Koneoppiminen unilääketieteessä

Pietari Mönkkönen
Luonnontieteiden kandidaatin tutkielma
Sovelletun fysiikan koulutusohjelma
Itä-Suomen yliopisto, Sovelletun fysiikan laitos
26. toukokuuta 2020

ITÄ-SUOMEN YLIOPISTO, Luonnontieteiden ja metsätieteiden tiedekunta
Sovelletun fysiikan koulutusohjelma, lääketieteellinen fysiikka
Pietari Mönkkönen: Koneoppiminen unilääketieteessä
Luonnontieteiden kandidaatin tutkielma, 25 sivua, 23 viitettä
Tutkielman ohjaajat: Sami Nikkonen, FM ja Timo Leppänen, FT
Toukokuu 2020

Avainsanat: koneoppiminen, tukivektorikone, satunnainen metsä, klusterointi, univaiheiden luokittelu, uniapnea

Tiivistelmä

Unihäiriöitä tai uniasairautta tutkittaessa potilaalle suoritetaan usein kliininen unirekisteröinti, jonka aikana kehosta mitataan useita signaaleja. Unirekisteröinnissä syntyy suuria määriä dataa, jonka läpikäynti vaatii yksityiskohtaista manuaalista työtä. Lisäksi eri ihmisten tekemissä analyyseissä on eroja, jotka voivat vaikuttaa tuktittavan henkilön diagnoosiin. Useat näistä analysointiprosesseista ovat kuitenkin hyvin mekaanisia. Eräs ratkaisu kyseisiin ongelmiin voisi olla koneoppimisen hyödyntäminen.

Tutkielman tavoitteena on selvittää, kuinka koneoppimista on hyödynnetty unilääketieteessä. Tutkielmassa esitellään koneoppimisen teoriaa, algoritmin opettamista, erilaisia oppimisskenaarioita ja ongelmia, johon koneoppimista voidaan soveltaa. Neuroverkot ja syväoppiminen jätettiin tutkielman ulkopuolelle.

Tutkielmassa perehdytään lyhyesti kolmeen eri koneoppimismenetelmään: Tukivektorikoneeseen, satunnaiseen metsään sekä klusterointiin. Tukivektorikone on menetelmä, joka yksinkertaisen luokitteluongelman tapauksessa, geometrisesti tulkittuna, sovittaa lähtömateriaaliin luokat toisistaan erottavan hypertason. Kyseinen hypertaso on sovitettu siten, että sen etäisyys lähimpiin datapisteisiin on mahdollisimman suuri. Satunnainen metsä on useisiin satunnaisuutta hyödyntäen luotuihin päätöspuu-luokittelijoihin perustuva menetelmä. Päätöspuu on varsin yksinkertainen luokittelija, mutta usean päätöspuun käyttö ja satunnaisuuden hyödyntäminen niiden muodostuksessa vähentävät yksittäisen päätöspuun ongelmia ja tällä tavoin saadaan muodostettua tehokas ja monikäyttöinen algortmi. Klusterointi on ohjaamattomaan oppimiseen lukeutuva menetelmä, jonka avulla lähtömateriaalista voidaan etsiä alaryhmiä tai muodostaa visualisaatioita. Tässä tutkielmassa klusterointia esitellään K:n keskiarvon klusterointimenetelmän sekä sidospohjaisen klusteroinnin avulla.

Unilääketieteessä yksi merkittävimmistä koneoppimisen sovelluskohteista on univaiheiden automaattinen luokittelu, johon ollaan sovellettu useita eri koneoppimismenetelmiä ja saadut tulokset ovat varsin tarkkoja. Toinen merkittävä sovelluskohde on uniapneaan keskittyvät sovellukset, joissa koneoppimista voidaan soveltaa muun muassa uniapnean vakavuuden arviointiin, kuorsauksen ja obstruktiivisen uniapnean yhteyden tutkimiseen sekä uniapnean fenotyyppien tutkimiseen. Tutkielmassa tarkastellaan lisäksi automaattista nukkuma-asennon tunnistamista sekä behavioraalisen unioireyhtymän automaattista havatsemista.

Sisältö

1 Johdanto

Henkilölle, jolla epäillään unihäiriötä tai unisairautta, suoritetaan usein kliininen unirekisteröinti. Unirekisteröinnin aikana mitataan useita eri signaaleja (kuten EEG, EKG, hengitysilmavirtaus ja veren happisaturaatio) [1] ja rekisteröidyn datan määrä on hyvin suuri. Tämän datan läpikäynti vaatii paljon yksityiskohtaista manuaalista työtä [1]. Lisäksi eri ihmisten tekemissä analyyseissä voidaan havaita selkeitä eroja [2], jotka voivat vaikuttaa tutkittavan henkilön diagnoosiin. Tämän seurauksena potilas ei välttämättä saa tarvitsemaansa hoitoa tai määrätty hoitomuoto ei ole potilaalle optimaalisin. Monet analysointiprosessit ovat kuintenkin hyvin mekaanisa ja tästä johtuen ne ovat hyviä sovelluskohteita koneoppimiselle.

Tutkielman tavoitteena on selvittää, kuinka koneoppimista on sovellettu unilääketieteessä. Tutkielmassa esitellään koneoppimisen teoriaa ja muutamien koneoppimisalgoritmien toimintaperiaatteita yleisellä tasolla. Tutkielman tarkoituksena on selvittää, minkälaisten unenaikaisten sairauksien, poikkeamien ja ilmiöiden havainnointiin ja analysointiin koneoppimista on mahdollista käyttää ja kuinka tarkkoja tuloksia koneoppimista hyödyntäen ollaan saatu. Tutkielma on kirjallisuuskatsaus ja materiaalina on käytetty aiheeseen liittyviä tieteellisiä julkaisuja ja kirjoja.

Koneoppiminen koostuu erilaisista laskennallisista menetelmistä, joita voidaan ”opettaa” samalla tavoin kuin ihminen oppii. Koneoppimisen avulla voidaan tehdä tarkkoja ennusteita tai havaita olennaisia kaavoja asioiden välillä. Koneoppimista hyödynnetään usein sovelluksiin, jotka ovat monimutkaisia tai vaativat suuren datamäärän läpikäyntiä. Koneoppiminen poikkeaa perinteisestä algoritmeihin perustuvasta ohjelmoinnista siten, että koneoppimislagoritmia ei tarvitse täsmällisesti ohjelmoida tehtävään, vaan se voidaan ennestään mitatun datan avulla opettaa suoriutumaan tehtävästä.

Tutkielmassa perehdytään tukivektorikoneen, satunnaisen metsän ja klusteroinnin toimintaperiaatteisiin. Tukivektorikone on menetelmä, joka esimerkiksi luokitteluongelmassa sovittaa lähtömateriaaliin hypertason, joka erottaa luokat toisistaan mahdollisimman hyvin. Mikäli luokat eivät ole erotettavissa hypertasolla, voidaan hyödyntää koordinaatistomuunnosta, jonka avulla sopiva hypertaso voidaan etsiä toisessa avaruudessa, tai voidaan käyttää kevyttä rajaehtoa, jolla osa pisteistä voidaan luokitella väärin, jotta sovitettu hypertaso yleistyisi paremmin. Satunnainen metsä puolestaan koostuu useasta satunnaisesti luodusta päätöspuu-luokittelijasta. Satunnaisen metsän rakenne on varsin helposti tulkittava päätöspuiden yksinkertaisen rakenteen vuoksi, mutta yksinkertaisuudestaan huolimatta satunnainen metsä on hvyin tehokas koneoppimismenetelmä. Klusteroinnin idea on jakaa lähtömateriaali erillisiin osajoukkoihin. Klusteroimalla lähtömateriaalista voidaan etsiä alaryhmiä tai visualisoida lähtömateriaalia. Klusterointi voidaan tehdä hyvin monella eri menetelmällä, mutta tässä tutkielmassa esitellään K:n keskiarvon klusterointimenetelmä ja sidospohjainen klusterointi.

Unilääketieteessä hyviä sovelluskohteita koneoppimiselle ovat erityisesti analysointiprosessien mekaaniset vaiheet. Näihin lukeutuu univaiheiden luokittelu, joka on hyvin tärkeä vaihe unen tutkimuksessa, mutta vaatii ihmiseltä paljon manuaalista työtä. Koneoppimsialgoritmi voidaan opettaa imitoimaan ihmisen tekemää univaiheiden luokittelua, joka perustuu ennalta määrättyihin sääntöihin. Koneoppimista voidaan hyödyntää myös unessa ilmenevien fysiologisten tilojen tutkimiseen ilman määrättyjä sääntöjä. Toinen hyvä sovelluskohde koneoppimiselle on uniapnean va-

kavuuden arviointi, joka perustuu uniapneapotilaalla ilmenevien hengityskatkosten määrään. Koneoppimisalgoritmi voidaan opettaa havaitsemaan hengityskatkot automaattisesti, jolloin ihmisen ei manuaalisesti tarvitse käydä unirekisteröintiä läpi. Koneoppimisen avulla voidaan myös tutkia kuorsauksen yhteyttä obstruktiiviseen uniapneaan tai taudin ilmenemismuotoja uniapneaa sairastavilla ihmisillä. Näiden sovelluskohteiden lisäksi koneoppimista voidaan soveltaa moneen muuhunkin ongelmaan unilääketieteessä. Tässä tutkielmassa käsitellään univaihe- ja uniapneasovellusten lisäksi nukkuma-asennon automaattista tunnistamista ja behavioraalisen unioireyhtymän automaattista havaitsemista.

2 Koneoppimisen perusteet

Koneoppimiseksi voidaan karkeasti määritellä laskennalliset menetelmät, jotka opetuksen myötä parantavat suorituskykyään ja pystyvät sen perusteella tekemään tarkkoja ennusteita syötetystä datasta [3]. Koneoppimisen päämääränä on olennaisten mallien tai kaavojen havaitseminen [4], sekä ennusteiden tekeminen [3]. Koneoppimisella on hyvin monia käyttökohteita, kuten internetin hakukoneet, kasvojentunnistus ja autojen elektroniset turvajärjestelmät [4].

Koneoppimista käytetään laajalti tieteellisessä tutkimuksessa esimerkiksi bioinformatiikassa, lääketieteessä ja tähtitieteessä [4]. Viime vuosikymmenten aikana koneoppimisesta on tullut hyvin yleinen työkalu tehtävissä, joissa suuresta datamäärästä pyritään poimimaan ollennaista tietoa [4], eli harjoitetaan niin kutsuttua tiedon louhintaa.

Koneoppimista hyödynnetään sovelluskohteisiin, joiden monimutkaisuuden ja suuren datamäärän johdosta ihminen ei voi täsmällisesti ohjelmoida algoritmia tehtävän suorittamiseen [4]. Sen sijaan algoritmin on pystyttävä oppimaan ja sopeuttamaan toimintansa kyseessä olevan tehtävän ja datan mukaisesti [4]. Lisäksi koneoppimista sovelletaan ongelmiin, joissa pyritään selvittämään yhteyttä jonkin syötteen ja ulostulon välillä [4]. Koneoppimisen vahvuus kyseisissä ongelmissa on se, ettei mahdollisesta syötteen ja ulostulon yhteydestä tarvitse tehdä juurikaan oletuksia, vaan koneoppimismentelmät pyrkivät luomaan mallin mahdollisimman vähillä oletuksilla [4].

2.1 Lähtömateriaali

Koneoppimisessa yksittäistä lähtömateriaalin alkiota kutsutaan esimerkkialkioksi (*example*) [3]. Esimerkkialkiolla on joukko mitattavia ominaisuuksia, joita kutsutaan muuttujiksi (*variable*) [5] tai piirteiksi (*feature*) [3]. Yleensä yksittäisen esimerkkialkion piirteet on ilmaistu reaalilukuina x_k, jolloin esimerkkialkio voidaan esittää vektorina $\bar{x}_i$, jonka alkioina on kutakin piirrettä vastaava reaaliluku [3]. Esimerkkialkio, jolla on n kappaletta piirteitä, voidaan siis matemaattisesti ilmaista pystyvektorina

$$\bar{x}_i = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}.$$

Esimerkiksi ihmisjoukkoa luokitellessa esimerkkialkiot ovat yksittäisiä henkilöitä ja piirteinä voi olla esimerkiksi henkilön ikä, pituus ja paino. Esimerkkialkioon $\bar{x}_i$ voi lisäksi olla liitettynä jokin vastealkio y_i, jota kutsutaan esimerkkialkion $\bar{x}_i$ leimaksi (*label*) tai ulostuloarvoksi (*output value*)[3, 5]. Vastealkio y_i voi olla esimerkiksi reaaliluku tai kategoria, joka kuvaa esimerkkialkiota $\bar{x}_i$ [3, 5].

2.2 Algoritmin opetus

Lähtömateriaalia $\{\bar{x}_i | i = 1, ..., m\}$ voidaan käyttää koneoppimisalgoritmin opettamiseen ja tulosten arviointiin [3]. Lähtömateriaalia voidaan kerätä esimerkiksi mittaamalla ilmiötä tai ongelmaa, johon koneoppimista halutaan soveltaa, ja valikoimalla sopivat piirteet mitatusta datasta [3]. Yleensä algoritmin opetus aloitetaan jakamalla

lähtömateriaali satunnaisesti kahteen ryhmään: opetusjoukkoon (*training sample*) ja testijoukkoon (*test sample*) [3]. Mikäli lähtomateriaalissa jokaista esimerkkialkioita $\bar{x}_i$ vastaa jokin ulostuloarvo y_i eli lähtömateriaali on muotoa $\{(\bar{x}_i, y_i)|i = 1, ..., m\}$, voidaan tehdä jako kolmeen ryhmään: opetusjoukkoon, testijoukkoon ja niin kutsuttuun validointijoukkoon (*validation sample*) [3].

Opetusjoukkoa käytetään nimensä mukaisesti koneoppimisalgoritmin opettamiseen [3]. Opetusvaiheessa algoritmi käy läpi opetusjoukon esimerkkialkiot ja rakentaa niiden pohjalta useita hypoteesimalleja, jotka poikkeavat toisistaan niin kutsuttujen vapaiden paramterien (*free parameters*) suhteen [3]. Validointivaiheessa validointijoukkoa käytetään hienosäätämään mallin vapaita parametrejä siten, että lopullinen malli ennustaa validointijoukon ulostuloarvot mahdollisimman tarkasti [3]. Validointijoukon tehtävä on estää ylioppimista, eli mallin liian tarkkaa sovittamista opetusjoukkoon [5]. Ylioppinut algoritmi ennustaa opetusjoukon esimerkkialkioiden ulostuloarvot hyvin tarkasti, muuta ei yleisty opetusjoukon ulkopuoliselle datalle [5]. Opetus- ja validointivaiheen jälkeen testijoukko syötetään algoritmiin tavoitteena selvittää, kuinka tarkasti opetettu algoritmi pystyy ennsutamaan testijoukolle oikeat ulostuloarvot y_i [3]. Tämä tapahtuu siten, että algoritmi käyttää testijoukon alkion $\bar{x}_i$ piirteitä ennustaakseen sille ulostuloarvon $\hat{y}_i$, jota verrataan todelliseen ulostuloarvoon y_i [3].

Toisinaan lähtömateriaali voi olla liian pieni erillisen testijoukon erotaamiselle [4]. Tällöin voidaan käyttää apuna ristiinvalidointia (*cross validation*) [4]. Yksi tapa tehdä ristiinvalidointi on niin kutsuttu k-kertainen ristiinvalidointi (*k-fold cross validation*) [4]. Kyseisessä menetelmässä lähtömateriaali jaetaan k:hon yhtäsuureen osajoukkoon ja algoritmia opetetaan k kertaa siten, että kukin osajoukko toimii vuorollaan testijoukkona [4]. Lopuksi jokaiselle testijoukolle saaduista tarkkuuksista lasketaan keskiarvo, joka toimii tarkkuuden arvona algoritmille [4]. Erikoistapaus k-kertaisesta ristiinvalidoinnista on tilanne $k = m$, missä m on esimerkkialkioiden määrä lähtömateriaalissa [4]. Tässä niin kutsutussa yksi-pois ristiinvalidoinnissa (*leave-one-out cross-validation*) kukin esimerkkialkio toimii vuorollaan yksinään testijoukkona muiden alkioiden koostaessa opetusjoukon [4].

2.3 Oppimisskenaariot

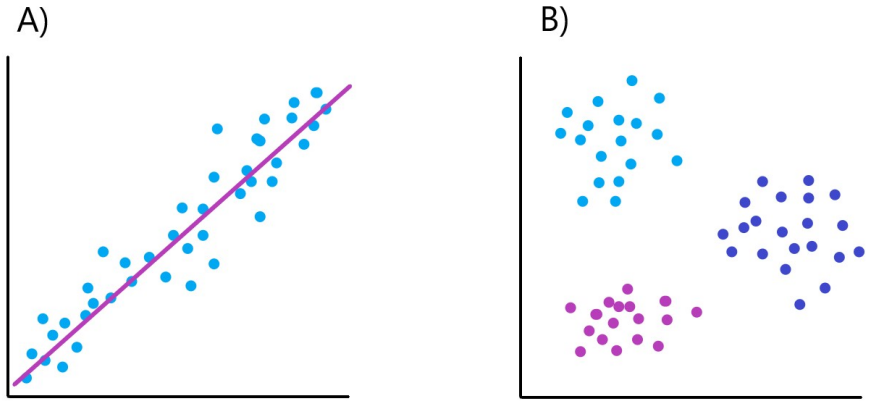
Tarkastellaan seuraavaksi kahta erilaista oppimisskenaariota: Ohjattua oppimista (*supervised learning*), jota käytetään ulostuloarvoiselle lähtömateriaalille $\{(\bar{x}_i, y_i)|i = 1, ..., m\}$, ja ohjaamatonta oppimista (*usupervised learning*), jota käytetään lähtömateriaalille, joka on muotoa $\{\bar{x}_i|i = 1, ..., m\}$ eli esimerkkialkioilla ei ole ulostuloarvoa [3, 5]. Näiden ohella on olemassa myös lukuisia muita oppimisskenaarioita, kuten osin ohjattu oppiminen (*semi-supervised learning*), transduktiivinen päättely (*transductive inference*) ja vahvistusoppiminen (*reinforcement learning*) [3].

Ohjatussa oppimisessa jokaista esimerkkialkiota $\bar{x}_i$ vastaa jokin ulostuloarvo y_i [5]. Tällöin pyritään löytämään koneoppimista hyödyntäen malli, joka liittää esimerkkialkion $\bar{x}_i$ ja sille kuuluvan ulostuloarvon y_i toisiinsa [5]. Tavoitteena on opettaa algoritmi, joka ennustaa sille syötettävien alkioiden vastaavat ulostulot tai selvittää syötteen ja ulostulon välistä yhteyttä [5]. Ohjaamatonta oppimista hyödynnetään tilanteessa, jossa näytteille $\bar{x}_i$ ei ole vastaavaa ulostuloa y_i [5]. Tällöin tarkoituksena ei ole tehdä ennusteita yksittäisille alkioille vaan etsiä jonkinlaisia rakenteita lähtömateriaalista [6]. Ohjaamattomalla oppimisella voidaan lähtömateriaalista etsiä esimerkiksi ryhmittymiä tai alkioiden välisiä yhteyksiä [5]. Näin ollen luokittelu tai sovitus joudutaan

tekemään ”sokkona”, kun käytössä ole ulostulomuuttujaa, jolla oppimista voitaisiin valvoa. Esimerkiksi klusterointi (*clustering*) on laajasti käytetty ohjaamattoman oppimisen menetelmä [5].

2.4 Sovelluskohteita

Yleisiä sovelluskohteita koneoppimiselle ovat regressio-ongelmat ja luokitteluongelmat [3, 5]. Regressio-ongelmasta puhutaan silloin, kun jokaista lähtömateriaalin esimerkkialkiota $\bar{x}_i$ vastaa kvantitatiivinen tai jatkuva ulostulomuuttuja y_i, eli muuttuja, joka saa arvon jatkuvalta reaalilukuväliltä [3, 5]. Kuvassa 1A on esimerkki regressiosta, jossa jatkuvaan dataan on sovitettu suora [5]. Luokitteluongelmasta puhutaan puolestaan silloin, kun jokaista lähtömateriaalin esimerkkialkiota $\bar{x}_i$ vastaa kvalitatiivinen tai kategorinen ulostulomuuttuja y_i, eli muuttuja, joka saa arvokseen jonkin luokan tai kategorian [3, 5]. Tällöin siis ulostuloarvot y_i voidaan jakaa erillisiin luokkiin. Kuvassa 1B on esitetty esimerkki luokittelusta.



Kuva 1: Kuvassa A on esimerkki regressio-ongelmasta. Siniset pallot kuvaavat jatkuva-arvoista lähtömateriaalia ja punainen viiva on pistejoukkoon sovitettu suora. Kuvassa B on esimerkki luokitteluongelmasta, jossa lähtömateriaalin pisteet on erotettu sijaintinsa perusteella erillisiksi osajoukoiksi.

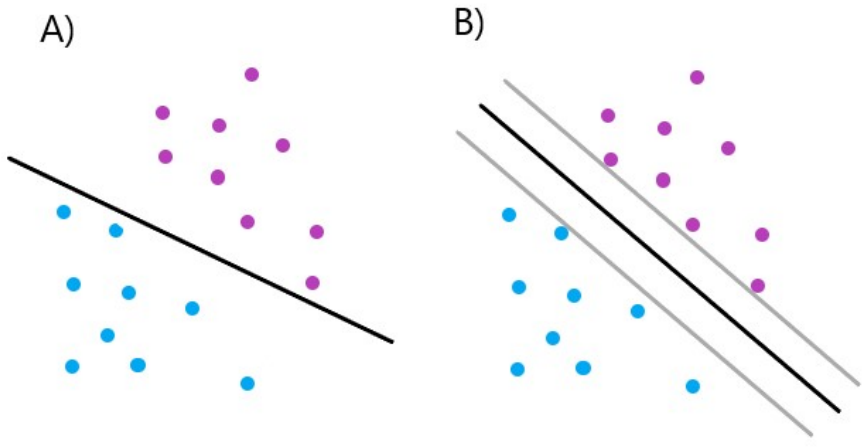
3 Koneoppimismenetelmiä

Tässä tutkielmassa perehdytään tarkemmin kolmeen koneoppimismenetelmään: tukivektorikoneeseen (*support vector machine*), satunnaiseen metsään (*random forest*) ja klusterointiin.

3.1 Tukivektorikone

Tukivektorikonemenetelmät ovat lineaarisia algoritmeja, joita voidaan soveltaa esimerkiksi luokitteluun ja regressio-ongelmiin [6]. Tukivektorikonemenetelmien katsotaan tällä hetkellä lukeutuvan modernin koneoppimisen tehokkaimpiin menetelmiin [3].

Tukivektorikonealgoritmien vaikean tulkittavuuden vuoksi sen toimintaa on mielekästä esitellä käyttäen yksinkertaista esimerkkiä. Tällaisiin esimerkkeihin lukeutuvat muun muassa lineaariset binääriluokitteluongelmat. Binääriluokittelussa esimerkkialkiot luokitellaan kahteen luokkaan, ja lineaarisuus tarkoittaa, että luokat ovat erotettavissa toisistaan syöteavaruuden hypertasolla [3]. Kyseessä oleva syöteavaruus rippuu lähtömateriaalista, mutta se voi olla esimerkiksi $\mathbb{R}^n$ osajoukko lähtömateriaalille, jonka esimerkkialkiot $\bar{x}_i$ ovat $\mathbb{R}^n$-vektoreita [3]. Lineaarisessa binääriluokitteluongelmassa yksinkertainen tukivektorikonealgoritmi geometrisesti tulkittuna sovittaa syöteavaruuteen hypertason, joka erottaa luokat toisistaan [3, 6]. Kysessä ei kuitenkaan ole mielivaltainen hypertaso, vaan algoritmin sovittama hypertaso erottaa luokat toisistaan siten, että hypertason kohtisuora etäisyys lähimpiin datapisteisiin on mahdollisimman suuri [3, 6]. Sovitetun tason ja sitä lähimpien pisteiden välistä kohtisuoraa etäisyyttä kutsutaan marginaaliksi [6]. Syöteavaruuden ollessa esimerkiksi $\mathbb{R}^2$, yksinkertainen tukivektorikone etsii suoran, joka erottaa luokat toisistaan mahdollisimman suurella marginaalilla [6] (kuva 2).



Kuva 2: Kuvassa A eri luokat (punainen ja sininen) erotettuna toisistaan mielivaltaisesti asetetulla hypertasolla. Kuvassa B musta viiva on tukivektorikoneen sovittama suurimman marginaalin hypertaso ja harmaat viivat kuvaavat marginaalihypertasoja.

3.1.1 Matemaattinen perusta

Tarkastellaan tukivektorimenetelmän matemaattista perustaa lineaarisen binääriluokitteluongelman tapauksessa. Tällöin siis lähtömateriaalin $S = \{(\bar{x}_i, y_i) \in \mathbb{R}^n \times \mathbb{R} | i \in [1, m]\}$ esimerkkialkiot jakautuvat kahteen eri luokkaan, jotka ovat lineaarisesti erotettavissa toisistaan. Tukivektorimenetelmien kanssa käytettävän termistön mukaan kyseessä on tällöin separoituva ongelma. Olkoon jokaisen esimerkkialkion $\bar{x}_i$ leima y_i määritelty siten, että $y_i \in \{-1, 1\}$ kaikilla $i \in [1, m]$.

Yleinen yhtälö hypertasolle avaruudessa $\mathbb{R}^n$ on

$$\bar{w} \cdot \bar{x} + b = 0, \tag{1}$$

missä $\bar{x}$ on yleinen tason piste, $\bar{w} \in \mathbb{R}^n$ on nollasta poikkeava hypertason normaalivektori ja $b \in \mathbb{R}$ on hypertason vakiotermi [3]. Yhtälöstä nähdään, että se on invariantti kerrottaessa nollasta poikkeavalla reaaliluvulla [3]. Näin ollen kaikille hypertasoille jotka eivät kulje esimerkkialkioiden $\bar{x}_i \in S$ läpi, voidaan skaalata normaalivektoria $\bar{w}$ ja vakiotermiä b siten, että

$$\min_{(\bar{x}_i, y_i) \in S} |\bar{w} \cdot \bar{x}_i + b| = 1. \tag{2}$$

Nimitetään kyseistä parin $(\bar{w}, b)$ määrittelemää hypertasoa kanoniseksi hypertasoksi [3].

Etäisyys mielivaltaisen pisteen $\bar{x}_0 \in \mathbb{R}^n$ ja yhtälön (1) määrittelemän hypertason välillä saadaan lausekkeesta

$$\frac{|\bar{w} \cdot \bar{x}_0 + b|}{\|\bar{w}\|}. \tag{3}$$

Näin ollen hypertason marginaali ρ eli etäisyys lähimpään esimerkkialkioon $\bar{x}_i$ on

$$\rho = \min_{(\bar{x}_i, y_i) \in S} \frac{|\bar{w} \cdot \bar{x}_i + b|}{\|\bar{w}\|} = \frac{1}{\|\bar{w}\|}, \tag{4}$$

jossa viimeinen yhtäsuuruus saadaan yhtälöstä (2) [3].

Otetaan tarkasteluun kanonisen hypertason lisäksi marginaalihypertasot, joilla on sama normaalivektori $\bar{w}$, mutta ovat marginaalin ρ etäisyydellä kanonisesta hypertasosta [3]. Kanoninen hypertaso on siis saman suuntainen kuin marginaalihypertasot ja kanonista hypertasoa lähimmät esimerkkialkiot positiivisella ja negatiivisella puolella sijaitsevat marginaalihypertasoilla [3]. Kuvassa 2B on havainnollistettu marginaalihypertasoja kaksiulotteisessa tapauksessa. Kanonisen hypertason määritelmän nojalla (yhtälö (2)) havaitaan, että marginaalihypertasojen yhtälöt ovat $\bar{w} \cdot \bar{x} + b = \pm 1$ [3].

Yleinen hypertaso $(\bar{w}, b)$ luokittelee esimerkkialkion $\bar{x}_i$ oikein silloin, kun lauseke $\bar{w} \cdot \bar{x}_i + b$ on samanmerkkinen kuin esimerkkialkiota $\bar{x}_i$ vastaava leima y_i [3]. Kanoniselle hypertasolle pätee yhtälön (2) nojalla $|\bar{w} \cdot \bar{x} + b| \geq 1$ kaikilla esimerkkialkioilla $\bar{x}_i$, joten kanoninen hypertaso luokittelee alkion $\bar{x}_i$ oikein, kun epäyhtälö $y_i(\bar{w} \cdot \bar{x}_i + b) \geq 1$ on voimassa [3]. Tukivektorikoneen tavoitteena on maksimoida marginaali, joka yhtälön (4) nojalla vastaa normin $\|\bar{w}\|$ minimoimista. Näin ollen tukivektorikoneen ratkaisu kyseiselle binääriluokitteluongelmalle saadaan seuraavasta optimointiongelmasta:

$$\min_{\bar{w}, b} \|\bar{w}\|, \tag{5}$$

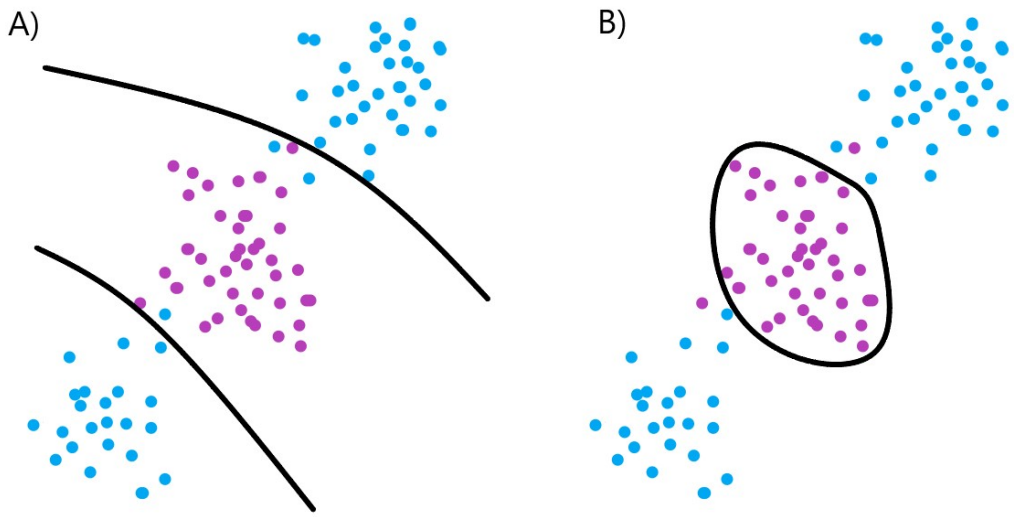
$$\text{kun } y_i(\bar{w} \cdot \bar{x}_i + b) \geq 1 \text{ kaikilla } i \in [1, m]. \tag{6}$$

Tukivektorikone pyrkii luokittelussaan maksimoimaan datapisteiden välisen etäisyyden kanonisesta hypertasosta, joten sovitus rippuu yleensä vain muutamasta tasoa lähinnä olevasta esimerkkialkiosta [6]. Näitä esimerkkialkioita, jotka määrittävät hypertason orientaation, kutsutaan tukivektoreiksi (*support vector*), joista algoritmin nimi on peräisin [6].

3.1.2 Epälineaarinen tapaus ja kevyt rajaehto

Tukivektorikonemenetelmän termistön mukaan ongelmaa, joka on lineaarinen ilman syöteavaruuden koordinaatistomuunnosta, nimitetään separoituvaksi ongelmaksi [3]. Aina eri luokkien esimerkkialkioita ei kuitenkaan voida erottaa hypertasolla, jolloin kysessä on ei-separoituva ongelma [3]. Tällöin avuksi voidaan ottaa koordinaatistomuunnos syöteavaruudesta johonkin sopivaan Hilbert-avaruuteen, jota kutsutaan piirreavaruudeksi (*feature space*) [7]. Piirreavaruudessa sopivasti valitun ytimen (*kernel*) avulla voidaan implisiittisesti löytää luokkia erottava hypertaso [6]. Tällainen muunnos on yleensä epälineaarinen ja piirreavaruuden dimensio voi poiketa syöteavaruuden dimensiosta ja olla jopa ääretön [7].

Jos lähtömateriaalissa on häiriötä tai esimerkkialkiot ovat limittäin myös piirreavaruudessa, mikä on yleistä käytännön ongelmissa [5], voidaan tukivektorikonealgoritmeissa hyödyntää kevyttä rajaehtoa (*soft margin*) [6]. Kevyt rajaehto sallii joidenkin esimerkkialkioiden väärän luokittelun, jolloin luokkien välisen hypertason löytäminen on mahdollista [6]. Kuvassa 3 on esitettynä ei-separoituva ongelma, jossa on käytetty epälineaarisia ytimiä sekä kevyttä rajaehtoa.



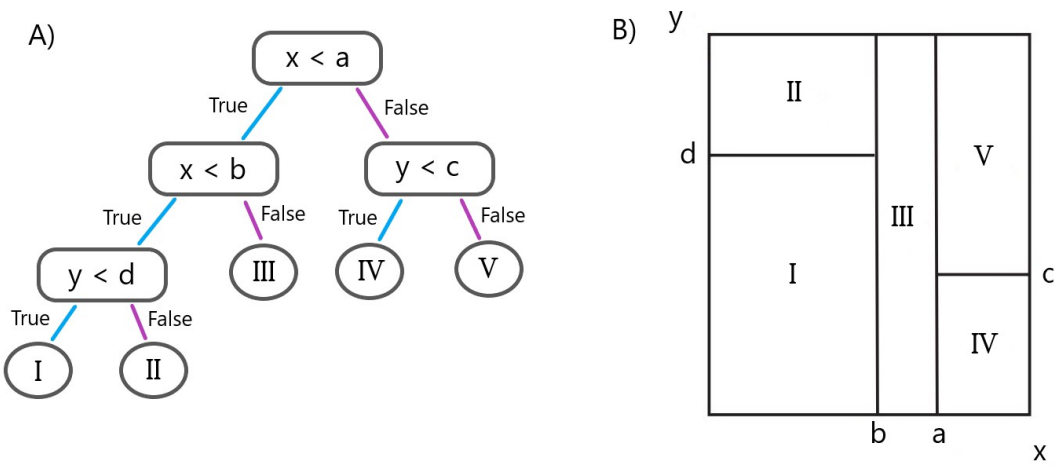
Kuva 3: Kuvissa A ja B on esitetty kaksi havainnollistavaa ratkaisua ei-separoituvalle ongelmalle käyttäen koordinaatistomuunnosta ja kevyttä rajaehtoa. Kuvan A tyyppinen ratkaisu voidaan saada käyttäen koordinaatistomuunnoksessa polynomiydintä. Kuvan B tyyppinen ratkaisu voidaan saada käyttäen säteittäistä (*radial*) ydintä. Molemmilla ydinvalinnoilla pystytään luokittelemaan kyseinen lähtömateriaali.

3.2 Satunnainen metsä

Satunnainen metsä on satunnaisuutta hyödyntävä koneoppimismenetelmä, jota voidaan käyttää esimerkiksi luokitteluun ja regressio-ongelmiin [5]. Menetelmä perustuu useisiin, varsin yksinkertaisiin päätöspuu-luokittelijoihin (*decision tree classifier*) ja niiden luonnissa käytettäviin satunnaismenetelmiin [8].

3.2.1 Päätöspuun toimintaperiaate

Päätöspuu on yksinkertainen ja helposti tulkittava algoritmi [5], joka luokittelee esimerkkialkion $\bar{x}_i$ sen piirteiden perusteella yhteen ennalta määrätyistä luokista [4]. Päätöspuu voidaan esittää yksinkertaisen puukaavion avulla [9]. Päätöspuuta havainnollistava puukaavio on esitetty kuvassa 4. Päätöspuu koostuu solmuista (*node*) ja niiden välisistä yhteyksistä [9]. Solmua, joka on määrätty puun ensimmäiseksi solmuksi, kutsuutaan juurisolmuksi (*root node*) [9]. Kaikki juurisolmun yhteydet johtavat aina pois päin juurisolmusta [9]. Jos solmusta A on yhteys solmuun B, niin solmu A on solmun B isäntäsolmu (*parent node*) ja solmu B on solmun A tytärsolmu (*child node*). Kaikkia päätöspuun solmuja, joista on yhteys uuteen solmuun, kutsutaan sisäsolmuiksi, ja solmuja, joista ei ole yhteyttä eteenpäin, kutsutaan lehtisolmuiksi (*leaf node*) [9].



Kuva 4: Kuvassa A esitetty päätöspuun rakenne. Ylimpänä on juurisolmu, josta on yhteys sisäsolmujen kautta lehtisolmuihin. Jos solmun ehto täyttyy, siirrytään vasempaan tytärsolmuun, ja jos ehto ei täyty, siirrytään oikeaan tytärsolmuun. Päätöspuu luokittelee mielivaltaisen pisteen (x, y) yhteen lehtisolmujen luokista I–V. Kuvassa B vastaavat luokat esitetty koordinaatistossa.

Jokainen päätöspuun solmu jakaa siihen saapuvan syötejoukon erillisiksi osajoukoiksi [9]. Jokainen solmu toimii siis luokittelijana, joka jakaa saapuvat esimerkkialkiot yhden tai useamman piirteen perusteella erillisiin osajoukkoihin [9]. Nämä osajoukot syötetään eteenpäin niitä vastaaville tytärsolmuille [9].Tytärsolmut puolestaan osittavat nämä osajoukot ja tämä jatkuu kunnes saavutaan lehtisolmuun [9]. Lehtisolmuihin päätyneille esimerkkialkioille annetaan kutakin lehtisolmua vastaava leima ja näin syötejoukko on saatu luokiteltua [9].

3.2.2 Päätöspuun rakenteen optimointi

Muodostettaessa päätöspuuta, paras rakenne puulle on luonnollisesti se, joka luokittelee syötejoukon oikeisiin luokkiin [9]. Käytännössä tällaisen rakenteen löytäminen mielivaltaiselle syötejoukolle ei ole mahdollista, joten tavoitteena on löytää rakenne, joka luokittelee algoritmin opetukseen käytettävän opetusjoukon oikeisiin luokkiin mahdollisimman hyvin [9]. Kaikkien mahdollisten rakenteiden joukossa saattaa olla useampi rakenne, jotka luokittelevat opetusjoukon yhtä hyvin [9]. Täten on mielekästä etsiä rakenteista yksinkertaisin eli pienin mahdollinen päätöspuu, jolloin päätöspuun rakenne on helpompi tulkita [9]. Pienimmän mahdollisen päätöspuun etsiminen on kuitenkin laskennallisesti haastavaa, joten käytännössä on järkevintä käyttää jonkinlaista tehokasta heuristiikkaa, jonka avulla voidaan rakentaa lähes optimaalisia päätöspuita (*near optimal decision tree*) [9].

Selvitettäessä lähes optimaalisen päätöspuun rakennetta, avuksi voidaan määritellä jokin epäpuhtauden mitta (*impurity measure*) [9]. Se mittaa päätöspuun solmujen "hyvyyttä"[9], eli sitä kuinka hyvä tai tehokas solmun tekemä jako on luokittelun suhteen. Mitä pienempi epäpuhtaus solmulla on, sitä parempi solmu on kyseessä [9]. Täten lähes optimaalinen päätöspuu voidaan rakentaa lähtien juurisolmusta siten, että valitaan aina seuraavaksi solmuksi puhtain solmu [9]. Tätä iteraatiota jatketaan, kunnes lähtöjoukkoa ei voida enää osittaa, jolloin saadun puun solmuista ei voi enää tehdä puhtaampia [9]. Päätöspuun rakentamista tällä tavoin kutsutaan ahneeksi (*greedy*) menetelmäksi [9].

Yksittäistä päätöspuuta muodostettaessa heikkoutena on se, että päätöpuun lopullinen rakenne on hyvin riippuvainen opetusjoukosta, eli toisistaan poikkeavilla opetusjoukoilla muodostettujen päätöspuiden ennusteet voivat poiketa toisistaan huomattavasti [5]. Jos esimerkiksi käytössä oleva opetusjoukko jaetaan mielivaltaisesti kahtia ja kummallekin puolikkaalle rakennetaan optimaalisin päätöspuu, voi päätöspuiden rakenne poiketa toisistaan suuresti, jolloin ne myös todennäköisemmin tekevät yksittäiselle syötealkiolle eri ennusteet [5]. Tähän ongelmaan saadaan apua satunnainen metsä -menetelmistä, joissa yksittäisten päätöspuiden ongelmia pyritään poistamaan käyttämällä useita päätöspuita yhtä aikaa [5].

3.2.3 Satunnaisen metsän muodostaminen

Satunnainen metsä viittaa kokoelmaan menetelmiä, joissa luokittelu- tai regressioalgoritmi koostetaan useista eri päätöspuista, jotka on luotu hyödyntäen satunnaisuutta [9]. Eri menetelmät poikkeavat toisistaan pääosin siinä, miten satunnaisuutta hyödynnetään päätöspuita muodostettaessa [9]. Puiden satunnaisella rakentamisella pyritään vähentämään yksittäisten päätöspuiden harhaa (*bias*) ja varianssia [5, 9]. Satunnainen metsä toimii siten, että jokainen yksittäinen päätöspuu luokittelee syötetyn esimerkkialkion itsenäisesti ja eniten päätöspuiden ääniä saanut luokka valitaan satunnaisen metsän päätökseksi [5].

Satunnainen metsä voidaan rakentaa esimerkiksi siten, että jokaista puuta rakennettaessa valitaan satunnaisesti osajoukko lähtömateriaalin piirteistä ja jokainen puu muodostetaan vain valittujen piirteiden perusteella [5]. Tällöin yksittäiset piirteet eivät dominoi etsittäessä jokaisen päätöspuun optimaalista rakennetta, joten päätöspuiden rakenne vaihtelee ja puiden yhteinen tulos on luotettavampi [5]. On myös olemassa useita muita tapoja muodostaa satunnainen metsä [9].

3.3 Klusterointi

Klusterointimentelmät ovat ohjaamattomaan oppimiseen lukeutuvia menetelmiä [5]. Klusterointia käytetään siis tilanteessa, jossa esimerkkialkiolle $\bar{x}_i$ ei ole vastealkiota y_i [5]. Tällaisesta datasta ei ole tarkoitus tehdä ennusteita yksittäisille esimerkkialkioille $\bar{x}_i$, vaan etsiä esimerkiksi alaryhmiä ja visualisoida koko lähtömateriaalia $\{\bar{x}_i | i = 1, ..., m\}$ [5].

Klusterointi käsittää suuren määrän erilaisia tapoja löytää lähtömateriaalista alaryhmiä eli niin kutsuttuja klustereita [5]. Lähtömateriaali pyritään jakamaan erillisiin ryhmiin, joissa yksittäisen ryhmän alkiot ovat keskenään samanlaisia ja eri ryhmien alkiot ovat keskenään puolestaan erilaisia [5].

Eri klusterointimenetelmät voivat vaihdella niille annettavan syötteen ja ulostulon osalta [4]. Rajataan tarkastelu yleisesti käytettyyn asetelmaan, jossa syöte ja ulostulo määritellään seuraavasti [4]:

syöte Koostuu lähtömateriaalista $S = \{\bar{x}_i | i = 1, \ldots, m\}$ ja jostakin etäisyyttä mittaavasta funktiosta $d : S \times S \to \mathbb{R}_+$, jolle pätee kaikilla $a \neq b$, $d(\bar{x_a}, \bar{x_b}) = d(\bar{x_b}, \bar{x_a})$, ja kaikilla $a = 1, \ldots, m$, $d(\bar{x}_a, \bar{x}_a) = 0$.

ulostulo Ulostulona saadaan jokin lähtömateriaalin ositus $C = (C_1, \ldots, C_k)$, jolle $\bigcup_{i=1}^{k} C_i = S$ ja kaikilla $i \neq j$ $C_i \cap C_j = \emptyset$ eli jokainen lähtömateriaalin alkio kuuluu johonkin osajoukkoon ja osajoukot ovat erillisiä.

Joidenkin menetelmien tapauksessa syötteeksi annetaan lisäksi haluttujen klustereiden määrä [4]. Alla on esitetty kaksi esimerkkiä klusteroinnista: k:n keskiarvon klusterointimenetelmä (*k-means clustering*) ja sidospohjainen klusterointi (*linkage-based clustering*).

3.3.1 K:n keskiarvon klusterointimenetelmä

K:n keskiarvon klusterointimenetelmä jakaa lähtömateriaalin k kappaleeseen erillisiä klustereita [5]. Menetelmän periaatteena on, että hyvä klusteri on osajoukko, jonka alkioiden keskenäinen vaihtelu on mahdollisimman pientä [5]. Tätä varten täytyy määritellä klustereille $C_i, i = 1, \ldots, k$ sisäisen vaihtelun mitta $W(C_i)$, joka kertoo kuinka suurta klusterin alkioiden keskenäinen vaihtelu on [5]. Tämän avulla ongelma voidaan ilmasita matemaattisesti muodossa [5]

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{i=1}^{k} W(C_i) \right\}. \tag{7}$$

Toisin sanoen tavoite on jakaa lähtömateriaali k kappaleeseen osajoukkoja siten, että osajoukkojen sisäisten vaihteluiden summa on mahdollisimman pieni [5].

Sisäisen vaihtelun mitta voidaan määritellä monilla eri tavoilla [5]. Yleisesti käytetty määritelmä on neliöllinen euklidinen etäisyys (*squared euclidean distance*), joka määritellään siten, että

$$W(C_i) = \frac{1}{|C_i|} \sum_{\bar{x}_a, \bar{x}_b \in C_i} \sum_{j=1}^{p} (x_{aj} - x_{bj})^2, \tag{8}$$

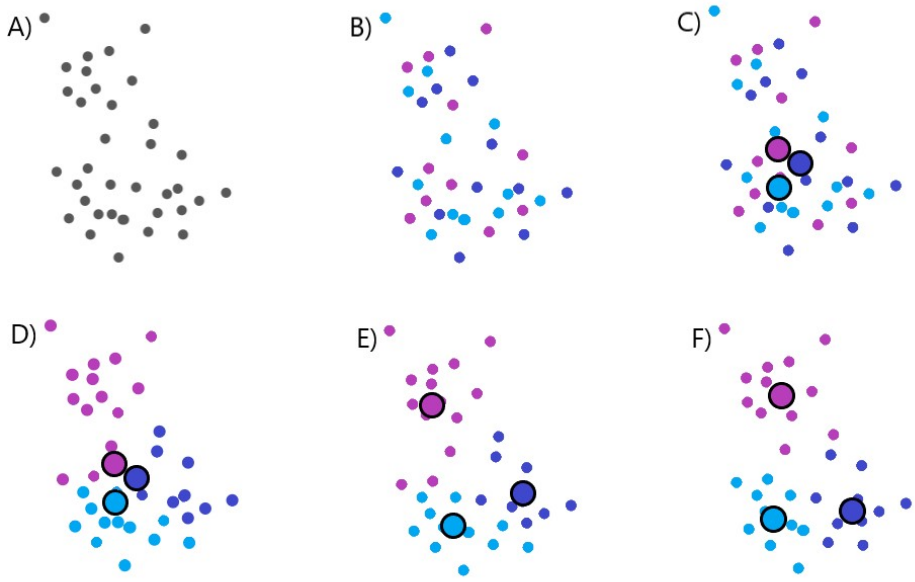
missä $|C_i|$ on esimerkkialkioiden määrä klusterissa C_i, p on piirteiden lukumäärä esimerkkialkiossa ja luvut x_{aj} ja x_{bj} ovat esimerkkialkioiden $\bar{x}_a$ ja $\bar{x}_b$ j:net koordinaattialkiot [5]. Sijoittamalla kaava (8) kaavaan (7) saadaan minimointiongelma muotoon

$$\min_{C_1,\ldots,C_k} \left\{ \sum_{i=1}^{k} \frac{1}{|C_i|} \sum_{\bar{x}_a,\bar{x}_b \in C_i} \sum_{j=1}^{p} (x_{aj} - x_{bj})^2 \right\}. \tag{9}$$

Kyseinen minimointiongelma on kuitenkin varsin haastava ratkaista, sillä m kappaleella esimerkkialkioita ja k kappaleella klustereita voidaan lähtömateriaali osittaa lähes k^m eri tavalla [5]. K:n keskiarvon klusterointimenetelmä pyrkii tarkan ratkaisun eli globaalin minimin sijaan löytämään lokaaliin minimin ongelmalle (9) [5]. Menetelmä toimii siten, että vaiheessa 1) jokaiselle esimerkkialkiolle asetetaan satunnaisesti jokin klusteri-indeksi 1:n ja k:n väliltä, missä k on haluttujen klustereiden lukumäärä [5]. Tämän jälkeen vaiheessa 2) suoritetaan seuraavaa iteraatiota:

a) Lasketaan painopiste (*centroid*) jokaiselle klusterille. Klusterin painopiste on jokaisen klusteriin lukeutuvan esimerkkialkion keskiarvo.

b) Jokaisen esimerkkialkion klusteri-indeksi asetetaan vastaamaan lähinnä olevan painopisteen klusteri-indeksiä.



Kuva 5: Havainnollistus k:n keskiarvon klusterointimenetelmästä klustereiden lukumäärällä $k = 3$. Kuvassa A on kuvattuna lähtömateriaali. Kuvassa B kuvattu vaihe 1, jossa esimerkkialkioille asetetaan satunnainen klusteri-indeksi. Kuvassa C kuvattu vaihe 2a, jossa satunnaisesti muodostetuille klustereille lasketaan painopisteet. Kuvassa D kuvattu vaihe 2b, jossa esimerkkialkioiden klusteri-indeksit on asetettu vastaamaan lähintä painopistettä. Kuvassa E on kuvattuna uuden iteraatiokierroksen vaihe 2a, jossa klustereille lasketaan uudet painopisteet. Kuvassa F on kuvattu iteraation lopullinen tulos.

Iteriaatiota jatketaan kunnes esimerkkialkioiden klusteri-indeksit eivät enää muutu, jolloin minimointiongelmaan (9) on saatu lokaali ratkaisu [5]. K:n keskiarvon klusterointimenetelmää on havainnollistettu kuvassa 5.

Koska menetelmä etsii lokaalin ratkaisun, joka riippuu vaiheen 1 satunnaisesta indeksivalinnasta, on tärkeää suorittaa algoritmi useaan otteeseen, jolloin tuloksista voidaan valita se, jolle asetetun ongelman (9) ratkaisu on pienin [5].

3.3.2 Sidospohjainen klusterointi

Sidospohjainen klusterointi (*linkage-based clustering*) on hierarkkisiin klusterointimenetelmiin lukeutuva klusterointimenetelmä [4, 5]. Sidospohjainen klusterointi etenee kierroksittain siten, että aluksi jokainen lähtömateriaalin piste on oma klusterinsa, ja jokaisella kierroksella yhdistetään toisiaan lähimpänä olevat klusterit yhdeksi klusteriksi [4]. Näin ollen klustereiden määrä vähenee yhdellä joka kierroksella, ja jos kierroksia jatketaan pysähtymättä, saadaan loplta yksi iso klusteri, joka pitää sisällään kaikki lähtömateriaalin pisteet [4]. Jotta menetelmää voidaan hyödyntää tehokkaasti, on asetettava kaksi määritelmää: kuinka mitataan kahden klusterin välinen etäisyys ja missä vaiheessa lopetetaan klustereiden yhdistäminen [4].

Kahden klusterin välinen etäisyys $D(A, B)$ voidaan määritellä monin eri tavoin [4]. Yleisimmät tavat etäisyyden määrittelyyn ovat yksittäissidosklusterointi (*single linkage clustering*), keskiarvosidosklusterointi (*average linkage clustering*) ja maksimisidosklusterointi (*max linkage clustering*) [4]. Olkoon $d : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_+$ jokin etäisyysmitta. Yksittäissidosklusteroinnissa kahden klusterin välinen etäisyys on määritelty pienimpänä mahdollisena etäisyytenä klusterien alkioiden välillä [4], eli

$$D(A, B) = \min\{d(\bar{x}, \bar{y}) : \bar{x} \in A, \bar{y} \in B\}. \tag{10}$$

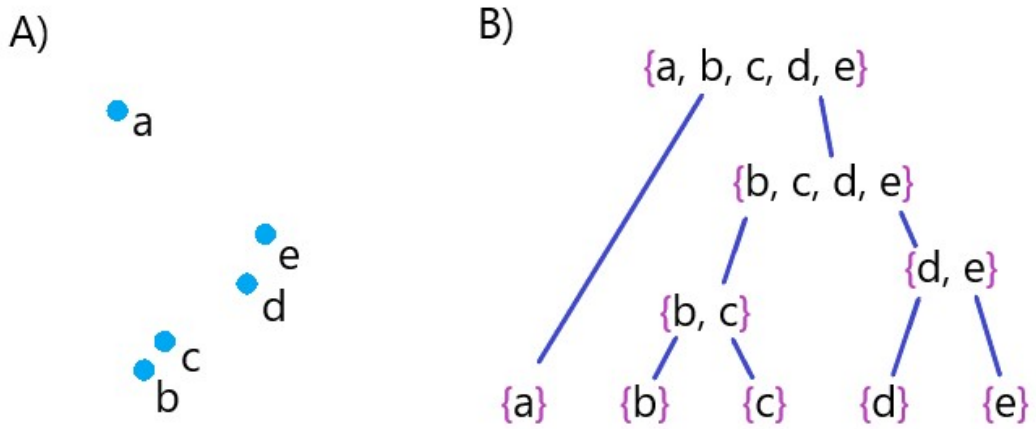
Keskiarvosidosklusteroinnissa kahden klusterin välinen etäisyys on määritelty kahden klusterin pisteiden etäisyyden keskiarvona [4], eli

$$D(A, B) = \frac{1}{|A||B|} \sum_{\bar{x} \in A} \sum_{\bar{y} \in B} d(\bar{x}, \bar{y}), \tag{11}$$

missä $|A|$ ja $|B|$ ovat alkioiden määrät klustereissa A ja B. Maksimisidosklusteroinninssa kahden klusterin välinen etäisyys on määritelty suurimpana mahdollisena etäisyytenä klusterien alkioiden välillä [4], eli

$$D(A, B) = \max\{d(\bar{x}, \bar{y}) : \bar{x} \in A, \bar{y} \in B\}. \tag{12}$$

Sidospohjaista klusterointia kutsutaan kokoavaksi (*agglomerative*) klusteroinniksi, sillä menetelmä aloittaa täysin yhdistelemättömästä datasta ja kokoaa niitä klustereiksi kierrosten edetessä [4]. Ilman katkaisusääntöä tämä voidaan kuvata klusterointipuulla (*dendogram*) [4]. Esimerkki klusterointipuusta on esitetty kuvassa 6. Mikäli menetelmällä halutaan muodostaa lähtömateriaalin ositus, on määriteltävä jokin katkaisusääntö [4]. Yleisesti käytetty katkaisusääntö on sama kuin K:n keskiarvon klusterointimenetelmässä, eli määritellään haluttu klusterien määrä k ja klustereiden yhdistäminen lopetetaan, kun k klusteria on saavutettu [4]. Toinen usein käytetty katkaisusääntö on etäisyyden yläraja [4]. Tällöin määritellään jokin luku $r \in \mathbb{R}_+$ ja klusterien yhdistäminen lopetetaan, kun klusterien välinen etäisyys on suurempaa kuin r [4].



Kuva 6: Kuvassa A klusteroitava lähtömateriaali ja kuvassa B sidospohjaisella klusteroinnilla muodostettu klusterointipuu.

4 Sovelluksia unilääketieteessä

4.1 Univaiheiden luokittelu

Henkilölle tehtävä unirekisteröinti on usein unipolygrafia (*polysomnography*), jossa henkilöstä mitataan useita kehon signaaleja monilla erilaisilla menetelmillä muutaman tunnin tai koko yön ajan [10]. Unipolygrafiassa mitattavat signaalit voivat hieman vaihdella tavoitteesta ja käytettävissä olevasta laitteistosta riippuen, mutta yleensä unipolygrafiamittauksessa rekisteröidään ainakin aivosähkökäyrä (*electroencephalography*), silmien liike (*electro-oculography*), leuan ja raajojen lihasten aktiivisuus (*electromyography*), sydänsähkökäyrä (*electrocardiography*), hengitysilmavirtaus, hengityksestä aiheutuvat pallean ja rintakehän liikkeet ja veren happisaturaatio [10].

Visuaalisessa univaiheiden luokittelussa unipolygrafiamittauksista saatu data jaetaan usein 30 sekunnin mittaisiin jaksoihin [11]. Jokaiselle näistä ajanjaksoista määritetään visuaalisesti jokin univaihe, ja mikäli samassa ajanjaksossa esiintyy useampaa unen vaihetta, valitaan se, kumpaa ajanjaksolla on enemmän [11]. Unelle määritellään nykyisen käytännön mukaan viisi eri vaihetta: hereilläolo W, NREM-unen (*nonrapid eye movement*) vaiheet N1, N2 ja N3 sekä REM (*rapid eye movement*) uni R [11]. NREM-unessa aivosähkökäyrässä muodostuu synkronoituja aaltoja ja silmien liike on vähäistä [11]. Vaiheessa N1 mitatun aivosähkökäyräsignaalin taajuss on 4–7 Hz, vaiheessa N2 taajuus pysyy samana, mutta lisäksi ilmenee K-komplekseja (*K complex*) ja unisukkuloita (*sleep spindle*), ja vaiheessa N3 taajuus putoaa 0,5–2 Hz [11]. REM-unessa puolestaan aivosähkököyrässä ei ole havaittavissa vastaava aaltoilua [11]. Sen sijaan silmien liike eri suuntiin on runsasta ja leuan lihasten aktiivisuus on hyvin alhaista [11]. Vanhemmassa luokittelumallissa vaihe N3 oli jaettu vielä kahteen erilliseen vaiheeseen, minkä johdosta osassa tutkimuksista uni jaetaan kuuteen vaiheeseen [11]. Univaiheet etenevät unisykleissä edellä mainitussa järjestyksessä alkaen vaiheesta N1 ja päättyen vaiheeseen R [11]. Yhden unisyklin pituus on keksimäärin 90 – 110 minuuttia, johon sisältyy kutakin vaihetta noin 5 – 15 minuuttia [12].

4.1.1 Ohjattu oppiminen

Klok ja Edin ryhmineen [13] muodostivat satunnaisen metsän, joka luokittelee unen hereilläoloon, NREM-uneen ja REM-uneen käyttäen aivosähkökäyrää ja silmien liikettä. Tavoitteena heillä oli luoda erityisesti luokittelija, joka pystyy luokittelemaan REM-unen aikana ilmenevää behavioraalista unioireyhtymää (*REM sleep behaviour disorder*) sairastavan potilaan unen edellä mainittuihin kolmeen luokkaan [13]. Lähtömateriaalina he käyttivät unipolygrafiamittauksia 100 koehenkilöstä, joilla tiedetään olevan erilaisia unihäiriöitä [13]. Kaksi asijantuntijaa luokittelivat koehenkilöiden unipolygrafiamittaukset hereilläoloon, NREM-uneen ja REM-uneen [13]. Satunnaisen metsän harjoittamista varten dataa esikäsiteltiin ja siitä poimittiin 247 piirrettä [13]. Satunnaisen metsän päätöspuiden määräksi asetettiin 200 [13]. Algoritmin tarkkuutta mitattiin 20-kertaisella ristiinvalidoinnilla ja opetetun algoritmin univaiheluokittelu vastasi $92,6\% \pm 6.4\%$ tarkkuudella asiantuntijoiden manualisesti tekemää analyysiä [13].

Rahman ym. [14] opettivat satunnaisen metsän sekä tukivektorikoneen luokittelemaan unen 2 – 6 luokkaan käyttäen silmien liikeestä mitattua signaalia [14]. Heidän lähtömateriaalinsa oli koostettu kolmesta eri tietokannasta ja koostui yhteensä 38 koehenkilön unipolygrafiamittauksista [14]. Algoritmit opetettiin kullekin tietokannalle erikseen [14]. Datan esikäsittelyn ja piirteiden valikoinnin pohjalta valittiin 30 piirrettä, joita käytettiin algoritmien opettamiseen [14]. Tukivektorikonetta opettaessa puolet lähtömateriaalista valittiin opetusjoukoksi toisen puolikkaan toimiessa testijoukkona [14]. Satunnaista metsää muodostaessa tarkkuuden arvioimiseen puolestaan käytettiin satunnaiselle metsälle ominaista ulkona-bootstrap-otoksesta -validointia (*Out-of-Bag validation*) [5, 14]. Tukivektorikoneella luokittelussa päästiin parhaimmillaan $91,7\% \pm 1,2\%$ tarkkuuteen [14]. Paras luokittelutarkkuus satunnaiselle metsälle oli puolestaan $91,0\% \pm 1,7\%$ [14].

4.1.2 Ohjaamaton oppiminen

Frilot ym. [15] tutkivat unessa ilmeneviä fysiologisia tiloja käyttäen klusterointia. Tavoitteena oli selvittää, voidaanko unesta löytää erillisiä vaiheita käyttämättä ihmisen määräämiä luokittelusääntöjä [15]. Lähtömateriaalina oli 149 koehenkilön unipolygrafiamittaukset [15]. Tuktimuksessa käytettiin luokitteluun aivosähkökäyrää, jolle suoritettiin toistuvuusanalyysi (*recurrence analysis*) [15]. Toistuvuusanalyysillä muodostettiin aivosähkökäyrästä jokaiselle sekunnille neljä toistuvuusmuuttujaa (*recurrence variable*), joista muodostettiin neliulotteinen vektori kuvaamaan aivosähkökäyrän käyttäytymistä kyseisellä ajanhetkellä [15]. Tämän jälkeen aivosähkökäyräsignaaleista muodostettiin 30 sekunnin ajanjaksoja ja jokaiselle ajanjaksolle laskettiin toistuvuusmuuttujavektori keskiarvona yksittäisistä jaksoon lukeutuvista vektoreista [15]. Tämän jälkeen 30 sekunnin ajanjaksojen vektoreille suoritettiin keskiarvosidosklusterointi, jossa klustereiden määräksi asetettiin neljä [15]. Toistuvuusmuuttujien arvojen vaihtelu muodostettujen klusterien välillä oli selkeää, minkä pääteltiin viittavan siihen, että kyseiset neljä klusteria ovat hyvin määriteltyjä fysiologisia unen tiloja [15].

4.2 Uniapnea

Uniapneasyndrooma (*sleep apnea syndrome*) on sairaus, jossa unen aikana ylähengitystiet väliaikaisesti tukkeutuvat ja hengitysilma ei pääse kulkeutumaan keuhkoihin

[16]. Unen aikaisia, yli 10 sekunnin mittaisia hengityskatkoksia kutsutaan apneoiksi, ja ne voidaan jakaa kolmeen tyyppiin: obstruktiiviseen apneaan (*obstructive apnea*), sentraaliseen apneaan (*central apnea*) ja sekamuotoiseen apneaan (*mixed apnea*) [16]. Näistä yleisin on obstruktiivinen apnea, jossa potilaan rintakehän ja vatsan lihakset ovat aktiivisia ja pyrkivät hengittämään apnean aikana [16]. Sentraalisessa apneassa puolestaan apnean ilmaantuessa hengityslihakset eivät ole aktiivisia [16]. Sekamuotoisessa apneassa yhdistyy molemmat edellä mainituista, eli apnean alkuvaiheessa on sentraalisen apnean piirteitä, jonka jälkeen hengityslihakset aktivoituvat kuten obstuktiivisessa apneassa [16]. Kun yli puolet apneoista ovat obstruktiivisia, puhutaan obstruktiivisesta uniapneasta ja vastaavasti kun yli puolet apneoista ovat sentraalisia, puhutaan sentraalisesta uniapneasta [10]. Apneoiden lisäksi potilaalla voi ilmetä hypopneoita, eli hengitysilmavirtauksen heikentymistä ylähengitysteiden osittaisen tukkeutumisen seurauksena [16]. Hypopnean seurauksena veren happikyllästeisyys laskee tai potilas havahtuu unesta [10]. Yleinen tapa arvioida uniapnean vakavuutta on määrittää apnea-hypopneaindeksi (AHI), joka kertoo keskimääräisen apneoiden ja hypopneoiden määrän nukuttua tuntia kohti [16]. AHI:n ollessa 5–15 on kyseessä lievä uniapnea, AHI:n ollessa 15–30 kyseessä on keskivaikea uniapnea ja AHI:n ollessa yli 30 kyseessä on vaikea uniapnea [16]

4.2.1 Uniapnean vakavuuden arviointi

Avici ja Akbas [17] muodostivat tutkimuksessaan satunnaisen metsän uniapnean vakavuusasteen luokitteluun. Algrotimi käytti luokitteluun nenästä mitattua hengitysilmavirtaa ja vatsasta ja rinnasta mitattuja hengitysliikkeen signaaleja [17]. Lähtömateriaalina heillä oli 8 koehenkilön unipolygrafiamittaukset [17]. Lähtömateriaalissa mitatut signaalit oli jaettu minuutin mittaisiin ajanjaksoihin, jotka asiantuntija oli luokitellut joko leimalla apnea tai leimalla ei apneaa [17]. Lähtömateriaalista poimittiin esikäsittelyllä 12 piirrettä, joita käytettiin satunnaisen metsän opettamiseen ja opetetun algoritmin tarkkuus testattiin 10-kertaisella ristiinvalidoinnilla [17]. Paras tarkkuus saatiin nenästä mitatulla hengityssignaalilla, jolloin satunnainen metsä luokitteli oikein 98,68% minuutin mittaisista ajanjaksoista [17].

Koley ja Dey [18] pyrkivät määrittämään koehenkilöiden apnea- ja hypopneatapaukset unen aikana sormen päästä mitattavasta happisaturaatiosignaalista hyödyntäen tukivektorikonetta. Lähtömateriaalina he käyttivät 26 koehenkilön unen aikana mitattuja happisaturaatiosignaaleja [18]. Lähtömateriaalia esikäsiteltiin ja siitä valittiin 34 piirrettä, joita käytettiin tukivektorikoneen opettamiseen [18]. Lähtömateriaalista 16 satunnaisesti valitun koehenkilön mittaukset toimivat opetusjoukkona muiden 10 koehenkilön koostaessa testijoukon [18]. Opetettu tukivektorikone onnistui havaitsemaan apnea- ja hypopneatapaukset 96,7% tarkkuudella.

4.2.2 Kuorsauksen yhteys uniapneaan

Alshaer ja muut [19] pyrkivät selvittämään yhteyttä uniapnean ja kuorsauksen välillä hyödyntäen satunnaista metsää kuorsausäänien havaitsemiseen ja luokitteluun. Lähtömateriaalina satunnaisen metsän opettamisessa he käyttivät 11 koehenkilön yhden yön kuorsausääniä, jotka asiantuntija oli luokitellut joko kuorsaukseksi tai muuksi ääneksi [20]. Kuorsausäänistä poimittiin 10 piirrettä, joiden perusteella satunnainen metsä opetettiin erottelemaan kuorsausäänet muista äänistä [20]. Satunnaisen

metsän luokittelutarkkuutta mitattiin yksi-pois -ristiinvalidoinnilla ja tulokseksi saatiin 97,8% [20].

Opetettua algoritmia sovellettiin 235 koehenkilön joukkoon, joilta oli mitattu unipolygrafian ohella kuorsausäänet [19]. Asijantuntija määritti unipolygrafian perusteella jokaiselle koehenkilölle apnea-hypopneaindeksin ja satunnainen metsä määritti jokaiselle koehenkilölle kuorsausindeksin, joka kertoo kuorsausäänien määrän tuntia kohden [19]. Korrelaatio apnea-hypopneaindeksin ja kuorsausindeksin välillä oli heikko, korrelaatiokertoimena $r = 0,32$ [19]. Obstruktiivista uniapneaa sariastavilla koehenkilöillä korrelaatio oli hieman voimakkaampaa kertoimella $r = 0,40$, mutta sentraalista uniapneaa sairastavilla koehenkilöillä korrelaatio oli hyvin heikkoa ja negatiivista kertoimella $r = -0,14$.

Tutkimuksen perusteella havaittiin että kuorsaamisen määrä on heikko indikaattori obstruktiivisen uniapnean vakavuudelle, vaikka kuorsaaminen on yksi yleisimmistä uniapnean oireista [19]. Lisäksi obstruktiivista uniapneaa sairastavien ja terveiden koehenkilöiden kuorsausindekseissä oli paljon päällekkäisyyttä [19].

4.2.3 Klusterointi uniapneassa

Joosten ym. [21] pyrkivät havaitsemaan erilaisia uniapnean ilmenemismuotoja eli fenotyyppejä käyttäen apuna klusterointia. Hypoteesina he olettivat, että lievää ja keskivaikeaa obstruktiivista uniapneaa sairastavien henkilöiden joukosta voidaan havaita erillisä uniapnean fenotyyppejä [21]. Lähtömateriaalina he käyttivät unipolygrafiamittauksia, väestötietoja ja kehonkoostumusmittauksia 1064 koehenkilöstä, jotka sairastivat obstruktiivista uniapneaa [21].

Koehenkilöt jaettiin tietojensa perusteella neljään, osittain päällekkäisiin pääryhmiin: pääosin selinmakuulla ilmenevä uniapnea, pääosin REM-unen aikana ilmenevä uniapnea, pääosin NREM-unen aikana ilmenevä uniapnea sekä ajoittainen uniapnea [21]. Lisäksi määriteltiin kaksi alaryhmää: vain selinmakuulla ilmenevä uniapnea sekä vain REM-unen aikana ilmenevä uniapnea [21]. Tämän jälkeen lähtömateriaalille suoritettiin K:n keskiarvon klusterointi, jossa klustereiden määräksi asetettiin kuusi [21]. Klusteroimalla muodostettuja ryhmiä verrattiin tutkijoiden määrittämiin ryhmiin ja ryhmiinjakojen välillä havaittiin selkeä yhtäläisyys, eli klusteroinnista saatiin tukea tutkijoiden määrittelemien ryhmien olemassaololle [21].

Määritettyjen ryhmien välillä oli selkeitä eroja koehenkilöiden ominaispiirteissä ja unipolygrafiaparametreissa [21]. Suoritetusta klusterianalyysistä saatiin tukea hypoteesille, jonka mukaan lievää tai keskivaikeaa uniapneaa sairastavilla henkilöillä voidaan havaita toisistaan eriäviä uniapnean fenotyyppejä [21].

4.3 Muita sovelluksia

4.3.1 Nukkuma-asennon tunnistaminen

Pan ym. [22] muodostivat satunnaisen metsän, joka tunnistaa henkilön nukkuma-asennon käyttäen sydänsähkökäyrää. Lähtömateriaali koostui 9 koehenkilön unipolygrafiamittauksista, joissa sydänsähkökäyräsignaali oli jaettu 30 sekunnin ajanjaksoihin [22]. Lähtömateriaalia esikäsiteltiin ja siitä valittiin 12 piirrettä [22]. Satunnaisen metsän puiden määräksi asetettiin 500 [22]. Vatsallaan nukkuminen oli hyvin vähäistä koehenkilöiden joukossa, joten ajanjaksot, joilla koehenkilöt nukkuivat vatsallaan, poistettiin lähtömateriaalista manuaalisesti ja tavoitteeksi otettiin luokitella

nukkuma-asento kolmeen luokkaan: selällään, vasemmalla kyljellä ja oikealla kyljellä [22].

Tutkimuksessa tarkasteltiin kolmea eri järjestelyä [22]. Ensimmäinen näistä oli koehenkilökohtainen järjestely, jossa jokaiselle koehenkilölle opetettiin oma luokittelijansa käyttäen 20% koehenkilön 30 sekunnin ajanjaksoista opetusjoukkona ja loppuja 80% ajanjaksoista testijoukkona [22]. Toisena oli koehenkilöstä riippumaton järjestely, jossa opetettiin kaikille koehenkilöille yhteinen luokittelija käyttäen leave-one-out-ristiinvalidointia [22]. Kolmantena oli koehenkilöstä riippumaton järjestely normalisoiduilla piirteillä, joka vastasi koehenkilöstä riippumatonta järjestelyä, mutta ennen luokittelua koehenkilöiden piirteet normalisoitiin [22].

Nukkuma-asennon luokittelutarkkuuksiksi saatiin koehenkilökohtaisella järjestelyllä $97,17\% \pm 2,74\%$, koehenkilöstä riippumattomalla järjestelyllä $44.73\% \pm 31,61\%$ ja koehenkilöstä riippumattomalla normalisoitujen piirteiden järjestelyllä $63,87\% \pm 16,32\%$.

4.3.2 Behavioraalisen unioireyhtymän havaitseminen

Cooray ym. [23] opettivat satunnaisen metsän havaitsemaan REM-unen aikana ilmenevän behavioraalisen unioireyhtymän käyttäen aivosähkökäyrää, silmien liikettä ja leuan lihasten aktiivisuutta. Behavioraalista unioireyhtymää sairastavan henkilön lihakset pysyvät normaalista poiketen aktiivisina REM-unen aikana ja potilas liikkuu nähdessään unia [23]. Behavioraalisen unioireyhtymän on osoitettu olevan muun muassa ennakoiva merkki myöhemmin kehittyvästä Parkinsonin taudista [23].

Lähtömateriaalina tutkimuksessa oli 55 behavioraalista unioireyhtymää sairastavan koehenkilön unipolygrafiamittaukset ja 55 terveen koehenkilön unipolygrafiamittaukset [23]. Unipolygrafiamittauksista oli luokiteltu univaiheet asiantuntijan toimesta [23]. Lähtömateriaalista poimittiin 10 piirrettä ja satunnaisen metsän puiden määräksi asetettiin 500 [23]. Opetettu satunnainen metsä onnistui havaitsemaan behavioraalista unioireyhtymää sairastavat koehenkilöt 96% tarkkuudella [23].

5 Pohdinta

Unilääketieteessä koneoppimiselle löytyy monia sovelluksia analysointiprosessien mekaanisista vaiheista. Koneoppimista on sovellettu laajasti erityisesti univaiheiden luokitteluun ja uniapnean vakavuuden arviointiin. Näissä saadut tarkkuudet ovat varsin hyviä ottaen huomioon sen, että eri ihmisten tekemissä luokitteluissa voi olla suuriakin eroja. Esimerkiksi univaiheiden luokittelussa eri ihmisten luokittelut vastaavat toisiaan noin 80%:sesti [2]. Univaiheiden ja uniapnean lisäksi koneoppimista on sovellettu lukuisiin muihin mekaanisiin prosesseihin. Esimerkiksi nukkuma-asennon luokittelu on ihmiselle helppoa esimerkiksi videon perusteella, mutta koko yön mittaisen videoinnin läpikäynti vaatii ihmiseltä pitkäveteistä manuaalista työtä. Sen sijaan koneoppimisalgoritmi pystyy tekemään luokittelun huomattavasti nopeammin, eikä vaadi juurikaan ihmisen työpanosta. Myös unisairauksien havaitseminen koneoppimisen keinoin vähentää huomattavasti ihmisen tarvetta mitatun datan läpikäynnissä. Näissä sovelluksissa voidaan koneoppimisen avulla säästää aikaa ja keskittää ihmisresursseja vaativimpiin tehtäviin.

Ohjaamaton oppiminen voi tuoda uutta näkökulmaa myös ennestään tutkittuihin asioihin. Hyvänä esimerkkinä toimii univaiheiden luokittelu, jossa luokittelussa käytettävät säännöt ovat tiedeyhteisön kehittämiä ja jokainen luokittelu riippuu luokittelun tehneestä ihmisestä. Ihmisriippuvuuden sekä tarkkojen luokittelusääntöjen vaikutuksesta voidaan päästää eroon käyttäen klusterointia. Klusteroinnin hyöty korostuu siinä, ettei algoritmille tarvitse syöttää malliksi ihmisen luokittelemaa dataa tai luokittelusääntöjä, vaan algoritmi itse muodostaa yhteyksiä havaintojen välille. Tässä tapauksessa tarkkojen luokittelusääntöjen sijasta korostuu oikeiden piirteiden löytäminen käytettävistä signaaleista, jotta unen eri vaiheet olisivat mahdollisimman selkeästi erotettavissa toisistaan. Klusteroimalla muodostettu luokittelija voi olla joustavampi esimerkiksi unisairauksista ja -häiriöistä johtuville yksilöllisille vaihteluille, joita sääntöihin perustuvassa luokittelussa on vaikeampi huomioida.

Koneoppimista voidaan hyödyntää myös uuden tutkimisessa. Tästä esimerkkinä kuorsauksen yhteys uniapnean vakavuuteen ja uniapnean fenotyyppien tutkiminen. Näiden asioiden tutkiminen vaatisi ihmiseltä paljon perehtymistä sekä datan läpikäyntiä, mutta koneoppimislagoritmilla näitä ilmiöitä voidaan tarkastella huomattavasti vähemmällä vaivalla. Uuden tiedon avulla voidaan esimerkiksi tehdä tarkempia diagnooseja ja kehittää yksilöllisempia hoitoja unisairauksista kärsiville potilaille.

Koneoppimisen kanssa tulee kuitenkin varoa, ettei koneelle anneta analysoinnissa liikaa vastuuta tai koneelta saataviin tuloksiin luoteta liikaa. Turvallisin paikka koneoppimisen hyödyntämiseen on hyvin tunnetut ja mekaaniset vaiheet analysointiprosessista. Tällöin koneoppimisalgoritmi voidaan opettaa perustuen laajaan tutkimysnäyttöön, ja ihmisen on helpompi valvoa, kuinka hyvin algoritmi tehtävästään suoriutuu. Uutta tutkittaessa koneoppimisella saataviin tuloksiin tulee suhtautua varauksella ja enemmän suuntaa antavina tuloksina kuin tarkkoina arvioina. Koneoppimislagoritmit sisältävät paljon parametreja, joita algoritmia muodostavan ihmisen tulee määrittää, eli lopputulos on aina riippuvainen myös algoritmin luoneesta ihmisestä. Lisäksi käytettävän datan esikäsittely ja käytettävien piirteiden valinta voivat vaikuttaa huomattavasti lopputulokseen. Usein tutkimuksissa lähtömateriaali koostuu varsin pienestä määrästä ihmisiä, joille unirekisteröinti on suoritettu identtisellä laitteistolla. Näin ollen tutkimuksissa saadut tarkkuudet voivat muuttua suuremmalla ihmisjoukolla ja erilaisilla mittauslaitteistoilla. Tutkimuksissa käytetyt sovellukset

vaativat kattavaa kliinistä validointia ennen laajempaa käyttöönottoa. Ihmistä koneoppiminen ei korvaa, mutta se voi toimia erittäin tehokkaana työkaluna.

Tulevaisuuden haasteena koneoppimisella on sen soveltaminen kliinisessä potilastutkimuksessa. Näyttöä koneoppimisen toimivuudesta ja tehosta on paljon, mutta käytännön potilastyössä koneoppimisen hyödyntäminen on vähäistä. Monet nykyisistä unilääketieteen säännöistä ja menetelmistä perustuvat vuosikymmeniä sitten laadittuihin ohjeisiin ja tulosten analysointi on usein, ongelmallisuudestaan huolimatta, ihmisen manuaalisesti tekemää. Vanhoihin tapoihin juurtunut tiedeyhteisö on vaikea saada muuttamaan menetelmiään, vaikka näyttöä paremmista, koneoppimista hyödyntävistä, menetelmistä on runsaasti.

Viitteet

[1] Penzel T ja Conradt R, *Computer based sleep recording and analysis*, Sleep Medicine Reviews, 4(2):131–148, 2000.

[2] Rosenberg RS ja Van Hout S, *The American Academy of Sleep Medicine Inter-Scorer Reliability Program: Sleep Stage Scoring*, Journal of Clinical Sleep Medicine, 9(1):81-87, 2013.

[3] Mohri M, Rostamizadeh A ja Talwalkar A, *Foundations of Machine Learning*, The MIT Press, 2012.

[4] Shalev-Shwartz S ja Ben-David S, *Understanding Machine Learning: From Theory to Algorithms*, Cambridge University Press, 2014.

[5] James G, Witten D, Hastie T ja Tibshirani R, *An Introduction to Statistical Learning*, Springer Science+Business Media New York, 2013.

[6] Sammut C ja Webb G, *Encyclopedia of Machine Learning*, Springer Science+Business Media New York, 2011.

[7] Steinwart I ja Cristmann A, *Support Vector Machines*, Springer-Verlag New York, 2008.

[8] Breiman L, *Random Forests*, Machine Learning, 45:5–32, 2001.

[9] Louppe G, *Understanding Random Forests: From Theory to Practice*, Väitöskirja, Department of Electrical Engineering & Computer Science, University of Liège, 2014.

[10] Kushida C, *Encyclopedia of Sleep*, Elsevier, 2013.

[11] Conrad I, Ancoli-Israel S, Chesson AL ja Quan SF, *The AASM Manual for the Scoring of Sleep and Associated Events: Rules, Terminology and Technical Specifications*, 1. painos, American Academy of Sleep Medicine, 2007.

[12] Fiorillo L, Puiatti A, Papandrea M, Ratti P, Favaro P, Roth C, Bargiotas P, Bassetti C ja Faraci F, *Automated sleep scoring: A review of the latest approaches*, Sleep Medicine Reviews, 48, 2019.

[13] Klok A, Edin J, Cesari M, Olesen A, Jennum P ja Sorensen H, *A new fully automated random-forest algorithm for sleep staging*, Conference Proceedings: IEEE Engineering in Medicine and Biology Society, Annual Conference 2018, 4920–4923, 2018.

[14] Rahman M, Bhuian M ja Hassan A, *Sleep stage classification using single-channel EOG*, Computers in Biology and Medicine, 102:211–220, 2018.

[15] Frilot C, McCarty D ja Marino A, *An original method for staging sleep vased on dunamical analysis of a single EEG signal*, Journal of Neuroscience Methods, 308:135–141, 2018.

[16] Pombo N, Garcia N ja Bousson K, *Classification techniques on computerized systems to predict and/or to detect Apnea : A systematic review*, Computer Methods and Programs in Biomedicine, 140:265-274, 2017.

[17] Avci C ja Akbas A, *Sleep apnea classification based on respiration signals by using ensemble methods*, Bio-Medical Materials and Engineering, 26:S1703-S1710, 2015.

[18] Koley B ja Dey D, *On-Line Detection of Apnea/Hypopnea Events Using SpO2 Signal: A Rule-Based Approach Employing Binary Classifier Models*, IEEE Journal of Biomedical and Health Informatics, 18:231-239, 2014.

[19] Alshaer H, Hummel R, Mendelson M, Marshal T ja Bradley D, *Objective relationship between sleep apnea and frequency of snoring assessed by machine learning*, Journal of Clinical Sleep Medicine, 15:463-470, 2019.

[20] Alshaer H, Pandya A, Bradley D ja Rudzicz F, *Subject independent identification of breath sounds components using multiple classifiers*, Conference Paper in Proceedings - ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing, 2014.

[21] Joosten S, Hamza K, Sands S, Turton A, Berger P ja Hamilton G, *Phenotypes of patients with mild to moderate obstructive sleep apnoea as confirmed by cluster analysis*, Respilogy, 17:99-107, 2012.

[22] Pan H, Xu Z, Yan H, Gao Y, Chen Z, Song J ja Zhang Y, *Lying position classification based on ECG waveform and random forest during sleep in healthy people*, BioMedical Engineering OnLine, 17:116, 2018.

[23] Cooray N, Andreotti F, Lo C, Symmonds M, Hu M ja De Vos M, *Detection of REM sleep behaviour disorder by automated polysomnography analysis*, Clinical neurophysiology, 130:505-514, 2019.