into out;
close out;

edit tulokset;
  read all var{ka imp} into tulokset;
close tulokset;

tulokset[1]=tulokset[1]+out[1];
tulokset[2]=tulokset[2]+out[2];

edit tulokset;
  ka=tulokset[1];
  imp=tulokset[2];
  replace;
close tulokset;

quit;

%mend kierros;

/* luuppi-makro kutsuu kierros-makroa iteratiivisesti,
   ja laskee neliösummien keskiarvot. */

%macro luuppi;
%do iter=1 %to &iter;
  %kierros;
%end;
```

```
proc iml;
  edit tulokset;
    read all var{ka imp} into apuva;
    ka=apuva[1]/&iter;
    imp=apuva[2]/&iter;
replace;
  close tulokset;
quit;
%mend luuppi;

/* Suoritetaan makrot */

%luuppi;

proc print data=tulokset;run;

/* Tehdään regressioanalyysi moniarvopaikatulla aineistolla */

data fpir2viim;
  set gradu.fpir2viim;
run;

proc mi data=fpir2viim out=out nimpute=3;
  var logfpir1 logfpir2;
run;

proc reg data=out outest=outreg covout noprint;
  by _imputation_;
  model logfpir1 = logfpir2;
run;

proc mianalyze data=outreg;
  var intercept logfpir2;
run;

/* Tehdään regressioanalyysi jossa on mukana vain
   täydelliset havainnot */

proc reg data=fpir2viim;
  model logfpir1 = logfpir2 / clb;
run;quit;
```

A.2 Yhden vasta-aineen malli luvussa 7.2

```
   *exponentti ja normaalijakauman yhdelmä;
proc iml;
  start mean(r);
```

```
    a=r[+]/nrow(r);
    return(a);
  finish mean;
  edit vastaine;
  read all var{loggad} into x;
  close vastaine;
  N=nrow(x);
  p1x=j(N,1,0.5);
  p2x=j(N,1,0.5);
  do t=1 to 100;
    p1=mean(p1x);
print(p1);
p2=mean(p2x);
mu=sum(p2x#x)/sum(p2x);
sigma2=sum(p2x#(x-mu)##2)/sum(p2x);
theta=sum(p1x#x)/sum(p1x);
px1=exp(-x/theta)/theta;
px2=(exp(-(x-mu)##2/2/sigma2))/(sqrt(2*3.14*sigma2));
p1x=(px1*p1)/(px1*p1+px2*p2);
p2x=1-p1x;
  end;
  print(mu); print(sigma2); print(theta);
quit;
   *kahden normaalijakauman yhdelmä;
proc iml;
  start mean(r);
    a=r[+]/nrow(r);
    return(a);
  finish mean;
  edit vastaine;
  read all var{loggad} into x;
  close vastaine;
  N=nrow(x);
  p1x=j(N,1,0.9);
  p2x=j(N,1,0.1);
  do t=1 to 200;
    p1=mean(p1x);
print(p1);
p2=mean(p2x);
mu_1=sum(p1x#x)/sum(p1x);
sigma2_1=sum(p1x#(x-mu_1)##2)/sum(p1x);
mu_2=sum(p2x#x)/sum(p2x);
sigma2_2=sum(p2x#(x-mu_2)##2)/sum(p2x);
px1=(exp(-(x-mu_1)##2/2/sigma2_1))/(sqrt(2*3.14*sigma2_1));
px2=(exp(-(x-mu_2)##2/2/sigma2_2))/(sqrt(2*3.14*sigma2_2));
p1x=(px1*p1)/(px1*p1+px2*p2);
p2x=1-p1x;
```

```
  end;
  print(mu_1); print(mu_2); print(sigma2_1); print(sigma2_2);
quit;
```

A.3 Tehokkaampi ahne algoritmi luvussa 7.5

```
  %luetaan aineisto
A = dlmread('fpir2viim_ei_otsikoita.txt', '\t');
[nrow,ncol]=size(A)
muutt_1=11;    %logfpir1
muutt_2=13;    %logica1
k=3;           %komponenttien maksimilukumäärä
  % tavallinen EM k-means alkuarvoilla
kajat = [];
klu = [];
for j=1:k   %lasketaan mallit joissa on 1,2,...,k komponenttia
   log_usk=[];
   ajat=[];
   for i=1:10   %lasketaan i-komponenttinen malli 10 kertaa, testiaineisto vaihtelee
      B=sortrows([A(1:nrow,muutt_1),A(1:nrow,muutt_2),transpose(randperm(nrow))],3);
      X=[B(1:nrow-30,1),B(1:nrow-30,2)];
      T=[B(nrow-29:nrow,1),B(nrow-29:nrow,2)];
      t=cputime;
      [W,M,R,Tlogl] = em(X,T,j,0,0,0);
      ajat = [ajat cputime-t];
      log_usk = [log_usk Tlogl];
   end
   klu = [klu mean(log_usk)];
   kajat = [kajat mean(ajat)];
   fprintf('keskim. log-usk. %f (keskihaj. %f). Paras: %f \n',mean(log_usk),std(log_usk),max(log_usk));
   fprintf('keskim. suoritusaika %.2f s\n', mean(ajat));
end
fprintf('1-k -komponenttisten mallien yhteisajoaika keskim. %.2f s\n', sum(kajat));
[turha maxkohta]=max(klu);
fprintf('Uskottavin komponenttien lukumäärä: %.0f \n',maxkohta);
  % tehokas ahne EM
ajat=[];
log_uskt=[];
for i=1:10
   B=sortrows([A(1:nrow,muutt_1),A(1:nrow,muutt_2),transpose(randperm(nrow))],3);
   X=[B(1:nrow-30,1),B(1:nrow-30,2)];
   T=[B(nrow-29:nrow,1),B(nrow-29:nrow,2)];
   t=cputime;
   [W,M,R,Tlogl] = em(X,T,k,2,0,0);
   ajat=[ajat cputime-t];
   [turha maksimikohta]=max(Tlogl);
   log_uskt=[log_uskt Tlogl];
```

```
   fprintf('uskottavin komponenttien lukumäärä %.0f \n',maksimikohta);
end
fprintf('Aikaa kului keskimäärin %.2f s\n',mean(ajat));
fprintf('log-usk. mean %f \n',mean(log_uskt,2));
fprintf('log-usk. std %f  \n',std(log_uskt,0,2));
fprintf('log-usk. max %f  \n',max(log_uskt,[],2));
  %piirretään kuva
B2=[A(1:nrow,muutt_1),A(1:nrow,muutt_2)];
[W,M,R,Tlogl] = em(B2,[],2,0,1,0);
```

A.4 Logistinen regressio luvussa 9

```
proc logistic data=fpir2viim descending;
  model sairas = fpir1 fpir2 ica1 ica2 ia2_1 ia2_2 gad1 gad2
        / selection=backward lackfit;
run;

* Ennustamista varten käytetään ristiinvalidointia.
  Lasketaan herkkyys ja tarkkuus ristiinvalidoimalla.;

data fpir2viim;
  set gradu.fpir2viim;
  where fpir2>. and ica1>. and ia2_1>. and gad1>.;
run;

proc means n nmiss data=fpir2viim;
  var sairas fpir1 fpir2 ica1 ia2_1 gad1;
run;

%macro osita;

proc iml;
edit fpir2viim;
  read all var{fpir1} into fpir1;
close fpir2viim;

lkm=nrow(fpir1);
osa=j(lkm,1,.);
do i=1 to lkm;
  osa[i]=round(ranuni(0)*(&rv-1))+1;
end;

create osa var{osa};
  append;
close osa;
quit;
```

```
data fpir2viim;merge fpir2viim osa;run;

proc freq data=fpir2viim;table osa / nocum;run;

%mend osita;

%macro kierros;

%local kierros_herkkyys;
%local kierros_tarkkuus;
%let kierros_herkkyys=0;
%let kierros_tarkkuus=0;

data opetus;
  set fpir2viim;
  where osa^=&kierros_ind;
run;

data testi;
  set fpir2viim;
  where osa=&kierros_ind;
run;

proc logistic data=opetus descending;
  ods output ParameterEstimates=parms (keep=variable estimate);
  ods select none;
  model sairas = fpir1 fpir2 ica1 ia2_1 gad1
        / selection=backward;
run;

proc iml;
edit parms;
  read all var{estimate} into estimate;
close parms;
edit testi;
  read all var{fpir1 fpir2 ica1 ia2_1 gad1 sairas} into testi;
  read all var{sairas} into sairas;
close testi;
n_sel=nrow(estimate);
n_testi=nrow(testi);
tulo=j(n_sel,1,0);
summa=j(n_testi,1,0);
ennuste=j(n_testi,1,0);
do j=1 to n_testi;
  do i=1 to n_sel-1;
    tulo[i]=estimate[i+1]*testi[j,i];
  end;
```

```
  summa[j]=sum(tulo)+estimate[1];
  if summa[j]>=0 then ennuste[j]=1;
  else ennuste[j]=0;
end;
create tulos var{sairas ennuste};
  append;
close tulos;
quit;

proc freq data=tulos noprint;
  tables sairas*ennuste / nopercent nocol out=out outpct;
run;
data out;
  set out;
  if (sairas=0 and ennuste=0) then tark=pct_row;
  if (sairas=1 and ennuste=1) then herk=pct_row;
run;
proc summary data=out;
  var tark herk;
  output out=kier&kierros_ind max=tark herk;
run;

%mend kierros;

%macro luuppi;
%osita;
%do kierros_ind=1 %to &rv;
  %kierros;
%end;
data tulokset;
  set %do kierros_ind=1 %to &rv;kier&kierros_ind %end;
;run;
%mend luuppi;

%global kierros_ind;
%global rv;
%let rv=5;
ods select all;
%luuppi;
ods select all;
proc means data=tulokset;var tark herk;run;
```

A.5 Neuroverkot luvussa 10.4

```
proc standard data=gradu.fpir2viim out=fpir2viim mean=0 std=1;
   var logfpir1 logfpir2 logica1 logica2 logiaav1 logiaav2
       logiaau1 logiaau2 logia2_1 logia2_2 loggad1 loggad2;
```

```
   where logfpir2>.;
run;

proc freq data=fpir2viim;tables sairas;run;

/* proc contents;run;
   proc means n nmiss;
     var sairas logfpir1 logfpir2 logica1 logiaau1 logia2_1 loggad1;
   run; */

data fpir2viim;
  set fpir2viim;
  if mod(id,5)=0 then gad1=.;
  bias=1;
run;

proc iml;   *tehdään tanh- ja sigmoidi-funktiot;
start tanh(r);
  if r<=20 then
    if r<-20 then a=-1;
else do;
  b=exp(r);
      c=exp(-r);
      a=(b-c)/(b+c);
    end;
  else a=1;
  return(a);
finish tanh;
start sigmoidi(r);
  if r<=20 then
    if r<-20 then a=0;
else do;
      b=exp(-r);
      a=1/(1+b);
    end;
  else a=1;
  return(a);
finish sigmoidi;
store module=_all_;
quit;

* Yhden piilotason multilayer perceptron -verkon (MLP) ratkaisu,
  kun data sisältää puuttuvia havaintoja.
  Tallenna read all riveille oikeat muuttujanimet. Koodi
  rajoittuu yhden output-muuttujan tapaukseen. Huomaa myös, että
  testi ja treeni datoissa tulee olla samat input-muuttujat.
;
```

```
*piilomuuttujien maksimilukumäärä.;
%global H_max;
%let H_max=5;

* jaetaan aineisto herkkyyden ja tarkkuuden
  ristiinvalidoinnin takia rv:hen osaan;
%global rv;
%let rv=5;

* jaetaan ei-testi-aineisto solmujen lukumäärän
  ristiinvalidoinnin takia rv2:hen osaan;
%global rv2;
%let rv2=5;


%macro osita;
proc iml;
edit fpir2viim;
  read all var{bias} into bias;
close fpir2viim;
lkm=nrow(bias);
osa=j(lkm,1,.);
do i=1 to lkm;
  osa[i]=round(ranuni(0)*(&rv-1))+1;
end;
create osa var{osa};
  append;
close osa;
quit;
data fpir2viim;merge fpir2viim osa;run;
proc freq data=fpir2viim;table osa / nocum;run;
%mend osita;

%macro osita2;
data fpir2viim2;
  set fpir2viim;
  where osa^=&rvind;
run;
proc iml;
edit fpir2viim2;
  read all var{bias} into bias;
close fpir2viim2;
lkm=nrow(bias);
osa2=j(lkm,1,.);
do i=1 to lkm;
  osa2[i]=round(ranuni(0)*(&rv-1))+1;
```

```
end;
create osa2 var{osa2};
  append;
close osa2;
quit;
data fpir2viim2;merge fpir2viim2 osa2;run;
proc freq data=fpir2viim2;table osa2 / nocum;run;
%mend osita2;

%macro kierros;
* jaetaan ei-testi-aineisto fpir2viim2 mallin valinnan
  takia ristiinvalidointiosiin (osa2);
%osita2;

proc iml;
load module=_all_;

*ristiinvalidointivirhe eri piilomuuttujien lukumäärillä;
sumtoterr2_H=j(&H_max,1,.);

do H=1 to &H_max;

sumtoterr2=0;

do rvind2=1 to &rv2;

  *luetaan rvind2:in määrämä osa validointiaineistoksi ja muut osat opetusaineistoksi;
  edit fpir2viim2;
    read all var{sairas logfpir1 logfpir2 logica1 logia2_1 loggad1}
where (osa2=rvind2) into printti;
    read all var{logfpir1 logfpir2 logica1 logia2_1 loggad1 bias}
where (osa2=rvind2) into testi;
    read all var{sairas}
where (osa2=rvind2) into testiout;
    read all var{logfpir1 logfpir2 logica1 logia2_1 loggad1 bias}
where (osa2^=rvind2) into reeni;
    read all var{sairas}
where (osa2^=rvind2) into reeniout;
  close fpir2viim2;

  N=nrow(reeni);  * N on opetusaineiston koko;
  N2=nrow(testi); * N2 on testiaineiston koko;
  D=ncol(reeni);  * D on input-muuttujien lkm;
  sigma=0.1;      * sigma on gaussien keskihajonta;
  sigmay=0.1;
  C=30;     * näin monta kertaa reeniaineisto käydään läpi;
  lo=0.1;         * output-painojen muutosnopeuden kerroin;
```

```
  li=0.2;         * input-painojen muutosnopeuden kerroin;

  print(N);
  print(N2);
  print(H);

  *määritellään painomatriisit;
  wo=j(H+1,1,.);        * painot ennen outputtia;
  wi=j(H+1,D,.);        * painot inputin jälkeen;
  do v1=1 to H+1;       * satunnaistetaan painot;
    wo[v1]=(ranuni(0)-0.5);
    do v2=1 to D;
  wi[v1,v2]=(ranuni(0)-0.5);
end;
  end;

  *määritellään muita matriiseja;
  hid=j(H+1,1,0);          * piilomuuttujien arvot;
  hid[H+1]=1;              * lisätään bias myös piilotasolle, inputtasolla se on datassa;
  hidapu=j(H+1,1,0);       * apu-muuttuja;
  outapu=j(H+1,1,0);       * apu-muuttuja;
  toterr=0;    * kokonaisvirhe, kun reeniaineisto on käyty läpi kerran;
  out=j(N,1,.);           * ennuste;
  erro=j(N,1,.);          * virhe;
  x=j(D,1,0);             * D-pituinen erotusvektori;
  apudata1=reeni;


*neuroverkon opettaminen;

  do j=1 to C;

    do i=1 to N;

      g2=0;
    g1=0;

     a=j(H+1,1,0);
     agg=j(H+1,1,0);
     aggsumma=j(H+1,1,0);

     a2=j(H+1,D,0);
     agg2=j(H+1,D,0);
     aggsumma2=j(H+1,D,0);

     ggsumma=0;
```

```
  taydi=1;

do q=1 to D;
  if reeni[i,q]=. then taydi=0;
end;

if taydi=0 then
  do p=1 to N;

    *p=floor(N*ranuni(0))+1; *valitaan reenihahmo satunnaisesti;
    taydp=1;
dim=D;
  do q=1 to D; *katsotaan onko puuttuvia, ja niiden varalta tehdään apumuuttujia;
  if reeni[p,q]=. then taydp=0;
      x[q]=(reeni[i,q]-reeni[p,q])**2;
      if x[q]=. then dim=dim-1;
  if reeni[i,q]=. then apudata1[i,q]=reeni[p,q];  *apudata on tehty alussa;
  end;
            *jos p on täydellinen, niin sillä voi paikata i:n puutteet;
  if taydp=1 then
  do;
      g2=exp(-x[+]/2/sigma/sigma)/((2*3.14*sigma*sigma)**(dim/2));
      do t=1 to H;
        apu1=0;
        do s=1 to D;
        apu1=apu1+apudata1[i,s]#wi[t,s];
      end;
        hidapu[t]=apu1;
      end;
      do t=1 to H;  *lasketaan piilomuuttujien arvot;
        apuva=hidapu[t];
        hid[t]=tanh(apuva);
      end;
    hid[H+1]=1;
      outapu=wo#hid;     *lasketaan target-muuttujan arvot;
  apuva=outapu[+];
      out[i]=sigmoidi(apuva);
      erro[i]=out[i]-reeniout[i];

  g1=exp(-(erro[i])**2/2/sigmay/sigmay)/((2*3.14*sigmay*sigmay)**(D/2));

      a=erro[i]*hid; *summien termit output-painoille;
    agg=a*g1*g2;
    aggsumma=aggsumma+agg;

      a2=erro[i]*wo#(1-hid##2)*apudata1[i,]; *summien termit input-painoille;
    agg2=a2*g1*g2;
```

```
    aggsumma2=aggsumma2+agg2;

  gg=g1*g2;  *molemmille painotyypeille tarvittavat summat;
  ggsumma=ggsumma+gg;

  g1=0;g2=0;

    end; *if:n;

  end; *m:n eli p:n;

 if taydi=1 then
do;
    do t=1 to H;
      apu1=0;
      do s=1 to D;
      apu1=apu1+reeni[i,s]*wi[t,s];
    end;
      hidapu[t]=apu1;
    end;
    do t=1 to H;  *lasketaan piilomuuttujien arvot;
      apuva=hidapu[t];
      hid[t]=tanh(apuva);
    end;
  hid[H+1]=1;
    outapu=wo#hid;     *lasketaan target-muuttujan arvo;
apuva=outapu[+];
    out[i]=sigmoidi(apuva);
    erro[i]=reeniout[i]-out[i];
dwo=(lo/j)*erro[i]*hid;
dwi=((li/j)*erro[i]*wo#(1-hid##2))*reeni[i,];
    wo=wo+dwo;
wi=wi+dwi;
end; *toisen if:n;

  if abs(ggsumma)>. & taydi=0 then
  do;
  dwo=-(lo/j)*aggsumma/ggsumma;
dwi=-(li/j)*aggsumma2/ggsumma;
    wo=wo+dwo; *uudet arvot output-painoille;
    wi=wi+dwi; *uudet arvot input-painoille;
end;

aggsumma=0;
ggsumma=0;
aggsumma2=0;
```

```
    end; *i:n;

  do i=1 to N;       *lasketaan kokonaisvirhe, neliöitten summa;
      toterr=toterr - ( reeniout[i]*log(out[i]) + (1-reeniout[i])*log(1-out[i]) );
  end;
    zz = j || toterr;
    print(zz);
    toterr=0;

  end; *j:n;

* ennustaminen;

  out2=j(N2,1,.);
  toterr2=0;
  x=j(D,1,0);
  apudata2=testi;
  *jos testiaineistossa on puuttuva,
   niin käytetään treeniaineistoa.;

  do i=1 to N2; *tehdään ennuste jokaiselle lapselle erikseen;

    g3summa=0; ag3summa=0;

    taydi=1;
  do l=1 to D;
    if testi[i,l]=. then taydi=0;
  end;

    if taydi=0 then do k=1 to N; *lasketaan N kpl alffoja ja geitä ja niiden tuloja;
    g3=0;
      *k=floor(N*ranuni(0))+1;
    taydk=1;
    dim=D;
      do l=1 to D;
        if reeni[k,l]=. then taydk=0;
        x[l]=(testi[i,l]-reeni[k,l])**2;
        if x[l]=. then dim=dim-1;
        if testi[i,l]=. then apudata2[i,l]=reeni[k,l];
      end;

    if taydk=1 then do;
    *print();
        g3=exp(-x[+]/2/sigma/sigma)/((2*3.14*sigma*sigma)**(dim/2));
        do t=1 to H;  *matriisikertolasku ei toimi puuttuville hav:lle!!;
          apu1=0;          *mutta suoritetaan se silmukoilla;
          do s=1 to D;
```

```
      apu1=apu1+apudata2[i,s]#wi[t,s];
    end;
        hidapu[t]=apu1;
      end;
      do t=1 to H;  *lasketaan piilomuuttujien arvot;
        apuva=hidapu[t];
        hid[t]=tanh(apuva);
      end;
    hid[H+1]=1;
      outapu=wo#hid;
  apuva=outapu[+];
      out2[i]=sigmoidi(apuva);

      ag3=out2[i]*g3;
  ag3summa=ag3summa+ag3;
  g3summa=g3summa+g3;
  end; *if:n loppu;

  end; *v:n eli k:n loppu;

if taydi=1 then do;
    do t=1 to H;
      apu1=0;
      do s=1 to D;
    apu1=apu1+testi[i,s]*wi[t,s];
  end;
      hidapu[t]=apu1;
    end;
    do t=1 to H;
      apuva=hidapu[t];
      hid[t]=tanh(apuva);
    end;
  hid[H+1]=1;
    outapu=wo#hid;
  apuva=outapu[+];
    out2[i]=sigmoidi(apuva);
end; *if:n loppu;

  if abs(g3summa)>. & taydi=0 then
    out2[i]=ag3summa/g3summa;

  g3=0; ag3=0; ag3summa=0; g3summa=0;

end; *i:n loppu;

z2=printti||out2;
print(z2);
```

```
  do i=1 to N2;
      toterr2=toterr2 - ( testiout[i]*log(out2[i]) + (1-testiout[i])*log(1-out2[i]) );
  end;

  print(toterr2);

  sumtoterr2=sumtoterr2+toterr2;
  toterr2=0;

end; *rvind2;

sumtoterr2_h[H]=sumtoterr2/&rv2;

end; *H;

print(sumtoterr2_h);
min_virhe=min(sumtoterr2_h);
print(min_virhe);
do b=1 to &H_max;
  if min_virhe=sumtoterr2_h[b] then min_lkm=b;
end;
print(min_lkm);

* Ja nyt kun malli (eli piilomuuttujien lukumäärä) on valittu,
  voidaan koko opetus+validointiaineistoa käyttää neuroverkon
  opettamiseen min_lkm:llä piilomuuttujalla sekä herkkyyden
  ja tarkkuuden laskemiseen testiaineistosta;

  edit fpir2viim;
    read all var{sairas logfpir1 logfpir2 logica1 logia2_1 loggad1}
where (osa=&rvind) into printti;
    read all var{logfpir1 logfpir2 logica1 logia2_1 loggad1 bias}
where (osa=&rvind) into testi;
    read all var{sairas}
where (osa=&rvind) into testiout;
    read all var{logfpir1 logfpir2 logica1 logia2_1 loggad1 bias}
where (osa^=&rvind) into reeni;
    read all var{sairas}
where (osa^=&rvind) into reeniout;
  close fpir2viim;

  H=min_lkm;

  N=nrow(reeni);      * N on opetusaineiston koko;
  N2=nrow(testi);     * N2 on testiaineiston koko;
  D=ncol(reeni);      * D on input-muuttujien lkm;
```

```
  print(N);
  print(N2);
  print(D);
  print(H);

  *määritellään painomatriisit;
  wo=j(H+1,1,.);        * painot ennen outputtia;
  wi=j(H+1,D,.);        * painot inputin jälkeen;
  do v1=1 to H+1;       * satunnaistetaan painot;
    wo[v1]=(ranuni(0)-0.5);
    do v2=1 to D;
  wi[v1,v2]=(ranuni(0)-0.5);
end;
  end;

  *määritellään muita matriiseja;
  hid=j(H+1,1,0);          * piilomuuttujien arvot;
  hid[H+1]=1;              * lisätään bias myös piilotasolle, inputtasolla se on datassa;
  hidapu=j(H+1,1,0);       * apu-muuttuja;
  outapu=j(H+1,1,0);       * apu-muuttuja;
  toterr=0;    * kokonaisvirhe, kun reeniaineisto on käyty läpi kerran;
  out=j(N,1,.);           * ennuste;
  erro=j(N,1,.);          * virhe;
  x=j(D,1,0);             * D-pituinen erotusvektori;
  apudata1=reeni;

  *neuroverkon opettaminen;

  do j=1 to C;

    do i=1 to N;

      g2=0;
    g1=0;

     a=j(H+1,1,0);
     agg=j(H+1,1,0);
     aggsumma=j(H+1,1,0);

     a2=j(H+1,D,0);
     agg2=j(H+1,D,0);
     aggsumma2=j(H+1,D,0);

     ggsumma=0;

      taydi=1;
```

```
do q=1 to D;
  if reeni[i,q]=. then taydi=0;
end;

if taydi=0 then
  do p=1 to N;

    *p=floor(N*ranuni(0))+1; *valitaan reenihahmo satunnaisesti;
    taydp=1;
dim=D;
  do q=1 to D; *katsotaan onko puuttuvia, ja niiden varalta tehdään apumuuttujia;
  if reeni[p,q]=. then taydp=0;
      x[q]=(reeni[i,q]-reeni[p,q])**2;
      if x[q]=. then dim=dim-1;
  if reeni[i,q]=. then apudata1[i,q]=reeni[p,q];  *apudata on tehty alussa;
  end;
             *jos p on täydellinen, niin sillä voi paikata i:n puutteet;
  if taydp=1 then
  do;
      g2=exp(-x[+]/2/sigma/sigma)/((2*3.14*sigma*sigma)**(dim/2));
      do t=1 to H;
        apu1=0;
        do s=1 to D;
        apu1=apu1+apudata1[i,s]#wi[t,s];
      end;
        hidapu[t]=apu1;
      end;
      do t=1 to H;  *lasketaan piilomuuttujien arvot;
        apuva=hidapu[t];
        hid[t]=tanh(apuva);
      end;
    hid[H+1]=1;
      outapu=wo#hid;     *lasketaan target-muuttujan arvot;
  apuva=outapu[+];
      out[i]=sigmoidi(apuva);
      erro[i]=out[i]-reeniout[i];

  g1=exp(-(erro[i])**2/2/sigmay/sigmay)/((2*3.14*sigmay*sigmay)**(D/2));

      a=erro[i]*hid; *summien termit output-painoille;
    agg=a*g1*g2;
    aggsumma=aggsumma+agg;

      a2=erro[i]*wo#(1-hid##2)*apudata1[i,];*summien termit input-painoille;
    agg2=a2*g1*g2;
    aggsumma2=aggsumma2+agg2;
```

```
  gg=g1*g2;  *molemmille painotyypeille tarvittavat summat;
  ggsumma=ggsumma+gg;

  g1=0;g2=0;

    end; *if:n;

  end; *m:n eli p:n;

 if taydi=1 then
do;
    do t=1 to H;
      apu1=0;
      do s=1 to D;
      apu1=apu1+reeni[i,s]*wi[t,s];
    end;
      hidapu[t]=apu1;
    end;
    do t=1 to H;  *lasketaan piilomuuttujien arvot;
      apuva=hidapu[t];
      hid[t]=tanh(apuva);
    end;
  hid[H+1]=1;
    outapu=wo#hid;     *lasketaan target-muuttujan arvo;
apuva=outapu[+];
    out[i]=sigmoidi(apuva);
    erro[i]=reeniout[i]-out[i];
dwo=(lo/j)*erro[i]*hid;
dwi=((li/j)*erro[i]*wo#(1-hid##2))*reeni[i,];
    wo=wo+dwo;
wi=wi+dwi;
end; *toisen if:n;

  if abs(ggsumma)>. & taydi=0 then
  do;
  dwo=-(lo/j)*aggsumma/ggsumma;
dwi=-(li/j)*aggsumma2/ggsumma;
    wo=wo+dwo; *uudet arvot output-painoille;
    wi=wi+dwi; *uudet arvot input-painoille;
end;

aggsumma=0;
ggsumma=0;
aggsumma2=0;

end; *i:n;
```

```
  do i=1 to N;       *lasketaan kokonaisvirhe, neliöitten summa;
      toterr=toter- ( reeniout[i]*log(out[i])+ 1-reeniout[i])*log(1-out[i]));
  end;
    zz = j || toterr;
    print(zz);
    toterr=0;

  end; *j:n;

* ennustaminen;

  out2=j(N2,1,.);
  toterr2=0;
  x=j(D,1,0);
  apudata2=testi;
  *jos testiaineistossa on puuttuva,
   niin käytetään treeniaineistoa.;

  do i=1 to N2; *tehdään ennuste jokaiselle lapselle erikseen;

    g3summa=0; ag3summa=0;

    taydi=1;
  do l=1 to D;
    if testi[i,l]=. then taydi=0;
  end;

    if taydi=0 then do k=1 to N; *lasketaan N kpl alffoja ja geitä ja niiden tuloja;
    g3=0;
      *k=floor(N*ranuni(0))+1;
    taydk=1;
    dim=D;
      do l=1 to D;
        if reeni[k,l]=. then taydk=0;
        x[l]=(testi[i,l]-reeni[k,l])**2;
        if x[l]=. then dim=dim-1;
        if testi[i,l]=. then apudata2[i,l]=reeni[k,l];
      end;

    if taydk=1 then do;
      *print();
        g3=exp(-x[+]/2/sigma/sigma)/((2*3.14*sigma*sigma)**(dim/2));
        do t=1 to H;  *matriisikertolasku ei toimi puuttuville hav:lle!!;
          apu1=0;          *mutta suoritetaan se silmukoilla;
          do s=1 to D;
          apu1=apu1+apudata2[i,s]#wi[t,s];
```

```
      end;
        hidapu[t]=apu1;
      end;
      do t=1 to H;  *lasketaan piilomuuttujien arvot;
        apuva=hidapu[t];
        hid[t]=tanh(apuva);
      end;
    hid[H+1]=1;
      outapu=wo#hid;
  apuva=outapu[+];
      out2[i]=sigmoidi(apuva);

      ag3=out2[i]*g3;
  ag3summa=ag3summa+ag3;
  g3summa=g3summa+g3;
  end; *if:n loppu;

  end; *v:n eli k:n loppu;

if taydi=1 then do;
    do t=1 to H;
      apu1=0;
      do s=1 to D;
      apu1=apu1+testi[i,s]*wi[t,s];
    end;
      hidapu[t]=apu1;
    end;
    do t=1 to H;
      apuva=hidapu[t];
      hid[t]=tanh(apuva);
    end;
  hid[H+1]=1;
    outapu=wo#hid;
  apuva=outapu[+];
    out2[i]=sigmoidi(apuva);
end; *if:n loppu;

  if abs(g3summa)>. /*0.000001*/ & taydi=0 then
    out2[i]=ag3summa/g3summa;

  g3=0; ag3=0; ag3summa=0; g3summa=0;

end; *i:n loppu;

out2diko=j(N2,1,.);
do b=1 to N2;
  if out2[b]>=0.5 then out2diko[b]=1;
```

```
    else out2diko[b]=0;
  end;

  create tulos var{out2diko testiout};
    append;
  close tulos;

quit;

proc freq data=tulos noprint;
  tables testiout*out2diko / nopercent nocol out=out outpct;
run;
data out;
  set out;
  if (testiout=0 and out2diko=0) then tark=pct_row;
  if (testiout=1 and out2diko=1) then herk=pct_row;
run;
proc summary data=out;
  var tark herk;
  output out=kier&rvind max=tark herk;
run;
%mend kierros;

%macro luuppi;
%osita;
%do rvind=1 %to &rv;
  %kierros;
%end;
data tulokset;
  set %do rvind=1 %to &rv;kier&rvind %end;
;run;
%mend luuppi;

%global rvind;
%luuppi;
proc means data=tulokset;var tark herk;run;
```

A.6 SVM luvussa 11.3

```
  % luetaan aineisto
A = dlmread('fpir2viim_ei_otsikoita.txt', '\t');
[nrow,ncol]=size(A);
muutt_1=11;    %logfpir1
muutt_2=12;   %logfpir2
sairas=2;   %sairas
% muutetaan terveet nollista miinus ykkösiksi
for i=1:nrow
```

```
   A(i,sairas)=2*A(i,sairas)-1;
end;
C=2;              %yläraja
ker='linear';     %ydinfunktio
  % tehdään svm-kuva
X=[A(1:nrow,muutt_1),A(1:nrow,muutt_2)];
Y=[A(1:nrow,sairas)];
[nsv alpha bias] = svc(X,Y,ker,C);
svcplot(X,Y,ker,alpha,bias);
  % lasketaan ristiinvalidoitu herkkyys ja tarkkuus
C=2;
lkm=5;
op=zeros(lkm,1);
on=zeros(lkm,1);
vp=zeros(lkm,1);
vn=zeros(lkm,1);
for s=1:nrow;
   satun(s)=floor(rand*lkm+1);
end;
for i=1:lkm
   jako=zeros(nrow,1);
   for s=1:nrow
      if (satun(s)==i)
         jako(s)=1;
      end;
   end;
   n_testi=sum(jako)
   n_opetus=nrow-n_testi;
   B=sortrows([A(:,muutt_1),A(:,muutt_2),A(:,sairas),jako],4);
   X1=B(1:n_opetus,1:2);
   Y1=B(1:n_opetus,3);
   X2=B(n_opetus+1:nrow,1:2);
   Y2=B(n_opetus+1:nrow,3);
   [nsv alpha bias] = svc(X1,Y1,ker,C);
   for k=1:n_opetus
      apu1(k)=alpha(k)*Y1(k)*X1(k,1);
      apu2(k)=alpha(k)*Y1(k)*X1(k,2);
   end;
   beta=[sum(apu1),sum(apu2)]
   eps=svtol(C);
   svii = find( alpha > eps & alpha < (C - eps));
   eka=svii(1)
   b0=-X1(eka,:)*transpose(beta)
   figure(i);
   hold on;
   for j=1:n_testi
      yht = X2(j,:)*transpose(beta)+b0;
```

```
      if (Y2(j)==1 & yht>0)
         op(i)=op(i)+1;
      elseif (Y2(j)==1 & yht<=0)
         vn(i)=vn(i)+1;
      elseif (Y2(j)==-1 & yht>0)
         vp(i)=vp(i)+1;
      else
         on(i)=on(i)+1;
      end;
      if (Y2(j)==1)
         plot(X2(j,1),X2(j,2),'b.','LineWidth',4);
      else
         plot(X2(j,1),X2(j,2),'r.','LineWidth',4);
      end;
   end;
   hold off;
end;
herk=100*(op)./(op+vn);
tark=100*(on)./(on+vp);
herkkyys=mean(herk)
tarkkuus=mean(tark)
```