Tallinna Ülikool
Matemaatika-loodusteaduskond
Informaatika osakond

Anti Soo

Andmekaevandus

Bakalaureusetöö

Juhendaja Jaagup Kippar

Tallinn 2006

Autor: .....”.....”..... 2006.a.

Juhendaja: .....”.....”..... 2006.a.

Osakonna juhataja .....”.....”..... 2006.a.

Sisukord

SISSEJUHATUS

Meie võime genereerida ning koguda andmeid on kasvanud tohutu kiirusega viimaste aastakümnete jooksul.Kui veel 1990ndatel peeti utoopiliseks analüüsitavaks andmemahuks gigabaite, siis tänapäeval hindavad paljud rahvusvahelised korporatsioonid oma andmemahte terabaitides (Liiv, 2005). Selle kõige põhjuseks on näiteks kasvõi toodete triipkoodide kasutamine, paljude äriliste protsesside arvutiseerimine, teaduslikud ja riiklikud transaktsioonid ning ka edasiminek andmekogumisvahendites, alates skaneerimisest kuni internetipoodideni välja. Loogiliselt peabki see maht pidevalt kasvama, sest salvestatakse ju andmete ajalugu. Näitena võib veel tuua interneti, mis on meid ujutanud üle meeletute koguste andmete ja informatsiooniga. Salvestatud andmete plahvatuslik kasv on loonud kriitilise vajaduse uute tehnoloogiate ning automatiseeritud vahendite järele, mis suudaksid intelligentselt muundada suured kogused andmeid meile kasulikuks informatsiooniks ning teadmisteks.

Andmekaevandus on vägagi mitmekülgne, koondades kokku sellised alad nagu andmebaasitehnoloogiad, tehisintellekt, masinõppimine, neurovõrgud, statistika, mustrite äratundmine, teadmuspõhised süsteemid, teadmiste omandamine, informatsiooni ammutamine ning andmete visualiseerimine (Han ja Kamber, 2000).

Kui globaalselt on andmekaevandamine edukalt toimiva suurettevõtte ärijuhtimisprotsessi üks kindel ning oluline osa, siis Eesti mastaabis hakkab antud valdkond alles hoogu koguma. Üheks peamiseks põhjuseks, miks meil varasemalt andmekaevandusele nii vähe tähelepanu on pööratud, võib pidada seda, et kohalikud ettevõtted on maailma mastaabis küllalti väikesed ning andmemahud ei ole senimaani ületanud tavaliste statistikavahendite jõudluse piire. Tänaseks päevaks on aga pangandus- ning kommunikatsioonisektorid ka Eestis andmekaevandust suuresti rakendanud ning see tendents on järjest kasvav tänu elektroonilise andmehulga meeletule arengule. Iga suurettevõtte ärijuhtimine peaks olema rajatud andmetel baseeruvatel teadmistele, mitte enam aga emotsioonidel ning juhtide sisetundele.

Käesoleva bakalaureusetöö peamiseks eesmärgiks on anda teoreetiline ülevaade andmekaevandamise protsessist, tutvustada enamkasutatavaid kaevandusmeetodeid ning tuua mõningad näited ka reaalelulistest kasutusvaldkondadest. Eelkõige on materjal mõeldud neile, kes puutuvad valdkonnaga kokku esmakordselt. Samuti heidame ka põgusa pilgu ühele vabavaralisele andmekaevanduskeskkonnale.

Antud bakalaureusetöö koosneb viiest peatükist, kus esimene peatükk annab üldise ülevaate andmekaevandamise ajaloost ning olemusest. Teine peatükk käsitleb detailsemalt enamkasutatavaid kaevandusmeetodeid. Kolmas ning neljas peatükk kirjeldavad andmekaevanduse rakendusvaldkondi ning ka esinevaid probleeme ja küsimusi. Viies peatükk on sissejuhatuseks ühe vabavaralisele andmekaevanduskeskkonnale.

1.Sissejuhatus andmekaevandamisse

1.1Andmebaasitehnoloogia evolutsioon

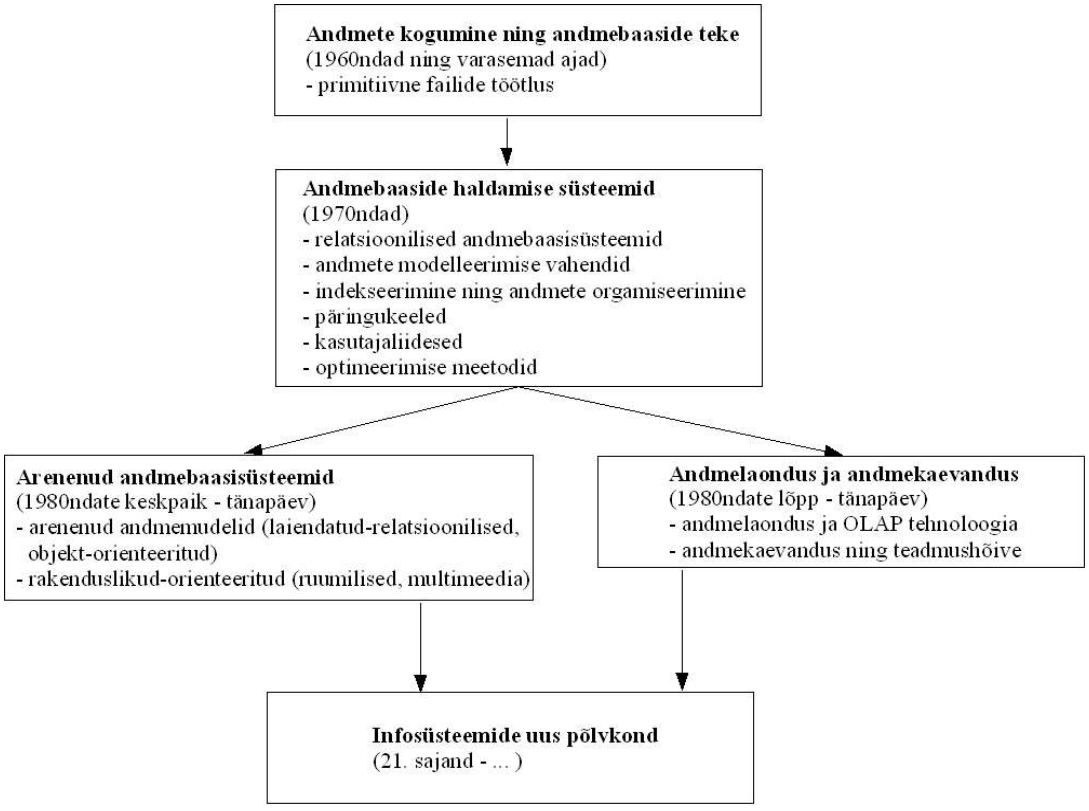
Peamiseks põhjuseks, miks andmekaevandus on tõusnud tänapäeva informatsioonitööstuse huviorbiiti, on ligipääs tohutu suurtele andmehulkadele ning kohene vajadus saada kätte sellistest andmetest kasulikku informatsiooni ning teadmisi (Han; Kamber; 2000). Kättesaadud informatsiooni saab kasutada alates äri juhtimisest, tootmise planeerimisest, turusituatsioonide analüüsimisest kuni elektroonika disaini ning teaduslike uurimustöödeni.

Andmekaevandust võib võtta kui loomulikku infotehnoloogia evolutsiooi tulemust. Evolutsioonilist rada võib jälgida alltoodud joonisel (joonis 1.1.1). Selle järgi võib tuua andmebaaside evolutsioonist välja järgmised etapid: andmete kogumine ning andmebaasi loomine, andmete haldamine (andmete hoiustamine ja andmete otsing, ka andmeoperatsioonid) ning andmete analüüs ja nende andmete mõistmine (andmeaidad ning andmekaevandus). Näiteks varasem andmete kogumine, algsete andmebaaside teke ning andmebaaside loomise mehhanismid olid eeltingimusteks hilisemate efektiivsete andmete salvestamise ning nende kasutamise mehhanismideks, samuti ka päringute ja andmebaasioperatsioonide jaoks. Peale arvukaid andmebaasisüsteeme, mis pakkusid päringuid ning andmeoperatsioone (sisestamine, muutmine, kustutamine jne), muutusid olulilsteks andmete analüüs ning nende andmete mõistmine.

Alates 1960ndatest on andmebaasid ning infotehnoloogia süstemaatiliselt arenenud primitiivsest failide töötlemisest välja keeruliste ja võimsate andmebaasisüsteemideni. 1970ndate aastate andmebaasisüsteemide teadustöö ning arendamine on välja viinud relatsiooniliste andmebaasisüsteemide tekkeni, andmete modelleerimisvahenditeni, indekseerimise ning andmete organiseerimistehnikateni. Lisaks on kasutajatele loodud mugav ning paindlik ligipääs andmetele päringukeelte, päringutöötluste ning heade kasutajaliideste kaudu. Võimsad meetodid nagu on-line transaktsioonide töötlused (*OLAP*), kus päringut vaadeldakse kui lugemisõigusega (*read-only*) transaktsiooni, on oluliselt kaasa aidanud relatsioonilise tehnoloogia evolutsioonile ning laialdasele aksepteerimisele, mis on peamiseks vahendiks suurte andmekoguste efektiivseks hoiustamiseks, kättesaamiseks ning haldamiseks.

1980ndate aastate keskpaiga andmebaasitehnoloogiat võib iseloomustada relatsioonilise tehnoloogia hoogsa adaptsiooniga ning aktiivse uurimustöö ja arendamisega võimsate andmebaaside valdkonnas. Sellel ajal võeti kasutusele niisugused arenenud andmemudelid nagu laiendatud-relatisioonilised, objekt-orienteeritud, objekt-relatsioonilised ning deduktiivsed rakenduslikud andmebaasisüsteemid. Andmete levitamiste, mitmekesisuse ning jagamise probleeme on uuritud üsna põhjalikult. Ebaühtlased andmebaasisüsteemid ning internetipõhised globaalsed infosüsteemid, nagu näiteks *WWW*, on samuti mänginud olulist rolli informatsioonitööstuse arengus.

Arvuti riistvara tehnoloogia stabiilne ning imetlusväärne areng viimase kolme aastakümne jooksul on viinud võimsate ning suure hulga arvutiteni, andmete kogumise varustuseni ning salvestusvahenditeni. Selline tehnoloogia kiire areng annab olulise tõuke andmebaaside ja informatsioonitööstuse arengule ning toob kaasa suure hulga andmebaase, andmebaasioperatsioone, andmete otsinguid ning nüüd ka andmete analüüsi.



Joonis 1.1.1. Andmebaasisüsteemide evolutsioon

Kaasajal saab andmeid salvestada mitmetes eritüüpi andmebaasides. Üks mõnede aastate eest välja tulnud süsteem on näiteks andmeladu – heterogeensetets allikatest pärit andmete hoidla, organiseeritud ühendatud skeemi alla ühte kohta, et hõlbustada strateegiliste otsuste tegemist. Andmelaonduse tehnoloogia sisaldab endas veel andmete puhastamist, andmete integratsiooni ning on-line analüütilist töötlemist ehk *OLAP* tehnoloogiat.

Andmete üleküllus koos vajadusega võimsate andmeanalüüsimisvahendite järele võib kirjeldada olukorrana: "andmerikas, kuid infovaene" (Han; Kamber; 2000). Kiiresti kasvav meeletu andmete kogus, mis kogutakse ning hoiustatakse suurtes ja arvukates andmebaasides, on ületanud juba ammu inimese võime neid andmeid hõlmata ilma võimsate vahenditeta. Tulemuseks on meil meeletud andmekogused, mida väga harva kasutatakse. Selle tagajärjel tehakse tihtipeale strateegilisi otsuseid lähtudes inimese intuitsioonist, mitte aga väärtuslikest teadmistest, mis võiksid tulla andmete töötlemiseset. Siinkohal tulevadki mängu andmekaevandamise vahendid, mis teostavad andmete analüüsi ning võivad avastada tähtsaid andmemustreid, toetades niimoodi äristrateegiaid, teadmiste aluseid ning ka teaduslikke ja meditsiinilisi uuringuid. Järjest suurenev lünk andmete ning informatsiooni vahel nõuab süstemaatilist andmekaevanduse vahendite arendamist, et muuta suur hunnik andmeid kasulikeks teadmisteks.

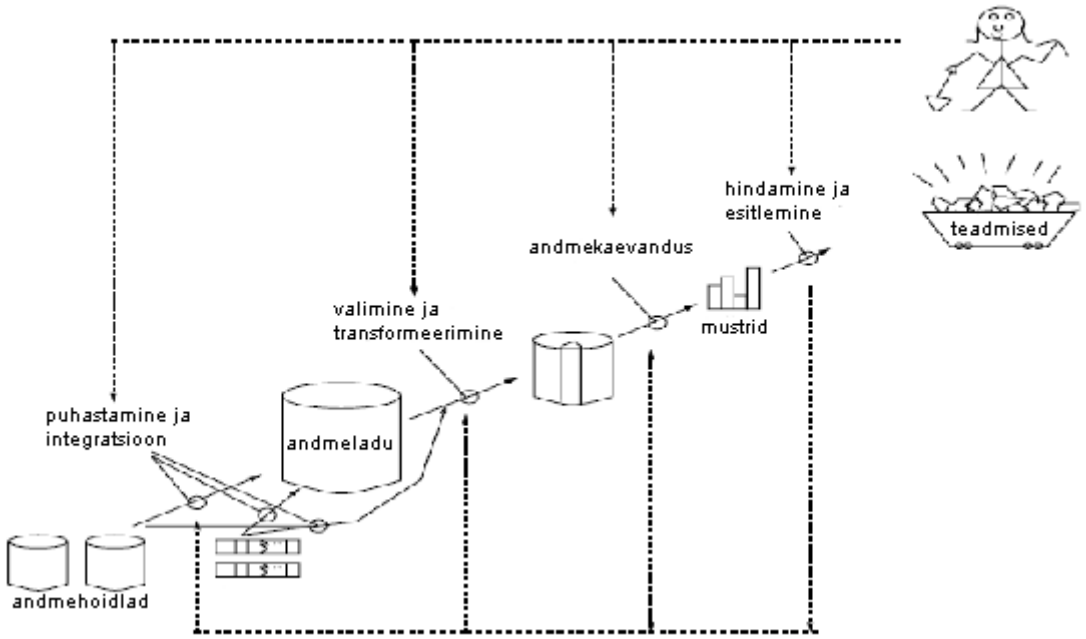
1.2Andmekaevanduse olemus

Andmekaevandamine ei erine põhimõtete poolest traditsioonilisest kaevandamisest – teatud oskusteabe ning vahendite abil püütakse pinnasest kätte saada väärtuslikke maavarasid (Liiv, 2005). Sarnaselt traditsioonilisele kaevandamisele toimub ka andmekaevandamises kindlate vahendite ning oskusteabega leida suurtest andmehulkadest kasulikku ja ootamatut informatsiooni, eesmärgiga saadud teadmisi edukalt ära kasutada. "Andmekaevandusele" võibolla sobivam ja täpsem nimetus oleks "Teadmiste kaevandamine andmetest", kuid see on pisut pikk ja ebamugav. Samuti lühike termin "Teadmiste kaevandamine" ei peegeldaks oma olemust – kaevandamine suurtest andmehulkadest. Siiski, "kaevandamine" on piisavalt ilmekas termin iseloomustamaks protsessi, mis leiab ülesse väikese koguse "kulda" suurest hulgast toormaterjalist. Seega muutus populaarseks väljendiks sõna andmekaevandus.

Andmekaevanduse kohta võib leida mitmesuguseid definitsioone:

- Lihtsalt öeldes viitab andmekaevandus teadmiste väljaotsimisele või "kaevandamisele" suurtest andmekogustest (Han,Kamber, 2000)
- Andmekaevandus on etapp teadmushõives, mille üldine eesmärk on leida andmetest paikapidavaid, uudseid, potensiaalselt kasulikke ning mõistetavaid mustreid (Fayyad,Piatetsky-Shapiro,Smyth, 1997)
- Mahukate andmete analüüs leidmaks uusi seaduspärasusi ja ootamatuid seoseid ning summeerida andmeid sellisel uudsel viisil, et need oleksid samaaegselt arusaadavad ning kasulikud (Hand,Mannila,Smyth, 2001)
- Mustrite avastamise protsess, mis peab olema automaatne või poolautomaatne. Leitud mustrite sisu peab juhatama teed mõne teatud eeliseni, tüüpiliselt ärilise konkurentsieeliseni (Written,Frank, 2000)
- Andmekaevandus on riistvarast, tarkvarast, oskustööjõust ning andmetest koosnevast võimalusest ära tunda tundmatuid kuid potensiaalselt kasulikke seoseid. See toetab andmete transformeerimist informatsiooniks, teadmisteks ning tarkuseks. Tsükkel, mis peaks toimima igas organisatsioonis (Labovitz, 2003).

Tihtipeale kiputakse pidama termineid "andmekaevandus" ning "teadmushõive andmebaasidest" sünonüümideks. Tegelikkuses vaadeldakse aga andmekaevandust teadmushõive ühe olulise etapina (joonis 1.2.1). Teadmushõivest teeme juttu pisut allpool.



Joonis 1.2.1. Andmekaevandus teadmushõive protsessina

Andmekaevanduses võib välja tuua kolm üldisemat eesmärki:

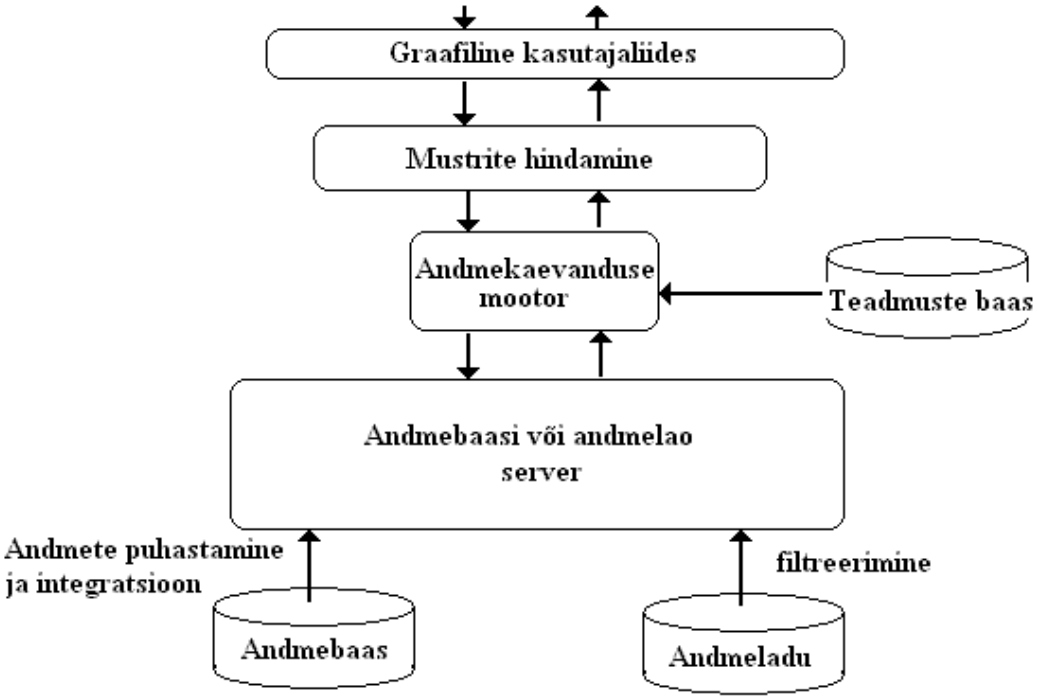
- kirjeldamine – keskendub andmete selgitamisele, mis võimaldaks andmekaevanduse analüütikutel nende andmete sisse näha ning neid interpreteerida
- prognoosimine – olemasolevate tunnuste põhjal tundmatute või tulevikuväärtuste leidmise ennustusmudelite loomine
- juhtimata ning järelvalveta avastamine – tegemist on erinevalt eelmistest protseduuri tüübiga, mis jaotab veel tehnikad juhitud tegevusteks (olemas küsimus ning püütakse leida sellele vastust) ning juhtimata, järelvalveta ning juhuslikeks tegevusteks (andmehulgale rakendatakse erinevaid tehnikaid, lootes leida midagi uut ning huvitavat)

Andmekaevandamise etapp toimib vastastikku kasutajaga või siis teadmuste baasiga. Huvitavad mustrid esitatakse kasutajale või salvestatakse uue teadmusena teadmuste baasis. Siinkohal tuletaks meelde, et andmekaevandamine on vaid üks etapp terves protsessis, ehkki elutähtis, kuna see avastab peidetud mustreid hinnangu andmiseks. Hoolimata sellest, et andmekaevandus on vaid osake teadmushõiveprotsessist, on termin "andmekaevandus" muutunud populaaremaks, kui pikk termin "teadmushõive andmebaasidest". Vaatleme andmekaevandust kui protsessi, mis avastab huvitavaid teadmisi suurtest andmekogustest, mis on salvestatud kas andmebaasidesse, andmeladudesse või teistesse informatsioonihoidlatesse.

Joonise 1.2.2 järgi võib välja tuua tüüpilise andmekaevandussüsteemi komponendid:

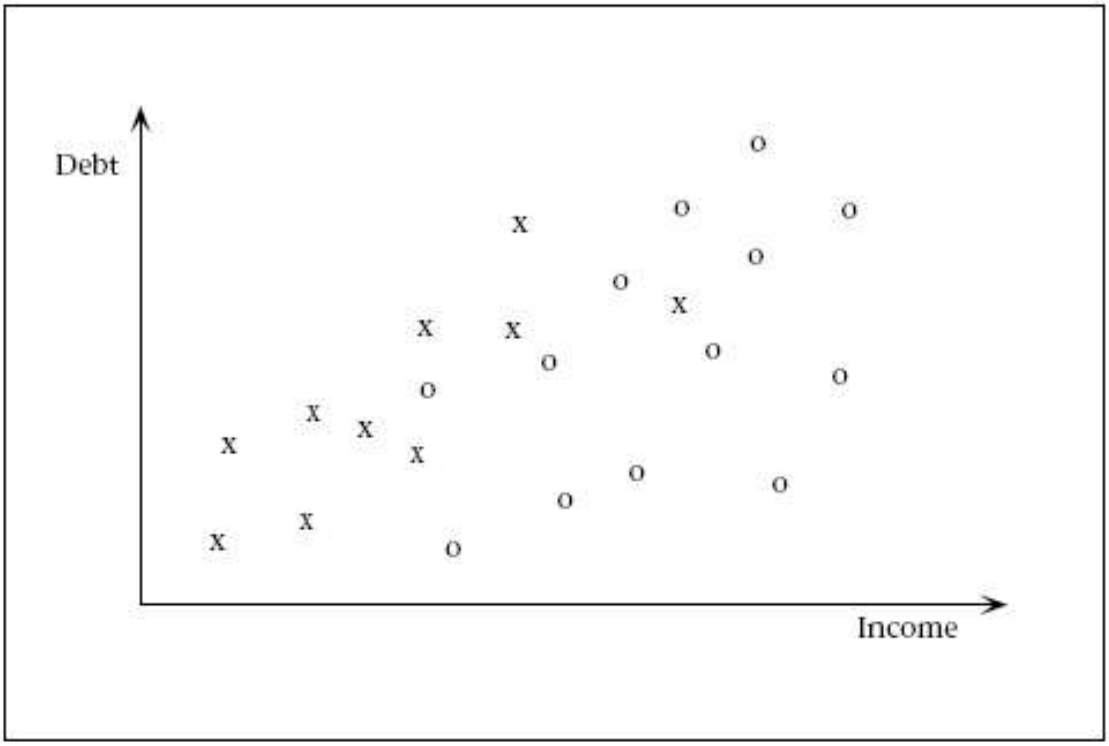
- **Andmebaas, andmeladu või muu informatsioonihoidla**. Selleks on üks või mitu andmebaasi, andmeladu või mõni muu informatsioonihoidla. Andmete puhastamist ning andmete integratsioonitehnikaid kasutataksegi üldjuhul siin.
- **Andmebaasi või andmelao server**. Andmebaasi või andmelao server peab hankima kasutaja poolt nõutud olulisi andmeid.
- **Teadmuste baas**. Selleks on teadmuste baas, mida kasutatakse otsingu juhtimiseks või siis leitud mustrite huvitavuse hindamiseks. Seesugune teadmus võib sisaldada ideede hierarhiaid, et organiseerida atribuute või atribuutide väärtusi erinevatesse tasemetesse. Samamoodi võib sinna teadmustesse arvata ka kasutaja arusaamad ja uskumused mustrite huvitatavusse ja uudsusesse.

- **Andmekaevanduse mootor**. See on üks põhiline osa andmekaevandamise süsteemist ning üldjuhul koosneb järgmistest funktsionaalsetest moodulitest nagu iseloomustamine, assotsatsioonide analüüs, kirjeldamine, evolutsiooni ja kõrvalekalde analüüs.
- **Mustrite hindamise moodul**. See komponent üldiselt kasutab huvitatavuse meetmeid ning suhtleb andmekaevandusmoodulitega, et leida huvitavaid mustreid. See pääseb ligi huvitatavuse künnistele, mis on salvestatud teadmuste baasis. Alternatiivselt võib see moodul olla ka integreeritud kaevandamise mooduliga, olenevalt sellest, millist andmekaevandamise meetodit kasutatakse. Efektiivse andmekaevanduse seisukohalt on oluline lükata mustrite huvitatavuse hinnang võimalikult sügavale kaevandusprotsessi sisse, et piirata oluliselt otsingut ning leida ainult huvitavaid mustreid.
- **Graafiline kasutajaliides**. Antud moodul suhtleb kasutajate ning andmekaevandussüsteemi vahel, lastes kasutajal suheldal süsteemiga ning määrata andmekaevanduspäringuid ning ülesandeid. Samuti pakub ta informatsiooni, et aidata fokusseeruda otsingule ning teostada vahetulemuste baasil uurivat andmekaevandust. Ühesõnaga see komponent laseb kasutajal uurida andmebaaside ja andmeladude skeeme ning andmestruktuure, hinnata kaevandatud mustreid ning visualiseerida mustreid erinevates vormides.



Joonis 1.2.2. Tüüpilise andmekaevandussüsteemi arhitektuur.

(Fayyad,Piatetsky-Shapiro,Smyth 1997) järgi saab vaadelda enamust (kui mitte lausa kõiki) andmekaevandusmeetoteid kui mõne elementaartehnika ja põhimõtte laiendustena ning hübriididena. Et seda pisut lihtsamal kujul arusaavamaks teha, võib vaadelda järgnevat joonist.



Joonis 1.2.3. Lihtne kahe klassiga andmehulk.

Joonisel 1.2.3 on kujutatud lihtne kahedimensiooniline 23 punktist koosnev andmehulk. Iga punkt joonisel kujutab inimest, kes on saanud mingil ajahetkel pangast laenu. Horisontaalne telg kujutab endast inimese sissetulekut ning vertikaalne telg laenu suurust. Andmed on kujutatud kahe klassina, kus X tähistab inimesi, kes on jätnud oma laenu maksmata ning O tähistab inimesi, kes maksavad oma laenu eeskujulikult. Seesugune lihne andmehulk võib esitleda ajaloolist andmehulka, mis sisaldab endast vajalikku informatsiooni pangale, et laene väljastada. Üldjuhul on tegelikkuses tegemist oluliselt suurema arvu dimensioonide ning andmepunktidega.

Andmelao perspektiivist võib andmekaevandust vaadata ka kui arenenumat astet *on-line* analüütilisest andmetöötlusest (*OLAP*). Siiski, andmekaevandus läheb oluliselt kaugemale andmeladude summeerivast analüütilisest töötlusest, hõlmates rohkem arenenumaid ja keerukamaid tehnikaid andmete mõistmiseks.

Andmekaevandus sisaldab endas kombinatsiooni mitmetest tehnikatest nagu andmebaasi tehnoloogia, statistika, masinõppimine, mustrite äratundmine, andmete visualiseerimine, informatsiooni kogumine, piltide ja signaalide töötlemine ning ruumiline andmeanalüüs. Andmekaevandamise tulemusena saab huvitavaid teadmisi, regulaarsusi või kõrgtaseme informatsiooni andmebaasidest vaadelda erinevate nurkade alt. Avastatud teadmisi võib rakendada otsuste tegemisel, protsesside juhtimisel, informatsiooni haldamisel jne. Seetõttu peetakse andmekaevandust üheks oluliseks osaks andmebaasisüsteemide juures ning ka üheks paljulubavaks andmebaasirakenduseks informatsioonitööstuses.

Siinkohal tuleks mainida, et andmekaevandamisega ei tegele pelgalt arvutid ning algoritmid. Andmekaevanduses on siiski esmatähtis roll inimesel ning eelmainitud on vaid abivahendid, kuna inimene üksi ei ole võimeline töötlema tohutuid andmehulkasid. Kuna arvutitel puudub loovus, siis andmekaevandusprotsesside automatiseerimine ei pruugi anda alati soovitud tulemusi.

1.3Andmekaevanduse ja statistika erinevus

Statistikat defineeritakse kui meetodit andmete kogumiseks, esitlemiseks, kokkuvõtmiseks, hüpoteeside testimiseks ning järelduste tegemiseks, kasutades induktiivseid ja deduktiivseid arutluskäike. Andmekaevandus on aga seevastu huvitavate struktuuride identifitseerimine olemasolevatest andmetest ning andmete vaheliste seoste ja mustrite leidmine kasutades induktiivseid arutluskäike ning teisi tehisintellekti tehnikaid (Fayyad,Grinstein,Wierse, 2002).

Digitaalsete andmehoidlate suurus on kahekordistunud iga üheksa kuu järel vähemalt viimasel aastakümnendil, mis on kaks korda kiirem, kui Moore'i seadus arvutusvõimsuse kohta samas perioodis (A.Burney,Manzoor,F.Burney, 2003). See on ka üks andmekaevanduse osatähtsuse tõusu põhjustest. Kui statistikute jaoks on juba üsnagi suured andmehulgad 1000 kirjet, siis andmekaevanduse puhul räägitakse lausa miljonitest või isegi mirjarditest kirjetest. Andmekaevandus tekkis, kuna hakati aru saama, et traditsioonilised statistikale orienteeritud ostsustusmeetodid ei tule enam toime suurte andmebaaside ja andmeladudega.
Põhiliseks erinevuseks andmekaevandamise ja statistika vahel toimimise loogikast lähtudes on see, et formaalne järeldav statistika on juhitud oletustest – formaliseeritakse hüpotees ning kontrollitakse seda teatud etteantud olulisuse nivool. Andmekaevandust pigem juhivad andmed – mustrid ja hüpoteesid genereeritakse andmetest automaatselt.

(Hand, 1998) järgi võib pidada suurimateks erinevusteks statistika ja andmekaevanduse vahel järgmist:

- **andmetabelite suurus** – statistikute jaoks on suur hulk juba mõnisada tunnust ning tuhande tunnuse analüüsimine juba suur katsumus. Andmekaevandus aga peab tome tulema juba miljonite ning miljardite tunnustega
- **vigased andmed** – teema on seotud eelmise erinevusega. Näiteks 0.1% puuduvaid või vigaseid andmeid ei avalda tavapärases statistikas just eriti suurt mõju. Seevastu suurte andmehulkade korral ei saa seda enam ignoreerida. Näiteks miljardi kirje puhul oleks see viga juba miljon kirjet.

- **mittestatsionaarsus** – kuna andmebaasid suurenevad pidevalt, siis ei ole tihtipeale aega hakata andmeid koguma ja analüüsima. Teinekord tuleb isegi analüüsida andmeid reaalajas. Probleemi toob esile informatsiooni vajamise kiirus, kus eelmise kuu tulemusi täna kätte saada on võibolla juba hilja.
- **mittearvulised väärtused –** klassikaline statistika tegeleb vaid numbrilise analüüsiga ning analüütiku ülesanne ongi informatsiooni numbriteks kodeerimine. Andmekaevanduse puhul tegeletakse aga kõikvõimalike andmetega, sealhulgas piltide, teksti või ka geograafiliste andmetega.

(Zamar, 2003) toob välja aga järgmised erinevused andmekaevanduse ja statistika vahel:

- **andmete hulk** – statistika puhul väike või keskmine, ehk siis kümned või sajad tunnused. Andmekaevanduse puhul aga andmete hulk suur – miljonid või miljardid.
- **andmetüübid** – statistika puhul kogutakse andmeid või pannakse kokku et katsetada mudelit või vastust kindlale küsimusele (*first hand data*). Andmekaevanduses aga kogutakse andmeid elektrooniliselt ning hoitakse andmehoidlates võimalikeks kasutamisteks tulevikus.

- **andmete töötlemine** – statistikas töödeldakse andmeid inimeste poolt arvutite abiga. Andmekaevanduses töötlevad andmeid arvutid ja algoritmid inimeste abiga.
- **tüüpilised ülesanded** – statistikas on tavalisteks ülesanneteks mudelite sobitamine, mudelite testimine, usalduse ning ennustuse intervall. Andmekaevanduses aga mustrite otsimine ja identifitseerimine, klassifitseerimine ning grupeerimine.
- **uurimustöö eesmärgid** – Statistika puhul on eesmärkideks arendada välja paremad statistilised protseduurid, õppida statistiliste ja matemaatiliste meetodite omadusi. Andmekaevanduses arendada paremaid/kiiremaid algoritme andmekaevandusülessannete jaoks, uurida empiiriliste andmekaevandusalgoritmide jõudlusi.

Statistikaga on tegeletud kogu 20nda sajandi jooksul, andmekaevandus aga tekkis alles kaheksakümndendate alguses ning selle eesmärgiks oli esialgu tegeleda statistikale sarnaste probleemidega (A.Burney,Manzoor,F.Burney, 2003).

Statistikal ning andmekaevandusel on üsnagi palju ühiseid jooni ning statistikud on juba alustanud diskussioone andmekaevandamise temaatikal, et proovida kasu lõigata väga lähedase valdkonna populaarsuse kasvust ning otsustada, kas hakata artiklites avaldama survet, et andmekaevandamine kuulutatakse statistika alamosaks (Liiv, 2005).

Andmekaevandus ning statistika on järjest rohkem kokku kasvamas lähitulevikus, kuna andmekaevandusest ei saa teadmushõivet ilma statistiliste vahenditeta ning statistika ei tule enam toime suurte ja keerukate andmehulkadega ilma andmekaevanduslike vahenditeta.

1.4Milliste andmehoidlatega on võimalik andmekaevandust teostada

Siinses peatükis vaatleme mõnda andmehoidlat, mille puhul saab teostada andmekaevandust. Põhimõtteliselt peaks andmekaevandust saama rakendada igasugusel andmehoidlal, sealhulgas relatsioonilistel andmebaasidel, andmeladudel, transaktsioonilistel andmebaasidel (mis võivad samaaegselt olla ka relatsioonilised), keerukamatel andmebaasidel, lihtsatel failidel ning ka veebis. Keerukamad andmebaasid sisaldavad endas objekt-orienteeritud ja objekt-relatsioonilisi andmebaase, aga ka spetsiifilisi rakenduslik-orienteeritud andmebaase nagu ruumilisi andmebaase, teksti andmebaase ning multimeedia andmebaase. Väljakutsed ja tehnikad võivad olla nende andmehoidlate puhul erinevad.

Relatsioonilised andmebaasid.

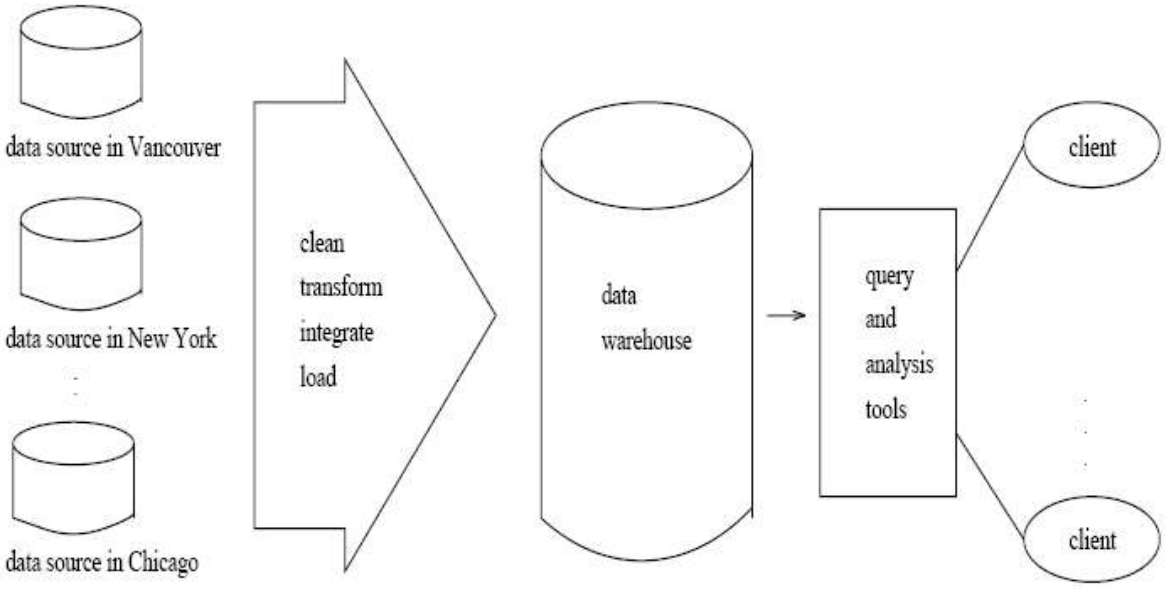
Andmebaasisüsteem koosneb hulgast vastastikkuses seoses olevatest andmetest ning teatud tarkvarast, et hallata ja pääseda ligi andmetele. Relatsiooniline andmebaas koosneb mitmetest tabelitest, millele kõigile on antud unikaalne nimi. Iga tabel koosneb hulgast atribuutidest ehk siis veergudest ning tavaliselt hoiustatakse selles suurel hulgal kirjed (ridu). Relatsioonilises andmebaasis on igal kirjel unikaalne identifikaator ehk *unique key*.

Relatsioonilistele andmetele saab ligi kasutades andmebaasi päringuid, mis on kirjutatud relatsioonilises päringukeeles nagu näiteks SQL. Samuti on võimalik andmete ligi pääseda ka graafilise kasutajaliidese abil. Päring võimaldab välja tuua teatud kasutaja poolt seatud tingimustele vastava andmehulga.

Relatsiooniliste andmebaaside puhul on võimalik andmekaevanduse käigus otsida trende või andmemustreid. Näiteks võivad andmekaevandussüsteemid analüüsida uue kliendi krediidiriski nende vanuse, sissetuleku ning varasema krediidiinformatsiooni järgi. Andmekaevandussüsteemid suudavad ka leida kõikvõimalikke kõrvalekaldeid - näiteks kui ühe toote müüginumbrid on oluliselt erinevad eelmise perioodi müügi-numbritega. Selliseid kõrvalekaldeid saab hiljem uurida ning välja selgitada tekkinud erinevuse põhjused, milleks võib olla pakendi muutus või märgatav hinnatõus. Relatsioonilised andmebaasid on andmekaevanduse jaoks ühed populaarsemaid ning informatsioonirikkamaid andmehoidlaid.

Andmeladu.

Andmelao puhul pärineb informatsioon tavaliselt mitmest erinevast allikast ning see on siis talletatud ühtse ning ühtlustatud skeemi järgi ning tavaliselt asub üks andmeladu ühes kindlas paigas ehk ühes serveris. Enne, kui andmed jõuavad andmelattu, toimub andmete puhastamine, transformatsioon, integratsioon ning perioodiline andmete värskendamine.



Joonis 1.4.1. Tüüpilise andmelao arhitektuur.

Et hõlbustada otsuste tegemist, on andmed andmeladudes organiseeritud peamiste subjektide ümber nagu näiteks klient, toode, tarnija või tegevus. Andmed salvestatakse selliselt, et pakkuda ajaloolist ülevaadet (näiteks viimase viie või kümne aasta andmeid) ning on tavaliselt summeeritud. Selle asemel, et salvestada iga müügitehingu detaile, hoitakse andmelaos hoopis iga kaupluse kohta ühe toote tüübi tehnigute summat.

Andmeladu tavaliselt modelleeritakse mitmedimensioonilise andmebaasistruktuurina, kus iga dimensioon vastab teatud atribuudile või atribuutide hulgale ning iga element sisaldab endas koondatud väärtust nagu näiteks loendatud väärtusi või väärtuste summat. Tihtipeale on tegelik andmelao füüsiline struktuur kas relatsiooniline andmete kogu või siis multidimensionaalne andmekuup. Viimane võimaldab andmeid vaadelda mitmedimensiooniliselt ning kiiret juurdepääsu summeeritud andmetele.

Andmelaosüsteemid on väga hästi sobivad OLAP operatsioonideks ning andmetöötlusteks tänu andmete mitmedimensioonilistele vaadetele ja summeeritud andmete eelarvutamisele.

Transaktsioonilised andmebaasid.

Üldsõnaliselt kujutavad transaktsioonilised andmebaasid endast faili, kus iga kirje kujutab endast mingisugust transaktsiooni ehk toimingut. Transaktsioonil on üljuhul unikaalne toimingu number ning hulk lisaväärtusi, mis siis kujutavad vastavat toimingut (näiteks poes ostetud tooted). Transaktsioonilises andmebaasis võivad veel olla lisaks nende toimingutega seotud tabelid, mis sisaldavad teisi andmeid peale konkreetse ostu. Nendeks andmeteks võivad olla näiteks toimingu kuupäev, kliendi identifikaator, müüja identifikaator ja palju muud.

Selline andmebaas on hea näiteks selleks, et andmekaevanduse abil leida, millised tooted müüvad koos hästi. Seesugused "ostukorviandmed" aitavad paigutada tooteid strateegiliselt ümber, et tõsta müüginumbreid. Kui teada, et tavaliselt ostetakse printer koos arvutiga, siis võib paigaldada kalli printeri odava arvuti kõrvale, lootes niimoodi müüa rohkem kalleid printereid. Tavaline andmepäring ei suuda ülaltoodud informatsiooni anda, kuid andmekaevandussüsteemid suudavad leida transaktsiooniliste andmete puhul seesuguseid tooted, mida sagedamini koos müüakse.

Keerukamad andmebaasisüsteemid ning andmebaasirakendused.

Relatsioonilisi andmebaase kasutatakse laialdaselt äriliste rakenduste puhul. Seoses keerukamate andmebaaside tekkimisega ning andmebaasirakenduste võimsa arenguga, on andmekaevandusel uusi ja keerukamaid väljakutseid.

Uued andmebaasirakendused suudavad käitleda ruumilisi andmeid (kaarte), projekteerimisandmeid (nagu ehitiste arhitektuur, süsteemide komponendid), hüperteksti, multimeediat (nagu teksti, pilte, videot ja ka heli), ajalisi andmeid ning ka internetis paiknevat kõikvõimalikke andmeid. Seesugused rakendused vajavad efektiivseid andmestruktuure ning skaleeritavaid meetodeid, et tulla toime keerukate objektide struktuuridega, kirjete varieeruva pikkusega, poolikult või üldse struktureerimata andmetega, teksti ning multimeediaga.

Et täita ülaltoodud nõudeid, ongi loodud keerukad andmebaasid ning spetsiifilised rakendustele-orienteeritud andmebaasisüsteemid. Nendeks on objekt-orienteeritud ja objekt-relatsioonilised andmebaasid, ruumilised andmebaasisüsteemid, ajutised ning perioodilised andmebaasisüsteemid, teksti ning multimeedia aga ka internetipõhised globaased infosüsteemid.

1.5Teadmushõive andmebaasist

Andmekaevandus ning teadmushõive andmebaasidest ei ole sünonüümid. Teadmushõive andmebaasidest kujutab endast tervet protseduuride jada, mida on tarvis teha selleks, et andmetest peidetud teadmisi kätte saada. Seega andmekaevandamine on vaid üks etapp teadmushõivest andmebaasidest (Liiv, 2005).

Teadmushõive protsessiks on koondada madala-astme andmed (mis on üldjuhul väga mahukad, et neist kergesti ülevaadet saada) teistesse, kompaktsematesse vormidesse (näiteks lühike raport) või kasulikumale kujule (näteks ennustusmudel tulevikujuhtude tarvis). Protsessi tuumaks on spetsiifilised andmekaevandusmeetodid mustrite avastamiseks ning eraldamiseks. (Fayyad,Piatetsky-Shapiro,Smyth, 1996).

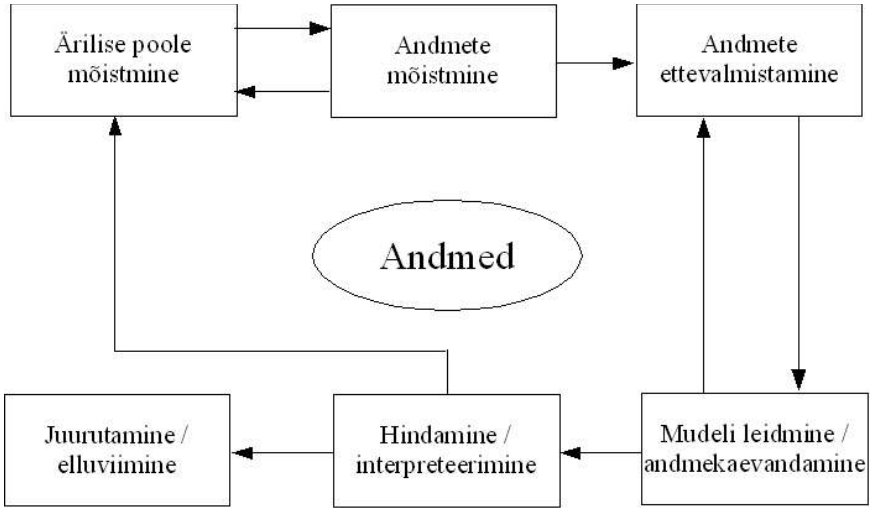
Teadmushõive olemuseks on samm-sammuline protsess leidmaks huvitavaid mustreid suurtest andmekogumitest. Andmekaevandus on üks samm selles protsessis (Imberman, 2001).

Nagu teadmushõive definitsioonidest näha ning nagu ka eespool mainitud, on andmekaevandamine ning teadmushõive andmebaasidest kaks erinevat asja, hoolimata sellest, et nende definitsioonid on küllaltki sarnased. Kuna aga andmekaevandamist ei saa läbi viia ilma eelnevate protseduurideta ning pole ka temast hilisemat kasu ilma hilisema saadud teadmiste rakendamiseta, siis lihtsustamise mõttes loetakse neid tihti teadlikult sünonüümideks.

(Liiv, 2005) toob oma artiklis välja teadmushõive andmebaasidest järgmise iteratiivse ning interaktiivse protsessina:

- **Valdkonnaga tutvumine** ning piisavad eelnevad teadmised, võimaldamaks protsessi eesmärke näha tellija vaatepunktist.
- **Andmete valik**, millest omakorda selekteeritakse sobivad atribuudid ning vajadusel ka alamhulk kirjeid.
- **Andmete puhastamine ning eeltöötlus** – võimaluse korral eemaldatakse müra, pannakse paika strateegia vigaste ja puuduvate andmetega ringikäimiseks, silutakse episoodilisi andmeid.
- **Andmete lihtsustamine ning nendele õige kuju andmine**.
- **Esimese etapi eesmärgid seotakse kindla andmekaevandamise tehnikaga** (näiteks summeerimine, klassifitseerimine, regressioonianalüüs, klasterdamine).
- **Avastav analüüsimine, andmekaevandamise algoritmide ja meetodi valik** mustrite leidmiseks.
- **Andmekaevandamine** – valitud meetodi ning konkreetse algoritmi rakendamine.
- **Leitud mustrite ja vihjete interpreteerimine**, võimalik tagasipöördumine kõigi esimese seitsme etapi juurde – selle etapi lõpuks võidakse proovida tulemust ka visualiseerida või välja pakkuda konkreetne mudel.
- **Leitud teadmistele vastav käitumine** – mudelite integreerimine asutuse süsteemidesse automatiseeritult, lihtne dokumenteerimine ja aruandlus või mudeli rakendamine turunduses või asutuse strateegia kujundamisel.

Konkureerivaid protsessimudeleid võib aga olla veelgi, kuid tekkis vajadus ühtsustatud protsessimudeli järele. Kolme ettevõtte (NCR, SPSS, DaimlerChrysler) koostöö tulemusena sündis uus tegevusalast sõltumatu standardiseeritud protsess andmekaevandamiseks *CRoss Industry Standard Process for Data Mining* ehk CRISP-DM. Erinevuseks varasemate protsessimudelitega oli selgelt äriliste huvide kaitsmine. Äriettevõtte puhul Iga projekt peaks algama ärilisest vajadusest ning lõppematulemuste rakendamisega konkurentsieelise saavutamise eesmärgil.



Joonis 1.5.1. CRISP-DM mudel.

(Liiv, 2005) järgi oleks äsjakirjeldatud protsessimudel järgmine:

- **ärilise poole mõistmine** – esimene etapp keskendub projekti ärilistele eesmärkidele ja nõuetele, püüab formuleerida selle teadmise andmekaevanduse probleemipüstitusena ning pakkuda välja esialgse plaani eesmärkide täitmiseks
- **andmete mõistmine** – etapp algab andmete kogumisega, sisaldab tegevusi nende struktuuri ja sisuga tutvumiseks ning kvaliteediprobleemide tuvastamiseks. Lisaks saadakse selles etapis juba esimesi vihjeid andmete kohta ning moodustatakse väljavõtteid, millest võiks hüpoteese genereeruda kõge edukamalt
- **andmete ettevalmistus** – kolmas etapp sisaldab endas kõiki vajaminevaid tegevusi algsetest allikatest lõpliku andmetabeli moodustamiseks. Lõplikuks kutsutakse andmetabelit siis, kui seda on sobiv ette sööta kaevandamisvahendile. Andmete ettevalmistuseks vajaminevaid tegevusi sooritatakse suure tõenäosusega korduvalt ning ilma kindla järjekorrata. Selliste tegevuste hulgas on näiteks tabelite, kirjete ning atribuutide valikud, samuti kõikvõimalikud puhastamised ja üldkujule transformeerimised
- **andmekaevandamine** – valitakse sobivaid tehnikaid ning rakendadakse neid andmetele. Tüüpiliselt on olemas mitu erinevat lähenemist samale probleemile, lisaks eeldavad mõned tehnikad andmetelt teatud kuju, mistõttu andmete ettevalmistamise juurde tagasipöördumine ei ole siin etapis harv juhus

- **hindamine/interpreteerimine** – selleks hetkeks on juba välja töötatud mudel või mitmeid mudeleid, mis näivad olevat väärtuslikud andmeanalüüsi seisukohalt. Enne elluviimist on tähtis, et mudel käidaks korralikult ka äriliste eesmärkide ning nõudmiste mõttes taas läbi kontrollimaks, kas mõnda eeldust või nõuet ei ole unustatud. Peale hindamise ning kogu senise protsessi ülevaatamise pannakse täpselt paika järmised sammud
- **juurutamine/elluviimine** – mudeli loomisega üldjuhul projekt ei lõpe. Isegi kui mudeli eesmärgiks on näiteks andmetest ülevaate saamine, tuleb saadud teadmine korrastamise ja struktureerimise abil viia sellisele kujule ning presenteerida taoliselt, et klientidel oleks sellest kasu. Tihti tuleb ka saadud mudel integreerida olemasolevatesse otsuste vastuvõtmise protsessidesse. Näiteks teatud objekte (ettevõtte kliente, tooteid) mõnest kindlast aspektist hindav mudel tuleb realiseerida korduvate ja regulaarsete arvutustöödena turunduse andmebaasides. Seega olenevalt nõudmistest võib kogu projekti väljund olla lihtsast tulemuste aruandest kuni keeruka korduva andmekaevandamisprotseduuri implementeerimiseni kogu ettevõttes. Tihti on antud etapi läbiviivaks pooleks juba töö tellinud klient, mitte täitev andmeanalüütik. Isegi kui andmeanalüütik ise ei tegele juurutamisega, peab ta siiski kliendile juba ette täpselt määratlema kõik vajalikud sammud mudelite elluviimiseks.

Eelpool toodud andmehõive protsessimudelitest aga on välja jäänud etapp, milleks on arendamine, jälgimine ning hooldamine. Selleks võib olla tehniline hooldamine, mille eesmärgiks on süsteemide muutumisel vigadele kiiresti jälile saamine, kuid olulisem on siiski ärilise keskkonna jälgimine ning arendamine, kuna see on pidevas muutuses – konkurendid võivad välja tulla uute toodetega ning elukvaliteet muutub pidevalt. See võib endaga kaasa tuua klientide käitumise muutuse ning varasemad koostatud mudelid ei pruugi enam toimida. Seetõttu võib kogu teadmushõive protsess muutuda ilma pideva rahastuseta üsnagi riskantseks ning mõttetuks.

2.Andmekaevandusmeetodid

Andmekaevanduses esineb üsnagi palju erinevaid ülesandeid ja küsimusi, mida tuleb hakata lahendama või siis uurima. Igal ülesandel on tavaliselt oma eeldused ja nõudmised ning lahenduste tulemused on tihtipeale erinevad. Nende lahendamiseks on mitmeid algoritme ja tehnikaid.

Siinses peatükis üritame anda ülevaate enimkasutatud andmekaevandusmeetoditest ning nende töötamispõhimõtetest.

2.1Klassifitseerimine

Klassifitseerimine: andmed klassifitseeritakse ühte või mitmesse eelnevalt defineeritud klassi (Fayyad,Piatetsky-Shapiro,Smyth, 1996).

Klassifitseerimine on protsess leidmaks mudelite või funktsioonide hulka, mis kirjeldaks ja eristaks andmete klasse ning mille eesmärgiks on kasutada saadud mudeleid, et ennustada objektide klasse, mille klassi nimetus puudub (Han, Kamber, 2000).

Tõenäoliselt on klassifitseerimine üks kõige rohkem uuritud andmekaevandusülesanne. Nagu ülalpool väljatoodud definitsioonidest juba ilmsiks tuli, kuulub iga andmekirje mingisse klassi, millel on mingisugune eesmärgi atribuut. Nendel atribuutidel võib olla väike arv eraldiseisvaid väärtuseid, mis kõik vastavad sellele klassile. Iga kirje koosneb kahest osast – ennustatavuse atribuudi väärtustest ning eesmärgi atribuudi väärtusest. Esimesed neist on selleks, et ennustada eesmärgi atribuudi väärtust. Seejuures need esimesed väärtused peaksid olema asjakohased eesmärgi atribuudi klassile. Näiteks kui eesmärgi atribuut peaks näitama, kas patsiendil on või võib tulevikus areneda teatud haigus, siis ennustatavuse atribuudid peaksid sisaldama vastavat meditsiinilist informatsiooni, mis on oluline antud ennustuse jaoks, mitte aga ebaolulisi atribuute, nagu patsiendi nimi.

Klassifitseerimise puhul jagatakse kaevandatavad andmed juhuslikult kahte erinevasse klassi - treeninghulka ning ning testimishulka. Treenimishulk tehakse tervenisti kättesaadavaks andmekaevandusalgoritmile, seega algoritm pääseb ligi nii ennustatavuse atribuutidele, kui ka eesmärgi atribuutidele.

Andmekaevandusalgoritmi eesmärk on avastada seos ennustatavuse atribuutide ning eesmärgi atribuutide vahel, kasutades selleks treenimisandmehulka. Seega, et algoritm suudaks avastada seoseid nende kahe atribuudi vahel, peab tal olema ligipääs mõlemate väärtustele treeningandmehulgas. Leitud seoseid kasutatakse, et ennustada klasse (eesmärgi atribuutide väärtuseid) testimisandmehulgast. Algoritmi jaoks sisaldab testimisandmehulk tundmatuid klasse ja andmeid. Peale seda, kui algoritm on ennustanud uued klassid, antakse talle tegelikud klassid vast-klassifitseeritud andmehulgast. Juhul, kui algoritmi poolt ennustatud klassid vastavad tegelikele andmehulkade klassidele, võib seda ennustust pidada õigeks. Aga kui saadud klassid ja tegelikud klassid ei kattu omavahel, siis ennustus ebaõnnestus. Algoritmi üks eesmärke on ka saavutada maksimaalne klassifitseerimistäpsus testimisandmehulgas, milleks on siisi õigete ennustuste arv jagatud kogu ennustuste arvuga.

Et antud juttu pisut arusaadavamaks teha, võib selle kohta tuua ühe illustreeriva näite. Tabel 2.1.1 kujutab endast treenimisandmehulka, mis koosneb kümnest andmereast ning kolmest ennustatavuse atribuudist – *vanus*, *sugu*, *palk* – ning ühest eesmärgi atribuudist – *ostab*. Näites on algoritmi eesmärk avastada reegleid, et ennustada ostmise väärtusi testimisandmehulgast. Selleks peab ta eelnevalt analüüsima antud tabelis väljatoodud treenimisandmehulka. Eesmärgi atribuut võib sisaldada väärtuseid *jah* või *ei*, näidates, kas vastav klient ostab mingit kindlat toodet või mitte.

Vanus	Sugu	Palk	**Ostab**
25	M	keskmine	**jah**
21	M	kõrge	**jah**
23	N	keskmine	**jah**
34	N	kõrge	**jah**
30	N	keskmine	**ei**
21	M	madal	**ei**
20	M	madal	**ei**
18	N	madal	**ei**
34	N	keskmine	**ei**
55	M	keskmine	**ei**

Tabel 2.1.1. Treenimisandmehulga näidis klassifitseerimisalgoritmi jaoks.

Üks võimalikke reeglite hulga, mis antud treenimisandmehulgal võiks põhineda, on toodud joonisel 2.1.1. Sellel väljatoodud neli reeglit vastavad 100% treeningandmetele ning kui antud kirje rahuldab reegli IF osa, siis vastav reegel THEN osas ennustab õige klassi vastava kirje jaoks. Siiski, 100% täpsus treeningandmehulgal ei pruugi anda sama tulemust testimisandmehulgal.

```
IF(Palk = madal) THEN (Ostab = ei )
IF(Palk = keskmine) AND (Vanus <= 25) THEN (Ostab = jah)
IF(Palk = keskmine) AND (Vanus > 25) THEN (Ostab = ei)
IF(Palk = kõrge) THEN (Ostab = jah)
```

Joonis 2.1.1. Klassifitseerimisreelgile näide tabeli 2.1.1 baasil.

2.1.1Täpsuse hindamine treenimise ja testimise teel

Klassifitseerimisalgoritmi edukuse saavutamiseks on oluline, et kasutataks juhuslikke osasid kogust olemasolevatest andmetest treenismis- ning testimisandmehulkade jaoks. Ülalpool juba tõime välja klassifitseerimistäpsuse definitsiooni – korrektselt klassifitseeritud ennustuste arv jagatud kogu ennustuste arvuga. Testimisandmehulgal arvutatud klassifitseerimistäpsus on vaid ennustatav täpsus tegelikul, tundmatul andmehulgal.

Tavaliselt koosneb klassifitseerimine kolmest etapist. Esiteks algoritm eraldab mingeid teadmisi treenimisandmetest. Teiseks mõõdetakse klassifitseerimistäpsust saadud teadmistel testimisandmete baasil. Saadud klassifitseerimistäpsus on hinnanguliselt õige ka kogu ülejäänud, tundmatu andmehulga jaoks. Teoreetiline ehk akadeemiline uuring tavaliselt lõppebki siinkohal, kuid päris elus toimivatel rakendustel on tavaliselt ka veel kolmas etapp – edaspidi kasutatakse algoritmi, et klassifitseerimiseks täiesti uusi andmeid ja klasse – andmeid, mis ei olnud esialgu kättesaadavad ning mille klassid on kasutajale samuti täiesti tundmatud.

Nüüd, kui on vaja hakata klassifitseerima täiesti uut andmehulka, saame kasutada kogu varasemat olemasolevat andmehulka (mida varem kasutati treenimiseks ning testimiseks) uue treenimisandmehulgana. See suurendab treenimiseks kasutatavat andmehulka, mida kasutatakse edaspidises klassifitseerimises. Nüüd muutuvad uued andmed testimisandmehulgaks.

Lisaks ülalpool kirjeldatule, kasutatakse ka veel ristvalideerimismeetodit. Antud meetodi puhul jagatakse andmed juhuslikul teel k erineva partitsiooni vahel. k siin tähistab kasutaja poolt defineeritud parameetrit ning tavaliselt valitakse k väärtuseks 10. Iga partitsioon peaks sisaldama samas suurusjärgus andmeid. Seega kui k=10, siis iga partitsioon peaks sisaldama 1/10 kogu originaalandmehulgast.

Antud juhul jookseks klassifitseerimisalgoritm 10 korda läbi. Igas tsükli ringis kasutatakse hetkel aktiivset partitsiooni testandmehulgana ning sellele eelnenud partitsiooni kui treeningandmehulka. Seega ristvalideerimismeetodi puhul kasutatakse igat partitsiooni täpselt ühe korra testandmetena ning k-1 korda treeningandmetena. Saadav klassifitseerimistäpsus on lihtsalt aritmeetiline keskmine, mis saadakse k-korda toimunud ristvalideerimisel.

Igal tsükli ringil kasutab klassifitseerimisalgoritm ainult ühte partitsiooni testimiseks ning kõiki teisi treenimiseks. Selle meetodi plussiks on, et kasutatakse võimalikult suurt andmehulka treenimiseks, miinuseks aga see, et antud protseduur on väga ressursinõudlik ning seega kasutatav pigem väiksematel andmehulkadel.

Kokkuvõtteks tuleks veelkord mainida, et mõlema esitatud protseduuri puhul on oluline, et partitsioonid ehk andmehulgad valitaks juhuslikult. Seega treenimisandmehulk ning testimisandmehulk peaksid olema statistiliselt iseseisvad üksteise suhtes.

2.2Sõltuvuse modelleerimine

Sõltuvuse modelleerimine kasutab meetodeid, mis kirjeldavad muutujate vahelisi sõltuvusi ja seoseid (Susan P. Imberman, 2001).

Sõltuvuse modelleerimine koosneb tähelepanuväärsete sõltuvuste mudelite leidmisest muutujate vahel. Sõltuvuse mudeleid on kahesuguseid - struktuurse taseme mudel, kus muutujad on lokaalselt sõltuvad üksteisest – ning kvantitatiivse taseme mudel, mis määratleb ära muutujate vaheliste sõltuvuste tugevused, kasutades mingisugust numbrilist skaalat (Fayyad,Piatetsky-Shapiro,Smyth, 1996).

Nagu juba nendest definitsioonidest välja tuleb, siis üldiselt tegeleb sõltuvuse modelleerimine atribuutide vaheliste sõltuvuste otsimisega. Sisuliselt sama ülesandega tegeles ka eelmises peatükis vaadeldud klassifitseerimine. Siiski on nende kahe meetodi vahel oluline erinevus. Klassifitseerimise puhul oli tegemist ainult ühe eesmärgiatribuudiga, mille väärtust püüti ennustada ning me olime üldjuhul huvitatud ainult sellest, kuidas antud atribuut sõltus ennustatavuse atribuutidest. Sõltuvusi viimaste vahel ilma eesmärgiatribuuti puudutamata aga ei peetud üldjuhul huvitavaks, kuna need ei ole vajalikud uute andmehulkade klassifitseerimiseks.

Seevastu sõltuvuse modelleerimise meetod kasutab laiemaid sõltuvuste hulkasid ning on ka üldjuhul seotud suurema otsimisalaga. Ülesanded võivad olla täiesti piiramatud sõltuvuste avastamise atribuutide valiku suhtes, aga ka sõltuvuste suuna suhtes. Erinevus klassifitseerimisest on viimase asümeetrilisus, kus ennustatavuse atribuudid võivad asuda ainult lause IF poole peal ning eesmärgiatribuudid lause THEN poole peal.

Nagu ka klassifitseerimise puhul valiti ainuke eesmärgiatribuut kasutaja poolt, valitakse ka sõltuvuse modelleerimisel eesmärgiatribuudid kasutaja poolt, kusjuures siin võivad need atribuudid paikneda nii IF kui ka THEN lause poole peal, aga mitte mõlemal poolel korraga ühes lauses. See piirang on oluline, et vältida mõtetute lausete genereerimist nagu näiteks "IF (A=1) AND (B=2) THEN (A=1)". Mitte-eesmärgiatribuudid võivad aga esineda ainult IF lause poolel. THEN lause pool aga saab korraga sisaldada vaid ühte eesmärgiatribuuti, kui samal ajal IF pool võib sisaldada mitut eesmärgi- kui ka ennustatavuse atribuuti. Siinkohal tooks veel välja erinevuse klassifitseerimisalgoritmiga, kus THEN poolel oli alati üks ja seesama eesmärgiatribuut. Sõltuvuse modelleerimises võivad erinevates lausetes esineda erinevad eesmärgiatribuudid.

2.3Assotsiatsioonireeglite analüüs

Assotsiatsioonireeglite analüüsi algoritmid arendati ostukorvi andmete analüüsimise eesmärgil. Need leidsid toodete grupid, mida ostjad tavatseid koos osta (Imberman, 2001).

Assotsiatsioonireeglite analüüsi eesmärgiks on omavahel püsival ja mõistlikul viisil seotud nähtuste, sündmuste, objektide vms leidmine (Tiidumaa, 2003).

Assotsiatsioonireeglite analüüs on assotsiatsioonireeglite leidmine näitamaks atribuutide väärtuseid, mis sageli ilmnevad koos antud andmehulgas. Assotsiatsioonireeglite analüüsi kasutatakse laialdaselt kaupluse ostukorvi tehingute analüüsimiseks (Han,Kamber, 2000).

Assotsiatsioonireegel on seos kujul X -> Y, kus X ning Y on omavahel mittesidusad andmehulgad. Iga reegel arvutatakse tavaliselt välja toetuse ning usalduse mõõtmisel. Assotsiatsioonireegeli toetuseks on nii X kui ka Y hulga andmete arvu suhe, kus väärtuseks on 'jah', jagatud kogu andmete arvuga. Usalduseks on aga hulkade X ja Y andmete arvu suhe, kus väärtuseks on 'jah', jagatud hulga X 'jah' väärtuste arvuga. Assotsiatsioonireegli meetod otsib andmehulgast välja kõik reeglid, kus toetus ning usaldus on suuremad või võrdsed kasutaja poolt defineeritud väärtustega.

Nagu ka definitsioonides ilmsiks tuli, kasutatakse nimetatud meetodit üsnagi palju just kaubanduses, seega võib protsessi piltlikumaks muutmiseks vaadelda seda kui suure toidupoe müügiprotsessi. Eksisteerib suur hulk kaubaartikleid ning kliente, kes sooritavad oste. Antud protsessist tekibki kaupluse andmebaasis nn ostukorv, millest saab vajalikku informatsiooni turundusosakond, et müüginumbreid võimalikult kõrgeks tõsta. Saadud ostukorvide pealt on võimalik leida erinevate toodete koos-ostmise seoseid.

Kuna aga selliseid reegleid võib tulla suhteliselt palju, siis on (Tiidumaa, 2003) jaganud need nelja peamisse gruppi:

- Juhuslikud seosed – kaks artiklikt võisid olla juhuslikult samal päeval poole hinnaga müügis ning seetõttu neil puudub omavaheline põhjuslik seos.
- Juba teadaolevad seosed – kahe artikli vaheline seos on juba varasemalt teada.
- Uued ent mitteolulised seosed – kahe artikli vaheline seos on küll uus, kuid antud ülesande jaoks olematu praktilise tähtsusega.
- Uued ja olulised seosed - kahe artikli vaheline seos on täiesti uus ja praktiliselt kasutatav: artiklid paigutatakse kõrvuti, et klient mõlemad kindlasti ostaks; artiklid paigutatakse üksteisest võimalikult kaugele, et klient peaks võimalikult paljude artiklite vahelt läbi käima; soodsama hinnaga artiklit ostes ei pane klient tähele, et teine artikkel on sama võrra kallim.

2.4Leitud reeglite kvaliteedi hindamine

Leitud reeglite kvaliteedi hindamine ei ole sugugi lihtne ülesanne, kuna see võib (ning üldjuhul ka peab) vastama mitmele kriteeriumile. Ideaaljuhul peaksid andmekaevandusalgoritmi poolt avastatud reeglid olema täpsed, arusaadavad ning huvitavad (üllatavad, kasulikud).
Seejuures tuleks veel märkida, et ühe kriteeriumi puhul edukas tulemus ei pruugi seda olla teiste kriteeriumide puhul. (Freitas, 2002) toob selle ilustreerimiseks ühe lihtsa näite: Meditsiinilisest andmebaasis leitakse järgmine reegel – IF (rase? = jah) THEN (sugu = naine). Antud reeglil on väga kõrge ennustatavuse täpsus ning seda võib pidada vägagi arusaadavaks ehk mõistetavaks tulemuseks. Siiski pole avastatud reegel sugugi huvitav, kuna avastatud seos on ilmselgelt loogiline.

2.5Klasterdamine

Klasterdamine on andmete grupeerimine klassidesse, nii et iga klastri objektid on omavahel väga sarnased ning erinevad teiste klastrite objektidest. Objektide vahelised erinevused põhinevad atribuutide väärtustel. Tihtipeale avaldatakse erinevused kauguse mõõtudena (Juhkam, 2004).

Klasterdamine on tavaline kirjeldav meetod, kus püütakse leida andmete kirjeldamiseks lõplikud hulgad kategooriaid või klastereid (Jain, Dubes, 1988).

Klasterdamise meetod kujutab endast kaevatavate andmete jaotamist mitmetesse gruppidesse või klastritesse niimoodi, et: (a) igas klastris paiknevad kirjed on üksteisele väga sarnased; ning (b) kirjed igas klastris on väga erinevad kirjetega, mis asuvad teistes klastrites (Freitas, 2002).

Klasterdamise ühe lihtsa näitena võib tuua välja pesu pesemise. Üldjuhul enamik inimesi sorteerib pesu enne masinassa panemist – riided, mida tuleb pesta madalal temperatuuril, heledad, tumedad ning eredavärvilised. Sorteerimine toimub nende riiete atribuutide ehk käitumise järgi pesumasinas (ühed annavad värvi ning teised võivad kahjustuda kõrgematel temperatuuridel). Kogu selle protsessi juures võivad aga ka ilmneda mõned keerulised otsused, näiteks millisesse hunnikusse paigutada valge särk punaste triipudega – kuna särk on enamjaolt valge, aga tal on ka värvilisi komponente).

Nagu esitatud definitsioonidest ning ka toodud lihtsast näites ilmneb, on klasterdamine üldjoontes sarnaste andmete grupeerimine.Seesugune klastrite otsimine on masinõppe seisukohalt varjatud mustrite otsimine ning teda võib pidada järelvalveta tegevuseks (ei nõua kasutajapoolset sekkumist).

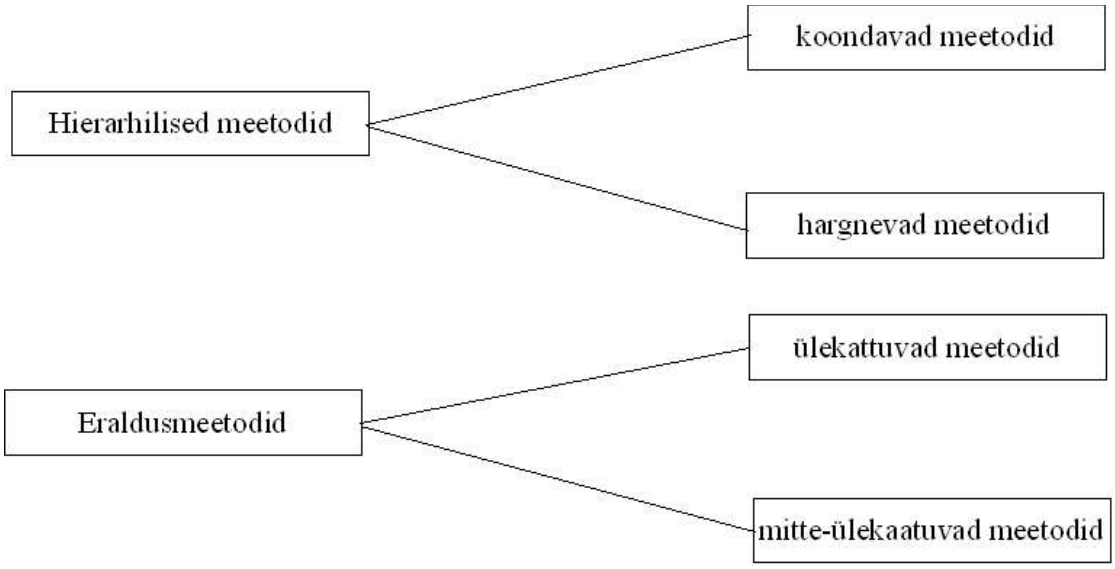
Klasteranalüüs sai populaarseks seoses andmebaasid mahtude tohutule kasvule. Andmekaevanduses kasutatakse seda meetodit nii iseseisva töömeetodina, kui ka ettevalmistuseks teiste meetodite kasutamiseks, näiteks klassifitseerimise jaoks. (Juhkam, 2004) toob välja järgmised põhilised nõuded klasterdamismeetodite jaoks:

- **Võime töötada suure andmehulga korral**: Suurtes andmebaasides on miljoneid kirjeid, mis nõuab klasterdamismeetoditelt efektiivsust.
- **Võime töötada erinevat tüüpi tunnustega**: Paljud algoritmid on loodud intervalltunnuste klasterdamiseks. Samal ajal andmebaasis esinevad ka binaarsed, nominaalsed ning järjestustunnused või nende tüüpide segu.
- **Suvalise kujuga klastrite avastamine**: Paljud klasterdamisalgoritmid avastavad klastreid, mis põhinevad *Eukleidilisel* või *Manhattani* kauguse mõõtudel. Sellised algoritmid kalduvad avastama sfäärilisi klastreid ühesuguse suurusega ja tihedusega, kuid klastrid võivad olla igasuguse kujuga.
- **Minimaalne nõutav sisendparameetrite arv**: Paljud algoritmid nõuavad kasutajatelt sisestada vajalikud parameetrid (näiteks klastrite arv). Klasterdamise tulemused võivad olla tundlikud sisendi suhtes, mille tõttu tulemuse headust on raske kontrollida.
- **Võime töötada "müraga"**: Paljud andmebaasid sisaldavad erandeid, puuduvaid ning vigaseid andmeid. Mõned klasterdamisalgoritmid on tundlikud sellise "müra" vastu.
- **Sõltumatus andmete järjekorrast**: Mõned klasterdamisalgoritmid on tundlikud andmete järjekorra suhtes ja võivad anda täiesti erinevaid tulemusi erineva järjestamise puhul.
- **Võime töötada mitmedimensionaalsete andmetega**: Andmebaas võib sisaldada palju atribuute, kuid paljud klasterdamise algoritmid töötavad hästi ainult väikese mõõtmiste arvuga.

Klasterdamisalgoritmide ülesanne on maksimaliseerida klasterisiseseid sarnasusi ning minimaliseerida klasteritevahelisi sarnasusi (Freitas, 2002). Paraku ainult nende kahe eesmärgi täitmine ei pruugi rahuldada head klasterdamislahendust kuna sellisel juhul on võimalik luua iga kirje jaoks omaette klaster. Seetõttu on veel oluline, et erinevate klastrite arv oleks võimalikult väike ning kirjete arv klastrites oleks võimalikult suur. Klasderdamise meetodi üks suurimaid väljakutseid ongi leida selline sobiv tasakaal, et need kolm tingimust oleksid rahuldatud.

2.5.1Ülevaade klasterdamismeetoditest

Klasterdamismeetodid tuleb valida sõltuvalt ülesande püstitusest ning andmete iseloomust. (Freitas, 2002) toob välja kaks enamkasutatavat klasterdamismeetodit: heirarhiline klasterdamine (*hierarchical clustering*) ning eraldusklasterdamine (*partitioning clustering*). Need jagunevad omakorda veel vastavalt siis hierarhilised meetodid – koondavad meetodid ning hargnevad meetodid – ja eraldusmeetodid – ülekattuvad ning mitte-ülekattuvad meetodid – nagu võib näha jooniselt 2.5.1.1.



Joonis 2.5.1.1. Peamised klasterdamismeetodid

Hierarhilised meetodid

Nagu eelnevalt sai mainitud, jagunevad hirarhilised meetodid koondavateks meetoditeks (*agglomerative methods*) ning hargnevateks meetoditeks (*divisive methods*). Koondava meetodi puhul jagatakse iga kirje alguses eraldi klastrisse. Järgmisel sammul koondatakse kaks kõige sarnasemat klastrit üheks klastriks. Protsess kordub kuni on täidetud vastav kriteerium või kuni on järele jäänud vaid üks klaster.

Hargnev meetod töötab vastupidiselt koondavale meetodile. Alustuseks paigutatakse kõik kirjed ühte klastrisse. Järgmise sammu puhul jaotatakse antud klaster väiksemateks klastriteks. Protsess kordub kuni etteantud kriteerium on täidetud või kuni käik kirjed on jagunenud iseseisvateks klastriteks.
Hargnevad meetodid on üldjuhul võrreldes koondavate meetoditeka oluliselt ressursinõudlikumad ning seetõttu on koondavad meetodid populaarsemad.

Eraldusmeetodid

Eraldusmeetodite puhul jagatakse andmebaasid etteantud arvu k -grupiks, kusjuures gruppide arv ei tohi ületada kirjete arvu ning iga grupp peab sisaldama vähemalt ühte kirjet ning iga kirje kuulub vähemalt ühte gruppi. Algoritm jaotab kirjed vastavalt siis k-klastrite vahel. Seejärel algoritm püüab parandada jaotust, vahetades punktid klastrite vahel. Eraldusmeetodid jagunevad ülekattuvateks meetoditeks – kirjed võivad paikneda mitmes grupis (klastris) – ning mitte-ülekattuvateks meetoditeks – iga kirje võib paikneda vaid ühes grupis.

Tihedusmeetodid

Enamus eraldusmeetodeid suudavad leida vaid sfäärilisele kujule lähedasi klastreid ning suvalise kujuga klastrite avastamine käib neil üle jõu. Viimast liiki klastrite leidmiseks kasutatakse tihedusmeetodeid (*density-based methods*). Nende meetodite puhul võetakse klastisse uued punktid niikaua, kuni punktide tihedus klastri ümbruses ületab mingi piiri. Ehk siis iga klastri punkti ümbruses peab sisalduma vähemalt mingi kindel arv punkte – siit tuleb ka mõiste "tihedus". Tihedusmeetodid aitavad leida erandeid või "müra".

Võremeetodid

Võremeetodid (grid-based methods) jagavad objektide ruumi lõplikuks ruutude arvuks, moodustades võrestiku struktuuri. Kõik klasterdamise operatsioonid toimuvad tekkinud võrestikul. Antud meetodi eeliseks on kiire tööaeg, kuna aeg ei sõltu objektide arvust, vaid hoopis ruutude arvust igas ruumi dimensioonis.

Mudelil põhinevad meetodid

Mudelil põhinevad meetodid (model-based methods) eeldavad mingisuguse parameetrilise mudeli kehtivust andmete kohta ning otsivad paremat mudelit iseloomustamaks andmeid. Algoritmid jaotavad kirjed klastritesse, konstrueerides tihedusfuntktsiooni, mis kajastab andmete ruumilist paiknemist. Mudelil põhinevatel meetoditega võib kindlaks teha klastrite arvu, põhinedes standardsele statistikale. Needmeetodid eeldavad aga mingisuguseid eelteadmisi andmete kohta.

2.6Induktiivse kalduvuse meetod

Induktiivne kalduvus masinõppe süsteemis on teatud oletuste hulk, mille abil on garanteeritud, et süsteem väljastab korrektsed hüpoteesid (Mitchell, 1996).

Induktiivne kalduvus on tingimuste hulk selliselt, et kui tingimused toimivad, siis süsteemi toimimine peaks olema edukas, kuid kui tingimused ei toimi, võib eeldada, et süsteem ei toimi nii hästi (Blockeel,Dehaspe, 2000).

(An inductive bias denotes any basis for choosing one generalisatioin (hypothesis) over another, other than a strict consistency with the observed training examples. Ler, 2004).
Induktiivne kalduvus on see, kui on kalduvus valida üks hüpotees teise – treenimisandmehulga puhul kindlalt esinenud hüpoteesi üle (Ler, 2004).

(Freitas, 2002) raamatust võib leida hea illustratiivse näite kirjeldamaks induktiivse kalduvuse meetodit. Oletame, et visatakse võltsmünti – kus tõenäosus, et tuleb kull või kiri, on erinev – 10 korda ning vaadelda viskamise tulemusi. Nüüd meil on ülesanne ennustada ette järgmise viske tulemused. Mittekallutatud ennustusstrateegia ennustaks tulemuseks nendes 10 viskes rohkem esinenud mündipoole. Antud strateegia võtab arvesse vaid tema valduses oleva treenimisandmehulga.

Kaalutlege nüüd strateegiat, mis ennustab tulemuseks kirja, kui kiri on esinenud vaid kolmel korral kümnest. Ilmselgelt on strateegial kalduvus eelistada kirja. Tekib küsimus, kas leitud kalduvus on hea või halb. Vastus aga sõltub sellest, kuidas antud münti on võltsitud, mis on aga meile teadmata ning mida me püüamegi avastada. Kui võltsitud mündil ongi kalduvus kirja poole, siis meie strateegia on toimiv, vastupidisel puhul meie strateegia ei toimi.

Esitatud näite puhul ilmneb, et kalduvuse efektiivsus sõltub algandmetest. Teisisõnu võib öelda, et iga kalduvus võib olla sobilik teatud andmehulkadele ning ebasobiv teistele andmehulkadele.

(Ler, 2004) toob välja kahte erinevat tüüpi kalduvusi:

- esituse kalduvus (representation bias) – Kasutatakse teatud esituskeelt, mis sisaldab pakutud hüpoteese ning seda keelt tutvustatakse andmekaevandusalgoritmile. Algoritm suudab leida vaid teadmisi, mida suunab antud esitluskeel. Näiteks kui algoritmile on esitletud vaid nominaalseid loogikaid, siis ei suuda ta kasutada järjestusloogikaid, nagu näiteks võrrelda kahte atribuuti – sissetulek ja väljaminek.
- preference bias (or search bias) - see kalduvus määrab, kuidas algoritm eelistab ühte hüpoteesi teistele. Reeglite ennustamise kontekstis kasutatakse seda kalduvust funktsiooni poolt, mis hindab pakutud reegli kvaliteeti ning genereerib sellest uusi reegleid .

2.7Otsustuspuu algoritmid

Otsustuspuu on graafiline puu struktuur, kus iga sõlm tähistab atribuudi väärtuse kontrollimist, iga haru esitab kontrollimise tulemust ning puu lehed tähistavad klasse või klasside jagunemist (Han, Kamber, 2000).

Otsustuspuu on lihtne, esitav vorm, et leitud mudel oleks kasutaja jaoks suhteliselt lihtsasti arusaadav (Fayyad,Piatetsky-Shapiro,Smyth, 1996).

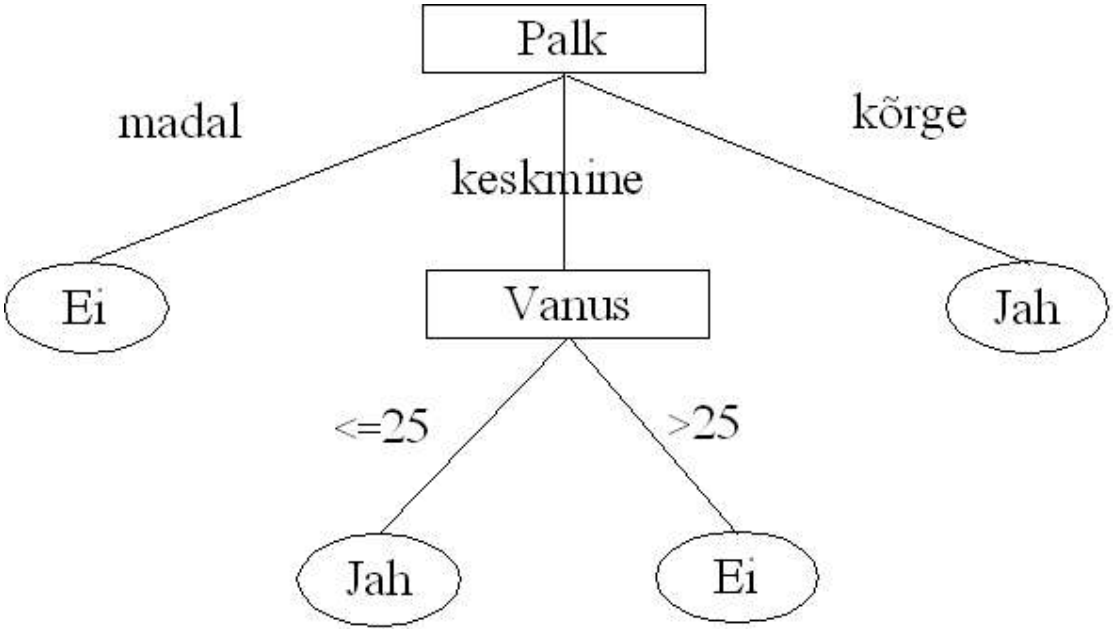
Otsustuspuu on oma olemuselt teadmiste esitlemise struktuur, mis koosneb sõlmedest ja harudes ning mis on organiseeritud puu kujuliselt nii, et: (a) iga sisemine (mitte-leht) sõlm kujutab endast ühte ennustatavuse atribuuti; (b) sisemisest sõlmest väljatulevad harud on tähistatud vastava sõlme atribuutide väärtustega; (c) iga leht tähistab klassi ehk eesmärgiatribuudi väärtust (Freitas, 2002).

Et esitatud definitsioone pisut illustreerida, vaatleme juba varasemast peatükist tuttavad näidisandmehulka (joonis **XXX**), mis koosneb kolmest atribuudist – vanus, sugu ning palk - ning eesmärgiatribuudist – ostab, mis näitab, kas vastav klient ostab antud toodet või mitte.

Vanus	Sugu	Palk	**Ostab**
25	M	keskmine	**jah**
21	M	kõrge	**jah**
23	N	keskmine	**jah**
34	N	kõrge	**jah**
30	N	keskmine	**ei**
21	M	madal	**ei**
20	M	madal	**ei**
18	N	madal	**ei**
34	N	keskmine	**ei**
55	M	keskmine	**ei**

Tabel 2.7.1. Treenimisandmehulga näidis otsustuspuu algoritmi jaoks.

Antud tabelit kasutades võib ehitada ühe lihtsa otsustuspuu:



Joonis 2.7.1. Otsustuspuu tabel 2.7.1 põhjal.

Vaatleme, kuidas kirje liigub esitatud otsustuspuus. Näitena võtame kirje (Vanus=23, Sugu=mees, Palk=keskmine, Ostab=?), kus ? tähtistab tundmatut väärtust. Esiteks kontrollib puu atribuudi Palk väärtust, kuna vastus on "keskmine", siis surutakse kirje edasi Vanuse sõlme. Nüüd kontrollib puu vanuse atribuudi väärtust, kuna vastus on 23, siis suundutakse lehte, mille väärtus on "Ostab". Kogu protsessi juures jäi kasutamata üks atribuut "Sugu", kuna see oli tulemuse saamiseks täiesti ebavajalik.

Otsustuspuu eeliseks teiste nn. "tasapinnaliste või lamedate" meetodite ees on see, et otustuspuu annab ka informatsiooni atribuutide olulisuse kohta. Mida lähemal on atribuut puu juurele, seda olulisem ta on klasside ennustamises antud andmehulgast. Lisaks on ka otsustuspuu meetod üsnagi kiire, kuna kasutab jaga-ja-valitse meetodit (*divide-and-conquer*). Puu laienedes alumised sõlmed sisaldavad järjest vähem kirjeid ning järgneva atribuudi valik toimud järjest kiiremini, kuna vaadeldavate atribuutide hulk järjest väheneb.

2.7.1Otsustuspuu ehitamine

Erinevate allikate põhjal alustatakse otsustuspuu konstrueerimist peamiselt ülalt alla. Kõik kirjed läbivad alustuseks puu juursõlme, seejärel jaotab algoritm need igas järgnevas sõlmes vastavalt valitud atribuutide järgi vastavatesse harudesse, kuni lõpuks jõuavad kirjed välja lehtedesse. Kogu selle protsessi eesmärk on andmehulgast erinevte klasside leidmine.

(Käärmann, 2003) kirjeldab otsustuspuu ehitusprotsessi järgnevalt - enamlevinud lähenemine otsustuspuu konstrueerimiseks on teha seda ülalt-alla tipu poolitamise teel, alustades kõikide õpiandmetega juurtipust ning rakendades rekursiivselt järgmist üldist eeskirja:

- Kui puu tippu kuuluvad vaid ühe klassi esindajad, siis märkida tipp leheks ning varustada vastava klassitunnusega.
- Hinnata headuse mõõduga kõikvõimalikke tipu lahutusi, so. tippu kuuluvate õpiobjektide jagunemisi alamtippude vahel. Valida parim lahutus ning formuleerida see tipu küsimuseks.
- Vastavalt valitud lahutusele luua alamtipud, jaotada nende vahel tipu objektid (õpiandmed) ning rakendada sama eeskirja alamtippudele.

(Freitas, 2002) järgi võib ühte enamlevinud algoritmitüüpi kirjeldada joonise 2.7.1.1 järgi, kus T tähistab treenimisandmeid vastaval puu sõlmel.

```
Algväärtusta T treenimisandmehulgaga;
(1) IF kui kõik kirjed hulgas T rahuldavad peatumiskriteeriumi
      THEN
(2)       loo leht mingi klassinimega ning peatu;
    ELSE
(3)       vali atribuut A, mida kasutatakse jaotusatribuudina ning vali
            kontrollribuut hulgast A, mis jaotaks andmed väljunditeks O1,..Ok;
(4)       tekita sõlm atribuudi A nimega ning tekita antud sõlmest harud igat
            liiki väljundite jaoks;
(5)       jaga T alamhulkadeks T1,..,Tk nii, et iga Tp i=1,..,k sisaldaks kõiki
            kirjeid hulgast T väljundiga Oi valitud kontrollatribuudi puhul;
(6)       korda seda algoritmi rekursiivselt iga alamhulga Ti, i=1,..,k puhul
    ENDELSE
   ENDIF
```

Joonis 2.7.1.1. Tavaline ülalt-alla otsustuspuu ehitamise algoritm

Sammus (1) kõige ilmsemaks peatumiskriteeriumiks on see, kui rekursiivse jaotamise protsessi tulemusena kõigil kirjetel treenimisandmehulgast on sama klass. Lisaks sellele kohustuslikule kriteeriumile, võib ka kasutada paljusid teisis. Mõnel juhul näiteks võib peatumine toimuda, kui on jõutud ettemääratud arvu kirjeteni.

Sammus (2) on äsjaloodud sõlmes kõigil kirjetel sama klass ning algoritm nimetab selle sõlme antud klassi järgi. Kui sõlmes on mitmeid klasse, siis nimetatakse see kõige sagedamini esineva klassi järgi.

Sammus (3) algoritm valib atribuudi ning ning kontrollib kirjete väärtuseid antud atribuudi järgi nii, et kontrollitud kirjed jagunevad erinevatesse klassidesse. Tekkinud väljundid peavad olema võimalikult "puhtad" ehk sisaldama võimalikult palju kirjeid samast klassist ning võimalikult vähe kirjeid erinevatest klassidest.

Sammud (4-6) on üsnagi mõistetavad joonise järgi, ega vaja laiemat lahtikirjutamist.

2.7.2Otsustuspuu kärpimine ehk tagasilõikamine

Otsustuspuu tagasilõikamise eesmärk on lihtsamate ning lühemate puude moodustamine, kuna seesugused puud on kasutajatele lihtsamini mõistetavad. Teiseks eesmärgiks võib veel pidada ka seda, et lõikamata puud võivad sisaldada palju mittevajalikke sõlmi ja lehti. Seejuures on veel oht treenimisandmehulga puhul tekkival ülekattumisel. Samas kui jälle liigselt puud tagasi lõigata, võib tekkida andmete alakattumise probleem.

(Käärmann, 2003) järgi konstrueeritakse esmalt täielik otsustuspuu, kus lehtedes on täpne klassijaotus ehk minimaalne entroopia. Seejärel hakatakse alt ülesse kontrollima naaberlehti – kas nende ühendamisel tekib piisavalt väike klassifitseerimistäpsuse kadu. Kui jah, siis tipud ühendadakse. (Freitas, 2002) nimetab seda meetodit kui järel-lõikamist. ning toob lisaks veel teise meetodi – eel-lõikamine. Vastupidiselt järel-lõikamisele, toimub hargnemise peatamine varasemalt, ehk siis puu hargemise protsessis juhu, kui edasine laienemine antud punktis ei tundu olevad huvitav või kasutlik.

Järel-lõikamine peaks tekitama suurema ennustustäpsusega otsustuspuid, kui eel-lõikamine, kuna suurem sõlmede hulk pakub rohkem informatsiooni lõikusprotseduuri jaoks. Igasuguse otsustuspuu efektiivsus sõltub seejuures siiski algandmetest. Eel-lõikuse eeliseks aga on see, et ta muudab algoritmi oluliselt võimekamaks, kuna tekib vähem sõlmi ning on seetõttu sobivam just eriti suurte andmebaaside puhul.

Ühe huvitava järel-lõikuse meetodi võib välja tuua (Quinlan, 1987) poolt. Antud meetod muudab puu esiteks IF-THEN klassifitseerimisreegliteks. Iga tee juursõlmest kuni leheni konverteeritakse reegliks: sisemised sõlmed ja nendele vastavad väljundharud konverteeritakse lause IF pooleks; lehed konverteeritakse lause THEN pooleks. Edasi eemaldatakse kõige ebaolulisemad reeglid, kuni antud protsess ei vähenda kogu puu klassifikatsioonitäpsust.

2.8Reeglite induktsiooni algoritmid

Reeglite induktsioonidega otsime parimaid ennustavaid reegleid, mis seoks kõiki või mingit alahulka algandmetest (Smyth, Goodman, 1991).

Reeglite induktsiooni algoritmide eesmärk on leida teatud reeglite hulk, mis vastaks mingisugustele eelnevalt kirjeldatud kriteeriumitele (Dung Duc, 2003).

Kirjanduses mainitakse reeglite induktsiooni kui algoritmi, mis avastab andmetevahelisi reegleid paindlikumalt võrreldes otsustuspuuga, ehk siis avastatud reeglid võivad katta andmehulki ka ülekattumisega. Sisuliselt tähendab see seda, et üks andmekirje võib olla kaetud mitme reegliga, erinevalt otsustuspuus, kus üks kirje oli kaetud täpselt ühe reegliga.

(Freitas, 2002) esitab reeglite induktsiooni algoritmi joonise 2.8.1 abil. Joonise osal (a) algväärtustatakse kõigepealt T kõiki treenimisandmehulga kirjetega ning avastatud reeglite hulk (*ReegliteHulk*) väärtustatakse tühihulgaga. Seejärel teeb algoritm silmuse ehk tsükli, kutsudes välja protseduuri *Indutseeri-Reegel* ning lisab antud protseduuri poolt loodud reegli algoritmi reeglite hulka. Edasi eemaldatakse kõik kirjed treenimishulgast T, mis on äsja indutseeritud reegliga kaetud. Antud tsüklit korratakse

seni, kuni peatumiskriteerium on täidetud ehk siis kuni kõik kirjed on kaetud vähemalt ühe indutseeritud reegliga või kui katmata kirjete arv on väiksem eelnevalt defineeritud arvust.

Joonise 2.8.1 osa (b) kujutab endast algoritmi protseduuri Indutseeri-Reegel. Protseduur käivitud tühja indutseeritud reegliga – reeglile ei ole veel seatud tingimusi. Seejärel moodustab protseduur silmuse, mille käigus valitakse parim tingimus indutseeritavale reeglile, mis baseerub etteantud hindamisfunktsioonile andmehulga T baasil. Antud silmust korratakse, kuni indutseeritavat reeglit saab veel täiendada uute tingimustega. Seejärel tagastab protseduur äsja indutseeritud reegli algoritmi (a) osale.

(a) Reeglite hulga konstrueerimine
(üks reegel korraga)

```
T = {kõik treenimisandmehulga kirjed)
ReegliteHulk = Ø

while (peatumiskriteerium pole rahuldatud)

 rakenda Indutseeri-Reegel(T);
 ReegliteHulk = ReegliteHulk U
                IndutseeritudReegel;
 T = T – {kirjed, mis on kaetud
          IndutseeritudReegliga};
endwhile
```

(b) Reegli konstrueerimine
(üks tingimus korraga)

```
protseduur Indutseeri-Reegel(T)
IndutseeritudReegel = Ø

while (IndutseeritudReeglit saab täiendada,
       lisades talle uusi tingimusi)

 vali ParimTingimus;
 IndutseeritudReegel = IndutseeritudReegel U
                       ParimTingimusl;
 tagasta (IndutseeritudReegel)

endwhile
```

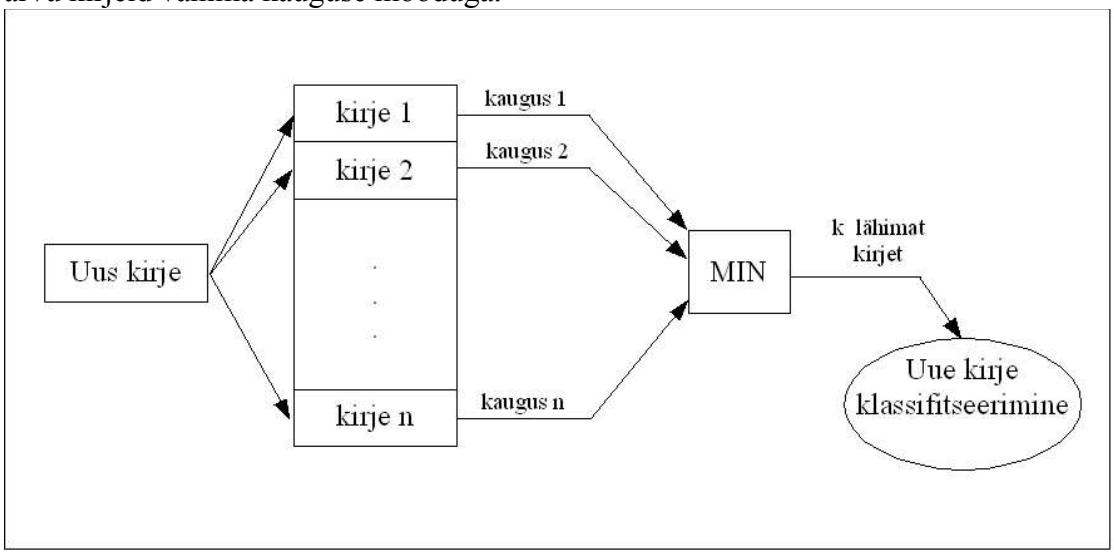
Joonis 2.8.1. Reeglite induktsiooni algoritm.

2.9Lähima naabri algoritmid

K-lähima naabri algorimt leiab k arvu naabreid, mis on kõige lähemal uuele treenimisandmehulga kirjele, baseerudes mingisugusel sobival sarnasusel või kauguse mõõdul (Khan,Ding,Perrizo, 2002).

Lähima naabri algoritm on ennustustehnika, mille eesmärk on ennustada millised ennustusväärtused ühel kirjel on, baseerudes sarnaste ennustusväärtustega kirjetel varasemast andmebaasist ning kasutada ennustusväärtust uue kirje jaoks selliselt kirjelt, mis on temale kõige "lähedasem" (Berson, Smith, Thearling, 1999).

Erinevalt otsustuspuu ja reeglite induktsiooni algoritmidest ei loo lähima naabri algoritm uut andmemudelit, vaid lihtsalt klassifitseerib uue kirje vastavasse klassi, kasutades selleks olemasolevaid teadmisi ja mudeleid. Uue kirje klassifitseerimiseks võrdleb algoritm seda kõigi treenimisandmehulgast salvestatud kirjetega. Seejärel saab ta vasteks arvu k kirjeid, mis on kõige lähedasemad uuele kirjele. k tähistab siin algoritmile ette antud tavaliselt üsnagi väikest algarvulist numbrit. Uus kirje liigitatakse selle klassi alla, millest tulenes kõige rohkem sarnaseid kirjeid. Antud protsessi kirjeldab joonis 2.9.1, kus MIN tähistab minimaalset operaatorit, mis valib k arvu kirjeid vähima kauguse mõõduga.



Joonis 2.9.1. Lähima naabri algoritmi põhiidee.

Tavaliselt kasutatakse uue kirje ja treenimisandmehulga kirjete vahelise kauguse hindamiseks atribuutide kaalu, et määrata iga atribuudi olulisus klassifitseerimisel.. Kui atribuut on ebaoluline, siis tema kaal peaks olema võimalikult väike, et selle atribuudi väärtustel oleks võimalikult vähene mõju uue ja olemasolevate kirjete erinevuse hindamisel. Ning vastupidi, kui atribuut on oluline, siis peaks tal olema võimalikut kõrge kaal, et antud atribuudi erinevustel oleks võimalikult suur mõju.

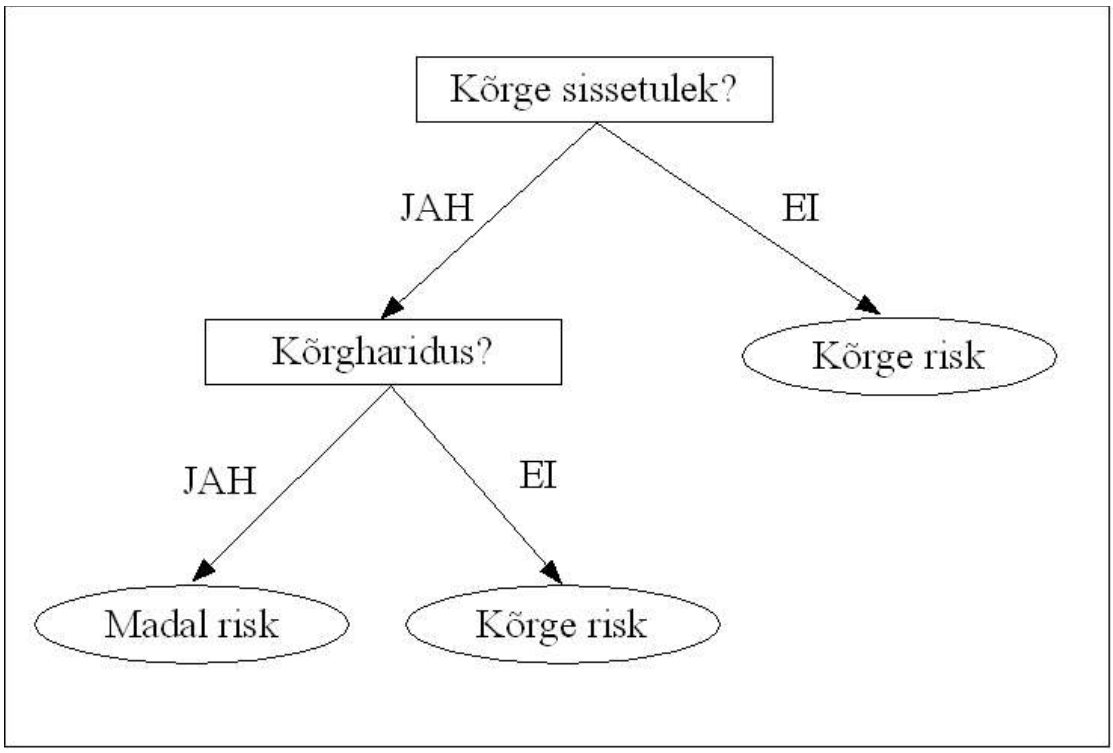
Kuigi lähima naabri algoritmid ei tekita uusi klasse või mudeleid, pakuvad need aga hoopis selgitust, kuidas uue kirje klassifitseerimine toimub. Uue kirje klassifitseerimisel näitab süsteem kasutajale kõige lähedasemaid kirjeid antud kirjele. Seesuguseid teadmisi suudab kasutaja valideerida ning see on kergesti tõlgendatav, senimaani, kuni lähimate naabrite ning atribuude arv ei ole väga suur.

3.Praktilised rakendusvaldkonnad

Siinses peatükis vaatleme lähemalt andmekaevanduse kasutusvaldkondi ning toome välja ka mõned huvitavamad praktiliste rakenduste näited. Andmekaevanduse esialgsed ning algelised rakendused said alguse eelkõige panganduse, kindlustuse ning telekommunikatsiooni valdkondades, kuid on tänapäeval juba laienenud praktiliselt kõikidesse eluvaldkondadesse. Loomulikult ei jõua me loetleda kõiki rakendusvaldkondi, kuid toome välja nendest olulisemad ja enamlevinud andmekaevandusrakendused.

3.1Pangandus

Panganduses on üheks andmekaevanduse rakenduseks näiteks kliendiprofiilide leidmine ning segmenteerimine – võimalikult täpselt ning minimaalsete kuludega tabada ära potensiaalsed toote või teenuse ostjad ning lisaks ka klientide riskantsuse hindamine näiteks laenu andmisel – kui suure tõenäosusega mingil kliendil võivad tekkida makseraskused.



Joonis 3.1.1. Laenuriski otsustuspuu.

Teiseks oluliseks rakenduseks oleks kliendikaotuse vältimine. Kuna pangaduses valitseb küllaltki tihe konkurents, siis seega ühele firmale uus klient tähendab teisele firmale kliendi kaotust. Andmekaevandus võimaldab sellest üsnagi varakult teada saada ning seega võib aidata ka takistada kliendi üleminekut konkurendi juurde. Seda sama mudelit saab kasutada ka hinnakujundamisel, et vältida klientide kaotust.

Lisaks eelnevale on panganduses kasutusel ka spetsiifilisemaid rakendusi nagu näiteks investeeringute optimaalne juhtimine ning riskide hindamine, krediidiriskide hindamine, krediidi kulukuse määra võimalikult täpne hindamine.

3.2Kindlustus

Kindlustussektorit võib pidada ka üheks esimeseks valdkonnaks, kus andmekaevandus kasutusele võeti. Põhilisteks kasutusaladeks on riskide ennustamine ja hindamine, hinnakujundus, kahjunõuete töötlemine ja analüüs, kindlustuspettuste ning keerukamate petuskeemide avastamine.

Kindlustuses kasutatakse andmekaevandust peamiselt kahel eesmärgil - avastada ja vähendada kindlustuspettuse juhtumeid ning oma klientide vajaduste paremaks mõistmiseks. Andmekaevandus on abiks näiteks kindlustusagentidele andes neile vajalikke eelteadmisi edukaks müügiks. Agentidel tekib võimalus teha õiged pakkumisi õigetele klientidele. Hindade langetamine ei ole alati just kõige parem lahendus konkurentsieelise saavutamiseks, andmekaevanduse abil on võimalik oma positsiooni parandada kasvõi näiteks õigele sihtgrupile suunatud vastava kampaaniaga (West, 2005).

Teiseks oluliseks kasutusalaks on kindlustuspettuste avastamine. Analüütik võib leida mustri, mis võib viidata pettusele. Sellele hüpoteesile rajanedes teostab ta erinevaid päringuid, et selgitada välja, kas on ikka tegemist pettusele omase käitumisega. Kui tulemused on negatiivsed, alustab analüütik uue hüpoteesi ja uute päringutega. Protsess on küllaltki aeganõudev ning eeldab ka analüütiku subjektiivset tulemuste hindamist. Paraku tekib siinkohal oht, et analüütilkule tundmatud petuskeemid võivad jäädagi avastamata (Moss, 2003).

3.3Jaekaubandus

Assotsiatsioonireeglite leidmise meetod on oma alguse saanud just jaekaubandusest ning esialgselt oligi ta tuntud kui ostukorvi analüüs. Ka tänapäeval kirjeldatakse seda meetodit just ostukorvi näite kaudu. Analüüsi olemuseks ongi näiteks kassas registreeritud ostukorvi sisu analüüs, leidmaks omavahel tugevas seoses olevaid kaupu, ehk siis mida ostetakse sageli koos. Siinkohal ei paku huvi kooslus, mis on ilmselge - saia ja leiba ostetakse ilmselgelt koos, kui näiteks samas õlut ja mähkmeid omavahel seostada oleks üsnagi ootamatu. Viimane näide on aga just antud valdkonnas huvipakkuvam.

Seesugusest andmekaevandusmeetodist saadud informatsiooni saavad kauplused rakendada üsna mitmet moodi:

- paigutades koososteta vad tooted lähestikku, et klient kindlasti mõlemat toodet ostaks
- asetada koososteta vad tooted üksteisest võimalikult kaugele, et klient peaks ühe toote juurest teiseni liikudes mööduma võimalikult paljudest riiulitest, lootuses, et asetab ka sealt midagi oma korvi.
- sooduskampaania varjust müüa teisi tooteid samavõrra kallimalt, kuna ostja ei pane seda sellisel juhul tihtipeale tähele.

3.4Telekommunikatsioon

Telekommunikatsiooni tööstusharu genereerib ja salvestab meeletutes kogustes andmeid. Nendeks on kõnede detailsed andmed, mis kirjeldavad telekommunikatsioonivõrgus liikuvaid kõnesid, võrgu andmeid, mis kirjeldavad riist- ja tarkvara komponentide seisundeid võrgus, ning klientide andmeid, mis kirjeldavad siis telekommunikatsiooni kliente. Andmete hulk on nii suur, et manuaalselt seda analüüsida on praktiliselt võimatu. Selle tõttu on välja arendatud hulk automaatseid süsteeme, mis tegelevad näiteks petturlike kõnede avastamisega aga ka võrgu vigade identifitseerimisega.

Üheks kõige tavalisemaks pettuste avastamise meetodiks on koostada vastavale pettusele kasutajaprofiil ning jälgida klientide tegevusi vastava profiili järgi. Antud meetod baseerub kõrvalekalde avastamisel. Helistamiskäitumine saadakse kliendi kõne detailide summeerimisel. Kui neid summeerimisi teostatakse reaalajas, siis avastatakse pettus kohe peale selle teostamist (Weiss, 2004).

3.5Maksuamet

Maksuametis on põhiliseks andmekaevanduse rakendamise valdkonnaks rahapesuskeemide tuvastamine. Üheks selliseks meetodiks on klasterdamine – kus sarnased ettevõtted jagatakse samasse klastrisse. Kui nüüd on teada mõned maksupettusega tegelenud firmad, siis tasub läbi vaadata kõik samasse klastrisse jäänud ettevõtted. Alati ei pruugi need küll kõik tegeleda maksudest kõrvalehiilimistega, kuid lootust sealt neid leida on küllaltki suur. Samamoodi on kasutatud ka andmekaevandusmeetodeid ümbrikupalkade maksmise tuvastamiseks.

3.6Kuritegevuse ning terrorismiga võitlemine

Kuritegevuse vastu võitlemise üks peamisi ning suuremaid valdkondi on olnud kelmuste avastamine. Kurjategijad tihtipeale panustavad väikestele kõrvalekalletele, mis peaks jääma märkamatuks suurte andmehulkade korral. Andmekaevanduse üks tugevaimaid külgi ongi just kõikvõimalike kõrvalekallete avastamine.
Samuti rakendatakse andmekaevandamist suhtlemisvõrgustike avastamiseks ning analüüsimiseks, mida on võimalik kaalutud graafina kujutada. Selline meetod toob välja omavahel kõige tihedamini suhtlevad inimesed ning aitab välja selgitada grupeeringute liidreid.

Terrorismi vastu võitlemine.
Washington, 8. august - rohkem kui aasta enne 11. septembri rünnakuid USA's - oli üks kõrgelt salastatud sõjaväeline üksus identifitseerinud Mohamed Atta ning veel kolm tulevast kaaperdajat, kui tõenäolised Al Qaeda võrgustiku liikmed Ameerika Ühendriikides. Üksus nimega Able Danger baseerus oma oletustes tugevalt

andmekaevandustehnikatel. 2000 aasta suvel pöördus antud sõjaväeline meeskond kongresmeni ning CIA poole, et oma valduses olevat infot jagada FBI'ga. Nende palve lükati tagasi.

Able Danger meeskond oli koostanud 2000 aasta suvel põranda-suuruse diagrammi kõikvõimalikest Al Qaeda võrgustiku liikmetest ümber maailma. Kõik see toetus erinevatel andmekaevandustehnikatel. Antud diagramm sisaldas praktiliselt kõiki 11. septembi terroristide andmeid ja omavahelisi seoseid, kuid mingil põhjusel ei jõudnud need CIA käest Ühendriikide Julgeolekubürooni.

Üheks sarnaseks jooneks nende terroristide puhul andmekaevanduse seisukohast oli suur hulk krediitkaarte, mitmesajatuhande-dollarilised laenukohustused ning vähene viibimisaeg Ühendriikide pinnal. Keerukaks tegi terroristide avastamise asjaolu, et kasutati erinevaid aliasi ning aadresse (Jehl, 2005).

Terroristide suhtluse jälgimine veebist:
Terrorismirühmitused kasutavad interneti infrastruktuuri, et vahetada informatsiooni ning värvata uusi liikmeid ja toetajaid. Näiteks kasutati kiiret interneti ühendust ka ülalpool kirjeldatud rünnakute organiseerimisel niinimetatud 'Hamburgi haru' poolt (viide artiklile?). See on ka üks suurimaid põhjuseid, miks erinevad riigikaitse agentuurid koguvad internetist terrorismiga seotud tegevuste kohta informatsiooni. Usutakse, et terroristide avastamine võrgu kaudu võib vältida tuleviku rünnakuid. Üks võimalus avastada terroristlikku tegevust on pealt kuulata võrgu liiklust veebilehtedel, mis on seotud mõne vastava organisatsiooniga ning avastada liikmeid IP aadresside järgi. Siiski on see üsnagi keeruline, kuna ei kasutata staatilisi IP-aadresse ega URL'e. Samuti muutuvad tihedalt vastavate lehekülgede hoiustamisserverid, et vältida pealtkuulamist. Et sellest probleemist jagu saada, püüavad riigikaitse organisatsioonid jälgida kogu ISP liiklust (Elovici,Kandel,Last,Shapira,Zaafrany, 2004).

3.7Tootmine

Tootmises on kõige tavalisemad andmekaevanduse kasutusvaldkonnad protsessi juhtimine, vigade ja defektide avastamine ning operatiivkulude vähendamine. Kriitiliste situatsioonideni viinud tingimuste leidmine andmekaevanduse abil aitab inseneridel probleeme ennetada, et vältida tulevasi vigu.

Iga kord, kui kaupa käidelda, kallineb kauba hind. Andmekaevandusvahendeid kasutades saavad tarnijad parandada nende kaupade logistilist liikumist ning seega siis vähendada toodete käitlemise kulusid. Samuti saavad tootjad andmekaevanduse abiga identifitseerida oma kliente piirkonniti ning seda, milliseid tooteid need ostavad. Seesugune teave aitab paremini valida toodete ladustamise kohti, et jaotus nendest tuleks võimalikult odav.

Üks olulisi punkte tootmise juures on ka kvaliteedi kontroll. Andmekaevandustehnikad aitavad identifitseerida defektide tekkimise keskkonda nagu näiteks nädala päev ja kellaaeg, millal tootmine toimus, milliseid komponente kasutati ning kes olid tööl antud hetkel vastaval tootmisliinil. Seesuguste faktorite hindamine ning tootmisprotsessi vajalike muutuste läbiviimine võib oluliselt tõsta toodete kvaliteeti. Kõrge kvaliteet aga aitab jällegi kasvatada müüginumbreid.

3.8Tekstianalüüs, dokumendihaldus

Tekstianalüüsis ning dokumendihalduses on andmekaevanduse peamisteks eesmärkideks dokumentide süntaksi ning semantika analüüs, mürarikaste tekstide analüüs ning ka tekstide automaatne lahterdamine autori ning teema kaupa.

Üheks seesuguseks andmekaevandusmeetodiks on lähima naabri meetod, kus ülesanneteks on kasutajat huvitavale dokumendile sarnaste dokumentide leidmine. Antud meetod aitab leida sarnaseid teoseid, millel on oluline sarnane iseloom. Andmekaevandus võib anda vastuse küsimustele, kas mingid dokumendid on kirjutatud ühe ja sama isiku poolt ning kas mingid dokumendid käsitlevad samu küsimusi ja temaatikat.

3.9Meditsiin

Ka meditsiinis kasutatakse aktiivselt andmekaevandust. Üheks aktiivseks kasutajaks on kindlasti ravimitootmisfirmad, kus see võimaldab efektiivset tulemuste analüüsimist ning suurt kulude kokkuhoidu. Samuti kasutatakse andmekaevandust, et avastada erinevate haiguste põhjuseid ning seoseid.

Siinkohal võib näitena tuua Austraalia tervisekindlustuse komisjoni ootamatu avastuse kahe patoloogilise testi vahel. Patsientidel tehti mõlemaid teste ning kindlustus maksis samuti mõlema testi eest. Andmekaevanduse tulemusena aga selgus, et vaid üks neist testidest oli vajalik, seega organisatsioonil oli võimalik kulude kokkuhoidmiseks üks test ära kaotada (Williams, 2004).

Teise näitena võib tuua Singapuris läbiviidud analüüsid, kus ligikaudu 10% elanikkonnast on diabeetikud. Sellel haigusel võib olla mitmeid kõrvalnähtusid nagu näiteks suurem oht silma-haigustele, maksa alatalitlusele või teistele komplikatsioonidele. Haiguse varajane avastamine ja õigeaegne ravi aitavad neid vältida. Singapuris alustati 1992. aastal diabeetikute aktiivse jälgimise programm, kus salvestati andmebaasi patsientide informatsiooni - kliinilisi sümptomeid, silmahaiguste diagnoose, nende ravi ja muud. Kaheksa aastat hiljem alustati kogu andmehulga peal andmekaevandamist, et leida huvitavaid mustreid. Eesmärgiks oli leida reegleid, mida arstid saaksid oma igapäevases töös ära kasutada ehk siis paremini aru saada diabeedi olemusest ning kuidas haigus on seotud erinevate elanikkonna segmentidega.

Arstid, kes projektis osalesid, leidsid, et paljud avastatud reeglid ja seosed kinnitasid nende varasemaid vaatluseid. Üllatavalt leiti aga ka palju selliseid, millest nad varem teadlikud ei olnud. Andmekaevanduse tulemusena said arstid paremini teada, kuidas diabeet areneb ning kuidas erinevad ravid mõjutavad selle protsessi (Apte,Liu,Pednault,Smyth, 2002).

3.10Militaarvaldkond

Andmekaevandust kasutab ka militaarvaldkond ning seda näiteks assotsatsioonireeglite kaevandamise näol. USA relvajõududes kasutatav automaatne raketitõrjesüsteem Patriot, mis leidis laialdast kasutust Lahesõjas, kasutab assotsiatsioonireeglitel põhinevat otsustussüsteemi. Automaatreziimil suudab see süsteem saamaegselt jälgida sadu lendavaid objekte ning otsustab teatud reeglite põhjal punktiarvestust pidades millist neist objektidest alla tulistada. Näiteks lennumasin, mis tuleb läbi eeldefineeritud «turvakoridori», saab mõned punktid. Kui tema lennutrajektoor pole aga piisavalt paralleelne turvakoridori omaga siis ta kaotab mõned punktid. Kui aga kiirus on teatud piires siis on see hea. Lendobjekt, mis siseneb turvakoridori altpoolt radari nähtavusala (nn pop-up), saab jälle miinuspunkte. Kui tema raadioseadmed ei suuda identifitseerida teda kui sõbralikku objekti, kaotab ta veel mõned punktid. Raketitõrjesüsteem otsustab objekti allatulistamise kasuks kui punktisumma on piisavalt madal.

3.11Sport

NBA katsetab andmekaevandusrakendusi koos video salvestamisega. Vastav tarkvara (The Advanced Scout software) analüüsib mängijate liikumist, et aidata treeneritel kujundada paremaid strateegiaid ning vaatemängulisemaid võistlusi. Tarkvara leiab iga mängija eripärad ning toob ka välja erinevused võrreldes mängijate keskmise tasemega. Kasutades NBA universaalset kella, on võimalik treeneril videopildis välja tuua iga mängija mõni konkreetne hüpe või sööt ilma, et oleks vaja kogu pikka linti läbi vaadata (Palace, 1996).

4.Andmekaevanduses esinevad probleemid

Andmekaevanduses esinevad probleemid ja väljakutsed jagunevad kolmeks peamiseks suunaks:

4.1Kaevandusmeetodite ning kasutajapoolse suhtlusega seotud probleemid.

Erisuguste teadmiste kaevandamine andmebaasist

Kuna erinevad kasutajad võivad olla huvitatud erisugustest teadmistest, siis peaks andmekaevandus katma küllaltki laia meetodite kasutuse, nende hulgas assotsiatsioonireeglite leidmise, klassifitseerimise, sarnasuse analüüsi ning ka klasterdamise. Kõik need meetodit võivad kasutada sama andmebaasi erinevat moodi ning saada ka erinevaid tulemusi (Han,Kamber, 2002).

Interaktiivne teadmiste kaevandamine

Kuna on üsnagi raske ette teada, mida andmebaasist leida võib, peaks kogu protsess olema interaktiivne. Suurte andmehulkade puhul võib kasutada sobivat näidismudelite leidmise tehnikat, et lihtsustada andmete uurimist. Interaktiivne kaevandamine võimaldab kasutajal keskenduda mustrite otsimisele, töödeldes andkekaevanduspäringuid edasi vastavalt saadud tulemustele. Teadmusi peaks ammutama ka samalaadselt OLAP tehnoloogiale, kasutades sealjuures ka erinevaid *pivot* meetodeid. Sedaviisi saab kasutaja suhelda andmekaevandussüsteemiga, vaadates avastatud mustreid erinevate vaatenurkade alt (Han,Kamber, 2002).

Eelteadmised ning taustainformatsioon

Andmete tausta ja seoste tundmaõppimine võimaldab paremini juhtida teadmushõive protsessi. Baasteadmised andmebaasist, nagu näiteks piirangud ning erinevad reeglid, aitavad paremini fokusseeruda ning kiirendada andmekaevandusprotsessi. Samuti aitavad need hinnata avastatud mustrite huvitavavust ning headust (Han,Kamber, 2002).

Tulemuste presentatsioon ja visualiseerimine

Avastatud teadmusi tuleks väljendada mõnes kõrgtaseme keeles, visuaalsel esitlusel või mõnel muul väljendaval viisil, et kaevandatud teadmised oleksid inimestele kergesti ja üheselt mõistetavad. See on just eriti oluline interaktiivse andmekaevandussüsteemi puhul. Süsteem peaks olema võimeline esitama teadmusi puudena, tabelitena, reeglitena, joonistena, risttabelitena või mõne muu samalaadse vahendi abil (Han,Kamber, 2002).

Müra ning puuduvad väärtused

Andmebaasid võivad üldjuhul salvestada lisaks terviklikele andmetele ka kõikvõimalikku müra, erandjuhte ning poolikuid andmeid. Seesugused objektid võivad ajada segadusse analüüsi protsessi, põhjustades näiteks ülekattumist koostatavate andmemudelite puhul. Tulemuseks on vähene täpsus avastatud mustrites. Kuigi enamus meetodeid eirab seesuguseid andmeid, võivad need aga olla teinekord just huvitavad ja vajalikud, kasvõi näiteks pettuste avastamsel (Han,Kamber, 2002). Probleemiks on ka puuduvad andmed, mis esineb just tihtipeale äriettevõtete andmebaasides ning tekib andmebaasi pideva arendamise ja muutmise tulemusena või siis tarkvara vigade tõttu. Puuduvad andmed võivad põhjustada tõrkeid andmekaevandamisalgoritmides (Smyth,Fayyad,Piatetsky-Shapiro, 1996).

Avastatud mustrite hindamise probleem.

Andmekaevandussüsteem võib avastada tuhandeid mustreid. Paljud nendest mustritest on kasutaja jaoks täiesti ebahuvitavad, esitades niigi teadaolevaid teadmisi või siis puudub nendes uudsus, samuti ei pruugi avastatud mustrid olla huvitavad või vajalikud. Hoolimata erinevate tehnikate arengust, jääb see probleem siiski püsima (Han,Kamber, 2002). Lisaks sellele on veel tihtipeale üsna keeruline teha avastatud mustrid inimkasutajale arusaadavaks (Smyth,Fayyad,Piatetsky-Shapiro, 1996).

4.2Jõudluse probleemid

Andmekaevandamisalgoritmide tõhusus ja mõõdetavus.
Et ammutada edukalt teadmisi suurtest andmehulkadest, peavad andmekaevandamisalgoritmid olema vägagi tõhusad ja mõõdetavad – algoritmi töötamisaega peab olema ennustatav ning vastuvõetav suurte andmebaaside puhul. Eksponentsiaalse või isegi keskmise keerukusega algoritmid jäävad üldjuhul praktilisest kasutusest välja. Andmebaasi seisukohalt on need kaks asja võtmeküsimusteks andmekaevandussüsteemi kasutuselevõtmise juures (Smyth,Fayyad,Piatetsky-Shapiro, 1996).

Paralleelsed, jaotatud ning juurdekasvu algoritmid.
Andmebaasite tohutu suurus, andmete laialdane jagunemine ning mõningate andmekaevandusmeetodite arvutuslik keerukus on faktorid, mis motiveerivad paralleelsete ning jaotatud andmete kaevandamisalgoritmide arendamist. Seesugused algoritmid jaotavad andmed partitsioonideks ning neid töödeldakse paralleelselt ning seerjärel tulemused ühendatakse. Paljude andmekaevandamisprotsesside ajaline keerukus ajendab ka juurdekasvualgoritmide väljatöötamist ning samuti on probleemiks dünaamilised andmebaasid, kus andmeid pidevalt muudetakse või kustutatakse. Seesugused algoritmid ei pea kogu kaevandusprotsessi viima iga kord läbi nullist, vaid teadmiste modifitseerimine toimub järk-järgult uute andmete lisandumisel (Han,Kamber, 2002).

Tööjõu probleemid.
Andmekaevandustarkvara ei asenda oskustöölisi. Kuigi andmekaevandamisalgoritmid on vägagi võimsad, võivad väheste teadmistega inimesed saada väärasid või kasutuid tulemusi. Samuti peab meeskond olema pühendunud tulemustele. Meeskonna suurus võib varieeruda mitmesugustel faktoritel ning lisaks ei ole andmekaevandus ainult ühekordne tegevus, vaid peab toimuma regulaarselt koos andmete ja keskkonna muutustega. Kogu teadmushõive protsess on pidev ning kestvat täiendamist ja õppimist.

4.3Probleemid, mis on seotud andmete mitmekesisusega

Toimetulek relatsiooniliste ning keeruliste andmetüüpidega.
Andmebaasides salvestatakse mitmesuguseid andmeid. Seoses relatsiooniliste andmebaaside ning andmeladude laialdase kasutamisega, peavad efektiivsed andmekaevandussüsteemid tulema toime kõikvõimalike andmetega. Siiski võivad andmebaasid sisaldada ka seesuguseid keerukaid andmetüüpe nagu hüpertekst, multimeedia, graafika, transaktsioonilised andmed ja veel teisigi andmeliike. Seetõttu on ka üsnagi ebareaalne arvata, et üks andmekaevandamissüsteem suudab nende kõigiga toime tulla, kuna andmetüüpe on palju ning ka kaevanduseesmärgid erinevad. Seetõttu on ka erisuguste andmetüüpide jaoks erinevad andmekaevandusprogrammid (Han,Kamber, 2002).

Kaevandamine heterogeensetest ja globaalsetest andmebaasidest.
Kohalikud võrgud ning internet võimaldavad ühendada omavahel mitmeid andmeallikaid, moodustades niimoodi hiiglaslikke, jaotatud ning heterogeenseid andmebaase. Teadmushõive erinevatest allikatest, kus andmed on pooleldi struktureeritud või täiesti struktureerimata, on üks andmekaevanduse suurimaid väljakutseid. Andmekaevandus võimaldab avaldada andmete regulaarsust ning mudeleid mitmetest heterogeensetest andmebaasidest, mis lihtsate päringusüsteemidega oleks ilmselgelt võimatu (Smyth,Fayyad,Piatetsky-Shapiro, 1996).

5. Andmekaevanduskeskkoond YALE

Järgnevas peatükis vaatleme ühte vabavaralist andmekaevanduskeskkonda YALE (*Yet Another Learing Environment*), proovime andmete importimist ja puhastamist ning katsetame ka erinevaid andmekaevandusmeetodeid. Antud peatüki eesmärk on katsetada andmekaevandamist valdkonnavõõra inimese poolt.

5.1 Ülevaade andmekaevanduskeskkonnast YALE

YALE on masinõppimise eksperimentide ning andmekaevanduse keskkond. Modulaarne ehitus võimaldab liita omavahel erinevaid operaatoreid, et viia läbi hulgaliselt õppimisprobleeme ning eksperimente. Andmete kasutamine on operaatorite jaoks nähtamatu ning need ei pea tulema toime tegeliku andmete formaadiga vaid keskkond suudab ise läbi viia vajaliku transformatsiooni. YALE kasutavad küllaltki paljud andmekaevandusfirmad ning andmekaevanduse arendajad.

Yale võimalused ja vahendid:

Sisend ja väljund – paindlikud operaatorid erinevate sisend ja väljund failiformaatidele, sealhulgas:

- tuntuimad andmekaevandus ja -õppimis skeemiformaadid (Arff, C4.5, csv, ...)
- hajusad failiformaadid (SVMlight, mySVM, ...)
- andmehulgad andmebaasidest (Oracle, mySQL, PostgreSQL, ...)
- dBase
- teksti ning audiofailid
- veel...

Masinõppimise algoritmid: enam kui 100 õppimisskeemi klassifitseerimiseks, klasterdamiseks ning paljudeks teisteks meetoditeks:

- *Support Vector Machines* (LibSVM, SMO, mySVM, ...)
- otsustuspuud ning reeglite õppijad (ID3, C4.5, PART, PRISM, ...)
- laisad õppijad (lähima naabri algoritmid, K*, LBR, ...)
- Bayes'i õppijad (naïve Bayes, Bayes Net, AODE, ...)
- logistilised õppijad (Logistic Regression, SimpleLogistic, ...)
- Gaussian Processes – gaussi protsessid?
- metaõppimine (AdaBoost, Bagging, Stacking, BayesianBoosting, ...)
- assotsiatsioonireeglite kaevandamine (Apriori, Tertius, ...)
- klasteranalüüs (eraldi plugin: k-Means, k-Medoids, Dbscan, SVClustering, ...)
- veel..

Weka operaatorid: kõiki Weka õppimiskeskkonna skeeme ja atribuutide hindamisvahendid on samuti võimalik antud keskkonnas kasutada.

Andmete eeltöötlemine: operaatorid, mida tihtipeale kasutatakse enne õppimisprotsessi alustamist:
- diskretiseerimine (eraldamine?) (Binning, Frequency, ...)
- näidiste ja tunnuste filtreerimine (Conditionen, ValueTypeFilter, ...)
- normaliseerimine (Interval, Standardization, z-Transformation, ...)
- valimid (Simple, ModelBased, ...)
- puuduvate ja lõpmatute andmete asendamine ja täiendamine
- ebavajalike vahendite eemaldamine
- veel...

Tunnuste operaatorid:
- tunnuste valik (Forward Selection, Backward Elimination, Genetic Algorithms, ...)
- tunnuste kaalumine ja olulisus (ChiSquared, Correlation, InfoGain, RelieF, ...)
- tunnuste koostamine (GGA, YAGGA, ...)
- veel...

Jõudluse hindamine: mitmed valideerimise ja hindamise skeemid, et hinnata andmehulga õppimise ja eeltöötlemise jõudlust/sooritust:
- rist-valideerimine (stratified, shuffled, non-shuffled, ...)
- treenimis- ning testandmehulga eraldamine (random, fixed, ...)
- olulisuse testid (ANOVA, t-Test, ...)
- veel...

Visualiseerimine: tulemuste logimine ning esitamine:
- online 1D, 2D ja 3D andmekaevanduse ja eksperimentide tulemuste joonised
- sisseehitatud histogrammid, jaotuse ning värvilised joonised
- *Learned Models* – puu vaade, klastermudelite graafikud
- SVM funktsioonid
- veel...

5.2Installeerimine ja käivitamine

YALE installeerimine on küllaltki lihtne, tuleb vaid fail alla laadida ning sobivasse kohta lahti pakkida kasutades näiteks WinZip'i või mõnda muud samalaadset tarkvara. Faili saab alla laadida aadressilt: http://www-ai.cs.uni-dortmund.de/SOFTWARE/YALE/download.html ning sealt tuleks valida link *Download YALE (SourceForge)*, failiks viimane versioon *yale-3.2-bin.tar.gz* ning edasi valida endale lähim allalaadimispaik.

Kuna YALE töötab Java keskkonnas, tuleks arvutisse installeerida ka ***Java Runtime Environment (JRE)*** version 1.5 (või uuem), mille saab SUN'i kodulehelt: http://java.sun.com/ .

Käivitamiseks Windows keskkonnas on mitu võimalust, kas topelt-klikkides *yale.jar* failil (mis ei ole aga üldjuhul soovitatav) või siis käsurealt, mida võib näha jooniselt 5.2.1.

Joonis 5.2.1. JALE käivitamine käsurealt.

5.3Andmed katsetamiseks.

Et läbi viia mõned lihtsamad katsed ja näited, on meil loomulikult tarvis selleks sobivat andmehulka. Vaatleme ühe ettevõtte laoseisu lihtsustatud ja vähendatud aruannet (joonis 5.3.1), mis koosneb väljadest: alusetyyp(millisel alusetüübil kaupa hoiustatakse), kaste(palju oli kaste alusel lattu saabudes), aluse staatus (kas alus on reservis, komplekteerimisel või välja viidud), sissetulek (kas kaup on tulnud sisse alusel või on saanud endale aluse alles laos), kaubatyyp (standardkaaluline kaup või erikaaluline kaup), yhik (kauba ühiku tüüp), sissetulek (mis kuus on kaup sisse tulnud).

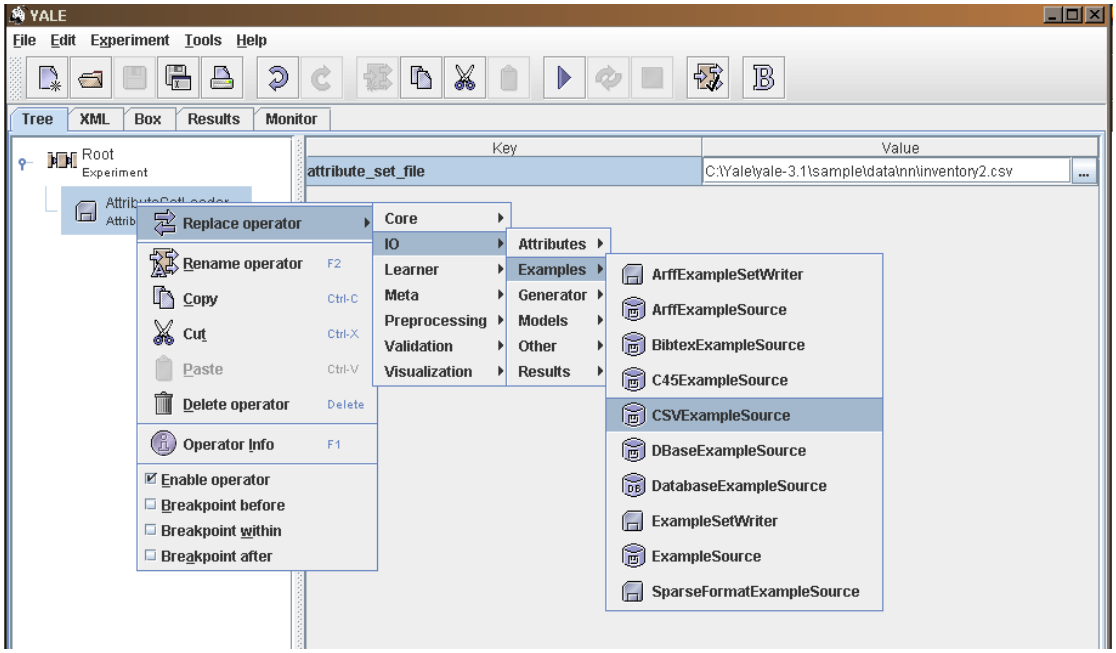
R...	alusetyyp ...	kaste (int4)	staatus (text)	sissetulek (...	kaubatyyp ...	yhik (text)	sissetuleku...
1	EUR	50	komplekteerimisel	alusega	standardkaal	KG	oktoober
2	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
3	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
4	EUR	35	komplekteerimisel	alusega	standardkaal	KG	oktoober
5	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
6	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
7	EUR	41	välja viidud	alusega	standardkaal	KG	september
8	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
9	EUR	51	välja viidud	alusega	standardkaal	KG	mai
10	EUR	51	komplekteerimisel	alusega	standardkaal	KG	mai
11	EUR	51	välja viidud	alusega	standardkaal	KG	mai
12	EUR	32	komplekteerimisel	alusega	standardkaal	KG	juuni
13	EUR	19	komplekteerimisel	alusega	standardkaal	KG	november
14	EUR	5	komplekteerimisel	alusega	standardkaal	KG	mai
15	EUR	17	komplekteerimisel	alusega	standardkaal	KG	veebruar
16	EUR	48	välja viidud	alusega	standardkaal	KG	november
17	EUR	24	reservis	alusega	standardkaal	KG	september

Joonis 5.3.1. Ettevõtte laoseisu aruanne.

Katsetamisandmehulga suuruseks on 1000 rida, mis peaks olema piisav lihtsamate andmekaevandusmeetodite katsetamiseks. Samuti on eemaldatud müra ning vigased ja puuduvad andmed. Järgmiseks sammuks on andmete väljastamine CSV formaati, et oleks võimalik samade andmetega sooritada erinevaid katseid, kuna on tegemist üsnagi muutuvate kirjetüüpidega – andmed võivad ajas muutuda.

5.4Andmete importimine

Et importida andmed CSV failist YALE keskkonda, tuleks teha *Root* operaatoril vajutus hiire paremal klahvil. Edasi valida *Replace operator – IO – Examples* ning CSV formaadi jaoks *CSVExampleSource*.



Joonis 5.4.1. Andmete importimine CSV formaadist.

Järgnevalt tuleks määrata ära andmefaili asukoht ning ka peamine atribuut, mis antud näite puhul on *alusetyyp*. Edasi vajutada menüüribas olevale sinisele noolekesele ning faili lugemise õnnestumise puhul väljastatakse kasutajale loetud andmete üldine kirjeldus koos statistikaga (joonis 5.4.2). Kui andmed on edukalt sisse loetud, siis tuleks edasi liikuda andmete eeltöötlemise, algoritmide kasutamise ning saadud tulemuste hindamiseni.

SimpleExampleSet

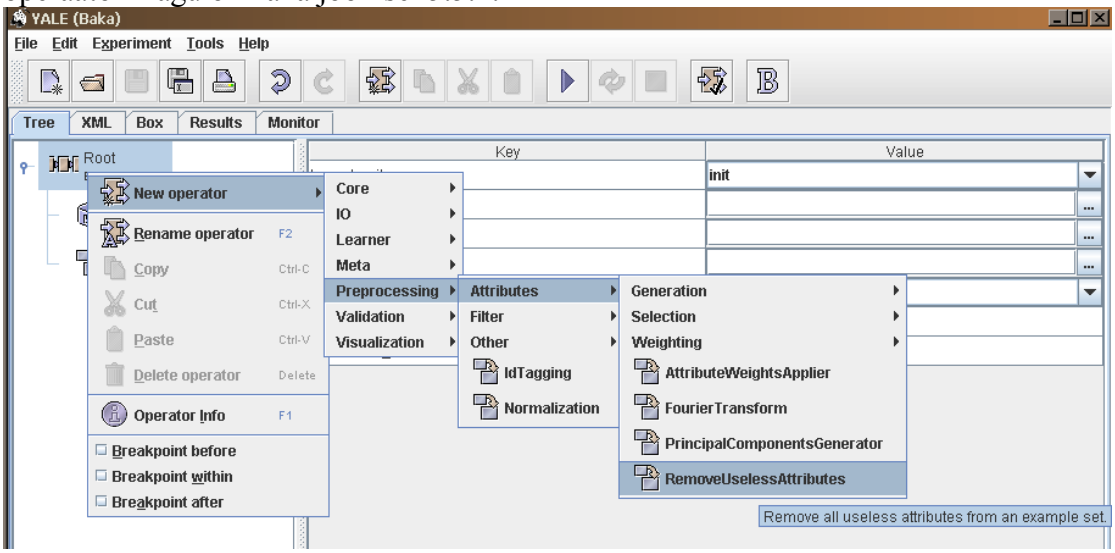
Number of examples: 1000
Number of attributes: 6

Index	Name	Generated from	Unit	Type	Blocktype	Blocknr.	Values
Regular attributes							
1	kaste	kaste		real	single_value	-	avg = 49.113 +/- 25.921; range = [0.000 ; 205.000]
2	staatus	staatus		nominal	single_value	-	mode = välja_viidud; komplekteerimisel (296), reser
3	sissetulek	sissetulek		nominal	single_value	-	mode = alusega; alusega (1000)
4	kaubatyyp	kaubatyyp		nominal	single_value	-	mode = standardkaal; erikaal (260), standardkaal (7
5	yhik	yhik		nominal	single_value	-	mode = KG; KG (847), PCS (153)
6	sissetulekuaeg	sissetulekuaeg		nominal	single_value	-	mode = november; aprill (34), august (21), detsemb
id							
0	alusetyyp	alusetyyp		nominal	single_value	-	mode = EUR; EUR (679), FIN (321)

Joonis 5.4.2. Andmed CSV failiformaadist.

5.5Andmete eeltöötlemine

Kuigi enne andmete importimist olid kõrvaldatud puuduvad ning vigased kirjed ja suurem eeltöötlus teostatud, tuleks siiski andmeid kontrollida ka andmekaevandustarkvara vahenditega, juhuks kui meil jäi midagi märkamata või ei ole teostatud eeltöötlus piisav. Kuna tihtipeale on meil atribuutide hulk küllaltki suur ning tulemused seetõttu küllalki keerukad ja kirjus, siis võiksime eemaldada kõik mittevajalikud atribuudid. Selleks lisame juuroperaatoril parema kliki abil uue operaatori nagu on näha joonisel 5.5.1.



Joonis 5.5.1. Mittevajalike atribuutide eemaldamine.

Kuna atribuudid "kaste" ja "staatus" osutusid mittevajalikeks, siis eemaldati need programmi poolt meie testandmetest (joonis 5.5.2). Samas ei pruugi olla kõik tarkvara poolt tehtavad otsused meie omadega kooskõlas ning vajadusel saab tehtud toiminguid ka tühistada. Kuna aga antud juhul meid huvitavad nii kastide hulk kui ka aluste staatus, siis tühistame mittevajalike atribuutide eemaldamise.

SimpleExampleSet

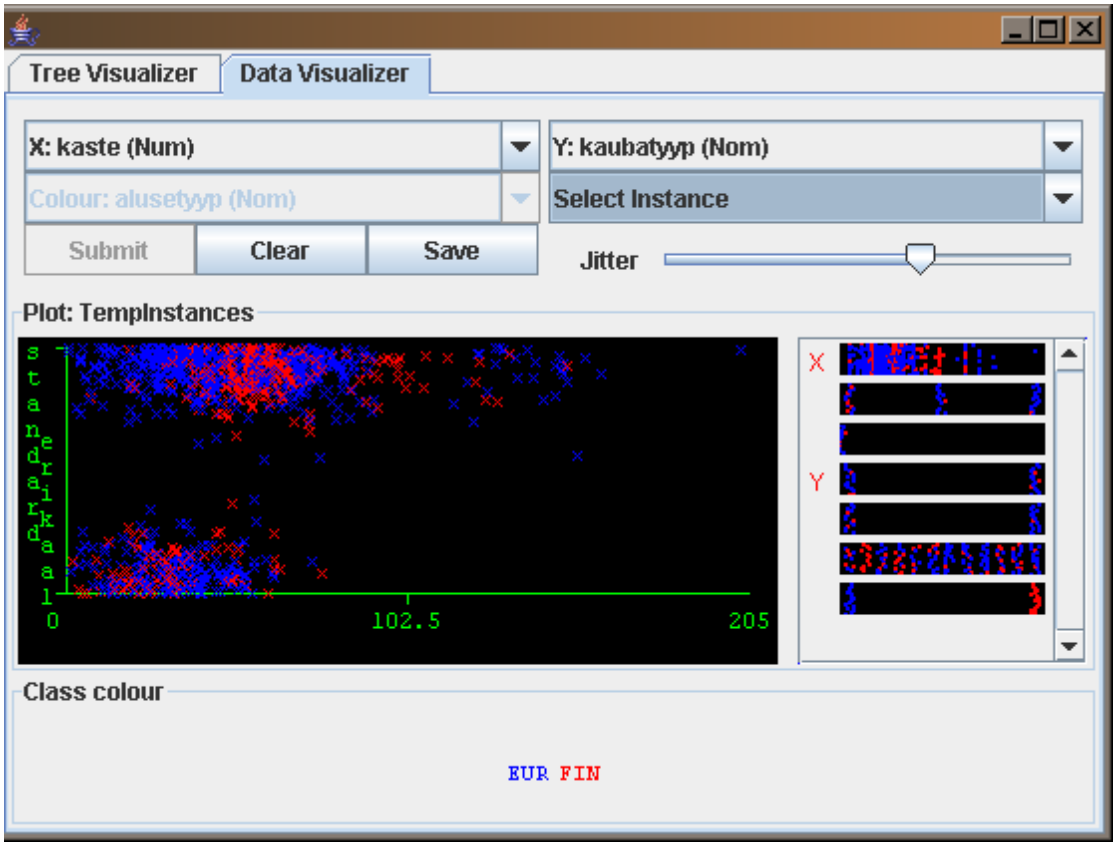
Number of examples: 1000
Number of attributes: 4

Index	Name	Generated from	Unit	Type	Blocktype	Blocknr.	Values
Regular attributes							
3	sissetulek	sissetulek		nominal	single_value	-	mode = alusega; alusega (1000)
4	kaubatyyp	kaubatyyp		nominal	single_value	-	mode = standardkaal; erikaal (260), standardkaal (7
5	yhik	yhik		nominal	single_value	-	mode = KG; KG (847), PCS (153)
6	sissetulekuaeg	sissetulekuaeg		nominal	single_value	-	mode = november; aprill (34), august (21), detsemb
id							
0	alusetyyp	alusetyyp		nominal	single_value	-	mode = EUR; EUR (679), FIN (321)

Joonis 5.5.2. Testandmehulk peale mittevajalike atribuutide eemaldamist.

5.6Õppimisalgoritmide kasutamine

Järgnevaks katsetame mõnda juhuslikult valitud algoritmi testandmehulga peal ning toome välja kuvapildistused tulemustest. Valitud algoritmideks osutusid *UserClassifier* (joonis 5.6.1), *RandomTree* (joonis 5.6.2), *OneR* (joonis 5.6.3) ja *DefaultLearner* (joonis 5.6.4).



Joonis 5.6.1. Algoritm *UserClassifier* koos andmete visualiseerimisega.

```
RandomTree
==========

sissetulekuaeg = aprill
|   yhik = KG
|   |   kaste < 47.5 : EUR (10/0)
|   |   kaste >= 47.5 : FIN (15/0)
|   yhik = PCS : EUR (9/0)
sissetulekuaeg = august
|   staatus = komplekteerimisel
|   |   kaubatyyp = erikaal : EUR (1/0)
|   |   kaubatyyp = standardkaal
|   |   |   kaste < 22.5 : EUR (1/0)
|   |   |   kaste >= 22.5 : FIN (14/0)
|   staatus = reservis : EUR (0/0)
|   staatus = välja_viidud : FIN (5/0)
sissetulekuaeg = detsember
```

Joonis 5.6.2 Algoritm *RandomTree*.

```
kaste:
        < 24.5  -> EUR
        < 25.5  -> FIN
        < 48.5  -> EUR
        < 50.5  -> FIN
        < 54.5  -> EUR
        < 55.5  -> FIN
        < 73.0  -> EUR
        < 82.0  -> FIN
        < 84.5  -> EUR
        < 112.5 -> FIN
        < 129.0 -> EUR
        < 134.0 -> FIN
        >= 134.0        -> EUR
 (927/1000 instances correct)
```

Joonis 5.6.3. Algoritm *OneR*.

```
DefaultModel

Model (type DefaultModel) for label #0: alusetyyp:=alusetyyp: nominal/single_value/no block/values=[EUR, FIN]
default value: EUR
```

Joonis 5.6.4. Algoritm *DefaultLearner*

5.7Tulemused/järeldused

Andekaevandusprotsess eeldab analüütikult tugevaid teadmisi andmete valdkonnast aga ka põhjalikku tarkvara tundmist. Antud katse puhul puudusid aga mõlemad vajalikud eeldused ning eesmärk oligi andmekaevanduse katsetamine valdkonnavõõrale inimesele.

Algoritmi UserClassifier abil võime näha erinevatel alusetüüpidel olevaid koguseid kaubatüüpide (standardkaal ja erikaal) lõikes. Antud kuvapildistusest (joonis 5.6.1) võib näha, et keskmistest kogustest on kõrvalekaldeid pigem standardkaalu kaupadel.

Algoritm RandomTree (joonis 5.6.2) kuvab meile otustuspuu ning selle järgi saab määrata atribuutide olulisust – mida lähemal juurele, seda olulisem atribuut. Saadud tulemuste huvitatavast on aga raske hinnata, tundmata andmete olemust ja iseloomu.

Kaks viimast algoritmi – OneR (joonis 5.6.3) ja DefaultLearner (joonis 5.6.4) ei andnud meile aga paraku mingisugust huvitavat ja kasulikku informatsiooni.

Antud eksperimendi tulemusena võib öelda, et analüütiku roll andmekaevandamise protsessi juures on küllalki oluline ning tulemused võivad paljuski sõltuda analüütikust – tema oskustest ning saadud tulemuste huvitatavuse hindamisest.

Kokkuvõte

Andmekaevandus on riistvarast, tarkvarast, oskustööjõust ning andmetest koosnevast võimalusest ära tunda tundmatuid kuid potensiaalselt kasulikke seoseid. See toetab andmete transformeerimist informatsiooniks, teadmisteks ning tarkuseks. Tsükkel, mis peaks toimima igas organisatsioonis (Labovitz, 2003).

"Andmekaevandusele" võibolla sobivam ja täpsem nimetus oleks "Teadmiste kaevandamine andmetest", kuid see on pisut pikk ja ebamugav. Samuti lühike termin "Teadmiste kaevandamine" ei peegeldaks oma olemust – kaevandamine suurtest andmehulkadest. Siiski, "kaevandamine" on piisavalt ilmekas termin iseloomustamaks protsessi, mis leiab ülesse väikese koguse "kulda" suurest hulgast toorest materjalist. Seega muutus populaarseks väljendiks antud tegevuse kohta sõna andmekaevandus. Tihtipeale kiputakse pidama termineid "andmekaevandus" ning "teadmushõive andmebaasidest" sünonüümideks. Tegelikkuses vaadeldakse aga andmekaevandust teadmushõive ühe olulise etapina.

Andmekaevandus sisaldab endas kombinatsiooni mitmetest tehnikatest nagu andmebaasi tehnoloogia, statistika, masinõppimine, mustrite äratundmine, andmete visualiseerimine, informatsiooni kogumine, piltide ja signaalide töötlemine ning ruumiline andmeanalüüs. Andmekaevandamisega ei tegele pelgalt arvutid ning algoritmid. Andmekaevanduses on siiski esmatähtis roll inimesel ning eelmainitud on vaid abivahendid, kuna inimene üksi ei ole võimeline töötlema tohutuid andmehulkasid. Kuna arvutitel puudub loovus, siis andmekaevandusprotsesside automatiseerimine ei pruugi anda alati soovitud tulemusi.

Andmekaevanduses esineb üsnagi palju erinevaid ülesandeid ja küsimusi, mida tuleb hakata lahendama või siis uurima. Igal ülesandel on tavaliselt oma eeldused ja nõudmised ning lahenduste tulemused on tihtipeale erinevad. Nende lahendamiseks on mitmeid algoritme ja tehnikaid.

Andmekaevanduse esialgsed ning algelised rakendused said alguse eelkõige panganduse, kindlustuse ning telekommunikatsiooni valdkondades, kuid on tänapäeval juba laienenud praktiliselt kõikidesse eluvaldkondadesse. Ettevõtetele, kes on nõus investeerima nii finantsiliselt, tehniliselt kui ka kvalifitseeritud tööjõu näol andmekaevandusse, võib antud protsess anda küllaltki suure konkurentsieelise.

Probleemid ja väljakutsed andmekavenduses jagunevad kolmeks peamiseks grupiks: kaevandusmeetodite ning kasutajapoolse suhtlusega seotud probleemid, jõudluse probleemid ning probleemid, mis on seotud andmete mitmekesisusega.

Töö käigus valmis küllaltki ülevaatlik materjal andmekaevanduse protsessist ning meetoditest. Samuti on välja toodud olulisimad kasutusvaldkonnad koos mõningaste näitedega, et pisut lähemalt välja tuua andmekaevanduse eesmärki. Seega võib öelda, et käesolevas töös on seatud eesmärgid täidetud.

Summary

Data mining is the task of discovering interesting patterns from large amounts of data where the data can be stored in databases, data warehouses, or other information repositories. In a simple way - data mining refers to extracting or knowledge from large amounts of data.
The major reason that data mining has become so popular in recent years is because of the wide availability of huge amounts of data and the imminent need for turning such data into useful information and knowledge. This knowledge can be used for applications from business management, production control and market analysis, to engineering design and science exploration.
Many people treat "data mining" as a synonym for "Knowledge Discovery in Databases", commonly it is viewed as simply an essential step in the process of knowledge discovery in databases.
Data mining involves an integration of techniques from database technology, statistics, machine learning, high performance computing, pattern recognition, data visualization, information retrieval, image and signal processing.
The main issues in data mining are divided into three main groups: Mining methodology and user-interaction issues, Performance issues and Issues relating to the diversity of database types.

The aim of the BA thesis is to introduce the Data Mining basic concepts and methods to those who are not familiar to this field. The motivation of this paper was to expand the IT materials in Estonian, especially in the Data Mining field. Another purpose was to show where the the Data Mining can be used and to bring out some examples.

The chapters of the BA thesis are divided into five main sections. The first chapter describes the concepts of the Data Mining, database systems' history and gives the idea of the term "Knowledge Discovery in Databases". Second chapter brings out the main methods of Data Mining and the third one shows the areas of Data Mining use with some examples. In the fourth chapter there is an overview of the main Data Mining issues. The final chapter is an introduction to the free Data Mining environment - YALE (Yet Another Learning Environment).

Viidatud allikad

I.**Liiv**, **2005**,
"Andmekaevandamine"

A.**Tiidumaa, 2003**,
"Assotsiatsioonireeglite leidmine suurtest andmehulkadest"
http://www.egeen.ee/u/vilo/edu/2003-04/DM_seminar_2003_II/Raport/P01/main.pdf

A.**Berson**, S.**Smith**, K.**Thearling, 1999**,
"Building Data Mining Applications for CRM"

C.**Apte**,B.**Liu**,E.**Pednault**,P.**Smyth**, **2002,**
"Business Applications of Data Mining"
http://www.research.ibm.com/dar/papers/pdf/business_applications_of_dm.pdf

H.**Blockeel**, L.**Dehaspe**; **2000,**
"Cumulativity as Inductive Bias"
http://eric.univ-lyon2.fr/~pkdd2000/Download/WS1_06.pdf

Alex A. **Freitas 2002,**
"Data Mining and Knowledge Discovery with Evolutionary Algorithms"

J.**Han** and M.**Kamber**; **2000,**
"Data Mining: Concepts and techniques"

G.**Williams, 2004,**
"Data Mining Desktop Survival Guide"
http://www.togaware.com/datamining/survivor/Other_Examples0.html

Gary M **Weiss**, 2004
"Data Mining in Telecommunications"
http://storm.cis.fordham.edu/~gweiss/papers/kluwer04-telecom.pdf

I.H.**Written**,E.**Frank**, **2000,**
"Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations"

D.J.**Hand**, **1998,**
"Data Mining: Statistics and More?"

B.**Palace, 1996,**
"Data Mining: What is Data Mining"
http://www.anderson.ucla.edu/faculty/jason.frand/teacher/technologies/palace/datamining.htm

J.R.**Quinlan**; **1987,**
"Decision trees as probabilistic classifiers"

Larissa T. **Moss, 2003,**
Defining Data Mining
http://www.businessintelligence.com/ex/asp/code.64/xe/article.htm

Susan P. I**mberman, 2001,**
"Effective Use Of The KDD Process and Data Mining For Computer Performance Professionals"
http://www.cs.csi.cuny.edu/~imberman/papers/6156.pdf

D.**West**, **2005,**
"Enhancing Value Through Data Mining"
http://www.insurancenetworking.com/protected/article.cfm?articleId=3631&searchTerm=SAS

Douglas **Jehl, 2005,**
"Four in 9/11 Plot Are Called Tied to Qaeda in '00"
http://www.nytimes.com/2005/08/09/politics/09intel.html?ex=1281240000&en=fde2e84ba45fe8ad&ei=5088&partner=rssnyt&emc=rss.html

U.**Fayyad**, G.**Piatetsky-Shapiro,** P.**Smyth, 1996**
"From Data Mining to Knowledge Discovery in Databases"

U.**Fayyad**, G.**Grinstein**, A.**Wierse, 2002,**
"Information Visualization in data mining"

R.H. **Zamar**, **2003**
"Intro to Data Mining"
http://ugrad.stat.ubc.ca/~stat447j/resources/

M.**Khan**, Q.**Ding**, W.**Perrizo, 2002,**
"k-Nearest Neighbor Classification on Spatial Data Streams Using P-Trees"
http://cs.hbg.psu.edu/~ding/publications/PAKDD02_KNN.pdf

M.**Juhkam, 2004,**
"Klasterdamine andmekaevanduses"
http://www.egeen.ee/u/vilo/edu/2003-04/DM_seminar_2003_II/Raport/P05/main.pdf

T.**Michell, 1996,**
"Machine Learning"

K.**Käärmann, 2003,**
"Otsustuspuudega klassifitseerimine"
http://www.egeen.ee/u/vilo/edu/2003-04/DM_seminar_2003_II/Raport/P02/main.pdf

D.**Hand**, H.**Mannila**, P.**Smyth, 2001,**
"Principles of Data Mining"

P.**Smyth**, R.M.**Goodman, 1991,**
"Rule Induction using Information Theory"
S.M. Aqil **Burney**, A.**Manzoor**, F.**Burney, 2003,**
"Similarities and Differences in Statistics and Data Mining"

P.**Smyth,** U.**Fayyad**, G.**Piatetsky-Shapiro**; **1996** ,
"The KDD Process for Extracting Useful Knowledge from Volumes of Data"

Y.**Elovici**,A.**Kandel**,M.**Last**,B.**Shapira**,O.**Zaafrany**, **2004**
Using Data Mining Techniques for Detecting Terror-Related Activities on the Web
http://www.ise.bgu.ac.il/faculty/mlast/papers/JIW_Paper.pdf

N.**Dung Duc; 2003,**
"Using Prior Knowledge in Rule Induction"

D.**Ler; 2004**;
"Utilising Multiple Inductive Bias in Classification: A Survey on Ensembles and Meta-learning"
http://www.it.usyd.edu.au/research/tr/tr558.pdf

Mark L. Labovitz, 2003;
"What Is Data Mining and What Are Its Uses?"
http://www.darwinmag.com/read/100103/mining.html