Suvi Kaasalainen

Emulsioräjähdysaineen panostusprosessin tiedonlouhinta

Tietotekniikan pro gradu -tutkielma

1. joulukuuta 2021

Jyväskylän yliopisto

Informaatioteknologian tiedekunta

Tekijä: Suvi Kaasalainen

Yhteystiedot: suvi.t.kaasalainen@student.jyu.fi

Ohjaajat: Joonas Hämäläinen ja Tommi Kärkkäinen

Työn nimi: Emulsioräjähdysaineen panostusprosessin tiedonlouhinta

Title in English: Knowledge Discovery from Emulsion Explosives Charging Process

Työ: Pro gradu -tutkielma

Opintosuunta: Ohjelmisto- ja tietoliikennetekniikka

Sivumäärä: 66+16

Tiivistelmä: Tämän tutkielman tavoitteena oli löytää uutta ja hyödyllistä tietoa emulsioräjähdysaineen panostusprosessista hyödyntäen tiedonlouhinnan menetelmiä. Aineistona oli panostusyksiköistä kerätty moniulotteinen aikasarjadata. Tutkielmassa käytiin läpi kohdealueeseen ja aikasarjojen erityispirteisiin sopivia ohjaamattoman oppimisen menetelmiä. Tiedonlouhinnan menetelmiksi valittiin kontekstuaalinen matriisiprofiili ja sen soveltaminen klusterointiin ja poikkeavuuksien havaitsemiseen seuraten Knowledge Discovery in Databases (KDD) -prosessia. Menetelmien avulla datasta löydettiin poikkeavuuksia. Tulosten avulla pyritään parantamaan panostusprosessin laatua sekä tarjoamaan tarkempaa tietoa asiakkaille.

Avainsanat: tiedonlouhinta, koneoppiminen, aikasarjat, räjähdysaineet

Abstract: The aim of this theses was to find novel and useful information from emulsion explosives charging process. Multivariate time series data was collected from charging units. Suitable unsupervised machine learning methods for times series data were discussed. Data mining methods used were contextual matrix profile applied to clustering and anomaly detection following the steps of Knowledge Discovery in Databases (KDD) process. Anomalies and discords were found as a result. Results and information are to be used to improve the quality of the charging process and provide more detailed information to customers.

Keywords: data mining, machine learning, time series, explosives

Esipuhe

Haluan kiittää Forcit Oy:tä mahdollisuudesta päästä toteuttamaan tutkielma työelämälähtöisen käytännön tutkimusongelman muodossa. Kohdealueeseen tutustuminen oli antoisaa ja opettavaista. Erityinen kiitos Forcit Oy:n Jami Kangasojalle kärsivällisyydestä ja asiantuntevasta ohjauksesta.

Kiitos myös läheisilleni tuesta koko opiskelujen aikana.

Jyväskylässä 1.12.2021

Suvi Kaasalainen

Kuviot

Taulukot

Sisältö

1 Johdanto

Kerättävän datan volyymi valmistavassa teollisuudessa on kasvanut huomattavasti viime vuosina internet-teknologian, esineiden internetin ja pilvipalveluiden kehityksen myötä. Vastaavasti algoritmien ja laskentatehon parantuessa koneoppimisen menetelmistä on tullut tehokkaita työkaluja mallien löytämiseksi datasta. Koneoppimista on hyödynnetty esimerkiksi prosessien optimoimiseen, ennakoivaan huoltoon sekä sovellusten monitoroimiseen ja kontrolloimisen. Tämä asettaa kuitenkin haasteen teollisuuden toimijoille, sillä koneoppimisen kenttä on monimuotoinen, saatavilla on monia algoritmeja, teorioita ja metodeja. (Wuest, ym. 2016)

Valmistavassa teollisuudessa asennetaan esimerkiksi yhä enemmän sensoreita prosessilaitteistojen, yksittäisten komponenttien ja tuotanto-olosuhteiden monitoroimiseksi. Sensorit tuottavat aikasarjadataa. Aikasarjojen analyysia voidaan hyödyntää erilaisiin tarkoituksiin, kuten poikkeavuuksien havaitsemiseen, mallien löytämiseen, indeksointiin, visualisointiin, segmentointiin, trendien tunnistamiseen ja ennustamiseen. Aikasarjadataan liittyy tyypillisesti moniulotteisuuden lisäksi suuri määrä poikkeavia havaintoja sekä häiriöitä ja kohinaa. Lisäksi aikasarjat ovat usein kestoltaan eri mittaisia. Nämä seikat muodostavat merkittävän haasteen aikasarjojen samankaltaisuuden mittaamiseen. (Aghabozorgi;Seyed Shirkhorshidi ja Ying Wah 2015)

Oy Forcit Ab (myöhemmin Forcit) on suomalainen kemian sektorilla toimiva yritys, joka kehittää, valmistaa ja myy räjähdysaineita pääasiassa Pohjoismaisille markkinoille. Lisäksi se harjoittaa konsultointi- ja koulutustoimintaa (Oy Forcit Ab 2019). Forcitin liiketoiminnan digitalisoimiseen liittyy järjestelmäkokonaisuus, jota hyödynnetään louhintaprosessin suunnittelusta jälkiseurantaan. Osana järjestelmää tietoa kerätään emulsiopanostusyksiköistä panostuksen aikana. Panostusyksiköistä kerätään dataa prosessilaitteiston ja panostusyksiköitä operoivan henkilöstön toiminnasta. (Halonen ja Kähäri 2018)

Tutkielma toteutetaan konstruktiivisena tutkimuksena. Lukan (2001) mukaan konstruktiivinen tutkimusote on innovatiivista konstruktioita tuottava metodologia. Sillä pyritään

ratkaisemaan reaalimaailman ongelmia ja tuottamaan kontribuutioita sille tieteenalalle, jolla sitä sovelletaan. Konstruktiivista tutkimusotetta kuvaavat seuraavat ydinpiirteet:

- Tutkimusote keskittyy tosimaailman ongelmiin, jotka koetaan tarpeelliseksi ratkaista.
- Tutkimusote tuottaa innovatiivisen konstruktion ja sisältää kehitetyn konstruktion toteuttamisyrityksen, jolla testataan käytännön soveltuvuutta.
- Tutkija ja käytännön edustaja tekevät läheistä tiimimäistä yhteistyötä, jossa odotetaan tapahtuvan kokemuksellista oppimista.
- Tutkimus on huolellisesti kytketty olemassa olevaan teoreettiseen tietämykseen.
- Tutkimuksessa kiinnitetään erityistä huomiota empiiristen löydösten reflektoimiseen takaisin teoriaan.

Konstruktiivinen tutkimusote valikoitui, koska tutkielman lähtökohtana oli reaalimaailman ongelma. Forcitin tavoitteena on saada keräämästään datasta hyödyllistä ja uutta tietoa. Tiedon jalostumiseen datasta tarvitaan konstruktio, joka pohjautuu aiempaan tutkimuskirjallisuuteen. Tutkielman toteuttaminen vaatii myös tutustumista kohdealueeseen.

Tutkielmassa sovelletaan KDD-prosessia. Knowledge Discovery in Databases (KDD) viittaa kokonaisprosessiin, jonka avulla datasta pyritään löytämään uutta ja hyödyllistä tietoa. Ongelma, jota prosessin avulla pyritään ratkaisemaan, on matalan tason datan, joka on sellaisenaan liian laaja ymmärrettäväksi, kuvaaminen kompaktimmin, abstraktimmin tai hyödyllisemmin. KDD-prosessin ydin on spesifien tiedonlouhintametodien soveltaminen mallien löytämiseksi ja määrittämiseksi datasta. Prosessin muut vaiheet, kuten aikaisemman tiedon huomioiminen, datan valinta, esikäsittely ja muunnokset sekä tulosten asianmukainen tulkinta varmistavat, että datasta pystytään johtamaan hyödyllistä tietoa. (Fayyad;Piatetsky-Shapiro ja Smyth 1996a)

Emulsioräjähdysaineen valmistukseen ja pumppaamiseen liittyvää dataa on kerätty panostusyksiköistä mahdollisimman paljon, ilman että on etukäteen kaikilta osin päätetty mihin ja miten dataa tullaan käyttämään. Tutkielman tavoitteena on löytää emulsioräjähdysaineen valmistuksen ja panostuksen aikana kerätystä datasta uutta ja hyödyllistä tietoa. Tarkoituksena on löytää uutta tietoa valmistusprosessista ja prosessilaitteiston toiminnasta. Uuden

tiedon avulla pyritään parantamaan prosessin laatua ja tuottavuutta sekä tarjoamaan asiakkaille tarkempaa tietoa prosessista. Tutkielmassa sovelletaan KDD-prosessia panostusyksiköistä kerättävään dataan. Tutkimuskysymykset ovat seuraavat:

1. **Mitä tiedonlouhinnan menetelmiä voidaan soveltaa panostusyksiköistä kerättyyn dataan?**
2. **Voidaanko panostusyksiköistä kerätystä datasta löytää uutta ja hyödyllistä tietoa tiedonlouhinnan menetelmillä?**

Tutkielmalla voidaan nähdä olevan myös laajempi konteksti: vastuullinen liiketoiminta. Forcit on sitoutunut laatu-, ympäristö- ja turvallisuusasioiden jatkuvaan parantamiseen, se on sitoutunut Responsiple Care – Vastuu Huomisesta -ohjelmaan, sen ympäristöjärjestelmä on ISO 14001 sertifioitu ja laatujärjestelmä ISO 9001 sertifioitu. Panostus ja räjäytystyöhön liittyy ympäristövaikutuksia. Räjähdyksessä muodostuu 700–1000 litraa kaasua räjähdysainekiloa kohden. Pieni osa muodostuvista kaasuista on myrkyllistä, kuten hiilimonoksidi ja typpioksidit. (T. Halonen 2015) Typen päälähde kaivoksen (jäte)vesissä on räjähtämätön räjähde. Typpipäästöjen yleisin haitta on vastaanottavan vesistön rehevöityminen. Lisäksi ne voivat olla vesieliöille haitallisia. (Jermakka, ym. 2015) Huolellisella panostustyöllä voidaan pienentää panostuksessa ja räjähdyksessä syntyviä ympäristövaikutuksia. (T. Halonen 2015). Kauppilan ym. mukaan (2015) kaivosteollisuuden vastuullinen liiketoiminta sisältää yhteiskuntavastuun kolme ulottuvuutta: talouden, ympäristön ja sosiaaliset näkökohdat. Vastuullisen liiketoiminnan avulla pyritään vastaamaan kestävän kehityksen haasteisiin. Yhteiskuntavastuun toteuttamisen toimenpiteet liittyvät myös lainsäädännön ja kannattavuuden tavoitteisiin. Prosessikehityksellä, tuotannon toimintavarmuuden edistämisellä, materiaali- ja energiatehokkuuden parantamisella edistetään samanaikaisesti sekä kestävän kehityksen tavoitteiden saavuttamista että yritysten menestymistä. Vaikka ympäristövaikutukset ovat tutkielman ulkopuolella, tutkielman tuloksilla saattaa olla epäsuoraa vaikutusta myös niihin parantuneen prosessin laadun ja ennakoitavuuden kautta.

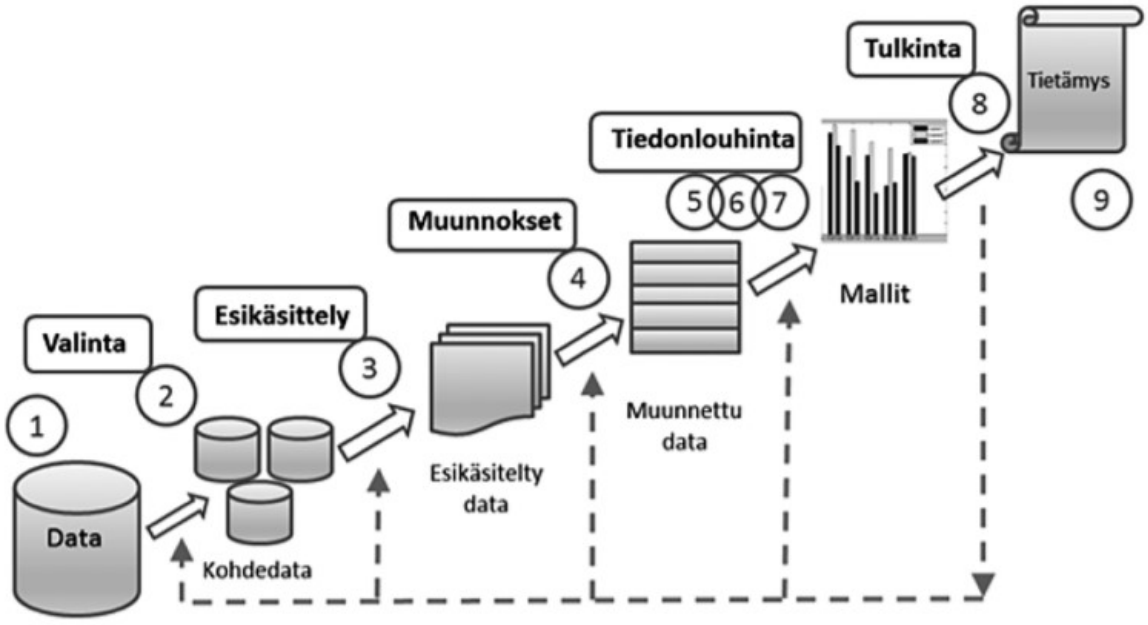
Tutkielmassa sovellettava KDD-prosessi kuvataan luvussa 2. 3. luvussa kuvataan aikasarjoille soveltuvia tiedonlouhinnan menetelmiä ja 4. luvussa käsitellään tarkemmin yksi soveltuvista menetelmistä eli klusterointi. 5. luvussa kuvataan työn empiirinen osuus eli

tiedonlouhinta panostusyksiköistä kerätylle datalle noudattaen KDD-prosessia. 6. luvussa esitetään johtopäätöksiä ja jatkotutkimuskohteita.

2 KDD-prosessi

Nykyiset laitteisto- ja tietokantateknologiat mahdollistavat tehokkaan datan tallentamisen ja hakemisen edullisesti. Tallennetut datajoukot ovat kuitenkin sellaisenaan arvoltaan vähäisiä. Ne tuottavat arvoa, kun niistä voidaan päätellä tietoa, jota voidaan käyttää hyödyksi. Datasta pyritään etsimään ja muodostamaan hahmoja (engl. *patterns*) tai malleja (engl. *models*) ja tämä informaatio voidaan johtaa tietämykseksi. KDD-prosessilla tarkoitetaan kokonaisprosessia, jonka avulla pyritään löytämään hyödyllistä tietoa datasta. Tiedonlouhinta, spesifien algoritmien soveltaminen hahmojen tai mallien määrittämiseksi datasta, on yksi erityinen askel prosessissa. Prosessin muita vaiheita ovat datan valinta, esikäsittely, muunnokset sekä tulkinta ja analyysi.

KDD-prosessi koostuu erilaisista vaiheista ja on luonteeltaan iteratiivinen. Oheisessa kuviossa on esitetty prosessin vaiheet (Kuvio 1).



Kuvio 1. Yleiskuvaus KDD-prosessin vaiheista (Fayyad, Piatetsky-Shapiro ja Smyth 1996a)

2.1 Datan valinta

Ensimmäisenä vaiheena (askel 1) ennen datan valintaa on muodostaa ymmärrys sovellusalueesta sekä aikaisemmasta tiedosta ja asiakkaan tavoitteista (Fayyad, Piatetsky-Shapiro ja Smyth ym. 1996a, 1996b, 1996c). Sovellusalueen tuntemus vaikuttaa myös seuraavissa vaiheissa tehtäviin valintoihin ja siten koko lopputulokseen.

Yleensä data halutaan valita niin (askel 2), että siitä määritetty informaatio pätee myös sellaiseen dataan, jota ei ole ollut saatavilla kyseisellä kohdealueella. Yhtä kiinnostuksen kohteena olevaa havaintoa (objektia, tietuetta) voidaan kuvata sen ominaisuuksien kokonaisuudella. Ominaisuuksia kutsutaan tietueen muuttujiksi tai attribuuteiksi (Bramer 2016) ja myös piirteiksi, ominaisuuksiksi, dimensioiksi, kentiksi (Zaki ja Meira 2014). Datajoukko esitetään usein $n \times d$ matriisina, missä rivit n kuvaa datan havaintoja ja sarakkeet d datan muuttujia. Kaikki datajoukot eivät ole matriisiformaatin muodossa, kuten monimutkaisemmat sekvenssi-, teksti, aikasarja-, kuva-, tai videomuodot, vaikka ne voidaan ainakin osittain muuntaa sellaiseksi (Hand;Mannila ja Smyth 2001, Zaki ja Meira 2014)

Kohteen ominaisuuksien mittaamiseen on erilaisia muuttujatyyppejä. Nominaali- eli luokitteluasteikkoa käytetään jakamaan datajoukko kategorioihin tai luokkiin. Nominaalimuuttujien arvoilla ei kuitenkaan ole yksiselitteistä järjestystä. Mikäli muuttujien arvot voidaan laittaa mielekkääseen järjestykseen, on kyseessä järjestys- eli ordinaaliasteikko. Binäärinen, dikotominen muuttuja on nominaaliasteikon erikoistapaus ja sillä on vain kaksi mahdollista arvoa kuten ei tai kyllä eli muuttujalla joko on jokin ominaisuus tai ei. Binäärimuuttuja koodataan `0-1`-muuttujaksi.

Edellä kuvatut datatyypit ovat ei-numeerisia, kategorisia muuttujia. Numeerinen muuttuja voi puolestaan olla jatkuva tai diskreetti eli epäjatkuva. Jos kahden arvon välissä on ääretön määrä arvoja, on muuttuja jatkuva. Mikäli muuttaja voi saada arvon äärellisestä tai numeroituvasti äärettömästä joukosta, on kyseessä epäjatkuva eli diskreetti muuttuja. Numeeriset muuttujat voidaan edelleen jakaa välimatka- ja suhdeasteikollisiin muuttujiin. (Zaki ja Meira 2014)

Datan suhde aikaan lisää vielä yhden dimension datatyyppien luokitteluun. Jos ajan kuluminen ei vaikuta muuttujan arvoon, on kyseessä staattinen data. Mikäli muuttujan arvo muuttuu ajan kuluessa, on muuttuja dynaaminen. Suurin osa tiedonlouhinnan menetelmistä sopivat staattiselle datalle ja dynaamisen datan louhiminen vaatii usein esikäsittelyä. (Kantardzic 2011) Datan tyyppi tulisi huomioida datan valinnan vaiheessa, koska sillä on vaikutusta myös valittavissa oleviin analyysimenetelmiin.

2.2 Datan esikäsittely

Datan puhdistukseen ja esikäsittelyyn (askel 3) kuuluu perusoperaatiot kuten kohinan poistaminen tarvittaessa, informaation kerääminen kohinan mahdollisista malleista tai selityksestä, strategian valinta puuttuvien arvojen käsittelyyn ja selvästi turhien ominaisuuksien poistaminen (Fayyad, Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c).

2.2.1 Virheelliset arvot

Ei voida olettaa, että data on virheetöntä. Reaalimaailmassa virheellisiä arvoja tallentuu useista syistä, kuten mittausvirheet, subjektiiviset päätelmät, epäkuntoiset laitteet tai automaattisten tallennuslaitteiden väärinkäyttö. Virheelliset arvot voidaan jakaa sellaisiin, jotka ovat mahdollisia ja niihin, jotka eivät ole mahdollisia. Kohina voidaan määritellä monella tavalla, mutta yleensä sillä tarkoitetaan häiriöstä johtuvaa epätarkkuutta tai satunnaisvirhettä. Kohina voidaan joko korjata tai poistaa kokonaan.

Virheelliset arvot ja kohina voi olla vaikea havaita ja paikallistaa erittäin suurista datajoukoista. Virheiden löytämiseksi dataa voidaan visualisoida, jolloin anomalia tai esimerkiksi arvojen odottamaton keskittymä voidaan havaita. Lisäksi numeeristen muuttujien kohdalla voidaan järjestää arvot nousevaan järjestykseen, jolloin virheelliset arvot voidaan huomata. Voidaan esimerkiksi huomata, että yhden muuttujan kaikki arvot ovat samoja, jolloin koko muuttuja voidaan jättää pois. Jos kaikki paitsi yksi arvo ovat samoja, voi kyseessä olla virheellinen arvo. Jos arvo on validi, silloin muuttuja tulisi käsittää kategorisena kahden arvon muuttujana.

On tärkeää erottaa todelliset virheelliset arvot ja poikkeavat havainnot (engl. *outliers*). Muista havainnoista merkittävästi poikkeava havainto voi olla täysin validi ja siten olla merkittävä löydös vaikkapa laitteiston toiminnan kannalta. Poikkeavia havaintoja ei pidä suoraan poistaa, vaan tutkia arvoa tarkemmin. (Bramer 2016)

2.2.2 Puuttuvat havainnot

Muuttujien arvot eivät aina tallennu reaalimaailman datajoukoissa. Puuttuvat arvot voivat johtua esimerkiksi inhimillisestä virheestä, laitteiston (hetkellisestä) toimintahäiriöstä (Bramer 2016) tai kerättävien muuttujien lisäämisestä, jos jokin muuttuja havaitaankin tärkeäksi (Fayyad;Piatetsky-Shapiro ja Smyth 1996c). Puuttuvien arvojen käsittelyyn on olemassa useita strategioita, joista yleisimmät ovat kohteen poistaminen ja puuttuvan arvon imputointi (Bramer 2016).

Yksinkertaisin tapa on poistaa kohteet, joista puuttuu vähintään yksi muuttujan arvo. Tällöin käytettävä datajoukko ei sisällä virheitä, mutta tulosten luotettavuus voi huonontua. Poistaminen voi tulla kyseeseen, jos puuttuvia arvoja on vain pieni osa datasta ja ne kohdistuvat dataan satunnaisesti.

Toinen tapa on arvioida kukin puuttuva arvo käyttämällä arvoja, jotka esiintyvät datajoukossa. Kategoristen muuttujien osalta puuttuva arvo voidaan korvata useimmin esiintyvällä arvolla, jos muuttuja on selvästi painottunut tiettyyn arvoon. Jatkuvien muuttujien osalta voidaan käyttää keskiarvoa. (Bramer 2016) Aikasarjojen osalta puuttuvat arvot voidaan muodostaa myös interpoloimalla (Shukla ja Marlin 2021).

Puuttuvien havaintojen korvaaminen millä tahansa arviolla voi luonnollisesti aiheuttaa kohinaa dataan. Jos puuttuvien arvojen osuus on pieni, vaikutus tuloksiin jää pieneksi. On tärkeää arvioida korvaavan arvon mielekkyys kohteelle. Ei ole olemassa strategiaa, joka olisi luotettavampi kuin toinen kaikissa tapauksissa. Käytettävä menetelmä tulee arvioida tapauskohtaisesti. (Bramer 2016)

2.2.3 Datan skaala

Erityisesti etäisyyteen perustuvat tiedonlouhintamenetelmät voivat vaatia normalisoitua dataa parhaan tuloksen saavuttamiseksi. Jos normalisointia ei tehdä, ne piirteet, joiden arvot ovat keskimäärin suuria painottuvat liikaa (Kantardzic 2011). Normalisointi voidaan suorittaa esimerkiksi min-max-skaalauksena, jossa muuttujan $X = x_1 \dots x_n$ arvot skaalataan välille $[0,1]$ seuraavasti (Zaki ja Meira 2014):

$$x_i' = \frac{x_i - min_i\{x_i\}}{max_i\{x_i\} - min_i\{x_i\}}. \tag{2.1}$$

Muuttujan arvot saadaan standardinormaalijakauman (engl. *z-normalized*) mukaisiksi muuntamalla X niin että sen keskiarvo $\mu = 0$ ja keskihajonta $\sigma = 1$ (Zaki ja Meira 2014):

$$x_i' = \frac{x_i - \hat{\mu}}{\hat{\sigma}}, \tag{2.2}$$

missä $\hat{\mu}$ on havaintojen keskiarvo ja $\hat{\sigma}^2$ keskihajonta.

Aikasarjojen osalta voidaan käyttää esimerkiksi amplitudin skaalausta, lineaarisen trendin poistamista, aika-akselin skaalausta ja kohinan poistamista tasoittamalla (engl. *smoothing*) esimerkiksi liukuvan keskiarvon avulla (Pechenizkiy, ym. 2010).

2.3 Datan muunnos

Neljännessä vaiheessa vähennetään datan dimensioita ja projisoidaan data niin, että alkuperäisen datan ominaisuudet säilyvät mahdollisimman muuttumattomina mutta muuttujien määrä vähenee. Muuttujien suuri määrä aiheuttaa ongelmia koska avaruus, josta malleja etsitään kasvaa eksponentiaalisesti dimensionaalisuuden kasvun myötä (Fayyad, Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c). Bellman (1961) kutsui ongelmaa dimensionaalisuuden kiroukseksi (engl. *curse of dimensionality*). Korkea dimensionaalisuus johtaa myös laskennallisiin ongelmiin, sillä etäisyyksien erottelukyky vähenee. Moniulotteisessa avaruudessa data hajoaa avaruuden pinnoille ja kulmiin jättäen keskustan tyhjäksi. Etäisyysmitat menettävät merkityksensä, sillä data ei riitä täyttämään moniulotteista avaruutta, ja datapisteet näyttävät olevan yhtä kaukana toisistaan. Kun datapisteet ovat harvassa, ei muodostu

tihentymiä eli etäisyyksiin tai lähimpään naapuriin perustuvat luokittelu- tai ryhmittelyalgoritmit eivät välttämättä toimi hyvin. (Verleysen ja François 2005)

Käytännössä halutaan valita ominaisuudet, jotka ovat relevantteja ja siten maksimoida suorituskyky ja minimoida mittaukset ja prosessointi. Dimensioiden vähentäminen tulisi johtaa (Kantardzic 2011):

1. Pienempään datamäärään algoritmin suoritusajan nopeuttamiseksi.
2. Parempaan tarkkuuteen niin että malli yleistää datan paremmin.
3. Yksinkertaisempaan, ymmärrettävämpään ja käytettävämpään louhinnan tulokseen.
4. Turhien muuttujien keräämisen lopettamiseen.

Dimensioiden vähentäminen voidaan toteuttaa karkeasti kahdella tavalla: piirteiden valinnalla (engl. *feature selection*) tai piirteiden irrotuksella (engl. *feature extraction*). Piirteiden valinnassa valitaan edustava osajoukko kaikista muuttujista ja piirteiden irrotuksessa muunnetaan tai yhdistetään olemassa olevat uudeksi pienemmäksi joukoksi muuttujia. (Kantardzic 2011) Pääkomponenttianalyysi (engl. *Principal Component Analysis, PCA*) on eniten käytetty piirreirrotuksen menetelmä dimensioiden vähentämiseen. Menetelmässä pyritään löytämään datalle projektio niin, että datan varianssi maksimoituu. Pääkomponenttianalyysi pystyy löytämään vain lineaarisia rakenteita, joten komponenttien epälineaariset suhteet saatetaan menettää prosessissa. (Verleysen ja François 2005) Piirreirrotukseen on olemassa myös muita lineaarisia ja epälineaarisia menetelmiä. Yksi epälineaarinen menetelmä on moniulotteinen skaalaus (engl. *multidimensional scaling*), joka pyrkii säilyttämään alkuperäisten datapisteiden väliset etäisyydet alhaisemmassa dimensioavaruudessa (Kantardzic 2011, Hand;Mannila ja Smyth 2001).

2.4 Tiedonlouhinta

Hand, Mannila ja Smyth (2001) mukaan tiedonlouhinnalla tarkoitetaan (usein suurten) havaintoaineistojen analysointia odottamattomien yhteyksien löytämiseksi ja yhteenvetojen tekemistä datasta uusilla tavoilla, jotka ovat sekä ymmärrettäviä että hyödyllisiä datan omistajille. Tiedonlouhinta-algoritmien soveltaminen on yksi erityinen KDD-prosessin vaihe. Tiedonlouhinta-algoritmi on hyvin määritelty proseduuri, joka ottaa syötteenä dataa ja

tulostaa mallin tai hahmon. Malli on korkean tason kuvaus suuresta datakokoelmasta summaten ja kuvaillen sen tärkeitä piirteitä. Malli on siis usein globaali siinä mielessä, että se pätee kaikkiin pisteisiin näyteavaruudessa. Hahmo on puolestaan lokaali kuvaus, joka pätee datan osajoukkoon osoittaen muutaman näyteavaruuden pisteen käyttäytymisen tai luonnehtien jonkin pysyvän, mutta epätavallisen rakenteen datassa.

Mallit voidaan edelleen jakaa kuvaileviin ja ennustaviin. Kuvaileva malli esittää datan pääpiirteet sopivassa muodossa. Pohjimmiltaan kyse on datan tiivistelmästä, jolloin kaikkein tärkeimpien aspektien tarkastelu mahdollistuu ilman, että ne hämärtyisivät tai peittyisivät datan koon vuoksi. Pyritään siis ymmärtämään datan muodostumisen pohjalla oleva prosessi. Kuvailevia malleja ovat esimerkiksi koko datan todennäköisyysjakauma (tiheysfunktion estimointi), riippuvuuksien mallintaminen muuttujien välillä sekä klusterointi ja segmentointi. Klusterointi esitellään tarkemmin luvussa 4. Sen sijaan ennustava mallin tavoitteena on ennustaa mihin luokkaan havainto kuuluu. Ennustavat mallit keskittyvät ennustuksen tarkkuuteen, ei niinkään siihen, millä tavalla malli reflektoi todellisuutta (Hand, Mannila ja Smyth 2001; Fayyad, Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c.)

Tiedonlouhintamenetelmät voidaan jakaa myös ohjattuun ja ohjaamattomaan oppimiseen datan perusteella. Mikäli valittu data on luokittelematonta, eli se ei sisällä erityistä tunnusta (engl. *label*), on kyseessä ohjaamaton oppiminen. Ohjatussa oppimisessa pyritään ennustamaan datajoukon tunnusten avulla uuden, ennennäkemättömän instanssin luokan tunnus. Jos tunnus on kategorinen, tiedonlouhinnan tehtävää kutsutaan luokitteluksi ja tunnuksen ollessa numeerinen kyseessä on regressio. (Bramer 2016) Tässä tutkielmassa keskitytään ohjaamattoman oppimisen menetelmiin, sillä tutkimusaineisto on luokittelematonta.

Luokittelu ja regressio ovat siis ennustavaa mallinnusta ja yksi tiedonlouhinnan tehtävistä. Muita tiedonlouhinnan tehtäviä ovat edellä mainittu kuvaileva mallinnus, eksploratiivinen data-analyysi (engl. *Exploratory Data Analysis, EDA*), mallien ja sääntöjen etsiminen (engl. *Discovery of Patterns and Rules*) ja hakeminen sisällön mukaan (engl. Retrieval by Content). (Hand;Mannila ja Smyth 2001). Ennen varsinaisen tiedonlouhinta-algoritmin soveltamista KDD-prosessissa päätetään mallin tavoite yhdistämällä se tiettyyn tiedonlouhinnan tehtävään (askel 5) (Fayyad; Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c).

Seuraavaksi valitaan tiedonlouhinnan algoritmi (askel 6). Fayyad, Piatetsky-Shapiro ja Smyth (1996a, 1996b, 1996c) mukaan tiedonlouhinnan algoritmi koostuu kolmen komponentin yhdistelmästä, joita ei kuitenkaan yleensä selvästi erikseen määritellä algoritmien kuvauksissa. Komponentit ovat:

- Malli: Koostuu mallin tehtävästä (kuten luokittelu ja klusterointi) ja mallin esitysmuodosta (kuten päätöspuu ja lineaarinen funktio). Malli sisältää parametrit, jotka määritellään datasta.
- Preferenssikriteeri: Muodostaa pohjan, jolla jokin malli preferoidaan, eli kuinka hyvin jokin malli ja sen parametrit sopivat dataan ja täyttävät tiedonlouhinnan tavoitteen.
- Etsintäalgoritmi: Spesifikaatio algoritmille tietyn mallin ja parametrien löytämiseksi, jossa huomioidaan data ja preferenssikriteeri.

Tiedonlouhintamenetelmistä jotkin sopivat tietyille ongelmille paremmin kuin toiset, joten algoritmien valinta on jossain määrin taiteenlaji. Suurin osa ajasta kuluukin ongelman oikeaan asetteluun ja ymmärtämiseen sekä oikeiden kysymysten esittämiseen, ei niinkään tietyn algoritmin optimoimiseen. Kun algoritmi on valittu, tiedonlouhinnan viimeisessä vaiheessa suoritetaan varsinainen tiedonlouhinta valitulla algoritmilla (askel 7). Edeltävien vaiheiden huolellisella suorittamisella voidaan vaikuttaa lopputuloksen onnistumiseen.

2.5 Tulkinta, arviointi ja hyödyntäminen

Viimeisissä vaiheissa louhittujen hahmojen tai mallien merkitys tulkitaan (askel 8). Tarvittaessa palataan aikaisempiin vaiheisiin. Tulkintaan kuuluu myös mallien tai mallin mukaisen datan visualisointi. (Fayyad, Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c.) Lisäksi poistetaan turhat tai merkityksettömät mallit ja muunnetaan hyödylliset ymmärrettävävään muotoon (Fayyad, Piatetsky-Shapiro ja Smyth 1996c).

Viimeisenä vaiheena on löydetyn tietämyksen hyödyntäminen (askel 9). Uusi tieto voidaan sisällyttää järjestelmään, voidaan ryhtyä tarvittaviin toimenpiteisiin tiedon pohjalta tai vain dokumentoida ja raportoida tieto kiinnostuneille tahoille. On myös syytä tarkistaa, ettei uusi

tieto ole ristiriidassa aiempien olettamuksien tai tiedon kanssa. (Fayyad, Piatetsky-Shapiro ja Smyth 1996a, 1996b, 1996c).

3 Aikasarjojen tiedonlouhinta

Aikasarja koostuu samaa ilmiötä kuvaavista mittausajankohdan mukaan järjestetyistä havainnoista. Mittausajankohtien tiheys vaihtelee esimerkiksi alle sekunnin automaattisesta mittaamisesta useiden vuosien jaksoihin tutkittavan ilmiön mukaan. (Han;Pei ja Kamber 2012) Luvussa kuvataan erityisesti aikasarjoille soveltuvia tiedonlouhinnan tehtäviä. Tehtäviä voivat olla esimerkiksi klusterointi (kuvataan tarkemmin luvussa 4), visualisointi, luokittelu, sääntöjen löytäminen (engl. *rule discovery*), poikkeuksien havaitseminen ja toistuvien kuvioiden (engl. *motif discovery*) löytäminen. Perinteiseen aikasarja-analyysiin verrattuna, kuten trendien ja kausivaihteluiden tunnistaminen ja ennustaminen, aikasarjojen tiedonlouhinta näyttäytyy pääasiallisesti samankaltaisuuden mittaamisen ongelmana.

3.1 Etäisyyden ja samankaltaisuuden mittarit

Yksinkertaisin ja yleinen etäisyysmitta on euklidinen etäisyys. Kahden aikasarjan $Q = q_1 \ldots q_n$ ja $C = c_1 \ldots c_n$ euklidinen etäisyys on

$$D(Q, C) = \sqrt{\sum_{i=1}^{n} (q_i - c_i)^2} \ . \tag{3.1}$$

Laskenta-aikaa voidaan vähentää hieman käyttämällä neliöllistä euklidista etäisyyttä

$$D_{squared}(Q, C) = \sum_{i=1}^{n} (q_i - c_i)^2 \ . \tag{3.2}$$

Euklidinen etäisyys on helppo laskea, mutta se on herkkä esimerkiksi datan kohinalle, poikkeamille, eri skaaloille ja vaihesiiroille. Siksi normalisointi on usein välttämätöntä. Joskus kuitenkin datan ”vääristymät” ovat kaikkein kiinnostavimpia löydöksiä datassa. (Pechenizkiy, ym. 2010) Joillain kohdealueilla normalisointi voi hävittää kiinnostavia poikkeavuuksia datasta (Akbarinia ja Cloez 2019, De Paepe, ym. 2020).

Pearsonin korrelaatiota voidaan käyttää myös etäisyysmittana, mutta sen heikkoutena on vain lineaaristen suhteiden havaitseminen ja herkkyys jopa yhdelle poikkeavalle arvolle.

Ristikorrelaatio on myös mahdollinen, mutta se ei pysty havaitsemaan viiveiden siirtymistä ajan suhteen. (Pechenizkiy, ym. 2010)

Dynaaminen aikasovitus (engl. *Dynamic Time Warping, DTW*) on aikasarjojen muotojen vertailuun perustuva menetelmä, jossa kaksi aikasarjaa pyritään sovittamaan optimaalisesti toisiinsa nähden epälineaarisesti niin, että niiden etäisyys minimoituu. Optimaalinen sovituspolku etsitään muodostamalla $|x| \times |y|$ kokoinen matriisi, joka täytetään x:n ja y:n jokaisen pisteparin (euklidisella) etäisyydellä. Jokainen mahdollinen sovituspolku (engl. *warp*) W = $w_1 \ldots w_k$ on polku matriisin läpi kahden aikasarjan välillä, ja etsitään optimaalista sovituspolkua, joka minimoi siirtymän (Pechenizkiy, ym. 2010):

$$DTW(x,y) = min\left\{\sqrt{\sum_{k=1}^{K} w_k}\Big/K\right. . \tag{3.3}$$

Optimaalisen sovituspolun etsintään asetetaan rajoitteita, jotta vältetään turhien polkujen löytyminen. Rajoitteita ovat esimerkiksi monotonisuusehto (engl. *monotonicity condition*) eli polun tulee olla etenevä niin ettei mennä alas eikä vasemmalle, jatkuvuusehto (engl. *continuity condition*) eli solujen yli ei voida hypätä sekä rajaehto (engl. *boundrary condition*) eli polkujen tulee alkaa pisteistä $w_1 = (1,1)$ ja loppua $w_k = (|x|, |y|)$. Lisäksi usein käytetään sovitusikkunaa $|i - j| \leq w$ estämään sovitusta jäämästä jumiin samanlaisiin sisäisiin sovituspolkuihin (engl. *pathological paths*). Optimaalinen sovituspolku voidaan etsiä dynaamisen ohjelmoinnin avulla rekursiivisesti (Pechenizkiy, ym. 2010):

$$D^2_{DTW}(x,y) = D^2(x_i, y_j) + min\begin{cases} D^2_{DTW}(x_i, y_{j-1}) \\ D^2_{DTW}(x_{i-1}, y_j) \\ D^2_{DTW}(x_{i-1}, y_{j-1}) \end{cases}, \tag{3.4}$$

missä x_i=First(x) on x:n ensimmäinen elementti ja x_{i-1} = Rest(x) on jäljelle jäävä aikasarja, josta First(x) on poistettu.

DTW:n aikavaativuus on $O(n^2)$. Laskentaa on mahdollista nopeuttaa hieman globaaleilla rajoitteilla (Sakoe-Chiba, 1987; Itakura, 1975). Algoritmin nopeuttamiseen voidaan käyttää

lisäksi esimerkiksi datan karkeistamista (engl. *lower-bounding*) (Keogh ja Ratanamahatana 2005).

Pisimmän yhteisen osajonon (Longest Common Subsequence, LCSS) lähestymistapa perustuu myös aika-akselin sovitukseen. Se ei ole yhtä herkkä poikkeaville havainnoille ja kohinalle kuin DTW, sillä aikasarjan osien yli voidaan hypätä sovituksessa. Kahden aikasarjan oletetaan olevan samankaltaisia, jos niillä on tarpeeksi yhteisiä osajonoja. (Vlachos, ym. 2003)

Tilastolliset mallit pystyvät tarjoamaan vain globaaleja kuvauksia aikasarjoista (Hand;Mannila ja Smyth 2001). Globaalit tilastolliset mallit tasoittavat erityisesti lokaalit muodot ja rakenteelliset ominaisuudet. Kuitenkin monet aikasarjat ovat luonnollista kuvata rakenteellisten piirteiden mukaan. Esimerkiksi sydänsähkökäyrällä on tunnusomainen lokaali visuaalinen muoto. Kiinnostavien hahmojen paikallistaminen tehokkaasti ja tarkasti aikasarjadatasta on tärkeä ja epätriviaali ongelma monilla sovellusalueilla, kuten kompleksisten systeemien monitoroinnissa ja diagnosoinnissa, biolääketieteessä sekä eksploratiivisessa data-analyysissä niin tieteellisissä kuin kaupallisissa aikasarjoissa. Mikäli aikasarjasta halutaan löytää osajono, joka on mahdollisimman samanlainen kuin kyselty osajono, on samankaltaisuuden mittaamiseen erilaisia menetelmiä.

Yksi lähestymistapa on suorittaa peräkkäinen skannaus kyselyjonolle yli koko aikasarjan pituuden, siirtäen kyselyjonoa yksi aikapiste kerrallaan ja laskien etäisyysmitan jokaisessa pisteessä. Euklidisen etäisyyden laskeminen raakaa voimaa käyttäen on paitsi laskennallisesti kohtuuttoman kallista, se keskittyy vain matalan tason mittauspisteisiin datassa rakenteellisten ominaisuuksien sijaan. Rakenteellisilla ominaisuuksilla tarkoitetaan esimerkiksi huippuja ja trendejä. Suora euklidiseen etäisyyteen perustuva sovitus on myös herkkä hienovaraiselle aika-akselin suuntaiselle ”venymiselle”. Euklidinen etäisyys voi täten olla suurempi kuin mitä ihmisen visuaalisesti observoima etäisyys kahden aikasarjan välillä näyttäisi olevan. (Hand;Mannila ja Smyth 2001)

Suosittu lähestymistapa on arvioida lokaalisti sekä kyselyjonon että kyselyn kohteena olevan aikasarjan muotoon perustuvia piirteitä ja näin verrata niiden korkeamman rakenteellisen tason samankaltaisuutta. Lähestymistavalla voidaan saavuttaa laskennallista etua, sillä

abstraktiossa data on tiivistetty eli signaalin epärelevantit yksityiskohdat voidaan jättää huomioimatta. Lisäksi datasta voidaan poimia rakenteellista informaatiota ihmiselle sopivassa, tulkittavassa muodossa. Yksi esimerkki tekniikasta on approksimoida signaali paloittain lineaarisiin (tai polynomisiin) segmentteihin. Segmentoitu sarja voidaan esittää lokaalisti parametrisoituina käyrien listana, ja rakenteelliset ominaisuudet, kuten huiput ja laaksot, voidaan laskea suoraan parametrisoidusta kuvauksesta. Tämän jälkeen voidaan käyttää todennäköisyysmallia odotetun muodon ja vaihtelevuuden parametrisoimiseksi näiden piirteiden suhteen, sallien joustavan ja mukautuvan pohjamallien perheen. Osajonon esiintyminen aikasarjasta muodostuu siten suurimman uskottavuuden estimoinniksi, joka maksimoi uskottavuusfunktion mallin parametrien suhteen. Esittämistapa on hyödyllinen signaaleille, joita ei voi käsitellä globaalien tilastollisten menetelmien avulla. Tällaisia signaaleja ovat epästationaariset aikasarjat, jotka sisältävät trendejä ja kausivaihteluita. (Hand;Mannila ja Smyth 2001)

Wang, Wirth ja Wang (2007) esittelevät joukon aikasarjojen rakenteen tilastosuureisiin perustuvia piirreirrotuksen menetelmiä. Tilastollisten piirteiden tulee kuvata aikasarjadata sen globaalin rakenteen koosteena. Tutkimuksessa käytetyt tilastosuureet olivat trendi, kausivaihtelu, autokorrelaatio (engl. *serial correlation*), epälineaarisuus (engl. *non-linearity*), vinous (engl. *skewness*), kurtoosi (engl. *kurtosis*), itsesimilaarisuus (engl. *self-similarity*), kaoottisuus (engl. *chaotic*) ja jaksollisuus (engl. *periodicity*), mutta sopivat suureet tulee valita tapauskohtaisesti. Lasketuista piirteistä muodostuu jokaiselle aikasarjalle yhtä pitkä vektori, joiden välille voidaan laskea (euklidinen) etäisyys.

Pechenizkiyn ym. (2010) mukaan aikasarjat voidaan kuvata esimerkiksi diskreetillä Fourier muunnoksella (engl. *Discrete Fourier Transform, DFT*) tai diskreetillä Wavelet-muunnoksella (engl. *Discrete Wavelet Transform, DWT*). DFT:n avulla voidaan laskea taajuuskomponenttien amplitudi ja vaihe. DFT on hyvä kompressoimaan monia luonnollisia signaaleja mutta ei sovellu hyvin eri mittaisille aikasarjoille eikä tue painotettuja etäisyysmittoja. DWF kuvaa aikasarjan aallokefunktioiden lineaarikombinaationa. Aallokemuunnosten suurin ero Fourier-muunnoksiin on aikaulottuvuuden säilyttäminen eli taajuudet voidaan paikallistaa ajan suhteen. DWT suoriutuu stationaaristen signaalien tiivistämisessä hyvin mutta sekään ei sovellu painotetuille etäisyysmitoille. Rakenteellisen samankaltaisuuden määrittelyssä

keskeiset kysymykset ovat mitä piirteitä tulisi etsiä, ja mitä etäisyysmittaa uudessa piirreavaruudessa tulisi käyttää.

3.2 Matriisiprofiili

Aikasarjojen samankaltaisuuden mittaamiseen on viime vuosina esitetty menetelmä, jota kutsutaan matriisiprofiiliksi (engl. *matrix profile*) (Yeh, ym. 2016). Matriisiprofiili tarjoaa ratkaisun aikasarjojen samankaltaisuuden perustehtävään eli etsitään dataobjektien joukolle lähin naapuri jokaiselle objektille (engl. *similarity join*). Etsintäalgoritmi käyttää standardoitua euklidista etäisyyttä alirutiinina hyödyntäen osajonojen päällekkäisyyttä klassisessa nopeassa Fourier-muunnoksessa (engl. *Fast Fourier Transform, FFT*). Menetelmässä muodostetaan kaksi meta-aikasarjaa, matriisiprofiili ja matriisiprofiili-indeksi, jotka annotoivat aikasarjan etäisyyden ja sijainnin sen kaikkiin osajonojen lähimpiin naapureihin siinä itsessään tai toisessa aikasarjassa. Nämä kaksi dataobjektia sisältävät implisiittisesti ratkaisun moneen aikasarjojen tiedonlouhintatehtävään. Matriisiprofiilia voidaan käyttää sellaisenaan ainakin toistuvien kuvioiden etsintään (engl. *motif discovery*), sellaisten poikkeusten havaitsemiseen, jotka ilmenevät uniikkeina sekä segmentointiin.

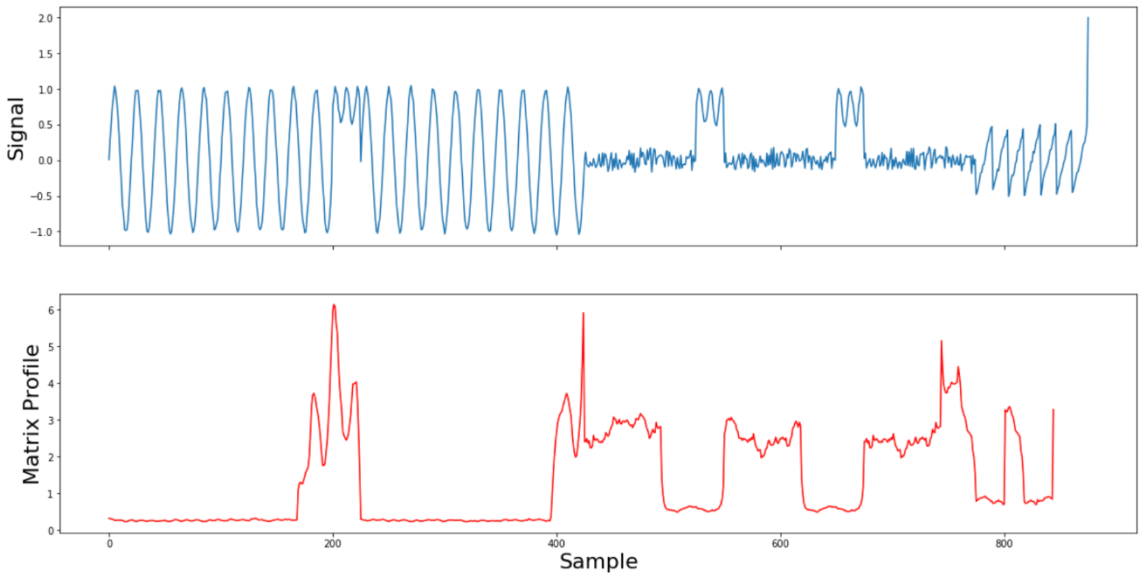
Matriisiprofiili lasketaan aikasarjalle $S \in \mathbb{R}$ (s_1, s_2, …, s_n), missä n on S:n pituus. Aikasarjassa ei olla kiinnostuneita sen globaaleista ominaisuuksista, vaan sen osajonojen samankaltaisuudesta. S:n osajono $S_{i,m}$ on m pituinen jatkuva osajoukko S :n arvoista, joka alkaa i:n positiosta. $S_{i,m} = s_i, s_{i+1}, \dots, s_{i+m-1}$, missä $1 \leq i \leq n - m + 1$. Aikasarjasta voidaan ottaa mikä tahansa osajono ja laskea sen etäisyys kaikkiin osajonoihin. Näin muodostuu etäisyysprofiili D. Triviaalit sovitukset osajonoon vältetään määrittämällä $m/2$ alue ennen ja jälkeen kyselyn sijainnista, joka jätetään huomioimatta. Kaikkien osajonojen joukon määrittely tehdään vain notaation vuoksi. Kaikkia osajonoja ei toteutuksessa irroteta, sillä se veisi turhaa laskenta-aikaa ja tilaa. S:n kaikkien osajonojen joukko A on järjestetty joukko kaikista mahdollisista osajonoista, jotka on saatu liu'uttamalla m:n pituista ikkunaa S:n läpi. $A = \{S_{1,m}, S_{2,m}, \dots, S_{n-m+1,m}\}$, missä m on käyttäjän määrittelemä osajonon pituus. $S_{i,m}$ merkitään $A[i]$.

Kiinnostuksen kohteena on lähimmän naapurin (1NN) suhteet osajonojen välillä. Olkoot kaikkien osajonojen joukot A ja B sekä kaksi osajonoa $A[i]$ ja $B[j]$, 1NN-yhdistefunktio $\theta_{1nn}\ (A_{[i]}, B_{[j]})$ on totuusarvomuuttujafunktio, joka palauttaa ”tosi”, vain jos $B[j]$ on $A[i]$:n lähin naapuri joukossa B. Funktion avulla voidaan luoda samankaltaisuusyhdistejoukko (engl. *similarity join set*) $J_{AB} = \{\langle A[i], B[j] \rangle \,|\theta_{1nn}\ (A[i], B[j])\}$, joka sisältää A:n jokaisen osajonon kaikkien parien lähimmät naapurit B:ssä. Merkitään $J_{AB} = A \bowtie_{\theta_{1nn}} B$. Jokaisen joukon parin välille lasketaan euklidinen etäisyys ja tallennetaan tulos järjestettyyn vektoriin, jota kutsutaan matriisiprofiiliksi. Aikasarjojen oletetaan olevan standardinormaalijakauman (kaava 2.2) mukaisia.

Euklidinen etäisyys $D_{ZE} = (A, B)$ kahden yhtä pitkän $A \in \mathbb{R}^m$ ja $B \in \mathbb{R}^m$ standardinormaalijakauman mukaiseksi muunnetun aikasarjan $\hat{A}$ ja $\hat{B}$ euklidinen etäisyys D_E joka voidaan laskea kaavan 3.1 mukaan. Matriisiprofiili P_{AB} on vektori, joka sisältää euklidisen etäisyyden jokaiselle parille J_{AB}:ssa. Mikäli joukosta A halutaan laskea etäisyydet siihen itseensä (engl. *self-similarity join set*), voidaan muodostunut matriisiprofiili nähdä annotoivana metaaikasarjana. Profiililla on tällöin kiinnostavia ja hyödynnettäviä ominaisuuksia: profiilin korkein kohta osoittaa poikkeaman, alhaisimmat pisteet osoittavat parhaimpaan toistuvan kuvion parin sijaintiin ja varianssi voidaan tulkita S:n kompleksisuuden mittana. Lisäksi aikasarjan tiheysfunktion estimaatio saadaan matriisiprofiilin arvojen histogrammin avulla.

Matriisiprofiilin i:s elementti kertoo siis etäisyyden sille osajonolle, joka alkaa i:stä, sen lähimpään naapuriin, mutta se ei kerro missä lähin naapuri sijaitsee. Tämä tieto tallennetaan matriisiprofiili-indeksiin. I_{AB} on kokonaislukujen vektori, missä $I_{AB}[i] = j$ jos $\{A_{[i]}, B_{[j]}\} \in J_{AB}$. Kun naapuri-informaatio tallennetaan tällä tavalla, voidaan lähin naapuri hakea tehokkaasti.

Kuviossa 2 on esitetty matriisiprofiili (punaisella), joka on laskettu yllä olevalle aikasarjalle. Toistuvat kuviot (engl. *motif*) aikasarjassa johtavat matalaan arvoon matriisiprofiilissa. Uusi, ennennäkemätön kuvio tai käytöksen muutos aikasarjassa johtaa puolestaan korkeaan arvoon matriisiprofiilissa. Tällaisen osajonon etäisyys lähimpään naapuriin on suuri.



Kuvio 2. Aikasarja ja siitä muodostettu matriisiprofiili

Yeh ym. (2016) esittelivät matriisiprofiilin lisäksi matriisiprofiilin laskemiseen STAMP-algoritmin (Scalable Time series Anytime Matrix Profile), jonka aikavaativuus n-mittaiselle aikasarjalle on $O(n^2 \log n)$. Algoritmissa käytetään MASS-algoritmia (Mueen's Algorithm for Similarity Search), jolla lasketaan iteratiivisesti etäisyydet jokaiselle osajonolle (Mueen, ym. 2017). Sittemmin suorituskykyä on saatu parannettua STOMP (Scalable Time series Ordered-search Matrix Profile) ja SCRIMP++ algoritmeilla aikavaativuuteen $O(n^2)$ (Zhu;Imamura, ym. 2017, Zhu;Yeh, ym. 2018).

3.3 Kontekstuaalinen matriisiprofiili

De Paepe ym. (2020) täydentävät matriisiprofiilin ideaa yleisempään muotoon. Kontekstuaalinen matriisiprofiili (engl. *Contextual Matrix Profile, CMP*) voidaan ymmärtää matriisiprofiilin konfiguroitavana kaksiulotteisena versiona, jossa pidetään kirjaa useammista ”*matcheista*” määritellyillä ikkunoiden alueilla siinä missä matriisiprofiili pitää kirjaa vain yhdestä. Datan visualisoinnin lisäksi menetelmää voidaan käyttää sellaisten poikkeamien etsintään, jotka eivät ilmene uniikkeina poikkeavuuksina (engl. *discord*) vaan ovat toistuvia. Aikasarjojen etäisyyksiä voidaan myös vertailla kontekstissaan esimerkiksi niin, että

konteksti on vuorokausi ja etäisyyksiä vertaillaan viikonpäivittäin, arkipäivittäin tai viikonloppupäivien osalta.

3.4 Poikkeavuuksien havaitseminen

Wang, Bah ja Hammand (2019) käyvät läpi kartoituksessaan erilaisia poikkeavuuksien havaitsemistekniikoita. Heidän mukaansa poikkeavuudet ymmärretään usein datapisteenä, joka poikkeaa merkittävästi muista datapisteistä tai joka ei noudata odotettua normaalia kaavaa siitä ilmiöstä, jota se kuvaa. Poikkeavuuksien löytäminen voi olla haastavaa, koska rajaarvot normaaleille ja poikkeaville voi olla asetettu väärin, normaali käyttäytyminen voi kehittyä ajan kuluessa, erilaiset sovellukset ja notaatiot tekevät yhdelle kohdealueelle kehitetyistä tekniikoista vaikeasti sovellettavia muilla alueilla, sekä kohina ja häiriöt datassa jäljittelevät todellisia poikkeavuuksia datassa, mikä tekee ne vaikeaksi tunnistaa ja poistaa. Poikkeavuuksista käytetään erilaisia nimityksiä eri kohdealueilla. Chandola, Banerjee ja Kumar (2009) mainitsevat poikkeavuuksien voivan olla esimerkiksi anomalioita (engl. *anomaly*), poikkeavia havaintoja (engl. *outlier*), epätavallisia havaintoja (engl. *discordant observation*), poikkeuksia (engl. *exception*) ja uutuuksia (engl. *novelty*). Näistä anomalia ja poikkeavat havainnot ovat yleisemmin käytettyjä. Aikasarjojen osalta epätavallisella havainnolla (engl. *discord*) tarkoitetaan sellaista aikasarjaa, missä aikasarjan osajonon etäisyys lähimpään naapuriin on suuri (Chandola;Cheboli ja Kumar 2009).

Wang, Bah ja Hammand (2019) jakavat erilaiset poikkeavuuksien havaitsemismenetelmät kategorioihin. Kategoriat ovat tilastollisiin menetelmiin perustuvat metodit, etäisyyksiin perustuvat metodit, tiheyteen perustuvat metodit, klusterointimetodit, graafeihin perustuvat menetelmät, yhdistelmämetodit sekä aktiiviseen- ja syväoppimiseen pohjautuvat menetelmät. Chandola, Banerjee ja Kumar (2009) puolestaan kategorisoivat poikkeavuuksien havaitsemismenetelmät luokittelutekniikoihin, lähimmän naapurin menetelmiin, klusterointipohjaisiin tekniikoihin, tilastollisiin tekniikoihin, informaatioteoriaan perustuviin tekniikoihin ja spektriin liittyviin tekniikoihin.

Lisäksi he (Chandola;Banerjee ja Kumar 2009) erottelevat menetelmät poikkeavuuden ilmenemistyypin mukaan. Ensimmäinen ja yleisin poikkeavuustyyppi on datajoukossa esiintyvä

yksittäinen poikkeava havainto. Tämän tyypin poikkeavuuden havainnointimenetelmät perustuvat analyysiin yksittäisen havainnon relaatiosta muihin havaintoihin. Menetelmät kuuluvat ohjaamattoman oppimisen menetelmiin eli etukäteistietoa luokista ei tarvita. Toinen poikkeavuustyyppi on muuten samantapainen kuin ensimmäinen, mutta havaintoa tarkastellaan tietyssä datan rakenteellisessa kontekstissa. Konteksti määrittää sekä normaalit havainnot että poikkeavat havainnot eli on olemassa malli molemmille tapauksille, johon dataa verrataan. Tällainen poikkeavuus on poikkeavuus vain kontekstissaan, esimerkiksi tietty lämpötila voi olla kesällä normaali, mutta talvella epänormaali. Kyseiset menetelmät ovat näin ollen ohjatun oppimisen menetelmiä. Kolmannen tyypin poikkeavat havainnot muodostavat poikkeavan osajoukon koko datajoukosta rakenteen perusteella. Näissä menetelmissä on olemassa malli vain normaaleille havainnoille.

Kirjallisuudesta löytyy joitain tämän tutkielman kohdealuetta sivuavia tutkimuksia, joissa pyritään havaitsemaan poikkeavuuksia. Seuraavassa esitellään kaksi tällaista tutkimusta. Ding ym. (2016) esittelevät tutkimusartikkelissaan menetelmän, joka havaitsee mahdollisia poikkeamia suuresta määrästä laitteiston monitorointidatasarjoja. Ehdotettu menetelmä, *Latent correlation-based anomaly detection* (LCAD), havaitsee poikkeavuuksia mallintamalla piileviä korrelaatioita monista korreloivista datasarjoista käyttäen todennäköisyysjakaumamallia. Heidän mukaansa yhdessä laitteessa tapahtuva poikkeama vaikuttaa myös muihin koko laitteiston sensorien keräämiin datasarjoihin. He analysoivat 5000 laitteiston vikaa 100:n betonipumppausauton tuottamasta datasta. Laitteiston vikaantuessa, kaikkien niiden yksittäisten sensorien keräämien sarjojen joihin vika vaikutti, toiminta oli normaalien rajaarvojen sisällä. Tästä huolimatta havaittiin, että eri sarjojen suhteet toisiinsa muuttuivat poikkeaviksi.

Martí ym. (2015) esittävät artikkelissaan ”Anomaly Detection on Sensor Data in Petroleum Industry Applications” algoritmin (YASA) joka on yhdistelmä segmentointialgoritmia ja yhden luokan tukivektorikone (engl. *one-class support vector machine*) -lähestymistapaa. Algoritmi on suunniteltu havaitsemaan poikkeamat raakaöljyn käsittelyyn liittyvässä prosessissa. Prosessista kerätään dataa 6–20 koneesta, joissa kussakin on 150–300 sensoria. Data tallennetaan kustakin muuttujasta, kuten paine ja lämpötila, viiden sekunnin välein. Data on tunnuksetonta, joten käytetään ohjaamattoman oppimisen lähestymistapaa.

Artikkelissa tunnistetaan tarve käyttää aikasarjojen segmentointia datan esikäsittelytekniikkana. Segmentoinnilla pyritään tunnistamaan homogeenisia datajaksoja, joita voidaan analysoida erikseen ja jonka päätarkoitus on dimensioiden vähentäminen. Tämän jälkeen käytetään yhden luokan tukivektorikone -algoritmia määrittämään, kuuluuko uusi data tiettyyn luokkaan vai ei.

Kuten huomataan, poikkeavuuksien havaitsemiseen on olemassa lukuisia menetelmiä, joista monet hyvin tapauskohtaisia. Menetelmän valintaan vaikuttavat ainakin kohdealue sekä ongelma-alue datan luonteen, poikkeavuustyypin, tunnuksellisen datan saatavuus ja menetelmän tuottaman tuloksen muodossa.

4 Aikasarjojen klusterointi

Klusteroinnilla tarkoitetaan (yleensä moniulotteisen) datajoukon jakamista ryhmiin niin, että yhden ryhmän pisteet ovat mahdollisimman samankaltaisia keskenään ja mahdollisimman erilaisia muihin ryhmiin nähden. Datajoukosta halutaan saada selville jotain sen populaation luonteesta, onko se heterogeeninen ja esiintyykö siinä luontaisia alaluokkia. Datan jakamiseen on olemassa suuri määrä erilaisia algoritmeja, jotka voidaan jakaa kolmeen päätyyppiin: osittavat menetelmät, hierarkkiset menetelmät ja mallipohjaiset menetelmät. (Hand;Mannila ja Smyth 2001)

4.1 Aikasarjojen klusteroinnin hyödyt

Aghabozorgi, Seyed Shirkhorshidi ja Ying Wah (2015) käyvät läpi kartoituksessaan erilaisia klusterointimenetelmiä aikasarjoille. He määrittelevät klusteroinnin menetelmäksi, jolla suuri datamäärä voidaan asettaa yhdenmukaisiin ryhmiin ilman ennakkotietoa ryhmistä. Klusterointi kuuluu ohjaamattoman oppimisen menetelmiin. Dataa kerätään monenlaisista sovelluksista aikasarjojen muodossa. He esittävät aikasarjojen klusteroinnille seuraavia motivaatiotekijöitä:

1. Aikasarjadata sisältää arvokasta tietoa, joka voidaan saada esiin etsimällä ja tunnistamalla datasta hahmoja (engl. *pattern discovery*). Klusterointi on yleinen menetelmä hahmojen löytämiseksi aikasarjasta.
2. Aikasarjatietokannat ovat suurikokoisia ja ihmisen on vaikea tarkastella niitä sellaisenaan. Klusteroinnin avulla data voidaan esittää joukkona ryhmiä, joissa aikasarjat ovat jäsennelty ryhmiin, jolloin niiden tulkinta helpottuu.
3. Aikasarjojen klusterointi on eniten käytetty kartoittava tutkimustekniikka ja mahdollistaa muiden, kompleksisempien menetelmien käytön, kuten indeksointi, poikkeuksien havaitseminen ja luokittelu.
4. Aikasarjojen visualisointi klusteroinnin tuloksena voi auttaa ymmärtämään paremmin datan rakennetta, klustereita, poikkeuksia ja muita säännöllisyyksiä datajoukoissa.

Kartoituksen mukaan aikasarjojen klusterointia voidaan hyödyntää anomalioiden ja uusien yllättävien hahmojen löytämiseen sekä dynaamisten muutosten havaitsemiseen. Aikasarjojen klusterointi voidaan luokitella kolmeen eri kategoriaan, joista kaksi ensimmäistä ovat yleisimpiä. Kokonaisten aikasarjojen klusteroinnissa joukko itsenäisiä aikasarjoja klusteroidaan etäisyyden suhteen. Jos puolestaan irrotetaan yhdestä pitkästä aikasarjasta osajonoja liukuvan ikkunan avulla ja nämä segmentit klusteroidaan, on kyse osajonojen klusteroinnista. Kolmantena kategoriana on aikapisteiden klusterointi, jossa yhdistetään aikapisteiden läheisyys ja niitä vastaavien arvojen samankaltaisuus. Menetelmä vastaa segmentointia sillä erotuksella, että kaikkien datapisteiden ei tarvitse kuulua mihinkään klusteriin, vaan ne luokitellaan kohinaksi.

Keogh, Lin ja Truppel (2003) ovat todenneet aikasarjan osajonojen klusteroinnin liukuvan ikkunan menetelmällä tuottavan merkityksettömiä tuloksia, sillä lähimmät vastaavuudet löytyvät aina osajonon välittömästä läheisyydestä, kun ikkunaa liu'utetaan askel kerrallaan. Ongelma voidaan välttää rajaamalla nämä triviaalit, lähimmän etäisyyden tuottavat alueet ulos laskennasta.

Kokonaisten aikasarjojen klusterointiin on valittavissa erilaisia tekniikoita lähestymistavan mukaan (Aghabozorgi;Seyed Shirkhorshidi ja Ying Wah 2015):

1. Konventionaaliset klusterointialgoritmit kustomoidaan niin, että ne ovat yhteensopivia aikasarjojen erityispiirteiden kanssa. Yleensä etäisyyden mittari muokataan yhteensopivaksi raa'an, muunnoksettoman aikasarjadatan kanssa.
2. Aikasarjadata muunnetaan yksinkertaisiksi, staattisiksi objekteiksi, jotka ovat konventionaalisten klusterointialgoritmien syötteinä.
3. Käytetään hybridimetodia, jossa yhdistetään eri menetelmiä askeleittain.

Aikasarjojen klusteroinnille on myös olemassa luokittelu samankaltaisuusmitan mukaan. Muotoon perustuvassa lähestymistavassa sovitetaan kaksi aikasarjaa mahdollisimman hyvin aika-akselin epälineaarisen venyttämisen ja supistamisen avulla. Muotoon perustuvissa menetelmissä käytetään yleensä raakadataa ilman muunnoksia ja perinteisiä klusterointimetodeja. Piirteisiin perustuvassa lähestymistavassa raakadata muunnetaan matalamman dimension piirrevektoreiksi. Tyypillisesti raakadatasta lasketaan samanpituiset piirrevektorit

kustakin aikasarjasta ja mitataan näiden euklidinen etäisyys. Mallipohjaisissa metodeissa raakadata muunnetaan mallin parametreiksi ja valitaan mallille sopiva etäisyysmittari ja klusterointialgoritmi. Mallipohjaisien lähestymistapojen ongelmana on huono skaalautuvuus ja niiden suorituskyky huononee, kun klusterit ovat lähellä toisiaan.

4.2 Aikasarjojen dimensioiden vähentäminen

Aikasarjat ovat luonteeltaan moniulotteisia ja sisältävät runsaasti kohinaa, mikä voi aiheuttaa ongelmia tiedonlouhinnalle ja analyysille. Datapisteiden tarkasteluavaruuden ulottuvuuksia voidaan vähentää niin, että koko datamatriisin keskeiset ja olennaiset ominaisuudet säilyvät. Aikasarjadatan dimensioiden vähentämisen tulosta kutsutaan piirreavaruudeksi (Pechenizkiy, ym. 2010). Aikasarjojen dimensioiden vähentämisen tekniikkana voidaan käyttää esimerkiksi DFT:ta, DWT:ta tai PCA:ta (Han;Pei ja Kamber 2012).

4.3 Etäisyysmittarin valinta klusteroinnissa

Aikasarjojen klusterointi on voimakkaasti sidoksissa etäisyyteen. Etäisyyttä voidaan mitata erilaisilla menetelmillä, joita kuvattiin luvussa 3.1. Aghabozorgi, Seyed Shirkhorshidi ja Ying Wah (2015) listaavat aikasarjojen klusteroinnin yhteydessä käytettyjä etäisyysmittoja kuten Hausdorffin etäisyys, muunnettu Hausdorffin etäisyys (MODH), Markovin piilomalliin (engl. *Hidden Markov Model*) perustuva etäisyys, dynaaminen aikasovitus (DTW), euklidinen etäisyys, euklidinen etäisyys pääkomponenttianalyysin osa-avaruudessa ja pisin yhteinen osajono (LCSS). Kaikkein yksinkertaisin tapa mitata kahden aikasarjan etäisyys on käsitellä se yksiulotteisena aikasarjana ja laskea etäisyys jokaisessa aikapisteessä.

Ongelmia aiheuttavat aikasarjoissa yleisesti esiintyvä kohina, poikkeamat, amplitudin skaalautuminen, ajan skaalautuminen, lineaarinen liikehdintä ja epäjatkuvuudet. Jotkut etäisyysmittarit ovat herkkiä datan ”vääristymille”, jolloin klusterointi saattaa kohdistua muodon samankaltaisuuden sijaan kohinan samankaltaisuuteen. Dynaamiseen optimointiin pohjautuvat menetelmät tuottavat hyviä ja tarkkoja tuloksia, mutta ovat aikavaativuudeltaan suuria. Suurten datajoukkojen käsitellyssä mittaamisen kompleksisuutta pyritään lievittämään, jolloin tarkkuus saattaa kärsiä. Ongelmaksi saattaa muodostua myös ulottuvuuksien

vähentämisen tuloksena saadun representaation yhteensopimattomuus etäisyysmittarin kanssa. Sopivan etäisyysmittarin valinta riippuu aikasarjan ominaisuuksista, sen pituudesta, representaatiometodista ja klusteroinnin tavoitteesta kokonaisuutena. (Aghabozorgi;Seyed Shirkhorshidi ja Ying Wah 2015).

4.4 Klusterointialgoritmit

4.4.1 Hierarkkinen klusterointi

Hierarkkinen klusterointi tapahtuu joko kokoavalla (engl. *agglomerative*) algoritmilla tai jakavalla (engl. *divisive*) algoritmilla. Samankaltaisten ryhmien sisäkkäisen hierarkian luominen perustuu aikasarjojen etäisyysmatriisiin. Kokoavissa menetelmissä jokainen alkio on aluksi oma klusterinsa, joita yhdistetään vähitellen sarjana suuremmiksi ryhmiksi etäisyysmatriisiin laskettujen etäisyyksien perusteella. Vastaavasti jakavissa menetelmissä klusteroinnin alkaessa kaikki alkiot kuuluvat samaan klusteriin ja se etenee sarjana tapahtumia, jossa n kappaletta klusteroitavana olevia havaintoja jaetaan useampaan klusteriin. Hierarkkisten menetelmien heikkous on se, että klusterit ovat pysyviä eikä niitä voi enää yhdistämisen tai jakamisen tapahduttua muuttaa virheiden korjaamiseksi. Lisäksi ne skaalautuvat huonosti suurille datajoukoille, sillä ne ovat laskennallisesti raskaita. (Zaki ja Meira 2014)

Hierarkkisen klusteroinnin etuna on klustereiden hierarkian helppo kuvaaminen visuaalisessa muodossa. Hierarkkisessa klusteroinnissa ei myöskään tarvitse tietää klustereiden määrää etukäteen, mikä on reaalimaailman ongelmien kannalta erinomainen ominaisuus. Mikäli käytetään elastista etäisyysmittaa, kuten dynaamista aikasovitusta (DTW) tai LCSS:a on mahdollista klusteroida myös eri mittaisia aikasarjoja. (Aghabozorgi;Seyed Shirkhorshidi ja Ying Wah 2015)

Kokoavassa hierarkkisessa klusteroinnissa algoritmi aloitetaan niin, että jokainen piste on omassa klusterissaan. Kaksi lähintä klusteria yhdistetään toisiinsa toistuvasti niin kauan, kunnes kaikki pisteet ovat yhden klusterin jäseniä. Olkoon joukko klustereita $C = \{C_1, C_2, .., C_m\}$, löydetään lähin klusteripari C_i ja C_j ja yhdistetään ne uudeksi klusteriksi $C_{ij} = C_i \cup C_j$. Seuraavaksi päivitetään klustereiden joukko poistamalla C_i ja C_j ja lisäämällä

C_{ij} seuraavasti $C = \left(C \setminus \{C_i, C_j\}\right) \cup \{C_{ij}\}$. Prosessi toistetaan, kunnes C sisältää vain yhden klusterin. Yhdistämisprosessi on mahdollista pystyttää niin, että jäljelle jää täsmälleen k klusteria, sillä klustereiden määrä vähenee yhdellä jokaisella askeleella. Algoritmin päävaihe on määrittää klustereiden lähin pari. Tarkoitukseen voidaan käyttää useita erilaisia etäisyysmittoja, jotka kuitenkin pohjautuvat kahden pisteen väliseen etäisyyteen. Tyypillisin etäisyys on euklidinen, mutta muitakin etäisyysmittoja tai käyttäjän määrittelemää etäisyysmatriisia voidaan käyttää. (Zaki ja Meira 2014, Bramer 2016)

Lähimmän naapurin (engl. *single link*) menetelmässä etsitään kahden klusterin lähimpien jäsenten etäisyys (Zaki ja Meira 2014):

$$\delta(C_i, C_j) = min\{\delta(x, y) | x \in C_i, y \in C_j\}, \tag{4.1}$$

missä $\delta(x, y)$ on kahden pisteen etäisyys. Nimi *single link* perustuu siihen havaintoon, jos kaksi klusteria yhdistetään vain kahden pisteen minimietäisyyden perusteella, klustereiden välillä on vain yksi linkki ja muut pisteet voivat olla kaukana toisistaan ja siten klusterin läpimitta voi olla suuri. Ominaisuudesta johtuen voi muodostua ketjumaisia klustereita, mikä voi olla joko hyvä tai huono asia (Hand;Mannila ja Smyth 2001).

Kauimpaan pisteiden etäisyyteen perustuva menetelmä, jota kutsutaan myös täydelliseksi linkitykseksi (engl. *complete link*), määritellään C_i:n pisteiden ja C_j:n pisteiden maksimietäisyytenä (Zaki ja Meira 2014):

$$\delta(C_i, C_j) = max\{\delta(x, y) | x \in C_i, y \in C_j\}. \tag{4.2}$$

Jos kaikki pisteparit yhdistetään kahdesta klusterista etäisyyden ollessa enintään $\delta(C_i, C_j)$, silloin kaikki mahdolliset parit olisi yhdistetty eli saadaan täydellinen linkitys. Tällöin klusterin läpimitta säilyy pienenä ja muodostuu kompakteja klustereita (Hand;Mannila ja Smyth 2001).

Keskiarvoon perustuvassa menetelmässä (engl. *average link*) kahden klusterin välinen etäisyys määritellään parien etäisyksien keskiarvona (Zaki ja Meira 2014):

$$\delta(C_i, C_j) = \frac{\sum_{x \in C_i} \sum_{y \in C_j} \delta(x, y)}{n_i \cdot n_j}, \tag{4.3}$$

missä $n_i = |C_i|$ on pisteiden määrä klusterissa C_i.

Klusterin keskipisteeseen perustuvassa menetelmässä (engl. *centroid link*) klustereiden välinen etäisyys määritellään klustereiden keskipisteiden etäisyytenä (Zaki ja Meira 2014):

$$\delta(C_i, C_j) = \delta(\mu_i, \mu_j), \tag{4.4}$$

missä $\mu_i = \frac{1}{n_i} \sum_{x \in C_i} x$.

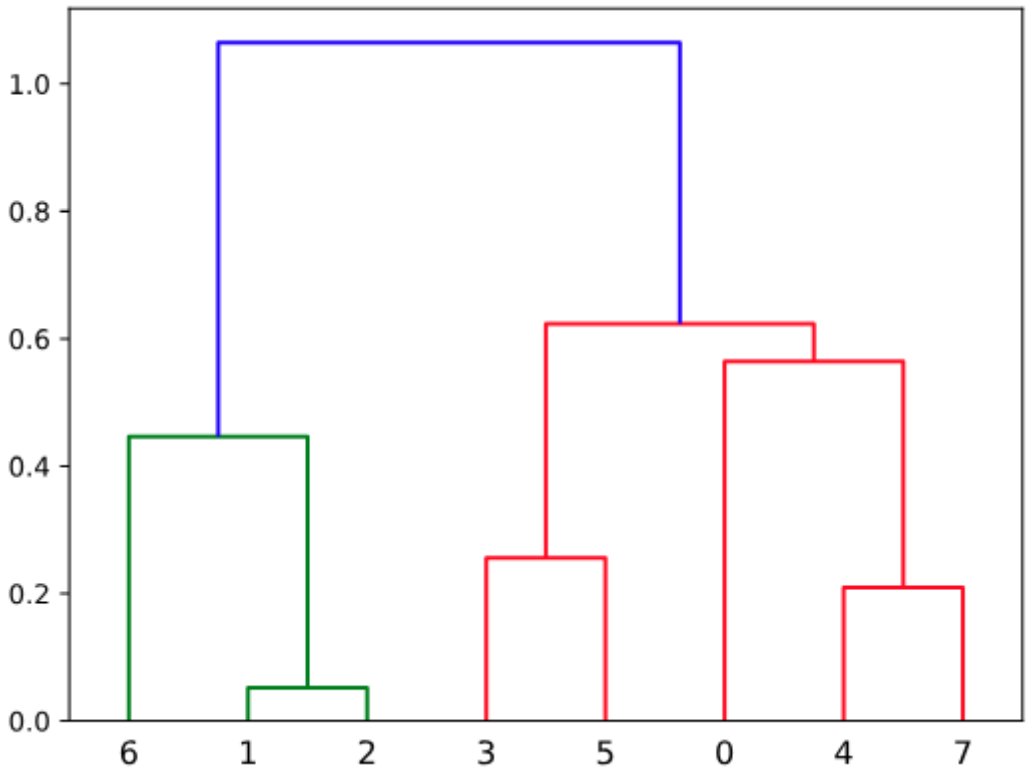
Wardin menetelmässä, eli pienimmän varianssin menetelmässä, yhdistettävät klusterit valitaan niin, että klusterin sisäisen varianssin kasvaminen minimoidaan. Klusterille C_i määritellään jäännösneliösumma (engl. *sum of squared error, SSE*) (Zaki ja Meira 2014):

$$SSE_i = \sum_{x \in C_i} \|x - \mu_i\|^2 . \tag{4.5}$$

Kahden klusterin C_i ja C_j välinen etäisyys määritellään SSE:n nettomuutoksena kun C_i ja C_j yhdistetään C_{ij}:ksi:

$$\delta(C_i, C_j) = SSE_{ij} - SSE_i - SSE_j. \tag{4.6}$$

Hierarkkisen klusteroinnin tuloksena saadaan sisäkkäisten klusterien hierakia, joka voidaan visualisoida binääripuulla eli dendrogrammilla (Kuvio 3) (Hand;Mannila ja Smyth 2001). Puun alimmalla tasolla jokainen piste on oma klusterinsa ja ylimmällä tasolla kaikki pisteet ovat osa yhtä klusteria. Näiden triviaalien tasojen välille voi muodostua mielekkäitä klustereita. (Zaki ja Meira 2014.) Pystyakselilta, kahden haaran yhtymäkohdassa, nähdään yhdistettävien klustereiden välinen etäisyys, kun ne yhdistettiin (Hand;Mannila ja Smyth 2001).



Kuvio 3. Dendrogrammi kuvaa klustereiden hierarkian

4.4.2 Osittava klusterointi

Klusterin edustajavektorin, prototyypin, etsiminen on oleellinen osatehtävä osittavien klusterointialgoritmien käytössä, kuten k-Means, k-Medoids ja Fuzzy C-Means (FCM). Aghabozorgi, Seyed Shirkhorshidi ja Ying Wah (2015) löysivät kartoituksessaan kolme aikasarjoille soveltuvaa prototyypin määrittelymetodin tyyppiä: joukon medoidi, joukon keskiarvo ja lokaali prototyypin etsintä. He toteavat, että tutkimuksissa ei ole todistettu käytettyjen menetelmien virheettömyyttä, kuitenkin prototyyppipohjaisten klustereiden laatu riippuu voimakkaasti prototyypin laadusta. Medoidin määrittäminen on eniten käytetty menetelmä, sillä keskiarvon laskeminen eri pituisista aikasarjoista ei ole triviaali tehtävä.

Osittavat klusterointimenetelmät jakavat datan siten, että kukin havainto kuuluu vain yhteen klusteriin. Osittavat menetelmät vaativat edustajavektorin määrittelyn ja niiden tarkkuus riippuu täysin tuon määrittelyn onnistumisesta ja niiden päivittämismetodista. Lisäksi

klustereiden määrä tulee tietää etukäteen. Haasteena on myös kustannusfunktion määrittäminen, jolla mitataan klusteroinnin onnistumista.

k-Means

Yksi käytetyimmistä osittavista algoritmeista on k-Means, jossa jokaisella klusterilla on sen objektien keskiarvoon perustuva edustajavektori. Perusideana on minimoida klusterin kaikkien objektien (yleensä euklidinen) kokonaisetäisyys klusterin keskuksesta (edustajavektorista). Eri pituisten aikasarjojen osalta keskiarvovektorin määrittäminen ei ole triviaali tehtävä. Ennen algoritmin suorittamista asetetaan klustereiden määrä K. Algoritmista on useita versioita, mutta perustoteutus on seuraava (Hand ym. 2001; Zaki ja Meira jr. 2014):

1. Valitaan K:n klusterin prototyypit satunnaisesti.
2. Asetetaan jokainen havaintopiste lähimpään klusteriin, eli klusteriin, jonka prototyyppiin havaintopisteellä on pienin euklidinen etäisyys.
3. Lasketaan klustereiden keskiarvovektorit uudelleen ja asetetaan ne uusiksi prototyypeiksi.
4. Toistetaan askeleita 2 ja 3, kunnes klusterit ja prototyypit eivät muutu.

k-Medoids

Yleinen tapa aikasarjojen edustajavektorin valinnassa on medoidi. Siinä missä keskiarvo on ryhmän laskennallinen edustaja, on medoidi todellinen ryhmän aikasarja, joka minimoi etäisyyden kaikkiin muihin ryhmän aikasarjoihin. Joukon kaikkien aikasarjojen välinen etäisyys lasketaan joko euklidisesti tai dynaamisella aikasovituksella (DTW) ja valitaan ryhmän medoidiksi vektori, jolla on pienin jäännösneliösumma (kaava 4.5).

4.4.3 Mallipohjainen klusterointi

Mallipohjainen klusterointi yrittää palauttaa alkuperäisen mallin datajoukosta. Jokaisella klusterilla oletetaan olevan oma malli, joka yritetään sovittaa dataan. Mallipohjaiset metodit käyttävät tyypillisesti tilastollisia lähestymistapoja, neuroverkkoja tai itseorganisoituvia

karttoja. Mallipohjaisella klusteroinnilla on kaksi yleistä heikkoutta: ensinnäkin parametrien asettaminen perustuu käyttäjän oletuksiin, jotka voivat olla virheellisiä ja toiseksi sen prosessointiaika on hidas. (Aghabozorgi;Seyed Shirkhorshidi ja Ying Wah 2015)

4.4.4 Tiheyspohjainen klusterointi

Zakin ja Meiran (2014) mukaan tiheyspohjaisessa klusteroinnissa klustereiden muodostaminen perustuu pisteiden lokaaliin tiheyteen pisteiden välisen etäisyyden sijasta. Tunnetuin tiheyspohjainen klusterointialgoritmi on DBSCAN (engl. *Density-Based Spatial Clustering of Applications with Noise*). Algoritmissa on kaksi parametria, joihin klustereiden muodostaminen perustuu. Pisteelle määritetään säde, joka määrittelee naapuruston, sekä minimimäärä pisteitä naapurustossa. Mikäli pisteen naapurusto sisältää vähintään määritellyn minimimäärän pisteitä, muodostuu klusteri.

Kantardzicn (2011) mukaan klusterit tunnistetaan selvästi korkeammasta pisteiden tiheydestä kuin klusterin ulkopuolisella alueella. Lisäksi alueilla, joilla esiintyy kohinaa, ovat tiheydeltään alhaisempia kuin mikään klusteri. Löydetyt klusterin voivat olla muodoltaan millaisia tahansa. Algoritmi on herkkä naapuruston säteen valinnalle varsinkin, jos klustereiden tiheys eroaa toisistaan.

4.5 Klusteroinnin validointi

Klusterointialgoritmit tapaavat löytää klustereita datajoukosta esiintyipä niitä tai ei (Jain 2010). Klusterointialgoritmien tuloksia tulisi voida arvioida objektiivisesti. Zaki ja Meira (2014) listaavat klusteroinnin validoinnin ja arvioinnin kolme perustehtävää: klusteroinnin arviointi pyrkii arvioimaan klusteroinnin laatua tai hyvyyttä, klusteroinnin stabiilius pyrkii ymmärtämään tulosten herkkyyttä algoritmien parametrien vaikutukselle, kuten klustereiden määrä, ja klusteroinnin tendenssi arvioi klusteroinnin mielekkyyttä eli onko datassa ylipäätään rakenteellisia ryhmiä. Näihin tehtäviin on olemassa lukuisia validointimittoja ja statistiikkoja, jotka voidaan jakaa kolmeen päätyyppiin:

1. **Ulkoinen validointi** käyttää kriteereitä, jotka eivät kuulu luontaisena osana datajoukkoon. Kriteeri voi olla esimerkiksi ennakkotieto klustereiden tunnuksista.
2. **Sisäinen validointi** käyttää datajoukosta itsestään johdettuja kriteerejä. Esimerkkinä klustereiden sisäisäisen tiiviyden ja ulkoisen erottelevuuden mittaaminen etäisyyksien avulla.
3. **Suhteellinen validointi** tähtää eri klusterointien vertaamiseen. Yleensä tämä tehdään vaihtamalla parametriasetuksia samaan algoritmiin.

Ulkoisen validoinnin kriteereihin ei yleensä voida turvautua, jos klusterointiin ryhdytään (Zaki ja Meira 2014). Yksi yleisesti käytetty sisäisen validoinnin menetelmä on Silhouette-indeksi (Rousseeuw 1987), joka mittaa sekä klustereiden sisäisäistä tiiviyttä tai samankaltaisuutta (engl. *intra*) sekä klustereiden keskinäistä eroavaisuutta (engl. *inter*). Se perustuu etäisyyden keskiarvoon lähimmän klusterin pisteisiin ja saman klusterin pisteisiin. Jokaiselle pisteelle x_i lasketaan kerroin (Zaki ja Meira 2014):

$$s_i = \frac{b(x_i) - a(x_i)}{max\{a(x_i), b(x_i)\}}, \tag{4.7}$$

missä $a(x_i)$ on pisteen x_i keskimääräinen etäisyys sen oman klusterin pisteisiin ja $b(x_i)$ pisteen x_i keskimääräinen etäisyys lähimmän klusterin pisteisiin.

Pisteen s_i arvo on välillä $[-1, +1]$. Jos s_i saa arvon lähellä +1, se indikoi, että piste x_i on lähellä oman klusterinsa pisteitä ja kaukana muiden klustereiden pisteistä. Lähellä nollaa oleva arvo indikoi, että piste on lähellä kahden klusterin rajaa. Jos piste saa lähellä -1:tä olevan arvon, piste on lähempänä muuta kuin omaa klusteriaan ja siten mahdollisesti asetettu väärään klusteriin. Kaikkien pisteiden kerroin saadaan kaikkien pisteiden s_i:n keskiarvona (Zaki ja Meira 2014):

$$SC = \frac{1}{n}\sum_{i=1}^{n} s_i\,. \tag{4.8}$$

Silhouette-indeksiä voidaan käyttää myös klustereiden määrän arvioimiseen. Lähestymistavassa huomioidaan eri k:n arvoilla laskettu SC:n arvo ja klustereiden sisäisten pisteiden s_j:n

arvot ja valitaan se k:n arvo, joka tuottaa parhaan klusterointituloksen näiden arvojen perusteella.

Sisäisen validoinnin menetelmiä on lukuisia. Muita validointi-indeksejä ovat muun muassa Dunn, Davies-Bouldin ja Calinski-Harabasz (Zaki ja Meira 2014). Silhouette-indeksin on todettu suoriutuvan yleisesti hyvin klustereiden määrän validoinnissa (Arbelaitz, ym. 2013, Jauhiainen ja Kärkkäinen 2017).

5 Panostusprosessin tiedonlouhinta

Tutkielmassa käytettävä aineisto on kerätty emulsioräjähdysaineen panostusyksiköiden toiminnasta. Kohdealueen ja datan erityspiirteet on huomioitava soveltuvien tiedonlouhintamenetelmien valinnassa ja soveltamisessa. Luvussa kuvataan tiedonlouhintaprosessi datalle KDD-menetelmän mukaisissa vaiheissa aloittaen kohdealueen esittelystä ja päättyen tulosten esittelyyn.

5.1 Kohdealue

5.1.1 Emulsioräjähdysaineet

Bulk-emulsioräjähdysaine on kaivosteollisuudessa ja kalliorakentamisessa sekä tarvekivi- tai vedenalaisessa louhinnassa käytettävä louhintaräjähdysaine (Jermakka, ym. 2015). Emulsiolla tarkoitetaan kahden toisiinsa liukenemattoman nesteen seosta. Emulsioräjähdysaineissa hapettimen vesiliuos on sekoitettu pieniksi pisaroiksi öljyn sekaan eli kyseessä on vesi-öljyssä-emulsio. (Korhonen 2005) Emulsioräjähde valmistetaan sekoittamalla ammoniumnitraatin vesiliuosta, öljyä ja lisäaineita. Emulsioräjähde sisältää tyypillisesti 70–80 % nitraattia, 10–20 % vettä, noin 4 % öljyä sekä 1–2 % lisäaineita. (Jermakka, ym. 2015) Nitraattiliuoksen ja öljyjen muodostama emulsio ei ole räjähtävä, vaikka se sisältää räjähdysaineen pääkomponentit eli hapettimen ja polttoaineen. Emulsio herkistetään niin, että se on sytytettävissä tavanomaisilla menetelmillä, ja että räjähdys etenee siinä. Emulsio herkistetään lisäämällä siihen kaasukuplia joko mekaanisesti tai kemiallisesti. Kaasukuplien on oltava sopivan kokoisia, niiden on jakauduttava tasaisesti emulsioon ja niitä on oltava riittävästi. (Korhonen 2005)

Emulsioräjähdysaineen valmistusprosessi voidaan toteuttaa kokonaan yhdessä paikassa, mutta se on myös jaettavissa niin, että osa tapahtuu tehtaalla ja osa liikkuvassa yksikössä. Herkistämätön emulsio voidaan valmistaa tehtaalla ja suorittaa herkistäminen panostusajoneuvossa. Samalla voidaan lisätä esimerkiksi kiinteää ammoniumnitraattia eli prilliä, joka lisää räjähdyskaasujen määrää. (Korhonen 2005)

Emulsioräjähdysaine panostetaan porareikään pumppaamalla. Panostustyön helpottamiseksi on kehitelty erilaisia panostuslaitteita. Emulsioräjähdysaineille kehitetyt laitteet mahdollistavat esimerkiksi lisäaineiden lisäämisen räjähdysaineeseen, jolloin saadaan muutettua sen räjähdysteknisiä ominaisuuksia.

Louhintaolosuhteet vaihtelevat: louhittavan kallion laatu voi vaihdella pehmeästä kovaan kiveen, panostettava räjäytyskenttä voi olla kostea, kivi voi rakoilla ja haluttu lohkarekoko vaihdella. Louhintaräjähdysaineen tehtävänä on kiven rikkomisen lisäksi siirtää lohkareet haluttuun paikkaan. Tarvittava panostusmäärä sovitetaan tapauskohtaisesti kivilajille ja olosuhteille sopivaksi. Halutut parametrit syötetään panostusajoneuvon logiikkaan ja laite hoitaa reikäpanoksen annostelun automaattisesti valitun reseptin mukaan. Käytetyt tekniikat mahdollistavat myös varsipanoksen keventämisen emulsioräjähteen tiheyttä muuttamalla, jotta saadaan aikaan paras mahdollinen kokonaispanos. Maanalaisessa käytössä panostuslaitteisto mahdollistaa panoksen räjähdysainemäärän säätämisen vaakarei'issä myös kesken reiän panostuksen. (T. Halonen 2015)

Räjähdysnopeus on voimakkaasti riippuvainen emulsioräjähdysaineen tiheydestä. Tiheyttä voidaan vaihdella 0,7–1,3 kg/dm³ rajoissa. 1,3:a suuremmat tiheydet eivät ole mahdollisia, koska tällöin herkistäviä kaasukuplia on liian vähän. Pienillä tiheyksillä voidaan saavuttaa 2000 m/s nopeuksia ja lähestyttäessä 1,3 kg/dm³ tiheyksiä nopeus nousee parhaimmillaan n. 6000 m/s. (Korhonen 2005)

5.1.2 Kemiitti 610

Kemiitti 610 on Forcitin panostuskohteessa valmistettava ammoniumnitraattiprilleillä lisäaineistettu emulsioräjähdysaine. Se soveltuu kaikenlaiseen kallion avolouhintaan. Tuote valmistetaan panostuskohteessa panostusajoneuvossa välivalmisteista ja panostetaan pumppaamalla. Hapetin (nitraattiliuos) ja polttoaine (öljyseos) ovat panostusajoneuvossa valmiina seoksena eli matriisina. Panostuskohteessa matriisiin lisätään ammoniumnitraattiprillejä ja se herkistetään kaasutusliuoksella valmiiksi tuotteeksi. Tuote pumpataan 40–100 metriä pitkän letkun avulla porareikään. Tuote herkistyy valmiiksi räjähdysaineeksi porareiässä tapahtuvan kemiallisen reaktion vaikutuksesta 10–30 minuutin kuluessa panostuksesta.

Panostamisen jälkeen tuotteen pinta nousee hieman porareiässä. Tuotteella saadaan porareikään progressiivinen panostus eli tiheys alenee pintaa kohti. Räjähdysainetta voidaan valmistaa ajoneuvossa olevista raaka-aineista 12–20 tonnia. Panostusnopeus vaihtelee 80–150 kg minuutissa. (Oy Forcit Ab. 2018)

5.2 Datan valinta

Tutkimusaineistona käytetään emulsiopanostusyksiköistä kerättyä prosessilaitteiston ja panostusyksiköitä operoivan henkilöstön toiminnasta kerättyä dataa. Data on luonteeltaan moniulotteista aikasarjadataa. Käytettävä data asettaa tiedonlouhinnalle haasteen useasta syystä: havaintoja on paljon, muuttujia on paljon, aikasarjat ovat eri mittaisia ja data sisältää paljon kohinaa sekä puuttuvia havaintoja. Data koostuu sensorien tuottamasta datasta sekä tapahtumapohjaisesta datasta. Tapahtumat ovat esimerkiksi operattoreiden asettamia asetusarvoja, joiden pohjalta PLC ohjaa prosessilaitteiston toiminnan. Data ei sisällä tunnuksia (engl. *label*), aiempaa luokittelutietoa eikä ennustettavia piirteitä, jonka vuoksi analysointiin käytettävä(t) menetelmä(t) tulee olla ohjamaattoman oppimisen menetelmä. Data ei sisällä henkilötietoja.

Pumppaamisen kokonaisprosessi sisältää osaprosesseja, joiden mukaan data on jaettu kahteen osaan: matriisilinjaan ja tuotelinjaan. Tämä jako johtuu siitä, että prosessin kaikkien osien ohjaus ei ole riippuvainen toisten osaprosessien toiminnasta eli esimerkiksi kaikkien komponenttien pumput eivät välttämättä ole käynnissä yhtä aikaa. Jaon ansiosta data on vertailukelpoista kaikkien reikien osalta. Tutkielmassa yksi havainto on yhden reiän pumppaamisen data. Reiät yksilöidään yksikön, päivämäärän ja reikänumeron mukaan. Reikänumero on juokseva yhdellä panostuskentällä. Prosessiarvot tallennetaan sekunnin välein. Kaikkiaan prosessiarvoja on noin 800, mutta kaikissa yksiköissä ei ole käytössä samat arvot, osa arvoista on päällekkäisiä ja osa on johdettu laskukaavojen avulla muista arvoista. Yhteistyössä kohdealueen asiantuntijoiden kanssa tarkasteluun valikoitui matriisilinjasta 7 prosessiarvoa ja tuotelinjasta 5 prosessiarvoa. Dimensioiden vähentämistekniikkana käytettiin siis tässä vaiheessa ominaisuuksien valintaa.

Forcitin toiveena oli vertailla eri yksiköiden dataa, joten aineistoon valikoitui kaksi Kemiitti 610 valmistavaa yksikköä. Yksiköstä 1 on mukana 16 panostuskentän data ja yksiköstä 2 15 panostuskentän data. Data on kerätty 1.4.2020-29.6.2020 välisenä aikana. Tiedonlouhinta suoritettiin Python ohjelmointikielen versiolla 3.7.

5.3 Esikäsittely

Datasta poistettiin niin kutsutut nollareiät. Nollareikien aikana letku täytetään vedellä ja siten data ei ole vertailukelpoista. Datan tallentamisessa on käytössä niin kutsuttu hystereesi, eli prosessiarvoa ei tallenneta, mikäli se ei ole muuttunut tarpeeksi edellisestä tallennetusta arvosta. Tällä tavalla tallennettavan datan määrää voidaan vähentää merkittävästi. Analysoinnin kannalta tämä tarkoittaa puuttuvia arvoja. Puuttuvat arvot korvattiin edellisellä arvolla. Ratkaisulla voi olla pieni vaikutus tuloksiin, koska arvon puuttuminen ei välttämättä johdu hystereesistä. Mikäli muuttujan arvot puuttuivat kokonaan jonkin muuttujan osalta, poistettiin reikä kokonaan. Esikäsittelyssä käytettiin Pandas ja Numpy kirjastoja.

Samankaltaisuuden mittaamista varten tuli määrittää ikkunan koko. Aikasarjojen pituuden persentiilit ovat taulukossa 1. Ikkunan koko $m = 20$ valikoitiin yhdessä kohdealueen asiantuntijan kanssa. Aikasarjan pituus tulee olla vähintään yhtä pitkä kuin ikkunan koko, joten alle 20 sekunnin aikasarjat poistettiin. Esikäsittelyn jälkeen yksiköstä 1 matriisilinjan dataa on 1907 reiästä ja yksiköstä 2 1814 reiästä. Tuotelinjasta samat lukemat ovat 2171 reikää ja 1765 reikää.

Taulukko 1. Aikasarjojen pituuden persentiilit

	25%	**50%**	**75%**
Aikasarjan pituus, sekuntia	22	32	47

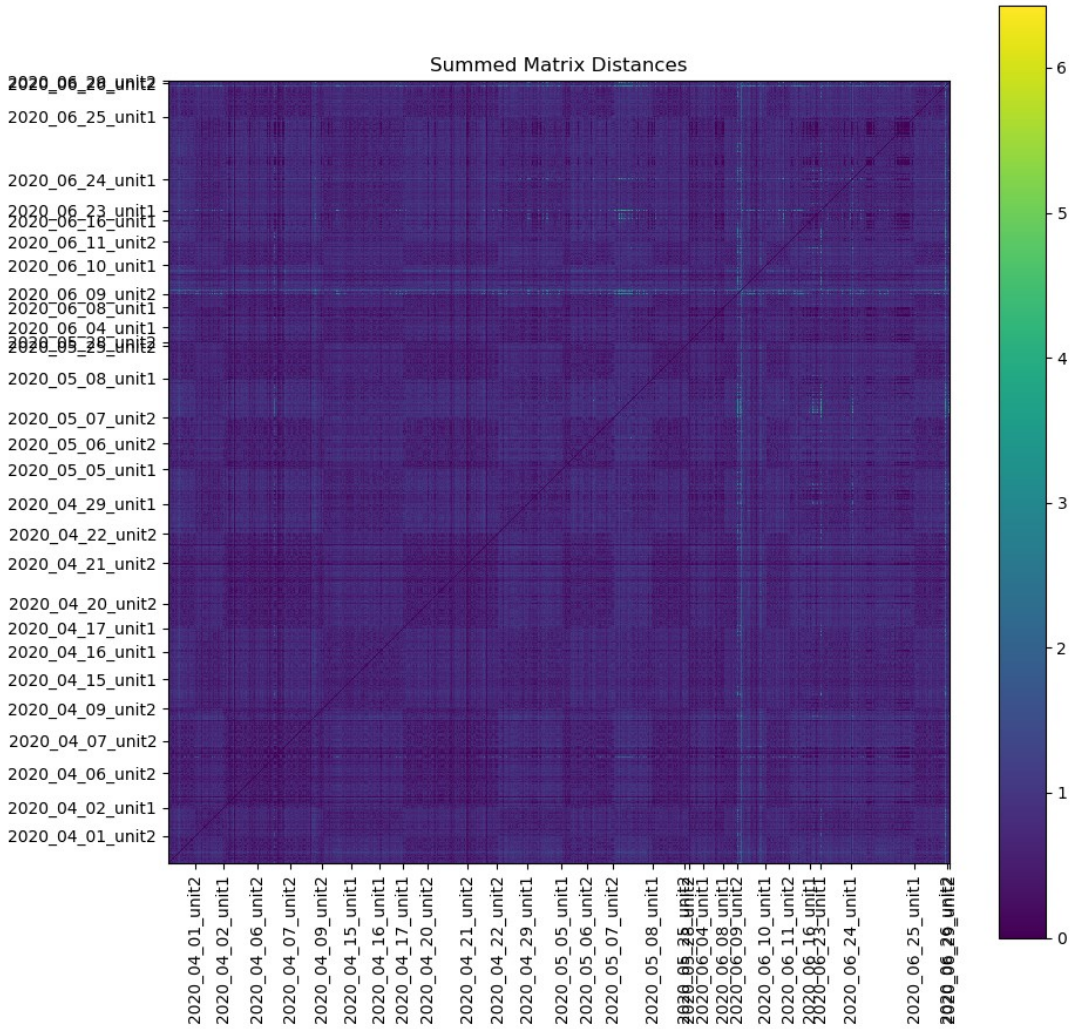
5.4 Tiedonlouhinta

Aikasarjojen samankaltaisuuden mittaamisen menetelmäksi valittiin kontekstuaalinen matriisiprofiili. Kontekstiksi määriteltiin reikien pumppaamisen välinen aika eli se aika, kun pumput ovat käynnissä. Kontekstuaalinen matriisiprofiili valittiin etäisyyden laskemisen menetelmäksi tavallisen matriisiprofiilin sijaan sen vuoksi, että tavallisessa matriisiprofiilissa liu'utettava ikkuna olisi laskenut etäisyyksiä myös kahden reiän väliltä ikkunan ollessa yhden reiän loppupäässä ja seuraavan alussa. Kontekstuaalinen matriisiprofiili muodostettiin ensin yksittäisille prosessiarvoille, jonka jälkeen etäisyydet skaalattiin min-max-skaalauksella (kaava 2.1) ja laskettiin yhteen. Matriisiprofiilin avulla laskettujen etäisyyksien pohjalta suoritettiin hierarkkinen klusterointi sekä poikkeavuuspisteytys. Hierarkkinen klusterointi valittiin, koska klustereiden määrä ei ollut etukäteen tiedossa, eikä datajoukon koko ollut liian suuri algoritmin aikavaativuuden näkökulmasta. Poikkeavuuspisteytys perustui yhden reiän keskimääräiseen etäisyyteen muihin reikiin nähden.

5.5 Tulokset

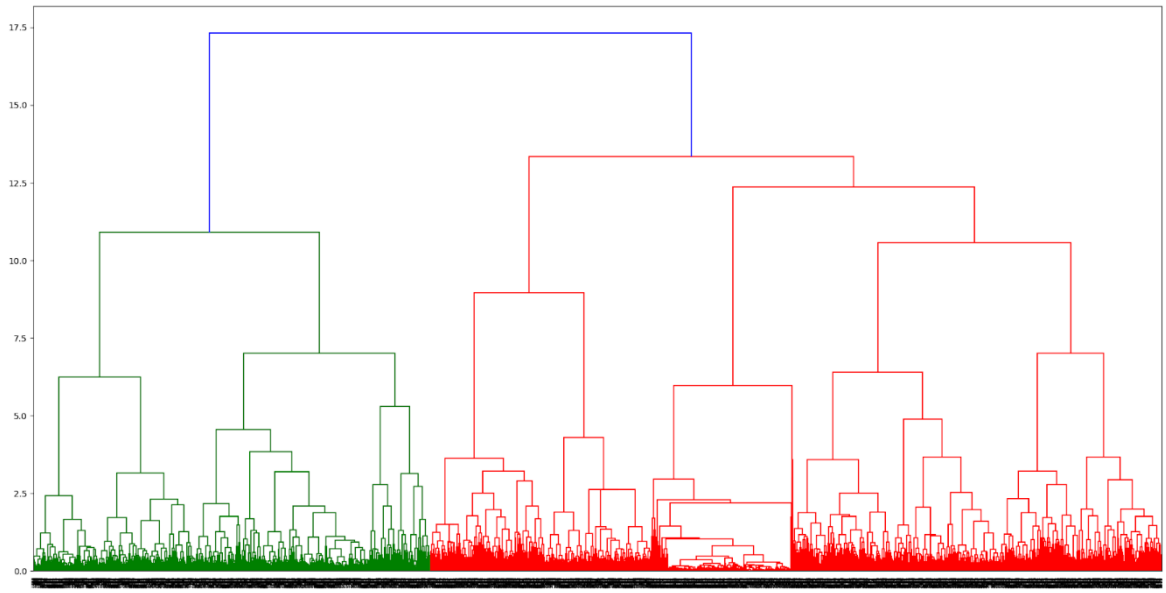
5.5.1 Tulokset standardinormaalijakautuneella datalla

Ensimmäisenä suoritettiin alkuperäisen matriisiprofiilin mukaisesti kontekstuaalisen matriisiprofiilin laskenta käyttäen standardinormaalijakauman mukaiseksi (kaava 2.2) muunnettua dataa ja etäisyyksien perusteella suoritettiin kokoava hierarkkinen klusterointi, jossa klustereiden välisen etäisyyden märittelyyn käytettiin Wardin menetelmää (kaava 4.6). Kuvion 4 etäisyyksien visualisoinnista on nähtävissä, että etäisyydet eivät ole suuria reikien välillä. Kuviossa tummempi väri merkitsee pientä etäisyyttä ja kenttä ja reikänumerot ovat aikajärjestyksessä. Matriisin koko on reikien määrä 3721×3721, sillä sovitus tehdään konteksteittain itseensä.



Kuvio 4. Matriisilinjan etäisyydet standardoidulla datalla

Matriisilinjan dendrogrammi kuviossa 5 osoittaa, että etäisyydet eri klustereiden välillä eivät ole suuria, sillä klustereiden yhdistymispisteissä korkeuksien erot näyttävät pieniltä. Dendrogrammista ei ole pääteltävissä selvästi klustereiden määrää.



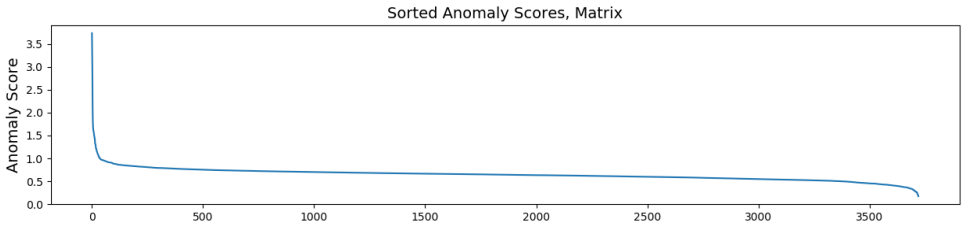
Kuvio 5. Matriisilinjan dendrogrammi standardoidulla datalla

Klustereiden määrän validointiin käytettiin Silhouette-indeksiä. Taulukossa 2 on esitetty tulokset klustereiden määrälle $k = 2 - 20$. Indeksin arvo on lähellä nollaa kaikilla klustereiden määrillä. Näyttäisi siltä, että datan standardointi tasoittaa aikasarjojen muodon niin, että ne ovat hyvin samankaltaisia ja klusterit eivät ole hyvin muodostuneita.

Taulukko 2. Matriisilinjan silhouette-indeksi standardoidulla datalla

2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
0.15	0.15	0.11	0.13	0.15	0.16	0.17	0.18	0.18	0.16	0.16	0.16	0.16	0.16	0.15	0.15	0.14	0.15	0.15

Etäisyyksistä laskettiin myös poikkeamapisteytys rei'ille. Kuviossa 6 on visualisoitu poikkeamapisteytys järjestettynä vähenevästi. Kuviosta on nähtävissä se, että muutamat reiät ovat saaneet korkean poikkeamapisteytyksen, mutta muuten etäisyydet ovat hyvin lähellä toisiaan. Standardointi näyttäisi erottelevan poikkeavat aikasarjat selvästi.

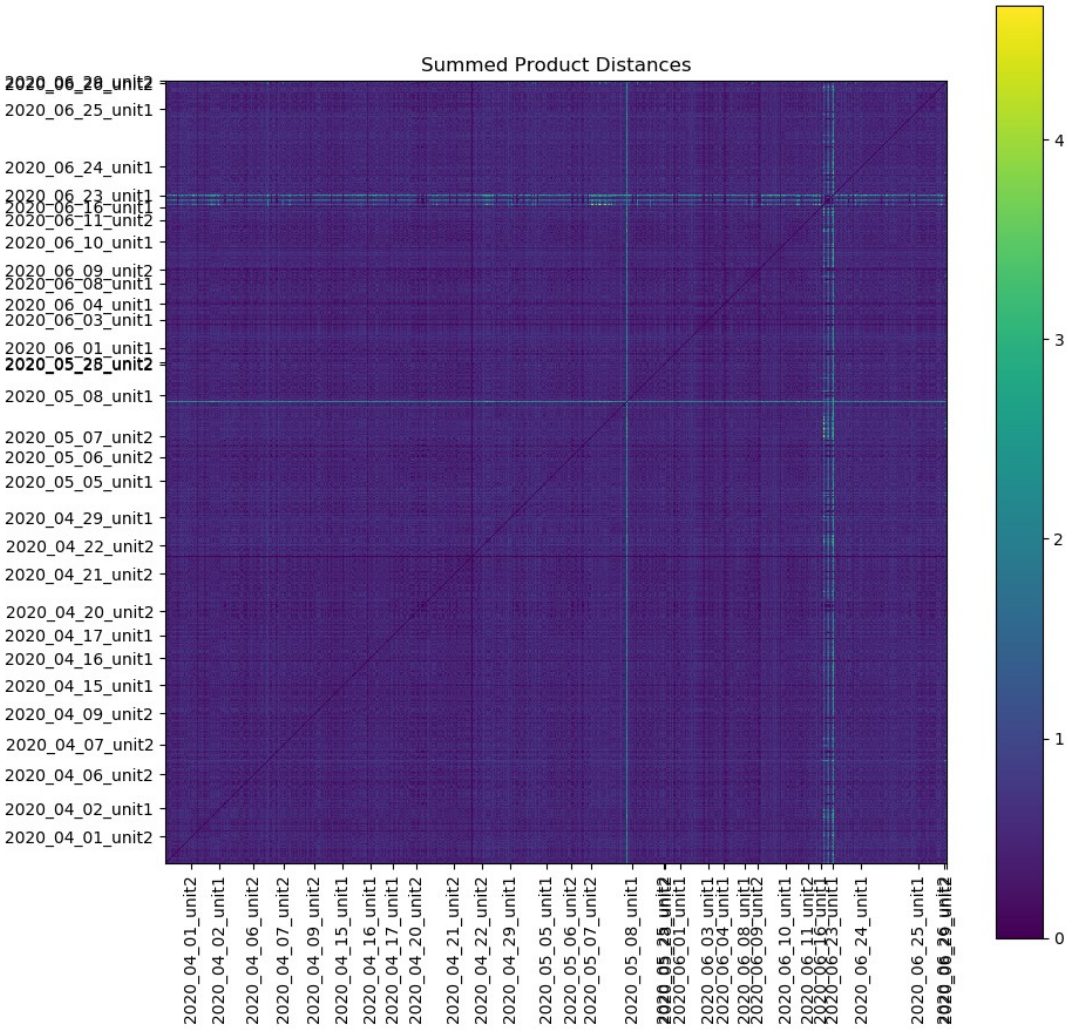


Kuvio 6. Matriisilinjan poikkeavuuspisteytys standardoidulla datalla

On normaalia, että yhden reiän pumppaus keskeytetään esimerkiksi siksi, että halutaan tarkistaa reiän täyttöaste. Automaation sijaan operaattori kuittaa reiän valmistumisen, jolloin reikänumero vaihtuu. Joskus reiän vaihtuminen saattaa jäädä epähuomiossa kuittaamatta, jolloin datassa samalla reikänumerolla on todellisuudessa pumpattu kaksi reikää. Molemmissa tapauksissa yhden reikänumeron aikana pumppaus keskeytetään ja jatketaan uudelleen. Näiden kahden tilanteen erottelemiseksi datasta ei ole ainakaan helppoa keinoa niin, että ensimmäinen tapaus on normaali ja tulisi säilyttää ja toinen poikkeus, joka pitäisi poistaa. Tällä on vaikutusta tuloksiin, useat poikkeavat reiät ovat sellaisia, joissa pumppaus on keskeytynyt ja jatkunut uudelleen. Useat reiät ovat poikkeavia edellä kuvatusta syystä ja niiden tarkempi tulkinta ei ole mielekästä eikä mahdollista tutkielman puitteissa.

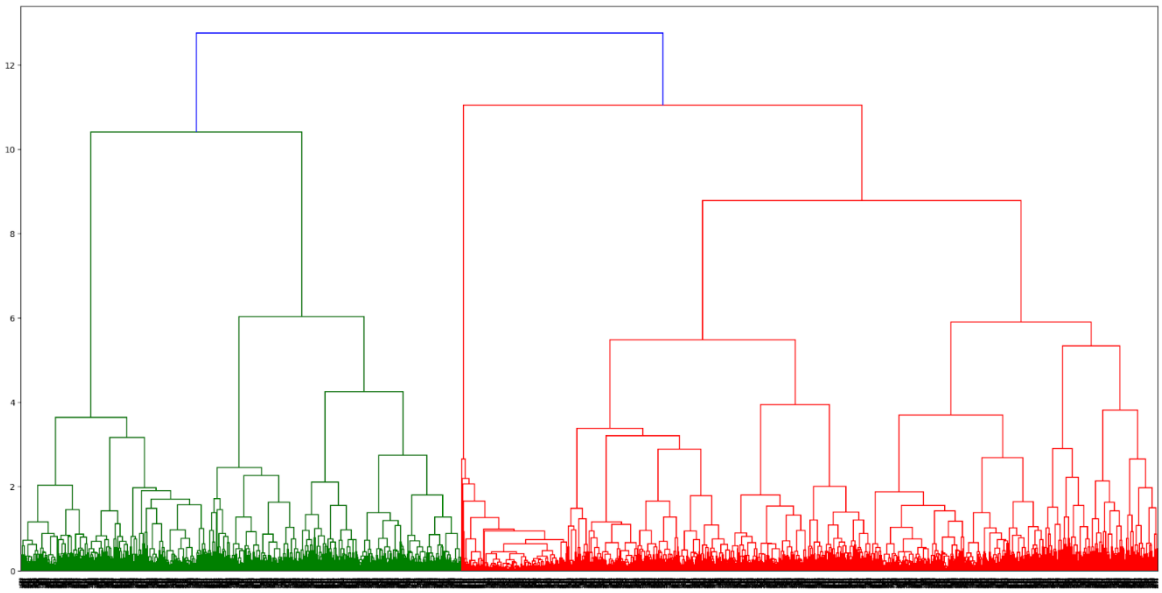
Koska eri prosessiarvojen etäisyydet on laskettu yhteen, tuloksesta ei selviä suoraan mikä tai mitkä muuttujat aiheuttavat poikkeavuuden. Matriisilinjan 20 eniten poikkeavaa reikää per prosessiarvo on visualisoitu liitteessä A. Kuvioista on havaittavissa kaksi aikasarjaa, 26.6.2020 yksikkö 2 reiät 183 ja 211, joilla on kaikissa seitsemässä prosessiarvossa poikkeava muoto. Poikkeava muoto on havaittavissa useammasta prosessiarvosta, joten kyse ei todennäköisesti ole kohinasta tai sensoriviasta. Prosessiarvossa 3 ja 4 on poikkeava reikä 29.4.2020 yksikön 1 reiässä 161 joka ilmenee myös notkahduksena prosessiarvoissa 5, 6 ja 7, mutta ei kuitenkaan käy nollassa. Tässäkin kyse on todennäköisesti todellisesta poikkeamasta.

Tuotelinjan osalta tilanne on sama, eli etäisyyksien vaihtelu on pientä. Oheisessa kuviossa (Kuvio 7) on visualisoitu tuotelinjan etäisyydet. Matriisin koko on 3936×3936. Kuviosta on nähtävissä kuitenkin muutamia poikkeavia reikiä, jotka erottuvat vaaleammalla värillä.



Kuvio 7. Tuotelinjan etäisyydet standardoidulla datalla

Tuotelinjan dendrogrammi (Kuvio 8) ei paljasta selvästi erottuvia klustereiden etäisyyksiä, sillä yhdistymispisteiden korkeuksien erot eivät ole selviä.



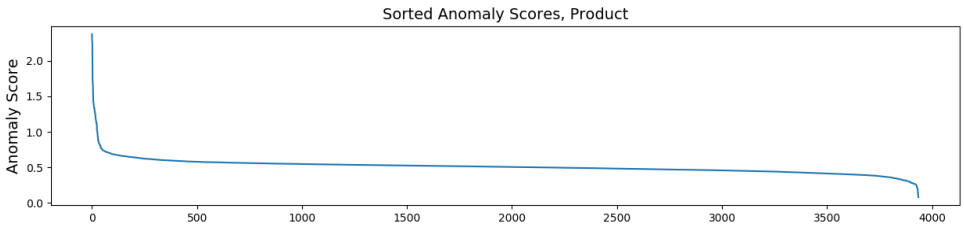
Kuvio 8. Tuotelinjan dendrogrammi standardoidulla datalla

Silhouette-indeksi arvoilla $K = 2 - 20$ (Taulukko 3) vahvistaa sen, ettei selviä klustereita muodostu, sillä arvot ovat lähellä nollaa.

Taulukko 3. Tuotelinjan silhouette-indeksi standardoidulla datalla

2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
0.15	0.08	0.12	0.14	0.13	0.13	0.12	0.13	0.13	0.13	0.13	0.12	0.11	0.12	0.11	0.11	0.16	0.11	0.10

Kuviossa 9 on esitetty tuotelinjan poikkeavuuspisteytys vähenevässä järjestyksessä. Tilanne on samantapainen kuin matriisilinjassa, etäisyyksien vaihtelu on pientä paitsi muutamat selvästi poikkeavat reiät. Datan standardointi näyttäisi erottelevan poikkeavat reiät selvästi.

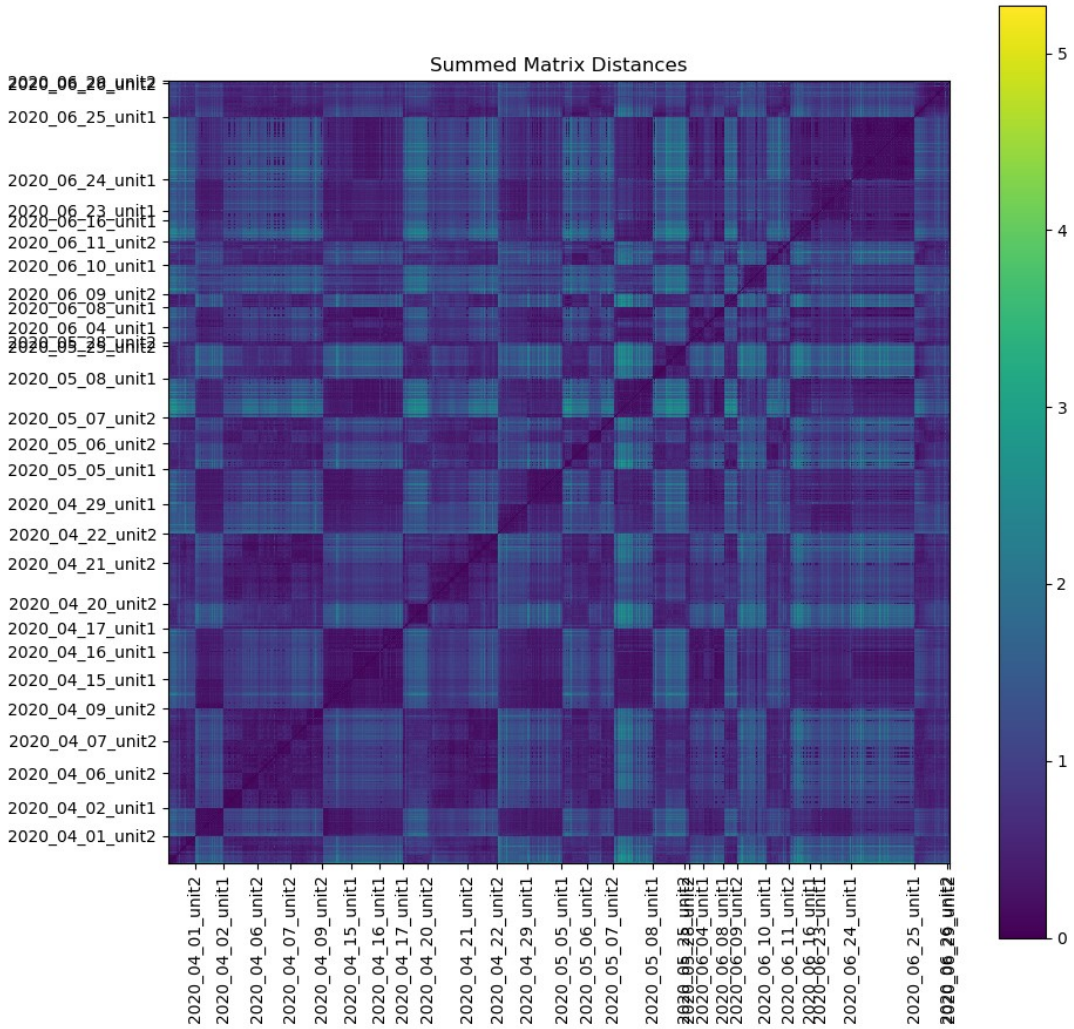


Kuvio 9. Tuotelinjan poikkeavuuspisteytys standardoidulla datalla

Liitteessä B visualisoidaan tuotelinjan eniten poikkeavat reiät. Esiin nousee 8.5.2020 yksikön 1 reikä numero 192 joka saa huomattavan korkeita arvoja kaikissa viidessä prosessiarvossa.

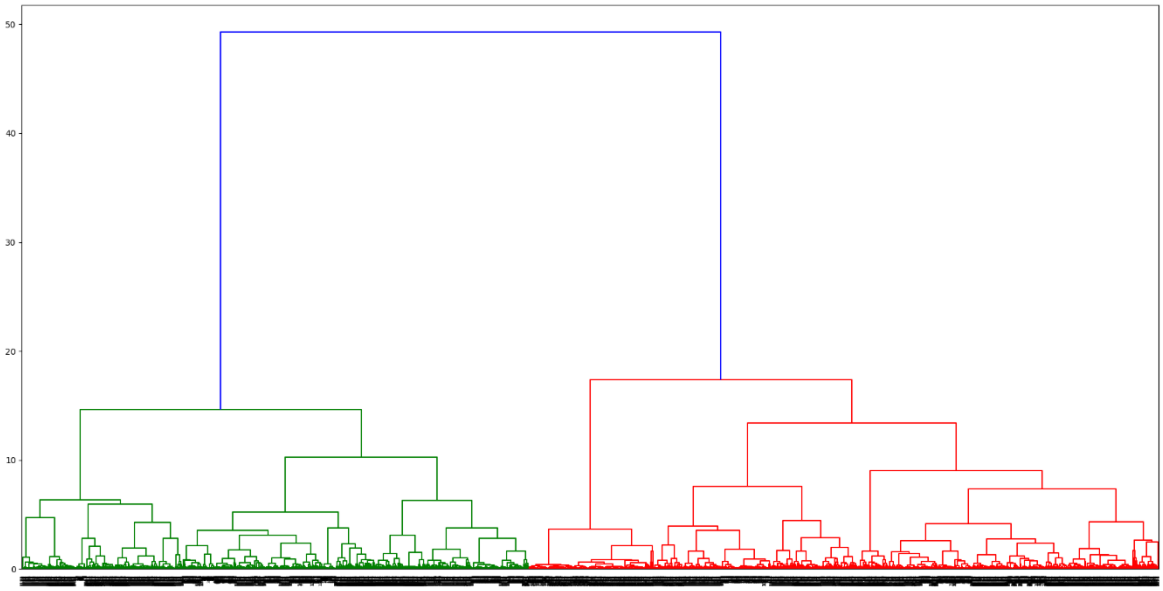
5.5.2 Tulokset raa'alla euklidisella etäisyydellä

Koska klusterointi standardinormaalijakautuneella datalla ei tuottanut hyvin klustereita erottelevaa rakennetta, haluttiin kokeilla etäisyysmitan vaihtamista suoraan euklidiseen etäisyyteen raa'alla skaalaamattomalla datalla. Kuviossa 10 on nähtävissä etäisyyksien jakaantumista yksikön mukaan matriisilinjan rei'issä.



Kuvio 10.Matriisilinjan etäisyydet raakadatalla

Matriisilinjan dendrogrammissa (Kuvio 11) etäisyys klustereiden välillä on suurimmillaan klustereiden määrällä $K = 2$.



Kuvio 11. Matriisilinjan dendrogrammi raakadatalla

Silhouette-indeksin arvot taulukossa Taulukko 4 vahvistavat dendrogrammin osoittaman. Parhaiten muodostuneet klusterit saadaan arvolla $K = 2$.

Taulukko 4. Matriisilinjan silhouette-indeksi raakadatalla

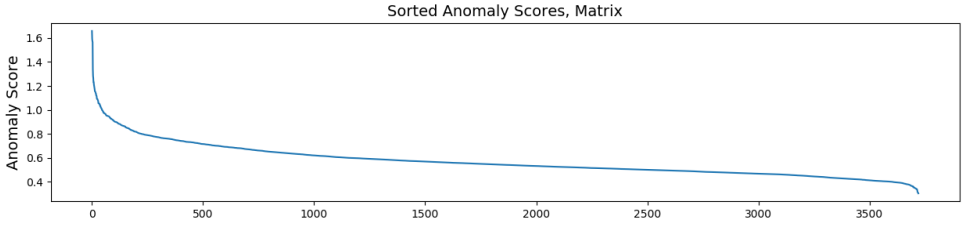
2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
0.49	0.32	0.27	0.28	0.26	0.28	0.27	0.27	0.27	0.27	0.28	0.28	0.29	0.29	0.29	0.29	0.27	0.28	0.28

Etäisyyksien visualisointi antoi vihjeen eri yksiköiden etäisyyksien eroista, joten jakaumaa haluttiin tarkastella tästä näkökulmasta. Taulukossa 5 on esitetty reikien jakautuminen klustereihin yksikön perusteella. Data jakautuu melko voimakkaasti kahteen klusteriin yksikön mukaan. Klusterissa 1 on yksi yksikön 1 reikä eli 4.6.2020 reikä 1. Yksikön 2 klusterissa 2 olevat 156 reikää jakaantuvat 15 eri kenttään.

Taulukko 5. Matriisilinjan jakautuminen klustereihin yksiköittäin

Klusteri	**1**	**2**
Yksikkö 1	1	1906
Yksikkö 2	1658	156

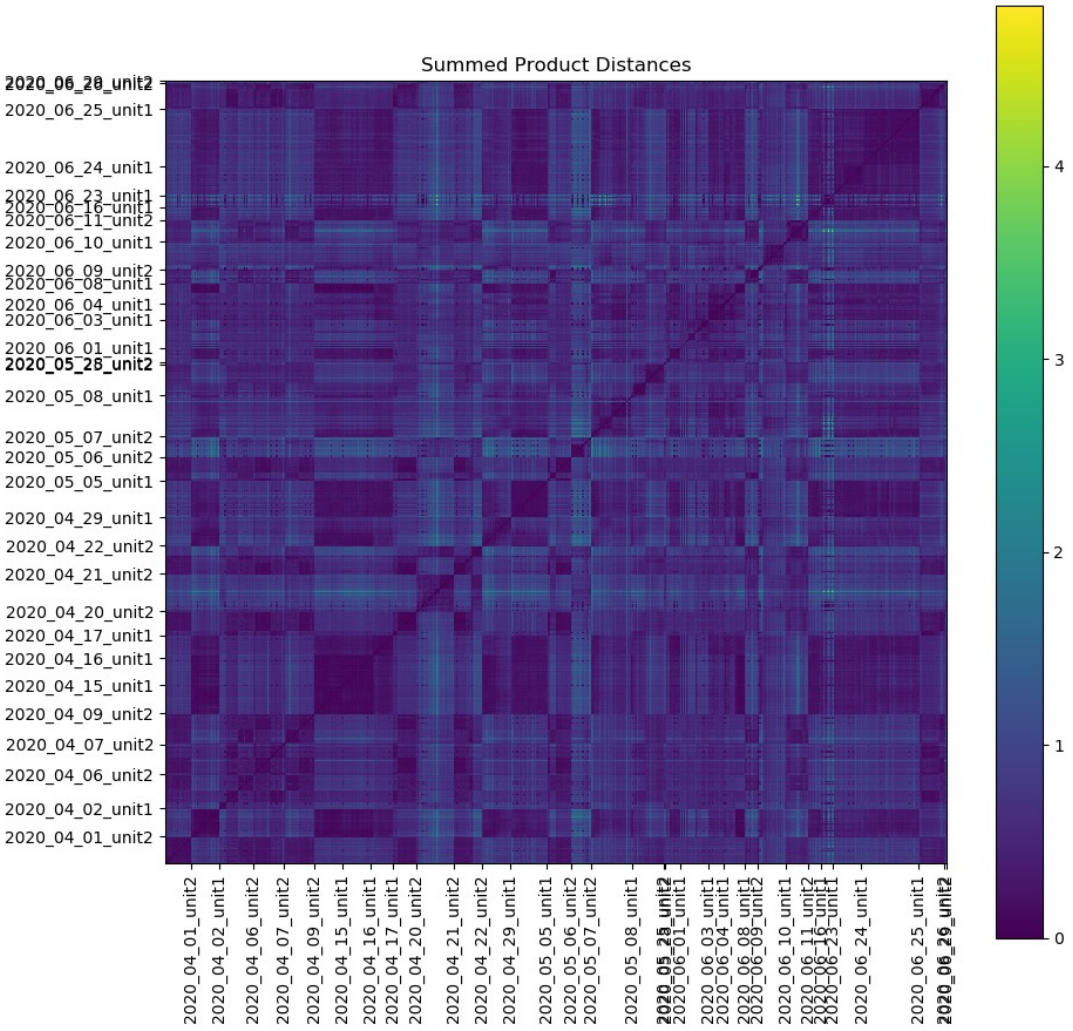
Etäisyyksistä laskettiin myös poikkeavuuspisteytys. Kuvaajasta (Kuvio 12) nähdään, että etäisyyksien vaihtelu on hieman suurempaa kuin standardoidulla datalla.



Kuvio 12.Matriisilinjan poikkeavuuspisteytys raakadatalla

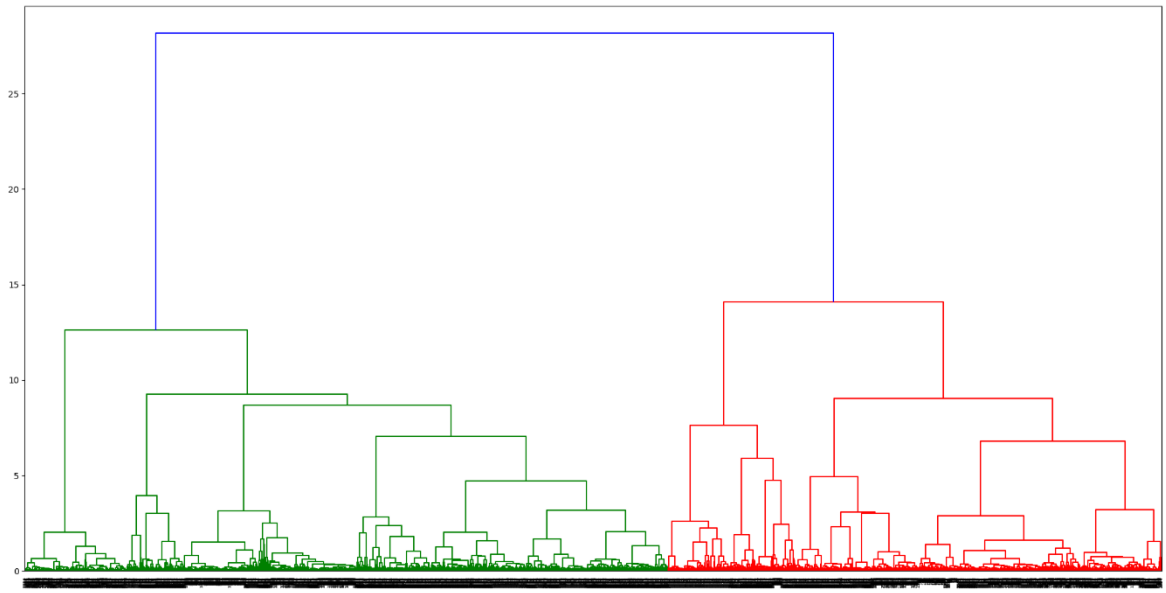
Liitteessä C esitetään 20 eniten poikkeavaa matriisilinjan reikää. Prosessiarvoissa 3 ja 4 nähdään huomattava poikkeama 10.6.2020 yksikön 1 reikänumerossa 46, mikä ei näy muissa prosessiarvoissa. Prosessiarvot 3 ja 4 ovat yhteydessä toisiinsa, eli ne mittaavat tiettyä osaprosessia. Näin ollen kyseisessä osaprosessissa on kuvaajien perusteella ollut jokin häiriö. Muiden reikien osalta nähdään normaalia korkeampia arvoja usean reiän kohdalla.

Tuotelinjan osalta raakadatalla mitatut etäisyydet on visualisoitu matriisissa (Kuvio 13). Etäisyydet näyttäisivät vaihtelevan yksiköiden mukaan.



Kuvio 13.Tuotelinjan etäisyydet raakadatalla

Kuviossa 14 on tuotelinjan dendrogrammi, josta näyttäisi erottuvan kaksi klusteria. Päätelmä perustuu suurimpaan etäisyyteen, eli korkeusero on suurin kahden klusterin kohdassa.



Kuvio 14.Tuotelinjan dendrogrammi raakadatalla

Silhouette-indeksi antaa suurimman arvon klustereiden määrälle $K = 2$ (Taulukko 6). Tosin ero muihin klustereiden määriin ei ole suuri.

Taulukko 6. Tuotelinjan silhouette-indeksi raakadatalla

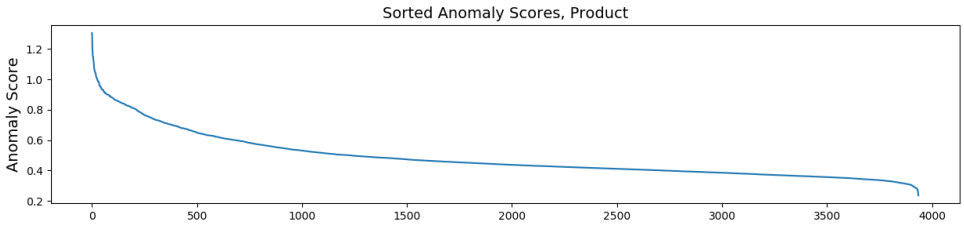
2	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
0.38	0.36	0.30	0.33	0.30	0.28	0.29	0.31	0.31	0.32	0.34	0.34	0.34	0.34	0.34	0.33	0.32	0.31	0.31

Data ei jakaudu aivan yhtä selvästi klustereihin yksikön mukaisesti kuin matriisilinja (Taulukko 7).

Taulukko 7. Tuotelinjan jakautuminen klustereihin yksiköittäin

Klusteri	**1**	**2**
Yksikkö 1	2080	91
Yksikkö 2	146	1619

Yksikön 1 toisessa klusterissa olevat 91 reikää ovat kaikki 3.6.2020 kentästä. Sen sijaan klusterissa 1 olevat 146 reikää yksiköstä 2 jakaantuvat 15 eri kentän kesken, mutta ovat samoja kuin matriisilinjan 146 poikkeavassa klusterissa olevaa reikää. Myös poikkeavuuspisteytys vaihtelee matriisilinjan tapaan enemmän raakadatalla kuin standardoidulla datalla (Kuvio 15).

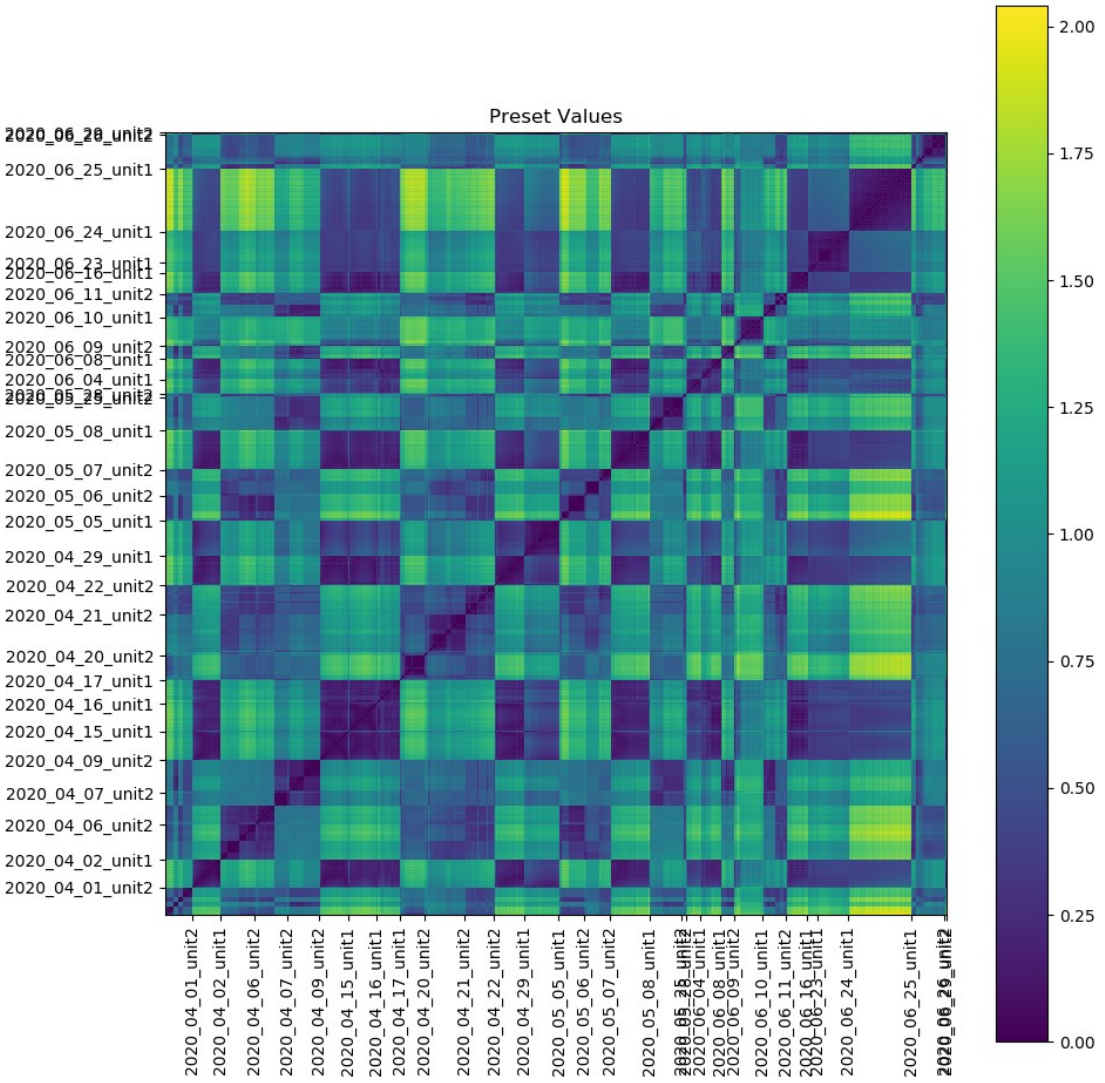


Kuvio 15.Tuotelinjan poikkeavuuspisteytys raakadatalla

Liitteessä D on visualisoitu tuotelinjan 20 eniten poikkeavaa reikää eri prosessiarvoilla. Poikkeavuuksien visualisoinnissa havaitaan poikkeava muoto prosessiarvoissa 3 ja 5 3.6.2020 yksikön 1 rei'issä 22, 24, 26 ja 27. Muiden reikien osalta nähdään poikkeavan korkeita arvoja.

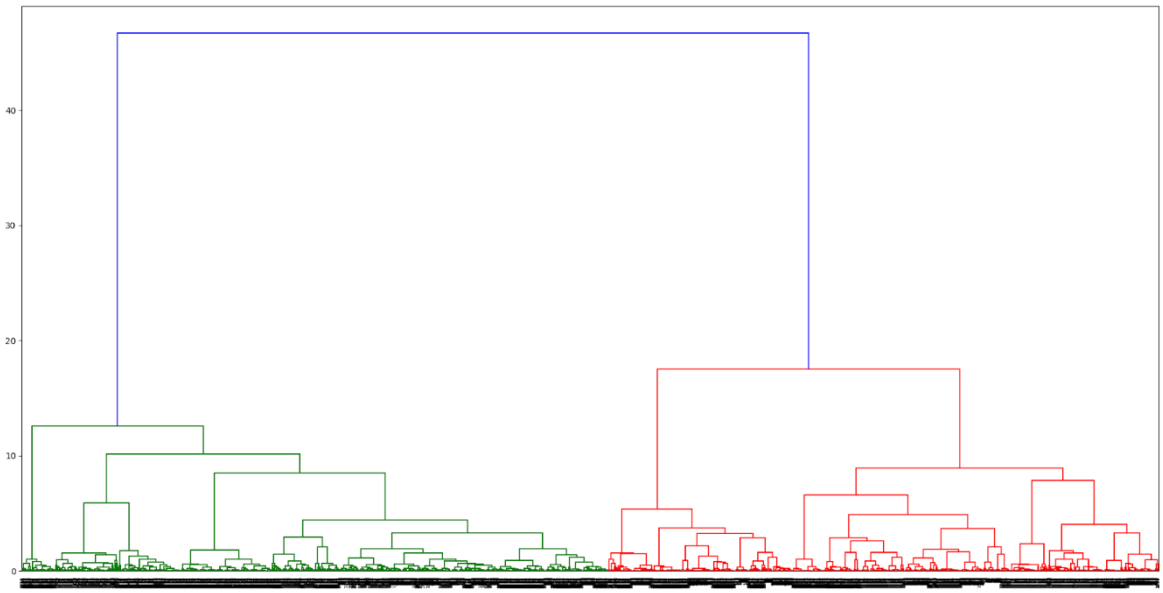
5.5.3 Asetusarvot

Prosessiarvojen tiedonlouhinnan tulosten perusteella haluttiin selvittää mahdollisia selittäviä tekijöitä tuloksille, joten analyysiin valittiin prosessin asetusarvoja. Asetusarvot eivät ole aikasarjan muodossa, vaan ne ovat staattisia välimatka-asteikollisia muuttujia. Muuttujia on 8 kappaletta. Muuttujien arvot skaalatiin min-max skaalauksella, jonka jälkeen suoritettiin kokoava hierarkkinen klusterointi, jonka etäisyysmittana käytettiin euklidista etäisyyttä. Etäisyyksien visualisointi osoittaa etäisyyksien eroavan kenttien perusteella melko voimakkaasti (Kuvio 16).



Kuvio 16.Asetusarvojen etäisyydet

Asetusarvojen dendrogrammi (Kuvio 17), Silhouette-indeksin tulokset (Taulukko 8) ja reikien jakautuminen klustereihin yksiköittäin (Taulukko 9) vahvistavat asetusarvojen olleen erilaisia kahdessa tarkasteltavassa yksikössä ja näin ollen vaikuttaneet myös toteutuneisiin prosessiarvoihin. Yksikön 2 klusterissa 1 olevat 12 reikää ovat kentästä 9.4.2020 reikä numero 1 ja 11 kappaletta reikiä kentästä 29.6.2020.



Kuvio 17.Asetusarvojen dendrogrammi

Taulukko 8. Asetusarvojen silhouette-indeksi

2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
0.57	0.51	0.50	0.41	0.39	0.41	0.42	0.44	0.47	0.45	0.44	0.43	0.43	0.44	0.44	0.40	0.42	0.43	0.44

Taulukko 9. Asetusarvojen jakautuminen klustereihin yksiköittäin

Klusteri	**1**	**2**
Yksikkö 1	1907	0
Yksikkö 2	12	1802

Eri menetelmillä löydettyjä poikkeavuuksia on esitetty liitteessä E. Taulukosta nähdään, että osa poikkeavuuksista löytyy useammalla menetelmällä, kun taas osa on havaittu vain yhdellä menetelmällä. Eri menetelmillä löydettyjä poikkeavuuksia voi selittää esimerkiksi se, että yksi menetelmä havaitsee paremmin muodon poikkeamat (normalisoitu data) ja toinen poikkeamat esimerkiksi datan skaalassa (raakadata). Menetelmän valinta ja kehittäminen riippuukin paljon siitä, mikä oikeastaan pitäisi tulkita poikkeavuudeksi kyseisellä datalla.

Taulukosta nousee esiin neljä reikää yksiköstä 2 29.6.2020. Näiden kohdalla poikkeavaa on ollut sekä asetusarvojen klusterissa, raakadatan klusteroinnissa niin matriisi- kuin tuotelinjoissa sekä poikkeavuuspisteytyksissä. Arvot eivät siis ole yksikön mukaisessa klusterissaan. Selityksenä voi olla esimerkiksi poikkeavat olosuhteet kyseisellä kentällä tai jokin muu syy poikkeaville asetusarvoille. Toisaalta esiin nousee myös yksikön 1 kenttä 8.5.2020, reikä 192. Koska klusteroinnin tulokset eivät ole poikkeavia, mutta poikkeavuus on löydetty sekä raa'alla että standardoidulla datalla, voi kyseisen reiän poikkeaman tulkita jonkinlaiseksi häiriöksi tai muuksi poikkeamaksi prosessilaitteiston toiminnassa.

Tutkielman tuloksena löydettiin siis eri menetelmillä poikkeamia eri reikien prosessiarvoissa. Poikkeamien juurisyiden analysointi ei ole tämän tutkielman puitteissa mahdollista, sillä se vaatii kohdealueen asiantuntemusta sekä muun tutkielman aineistoon kuulumattoman tiedon analysointia ja yhdistämistä tuloksiin.

6 Johtopäätökset

Tutkielman lähtökohtana oli tiedonlouhinnan menetelmien soveltaminen emulsioräjähdysaineen valmistus- ja panostusprosessista kerättyyn dataan. Tutkielman tavoitteena oli löytää uutta ja hyödyllistä tietoa ja sen avulla pyrkiä parantamaan prosessin laatua ja tuottavuutta sekä tarjota asiakkaille tarkempaa tietoa prosessista.

Ensimmäisenä tutkimuskysymyksenä oli selvittää, mitä tiedonlouhinnan menetelmiä voidaan soveltaa panostusyksiköistä kerättyyn dataan. Löydettyjä soveltuvia menetelmiä ovat klusterointi ja poikkeavuuksien havaitseminen. Datan luonteen vuoksi menetelmien tuli olla ohjaamattoman oppimisen menetelmiä. Kuten todettua, menetelmien valinta on oma taiteenlajinsa (Fayaad ym. 1996) ja riippuu niin kohdealueesta kuin datan ominaisuuksista (Aghabozorgi ym. 2015). Tiedonlouhinnan tehtävänä on löytää datasta uutta tietoa, joten erilasten menetelmien kokeileminen saattaa paljastaa jotain epätavallista ja yllättävää (Hand ym. 2001). Tutkielma tarjosi yleiskuvan erilaisista datan hyödyntämismahdollisuuksista ja pohjan tuleville tiedonlouhinnan tehtäville. Tutkielmassa esiteltiin erilaisia tiedonlouhintamenetelmiä ja aikasarjojen samankaltaisuuden mittaamismenetelmiä, jotka soveltuvat kohdealueen datalle ja joista useita voidaan kokeilla ja hyödyntää jatkossa.

Toisena tutkimuskysymyksenä oli ”Voidaanko panostusyksiköistä kerätystä datasta löytää uutta ja hyödyllistä tietoa tiedonlouhinnan menetelmillä?” Valituilla menetelmillä löydettiin datasta erityisesti poikkeavuuksia, joiden juurisyitä kannattaa tutkia. Klusteroinnin tuloksena data jakautui klustereihin yksikön mukaan. Asetusarvot vaikuttavat ymmärrettävästi toteutuneisiin prosessiarvoihin, mikä ei sinänsä ole uutta tietoa, mutta selvä ero yksiköiden välillä oli hieman yllättävää. Louhintaolosuhteiden vaikutusta asiaan ei voitu tämän tutkielman puitteissa todeta. Soveltuvien menetelmien testaaminen yhden tutkielman puitteissa ei ole mahdollista. Tutkielma eteni kuitenkin KDD-prosessin mukaisesti iteratiivisesti, eli ensimmäisen valitun menetelmän tuottamat tulokset eivät olleet kaikilta osin tyydyttäviä, joten palatiin askel taaksepäin. Uuden tiedon löytämiseksi on hyödyllistä kokeilla erilaisia menetelmiä eikä tyytyä ensimmäisen menetelmän tuottamiin tuloksiin.

Aikasarjojen samankaltaisuuden mittaaminen on tapaus- ja kohdealuekohtaista. Esimerkiksi paljon käytetyt aika-akselin sovituksen menetelmät (kuten DTW ja LCSS) soveltuvat eri

mittaisten aikasarjojen samankaltaisuuden mittaamiseen, mutta niillä saattaa olla myös ei toivottuja ominaisuuksia tietyillä kohdealueilla. Panostusyksiköiden osalta prosessilaitteiden toiminta, eli tässä tapauksessa pumppujen käynnistyminen, tuottaa tyypillisen muodon usean prosessiarvon osalta. Pumpun käynnistyminen muodostaa kuvaajaan jyrkän nousun ensimmäisten sekuntien aikana, jonka jälkeen käyrä tasoittuu. Mikäli tyypilliseen muotoon tulee viivettä, dynaamisella aikasovituksella tulkittuna muodot saatettaisiin tulkita samankaltaisina ja etäisyys pienenä, mikä kuitenkin olisi prosessilaitteiston toiminnan kannalta selkeästi poikkeavaa. Tämän vuoksi etäisyyden mittaamisen menetelmäksi ei valittu aika-akselia sovittavaa menetelmää. Tyypillisestä muodosta johtuen menetelmänä voisi kokeilla aikasarjojen katkaisemista saman mittaisiksi niin että huomioitaisiin vain todennäköisesti kiinnostavin aikasarjojen alku.

Valittua aikasarjojen samankaltaisuuden mittaamisen menetelmää, eli kontekstuaalista matriisiprofiilin avulla laskettua etäisyysmatriisia, ei ole tiettävästi käytetty tutkimuksissa aiemmin vastaavalla tavalla klusterointiin. Tutkielman perusteella voitaneen todeta, että menetelmä soveltuu aikasarjojen klusteroinnin etäisyysmitaksi, mutta menetelmä vaatii jatkotutkimusta. Menetelmällä on myös omat heikkoutensa. Mikäli aikasarjojen pituudet vaihtelevat huomattavasti, voi ikkunan koon määrittäminen olla hankalaa. Kuten esimerkiksi tämän tutkielman kohdalla, jos kaksi aikasarjaa ovat pituudeltaan yli kaksinkertaisia ikkunan kokoon verrattuna, voi lyhin etäisyys löytyä molempien sarjojen loppupäästä ja kiinnostavat poikkeavuudet muodossa jäädä havaitsematta. Lisäksi eri prosessiarvojen etäisyyksien yhdistäminen tekee tulosten tulkinnasta haastavaa. Tutkimuksessa esitettyä menetelmää voisi jatkokehittää soveltaen selittävän koneoppimisen menetelmiä (engl. *interpretable machine learning*).

Kohdealueelle on löydettävissä useita jatkotutkimuskohteita. Tutkielmassa ei ollut käytettävissä esimerkiksi emulsion kaasuuntumismittauksien tuloksia eikä tietoa panostuskohteesta ja siihen liittyvistä louhintaolosuhteista. Mittaustuloksia voisi hyödyntää vertaamalla eri olosuhteissa ja eri asetusarvoilla pumpattua emulsiota toteutuneisiin emulsion tiheyksiin.

Panostusprosessiin liittyy runsaasti sellaisia tilanteita, jotka ovat normaaleja toiminnan kannalta, mutta näyttäytyvät datan kannalta poikkeamina ja niiden tapahtumisfrekvenssi on

harvinainen datan kokonaismäärään verrattuna. Datan hyödyntämisen kannalta olisi hyödyllistä systemaattisesti dokumentoida tapahtumat ja tunnistaa niiden vaikutus dataan. Tunnistamisen myötä datan esikäsittelyä olisi mahdollista lisätä ja tarkentaa, ja siten saada tiedonlouhinnan menetelmillä hyödyllistä uutta tietoa. Mikäli laitteiston vikaantumisten tai muiden häiriöiden ajankohdat ja oleelliset yksityiskohdat yhdistetään tulevaisuudessa prosessidataan, voidaan tietoa hyödyntää vertaamalla vikaantumisen aikaista ja edeltävää dataa esimerkiksi matriisiprofiilin avulla muuhun dataan. Tällä tavalla voidaan mahdollisesti ennakoida vikaantuminen ja asettaa hälytyksiä niin, että laitteisto voidaan huoltaa ennen vian esiintymistä. Lisäksi voidaan siirtyä ohjatun tiedonlouhinnan menetelmiin kuten luokitteluun.

Datan hyödyntämispotentiaali on monipuolinen, kun keräämisprosessi on yhdenmukainen eri yksiköissä. Kuten tyypillistä, kerätty data sisälsi puuttuvia arvoja mikä hankaloitti analyysiä sekä haittaa yleisestikin kerätyn datan hyödyntämistä. Lisäksi datassa on jonkun verran päällekkäisiä muuttujia, jotka mittaavat samaa asiaa. Turhien muuttujien keräämisestä tulee turhia kustannuksia datan määrän kasvaessa jatkuvasti. Eri yksiköiden vertaaminen ei aina ole mielekästä komponenttierojen vuoksi, mutta esitettyjä menetelmiä voidaan soveltaa vain yhteen yksikköön kerrallaan. Tutkielma tarjosi hyvän lähtökohdan datan hyödyntämiseen emulsioräjähdysaineen valmistamisen ja panostamisen prosessin kehittämisessä. Tulosten tarkempi tulkinta, analysointi ja hyödyntäminen jää kohdealueen asiantuntijoiden tehtäväksi.

Lähteet

Aghabozorgi, Saeed, Ali Seyed Shirkhorshidi, ja Teh Ying Wah. ”Time-series clustering – A decade review.” *Information Systems* 53 (2015): 16 - 38.

Akbarinia, Reza, ja Bertrand Cloez. ”Efficient Matrix Profile Computation Using Differ-ent Distance Functions.” *ArXiv.org*, 2019.

Arbelaitz, Olatz, Ibai Gurrutxaga, Javier Muguerza, Jesús M. Pérez, ja Iñigo Perona. ”An extensive comparative study of cluster validity indices.” *Pattern Recognition* 46, nro 1 (2013): 243 - 256.

Bellman, Richard. ”Adaptive control process: a guided tour.” (Princeton University Press) 1961.

Bramer, Max. *Principles of Data Mining.* London: Springer-Verlag, 2016.

Chandola, Varun, Arindam Banerjee, ja Vipin Kumar. *Anomly Detection: A Survey.* ACM Computing Surveys 41, no. 3 3:14.1-15.58, 2009.

Chandola, Varun, Deepthi Cheboli, ja Vipin Kumar. *Detecting Anomalies in a Time Series Database.* Technical Report, University of Minnesota Digital Conservancy, 2009.

De Paepe, Dieter, ym. ”A generalized matrix profile framework with support for contextual series analysis.” *Engineering Applications of Artificial Intelligence*, 2020.

Ding, Jianwei, Yingbo Liu, Li Zhang, Jianmin Wang, ja Yonghong Liu. ”An anomaly detection approach for multiple monitoring data series based on latent correlation probabilistic model.” *Applied Intelligence* 44, nro 2 (2016): 340-361.

Fayyad, Usama, Gregory Piatetsky-Shapiro, ja Padhraic Smyth. ”From Data Mining to Knowledge Discovery in Databases.” *AI Magazine* 17, nro 3 (1996a): 37.

—. ”Knowledge Discovery and Data Mining: Towards a Unifying Framework.” *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.* Portland, Oregon: AAAI Press, 1996b. 82 - 88.

Fayyad, Usama, Gregory Piatetsky-Shapiro, ja Padhraic Smyth. ”The KDD Process for Extracting Useful Knowledge from Volumes of Data.” *Communications of the ACM* 39, nro 11 (1996c): 27 - 34.

Halonen, Mari, ja Elina Kähäri. ”Tuotekehityksen tulos: ForDEX-järjestelmäkokonaisuus tuo työhön helppoutta ja joustavuutta digitaalisilla työkaluilla. Ruutiset, 2:5-6. .” *Ruutiset* (Oy Forcit Ab), nro 12 (2018): 5 - 6.

Halonen, Tommi. ”Räjäytystyöt ja -kalusto.” Teoksessa *Kaivos- ja louhintatekniikka*, tekijä: Pekka Lappalainen, Antero Hakapää ja Tauno Paalumäki, 183 - 199. Helsinki: Opetushallitus: Kaivosteollisuus, 2015.

Han, Jiawei, Jian Pei, ja Micheline Kamber. *Data Mining: Concepts and Techniques.* Kolmas painos. Waltham, Mass.: Morgan Kaufmann, 2012.

Hand, David, Heikki Mannila, ja Padhraic Smyth. *Principles of data mining.* Cambridge: MIT Press, 2001.

Jain, Anil K. ”Data clustering: 50 years beyond K-means.” *Pattern Recognition Letters* 31, nro 8 (2010): 651 - 666.

Jauhiainen, Susanne, ja Tommi Kärkkäinen. ”A Simple Cluster Validation Index with Maximal Coverage.” *ESANN 2017 : Proceedings of the 25th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning.* ESANN, 2017. 293-298.

Jermakka, Johannes, ym. *Nitrogen compounds at mines and quarries : Sources, behaviour and removal from mine and quarry waters - Literature study.* VTT Technology No. 226, Espoo: VTT Technical Research Centre of Finland, 2015, 144.

Kantardzic, Mehmed. *Data mining : concepts, models, methods, and algorithms.* Hoboken, New Jersey: Jhon Wiley, 2011.

Kauppila, Päivi, Hanna Lampinen, Lauri Siirama, ja Pekka Suomela. ”Ympäristövaikutusten hallinta ja ohjaus.” Teoksessa *Kaivos- ja louhintatekniikka*, tekijä: Pekka Lappalainen,

Antero Hakapää ja Tauno Paalumäki, 429 - 450. Helsinki: Opetushallitus: Kaivosteollisuus, 2015.

Keogh, Eamonn, ja Chotirat Ann Ratanamahatana. ”Exact Indexing of Dynamic Time Warping Knowledge and Information Systems.” *Knowledge and Information Systems* 7, nro 3 (2005): 358 - 386.

Keogh, Eamonn, Jessica Lin, ja Wagne Truppel. ”Clustering of time series subsequences is meaningless: implications for previous and future research.” *Proceedings of the Third IEEE International Conference on Data Mining.* IEEE Computer Society, 2003. 115 - 122.

Korhonen, Pirjo. *Räjähdekirja.* Helsinki: Hanko: Suomen kemian seura; Räjähdeyhdistys, 2005.

Lukka, Kari. *Kari Lukka: Konstruktiivinen tutkimusote.* 2001. https://metodix.fi/2014/05/19/lukka-konstruktiivinen-tutkimusote/.

Martí, Luis, Nayat Sanchez-Pi, Jose Manuel Molina, ja Ana Cristina Bicharra Garcia. ”Anomaly Detection Based on Sensor Data in Petroleum Industry Applications.” *Sensors* 15, nro 2 (2015): 2774-2797.

Mueen, Abdullah, ym. ”The Fastest Similarity Search Algorithm for Time Series Subsequences under Euclidean Distance.” 2017.

Oy Forcit Ab. *Yritys.* 2019. https://www.forcit.fi/fi/forcit/yritys/ (haettu 13. 12 2019).

Oy Forcit Ab. ”Kemiitti 610 Tuotetieto 31.7.2018.” 2018. https://forcit.fi/assets/product-brochures/KEMIITTI-610-INFO-FI.pdf.

Pechenizkiy, Mykola, Andriy Ivannikov, Sami Äyrämö, ja Tommi Kärkkäinen. *Towards better understanding and control of CFB-boilers: review of recent research in mining time series data.* University of Jyväskylä, Jyväskylä: Reports of the Department of Mathematical Information Technology / University of Jyväskylä. Series C, Software and computational engineering, 2010.

Rousseeuw, Peter J. ”Silhouettes: A graphical aid to the interpretation and validation of cluster analysis.” *Journal of Computational and Applied Mathematics* 20 (1987): 53 - 65.

Shukla, Satya Narayan, ja Benjamin M Marlin. ”A Survey on Principles, Models and Methods for Learning from Irregularly Sampled Time Series.” *arXiv.org* (Cornell University Library), 2021.

Verleysen, Michel, ja Damien François. ”The Curse of Dimensionality in Data Mining and Time Series Prediction.” *Computational Intelligence and Bioinspired.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2005. 758-770.

Vlachos, Michail, Marios Hadjieleftheriou, Dimitrios Gunopulos, ja Eamonn Keogh. ”Indexing multi-dimensional time-series with support for multiple distance measures.” *KDD '03: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* New York, NY, USA: Association for Computing Machinery, 2003.

Wang, Hongzhi, Mohamed Bah, ja Mohamed Hammad. ”Progress in Outlier Detection Techniques: A Survey.” *IEEE Access* 7 (2019): 107964-108000.

Wang, Xiaozhe, Anthony Wirth, ja Liang Wang. ”Structure-Based Statistical Features and Multivariate Time Series Clustering.” *Seventh IEEE International Conference on Data Mining (ICDM 2007).* 2007. 351-360.

Wuest, Thorsten, Daniel Weimer, Christopher Irgens, ja Klaus-Dieter Thoben. ”Machine learning in manufacturing: advantages, challenges, and applications.” *Production & Manufacturing Research* (Taylor & Francis) 4, nro 1 (2016): 23 - 45.

Yeh, Chin-Chia Michael, ym. ”Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets.” *2016 IEEE 16th International Conference on Data Mining (ICDM).* 2016. 1317-1322.

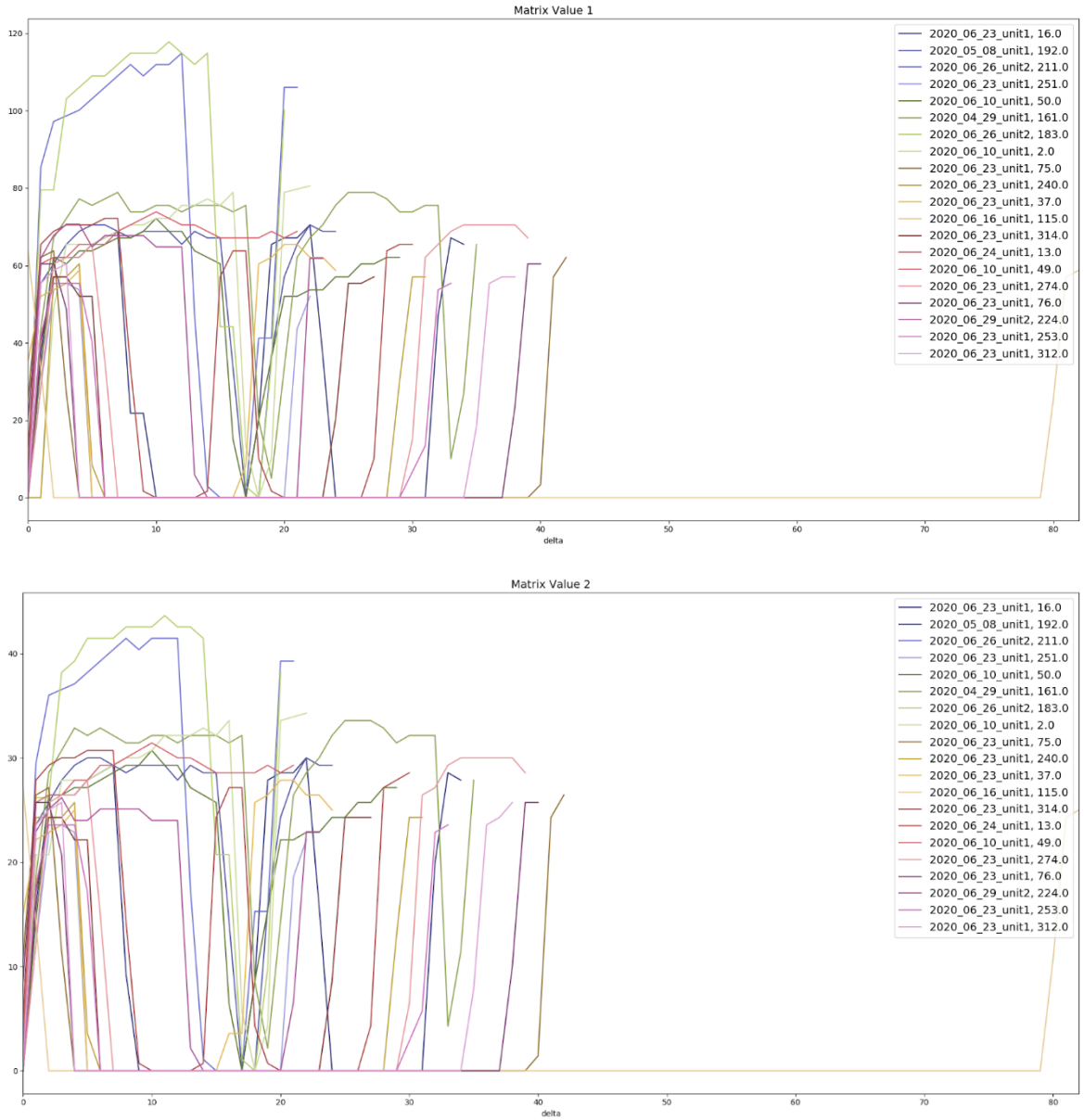
Zaki, Mohammed J., ja Wagner Meira. *Data mining and analysis : fundamental concepts and algorithms.* New York: Cambridge University Press 2014, 2014.

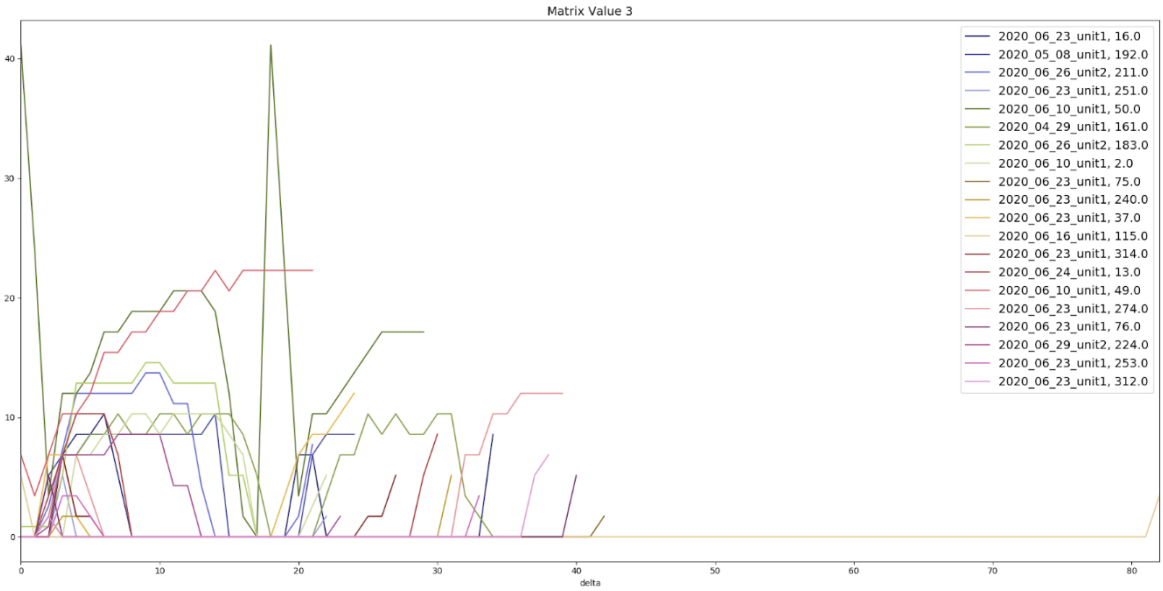
Zhu, Yan, Chin-Chia Michael Yeh, Zachary Zimmerman, Kaveh Kamgar, ja Eamonn Keogh. ”Matrix Profile XI: SCRIMP++: Time Series Motif Discovery at Interactive Speeds.” *2018 IEEE Internation-al Conference on Data Mining (ICDM).* 2018. 837 - 846.

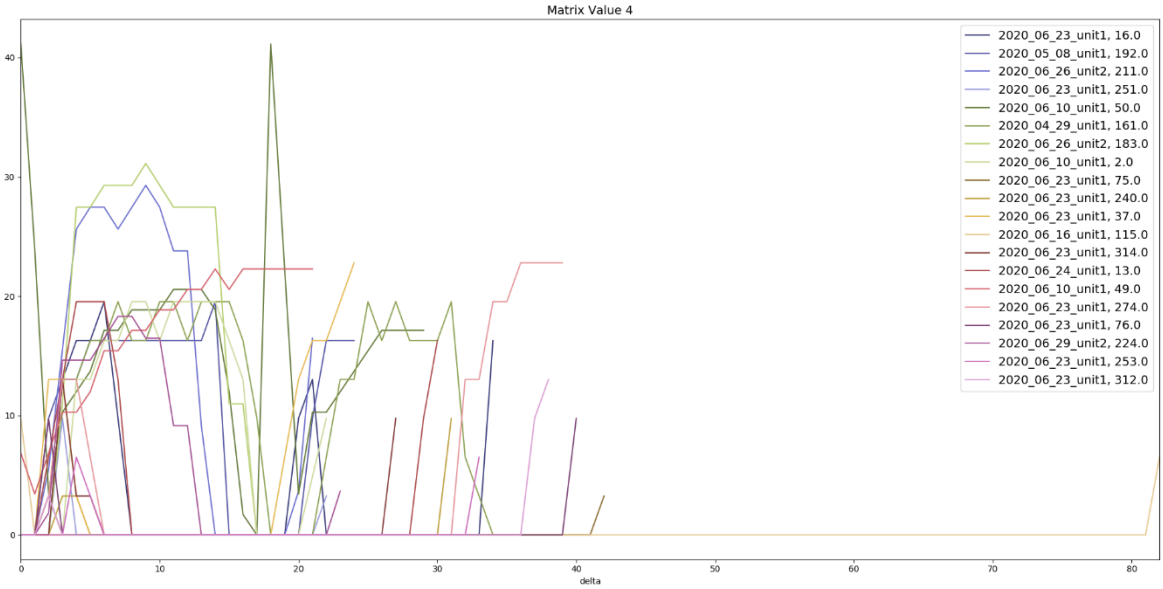
Zhu, Yan, Makoto Imamura, Daniel Nikovski, ja Eamonn Keogh. ”Matrix Profile VII: Time Series Chains: A New Primitive for Time Series Data Mining.” *2017 IEEE International Conference on Data Mining (ICDM).* 2017. 695 - 704.

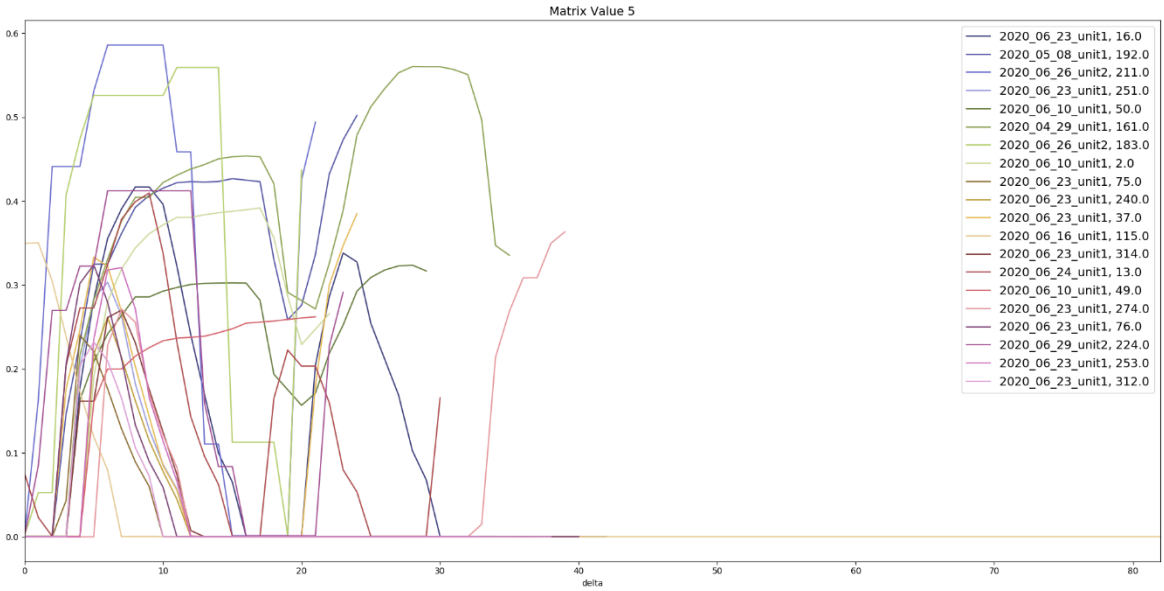
Liitteet

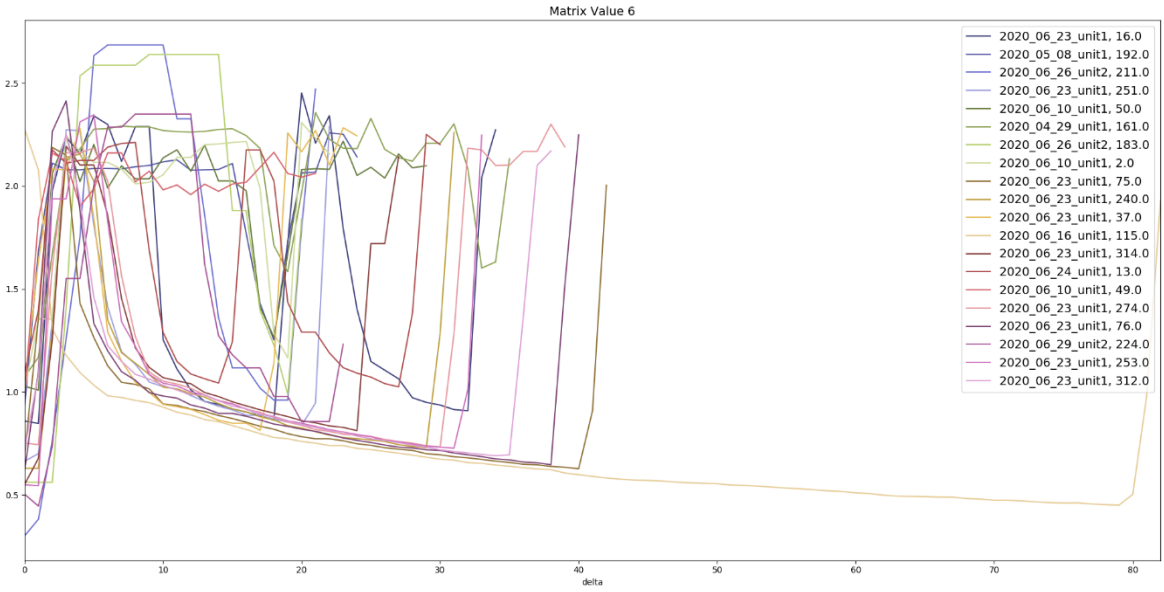
A Matriisilinjan 20 eniten poikkeavaa reikää standardoidulla datalla, prosessiarvot 1–7

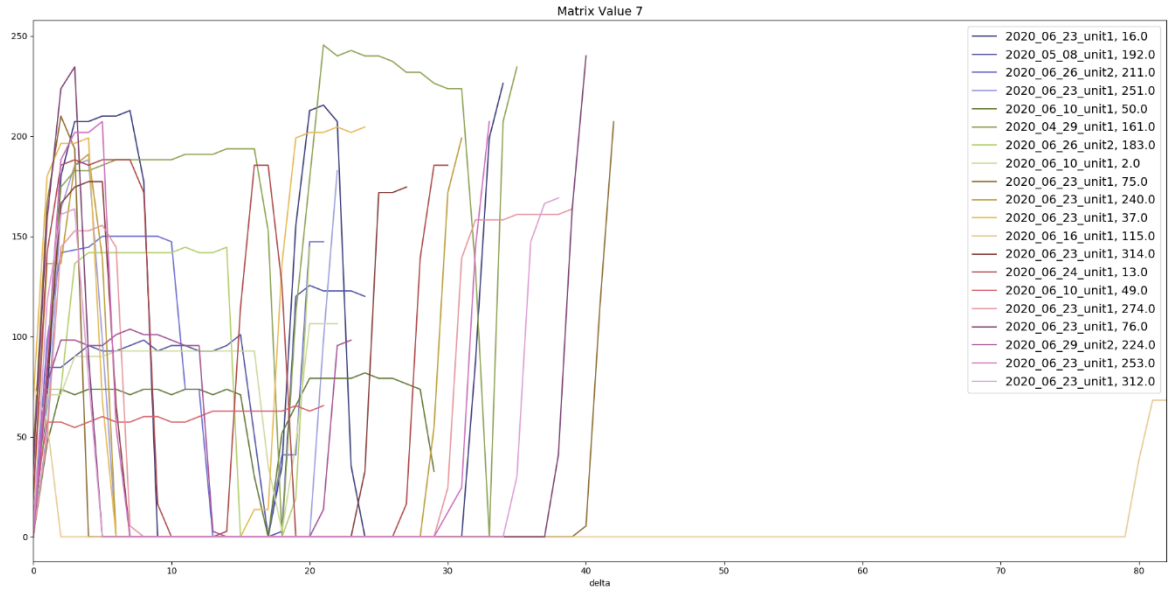



Matrix Value 3
2020_06_23_unit1, 16.0
2020_05_08_unit1, 192.0
2020_06_26_unit2, 211.0
2020_06_23_unit1, 251.0
2020_06_10_unit1, 50.0
2020_04_29_unit1, 161.0
2020_06_26_unit2, 183.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 37.0
2020_06_16_unit1, 115.0
2020_06_23_unit1, 314.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 49.0
2020_06_23_unit1, 274.0
2020_06_23_unit1, 76.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 312.0
0
10
20
30
40
0
10
20
30
40
50
60
70
80
delta

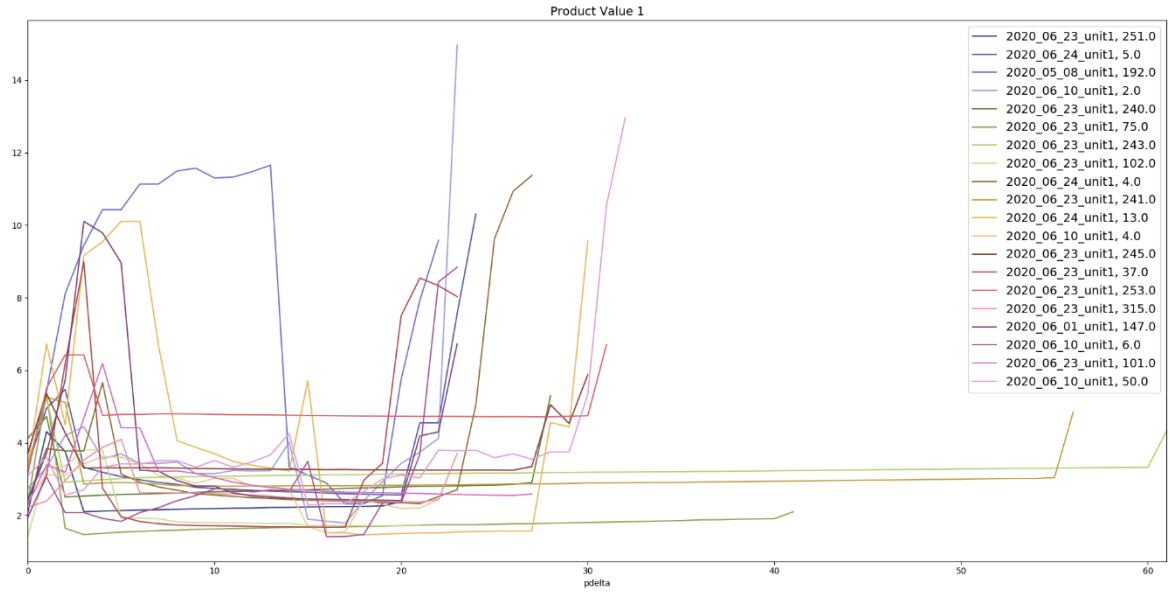

Matrix Value 4
2020_06_23_unit1, 16.0
2020_05_08_unit1, 192.0
2020_06_26_unit2, 211.0
2020_06_23_unit1, 251.0
2020_06_10_unit1, 50.0
2020_04_29_unit1, 161.0
2020_06_26_unit2, 183.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 37.0
2020_06_16_unit1, 115.0
2020_06_23_unit1, 314.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 49.0
2020_06_23_unit1, 274.0
2020_06_23_unit1, 76.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 312.0
0
10
20
30
40
0
10
20
30
40
50
60
70
80
delta

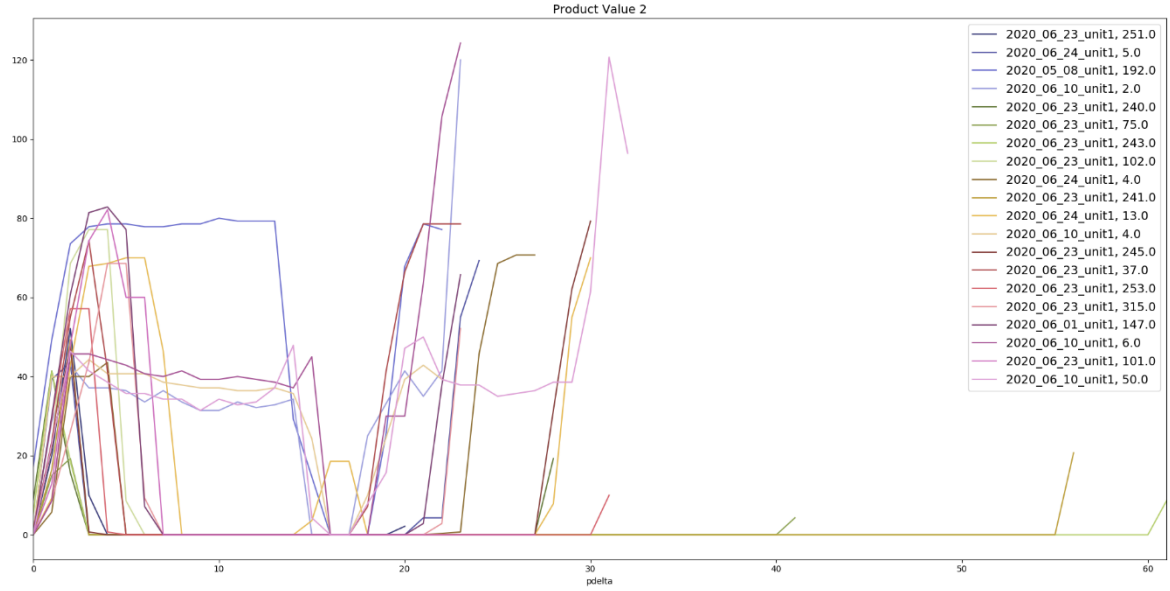

Matrix Value 5
2020_06_23_unit1, 16.0
2020_05_08_unit1, 192.0
2020_06_26_unit2, 211.0
2020_06_23_unit1, 251.0
2020_06_10_unit1, 50.0
2020_04_29_unit1, 161.0
2020_06_26_unit2, 183.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 37.0
2020_06_16_unit1, 115.0
2020_06_23_unit1, 314.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 49.0
2020_06_23_unit1, 274.0
2020_06_23_unit1, 76.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 312.0
delta

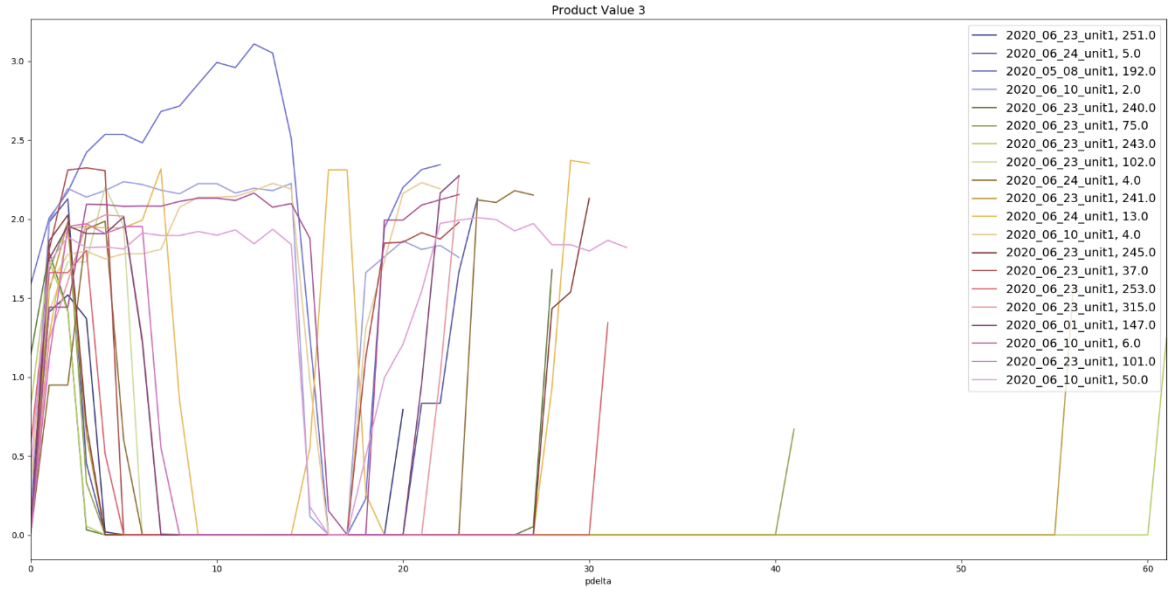

Matrix Value 6
2020_06_23_unit1, 16.0
2020_05_08_unit1, 192.0
2020_06_26_unit2, 211.0
2020_06_23_unit1, 251.0
2020_06_10_unit1, 50.0
2020_04_29_unit1, 161.0
2020_06_26_unit2, 183.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 37.0
2020_06_16_unit1, 115.0
2020_06_23_unit1, 314.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 49.0
2020_06_23_unit1, 274.0
2020_06_23_unit1, 76.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 312.0
delta

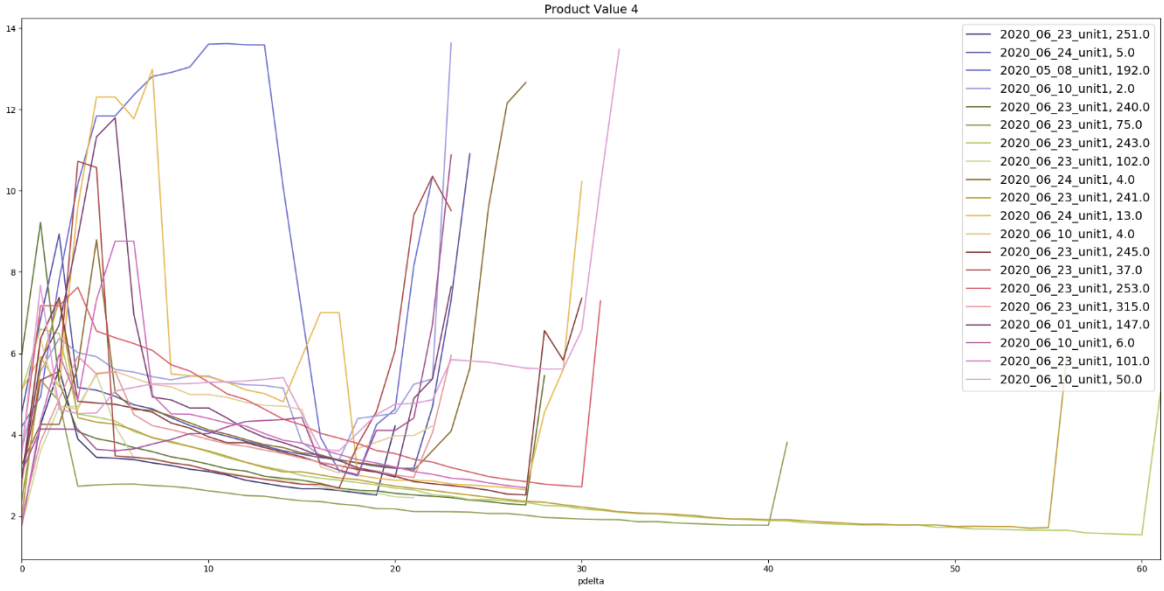


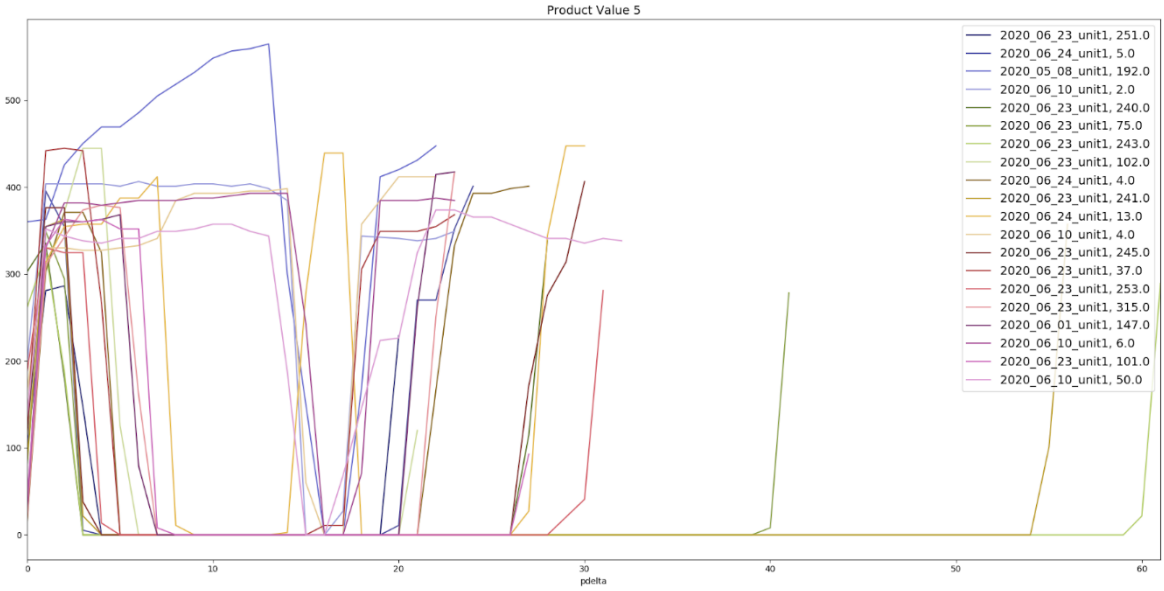
B Tuotelinjan 20 eniten poikkeavaa reikää standardoidulla datalla, prosessiarvot 1–5



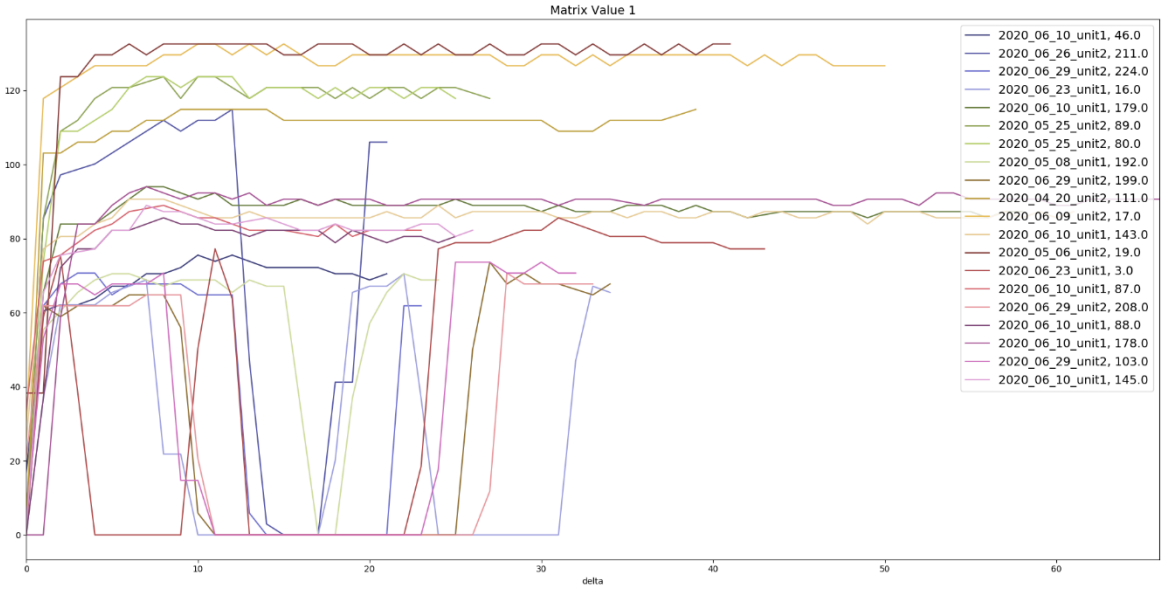

Product Value 2
2020_06_23_unit1, 251.0
2020_06_24_unit1, 5.0
2020_05_08_unit1, 192.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 243.0
2020_06_23_unit1, 102.0
2020_06_24_unit1, 4.0
2020_06_23_unit1, 241.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 4.0
2020_06_23_unit1, 245.0
2020_06_23_unit1, 37.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 315.0
2020_06_01_unit1, 147.0
2020_06_10_unit1, 6.0
2020_06_23_unit1, 101.0
2020_06_10_unit1, 50.0
0
20
40
60
80
100
120
0
10
20
30
40
50
60
pdelta

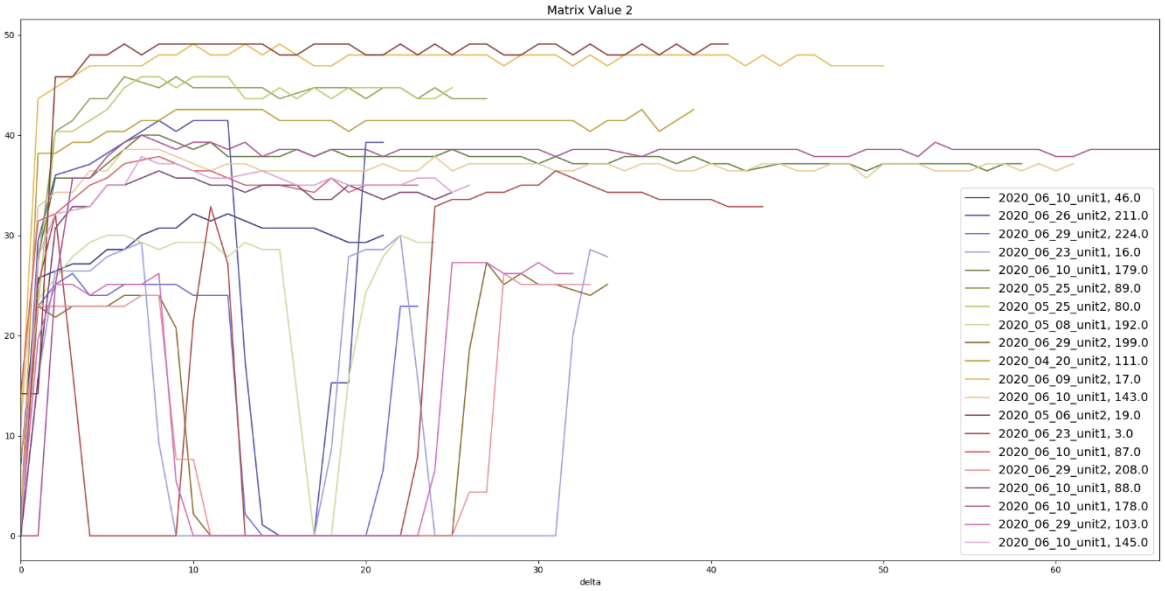

Product Value 3
2020_06_23_unit1, 251.0
2020_06_24_unit1, 5.0
2020_05_08_unit1, 192.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 243.0
2020_06_23_unit1, 102.0
2020_06_24_unit1, 4.0
2020_06_23_unit1, 241.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 4.0
2020_06_23_unit1, 245.0
2020_06_23_unit1, 37.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 315.0
2020_06_01_unit1, 147.0
2020_06_10_unit1, 6.0
2020_06_23_unit1, 101.0
2020_06_10_unit1, 50.0
0.0
0.5
1.0
1.5
2.0
2.5
3.0
0
10
20
30
40
50
60
pdelta

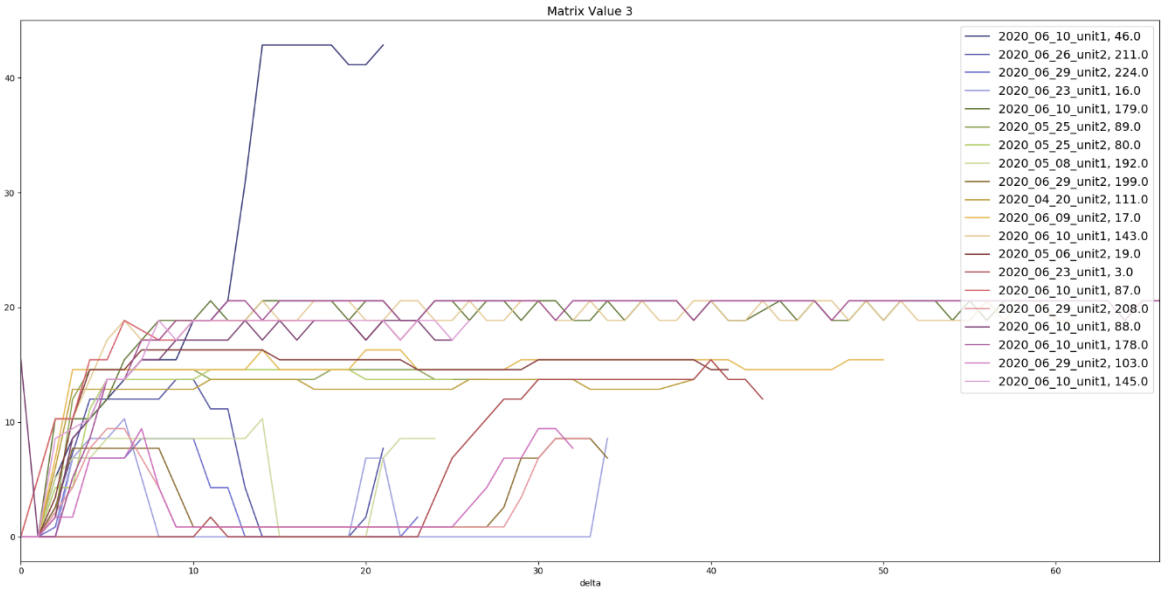

Product Value 4
2020_06_23_unit1, 251.0
2020_06_24_unit1, 5.0
2020_05_08_unit1, 192.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 243.0
2020_06_23_unit1, 102.0
2020_06_24_unit1, 4.0
2020_06_23_unit1, 241.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 4.0
2020_06_23_unit1, 245.0
2020_06_23_unit1, 37.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 315.0
2020_06_01_unit1, 147.0
2020_06_10_unit1, 6.0
2020_06_23_unit1, 101.0
2020_06_10_unit1, 50.0
pdelta

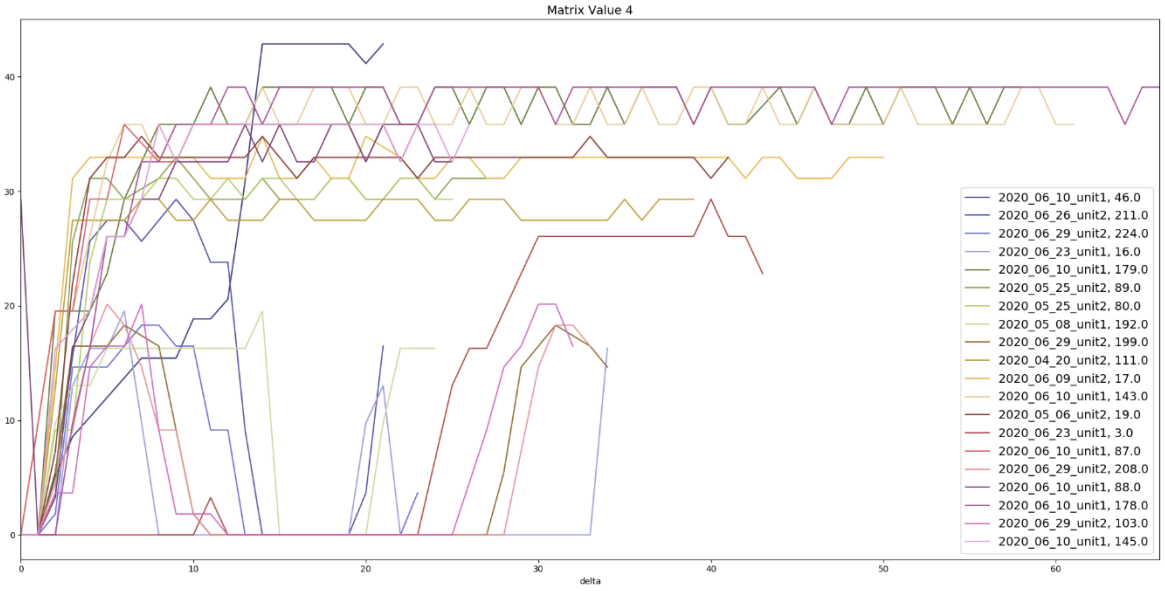

Product Value 5
2020_06_23_unit1, 251.0
2020_06_24_unit1, 5.0
2020_05_08_unit1, 192.0
2020_06_10_unit1, 2.0
2020_06_23_unit1, 240.0
2020_06_23_unit1, 75.0
2020_06_23_unit1, 243.0
2020_06_23_unit1, 102.0
2020_06_24_unit1, 4.0
2020_06_23_unit1, 241.0
2020_06_24_unit1, 13.0
2020_06_10_unit1, 4.0
2020_06_23_unit1, 245.0
2020_06_23_unit1, 37.0
2020_06_23_unit1, 253.0
2020_06_23_unit1, 315.0
2020_06_01_unit1, 147.0
2020_06_10_unit1, 6.0
2020_06_23_unit1, 101.0
2020_06_10_unit1, 50.0
pdelta

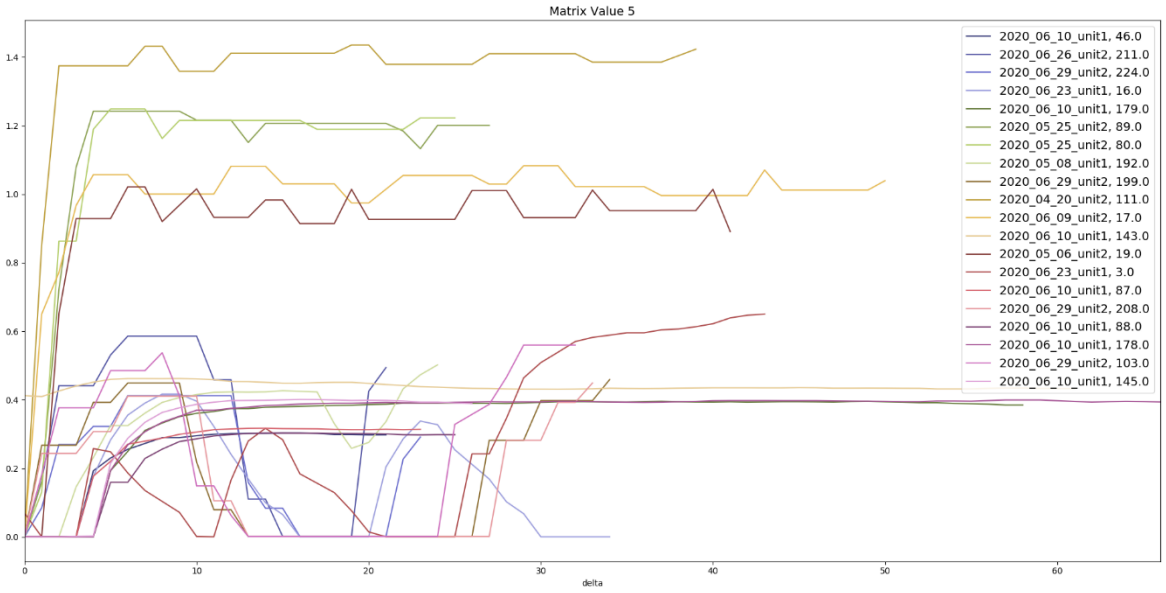
C Matriisilinjan 20 eniten poikkeavaa reikää raakadatalla, prosessiarvot 1–7

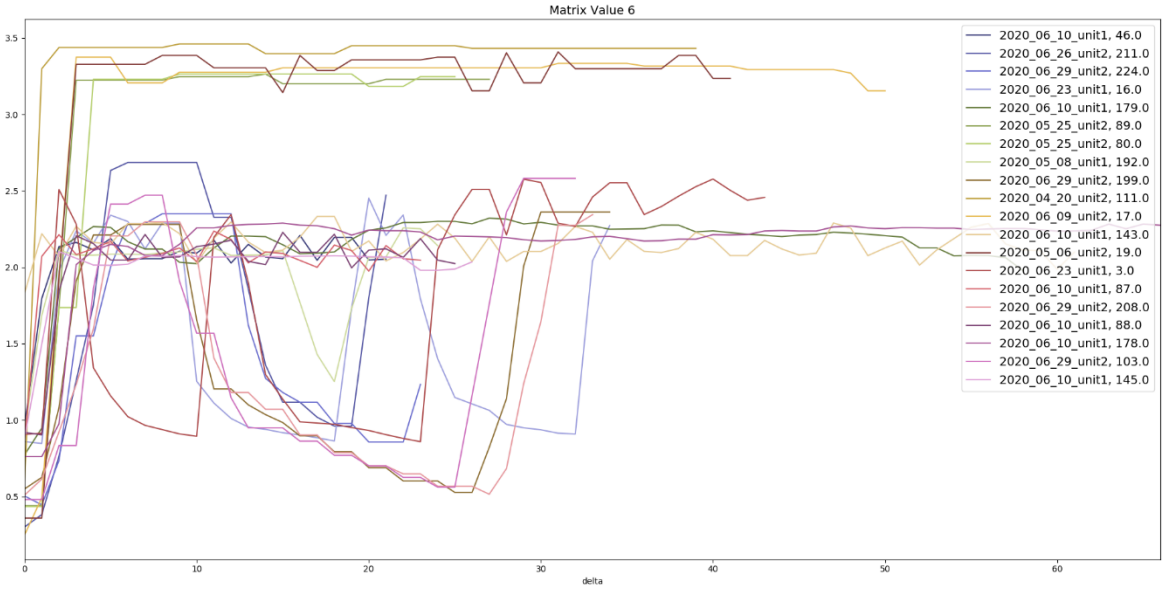


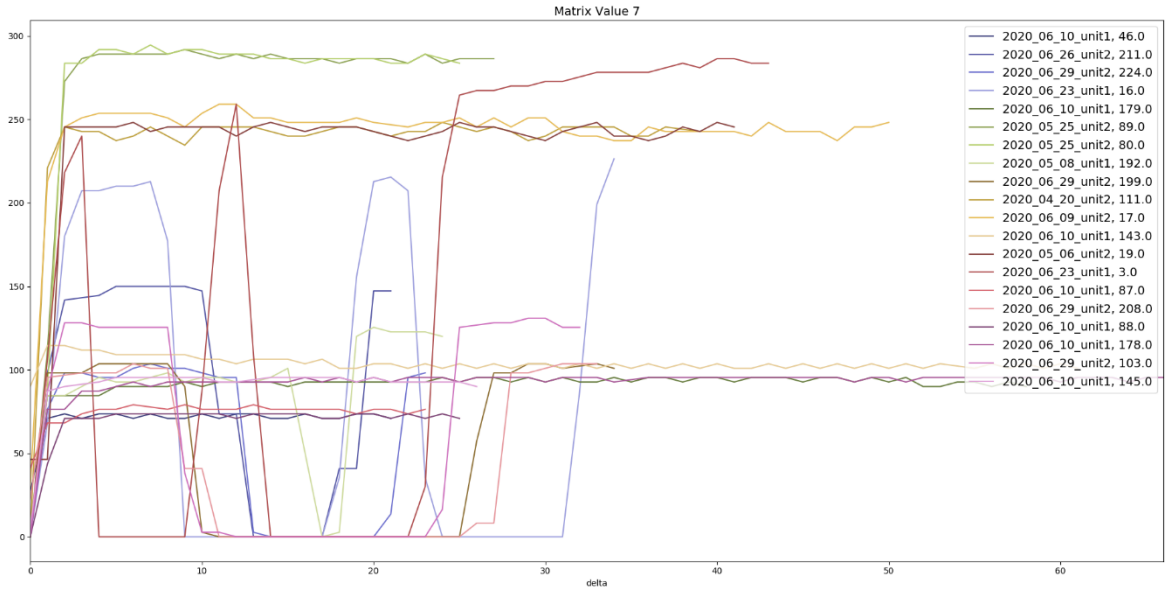



Matrix Value 3
2020_06_10_unit1, 46.0
2020_06_26_unit2, 211.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 16.0
2020_06_10_unit1, 179.0
2020_05_25_unit2, 89.0
2020_05_25_unit2, 80.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 199.0
2020_04_20_unit2, 111.0
2020_06_09_unit2, 17.0
2020_06_10_unit1, 143.0
2020_05_06_unit2, 19.0
2020_06_23_unit1, 3.0
2020_06_10_unit1, 87.0
2020_06_29_unit2, 208.0
2020_06_10_unit1, 88.0
2020_06_10_unit1, 178.0
2020_06_29_unit2, 103.0
2020_06_10_unit1, 145.0
0
10
20
30
40
0
10
20
30
40
50
60
delta

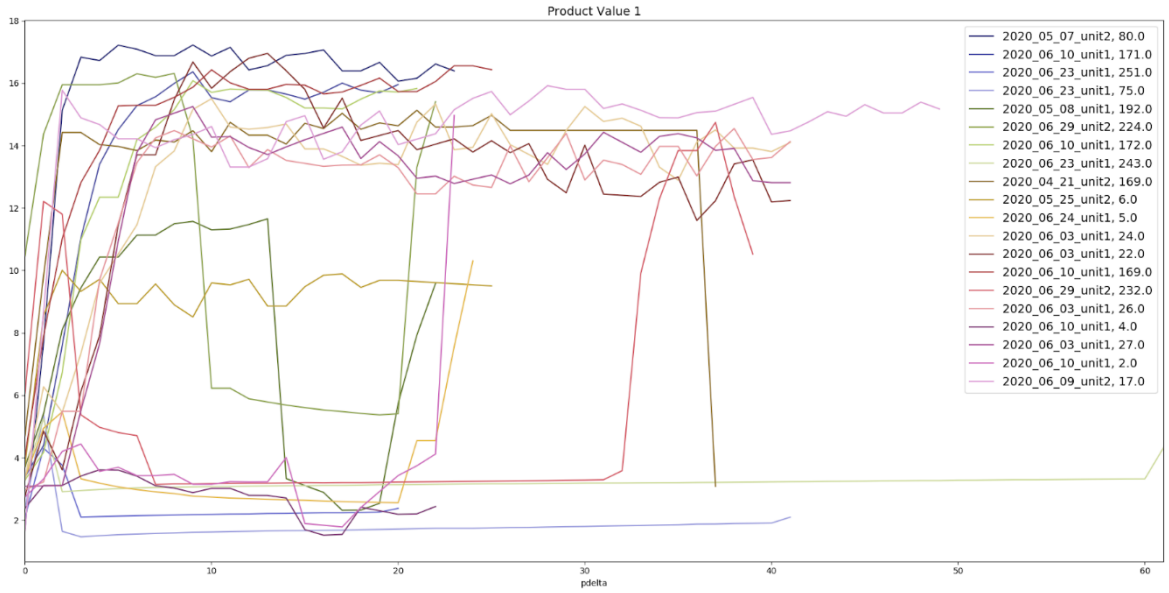

Matrix Value 4
2020_06_10_unit1, 46.0
2020_06_26_unit2, 211.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 16.0
2020_06_10_unit1, 179.0
2020_05_25_unit2, 89.0
2020_05_25_unit2, 80.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 199.0
2020_04_20_unit2, 111.0
2020_06_09_unit2, 17.0
2020_06_10_unit1, 143.0
2020_05_06_unit2, 19.0
2020_06_23_unit1, 3.0
2020_06_10_unit1, 87.0
2020_06_29_unit2, 208.0
2020_06_10_unit1, 88.0
2020_06_10_unit1, 178.0
2020_06_29_unit2, 103.0
2020_06_10_unit1, 145.0
0
10
20
30
40
0
10
20
30
40
50
60
delta

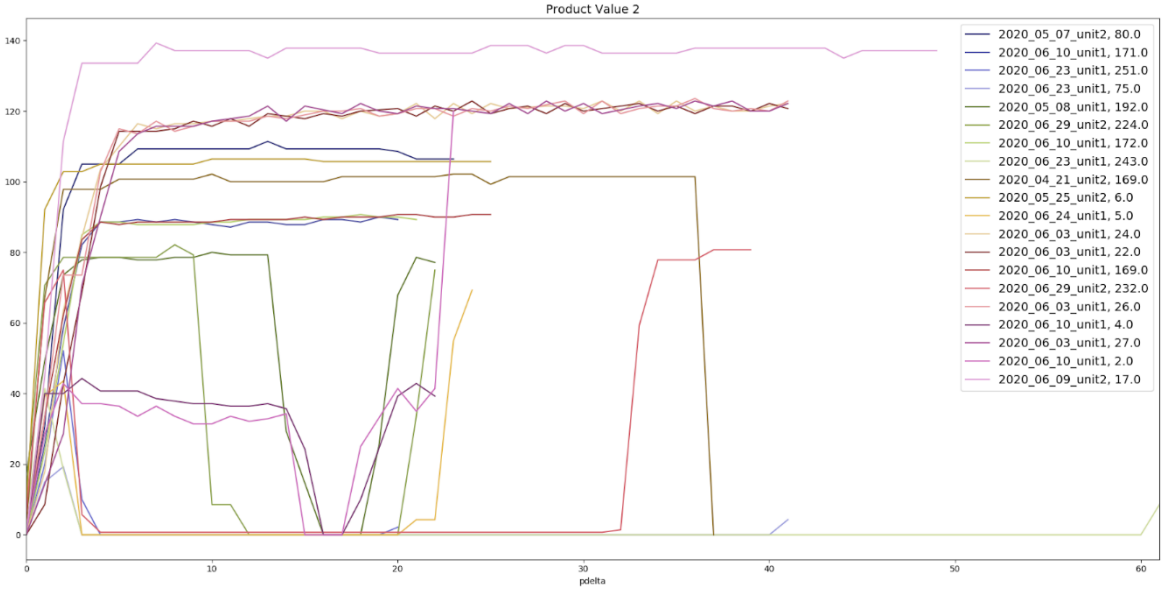

Matrix Value 5
2020_06_10_unit1, 46.0
2020_06_26_unit2, 211.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 16.0
2020_06_10_unit1, 179.0
2020_05_25_unit2, 89.0
2020_05_25_unit2, 80.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 199.0
2020_04_20_unit2, 111.0
2020_06_09_unit2, 17.0
2020_06_10_unit1, 143.0
2020_05_06_unit2, 19.0
2020_06_23_unit1, 3.0
2020_06_10_unit1, 87.0
2020_06_29_unit2, 208.0
2020_06_10_unit1, 88.0
2020_06_10_unit1, 178.0
2020_06_29_unit2, 103.0
2020_06_10_unit1, 145.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
0
10
20
30
40
50
60
delta

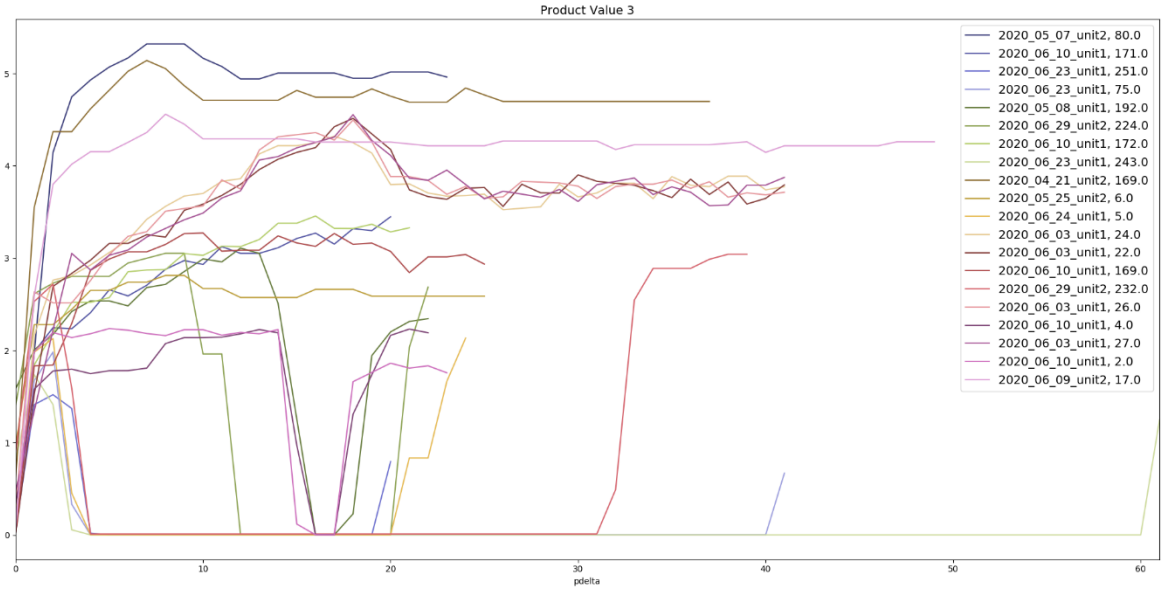

Matrix Value 6
2020_06_10_unit1, 46.0
2020_06_26_unit2, 211.0
2020_06_29_unit2, 224.0
2020_06_23_unit1, 16.0
2020_06_10_unit1, 179.0
2020_05_25_unit2, 89.0
2020_05_25_unit2, 80.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 199.0
2020_04_20_unit2, 111.0
2020_06_09_unit2, 17.0
2020_06_10_unit1, 143.0
2020_05_06_unit2, 19.0
2020_06_23_unit1, 3.0
2020_06_10_unit1, 87.0
2020_06_29_unit2, 208.0
2020_06_10_unit1, 88.0
2020_06_10_unit1, 178.0
2020_06_29_unit2, 103.0
2020_06_10_unit1, 145.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
0
10
20
30
40
50
60
delta

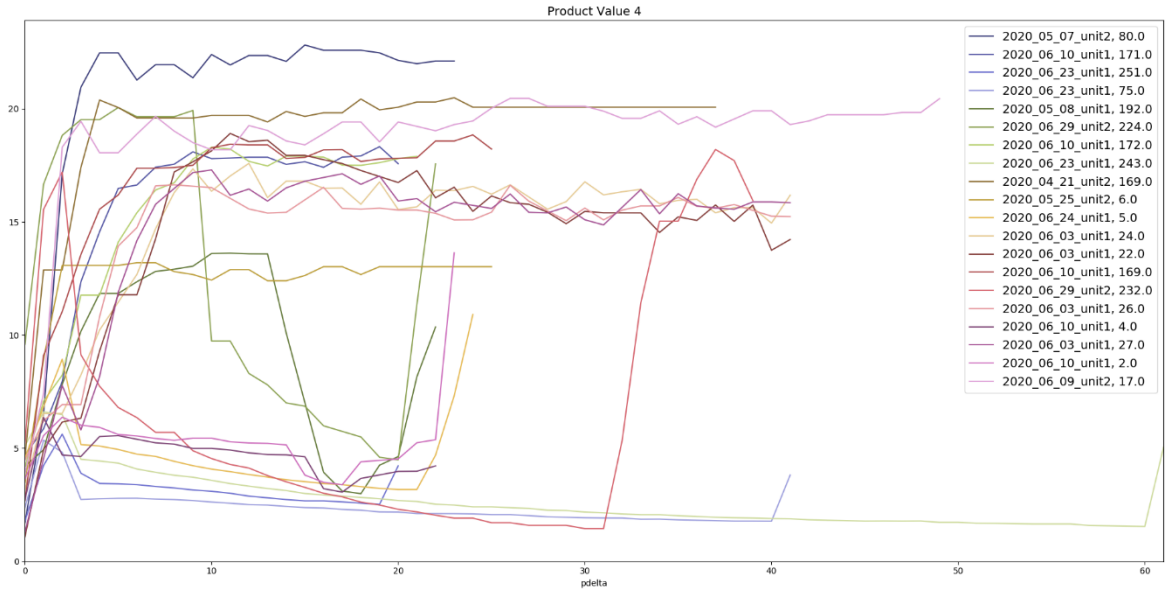


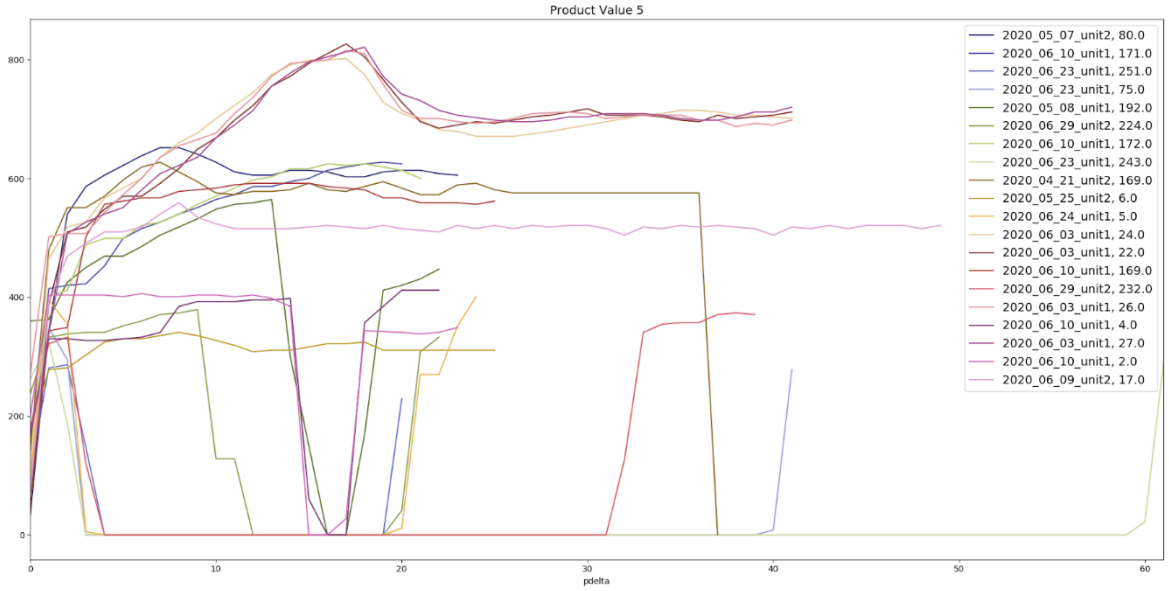
D Tuotelinjan 20 eniten poikkeavaa reikää raakadatalla, prosessiarvot 1–5




Product Value 2
2020_05_07_unit2, 80.0
2020_06_10_unit1, 171.0
2020_06_23_unit1, 251.0
2020_06_23_unit1, 75.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 224.0
2020_06_10_unit1, 172.0
2020_06_23_unit1, 243.0
2020_04_21_unit2, 169.0
2020_05_25_unit2, 6.0
2020_06_24_unit1, 5.0
2020_06_03_unit1, 24.0
2020_06_03_unit1, 22.0
2020_06_10_unit1, 169.0
2020_06_29_unit2, 232.0
2020_06_03_unit1, 26.0
2020_06_10_unit1, 4.0
2020_06_03_unit1, 27.0
2020_06_10_unit1, 2.0
2020_06_09_unit2, 17.0
pdelta


Product Value 3
2020_05_07_unit2, 80.0
2020_06_10_unit1, 171.0
2020_06_23_unit1, 251.0
2020_06_23_unit1, 75.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 224.0
2020_06_10_unit1, 172.0
2020_06_23_unit1, 243.0
2020_04_21_unit2, 169.0
2020_05_25_unit2, 6.0
2020_06_24_unit1, 5.0
2020_06_03_unit1, 24.0
2020_06_03_unit1, 22.0
2020_06_10_unit1, 169.0
2020_06_29_unit2, 232.0
2020_06_03_unit1, 26.0
2020_06_10_unit1, 4.0
2020_06_03_unit1, 27.0
2020_06_10_unit1, 2.0
2020_06_09_unit2, 17.0
pdelta


Product Value 4
2020_05_07_unit2, 80.0
2020_06_10_unit1, 171.0
2020_06_23_unit1, 251.0
2020_06_23_unit1, 75.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 224.0
2020_06_10_unit1, 172.0
2020_06_23_unit1, 243.0
2020_04_21_unit2, 169.0
2020_05_25_unit2, 6.0
2020_06_24_unit1, 5.0
2020_06_03_unit1, 24.0
2020_06_03_unit1, 22.0
2020_06_10_unit1, 169.0
2020_06_29_unit2, 232.0
2020_06_03_unit1, 26.0
2020_06_10_unit1, 4.0
2020_06_03_unit1, 27.0
2020_06_10_unit1, 2.0
2020_06_09_unit2, 17.0
0
5
10
15
20
0
10
20
30
40
50
60
pdelta


Product Value 5
2020_05_07_unit2, 80.0
2020_06_10_unit1, 171.0
2020_06_23_unit1, 251.0
2020_06_23_unit1, 75.0
2020_05_08_unit1, 192.0
2020_06_29_unit2, 224.0
2020_06_10_unit1, 172.0
2020_06_23_unit1, 243.0
2020_04_21_unit2, 169.0
2020_05_25_unit2, 6.0
2020_06_24_unit1, 5.0
2020_06_03_unit1, 24.0
2020_06_03_unit1, 22.0
2020_06_10_unit1, 169.0
2020_06_29_unit2, 232.0
2020_06_03_unit1, 26.0
2020_06_10_unit1, 4.0
2020_06_03_unit1, 27.0
2020_06_10_unit1, 2.0
2020_06_09_unit2, 17.0
0
200
400
600
800
0
10
20
30
40
50
60
pdelta

E Eri menetelmillä löytyneiden poikkeavuuksien vertailu

Yksikkö, kenttä, reikä	Standardoitu matriisilinjan poikkeavuuspisteytys	Standardoitu tuotelinjan poikkeavuuspisteytys	Raaka matriisilinjan poikkeavuuspisteytys	Raaka tuotelinjan poikkeavuuspisteytys	Matriisilinjan klusterointi	Tuotelinjan klusterointi	Asetusarvojen klusterointi
2, 20.4.2020, 111			x				
2, 21.4.2020, 169				x			
1, 29.4.2020, 161	x						
2, 6.5.2020, 19			x				
2, 7.5.2020, 80				x			
1, 8.5.2020, 192	x	x	x	x			
2, 25.5.2020, 6				x			
2, 25.5.2020, 80			x				
2, 25.5.2020, 89			x				
1, 1.6.2020, 147		x					
1, 3.6.2020, 22				x		x	
1, 3.6.2020, 24				x		x	
1, 3.6.2020, 26				x		x	
1, 3.6.2020, 27				x		x	
2, 9.6.2020, 17			x	x			
1, 10.6.2020, 2	x	x		x			

Yksikkö, kenttä, reikä	Standardoitu matriisilinjan poikkeavuuspisteytys	Standardoitu tuotelinjan poikkeavuuspisteytys	Raaka matriisilinjan poikkeavuuspisteytys	Raaka tuotelinjan poikkeavuuspisteytys	Matriisilinjan klusterointi	Tuotelinjan klusterointi	Asetusarvojen klusterointi
1, 10.6.2020, 4		x		x			
1, 10.6.2020, 6		x					
1, 10.6.2020, 46			x				
1, 10.6.2020, 49	x						
1, 10.6.2020, 50	x	x					
1, 10.6.2020, 88			x				
1, 10.6.2020, 143			x				
1, 10.6.2020, 145			x				
1, 10.6.2020, 169				x			
1, 10.6.2020, 171				x			
1, 10.6.2020, 172				x			
1, 10.6.2020, 178			x				
1, 10.6.2020, 179			x				
1, 10.6.2020, 208			x				
1, 16.6.2020, 115	x						
1, 23.6.2020, 3			x				
1, 23.6.2020, 16	x		x				

Yksikkö, kenttä, reikä	Standardoitu matriisilinjan poikkeavuuspisteytys	Standardoitu tuotelinjan poikkeavuuspisteytys	Raaka matriisilinjan poikkeavuuspisteytys	Raaka tuotelinjan poikkeavuuspisteytys	Matriisilinjan klusterointi	Tuotelinjan klusterointi	Asetusarvojen klusterointi
1, 23.6.2020, 37	x	x					
1, 23.6.2020, 75	x	x		x			
1, 23.6.2020, 76	x						
1, 23.6.2020, 101		x					
1, 23.6.2020, 102		x					
1, 23.6.2020, 240	x	x					
1, 23.6.2020, 241		x					
1, 23.6.2020, 243		x		x			
1, 23.6.2020, 245		x					
1, 23.6.2020, 251	x	x		x			
1, 23.6.2020, 253	x	x					
1, 23.6.2020, 274	x						
1, 23.6.2020, 312	x						
1, 23.6.2020, 314	x						
1, 23.6.2020, 315		x					
1, 24.6.2020, 4		x					
1, 24.6.2020, 5		x		x			

Yksikkö, kenttä, reikä	Standardoitu matriisilinjan poikkeavuuspisteytys	Standardoitu tuotelinjan poikkeavuuspisteytys	Raaka matriisilinjan poikkeavuuspisteytys	Raaka tuotelinjan poikkeavuuspisteytys	Matriisilinjan klusterointi	Tuotelinjan klusterointi	Asetusarvojen klusterointi
1, 24.6.2020, 13	x	x					
2, 26.6.2020, 183	x				x		
2, 26.6.2020, 211	x		x		x		
2, 29.6.2020, 103			x		x	x	x
2, 29.6.2020, 199			x		x	x	x
2, 29.6.2020, 224	x		x	x	x	x	x
2, 29.6.2020, 232	x			x		x	x