Visa Nykänen

Päätöspuiden käyttö koneoppimisessa

Tietotekniikan kandidaatintutkielma

18. joulukuuta 2017

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Visa Nykänen

Yhteystiedot: Visa.K.Nykanen@student.jyu.fi

Työn nimi: Päätöspuiden käyttö koneoppimisessa

Title in English: The use of decision trees in machine learning

Työ: Kandidaatintutkielma

Sivumäärä: 28+0

Tiivistelmä: Päätöspuihin perustuvia menetelmiä ollaan käytetty laajasti koneoppimisen eri tehtävissä. Tämä tutkielma kokoaa tietoa siitä miten ja mihin näitä menetelmiä ollaan käytetty. Käydään läpi yksittäisen päätöspuun luominen datasta ja tämän pohjalta tutustutaan kokoonpanomenetelmiin, joissa kootaan useita päätöspuita. Lopulta tutustutaan vielä joihinkin esiteltyjen menetelmien sovelluksiin.

Avainsanat: Koneoppiminen, Päätöspuu, Kokoonpanomenetelmä, Satunnaismetsä, Bootsrap-aggregointi, Tehostaminen

Abstract: Methods based on decision trees have been utilized widely in different tasks of machine learning. This study compiles information on how and for what purposes have these methods been used. Construction of a decision tree based on data is explained and with that as a basis ensemble methods, which comprise many decision trees, are introduced. Finally some applications of the introduced methods are discussed.

Keywords: Machine Learning, Decision Tree, Ensemble Method, Random Forest, Bagging, Boosting

Kuviot

Sisältö

1 Johdanto

Päätöspuut ovat yleisesti tapa kuvata päätöksentekoa puurakenteen avulla siten, että kukin oksa kuvaa yhtä havaintoa ja kukin lehtisolmu kuvaa lopputulosta. Koneoppimisessa Maimon, O. & Rokach, L. (2008, luku 1) mukaan päätöspuu toimii ennakoivana mallina, joka luodaan käytössä olevan opetusdatan pohjalta. Tässä tapauksessa oksat kuvaavat datapisteestä tehtyjä havaintoja ja lehtisolmut johtopäätöksiä datapisteen siitä muuttujasta, jota päätöspuun avulla halutaan ennakoida eli kohdemuuttujasta. Kohdemuuttujan ollessa diskreetti sanotaan päätöspuun olevan luokittelupuu. Jatkuvan kohdemuuttujan tapauksessa kyseessä on taas regressiopuu. Eräs mielenkiintoinen päätöspuiden ominaisuus onkin se, että niitä voidaan soveltaa yleisimmistä ohjatun oppimisen tehtävistä molempiin: regressioon sekä luokitteluun.

Tässä tutkielmassa tutkitaan kirjallisuuden avulla päätöspuiden käyttötapoja ja -tarkoituksia koneoppimisen sovelluksissa. Esitetään päätöspuihin liittyviä menetelmiä, selvittäen samalla minkälaisiin ongelmiin niitä on kehitetty vastaamaan. Tarkoituksena on antaa lukijalle selkeä yleiskatsaus siitä miten ja mihin päätöspuihin perustuvia menetelmiä tulisi koneoppimisessa käyttää. Tavoite olisi, että lukija tämän tutkielman sekä lähdekirjallisuuden avulla kykenisi hyödyntämään esiteltyjä menetelmiä. Käytännöllisen suuntautumisen nojalla kiinnitetään huomiota erityisesti menetelmiin, jotka ovat osoittautuneet suosituiksi käytännössä.

Luvussa 2 selvitetään päätöspuun johtamista datan pohjalta käymällä yleisesti läpi siihen liittyvät johtamisen ja karsimisen menetelmät; esitellään tarkemmin myös muutama päätöspuun luomiseen tarkoitettu algoritmi. Luvussa 3 taas kartoitetaan koneoppimisen menetelmiä, jotka toteutuksissaan perustuvat useiden päätöspuiden käyttöön. Lopulta luvussa 4 selvitetään minkälaisissa sovelluksissa päätöspuihin perustuvia menetelmiä ollaan käytetty.

2 Päätöspuun luominen datasta

Päätöspuun luominen datan pohjalta jakautuu yleensä kahteen vaiheeseen: kasvattaminen ja karsiminen. Ylhäältä alaspäin kasvattaessa valitaan datasta jollain kriteerillä parhaaksi arvioitu ominaisuus ja jaetaan data osiin tämän ominaisuuden arvojen perusteella. Sama prosessi toistetaan rekursiivisesti kullekin jaossa syntyneelle datan osalle, kunnes näin muodostuva puu toteuttaa valitun pysäytyskriteerin. Myös alhaalta ylöspäin toteutettavia kasvatusmenetelmiä ollaan ehdotettu (esim Barros, R. C. & Jaskowiak, A. J. & Cerri, R. et al. (2014)), mutta ne ovat jääneet kirjallisuudessa huomattavasti vähemmälle huomiolle. Karsimisvaihe toteutetaan muuttamalla kasvatetusta puusta alipuita lehtisolmuiksi, mikäli vähentämättä mallin tarkkuutta näin saadaan paranneltua puuta jollain kriteerillä. Tässä luvussa paneudutaan tarkemmin edellä esitettyihin käsitteisiin. Käydään läpi muutama tunnettu päätöspuiden johtamiseen tarkoitettu algoritmi pohjustaen niitä niiden käyttämillä kasvatus- ja karsimisvaiheen menetelmillä. Näistä menetelmistä poikkeavia menetelmiä kokoaa esimerkiksi Maimon, O. & Rokach, L. (2002). Merkittävä havainto esiteltävistä menetelmistä on se, että ne ovat kaikki ahneita algoritmeja, joissa valitaan kussakin vaiheessa jollain kriteerillä paras etenemistapa. Tämä johtuu siitä, että optimaalisen puun löytämisen ollaan osoitettu olevan NP-täydellinen ongelma (Hyafil, L. & Rivest, R. L. 1976).

2.1 Kasvatusvaihe

2.1.1 Jakokriteerit

Päätöspuun jakokriteereitä käytetään valitsemaan se ominaisuus, jonka perusteella kussakin solmussa data jaetaan. Jakokriteerit voidaan karkeasti luokitella informaatioteoriaan, etäisyysmittoihin tai riippuvuusmittoihin perustuviksi (Ben-Bassat, M. 1982). Tässä tutkielmassa kiinnitetään erityisesti huomiota informaatioteoreettisiin jakokriteereihin. Informaatioteoriaan perustuvissa kriteereissä tarvitaan epäpuhtausmittaa (engl. *impurity measure*). Epäpuhtausmitta on funktio, joka laskee tar-

kasteltavan kohdemuuttujan todennäköisyysjakauman perusteella sen, kuinka selkeästi kohdemuuttujan luokat ovat erillään toisistaan. Epäpuhtausmitan tulee olla minimissään silloin, kun muuttujan todennäköisyysjakaumassa jokin todennäköisyys saa arvon yksi ja maksimissaan, kun todennäköisyydet ovat tasajakaumasta. Lisäksi epäpuhtausmitan vaaditaan olevan ei-negatiivinen, symmetrinen muuttujan todennäköisyysjakauman suhteen sekä kaikkialla differentioituva. Jakokriteereissä epäpuhtausmittaa käytetään etsimällä sen perusteella se ominaisuus, joka maksimoi odotusarvoisen epäpuhtauden vähenemisen, kun data jaetaan kyseisen ominaisuuden suhteen (Maimon, O. & Rokach, L. 2002).

Olkoon O käytössä oleva opetusdata, y tarkasteltava luokka-attribuutti, c tarkasteltavan luokka-attribuutin mahdollisten arvojen määrä, $\{p_1, p_2, .., p_c\}$ opetusdatan suhteelliset frekvenssit kussakin luokassa, a se ominaisuus, jonka hyvyyttä jakokohtana arvioidaan ja $O_{a,v} = \{x \mid x \in O,\ x_a = v\}$ se joukko jossa datapisteiden ominaisuus a saa arvon v.

Käytettäessä epäpuhtausmittana informaatioteoriasta saatavaa entropian suuretta, kutsutaan jakokriteeriä nimellä informaatiolisä (engl. *information gain*). Entropia määritellään kaavalla:

$$H(O,y) = -\sum_{i=1}^{c} p_i \, log(p_i).$$

Informaatiolisä ominaisuuden a suhteen taas lasketaan kaavalla

$$IG(O,a) = H(O,y) - \sum_{v \,\in\, arvot(a)} \frac{|O_{a,v}|}{|O|} H(O_{a,v}, y).$$

Toinen tunnettu jakokriteereissä hyödynnetty epäpuhtausmitta on niin sanottu Gini-epäpuhtaus

$$Gini(O,y) = 1 - \sum_{i=1}^{c} p_i^2,$$

jonka avulla lasketaan Gini-lisä (engl. *gini gain*)

$$GG(O,a) = Gini(O,y) - \sum_{v \,\in\, arvot(a)} \frac{|O_{a,v}|}{|O|} Gini(O_{a,v}, y).$$

Informaatiolisän ja Gini-lisän käyttäytyminen ollaan todettu hyvin samankaltaisiksi. Ne tekevät poikkeavia päätöksiä, siitä jaetaanko data tarkasteltavan ominaisuu-

den perusteella vain 2 % tapauksista (Raileanu, L. E. & Stoffel, K. 2004). Yhdeksi ongelmaksi informaatiolisän käytössä ollaan havaittu se, että se suosii ominaisuuksia, jotka voivat saada useita arvoja. Tämä johtuu siitä, että on todennäköistä, ettei opetusdatassa esiinny edes kahta havaintoa, joilla tällainen ominaisuus saisi saman arvon (Quinlan, J. R. 1986). Tähän ongelmaan ratkaisuksi ollaan ehdotettu muun muassa erilaisia normalisointi-metodeja. Eräs näistä on Quinlanin itsensä ehdottama lisäsuhde (engl. *gain ratio*)

$$GR(O,a) = \frac{IG(O,a)}{H(O,a)},$$

joka normalisoi informaatiolisän jakamalla sen opetusdatan entropialla tarkasteltavan ominaisuuden suhteen. Tämän erääksi ongelmaksi on todettu se, ettei se ole määritelty kun $H(O,a) = 0$ ja se, että se saattaa suosia ominaisuuksia, joiden entropia on vähäinen, vaikka niistä saatava informaatiolisä ei olisi merkittävä. Ehdotetaankin, että lisäsuhdetta tulisi käyttää siten, että lasketaan ensin informaatiolisä kaikille ominaisuuksille ja tarkastellaan lisäsuhdetta vain niillä ominaisuuksilla, joilla informaatiolisä on keskimääräistä suurempi. Formaalimpi tapa normalisoida informaatiolisä on De Mántaras, R. L. (1991) esittelemä etäisyysmitta (engl. *distance measure*), jossa normalisointi tehdään yhteisinformaation (engl. *mutual information*) avulla.

Edellä esitellyt jakokriteerit toimivat ainoastaan luokittelupuiden tapauksessa. Regressiopuissa yksi jaon tekemiseen soveltuva kriteeri on pienimmän virheneliösumman menetelmä (Breiman, L. et al. 1984, luku 8) vastaavasti kuin lineaarisessa regressiossa. Regressiopuut arvioivat kohdemuuttujaa lehtisolmuissa sijaitsevien datapisteiden kohdemuuttujan arvojen keskiarvon avulla. Parhaaksi jakokohdaksi valitaan se ominaisuus, joka minimoi virheneliösumman

$$SSE(O,a) = \sum_{v \in arvot(a)} \sum_{x \in O_{a,v}} (x_y - \bar{O}_{a,v})^2$$

verrattaessa lehtisolmuissa sijaitsevien datapisteiden x kohdemuuttujan todellisia arvoja x_y ja niistä laskettua keskiarvoa $\bar{O}_{a,v}$.

Useissa jakokriteereitä vertailevissa tutkimuksissa ollaan todettu, että yksittäistä parasta jakokriteeriä on vaikea määrittää. Se, miten hyvin tietty jakokriteeri pärjää eri

sovelluksissa, vaihtelee huomattavasti (Maimon, O. & Rokach, L. 2002). Lisäksi ollaan todettu, että jakokriteerin valinta ei vaikuta siihen, kuinka hyvin lopullinen puu mallintaa tarkasteltavaa ongelmaa kovinkaan paljoa (Breiman, L. et al. 1984, luku 4). Kaikki tässä luvussa mainitut jakokriteerit perustuvat jaon tekemiseen yhden ominaisuuden perusteella, mutta myös useita ominaisuuksia huomioon ottavia jakokriteereitä ollaan esitelty. Ne perustuvat usein tarkasteltavien ominaisuuksien lineaarikombinaatioihin ja muutama esimerkki niistä on perseptroni-oppiminen, mäenkiipeämismetodit (Murthy, S. K. 1998) sekä tukivektorikoneet (Bennet, K. P. & Blue, J. A. 1998).

2.1.2 Pysäytyskriteerit

Pysäytyskriteeri määrää sen, milloin luotavaa puuta ei enää tarvitse jakaa useampiin alipuihin. Maimon, O. & Rokach, L. (2008, luku 6) luettelee erilaisia pysäytyskriteereitä seuraavasti:

- Kaikki opetusdatan alkiot on luokiteltu.
- Ollaan saavutettu puun maksimisyvyys.
- Korkein jakokriteerin arvo ei saavuta jotain sille asetettua rajaa.
- Jos solmu vielä jaettaisiin, olisi jommassa kummassa tai molemmissa sen lapsisolmuista liian vähän opetusdatan alkioita ennalta määrättyyn minimiin nähden.

Yleisessä tapauksessa puuta tullaan karsimaan kasvattamisen jälkeen, jolloin on perusteltua valita löysä lopetuskriteeri. Tiukoilla lopetuskriteereillä pystytään hallitsemaan puun kokoa niin sanotussa esikarsinnassa. Esikarsinnan vaarana on, ettei sen avulla luotu puu välttämättä mallinna edes opetusdataa kyllin hyvin. Käyttökelpoinen menetelmä se saattaa kuitenkin olla tilanteissa, joissa mallin rakentamisen tehokkuutta pidetään tärkeämpänä kuin mallin tarkkuutta (Breslow, L. & Aha, D. 1996). Tämä huomio on hyvä pitää mielessä lukiessa lukua 3, jossa käsitellään useita päätöspuita hyödyntäviä menetelmiä.

2.2 Karsimismenetelmät

Kasvatusvaiheessa syntyneellä puulla on taipumus olla tarpeettoman suuri ja monimutkainen. Tämä on ongelmallista, sillä suuret puut ovat taipuvaisia ylisovittamaan opetusdataa eli mallintamaan tutkittavan ilmiön lisäksi opetusdataan liittyvää satunnaisvaihtelua. Tämä aiheuttaa sen, että luotu malli yleistyy huonosti opetusdatan ulkopuolelle. Muita suurten puiden ongelmia ovat niiden vaikea luettavuus, lehtisolmut, joissa on vain muutama opetusdatan alkio sekä toistuvat alipuut (Breslow, L. & Aha, D. 1996). Breiman, L. et al. (1984, luku 3) esittelee taulukon, joka havainnollistaa puun koon vaikutusta luokitteluvirheeseen: mallin yleistyvyys opetusdatan ulkopuolelle paranee puun kokoa pienennettäessä tiettyyn pisteeseen saakka, jonka jälkeen malli sovittaa sekä opetusdatan että testidatan huonosti. Päätöspuiden koon hallitsemiseen ollaankin esitetty useita ratkaisuja, joista yleisimmäksi on muodostunut alun perin Breiman, L. et al. (1984, luku 3) ehdottama jälkikarsinta (engl. *post pruning*), jota nimitetään yleisesti vain karsinnaksi. Esitellään nyt muutama karsimiseen käytetty menetelmä.

Olkoon $R(T)$ puun T virhe opetusdatassa eli niin sanottu takaisinsijoitusvirhe (engl. *resubstitution error*), $lehdet(T)$ puun T lehtisolmujen joukko sekä $karsi(T,t)$ puu T, josta ollaan karsittu alipuu t.

Tutustutaan ensin Breiman, L. et al. (1984, luku 3) esittelemään menetelmään minimaalinen hinta-kompleksisuus-karsiminen (engl. *minimal cost-complexity pruning*). Sen tavoite on hallita puun kompleksisuutta eli lehtisolmujen määrää sekä hintaa eli takaisinsijoitusvirhettä. Menetelmä etenee kahdessa vaiheessa: ensin luodaan jono toinen toistaan pienempiä puita $T_0, T_1, ..., T_n$ karsimalla aina edeltävästä puusta se alipuu t, joka minimoi seuraavan lausekkeen:

$$\alpha = \frac{R(karsi(T,t)) - R(T)}{|lehdet(T)| - |lehdet(karsi(T,t))|}.$$

Tässä siis T_0 on alkuperäinen kasvatusvaiheessa muodostunut puu ja T_n sen juurisolmu. Toisessa vaiheessa valitaan karsituista puista paras testaamalla niitä joko tätä varten varatulla karsintadatalla tai opetusdatan osajoukoilla, niin sanotusti ristivalidoimalla. Ristivalidointia laskennallisen raskautensa takia suositellaan lähinnä

siinä tapauksessa, että alkuperäinen data on niin pieni, että sen jakaminen opetus- ja karsintadataksi heikentäisi kasvatusvaiheessa luotavan puun tarkkuutta (Murthy, S. K. 1998). Parhaan puun valintaan esitetään kaksi mahdollista tapaa. Voidaan valita yksinkertaisesti se puu, joka minimoi luokitteluvirheen testivaiheessa (0-SE). Toinen valintamenettely on valita se puu, joka minimoi lehtisolmujen määrän niiden puiden joukossa, joiden luokitteluvirhe on korkeintaan yhden keskivirheen päässä luokitteluvirheen minimistä (1-SE).

Menetelmäksi, joka poistaa tarpeen erilliselle testidatalle tai ristivalidoinnille, Quinlan, J. R. (1993) ehdottaa pessimistisen karsimisen jälkeläistä virheeseen perustuvaa karsimista (engl. *error based pruning*). Tämä menetelmä perustuu todellisen luokitteluvirheen estimointiin takaisinsijoitusvirheen avulla. Arviointi toteutetaan takaisinsijoitusvirheen luottamusvälin ylärajalla jollain valitulla merkitsevyystasolla, tässä virheiden oletetaan noudattavan binomijakaumaa. Puuta käydään läpi alhaalta ylöspäin laskien kyseinen virhearvio kussakin solmussa: koko tarkasteltavalle alipuulle, jonka juurena solmu toimii; tilanteelle, jossa solmu korvattaisiin lehtisolmulla ja tilanteelle, jossa solmu korvattaisiin sen eniten dataa sisältävällä alipuulla. Se, kuinka solmu lopulta käsitellään, valitaan pienimmän lasketun virhearvion mukaan. Tässä menettelyssä merkille pantavaa on se, että toisin kuin esimerkiksi hinta-kompleksisuuskarsimisessa, voidaan näin koko solmun karsimisen sijaan korvata se jollain sen alipuulla.

Esitellyistä menetelmistä virheeseen perustuva karsiminen tapaa karsia puuta liian vähän, kun taas hinta-kompleksisuus-karsiminen tapaa karsia puuta liian paljon varsinkin valittaessa paras puu 1-SE-menetelmällä (Maimon, O. & Rokach, L. 2008, luku 6). Toinen hinta-kompleksisuus-karsimisen ongelma on sen epävakaus ristivalidointia hyödynnettäessä (Breslow, L. & Aha, D. 1996). Vastaavasti kuin jakokriteerien tapauksessa, myös karsintamenetelmän suoriutuminen riippuu sovellustilanteesta (Mingers, J. 1989). Menetelmää valittaessa huomioon tulee ottaa esimerkiksi opetusdatan koko sekä erillisen karsintadatan saatavuus.

2.3 Algoritmeja

Päätöspuun johtamiseen tarkoitetut algoritmit koostuvat säännöistä, joiden perusteella puu luodaan. Näihin kuuluu edellä esiteltyjen jakokriteerien ja karsimismenetelmien lisäksi muun muassa se, miten puuttuvat arvot käsitellään puuta luodessa. Näiden algoritmien hyödyllisyydestä kertoo se, että niistä kaksi, CART ja C4.5, mainittiin tiedonlouhinnan algoritmien kärkikymmenikössä (Wu, X. et al. 2008). Tässä osiossa tutustutaan niihin tarkemmin.

C4.5 on Quinlan, J. R. (1993) aikaisemman ID3-algoritmin pohjalta kehittämä algoritmi. Se käyttää jakokriteerinä lisäsuhdetta ja karsimismenetelmänä virheeseen perustuvaa karsimista. Puuttuvat arvot siinä käsitellään ottamalla jakokriteerissä huomioon puuttuvien arvojen määrä. Edelleen jakoa tehdessä lisätään kaikki datapisteet, joista puuttuu jakokohtana olevan ominaisuuden arvo, kaikkiin jaossa syntyviin lehtisolmuihin painottaen niitä lehtisolmua vastaavan ominaisuuden arvon todennäköisyydellä. C4.5 tarjoaa myös karsinnasta poikkeavan tavan yksinkertaistaa puuta esittämällä se sarjana ehtolauseita ja yksinkertaistamalla alkuperäisen puun sijaan tätä sääntöjoukkoa. Tämän menetelmän ollaan todettu luovan tarkempia puita kuin karsittu C4.5 lisätyn laskennallisen vaativuuden hinnalla (Breslow, L. & Aha, D. 1996).

CART on (Breiman, L. et al. 1984) esittelemä algoritmi päätöspuiden luomiseksi, sen nimi on lyhennelmä sanoista ”Classification and Regression Trees” eli luokitteluja regressiopuut. Nimensä mukaisesti se kykenee käsittelemään sekä jatkuvaa että diskreettiä kohdemuuttujaa. Toinen CART:n ominaispiirre on se, että se luo aina binäärisiä puita käyttäen luokittelupuissa jakokriteerinä Gini-lisästä johdettua kahtiajakokriteeriä ja regressiopuissa joko esiteltyä pienimmän virheneliösumman menetelmää tai niin sanottua vähäisimmän absoluuttisen hajonnan menetelmää. Karsimismenetelmänä siinä käytetään hinta-kompleksisuus-karsimista. Puuttuvien arvojen käsittely toteutetaan etsimällä jokaiselle jakokohdalle sijaisjakaja (engl. *surrogate split*) sellaisesta ominaisuudesta, jonka perusteella tehtävä jako muistuttaa eniten alkuperäistä jakokohtaa. Näin menetellen voidaan käyttää sijaisjakajaa, mikäli käsiteltävästä datapisteestä puuttuu varsinaisena jakokohtana toimiva ominaisuus.

Muita CART:n ominaisuuksia ovat esimerkiksi mahdollisuus luokkien tasapainottamiseen ja virheellisen luokittelun hinnan määrittämiseen.

Lim, T. S. et al. (2000) arvioi useiden päätöspuualgoritmien vertailussa, etteivät menetelmien virheet poikkea toisistaan merkitsevästi. Eroja kuitenkin havaittiin menetelmien tehokkuudessa: erityisesti C4.5 havaittiin nopeammaksi kuin CART sekä sääntöjoukkoihin että puihin perustuvilla versioilla. Joka tapauksessa menetelmän valinta tulee tehdä tapauskohtaisesti, sillä esimerkiksi jatkuva kohdemuuttuja rajaa C4.5:n pois mahdollisista vaihtoehdoista.

3 Useiden päätöspuiden käyttöön perustuvat menetelmät

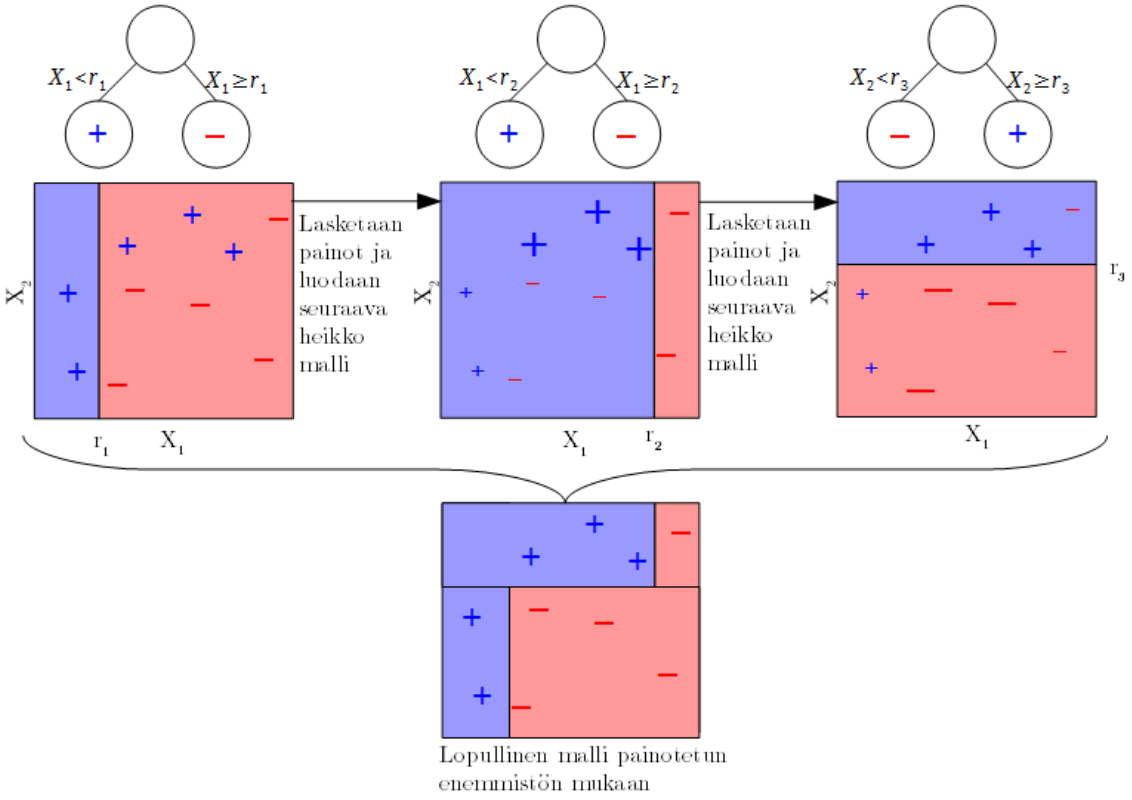
Yksittäiset päätöspuut ennakoivana mallina sisältävät monia toivottuja ominaisuuksia kuten sen, että ne ovat tehokkaita ja sen, että niitä on helppo lukea. Ongelmallista niissä on kuitenkin se, ettei niiden luomiseen ole löytynyt tapaa, jossa tarkkuuden parantaminen opetusdatassa ei huonontaisi mallin yleistyvyyttä (Ho, T. K. 1995). Tämän voi tulkita niin, etteivät karsimismenetelmät täysin pysty hallitsemaan päätöspuiden taipumusta ylisovittamiseen. Tähän tehtävään paremmin kykenevät menetelmät, joissa luodaan ensin useita malleja soveltaen jotain koneoppimismenetelmää ja kootaan näiden mallien tulokset lopullisen mallin saamiseksi. Vaikka mallin yleistyvyys näin menetellessä paraneekin, tulee huomioida, että yksittäiseen päätöspuuhun verrattuna sen tulkinnallisuus huononee ja sen tarvitseman muistin määrä kasvaa huomattavasti.

Esitellyn tyylisiä menetelmiä nimitetään kokoonpanomenetelmiksi (engl. *ensemble method*) ja niissä sovellettua koneoppimismenetelmää pohjaoppijaksi (engl. *base learner*). Eräs syy sille, miksi päätöspuut ovat osoittautuneet suosituiksi pohjaoppijoiksi, on juurikin niiden laskennallinen tehokkuus (Opitz, D. & Maclin, R. 1999), jonka merkitys korostuu, kun malleja täytyy luoda useita. Yleisiksi päätöspuita hyödyntäviksi kokoonpanomenetelmiksi ovat muodostuneet tehostaminen (engl. *boosting*), bootstrap-aggregointi (engl. *bootstrap-aggregation, bagging*) sekä tämän erikoistapaus satunnaismetsä. Tässä luvussa esitellään mainittuja menetelmiä ja vertaillaan niiden suoriutumista toisiinsa sekä yksittäisiin päätöspuihin nähden. Pohjaoppijana oletetaan toimivan päätöspuu, vaikka osa teoriasta pätee muussakin tapauksessa.

3.1 Tehostaminen

Tehostamismenetelmät syntyivät alunperin Schapire, R. E. (1990) esittämänä vastauksena teoreettiseen kysymykseen siitä, voidaanko useista heikoista oppijoista koostaa vahva oppija. Tässä heikko oppija tarkoittaa koneoppimismenetelmää, jon-

ka muodostamilta malleilta vaaditaan vain se, että niiden tarkkuus on parempi kuin puhdas arvaus. Tehostaminen pitää sisällään eri menetelmiä, joiden toiminta etenee muodostamalla jono heikon oppijan luomia malleja painottaen niitä datapisteitä, joiden kohdalla edeltävät mallit ovat tehneet virheitä. Näin muodostuneet mallit yhdistetään antaen korkeampi paino niille malleille, jotka tekivät vähän virheitä. Tätä prosessia havainnollistetaan kuvassa 1 pohjaoppijan ollessa päätöskanto eli päätöspuu, joka tekee ennusteensa vain yhden ominaisuuden arvojen perusteella. Eri tehostamismenetelmät poikkeavat toisistaan esimerkiksi siinä, miten ne toteuttavat virheellisesti luokiteltujen datapisteiden painottamisen.



Kuvio 1. Tehostamisen toiminta

Mukautuva tehostamisalgoritmi AdaBoost on Schapire, R. E. & Freund, Y. (1997) esittelemä tehostamismenetelmä, joka kehitettiin alunperin vastaamaan vain kaksiluokkaiseen luokitteluteht\u00e4v\u00e4\u00e4n. Eräs tapa yleistää se moniluokkaiseksi toimii yksinkertaistamalla ongelma sarjaksi kaksiluokkaisia ongelmia (Schapire, R. E. & Singer, Y. 1999), myös yleistyksiä regressiotehtäviin on ehdotettu Solomatine, D.P. &

Shrestha, D. L. (2006). Tutustutaan tässä sen toimintaan alkuperäisessä versiossaan mukaillen Wu, X. et al. (2008). Olkoon K valittu kierrosten lukumäärä, $k \in \{1,...,K\}$ menossa oleva kierros, n opetusdatan alkioiden lukumäärä, (x_i, y_i) missä $i \in \{1,...,n\}$ opetusdatan alkio, jossa luokkamuuttuja y_i koodattuna joukkoon $\{-1,1\}$ sekä $w_k(i)$ i:nnen datapisteen paino kierroksella k. Asetetaan ensin $w_1(i) = \frac{1}{n}$ kaikilla $i \in 1,...,n$, menetelmä etenee seuraavasti:

- Luodaan heikko malli h_k käyttäen datapisteiden painoina arvoja $w_k(i)$.
- Lasketaan virheellisesti luokiteltujen datapisteiden suhteellinen frekvenssi opetusdatassa ε_k.
- Asetetaan
$$\alpha_k = \frac{1}{2} ln(\frac{1-\varepsilon_k}{\varepsilon_k}).$$
Näitä käytetään lopullisessa mallissa painottamaan luotuja heikkoja malleja. Nähdäänkin, että yksittäisen heikon mallin paino on pieni, jos se tekee paljon virheitä ja suuri, jos se tekee niitä vähän.
- Päivitetään datapisteiden painot seuraava kierrosta varten
$$w_{k+1}(i) = \frac{w_k(i)}{N_k} \times \begin{cases} exp(-\alpha_k), & \text{jos } h_k(x_i) = y_i \\ exp(\alpha_k), & \text{jos } h_k(x_i) \neq y_i \end{cases},$$
tämä saadaan sievennettyä, koska $y_i \in \{-1,1\}$:
$$w_{k+1}(i) = \frac{w_k(i) exp(-\alpha_k y_i h_k(x_i))}{N_k}.$$
N_k valitaan siten, että w_k pysyy jakaumana.
- Kun kaikki K heikkoa mallia ollaan luotu, saadaan lopullisen mallin ennuste havainnon x luokalle seuraavan lausekkeen etumerkistä:
$$\sum_{k=1}^{K} \alpha_k h_k(x).$$
Valitaan siis se luokka, johon painotetusti suurin osa heikoista malleista luokitteli x:n.

Eräs merkittävä variaatio esitellylle menetelmälle on datapisteiden painottamisen sijaan, ottaa heikkoja malleja luotaessa huomioon vain osa opetusdatasta tehden va-

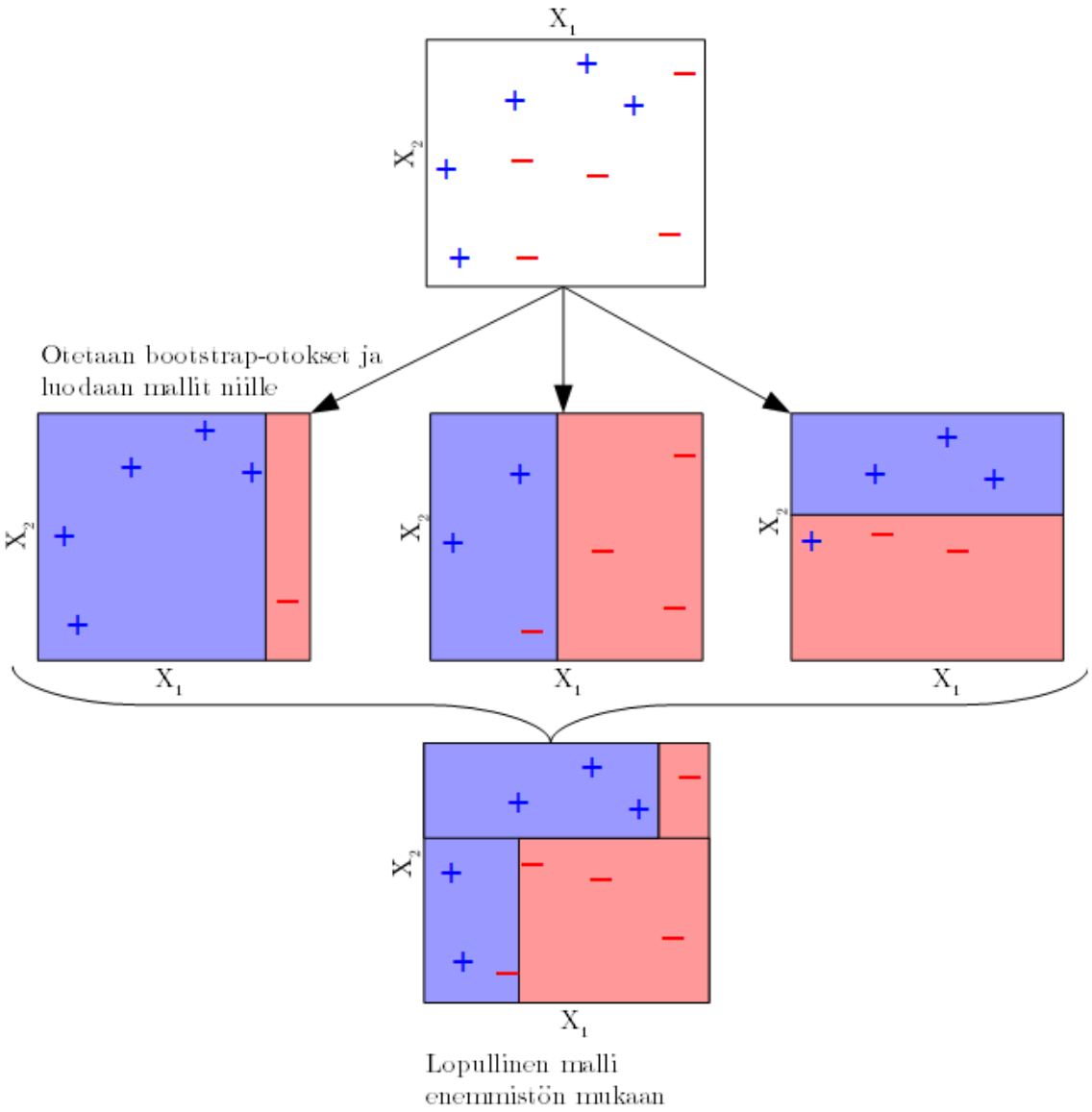
linta mukaan otettavista datapisteistä painoista saatavien todennäköisyyksien avulla (Dietterich, T. 2000). Kaikilla variaatioillaan AdaBoost on käytännössä osoittautunut erittäin tehokkaaksi ja vertautuu useimmissa tapauksissa hyvin muihin koneoppimismenetelmiin nähden. Erityisesti Schapire, R. E. (1999) näyttää vertailussa, että AdaBoostilla tehostettu C4.5 pärjää miltei aina paremmin kuin yksittäinen C4.5:llä luotu puu. Saman vertailun tuloksissa joissain tehtävissä jopa AdaBoostilla tehostettu päätöskanto päihittää C4.5:n. Quinlan, J. R. (1996) tulosten mukaan tehostaminen vähentää C4.5:n virheiden määrää keskimäärin 15 %.

Toinen merkittävä tehostamismenetelmien joukko perustuu havaintoon, että tehostaminen voidaan nähdä optimointitehtävänä funktioavaruudessa. Tähän joukkoon kuuluvat menetelmät siis pyrkivät löytämään sellaisen mallin, joka minimoi valitun häviöfunktion arvon ja niitä kutsutaan gradienttitehostamiseksi (engl. *gradient boosting*). Minimointi suoritetaan iteratiivisesti gradienttilaskeutumismenetelmällä eli siirtämällä luotavaa mallia kullakin iteraatiolla sellaiseen suuntaan, jossa häviöfunktion arvot pienenevät nopeiten. Eräs mielenkiintoinen huomio gradienttitehostamisesta on se, että myös AdaBoost voidaan esittää gradienttitehostamismenetelmänä tietyllä häviöfunktion valinnalla (Mason, L. & Baxter, J. & Bartlett, P. & Frean, M. 1999). Erilaiset gradienttitehostamismenetelmät ovat osoittautuneet käytännössä erittäin suosituiksi ja tehokkaiksi. Erityisesti Cheng, T. & Guestrin, C. (2016) esittelemä XGBoost-gradienttitehostamismenetelmä on ollut viime aikoina kovassa suosiossa.

3.2 Bootstrap-aggregointi

Bootstrap-aggregointi on Breiman, L. (1996) esittelemä kokoonpanomenetelmä. Se eroaa tehostamisesta olennaisesti siten, että siinä pohjaoppijan mallit luodaan toisistaan riippumatta. Menetelmä etenee ottamalla opetusdatasta valittu määrä bootstrap-otoksia takaisinsijoittaen eli satunnaisesti valittuja otoksia siten, että valitut datapisteet voidaan valita tulevissakin otoksissa. Kustakin bootstrap-otoksesta luodaan pohjaoppijan avulla malli ja näiden mallien ennusteista lasketaan lopullisen mallin ennuste. Ennuste lasketaan keskiarvona, jos kohdemuuttuja on jatkuva ja

enemmistön valitsemana luokkana diskreetin kohdemuuttujan tapauksessa. Menetelmää havainnollistetaan kuviossa 2 pohjaoppijan ollessa päätöskanto.



Kuvio 2. Bootstrap-aggregoinnin toiminta

Eräs merkittävä variaatio bootstrap-aggregoinnille on Ho, T. K. (1995) esittelemän satunnaisaliavaruusmenetelmän (engl. *random subspace method*) pohjalta Breiman, L. (2001) kehittelemä menetelmä nimeltään satunnaismetsä. Satunnaismetsä lisää bootstrap-aggregointiin satunnaisuutta siten, että päätöspuita luotaessa paras jakokohta valitaankin tarjolla olevien ominaisuuksien satunnaisesta osajoukosta kaik-

kien ominaisuuksien sijaan. Kyseinen menettely vähentää ylisovittamisen mahdollisuutta entisestään, sillä näin luodut puut poikkeavat toisistaan enemmän kuin satunnaistamattomat puut.

Myös bootstrap-aggregointi ja erityisesti satunnaismetsät ovat pärjänneet hyvin muihin koneoppimismenetelmiin nähden. Quinlan, J. R. (1996) näyttää boostrap-aggregoinnin vähentävän C4.5:n tekemien virheiden määrää keskimäärin 10 %. Caruana, R. & Niculescu-Mizil, A. (2006) suorittamassa vertailussa bootstrap-aggregointi sijoittui kolmanneksi ja satunnaismetsät toiseksi häviten ainoastaan tehostamiselle. Kuitenkin Caruana, R. & Karampatziakis, N. & Yessenalina, A. (2008) näyttää, että käsiteltäessä todella moniulotteista dataa päihittää satunnaismetsät myös tehostamisen. Breiman, L. (2001) suorittamassa AdaBoostin ja satunnaismetsien vertailussa selviää, että myös meluisassa datassa satunnaismetsät pärjäävät tehostamista paremmin, sillä AdaBoost on satunnaismetsiä alttiimpi ylisovitukselle. Kiinnostavaa olisi saada tutkimustietoa satunnaismetsän pärjäämisestä nykyaikaisempaan tehostamismenetelmään XGBoostiin verrattuna, sillä siinä tehostamisen aikaisempiin ongelmiin, kuten ylisovittamiseen meluisassa datassa, ollaan kiinnitetty huomiota.

4 Sovelluksia

Esiteltyjä päätöspuihin perustuvia menetelmiä ollaan sovellettu laajasti koneoppimisen eri tehtäviin. Tässä luvussa tarkastellaan minkälaisissa tilanteissa esiteltyjä menetelmiä voidaan käyttää yleisesti sekä esimerkkien avulla. Esiteltyjen menetelmien mahdollisten sovellusalueiden valtavan määrän takia ei tutkielman rajoissa kovin kattavaa käsitystä kyetä aiheesta antamaan. Rajoitutaan käymään läpi muutama tunnettu esiteltyjen menetelmien sovellus yksityiskohtaisesti, jotta saadaan käsitystä siitä, miten menetelmiä sovelletaan käytännössä. Tämän lisäksi luetellaan joitain esiteltyjen menetelmien viimeaikaisia tehtäviä eri sovellusalueilla, jotta saataisiin käsitys siitä, kuinka laajasti menetelmiä ollaan sovellettu. Lopuksi pyritään luomaan yleiskäsitystä siitä, mihin päätöspuumenetelmät ylipäätään soveltuvat.

Kinect on liikkeentunnistusohjelma, joka hyödyntää satunnaismetsiä ruumiinosien luokitteluun. Merkittävän siitä tekee se, että se saavutti huomattavasti aikaisempia liikkeentunnistusjärjestelmiä suuremman nopeuden ja kykenee käsittelemään 200 kehystä sekunnissa. Tutustutaan tässä siihen, kuinka se käyttää satunnaismetsiä tarkemmin Shotton, J. & Cook, M. & Sharp, T. & Moore, R. et al. (2011) mukaan. Kinect toimii syvyyskameran avulla, joten pikselille p pystytään määrittämään kuvassa syvyys $d(p)$. Luokittelu eri ruumiinosiin tehdään kullekin pikselille erikseen käyttämällä ominaisuuden arvoina syvyysarvojen $d(p+u)$ ja $d(p+v)$ erotusta, missä u ja v ovat $d(p)$:llä normalisoituja siirtymäparametreja. Jos siis esimerkiksi $p+u$ olisi p:n oikealla puolella ja $p+v$ p:n vasemmalla puolella suunnilleen yhtä kaukana p:stä, voitaisiin ominaisuuden arvon avulla päätellä kuuluuko p johonkin ohueen ruumiinosaan. Satunnaismetsä luodaan kasvattamalla kukin puu eri joukolle satunnaisesti syntetisoituja kuvia 900 000 kuvan opetusdatasta ottaen kustakin kuvasta 2000 satunnaista pikseliä. Ominaisuuksia varten valitaan satunnaisesti sekä pari (u,v) että raja r ominaisuuden arvolle, jonka suhteen pikselit jaetaan. Data jaetaan siis aina kahteen joukkoon, joista toisessa $d(p+u)-d(p+v)<r$ ja toisessa $d(p+u)-d(p+v)\geq r$. Puiden kasvatuksessa käytetään jakokriteerinä informaatiolisää ja pysäytyskriteerinä alarajaa informaatiolisän arvolle sekä ylärajaa puun sy-

vyydelle.

Toinen merkittävä kokoonpanomenetelmien sovellus on Viola, P. & Jones, M. (2001) esittelemä AdaBoostilla tehostettuihin päätöskantoihin perustuva esineentunnistusjärjestelmä, joka soveltuu erityisesti kasvojen tunnistamiseen. Kohdemuuttuja saa siis arvon -1, mikäli kuvassa ei ole kasvoja ja arvon 1, mikäli on. Kuvia käsitellään aina yksi 24×24-kokoinen ikkuna kerrallaan ja ominaisuuksina käytetään ikkunan eri alueista tietyntyyppisten mustavalkeiden suorakulmioiden avulla laskettavia arvoja. Nämä ominaisuudet kertovat ikkunan kontrastista siten, että valkoisen alueen alle jäävien pikselien kirkkauden summasta vähennetään vastaava arvo mustan alueen alle jäävistä pikseleistä. Esimerkiksi kuvan keskelle sijoitetun suorakulmion, joka on sivuilta musta ja keskeltä valkoinen, avulla lasketun arvon ollessa suuri voitaisiin saada näyttöä siitä, että kuvassa on nenä ja siten myös kasvot.

Sovelluksessa AdaBoostissa pohjaoppijana toimiva päätöskanto etsii nyt sen suorakulmion, jonka mukaan aikaisempien kierrosten virheiden mukaan painotetut kuvat saadaan luokiteltua parhaiten. Valituille ominaisuuksille se laskee myös optimaalisen raja-arvon, jonka suhteen kuvat luokitellaan. Menetelmän nopeutta parantaa entisestään kaksi tekijää, joista ensimmäinen on tapa laskea esitellyn tyyppisten ominaisuuksien arvoja vaatien korkeintaan 9 laskutoimitusta. Toinen nopeuttava tekijä on menetelmä hylätä selkeästi kasvoja sisältämättömät ikkunat nopeasti. Tämä toteutetaan luokittelemalla kaikki ikkunat ennen varsinaisella mallilla luokittelua jonolla yksinkertaisempia AdaBoost-malleja, jotka minimoivat mahdollisuutta luokitella kasvoja sisältämättömiä kuvia väärin. Mielenkiintoinen huomio esitellyistä sovelluksista koskee niiden käyttämien ominaisuuksien samankaltaisuuksia ja eroja: Viola-Jones käsittelee kokonaisia alueita kerrallaan ja käyttää kontrastia selvittämään eri alueiden syvyyseroja, Kinect sen sijaan käsittelee yksittäisiä pikseleitä ja pääsee käyttämänsä laitteiston avulla suoraan käsiksi niiden syvyysarvoihin.

Luetellaan seuraavaksi joitain esiteltyjen menetelmien sovelluksia eri tieteenaloilla. Käydään lyhyesti läpi, mitä menetelmää ollaan käytetty ja mihin tehtävään.

- Fysiikan alalla Aad, G. et al. (2014) hyödynsi tehostettuja päätöspuita analy-

soimaan Higgsin bosonin hajoamista W-hiukkasiksi.

- Lee, T. M. & Markowitz E. M. et al. (2015) suorittamassa sosiologisessa tutkimuksessa selvitettiin satunnaismetsien avulla, mitkä muuttujat selittävät väestön tietämystä ilmastonmuutoksesta.
- Lääketieteessä Gray, K. R. & Aljabar, P. & Hammers, A. et al. (2013) esittelee satunnaismetsiin perustuvan tavan luokitella Alzheimerin taudin vakavuutta MRI- ja PET-kuvien sekä geneettisten suureiden avulla.
- Biologian alalla Jia, J. & Liu, Z. & Xiao, X. et al. (2016) esitteli satunnaismetsiin perustuvan menetelmän selvittää, millä proteiinin sisältämillä lysiinin jäämillä on sellainen ominaisuus, että ne voidaan sukkinyloida.
- Maantieteessä Youssef, A. M. & Pourghasemi, H. R. & Pourtaghi, Z. S. & Al-Katheeri, M. M. (2016) selvitti maanvyörymäalttiuden kartoittamista tehostettujen päätöspuiden, satunnaismetsien, CART:n sekä yleistettyjen lineaarimallien avulla.
- Taloustieteen alalla Dursun, D. & Kuzey, C. & Uyar A. (2013) selvitti päätöspuiden käyttöä yritysten menestyksen ennakointiin erilaisten rahoitukseen liittyvien suureiden avulla.
- Tietotekniikan alalla Singh, K. & Guntuku, S. C. & Thakur, A. & Hota, C. (2014) esittelee satunnaismetsiin perustuvan menetelmän P2P-bottiverkkojen havaitsemiseen DDoS-hyökkäysten ehkäisemiseksi.
- Min, F. & Zhang, H. (2016) esitteli satunnaismetsiin perustuvan suosittelijajärjestelmän.

Koska esiteltyjä menetelmiä on sovellettu niin laajasti, on vaikeaa luoda kattavaa yleiskäsitystä siitä, minkälaisiin tehtäviin ne soveltuvat. Joitain havaintoja siitä, minkälaisissa tilanteissa eri menetelmät ovat osoittautuneet toisia menetelmiä suositummiksi, voidaan kuitenkin tehdä. Aihetta koskevan kirjallisuuden määrästä voisi päätellä, että esimerkiksi kuvien ja äänen analysointiin liittyvissä tehtävissä vaikuttaisivat niin sanotut syvät neuroverkot olevan nykyään päätöspuumenetelmiä suositumpia. Tässä tutkielmassa on kuitenkin esitelty sovelluksia, jotka hyödyntävät kokoonpanomenetelmiä menestyksekkäästi kuva-analyysiin, joten vertaileva tutkimus aiheesta voisi olla tarpeen. Myös ohjaamattoman oppimisen klusterointiteh-

tävissä on yleisempää soveltaa muita menetelmiä, vaikka osalle tutkielmassa esitellyistä menetelmistä ollaankin esitelty klusterointiin soveltuvia variaatioita (esim. Yu, P. S. & Yiyuan, X. & Bing, L. (2000)).

Kerraten luvussa 3 esiintyneiden vertailujen tuloksia käydään vielä lyhyesti läpi miten esitellyt menetelmät vertautuvat toisiinsa.

- Yksittäistä päätöspuuta kannattaa käyttää, kun tärkeinä arvoina pidetään luotavan mallin kokoa, tulkinnallisuutta ja tehokkuutta. Tarkkuus kärsii ylisovittamisen takia enemmän kuin kokoonpanomenetelmissä.
- Tehostaminen on hyvä vaihtoehto kun arvostetaan mallin tarkkuutta ja käytössä oleva data ei ole kovin meluisaa. On tosin olemassa menetelmiä tehostamisen toiminnan parantamiseen myös meluisassa datassa.
- Bootstrap-aggregointi sekä satunnaismetsät ovat myös hyviä vaihtoehtoja kun tarkkuus on tärkeää. Satunnaismetsät pärjäävät vertailujen perusteella esitellyistä menetelmistä parhaiten todella moniulotteisessa sekä meluisassa datassa.

5 Yhteenveto

Tässä tutkielmassa käsiteltiin päätöspuiden käyttöä koneoppimisessa. Aluksi käytiin läpi päätöspuihin perustuvia koneoppimisen menetelmiä lähtien yksittäisistä päätöspuista ja päätyen kokoonpanomenetelmiin. Havaittiin ylisovittaminen yksittäisten päätöspuiden erääksi merkittäväksi ongelmaksi ja käytiin läpi keinoja, joita sen hallitsemiseen ollaan käytetty. Näiden osioiden avulla luotiin kattava, muttei perusteellinen yleiskatsaus siihen, miten päätöspuita ollaan käytetty. Merkittävä havainto tutkielmasta onkin se, kuinka monipuolista päätöspuiden käyttö on koneoppimisessa ollut ottaen huomioon niiden näennäinen yksinkertaisuus.

Lopulta tutustuttiin esiteltyjen menetelmien joihinkin sovelluksiin eri aloilla. Tässä osiossa pyrittiin pääasiallisesti vastaamaan kysymykseen siitä, mihin esiteltyjä menetelmiä ollaan käytetty. Tähän kysymykseen yleistävän vastauksen löytäminen todettiin hankalaksi aiheen laajuuden vuoksi. Mielenkiintoista olisikin saada lisää tutkimustietoa siitä, minkälaisessa datassa eri koneoppimismenetelmät pärjäävät toisiaan paremmin; esimerkiksi siitä, milloin syvät neuroverkot pärjäävät kokoonpanomenetelmiä paremmin ja milloin huonommin. Tämä olisi hyödyllistä, sillä valinta koneoppimismenetelmästä joudutaan usein tekemään vertailemalla eri menetelmiä tapauskohtaisesti, mikä saattaa olla hyvinkin aikaavievä prosessi. Kaiken kaikkiaan tutkielma antaa hyvät valmiudet ymmärtää ja soveltaa esiteltyjä päätöspuumenetelmiä sekä kokoaa kattavasti aiheeseen liittyvää aiempaa kirjallisuutta.

Kirjallisuutta

Aad G. et al. 2014 *Observation and measurement of Higgs boson decays to WW* with the ATLAS detector*, Physical Review D, 92

Barros, R. C. & Jaskowiak, A. J. & Cerri, R. et al. 2014 *A framework for bottom-up induction of oblique decision trees*. Neurocomputing, 135, s.3-12

Bennet, K. P. & Blue, J. A. 1998. *A support vector machine approach to decision trees*. In Proceedings of IEEE World Congress on Computational Intelligence, 3, s.2396–2401.

Ben-Bassat, M. 1982. *Use of distance measures, information measures and error bounds in feature evaluation*. Handbook of Statistics, 2, s.773-791

Breiman, L. & Friedman, J.H. & Olshen, R.A. & Stone, C.J.(1984). *Classification and regression trees*. Belmont,CA:Wadsworth International Group.

Breiman, L. 1996. *Bagging Predictors*. Machine Learning, 24, s.123-140

Breiman, L. 2001. *Random Forests*. Machine Learning, 45, s.5-32

Breslow, L. & Aha, D. 1996. *Simplifying Decision Trees: A Survey*. Knowledge Engineering Review, 12, s.1-40.

Caruana, R. & Niculescu-Mizil, A. 2006. *An empirical comparison of supervised learning algorithms*. Proceedings of the 23rd international conference on Machine learning, s.161-168

Caruana, R. & Karampatziakis, N. & Yessenalina, A. 2008. *An empirical evaluation of supervised learning in high dimensions*. Proceedings of the 25th international conference on Machine learning, s.96-103

Chen, L. & Chu, C. & Huang, T. & Cai, Y. 2015. *Prediction and analysis of cell-penetrating peptides using pseudo-amino acid composition and random forest models*. Amino Acids, 47, s.1485-1493

Cheng, T. & Guestrin, C. 2016. *XGBoost: A Scalable Tree Boosting System*. Proceedings of the 22nd international conference on knowledge discovery and data mining. pp 785–794

De Mántaras, R.L. 1991. *A Distance-based Attribute Selection Method for Decision Tree Induction*. Machine Learning, 6, s.81-92.

Dietterich, T. 2000. *Ensemble Methods in Machine Learning*. Lecture Notes in Computer Science, 1857, s. 1–15.

Dursun D. & Kuzey C. & Uyar A. 2013. *Measuring firm performance using financial ratios: A decision tree approach*. Expert Systems With Applications, 40, s. 3970-3983

Gray, K. R. & Aljabar, P. & Hammers, A. et al. 2013. *Random forest-based similarity measures for multi-modal classification of Alzheimer's disease*. NeuroImage, 65, s.167-175

Ho, T. K. 1995, *Random decision forests*, Proceedings of 3rd International Conference on Document Analysis and Recognition, 1, s.278-282

Hyafil, L. & Rivest, R. L. 1976. *Constructing optimal binary decision trees is NP-complete*. Information Processing Letters, 5, s.15-17

Jia, J. & Liu, Z. & Xiao, X. et al. 2016. *pSuc-Lys: Predict lysine succinylation sites in proteins with PseAAC and ensemble random forest approach*. Journal of Theoretical Biology, 394, s.223-230

Lee, T. M. & Markowitz E. M. et al.2015 *Predictors of public climate change awareness and risk perception around the world*. Nature Climate Change, 5, s.1014-1020

Lim, T. S. & Loh, W. Y. & Shih, Y. S. 2000. *A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms*. Machine Learning, 40, s.203-228.

Maimon, O. & Rokach, L. 2002. *Top Down Induction of Decision Trees Classifiers - A Survey*. IEEE Transactions On Systems, Man and Cybernetics, PART C, 11, s. 476–487.

Maimon, O. & Rokach, L. 2008. *Data Mining with Decision Trees: Theory and Applications*. World Scientific Publishing.

Mason, L. & Baxter, J. & Bartlett, P. & Frean, M. 1999. *Boosting algorithms as gradient descent*. Proceedings of the 12th International Conference on Neural Information Processing Systems, s.512-518

Min, F. & Zhang, H. 2016. *Three-way recommender systems based on random forests*. Knowledge-based Systems, 91, s.275-286

Mingers, J. 1989. *An Empirical Comparison of Pruning Methods for Decision Tree Induction*. Machine Learning, 4, s.227-243

Murthy, S.K. 1998. *Automatic Construction of Decision Trees from Data: A Multi-Disciplinary Survey*. Data Mining and Knowledge Discovery, 2, s. 345–389.

Opitz, D. & Maclin, R. 1999. *Popular ensemble methods: an empirical study*. Journal of Artificial Intelligence Research, 11, s.169–198.

Quinlan, J. R. 1986. *Induction of Decision Trees*. Machine Learning, 1, s.81-106.

Quinlan, J. R. 1993. *C4.5: Programs For Machine Learning*. Morgan Kaufmann, Los Altos.

Quinlan, J. R. 1996. *Bagging, boosting, and C4.5*. Proceedings of the Thirteenth National Conference on Artificial Intelligence, s. 725–730.

Raileanu, L. E. & Stoffel, K. 2004. *Theoretical Comparison between the Gini Index and Information Gain Criteria*. Annals of Mathematics and Artificial Intelligence 41, s. 77-93.

Schapire, R. E. 1990. *The strength of weak learnability*. Machine Learning, 5, s.197-227

Schapire, R. E. & Freund, Y. 1997. *A decision-theoretic generalization of on-line learning and an application to boosting*. Journal of Computer and System Sciences, 55, s.119-139

Schapire, R. E. & Freund, Y. 1999. *Improved Boosting algorithms using confidence-rated predictions*. Machine Learning, 37, s.297-336

Schapire, R. E. 1999. *A brief introduction to boosting*. Proceedings of the 16th international joint conference on Artificial intelligence, 2, s.1401-1406

Shotton, J. & Cook, M. & Sharp, T. & Moore, R. et al. 2011 *Real-time human pose recognition in parts from single depth images*. Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition. s.1297-1304

Singh, K. & Guntuku, S. C. & Thakur, A. & Hota, C. 2014. *Big Data Analytics framework for Peer-to-Peer Botnet detection using Random Forests*. Information Sciences, 278, s.488-497

Solomatine, D.P. & Shrestha, D. L. 2006. *Experiments with AdaBoost.RT, an improved boosting scheme for regression*. Neural Computation, 8, s.1678-1710

Viola, P. & Jones, M. 2001. *Rapid object detection using a boosted cascade of simple features*. Proceedings of the Conference on Computer Vision and Pattern Recognition, s. 511-518

Wu, X. et al. 2008. *Top 10 Algorithms in Data Mining*. Knowledge and Information Systems, 14, s. 345–389.

Youssef, A. M. & Pourghasemi, H. R. & Pourtaghi, Z. S. & Al-Katheeri, M. M. 2016 *Landslide susceptibility mapping using random forest, boosted regression tree, classification and regression tree, and general linear models and comparison of their performance at Wadi Tayyah Basin, Asir Region, Saudi Arabia*. Landslides, 16, s.839-856

Yu, P. S. & Yiyuan, X. & Bing, L. 2000. *Clustering through decision tree construction* Proceedings of the ninth international conference on Information and knowledge management, s.20-29