KATEGORISEN DATAN RYHMITTELY

Antti Pöllänen

17.6.2010

Itä-Suomen yliopisto
Tietojenkäsittelytieteen laitos
Pro gradu -tutkielma

Tiivistelmä

Tutkielmassa tutustutaan ryhmittelyn perusteisiin, kategorisen datan perusominaisuuksiin sekä erityisesti kategorisen datan ryhmittelyyn. Kategorisen datan ryhmittelyä lähestytään numeerisen datan ryhmittelyn kautta. Ryhmittelymenetelmissä keskitytään kokoaviin hierarkisiin menetelmiin ja iteratiivisiin menetelmiin. Erityisesti mielenkiinnon kohteena ovat entropiaa ryhmittelykriteeniä käyttävät menetelmät. Ryhmittelymenetelmistä käsitellään tarkemmin ACE, k-distributions, k-histograms, k-medoids, k-modes ja ROCK menetelmiä. Näiden menetelmien lisäksi perehdytään entropian käyttämiseen iteratiivisessa menetelmässä, josta käytetään tässä tutkielmassa nimeä k-entropies. Kokeellisessa osassa vertaillaan menetelmien käyttämää aikaa sekä niiden tuottamien ryhmittelyjen laatua. Laadun mittarina käytetään ensisijaisesti entropiaa ja lisäksi menetelmien soveltuvuutta käytäntöön tarkastellaan luokitteluvirheen avulla.

ACM-luokat (ACM Computing Classification System, 1998 version): H.3.3, I.5.3

Avainsanat: entropia, kategorinen data, klusterointi, kokoava, k-means, ryhmittely

Esipuhe

Erikoistyöni ja MOPSI projekti ovat suurelta osin edesauttaneet tämän työn syntyä. MOPSI projekti (Mobiilit PaikkatietoSovellukset ja Internet) on Teknologian ja innovaatioiden kehittämiskeskuksen (TEKES) ja Euroopan aluekehitysrahaston (EAKR) rahoittama.

Haluan kiittää ohjaajaani FT Ville Hautamäkeä väsymättömästä ohjauksesta. Kiitokset myös professori Pasi Fräntille, joka johdatti minut aiheen pariin.

Lyhenteet

ACE	Agglomerative Categorical clustering with Entropy criterion
CU	Category Utility
DET	Detection Error Tradeoff
MSE	Mean Square Error
ROCK	A Robust Clustering Algorithm for Categorical Attributes

Symbolit

A_i	attribuutti i
$\mathbf{C}$	ryhmittely
$\mathbf{C}_i$	ryhmä i
D	aineiston alkioiden ulotteisuus
$F(A_i = V_v)$	attribuutissa i kategorian v esiintymien lukumäärä
$F(A_i = V_v \mid S)$	attribuutissa i kategorian v esiintymien lukumäärä joukossa S
I	iteraatioiden lukumäärä
K	ryhmien lukumäärä
M_i	attribuutin i kategorioiden lukumäärä
M_{avg}	attribuutin kategorioiden lukumäärän keskiarvo
N	aineiston koko
N_i	ryhmän i koko
$O(f(x))$	funktion $f(x)$ asymptoottinen aikavaativuus
$p(A_i = V_v)$	attribuutissa i kategorian v esiintymien osuus
$p(A_i = V_v \mid S)$	attribuutissa i kategorian v esiintymien osuus joukossa S
R_{avg}	naapureiden keskimääräinen lukumäärä
R_{max}	naapureiden suurin lukumäärä
T_L	logaritmifunktion aikavaativuus
V_i	kategoria i
$\mathbf{X}$	aineisto
$\mathbf{x}_i$	aineiston i:s vektori
x_{ij}	aineiston i:nnen vektorin j:s arvo

Sanasto

aineisto	dataset
hyvyysfunktio	goodness measure
Jaccard-kerroin	Jaccard coefficient
kokoava	agglomerative
keskinäisinformaatio	mutual information
kynnysarvo	threshold
käsitehierarkia	concept hierarchy
käsitteiden kerääminen	concept acquisition
maksufunktio	cost function
näytteistäminen	sampling
ohjaamaton oppiminen	unsupervised learning
ohjattu oppiminen	supervised learning
osajoukko	subset
osittava	divisive
ositus	partitioning
poikkeama	outlier
puhtaus	purity
ryhmiteltävyys	clustering tendency
ryhmittely	clustering
ryhmittelyn oikeellisuus	cluster validity
sekaannusmatriisi	confusion matrix
tietämyksen muodostus	knowledge discovering
yhteistodennäköisyys	joint probability
yksinkertainen epäyhteensopivuus	simple matching dissimilarity
yksinkertainen yhteensopivuuskerroin	simple matching coefficient

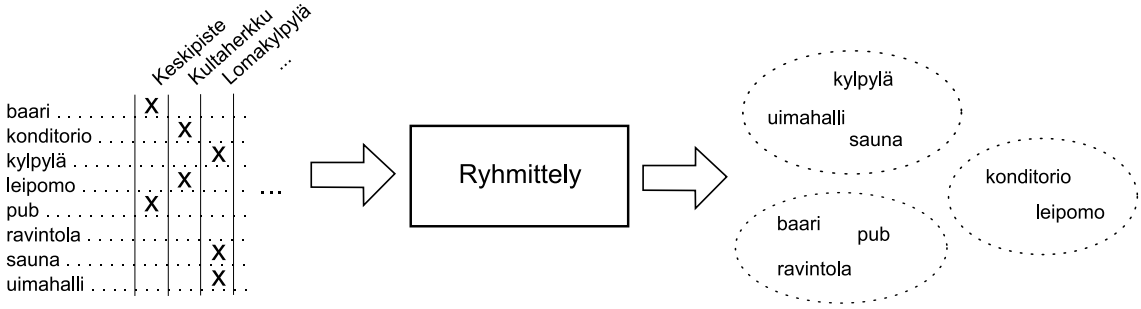
Sisältö

1 Johdanto

Dataa on olemassa valtavia määriä eri aloilta, kuten lääketieteestä, geologiasta, taloudesta ja markkinoinnnista. Ihmiset eivät pysty tutkimaan manuaalisesti kaikkea dataa ajan puutteen ja työn kustannuksien takia. Helposti päädytään tilanteeseen, että kerättyä dataa ei hyödynnetä. Syntyy datahautoja - arkistoja joita käytetään harvoin. Datan hyödyntämiseksi täytyy kehittää automatisoituja menetelmiä [17].

Ryhmittely on yksi keino datan käsittelyyn. Ryhmittelyssä pyritään jakamaan dataa ryhmiin siten, että ryhmien sisällä erot ovat mahdollisimman pieniä ja ryhmien välillä erot ovat mahdollisimman suuria. Ryhmittelyä voidaan hyödyntää useissa eri sovelluksissa. Sovellusten tärkeimmät osa-alueet ovat tiedon tiivistäminen, hypoteesien etsiminen ja testaaminen sekä ennustaminen [35].

Esimerkiksi MOPSI projektissa palveluhakemistossa on palveluja, joihin kuhunkin liittyy joukko avainsanoja. Vastaavasti avainsanoihin liittyy joukko palveluja. Avainsanat voidaan ryhmitellä palvelujen perusteella ja lopputuloksena saadaan avainsanaryhmiä. Ideaa on havainnollistettu kuvassa 1. Ryhmien avulla voidaan tarjota käyttäjälle avainsanat selkeästi jäsenneltynä. Ero perinteiseen palveluhakemistoon on siinä, että jäsentely saadaan tuotettua automaattisesti.



Kuva 1: Avainsanojen ryhmittely

Usein ryhmittelyssä on totuttu käsittelemään numeerista dataa, josta esimerkkejä ovat kappaleen massa ja sijainti koordinaatistossa. Yksi tärkeimmistä numeerisen datan ominaisuuksista on eri arvojen välinen etäisyys, joka on myös ryhmittelymenetelmien perusta. Etäisyyttä ei voida määritellä vastaavalla tavalla kategoriselle datalle, missä muuttujan mahdolliset arvot ovat kategorioita. Esimerkiksi henkilön sukupuoli on kategorinen muuttuja, jonka mahdollisia arvoja mies ja nainen. Käytännössä voidaan ainoastaan suorittaa vertailu kuuluvatko arvot

samaan kategoriaan vai eivät. Tämä rajoite asettaa haasteita myös ryhmittelymenetelmille eikä numeeriselle datalle tarkoitettuja menetelmiä voida hyödyntää suoraan kategoriselle datalle. Koska kategorista dataa kuitenkin esiintyy monissa sovelluksissa, on tärkeää kehittää menetelmiä kategorisen datan ryhmittelyyn. Kategorisen datan käsittely on erityisen tärkeää tiedonlouhinnassa, jossa ryhmittely on yksi merkittävä apuväline.

Tutkielman toisessa luvussa tutustutaan ryhmittelyn perusteisiin ja selitetään mitä on kategorinen data. Luvussa 3 pohditaan kuinka ryhmittelyn laatua voidaan arvioida kategorisen datan tapauksessa. Luvuissa 4, 5 ja 6 kuvataan kategorisen datan ryhmittelymenetelmiä teoreettisesta näkökulmasta. Luvussa 7 ja 8 vertaillaan eri ryhmittelymenetelmiä käytännössä ja lopuksi luvussa 9 on lyhyt yhteenveto tutkielman pääkohdista.

2 Ryhmittely ja kategorinen data

2.1 Ryhmittely

Luokittelulla tarkoitetaan datan jakamista ennalta määriteltyihin *luokkiin*. Tätä kutsutaan *ohjatuksi oppimiseksi*. *Ryhmittelyllä* tarkoitetaan datan jakamista samankaltaisiin ryhmiin datan ominaisuuksien perusteella. Ryhmittely poikkeaa luokittelusta siinä, että ryhmittelyn apuna ei ole ennalta tiedossa olevia *luokkarajoja*. Tämän vuoksi ryhmittelyä sanotaan *ohjaamattomaksi oppimiseksi* [35]. On myös mahdollista sanoa, että ryhmittely on havaintojen kautta oppimista ja luokittelu on esimerkkien kautta oppimista [17].

Ryhmittelyongelmassa [36] on alussa *aineisto* $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N\}$. Aineiston yksittäinen *datavektori* tai lyhyemmin *vektori* sisältää D *attribuuttia* eli $\mathbf{x}_j = (x_{j1}, x_{j2}, ..., x_{jD})$. Vektoreille pyritään löytämään *ositus* $\mathbf{C} = \{\mathbf{C}_1, \mathbf{C}_2, ..., \mathbf{C}_K\}$, joka toteuttaa rajoitteet 1-3.

$$\mathbf{C}_i \neq \emptyset, i = 1, ..., K \tag{1}$$

$$\bigcup_{i=1}^{K} \mathbf{C}_i = \mathbf{X} \tag{2}$$

$$\mathbf{C}_i \cap \mathbf{C}_j = \emptyset, i, j = 1, ..., K \wedge i \neq j. \tag{3}$$

Ehto 1 takaa, ettei yksikään ryhmä ole tyhjä. Ehto 2 takaa, että jokainen vektori kuuluu ainakin yhteen ryhmään ja ehto 3 takaa, että vektorit kuuluvat vain yhteen ryhmään.

Näiden ehtojen vallitessa ryhmittelyalgoritmeilla pyritään ryhmittelemään tietoa siten, että ryhmien sisällä data on mahdollisimman samanlaista ja eri ryhmien välillä data on mahdollisimman erilaista [29; 26]. Algoritmisesta näkökulmasta on tärkeää antaa tarkka määritelmä datan samanlaisuudelle ja erilaisuudelle. Aihetta käsitellään tarkemmin luvussa 3.

Ei ole olemassa yleistä ryhmittelymenetelmää, joka soveltuisi kaikkiin mahdollisiin moniulotteisissa aineistoissa esiintyviin rakenteisiin [23]. Eri menetelmät, tai jopa samat menetelmät eri parametreilla, saattavat tuottaa hyvinkin erilaisia ryhmittelytuloksia, jotka kaikki näyttävät hyväksyttäviltä [5].

2.2 Mitä on kategorinen data

Erilaiset muuttujat voidaan jakaa mitta-asteikon mukaisesti neljään eri luokkaan [2; 34]:

- Nominaaliasteikko
- Järjestysasteikko
- Välimatka-asteikko
- Suhdeasteikko

Nominaaliasteikossa arvoja ei voida järjestää. Käytännössä arvoista voidaan vain sanoa, ovatko arvot samoja vai eivät. Nominaalisia muuttujia ovat esimerkiksi henkilön sukupuoli (mies, nainen) tai siviilisääty (naimaton, naimisissa, eronnut, leski). *Järjestysasteikossa* arvot voidaan järjestää, mutta niiden välillä ei ole mielekästä etäisyyttä. Järjestysasteikoon kuuluu usein kyselyissä nähtävät vaihtoehdot: täysin eri mieltä, jokseenkin eri mieltä, jokseenkin samaa mieltä, täysin samaa mieltä.

Välimatka-asteikossa arvot voidaan järjestää ja arvojen välinen etäisyys voidaan laskea, mutta arvojen suhteellinen vertailu ei ole mielekästä. Eri lämpötila-asteikot kuten Celcius ja Farenheit ovat esimerkkejä välimatka-asteikosta. Lämpötilaerojen vertaaminen on vielä mielekästä, mutta suhteellinen vertailu tulee mahdottomaksi viimeistään siinä vaiheessa kun vaihdetaan lämpötila-asteikkoa. *Suhdeasteikossa* järjestyksen ja etäisyyden lisäksi arvoilla on yksikäsitteinen nollapiste. Tällöin myös arvojen suhteellinen vertailu on mielekästä. Suhdeasteikkoon sijoittuu muun muassa pituus, jonka mukaan voidaan järjestää. Lisäksi laskea pituuseroja ja sanoa että toinen kappale on kaksi kertaa toista pidempi. Taulukossa 1 on taulukoitu eri mitta-asteikkojen operaatiot.

Asteikko	Yhtäsuuruus ($=$)	Vertailu ($<, >$)	Erotus ($-$)	Suhde ($:$)
Nominaali	✓			
Järjestys	✓	✓		
Välimatka	✓	✓	✓	
Suhde	✓	✓	✓	✓

Taulukko 1: Operaatiot eri mitta-asteikoissa

Numeerisella datalla tarkoitetaan välimatka- ja suhdeasteikon arvoja sisältävää dataa. *Kategorinen data* on puolestaan nominaali- ja järjestysasteikon arvoja sisältävää dataa [2].

Kategorinen data voidaan kääntää suomeksi myös *luokittelevaksi dataksi*. Voidaan ajatella, että attribuutin arvot kuuluvat tiettyihin luokkiin.

Edellä mainittu on tarkka jaottelu numeeriselle ja kategoriselle datalle. Kuitenkin ryhmittelyssä riittää toisinaan jaottelu pelkästään numeeriseen ja kategoriseen dataan [20]. Yleensä kategorisella datalla tarkoitetaan nimenomaan nominaaliasteikon dataa, vaikka sitä ei erikseen mainita. Myöskin tässä tutkielmassa kategorinen datalla tarkoitetaan erityisesti dataa nominaaliasteikkossa. Etuna kategorisen datan olettamisessa nominatiiviseksi on yksinkertaisuus. Kaikki kategorinen data voidaan käsitellä samalla tavalla, koska nominatiivisen datan käsittelyssä käytettävä operaatiojoukko on pienin mahdollinen. Haittana on tietysti se, että kaikkea mahdollista tietoa ei hyödynnetä ja ryhmittelyn laatu voi kärsiä. Esimerkiksi koulutustasoa ilmaisevan arvon tapauksessa kandidaatit kannattaa sijoittaa maistereiden kanssa samaan ryhmään ennemmin kuin muodostaa kandidaateista ja tohtoreista erillinen ryhmä. Ilman järjestystietoa ei ole mahdollista sanoa kumpi vaihtoehto on parempi.

Binääridata on kategorisen datan erikoistapaus. Binäärimuuttujalla on kaksi mahdollista arvoa: tosi tai epätosi [17]. Binäärimuuttujat voivat olla symmetrisiä tai epäsymmetrisiä. Symmetrisen binäärimuuttujan molemmilla arvoilla on sama painoarvo. Esimerkki symmetrisesta binäärimuuttujasta on muuttuja, joka ilmaisee onko henkilö mies. Molemmat sukupuolet ovat yhtä yleisiä. Epäsymmetrisen binäärimuuttujan arvot eivät ole yhtä merkityksellisiä. Esimerkiksi positiivista HIV-testin tulosta ilmaiseva muuttuja on epäsymmetrinen, koska positiivinen tulos on harvinaisuus [17].

Transaktiodata on eräs binääridatan erikoistapaus. Transaktiodata muodostuu tapahtumista kuten esimerkiksi ostotapahtumista. Ostotapahtumassa attribuutteina ovat kaikki mahdolliset tuotteet ja ainoastaan ostetut tuotteet saavat arvon tosi. Tehokkaalla ja tarkalla transaktiodatan ryhmittelyllä on paljon sovellusalueita [37] ja useat menetelmät [37; 29] onkin suunniteltu erityisesti tämän tyyppiselle datalle. Aineistojen suuri koko, suuri attribuuttien lukumäärä sekä attribuuttien epäsymmetrisyys tekevät transaktiodatan ryhmittelystä haastavan tehtävän [37].

Nominaaliset eli kategoriset muuttujat voidaan esittää epäsymmetrisien binäärimuuttujien joukkona [17]. Useissa eri lähteissä on käsitelty kategoristen muuttujien muuttamista binäärisiksi [26], [30]. Kategorinen attribuutti, joka sisältää M mahdollista arvoa, voidaan muuttaa joukoksi binäärisiä *indikaattorimuuttujia* [26]. Näitä indikaattorimuuttujia on M kappaletta. Jos indikaattorimuuttujia merkitään $v^{(1)}, v^{(2)}, ..., v^{(M)}$, tarkoittaa $v^{(i)} = 1$ sitä, että

kategorinen attribuutti saa i:nnen arvon. Tällöin muut kyseisen indikaattorimuuttujajoukon muuttujat saavat arvon nolla.

2.3 Kategorisen datan ryhmittelyn tarpeellisuus

Yleisimmät tarkoitukset ryhmittelylle on datamäärän pienentäminen ja käsitteiden kerääminen. Jollain aloilla, kuten astronomiassa, tietoa kerätään paljon ja ryhmittely on hyödyllinen tiivistysmenetelmä pienentämään valtavia määriä moniulotteista dataa. Toisilla aloilla ryhmät ovat mielenkiintoisia ainoastaan, jos niille on mielenkiintoinen tulkinta tai jos ne tarjoavat uutta tietoa. Esimerkiksi markkinoinnissa kuluttajaryhmät ovat tärkeitä yrityksen markkinointistrategian kannalta [30]. Kategorista dataa löytyy käytännössä kaikkialla [41], joten sen käsittelyltä ei voida välttyä myöskään ryhmittelyssä.

Yleensä on totuttu ryhmittelemään numeerista dataa ja kategorinen data on jäänyt vähemmälle huomiolle. Aiemmin kategorista dataa on pyritty ryhmittelemään muuntamalla kategoriset muuttujan numeerisiksi [23; 30]. Kuitenkin kategorista dataa on paljon eikä numeeriselle datalle tarkoitetut menetelmät kykene ryhmittelemään kategorista dataa hyvin [16; 26]. Näin ollen on tärkeää kehittää kategoriselle datalle sopivia ryhmittelymenetelmiä [20]. Lisäksi aineistoissa voi olla sekä numeerisia että kategorisia attribuutteja [17]. Yksi lähestymistapa on ryhmitellä aineisto erikseen jokaisella attribuuttityypillä, mutta ongelmaksi jää tulosten yhdistäminen. Toinen mahdollinen lähestymistapa on muuttaa kaikki attribuutit samantyyppisiksi. On olemassa ryhmittelymenetelmiä, joihin on sisäänrakennettu tapa käsitellä erityyppisiä attribuutteja. Tässä tutkielmassa keskitytään kuitenkin vain ja ainoastaan kategoristen muuttujien ryhmittelyyn.

Tiedonlouhinnasta ja tietämyksen muodostamisesta on tulossa entistä tärkeämpi tehtävä [16]. Esimerkiksi yrityksillä on olemassa valtavat määrät tietoa kerättynä tietokantoihin, koskien muun muassa asiakkaiden ostokäyttäytymistä ja yhteydenottoja. Yritykset voisivat tehostaa markkinointia ja mainontaa, jos aikaisemmin tuntemattomat asiakkaiden käyttäytymiskaavat voitaisiin selvittää aikaisemman tiedon perusteella. Suomessa erilaisien kanta-asiakaskorttijärjestelmien avulla saadaan koottua tietoa asiakkaiden ostokäyttäytymisestä. Tiedonlouhinnalla on toki merkitystä myös ihmisten yksityisyyden kannalta, mutta tässä tutkielmassa keskitytään kuitenkin puhtaasti teknisiin ongelmiin.

Tiedonlouhinnassa on datan käsittelyyn muitakin menetelmiä kuin ryhmittelymenetelmät [17]. Osa näistä soveltuu myös kategoriselle datalle. Ryhmittely voi toimia itsenäisenä työkaluna datan jakaumien ja ryhmien ominaisuuksien selvittämisessä. Vaihtoehtoisesti ryhmittelyllä voidaan esikäsitellä data muita tiedonlouhinnan menetelmiä varten [17].

2.4 Kategorisen datan ryhmittelyn haasteet

Ensimmäinen haaste kategorisen datan ryhmittelyssä on se, ettei arvoilla ole luonnollista järjestystä ja perinteiset etäisyysfunktiot ovat tehottomia [40]. Kategorisen datan ryhmittely vaatii uudenlaista lähestymistapaa, koska etäisyydet ja kriteerit täytyy määritellä uudelleen. Etäisyyksiä tarkastellaan lähemmin kohdassa 2.6 ja kriteeriä luvussa 3.

Intuitiivisesti kategorisen datan ryhmittely tuntuu helpommalta kuin numeerisen datan ryhmittely, koska arvojoukko on äärellinen ja usein attribuuteilla on vain muutamia arvoja. Kuitenkin vektorit voidaan järjestää edelleen yhtä monella tavalla eri ryhmiin ja vaikka yksittäisen attribuutin arvojoukko on pieni, usein suuri ulottuvuuksien määrä vaikeuttaa ryhmittelyä [14; 40]. Lisäksi ongelmaksi mainitaan se, etteivät kaikki attribuutit ole olennaisia oikean ryhmittelyn kannalta [40]. Transaktiodatan tapauksessa attribuuttien eri arvot eivät ole yhtä merkityksellisiä [16].

Tiedonlouhinnassa, jossa kategorista dataa usein joudutaan käsittelemään, menetelmille on pitkä lista tärkeitä ominaisuuksia. Näitä ominaisuuksia ovat skaalautuvuus, eri tyyppisten attribuuttien käsittely, eri muotoisten ryhmien löytäminen, parametrien pieni lukumäärä, poikkeamien käsittely, riippumattomuus vektoreiden järjestyksestä, moniulotteisen datan käsittely, rajoitteiden käsittely, tulosten tulkittavuus ja käytettävyys [17].

2.5 Kategorisen datan visualisointi

Datan visualisointia voidaan hyödyntää sekä datan analysoinnissa että esittämisessä. Analysoinnissa datan visualisointi on lähestymistapa, jossa keskitytään oivaltavaan graafinen esitykseen [12]. Oivaltavuus tarkoittaa sitä, että visualisoinnin avulla pyritään paljastamaan uusia piirteitä. Oivaltavuuteen pyrkivät menetelmät tuovat esiin piirteitä, joita ei tiedetä eikä oleteta olevan. Datan esittämisessä puolestaan pyritään tuomaan esityksen lukijalle tietty asia selväksi. Eri tarkoituksi varten on olemassa monia eri tapoja visualisoidan dataa [12].

Esitystavan valinnassa on tärkeää huomioida mitä visualisoinnilla halutaan saavuttaa.

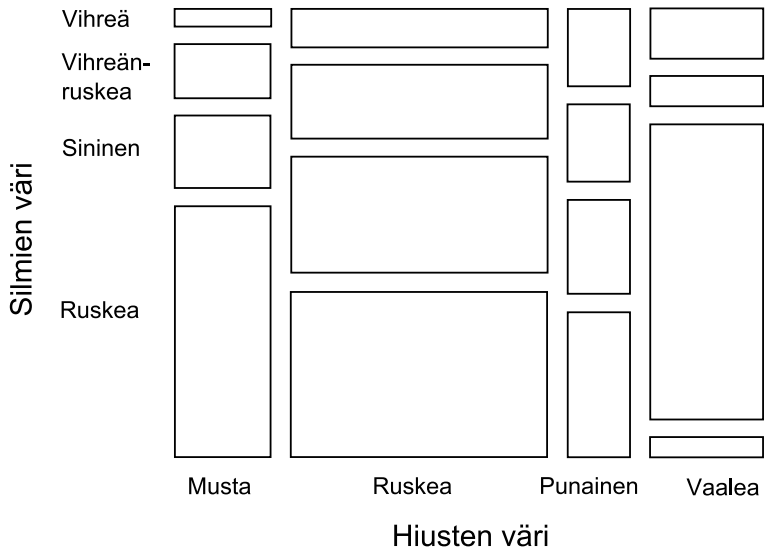
Kategoriselle datalle ei ole yksiselitteistä geometrista tulkintaa, joten visualisointi geometrian kautta ei ole soveliasta. Kategorista dataa visualisoidaan useimmiten taulukoiden avulla [11]. Yleensä eri sovelluksissa data tallennetaan *vektorimuodossa* eli yksittäisinä vektoreita [12]. Taulukoita varten tarvitaan *frekvenssimuotoa*, jossa lasketaan eri kategorioiden esiintymien lukumäärät ja luodaan taulukko. Taulukossa 2 on esitetty dataa, jonka vektoreilla on kolme attribuuttia.

Lääke	Sukupuoli	Ei vaikutusta	Vähäinen	Merkittävä	Yhteensä
Oikea lääke	Nainen	6	5	16	27
	Mies	7	2	5	14
Lumelääke	Nainen	19	7	6	32
	Mies	10	0	1	11
Yhteensä		42	14	28	84

Taulukko 2: Esimerkkki kategorisen datan esittämisestä frekvenssimuodossa.

Vektorimuodon etuna on se, ettei tietoa yksittäisistä tapauksista menetetä. Taulukosta ei voida läheskään aina palauttaa alkuperäisiä datavektoreja [12]. Sen sijaan vektoriesityksestä voidaan aina rakentaa taulukko.

Jos ulottuvuuksia on vähän, datasta voidaan piirtää mosaiikkikuvaaja [12; 33]. Kuvassa 2 on esimerkki mosaiikkikuvaajasta.

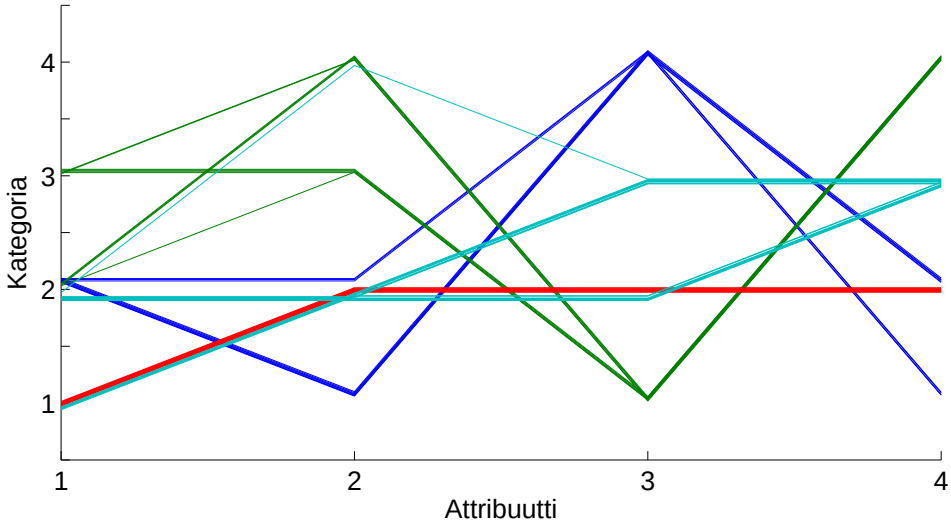


Kuva 2: Mosaiikkikuvaaja

Kuvasta 2 nähdään eri hiusten ja silmien värien riippuvuus. Jokaisen sarakkeen leveys vastaa

hiusten värin prosentuaalista osuutta koko aineistossa. Korkeus puolestaan määräytyy jokaiseen sarakkeeseen erikseen silmien värin esiintymien mukaan. Kuvaajasta huomaa helposti tavallista vahvemmat korrelaatiot kuten sinisten silmien ja vaaleiden hiuksien yhteyden.

Jos ulottuvuuksia on enemmän, voidaan käyttää rinnakkaisia koordinaatteja [33]. Kuvassa 3 on esimerkki rinnakkaisista koordinaateista, jossa värien avulla ilmaistaan vektoreiden luokat. Esimerkkissä on yhteensä 47 vektoria. Data on otettu soijapapuaineistosta, joka on esitelty tarkemmin kohdassa 7.2.3. Neljä attribuuttia on valittu siten, että eri luokkien erot tulevat selkeästi esille.



Kuva 3: Rinnakkaiskoordinaatisto-kuvaaja

Rinnakkaisissa koordinaateissa ulottuvuuksien järjestyksellä on suuri merkitys kuvaajan tulkittavuuteen [33].

Ryhmittelyn näkökulmasta koko aineiston visualisoiminen edellä nähdyillä tavoilla ei paljasta tietoa ryhmittelystä. Sen sijaan ryhmien visualisoiminen erikseen voi paljastaa ryhmien erikoispiirteitä.

2.6 Etäisyys- ja samankaltaisuusmitat

Lähes kaikki ryhmittelymenetelmät lähtevät liikkeelle siitä, että datavektorien välille määritellään etäisyys tai samankaltaisuus [34]. Olkoon $\mathbf{X}$ vektorijoukko. Funktio $\mathrm{d} : \mathbf{X} \times \mathbf{X} \rightarrow \mathbb{R}$ on etäisyysfunktio, jos mielivaltaisilla arvoilla $x, y \in \mathbf{X}$ toteutuu ehdot 4-6.

$$\mathrm{d}(x, y) \geq d_0 \tag{4}$$

$$\mathrm{d}(x, x) = d_0 \tag{5}$$

$$\mathrm{d}(x, y) = \mathrm{d}(y, x) \tag{6}$$

Kaavoissa käytetty d_0 on jokin äärellinen reaaliluku. Ehdot 4 ja 5 määrittävät, että etäisyysfunktio d saa pienimmän arvonsa kun vektorit ovat identtiset. Lisäksi 6 ilmaisee funktion symmetrisyyden. Etäisyysfunktiota voidaan sanoa metriseksi, jos ehtojen 4-6 lisäksi toteutuu ehdot 7 ja 8.

$$\mathrm{d}(x, y) = d_0 \Rightarrow x = y \tag{7}$$

$$\mathrm{d}(x, z) \leq \mathrm{d}(x, y) + \mathrm{d}(y, z), z \in \mathbf{X} \tag{8}$$

Vastaavasti funktio s on samankaltaisuusmitta, jos ehdot 9-11 toteutuvat.

$$\mathrm{s}(x, y) \leq s_0 \tag{9}$$

$$\mathrm{s}(x, x) = s_0 \tag{10}$$

$$\mathrm{s}(x, y) = \mathrm{s}(y, x) \tag{11}$$

Kaavoissa käytetty s_0 on reaaliluku. Ero etäisyysfunktioihin tulee ominaisuuksista 9 ja 10 eli funktio saa suurimman arvon kun vektorit ovat identtisiä. Samankaltaisuusmitta on metrinen, jos ehdot 12 ja 13 toteutuvat.

$$\mathrm{s}(x, y) = s_0 \Rightarrow x = y \tag{12}$$

$$[\mathrm{s}(x, y) + \mathrm{s}(y, z)]\mathrm{s}(x, z) \geq \mathrm{s}(x, y)\mathrm{s}(y, z) \tag{13}$$

Ehto 13 voidaan tulkita siten, että jos tiedetään pisteen y samankaltaisuus pisteisiin x ja z, voidaan päätellä rajoitteita pisteiden x ja z keskenäiselle samankaltaisuudelle.

Numeerisen datan tapauksessa etäisyyksien ja samankaltaisuuksien mittaaminen perustuu usein geometrisiin ominaisuuksiin [26]. Parhaiten tunnettu on Euclidinen-etäisyys $\mathrm{d_E}(\mathbf{x}, \mathbf{y})$ [23].

$$\mathrm{d_E}(\mathbf{x}, \mathbf{y}) = \left(\sum_{i=1}^{D} w_i \|x_i - y_i\|^2 \right)^{\frac{1}{2}} \tag{14}$$

Kategorisen datan tapauksessa mittaamisesta tulee vaikeampaa, koska geometrinen tulkinta ei ole niin yksiselitteistä. Yksi yleisesti käytetty kategorisen datan etäisyys mitta on Hamming-etäisyys $\mathrm{d_H}(\mathbf{x_i}, \mathbf{x_j})$ [21].

$$\mathrm{d_H}(\mathbf{x_i}, \mathbf{x_j}) = \sum_{d=1}^{D} \delta(x_{id}, x_{jd}) \tag{15}$$

$$\delta(x_{id}, x_{jd}) = \begin{cases} 1, & \text{jos } x_{id} \neq x_{jd} \\ 0, & \text{jos } x_{id} = x_{jd} \end{cases} \tag{16}$$

Vektoreiden välisen etäisyyden lisäksi voidaan mitata kahden ryhmän välistä etäisyyttä [9] tai vektorin ja ryhmän välistä etäisyyttä. Kategoristen muuttujille on olemassa paljon erilaisia tapoja mitata etäisyyttä ja samankaltaisuutta, joita on kuvattu tarkemmin eri menetelmien yhteydessä. Vektorien välistä mittaa käytetään menetelmissä k-modes ja k-medoids. Vektorin ja ryhmän välistä mittaa käytetään menetelmissä k-histograms, k-distributions ja k-entropies. Kahden ryhmän välistä mittaa käytetään kokoavissa menetelmissä ACE ja ROCK.

Intuitiivisesti samankaltaisuus- ja etäisyysmitat ovat päinvastaisia [35]. Samankaltaisuusmitan muuttaminen etäisyysmitaksi ja etäisyysmitan muuttaminen samankaltaisuusmitaksi on melko triviaali tehtävä. Toisin sanoen ei ole väliä määrittelemmekö etäisyyden vai samankaltaisuuden.

3 Ryhmittelytuloksen arvioiminen

Ryhmittely on ohjaamatonta oppimista ja aina ei edes tiedetä mikä on oikea tulos. *Arviointikriteeri* määrittää ryhmittelyn laadun. Arviointikriteerin avulla erilaisia ryhmittelyjä voidaan asettaa paremmuusjärjestykseen. Eräs tapa ryhmittelytuloksen arvioimiseksi voi olla visualisointi [20]. Visualisointi ei kuitenkaan ole erityisen tarkka menetelmä ja suurten aineistojen visualisointi on mahdotonta. Numeerisen datan tapauksessa arviointikriteeri perustuu useimmiten datan geometrisiin ominaisuuksiin [35]. Kategoriselle datalle geometriset lähestymistavat ovat kuitenkin sopimattomia [26] eikä ryhmittelyn laadun arvioimiseen ei ole yhtä yleisesti hyväksyttyä kriteeriä. On olemassa useita erilaisia arviointikriteerejä kuten *category utility*, *information loss* ja *luokitteluvirhe* [2].

Ryhmittelyongelman optimaalisen ratkaisun etsiminen on lähes mahdoton tehtävä suuremmilla syötteillä. Esimerkiksi on olemassa 10^{68} eri tapaa ryhmitellä sadan alkion aineisto viiteen ryhmään [34]. Yleensä joudutaan tyytymään epäoptimaaliseen tulokseen, jolloin on tärkeä tietää kuinka hyvä tulos on.

Kriteeri ei pelkästään kerro minkälaisia tuloksia eri ryhmittelymenetelmillä saadaan aikaan vaan se määrittää myös minkälaisia tuloksia tavoitellaan ja mikä on optimaalinen ratkaisu. Kriteeri siis määrittää ongelman ja vaikuttaa siihen kuinka vaikeaa ratkaisun löytäminen on. Tätä kautta arviointiin käytettävä kriteeri määrittää myös ne sisäiset kriteerit, joita käytetään ryhmittelymenetelmässä [5].

Ryhmittelytuloksen arvioimiseen käytettävät kriteerit voidaan jakaa kolmeen kategoriaan [22; 35; 36]:

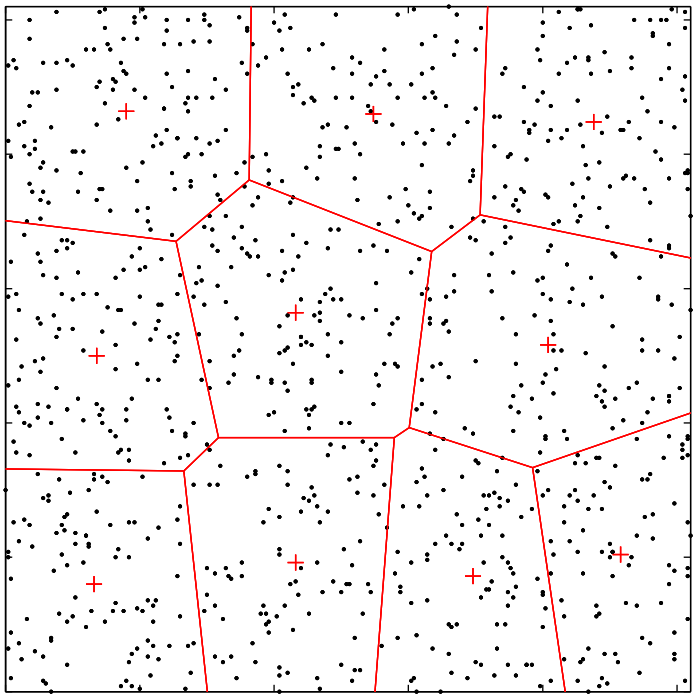
- Ulkoiset kriteerit
- Sisäiset kriteerit
- Suhteelliset kriteerit

Ulkoiset kriteerit testaavat kuinka hyvin ryhmittelytulos sopii ennakkotietoihin. Esimerkiksi luokitteluvirhe mittaa kuinka hyvin ryhmät vastaavat ennalta määrättyä luokittelua. Sisäiset kriteerit testaavat kuinka hyvin ryhmittely vastaa datan sisäistä rakennetta. Sisäiset kriteerit eivät ole riippuvaisia ennakkotiedoista vaan ne tarkastelevat ryhmittelytulosta suoraan alkuperäiseen

dataan perustuen. Suhteelliset kriteerit vertailevat kahden rakenteen paremmuutta. Suhteelliset kriteerit auttavat vertailemaan eri ryhmittelytuloksia ja päättämään mikä ryhmittelytulos kuvastaa parhaiten datan rakennetta.

Ryhmittelytuloksen arvioimiseen liittyy tehtävä oikean ryhmien lukumäärän etsimiseksi [20]. Useat menetelmät eivät käsittele tätä ongelmaa, vaikka useimmiten käytännön sovelluksissa ryhmien lukumäärää ei tiedetä ennakkoon. Myöskään tässä tutkielmassa ei syvennytä ryhmien lukumäärän valintaan. Lukumäärä oletetaan tunnetuksi tai verrataan ryhmien lukumäärän vaihtelun vaikutuksia. Lukumäärän valintaa käsittelee muun muassa [2; 7; 16; 39].

Ryhmittelymenetelmät löytävät aina jonkun ryhmittelyn, vaikka todellisuudessa datassa ei olisi minkäänlaista loogista ryhmäjakoa [22; 36]. Tuloksia analysoitaessa täytyy varmistaa, että data sisältää ryhmäjaon eli toisin sanoen tutkia datan *ryhmiteltävyyttä* [35]. Kuvassa 4 on esimerkki lohkotusta aineistosta, joka on todellisuudessa satunnaista dataa.



Kuva 4: Lohkottu aineisto

Toisaalta myös eri menetelmät ja eri parametrit tuottavat erilaisia ryhmittelyjä. Tämän vuoksi tehokkaat arviointistandardit ja kriteerit ovat tärkeitä, että voidaan olla varmoja ryhmittelyn laadusta. Arvioinnin pitäisi olla objektiivista eikä viitata mihinkään algoritmiin. Arvioinnin pitäisi olla myös hyödyllinen ryhmien lukumäärän valinnassa ja ryhmiteltävyyden arvioimisessa.

3.1 Ryhmittelytuloksien arviontimenetelmiä

3.1.1 Luokitteluvirhe

Luokitteluvirhe perustuu ennakkoon tunnetun luokkajaon tuntemiseen ja ryhmittelytuloksen vertaamiseen tähän luokkajakoon. Luokitteluvirhe voidaan määritellä usealla eri tavalla. Seuraavassa on esitelty usein käytetty tapa, joka on esitetty lähteessä [20]. Ensin määritellään luokittelutarkkuus r.

$$r = \frac{1}{N} \sum_{i=1}^{K} a_i \tag{17}$$

Muuttuja N on aineiston vektoreiden lukumäärä, K on ryhmien lukumäärä ja a_i on oikeaan luokkaan ryhmiteltyjen vektoreiden lukumäärä. Jokainen ryhmä kuvautuu tasan yhteen luokkaan ja jokainen luokka tasan yhtä ryhmää siten, että r on mahdollisimman pieni [2]. Lopullinen luokitteluvirhe e saadaan käyttämällä luokittelutarkkuutta.

$$e = 1 - r \tag{18}$$

Vaikka on mielenkiintoista nähdä kuinka hyvin ryhmittelyä voitaisiin soveltaa reaalimaailmaan, luokitteluun perustuva arviointi sisältää useita ongelmia. Ensiksi optimaalinen luokittelu ei välttämättä takaa optimaalista ryhmittelyä [2]. Näin ollen on mahdollista saada huonompi arviointi, vaikka intuitiivisesti ajateltuna ryhmittelytulos paranisi. Vastaavasti on mahdollista saada parempi arviointi, vaikka ryhmittely olisi epäloogisempi. Todellisuudessa luokitteluvirhe kuitenkin korreloi muiden arviointimenetelmien tuloksien kanssa [2; 3]. Toiseksi käytännön sovellutuksissa luokitteluvirheen käyttö on mahdotonta, koska oikeaa luokittelua ei yleensä ole tarjolla [20]. Koska oikeaa luokittelua ei tiedetä, ei voida arvioida ryhmittelyn laatua. Tästä johtuen ei voida myöskään arvioida onko saatu ryhmittely mielekäs vai onko data ainoastaan lohkottu keinotekoisesti eri osiin.

Valitettavan usein menetelmien arvioimiseen näkee kuitenkin käytettävän ainoastaan ulkoiseen luokitteluun perustuvaa arviointia [4; 8; 16; 29; 19; 18; 20; 24; 25; 31; 32; 37; 38]. Monet [4; 19; 18; 20; 24; 25; 31] viittaavat erityisesti lähteen [20] esittämään luokitteluvirheeseen. Ulkoisesta luokittelusta on kyse myös silloin, kun käytetään käsitteitä puhtaus ja sekaannusmatriisi.

3.1.2 Category utility

Category Utility [15] on määritelty erotuksena, joka syntyy ryhmittelytulokseen perustuvan oikein arvattujen attribuuttien arvojen odotusarvon ja ilman lisätietoja oikein arvattujen attribuuttien odotusarvon välillä [2]. Category utilityn arvo nolla ilmaisee, että ryhmittely ei paljasta aineistosta olennaisia piirteitä [10]. Mitä suuremman arvon ryhmittely saa, sitä parempi ryhmittely on. Category utility on ekvivalentti keskinäisinformaation kanssa [15]. Category utilityn formaali määritelmä noudattaen lähdettä [2]:

$$CU = \sum_{k=1}^{K} \frac{N_k}{N} \sum_{i=1}^{D} \sum_{j=1}^{M_i} (p(A_i = V_j|\mathbf{C}_k)^2 - p(A_i = V_j)^2) \tag{19}$$

Laskennassa käydään läpi kaikki ryhmät K, kaikki attribuutit D ja kaikki attribuuttien arvot $M_{1..D}$. Jos ryhmän sisäinen jakauma vastaa koko aineiston jakaumaa, tällöin $p(A_i = V_j|\mathbf{C}_k) = p(A_i = V_j)$ ja $p(A_i = V_j|\mathbf{C}_k)^2 - p(A_i = V_j)^2 = 0$ [10].

Category utilitya on käytetty muutamissa kategorista ryhmittelyä käsittelevissä lähteissä [2; 3; 5].

3.1.3 Entropia

Entropia on informaatioteoriassa käytetty mitta, joka kuvaa epäjärjestyksen määrää [5]. Seuraavaksi kuvataan entropian laskenta noudattaen lähteitä [7] ja [26]. Ensin määritellään klassinen entropia koko aineistolle.

$$H(\mathbf{X}) = -\sum_{i=1}^{D} \sum_{j=1}^{M_i} p(A_i = V_j) \log p(A_i = V_j) \tag{20}$$

Entropia voidaan laskea myös mille tahansa osa-joukolle. Ryhmittelyn tuloksena saadaan ositus $\mathbf{C} = \mathbf{C}_1, ..., \mathbf{C}_K$. Tällöin tavoitteena on maksimoida funktiota $H_C(\mathbf{C})$.

$$H_C(\mathbf{C}) = \frac{1}{D}\left(H(\mathbf{X}) - \frac{1}{N}\sum_{k=1}^{K} N_k H(\mathbf{C}_k)\right) \tag{21}$$

Koska $\hat{H}(\mathbf{X})$ on tietylle aineistolle aina vakio, päädytään minimoimaan lauseketta

$\frac{1}{N}\sum_{k=1}^{K} N_k H(\mathbf{C}_k)$, josta käytetään myös nimitystä ”entropian odotusarvo” (engl. expected entropy) [7]. Jatkossa käytetään merkintää $H_E(\mathbf{C})$. Kuva 5 esittää entropian laskemisen yksittäiselle ryhmälle C pseudokoodina.

```
ClusterEntropy(C) {
  error := 0.0;

  FOR i := 1 TO Attributes(X) DO {
    FOR j := 1 TO Categories(a) DO {
      IF CategoryInstances(C, i, j) > 0 {
        probability := CategoryInstances(C, i, j) / Size(C);
        error       := error - probability * log(probability);
      }
    }
  }

  RETURN error * Size(C);
}
```

Kuva 5: Ryhmän entropian laskenta

On esitetty, että entropian ja usein kategorisen datan ryhmittelyssä käytettyjen etäisyyksien välillä on yhteys [26].

3.2 Arviointimenetelmän valinta

Koska arviointiin käytettävä kriteeri määrittää ryhmittelymenetelmän sisäiset kriteerit [5], ei ole mielekästä käyttää useita erilaisia arviointimenetelmiä. Lisäksi yksikäsitteistä ja yleisesti hyväksyttyä arviointimenetelmää ei ole, joudutaan tekemään valinta eri menetelmien väliltä. Toisin sanoen on valittava ensisijainen arviointimenetelmä, jota optimoidaan ryhmittelymenetelmiä kehittäessä.

Sisäisen kriteerin valinta ei ole triviaali tehtävä [20]. Tässä tutkielmassa arviointimenetelmäksi on valittu entropia. Entropia on hyvin määritelty ja tunnettu. Lisäksi tulkinta on mielekäs. Entropia kuvastaa epäjärjestyksen määrää ja intuitiivisesti ajateltuna epäjärjestyksen määrän pienentyessä ryhmien sisällä vektorit ryhmän sisällä tulevat yhdenmukaisemmiksi. Entropiaa on käytetty useassa kategorisen datan ryhmittelyä käsittelevissä artikkeleissa menetelmien

vertailussa ja ryhmittelymenetelmien kehittämisen pohjana [5; 7; 26]. Entropiaa on käytetty myös ryhmien lukumäärän määrittämiseen [7]. Eräs mielenkiintoinen entropian sovellusalue voisi olla aineiston ryhmiteltävyyden tutkiminen. Alkuperäisen aineiston ja ryhmitellyn aineiston entropian vertaaminen saattaisi kertoa onko ryhmittelyn avulla löydetty merkityksellisiä rakenteita vai onko aineisto ainoastaan lohkottu osiin.

Aiemmin mainituista ongelmista huolimatta tutkielmassa vertaillaan myös luokitteluvirhettä, koska menetelmä on niin yleisesti käytetty. Lisäksi luokitteluvirhe antaa paremman kosketuksen käytännön ongelmiin kuin entropia. On kuitenkin syytä huomata, että entropia on ensisijainen kriteeri ja entropia määrää menetelmien paremmuusjärjestyksen. Menetelmiä paranneltaessa entropiaa pyritään minimoimaan luokitteluvirheestä välittämättä.

4 K-means algoritmi ja sen soveltaminen kategoriselle datalle

K-means menetelmä [27] on suosituin ja parhaiten tunnettu ryhmittelymenetelmä. Algoritmin yksinkertaisuus ja tehokkuus on inspiroinut tutkijoita kehittämään siitä useita muunnoksia. K-means menetelmän suurimpia etuja on sen laskennallinen tehokkuus [35]. Tehokkuus on olennainen ominaisuus useissa sovelluksissa. Tehokkuuden merkitys korostuu erityisesti tiedonlouhinnassa, jossa käsiteltävää dataa voi olla gigatavuja tai jopa teratavuja [20]. Tällöin neliöllinen aikavaativuus ei ole riittävän nopea.

4.1 Iteratiivisten ryhmittelyalgoritmien yleinen toimintaperiaate

Kuvassa 6 on kuvailtu k-means menetelmän perusvaiheet.

```
Valitaan k satunnaista vektoria ryhmien edustajiksi.

REPEAT
    Lasketaan uusi ositus käyttäen ryhmien edustajia.
    Lasketaan uudet ryhmien edustajat käyttäen ositusta.
UNTIL Kierroksella ei korjata mitään.
```

Kuva 6: K-means algoritmi

Menetelmässä valitaan aluksi K satunnaista vektoria ryhmien edustajiksi. *Ryhmän edustaja* on vektori, joka sisältää eri attribuuttien keskiarvot. Tämän jälkeen vuorotellen korjataan ositusta perustuen ryhmän edustajaan ja ryhmän edustajaa perustuen ositukseen niin kauan kunnes muutoksia ei enää tule. Oikea ositus saadaan asettamalla vektorit ryhmään, jonka ryhmän edustajan on lähimpänä vektoria. On syytä huomata, että lähin tarkoittaa etäisyysmitan tapauksessa pienintä etäisyyttä ja samankaltaisuusmitan tapauksessa suurinta samankaltaisuutta.

K-means menetelmällä on heikkoutensa. Ensimmäiseksi menetelmä löytää paikallisesti optimaalisen ryhmittelyn, mutta ei takaa parasta ryhmittelyä kaikista mahdollisista ryhmittelyistä. Toiseksi ryhmien lukumäärä on annettava parametrina ja kolmanneksi menetelmä on herkkä aineistoissa oleville poikkeamille [35]. *Poikkeamat* ovat vektoreita, jotka

ovat selkeästi eristäytyneitä muusta aineistosta [16].

4.2 Ryhmittely optimointiongelmana

K-means voidaan ajatella epälineaarisena optimointiongelmana [19], jossa pyritään minimoimaan funktiota $P(W, Q)$ rajoitteilla 23 ja 24.

$$P(W, Q) = \sum_{l=1}^{K} \sum_{i=1}^{N} w_{i,j} d(\mathbf{x}_i, Q_l) \tag{22}$$

$$\sum_{l=1}^{K} w_{i,j} = 1, 1 \leq i \leq N \tag{23}$$

$$w_{i,j} \in \{0, 1\}, 1 \leq i \leq n, 1 \leq l \leq K \tag{24}$$

W on $N \times K$-partitiointimatriisi. Jos alkio i kuuluu ryhmään j, $w_{i,j} = 1$. Muutoin $w_{i,j} = 0$. Q on ryhmän edustajisto eli joukko keskiarvovektoreita. Funktio d on etäisyysfunktio. K-meansin muunnoksissa on kyse ryhmän edustajiston Q ja etäisyysfunktion d muuttamisesta.

K-means algoritmi toimii siten, että pidetään vuorotellen arvot W ja Q kiinnitettyinä ja muokataan kiinnittämätön arvo optimaaliseksi. Toisin sanoen ongelma P voidaan ratkaista ratkaisemalla iteratiivisesti kaksi seuraavaa ongelmaa:

1. Ongelma P1: Kiinnitetään $Q = \hat{Q}$ ja ratkaistaan redusoitu ongelma $P(W, \hat{Q})$.
2. Ongelma P2: Kiinnitetään $W = \hat{W}$ ja ratkaistaan redusoitu ongelma $P(\hat{W}, Q)$.

4.3 Iteratiivinen ryhmittely kategoriselle datalle

K-means ei sovellu suoraan kategorisen datan käsittelyyn [20]. Yksinkertaisimmillaan k-means algoritmia voidaan soveltaa kategoriselle datalle siten, että muutetaan kaikki attribuutit binäärisiksi [30]. Tällöin jokainen eri kategoria vastaa omaa binääristä ulottuvuutta ja dataa voidaan käsitellä ikäänkuin numeerisena. Koska ulottuvuuksien lukumäärä on haaste kategorisen datan ryhmittelyssä [40], attribuuttien muuttaminen binääriattribuuteiksi on kyseenalaista. Muunnos kasvattaa ulottuvuuksien lukumäärää entisestään. Lisäksi muunnos ei

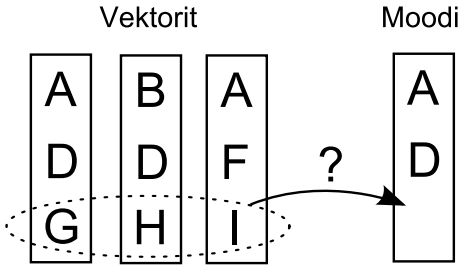
välttämättä kuvaa hyvin attribuuttien ominaisuuksia.

K-means menetelmän periaatetta voidaan kuitenkin hyödyntää tehokkaasti kategorisen datan ryhmittelyyn. Edellytyksenä on etäisyyden ja ryhmän edustaja uudelleen määrittäminen. Useita k-means menetelmän periaatetta noudattavia algoritmeja on kehitetty kategorista dataa varten. Näitä ovat muun muassa k-distributions [6], k-histograms [19], k-medoids [35], k-modes [20], k-populations [24] ja k-representives [31]. Seuraavissa luvuissa tutustaan osaan näistä menetelmistä.

4.4 K-modes

K-modes menetelmässä käytetään ryhmän edustajana keskiarvon (mean) sijaan moodiarvoa (mode) ja geometrisen etäisyyden sijaan käytetään yksinkertaista epäyhteensopivuutta. *Moodiarvo* on se arvo, joka saadaan laskemalle eri arvojen esiintymien lukumäärä ja valitsemalla useimmiten esiintyvä arvo. *Yksinkertainen epäyhteensopivuus* puolestaan saadaan vertaamalla ryhmän edustajaa eli moodia arvoa vektoriin. Eriävien arvojen lukumäärä kertoo suoraan etäisyyden. Yksinkertaisen epäyhteeensopivuuden määritelmä vastaa lähteen [21] Hamming-etäisyyden määritelmää.

Kuvassa 7 on visualisointi ryhmän moodivektorin laskemisesta. Vektoreilla on kolme attribuuttia.



Kuva 7: Moodi-arvo

Moodiarvo ei ole aina yksikäsitteinen [20]. Esimerkiksi kuvasta 7 nähdään, että viimeisen attribuutin moodiarvo voisi olla yhtä hyvin G, H tai I. Kuvassa 8 on moodiarvon laskeminen pseudokoodina. Funktio *ModeValue* palauttaa ryhmälle C attribuutin i moodiarvon.

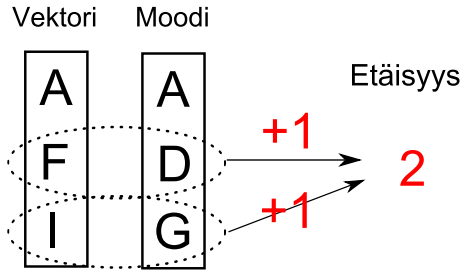
```
ModeValue(C, i) {
  max_category := Category(C, i);
  max_instances := CategoryInstances(C, i, max_category);

  FOR j = 1 TO Categories(X, i) {
    IF (CategoryInstances(C, i, j) > max_instances) {
      max_category := j;
      max_instances := CategoryInstances(C, i, j);
    }
  }

  RETURN max_category;
}
```

Kuva 8: Moodiarvon laskeminen

On syytä huomata, että metodin alussa oletus moodiksi asetetaan nykyinen moodiarvo. Tällä moodiarvo pysyy samana tasapelitilanteissa. Kuvassa 9 on puolestaan laskettu yksinkertainen epäyhteensopivuus saadun moodivektorin ja ensimmäisen vektorin välillä.



Kuva 9: Yksinkertainen epäyhteensopivuus

Kuvassa 9 ensimmäisen attribuutin sekä moodivektorissa että datavektorissa A eli etäisyys on nolla. Sen sijaan toisen ja kolmannen attribuutin arvot ovat eriävät, joten etäisyydeksi saadaan yhteensä 2. Vastaava pseudokoodi yksinkertaisen epäyhteensopivuuden laskemiseen on esitetty kuvassa 10. Kannattaa huomioida, että funktion *SimpleMatchingDistance* molemmat parametrit ovat vektoreita.

Ajatus k-modes algoritmin kehittämiseen on lähtenyt alunperin havainnosta, että k-means algoritmi toimii tehokkaasti, muttei osaa käsitellä kategorisia attribuutteja. K-modes menetelmää aikaisemmat ryhmittelymenetelmät ovat keskittyneet joko kategorisen datan ryhmittelyyn tai suurten datamäärien ryhmittelyyn, mutta harva menetelmä kykenee yhdistämään nämä ominaisuudet [20]. Esimerkiksi hierarkiset menetelmät kykenevät

```
SimpleMatchingDistance(X, Y) {
  distance := 0;

  FOR i = 1 TO Attributes(X) {
    IF (Category(X, i) != Category(Y, i)) {
      distance := distance + 1;
    }
  }

  RETURN distance;
}
```

Kuva 10: Yksinkertainen epäyhteensopivuus

käsittelemään myös kategorista dataa, mutta neliöllinen aikavaativuus tekee niistä hyödyttömiä suurten tietokantojen ryhmittelyssä. K-modes menetelmä soveltuu suurten kategoristen datamäärien ryhmittelyyn [20]. Lisäksi k-modes menetelmä on laajennettavissa *k-prototypes* menetelmäksi [20]. K-prototypes menetelmällä voidaan ryhmitellä aineistoja, jossa on sekä numeerisia että kategorisia attribuutteja. K-modes menetelmän jälkeen on kehitetty paljon uusia menetelmiä, mutta silti menetelmään viitataan edelleen paljon ja sitä voidaan pitää kategorisen datan ryhmittelyn uranuurtajana.

Liukulukuaritmetiikassa on hyvin epätodennäköistä törmätä tasapeleihin. Kategorisen datan tapauksessa tasapelit eivät olekaan epätavallisia tapahtumia [13]. Tasapelien käsittely voi vaikuttaa merkittävästi lopputuloksen. K-modes algoritmissa on kaksi paikkaa, jossa tasapeli voi ilmetä. Näitä nimitetään tyypin 1 ja tyypin 2 tasapeleiksi.

- Tyypin 1 tasapeli ilmenee moodin laskemisessa. Ryhmässä voi attribuuteilla olla sama määrä eri arvoja. Tällöin attribuutin moodiarvoksi on kaksi tai useampia vaihtoehtoja, jotka ovat kaikki yhtä optimaalisia. Pahin mahdollinen tapaus on se, että attribuutin arvot ovat täysin tasaisesti jakautuneita, jolloin vaihtoehtoja on useita eikä ole mitään tapaa sanoa mikä niistä olisi paras.
- Tyypin 2 tasapeli ilmenee vektoreiden etäisyyden laskemisessa eri ryhmiin. Tällöin vektori on yhtä lähellä kahta tai useampaa eri ryhmää.

Tasapelitilanteiden lisäksi ryhmän tyhjeneminen eli ryhmän kuoleminen on myös huomattavasti

todennäköisempää k-modes algoritmissa verrattuna k-means algoritmiin. Nämä ongelmat on huomioitu toteutuksessa, josta tarkemmin luvussa 7.3.

Alkuperäisessä k-modes menetelmässä [20] iterointi on tehty hieman eri tavalla kuin k-means menetelmässä. Sen sijaan että moodeja ja osituksia päivitettäisiin vuorotellen, moodeja päivitetään heti vektorin siirron jälkeen. Muutos on tehty tehokkuuden vuoksi [20]. Puhtaasti k-means menetelmään pohjautuvaa k-modes algoritmia on käytetty muun muassa lähteessä [6] ja tätä versiota käytetään myös tässä tutkielmassa.

Jokaisella iteraatiolla suoritetaan osituksen ja ryhmän edustajan päivittäminen. Osituksen päivittämisessä lasketaan jokaisen vektorin etäisyys jokaiseen ryhmään. Tämä tarkoittaa etäisyyden laskemista NK kertaa. Etäisyyden laskeminen vaatii vektorin jokaisen attribuutin käsittelyä eli D operaatiota. Yhteensä uudelleen sijoittaminen vie yhdellä kierroksella NKD operaatiota. Ryhmän edustajan päivittämisessä täytyy käydä läpi kaikkien attribuuttien kaikki kategoriat. Kaikkien ryhmien edustajien päivittäminen vaatii KDM_{avg} operaatiota jokaisella iteraatiolla. Aineisto iteroidaan läpi I kertaa, joten kaikkiaan tarvitaan $I(NKD + KDM_{avg})$ operaatiota. Koska attribuuttien arvoja ei voi olla enempää kuin aineiston vektoreita eli $M_{avg} \leq N$, kokonaisaikavaativuudeksi saadaan $O(INKD)$.

4.5 K-medoids

K-medoids menetelmä on muunnelma k-means menetelmästä, joka käyttää keskiarvon sijasta ryhmän edustajana medoidia. *Medoidi* on ryhmän vektori, jonka yhteenlaskettu etäisyys ryhmän muihin vektoreihin on pienin mahdollinen [35]. Medoidin käyttämisellä ryhmän edustajana on kaksi etua verrattuna keskiarvoon. Ensiksi medoidi ei ole riippuvainen siitä onko tieto diskreettiä vai jatkuvaa. Vektoreiden välisen etäisyyden määrittäminen riittää. Toiseksi medoidi ei ole yhtä herkkä virheellisille poikkeamille datassa.

Yleinen k-medoids menetelmä vaatii ryhmän sisällä kaikkien vektoreiden välisen etäisyyden laskemisen, jonka seurauksena aikavaativuus on neliöllinen. Kategorisen datan ja Hamming-etäisyyden tapauksessa laskentaa voidaan kuitenkin nopeuttaa. Medoidiksi valitaan vektori, jonka Hamming-etäisyyksien summa muihin vektoreihin on pienin.

Hamming-etäisyys on kahden vektorin välisten eriävien attribuuttien lukumäärä. *Yksinkertainen yhteensopivuuskerroin* on kahden vektorin samojen attribuuttien lukumäärä. Kahden vektorin Hamming-etäisyyden ja yksinkertaisen yhteensopivuuskertoimen summa on aina attribuuttien lukumäärä. Toisin sanoen medoidiksi valitun vektorin yhteenlasketut yhteensopivuuskertoimet muihin vektoreihin on suurin mahdollinen.

Yksinkertaisten yhteensopivuuskerrointen summa voidaan laskea nopeasti ryhmän attribuuttien esiintymien lukumääristä. Esimerkiksi jos tietty kategoria esiintyy ryhmässä viisi kertaa, kategorian omaava vektori on yhteensopiva neljän muun vektorin kanssa kyseisessä ryhmässä. Kun yhteensopivuudet lasketaan yhteen kaikista vektorin kategorioista, saadaan lopputulokseksi yksinkertaisten yhteensopivuuskerrointen summa.

Tarkastellaan esimerkkiä. Taulukossa 3 on ryhmän vektorit ja tehtävänä on etsiä medoidi. Vektoreista voidaan triviaalisti laskea taulukossa 4 näkyvät kategorioiden esiintymät. Taulukosta 4 voidaan katsoa vektorin kategorioita vastaavat esiintymät, vähentää niistä yksi ja laskea arvot yhteen. Tulokset on esitetty taulukossa 5.

	Attribuutti 1	Attribuutti 2	Attribuutti 3
Vektori 1	A	D	G
Vektori 2	B	D	H
Vektori 3	A	F	I

Taulukko 3: Ryhmän vektorit

Attribuutti 1	A: 2	B: 1	
Attribuutti 2	D: 2	F: 1	
Attribuutti 3	G: 1	H: 1	I: 1

Taulukko 4: Ryhmän kategorioiden esiintymät

	Attribuutti 1	Attribuutti 2	Attribuutti 3	Summa
Vektori 1	A: 2-1=**1**	D: 2-1=**1**	G: 1-1=**0**	**2**
Vektori 2	B: 1-1=**0**	D: 2-1=**1**	H: 1-1=**0**	**1**
Vektori 3	A: 2-1=**1**	F: 1-1=**0**	I: 1-1=**0**	**1**

Taulukko 5: Sopivuus muihin vektoreihin

Taulukosta 5 nähdään, että vektorin 1 summa on suurin ja se valitaan medoidiksi. Tulos on helppo todeta oikeaksi, sillä vektorin 1 yhteenlaskettu etäisyys muihin vektoreihin on 4, kun

taas vektoreilla 2 ja 3 yhteenlaskettu etäisyys on 5.

Pseudokoodiesitystä varten määritellään ensimmäiseksi kuvan 11 apufuntio *CategoryEncounters*, joka laskee vektorin kategorioiden esiintymien lukumäärän ryhmässä.

```
CategoryEncounters(C, X) {
  hits := 0;

  FOR i TO Attributes(C) {
    category := Category(max_vector, i)
    hits := CategoryInstances(C, i, category);
  }

  RETURN hits;
}
```

Kuva 11: Vektorin kategorioiden esiintymien lukumäärän laskeminen ryhmässä

Tämän jälkeen voidaan määritellä kuvan 12 funktio *FindMedoid*, joka etsii medoidin.

```
FindMedoid(C, i) {
  max_vector := Medoid(C);
  max_encounters := CategoryEncounters(C, max_vector);

  FOR v IN Vectors(C) {
    encounters := CategoryEncounters(C, v);

    IF (encounters > max_encounters) {
      max_encounters := encounters;
      max_vector := vector;
    }
  }

  RETURN max_vector;
}
```

Kuva 12: Medoidin etsintä

Aluksi medoidiksi asetetaan ryhmän nykyinen medoidi *Medoid(C)* ja vektorin nykyinen sopivuus edustajaksi. Tämän jälkeen käydään kaikki ryhmän vektorit läpi ja etsitään parhaiten edustajaksi sopiva vektori. Alustuksella suositaan nykyistä medoidia tasapelitilanteissa.

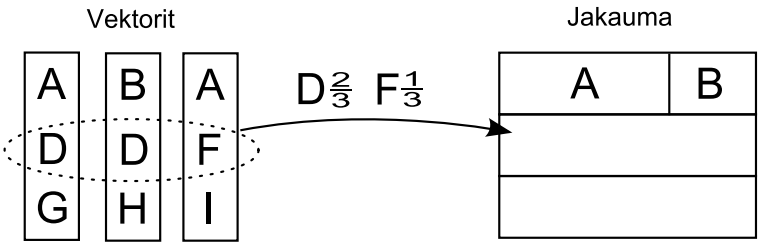
Yleistä periaatetta noudattavassa k-medoids lasketaan $\sum_i^K |\mathbf{C}_i|^2$ vektorien välistä

etäisyyttä jokaisella kierroksella. Tasakokoisilla ryhmillä tämä tarkoittaisin $\frac{N}{K}^2$ etäisyyttä. Hamming-etäisyyden tapauksessa voidaan kuitenkin hyödyntää kategorioiden esiintymien lukumäärää. Osituksen päivittäminen suoritetaan vastaavalla tavalla kuin k-modes. Ryhmän edustajien etsimiseksi täytyy kaikkien vektoreiden arvoja verrata ryhmän jakaumaan. Koska jakauma voidaan päivittää osituksen yhteydessä, tarvitaan iteraatiota kohti $I(NKD + ND)$ operaatiota. Kokonaisaikavaativuus $O(INKD)$ on sama kuin k-modes menetelmällä.

4.6 K-histograms

K-histograms menetelmä [19] tarkentaa ryhmän edustajaa käsittämään jokaisen attribuutin kohdalla prosentuaalista jakaumaa eri arvojen välillä. Samoin etäisyys vektorin ja ryhmän välillä lasketaan ryhmässä esiintyvien attribuuttien prosenttiosuuksien avulla. Vektorin etäisyys ryhmän edustajaan on sitä pienempi, mitä suurempi vektorin kategorioiden prosentuaalinen osuus on ryhmässä. Menetelmässä ryhmän edustaja on yksikäsitteinen toisin kuin k-modes algoritmissa. Sen lisäksi ryhmän edustaja kuvaa tarkemmin ryhmää, koska se huomioi myös vähemmän esiintyvät arvot.

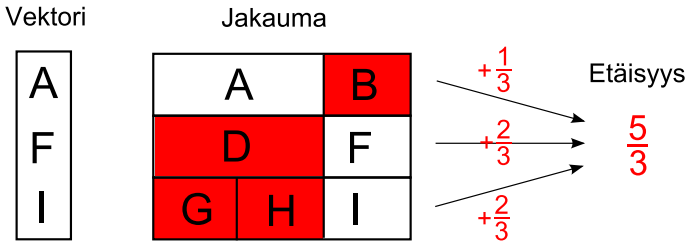
Kuvassa 13 on visualisointi k-histograms menetelmän ryhmän edustajan ja etäisyyden laskemisesta. Yksi rivi edustajassa kuvaa yhden attribuutin arvojakaumaa. Etäisyydessä selkein ero moodiarvoon nähdään attribuutissa 3. Moodiarvo olisi hukannut paljon tietoa, koska vain yksi arvo voi olla moodiarvo. Sen sijaan jakauma huomioi kaikki kolme arvoa.



Kuva 13: Ryhmän edustaja k-histograms menetelmässä

Kuvassa 14 on laskettu etäisyys yhden datavektorin ja ryhmän edustajana toimivan jakauman välillä. Etäisyyttä kasvattavat osat jakaumassa on merkitty punaisella. Esimerkkitapauksesta voidaan siirtyä yleiseen laskentakaavaan, joka on esitetty pseudokoodina kuvassa 15. Alussa etäisyydeksi määritellään attribuuttien lukumäärä, joka tarkoittaa maksimietäisyyttä. Tämän

jälkeen vektorin jokaisen kategorian prosenttiosuus vähennetään etäisyydestä.



Kuva 14: Etäisyys k-histograms menetelmässä

```
KHistogramsDistance(C, X) {
  distance := Attributes(X);

  FOR i = 1 TO Attributes(X) {
    category := Category(X, i);
    proportion := CategoryInstances(C, i, category) / Count(C);
    distance := distance - proportion;
  }

  RETURN distance;
}
```

Kuva 15: K-histograms menetelmän etäisyys pseudokoodina

Kaksi menetelmää – k-histograms [19] ja k-representatives [31] – perustuvat samanlaisiin ryhmän edustajiin ja etäisyyksiin. Menetelmien ainoana erona on iterointi. Ero on vastaava kuin alkuperäisessä k-modes toteutuksessa [20] ja k-modes muunnoksessa [6]. K-representives algoritmissa ryhmiä päivitetään vasta kun kaikki vektorit on uudelleen ositettu. K-histogram algoritmissa sen sijaan päivitetään ryhmiä heti vaihdon jälkeen. Tässä tutkielmassa käytetään k-histograms menetelmää. Alunperin tarkoituksena oli pitää osituksen ja ryhmän edustajan korjaaminen erillisinä kuten k-representatives menetelmässä. Testiajot kuitenkin paljastivat, että k-representatives ei pysähdy aina. Ongelmia esiintyy erityisesti silloin kun ryhmien lukumäärä on suuri aineiston kokoon suhteutettuna. Lisäksi on mainittava, että k-histograms (suomennettuna k-histogrammia) on huomattavasti kuvaavampi nimi kuin k-representatives (suomennettuna k-edustajaa). Uudenlainen iterointitapa on esitetty pseudokoodina kuvassa 16.

Funktio *NextVector* palauttaa seuraavan aineiston vektorin. Aineiston viimeisen vektorin seuraajaksi tulkitaan aineiston ensimmäinen vektori. Silmukassa aineiston vektoreita iteroidaan läpi niin pitkään kunnes kaikki vektorit on käyty peräkkäin läpi ilman muutoksia. Jokaisella

```
IterateKHistograms(X) {
  vector := LastVector(X);
  immutable := 0;

  WHILE (immutable < Size(X)) {
    vector := NextVector(X, vector);
    nearest := FindNearest(vector);

    IF (Cluster(vector) != nearest) {
      ChangePartition(vector, nearest);
      immutable := 0;
    } ELSE {
      immutable := immutable + 1;
    }
  }
}
```

Kuva 16: Iterointi välittömillä siirroilla

otetaan uusi vektori käsittelyyn ja etsitään vektorin lähin ryhmä. Jos ryhmä vaihtuu, jatketaan iterointia vielä vähintään N kertaa.

Koska k-histograms menetelmässä vektorit siirretään välittömästi, iteraatio voidaan ymmärtää yksittäisen vektorin testaamisena. On kuitenkin mielekkäämpää ajatella iteraatiota kaikkien vektoreiden läpikäyntinä kertaalleen, koska silloin iteraatio on vertailukelpoinen muiden menetelmien iteraation kanssa. Tällöin iteraation aikana käsitellään N vektoria. Aluksi lasketaan etäisyys jokaiseen ryhmään, joita on K kappaletta. Yhden etäisyyden laskeminen vaatii D operaatiota, joka tarkoittaa KD operaatiota lähimmän ryhmän etsimiseen yhdelle vektorille. Tuloksen perusteella vektoria saatetaan joutua siirtämään ja pahimmassa tapauksessa siirto suoritetaan joka kerta. Siirtäminen vaatii D operaatiota. Kaikkiaan iteraation aikana suoritetaan $N(KD + D)$ operaatiota. Kaikkien iteraatioiden aikana suoritetaan $IN(KD + D)$ operaatiota. Kokonaisaikavaativuudeksi saadaan $O(INKD)$, joka on yllätten sama kuin k-modes ja k-medoids menetelmillä. Kun edustajan päivittämistä ei tehdä erikseen, ei missään vaiheessa tarvitse erikseen käsitellä kaikkia kategorioita läpi.

4.7 K-distributions

K-distributions menetelmä [6] käyttää ryhmän edustajana jakaumaa kuten k-histograms, mutta noudattaa iteroinnissa alkuperäistä k-means menetelmää. K-distributions menetelmän todellinen erikoispiirre on osituksessa käytettävä vektorin ja ryhmän välinen samankaltaisuusmitta $\mathrm{s}(i, j)$, joka lasketaan eri attribuutin esiintymistodennäköisyyksien yhteistodennäköisyyden avulla.

$$\mathrm{s}(i, j) = \prod_{k=1}^{D} P(i, x_{jk}, k) \tag{25}$$

Vektori i sijoitetaan siihen ryhmään j, joka tuottaa vektorin kanssa suurimman funktion s arvon. Tätä varten tarvitaan funktio P.

$$P(i, v, k) = \frac{F(A_k = V_v|\mathbf{C}_i) + \dfrac{1}{M_k}}{N_i + 1.0} \tag{26}$$

Merkinnöistä $F(A_k = V_v|\mathbf{C}_i)$ tarkoittaa kategorian v lukumäärää ryhmässä, M_k attribuutin k erilaisien kategorioiden lukumäärä ja N_i ryhmän i kokoa. Taulukossa 6 on esimerkki kertoimien määräytymisestä.

Vektori	Jakauma			Kerroin
A	**A: 2**	B: 1	C: 0	2
F	D: 2	E: 0	**F: 1**	1
I	G: 1	H: 1	**I: 1**	1

Taulukko 6: Esimerkki kertoimien määrittämisestä

Taulukon 6 ja kaavan 25 perusteella voidaan laskea vektorin ja ryhmän välinen etäisyys $\mathrm{s}(i, j)$. Oletuksena kaikilla attribuuteilla on kolme mahdollista arvoa.

$$\mathrm{s}(i, j) = \frac{2 + \frac{1}{3}}{3 + 1.0} * \frac{1 + \frac{1}{3}}{3 + 1.0} * \frac{1 + \frac{1}{3}}{3 + 1.0} = \frac{112}{1728} \approx 0.0648 \tag{27}$$

Kuvassa 17 on esitetty sama pseudokoodina. Tämä laskentatapa aiheuttaa ongelmia, jos

```
KDistributionsSimilarity(C, X) {
  product := 1;

  FOR i = 1 TO Attributes(X) {
    category    := Category(X, i);
    instances   := CategoryInstances(C, i, category);
    estimate    := 1 / Categories(i);
    probability := (instances + estimate) / (Count(C) + 1);
    product     := product * probability;
  }

  RETURN product;
}
```

Kuva 17: Samankaltaisuuden laskeminen k-distributions menetelmässä

ulottuvuuksia ja kategorioita on paljon. Todennäköisyydet ovat ykköstä pienempiä ja tulo lähestyy nollaa. Helpoin tapa kiertää ongelma, on suorittaa laskenta logaritmia apuna käyttäen. Tarkastellaan epäyhtälöitä 28-30, jossa vertaillaan kahden tulon suuruusjärjestystä. Luvut x_i ja y_i ovat reaalilukuja.

$$x_1 x_2 ... x_n < y_1 y_2 ... y_n \tag{28}$$

$$\log(x_1 x_2 ... x_D) < \log(y_1 y_2 ... y_D) \tag{29}$$

$$\log x_1 + \log x_2 + ... + \log x_n < \log y_1 + \log y_2 + ... + \log y_n \tag{30}$$

Alkuperäinen epäyhtälö 28 on ekvivalentti epäyhtälön 29 kanssa, koska logaritmifunktio on aidosti kasvava. Epäyhtälö 30 on puolestaan ekvivalentti logaritmifunktion laskusääntöjen perusteella. Saman voi kirjoittaa pseudokoodina kuvan 18 mukaisesti.

Kuten k-means ja k-modes, k-distributions tuottaa alustuksesta riippuvan paikallisen optimiratkaisun. Kirjoittajien [6] testien mukaan k-distributions tuottaa selvästi k-modes algoritmia parempia ryhmittelytuloksia. Lisäksi yleensä ryhmittelyssä eri attribuutit oletetaan epärealistisesti toisistaan täysin riippumattomiksi. Yhteistodennäköisyyden avulla voitaisiin mahdollisesti huomioida myös attribuuttien riippuvuus ja tätä kautta parantaa tulosta entisestään [6].

```
ModifiedKDistributionsSimilarity(C, X) {
  sum := 0;

  FOR i = 1 TO Attributes(X) {
    category    := Category(X, i);
    instances   := CategoryInstances(C, i, category);
    estimate    := 1 / Categories(i);
    probability := (instances + estimate) / (Count(C) + 1);
    sum         := sum + log(probability);
  }

  RETURN sum;
}
```

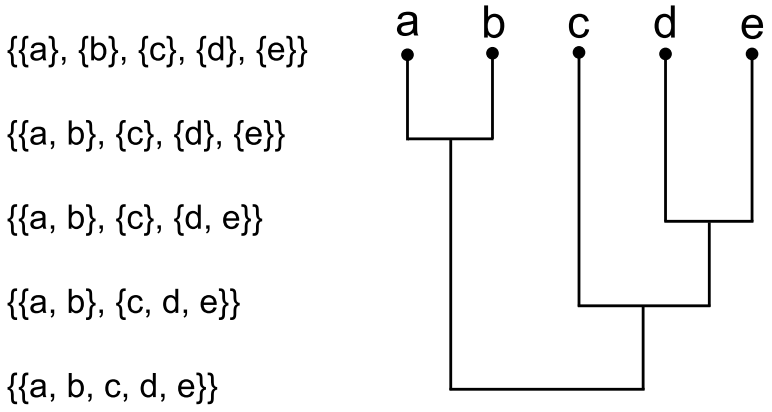
Kuva 18: Samankaltaisuuden laskeminen logaritmien summana

K-distributions menetelmän huonona puolena mainittakoon se, ettei ositusvaiheen laskentamenetelmää ole täysin perusteltu. Kategorian esiintymisten lukumäärä jaettuna ryhmän koolla on intuitiivinen lähtökohta. Termin $\frac{1}{M_k}$ lisääminen estää todennäköisyyden 0, jos kategoriaa ei esiinny ryhmässä lainkaan. Yhden kategorian todennäköisyys 0 puolestaan estäisi vektorin siirtymisen ryhmään, vaikka vektori olisi muilta osin ryhmään sopiva. Avoimeksi kuitenkin jää miksi juuri termi $\frac{1}{M_k}$ on valittu eikä esimerkiksi termiä 1 sekä miksi ryhmän kokoa kasvatetaan juuri yhdellä. Kyse on kirjoittajien mukaan estimoinnista ja joka tapauksessa menetelmä tuottaa hyviä tuloksia testien perusteella [6].

Jakauman tapauksessa osituksen päivittäminen on lähes k-modes menetelmää vastaava, mutta etäisyyden laskemisessa joudutaan lisäksi käyttämään logaritmi-funktiota, jolloin päädytään $NKDT_L$ operaatioon. Erillinen jakauman päivittäminen sen sijaan vaatii jakauman alustamisen nollilla KDM_{avg} sekä jokaisen vektorin attribuuttien läpikäymistä ND. Yhteensä operaatioita tarvitaan $I(NKDT_L + KDM_{avg} + ND)$. Koska attribuutilla voi olla eri kategorioita enintään yhtä paljon kuin vektoreita eli $M_{avg} \leq N$, saadaan aikavaativuudeksi $O(INKDT_L)$.

5 Hierarkiset algoritmit

Hierarkiset ryhmittelymenetelmät etenevät nimensä mukaisesti hierarkisesti ja tuottavat ryhmittelyhierarkian. Kuva 19 havainnollistaa hierarkiaa *dendogrammilla*.



Kuva 19: Dendorammi mukaillen lähdettä [35]

Kuvassa 19 a-e ovat vektoreita. Vasemmalla on viisi erilaista ryhmittelyä, jotka määräytyvät oikealla olevan dendogrammin perusteella. Hierarkiset ryhmittelymenetelmät voidaan jakaa kokoaviin ja osittaviin menetelmiin [9]. *Kokoavissa menetelmissä* lähtötilanteena on N ryhmää, jolloin jokainen datavektori sijaitsee omassa ryhmässä. Jokaisella kierroksella yhdistetään kaksi samankaltaisinta ryhmää yhdeksi ryhmäksi. Tämä vastaa kuvassa 19 siirtymistä ylhäältä alaspäin. *Osittavissa menetelmissä* lähtökohtana on yksi ryhmä, joka sisältää kaikki vektorit. Jokaisella kierroksella jaetaan rakenteeltaan hajanaisin ryhmä kahteen osaan siten, että uusien ryhmien rakenne olisi mahdollisimman yhtenäinen. Kuvassa 19 tämä vastaa siirtymistä alhaalta ylöspäin. Yhdistämistä tai jakamista jatketaan niin kauan kunnes ryhmiä on haluttu määrä.

Koska tässä luvussa tutustutaan kahteen kokoavaan menetelmään, tutustustaan hieman tarkemmin kokoavaan ryhmittelyyn yleisesti. Yleinen kokoava periaate on kuvattu pseudokoodina kuvassa 20. Näiden vaiheiden toteuttamiseksi yhdistämiskustannuksia säilytetään usein erillisessä tietorakenteessa, jonka avulla vältetään laskemasta yhdistämiskustannuksia uudelleen jokaisella kierroksella ja yhdistettävä pari saadaan haettua nopeammin. Yhdistämiskustannukset lasketaan kokonaisuudessaan vain alustusvaiheessa ja kustannuksia päivitetään aina ryhmien yhdistämisen jälkeen.

```
Sijoitetaan jokainen vektori omaan ryhmään.

REPEAT
  Etsitään pari, jonka yhdistämiskustannus on pienin.
  Yhdistetään pari uudeksi ryhmäksi.
  Poistaan vanhat ryhmät.
UNTIL ryhmien lukumäärä on oikea
```

Kuva 20: Yleinen kokoava periaate

5.1 Hierarkisten menetelmien edut ja haitat

Hierarkia on käytännöllinen monissa sovelluksissa [9]. Esimerkiksi taksonomiassa, jossa tutkitaan elävien organismien luokittelua. Tarvittaessa yhden suorituksen aikana saadaan ryhmittely kaikilla mahdollisilla ryhmien lukumäärillä. Tämä voi olla hyödyllistä myös silloin, kun halutaan löytää vain yksi ryhmittely. Hierarkiasta voidaan etsiä datalle parhaiten soveltuva ryhmittely ja päätellä oikea ryhmien lukumäärä [9]. Tekniikkaa on sovellettu muun muassa ACE menetelmässä [7].

Hierarkisten menetelmien tuottamat hierarkiat ovat hyödyllisiä sovelluksissa, joissa hierarkia on luonnostaan olemassa. Kaikki aineistot eivät kuitenkaan ole rakenteeltaan hierarkisia. Hierarkian määrääminen tälläiselle aineistolle on ilmeisen vaarallista [9]. Yksi vaara piilee siinä, että jos hierarkinen menetelmä yhdistää tai erottaa väärät vektorit toisistaan, tilanne ei korjaannu seuraavilla kierroksilla [9; 35]. Ei-hierarkisten aineistojen tapauksessa on harkittava tarkkaan hierarkisten menetelmien käyttöä.

Hierarkiset menetelmät eivät ole erityisen tehokkaita. Yleistä kokoavaa periaatetta noudattavan menetelmän aikavaativuus on useimmiten $O(N^3)$ ja parhaimmillaankin $O(N^2)$ [35]. Lisäksi jos pidetään kirjaa kaikkien ryhmäparien yhdistämiskustannuksesta, tilavaativuudeksi saadaan $O(N^2)$. On selvää etteivät nämä menetelmät ole suoraan sovellettavissa suurille aineistoille. Tehokkuuden parantamiseksi voidaan käyttää esimerkiksi näytteistämistä kuten esimerkiksi ROCK-menetelmässä on tehty [16].

5.2 Entropiaan perustuva kokoava menetelmä

ACE (Agglomerative Categorical clustering with Entropy criterion) [7] on kokoava menetelmä, joka optimoi ryhmien yhdistämisessä entropiaa. Alussa jokainen vektori on omassa ryhmässään. Iteroinnissa jokaisella kierroksella yhdistetään kaksi ryhmää, joiden yhdistäminen aiheuttaa mahdollisimman pienen kokonaisentropian kasvun. Entropia kasvaa sitä enemmän mitä erilaisempia yhdistettävät ryhmät ovat. Jos ryhmien arvojakaumat ovat täysin identtiset, entropia ei kasva lainkaan.

Entropia on luonnollinen samankaltaisuusmitta, jonka päälle voidaan rakentaa kokoava algoritmi [7]. Entropiasta saadaan johdettua ryhmien yhdistämiselle kustannusfunktio I_m.

$$I_m(\mathbf{C}_p, \mathbf{C}_q) = (N_p + N_q)H(\mathbf{C}_p \cup \mathbf{C}_q) - (N_p H(\mathbf{C}_p) + N_q H(\mathbf{C}_q)) \tag{31}$$

Funktio I_m kertoo kuinka paljon kokonaisentropia kasvaa, jos ryhmät $\mathbf{C}_p$ ja $\mathbf{C}_q$ yhdistetään. Muuttujat N_p ja N_q tarkoittavat ryhmiin kuuluvien vektoreiden lukumäärää. Sama kustannusfunktio voidaan esittää pseudokoodina kuvan 21 mukaisesti.

```
ACEMergeCost(P, Q) {
  costP  := Count(P) * ClusterEntropy(P);
  costQ  := Count(Q) * ClusterEntropy(Q);
  before := costP + costQ;
  after  := (Count(P)+Count(Q)) * ClusterEntropy(Union(P, Q));
  RETURN before - after;
}
```

Kuva 21: ACE kustannusfunktio

Menetelmässä käytetään kolmea merkittävää tietorakennetta. *Yhteenvetotaulussa* (summary table) on eri ryhmien arvojakaumat ja tämän avulla saadaan päivitettyä yhdistämiskustannukset nopeasti riippumatta vektoreiden lukumääristä. *I_m-taulussa* (I_m-table) on ryhmien yhdistämiskustannukset. I_m-taulun arvot ovat *I_m-keosta* (I_m heap), josta saadaan etsittyä pienin arvo nopeasti [7].

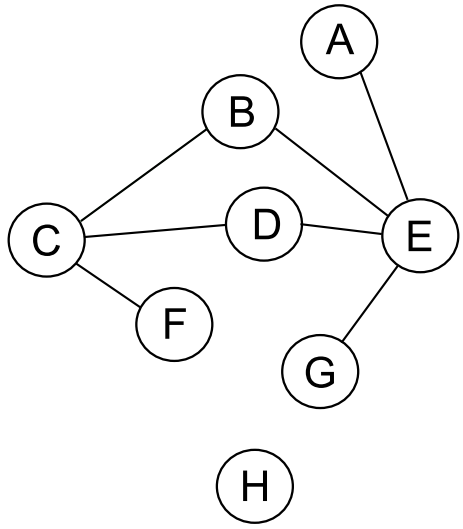
ACE menetelmä noudattaa tarkasti yleistä kokoavaa periaatetta. Kun asetetaan D ulottuuksien

lukumääräksi ja M_{avg} attribuuttien keskimääräiseksi kategorioiden lukumääräksi, saadaan algoritmille aikavaativuus $O(N^2 D M_{avg})$. Käytännön kannalta pahimmaksi rajoitteeksi muodostuu kuitenkin tilavaativuus, koska alussa jokainen vektori muodostaa oman ryhmänsä. Koska I_m-taulussa ja keossa on jokaisen ryhmäparin yhdistämiskustannus, päädytään tilavaativuuteen $O(N^2)$ [7].

Monista muista ryhmittelymenetelmistä poiketen ACE ryhmittelyä voidaan hyödyntää myös ryhmien lukumäärän määrittämiseen. Idea perustuu intuitiiviseen havaintoon, että entropia alkaa kasvamaan voimakkaasti siinä vaiheessa kun ryhmien lukumäärä menee liian pieneksi. Nämä pisteet löytyvät toisella derivaatalla. Koska entropiaa ilmaiseva funktio on diskereetti, derivointia ei voida tehdä suoraan. Derivaatoista saadaan kuitenkin riittävän hyvät arviot. Menetelmä ei kuitenkaan anna suoraa tulosta, vaan joukon hyviä vaihtoehtoja ryhmien lukumääräksi, joiden väliltä voidaan tehdä lopullinen valinta [7].

5.3 Linkkeihin perustuva kokoava menetelmä

ROCK (RObust Clustering using linKs) [29] on kokoava menetelmä, joka nojautuu naapureiden ja linkkien käsitteisiin. Kaksi vektoria ovat *naapureita*, jos ne ovat tarpeeksi samankaltaiset. Samankaltaisuus määritellään samojen attribuuttien lukumäärän perusteella. Kahden vektorin välillä on *linkki*, jos vektoreilla on yhteinen naapuri. Kuvassa 22 on esimerkki linkkien laskemisesta.



Kuva 22: Linkkien laskeminen vektoreiden välillä.

Kuvassa 22 verkon solmut ovat aineiston vektoreita ja kaari kuvaa vektoreiden naapuruutta. Tässä esimerkki tapauksessa vektoreiden C ja E välillä on kaksi linkkiä, koska niillä on kaksi yhteistä naapuria B ja D. Formaalisti määriteltynä vektorit $\mathbf{x}_i$ ja $\mathbf{x}_j$ ovat naapureita, jos epäyhtälö 32 on tosi.

$$sim(\mathbf{x}_i, \mathbf{x}_j) \geq \theta \tag{32}$$

Epäyhtälön theta-arvo (θ) on käyttäjän määrittelemä parametri ja funktio $sim(\mathbf{x}_i, \mathbf{x}_j)$ on määritelty kaavassa 33.

$$sim(T_1, T_2) = \frac{\mid T_1 \cap T_2 \mid}{\mid T_1 \cup T_2 \mid} \tag{33}$$

Joukko T_1 on ensimmäisen vektorin attribuuttien joukko ja vastaavasti T_2 on toisen vektorin attribuuttien joukko. Käytännössä kaava tarkoittaa vektoreiden yhteisten attribuuttien lukumäärän jakamista kaikkien vektoreissa esiintyvien attribuuttien lukumäärällä. Kun linkit on tiedossa, funktio g kertoo ryhmien $\mathbf{C}_i$ ja $\mathbf{C}_j$ samankaltaisuuden.

$$g(\mathbf{C}_i, \mathbf{C}_j) = \frac{link(\mathbf{C}_i, \mathbf{C}_j)}{(N_i + N_j)^{1+2f(\theta)} - N_i^{1+2f(\theta)} - N_j^{1+2f(\theta)}} \tag{34}$$

Muuttujat N_i ja N_j ilmaisevat ryhmien koot. Ehdotus funtiolle $f(\theta)$ [29]:

$$f(\theta) = \frac{1-\theta}{1+\theta} \tag{35}$$

Näiden käsitteiden pohjalta voidaan kuvata ROCK menetelmä kuvan 23 pseudokoodin mukaisesti. Koska ROCK menetelmä sisältää paljon yksityiskohtia ja eri vaiheita, käytetään selkeyden vuoksi tavallista abstraktimpaa tasoa. Yksityiskohtaisemmat esitykset löytyvät lähteistä [29; 16].

```
Sijoitetaan jokainen vektoria omaan ryhmään.
Etsitään ryhmien naapurit.
Lasketaan naapurien pohjalta ryhmien väliset linkit.
Poistetaan ryhmät, joilla ei ole linkkejä (1).

FOREACH ryhmä:
  Lasketaan yhdistämiskustannukset muihin ryhmiin.
  Rakennetaan ryhmäpareista lokaalikeko (2).
END

Rakennetaan lokaaleista keoista globaalikeko (3a).

REPEAT
  Haetaan keoista parhaiten yhteensopiva pari (3b).
  Yhdistetään pari uudeksi ryhmäksi (4).
  Poistetaan vanhat ryhmät.
  Päivitetään kekoja muuttuneiden ryhmien osalta.
UNTIL keot ovat tyhjiä (5).
```

Kuva 23: ROCK pseudokoodina

Kuvan 23 pseudokoodista on syytä tehdä useita tärkeitä havaintoja. Alkuvaiheessa (1) poistetaan linkittömät ryhmät, jotka ovat muusta aineistosta erillisiä. Nämä ovat menetelmän löytämiä poikkeamia, joita ei voi ryhmitellä ollenkaan. Mitä suuremmaksi θ arvo asetetaan, sitä useampi vektori hylätään poikkeamana. Lokaaleihin kekoihin (2) sijoitetaan ainoastaan ne ryhmäparit, joiden välillä on linkkejä. Jos vektoreiden välillä ei ole linkkejä, niiden yhdistämiskustannus on ääretön eikä paria sijoiteta kekoon. Algoritmi käyttää useita kekoja (3a ja 3b) yhdistettävän parin etsimiseen. Monimutkaisuuden ei pidä antaa hämätä, koska kyse on kuitenkin vain parhaiten yhteensopivan parin etsimisestä. Yhdistämisvaiheessa (4) uuden ryhmän linkkien lukumäärä muihin ryhmiin määräytyy yhdistettävien ryhmien linkkien summasta. Linkit määräävät myös uuden ryhmän yhdistämiskustannukset. Algoritmin pysähtymistä (5) ei määrää ulkoinen parametri vaan pysähtyminen määräytyy keon tyhjenemisen mukaan. Keon tyhjeneminen määrittelee myös ryhmien lukumäärän.

Kokonaisuudessaan ROCK menetelmässä pyritään maksimoimaan arvoa E_{ROCK}.

$$E_{ROCK} = \sum_{i=1}^{K} N_i \sum_{x,y \in \mathbf{C}_i} \frac{link(x,y)}{N_i^{1+2*f(\theta)}} \tag{36}$$

Funktion $f(\theta)$ valintaan vaikuttaa aineiston rakenne ja se minkälaisia ryhmiä etsitään [16]. Valitettavasti funktion valinnalle ei ole eksaktia perustelua, jonka vuoksi myöskään funktiosta riippuvaa arvoa E_{ROCK} ei voida käyttää arvioinnissa.

ROCK algoritmia soveltuu erityisesti transaktiodatan tiedonlouhintaan [29]. Tietokanta voi sisältää useita tuhansia erilaisia attribuutteja, vaikka yksittäinen maksutapahtuma sisältää yleensä huomattavasti vähemmän merkitseviä attribuutteja. Ongelman ajatteleminen linkkien kautta toimii tällöin hyvin [29]. Idea on intuitiivisesti erittäin järkevä. Esimerkiksi jos kahdesta maksutapahtumasta molemmista puuttuu jonkin tietty tuote, ei ole mielekästä päätellä tuotteen puuttumisesta mitään. Perinteisillä etäisyyksillä olisi mahdollista saada pieni etäisyys useiden tälläisien attribuuttien vuoksi ja oikeasti merkitsevien attribuuttien merkitys katoaisi.

Merkitsemällä R_{max} naapureiden suurinta mahdollista määrää ja R_{avg} naapureiden keskimääräistä määrää, saadaan ROCK menetelmälle aikavaativuudeksi $O(N^2 + NR_{max}R_{avg} + N^2 \log N)$ ja tilavaativuudeksi $O(min\{N^2, NR_{max}R_{avg}\})$ [16].

ROCK algoritmin aikavaativuus on liian suuri isoimmille aineistoille [16]. Menetelmän kehittäjät tarjoavat ratkaisuksi näytteistämistä. Näytteistämisen perusajatuksena on valita pieni osajoukko alkuperäisestä aineistosta, jonka jälkeen osajoukko ryhmitellään ja loput aineiston vektorit sijoitetaan lähimpiin ryhmiin. Voidaan kuitenkin väittää, ettei suurien ja monimutkaisien aineistojen rakennetta saada kuvattua pienellä osajoukolla [20]. Näitä aineistoja esiintyy erityisesti tiedonlouhinnassa.

ROCK menetelmä sisältää poikkeamien käsittelyn [16]. Poikkeamat hylätään ja jätetään ryhmittelemättä. Poikkeamien käsittely on hyödyllistä monissa käytännön sovelluksissa. Valitettavasti vertailu muihin menetelmiin hankaloituu, koska koko aineistoa ei ole ryhmitelty. Usein nämä poikkeamat ovat vieläpä hankalia tapauksia, koska ne eivät kuulu kunnolla mihinkään ryhmään. Tämä vaikuttaa merkittävästi ryhmittelytuloksen arviointiin. Lisäksi ROCK menetelmässä poikkeamien käsittely on sisäänrakennettu siten, että poikkeamien ryhmittely tuottaisi olennaisesti erilaisen menetelmän.

Poikkeamien käsittelyn lisäksi ROCK menetelmä sisältää ryhmien lukumäärän valinnan. Käyttäjä ei voi määrittää ryhmien lukumäärää ROCK menetelmän ehdottamaa lukumäärää

pienemmäksi. Myös tämä heikentää vertailukelpoisuutta muihin menetelmiin. Vaikka ryhmien lukumäärän valinnan ja poikkeamien käsittelyn automaattisuus voidaan ajatella menetelmän etuna, ovat ne toisaalta rajoitteina selkeä haitta. Usein ryhmien lukumäärän valintaa ja poikkeamien käsittelyä pidetään erillisinä ongelmina, joihin ryhmittelymenetelmä ei ota kantaa [2].

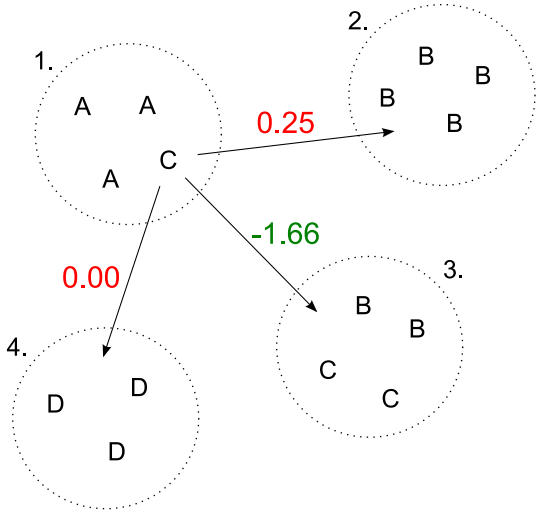
6 K-means ja entropiakriteeri

Tässä luvussa käsitellään kategorisen datan ryhmittelyyn, jonka lähtökohtana on k-means perusidea ja entropian minimoimisen yhdistäminen.

6.1 Perusteet

Kuten luvussa 4 nähtiin, k-means algoritmilla on useita hyviä ominaisuuksia ja k-meansin perusajatusta hyödyntäviä menetelmiä on esitelty useita. Valitettavasti yksikään näistä ei suoranaisesti optimoi entropiakriteeriä. Toisin sanoen menetelmät eivät löydä entropian näkökulmasta lokaalia optimia, vaikka kaikki menetelmät pyrkivät löytämään mahdollisimman yhdenmukaisia ryhmiä. Tästä syystä olisi mielenkiintoista määritellä k-means perusajatusta noudatteleva menetelmä, joka optimoi entropiaa. Käytetään tästä menetelmästä nimitystä k-entropies.

Yksinkertaisin tapa määrittää optimaalinen etäisyys ryhmän ja vektorin välille on laskea entropiassa tapahtuva muutos, jos vektori siirtyisi ryhmään. Jos entropia kasvaa, muutosta ei kannata tehdä. Sen sijaan jos entropia pienenee, on vektorin siirtäminen uuteen ryhmään kannattavaa. Kuvassa 24 on idean visualisointi yksiulotteisella datalla.

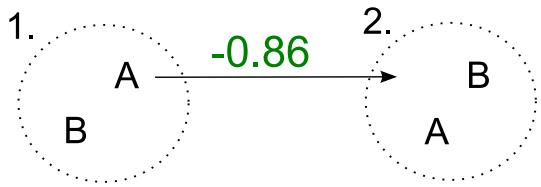


Kuva 24: Siirrosta aiheutuva entropian muutos

Kuvassa on esitetty entropian muutokset vektorin C mahdollisille siirroille. Tässä tapauksessa vektori siirretään ryhmään 3, koska se tuottaa pienimmän mahdollisen entropian. Siirto

vaikuttaa ainoastaan muuttuvien ryhmien entropiaan. Ryhmien 2 ja 4 entropia ei muutu.

Idea ei ole uusi. Samaan ideaan, mutta satunnaisiin vektoreiden siirtoihin perustuva menetelmä on esitetty [26]. Tavoitteena on kuitenkin determinisesti etenevä menetelmä. Valitettavasti k-means iterointi ei toimi tällä etäisyysmetriikalla suoraan. Yksinkertaisena vastaesimerkkinä voidaan ajatella yksiulotteista aineistoa, jossa on neljä vektoria. Kategoriat A ja B esiintyvät molemmat kaksi kertaa. Ryhmittelyssä kahteen ryhmään alustus on suoritettu siten, että molemmissa ryhmissä esiintyy molemmat arvot kerran. Kuvassa 25 on visualisointi vastaesimerkistä.



Kuva 25: Vastaesimerkki k-means pohjaisen menetelmän toimivuudelle

Yksinkertaisella laskulla voidaan todentaa, että ensimmäisen ryhmän vektorin A siirtäminen toiseen ryhmään pienentää entropiaa. Kun tarkastellaan kuvaa tarkemmin, tilanne on kaikille vektoreille sama. Kaikkia vektoreita täytyy siirtää. Jos kaikki vektorit siirretään, päädytään täysin vastaavaan tilanteeseen ja muodostuu ikuinen silmukka. Vaikka esimerkin tilanne voitaisiin vielä välttää duplikaattien vektoreiden käsittelyllä, ongelma voidaan aiheuttaa monimutkaisemmalla esimerkillä myös ilman duplikaatteja. Käytännössä ongelma aiheutuu siitä, että entropian muutosta laskettaessa ei huomioida muiden vektoreiden mahdollista siirtymistä. Ongelma voidaan korjata siirtämällä vektori oikeaan ryhmään heti testauksen jälkeen.

On helppo päätellä, että välittömillä siirroilla algoritmi pysähtyy. Jokainen siirto pienentää entropiaa eli koskaan ei voida päätyä aikaisempaan ryhmittelyyn. Erilaisia ryhmittelyjä on äärellinen määrä, joten algoritmi pysähtyy väistämättä.

Vektorin siirto voidaan laskea yksinkertaisesti käyttäen aliluvussa 3.1.3 kuvassa 5 esitettyä funktiota kuvan 26 mukaisesti. Ensin lasketaan nykyinen kokonaisentropia ryhmässä, josta vektoria ollaan siirtämässä, ja ryhmässä, johon vektoria ollaan siirtämässä. Sen jälkeen suoritetaan siirto ja lasketaan ryhmien entropiat uudelleen. Lopuksi vektori siirretään takaisin.

Entropian muutos saadaan vähentämällä siirtoa edeltävä entropia siirron jälkeisestä entropiasta.

```
EntropyChange(C,C',V) {
  before := ClusterEntropy(C) + ClusterEntropy(C');
  ChangePartition(V, C, C');
  after  := ClusterEntropy(C) + ClusterEntropy(C');
  ChangePartition(V, C', C);

  change := after - before;
  RETURN change;
}
```

Kuva 26: Yksinkertainen tapa laskea entropian muutos vektoria siirrettäessä

Lähin ryhmä valitaan sen perusteella, missä vektori tuottaa pienimmän mahdollisimman kokonaisentropian. Siirrosta aiheutuva entropian muutos täytyy olla negatiivinen, koska vektorin pitäminen nykyisessä ryhmässä voidaan tulkita muutoksena, jonka suuruus on nolla. Pseudokoodi ryhmän valintaan on kuvassa 27. Itse pääohjelma etenee täysin vastaavalla tavalla kuin k-histograms.

```
FindNearest(C, V) {
  nearest   := C;
  minChange := 0;

  FOR C' := 1 TO K DO {
    IF (C' != C) {
      change = EntropyChange(C, C', V);

      IF (change < minChange) {
        nearest   := C';
        minChange := change;
      }
    }
  }

  RETURN nearest;
}
```

Kuva 27: Minimientropian tuottavan ryhmän valinta

K-entropies menetelmän tapauksessa etäisyydenlaskenta on raskasta, mutta osituksen päivittämistä ei suoriteta erikseen. Etäisyyden laskemisen aikavaativuus on ryhmän entropian laskemiseen ja vektorin siirtoihin kuluva aika. Vektorin siirto vaatii $2 * D$ operaatiota. Lisäksi

entropia on laskettava kahdelle ryhmälle, joka vaatii logaritmin laskemisen $4 * DM_{avg}$ kertaa. Kun logaritmin tarvitsemaa aikaa merkitään muuttujalla T_L, saadaan tulokseksi $NK(2 * D + 4 * DM_{avg}T_L)$ operaatiota. Tämä tarkoittaa lopulta aikavaativuutta $O(INKDM_{avg}T_L)$.

Seuraavaksi on esitetty menetelmä, jonka avulla entropian muutos saadaan laskettua nopeammin ylläpitämällä kustannustaulukoita. Tässä tutkielmassa pitäydytään kuitenkin äsken esitetyssä menetelmässä, koska todellisuudessa kustannustaulukoiden päivittäminen hidastaa kokonaissuoritusaikaa.

6.2 Nopeutettu entropian muutoksen laskeminen

K-entropies menetelmässä vektorin siirrosta aiheutuvan entropian muutoksen laskemista voidaan nopeuttaa siirron nopeuden kustannuksella. Jokaisessa ryhmässä jokaiselle kategorialle lasketaan lisäys- ja vähennyskustannus eli entropian muutos, jos kategoriaa lisättäisiin ryhmään tai vähennettäisiin ryhmästä. Kun kaikkien kategorioiden entropian muutos on saatu laskettua, yksittäisen vektorin siirtämisestä aiheutuva entropian muutos saadaan laskemalla yhteen vektorin sisältämien kategorioiden lisäys- ja vähennyskustannukset. Jos vektori siirretään, siirrossa mukana olleiden ryhmien lisäys- ja vähennyskustannukset on päivitettävä. Aluksi määritellään laskentaa yksinkertaistava apufunktio $E(i, c)$.

$$E(i,c) = \begin{cases} \frac{i}{c} \log \frac{i}{c}, & \text{jos } i > 0 \text{ ja } c > 0 \\ 0, & \text{jos } i = 0 \text{ tai } c = 0 \end{cases} \tag{37}$$

Yksittäiselle attribuutille voidaan laskea kokonaisentropia 38. Lisäksi voidaan laskea yleinen lisäyskustannus 39 ja vähennyskustannus 40.

$$S(k,a) = \sum_{i=0}^{M_a} E(F(A_a = V_i|\mathbf{C}_k), N_k) \tag{38}$$

$$I_s(k,a) = \sum_{i=0}^{M_a} E(F(A_a = V_i|\mathbf{C}_k), N_k + 1) \tag{39}$$

$$D_s(k,a) = \sum_{i=0}^{M_a} E(F(A_a = V_i|\mathbf{C}_k), N_k - 1) \tag{40}$$

On huomattava, ettei lisäys- ja vähennyskustannuksen laskemisessa vielä oleteta minkään tietyn kategorian muutosta. Todellinen kustannus 42 tietyn kategorian lisäämiselle ryhmään saadaan käyttäen korjauskerrointa 41.

$$I_f(k,a,x) = E(F(A_a = V_i|\mathbf{C}_k) + 1, N_k + 1) - E(F(A_a = V_i|\mathbf{C}_k), N_k + 1) \tag{41}$$

$$I(k,a,x) = N_k * S(k,a) + (N_k + 1) * (I_s(k,a) + I_f(k,a,x)) \tag{42}$$

Todellinen kustannus tietyn kategorian vähentämiselle 44 ryhmästä saadaan käyttäen korjauskerrointa 43.

$$D_f(k,a,x) = E(F(A_a = V_i|\mathbf{C}_k) - 1, N_k - 1) - E(F(A_a = V_i|\mathbf{C}_k), N_k - 1) \tag{43}$$

$$D(k,a,x) = N_k * S(k,a) + (N_k - 1) * (D_s(k,a) + D_f(k,a,x)) \tag{44}$$

Vektorin lopullinen siirtokustannus saadaan laskemalla yhteen kaikkien kategorioiden lisäys- ja vähennyskustannukset. Sama voidaan ilmaista pseudokoodina. Aluksi määritellään yksittäisen kategorian entropia kuvan 28 mukaisesti.

```
CategoryEntropy(instances, count) {
  entropy := 0;

  IF (instances > 0 AND count > 0) {
    probability := instances / count;
    entropy = -probability * log(probability);
  }

  RETURN entropy;
}
```

Kuva 28: Yhden kategorian tuottaman entropian laskeminen

Funktiota $CategoryEntropy$ apuna käyttäen voidaan laskea ryhmälle jokaisen kategorian

vähentämisestä ja lisäämisestä koituva entropian muutos kuten kuvassa 29. Lopuksi voidaan laskea vektorin siirrosta aiheutuvat kulut ajassa $O(D)$ kuvan 30 funktion avulla.

```
CountCategoryCosts(cluster) {
  size := Size(C);

  FOR i := 1 TO Attributes(X) DO {
    current := 0; increase := 0; decrease := 0;

    FOR j := 1 TO Categories(i) DO {
      inst     := CategoryInstances(cluster, i, j);
      current  := current  + CategoryEntropy(inst, size);
      increase := increase + CategoryEntropy(inst, size+1);
      decrease := decrease + CategoryEntropy(inst, size-1);
    }
    current := size * current;

    FOR j := 1 TO Categories(i) DO {
      inst := CategoryInstances(cluster, i, j);

      increaseFix   := CategoryEntropy(inst+1, size+1)
                     - CategoryEntropy(inst,   size+1);
      fixedIncrease := (size + 1) * (increase + increaseFix);
      increaseCost(cluster, i, j) := fixedIncrease - current;

      decreaseFix   := CategoryEntropy(inst-1, size-1)
                     - CategoryEntropy(inst,   size-1);
      dixedDecrease := (size - 1) * (decrease + decreaseFix);
      decreaseCost(cluster, i, j) := fixedDecrease - current;
    }
  }
}
```

Kuva 29: Siirtojen aiheuttamien entropiamuutosten laskeminen

Jokaisen siirron jälkeen joudutaan päivittämään kustannustaulukot kahdelle ryhmälle. Tämä tarkoittaa kompleksisuusanalyysissa kaikkien vektoreiden, attribuuttien ja kategorioiden läpi käymistä kahdesta ryhmästä. Jos oletetaan kaikki ryhmät yhtäsuuriksi, saadaan $O(\frac{2}{K}DM_{avg})$. Tämän jälkeen entropian muutos yksittäiseen ryhmään saadaan lineaarisessa ajassa attribuuttien lukumäärän suhteen. Yhden kierroksen aikana tämä tarkoittaa aikavaativuutta $O(KND)$. Jos keskimääräinen kierroksen aikana tapahtuvien siirtojen lukumäärä on X, saadaan kokonaisaikavaativuudeksi $O(IKND + IX\frac{2}{K}DM_{avg})$. Vaikka vektorin etäisyyden laskeminen nopeutuu, päivityksestä syntyvä kustannus hidastaa kokonaissuoritusaikaa.

```
EntropyChange'(C, C', vector) {
  distance := 0;

  FOR i = 1 TO Attributes(X)  {
    v        := Category(vector, i);
    change   := IncreaseCost(C', i, v) + DecreaseCost(C, i, v);
    distance := distance + change;
  }

  RETURN distance;
}
```

Kuva 30: Entropian muutoksen laskeminen muutostaulukon avulla

6.3 Samankaltaisia menetelmiä

6.3.1 Entropian minimointi satunnaistetulla algoritmilla

Kuten edellisessä kohdasa mainittiin, samaan ideaan perustuva satunnainen vektoreiden testaaminen satunnaisiin ryhmiin on olemassa [26]. K-entropies menetelmästä poiketen tässä menetelmässä valitaan satunnaisesti vektori ja toinen ryhmä. Jos vektorin siirtäminen toiseen ryhmään pienentää entropiaa, suoritetaan siirto. Tätä jatketaan, kunnes on kulunut riittävän pitkä aika siten, ettei siirtoja ole suoritettu.

Satunnaistettu iterointi pienentää suoritusaikaa, koska täydellistä läpikäyntiä ei tarvitse tehdä. Toisaalta koska kaikkia vektoreita ei aina käydä läpi, on ryhmittelyn laatu keskimäärin heikompi kuin k-entropies menetelmässä. Satunnaisuus aiheuttaa kuitenkin sen, että suoritusaika ja laatu voi vaihdella merkittävästi eri ajojen välillä. Valinta deterministisen ja satunnaisen iteroinnin välillä riippuu käyttötarkoituksesta.

Satunnaisuus voi huolimattomasti käytettynä aiheuttaa algoritmin pysähtymättömyyden. Tässä tapauksessa ongelmaa ei kuitenkaan ole, koska jokainen siirto pienentää entropiaa kuten k-entropies menetelmän tapauksessa. Näin ollen ei voida koskaan päätyä samaan ryhmittelyyn ja ryhmittelyjen rajallinen määrä takaa myös algoritmin pysähtymisen.

6.3.2 Coolcat

Coolcat menetelmä [5] on heuristinen ryhmittelymenetelmä suurien kategoristen aineistojen ryhmittelyyn. Entropian käyttäminen ryhmittelykriteerinä tekee Coolcat menetelmästä erityisen mielenkiintoisen.

Coolcat menetelmässä on alustus- ja lisäysvaihe [5]. *Alustusvaiheessa* valitaan joukko näytteitä kaikista vektoreista. Näytteistä valitaan kaksi mahdollisimman erilaista vektoria. Tämän jälkeen etsitään yhteensä K vektoria ja jokaisella kierroksella pyritään valitsemaan vektori siten, että minimietäisyys kaikkiin jo valittuihin vektoreihin on mahdollisimman suuri. Alustusvaiheen lopputuloksena on K ryhmää, joissa jokaisessa on yksi vektori. *Lisäysvaiheessa* käydään läpi ryhmiin sijoittamattomat vektorit ja sijoitetaan ne ryhmiin minimoiden entropiaa. Toisin sanoen kyseessä on laskukaava, jota käytetään k-entropies menetelmässä.

Alkioiden käsittelyjärjestys vaikuttaa lopputulokseen. On mahdollista, että alunperin hyvin ryhmään sopinut vektori ei enää sovikaan hyvin ryhmään muiden vektoreiden lisäämisen jälkeen. Vaikutusta pyritään ehkäisemään sijoittamalla $m\%$ huonoiten ryhmiin sopivia vektoreita uudelleen. Huonoiten sopivien alkioiden etsimisessä käytetään samankaltaisuusmittaa, jonka arvo on kategorioiden esiintymistodennäköisyyksien tulo. Tämä samankaltaisuusmitta muistuttaa k-distributions menetelmässä käytettyä mittaa. Kuvassa 31 on ilmaistu sama pseudokoodina.

Coolcat menetelmä eroaa k-entropies menetelmästä alustusvaiheessa sekä vektoreiden uudelleen sijoittamisessa. Tietyllä tavalla voidaan ajatella, että Coolcat on k-entropies menetelmän kaltainen, mutta suorittaa huomattavasti vähemmän iterointia. Voidaan nähdä, että k-entropies tuottaa parempia tuloksia nopeuden kustannuksella, koska se käy vektorit useaan kertaan läpi.

```
Coolcat(X) {
  Valitaan S vektorin näyte.
  Valitaan näytteestä 2 mahdollisimman erilaista vektoria.

  FOR i := 3 TO K {
    Valitaan vektori maksimoiden minimietäisyyttä jo valittuihin.
  }

  Luodaan valituista vektoreista omat ryhmänsä.

  FOREACH vektori, jota ei valittu {
    Etsitään ryhmä, joka on entropian näkökulmasta lähimpänä.
    Sijoitetaan vektori ryhmään.
  }

  Poistetaan m prosenttia huonoiten ryhmiin sopivia vektoria.

  FOREACH poistettu vektori {
    Sijoitetaan vektori uudelleen.
  }
}
```

Kuva 31: Coolcat pseudokoodina

7 Ryhmittelymenetelmien vertaileminen

Tässä luvussa vertaillaan ryhmittelymenetelmiä käytännössä todellisilla aineistoilla. Aineistojen lisäksi esitellään toteutuksien yksityiskohtia. Ryhmittelyn laadun vertailemiseksi käytetään entropiaa ja ryhmittelymenetelmien soveltuvuutta käytäntöön tarkastellaan luokitteluvirheen avulla. Koska todelliset aineistot ovat usein suuria ja moniulotteisia, vertaillaan myös ajankäyttöä.

7.1 Koejärjestely

Aikaisemmin luvussa 2.4 käsiteltiin kategorisen datan ryhmittelymenetelmän toivottuja ominaisuuksia. Testitapauksilla pyritään selvittämään kuinka hyviä ryhmittelytuloksia menetelmät tuottavat eri tyyppisillä aineistoilla.

Todellisissa sovelluksissa kohdataan usein tilanteita, joissa tietoja puuttuu [35]. Esimerkiksi sieniaineistossa 7.2.1 puuttuu arvoja. Numeerisen datan tapauksessa ongelma voidaan ratkaista laskemalla attribuutin keskiarvo ja korvata puuttuvat arvot keskiarvolla. On myös mahdollista päätellä muun datan ja vektorin muiden arvojen avulla todennäköisin arvo. Ongelmaan ei ole olemassa yksikäsitteistä parasta ratkaisua. Tutkielmassa on päädytty poistamaan vektorit tai attribuutit, jotka sisältävät puuttuvia arvoja. Lisäksi on vältetty aineistoja, joista puuttuu paljon arvoja.

7.2 Aineistot

Tässä luvussa kuvataan vertailuun valittuja aineistoja, jotka on poimittu lähteestä [1]. Aineistoihin on valittu tavallista kategorista dataa, binääridataa ja transaktiodataa. Aineistojen koot vaihtelevat muutamista kymmenistä vektoreista useisiin miljooniin vektoreihin. Aineistojen ulottuvuuksien määrä vaihtelee 16 ulottuvuudesta 70 ulottuvuuteen.

7.2.1 Sieniaineisto

Sieniaineisto sisältää havaintoja 23 eri sienilajista. Aineisto on erittäin suosittu, ellei jopa suosituin, testiaineisto kategorisen datan ryhmittelyä käsittelevissä julkaisuissa. Havaintoja on 8124 kappaletta ja jokaisesta havainnosta on 22 eri attribuuttia. Näillä attribuuteilla on yhteensä 126 mahdollista arvoa eli kategoriaa. Attribuutit kuvaavat muun muassa sienen osien eri värejä ja muotoja. Luokkajako myrkyllisiin ja syötäviin sieniin tunnetaan. Tässä tapauksessa myrkyllinen tarkoittaa varmasti myrkyllistä, mahdollisesti myrkyllistä ja ei syötäväksi suositeltavaa. Aineistosta saadaan käytännönläheinen ongelma kun yritetään löytää sienet, jotka ovat varmasti syötäviä.

Tästä aineistossa on 2480 puuttuvaa arvoa attribuutin 11 kohdalta. Koska puuttuvia arvoja on paljon ja ne sijoittuvat samaan attribuuttiin, ongelma on ratkaistu poistamalla attribuutti kokonaan. Jos puuttuvia arvoja olisi vähemmän, voitaisiin lisätä erillinen kategoria puuttuville arvoille. Poistetulla attribuutilla on seitsemän mahdollista arvoa, joten karsitussa aineistossa on 21 attribuuttia ja 119 eri kategoriaa.

Sieniaineistolle ryhmittely suoritetaan sata kertaa ryhmien lukumäärille aina 512 ryhmään asti. Ryhmien lukumäärillä 1024, 2048 ja 4096 ryhmittely toistetaan ainoastaan kymmenen kertaa. Tuloksissa huomioidaan tulosten keskiarvo. Poikkeuksena kuitenkin kokoavat menetelmät ACE ja ROCK, joiden tuloksen perustuvat yhteen ajoon.

7.2.2 Kongressiaineisto

Kongressiaineisto sisältää vuoden 1984 Yhdysvaltain kongressin ääniä. Aineisto kuvaa 435 poliitikon antamia ääniä 16 eri äänestyksessä. Kaikkiaan arvoilla on 46 kategoriaa. Kuten sieniaineisto, kongressiaineisto on erittäin suosittu kategorisen datan ryhmittelyä käsittelevissä julkaisuissa.

Yleisimmin arvot ovat joko puolesta tai vastaan. Lisäksi attribuutilla voi olla kolmas arvo kuvaamaan tapauksia, joissa poliitikon kanta ei tule julki. Puuttuvat vastaukset ovat jakautuneet eri attribuuteille siten, ettei puuttuvien arvojen poistaminen ole mahdollista. Tämän

vuoksi attribuuteille annetaan kolme mahdollista arvoa: kyllä, ei tai muu. Poliitikkojen jako republikaaneihin ja demokraatteihin tunnetaan.

Kongressiaineistolle ryhmittely suoritetaan sata kertaa kaikilla ryhmien lukumäärillä ja tuloksissa huomioidaan tulosten keskiarvo.

7.2.3 Soijapapuaineisto

Soijapapuaineisto sisältää havaintoja soijapavuista. Aineistoja on kaksi erilaista, joista käytetään pienempää. Alkuperäisestä 307 vektoria sisältävästä aineistosta on johdettu 47 vektoria sisältävä aineisto, jossa kaikki attribuutit ovat nominatiivisia eikä siinä ole puuttuvia arvoja. Attribuutteja on 35 kappaletta, joka on paljon vektoreiden lukumäärään nähden. Kategorioita on 72 kappaletta. Aineisto voidaan luokitella neljään eri luokkaan sairauksien mukaan. Kolmessa luokassa on kymmenen havaintoa ja neljännessä luokassa 17.

Soijapapuaineistolle ryhmittely suoritetaan sata kertaa kaikilla ryhmien lukumäärillä ja tuloksissa huomioidaan tulosten keskiarvo.

7.2.4 Sydänaineisto

Sydänaineisto sisältää tietoa sydänkuvauksiin sovelletusta yksifotoniemissiotomografiasta (Single Proton Emission Computed Tomography). Alkuperäisistä SPECT-kuvista on poimittu 44 jatkuvaa ominaisuushahmoa ja jatkoprosessoinnin tuloksena on saatu 22 binääriattribuutteja. Lopullisessa aineistossa on 267 vektoria, joissa jokaisessa on 22 binääristä attribuuttia eli yhteensä 44 kategoriaa. Kuvattujen potilaiden sydäntoiminta voidaan luokitella kahteen luokkaan: säännönmukaisiin ja poikkeaviin.

Sydänaineistolle ryhmittely suoritetaan sata kertaa kaikilla ryhmien lukumäärillä ja tuloksissa huomioidaan tulosten keskiarvo.

7.2.5 Kasviaineisto

Kasviaineisto on transaktiodataa eri kasvien kasvupaikoista. Aineistossa on 34781 vektoria ja 70 attribuuttia. Koska transaktiodata on binääristä, tarkoittaa se 140 kategoriaa. Jokainen kasvia kohti on yksi vektori ja jokaista aluetta varten on oma attribuutti. Kasveille pyritään löytämään sopiva jaottelu kasvupaikkojen perusteella. Oikeaa luokittelua ei kuitenkaan ole tiedossa.

Kasviaineistolle ryhmittely suoritetaan kymmenen kertaa kaikilla ryhmien lukumäärillä ja tuloksissa huomioidaan tulosten keskiarvo. ACE ja ROCK menetelmillä tätä aineistoa ei ryhmitellä liian suuren muistin kulutuksen takia.

7.2.6 Väestöaineisto

Väestöaineisto on peräisin väestönlaskennasta ja se on valittu suorituskyvyn testaamiseksi. Aineisto sisältää peräti 2458285 vektoria, 68 attribuuttia ja 396 kategoriaa.

Kaikki attribuutit eivät ole nominatiivisia, vaan aineisto sisältää myös ordinaalisia attribuutteja. Nominatiivista dataa käsittelevien menetelmien ryhmittelytuloksen laadun arvioiminen ei ole mielekästä, koska menetelmiä laadittaessa ei ole huomioitua ordinaalisia attribuutteja. Ryhmittelymenetelmät kykenevät kuitenkin käsittelemään aineistoa, koska attribuuttien arvojoukot ovat diskreettejä.

Muista testeistä poiketen ryhmien lukumäärä kiinnitetään ja testissä tarkastellaan ajankäyttöä aineiston koon suhteen. Ryhmien lukumääräksi on valittu 16 ja aineiston koko vaihtelee 1000 vektorista aina 256000 vektoriin. Ryhmittely toistetaan kaikilla asetuksilla kymmenen kertaa. ACE ja ROCK eivät ole mukana vertailussa.

7.2.7 Yhteenveto aineistoista

Taulukkoon 7 on kerätty eri aineistojen tärkeimmät ominaisuudet.

Aineisto	Vektoreita	Attribuutteja	Kategorioita
Sieniaineisto	8124	21	119
Kongressiaineisto	435	16	46
Soijapapuaineisto	47	35	72
Sydänaineisto	267	22	44
Kasviaineisto	34781	70	140
Väestöaineisto	2458285	68	396

Taulukko 7: Yhteenveto aineistoista

7.3 Toteutusten yksityiskohdat

K-modes menetelmässä tyypin 1 tasapeli, eli tasapeli moodiarvoa laskettaessa, ratkaistaan valitsemalla aina ensisijaisesti nykyinen moodiarvo. Jos nykyinen moodi ei kuitenkaan ole parhaiden joukossa, valitaan ensimmäinen löydetty arvo. Myös tyypin 2 tasapelissä, eli vektoreiden uudelleen sijoittamisessa, valitaan vastaavasti ensisijaisesti nykyinen ryhmä tai ensimmäinen löydetty ryhmä. K-medoids menetelmässä toimitaan vastaavalla tavalla nykyistä medoidia suosien.

Jakaumapohjaisissa menetelmissä tasapelejä ei synny edustajaa laskettaessa ja osituksessa tasapelit ovat huomattavasti harvinaisempia, koska laskennassa käytetään liukulukuaritmetiikkaa. Tasapelitilanteissa liukulukulaskennan pyöristysvirheet voisivat kuitenkin satunnaistaa suoritusta ja pahimmillaan aiheuttaa algoritmin pysähtymättömyyden. Siksi vektorin nykyistä ryhmää suositaan pienellä marginaalilla, joka on 0.000001.

Iteratiivisissa menetelmissä ryhmien tyhjeneminen ositusta päivittäessä on kategorisen datan ryhmittelyssä yleisempää kuin numeerisen datan ryhmittelyssä. Kaikissa iteratiivissa menetelmissä viimeisen vektorin siirtäminen pois ryhmästä on estetty.

Välitöntä päivitystä käyttävissä k-histograms ja k-entropies menetelmissä suoritetaan alussa aineiston järjestyksen sekoittaminen. Sekoittaminen on toteutettu siten, että alkuperäistä aineistoa pidetään muuttumattomana ja järjestys määräytyy erillisen järjestystaulukon avulla. Sekoittamisella pyritään estämään aineiston järjestyksen vaikutus algoritmin kulkuun. Ilman sekoittamista samat vektorit käsiteltäisiin aina ensimmäisenä ja näihin kohdistuneet päivitykset vakuttaisivat algoritmin kulkuun. Muissa iteratiivisissa menetelmissä ei esiinny samaa ongelmaa, koska kaikki päivitykset tehdään yhtäaikaa.

ROCK algoritmia on hieman yksinkertaistettu. Yksinkertaistetussa versiossa ei käytetä ollenkaan lokaaleja kekoja. Lokaalien kekojen käyttämisestä ei ole hyötyä ja se monimutkaistaa menetelmää turhaan. Kekojen avulla etsitään parhaiten yhteensopivia ryhmiä. Koska ryhmäpareja on enimmillään N^2, käyttämällä ainoastaan yhtä kekoa operaatioiden aikavaativuus on $\log(N^2)$ eli $2\log(N)$. Sen sijaan, jos globaalikeko sisältää N lokaalia kekoa ja näistä jokainen N ryhmäparia, operaatioiden aikavaativuus on $\log(N) + \log(N)$ eli $2\log(N)$. Aikavaativuus on sama kuin yhden keon tapauksessa. Algoritmi yksinkertaistuu huomattavasti, koska ei tarvitse käsitellä kuin yhtä kekoa eikä myöskään tarvitse ratkaista sitä kumman ryhmän lokaaliin kekoon ryhmäpari sijoitetaan. Yhden keon käyttäminen saattaa muuttaa ryhmien yhdistämisjärjestystä tasapelitilanteissa verrattuna useamman keon käyttämiseen. Toisaalta sama tilanne voitaisiin aiheuttaa muuttamalla vektoreiden järjestystä aineistossa ja menetelmä ei saisi olla herkkä vektoreiden järjestykselle.

ACE menetelmän alustusvaihetta on nopeutettu. Kahden ryhmän, joissa on molemmissa vain yksi vektori, yhdistämisestä aiheutuva entropian kasvu voidaan laskea yksinkertaisemmin. Yksittäisen vektorin muodostaman ryhmän entropia on aina 0. Jos kahdella vektorilla attribuutin arvo on sama, voidaan päätellä ettei kyseinen attribuutti lisää entropiaa lainkaan. Jos vektoreilla on eri attribuutin arvo, yhdistäessä kahdella attribuutin kategorialla on todennäköisyys 0.5. Yksittäisen kategorian aiheuttama entropia on $0.5\log(0.5)$ ja koska kategorioita on kaksi, jokainen eriävä attribuutti kasvattaa yhdistämiskustannusta $\log(0.5)$. Kertomalla $\log(0.5)$ eriävien attribuuttien lukumäärällä ja kertomalla tulo lopullisella ryhmän koolla 2, saadaan lopputuloksena ryhmien yhdistämiskustannus. Nopeutus ei kuitenkaan vaikuta algoritmin kokonaisaikavaativuuteen.

Sekä ROCK että ACE menetelmissä on mahdollista törmätä tasapeleihin yhdistämisvaiheessa. Toisin sanoen kahden tai useamman ryhmäparin yhdistämiskustannus on sama. Ongelma on ratkaistu satunnaisella valinnalla. Tästä syystä menetelmät eivät ole täysin deterministisiä, vaikka useimmiten tulos on sama riippumatta yhdistämisjärjestyksestä.

7.4 Tulokset

Tässä luvussa vertaillaan menetelmiä käytännössä. Vertailun helpottamiseksi taulukkoon 8 on kerätty yhteenveto eri menetelmien tärkeimmät ominaisuudet.

Menetelmä	Tyyppi	Aikavaativuus	Tilavaativuus
K-entropies	Välitön päivitys	$O(INKDM_{avg}T_L)$	$O(N)$
K-distributions	K-means	$O(INKDT_L)$	$O(N)$
K-histograms	Välitön päivitys	$O(INKD)$	$O(N)$
K-medoids	K-means	$O(INKD)$	$O(N)$
K-modes	K-means	$O(INKD)$	$O(N)$
ACE	Kokoava	$O(N^2DM_{avg})$	$O(N^2)$
ROCK	Kokoava	$O(N^2 + NR_{max}R_{avg} + N^2 \log N)$	$O(min\{N^2, NR_{max}R_{avg}\})$

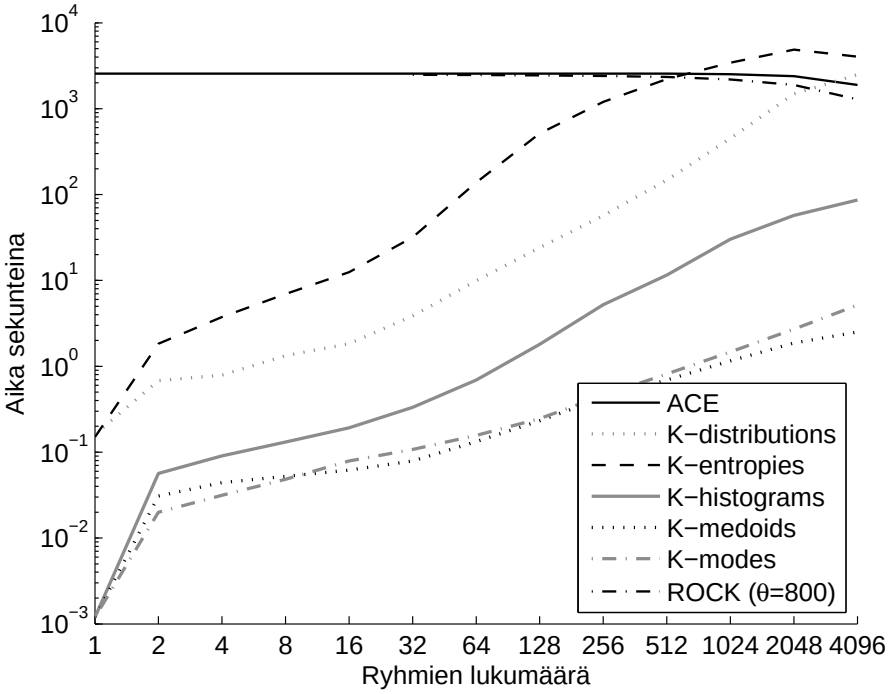
Taulukko 8: Menetelmien tärkeimmät ominaisuudet

7.4.1 Ryhmittelyyn käytetty aika

Tässä kappaleessa vertaillaan eri menetelmien ajankäyttöä. Pienien aineistojen kohdalla ajankäytössä ei näy merkittäviä eroja, joten vertailua on tehty ainoastaan suuremmilla aineistoilla. Valitettavasti kokoavilla menetelmillä ei pysty ryhmittelemään suurimpia aineistoja suuren ajan- ja muistinkäytön vuoksi. Tästä syystä keskitytään iteratiivisiin menetelmiin. Iteratiivisien menetelmien tapauksessa on mielenkiintoista tarkkailla myös iteraatioiden lukumäärää. Aloitetaan sieniaineiston ryhmittelyyn käytetystä ajasta, joka on esitetty kuvassa 32.

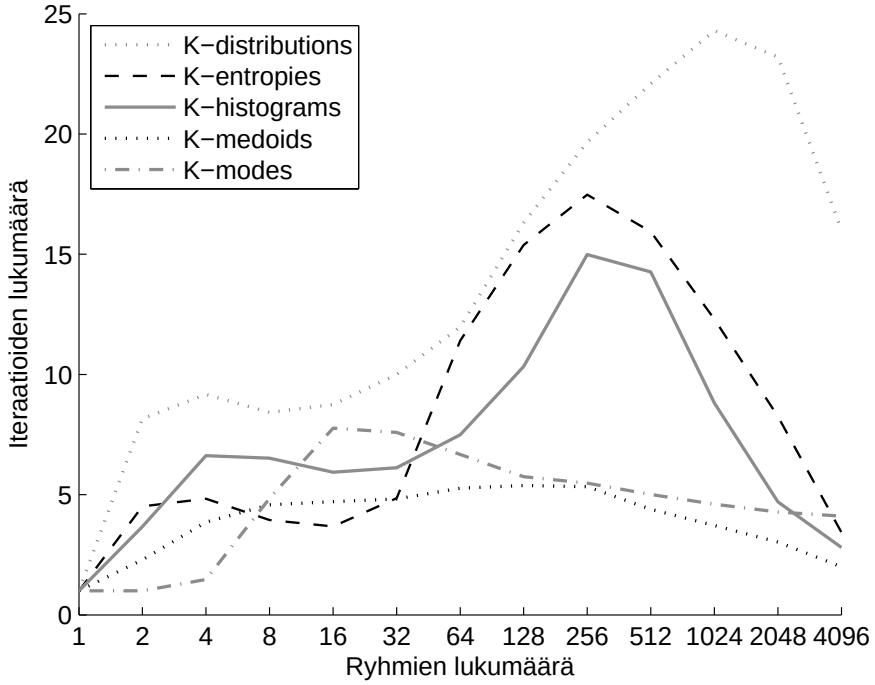
Kokoavien menetelmien ajankäyttö pienenee isommilla ryhmien lukumäärillä, koska lähtökohtana on N ryhmää ja ryhmät vähenevät suorituksen edetessä. Kuvassa ajankäytön vaihtelu on pientä, koska suurin osa ajasta kuluu suorituksen alussa. Alussa ryhmien yhdistäminen on raskaampaa, koska päivitettävien yhdistämiskustannusten lukumäärä riippuu ryhmien lukumäärästä.

Vektoria ryhmän edustajana käyttävät menetelmät k-modes ja k-medoids ovat nopeimpia. Ero k-histograms menetelmään on yllättävän suuri, vaikka näiden kolmen menetelmän teoreettinen aikavaativuus on sama. Toinen huomiota herättävä asia on k-distributions ja k-entropies menetelmien välisen eron kaventuminen lopussa. Muilta osin menetelmien järjestys



Kuva 32: Käytetty aika sieniaineistolle

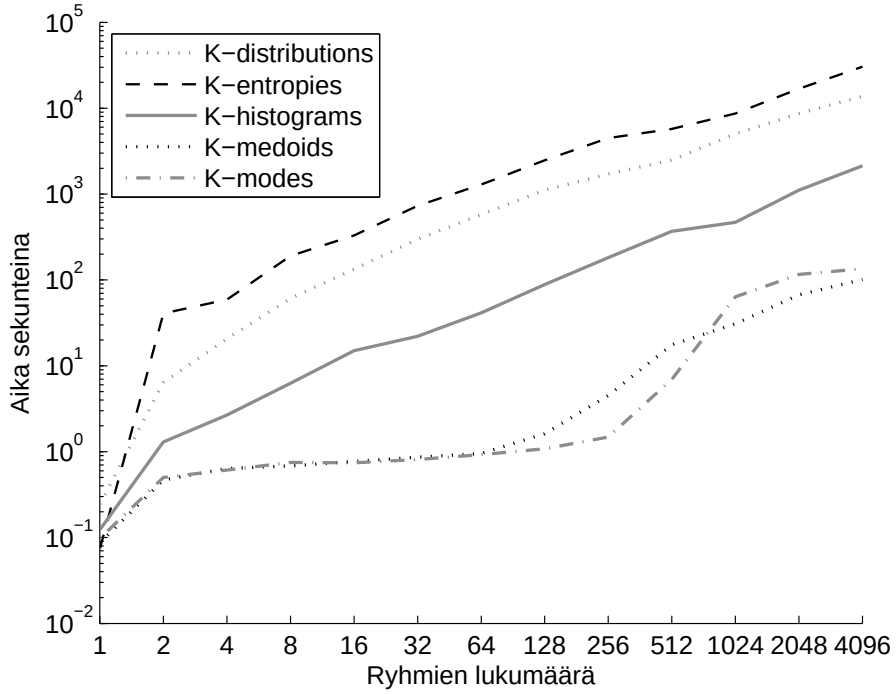
noudattaa tarkasti teoreettisessa tarkastelussa ennustettua järjestystä. Tärkeä selittävä tekijä on iteraatioiden lukumäärä, joka on esitetty kuvassa 33.



Kuva 33: Iteraatioiden lukumäärä sieniaineistolle

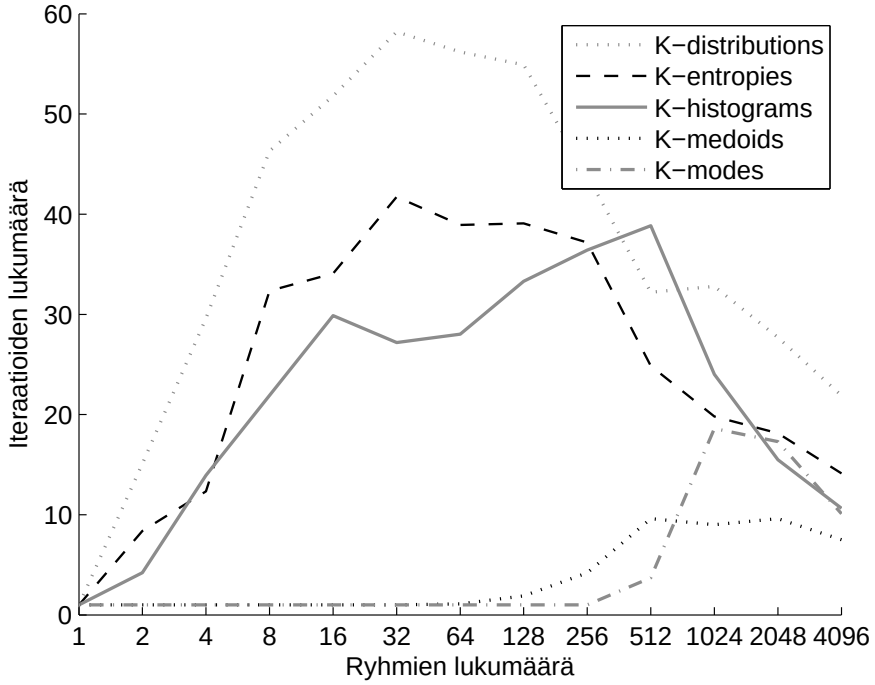
Iteraatioiden lukumäärä selittää k-distributions ja k-entropies menetelmien välisen eron kavantumisen. Kaikissa menetelmissä iteraatioiden lukumäärä alkaa putoamaan kun ryhmien lukumäärä kasvaa riittävän suureksi. K-entropies menetelmässä tämä putoaminen tapahtuu aikaisemmin kuin k-distributions menetelmässä.

Kuvassa 34 on ryhmittelyssä käytetty aika kasviaineistolle. Tulokset ovat samansuuntaisia kuin sieniaineistolla. K-modes ja k-medoids menetelmien ajankäyttö on kuitenkin poikkeavaa pienemmillä ryhmien lukumäärillä. Tarkastelemalla iteraatioita kuvasta 35 ero on nähtävissä vieläkin selkeämmin.



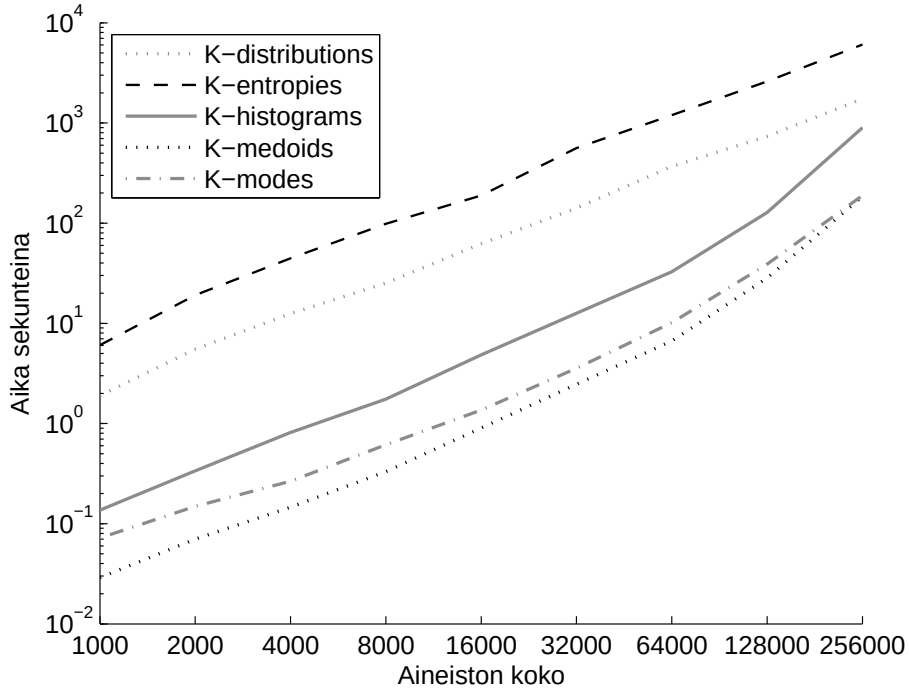
Kuva 34: Käytetty aika kasviaineistolle

Kuvasta 35 nähdään, että k-modes ja k-medoids pysähtyvät pienillä ryhmien lukumäärillä yhden iteraation jälkeen. Yksi iteraatio tarkoittaa sitä, ettei menetelmä tee yhtään muutoksia alustuksen jälkeen. Selitys löytyy aineistosta ja ryhmän edustajasta. Aineisto on transaktiodataa, jossa arvoa epätosi kuvaava kategoria on selkeästi yleisin. Näin ollen ryhmän edustajaa kuvaava vektori on useimmiten nollavektori. Pysähtyminen ensimmäiselle kierrokselle tarkoittaa täysin satunnaista ryhmittelyä, joten on selvää etteivät nämä menetelmät sovellu transaktiodatan ryhmittelyyn.

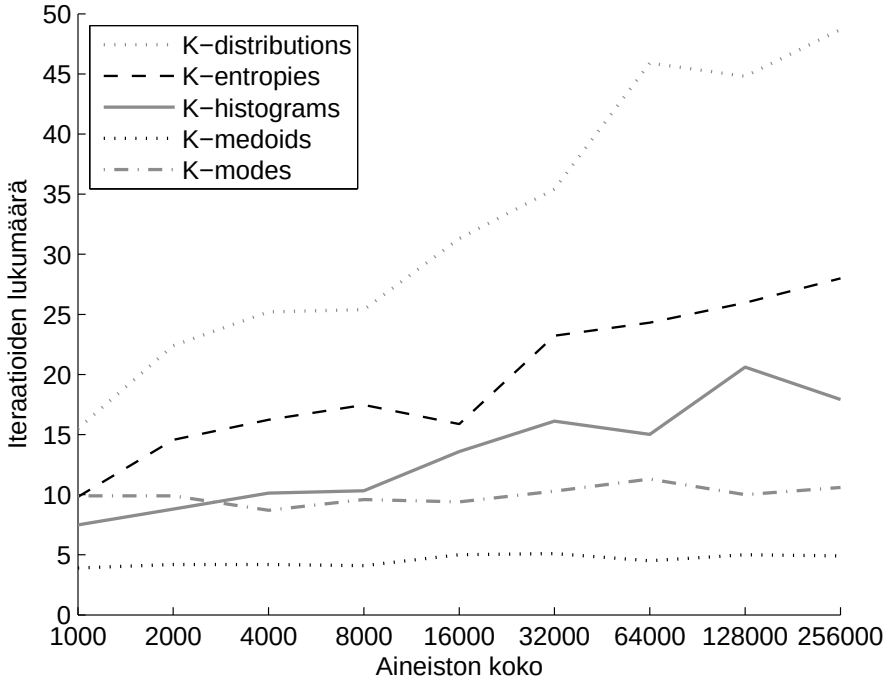


Kuva 35: Iteraatioiden lukumäärä kasviaineistolle

Lopuksi tarkastellaan aikaa aineiston koon kasvaessa. Ryhmien lukumäärä pidetään 16. Kuvassa 36 on käytetty aika väestöaineistolle ja kuvassa 37 vastaavien ajojen iteraatiot.



Kuva 36: Käytetty aika väestöaineistolle



Kuva 37: Iteraatioiden lukumäärät väestöaineistolle

Kuvaaja noudattaa pitkälti aikaisempia tuloksia. Lopussa huomio kuitenkin kiinnittyy k-histograms menetelmän voimakkaaseen ajankäytön lisääntymiseen. Tulos ei ole selitettävissä iteraatioiden lukumäärällä. Selitys löytyy kuitenkin laitteistosta. K-histograms menetelmässä vektorit on sekoitettu pitämällä aineisto alkuperäisessä järjestyksessä ja hakemalla seuraava käsiteltävä vektori erillisen järjestystaulukon avulla. Toisin sanoen menetelmä käsittelee muistissa olevaa dataa hajanaisessa järjestyksessä. Kun aineisto kasvaa liian suureksi, sitä ei voida pitää kokonaisuudessaan välimuistissa. Laitteisto osaa puskuroida dataa välimuistiin menetelmissä, joissa aineistoa käydään läpi oikeassa järjestyksessä.

Kokeiden pohjalta voidaan sanoa, että aikavaativuuksanalyysissä käytetty iteraatioiden lukumäärä I vaihtelee eri menetelmissä. Vektoripohjaiset menetelmät vaativat vähemmän iteraatioita kuin jakaumapohjaiset. Tulos on intuitiivisesti järkevä, koska vektoripohjaisien menetelmien erottelukyky on heikompi. Iteraatioiden lukumäärään vaikuttavia muuttujia ovat menetelmä, ryhmien lukumäärä ja aineisto. Iteraatioiden lukumäärä ei kuitenkaan selitä kaikkea, vaan sen lisäksi jakaumapohjaisissa menetelmissä joudutaan suorittamaan kerto- ja jakolaskuja sekä k-distributions ja k-entropies menetelmässä käytetään logaritmi-funktiota. Taulukossa 9 on esitetty eri operaatioiden käyttämä aika testausympäristössä. Tuloksia varten

on suoritettu 10 000 iteraatiota, joissa jokaisessa suoritetaan 1000 laskuoperaatiota. Ilman operaatiota suoritettavan iteroinnin tulos kertoo, että silmukan aiheuttama kustannus ei ole merkittävä.

Operaatio	Aika	Kerroin
Ei operaatioita	0.00	0.00
Yhteenlasku	2.00	1.00
Vähennyslasku	2.00	1.00
Kertolasku	2.68	1.34
Jakolasku	6.52	3.26
Logaritmi	46.7	23.3

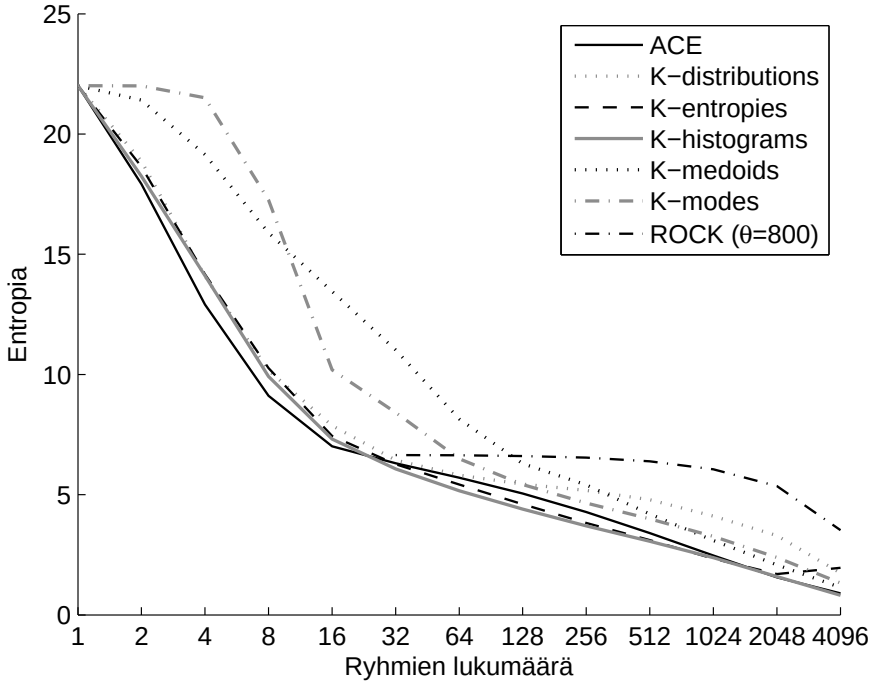
Taulukko 9: Eri operaatioiden ajankäyttö

Taulukossa 9 esitettyjen tuloksien perusteella voidaan selittää eri menetelmien eroja käytännössä. Tarkan vakiokertoimen laskeminen edellyttäisi funktiokutsujen ja silmukoiden aiheuttamien kustannuksien huomioimisen sekä erilaisten iterointitapojen vaikutuksen huomioimisen välimuistien käytössä. Kustannukset vaihtelevat eri laitteilla.

7.4.2 Ryhmittelymenetelmien tuottamat entropiat

Seuraavaksi tarkastellaan ryhmittelymenetelmien tuottamia ryhmittelyjä entropiaa käyttäen. Kuvassa 38 on eri ryhmittelymenetelmien tuottamat entropiat sieniaineistolle.

Kuvasta 38 havaitaan taitekohta 16 ja 32 ryhmän kohdalla. Taitekohta selittyy sillä, että aineistossa on 24 sienilajia. Entropia pienenee voimakkaasti kun sienilajeja jaetaan eri ryhmiin, koska eri lajien edustajat ovat selkeästi erilaisia. Sen sijaan entropia ei pienene yhtä merkittävästi, jos laji jaetaan useampaan ryhmään, koska erot saman lajien edustajien välillä ovat pieniä. Vektoripohjaiset menetelmät menestyvät heikosti vertailussa. Erot ovat erityisen voimakkaita alle 128 ryhmällä. Tulos on odotettu, koska vektoripohjainen edustaja ei kuvaa ryhmää hyvin. ACE ja jakaumapohjaiset menetelmät ovat melko tasaväkisiä lukuunottamatta K-distributions ja k-entropies menetelmissä tapahtuvaa laadun heikkenemistä suuremmilla ryhmien lukumäärillä. Syynä tähän voi olla suuri ryhmien lukumäärä verrattuna aineiston kokoon. Esimerkiksi 4096 ryhmän tapauksessa jokaisessa ryhmässä on keskimäärin alle kaksi vektoria.

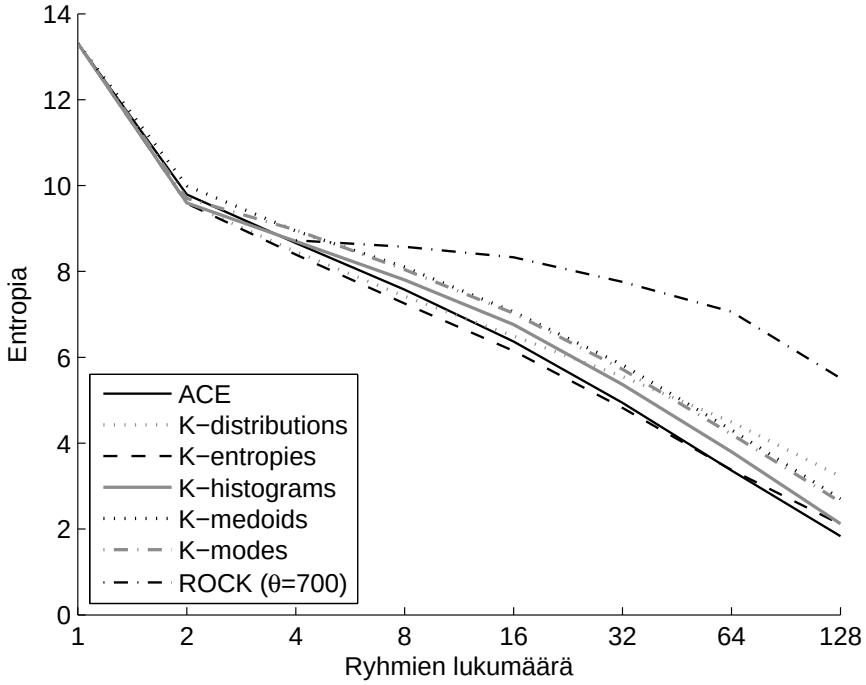


Kuva 38: Entropia sieniaineistolle

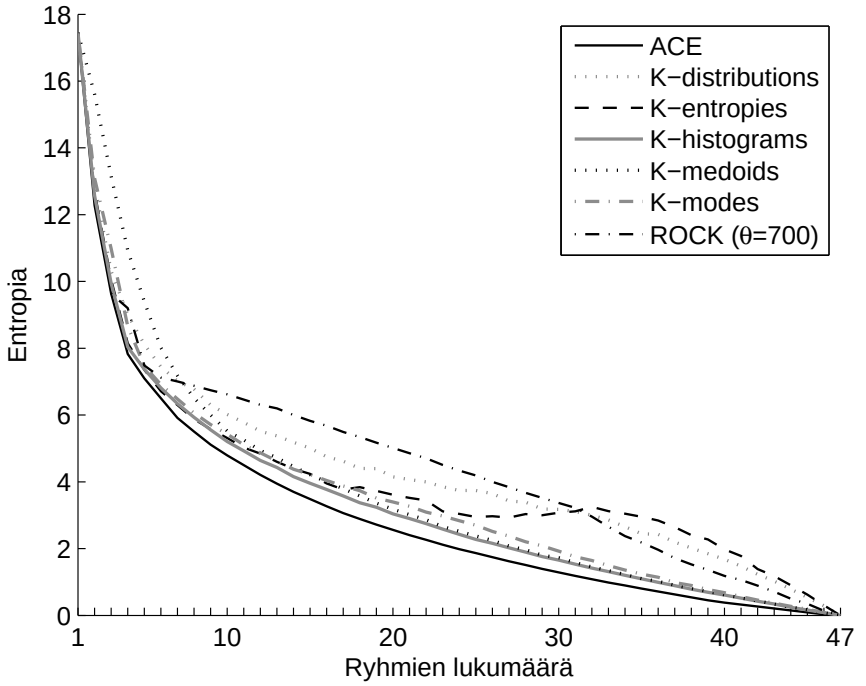
ROCK menetelmän tulos on huomiota herättävä. Kyseessä ei kuitenkaan ole virhe. Usein ROCK menetelmässä ryhmä, joka muodostuu edellisellä kierroksella, on yhdistämisessä mukana myös seuraavalla kierroksella. Voidaan ajatella, että menetelmä rakentaa yhden ryhmän kerrallaan valmiiksi ennen siirtymistä seuraavaan ryhmään. Tästä johtuen välitulokset eivät ole hyviä. Lisäksi pienimpiä ryhmien lukumääriä ei ole mahdollista saavuttaa. Menetelmä kuitenkin tuottaa hyvän tuloksen sillä ryhmien lukumäärällä, jota menetelmä tarjoaa oikeaksi ryhmien lukumääräksi.

Kuvassa 39 on kongressiaineiston tulokset. Kuvasta voidaan erottaa selkeä hidastuminen entropian pienenemisessä kahden ryhmän jälkeen. Hidastuminen selittyy kongressin jakautumisella republikaaneihin ja demokraatteihin. Kongressiaineiston tulokset ovat lähes kaikilta osin sieniaineiston tuloksia vastaavia.

Sen sijaan soijapapuaineisto, joka on esitetty kuvassa 40, tuottaa hieman erilaisia tuloksia. ACE tuottaa selkeästi muita parempia tuloksia. Yllättäen k-distributions menetelmä tuottaa aikaisempaa huonompia tuloksia, mutta sen sijaan k-medoids ja erityisesti k-modes tuottavat tavallista parempi tuloksia.



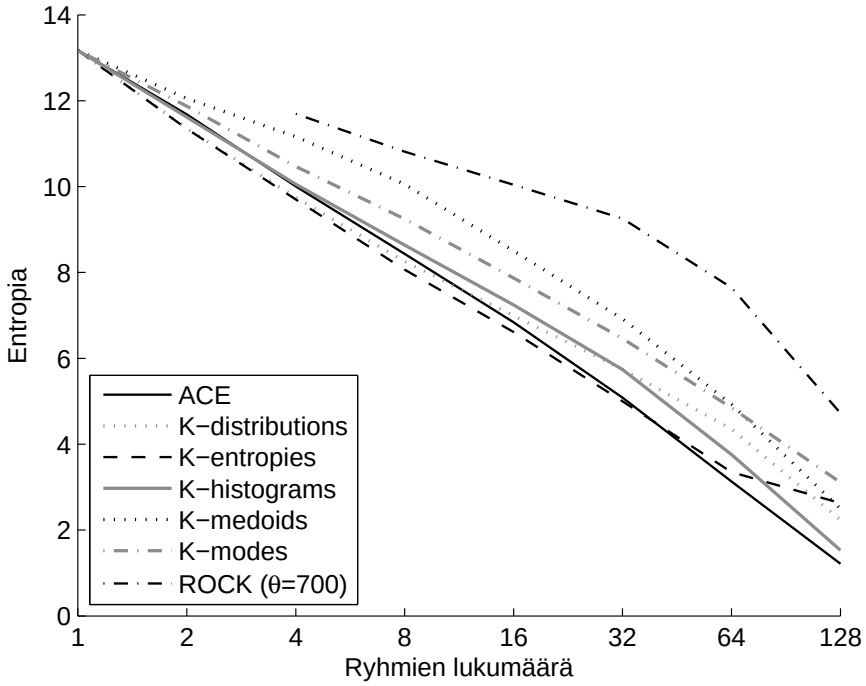
Kuva 39: Entropia kongressiaineistolle



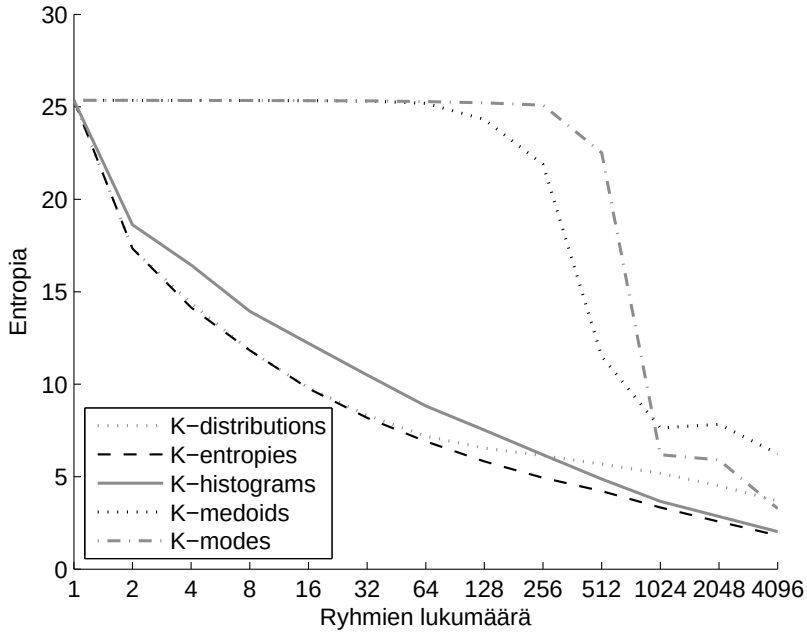
Kuva 40: Entropia soijapapuaineistolle

Kuvassa 41 on entropiat sydänaineistolle. Tulokset ovat hyvin samansuuntaisia kuin sieni- ja kongressiaineiston tapauksessa. Lopuksi kuvassa 42 on ryhmittelymenetelmien tuottamat

entropiat kasviaineistoa ryhmiteltäessä.



Kuva 41: Entropia sydänaineistolle



Kuva 42: Entropia kasviaineistolle

Kuvan 42 silmiinpistävin osa on k-modes ja k-medoidsin tuottamat erittäin huonot tulokset alussa. Kyse on ongelmasta, joka esiteltiin aikatarkastelussa aliluvussa 7.4.1. Transaktiodata

aiheuttaa vektoripohjaisien menetelmien pysähtymisen ensimmäisellä kierroksella, koska edustajiksi tulee nollavektoreita. Ensimmäisellä kierroksella pysähtyminen tarkoittaa satunnaista ryhmittelyä, joka näkyy kuvaajassa korkeana entropiana.

Taulukossa 10 on yhteenveto keskimääräisistä entropioista ja keskihajonnasta kiinnitetyillä ryhmien lukumäärillä. Ryhmien lukumäärät on valittu kuvaajien perusteella siten, että aineiston ominaisuudet ja tulokset tulevat parhaiten esille.

Menetelmä	Sieni $K = 16$		Kongressi $K = 2$		Soijapapu $K = 4$		Sydän $K = 8$		Kasvi $K = 1024$		Väestö $K = 16$	
	ka	kh	ka	kh	ka	kh	ka	kh	ka	kh	ka	kh
ACE	**7.01**	n/a	9.79	0.16	**7.83**	**0**	8.43	0.07	n/a	n/a	n/a	n/a
ROCK	n/a	n/a	**9.30**	**0**	9.20	**0**	10.82	**0**	n/n	n/a	n/a	n/a
k-medoids	13.46	0.71	10.00	0.95	10.94	1.57	10.06	0.44	7.64	0.33	32.61	0.87
k-modes	10.19	1.45	9.70	0.03	8.66	1.06	9.25	0.29	6.19	0.18	31.00	1.26
k-distributions	7.87	0.44	9.59	0.01	9.00	0.93	8.25	0.13	5.19	0.09	28.58	**0.27**
k-histograms	7.31	**0.18**	9.60	0.01	8.04	0.53	8.64	0.15	3.67	**0.02**	29.17	0.48
k-entropies	7.44	**0.18**	9.58	**0**	8.15	0.56	**8.06**	0.07	**3.33**	0.05	**28.51**	0.50

Taulukko 10: Taulukko entropioiden keskiarvosta ja keskihajonnasta

7.4.3 Ryhmittelymenetelmien tuottamat luokitteluvirheet

Kuten aikaisemmin mainittiin, luokitteluvirhe ei välttämättä kuvaa hyvää ryhmittelyä, mutta se on mielenkiintoinen käytännön näkökulmasta. Taulukossa 11 on eri menetelmien antamien tuloksien luokitteluvirheet. Jokaisen aineiston paras tulos on lihavoitu.

Ryhmittelymenetelmä	Sieniaineisto	Soijapapuaineisto	Äänestysaineisto	Sydänaineisto
ACE	**0.110**	**0.000**	0.136	0.378
K-distributions	0.300	0.156	0.127	0.379
K-entropies	0.310	0.077	0.131	**0.371**
K-histograms	0.253	0.061	**0.125**	0.442
K-medoids	0.472	0.367	0.172	0.432
K-modes	0.496	0.157	0.138	0.424
ROCK	n/a	0.489	0.414	n/a

Taulukko 11: Ryhmittelymenetelmien tuottamat luokitteluvirheet

Luokitteluvirhe voidaan laskea järkevästi ainoastaan oikealla ryhmien lukumäärällä. Soijapapuaineistossa on neljä luokkaa. Muissa aineistoissa on kaksi luokkaa. Vertailussa ei ole väestö- ja kasviaineistoissa, koska luokkajakoa ei ole tiedossa.

Kokonaisuudessaan voidaan sanoa, että ACE tuottaa parhaat luokitteluvirheet jättäen iteratiiviset menetelmät selkeästi jälkeensä. Jakaumaa ryhmien edustajina käyttävät menetelmät k-distributions, k-entropies ja k-histograms tuottavat keskimäärin parempia tuloksia kuin vektoria ryhmien edustajina käyttävät k-medoids ja k-modes. Lisäksi k-modes on selkeästi k-medoidsia parempi. ROCK menetelmän vertailu ei ole tässä tapauksessa mielekästä, koska ROCK ei osaa optimoida tulosta kiinnitetyllä ryhmien lukumäärällä eikä välttämättä edes tuottaa oikeaa määrää ryhmiä.

8 Case-esimerkki: myrkkysienien tunnistaminen

Ryhmittelyä voidaan käyttää apuna lukuisissa eri sovelluksissa ja aina ei välttämättä tarvitse edes tuntea sovellusaluetta. Sienihavaintojen jakaminen syötäviin ja myrkyllisiin on eräs mahdollinen sovellusalue. Oletuksena on se, että havaintojen yksityiskohdat korreloivat sienien myrkyllisyyden kanssa.

Aluksi sieniaineisto ryhmitellään kahteen ryhmään. Vaikka luokkajakoa ei käytetä apuna, voidaan ryhmittely suorittaa useasti ja valita paras tulos entropian perusteella. Ryhmittelyn jälkeen on tehtävä vaikea valinta siitä kummassa ryhmässä on syötäviä ja kummassa myrkyllisiä sieniä. Valinta voitaisiin tehdä, jos tunnettaisiin muutamia esimerkkitapauksia. Oikaistaan valinnassa kuitenkin hieman ja oletetaan jaottelu tunnetuksi. Kun tunnetaan osiointi ja ryhmien tyypit, voidaan ryhmien edustajia pitää tyypillisinä esimerkkeinä syötävistä ja myrkyllisistä sienistä.

Seuraavassa vaiheessa on pääteltävä kuinka todennäköisesti yksittäiset havainnot kuuluvat kyseisiin ryhmiin. Sumeita ryhmittelymenetelmiä lukuun ottamatta ryhmittelymenetelmät asettavat vektorit jompaan kumpaan ryhmään riippumatta siitä, kuinka pienestä erosta on kysymys. Eron suuruus on kuitenkin seuraavassa luokittelussa olennainen, koska todennäköisyyden perusteella voidaan karsia pois rajatapauksia.

Olkoon D_e vektorin etäisyys syötävien sienien edustajasta ja D_p vektorin etäisyys myrkyllisten sienien edustajasta. Määritellään pistemäärä S_e sienen myrkyllisyydelle.

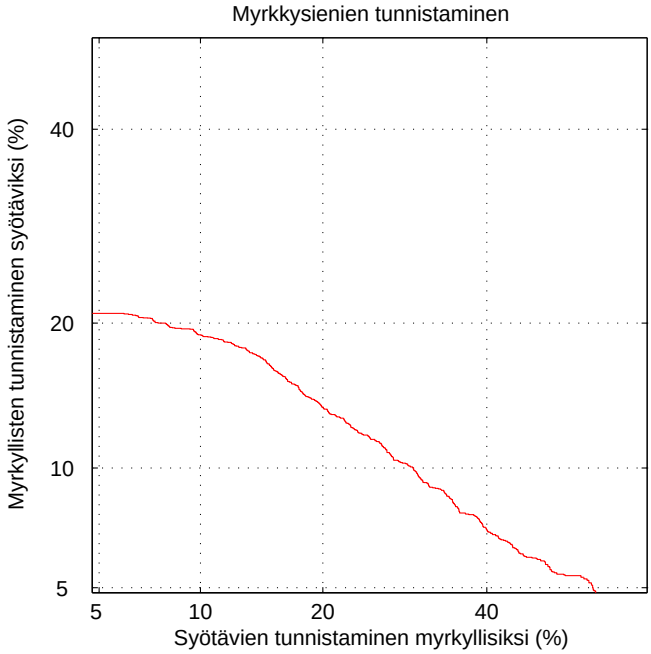
$$S_e = D_e - D_p \tag{45}$$

Kun pistemäärä tiedetään, voidaan määritellä kynnysarvo T.

$$S_e > T \tag{46}$$

Jos epäyhtälö on tosi, luokitellaan sieni myrkylliseksi. Muussa tapauksessa sientä voidaan

pitää syötävänä. Mitä pienempi kynnysarvo T on, sitä suuremmalla todennäköisyydellä sieni luokitellaan myrkylliseksi. Toisin sanoen myös syötäviä sieniä luokitellaan virheellisesti myrkylliseksi. Toisaalta mitä suurempi kynnys arvo T on, sitä todennäköisemmin myrkyllinen sieni luokitellaan syötäväksi. Virheiden vaihtelua kynnysarvon muuttuessa voidaan kuvata DET-kuvaajalla kuten kuvassa 43. DET-kuvaajaa on esitelty tarkemmin muun muassa lähteessä [28].



Kuva 43: Sienien myrkyllisyyden tunnistaminen

Kuvasta 43 nähdään kuinka myrkyllisten sienien luokittelu syötäviksi vähenee sitä mukaan kuin syötäviä sieniä luokitellaan enemmän ja enemmän myrkylliseksi. Lopullinen kynnysarvon valinta riippuu siitä, kuinka suuria ovat näistä kahdesta virheestä aiheutuvat haitat. Myrkkysienien tapauksessa todennäköinen vaihtoehto on minimoida myrkyllisten sienien luokittelua syötäviksi.

Rehellisyyden nimissä on todettava, etten käyttäisi tätä järjestelmää oikeasti sienien myrkyllisyyden tunnistamiseen. Tulos on kuitenkin kohtalainen kun huomioidaan, ettei ongelman ratkaisemiseen ole tarvittu sienituntemusta.

9 Yhteenveto

Tutkielmassa perehdyttiin kategorisen datan ryhmittelyyn liittyvät käsitteisiin ja ominaisuuksiin. Ryhmittelymenetelmistä käytiin läpi k-modes, k-medoids, k-histograms, k-distributions, k-entropies, ACE ja ROCK. Nämä menetelmät jakautuvat iteratiivisiin ja kokoaviin. Iteratiiviset menetelmät jakautuvat edelleen jakaumaa ja vektoria ryhmän edustajana käyttäviin menetelmiin.

Kokoava ACE tuotti useimmiten parhaat tulokset. Jakaumapohjaiset iteratiiviset menetelmät tuottivat ACE menetelmän jälkeen seuraavaksi parhaat tulokset jättäen jälkeensä vektoripohjaiset menetelmät. Myöskään ROCK ei menestynyt vertailussa, joskaan tulokset eivät olleet edes vertailukelpoisia. Samansuuntaisia tuloksia saatiin sekä entropialla että luokitteluvirheellä mitattuna. Tämä antaa viitteitä siitä, että entropia on vahvojen teoreettisten perusteiden lisäksi myös sopiva kriteeri monissa reaalimaailman ongelmissa.

Ajankäytössä vektoripohjaiset menetelmät sen sijaan pärjäsivät parhaiten. Jakaumapohjaiset olivat selkeästi vektoripohjaisia menetelmiä hitaampia, puhumattakaan kokoavista menetelmistä, jotka liikkuivat täysin eri kertaluokassa. Näin ollen laadun ja ajankäytön välillä on selkeä korrelaatio. Menetelmän valinnalla voidaan tasapainoilla korkeamman laadun ja lyhyemmän suoritusajan välillä. Poikkeuksena on ROCK, joka ei onnistunut erityisen hyvin kummallakaan osa-alueella. ROCK menetelmän vahvuudeksi voidaan kuitenkin lukea kokonaisuuden hallinta automaattisen poikkeamien käsittelyn ja ryhmien lukumäärän valinnan ansiosta. Kaikkia menetelmiä voidaan pitää jossain suhteessa onnistuneina.

Tutkimatta jäi suuri joukko jo olemassa olevia menetelmiä. Myöskään tärkeisiin kysymyksiin kuten poikkeamien käsittelyyn ja ryhmien lukumäärän valintaan ei otettu kantaa. Kategorisen datan ryhmittelyyn tarkoitetut menetelmät ovat melko hajanainen joukko joka selittyy sillä, että myös ongelmat ovat hyvin laajalta alueelta. Kuten muissakin ryhmittelyongelmissa, ei ole olemassa yhtä menetelmää, joka kykenisi ratkaisemaan kaikki ongelmat. Eri ongelmiin täytyy erikseen etsiä parhaat soveltuvat ryhmittelymenetelmät ja soveltaa niitä sopivalla tavalla.

Viitteet

[1] A. ASUNCION, D. N. UCI machine learning repository, 2007.

[2] ANDRITSOS, P. *Scalable Clustering of Categorical Data And Applications*. PhD dissertation, University of Toronto, Department of Computer Science, 9 2004.

[3] ANDRITSOS, P., TSAPARAS, P., MILLER, R. J., AND SEVCIK, K. C. Limbo: Scalable clustering of categorical data. *Advances in Database Technology - EDBT 2004* (2004), 531–532.

[4] ARANGANAYAGI, S., AND THANGAVEL, K. Clustering categorical data based on representatives. In *ICCIT '08: Proceedings of the 2008 Third International Conference on Convergence and Hybrid Information Technology* (Washington, DC, USA, 2008), IEEE Computer Society, pp. 599–604.

[5] BARBARÁ, D., LI, Y., AND COUTO, J. Coolcat: an entropy-based algorithm for categorical clustering. In *CIKM '02: Proceedings of the eleventh international conference on Information and knowledge management* (New York, NY, USA, 2002), ACM, pp. 582–589.

[6] CAI, Z., WANG, D., AND JIANG, L. K-distributions: A new algorithm for clustering categorical data. In *Lecture Notes in Computer Science* (8 2007), vol. 4682, Springer Berlin / Heidelberg, pp. 436–443.

[7] CHEN, K., AND LIU, L. The "best k" for entropy-based categorical data clustering. In *SSDBM'2005: Proceedings of the 17th international conference on Scientific and statistical database management* (Berkeley, CA, US, 2005), Lawrence Berkeley Laboratory, pp. 253–262.

[8] CHUAN CHU, S., CHIH HSIN, Y., AND HUI WANG, M. Genetic distance measure for k-modes algorithm. *International Journal of Innovative Computing, Information and Control 2*, 1 (2006), 33–40.

[9] EVERITT, B. S. *Cluster Analysis*. Arnold, London, UK, 1993.

[10] FISHER, D. H. Knowledge acquisition via incremental conceptual clustering. *Machine Learning 2*, 2 (Sep 1987), 139–172.

[11] FRIENDLY, M. Visualizing categorical data: Data, stories and pictures. In *Proc. SAS User Group Conf.* (1991), pp. 190–200.

[12] FRIENDLY, M. *Visualizing Categorical Data*. SAS Publishing, 2001.

[13] FUNDERLIC, R. E., CHU, M. T., ORLOWSKI, N., SCHLORFF, D., BLEVINS, J., AND AS, D. C. Convergence and other aspects of the k-modes algorithm for clustering categorical data. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.67.8748, 2004. Julkaisematon käsikirjoitus.

[14] GANTI, V., GEHRKE, J., AND RAMAKRISHNAN, R. Cactus—clustering categorical data using summaries. In *KDD '99: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 1999), ACM, pp. 73–83.

[15] GLUCK, MARK A., J. E. C. Information, uncertainty, and the utility of categories. In *Program of the Seventh Annual Conference of the Cognitive Science Society* (Irvine, CA, 1985), Lawrence Erlbaum Associates, pp. 283–287.

[16] GUHA, S., RASTOGI, R., AND SHIM, K. ROCK: A robust clustering algorithm for categorical attributes. *Information Systems 25*, 5 (2000), 345–366.

[17] HAN, J., AND KAMBER, M. *Data Mining: Concepts and Techniques*. Academic Press, London, United Kingdom, 2001.

[18] HE, Z., DENG, S., AND XU, X. X.: Improving k-modes algorithm considering frequencies of attribute values. In *Lecture Notes in Artificial Intelligence* (2005), pp. 157–162.

[19] HE, Z., XU, X., DENG, S., AND DONG, B. K-histograms: An efficient clustering algorithm for categorical dataset. *CoRR abs/cs/0509033* (2005).

[20] HUANG, Z. Extensions to the k-means algorithm for clustering large data sets with categorical values. *Data Min. Knowl. Discov. 2*, 3 (1998), 283–304.

[21] HUANG, Z., AND NG, M. A fuzzy k-modes algorithm for clustering categorical data. *Fuzzy Systems, IEEE Transactions on 7*, 4 (Aug 1999), 446–452.

[22] JAIN, A. K., AND DUBES, R. C. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[23] JAIN, A. K., MURTY, M. N., AND FLYNN, P. J. Data clustering: a review. *ACM Comput. Surv. 31*, 3 (September 1999), 264–323.

[24] KIM, D.-W., LEE, K., LEE, D., AND LEE, K. H. A k-populations algorithm for clustering categorical data. *Pattern Recognition 38*, 7 (2005), 1131 – 1134.

[25] LEI, M., HE, P., AND LI, Z. An improved k-means algorithm for clustering categorical data *. *Journal of Communication and Computer 3*, 8 (2006), 20 – 24.

[26] LI, T., MA, S., AND OGIHARA, M. Entropy-based criterion in categorical clustering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning* (New York, NY, USA, 2004), ACM, p. 68.

[27] MACQUEEN, J. B. Some methods for classification and analysis of multivariate observations. In *Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability* (1967), L. M. L. Cam and J. Neyman, Eds., vol. 1, University of California Press, pp. 281–297.

[28] MARTIN, A., DODDINGTON, G., KAMM, T., ORDOWSKI, M., AND PRZYBOCKI, M. The det curve in assessment of detection task performance. In *Proc. Eurospeech '97* (Rhodes, Greece, 1997), pp. 1895–1898.

[29] RAJEEV, S. G., RASTOGI, R., AND SHIM, K. Rock: A robust clustering algorithm for categorical attributes. In *Information Systems* (1999), pp. 512–521.

[30] RALAMBONDRAINY, H. A conceptual version of the k-means algorithm. *Pattern Recogn. Lett. 16*, 11 (1995), 1147–1157.

[31] SAN, O. M., NAM HUYNH, V., AND NAKAMORI, Y. An alternative extension of the k-means algorithm for clustering categorical data. *International Journal of Applied Mathematics and Computer Science 14* (2004), 241–247.

[32] SIMOVICI, D., SINGLA, N., AND KUPERBERG, M. Metric incremental clustering of nominal data. In *ICDM '04: Proceedings of the Fourth IEEE International Conference on Data Mining* (Washington, DC, USA, 2004), IEEE Computer Society, pp. 523–526.

[33] SPENCE, R. *Information Visualization*. Addison-Wesley, 2001.

[34] SPÄTH, H. *Cluster Analysis Algorithms*. Ellis Horwood Limited, Berkeley, CA, US, 1980.

[35] THEODORIDIS, S., AND KOUTROUMBAS, K. *Pattern Recognition, Third Edition*. Academic Press, February 2006.

[36] XU, R., AND WUNSCH, D. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on 16*, 3 (2005), 645–678.

[37] YANG, Y., GUAN, X., AND YOU, J. Clope: a fast and effective clustering algorithm for transactional data. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining* (New York, NY, USA, 2002), ACM, pp. 682–687.

[38] YI ZHANG, ADA WAI-CHEE, F. C. H. C. P.-A. H. Clustering categorical data. In *Proceedings of the 16th International Conference on Data Engineering* (San Diego, California, USA, 2000), IEEE, p. 305.

[39] YONG LIAO, H., AND NG, M. K. Categorical data clustering with automatic selection of cluster number. In *Fuzzy Inf. Eng.* (2008), vol. 1, pp. 5–25.

[40] ZAKI, M. J., PETERS, M., ASSENT, I., AND SEIDL, T. Clicks: an effective algorithm for mining subspace clusters in categorical datasets. In *KDD '05: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining* (New York, NY, USA, 2005), ACM, pp. 736–742.

[41] ZHANG, P., WANG, X., AND SONG, P. X. Clustering categorical data based on distance vectors. *Journal of the American Statistical Association 101* (March 2006), 355–367.