ID3 JA SUMEA PÄÄTTELY

Jani Juhola

5.9.2005

Joensuun yliopisto
Tietojenkäsittelytiede
Pro gradu -tutkielma

TIIVISTELMÄ

ID3-päätöspuualgoritmi on suosittu ja melko yksinkertainen oppimisalgoritmi ja sen on todettu toimivan hyvin, kun luokiteltavana on tarkkaa, täsmällistä tietoa. ID3:n luokittelutarkkuutta voidaan parantaa erilaisilla laajennuksilla, kuten C4.5:lla. C4.5 on ID3:n laajennettu ohjelmistokokonaisuus, joka koostuu useista laajennusmahdollisuuksista, joista tutkielmassa käydään läpi tärkeimpiä: jatkuva-arvoisten attribuuttien käsittely, puuttuvien attribuuttiarvojen käsittely, päätöspuun karsiminen ja suhteutettu Gain-arvo.

Kun luokiteltavana on epävarmaa tietoa, täsmällisen ID3:n luokittelutarkkuus heikkenee. Täsmällisen ID3:n luokittelutarkkuutta voidaan parantaa sumealla päättelyllä. Tutkielmassa esitellään kirjallisuudessa esitetty UR-ID3, joka yhdistää sumean luokittelun ja ID3-päätöspuun. Tutkielmaan liittyy empiirinen osa, jossa vertaillaan ID3- ja UR-ID3-algoritmit toteuttavan SP-ID3-järjestelmän avulla ID3- ja UR-ID3-algoritmien luokittelutarkkuutta kirjallisuudessa hyvin tunnettuun Iris-aineistoon. Saadut tulokset osoittavat sumeuttamisen parantavan luokittelutarkkuutta.

Avainsanat: ID3, CLS, C4.5, sumea luokittelu, UR-ID3

SISÄLLYS

1 JOHDANTO

Päätöspuu on jonkin aiheen aineistosta suoraan generoitu kompakti luokittelurakenne, jonka tavoitteena on uusien tilanteiden mahdollisimman tarkka luokittelu [6,13]. Päätöspuuinduktiossa oppimisalgoritmille annetaan joukko esimerkkitapauksia, jotka kuvaavat jotain aihetta tai käsitettä. Esimerkkiaineiston perusteella algoritmi muodostaa säännöstön, jonka avulla voidaan sanoa kuuluuko jokin yksittäistapaus tiettyyn luokkaan vai ei. Päätöspuualgoritmit jakaantuvat erilaisiin tapoihin, joista yksi on TDIDT (top-down induction of decision trees). TDIDT-prosessissa muodostetaan päätöspuu, jonka solmut ovat attribuutteja, oksat attribuuttien arvoja ja lehdet luokkia [6]. Puun juuresta lehtisolmuun kulkemalla saadaan sääntö lehteä vastaavalle tulosluokalle. Oppimisprosessin vaativuudesta johtuen kaikkia mahdollisia puita ei käydä läpi, vaan puu muodostetaan yleensä ahneesti jonkin valintakriteerin mukaan. Näin saavutetaan laskennallisesti tehokas menetelmä, joka ei kuitenkaan takaa, että saatu päätöspuu olisi paras mahdollinen [13]. Yksi tunnetuimmista TDIDT-oppimisalgoritmeista on ID3, jonka suosio perustuu sen yksinkertaisuuteen ja tehokkuuteen. Jos aineisto ei ole ristiriitainen, luotu päätöspuu luokittelee täydellisesti koko esimerkkiaineiston [14].

Vaikka ID3:lla voidaan saavuttaa hyviä tuloksia täsmällisellä aineistolla, sen tehokkuus heikentyy huomattavasti, kun luokitteluaineisto sisältää epävarmaa, todellista tietoa [11]. Sumeus on yksi tapa mallintaa matemaattisesti reaalimaailman epävarmuutta [17]. Ihmisajattelukin voidaan mieltää sumeana luokitteluprosessina, kun yritämme päätöksiä tehdessämme vähentää tiedon liikakuormitusta. Kun perinteisessä joukko-opissa alkion kuuluvuus joukkoon on joko tosi tai epätosi, sumea joukko-oppi mahdollistaa myös osittaisen kuuluvuuden. Päätöspuuluokittelussa tämä tarkoittaa esimerkiksi sitä, että täsmällisiä lukuja ei enää käsitellä tarkkoina, vaan luvut sumeutetaan, jolloin niiden tarkkuus hämärtyy [11]. Tämä ei tarkoita kuitenkaan epävarmempaa tulosta, vaan yleensä tulos on parempi kuin täsmällisellä menetelmällä.

Tutkielma keskittyy oppimisjärjestelmän kahteen osa-alueeseen, päätöspuuinduktioon ja luokitteluun, ja erityisesti kuinka kumpaakin vaihetta voidaan parantaa erilaisilla laajennuksilla. Luvussa 2 esitellään perinteinen täsmällinen ID3-algoritmi ja sitä edeltävä CLS-algoritmi. Lisäksi luvussa esitellään ID3:n tunnetuin laajennus C4.5, josta käydään

läpi yleisimmät laajennusmenetelmät. Luvussa 3 tarkastellaan sumean päättelyn tuomia etuja luokittelujärjestelmässä. Luvussa esitellään UR-ID3, jolla voidaan käyttää sumeutta hyväksi ID3-päätöspuuluokittelussa. Luvussa 4 vertaillaan sumean ja täsmällisen ID3:n luokittelutarkkuuksia. ID3-toteutuksena käytetään Juholan [8] SP-ID3-järjestelmää. Luvussa 5 on tutkielman yhteenveto.

2 ID3-PÄÄTÖSPUUALGORITMI

Induktioluokittelun perimmäisenä ajatuksena on löytää aineistosta piilotetut lait ja säännöt, joiden mukaan luodaan malli uusien tapausten luokitteluun [6]. Päätöspuuluokittelun tavoitteena olisi löytää minimaalinen päätöspuu, joka luokittelisi testiaineiston mahdollisimman hyvin. Minimaalinen päätöspuu tarkoittaisi tässä puuta, jolla on vähän lehtisolmuja. Kaikkien esimerkkiaineiston päätöspuiden tutkiminen ei ole järkevää, koska kyseessä on NP-täydellinen ongelma [13]. Esimerkiksi pienestä esimerkkiaineistosta, jolla on viisi attribuuttia ja vain 20 esimerkkitapausta, voidaan muodostaa enemmän kuin 10^6 erilaista päästöpuuta (riippuen siitä kuinka monta arvoa eri attribuuteilla on). Ongelman vaikeuden vuoksi, monien päätöspuualgoritmien, kuten ID3, perustana on ahne, ei-peruuttava menetelmä [9].

Keskeisin ongelma päätöspuun luonnissa on attribuutin valinta puun solmuksi. ID3:ssa solmuksi asetettava attribuutti valitaan informaatioteorian entropiaan perustuvalla menetelmällä. Tämä ns. erotuskykyisimmän attribuutin valinta pitää puurakenteen yksinkertaisena [13]. Attribuutin valinta perustuu olettamukseen, että päätöspuun kompleksisuus liittyy vahvasti informaation määrään, jonka kyseinen attribuutti pystyy välittämään [9]. Päätöspuu, jolla on korkea luokittelutarkkuus, sisältää siis ne attribuutit, jotka myötävaikuttavat luokitteluprosessia merkittävästi [4]. Täten luokittelulle irrelevantit attribuutit eivät välttämättä sisälly ID3-pohjaiseen päätöspuuhun [14].

Päätöspuuluokittelun tavoitteena on muodostaa esimerkkiaineistosta kompakti sääntöjoukko, joka luokittelisi mahdollisimman hyvin kaikki testiaineiston tapaukset. Sääntöjoukko voi luokitella testiaineiston hyvin, mutta ei välttämättä täysin virheettömästi. Päätöspuun luokittelua ja tehokkuutta voidaan mahdollisesti vielä parantaa puun karsinnalla [13].

Luokiteltavan aineiston jokainen tapaus koostuu sen attribuuteista ja tulosluokasta. Jokaisella attribuutilla ja tulosluokalla on yksi tai useampia arvoja. Tapaus kuuluu johonkin tulosluokkaan sen attribuuttien arvojen perusteella. Alkuperäinen aineisto jaetaan kahteen osajoukkoon, esimerkkiaineistoon ja testiaineistoon. Luokiteltavalle aineistolle asetetaan useita kriteerejä [9]. Aineiston jokaisen tapauksen tulee olla määritelty samoilla attribuuteilla. Attribuutit voivat olla kategorisia tai jatkuvia, mutta ne eivät saa poiketa

tapauskohtaisesti toisistaan. Tulosluokat täytyy määritellä etukäteen ja yhdelle tapaukselle sallitaan vain yksi tulosluokka. Sekä attribuuttien ja tulosluokan arvojen täytyy olla erillisiä. Alkuperäisen aineiston esijalostus voi parantaa luokittelujärjestelmän tarkkuutta ja tehokkuutta [6]. Ennen oppimisprosessia aineisto voidaan puhdistaa *kohinasta* (noise) ja attribuuttien puuttuvat arvot käsitellä. Luokittelulle irrelevanttien attribuuttien poistaminen aineistosta yksinkertaistaa oppimisprosessia. Myös jatkuva-arvoisten attribuuttien yleistäminen tiivistää alkuperäistä aineistoa ja vähentää luokittelussa tarvittavien operaatioiden määrää [6].

ID3-päätöspuu on eniten käytetty luokittelumalli, koska sen oppimisprosessi on helppo ja tehokas [11]. ID3:n kirjainpari ID tulee sanoista Iterative Dichotomizer ja numero kolme tarkoittaa yksinkertaisesti versiota 3 [15].

2.1 Päätöspuuinduktio

Päätöspuun rakenteessa jokainen sisäsolmu edustaa jotain aineiston attribuuttia ja solmusta lähtevät oksat esittävät attribuutin arvoja. Lehtisolmut ilmentävät tulosluokkia eli päätöksiä. Tässä esiteltävät päätöspuualgoritmit, CLS ja ID3, ovat ns. TDIDT-menetelmiä (Top Down Induction of Decision Tree), jotka rakentavat päätöspuuta rekursiivisesti juuresta lehtisolmuihin, jakaen samalla esimerkkiaineistoa pienempiin osajoukkoihin [6]. Jokainen jako suoritetaan erotuskykyisimmän attribuutin arvojen mukaan. Prosessin tavoitteena on, että jossakin vaiheessa saavutetaan tilanne, jolloin alijoukon kaikki tapaukset kuuluisivat samaan tulosluokkaan.

2.1.1 CLS

ID3:n alkuperäinen idea perustuu Huntin et al. [7] menetelmään CLS (Concept Learning System). Hunt ja hänen ryhmänsä olivat yksi ensimmäisistä, jotka tutkivat käsitteellisen oppimisjärjestelmän rakentamista aineistosta. Päätöspuu rakennetaan CLS-algoritmilla esimerkkiaineistosta *S* kuvan 1 mukaisesti, missä tulosluokka $C_i \in \{C_1, C_2\}$ [7]:

Luo_CLS (*S*: esimerkkitapausten joukko)

Jos *S* sisältää vain yhden tai useamman tapauksen, jotka kuuluvat samaan luokkaan C_i **niin**

Palauta luo päätöspuulle lehtisolmu nimiöllä C_i

Jos *S* ei sisällä yhtään tapausta **niin**

Palauta luo lehtisolmu ja aseta nimiöksi mielivaltaisesti toinen tulosluokista

Valitse joukosta *S* erotuskykyisin attribuutti *A*

Jokaiselle attribuutin *A* arvolle a_i

Luo oksa solmusta *A* arvolla a_i

Olkoon s_i esimerkkitapausten alijoukko, joilla attribuutti $A=a_i$

$s_i = s_i - A$

Luo_CLS (s_i)

Kuva 1. CLS-päätöspuualgoritmi.

Esimerkkiaineisto *S* jaetaan siis osajoukkoihin $\{s_1,s_2,...,s_v\}$ erotuskykyisimmän attribuutin *A* arvojen $\{a_1,a_2,...,a_v\}$ mukaan. Attribuutin *A* valintaan käytetään jotain heuristista kriteeriä. Jokaisen oksan alipuu luodaan alijoukon s_i mukaan, joka sisältää kaikki aineiston *S* tapaukset, joilla attribuutti *A* sai arvon a_i. Päätöspuu saa siis solmun *A*, jolla on *v* kpl oksia arvoilla $\{a_1,a_2,...,a_v\}$. Tämän jälkeen sovelletaan prosessia rekursiivisesti jokaiselle alijoukolle s_i.

2.1.2 ID3-päätöspuuinduktio

ID3–algoritmin tärkein prosessi on päätöspuun johtaminen esimerkkiaineistosta. CLS on rajoitettu vain kahteen tulosluokkaan, mutta ID3 mahdollistaa useiden tulosluokkien käsittelyn. Lisäksi ID3:ssa erotuskykyisimmän attribuutin valinta suoritetaan informaatioteoriaan perustuvalla menetelmällä (katso kohta 2.1.3). Kuvassa 2 esitettävä ID3-päätöspuuinduktion algoritmi perustuu osittain Hanin ja Kamberin [6] esittämään versioon Quinlanin ID3:sta:

Luo_ID3 (*S*: esimerkkitapausten joukko)

Jos joukon *S* esimerkkitapaukset kuuluvat samaan luokkaan C_i **niin**

// Lopetusehto 1

Palauta luo lehtisolmu nimiöllä C_i

Jos joukossa *S* on vain yksi attribuutti, jolla yksi arvo **niin**

// Lopetusehto 2

Palauta luo lehtisolmu ja nimeä se luokan C_i mukaan, jolla on enemmistö esimerkkiaineistossa

Valitse joukosta *S* erotuskykyisin attribuutti *A*

Luo uusi solmu nimellä *A*

Jokaiselle attribuutin *A* arvolle a_i

Luo oksa solmusta *A* arvolla a_i

Olkoon s_i esimerkkitapausten alijoukko, joilla attribuutti $A = a_i$

$s_i = s_i - A$

Jos joukko s_i on tyhjä **niin**

// Lopetusehto 3

Luo lehtisolmu ja nimeä se luokan C_i mukaan, jolla on enemmistö koko alkuperäisestä esimerkkiaineistosta

muuten

Luo_ID3 (s_i)

Kuva 2. ID3-algoritmin päätöspuuinduktio.

Rekursion kolme lopetusehtoa ovat seuraavat [6]:

1: Kaikki esimerkkitapaukset kuuluvat samaan tulosluokkaan, jolloin päätöksiä voi olla vain yksi.

2: Alijoukossa on jäljellä vain yksi attribuutti, jolla on vain yksi arvo. Lopetusehdon 1 perusteella voidaan myös olettaa, että attribuutin tapahtumat eivät kuulu samaan tulosluokkaan. Koska alijoukkoa ei voida enää jakaa, päivitetään puuhun lehtisolmu, jonka päätökseksi tulee tulosluokka, jonka esiintymiä on alijoukossa eniten.

3: Erotuskykyisimmän attribuutin arvolla ei ole enää esimerkkitapauksia jäljellä, jolloin lehtisolmun päätökseksi tulee se tulosluokka, jolla on eniten esiintymiä koko esimerkkiaineistosta.

Lopetusehdot 2 ja 3 perustuvat siis Hanin ja Kamberin päätöspuuinduktioon. Lopetusehto 2 saavutetaan, kun aineisto sisältää tapauksia, joiden attribuuttien arvot ovat täysin samat, mutta jotka kuuluvat eri luokkiin. Quinlanin alkuperäisessä ID3:ssa [14] luokat valitaan satunnaisesti kummassakin lopetusehdossa 2 ja 3. Quinlanin C4.5 [13] valitsee päätösluokan ehdossa 2 samalla tavalla kuin Han ja Kamber. Lopetusehdolla 3 C4.5 valitsee päätökseksi sen luokan, jolla on enemmistö puun edellisellä tasolla.

Kun päätöspuu on valmis, muodostetaan sen kaikista poluista sääntöjoukko, jonka avulla testiaineisto luokitellaan [6]. Päätöspuun juurisolmusta lehtisolmuun kulkevaa polkua sanotaan säännöksi. Säännöt esitetään JOS-NIIN (IF-THEN) muodossa: polun jokaisen attribuutin arvot muodostavat säännön JOS-osan ja lehtisolmun päätösennusteesta muodostetaan NIIN-osa. JOS-osa määrää myös järjestyksen, missä testitapausten attribuutteja vertaillaan. Hanin ja Kamberin mukaan [6] sääntöjoukko on ihmisajattelulle helpompi kuin puurakenne varsinkin, jos päätöspuu on suurikokoinen.

2.1.3 Informaatio ja entropia

Hunt et al. [7] pohtivat CLS:ssä useita tapoja, joilla jakoattribuutti voitaisiin valita. Useimmat testit perustuivat attribuuttien ja luokkien frekvensseihin ja niiden vertailuun. Hunt et al. kuitenkin ehdottavat, että informaatioteoriaan perustuva ratkaisumalli saattaisi olla hyödyksi päätöspuuta rakennettaessa.

ID3 käyttää erotuskykyisimmän attribuutin valintaan kriteerinä *Gain-arvoa* (information gain) [13]. Attribuutti, joka omaa suurimman informaatio hyödyn, valitaan erotuskykyisimmäksi attribuutiksi, jonka arvojen mukaan muodostetaan seuraavat alijoukot. Valittu attribuutti minimoi informaation, joka tarvitaan luokittelemaan jäljellä olevat tapaukset johonkin tulosluokkaan. Tällainen informaatioteoriaan perustuva lähestymistapa minimoi testivaiheiden määrän päätöspuun juuresta lehtisolmuihin ja takaa, että luodun päätöspuun

rakenne on yksinkertainen [6]. Suure *Gain* perustuu viestin informaation määrän (entropian) määritelmään [13]:

Määritelmä 1. Viestin välittämä informaatio riippuu sen todennäköisyydestä ja se voidaan mitata bitteinä: miinus kaksikantainen logaritmi viestin todennäköisyydestä.

Entropia-käsite tarkoittaa tässä samaa kuin informaation määrä. Entropia ilmaisee epävarmuutta tulevasta. Mitä suurempi attribuutin entropia on, sitä satunnaisempaa on sen tuottama tieto. Seuraavaksi käydään läpi, kuinka informaation määrä lasketaan esimerkkiaineiston attribuutille. Kaikki seuraavat määritelmät ja notaatio perustuvat Hanin ja Kamberin kirjaan [6].

Olkoon S on esimerkkitapausten joukko ja s kaikkien esimerkkitapausten lukumäärä. Luokkia C_i on yhteensä m kappaletta. Suure *Gain(A)* kertoo informaation määrän, joka saadaan, kun esimerkkijoukko S jaetaan attribuutin A mukaan. Gain-kriteeri on taas tapahtuma, jossa valitaan attribuutti, joka maksimoi informaation määrän eli omaa suurimman Gain-arvon. Esimerkkijoukon S attribuutille A lasketaan Gain-arvo kaavalla,

$$Gain(A) = I(S) - I_A(S), \tag{1}$$

missä *I(S)* tarkoittaa esimerkkijoukon S keskimääräistä informaatiota, joka tarvitaan luokittelemaan yksi esimerkkitapaus. Voidaan myös sanoa, että *I(S)* on esimerkkijoukon S entropia ennen jakoa alijoukkoihin. $I_A(S)$ on attribuutin A entropia. Olkoon s_i esimerkkitapausten lukumäärä luokassa C_i. Tällöin esimerkkijoukon S entropia saadaan kaavalla

$$I(S) = I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i), \tag{2}$$

missä p_i on todennäköisyys, että mielivaltainen tapaus kuuluu luokkaan C_i eli s_i / s.

Olkoon attribuutilla A arvot $\{a_1, a_2, ..., a_v\}$. Attribuutin A entropia saadaan kaavalla

$$I_A(S) = \sum_{j=1}^{v} \frac{s_{1j} + ... + s_{mj}}{s} I(s_{1j}, ..., s_{mj}), \tag{3}$$

missä termi $(s_{ij}+...+s_{mj}) / s$ on luokkaan kuuluvien esimerkkitapausten lukumäärä osajoukossa jaettuna koko joukon S esimerkkien lukumäärällä s. Alijoukolle S_j entropia lasketaan kaavalla

$$I(S_{1j}, S_{2j}, \ldots, S_{mj}) = -\sum_{i=1}^{m} p_{ij} \log_2 (p_{ij}), \tag{4}$$

missä $p_{ij} = S_{ij} / |S_j|$ eli todennäköisyys, että esimerkkitapaus alijoukossa S_j kuuluu luokkaan C_i. Entropia sijoittuu aina välille nollasta yhteen, jossa $I_a(A)=0$ tarkoittaisi, että joukko S on luokiteltu täydellisesti ts. kaikki esimerkkijoukon tapaukset kuuluvat samaan luokkaan. $I_a(A)=1$ tarkoittaisi, että tieto on täysin satunaista.

Kun koko esimerkkijoukon S entropiasta vähennetään attribuutin A entropia, saadaan $Gain(A)$ eli informaation määrä, kun joukko S jaetaan attribuutin A mukaan. Gain-kriteerissä erotuskykyisimmäksi attribuutiksi valitaan se, jonka Gain-arvo on suurin. Jos attribuutti A on erotuskykyisin, jaetaan joukko S alijoukkoihin $\{S_1,S_2,...,S_v\}$, missä S_j sisältää ne tapaukset joilla A sai arvon a_j.

Erotuskykyisimmän attribuutin valintaan on olemassa kaksi näkökulmaa. Jos halutaan mieltää valittava attribuutti sellaisena, jolla saavutettaisiin suurin informaatiohyöty luokittelussa, valitaan attribuutti Gain-kriteerin perusteella. Toisaalta koska entropia ilmaisee epävarmuutta tulevasta (mitä suurempi entropia sitä suurempi epävarmuus tulevasta), voitaisiin erotuskykyisimmäksi attribuutiksi valita yhtä hyvin pienimmän entropian tuottava attribuutti.

Vaikka Gain-kriteerin käytön on todettu tuottavan suppeita päätöspuita [13], on sillä Quinlanin mukaan yksi vakava puute: Gain-kriteeri suosii puolueellisesti attribuutteja, joilla on monta arvoa. Moniarvoiset attribuutit pilkkoisivat aineiston suureen määrään pieniä alijoukkoja, jotka olisivat melko tasarakenteisia, mutta ennustavan luokittelun näkökannalta melko tarpeettomia. Eli Gain-kriteeriä käyttämällä päätöspuista tulee matalia, mutta puu saattaa sisältää ylisovittamista. Selvittääkseen tämän ongelman Quinlan esittää C4.5:ssa suhteutetun Gain-arvon (Gain-ratio), jossa Gain-kriteeri normalisoidaan.

2.1.4 Esimerkki ID3-päätöspuuinduktiosta

Tässä kohdassa käydään läpi esimerkein kaikki edellä esitetyt laskukaavat ja lyhyesti myös päätöspuun muodostus. Aineisto ja kaikki seuraavat esimerkkilaskut perustuvat Kantardizicin kirjaan [9]. Taulukon 1 aineistossa on 14 tapausta, jotka koostuvat kolmesta attribuutista (*A*, *B* ja *C*) ja tulosluokasta. Attribuutteja *A* ja *C* käsitellään kategorisina ja attribuuttia *B* jatkuva-arvoisena (ks. kohta 2.3.1). Tulosluokkia on tässä esimerkissä vain kaksi.

Päätöspuun muodostus aloitetaan erotuskykyisimmän attribuutin valinnalla, koska kaikki esimerkkitapaukset eivät kuulu samaan tulosluokkaan. Jokaiselle attribuutille lasketaan sen tuottama informaation määrä eli Gain-arvo. Lasketaan esimerkkinä *Gain(A)*.

Taulukko 1. Esimerkkiaineisto S.

A	B	C	Tulosluokka
1	70	TRUE	CLASS1
1	90	TRUE	CLASS2
1	85	FALSE	CLASS2
1	95	FALSE	CLASS2
1	70	FALSE	CLASS1
2	90	TRUE	CLASS1
2	78	FALSE	CLASS1
2	65	TRUE	CLASS1
2	75	FALSE	CLASS1
3	80	TRUE	CLASS2
3	70	TRUE	CLASS2
3	80	FALSE	CLASS1
3	80	FALSE	CLASS1
3	96	FALSE	CLASS1

Taulukko 2. Attribuutin A arvojen muodostamat frekvenssit.

	CLASS1	CLASS2	Yhteensä
A=1	2	3	5
A=2	4	0	4
A=3	3	2	5
Yhteensä	9	5	14

Kun aineisto jaetaan attribuutin *A* arvojen (1, 2 ja 3) mukaan, saadaan taulukossa 2 näkyvät frekvenssit. Tulosluokkia on kaksi, joista luokkaan *CLASS1* kuuluu yhdeksän tapausta ja luokkaan *CLASS2* viisi. Täten joukon *S* koko entropia *I(S)* ennen jakoa on

$$I(S) = -9/14\ log_2(9/14) - 5/14\ log_2(5/14) = 0.940\ bittiä$$

Vastaavasti *A*:n arvojen *1, 2* ja *3* mukaan laskettu entropia on

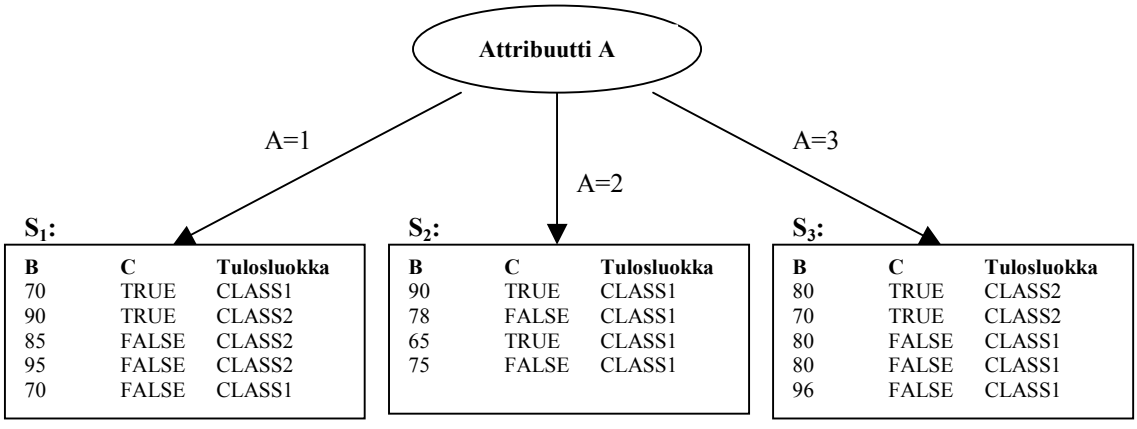
$$I_A(S) = 5/14\ (\ -2/5\ log_2(2/5) - 3/5\ log_2(3/5)) +$$
$$4/14\ (\ -4/4\ log_2(4/4) - 0/4\ log_2(0/4)) +$$
$$5/14\ (\ -3/5\ log_2(3/5) - 2/5\ log_2(2/5)) = 0.694\ bittiä$$

Informaation määrä, *Gain(A)*, joka saadaan, kun jaetaan aineisto *S* attribuutin *A* mukaan on

$$Gain(A) = I(S) - I_A(S) = 0.940 - 0.694 = 0.246$$

Kun informaation määrä lasketaan attribuuteille *B* ja *C*, saadaan *Gain(B)* = 0.103 *bittiä* ja *Gain(C)* = 0.048 *bittiä*. Attribuutti *B*:n Gain-arvo on laskettu ID3:n laajennuksella C4.5, joka mahdollistaa jatkuva-arvoisten attribuuttien käsittelyn. Laajennus esitellään kohdassa 2.3.1.

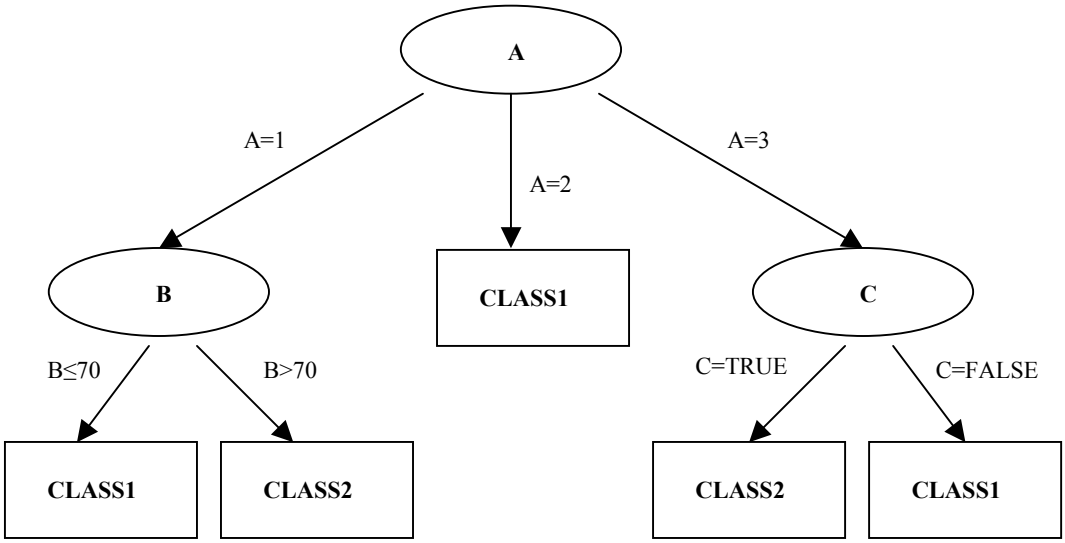
Gain-kriteeri valitsee attribuutin *A* erotuskykyisimmäksi, koska sen tuottamalla jaolla saavutetaan suurin informaation määrä. Puun ensimmäinen taso muodostetaan attribuutin *A* arvojen mukaan jakamalla aineisto *S* alijoukkoihin (alipuihin) s_1, s_2, ja s_3 (kuva 3).



Kuva 3. Päätöspuu, kun aineisto jaetaan attribuutin A mukaan.

Kuten kuvasta 3 näkyy, alipuun s_2 kaikki tapaukset kuuluvat samaan tulosluokkaan, joten alipuusta tulee lehtisolmu nimiöllä (päätöksellä) *CLASS1*. Sen sijaan alipuut s_1 ja s_3 on jaettava edelleen erillisiin alijoukkoihin valitsemalla erotuskykyisempi attribuutti eli joko

B tai *C*. Kuvassa 4 näkyy aineiston lopullinen päätöspuu. Päätöspuuta vastaava sääntöjoukko näkyy kuvassa 5.



Kuva 4. Lopullinen päätöspuu taulukon 1 aineistosta.

Jos Attribuutti A = 1 *ja* Attribuutti B <= 70 ***niin***	Luokittelu = CLASS1;
Jos Attribuutti A = 1 *ja* Attribuutti B >70 ***niin***	Luokittelu = CLASS2;
Jos Attribuutti A = 2 ***niin***	Luokittelu = CLASS1;
Jos Attribuutti A = 3 *ja* Attribuutti C=TRUE ***niin***	Luokittelu = CLASS2;
Jos Attribuutti A = 3 *ja* Attribuutti C=FALSE ***niin***	Luokittelu = CLASS1.

Kuva 5. Päätöspuuta (kuva 4) vastaava sääntöjoukko.

2.2 ID3-algoritmi

Alkuperäiseen Quinlanin ID3-algoritmiin [14] kuuluu iteratiivinen *ikkunointi* (windowing), joka mahdollistaa täydellisesti luokittelevan päätöspuun muodostamisen suurista aineistoista. Ikkunoinnissa esimerkkiaineistoon lisätään aina ne tapaukset koko alkuperäisestä aineistosta, jotka päätöspuu luokitteli väärin. Tämän jälkeen päivitetystä esimerkkiaineistosta luodaan uusi päätöspuu, joka testataan jälleen alkuperäisellä aineistolla. Prosessia jatketaan, kunnes päätöspuun luokittelussa ei esiinny enää virheitä. Yleensä oikea päätöspuu löydetään neljän kierroksen jälkeen [14]. Kun perus päätöspuuinduktio (kuva 2) ja ikkunointi yhdistetään, saadaan kuvan 6 esittämä ID3-algoritmi [14]:

Valitse satunnainen alijoukko alkuperäisestä aineistosta esimerkkiaineistoksi eli ikkunointi

Toista

Suorita päätöspuuinduktio esimerkkiaineistosta (kuvan 2 algoritmilla)

Testaa päätöspuu alkuperäisellä aineistolla ja lisää esimerkkiaineistoon tapaukset, jotka luokiteltiin väärin

kunnes ei löydy enää väärin luokiteltuja tapauksia

Kuva 6. ID3-algoritmi.

Quinlan [14] esittää myös esimerkkiaineiston päivitykseen toisen tavan, jossa ikkunan koko pidetään kiinteänä. Menetelmässä yritetään etsiä esimerkkiaineistosta ne tapaukset, jotka ovat tärkeitä luokittelulle. ”Avaintapaukset” jätetään esimerkkiaineistoon ja loput korvataan alkuperäisen aineiston väärinluokitetuilla tapauksilla.

2.3 C4.5 laajennus

ID3 on saanut lukuisia laajennuksia, joista Quinlanin itse kehittämä C4.5 on ehkä tunnetuin. Algoritmin C4.5 perustana on yhä ID3, mutta se laajentaa luokitteluprosessia mm. seuraavilla menetelmillä [13]:

- suhteutetun Gain-arvon (Gain-ratio) käyttö Gain-kriteerin sijasta
- ylisovittamisen välttäminen
- jatkuva-arvoisten (numeeristen) attribuutien käsittely (ID3:ssa vain kategorisia attribuutteja)
- puuttuvien attribuuttiarvojen käsittely
- tehokkuuden parantaminen aineiston ikkunoinnilla (laajennettu versio ID3:n ikkunoinnista)
- päätöspuun karsiminen
- sääntöjen jälkikarsinta

C4.5 laajentaa ja parantaa ID3:sta monella tavalla, mutta samalla vaikeuttaa itse luokittelusysteemin rakentamista ja voi jopa sisältää järjestelmälle turhia laajennuksia.

Systeemin vaatimuksista riippuen voidaan perinteiseen ja helppotajuiseen ID3:een ottaa vain osa C4.5 tarjoamista laajennuksista.

2.3.1 Jatkuvien attribuuttien käsittely

Aineisto saattaa sisältää attribuutteja, joiden arvojen käsittely erillisinä johtaisi suuriin päätöspuihin. Tästä tilanteesta päädytään ongelmaan, jossa täytyy päättää, kuinka moneen osaan jatkuvat arvot jaetaan ja mihin kohtaan rajat asetetaan. Algoritmi C4.5 liittää ID3:een algoritmin, jolla etsitään jatkuva-arvoisen attribuutin optimaaliset jakokohdat [13]. Jatkuvien attribuuttien käsittely suoritetaan seuraavasti [9]:

Olkoon attribuutin A arvot suuruusjärjestyksessä $\{v_1,v_2,\ldots,v_m\}$. Mahdollisia kynnysarvoja ovat $(v_i+v_{i+1})/2$, kun $i=1..m-1$. C4.5 valitsee kynnysarvoksi jokaisen välin keskikohtaa pienemmän arvon v_i. Tällä varmistetaan, että kaikki päätöspuuhun tulevat kynnysarvot todella esiintyvät aineistossa (täten mahdollisia jakokohtia on $m-1$ kappaletta). Kynnysarvo v_i jakaa attribuutin A arvot edelleen kahteen osajoukkoon $\{v_1,v_2,\ldots,v_i\}$ ja $\{v_{i+1},v_{i+2},\ldots,v_m\}$. Jatkuva-arvoisten attribuuttien kynnysten etsintä suoritetaan erotuskykyisimmän attribuutin valinnan aikana. Kun kategorisille attribuuteille lasketaan kullekin yksi Gain-arvo, jatkuva-arvoiselle attribuutille A lasketaan informaation määrä jokaiselle kynnysarvon luomalle jaolle ($attribuutiA \leq v_j$ *tai* $attribuuttiA > v_j$). Näistä kynnysarvoista valitaan se, jolla saavutetaan suurin informaation määrä ja verrataan edelleen sitä esimerkkiaineiston muiden attribuuttien Gain-arvoihin. Jos joku muu attribuutti valitaan erotuskykyisimmäksi, ei A:ta ja esimerkkiaineistoa enää jaeta v_j:n mukaan. Uuden mahdollisen jakokynnyksen etsintä suoritetaan taas puun seuraavalla tasolla.

Täydennetään kohdan 2.1.4 esimerkkiä [9] ja lasketaan Gain-arvo jatkuva-arvoiselle attribuutille B, jolla on järjestetyt arvot {65, 70, 75, 78, 80, 85, 90, 95, 96}. Mahdollisten kynnysarvojen joukko Z on {65, 70, 75, 78, 80, 85, 90, 95}. Jokaiselle Z:n arvon muodostamalle jaolle lasketaan Gain-arvo samalla tavalla kuin kategorisilla attribuuteille (poikkeuksena, että ”kategorioita” on aina vain kaksi kappaletta eli esimerkiksi jako Z:n ensimmäisellä alkiolla olisi $attribuuttiA \leq 65$ *tai* $attribuutti\ A > 65$). Päätöspuun ensimmäisellä tasolla attribuutin B optimaalisin kynnysarvo on 80, jolla saavutetaan suurin informaation määrä. Jako $Gain(attribuuttiC \leq 80$ *tai* $attribuuttiC > 80)$ (tai lyhyesti

Gain(C)) otetaan edelleen vertailuun ehdokkaaksi erotuskykyisimmän attribuutin valintaan aineiston muista attribuuteista. *Gain(C)*:n laskenta suoritetaan seuraavasti

$$I_C(S) = 9/14(-7/9\ log_2(7/9)-2/9log_2(2/9))+$$
$$5/14(-2/5\ log_2(2/5)-3/5log_2(3/5))$$
$$= 0.837\ bittiä$$
$$Gain(C) = 0.940 - 0.837 = 0.103\ bittiä$$

Vertailussa muihin attribuutteihin, erotuskykyisimmäksi päätöspuun juureen tulee attribuutti *A*, koska sillä on suurin Gain-arvo (ks kohta 2.1.4). Päätöspuun seuraavalla tasolla attribuutti *B* on vasemman alipuun erotuskykyisin ja optimaalisin jakokynnys alipuussa on 70 (ks. kuva 4).

2.3.2 Suhteutettu Gain-arvo (Gain-ratio)

Vaikka Gain-kriteerin käytön on todettu tuottavan suppeita päätöspuita, on sillä Quinlanin [13] mukaan yksi vakava puute: Gain-kriteeri suosii puolueellisesti attribuutteja, joilla on monta arvoa. Moniarvoiset attribuutit pilkkovat aineiston suureen määrään pieniä alipuita, jotka olisivat melko tasarakenteisia, mutta saattavat sisältää ylisovittamista ja ovat luokittelun kannalta melko tarpeettomia [13]. Selvittääkseen tämän ongelman Quinlan esittää C4.5:ssa suhteutetun Gain-kriteerin (Gain-ratio), joka on hänen testeissä tuottanut suppeampia päätöspuita ja on siten parempi jakokriteeri kuin tavallinen Gain-kriteeri. Suhteutettu Gain-arvo, *Gain ratio(A)*, saadaan kun attribuutin *A* tavallinen *Gain(A)* normalisoidaan jakamalla se jakoinformaatiolla,

$$Gain\ ratio(A) = Gain(A) / Split\ info(A), \quad (5)$$

missä attribuutin *A* jakoinformaatio, *Split info(A),* saadaan kaavalla

$$Split\ info(A) = -\sum_{i=1}^{n} \frac{|S_i|}{|S|} \times \log_2\left(\frac{|S_i|}{|S|}\right). \quad (6)$$

Jakoinformaatio edustaa mahdollista informaation määrää, joka saadaan, kun esimerkki-joukko *S* jaetaan alijoukkoihin S_i, joita on n kappaletta. Jos jaolla on vain vähän merkitystä,

jakoinformaatio on pieni ja Gain-ratiosta tulee epävarma. Tämän välttämiseksi erotuskykyisimmäksi attribuutiksi valitaan se, jolla on suurin Gain-ratio. Suhteutettu Gain-arvo esittää hyödyllisen informaation osuutta, joka saavutetaan attribuutin arvojen mukaisella jaolla.

Laajennetaan kohdan 2.1.4 päätöspuuesimerkkiä ja lasketaan suhteutettu Gain-arvo attribuutille *A*. Jakoinformaatio attribuutille *A* on

$$\begin{aligned} \textit{Split info(A)} &= -5/14\ log_2(5/14) - 4/14\ log_2(4/14) - 5/14\ log_2(5/14) \\ &= 1.577\ \textit{bittiä} \end{aligned}$$

Suhteutettu *Gain-ratio(A)* saadaan, kun jaetaan *Gain(A)* = 0.246 jakoinformaatiolla

$$\textit{Gain-ratio(A)} = 0.246 / 1.557 = 0.156$$

Suhteutettu Gain-ratio lasketaan myös muille attribuuteille ja erotuskykyisimmäksi attribuutiksi valitaan se, jonka Gain-ratio on suurin.

2.3.3 Puuttuvien attribuuttiarvojen käsittely

Reaalimaailmasta kerätty aineisto saattaa olla usein epätäydellistä. ID3-algoritmin yhtenä oletuksena on, että luokiteltava aineisto on täysin määritelty ja tieto on varmaa, niinpä epätäydelliset tapaukset täytyy joko poistaa tai puuttuvat arvot täytyy korvata jollain menetelmällä. C4.5 sisältää laajennuksen, jolla attribuutin puuttuvat arvot voidaan käsitellä jo päätöspuun rakennusvaiheessa [13]. Menetelmässä tapaukset, joilla on tuntemattomia arvoja, jaetaan osatapauksina jokaiseen alijoukkoon. Tapaus jaetaan osatapauksiksi vasta sitten, kun puuttuvan arvon omaava attribuutti on valittu erotuskykyisimmäksi. Lisäksi menetelmässä tapaukset painotetaan s. e. normaalin (tunnetun) tapauksen painoarvo on yksi, mutta kuhunkin alijoukkoon lisätyn tuntematonta arvoa vastaavan tapauksen painoarvo on alijoukon tapausten suhde kaikkiin tunnettuihin tapauksiin.

Puuttuvien arvojen käsittely muuttaa informaation määrien laskemista:

$$Gain\ (A) = F \times (\ I\ (S) - I_A(S)), \qquad (7)$$

missä F ilmaisee todennäköisyyden, jolloin attribuutti A on tunnettu. F on siis murtoluku, joka saadaan, kun attribuutin A mukaan määräytyvän alijoukon tunnettujen tapausten lukumäärä jaetaan alijoukon kaikkien tapausten lukumäärällä. Myös jakoinformaation (*Split info(A)*) laskenta muuttuu. Jos attribuutilla on n kpl arvoja, jakoinformaatio lasketaan $n+1$ alijoukolle. Koska kategorioita on yksi enemmän, tarkoittaa se käytännössä, että jakoinformaation määrä kasvaa verrattuna täysin määritellyn aineiston jakoinformaatioon.

Päätöspuun jokaiselle lehdelle lasketaan painoarvojen lukupari, (N / E), missä N on niiden esimerkkiaineiston osatapausten painoarvojen summa, jotka kuuluvat lehtisolmuun ja E on niiden osatapausten painoarvojen summa N:stä, jotka kuuluvat johonkin muuhun luokkaan kuin lehtisolmun päätösluokkaan.

Seuraavaksi esitettävän esimerkin aineisto ja laskut perustuvat osittain Kantardizicin kirjaan [9] ja Quinlanin teokseen [13].

Taulukko 3. Esimerkkiaineisto S.

A	B	C	Tulosluokka
1	70	TRUE	CLASS1
1	90	TRUE	CLASS2
1	85	FALSE	CLASS2
1	95	FALSE	CLASS2
1	70	FALSE	CLASS1
?	90	TRUE	CLASS1
2	78	FALSE	CLASS1
2	65	TRUE	CLASS1
2	75	FALSE	CLASS1
3	80	TRUE	CLASS2
3	70	TRUE	CLASS2
3	80	FALSE	CLASS1
3	80	FALSE	CLASS1
3	96	FALSE	CLASS1

Taulukko 4. Attribuutin A arvojen muodostamat frekvenssit.

	CLASS1	CLASS2	Yhteensä
A=1	2	3	5
A=2	3	0	3
A=3	3	2	5
Yhteensä	8	5	13

Attribuutin *A* Gain-arvo lasketaan muuten kuten Gain(A), mutta tapausta, jolla ei ole attribuutin arvoa, ei oteta huomioon Gain-arvon laskentaan. Kun taulukon 3 aineisto jaetaan attribuutin A tunnettujen arvojen mukaan, saadaan taulukossa 4 näkyvät frekvenssit. Tulosluokkaan *CLASS1* kuuluu nyt kahdeksan tapausta ja luokkaan *CLASS2* viisi, jolloin joukon *S* koko entropia *I(S)* ennen alijoukkoihin jakoa on

$$I(S) = -8/13\ log_2(8/13) - 5/13\ log_2(5/13)$$
$$= 0.961\ bittiä.$$

Vastaavasti *A*:n arvojen (1, 2 ja 3) mukaan laskettu entropia on

$$I_A(S) = 5/13\ (\ -2/5\ log_2(2/5) - 3/5\ log_i(3/5))+$$
$$3/13\ (\ -3/3\ log_2(3/3) - 0/3\ log_2(0/3))+$$
$$5/13\ (\ -3/5\ log_2(3/5) - 2/5\ log_2(2/5))$$
$$= 0.747\ bittiä.$$

Informaation määrä, *Gain(A)*, kerrotaan nyt murtoluvulla *F*. Koska taulukon 3 aineistossa oli vain yksi epätäydellinen tapaus, on $F = 13/14$.

$$Gain(A) = F \times (\ I(S) - I_A(S)) = 13/14 \times (\ 0.961 - 0.747\ ) = 0.199\ bittiä.$$

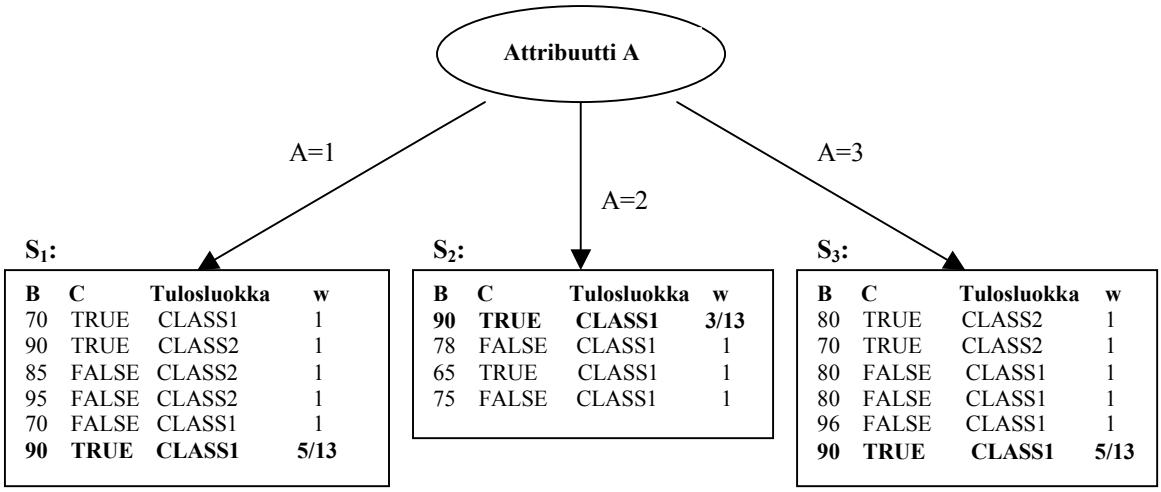
Jakoinformaatio lasketaan koko aineistosta ja epätäydellinen tapaus otetaan mukaan uutena kategoriana:

$$Split\ info(A) = -5/14\ log_2(5/14)\ \ (A\ arvolla\ 1)$$
$$-3/14\ log_2(3/14)\ \ (A\ arvolla\ 2)$$
$$-5/14\ log_2(5/14)\ \ (A\ arvolla\ 3)$$
$$-1/14\ log_2(1/14)\ \ (A\ arvolla\ ?)$$
$$= 1.809\ bittiä.$$

Suhteutettu *Gain-ratio(A)* saadaan, kun jaetaan *Gain(A)* =0.246 jakoinformaatiolla

$$Gain\text{-}ratio(A) = Gain(A) / Split\ info(A)$$
$$= 0.199 / 1.809 = 0.110.$$

Kun aineisto jaetaan attribuutin *A* arvojen mukaan alijoukkoihin, täydelliset 13 tapausta jaetaan normaalisti. Epätäydellinen tapaus sisällytetään jokaiseen alijoukkoon arvojen mukaisilla painoarvoilla 5/13, 3/13 ja 5/13 (kuva 7).

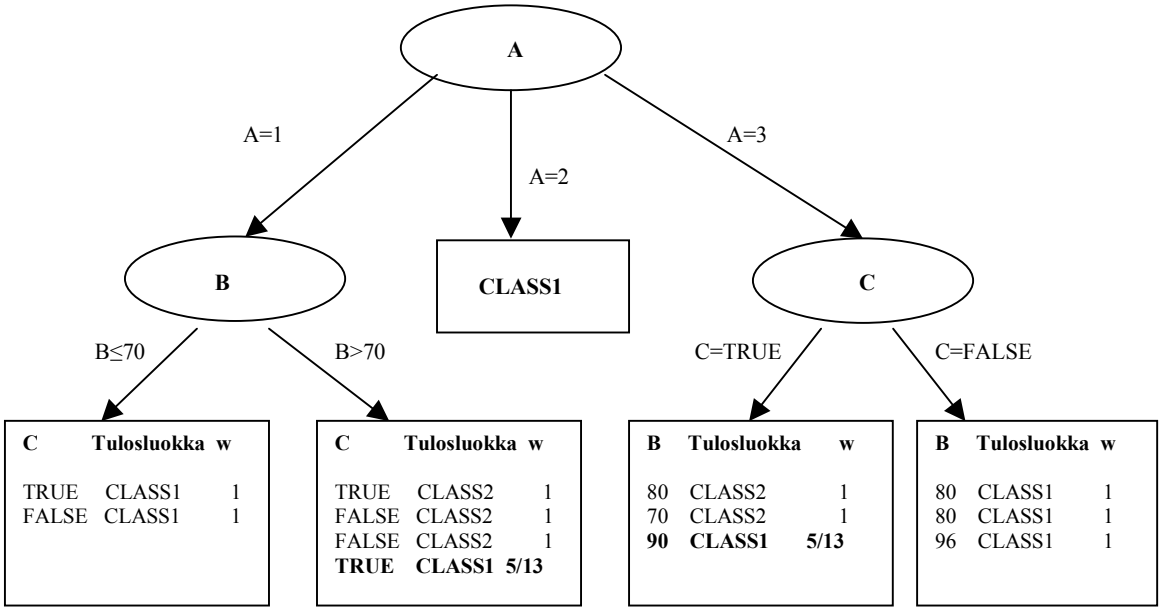


Kuva 7. Attribuutin *A* arvojen mukaiset alipuut.

Kun joukkoa s_1 osioidaan edelleen alijoukkoihin (kuva 8), valitaan erotuskykyisimmäksi attribuutiksi *B* ja jako suoritetaan arvon 70 mukaan, jolloin alijoukkojen luokkajakaumat ovat

$B \leq 70$: 2 tapausta luokassa *CLASS1* ja 0 tapausta luokassa *CLASS2;*

$B > 70$: 5/13 tapausta luokassa *CLASS1* ja 3 tapausta luokassa *CLASS2.*



Kuva 8. Päätöspuu taulukon 3 esimerkkiaineistosta.

Ensimmäinen alijoukko koostuu vain kahdesta luokan *CLASS1* tapauksesta, mutta toisen alijoukon tapaukset kuuluvat yhä molempiin luokkiin. Tapauksia ei kuitenkaan voi jakaa enää alijoukkoihin. Sama tilanne kohdataan joukon s_3 partitioinnissa; tapauksia ei voida jakaa yhteen tulosluokkaan kuuluviin alijoukkoihin. Jokaiseen lehtisolmuun lasketaan vielä painoarvopari. Esimerkiksi *CLASS2* (3.4 / 0.4) tarkoittaa, että koko lehtisolmun painoarvo on 3.4 (tai 3 + 5/13), josta painoarvolla 0.4 osatapausta ei kuulunut luokkaan *CLASS2*. Kuvan 8 päätöspuun mukainen lopullinen sääntöjoukko on

Jos $A = 1$ ja $B \leq 70$	***niin***	Luokittelu = *CLASS1* (2.0 / 0);
Jos $A = 1$ ja $B > 70$	***niin***	Luokittelu = *CLASS2* (3.4 / 0.4);
Jos $A = 2$	***niin***	Luokittelu = *CLASS1* (3.2 / 0);
Jos $A = 3$ *ja* C = TRUE	***niin***	Luokittelu = *CLASS2* (2.4 / 0.4);
Jos $A = 3$ *ja* C = FALSE	***niin***	Luokittelu = *CLASS1* (3.0 / 0).

Kun päätöspuulla luokitellaan testitapaus, jonka attribuutin *A* arvo on *1* ja attribuutin *B* arvo on tuntematon, luokittelu siirtyy heti ensimmäisen säännön mukaan attribuutin *B* tarkasteluun. Koska testitapauksella ei ole *B*:n arvoa, ei päätöstä voida määritellä suoraan. Voidaan kuitenkin päätellä, että jos tapauksen *B* arvo on pienempi tai yhtä suuri kuin 70, olisi luokittelun päätös *CLASS1*. Jos tapauksen *B* > 70, kuuluisi testitapaus todennäköisyydellä 88 % (=100*3/3.4) luokkaan *CLASS2* ja 12 % (=100*0.4/3.4) todennäköisyydellä luokkaan *CLASS1*. Kun päätöspuu rakennettiin, nämä osiot sisälsivät 2.0 ja 3.4 tapausta. Kun ehdolliset päätökset yhdistetään suhteellisilla painoilla, 2.0/5.4 ja 3.4/5.4, testitapauksen lopullinen luokkajakauma on

CLASS1:	2.0 / 5.4 * 100 % + 3.4 / 5.4 * 12 % = 44 %
CLASS2:	3.4 / 5.4 * 88 % = **56 %**

2.3.4 Puun karsiminen

Jos esimerkkiaineisto sisältää virheellistä tietoa, päätöspuun jotkin haarat muotoutuvat näiden virheiden mukaan. Samoin, jos esimerkkitapauksia on vähän, päätöspuu saattaa luokitella vain joitakin aineiston erikoistapauksia. Näitä tilanteita kutsutaan päätöspuun *ylisovittamiseksi* (overfitting) [13]. Ylisovittaminen siis heikentää päätöspuun luokittelu-

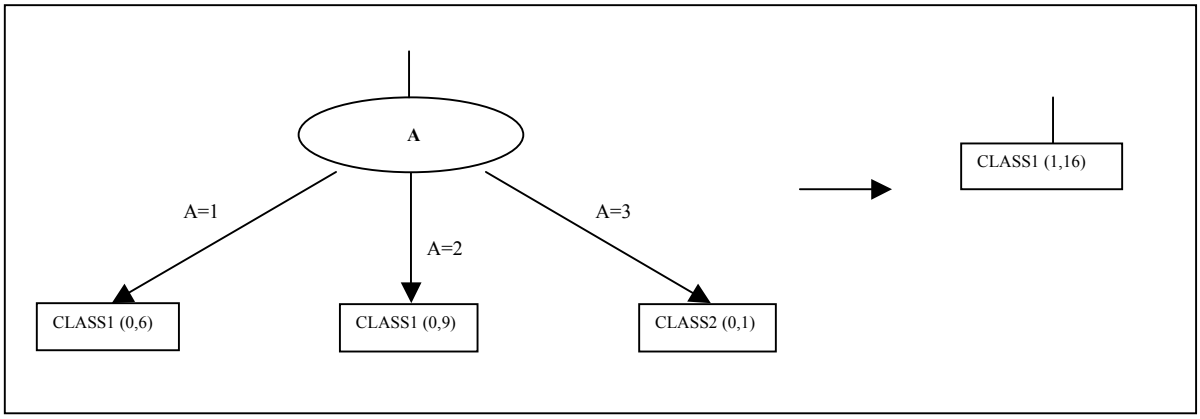
tarkkuutta. Päätöspuun karsinnalla pyritään välttämään ylisovittamista. Päätöspuiden rakennetta voidaan yleensä yksinkertaistaa korvaamalla jokin alipuu yhdellä sen lehtisolmuista. Päätöspuun koon pienenemisen lisäksi karsimisen tavoitteena on, että puun luokittelun virhetaso alenee.

C4.5 laajentaa ID3:sta *jälkikarsivalla* (postpruning) menetelmällä, jossa päätöspuu karsitaan vasta, kun se on muodostettu kokonaan. Lisäksi se käyttää erityistä menetelmää, *pessimististä karsintaa* (pessimistic pruning), virhearvion ennustamiseksi [13]. Puun jokaiselle lehtisolmulle lasketaan parametri U_{cf}, joka kertoo ylärajan virheen todennäköisyydelle. C4.5 käyttää 25 % luottamustasoa, jolloin $U_{25\%}(E,N) = p$. Binomijakaumaa [10] käytettäessä todennäköisyys saadaan ratkaisemalla p kaavasta,

$$\sum_{i=0}^{E} \binom{N}{i} p^i (1-p)^{N-i} = 0.25, \tag{8}$$

missä N on esimerkkitapausten lukumäärä, jonka kyseinen lehtisolmu saavuttaa ja E on niiden tapausten lukumäärä N:stä, jotka luokiteltiin väärin. Edelleen jokainen virhearvion todennäköisyys kerrotaan lehtisolmun kattavilla esimerkkitapausten lukumäärällä eli N:llä, jolloin saadaan lehtisolmun virheiden lukumääräarvio. Koko alipuun ennustettu virhearvio saadaan, kun jokaisen lehtisolmun virheiden lukumääräarviot lasketaan yhteen. Vertailuarvona lasketaan korvaavien lehtisolmujen virhearviot. Korvaava lehtisolmu tarkoittaa solmua, jonka nimiönä on jokin alipuun luokka-arvo ja se saavuttaa saman määrän esimerkkitapauksia kuin alipuu ja E on vähintään yksi.

Jos alipuulla on suurempi virhearvio kuin korvaavalla lehtisolmulla, päätöspuuta karsitaan ja alipuu korvataan pienimmän virhearvion omaavalla lehtisolmulla. Seuraavana esitettävä esimerkki perustuu Kantardizicin kirjaan [9].



Kuva 9. Alipuun karsinta.

Kun halutaan selvittää, voidaanko alipuu korvata lehtisolmulla, lasketaan ennustettu virhearvo (virheiden lukumäärä) solmulle *A* ja koko alipuulle. Kuvan 9 alipuu saavuttaa yhteensä 16 esimerkkitapausta, jotka jakautuvat attribuutin *A* arvojen mukaan. Tarkoituksena olisi selvittää, kannattaako alipuu korvata lehtisolmulla *CLASS1*.

Ensimmäiselle lehtisolmulle (A=1) N=6 ja E=0, joten kun virhearvon ylärajan laskentaan käytetään oletusarvoisesti 25 % luottamustasoa, saadaan arvo $U_{25\%}(0,6) = 0.206$. Jos lehtisolmulla luokitellaan kuusi uutta testitapausta, saadaan 6 * 0.206. Jäljellä oleville lehtisolmuille saadaan vastaavalla menetelmällä arvot $U_{25\%}(0,9) = 0.143$ ja $U_{25\%}(0,1) = 0.750$. Ennustettu virheiden lukumäärä koko alipuulle on siis

$$6 * 0.206 + 9 * 0.143 + 1 * 0.750 = 3.273.$$

Jos alipuu korvattaisiin lehtisolmulla *CLASS1*, se kattaisi edelleen samat 16 tapausta yhdellä virheellä, koska yksi tapaus kuului luokkaan *CLASS2* (*A*:n arvolla 3):

$$16 * U_{25\%}(1,16) = 16 * 0.157 = 2.512.$$

Koska koko alipuulla on suurempi virhearvo, se karsitaan lehtisolmuksi *CLASS1*(1,16).

3 LUOKITTELU SUMEALLA ID3-PÄÄTÖSPUULLA

Standardi ID3 toimii hyvin, jos luokiteltava aineisto on täsmällinen. Jos aineisto sisältää epävarmuutta ja kohinaa, sen suorituskyky laskee [11, 3]. Sumeus voidaan yhdistää oppimisjärjestelmiin monella tavalla, usein menetelmät jakaantuvat esisumeuttaviin ja jälkisumeuttaviin menetelmiin [2]. Lisäksi on olemassa menetelmiä, jotka käyttävät sumeutta hyväksi myös päätöspuun rakennusvaiheessa [16]. Tutkielmassa esitettävä UR-ID3-algoritmi kuuluu jälkisumeuttaviin, joissa ensin luodaan täsmällinen päätöspuu ja sumeus otetaan mukaan vasta luokitteluvaiheessa.

3.1 Sumea joukko-oppi

Sumea joukko-oppi (fuzzy set theory) on eräs tapa mallintaa matemaattisesti epävarmuutta. Kun perinteisessä, kaksiarvoisessa joukko-opissa alkio kuuluu tai ei kuulu joukkoon, sumea joukko-oppi mahdollistaa myös alkion osittaisen kuulumisen joukkoon [17]. Sumean joukon, kuten nimi ilmaisee, rajat eivät ole tarkkoja. Siirtyminen joukon ulkopuolelta sisäpuolelle on asteittaista ja apuna käytetään *jäsenyysfunktiota* (membership function). Sumeiden joukkojen ensimmäisen artikkelin kirjoitti vuonna 1965 iranilaissyntyinen professori Lotfi A. Zadeh [17]. 1960- ja 1970-luvulla hän esitti *sumean joukko-opin* (fuzzy set theory) ja *sumean logiikan* (fuzzy logic) perusteet. Zadehin mukaan tämän kokonaisuuden tavoitteena on inhimillisen kielellisen päättelyn jäljittely tietokoneympäristössä.

Seuraavaksi käydään läpi sumean joukko-opin käsitteitä ja määritelmiä, joita tullaan tarvitsemaan jatkossa. Kaikki seuraavat määritelmät perustuvat Zimmermanin teokseen [17]. Esitetyt esimerkit ja notaatio perustuvat Kantardzicin kirjaan [9].

Jos X on alkioiden x avaruus, klassisen joukon määrittelyssä alkio $x \in X$ joko kuuluu tai ei kuulu joukkoon A. Määrittelemällä karakteristinen funktio jokaiselle alkiolle $x \in X$, voidaan klassinen joukko esittää järjestettyjen parien $(x,0)$ tai $(x,1)$ joukkona, missä $(x,1)$ tarkoittaisi joukon jäsenyyttä eli $x \in A$ ja $(x,0)$ ei-jäsenyyttä eli $x \notin A$.

Määritelmä 2. Jos U on perus- eli referenssijoukko ja u edustaa U:n objektia, *sumea joukko* $A=\{ (u,\mu_A(x)) \mid u \in U \}$.

Sumea joukko A perusjoukossa U on karakterisoitu jäsenyysfunktiolla μ_A, joka saa arvoja väliltä [0,1]. μ_A:ta voidaan kutsua myös jäsenyyden asteeksi. Kun u on varmasti sumean joukon A jäsen, $\mu_A(u) = 1$. Kun taas $\mu_A(u) = 0$ tarkoittaa, että u ei varmasti kuulu sumeaan joukkoon A. Kun $0 < \mu_A(u) < 1$, objekti u kuuluu joukkoon A vain osittain. Lisäksi mitä pienempää osittainen kuuluminen on, sitä lähempänä on nollaa vastaava jäsenyysaste. Jos $\mu_A(u)$ saa vain arvoja 0 tai 1, kaikilla $u \in U$, joukossa A ei esiinny sumeutta eli se on *täsmällinen joukko* (crisp set).

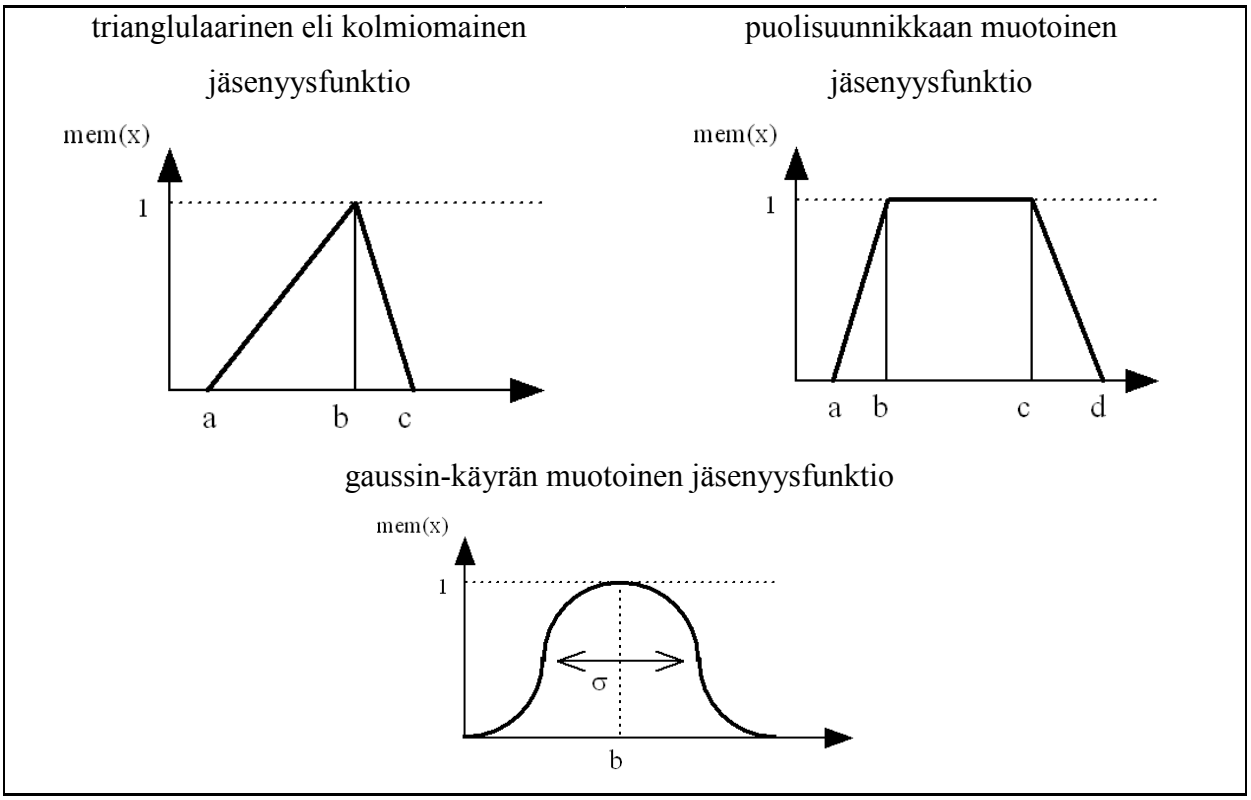
Sumeiden joukkojen jäsenyysfunktioiden muoto on yleensä rajoittunut tiettyihin funktioihin, jotka voidaan tarkentaa vain muutamalla parametrilla [9]. Tunnetuimmat muodot ovat kolmiomainen (triangular), puolisuunnikas (trapezoidal) ja Gaussin-käyrä (kuva 10). Zimmermannin mukaan [17] laskennallisen tehokkuuden vuoksi sumean joukon jäsenyysfunktiona käytetään yleensä puolisuunnikkaan muotoista tai kolmikulmaista jäsenyysfunktiota.

Tarkastellaan esimerkkinä henkilön pituutta. Jos joukko määritellään A = {”henkilö on pitkä”}, täsmällinen rajaus A ={x | x ≥ 190} tarkoittaisi tällöin, että 189 cm pituinen henkilö ei olisi pitkä. Jos joukko A määriteltäisiin sumealla jäsenyysfunktiolla $\tilde{A}=\{(x), \mu_{\tilde{A}}(x) \mid x \in X\}$, missä

$$\mu_{\tilde{A}}(x) = \begin{Bmatrix} 0 & , jos & x \le 170 \\ (x-170)/(190-170) & , jos & 170 < x < 190 \\ 1 & , jos & x \ge 190 \end{Bmatrix}$$

Eli tällöin esimerkiksi 189 cm pituisen henkilön jäsenyysaste sumeassa joukossa A on 0.95. Kun taas 175 cm pituisen henkilön jäsenyysaste on enää vain 0.33.

x = 189 *cm*, $\mu_{\tilde{A}}(x)$ = (189 - 170) / (190 - 170) = <u>0.95</u>
x = 175 *cm*, $\mu_{\tilde{A}}(x)$ = (175 - 170) / (190 - 175) = <u>0.33</u>



Kuva 10. Jäsenyysfunktion kolme eri muotoa.

Jäsenyysfunktiolla on useita ominaisuuksia ja tunnuslukuja, joita käytetään sumeiden joukkojen operaatioissa ja sumeissa päättelyjärjestelmissä. Seuraavaksi käydään läpi sumean joukko-opin tärkeimmät käsitteet, joita tullaan tarvitsemaan jatkossa.

Määritelmä 3. Sumean joukon *A tuki* (support) on kaikkien niiden pisteiden *u* tarkka joukko, jolle $\mu_A(u) > 0$ eli $Support(A) = \{u \mid \mu_A(u) > 0\}$ ja $u \in U$.

Määritelmä 4. Sumea joukko A on *normaali*, jos on olemassa piste $u \in U$, jolle $\mu_A(u) = 1$.

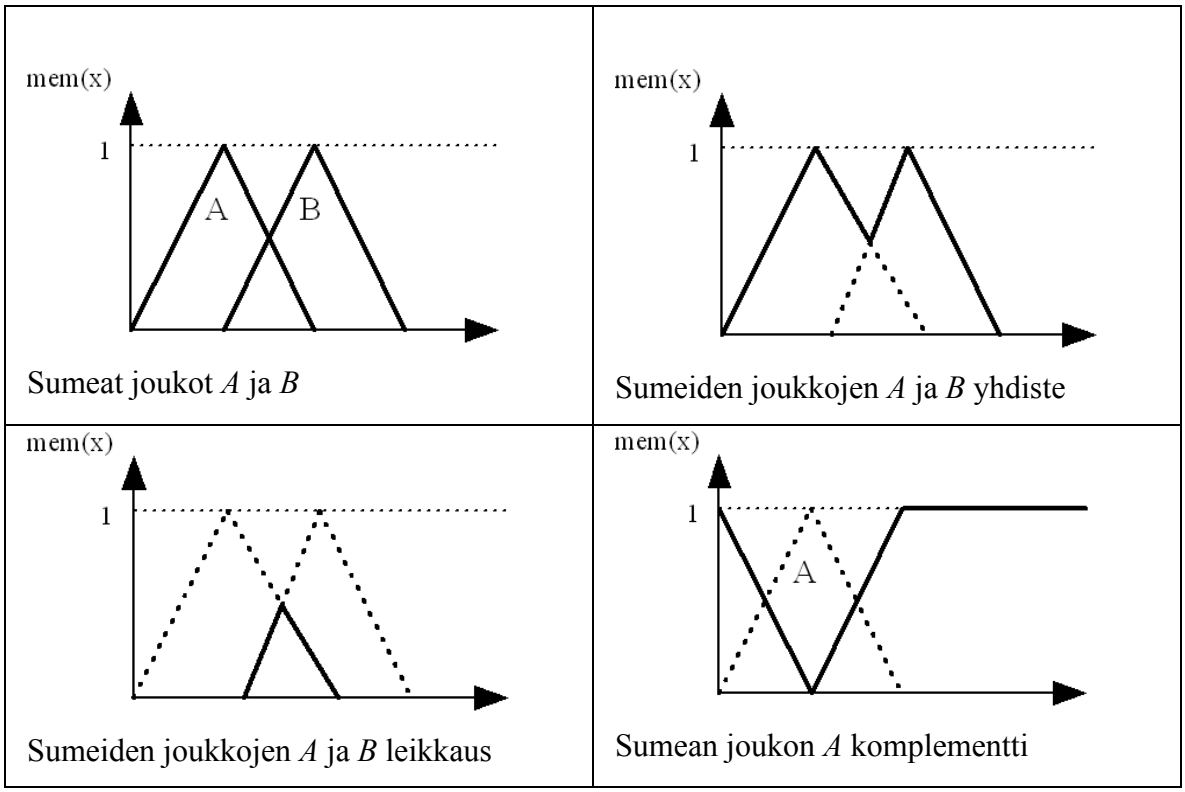
Määritelmä 5. Olkoon sumeiden joukkojen *A* ja *B* jäsenyysfunktiot referenssijoukossa *U*, μ_A ja μ_B. Sumeiden joukkojen *A* ja *B leikkaus* (intersection) on sumea joukko *C, kun* $C = A \cap B$ ja *C*:n jäsenyysfunktio on $\mu_C(u) = \mu_{A \cap B}(u) = \min\{\mu_A(u), \mu_B(u)\}$, kun $u \in U$.

Määritelmä 6. Joukkojen *A* ja *B yhdiste* (union) on sumea joukko *D, kun* $D = A \cup B$ ja *D*:n jäsenyysfunktio on $\mu_D(u) = \mu_{A \cup B}(u) = \max\{\mu_A(u), \mu_B(u)\}$, kun $u \in U$.

Määritelmä 7. Joukon *A komplementti* on joukko $\overline{A}$, kun $\mu_{\overline{A}}(u) = 1 - \mu_A(u)$.

Määritelmä 8. Joukko A on joukon B *alijoukko*, jos ja vain jos $\mu_A(u) \leq \mu_B(u), \forall u \in U$.

Kuvaesimerkit määritelmien 5 - 7 operaatioista näkyvät kuvassa 11.



Kuva 11. Sumeiden joukkojen A ja B perusoperaatiot.

Arkipäivän likimääräisiin lukumääriin liittyviä ilmaisuja (esim. desimaaliluvun pyöristäminen kokonaisluvuksi) voidaan matemaattisesti mallintaa *sumeiden lukujen* (fuzzy number) avulla [17]. Sumea luku on itse asiassa tavallinen sumea joukko, jolla on seuraavat ehdot:

1. On olemassa vain yksi $u \in U$, jolle $\mu_A(u) = 1$. Eli jäsenyysfunktion täytyy olla normalisoitu ja sillä on vain yksi huippuarvo *u*.
2. Jäsenyysfunktion täytyy kasvaa ja laskea monotonisesti kummaltakin puolelta pistettä *u*. Tämä varmistaa, että on olemassa vain yksi huippuarvo *u*.

Esimerkiksi kuvan 11 sumeat joukot A ja B voitaisiin määritellä yhtä hyvin sumeiksi luvuiksi.

Jotta kahta sumeata lukua voidaan verrata toisiinsa, täytyy niille määritellä jokin epätarkkaa informaatiota määrittelevä mitta. Eräs tapa kahden sumean luvun yhtäläisyyden vertailemiseen on *tukiparin* (support pair) muodostaminen. Tukipari esitetään välinä $[S_n, S_p]$, jossa ensimmäinen luku ilmaisee tuen päätelmän välttämättömyydelle ja jälkimmäinen mahdollisuudelle [17]. Jos väitteen tiedetään olevan tosi, sitä tuetaan välttämättä asteella yksi. Jos taas sen tiedetään olevan epätosi, sen negaatiota tuetaan asteella yksi. Epävarmuutta ilmenee, kun väitettä ja sen negaatiota tuetaan välttämättä asteilla x ja y, jotka ovat positiivisia lukuja ja pienempiä kuin yksi. Täyttä tukea merkitään $[S_n, S_p] = [1,1]$. Kahden sumean joukon A ja B samanlaisuuden *mahdollisuustuki* (possible support) saadaan kaavasta

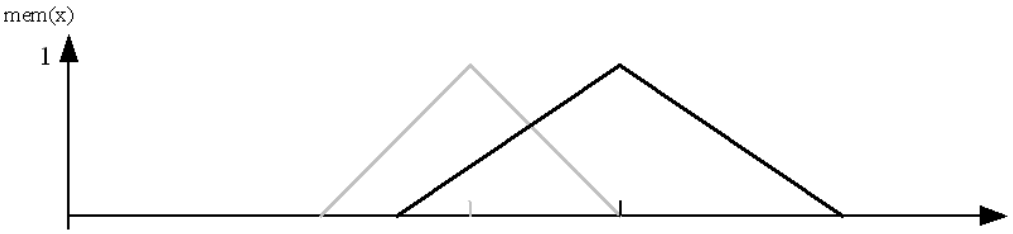
$$S_p = max\ [\ min(A(x), B(x))\ ],\ x \in X. \qquad (9)$$

Mahdollisuustuki on tapauksen todennäköisyyden yläraja. Jos tapaus ei ole mahdollinen, on se myös epätodennäköinen. On kuitenkin huomattava, että korkea mahdollisuustuki ei välttämättä johda korkeaan todennäköisyyteen [17]. Tukiaste yhtäläisyyden välttämättömyydelle eli *välttämättömyystuelle* (necessary support) saadaan kaavalla

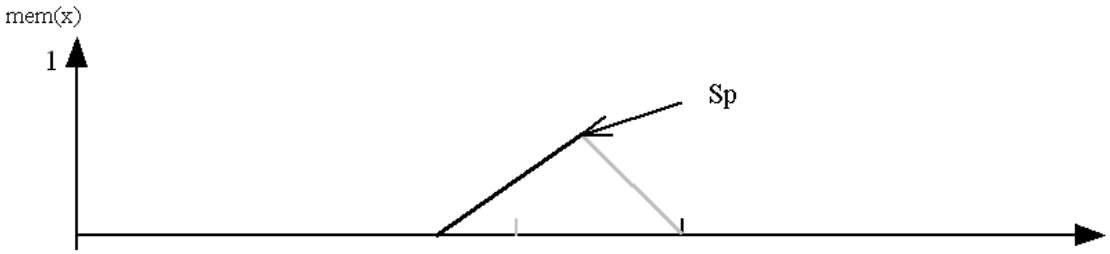
$$S_n = min\ [\ min(max\{A(x), 1\text{-}B(x)\}), min(max\{1\text{-}A(x), B(x)\})\ ]\ ,\ x \in X. \qquad (10)$$

Koska $min(max\{A(x), 1\text{-}B(x)\})$ ja $min(max\{1\text{-}A(x), B(x)\})$ ovat yleensä erisuuruisia, näistä valitaan minimi [11]. Välttämättömyystuki määrittelee asteen, jossa B sisältyy A:han.

Seuraavaksi käydään esimerkkikuvin läpi, kuinka mahdollisuustuki ja välttämättömyystuki lasketaan. Kuvassa 12 on kaksi sumean luvun jäsenyysfunktiota (joukko A mustalla ja B harmaalla). Lukujen välinen mahdollisuus on leikkauksen maksimikohta (kuva 13). Mahdollisuustuki, S_p, ilmaisee siis suuruuden kohdassa, jossa A ja B limittyvät.

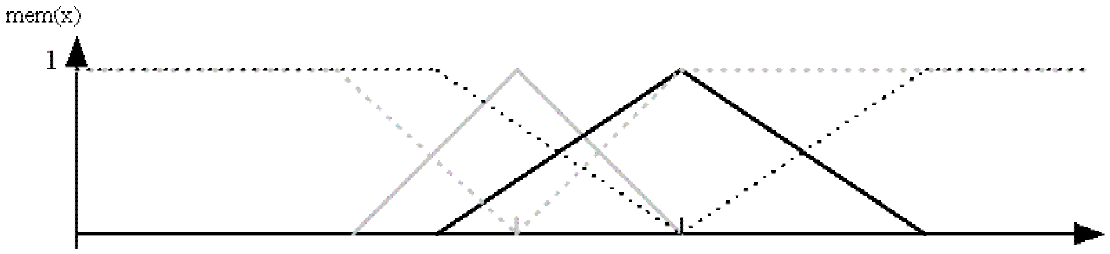


Kuva 12. Kahden sumean luvun jäsenyysfunktiot.

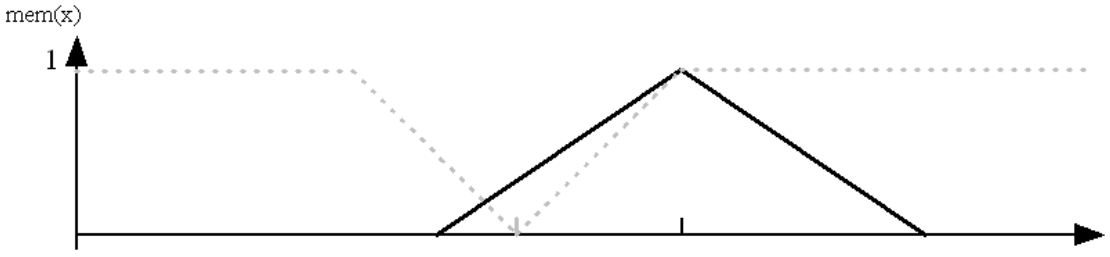


Kuva 13. Kahden sumean luvun mahdollisuustuki S_p.

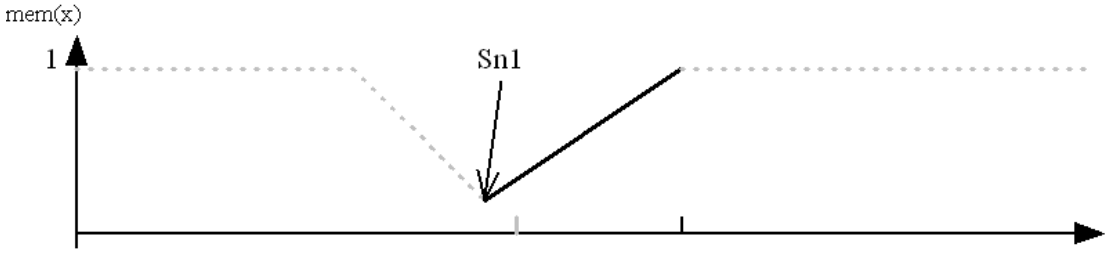
Välttämättömyystuen, S_n, laskemiseksi täytyy laskea kaksi ehdokasta, S_{n1} ja S_{n2}. Näistä luvuista valitaan tukiparin välttämättömyystueksi pienempi [11]. Kuvassa 14 näkyy kaksi sumeaa lukua (*A* mustalla ja *B* harmaalla) ja niiden komplementit katkoviivoilla. S_{n1} saadaan sumean luvun *A*:n ja sumean luvun *B*:n komplementin yhdisteen minimikohdasta (kuva 15 ja 16).



Kuva 14. Kahden sumean joukon jäsenyysfunktiot ja niiden komplementit.

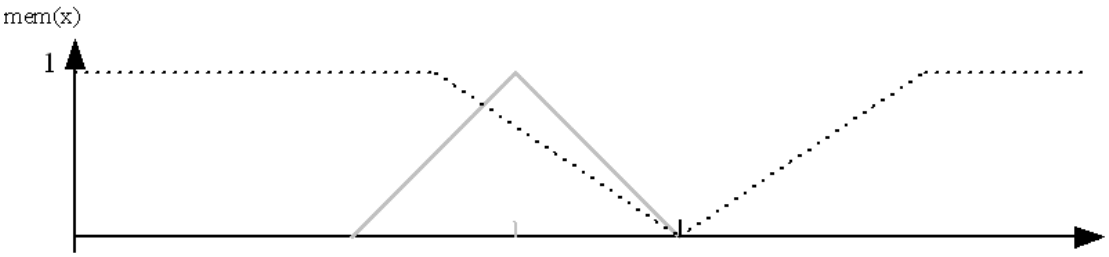


Kuva 15. Sumea luku *A* (musta) ja sumean luvun *B* komplementti (harmaa)

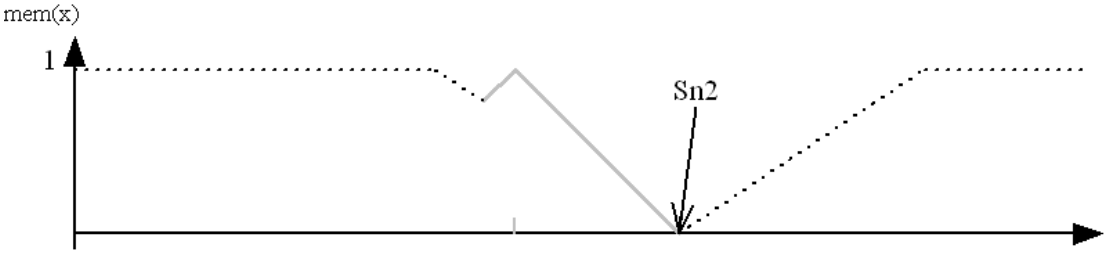


Kuva 16. Sumeiden lukujen ensimmäinen mahdollinen välttämättömyystuki S_{n1}.

Vastaavasti S_{n2} saadaan sumean luvun B:n ja sumean luvun A:n komplementin yhdisteen minimikohdasta (kuvat 17 ja 18).

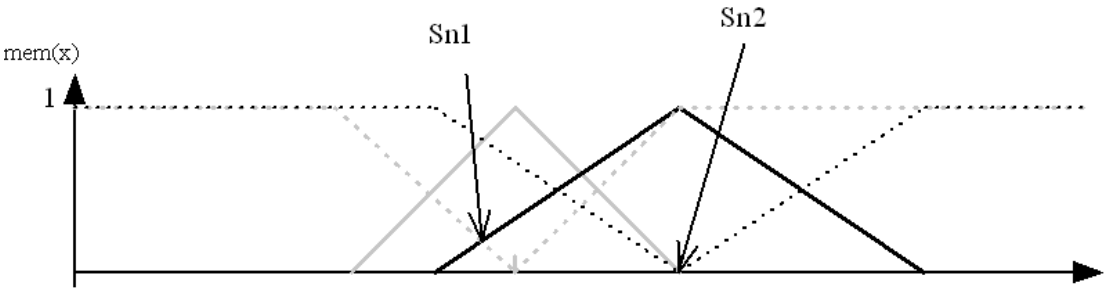


Kuva 17. Sumean luvu A komplementti (musta) ja sumea luku B (harmaa).



Kuva 18. Sumeiden lukujen A ja B toinen välttämättömyystuki-ehdokas S_{n2}.

Koska yleensä S_{n1} on eri kuin S_{n2}, otetaan välttämättömyystueksi niiden minimi, jolloin esimerkissä kuvan 19 perusteella $S_n = min\ (S_{n1}, S_{n2}) = S_{n2}$.



Kuva 19. Välttämättömyystueksi otetaan ehdokkaista pienempi S_{n2}.

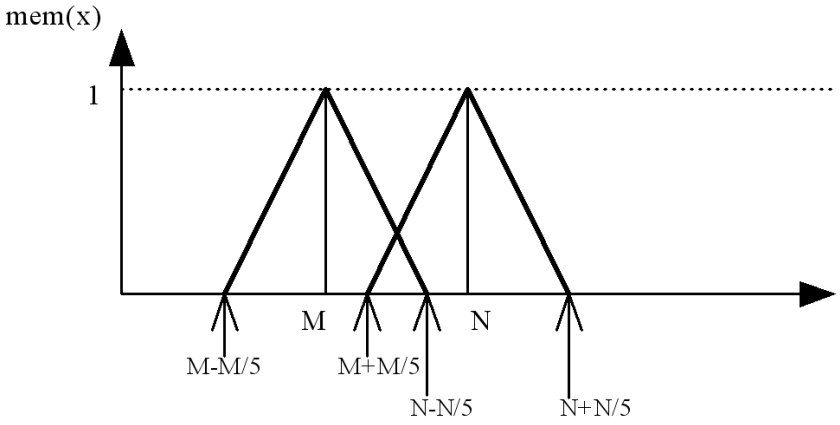
3.2 UR-ID3

Maher ja St. Clair esittelivät vuonna 1993 menetelmän UR-ID3 (Uncertain Reasoning ID3) [11], joka yhdistää epävarman päättelyn ja ID3-päätöspuuprosessin. UR-ID3 tavoitteena on epävarman ja epätarkan aineiston luokittelutarkkuuden parantuminen perinteiseen ID3:een

verrattuna. Menetelmän luokittelutarkkuus on Maherin ja St. Clairin [11] testien mukaan täysin ylivertainen verrattuna tavalliseen ID3:een. Tätä puoltaa Chin ja Yanin [3] UR-ID3:een perustuva sovellus, jossa tutkitaan sumean luokittelun tarkkuutta hahmontunnistuksessa. Beasaid et al. [1] esittelevät oman sydänkäyrädiagnoosi päätösjärjestelmän, joka perustuu osittain UR-ID3:n jälkisumeuttavaan menetelmään [3]. Diagnoosijärjestelmä saavutti sumeuden avulla täyden 100 % luokittelutarkkuuden.

UR-ID3 prosessi alkaa perinteisen ID3-päätöspuun luonnilla esimerkkiaineistosta, jonka attribuuttien oletetaan olevan erillisiä ja varmoja. Tämän jälkeen päätöspuun solmut ja lehdet ”sumeutetaan” kolmiomaisella jäsenyysfunktiolla. Itse ID3-puurakennne ja sen synnyttämä sääntöjoukko pysyvät muuttumattomina ja sumeutta käytetään vasta testiaineiston luokittelussa, kun testitapausten arvoja tarkastellaan aina epävarmoina. Chiang ja Hsu [2] kategorioivatkin UR-ID3:n jälkisumeuttavaksi menetelmäksi. Seuraavana esitetty UR-ID3 algoritmi ja sen esimerkki perustuvat Maherin ja St. Clairin kehittämään menetelmään [11].

UR-ID3 alkaa tavallisen ID3-päätöspuun luomisella esimerkkiaineistosta. Päätöspuun luontivaiheessa jokaiseen lehtisolmuun täytyy tallentaa lukumäärätieto niistä esimerkkitapauksista, jotka johtivat lehtisolmuun. Täsmällisen päätöspuun luonnin jälkeen sekä puun attribuuttien arvoja että testitapauksien arvoja pidetään sumeina eli päätöspuun oksat sumeutetaan jäsenyysfunktiolla. Kuvassa 20 nähdään sumeuttamisen periaate. Arvot *M* ja *N* on määritelty yhden tarkan pisteen sijasta jäsenyysfunktiolla, jonka approksimaattina eli jakovakiona on käytetty lukua *5*. Jakovakio voidaan parametrisoida ja käsitellä sumeutuksen varmuustekijänä [11]. Yhtäläisyyden sumean vertailun erityistapauksena on kahden saman arvon vertailu, jolloin tukipari määritellään $[S_n, S_p] = [1,1]$. Maherin ja St. Clairin [11] mukaan laskennallinen vaativuus laskee tällöin täsmällisen ID3:n tasolle.



Kuva 20. Kahden tarkan arvon sumeuttaminen kolmiomaisella jäsenyysfunktiolla.

UR-ID3 laskee testitapauksen attribuuttien mukaan puun jokaiselle lehtisolmulle tukiparin. Tukipari lasketaan poluittain vertaamalla puun joka tasolla testitapauksen attribuutin arvoa solmun attribuutin arvoon. Näille kahdelle sumealle arvolle lasketaan yhtäläisyyden sumea tukipari. Tämä tukipari edustaa testitapauksen attribuuttien arvojen todennäköisyyttä juurisolmusta lehtisolmuun [11]. Kun päätöspuun polkua P_j pitkin kuljetaan lehtisolmua L_j kohti, jokaisella tasolla saadaan uusi tukipari. Jotta jokaiselle lehtisolmulle saataisiin vain yksi tukipari, täytyy peräkkäisten tasojen tukiparit $[a_1,b_1]$ ja $[a_2,b_2]$ yhdistää. UR-ID3 käyttää tähän loogista AND-operaatiota:

$$[a_1, b_1]\,\mathrm{AND}\,[a_2, b_2] = [a_1 a_2, b_1 b_2]. \qquad (11)$$

Näin tukiparien yhdistämistä jatketaan attribuutti kerrallaan polkua P_j pitkin lehtisolmuun asti. Tuloksena saadaan yksi tukipari, θ_j, joka liitetään lehtisolmuun L_j. Täsmällisen päätöspuun rakennusvaiheessa jokaiseen lehtisolmuun tallennettiin kokonaisluku, N_j, joka kertoo kuinka monta esimerkkitapausta johti polkua pitkin lehtisolmuun L_j. Tätä lukua käytetään hyväksi päätöksen painoarvon laskemisessa. Päätöksen tukipari painotetaan jokaiselle *j* kaavalla

$$\theta_j' = \theta_j * (N_j / \sum N_k), \qquad (12)$$

missä $\sum N_k$ merkitsee esimerkkiaineiston kaikkien tapausten lukumäärää ja N_j lehtisolmuun johtaneiden esimerkkitapausten lukumäärää.

Painotus suoritetaan jokaiselle lehtisolmulle, jolloin tulokseksi saadaan kaikkien päätösten tukiparien joukko

$$\theta' = \{\theta_1', \theta_2', \ldots, \theta_t'\}. \tag{13}$$

Jokaista tulosluokkaa C_i vastaan on olemassa alijoukko $L_{Ci} \subseteq L$ ts. joukko L_{Ci} sisältää ne lehtisolmut, jotka kuuluvat tulosluokkaan C_i. Koska jokaiseen lehtisolmuun L_{Ci} on liitetty tukipari, täytyy nämä parit yhdistää yhdeksi tukipariksi (θ_{Ci}), joka vastaa koko päätöksen C_i tukiparia. Yhdistämiseen käytetään OR-operaatiota:

$$[a_1, b_1] \,\mathrm{OR}\, [a_2, b_2] = [a_1 + a_2 - a_1 a_2, b_1 + b_2 - b_1 b_2] \tag{14}$$

Tuloksena saadaan jokaiselle mahdolliselle tulosluokalle yksi tukipari [S_n, S_p], joten testitapaus voidaan nyt luokitella valitsemalla sopivan tukiparin omaava tulosluokka. UR-ID3 valitsee päätökseksi sen luokan jolla on suurin välttämättömyystuki S_n [11].

Kun täsmällinen luokittelu etsii testitapaukselle vain yhtä oikeaa polkua juuresta yhteen lehtisolmuun, UR-ID3:n sumea luokittelu laskee päätöksen puun kaikkien solmujen mukaan.

3.2.1 Esimerkki sumeasta luokittelusta

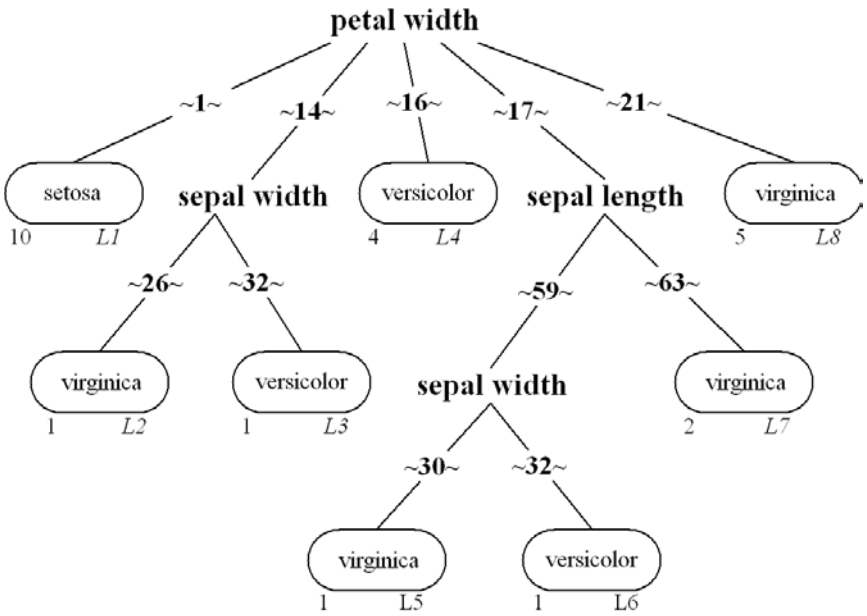
Maherin ja St. Clairin mukaan [11] kuvan 21 esittämä UR-ID3 päätöspuu on luotu normaalilla ID3-algoritmilla, jonka attribuuttien arvoja pidetään tarkkoina. Esimerkkiaineistoksi on otettu 25 tapausta.

Kahden sumean luvun *A* ja *B* mahdollisuustuki voidaan laskea seuraavasti [11, 8],

$$S_p = \frac{x(A - B) + A + B}{A + B}, \tag{15}$$

kun $B \geq A$ ja sumean jakson ääripisteet määrittävä jakovakio on x. Esimerkiksi, jos jakovakio $x=5$, sumeiden lukujen *~17 ja ~18* mahdollisuustuki on

$$S_p\ (\sim 17, \sim 18) = \frac{5*(17-18)+17+18}{(17+18)} = \underline{0.857}.$$



Kuva 21. UR-ID3 päätöspuu.

Välttämättömyystueksi valitaan minimi kahdesta luvusta, S_{n1} tai S_{n2}. Vasemmanpuoleinen S_{n1} ja oikeanpuoleinen S_{n2} saadaan seuraavilla kaavoilla [11, 8],

$$S_{n1} = \frac{x(A-B)+B}{(A+B)}, \tag{16}$$

$$S_{n2} = \frac{x(A-B)+A}{(A+B)}, \tag{17}$$

missä A ja B ovat sumeita lukuja, $B \geq A$ ja jakovakiona on x. Esimerkiksi, jos lasketaan välttämättömyystuki arvoille ~17 ja ~18, täytyy laskea S_{n1} ja S_{n2} ja valita näistä pienempi. Jakovakiona x on *5*:

$$S_{n1} = \frac{5*(17-18)+18}{(17+18)} = 0.371 \text{ ja } S_{n2} = \frac{5*(17-18)+17}{(17+18)} = 0.343.$$

S_n *(~17,~18) = min(*0.371, 0.343*)=*<u>0.343.</u>

Seuraavaksi käydään läpi yhden testitapauksen luokittelun vaiheet. Päätöspuuna käytetään kuvan 21 mukaista puuta ja luokiteltavan testitapauksen attribuuteilla on seuraavat arvot [11]:

petal_width = ~ 18
petal_length= ~ 45
sepal_width= ~ 29
sepal_length= ~ 60

Taulukko 5. Testitapauksen ja kuvan 21 mukaisen päätöspuun sumeat vertailut.

Vertailu	**Tukiparit [S_n, S_p]**		
	Taso1	**Taso2**	**Taso3**
~18 ⊕ ~1	[0, 0]		
~18 ⊕ ~14	[0, 0.375]		
~18 ⊕ ~16	[0.176, 0.706]		
~18 ⊕ ~17	[0.343, 0.857]		
~18 ⊕ ~21	[0.077, 0.615]		
~29 ⊕ ~26		[0.2, 0.727]	
~29 ⊕ ~32		[0.230, 0.754]	
~60 ⊕ ~59		[0.454, 0.958]	
~60 ⊕ ~63		[0.366, 0.878]	
~29 ⊕ ~30			[0.407, 0.915]
~29 ⊕ ~32			[0.230, 0.754]

Laskujen suoritusjärjestys ei ole kiinnitetty, joten tässä lasketaan ensin tukiparit puun jokaiselle solmulle, jonka jälkeen ne yhdistetään. Päätöspuun ensimmäisellä tasolla verrataan juurisolmun attribuutin (petal_width) kaikkia arvoja tapauksen attribuutin arvon kanssa (~ 18). Esimerkiksi, kun verrataan sumeita lukuja ~18 ja ~16, saadaan välttämättömyystuen ehdokkaiksi S_{n1} = 0.235 ja S_{n2}=0.176, joista valitaan pienempi eli S_n = *min* (S_{n1}, S_{n2}) = S_{n2}. Kun lasketaan vielä yhtäläisyyden mahdollisuustuki, saadaan tukipariksi [S_n, S_p] = [0.176, 0.706]. Puun seuraavalla tasolla verrataan testitapauksen attribuuttien sepal_width ja sepal_length arvoja päätöspuun vastaaviin arvoihin. Päätöspuun alimmalla tasolla vertailut suoritetaan vain attribuutin sepal_width kanssa. Testitapauksen arvojen vertailut kuvan 21 päätöspuun solmujen kanssa näkyvät taulukossa 5.

Taulukko 6. Tukiparien yhdistäminen poluittain.

Polun tukiparit	Painotus	Painotettu tukipari
$P_1 : \theta_1 = [0, 0]$	10/25	$\theta_1' = [0, 0]$
$P_2 : \theta_2 = [0, 0.375] \wedge [0.2, 0.727]$	1/25	$\theta_2' = [0, 0.011]$
$P_3 : \theta_3 = [0, 0.375] \wedge [0.230, 0.754]$	1/25	$\theta_3' = [0, 0.011]$
$P_4 : \theta_4 = [0.176, 0.706]$	4/25	$\theta_4' = [0.028, 0.113]$
$P_5 : \theta_5 = [0.343, 0.857] \wedge [0.454, 0.958] \wedge [0.407, 0.915]$	1/25	$\theta_5' = [0.003, 0.030]$
$P_6 : \theta_6 = [0.343, 0.857] \wedge [0.454, 0.958] \wedge [0.230, 0.754]$	1/25	$\theta_6' = [0.001, 0.025]$
$P_7 : \theta_7 = [0.343, 0.857] \wedge [0.366, 0.878]$	2/25	$\theta_7' = [0.010, 0.060]$
$P_8 : \theta_8 = [0.077, 0.615]$	5/25	$\theta_8' = [0.015, 0.123]$

Jotta puun jokaiselle lehtisolmulle saadaan yksi tukipari, ne yhdistetään poluittain käyttämällä AND-operaatiota (taulukko 6). Esimerkiksi polun P_3 tukipari θ_3 saadaan, kun vertailujen $\sim 18 \oplus \sim 14$ ja $\sim 29 \oplus \sim 32$ tukiparit yhdistettään AND-operaatiolla: $[0,0.375] \wedge [0.230, 0.754] = [0*0.230, 0.375*0.754] = [0, 0.283]$. Kun saatu tukipari vielä painotetaan, saadaan lehtisolmulle lopullinen tukipari eli $\theta_3' = 1/25 * [0, 0.283] = [0, 0.011]$.

Taulukko 7. Lehtisolmujen tukiparien yhdistäminen päätöksen tukipariksi.

Päätös	Kaava	Päätöksen tukipari
θ_{Setosa}	θ_1'	[0, 0]
$\theta_{Versicolor}$	$\theta_3' \vee \theta_4' \vee \theta_6'$	[0.029, 0.145]
$\theta_{Virginica}$	$\theta_2' \vee \theta_5' \vee \theta_7' \vee \theta_8'$	[0.028, 0.210]

Kun jokaisen lehtisolmun tukipari on laskettu, yhdistetään ne luokittain OR-operaatiolla päätösten tukipareiksi (taulukko 7). Esimerkiksi päätökselle Versicolor lasketaan ensin $\theta_3' \vee \theta_4' = [0, 0.011] \vee [0.028, 0.113] = [0 + 0.028 – 0 * 0.028, 0.011 + 0.113 – 0.011 * 0.113] = [0.028, 0.123]$. Kun saatu tukipari yhdistetään vielä polun θ_6' tukiparin kanssa, saadaan $[0.028, 0.123] \vee [0.001, 0.025] = [0.029, 0.145]$.

Esimerkin luokittelun päätöksenä on tulosluokka Versicolor, koska sen välttämättömyys-tuki on korkein.

4 ID3-PÄÄTÖSPUUN LUOKITTELUTARKKUUS

Tässä luvussa tutkitaan ID3-algoritmin luokittelutarkkuuksia käyttämällä Juholan [8] toteuttamaa SP-ID3 järjestelmää, jolla voidaan verrata sumean ja täsmällisen ID3-päätöspuun luokittelutarkkuuksia. Järjestelmän ID3-algoritmin päätöspuuinduktio perustuu Hanin ja Kamberin [6] versioon Quinlanin ID3:sta. Lisäksi järjestelmä mahdollistaa aineiston ikkunoinnin. Sumealuokittelu perustuu Maherin ja St. Clairin [11] UR-ID3:een, joka esiteltiin kohdassa 3.2.

Aineistona vertailussa käytetään Iris (suom. kurjenmiekka) aineistoa, jonka R. A. Fisher esitteli vuonna 1936 artikkelissaan "The Use of Multiple Measurements in Axonomic Problems" (Annals of Eugenics 7, 179 - 188) [11]. Aineisto koostuu neljästä kukan attribuutista: sepal width, sepal length, petal width ja petal length. Tapauksilla on kolme tulosluokka, jotka kuvaavat kukin kukan alalajia: setosa (1), versicolor (2) ja virginica (3). Kaikki mittaukset ohjelmassa ovat pituuksia millimetreinä. Iris-aineiston [5, 12] kaikki 150 tapausta on esitetty liitteessä 1. Aineiston käsittelyn periaate käy ilmi kohdan 3.2.1 esimerkistä.

Maher ja St. Clair [11] valitsivat Iris aineiston, koska sitä on käytetty laajasti kirjallisuudessa luokitteluongelmissa. Lisäksi koska aineisto on kerätty reaalimaailmasta ja se on melko kohinaton, se soveltuu hyvin sumeaan päättelyyn [11]. Maher ja St. Clair toteavat UR-ID3:n suorituksen olevan täysin ylivertainen tavalliseen ID3:een verrattuna [11]. Heidän testeissään esimerkkiaineistona on 75 % Iris-aineistosta (113 tapausta) ja loput 25 % (37 tapausta) kuuluvat testiaineistoon. Tapaukset valittiin esimerkkiaineistoon satunnaisesti. Taulukossa 8 on esitetty Maherin ja St. Clairin saamat tulokset. UR-ID3 luokittelee huomattavasti paremmin kuin täsmällinen ID3. UR-ID3 saavuttaa tasaisesti yli 94 % luokittelutarkkuuden huolimatta päätöspuun koosta. Sen sijaan täsmällisen ID3:n tulokset hajaantuvat hieman ja jäävät kaikki alle 80 % luokittelutarkkuudesta. Sumean luokittelun approksimaattina eli jakovakiona (ks. kohta 3.2) oli Maherin ja St. Clarin testeissä luku viisi.

Taulukko 8. URID3:n luokittelutarkkuuksia Iris-aineistolla.

Aineisto	ID3	UR-ID3	Sisäsolmuja	Lehtisolmujen lkm
Iris1	78.9 %	94.7 %	6	38
Iris2	71.1 %	94.7 %	4	52
Iris3	75.7 %	94.6 %	4	46

4.1 Testien tuloksia

Juholan [8] SP-ID3-järjestelmän luokittelualgoritmi sisältää muutaman tarkennuksen Maherin ja St. Clairin UR-ID3 algoritmiin [11]. Sumean luokittelun tulokseksi otetaan ensisijaisesti suuremman välttämättömyysasteen (S_n) omaava tulosluokka. Jos kahden luokan välttämättömyysasteet ovat samat, tehdään päätös mahdollisuusasteen (S_p) mukaan. Lisäksi jos jokaisen tulosluokan tukipari on $[S_n, S_p] = [0,0]$, ei päätöstä voida tehdä.

Tutkimus suoritetaan kahdessa vaiheessa. Ensimmäisenä käydään läpi aineiston luokittelu ilman ikkunointia ja toiseksi testataan iteroinnin vaikutusta luokittelutarkkuuteen. Jokaisessa testissä luodaan uusi päätöspuu esimerkkiaineistosta, johon otetaan esimerkkitapauksia satunnaisesti alkuperäisestä aineistosta 10 % - 75 %. Loput tapaukset muodostavat aina testiaineiston.

Sumeuden määräävinä jakovakioina on käytetty lukuja 99, 5 ja ½. Vakiot on valittu siten, että luvut 99 ja 1/2 edustavat sumeuden kumpaakin ääripäätä: jakovakiolla 1/2 saadaan aikaan suurin sumeus ja vastaavasti luvulla 99 pienin. Maher ja St. Clair käyttävät jakovakiona UR-ID3:ssa lukua 5 [11].

4.1.1 Luokittelu ilman ikkunointia

Tutkimuksen tulokset esitetään taulukoissa 9 - 12. Taulukon ensimmäisessä sarakkeessa on testin numero, toisessa täsmällisen ID3:n luokittelutarkkuus (testiaineistosta oikein luokiteltujen osuus prosentteina). Seuraavissa kolmessa sarakkeessa on kerrottu sumean ID3:n luokittelutarkkuudet eri jakovakioilla (99, 5 ja ½). Jokaisen testin päätöspuun koko, sisäsolmujen ja lehtisolmujen lukumäärät, on esitetty viimeisenä. Sarakkeista on lopuksi laskettu vielä keskiarvot ja keskihajonnat.

Taulukko 9. Esimerkkiaineistona 10 % koko aineistosta (15 tapausta).

Testi	Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
		99	5	½	sisä	lehti
1	27.4	37	90.4	65.2	1	13
2	23	28.1	92.6	71.1	1	12
3	31.1	42.2	90.4	94.1	1	12
4	51.1	51.1	80	88.1	1	9

Taulukko 9. Esimerkkiaineistona 10 % koko aineistosta (15 tapausta) (jatkuu).

Testi	Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
		99	5	½	sisä	lehti
5	24.4	49.6	58.5	52.6	1	13
6	57	57	83.7	94.1	1	10
7	41.5	48.9	94.8	94.8	1	12
8	31.1	35.6	94.1	71.1	1	11
9	29.6	45.2	94.8	74.1	1	12
10	53.3	53.3	91.1	78.5	1	10
Ka	**36.95**	**44.8**	**87.04**	**78.37**	**1**	**11.4**
Kh	12.72	9.02	11.13	14.22		

Kun esimerkkiaineistona on vain 10 % eli 15 tapausta koko aineistosta ja testitapauksia loput 135 tapausta, tulokset eivät ole kovin lupaavia. Jo tässä vaiheessa näkyy sumean luokittelun tarkkuus (jakovakiolla 5), vaikka päätöspuiden koot ovat pieniä (1 sisäsolmu ja 9-13 lehtisolmua). Kun sumean ID3:n jakovakiona on viisi, kuudessa testissä kymmenestä saavutetaan yli 90 % luokittelutarkkuus. Taulukon 9 testissä 2 näkyy kuinka suuria eroja voidaan saavuttaa täsmällisen ja sumea luokittelun välillä. Kun täsmällinen ID3 saavuttaa vain 23 % luokittelutarkkuuden, sumea ID3 jakovakiolla 5 luokittelee 92,6 % testitapauksista oikein. Toisaalta taulukon 9 testin 5 tulos on sumealle luokittelulle todella heikko (58,5 %) ja kaikkien 10 % -testien hajonta on sumealle luokittelun jakovakiolla 5 melko suuri. Jakovakiolla 99 sumea luokittelu onnistuu melkein yhtä heikosti kuin täsmällisellä ID3:lla. Jakovakion ½ antamat luokittelutarkkuudet tyydyttäviä, mutta silläkin hajonta on melko suurta. Voidaan siis päätellä, että 10 % 150 tapauksen aineistosta ei keskimäärin riitä erinomaisen tuloksen saavuttamiseen edes sumealle ID3:lle (edes jakovakiolla 5).

Taulukko 10. Esimerkkiaineiston koko 75% aineistosta (112 tapausta).

Testi	Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
		99	5	½	sisä	lehti
1	78.9	76.3	94.7	97.4	5	51
2	84.2	84.2	94.7	86.8	6	53
3	89.5	92.1	92.1	94.7	4	32
4	76.3	89.5	97.4	94.7	5	50
5	89.5	81.6	97.7	94.7	5	33
6	92.1	92.1	94.7	94.7	4	31
7	65.8	81.6	97.4	86.8	5	48
8	78.9	81.6	89.5	89.5	3	26
9	76.3	81.6	86.8	84.2	3	46
10	94.7	97.4	97.7	100	6	43
Ka	**82.62**	**85.8**	**94.27**	**92.35**	**4.6**	**41.3**
Kh	8.97	6.59	3.76	5.19		

Taulukko 11. Esimerkkiaineiston koko 50% aineistosta (75 tapausta)

Testi	Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
		99	5	½	sisä	lehti
1	73.3	84.0	93.3	64.0	3	39
2	80.0	81.3	94.7	81.3	4	28
3	85.3	85.3	96.0	93.3	2	28
4	72.0	80.0	92.0	88.0	2	36
5	86.7	88.0	97.3	94.7	5	34
6	92.0	92.0	94.7	94.7	3	26
7	57.3	62.7	94.7	94.7	2	34
8	70.7	77.3	92.0	93.3	4	41
9	66.7	74.7	98.7	92.0	5	43
10	88.0	88.0	94.7	93.3	3	24
Ka	**77.20**	**81.33**	**94.81**	**88.93**	**3.3**	**33.3**
Kh	11.01	8.40	2.13	9.70		

Taulukko 12. Esimerkkiaineistona 25% alkuperäisestä aineistosta (38 tapausta)

Testi	Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
		99	5	½	sisä	lehti
1	51.8	65.2	95.5	87.5	3	25
2	59.8	71.4	95.5	82.1	2	26
3	67.9	75	95.5	67	3	29
4	56.3	65.2	92	87.5	2	26
5	87.5	87.5	94.6	94.6	2	21
6	47.3	59.8	95.5	92.9	1	23
7	81.3	81.3	93.8	81.3	1	19
8	76.8	76.8	93.8	93.8	1	15
9	66.1	75	93.8	64.3	2	28
10	82.1	82.1	89.3	93.8	1	16
Ka	**67.69**	**73.93**	**93.93**	**84.48**	**1.8**	**22.8**
Kh	13.87	8.66	1.99	11.02		

Taulukkojen 10 - 12 mukaan täsmällisen ID3 luokittelutarkkuus alenee ja tuloksien hajonta suurenee, kun esimerkkiaineiston koko pienenee 75 prosentista 25:een. Samanlainen kehitys on nähtävissä myös sumealla ID3:lla, kun jakovakiona on 99 tai ½. Sen sijaan jakovakiolla 5 luokittelutarkkuus pysyy suunnilleen samana ja hajontakin pienenee.

Jakovakio 99 antaa tuloksen, jota voidaan verrata täsmällisen ID3:n luokittelutarkkuuteen. Koska luku 99 sumeuttaa lukuja vain hieman, on tulokset lähempänä täsmällisen luokittelun tarkkuutta kuin kahden muun jakovakion luokittelutarkkuutta. Luokittelun tarkkuus jakovakiolla 99 on kuitenkin yleensä korkeampi kuin täsmällisen ID3, koka päätös lasketaan sumeassa ID3:ssa koko päätöspuun mukaan. Vain taulukon 10 testissä 5 täsmällinen ID3 luokitteli testitapaukset paremmin.

Jakovakio 5 on testien mukaan täysin ylivertainen verrattuna muihin jakovakioihin. Kolmestakymmenestä testistä vain kolmella aineistolla luokittelutarkkuus on alle 90 %. Huomattavaa on se, kuinka korkea on sumean luokittelun tarkkuus, kun esimerkkiaineistona oli vain 25 % koko aineistosta (taulukko 12). Oikein luokiteltuja testitapauksia saatiin 93.93 % ja hajonnaksikin tuli vain 1.99. Erityisesti taulukon 12 testissä 6 näkyy jakovakion oikean valinnan tärkeys ja siten sumean luokittelun paremmuus; täsmällinen ID3 luokitteli 112 testitapauksesta vain 47.3 % oikein ja sumea ID3 jakovakiolla 99 ylsi vain 59.8 % tarkkuuteen. Jakovakiolla ½ sumea ID3 saavutti jo 92.9 % ja jakovakiolla 5 saavutettiin 95.5 % luokittelutarkkuus. Testeistä näkyy selvästi, että esimerkkiaineiston koolla ei ole kovin suurta merkitystä, jos luokitteluun käytetään sumeaa ID3:sta ja jakovakio on aineistolle sopiva.

Jakovakiolla ½ saadaan aikaan suurin sumeus ja sillä saavutettiinkin usein paremmat tulokset kuin täsmällisellä ID3:lla ja jakovakiolla 99. Mutta mitä pienempi esimerkkiaineisto on, sitä huonommaksi ja epävarmemmaksi se tulee verrattuna jakovakioon 5. Taulukon 12 testeissä 3 ja 9, joissa käytettiin 25 % esimerkkiaineistoa, sumea ID3, jakovakiolla ½ saavuttaa jopa huonomman luokittelutarkkuuden kuin täsmällinen ID3. Tästä voidaan päätellä, että pienellä jakovakiolla saavutettu laaja sumeus ei anna välttämättä parempaa tulosta.

Jakovakion valinnan voidaan ajatella riippuvan aineiston arvojen suuruudesta. Jos aineisto sisältää paljon suuria lukuja, tulee myös jakovakion olla tarpeeksi suuri, jotta luvuille saataisiin tarpeeksi sumeutta. Pieni jakovakio aiheuttaisi suurille luvuille liikaa sumeutta. Vastaavasti aineiston pieniarvoiset luvut vaativat pienen jakovakion, jotta saavutettaisiin edes jonkinlaista sumeutta. Testien mukaan jakovakio 5 on sopiva Iris-aineiston sumeaan luokitteluun.

4.1.2 Luokittelu ikkunoinnilla

Juholan [8] SP-ID3- järjestelmän iteraatioiden lukumäärä on rajoitettu maksimissaan neljään kertaan ikkunointia käytettäessä. Quinlanin mukaan paras päätöspuu löydetään yleensä neljän toiston jälkeen [14]. Ikkunoinnin iteraatiotoistojen lukumäärät vaihtelevat seuraavissa testeissäni kahdesta neljään. Neljännellä kerralla ei kuitenkaan välttämättä

saavuteta täsmällisellä ID3:lla täydellisesti luokittelevaa päätöspuuta. Koska jokaisella iteraatiolla esimerkkiaineistoon lisätään edellisellä kerralla väärinluokitellut tapaukset, testitapausten määrä vähenee aina saman verran**.**

Taulukko 13. Esimerkkiaineistona 75 % koko aineistosta.

Testi		Tapauksia		Täsmällinen ID3	Sumea	ID3		Päätöspuun koko	
	iter.	Esim	Testi		99	5	½	sisä	lehti
1	1	112	38	78.9	78.9	86.8	94.7	4	51
	2	120	30	93.3	93.3	100	93.3	6	59
	3	122	28	100	100	100	96.4	6	63
2	1	112	38	84.2	84.2	89.5	84.2	4	32
	2	118	32	87.5	93.8	100	96.9	6	61
	3	122	28	100	100	100	100	6	64
3	1	112	38	73.7	92.1	92.1	89.5	6	53
	2	122	28	96.4	100	100	100	6	62
	3	123	27	100	100	100	100	6	62
4	1	112	38	84.2	84.2	97.4	84.2	5	32
	2	118	32	75	81.3	100	96.9	6	54
	3	126	24	100	100	100	100	6	64
5	1	112	38	81.6	86.8	92.1	92.1	4	50
	2	119	31	90.3	83.9	96.8	96.8	6	37
	3	122	28	100	96.4	100	100	6	61
6	1	122	38	78.9	81.6	92.1	84.2	5	51
	2	120	30	96.7	96.7	96.7	96.7	6	63
	3	121	29	96.6	96.6	100	100	6	62
	4	122	28	100	100	100	100	6	64
7	1	122	38	73.7	81.6	97.4	94.7	5	49
	2	122	28	100	100	100	100	6	61
8	1	122	38	86.8	89.5	92.1	86.8	4	29
	2	177	33	75.8	87.9	100	100	6	53
	3	125	25	100	100	100	100	6	59
9	1	122	38	73.7	73.7	86.8	86.8	3	43
	2	122	28	96.4	96.4	100	100	6	60
	3	123	27	100	100	100	100	6	65
10	1	112	38	76.3	84.2	92.1	92.1	4	46
	2	121	29	100	100	100	96.6	6	60

Taulukko 14. Esimerkkiaineistona 50 % koko aineistosta.

Testi		Tapauksia		Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
	iter.	Esim	Testi		99	5	½	sisä	lehti
1	1	75	75	84	84	94.7	93.3	2	26
	2	87	63	66.7	74.6	100	98.4	6	45
	3	108	42	100	100	100	100	6	60
2	1	75	75	96	90.7	96	96	4	27
	2	78	72	80.6	87.5	98.6	98.6	5	47
	3	92	58	94.8	96.6	100	100	6	58
	4	95	55	89.1	85.5	100	100	6	37
3	1	75	75	84	84	92	92	2	22
	2	87	63	74.6	84.1	100	93.7	6	49
	3	103	47	93.6	93.6	100	100	6	58
	4	106	44	100	100	100	100	6	63
4	1	75	75	77.3	85.3	94.7	77.3	4	42
	2	92	58	87.9	84.5	96.6	96.6	6	56
	3	99	51	82.4	82.4	96.1	94.1	7	37
	4	108	42	100	100	100	100	6	60
5	1	75	75	80	81.3	93.3	93.3	3	24
	2	90	60	65	76.7	93.3	68.3	6	49
	3	111	39	100	100	100	100	6	63
6	1	75	75	73.3	74.7	93.3	89.3	2	36
	2	95	55	85.5	85.5	98.2	96.4	5	35
	3	103	47	95.7	95.7	100	100	6	57
	4	105	45	97.8	100	100	100	6	63
7	1	75	75	81.3	80	96	89.3	3	39
	2	89	61	86.9	88.5	95.1	95.1	6	33
	3	97	53	94.3	92.5	98.1	92.1	6	59
	4	100	50	96	98	100	100	6	61
8	1	75	75	70.7	76	94.7	93.3	3	36
	2	97	53	90.6	92.5	98.1	98.1	6	58
	3	102	48	95.8	95.8	100	100	6	63
	4	104	46	100	100	100	100	6	61
9	1	75	75	76	88	94.7	89.3	3	41
	2	93	57	89.5	89.5	94.7	80.7	5	31
	3	99	51	94.1	92.2	100	100	6	56
	4	102	48	93.8	93.8	97.9	93.8	7	46
10	1	75	75	93.3	90.7	94.7	93.3	2	23
	2	80	70	80	90	98.6	98.6	5	44
	3	94	56	85.7	82.1	100	98.2	6	36
	4	102	48	93.8	97.9	100	100	6	61

Kun esimerkkiaineistona on 75 % koko aineistosta (taulukko 13), saavutetaan iteraatiotoistoilla aina täydellisesti luokitteleva päätöspuu. Esimerkkiaineiston koko kasvaa toistoilla 121-126 tapaukseen ja päätöspuun koko aina kuuteen sisäsolmuun ja 59-65 lehtisolmuun. On kuitenkin huomattava, että sumea ID3 saavuttaa 100% luokittelutarkkuuden usein ennen täsmällistä ID3:sta. Kun esimerkkiaineistoksi otetaan 50 % alkuperäisestä aineistosta (taulukko 14), ei neljä iteraatiotoistoa enää välttämättä riitä täsmällisen 100 %

luokittelutarkkuuden saavuttamiseen. Viidessä testissä täsmällinen ID3 ei saanut täydellistä luokittelutarkkuutta. Taulukon 14 testissä 2 täsmällinen ID3 saavutti 96 % tarkkuuden, joka on täsmällisen ID3:n paras luokittelutarkkuus taulukkojen 11 ja 14 (ensimmäisellä iteraatiotoistolla) testeissä. Kun esimerkkiaineistoon lisättiin uudet tapaukset, ei seuraavilla toistoilla enää saavutettu yhtä korkeaa tarkkuutta. Luokittelutarkkuudelle olisikin hyvä antaa kynnyskriteeri, jonka ylitettyä ei ikkunointia enää jatketa. Esimerkiksi taulukon 14 testin 2 mukaan täsmällisen ID3:n luokittelutarkkuuden kynnys voisi olla 95 %, jolloin ikkunointi lopetetaan ensimmäisen iteraation jälkeen. Toisaalta, jos tarkastellaan sumeaa luokittelua, 100 % luokittelutarkkuus saavutetaan vasta kolmannella iteraatiokerralla. Esimerkkiaineiston koko kasvaa toistoilla 95-108 tapaukseen ja päätöspuun koko yleensä kuuteen sisäsolmuun ja useimmiten 60-63 lehtisolmuun.

Taulukko 15. Esimerkkiaineistona 25 % koko aineistosta.

Testi		Tapauksia		Täsmällinen ID3	Sumea	ID3		Päätöspuun koko	
	iter.	Esim	Testi		99	5	½	sisä	lehti
1	1	38	112	55.4	64.3	95.5	89.3	3	26
	2	88	62	82.3	88.7	100	98.4	6	51
	3	99	51	88.2	88.2	96.1	94.1	6	38
	4	105	45	97.8	97.8	97.8	97.8	7	42
2	1	38	112	43.8	57.1	94.6	90.2	2	23
	2	101	49	83.7	87.7	95.9	95.9	4	36
	3	109	41	92.7	92.7	97.6	97.6	6	61
	4	112	38	100	100	100	100	6	60
3	1	38	112	87.5	87.5	95.5	95.5	2	20
	2	52	98	54.1	61.2	96.9	94.9	5	32
	3	97	53	98.1	98.1	98.1	100	6	56
	4	98	52	92.3	92.3	100	100	6	58
4	1	38	112	58.9	67.9	92.9	93.8	1	24
	2	84	66	92.4	90.9	97	97	5	35
	3	89	61	93.4	93.4	100	100	6	58
	4	93	57	98.2	100	100	100	6	60
5	1	38	112	69.9	69.9	92.9	79.5	2	16
	2	72	78	74.4	76.9	96.2	61.5	5	41
	3	92	58	98.3	94.8	98.3	100	6	57
	4	93	57	93	94.7	98.2	100	6	57
6	1	38	112	57.1	55.4	70.5	94.6	2	19
	2	86	64	56.3	65.6	95.3	89.1	5	46
	3	114	36	94.4	86.1	100	100	6	39
	4	116	34	100	100	100	100	6	59
7	1	38	112	80.4	78.6	95.5	90.2	3	24
	2	60	90	62.2	70	98.9	90	5	40
	3	94	56	100	96.4	100	100	6	58

Taulukko 15. Esimerkkiaineistona 25 % koko aineistosta (jatkuu).

Testi		Tapauksia		Täsmällinen ID3	Sumea ID3			Päätöspuun koko	
	iter.	Esim	Testi		99	5	½	sisä	lehti
8	1	38	112	83	83	94.6	94.6	1	19
	2	57	93	58.1	65.6	95.7	64.5	5	34
	3	96	54	96.3	94.4	100	100	6	59
	4	98	52	100	100	100	100	6	58
9	1	38	112	73.2	73.2	92.9	92	1	15
	2	68	82	64.6	68.3	96.3	82.9	5	40
	3	97	53	94.3	98.1	98.1	96.2	6	56
	4	100	50	100	100	100	100	6	63
10	1	38	112	82.1	82.1	94.6	93.8	1	16
	2	58	92	73.9	78.3	97.8	96.7	5	39
	3	82	68	89.7	92.6	100	100	6	54
	4	89	61	98.4	98.4	100	98.4	6	58

Kun esimerkkiaineistona on 25 % (taulukko 15), sumea ID3 jakovakiolla 5 saavuttaa iteraatiotoistoilla testiä 5 lukuun ottamatta täydellisen luokittelutarkkuuden. Kuten jo testeissä ilman iteraatiota todettiin, sumea ID3 jakovakiolla 5 luokittelee hyvin jo ensimmäisellä iteraatiotoistolla. Esimerkkiaineiston koko kasvaa ikkunoinnilla noin 100 tapaukseen ja päätöspuun koko noin kuuteen sisäsolmuun ja noin 57 lehtisolmuun.

Ikkunoinnilla saavutetaan tarkemmin luokittelevia päätöspuita, mutta päätöspuun koko voi kasvaa huomattavasti, kun iteraatiotoistoja lisätään. Jos luokittelu suoritetaan sumean ID3:n mukaan, ei ikkunoinnista ole kovin suurta hyötyä. Yleensä oikean jakovakion valinnalla saavutetaan yli 90 % luokittelutarkkuus jo ensimmäisellä kerralla (jakovakioilla 5 ja ½). Jos ikkunointia käytetään, sumealla ID3:lla ja sopivalla jakovakiolla saavutetaan yli 90 % luokittelutarkkuus noin kahdella iteraatiotoistolla. Kuten kohdassa 4.2.1 todettiin, esimerkkiaineiston koolla ei ole kovin suurta merkitystä, kun käytetään sumeaa ID3:sta; yli 90 % luokittelutarkkuus saavutetaan, vaikka esimerkkiaineistona on vain 25 % alkuperäisestä aineistosta.

5 YHTEENVETO

ID3 on yksi tunnetuimmista päätöspuualgoritmeista. Vaikka ID3 ei ehkä edusta enää oppimiskyvyiltään nykypäivän kärkeä, voidaan sen luokittelutarkkuutta parantaa erilaisilla laajennuksilla. Quinlanin oma C4.5 tarjoaa useita parannuksia perinteiseen ID3:een. Jatkuva-arvoisten attribuuttien käsittelyllä vältytään jokaisen arvon kategorioimiselta ja näin yksinkertaistetaan päätöspuurakennetta. Suhteutetun Gain-arvon käyttö päätöspuuinduktiossa auttaa välttämään puun ylisovittamista. Suurikokoiset päätöspuut voidaan myös karsia C4.5 mahdollistamalla laajennuksella. Lisäksi C4.5 mahdollistaa aineiston puuttuvien attribuuttiarvojen käsittelemisen. Monissa tapauksissa ID3:n ja C4.5:n tarkka rajaus on hämärtynyt, koska C4.5 laajentaa melkein jokaista ID3:n oppimisvaihetta.

Jos aineisto sisältää epätarkkuutta ja ristiriitaisuutta, aineistosta luotu päätöspuu ei välttämättä luokittele täydellisesti koko esimerkkiaineistoa. Järjestelmästä riippuen päätöspuun luokittelutarkkuudelle voidaan sallia joku virhemarginaali, mutta joissain asiantuntijajärjestelmissä, esimerkiksi potilasdiagnoosi, virheisiin ei ole juuri varaa. Sumeuden käyttö päätöspuuluokittelussa sietää epätarkan, ristiriitaisen ja jopa puuttuvan tiedon käsittelyn.

UR-ID3 on melko helposti ymmärrettävä kaksivaiheinen ID3-päätöspuun sumeutusprosessi. Ensin aineistosta luodaan täsmällinen ID3-päätöspuu, jonka jälkeen aineisto luokitellaan sumeasti päätöspuun mukaan. Luokitteluvaiheessa kaikkia attribuuttien arvoja tarkastellaan sumeina. Kun perinteisesti testitapaukselle etsitään ainoastaan yksi polku puun juuresta yhteen lehtisolmuun, UR-ID3 tekee päätöksen puun kaikkien polkujen ja lehtisolmujen mukaan eli koko puurakenteen mukaan. Sumeutus antaa päätöksen tekijälle enemmän tietoa kuin perinteinen menetelmä. Sumea päätöspuu voi antaa useita tulosmahdollisuuksia, joista valitaan paras. Valintaa auttavat päätöksien tukiarvot.

Koska UR-ID3 on jälkisumeuttava menetelmä, itse päätöspuuinduktiossa voitaisiin käyttää esimerkiksi C4.5:n tarjoamia laajennuksia. Näin päätöspuun sumeaa luokittelutehokkuutta voitaisiin ehkä parantaa vielä entisestään. Itse asiassa UR-ID3:n sumeaa luokittelumenetelmää voitaisiin soveltaa laajennuksena myös muihin päätöspuualgoritmeihin, kunhan puun luontivaiheessa otetaan huomioon sumeuden asettamat vaatimukset. UR-ID3

sopiikin mainiosti tilanteeseen, jossa jo luotua päätöspuuta halutaan laajentaa sumealla päättelyllä.

Saatujen empiiristen tulosten perusteella voidaan sanoa, että sumeutus parantaa ID3-päätöspuun luokittelutarkkuutta testatulla sovellusalueella; sumea luokittelu sopii erinomaisesti testeissä käytetylle Iris-aineistolle. Saadut tulokset ovat yhdenmukaisia verrattaessa niitä taulukon 8 osoittamiin Maherin ja St. Clairin saamiin tuloksiin. Huomattavaa oli kuinka hyvin sumea luokittelu toimii, vaikka esimerkkiaineistona oli vain pieni osa alkuperäisestä aineistosta. Koska täsmällisessä luokittelussa etsitään vain yhtä oikeaa sääntöpolkua juuresta lehtisolmuun, sen luokittelunopeus on huomattavasti parempi kuin sumean luokittelun, jossa yksittäinen päätös lasketaan koko päätöspuun mukaan. Monimutkaisella ja laajalla aineistolla sumea luokittelu saattaa viedä melkoisesti aikaa.

Vaikka perinteisellä täsmällisellä ID3-algoritmilla voidaan saavuttaa hyviä luokittelu-tuloksia, sen virhealttiutta voidaan pienentää selvästi C4.5:n laajennuksilla ja sumealla luokittelulla.

VIITELUETTELO

[1] Bensaid A.M., Bouhouch N., Bouhouch R., Fellat R., ja Amri R.: *Classification of ECG Patterns Using Fuzzy Rules Derived from ID3-Induced Decision Trees*, IEEE, 34-38, 1998.

[2] Chiang, I-J. ja Hsu J. : *Fuzzy classification trees for data analysis*, Fuzzy Sets and Systems 130, 87-99, 2002.

[3] Chi Z. Ja Yan H.: *ID3-Derived Fuzzy Rules and Optimized Defuzzification fir Handwritten Numeral Recognition*, IEEE Transactions On Fuzzy Systems, Vol 4, No.1, 24-31, 1998.

[4] Flachsbart B., Clair D.C., Holland J. ja Bond W.E.: *Using The ID3 Classification Algorithm to Reduce Data Density*, ACM Press, 1994.

[5] Fomby T. B.: Exercises and Data sets, Internet WWW-sivu, URL: http://faculty.smu.edu/ tfomby/eco5385/data/Iris.xls (2.8.2005).

[6] Han, J. ja Kamber M. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, 2001.

[7] Hunt E.B., Marin J. ja Stone P.J.: *Experiments in Induction*, Academic Press, New York 1966.

[8] Juhola J.: *SP-ID3-järjestelmä*, Tietojenkäsittelytieteen laudatur-harjoitustyö, Joensuun yliopisto, 2005.

[9] Kantardzic, M.: *Data Mining - Concepts, Models, Methods, and Algorithms*, IEEE Press, A John Wiley & Sons, Inc., New Jersey, 2003.

[10] Karttunen H.: *Datan käsittely*, CSC – Tieteellinen laskenta Oy, 1994.

[11] Maher, P. E. ja St.Clair D.: *Uncertain Reasoning in an ID3 Machine Learning Framework,* IEEE International Conference on Fuzzy Systems, 7-12, San Francisco, 1993.

[12] Manchester Metropolitan University: Department of Biological Sciences, Multivariate Statistics, Internet WWW-sivu, URL: http://149.170.199.144/multivar/datafile/data.htm (2.8.2005).

[13] Quinlan J.R.: *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, California, 1993.

[14] Quinlan J.R.: *Learning Efficioent Classification Procedures And Their Application To Chass End Games*, Machine Learning - An Artificial Intelligence Approach, sivut 463-482, Morgan Kaufmann, California, USA, 1983.

[15] Wilson B.: The Machine Learning Dictionary for COMP9414, internet-sivu, http://www.cse.unsw.edu.au/~billw/mldict.html (2.8.2005).
[16] Yuan Y. ja Shaw M. J.: *Induction of fuzzy decision trees*. Fuzzy Sets and Systems 69 (2): 125-139, 1995.
[17] Zimmermann, H.J.: *Fuzzy Set Theory and Its Applications*, Kluwer Academic Publishers, 2nd ed. (1994), Boston, USA.

LIITE 1: IRIS-AINEISTO

Tutkielman lähteenä on Internetistä löytyvä Iris-aineisto [5, 12].

Aineisto sarakkeittain:

Case	tapauksen numero (yht 150 tapausta)
Spec (Species_No)	tulosluokat: setosa (1), versicolor (2) ja virginica (3)
SL	Sepal Length attribuutti
PW	Petal Width attribuutti
PL	Petal Length attribuutti
SW	Sepal Width attribuutti

Case	Spec	SL	SW	PL	PW	Case	Spec	SL	SW	PL	PW
1	1	5,1	3,5	1,4	0,2	31	1	4,8	3,1	1,6	0,2
2	1	4,9	3	1,4	0,2	32	1	5,4	3,4	1,5	0,4
3	1	4,7	3,2	1,3	0,2	33	1	5,2	4,1	1,5	0,1
4	1	4,6	3,1	1,5	0,2	34	1	5,5	4,2	1,4	0,2
5	1	5	3,6	1,4	0,2	35	1	4,9	3,1	1,5	0,2
6	1	5,4	3,9	1,7	0,4	36	1	5	3,2	1,2	0,2
7	1	4,6	3,4	1,4	0,3	37	1	5,5	3,5	1,3	0,2
8	1	5	3,4	1,5	0,2	38	1	4,9	3,6	1,4	0,1
9	1	4,4	2,9	1,4	0,2	39	1	4,4	3	1,3	0,2
10	1	4,9	3,1	1,5	0,1	40	1	5,1	3,4	1,5	0,2
11	1	5,4	3,7	1,5	0,2	41	1	5	3,5	1,3	0,3
12	1	4,8	3,4	1,6	0,2	42	1	4,5	2,3	1,3	0,3
13	1	4,8	3	1,4	0,1	43	1	4,4	3,2	1,3	0,2
14	1	4,3	3	1,1	0,1	44	1	5	3,5	1,6	0,6
15	1	5,8	4	1,2	0,2	45	1	5,1	3,8	1,9	0,4
16	1	5,7	4,4	1,5	0,4	46	1	4,8	3	1,4	0,3
17	1	5,4	3,9	1,3	0,4	47	1	5,1	3,8	1,6	0,2
18	1	5,1	3,5	1,4	0,3	48	1	4,6	3,2	1,4	0,2
19	1	5,7	3,8	1,7	0,3	49	1	5,3	3,7	1,5	0,2
20	1	5,1	3,8	1,5	0,3	50	1	5	3,3	1,4	0,2
21	1	5,4	3,4	1,7	0,2	51	2	7	3,2	4,7	1,4
22	1	5,1	3,7	1,5	0,4	52	2	6,4	3,2	4,5	1,5
23	1	4,6	3,6	1	0,2	53	2	6,9	3,1	4,9	1,5
24	1	5,1	3,3	1,7	0,5	54	2	5,5	2,3	4	1,3
25	1	4,8	3,4	1,9	0,2	55	2	6,5	2,8	4,6	1,5
26	1	5	3	1,6	0,2	56	2	5,7	2,8	4,5	1,3
27	1	5	3,4	1,6	0,4	57	2	6,3	3,3	4,7	1,6
28	1	5,2	3,5	1,5	0,2	58	2	4,9	2,4	3,3	1
29	1	5,2	3,4	1,4	0,2	59	2	6,6	2,9	4,6	1,3
30	1	4,7	3,2	1,6	0,2	60	2	5,2	2,7	3,9	1,4

Case	Spec	SL	SW	PL	PW	Case	Spec	SL	SW	PL	PW
61	2	5	2	3,5	1	106	3	7,6	3	6,6	2,1
62	2	5,9	3	4,2	1,5	107	3	4,9	2,5	4,5	1,7
63	2	6	2,2	4	1	108	3	7,3	2,9	6,3	1,8
64	2	6,1	2,9	4,7	1,4	109	3	6,7	2,5	5,8	1,8
65	2	5,6	2,9	3,6	1,3	110	3	7,2	3,6	6,1	2,5
66	2	6,7	3,1	4,4	1,4	111	3	6,5	3,2	5,1	2
67	2	5,6	3	4,5	1,5	112	3	6,4	2,7	5,3	1,9
68	2	5,8	2,7	4,1	1	113	3	6,8	3	5,5	2,1
69	2	6,2	2,2	4,5	1,5	114	3	5,7	2,5	5	2
70	2	5,6	2,5	3,9	1,1	115	3	5,8	2,8	5,1	2,4
71	2	5,9	3,2	4,8	1,8	116	3	6,4	3,2	5,3	2,3
72	2	6,1	2,8	4	1,3	117	3	6,5	3	5,5	1,8
73	2	6,3	2,5	4,9	1,5	118	3	7,7	3,8	6,7	2,2
74	2	6,1	2,8	4,7	1,2	119	3	7,7	2,6	6,9	2,3
75	2	6,4	2,9	4,3	1,3	120	3	6	2,2	5	1,5
76	2	6,6	3	4,4	1,4	121	3	6,9	3,2	5,7	2,3
77	2	6,8	2,8	4,8	1,4	122	3	5,6	2,8	4,9	2
78	2	6,7	3	5	1,7	123	3	7,7	2,8	6,7	2
79	2	6	2,9	4,5	1,5	124	3	6,3	2,7	4,9	1,8
80	2	5,7	2,6	3,5	1	125	3	6,7	3,3	5,7	2,1
81	2	5,5	2,4	3,8	1,1	126	3	7,2	3,2	6	1,8
82	2	5,5	2,4	3,7	1	127	3	6,2	2,8	4,8	1,8
83	2	5,8	2,7	3,9	1,2	128	3	6,1	3	4,9	1,8
84	2	6	2,7	5,1	1,6	129	3	6,4	2,8	5,6	2,1
85	2	5,4	3	4,5	1,5	130	3	7,2	3	5,8	1,6
86	2	6	3,4	4,5	1,6	131	3	7,4	2,8	6,1	1,9
87	2	6,7	3,1	4,7	1,5	132	3	7,9	3,8	6,4	2
88	2	6,3	2,3	4,4	1,3	133	3	6,4	2,8	5,6	2,2
89	2	5,6	3	4,1	1,3	134	3	6,3	2,8	5,1	1,5
90	2	5,5	2,5	4	1,3	135	3	6,1	2,6	5,6	1,4
91	2	5,5	2,6	4,4	1,2	136	3	7,7	3	6,1	2,3
92	2	6,1	3	4,6	1,4	137	3	6,3	3,4	5,6	2,4
93	2	5,8	2,6	4	1,2	138	3	6,4	3,1	5,5	1,8
94	2	5	2,3	3,3	1	139	3	6	3	4,8	1,8
95	2	5,6	2,7	4,2	1,3	140	3	6,9	3,1	5,4	2,1
96	2	5,7	3	4,2	1,2	141	3	6,7	3,1	5,6	2,4
97	2	5,7	2,9	4,2	1,3	142	3	6,9	3,1	5,1	2,3
98	2	6,2	2,9	4,3	1,3	143	3	5,8	2,7	5,1	1,9
99	2	5,1	2,5	3	1,1	144	3	6,8	3,2	5,9	2,3
100	2	5,7	2,8	4,1	1,3	145	3	6,7	3,3	5,7	2,5
101	3	6,3	3,3	6	2,5	146	3	6,7	3	5,2	2,3
102	3	5,8	2,7	5,1	1,9	147	3	6,3	2,5	5	1,9
103	3	7,1	3	5,9	2,1	148	3	6,5	3	5,2	2
104	3	6,3	2,9	5,6	1,8	149	3	6,2	3,4	5,4	2,3
105	3	6,5	3	5,8	2,2	150	3	5,9	3	5,1	1,8