LAPPEENRANTA-LAHTI UNIVERSITY OF TECHNOLOGY
LUT School of Engineering Science
Chemical Engineering
Chemical and Process Engineering

Tatu Toroskainen

ÄLYKÄS DATAN KÄYTTÖ TEOLLISUUDEN KUNNOSSAPIDON JA KÄYTTÖVARMUUDEN PARANTAMISESSA

Työn tarkastajat: Professori, TkT Tuomas Koiranen
Jari Simola
Ohjaajat: Simo Klemettilä
Anssi Anttila

Tiivistelmä

Lappeenrannan-Lahden teknillinen yliopisto

School of Engineering Science

Master's Programme in Chemical and Process Engineering

Diplomityö

2019

Tatu Toroskainen

Älykäs datan käyttö teollisuuden kunnossapidon ja käyttövarmuuden parantamisessa

167 sivua, 67 kuvaa, 31 taulukkoa, 4 liitettä

Tarkastajat ja ohjaajat: TkT Tuomas Koiranen (LUT), Simo Klemettilä, Jari Simola ja Anssi Anttila

Hakusanat: Big data, Internet of Things, IoT, data-analyysi, kunnossapito, juurisyyanalyysi, poikkeuksien havaitseminen

Paperitehtailla ja teollisuudessa yleisesti kerätään valtava määrä prosessien tuotanto- ja kunnossapitodataa. Tämän datan analysointi on perinteisin menetelmin sekä muuttujien- että datapisteiden määrän vuoksi erittäin hankalaa, ellei jopa mahdotonta. Eri yritykset ovat luoneet ratkaisuja tämän ns. Big datan käsittelyyn. Dataa voidaan käsitellä sekä tilastollisilla että oppivilla menetelmillä. Ratkaisut vaativat suurien datamäärien kanssa usein hajautetun järjestelmän.

Tässä diplomityössä tutkitaan data-analytiikan soveltamista teollisuuden tuotannon ja kunnossapidon ongelmien ratkaisemiseen. Kirjallisuusselvitys kattaa data-analytiikkamenetelmiä, -käyttötapauksia ja -ohjelmistoja sekä kunnossapidon ja prosessi-automaation suppeat esittelyt.

Käytännön osassa vertaillaan eri ohjelmistoja koekäytön perusteella. Vertailussa on mukana neljä eri ohjelmistoa. Parhaaksi ohjelmistoksi valittiin pisteytyksien perusteella KNIME, joka on työnkulkuihin perustuva analytiikkaohjelmisto. KNIMEn ja visualisointityökaluna toimineen TIBCO Spotfiren avulla löydettiin juurisyy teollisen prosessin tuotannon ongelmalle. KNIMEllä testattiin lisäksi erilaisia keinoja havaita poikkeuksia vikaantuneen prosessipumpun historia-värähtelydatasta. Älykkäillä menetelmillä pumpun poikkeava toiminta olisi ollut mahdollista havaita jo noin viikkoa ennen todellista ongelmatilannetta.

Abstract

Lappeenranta-Lahti University of Technology

School of Engineering Science

Master's Programme in Chemical and Process Engineering

Master's Thesis

2019

Tatu Toroskainen

Intelligent use of data for improvement of industrial maintenance and reliability

167 pages, 67 figures, 31 tables, 4 appendix

Examiners and supervisors: D.Sc. (Tech.) Tuomas Koiranen (LUT), Simo Klemettilä, Jari Simola and Anssi Anttila

Keywords: Big data, Internet of Things, IoT, data analysis, maintenance, root cause analysis, anomaly detection

Huge amount of process and maintenance data is collected in paper mills and other industrial facilities. Analyzing this data by traditional means is difficult due to its volume and dimensionality. Companies have created different solutions to handling this so called "Big Data". Due to the volume of the data, distributed systems are often necessary when handling it.

This thesis explores data analysis in the context of solving industrial production and maintenance problems. The literature study covers several data analysis methods, use cases and software, as well as concise introductions to process automation and maintenance.

In the practical part, four different analytics software were compared and scored with different criteria. A workflow based software called KNIME was selected as the best option based on the scoring. With KNIME for analysis and TIBCO Spotfire for visualization, the root cause of an industrial production problem was found. In addition, different anomaly detection methods were used to detect problems from a faulty process pump's historical vibration data. Intelligent methods could have detected anomalous behavior of the pump approximately a week before the actual problem in the pump was detected visually.

Alkusanat

Tämä diplomityö tehtiin Empower IN Oy:lle kevään ja kesän 2019 aikana. Työn kirjallisuusosa tehtiin Lappeenrannassa UPM Kaukaan tehdasalueella ja käytännön osa Inkeroisissa Stora Enson Anjalankosken paperitehtaalla. Haluan kiittää Empowerin henkilöstöä mukavasta ja kannustavasta työilmapiiristä molemmilla paikkakunnilla. Erityiset kiitokset ansaitsevat työni ohjaajat Simo Klemettilä, Jari Simola ja Anssi Anttila, sekä vastaava professori Tuomas Koiranen.

Lisäksi haluan kiittää kotijoukkojani ja ystäviäni heidän antamastaan tuesta koko opiskelujeni ajalta.

Inkeroisissa 5.7.2019

Tatu Toroskainen

Sisällysluettelo

Tiivistelmä 2
Abstract 3
Alkusanat 4
Sisällysluettelo 5
Symboli- ja lyhenneluettelo 8
KIRJALLISUUSOSA 12
1. Johdanto 12
1.1 Työn tarkoitus ja rajaus 13
2. Big data ja IoT 13
2.1 Big Datan käytön rajoitukset 15
2.2 Käyttötapaukset älykkäästä datan käytöstä kirjallisuudessa 16
2.2.1 Case MeadWestvaco Covington (Fu & Hart, 2016) 16
2.2.2 Case UPM Kaukas (Nykyri, 2018) 17
2.2.3 Case Valukomponenttitehdas (Lee, et al., 2017) 18
2.2.4 Case Poikkeuksien havaitseminen roottorin toiminnassa (Silipo, et al., 2018) 20
3. Data-analyysi ja tiedon louhinta 21
3.1 Koneoppiminen 22
3.2 Korrelaatio 24
3.2.1 Kovarianssi 24
3.2.2 χ^2-testi 29
3.2.3 Varianssianalyysi (ANOVA) ja F-stat 33
3.2.4 Levenen testi 36
3.3 Datan luokittelu 38
3.3.1 Regressio 41
3.3.2 Päätöspuu 42
3.3.3 Support Vector Machine 47
3.3.4 Naiivi Bayes 48
3.3.5 Neuroverkko (Neural Network) 51
3.4 Klusterointi 54
3.4.1 K-Means 55
3.5 Poikkeuksien havaitseminen 56
3.6 Yhteenveto data-analyysin eri menetelmistä 57
3.7 Datan esikäsittely 58
4. Kaupalliset datankäsittelyohjelmistot ja teknologiat 59

4.1 Microsoft Azure Machine Learning Studio 59
4.2 GE Intelligent Platforms 61
4.3 TIBCO Spotfire 62
4.4 Apache Hadoop 63
4.5 Apache Spark 65
4.6 IBM Watson 67
4.7 Umetrics Easy Analytics 68
4.8 KNIME 69
4.9 Ohjelmiston perusedellytykset ja harkinnan kohteet 71
4.10 Testi- ja tutkimusdata ohjelmistoille 72
5. Tyypillinen prosessiautomaatio ja kunnossapito 74
5.1 Kunnossapiton merkitys ja lajit 74
5.2 Prosessiautomaatio ja anturointi 77
KÄYTÄNNÖN OSA 79
6. Ohjelmistojen testaaminen: kriteerit ja metodiikka 79
6.1 Yleinen helppokäyttöisyys ja saatavilla oleva tuki 82
6.2 Datan esikäsittely ja tiedostojen luku 84
6.3 Menetelmien määrä 88
6.4 Visualisointi- ja raportointiominaisuudet 91
6.5 Menetelmien lisäysmahdollisuudet ja helppous 93
6.6 Hinta 94
6.7 Yhteenveto ohjelmistoista 95
7. Dataan pohjautuva juurisyyanalyysi teollisuustapauksessa 96
7.1 Esikäsittely 100
7.2 Tilastolliset analyysit 101
7.2.1 Ominaisuuksien korrelaatio ajotilanteen kanssa ja keskenään 103
7.2.2 Ominaisuuksien varianssianalyysi Ajotilanteen suhteen 105
7.2.3 χ^2-testi Ajotilanteen ja ominaisuuksien välillä 110
7.2.4 Yhteenveto tilastollisista analyyseista 116
7.3 Ongelmatilanteiden luokittelu 116
7.3.1 Testijärjestelyt 118
7.3.2 Logistinen regressio 121
7.3.3 Päätöspuu 123
7.3.4 Random Forest 124
7.3.5 Naiivi Bayes 126
7.3.6 Neuroverkko 127
7.3.7 Yhteenveto ongelmatilanteiden luokitteluista 128

8. Värähtelytason poikkeuksien havainnointi pumpussa 130
8.1 Autokorrelaatioon perustuva värähtelyn tarkkailu 132
8.1.1 Autokorrelaation soveltaminen pumpun värähtelyn tarkkailussa 135
8.2 (Auto)regressioon perustuva värähtelyn tarkkailu 141
8.2.1 Autoregression soveltaminen pumpun värähtelyn tarkkailussa 144
8.3 PCA-rekonstruktioon perustuva värähtelyn tarkkailu 150
8.3.1 PCA-rekonstruktion soveltaminen pumpun värähtelyn tarkkailussa 152
8.4 Yhteenveto värähtelyn tarkkailusta pumpussa 157
9. Johtopäätökset 158
10. Kirjallisuus 160
11. Liitteet 167

Symboli- ja lyhenneluettelo

Symbolit

a_r	Kulmakerroin ominaisuudelle x_r
b	Vakio tai/eli taipumus
C	Kendallin korrelaation yhdenmukaisten parien määrä
c_h	Hyödyntämättömän elinajan kustannukset
c_{oh}	Laitteen hajoamisen kustannukset
C_{tot}	Kunnossapidon kustannukset laitteelle
C_{yx}	Korrelaatiokerroin ominaisuuksien x ja y välillä
D	Kendallin korrelaation ristiriitaisten parien määrä
E	Ennustettu arvo
e_{ij}	Odotettu arvo solulle (i, j) χ^2 -testissä
F	ANOVAn laskettu suure
f	Ominaisuuksien määrä
f_a	Aktivaatiofunktio
f_B	Ryhmien välinen vapausaste (ANOVA)
F_{Chi}	χ^2 -testin vapausaste
f_{oh}	Laitteen vikataajuus
f_W	Ryhmien sisäinen vapausaste (ANOVA)
G	Gini Index
h	Eri tietotyyppien määrä päätöspuun haarassa
k	Luokkien lukumäärä
k_h	Haarojen määrä

N	Havaintojen kokonaismäärä
n	Havaintojen lukumäärä
N_j	Otoksen (eli luokan) j havaintojen määrä
n_o	Otosten lukumäärä
n_{tot}	Havaintojen kokonaismäärä
o_i	Lokeron i havainnot yhteensä
o_{ij}	Havaintojen määrä solussa (i, j)
o_j	Lokeron j havainnot yhteensä
$P(A)$	Todennäköisyys, että A on tapahtunut
$P(A\|B)$	Todennäköisyys, että A on tapahtunut, kun B:n tiedetään tapahtuneen
p_i	Tiedon i osuus eli todennäköisyys datassa
S	Entropia välillä [0, 1]
$SSBG$	Ryhmien välisten neliöiden summa (sum of squares between groups)
$SSWG$	Ryhmien sisäisten neliöiden summa (sum of squares within groups)
s_x	Muuttujan x mittaustulosten keskihajonta
S_{xy}	Pearsonin korrelaation arvo ominaisuuksien x ja y välillä
s_y	Muuttujan y mittaustulosten keskihajonta
T	Todellinen arvo
t_h	Laitteen huollettavan tai vaihdettavan komponentin hyödyntämätön elinaika
W	Levenen testin tarkkailtava arvo
w_n	Painoarvo muuttujalle x_n
x	Muuttujan x arvo
$\bar{x}$	Muuttujan x keskiarvo
x_{ij}	Muuttujan x arvo indeksillä i luokassa j

$\bar{y}$	Muuttujan y keskiarvo
y_{calc}	Mallinnettavan muuttujan ennustettu arvo regressiossa
y_l	Muuttujan y arvo havainnossa *l*
z	aktivaatiofunktion syöte
$Z_{..}$	Z_{ji}:n kokonaiskeskiarvo
$Z_{j.}$	Z_{ji}:n keskiarvo luokassa j
Z_{ji}	Mittauksen j,i hajonnan itseisarvo luokassa, eli $\lvert x_{j,i} - \bar{x}_{j.} \rvert$
$\bar{x}_{j.}$	Muuttujan x keskiarvo luokassa j
$\bar{x}_{tot}$	Kaikkien havaintojen x keskiarvo
$\hat{y}$	Aktivaatiofunktion *f* arvo syötteellä *z*
τ	Kendallin korrelaation arvo

Lyhenteet

ANOVA	Analysis of Variance
AUC	Area Under Curve
CRAN	Comprehensive R Archive Network
FN	False Negative
FP	False Positive
HDFS	Hadoop Distributed File System
IIoT	Industrial Internet of Things
IoT	Internet of Things
kNN	k-Nearest-neighbor, eli lähimpien naapureiden määrä SMOTEssa
ML	Machine Learning
MLP	Multi-Layer Perceptron

MS	Microsoft
PCA	Principal Component Analysis
RAM	Random-Access Memory
RFID	Radio-Frequency Identification
ROC	Receiver Operating Characteristic (curve)
SMOTE	Synthetic Minority Oversampling Technique
SSBG	Sum of Squares Between Groups
SSWG	Sum of Squares Within Groups
SVM	Support Vector Machine
TERR	TIBCO Enterprise Runtime for R
TN	True Negative
TP	True Positive
Weka	Waikato Environment for Knowledge Analysis

KIRJALLISUUSOSA

1. Johdanto

Teollisuudessa kerätään valtava määrä tuotanto- ja kunnossapitodataa. Tämän datan hyödyntäminen on suuren määränsä vuoksi ollut ennen viime vuosia hankalaa. Nykyaikaisilla hajautetuilla järjestelmillä ja pilvilaskennalla on kuitenkin mahdollista käsitellä suuria datamääriä, ja oppivat järjestelmät kykenevät tekemään johtopäätöksiä datan perusteella. Ehkäisevä kunnossapito on yksi sovelluskohde, jonka parantamiseen ns. Big dataa voidaan hyödyntää. (Wan, et al., 2017)

Big data tarkoittaa eri lähteistä saatavia datajoukkoja, joita on määränsä ja tulonopeutensa vuoksi vaikeaa tai mahdotonta käsitellä perinteisin keinoin (Furht & Villanustre, 2016). Analysoimalla datamassoja moderneilla ohjelmistoilla on mahdollista löytää riippuvuuksia ja muuta arvokasta informaatiota tiedosta, jota ei aiemmin ole voitu käsitellä. Big dataa on alettu viimevuosina hyödyntämään eri sovelluksissa kielien analysoinnista ja automaattisesta kääntämisestä (Ryżko, et al., 2016) paperitehtaiden tuotannon laadun parantamiseen (Fu & Hart, 2016).

Big dataa hyödyntävien ohjelmistojen toiminnan edellytyksiä ovat datan määrän ja vaihtelevuuden vuoksi mm. korkea luotettavuus ja virhetoleranssi (Furht & Villanustre, 2016). Ohjelmiston valinta Big datan analysointiin tapahtuukin monen käyttökohteesta riippuvan tekijän perusteella.

Big datan käsittely tarjoaa mahdollisuuden parantaa tuotantoprosessien toimintavarmuutta erottamalla epäselvät häiriötilanteet normaalista ajotilanteesta sekä tehostamalla kunnossapitoa. Kunnossapitoa voidaan tehostaa mm. ennakoimalla tulevia ongelmia datan avulla ja aikatauluttamalla korjaustoimenpiteitä ennen ongelmien ilmenemistä. Ongelmien juurisyitä tarkkailemalla data voi myös antaa suuntaa tuotantoparametrien muutoksiin ja optimointiin. Sekä kunnossapidon tehostamisella että ongelmien juurisyiden löytämisellä voi olla huomattavia kustannusvaikutuksia. Perinteinen laitteiden kunnossapito kattaa jopa yli 60 % laitteistojen elinkaaren kustannuksista (O'Donovan, et al., 2015) ja ongelmien estäminen juurisyyn löytämisellä voi estää kymmeniä tuhansia euroja tunnilta maksavia tuotannon keskeytyksiä.

1.1 Työn tarkoitus ja rajaus

Tämän diplomityön kirjallisuusosassa käsitellään Big dataan liittyviä käyttötapauksia, analysoinnissa käytettäviä menetelmiä, käsittelyyn soveltuvia ohjelmistoja, sekä tyypillistä teollisuuden prosessiautomaatiota ja -kunnossapitoa. Käytännön osassa käsitellään teollisuustapauksessa syntyneen ongelman juurisyiden etsimistä ja ongelmien mallintamista, sekä Anjalankosken paperitehtaalta kerättyä kunnossapidon värähtelydataa kirjallisuusosassa valitun ohjelmiston avulla.

2. Big data ja IoT

”**Big data**” eroaa perinteisestä datasta monilla tavoin. Se on määrältään ja tulonopeudeltaan huomattavasti perinteistä dataa suurempaa, eikä se ole useimmiten jäsennelty yhtä tarkasti. Englanniksi tätä kuvaillaan usein kolmella V:llä: Volume, Velocity ja Variety. Data saadaan hajautetusti eri lähteistä, ja sitä on vaikea hallita ja tulkita perinteisin menetelmin. Perinteistä dataa säilytetään yleensä relaatiotietokannoissa, Big dataa hajautetusti eri tiedostoissa. Big dataa käsitellessä tietokoneen laskenta-aika on yleensä suurin pullonkaula. (Furht & Villanustre, 2016)

Teollisuuden Big dataa tulee työstää järjestelmällisesti. Työstö alkaa datan keräämisellä antureista eri lähteistä. Keräämisen jälkeen data varastoidaan keskitettyyn kohteeseen. Data käsitellään oikeean muotoon ja turha tieto poistetaan. Tämän jälkeen data linkitetään analysoivaan ohjelmistoon, joka tuottaa tietoa ja tilastoja, sekä oppii ja ennustaa datan perusteella määriteltyjä asioita. Tämä tieto muunnetaan ihmiselle ymmärrettävään muotoon esim. visualisoimalla. (Furht & Villanustre, 2016)

Big datan käsittelyyn liittyy monia teknologioita. Dataa voidaan käsitellä mm. rinnakkaisprosessoinnilla ja pilvitietojenkäsittelyllä. Tiedon louhintamenetelmät ovat monissa Big data sovelluksissa tarpeellisia, sillä kaikki saatavilla oleva tieto ei ole olennaista. Hajautetut tietokantaratkaisut ja laajennettavat tietovarastot sallivat datan tallentamisen. (Furht & Villanustre, 2016)

Big dataa voidaan käyttää tekstianalyysiin, äänianalyysiin, videoanalyysiin, sosiaalisen median analyysiin, juurisyyanalyysiin tai ennakoivaan analyysiin (Furht & Villanustre, 2016). Tuotannon ja käyttövarmuuden parantamisessa keskeisiä käyttökohteita ovat juurisyyanalyysi ja ennakoiva analyysi. Analyysien avulla voidaan selvittää vanhoja ongelmia ja ennustaa tulevaisuuden tapahtumia käyttämällä historia- tai online-dataa. Datan avulla kunnossapidolliset ja luotettavuuteen vaikuttavat ongelmat voidaan havaita ajoissa, mikä puolestaan parantaa tuotteen saatavuutta (Alzghoul & Löfstrand, 2011).

Big datan avulla voidaan harjoittaa sekä deskriptiivistä että prediktiivistä analytiikkaa. Deskriptiivisellä analytiikalla tarkoitetaan historiadatan käsittelyä mahdollisten piilevien kaavojen löytämiseksi. Prediktiivisellä analytiikalla tarkoitetaan tulevaisuuden ennustamista joko tapahtumien tai trendien osalta. (Pyne, et al., 2016)

IoT, eli Internet of Things, tarkoittaa teknistä arkkitehtuurimallia, jossa laitteet keräävät ja lähettävät dataa laajempaan verkkoon. Verkkoon tietoa keräävät laitteet voivat olla mitä tahansa. Anturit, älypuhelimet, tietokoneet ja kaikki muut verkkoon kytketyt laitteet ovat osa Internet of Thingsia. Termin alkuperä on johdettavissa Kevin Ashtoniin, yhteen RFID:n (Radio-frequency identification) kehittäjistä. Hän pohti inventaarion, tässä tapauksessa suositun huulipunavärin, hallintaa vähittäiskaupassa. Jos tuote kommunikoisi hyllyn (järjestelmän) kanssa, olisi Ashtonin tai jonkun muun mahdollista tietää, mitä hyllyssä on. Myöhemmässä yhteistyössä MIT:n kanssa Ashton käytti ideaan liittyen termiä Internet of Things. (Klein, 2017)

Teollisuuden puolella käytetään IoT:n lisäksi myös termiä IIoT eli Industrial Internet of Things. (Pyne, et al., 2016) IIoT on Big datan tuottaja teollisuuden prosesseissa. Se kattaa laitteistojen automaation, anturoinnin ja muut dataa tuottavat lähteet. IoT:n yleinen arkkitehtuuri voidaan jakaa kolmeen tasoon taulukon 1 mukaisesti (Nykyri, 2018).

Taulukko 1 IoT -arkkitehtuurin tasot.

Sovellustaso	Datan käsittelyyn liittyvät asiat, kuten analysointityökalut, visualisointi, pilvipalvelut ja hälytykset
Yhteystaso	Yhteyteen liittyvät asiat, kuten langattomat verkot ja tukiasemat
Havaintotaso	Dataa keräävät asiat ja esineet, kuten virta-, värähtely- ja lämpötila-anturit ja älylaitteet

Nykyään sekä IoT:n että Big datan määritelmät ovat hieman epäselviä, ja vaihtelevat henkilöstä toiseen. (Klein, 2017) Tässä työssä IoT:sta puhuttaessa tarkoitetaan taulukon 1 mukaista arkkitehtuuria, eli antureita ja muita laitteita, jotka ovat yhteydessä laajempaan verkkoon ja sovelluksiin.

2.1 Big Datan käytön rajoitukset

Big datan käytössä on monia ongelmia. Osaa käyttöön liittyvistä ongelmista kyetään lievittämään data-analyysin esikäsittelykeinolla tai muilla tavoilla. Osa ongelmista on perustavamman laatuisia. Nämä ongelmat on pakko hyväksyä.

Ongelmadatan pieni määrä suhteessa normaalin toiminnnan dataan on osassa tapauksista lievennettävissä oleva ongelma. Pienellä ongelmadatamäärällä on haastavaa kouluttaa tehokasta, yleistävää koneoppimisjärjestelmää. Ongelmaa voidaan lievittää tilastotieteen keinoin, muun muuassa SMOTE (Synthetic Minority Oversampling Technique) -menetelmällä, joka on esitelty myöhemmin lyhyesti luvussa 3.7. (Microsoft [2], 2015)

Onlinekäsittelyssä datan määrästä johtuen sen siirtäminen esimerkiksi pilvipalveluun ei ole aina mahdollista. Jos datan volyymi on riittävän suurta, on ainoa vaihtoehto paikallinen prosessointi tai esikäsittely. Datan siirtomäärän tulisi olla matala suhteessa laskentaan tarvittavaan aikaan. (Pyne, et al., 2016)

Datan tietoturvallisuus pilvipalveluissa on huomiota vaativa asia, joka voi muodostua ongelmaksi. Asiakkaiden- tai yrityksen omien tietojen vuotaminen voi olla erittäin haitallista yrityksen imagolle ja asiakassuhteille, minkä lisäksi tietoturvalainsäädännön vaatimukset on täytettävä. Käyttäjän päässä toimivat salausmenetelmät takaavat tiedon tietoturvan, mutta rajoittavat toisaalta esimerkiksi haku- ja prosessointitoimintoja. (Pyne, et al., 2016)

Kuten kaikissa muissakin jatkuvissa prosesseissa, Big datan onlinekäsittelyssä syntyy ajan myötä väistämättä ongelmia esimerkiksi laitteiden kaatumisena. Syntyvien ongelmien vuoksi käsittelevän ohjelmiston on pystyttävä sietämään virhetilanteita. (Pyne, et al., 2016)

Ongelmanlähteet, joista ei ole (ainakaan tarkkaa) dataa eivät luonnollisesti ole analysoitavissa. Esimerkiksi raaka-aineista aiheutuvat ongelmat tuotantoprosessin myöhemmissä osissa ovat tällaisia ongelmia, jos toimittajalla ei ole dataa.

Pilvilaskennassa ja hajautetussa laskennassa usean systeemissä kiinni olevan laitteen on kopioitava dataa. Kopiointi hidastaa itse laskentaprosessia, mutta on välttämätöntä virheiden sietokyvyn kasvattamisessa. Kopiointi voidaan tehdä usealla eri perusteella. (Pyne, et al., 2016)

2.2 Käyttötapaukset älykkäästä datan käytöstä kirjallisuudessa

Tässä luvussa esitellään muutama kirjallisuudesta löytyvä yksityiskohtaisesti kuvattu käyttötapaus Big dataan ja älykkääseen kunnossapitoon liittyen. Lyhyitä kuvauksia tapauksista löytyy kirjallisuudesta runsaasti, mutta yksityiskohtaisesti kuvattuja tapauksia niukemmin.

2.2.1 Case MeadWestvaco Covington (Fu & Hart, 2016)

Covingtonin paperitehtaalla syntyi vuoden 2013 syksystä eteenpäin huomattavasti aiempaa enemmän painaumajälkiä tuotettuun paperiin. Jälkien aiheuttamat ylimääräiset pesut aiheuttivat 29 % nettotuotannon laskun paperikoneella 1. Työryhmä perustettiin tutkimaan ongelmien lähdettä ja juurisyitä alkuvuonna 2014. Jälkien todettiin syntyvän pääasiassa kuivatusosalla, kun kuivatussylinteriin tarttui materiaalia. Syyt materiaalin tarttumiseen saattoivat työryhmän mukaan löytyä laajasti tehtaan eri osista, sellun valmistuksesta ja valkaisusta paperikoneelle. Mahdollisiksi ongelmanaiheuttajiksi todettiin mm. rainan kosteus kuivatusosaan mennessä, kuitujen lujuus, laitteiston tila ja -toimintaparametrit.

Käyttödataa kerättiin tehtaan eri järjestelmistä, ja varmasti irrelevantit datanlähteet karsittiin pois työryhmän toimesta. Datanlähteiden karsiminen laski määrän lähes 60 000:sta alle 8 000:een. **GE Intelligent Platforms (Csense Troubleshooter)** valittiin datan käsittelyyn. Datapisteitä kerättiin prosessista minuutin välein. Ongelmat eivät olleet sidonnaisia tiettyihin paperilaatuihin, joten ongelmatilanteita käsiteltiin päiväkohtaisesti. Prosessidatasta laskettiin päivittäiset keskiarvot ja varianssit, ja on/off arvoja käsiteltiin arvoilla 1-1 440 [minuuttia päivässä]. Määritelty data ladattiin ajalta marraskuu 2011 – elokuu 2014 viidentoista tehtävään ohjelmoidun tietokoneen avulla. Datan lataamiseen kului

noin viikko. Työryhmältä kului latauksen lisäksi kaksi viikkoa datan muotoiluun analyysiohjelmistolle käytettävään muotoon, Excel -taulukoiksi.

GE Intelligent Platforms (Csense Troubleshooter) käytti algoritmeja entropy gain (esitelty luvussa 3.3.2), chi-square (esitelty luvussa 3.2.2), ja Gini index (esitelty luvussa 3.3.2) muuttujien tulkintaan päätöspuussa (esitelty luvussa 3.3.2). Työryhmä määritteli ohjelmistolle normaalin toiminnan päivät ja korkean rejektimäärän päivät. Analyysissa käytettiin 15 korkean rejektimäärän päivää ja 230 normaalin toiminnan päivää. Lisäksi määriteltiin ongelmakohdetta edeltäville alueille viiveajat ongelmatilanteiden ja ylävirrassa olevien vaiheiden yhdistämiseksi. Esimerkiksi sellun valmistuksen ja ongelmakohdan välillä käytettiin 18 tunnin viivettä.

Ohjelmisto löysi useita mahdollisia juurisyitä jälkien synnylle sellunvalmistuksesta ja paperikoneelta. Esimerkiksi tikkupitoisuuden muutos aiheutti ylivalkaisua valkaisulaitoksella, mikä puolestaan saattoi heikentää sellun lujuusominaisuuksia sekä kuitujen sitoutumista. Myös mm. valkolipeän väkevyyden vaihtelu ja kappa-arvon vaihtelu vaikuttivat heikompien, ylivalkaistujen kuitujen muodostumiseen. Klooridioksidin matalan virtausnopeuden todettiin aiheuttavan yllä mainittujen arvojen vaihtelua.

Paperikoneella ongelmapäivät korreloivat matalan puristusvoiman, vähäisen alipaineen ja alhaisen höyrynpaineen kanssa. Kosteampi raina kuivatusosan alussa aiheutti suuremman todennäköisyyden materiaalin tarttumiseen kuivatussylinterille.

Löydettyjen korrelaatioiden perusteella tehtiin muutoksia prosessien toimintaan. Esimerkiksi ylivalkaisua pyrittiin välttämään ilman sellun vaaleuden menetystä. Muutosten jälkeen paperissa olevien jälkien määrän todettiin pienenneen huomattavasti, vähentäen tehtaan pesuseisokkiaikaa. Tulos viittaa onnistuneeseen Big data -analyysin ja prosessitiedon yhdistämiseen ongelman juurisyiden selvittämiseksi.

2.2.2 Case UPM Kaukas (Nykyri, 2018)

Vuonna 2018 tehdyssä diplomityössä pyrittiin ennustamaan UPM Kaukaan tehtaalla sähkömoottorin ylikuormituksia ja sellun tuotannon määrää datan perusteella. Moottorien ongelmatilanteiden ennustaminen koettiin tärkeäksi, sillä tehtaalla on tuhansia sähkömoottoreita.

Sähkömoottorin datan analysointi suoritettiin Pythonilla ja sen kirjastoilla scikit-learn ja Pandas. Dataa saatiin Metso DNA automaatiosysteemistä Excel add-inin avulla. Aikahaarukkana datalle toimi heinäkuu 2016 - heinäkuu 2017. Data-analyysin kohteena toimi pumppua pyörittävä 315 kW vaihtovirtamoottori. Kerätty data sisälsi tietoa viereisten moottorien virroista, virtauksen massasta, venttiilien asennoista, ja muutamasta muusta parametrista. Lisäksi käytettiin Metson DNA:sta saatua dataa totuusarvomuodossa virtapiikkien luokitteluun.

Random Forest -algoritmia hyödyntämällä kyettiin havaitsemaan historia-testidatasta 67/68 tapauksista, joissa ylikuormitus oli tapahtunut. Vääriä positiivia havaintoja oli 34. Kokonaisuudessaan havaintoja oli lähes 8 800.

Työssä koitettiin myös ylikuormituksien ennustamista 10 minuuttia tulevaisuuteen. Havaintotarkkuus oli ennustuksessa huomattavasti heikompi kuin historiadatan kanssa. Parhaiten selviytyivät Random Forest ja Gradient Boosting -algoritmit, jotka on esitelty lyhyesti luvussa 3.3.2.

Sama ennustava malli rakennettiin myös Microsoft Azure -pilvipalveluun demona. Oman sovelluksen lisäksi kokeiltiin Azuren valmiita ratkaisuja, mutta nämä todettiin heikoiksi monimutkaisuutensa ja huonon muokattavuutensa vuoksi.

Sellun tuotannon määrää yritettiin ennustaa saman moottorista ja sen ympäristöstä saadun datan perusteella. Multi-layer Perceptron Regressor -algoritmilla, eli neuroverkkoregressiolla, päästiin parhaisiin tuloksiin usealla eri ennustusajalla. Tuloksien tarkkuus oli yli 90 % tunnin ennustuksiin asti.

2.2.3 Case Valukomponenttitehdas (Lee, et al., 2017)

Etelä-Korealaiselle auton osia ja elektroniikkakomponentteja valmistavalle sadan henkilön yritykselle suunniteltiin valun ongelmia tarkkaileva Big data -järjestelmä. Tarkkailun kohteena toimi yksittäinen kylmäkammiovalulaite. Valamisessa kuumaa, sulaa, paineistettua metallia injektoidaan muottiin. Metallin kiinteydyttyä se poistetaan muotista. Valoksille oli etukäteen määritelty kahdeksan erilaista ongelmatapausta, joihin sisältyivät mm. halkeamat ja epämuodostumat. 75 eri muuttujaa tarkkailtiin vikojen analysoinnissa.

Datan käsittely tehtaalla tapahtuu Hadoop-pohjaisessa järjestelmässä. Hadoop on esitelty luvussa 4.4. Data liikkuu antureista älykkään välilaitteen läpi, josta se siirtyy tehtaalla perinteisesti käytettyyn järjestelmään, joka kokoaa tiedot. Samaan järjestelmään syötetään myös laadunvarmistustiedot, eli käytännössä ongelmahetket, käsin. Perinteisestä järjestelmästä data siirretään Big data -järjestelmään, joka esikäsittelee ja tallentaa tiedon hajautetusti.

Big data -järjestelmässä lokidata ensin luokitellaan ja muunnetaan määriteltyyn muotoon. Muodon muuntamisen jälkeen data suodatetaan, eli mm. tyhjät ja merkityksettömät arvot poistetaan. Suorittamalla esikäsittely ennen tallentamista tiedot vievät vähemmän tilaa.

Laadunvarmistustoimenpiteet suoritetaan tehtaalla erissä. Järjestelmä tuottaa dataa jokaiselta valulta erikseen, mutta tarkistusdata saadaan harvemmalla syklillä. Yhdellä syklillä tuotetuista osista tarkastetaan virheiden lukumäärä, ja valuprosessin arvoja tarkkaillaan eri keinoin syklin aikana, esimerkiksi keskiarvojen, maksimiarvojen ja keskihajontojen avulla.

Eri prosessiparametrien merkityksellisyys selvitettiin korrelaatioanalyysin avulla. Analyysin mukaan kaikkiaan 13 parametria ovat kriittisiä prosessin virheettömän toiminnan kannalta. Ongelmiin eniten vaikuttavat parametrit kohdelaitteella olivat matala iskunopeus, korkea nopeus, ja maksiminopeus. Korrelaatioanalyysin seurauksena tehtaalle on tarkoitus luoda neuroverkko, joka tarkkailee 13 merkittävintä parametria viallisten tuotteiden ennustamiseksi. Projekti oli artikkelin kirjoittamisen aikaan vielä kesken, joten saatujen tuloksien validointia ei ole suoritettu.

Projektin aikana ilmeni muutama merkittävä haaste. Vanhanaikainen perinnejärjestelmä oli haastava yhdistää uuteen järjestelmään, sillä laitteiston suora liittäminen omaan järjestelmään oli kielletty valmistajan puolelta. Järjestelmä ei ollut myöskään kykenevä tarkkailemaan yksittäisiä valoksia, minkä vuoksi tarkkailu suoritettiin valujoukoille. Ongelma olisi ollut ratkaistavissa, mikäli yrityksellä olisi resursseja liittää RFID -siru jokaiseen valokseen.

2.2.4 Case Poikkeuksien havaitseminen roottorin toiminnassa (Silipo, et al., 2018)

KNIME AG pyrki havaitsemaan ennalta määrittelemättömiä ongelmia roottorin toiminnassa. Tarkkailun kohteena oleva roottori oli anturoitu 28:lla anturilla, jotka oli sijoitettu roottorin kahdeksaan eri osaan. Antureita oli tuloakselissa (input shaft), neljässä eri laakerissa, kahdessa hammaspyörässä, sekä momentin mittauksessa. Kukin anturi tuotti oman sarjansa aikaleimattua informaatiota. Yhteensä tietoa kerättiin ajalta 1.1.2007 – 20.4.2009. Kyseisellä aikavälillä tapahtui yksi laiterikko 21.7.2008. Tutkimus tehtiin anonyymista lähteestä saadulla datalla jälkikäteen.

Antureiden signaalit käsiteltiin nopealla Fourier'n muunnoksella, jonka avulla saatiin yhteensä 313 aikaleimattua sarjaa. Kukin sarja käsitteli tiettyä 100 Hz laajuista taajusaluetta tietyssä paikassa. Taajuusalueet olivat välillä 0 – 1 200 Hz. Datan puuttuvat arvot korvattiin edellisellä mitatulla arvolla.

Poikkeuksien havaitsemiseksi sovellettiin kahta menetelmää, perinteistä Control Chart -menetelmää, sekä autoregressioon perustuvaa menetelmää.

Control Chart -menetelmässä laskettiin kullekin sarjalle kumulatiiviset keskiarvot ja keskihajonnat. Rajoina menetelmässä käytettiin arvoja keskiarvo ± 2*keskihajonta. Ensimmäisen tason hälytys annettiin rajojen ylittyessä tai alittuessa. Ensimmäisen tason hälytystä ei vielä välitetty henkilöstölle, vaan sitä seuraamalla määritettiin hälytystaso 2. Mitatuille pisteille annettiin arvo 0 normaalin toiminnan tapauksessa, ja arvo 1 hälytyksen tapauksessa.

Hälytystaso 2 määriteltiin päiväkohtaisesti. Tämä tapahtui laskemalla kaikista aikaleimallisista sarjoista hälytystasojen keskiarvo. Kun hälytystasojen keskiarvo ylitti arvon 0,25, annettiin toisen tason hälytys laitteiston tarkastamiseksi.

Control Chart -menetelmällä suoritetussa roottorin kunnon tarkkailussa havaittiin laitteen vaurioituminen ajoissa, mutta menetelmä aiheutti myös ennen todellisen ongelman syntyä kaksi väärää hälytystä yli vuoden ennen todellista rikkoutumista.

Autoregressiivisessä menetelmässä opetettiin kullekin aikaleimalliselle sarjalle erikseen lineaarisen regression malli ennustamaan tietyn hetkisen mittauksen arvo käyttämällä kymmentä aiempaa mittausarvoa. Mallien ennustamien arvojen ja todellisten mittaustulosten arvojen eroille laskettiin keskiarvot ja keskihajonnat. Ideana menetelmässä

on, että kyseiset autoregressiiviset mallit ovat hyviä ennustamaan normaalin toiminnan arvot, mutta huonoja ennustamaan poikkeavan toiminnan arvot. Tarkkailemalla ennustuksen tarkkuutta voidaan siten tehdä johtopäätöksiä laitteen kunnosta.

Autoregressiiviselle menetelmälle rakennettiin vastaavasti ensimmäisen tason hälytysraja, jolloin ennustuksen ero todellisuudesta oli yli kaksi keskihajontaa. Jokaiselle aikaleimalle jokaiselle muuttujalle sijoitettiin hälytysrajan ylityksen perusteella arvo 0 tai 1.

Toisen tason hälytysrajan määritystä varten jokaisen sarjan jokaiselle pisteelle laskettiin viimeisen 21 ensimmäisen hälytystason arvon (0 tai 1) keskiarvo. Laskemalla näistä keskiarvoista rivikohtainen jokaisen sarjan huomioiva keskiarvo saatiin toisen hälytysrajan määrityksessä käytettävä arvo. Rajana hälytykselle toimi 0,01, ja tilanteessa, jossa tämä ylittyi, lähetettiin kunnossapidolle tarkastuskehotus.

Tarkkailun kohteena olleen laitteen vaurioituminen oli helposti havaittavissa tarkkailemalla toisen asteen hälytyksiä, sekä eri sarjojen kontribuutiota näihin hälytyksiin. Vaurion alku oli havaittavissa jo maaliskuussa ja kävi todella selväksi toukokuussa, kun varsinainen rikkoutuminen tapahtui vasta heinäkuun puolen välin jälkeen.

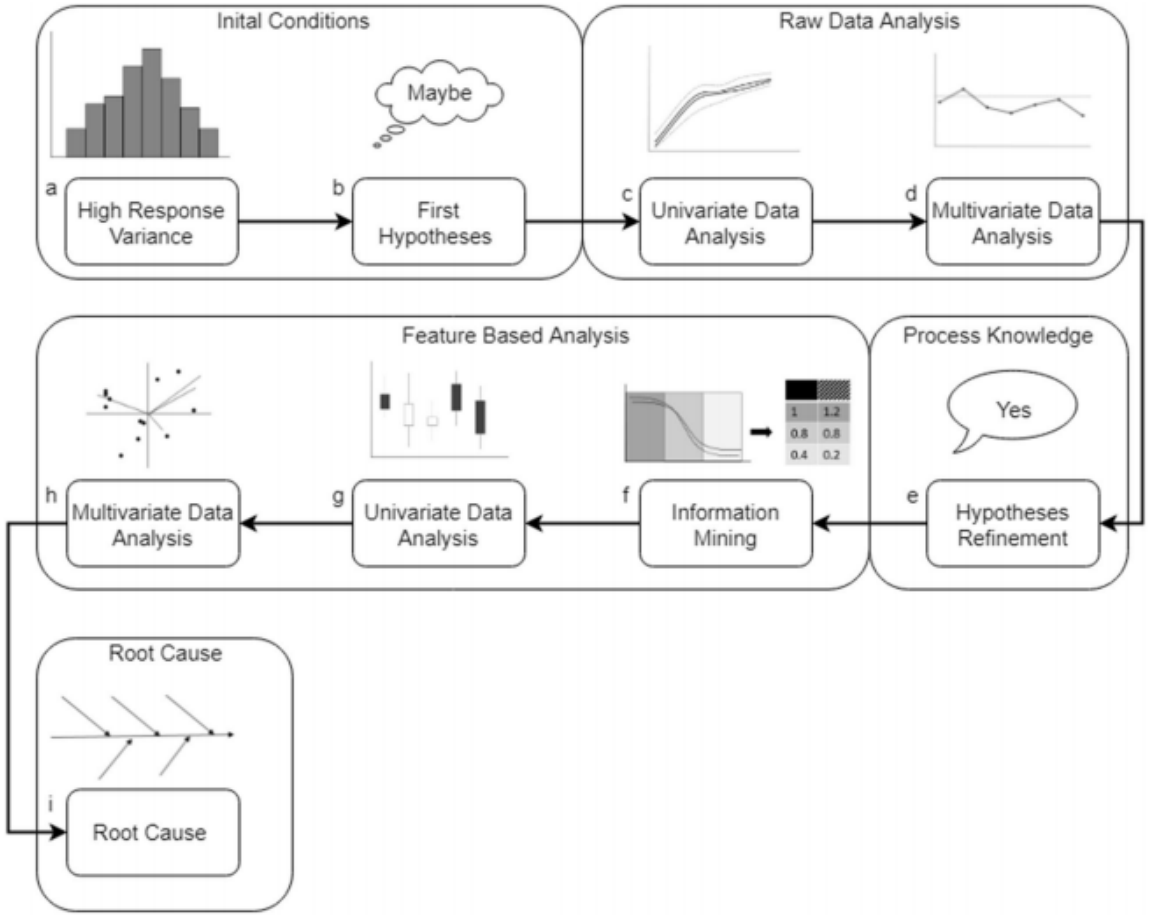
3. Data-analyysi ja tiedon louhinta

Data-analyysi määritellään tietokoneella tehtäväksi suurten datamäärien analysoinniksi päätöksenteon tueksi. Data-analyysiin liittyy monia tieteenhaaroja tilastotieteistä keinoälyyn. Data-analyysi on järjestelmällinen prosessi, johon kuuluvat tiedon valinta ja arviointi, puhdistus ja suodatus, visualisointi ja analysointi, sekä tulkinta ja tulosten arviointi. (Runkler, 2016)

Tiedon louhinta, eli data-mining, tarkoittaa kiinnostavan tiedon erottamista datamassasta. Kiinnostavan tieto tarkoittaa tiedon louhinnan yhteydessä tietoa, joka on ymmärrettävää, hyödyllistä, uutta ja oikeaa. (Runkler, 2016) Käytännössä tiedon louhintaan siis sisältyvät kaikki tässä luvussa esitellyt tekniikat.

Kirjallisuudessa esitetään raakadata-analyysin ja ominaisuuksiin perustuvan analyysin yhdistelmän olevan tehokas keino juurisyyanalyysissa. Raakadata-analyysia hyödynnetään ensin ensimmäisten hypoteesien muodostamisessa, minkä jälkeen hypoteeseja lähdetään

koettamaan ominaisuuksiin perustuvilla analyysikeinoilla, kuten mallien sovituksella. Analyysin systematiikka on esitetty kuvassa 1. (Borchert, et al., 2018)



Kuva 1 Juurisyyanalyysin systemaattinen suorittaminen dataan perustuen. (Borchert, et al., 2018)

3.1 Koneoppiminen

Koneoppiminen (eng. Machine Learning) tarkoittaa systeemin oppimista datan avulla. Oppiminen ei siis ole täsmällisesti ohjelmoitua, vaan systeemi etsii riippuvuuksia ja kuvioita datasta. Opitun perusteella systeemi yleistää tulokset ja pystyy ennustamaan tapahtumia uudesta datasta. (Guller, 2015)

Koneoppimisen tarkkailun kohteita kutsutaan joko muuttujiksi tai ominaisuuksiksi. Toisinaan muuttujia kutsutaan myös ulottuvuuksiksi. Esimerkiksi moniulotteisesta datasta

puhuminen on yleistä. Kategorisista muuttujista tai ominaisuuksista puhutaan, kun tarkoitetaan muuttujaa, jolla on tietty määrä arvoja, joita ei voida laittaa järjestykseen. Tämä voi tarkoittaa sosiaalisen median tapauksessa esimerkiksi sukupuolta, tai teollisuuden laitteen tapauksessa kunnossa tai epäkunnossa olemista. Numeerisista muuttujista puhutaan, kun muuttujan arvot voidaan laittaa suuruusjärjestykseen. Numeeriset muuttujat voivat olla jatkuvia (esim. lämpötila) tai diskreettejä (esim. avoimien venttiilien määrä). (Guller, 2015)

Koneoppimisessa systeemi merkitsee datan esim. klusteroinnin avulla. Systeemi pyrkii mallien avulla löytämään yhteyksiä datajoukoista laskeakseen tai ennustaakseen riippuvia muuttujia. Malli sovitetaan dataan, ja mallin avulla merkitään uusia havaintoja. Mallin sovittaminen on laskennallisesti raskasta, mutta sovituksen jälkeen sen hyödyntäminen on kevyttä. (Guller, 2015)

Koneoppimissysteemin ennustamiskyvyn testaamiseen käytetään testidataa. Testidatan tulee olla eri dataa kuin mallin opetuksessa on käytetty data edustavan testituloksen saamiseksi. Yleisesti historiadataa käytettäessä noin 80 %:a datasta käytetään mallin opettamiseen, ja 20 %:a mallin testaamiseen. Mikäli samaa dataa käytetään testauksessa ja opetuksessa, antaa malli testissä todellisuutta paremman ennustuskyvyn. (Guller, 2015)

Koneoppimista voidaan käyttää laajasti eri aloilla teollisuudesta markkinoiden ennustamiseen, ja koneoppimisen käyttö yleistyy koko ajan. Koneoppimisen käyttökohteet voidaan jakaa karkeasti luokitteluun, klusterointiin, regressioon, poikkeuksien havainnointiin, suositteluun ja muuttujien karsimiseen eli dimensioiden vähentämiseen. Itse koneoppimisalgoritmit voidaan karkeasti jaotella ohjattuihin ja ohjaamattomiin algoritmeihin. (Guller, 2015)

Koneoppiminen vaatii myös hyperparametreiksi kutsuttuja syötteitä. Hyperparametreilla määritellään esimerkiksi, kuinka kauan mallin sovittamiseen voidaan käyttää aikaa, tai miten tarkkoja ennustuksia halutaan. (Guller, 2015) Kaikkia tässä luvussa esiteltyjä metodeja voidaan käyttää koneoppimisessa tai koneoppimisen tukena data-analyysissa.

3.2 Korrelaatio

Korrelaatio määrittää eri ominaisuuksien välisen yhteyden. Korrelaatio voi olla lineaarista tai epälineaarista. Lineaarista korrelaatiota hyödyntävät metodit ovat luotettavia ja tehokkaita tapoja käsitellä lineaarisia riippuvuussuhteita. Epälineaarista korrelaatiota käyttävät metodit voivat löytää monimutkaisempia riippuvuussuhteita, mutta vaativat käyttäjältään suurempaa tarkkuutta ja tietotaitoa. Korrelaatio eri ominaisuuksien välillä ei aina viittaa syy-seuraussuhteisiin. Kahden muuttujan välillä olevassa korrelaatiossa voi olla kyse syy-seuraussuhteesta, sattumasta, tai kolmannesta tekijästä, jolla on syy-seuraussuhde molempiin tekijöihin. (Runkler, 2016)

Lineaarikorrelaatiossa ominaisuudet ovat yhteydessä toisiinsa nimensä mukaan lineaarisesti. Lineaarikorrelaatiota voidaan mitata usealla mittarilla.

3.2.1 Kovarianssi

Kovarianssi määrittää ominaisuuksien riippuvuussuhteen yksittäisen numeron avulla. Mikäli kovarianssi on suuri positiivinen luku, ovat ominaisuudet voimakkaasti lineaarisesti riippuvaisia. Lähellä nollaa olevat luvut kertovat riippuvuuden olevan pientä. Suuret negatiiviset luvut kertovat voimakkaasta käänteisestä riippuvuudesta. Kovarianssi sarjojen välillä voidaan laskea yhtälöllä 1. (Runkler, 2016)

$$C_{yx} = \frac{1}{n-1}\sum_{l=1}^{n}(x_l - \bar{x})\,(y_l - \bar{y}) \quad (1)$$

jossa

C_{yx}	Korrelaatiokerroin ominaisuuksien x ja y välillä
n	Havaintojen lukumäärä
x_l	Ominaisuuden x arvo havainnossa l
$\bar{x}$	Ominaisuuden x keskiarvo
y_l	Ominaisuuden y arvo havainnossa l
$\bar{y}$	Ominaisuuden y keskiarvo

Pearsonin korrelaatiossa otetaan huomioon mittaustulosten suuruusluokka lisäämällä ominaisuuksien mittaustulosten keskihajontojen tulo kovarianssin jakajaksi. Näin saadaan suuruudeltaan välillä [-1, 1] oleva lineaarista korrelaatiota kuvaava arvo, mikä on kuvattu yhtälössä 2.

$$S_{xy} = \frac{C_{xy}}{s_x s_y} \quad (2)$$

jossa

S_{xy}	Pearsonin korrelaation arvo ominaisuuksien x ja y välillä
s_x	Ominaisuuden x mittaustulosten keskihajonta
s_y	Ominaisuuden y mittaustulosten keskihajonta

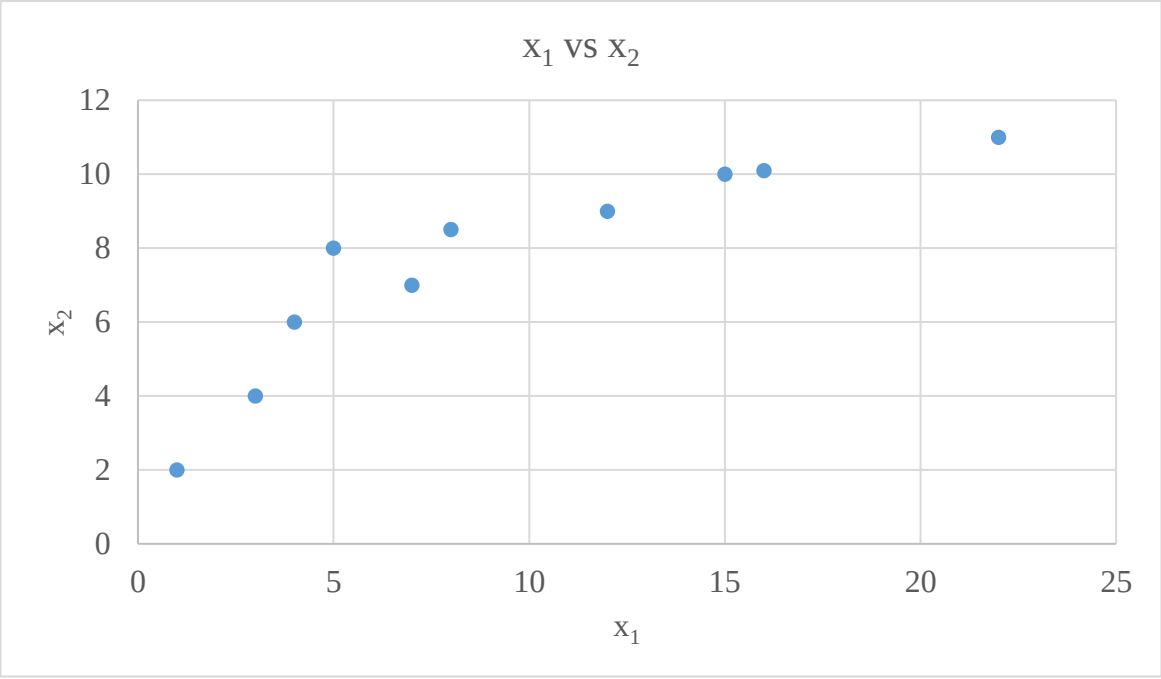
Lineaarikorrelaation lisäksi voidaan laskea sijoitukseen perustuvaa korrelaatiota (Rank Correlation) muutamalla menetelmällä. Esimerkiksi Spearmanin (Spearman, 2010) ja Kendallin korrelaatiot perustuvat arvojen sijoitukseen suhteessa muihin saman muuttujan arvoihin. Toisin sanoen suureen pienin arvo saa arvon 1, seuraavaksi pienin 2, jne. kunnes

suurin arvo saa arvokseen *n*. Spearmanin korrelaatiossa suureiden sijoituksille sovelletaan täsmälleen samaa yhtälöä kuin Pearsonin korrelaatiossa suureiden arvoille.

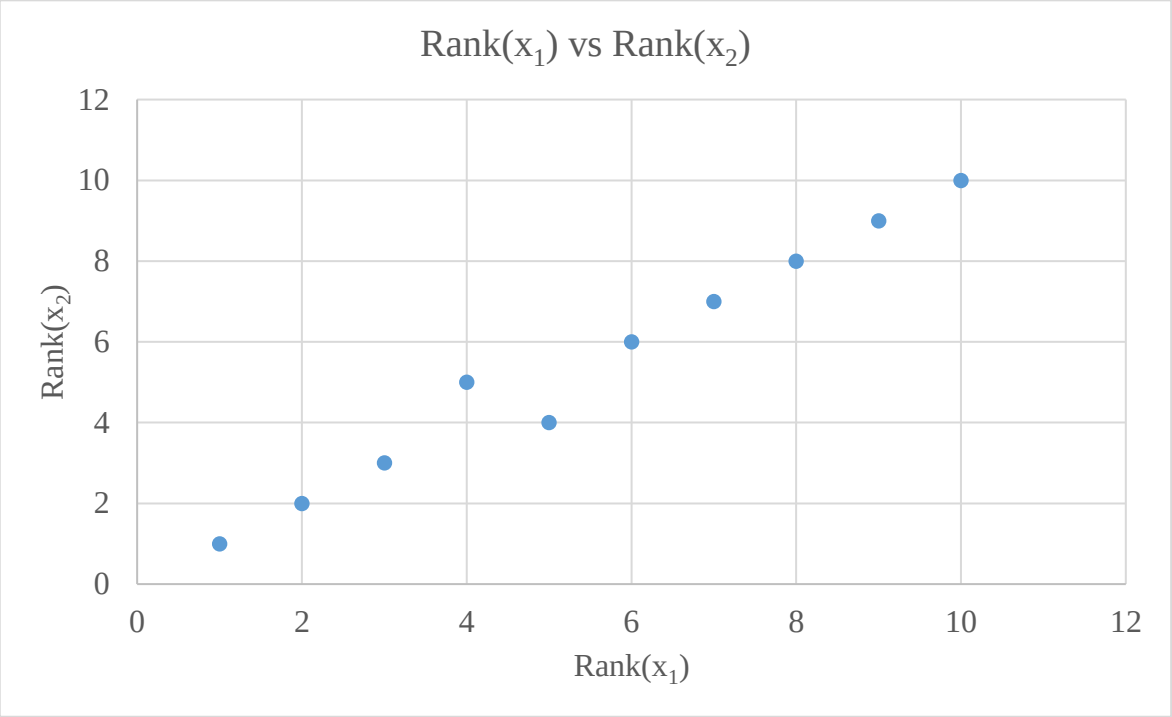
Taulukko 2 ja kuvat 2-3 sisältävät manuaalisesti rakennetun esimerkkidatan, jota tutkimalla voidaan havainnollistaa Spearmanin korrelaation ja Pearsonin korrelaation eroa. Kuva 2 sisältää esimerkin alkuperäiset arvot, kuva 3 sijoitukset.

Taulukko 2 Kuvitteellinen data muuttujille x_1 ja x_2, sekä näiden sijoitukset suhteessa muihin arvoihin.

x_1	**x_2**	**Rank(x_1)**	**Rank(x_2)**
1	2	1	1
5	8	4	5
12	9	7	7
3	4	2	2
15	10	8	8
16	10,1	9	9
22	11	10	10
4	6	3	3
8	8,5	6	6
7	7	5	4



Kuva 2 Suureiden x_1 ja x_2 arvot toistensa suhteen.



Kuva 3 Suureiden x_1 ja x_2 sijoitukset toistensa suhteen.

Kuvia 2-3 vertaamalla nähdään, että kuvan 2 mukainen epälineaarinen kasvaminen muuttuu sijoituksien avulla lineaariseksi. Soveltamalla alkuperäistä dataa ja sijoitusdataa yhtälöön 2 saadaan Pearsonin korrelaatioksi 0,885 ja Spearmanin korrelaatioksi 0,988. Epälineaariset, mutta jatkuvasti samaan suuntaan toistensa suhteen muuttuvat muuttujaparit saavat siis Spearmanin sijoituksiin perustuvalla korrelaatiolla suuremmat korrelaation arvot kuin suoralla lineaarikorrelaatiolla, mikä voi epälineaarisia systeemeitä tarkastellessa antaa lisäinformaatiota.

Kendallin sijoituksiin perustuvassa korrelaatiossa lasketaan korrelaatiota kuvaavalle suureelle τ arvo perustuen ristiriitaisin (Discordant) ja yhdenmukaisiin (Concordant) pareihin tutkittavissa sijoituksissa. Suureen arvo lasketaan yhtälöllä 3. (Statistics How To, 2019)

$$\tau = \frac{C-D}{C+D} \quad (3)$$

jossa C (Taulukossa alla olevien) yhdenmukaisten parien määrä

D (Taulukossa alla olevien) ristiriitaisten parien määrä

Käytännössä Kendallin korrelaatiota laskiessa jompi kumpi tutkittavista suureista järjestetään suuruusjärjestykseen, minkä jälkeen toisen arvoja tarkkaillaan. Tarkkailtaessa järjestämättömän suureen tietyn rivin arvoa, kaikki alla olevissa riveissä olevat suuremmat arvot ovat yhdenmukaisia, kaikki pienemmät ristiriitaisia. Näiden arvojen määrät lasketaan kullekin riville, minkä jälkeen arvot lasketaan yhteen.

Yllä esitellyssä esimerkkitapauksessa parien järjestäminen ja laskeminen tapahtuvat taulukon 3 mukaisesti:

Taulukko 3 Yhdenmukaisten ja ristiriitaisten arvojen laskeminen Kendallin korrelaatioon. Taulukon Rank(x_2) -arvoja verrataan alempien rivien arvoihin.

Rank(x_1)	Rank(x_2)	C	D
1	1	9	0
2	2	8	0
3	3	7	0
4	5	5	1
5	4	5	0
6	6	4	0
7	7	3	0
8	8	2	0
9	9	1	0
10	10		

Yhdenmukaisten arvojen kokonaismääräksi saadaan esimerkissä C = 44, ristiriitaisten arvojen määräksi D = 1. Kun arvot sijoitetaan yhtälöön 3, saadaan Kendallin korrelaatiota kuvaava suure τ = 0,956.

Erilaisilla korrelaation laskentamenetelmillä saadaan tilanteesta riippuen huomattavankin erilaisia tuloksia. Korrelaatioiden laskeminen on nykyisin ohjelmallisesti niin nopeaa, että kaikki korrelaatiot voidaan laskea käytännössä missä vain tapauksessa.

3.2.2 χ^2-testi

Etsimällä lineaarisia korrelaatioita eivät voimakkaan epälineaariset korrelaatiot näyttäydy välttämättä lainkaan. Yksi keino mitata epälineaarista riippuvuutta on χ^2-testi (Chi-Squared test). (Runkler, 2016)

χ^2-testissä tutkittavien suureiden eri arvot jaetaan eri lokeroihin histogrammien rakentamiseksi. Jatkuvat muuttujat diskretisoidaan. Mittausdata, joka sisältää 1000 mittausta voidaan jakaa esimerkiksi viiteen lokeroon. Mitattavat suureet voivat olla esimerkiksi telan pyörimisnopeus ja -värähtely. Suureita mitataan samaan aikaan esimerkiksi 10 sekunnin keskiarvoilla. Kuvassa 4 on rakennettu täysin kuvitteelliset, yksiköttömät histogrammit kahdelle mitattavalle suureelle.

Kuva 4 Kahden kuvitteellisen suureen histogrammit.

Kuvan 4 tapauksessa nähdään jo vilkaisulla, että suureiden välillä on todennäköisesti jonkinlaista epälineaarista korrelaatiota, joka havaittaisiin mahdollisesti kovarianssillakin. Tilanteissa joissa näin ei ole, tai joissa tietokone laskee korrelaatiota itsenäisesti, rakennetaan näistä kahdesta histogrammista taulukko, joissa saman mittauksen eri suureiden arvot lasketaan samaan lokeroon. Esimerkin histogrammeihin liittyvät kuvitteelliset arvot ovat taulukossa 4.

Taulukko 4 Esimerkkinä toimivat havainnot. Vasemmassa sarakkeessa on suureen 2 (j) lokerot, ylimmässä rivissä suureen 1 (i) lokerot. Risteämäkohdissa on molempiin lokeroihin samanaikaisesti osuvien mittausten määrä.

HAVAINNOT	50-60	60-70	70-80	80-90	90-100	YHTEENSÄ
300-400	1	11	12	2	5	31
400-500	3	31	220	180	6	440
500-600	4	22	12	274	8	320
600-700	1	31	10	69	9	120
700-800	2	26	22	35	4	89
YHTEENSÄ	11	121	276	560	32	1000

Kun todelliset havainnot on taulukoitu, lasketaan niin sanotun nollahypoteesin mukaiset arvot kullekin solulle. Tämä tarkoittaa arvoja, jotka olisivat odotettavissa tapauksessa, jossa suureet 1 ja 2 eivät liity toisiinsa millään tavoin. Arvot lasketaan kullekin lokerolle yhtälöllä 4, jolloin saadaan taulukon 5 mukaiset arvot. (Sigma Plus Statistiek, 2019)

$$e_{ij} = \frac{o_i o_j}{N} \tag{4}$$

jossa e_{ij} Odotettu arvo solulle (i, j)

o_i Lokeron i havainnot yhteensä

o_j Lokeron j havainnot yhteensä

N Havaintojen kokonaismäärä

Ensimmäiseen soluun [1, 1] eli [50-60, 300-400] Nollahypoteesin mukainen arvo lasketaan seuraavasti:

$$e_{1,1} = \frac{11*31}{1000} = 0{,}341$$

Taulukko 5 Nollahypoteesin mukaiset odotusarvot eri soluille.

OLETETUT (NULL HYPOTHESIS)	50-60	60-70	70-80	80-90	90-100	YHTEENSÄ
300-400	0,341	3,751	8,556	17,360	0,992	31
400-500	4,840	53,240	121,440	246,400	14,080	440
500-600	3,520	38,720	88,320	179,200	10,240	320
600-700	1,320	14,520	33,120	67,200	3,840	120
700-800	0,979	10,769	24,564	49,840	2,848	89
YHTEENSÄ	11	121	276	560	32	1000

Nollahypoteesin mukaisten odotusarvojen ja havainnoitujen arvojen perusteella voidaan laskea χ^2 -pistearvot kullekin solulle. Näiden arvojen summa on χ^2. χ^2 -arvon laskemiseen käytetään yhtälöä (5). Pistearvot kullekin solulle on esitetty taulukossa 6. (Sigma Plus Statistiek, 2019)

$$\chi^2 = \sum \frac{(o_{ij} - e_{ij})^2}{e_{ij}} \quad (5)$$

jossa o_{ij} havaintojen määrä solussa (i, j)

Ensimmäiseen soluun [1, 1] eli [50-60, 300-400] χ^2 -pistearvo lasketaan seuraavasti:

$$\chi^2_{1,1} = \frac{(1-0{,}341)^2}{0{,}341} = 1{,}274$$

Taulukko 6 Kullekin solulle (i, j) lasketut χ^2 .pistearvot.

X^2 -PISTEET	50-60	60-70	70-80	80-90	90-100
300-400	1,274	14,009	1,386	13,590	16,194
400-500	0,700	9,290	79,991	17,894	4,637
500-600	0,065	7,220	65,950	50,151	0,490
600-700	0,078	18,705	16,139	0,048	6,934
700-800	1,065	21,542	0,268	4,419	0,466

Kun kaikki pisteet esimerkkitapauksessa lasketaan yhteen, saadaan χ^2 = 352,503. Jotta arvolla olisi merkitystä, täytyy testin vapausaste määrittää. Vapausaste lasketaan yhtälöllä (6).

$$F_{Chi} = (i-1)(j-1) \quad (6)$$

jossa F_{Chi} χ^2 -testin vapausaste

i Suuretta 1 kuvaavien lokeroiden lukumäärä

j Suuretta 2 kuvaavien lokeroiden lukumäärä

Vapausasteeksi testissä saadaan 16. Vapausasteen saamisen jälkeen voidaan laskea jakaumasta tai tulkita taulukosta todennäköisyys havainnoille, mikäli kyseessä olevat suureet ovat toisistaan täysin riippumattomat. (Sigma Plus Statistiek, 2019) Esimerkkitapauksen vapausasteella 16, liittessä I olevan taulukon (Nazarathy, 2019) mukaan täysin toisistaan riippumattomilla suureilla 1 ja 2 on 1 % mahdollisuus jakautua satunnaisesti soluihin niin, että χ^2 -arvo on yli 32,00. Esimerkkitapauksessa χ^2 -arvo on 352,503 eli satunnaisuuden mahdollisuus on käytännössä nolla. Näin ollen nollahypoteesi voidaan todeta virheelliseksi eli suureiden 1 ja 2 välillä on korrelaatiota.

3.2.3 Varianssianalyysi (ANOVA) ja F-stat

Analysis of variance, eli ANOVA, on menetelmä, jossa eri luokkiin sijoittuvien arvojen välisiä ja sisäisiä variansseja verrataan kokonaisvarianssiin muuttujan merkityksellisyyden selvittämiseksi. Menetelmä ottaa huomioon luokkien keskiarvojen varianssin kokonaiskeskiarvosta sekä luokkien sisäisen varianssin. Menetelmän etu on sen toimivuus kategorisen luokkamuuttujan tutkimisessa numeeristen muuttujien suhteen. (TIBCO Software Inc. [5], 2014) Alla on esitetty menetelmä laskuesimerkin avulla. Laskuesimerkissä käytetään taulukossa 7 esitettyä datajoukkoa. Datajoukon arvot ovat

arvottu Excelin RAND() -funktiolla, käyttäen luokan 0 arvoille väliä 0,4 – 0,5 ja luokan 1 arvoille väliä 0,47 – 0,57.

Taulukko 7 Esimerkissä käytettävät muuttujan arvot ja niiden kategoriset luokat.

Luokka	Arvo
0	0,442
0	0,497
0	0,451
0	0,455
0	0,479
0	0,477
0	0,451
0	0,459
0	0,475
0	0,450
0	0,405
0	0,408
0	0,441
1	0,525
1	0,479
1	0,506
1	0,540
1	0,556
1	0,483

Menetelmän käyttö aloitetaan laskemalla keskiarvot ensin koko datajoukolle, minkä jälkeen kunkin luokan arvoille lasketaan omat keskiarvonsa. Esimerkin arvoilla koko datasetin keskiarvo on 0,473, luokkaan 0 kuuluvien arvojen keskiarvo 0,453, ja luokkaan 1 kuuluvien arvojen keskiarvo 0,515.

Keskiarvojen laskemisen jälkeen lasketaan kullekin kunkin ryhmän pisteelle varianssi ja arvot lasketaan yhteen. Näin saadaan ryhmien sisäisten neliöiden summa:

$$SSWG = \sum_j^k \sum_i^{n_j} \left(x_{ij} - \bar{x}_j\right)^2 \quad (7)$$

jossa *SSWG* Ryhmien sisäisten neliöiden summa (sum of squares within groups)

k Luokkien lukumäärä, esimerkissä 2

n_j Havaintojen lukumäärä luokassa j

x_{ij} Havainnon arvo indeksillä i luokassa j

$\bar{x}_j$ Luokan j keskiarvo

Arvoksi *SSWG* saa esimerkkitapauksessa 0,0133. Tämän neliösumman lisäksi ANOVA:ssa tarvitaan ryhmien välisten neliöiden summa:

$$SSBG = \sum_j^k n_j \left(\bar{x}_j - \bar{x}_{tot}\right)^2 \quad (8)$$

joss *SSBG* Ryhmien välisten neliöiden summa (sum of squares between groups)

n_j Havaintojen määrä luokassa *j*

$\bar{x}_j$ Luokan *j* keskiarvo

$\bar{x}_{tot}$ Kaikkien havaintojen keskiarvo.

SSBG:n arvoksi tulee esimerkissä 0,0157. Riippuvuudesta kertovan F-arvon laskemiseksi tarvitaan vielä ryhmien sisäinen vapausaste f_W ja ryhmien välinen vapausaste f_B. Vapausasteet voidaan laskea yksinkertaisilla yhtälöillä 9 ja 10.

$$f_W = n_{tot} - k \tag{9}$$

$$f_B = k - 1 \tag{10}$$

joissa n_{tot} havaintojen kokonaismäärä

k luokkien lukumäärä

Vapausasteiksi esimerkkiarvoilla saadaan siis f_W = 17 ja f_B = 1. F-arvo, eli muuttujan merkitsevyyttä kuvaava suure, voidaan laskea nyt yhtälöllä 10.

$$F = \frac{\frac{SSBG}{fB}}{\frac{SSWG}{fW}} \tag{11}$$

Esimerkkidatalla arvo F on 20,051. Arvoa voidaan käyttää suoraan muuttujien merkitsevyyden vertailussa. Kun mittauspisteiden määrä on eri muuttujille sama tai lähes sama, kertoo F:n suuruus muuttujien merkitsevyysjärjestyksen suhteessa luokkiin. F-stat voidaan χ^2 -arvon tavoin muuttaa vapausasteiden avulla riippumattomuuden todennäköisyydeksi. Todennäköisyysarvo p voidaan tulkita joko taulukosta tai kumulatiivisesta funktiosta. Taulukon (National Institute of Standards and Technology, 2013) avulla tulkittuna vapausasteilla f_W = 17 ja f_B = 1 F-arvon ollessa 8,4, on muuttuja ryhmien suhteen 1 % todennäköisyydellä riippumaton. Esimerkin tapauksessa F on huomattavasti suurempi, joten riippuvuus on käytännössä varmaa, kuten sen kuuluukin olla tapauksen järjestelyn perusteella.

3.2.4 Levenen testi

Levenen testiä voidaan käyttää tasaisten varianssien oletuksen tarkistamiseen eri otoksissa. Tämä oletus on voimassa esimerkiksi edellisessä luvussa esitellyssä ANOVA:ssa. Testin nollahypoteesina toimii siis eri otosten (luokkien) varianssien yhtäsuuruus $\sigma_1^2 = \cdots = \sigma_n^2$.

Nollahypoteesi hylätään, jos jollakin varianssiparilla yhtäläisyysehto ei täyty. (NIST/SEMATECH, 2012) Luvun 3.2.3 kahden luokan esimerkissä nollahypoteesi on siis $\sigma_1^2 = \sigma_2^2$.

Levenen testin tarkkailtava arvo on *W*. Arvoa verrataan samassa tapauksessa ANOVA:sta saatuun F-arvoon. Jos Levenen testin arvo *W* on suurempi kuin ANOVA:n *F*, hylätään nollahypoteesi, eli luokkien variansseja ei voida pitää samoina. *W* on määritelty yhtälössä (12) (NIST/SEMATECH, 2012)

$$W = \frac{(N-n_o)}{(n_o-1)} * \frac{\sum_{j=1}^{n_o} N_j \left(\bar{Z}_{j.}+\bar{Z}_{..}\right)^2}{\sum_{j=1}^{n} \sum_{i=1}^{N_j} \left(Z_{ji}+\bar{Z}_{j.}\right)^2} \qquad (12)$$

jossa

W	Levenen testin tarkkailtava arvo	
N	Havaintojen kokonaismäärä	
N_j	Otoksen (eli luokan) j havaintojen määrä	
n_o	Otosten lukumäärä	
Z_{ji}	Mittauksen j,i hajonnan itseisarvo luokassa, eli $\lvert x_{j,i} - \bar{x}_{j.} \rvert$	
jossa	$x_{j,i}$	mittaus i luokassa j
	$\bar{x}_{j.}$	luokan j keskiarvo
$Z_{j.}$	Z_{ji}:n keskiarvo luokassa j	
$Z_{..}$	Z_{ji}:n kokonaiskeskiarvo	

Levenen testin arvoksi *W* tulee luvun 3.2.3 esimerkkitapauksessa 0,650, mikä on pienempi kuin ANOVA:n *F*-arvo 20,051. Näin ollen nollahypoteesi hyväksytään ja varianssien eri ryhmissä katsotaan olevan samat. Levenen testi voi antaa ylimääräistä tietoa esimerkiksi ongelmien juurisyyn selvittämisessä, sillä luokkien varianssien erilaisuus voi olla kiinnostava tieto.

3.3 Datan luokittelu

Datan ohjatussa luokittelussa pyritään yhdistämään eri objekteja ominaisuuksien perusteella luokkiin. Tämä voi tarkoittaa esimerkiksi ihmisen luokittelemista sairaaksi tai terveeksi verenpaine- ja kehonlämpödatan perusteella tai laitteen luokittelua epäkuntoiseksi värähtelydatan ja energiankulutuksen perusteella. Väärän luokittelun mahdollisuus otetaan huomioon, ja luokittelun tarkkuutta arvioidaan eri mittareiden avulla. Dataa voidaan luokitella mm. Naiivilla Bayesilaisella luokittelulla (naive Bayes classifier), lineaarisella diskriminanttianalyysillä (linear discriminant analysis) sekä monella muulla menetelmällä. Luokittelua käyttävä järjestelmä opetetaan tunnistamaan esim. sairaustapaukset opetusdatan avulla, minkä jälkeen opetuksen onnistuminen validoidaan käyttämällä eri tunnettua dataa. (Runkler, 2016) Juurisyyanalyysi on tyypillisesti luokitteluongelma, sillä data merkataan esimerkiksi aikaleiman perusteella ongelmadataksi tai normaaliksi dataksi.

Luokittelun jälkeen kohde, esimerkiksi kunnossapidon tarkkailema kone, on joko (Runkler, 2016):

1. Luokiteltu huoltoa tarvitsevaksi ja todellisuudessa tarvitsee huoltoa (TP, true positive)
2. Luokiteltu kunnossa olevaksi ja todellisuudessa kunnossa (TN, true negative)
3. Luokiteltu huoltoa tarvitsevaksi ja todellisuudessa kunnossa (FP, false posititive)
4. Luokiteltu kunnossa olevaksi ja todellisuudessa tarvitsee huoltoa (FN, false negative)

Luokittelut voidaan jakaa siis neljään laatikkoon, jolloin saadaan nk. ”sekaannusmatriisi” (Confusion Matrix). Esimerkki sekaannusmatriisista on taulukossa 8.

Taulukko 8 Koneen seuraavan vuorokauden aikana hajoamisen ennustamisesta tehty kuvitteellinen sekaannusmatriisi. Vasen yläkulma TN, vasen alakulma FN, oikea yläkulma FP, oikea alakulma TP.

	Ennustus: Ei hajoa	Ennustus: Hajoaa
Todellisuus: Ei hajoa	330	44
Todellisuus: Hajoaa	59	23

Sekaannusmatriisissa ”hyviä” arvoja ovat siis vasen yläkulma ja oikea alakulma. Väärillä luokitteluilla voi esimerkin mukaisessa kunnossapidon tapauksessa olla merkittäviä kustannusvaikutuksia. Kunnossa olevalle, mutta huoltoa tarvitsevaksi luokitellulle laitteelle saatetaan tehdä ylimääräisiä huoltotoimenpiteitä. Huoltoa tarvitseva, mutta kunnossa olevaksi luokiteltu laite sen sijaan voi hajota prosessissa lopullisesti, aiheuttaen mahdollisesti koko prosessin pysähtymisen ja muita ongelmia. FP ja FN tapauksia kutsutaan myös tyypin 1 ja tyypin 2 virheiksi (Runkler, 2016). Luokittelua voidaan tarkkailla monilla tavoin yllä mainittujen neljän luokittelutuloksen perusteella, esimerkiksi yhtälöissä 13-16 esitettyjen täsmällisyyden (precision), tarkkuuden (accuracy) ja löytöasteen (recall) avulla (Barga, et al., 2015). Epätasapainoisissa tilanteissa, eli tilanteissa joissa eri luokkiin kuuluvien pisteiden määrä eroaa huomattavasti, voidaan käyttää myös tasapainotettua tarkkuutta (balanced accuracy) (Josefsson, 2017).

$$Täsmällisyys = \frac{TP}{TP+FP} \tag{13}$$

$$Tarkkuus = \frac{TP+TN}{TP+TN+FP+FN} \tag{14}$$

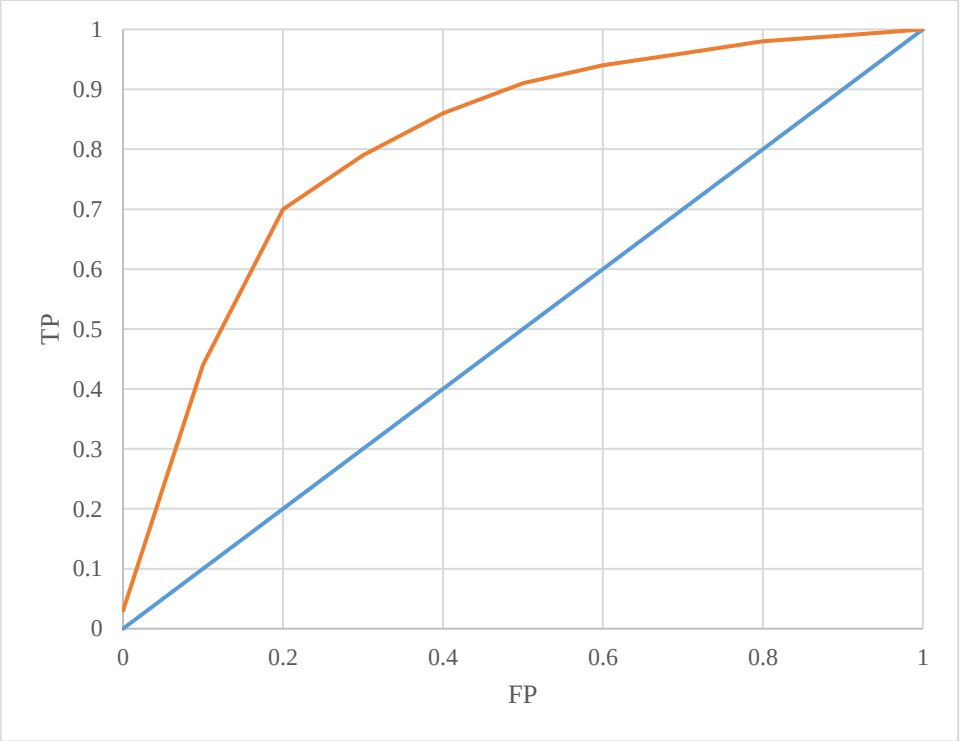
$$Löytöaste = \frac{TP}{TP+FN} \tag{15}$$

$$Tasapainotettu\ tarkkuus = \frac{\left(\frac{TN}{TN+FN}+\frac{TP}{TP+FP}\right)}{2} \tag{16}$$

Yksittäisen mittarin tarkkailu ei välttämättä kerro totuutta, sillä on helppoa rakentaa luokittelija, joka tunnistaa vain selvimmät tapaukset ja ei tee niiden osin virheitä. Hyvä luokittelija sen sijaan löytää myös epäselvempiä tapauksia, mutta ei silti tee huomattavia määriä virheitä. (Runkler, 2016)

Luokittelijan tehokkuutta eri TP arvoilla voidaan tarkkailla ROC-käyrän (Receiver Operating Characteristic curve) avulla. Käyrä kertoo löydettyjen TP ja FP tapausten suhteen

eri luokittelijan herkkyyksillä. (Barga, et al., 2015) Luokittelijat antavat tyypillisesti todennäköisyyden niin tutkittavalle kuin jokaiselle muullekin luokalle. Tämän todennäköisyyden rajaa muuttamalla saadaan eri luokittelutuloksia. Kaikki luokittelijat eivät kuitenkaan anna todennäköisyyksiä eri pisteille. Esimerkiksi päätöspuuluokittelija antaa normaalisti vain luokkatiedon (Fawcett, 2003). Tällöin luotu ROC-käyrä sisältää ainoastaan yhden pisteen. Osassa tapauksista luokittelijan sisälle katsottaessa nähdään kuitenkin todennäköisyyksiä, esimerkiksi päätöspuun tapauksessa eri haarojen todennäköisyydet (Fawcett, 2003). Yleisesti kaksiluokkaluokittelussa luokittelu tehdään, kun datapisteillä on yli 50 % todennäköisyys kuulua luokkaan. Vaihtamalla arvo esimerkiksi 30 %:iin löydetään todennäköisesti lisää TP tapauksia, mutta myös FP tapausten määrä kasvaa. Kuvitteellinen esimerkki ROC-käyrästä on esitetty kuvassa 5.



Kuva 5 ROC-käyrä luokittelijan toiminnalle kuvitteellisessa tapauksessa. Diagonaali kuvaa täysin satunnaista luokittelua.

Esimerkkikuvassa näkyy yleinen ilmiö luokittelijoiden suorituskyvyssä: kun TP tapausten määrä kasvaa, kasvaa myös FP tapausten määrä. Mitä jyrkempi käyrä on, sitä tehokkaampaa luokittelijan toiminta on. ROC-käyrää voidaan käyttää muuttujavalintojen ja oppimisen

hyperparametrien optimoinnissa – mitä suurempi pinta-ala käyrän alle jää, sitä paremmin luokittelu onnistuu. Tätä pinta-alaa kuvataan termillä AUC, eli Area Under Curve.

3.3.1 Regressio

Regressio on yksi datan luokittelussa ja arvojen ennustamisessa eniten käytetyistä menetelmistä. Regression avulla kyetään arvioimaan ominaisuuksien välisiä riippuvuussuhteita käytännössä. Luokittelun tapauksessa puhutaan logistisesta regressiosta. Kovariansseja voidaan käyttää lineaarisen regression mallien rakentamiseen ja relevanttien ominaisuuksien löytämiseen. Epälineaarisia riippuvuussuhteita voidaan myös kuvata lineaarisella regressiolla, mikäli tekijöitä voidaan korvata toisilla, lineaarisesti käyttäytyvillä tekijöillä. Ns. universaalit approksimaattorit kykenevät epälineaariseen regressioon ja niitä käytetään mm. neuroverkoissa. (Runkler, 2016)

Yksinkertaisin regressiomalli on tavallinen suoran yhtälö (Runkler, 2016). Summayhtälöä 17 käytetään useampien ominaisuuksien regression eli yhtälöryhmän sovittamisen yhteydessä.

$$y_{calc} \approx \sum_{r=1}^{f}(a_r * x_r) + b \tag{17}$$

jossa

	y_{calc}	mallinnettavan ominaisuuden ennustettu arvo
	x_r	ominaisuuden x_r arvo
	a_r	kulmakerroin ominaisuudelle x_r
	b	vakio
	f	ominaisuuksien määrä

Esimerkiksi kolmelle muuttujalle ennustava malli näyttäisi siis yhtälön 18 kaltaiselta.

$$y_{calc} = a_1 x_1 + a_2 x_2 + a_3 x_3 + b \tag{18}$$

Yleistettävissä olevassa usean muuttujan mallissa suoran yhtälön parametri a_l tulee arvioida kullekin muuttujalle regressiomallin rakentamiseksi, parametri b on vakio. Tämä tapahtuu minimoimalla mittauspisteiden ja sovitetun mallin välistä virhettä. Mielivaltaisella määrällä muuttujia parametreille a_j matriisissa A saadaan arvot, kun muodostetaan kaikki mittauspisteet k sisältävät matriisit 19 – 21 (Runkler, 2016):

$$V = \begin{pmatrix} y_1 - \bar{y} \\ \dots \\ y_k - \bar{y} \end{pmatrix} \tag{19}$$

$$O = \begin{pmatrix} x_{1,1} - \bar{x}_1 & \dots & x_{1,f} - \bar{x}_f \\ \dots & \dots & \dots \\ x_{k,1} - \bar{x}_1 & \dots & x_{k,f} - \bar{x}_f \end{pmatrix} \tag{20}$$

$$A = \begin{pmatrix} a_1 \\ \dots \\ a_f \end{pmatrix} \tag{21}$$

Näistä matriiseista lasketaan A:n sisältämille parametreille arvot yhtälöllä 22 (Runkler, 2016):

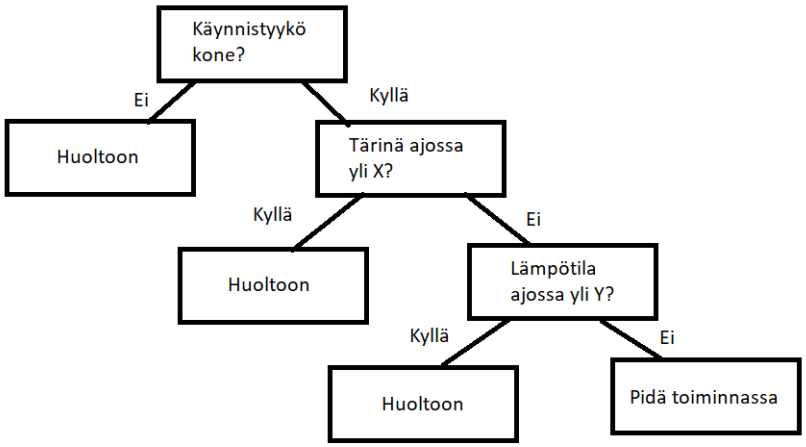
$$(O^T \cdot O)^{-1} \cdot O^T \cdot V = A \tag{22}$$

Lasketun mallin avulla uusien datapisteiden ennustetut arvot voidaan laskea.

3.3.2 Päätöspuu

Päätöspuu (eng. Decision Tree) on luokitteluun ja regressioon käytetty menetelmä, jossa kuljetaan puun juuresta ylöspäin. Päätöspuussa liikutaan ainoastaan yhteen suuntaan,

edeltävään askeleeseen ei koskaan palata. Käytännössä päätöspuu on sarja hierarkisia ”if-then-else” rakenteita, joiden perusteella luokittelu tehdään. Päätöspuita hyödynnetään mm. koneoppimisessa, jossa päätöspuut ovatkin suosittuja menetelmiä yksinkertaisuutensa, helppokäyttöisyytensä ja sekä numeeristen- että kategoristen muuttujien hallinnan vuoksi. Päätöspuilla ei kuitenkaan usein päästä samaan ennustustarkkuuteen kuin monimutkaisemmilla menetelmillä. (Guller, 2015) Yksinkertainen esimerkki kuvitteellisesta koneen huoltamisen tarpeesta kertovasta päätöspuusta on esitetty kuvassa 6.



Kuva 6 Koneen huollontarvetta kuvaava yksinkertainen kuvitteellinen päätöspuu.

Päätöspuiden tarkkuutta voidaan parantaa käyttämällä useampaa puuta. Yksi useamman puun keinoista on Random Forest, jossa jokainen puu edustaa satunnaisesta osasta dataa opittuja asioita. Jokaisella puulla on oma osansa ominaisuuksia, joihin oppiminen ja päätöksenteko perustuvat. Kun ennuste halutaan, Random Forest saa ominaisuuksien arvot syötteenä ja tekee päätöksen yksittäisten puiden antamien tulosten keskiarvona. (Guller, 2015)

Toinen usean puuhun perustuva keino on Gradient-Boosted Trees. Tämä algoritmi rakentaa ”metsän” yksi puu kerrallaan niin, että jokainen puu hyödyntää edellisestä puusta saatua tietoa. Algoritmi on altis ylisovitukselle (over-fitting) riippuen puiden määrästä. Mallin opettaminen voi olla nopeaa tai hidasta, riippuen puiden koosta ja määrästä. (Guller, 2015)

Päätöspuun haarautumisen perusteet

Päätöspuiden haarojen luomiseen voidaan käyttää erilaisia metodeja. Esimerkiksi algoritmeja **Gini Index** ja **Entropy** voidaan käyttää tiedonlisäyksen (Information Gain) maksimointiin.

Entropiaan perustuvalla algoritmilla päätöspuun haarautuminen perustuu entropian vähenemiseen, eli informaation lisääntymiseen haaroissa. Entropia on päätöspuuluokittelun yhteydessä määritelty yhtälöllä 23 (Dougherty, 2013):

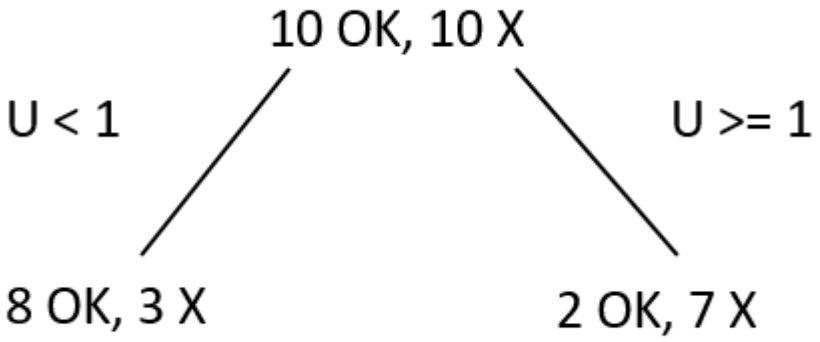
$$S(P) = -\sum_{i=1}^{h} p_i \log_2 p_i \qquad (23)$$

jossa S Entropia välillä [0, 1]

p_i Tiedon i osuus eli todennäköisyys datassa

h Eri tietotyyppien määrä

Yhtälön toiminnan havainnollistamiseksi otetaan esimerkiksi päätöspuu, jossa on 10 toimivaa laitetta (OK) ja 10 huoltoa vaativaa laitetta (X). Kuvitteellisessa tapauksessa nämä jakautuvat ominaisuuden *U* perusteella ensimmäisessä haarassa kuvan 7 mukaisesti:



Kuva 7 Kuvitteellinen päätöspuun ensimmäinen haarauma.

Juurelle ja haaroille voidaan laskea entropiat seuraavasti yhtälön 23 avulla:

$$S(Juuri) = -\frac{10}{20} * \log_2 \frac{10}{20} - \frac{10}{20} * \log_2 \frac{10}{20} = 1$$

$$S(Vasen) = -\frac{8}{11} * \log_2 \frac{8}{11} - \frac{3}{11} * \log_2 \frac{3}{11} = 0{,}834$$

$$S(Oikea) = -\frac{2}{9} * \log_2 \frac{2}{9} - \frac{7}{9} * \log_2 \frac{7}{9} = 0{,}764$$

Entropian muutoksen vertaamista varten haarojen entropiat kerrotaan haaraan menneen datan määrällä jaettuna alkuperäisen datan määrällä, eli vasemmalle haaralle kerroin on 11/20 ja oikealle 9/20. Tämän jälkeen haarojen tulosten summa vähennetään alkuperäisestä entropiasta. Informaation lisäys, eli entropian väheneminen, voidaan laskea yhtälöllä 24 (Dougherty, 2013):

$$(Information)\ Gain = S(Juuri) - \sum_{j=1}^{k} \frac{n_j}{n} S(i) \qquad (24)$$

jossa n_j Datapisteiden määrä haarassa j

k_h Haarojen määrä

Soveltamalla yhtälöä esimerkkitapaukseen saadaan:

$$Information\ Gain = 1 - \left(\frac{11}{20} * 0{,}834 + \frac{9}{20} * 0{,}764\right) = 0{,}191$$

Entropiaa hyödyntävä päätöspuu maksimoi Information Gainin määrän kussakin haarassa, eli iteroi U:n arvoa.

Gini Indexiä hyödyntävän haarautumisen maksimoitava yhtälö on sama kuin Entropian Information Gain, tosin Gini Indexin tapauksessa puhutaan Information Gainin sijaan

pelkästä Gainista. Yhtälössä 24 termien S paikalle kuitenkin sijoitetaan termit G, jotka lasketaan yhtälöllä (Dougherty, 2013):

$$G(p) = 1 - \sum_i^h p_i^2 \quad (25)$$

jossa G Gini Index

Sijoittamalla kuvan 7 mukaiset arvot yhtälöön 25 saadaan:

$$G(Juuri) = 1 - \left(\left(\frac{10}{20}\right)^2 + \left(\frac{10}{20}\right)^2\right) = 0{,}5$$

$$G(Vasen) = 1 - \left(\left(\frac{8}{11}\right)^2 + \left(\frac{3}{11}\right)^2\right) = 0{,}397$$

$$G(Oikea) = 1 - \left(\left(\frac{2}{9}\right)^2 + \left(\frac{7}{9}\right)^2\right) = 0{,}346$$

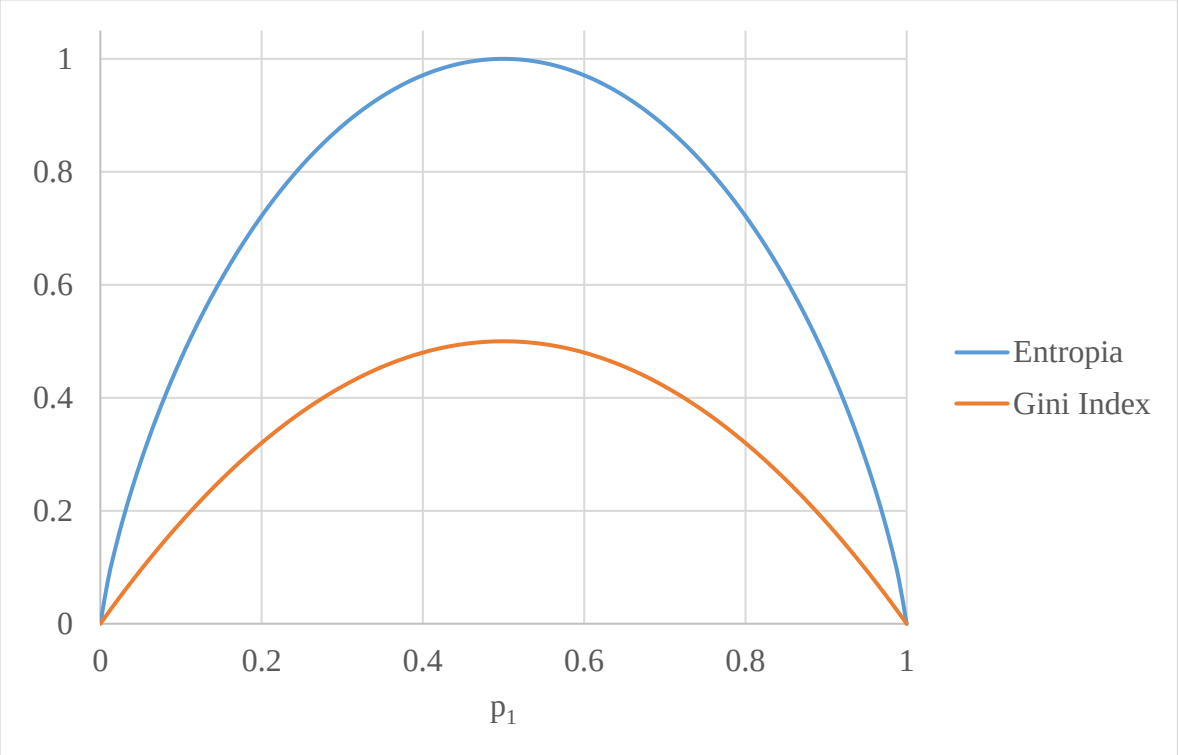
Sijoittamalla Gini Indexin arvot G yhtälöön 24 arvojen S paikalle saadaan:

$$Gain = 0{,}5 - \left(\frac{11}{20} * 0{,}397 + \frac{9}{20} * 0{,}346\right) = 0{,}126$$

Sekä entropian että Gini Indexin tapauksissa nähdään jaottelun tuottavan konkreettista hyötyä yhtälön 24 tuloksen perusteella.

Sekä entropian yhtälö 23 että Gini Indexin yhtälö 25 toimivat alueella p_i = [0, 1] ja binäärisessä tapauksessa saavat maksimiarvonsa, kun $p_1 = p_2 = 0{,}5$. Samalla arvolla maksimoitava yhtälö 24 saa arvon 0, eli huonoimman mahdollisen arvon. Parhaimman arvonsa yhtälöt saavat, kun jompi kumpi todennäköisyyksistä on 1, eli haara sisältää vain

toisen luokan arvoja. Yhtälöt 23 ja 25 käyttäytyvät kuvan 8 mukaisesti, kun yhtälöitä käytetään binääriseen jaotteluun ja p_2 on p_1:n komplementti.

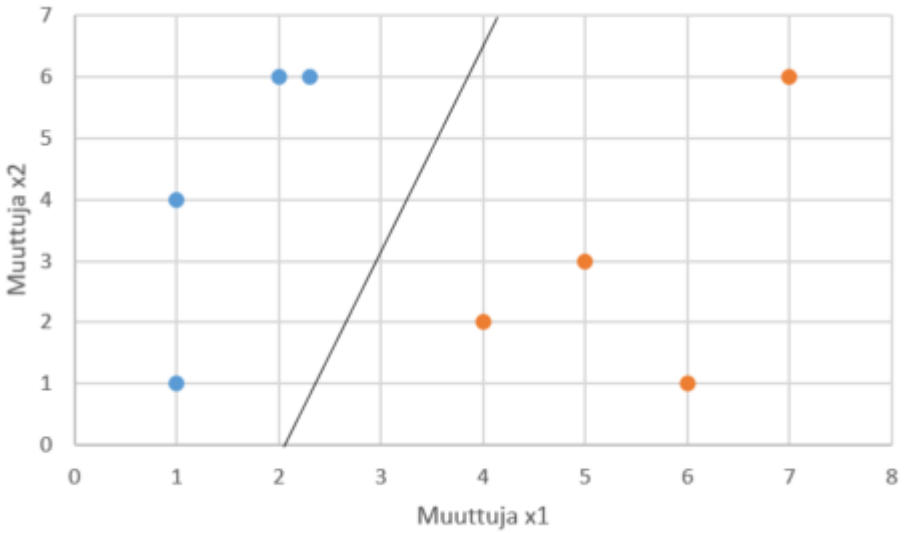


Kuva 8 Entropian ja Gini Indexin käyttäytyminen välillä $p_1 = [0, 1]$.

Hyödynnettäessä Entropiaa ja Gini Indexiä binäärisen päätöspuun muodostamiseen ovat siis suuret todennäköisyyserot p_1:n ja p_2:n välillä jakautumisen jälkeen optimaalisia. Kaikkein paras tapaus olisi kuvan 7 tapauksessa jakautuminen niin, että vasen haara sisältää kaikki yhden luokan arvot ja oikea haara kaikki toisen luokan arvot. Tällöin molempien haarojen entropiat ja Gini Indexit olisivat 0, ja Information Gain ja Gain vastaavasti 1 ja 0,5.

3.3.3 Support Vector Machine

Support Vector Machine (SVM) on luokittelumenetelmä, joka pyrkii erottamaan eri luokat datajoukoista rakentamalla mahdollisimman hyvän moniulotteisen tason luokkien välille (eng. hyperplane). Parhaassa mahdollisessa moniulotteisessa tasossa on eri luokkien lähimpien pisteiden etäisyys tasoon mahdollisimman suuri. Kaksiulotteisessa muuttuja-avaruudessa SVM toimii kuvan 9 mukaisesti. (Guller, 2015)



Kuva 9 Kaksiulotteinen SVM:n optimaalinen taso eli viiva. Viivan kohtisuora etäisyys sitä lähimmistä pisteistä on mahdollisimman suuri molemmin puolin.

Kahdella muuttujalla eri luokat eivät välttämättä erotu toisistaan. Kun siirrytään monimuuttujaisempaan avaruuteen, rakennetaan useampiulotteinen taso samalla periaatteella, eli kolmiulotteisessa avaruudessa kyseessä on normaali taso. Yli kolmiulotteisessa avaruudessa ihmisen hahmotuskyky käytännössä lakkaa toimimasta. Näin voidaan saada erottumaan luokat, jotka eivät ilmene vähempiulotteisessa avaruudessa. Ihanteellisen moniulotteisen tason rakentaminen on laskennallisesti sitä raskaampaa, mitä enemmän ulottuvuuksia ja datapisteitä on. (Guller, 2015)

3.3.4 Naiivi Bayes

Naiivi Bayes on puhtaasti todennäköisyyksiin perustuva luokittelija-algoritmi, joka yksinkertaisuudestaan huolimatta on tehokas erityisesti moniulotteisten datajoukkojen käsittelyssä. Algoritmin idea on laskea todennäköisyys, jolla tapahtuma A on tapahtunut, kun tapahtuman B tiedetään tapahtuneen. Matemaattisesti tämä voidaan kuvata yhtälön 26 mukaisesti. (Guller, 2015)

$$P(A|B) = \frac{P(B|A)*P(A)}{P(B)} \quad (26)$$

jossa
- *P(A|B)* Todennäköisyys, että A on tapahtunut, kun B:n tiedetään tapahtuneen
- *P(B|A)* Todennäköisyys, että B on tapahtunut, kun A:n tiedetään tapahtuneen
- *P(A)* Todennäköisyys, että A on tapahtunut
- *P(B)* Todennäköisyys, että B on tapahtunut, toimii normalisointiterminä

Yhtälön 26 avulla eri parametreille saatujen todennäköisyyksien avulla voidaan rakentaa luokittelija, joka sijoittaa mitatun pisteen todennäköisimpään luokkaan. Kyseisessä luokittelussa kaikki ominaisuudet oletetaan toisistaan riippumattomiksi, mikä tekee menetelmästä ”naiivin”. (Gandhi, 2018) Alla oleva esimerkki esittelee Naiivin Bayesilaisen luokittelijan toiminnan. Kuvitteelliset alkuarvot luokittelulle on esitetty taulukossa 9. Todellisuudessa dataa tarvitaan huomattavasti enemmän tarkan luokittelutuloksen saamiseksi.

Taulukko 9 Kuvitteelliset referenssiarvot Naiiville Bayes -algoritmille.

Koneen tila, *y*	Lämpötila, x_1	värähtelyn määrä, x_2	Virran tarve, x_3
Vaatii huoltoa	Korkea	Suuri	Suuri
Ei vaadi huoltoa	Matala	Matala	Suuri
Ei vaadi huoltoa	Normaali	Normaali	Matala
Ei vaadi huoltoa	Normaali	Matala	Normaali
Vaatii huoltoa	Matala	Suuri	Normaali
Ei vaadi huoltoa	Matala	Suuri	Matala
Vaatii huoltoa	Matala	Matala	Matala

Luokkia *y* on siis kaksi, vaatii huoltoa (y_1) ja ei vaadi huoltoa (y_2). Muuttujia *x* on kolme ja niillä on kullakin kolme diskreettiä arvoa. Koneen huollon vaatiminen tai kunnossa oleminen voidaan kuvata kolmella muuttujalla yhtälöllä (27)

$$P(y_i|x_1, x_2, x_3) = \frac{P(y_i)P(x_1|y_i)P(x_2|y_i)P(x_3|y_i)}{P(y_1)P(x_1|y_1)P(x_2|y_1)P(x_3|y_1)+P(y_2)P(x_1|y_2)P(x_2|y_2)P(x_3|y_2)} \quad (27)$$

Valitaan muuttujille sattumanvaraiset arvot, $y = y_1$, x_1 = Matala, x_2 = Suuri ja x_3 = Suuri. Taulukosta pystytään tulkitsemaan todennäköisyyksien arvot. Esimerkit kolmesta todennäköisyyden arvosta on esitetty alla.

Todennäköisyys, että kone vaatii huoltoa:

$P(y = y_1) = 3/7$

Todennäköisyys, että lämpötila on matala, kun koneen tiedetään vaativan huoltoa:

$P(x_1 = Matala \mid y = y_1) = 2/3$

Todennäköisyys, että lämpötila on matala:

$P(x_1 = Matala) = 4/7$

Kun kaikki arvot on tulkittu vastaavasti taulukosta, sijoitetaan ne yhtälöön (27) esimerkin vuoksi sekä tapauksessa $y = y_1$ että $y = y_2$:

$$P(y = y_1|x_1 = Matala, x_2 = Suuri, x_3 = Suuri) = \frac{\frac{3}{7} * \frac{2}{3} * \frac{2}{3} * \frac{1}{3}}{\frac{3}{7} * \frac{2}{3} * \frac{2}{3} * \frac{1}{3} + \frac{4}{7} * \frac{2}{4} * \frac{1}{4} * \frac{1}{4}} \approx 0{,}7805$$

$$P(y = y_2|x_1 = Matala, x_2 = Suuri, x_3 = Suuri) = \frac{\frac{4}{7} * \frac{2}{4} * \frac{1}{4} * \frac{1}{4}}{\frac{3}{7} * \frac{2}{3} * \frac{2}{3} * \frac{1}{3} + \frac{4}{7} * \frac{2}{4} * \frac{1}{4} * \frac{1}{4}} \approx 0{,}2195$$

Kun koneen lämpötila on matala, sen värähtelyn määrä suuri ja sen virran tarve on suuri, kuuluu kone naiivin Bayes -luokittelun mukaan 78,05 % todennäköisyydellä huoltoa vaativien koneiden luokkaan. Kunnossa olevien koneiden luokkaan se kuuluu 21,95 % todennäköisyydellä. Koska todennäköisyys huollon vaatimiselle on suurempi, luokitellaan piste x_1 = Matala, x_2 = Suuri ja x_3 = Suuri huoltoa vaativien luokkaan. Mikäli kyseessä olisi useamman luokan tapaus, laskettaisiin kaikki todennäköisyydet ja luokiteltaisiin piste suurimpaan todennäköisyyteen. Laskuesimerkki on yleistettävissä *n* muuttujalle *y* luokkaan yhtälöllä (28). (Gandhi, 2018)

$$y = argmax_y \, P(y) \prod_{i=1}^{n} \frac{P(x_i|y)}{P(x_i)} \propto argmax_y \, P(y) \prod_{i=1}^{n} P(x_i|y) \quad (28)$$

Huomattavaa on, että jakajana toimiva arvo ei muutu eri luokkien välillä. Tämän vuoksi luokittelua voidaan yksinkertaistaa jättämällä jakaja pois ja laskemalla ainoastaan osoittajan suuruutta, kuten yhtälön 28 oikealla puolella olevassa osassa. (Gandhi, 2018) Naiivia Bayes -luokittelijaa voidaan käyttää jatkuvien muuttujien tapauksessa lokeroimalla arvot.

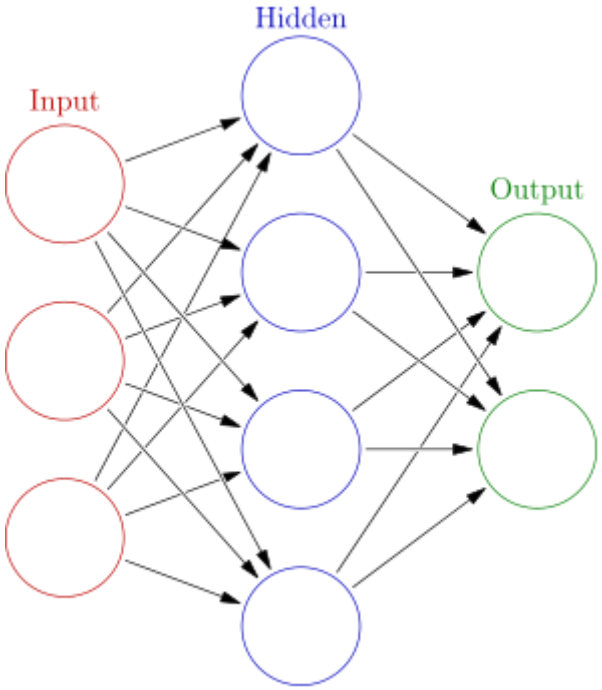
3.3.5 Neuroverkko (Neural Network)

Neuroverkkoalgoritmit ovat aivojen toimintaa matkivia algoritmeja, joita käytetään mm. luokitteluun. Neuroverkon tuottamaa luokittelijaa kutsutaan multi-layer perceptron -luokittelijaksi. (Guller, 2015) Neuroverkkoalgoritmit ovat yleisimpiä Big datan käytössä hyödynnettyjä algoritmeja (Chiroma, et al., 2018).

Yksinkertaisesti kuvattuna neuroverkko koostuu suuresta määrästä yksittäisiä biologian mukaan nimettyjä neuroneita, joista kukin suorittaa tietyn laskuoperaation. Näistä yksiköistä voidaan rakentaa huomattavankin monimutkainen kokonaisuus, jota kutsutaan verkkoarkkitehtuuriksi. Yksiköiden laskentamenetelmiä ja yhteyksiä toisiinsa voidaan muuttaa, mikä puolestaan muuttaa neuroverkon oppimiskykyä ja ennustustarkkuutta. (Mechelucci, 2018)

Verkon syötetasoon syötetään muuttujia, kukin omaan yksikköönsä (node), minkä jälkeen tietoa käsitellään yhdessä tai useammassa piilossa olevassa kerroksessa. Piilossa olevissa

kerroksissa on yksiköitä, jotka käsittelevät tietoa jonkin funktion avulla ennen kuin lähettävät sen eteenpäin. Kukin piilossa oleva yksikkö saa tietoa jokaiselta edeltävän tason yksiköltä ja lähettää sen jokaiselle seuraavan tason yksikölle. Viimeinen kerros on ulostulokerros, jossa syötetty data on jaettu luokkiin. (Guller, 2015) Visuaalinen esimerkki yhden piilossa olevan kerroksen suorasta neuroverkosta on kuvassa 10.



Kuva 10 Yhden piilossa olevan kerroksen neuroverkko visualisoituna. (Glosser.ca, 2013)

Yksinkertaisessa tapauksessa neuroverkon piilossa oleva yksikkö laskee nk. aktivaatiofunktiolle arvon käyttäen kaikkia syötettyjä muuttujia, näiden painoarvoja ja vakiota. Aktivaatiofunktio voi olla esimerkiksi sigmoidifunktio tai hyperbolinen funktio (Mechelucci, 2018). Matemaattisesti tämä voidaan kuvata yhtälöllä 29 (Mechelucci, 2018).

$$\hat{y} = f_a(z) = f_a(w_1 x_1 + \cdots + w_n x_n + b) \quad (29)$$

jossa

f_a	aktivaatiofunktio
z	aktivaatiofunktion syöte
x_n	arvo muuttujalle indeksillä n
w_n	painoarvo muuttujalle x_n
b	vakio tai/eli taipumus
$\hat{y}$	aktivaatiofunktion f arvo syötteellä z

Piilossa olevissa kerroksissa voi olla mikä tahansa määrä yksiköitä. Jos piilossa olevia kerroksia on enemmän kuin yksi, puhutaan syväoppimisalgoritmeista. Kerrosten ja yksiköiden määrän kasvu parantaa esimerkiksi kuvan tunnistuksessa yleensä luokittelutulosta. (Guller, 2015) Suurempi määrä kerroksia ja yksiköitä ei kuitenkaan ole itseisarvoisesti hyvä asia.

Riittävällä määrällä piilossa olevia kerroksia voidaan yksinkertaisilla aktivaatiofunktioilla approksimoida käytännössä mitä vain epälineaarista funktiota. Käytännössä neuroverkon oppiminen tarkoittaa yhtälöstä 29 löytyvien painoarvojen löytämistä. Painoarvoja muutetaan iteroimalla, esimerkiksi optimaalisia arvoja etsivällä gradient descent nimisellä numeerisella menetelmällä. Menetelmässä aloitetaan satunnaisista painoarvoista ja niitä muutetaan nk. kustannusfunktion (cost function) $\nabla J(w_n)$ ja hyperparametrina toimivan oppimisnopeuden γ perusteella. Iterointikierroksia voi olla tarpeen mukaan muutamista miljooniin. (Mechelucci, 2018)

Neuroverkot soveltuvat datan luokitteluun, jos lineaarinen luokittelu ei ole mahdollista. Verkot ovat laskennallisesti raskaampia, kuin yksinkertaisemmat luokittelualgoritmit. (Guller, 2015)

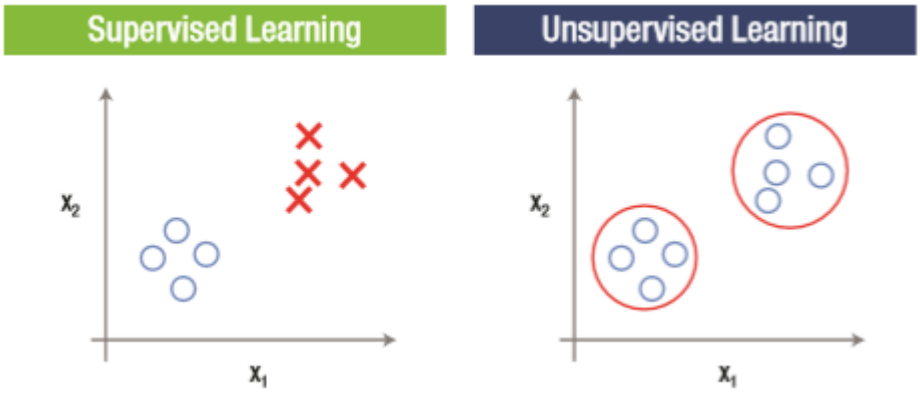
Verkkotyypit voidaan jakaa karkeasti kahteen luokkaan: ”muistillisiin”- (recurrent) ja eteenpäin kytkettyihin (feed-forward) verkkohin (Chiroma, et al., 2018). Eteenpäin kytketyt neuroverkot käsittelevät ainoastaan yhtä datapistettä kerrallaan. Muistilliset verkot

kykenevät hyödyntämään aiemmissa datapisteissä ollutta informaatiota, jos se on tallennettuna muistiin. Käytännössä voidaan ajatella verkon piilossa olevan kerroksen yksiköiden ottavan yhteyttä muistiin, josta ne saavat kontekstia. (Skymind, 2019)

Neuroverkkoja on kappaleessa esitettyjen verkkojen lisäksi monenlaisia. 27 eri verkkotyyppiä on kuvattu pinnallisella tasolla The Asimov Instituten verkkosivuilla. Sivusto sisältää myös lähteet verkkojen alkuperäisiin julkaisuihin. (van Veen, 2016)

3.4 Klusterointi

Klusterointi tarkoittaa löyhästi määriteltynä datan ryhmittelyä valitun algoritmin avulla. Klusterointi on ns. ohjaamaton menetelmä. Ohjaus määritellään klusteroinnin ja luokittelun yhteydessä alkuperäisen datan etukäteen määritellyllä luokkaominaisuudella. Mikäli data sisältää luokkaominaisuuden tai vastaavan mallin opetuksessa hyödynnetyn dataa kuvaavan suureen, on kyse ohjatusta luokittelusta. Klusteroinnin tapauksessa alkuperäinen luokkaominaisuus puuttuu. Klusterointia ei ole määritelty tarkasti, mutta sen pääasiallisena tarkoituksena on datan tutkiminen, hypoteesejen vahvistaminen tai datan esikäsittely jotakin tehtävää varten (Pyne, et al., 2016). Kuva 11 havainnollistaa luokittelun ja klusteroinnin eroa (Haroon, 2017).



Kuva 11 Ohjattu luokittelu (vasemmalla) ja ohjaamaton klusterointi (oikealla). Menetelmien perustavanlaatuinen ero on algoritmin lähtötiedoissa, ja niiden kautta toiminnassa. (Haroon, 2017)

Tietokoneen klusteroinnilla löytämät ryhmät eivät välttämättä vastaa mitään fyysistä luokkaa, kuten epäkunnossa olevia koneita. Klusterointialgoritmejä on monenlaisia. Kullakin on omat vahvuutensa ja heikkoutensa erilaisten datajoukkojen käsittelyssä. Klusterointiin liittyy sivuavia keinoja, joiden avulla määritellään, onko käsiteltävä data yleisesti edes jakautunut eri ryhmiin ja kuinka monta näitä ryhmiä on. (Runkler, 2016)

Vaikka klusteroinnin löytämät ryhmät eivät välttämättä vastaakaan fyysisiä luokkia, käytetään ryhmiä usein luokkien sijasta tilanteissa, joissa luokittaminen ei ole helppoa. Klusteroinnin löytämät ryhmät saattavat olla monimutkaisen muotoisia ja eri klusterointialgoritmit voivat löytää samasta datasta eri ryhmiä. Osa klusterointialgoritmeista vaatii alustuksena ryhmien määrän tai muita parametreja, osa löytää nämä itse. Algoritmit eroavat toisistaan huomattavastikin laskenta-ajassa, mikä näkyy ryhmien erottelukyvyssä. (Runkler, 2016)

Mikään yksittäinen klusterointialgoritmi ei ole ratkaisu kaikkiin ongelmiin. Esimerkiksi Expectation-Maximization ja Fuzzy-C-means -algoritmit tuottavat erittäin hyviä ryhmiä, paitsi jos käsiteltävä data on erittäin moniuloitteista. Ne myös vaativat paljon laskentatehoa, joten niiden käyttö on monessa tilanteessa epä-optimaalista. BIRCH, DENCLUE ja OptiGrid -algoritmit sopivat suurien datamäärien ryhmittelyyn. Näistä kaksi viimeistä pystyvät käsittelemään myös todella moniulotteista dataa. (Fahad, et al., 2014)

Klusterointialgoritmit voidaan jakaa karkeasti neljään eri luokkaan (Fahad, et al., 2014): osiointiin perustuviin algoritmeihin, hierarkiaan perustuviin algoritmeihin, ruudukkoon perustuviin algoritmeihin ja malleihin perustuviin algoritmeihin.

3.4.1 K-Means

K-Means algoritmi on yksi perinteisistä ja laajimmin käytetyistä klusterointialgoritmeista. (Pyne, et al., 2016) Se jakaa datajoukon *k* ryhmään. *k*:n arvo on määritelty käyttäjän toimesta. K-means etsii klusterien keskipisteet iteratiivisesti ja määrittää klusteriin kuuluvat pisteet minimoimalla klusteriin sisältyvien pisteiden etäisyyksien neliösummia. (Guller, 2015) Klustereiden keskipisteiden alkuarvojen etsintään on kehitetty useita eri algoritmeja. Eri aloituspisteet vaikuttavat menetelmän tuloksiin.

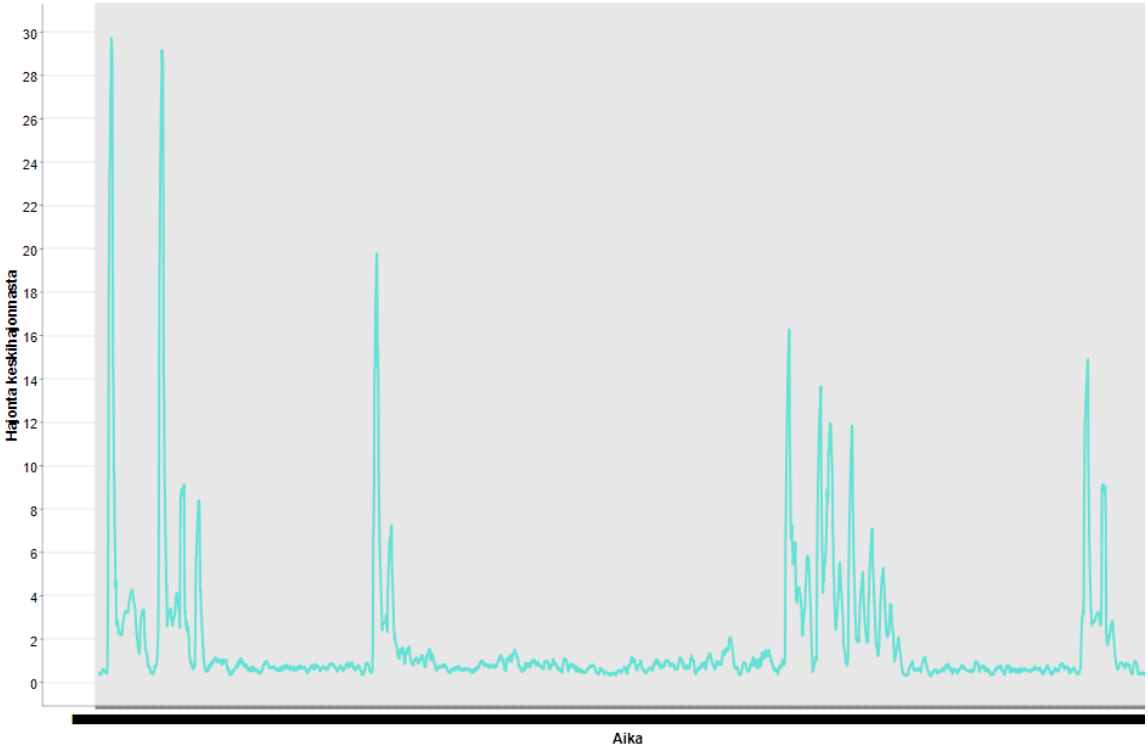
3.5 Poikkeuksien havaitseminen

Poikkeuksien havaitsemisessa pyritään löytämään muusta datasta huomattavasti eroavia pisteitä. Poikkeavan pisteen odotetaan viittaavan poikkeavaan tapahtumaan. Poikkeuksien havaitsemista hyödynnetään merkitsemättömän datan yhteydessä. (Guller, 2015)

Poikkeuksia havaitsemalla voidaan esimerkiksi teollisuudessa löytää viallisia tuotteita tai hajoamassa oleva laite. Aina datapisteen poikkeama ei kuitenkaan kerro mistään konkreettisesta, havainnoitavasta tapahtumasta. Poikkeuksien havaitsemisella on käyttökohteita petoksien havaitsemisesta hakkeroinnin havaitsemiseen. (Guller, 2015)

Poikkeuksien havainnoinnin avulla voidaan tarkkailla yksittäisen laitteen kuntoa. Mallintamalla normaalia toimintaa esimerkiksi autoregressiivisellä mallinnuksella voidaan havaita ongelmatilanteita, kun mallinnus antaa huonoja tuloksia. Mallinnuksessa pyritään arvioimaan tulevan mittauksen tulos edellisten mittausten perusteella. Kun normaaliin toimintaan perustuvan mallin antaman arvion ja todellisen mittauksen ero on suuri, esimerkiksi kaksi tai kolme kertaa yli keskihajonnan, on oletettavissa, että on tapahtunut jotakin poikkeavaa. Yksittäisiä poikkeamia voi tapahtua monista syistä, minkä vuoksi yksittäinen rajan ylittävä arvo ei todennäköisesti ole kiinnostava. Kun tapahtumien määrä tietyllä aikaikkunalla kasvaa, on ongelmatilanne todennäköisempi. Ongelmatilanteiden määrälle aikaikkunalla voidaan asettaa raja, joka tiedottaa kunnossapitoa esimerkiksi sähköpostilla tarkastamaan laitteen. (Silipo, et al., 2018)

Kuvassa 12 on esitetty todellisesta teollisesta datasta kymmenen edellisen minuutin autoregressiolla tehty havainnointi tuotannon poikkeuksista. Datasta on käytetty vain yhtä prosessimuuttujaa, joka on yksi prosessin poikkeuksiin vaikuttavista tekijöistä. Esimerkin informaatio ei ole hyödyllistä, sillä se ei ole ennustavaa, mutta havainnollistaa tehdyn mallin poikkeuksien havainnointikykyä. Hajontojen määrittämiseen on käytetty edellisen 20 hajonnan keskiarvoa.



Kuva 12 Tuotannon poikkeuksien havainnointi teollisessa prosessissa.

3.6 Yhteenveto data-analyysin eri menetelmistä

Taulukossa 10 on esitetty kappaleessa esitellyt menetelmät ja algoritmit tiivistettynä.

Taulukko 10 Esitellyt data-analyysimenetelmät tiivistettynä.

Menetelmäluokka	**Tarkoitus**	**Esimerkkimenetelmät ja -algoritmit**
Korrelaatio	Kertoo muuttujien välisistä riippuvuussuhteista. Auttaa löytämään merkittävimmät muuttujat.	Kovarianssi, χ^2-testi, ANOVA
Luokittelu	Sijoittaa datapisteitä tiettyihin ennalta määriteltyihin luokkiin. Voi hyödyntää koneoppimista tai tilastollisia menetelmiä.	Päätöspuut, Support Vector Machine, Naiivi Bayes, Neuroverkot, (Regressio)
Klusterointi	Sijoittaa datapisteitä ennalta määrittämättömiin ryhmiin. Hyödyntää koneoppimista.	K-means, BIRCH, DENCLUE, Optigrid

3.7 Datan esikäsittely

Dataa on hyödyllistä esikäsitellä eri tavoin käyttökohteesta riippuen, sillä monien laskenta-algoritmien laskenta-aika skaalautuu datapisteiden- ja muuttujien määrän neliön tai kuution mukaan (L'Heureux, et al., 2017). Esimerkiksi ehkäisevässä kunnossapidossa antureista saatava värähtely- ja lämpötiladata voi olla sekunnin välein mitattua. Sovelluskohde ei kuitenkaan suurimmassa osassa tapauksista juurikaan hyödy näin tiheästä mittausvälistä, sillä kunnossapidolliset ongelmat kehittyvät yleisesti verrattain pitkällä aikavälillä. Näin ollen datasta voidaan laskea esimerkiksi minuutin tai tunnin keskiarvot, mikä vähentää huomattavasti koneoppimisessa käytettävän datan määrää ja siten laskenta-aikaa. (Microsoft [2], 2015) Toinen vaihtoehto on valikoida vain osa datapisteistä.

Vähäinen ongelmadatan määrä on ongelmallista koneoppimisen menetelmissä, kuten luokittelussa. Yksi ratkaisu ongelmaan on keinotekoinen ongelmadatan monistaminen **SMOTE**:lla (Synthetic Minority Over-sampling Technique) tai vastaavalla menetelmällä. SMOTE:ssa ylimääräistä dataa luodaan lisäämällä uusia ongelmapisteitä olemassa olevien ongelmapisteiden väliin. (Prusty, et al., 2017) SMOTE lisää monesti luokittelijan löytöastetta, mutta myös false positive -luokittelujen määrä kasvaa.

Esikäsittelyssä tulee ottaa huomioon myös datan laatu ja poikkeamat. Puuttuvat arvot käsitellään aina jollakin menetelmällä. Ohjelmisto tai algoritmi voi esimerkiksi ohittaa kokonaan tyhjiä arvoja sisältävät rivit tai sijoittaa tyhjään arvoon esimerkiksi sarakkeen keskiarvon tai edellisen rivin arvon. Käsittelymenetelmä tulee valita sovelluskohteen mukaan. Vakioina pysyvät sarakkeet tai todella pienen hajonnan omaavat sarakkeet on myös hyödyllistä poistaa, sillä ne eivät tuo mallin rakennukseen uutta informaatiota.

PCA (Principal Component Analysis), eli pääkomponenttianalyysi, jakaa datan keskenään mahdollisimman vähän korreloiviin osiin ja täten vähentää huomattavasti muuttujien määrää. Menetelmässä määritetään kovarianssien tai muiden riippuvuutta mittaavien suureiden avulla keskenään lineaarisesti korreloivat ominaisuudet ja yhdistetään nämä. Ennalta määrättyjen parametrien perusteella jäljelle jää N komponenttia, jotka ovat mahdollisimman riippumattomia toisistaan. Komponenttien määrä voidaan eri ohjelmistoissa joko määrittää erikseen tai säilytetyn informaation määrän avulla. PCA on herkkä datan suuruusluokille, joten ennen menetelmän käyttöä on tärkeää poistaa voimakkaasti poikkeavat datapisteet ja normalisoida ominaisuudet. PCA:n tuottamat

komponentit on mahdollista muuntaa takaisin alkuperäisiksi, mutta rekonstruktio ei ole täydellinen. (Vidal, et al., 2016)

4. Kaupalliset datankäsittelyohjelmistot ja teknologiat

Eri kaupallisten ohjelmistojen tarjoamat työkalut ja hinnat eroavat huomattavasti. Tässä kappaleessa keskitytään muutaman eri datankäsittelyohjelmistojen tarjoamiin ominaisuuksiin esittelymateriaalin ja kirjallisuuden perusteella. Markkinoilta löytyy kappaleessa esitettyjen ohjelmistojen lisäksi kymmeniä, ellei satoja muita ratkaisuja.

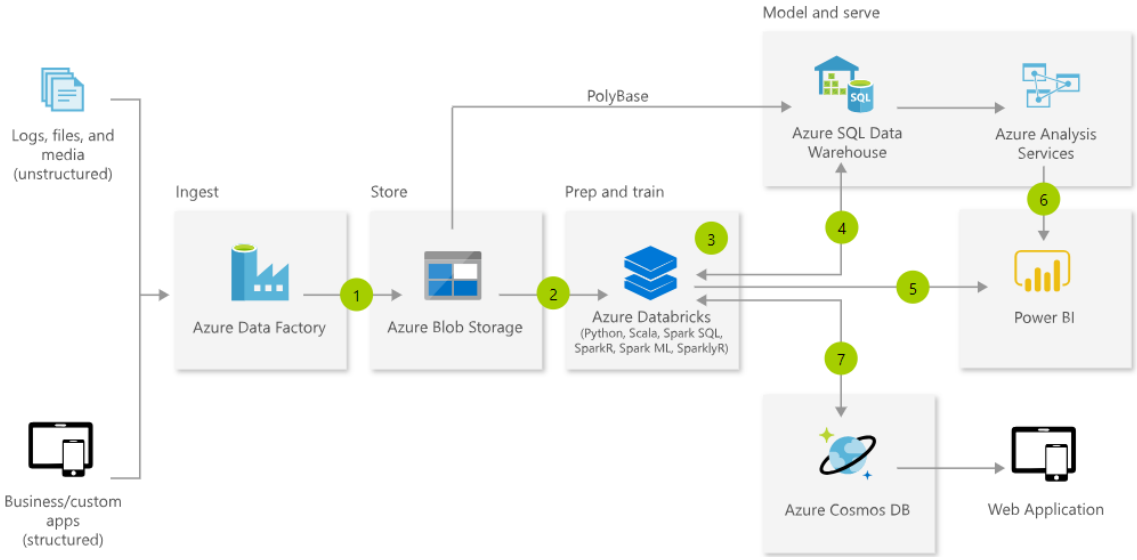
4.1 Microsoft Azure Machine Learning Studio

Azure on Microsoftin toteuttama monikäyttöinen pilvipalvelu. Azure sallii käyttäjän omien sovellusten kehittämisen ja suorittamisen pilvessä, ja sen pilvipalveluita voi käyttää niin mobiililaitteilla kuin perinteisillä tietokoneilla. Azuren palveluiden mainostetaan olevan tilankäyttöön ja laskentatehon tarpeeseen mukautuvia, minkä vuoksi Azure on kustannustehokas vaihtoehto. Azure tarjoaa myös keinoäly- ja koneoppimispalveluita, joita voidaan hyödyntää ennustavassa kunnossapidossa ja juurisyyanalyysissa. (Microsoft [1], 2019)

Azure tarjoaa esimerkkitapauksen ja pohjan älykkäälle ennakoivalle kunnossapidolle. Esimerkissä käsitelty data on suuruusluokaltaan melko pientä. Esimerkki neuvoo käyttäjää ennakoivan kunnossapidon ratkaisun rakentamisessa. Ratkaisu sisältää laitevikojen ennakoinnin, juurisyyanalyysin, vian luokittelun, vikojen havaitsemisen, ja kunnossapitosuosituksen. Se keskittyy ennen kaikkea ennustamaan vian synnyn, jotta kunnossapitotoimenpiteet voidaan suorittaa ennen vian ilmenemistä. Esimerkkitapauksessa tarkastellaan simuloitua lentokoneen moottoria. Esimerkkiratkaisu käyttää kolmea mallia: regressiota (Decision Forest Regression, Neural Network Regression, Poisson Regression, Boosted Decision Tree Regression) jäljellä olevan käyttöiän selvittämiseen, binääristä luokittelua (Two-Class Logistic Regression, Two-Class Decision Forest, Two-Class Boosted Decision tree, Two Class Neural Network) hajoamisen todennäköisyyden tietyllä

aikavälillä määrittämiseen, ja moniluokkaluokittelua (Multiclass Neural Network ja Multiclass Logistic Regression) eri aikaikkunoiden aikana tapahtuvan hajoamisen todennäköisyyksiin. (Microsoft [2], 2015)

Microsoft myös mainostaa Azuren soveltuvan Big datan käyttöön. Kuvassa 13 esitetty tarvittava arkkitehtuuri sallii datan yhdistämisen missä tahansa kokoluokassa ja koneoppimismallin valjastamisen datan tulkintaan. (Microsoft [3], 2019)



Kuva 13 Microsoftin esittämä Big datan hyödyntämiseen tarvittava arkkitehtuuri Azuressa. (Microsoft [3], 2019)

Yllä kuvatun arkkitehtuurin ylläpito maksaisi Microsoftin laskurin (Microsoft [7], 2019) oletusarvoilla noin 7 000 €/kk. Oletusarvot eivät kuitenkaan varmasti ole suurimmassa osassa käyttökohteita kovin lähellä todellisuutta, sillä laskennan ja datan määrä vaikuttavat huomattavasti hintaan. Hinta olettaa ympärivuorokautisen ja -kuukautisen laskentakäytön.

Azure sopii esimerkin perusteella ainakin yksittäisten laitteiden kunnon tarkkailuun, ja tähän tarkoitukseen implementoinnin ei pitäisi olla erityisen vaikeaa. Myös suuremman datamäärän hyödyntäminen on mahdollista, mutta vaatii varmasti huomattavasti enemmän osaamista ja aikaa.

Ennustavaan analyysiin löytyy ohjaavaa materiaalia. (Barga, et al., 2015) Azuren sisällä voi suorittaa esimerkiksi Hadoopia (luku 4.4) ja Sparkia (luku 4.5) mm. Azure Databricksissa (Microsoft [4], 2019) ja Azure HDInsightissa.

Microsoft tarjoaa Azuren koneoppimispalveluihin nopeaan algoritmin valintaan tarkoitetun dokumentin, joka on esitetty liitteessä II. (Microsoft [6], 2017) Periaatteen tasolla valinnan ei pitäisi olla sidottu Azureen, joten dokumentti lienee hyödyllinen myös muiden ohjelmistojen kanssa toimiessa.

Yksi Azuren Big data -palveluita hyödyntävistä yrityksistä on Daimler AG. Daimler ylistää Azuren tietoturvamahdollisuuksia ja hyödyntää analyysipalveluita ympäri maailman. Yksi käyttökohde löytyy autojen huolloista. Historiadataa hyödyntämällä ja auton lokitietojen perusteella kyetään löytämään ongelmien juurisyitä, mikä nopeuttaa huoltotoimenpiteitä säästäen aikaa ja rahaa. Daimler myös kokee pilvipalvelun nopeuttaneen järjestelmän käyttöönottoa ja vähentäneen kustannuksia verrattuna omaan palvelinklusteriin. (Microsoft [8], 2019)

4.2 GE Intelligent Platforms

APM Reliability on osa GE Digitalin (General Electricin ohjelmistoyhtiö) Predix APM (Predix Asset Performance Management) ohjelmistopakkausta. Ohjelmisto tarjoaa mahdollisuuden havaita ja diagnosoida laitteistojen ongelmia ennen niiden tapahtumista. Se lisää laitteiston luotettavuutta samalla optimoiden ylläpitokustannuksia. Se yhdistää dataa erilaisista lähteistä ja analysoimalla muuntaa sen tiedoksi, jonka avulla voidaan tehdä toimenpiteitä. Se toimii kaikilla teollisuuden aloilla ja koko tehtaan alueella. Ohjelmisto pystyy käsittelemään sekä jäsenneltyä että jäsentelemätöntä dataa. (GE Digital, 2019)

Luvussa 2.2.1 esitetty käyttötapaus Covingtonin paperitehtaalla hyödynsi GE Intelligent Platfroms Csense troubleshooteria juurisyyanalyysin toteuttamiseen valtavilla datamäärillä. Käyttötapauksessa ohjelmistolla saatiin selkeitä hyötyjä tehtaan toimintaan.

4.3 TIBCO Spotfire

TIBCO Spotfire mainostaa itseään helppo- ja nopeakäyttöisenä ohjelmistona. Se kykenee älykkäisiin visualisointiratkaisuihin, datan käsittelyyn, ennakoivaan analyysiin, sekä online-datan käyttöön analyysissä. TIBCO mainostaa Spotfiren olevan suoraan käännettynä ”ainoa alusta joka antaa liiketoiminnan käyttäjille intuitiivisen ja helppokäyttöisen rajapinnan kaikkeen Big datan analysointiin liittyvään teknologiaan ilman tiedonkäsittelyllistä- tai tietoteknistä asiantuntemusta”. Spotfire kykenee hyödyntämään mm. Hadoopia datan lähteenä. (TIBCO Software Inc., 2019) Hadoop on myöhemmin luvussa esitelty avoimen lähdekoodin ohjelmisto suurten tietomäärien käsittelyyn.

Spotfire pystyy sekä historia- että online-datan käsittelyyn. Online-datan perusteella ohjelmisto pystyy mm. antamaan hälytyksiä poikkeustilanteissa. Ohjelmisto kykenee käsittelemään miljardeja datarivejä. (TIBCO Software Inc. [2], 2019)

Gartner totesi vuoden 2018 analyysissaan TIBCOn ohjelmistojen olevan parhaita koneoppimis- ja datankäsittelyohjelmistoja tuotannon parantamiseen. Raportissa analysoitiin 17:ta eri myyjää. (TIBCO Software Inc. [2], 2019) Toisessa Gartnerin analyysissa kehutaan TIBCO:n ohjelmistojen käytön yksinkertaisuutta hyvällä ominaisuusvalikoimalla (Gartner Inc., 2019).

Spotfiren pilvipalveluiden hinnoittelu vaihtelee käyttäjän käytössä olevien ominaisuuksien mukaan. Tarjolla on ilmainen 30 päivän koekäyttömahdollisuus, sekä kolme erilaista maksullista lisenssiä. Korkeimman tason ”Analyst”-lisenssi maksaa 125 dollaria kuukaudessa tai 1250 dollaria vuodessa. ”Business Author” -lisenssi maksaa 65 dollaria kuukaudessa tai 650 dollaria vuodessa. ”Consumer”-lisenssi maksaa 25 dollaria kuukaudessa tai 250 dollaria vuodessa. Lisenssien lisäksi TIBCO Software myy tallennustilaa pilvipalvelussaan hintaan 250 dollaria/vuosi/25 GB (TIBCO Software Inc. [3], 2019)

Mm. Mercedes AMG Petronas Formula 1 -tiimi käyttää Spotfirea analyyseissaan. Tiimi on kyennyt tekemään parannuksia suureen datamäärän perustuvan tiedon pohjalta. Analysoimalla eri vaihteistoasetuksia Spotfirellä, tiimi pystyi nopeuttamaan vaihteiston toimintaa noin 50 millisekuntilla per kierros, mikä on merkittävä muutos erityisesti aika-ajoissa. Myös todella lyhyiden kaasutuksen katkojen juurisyy löydettiin, ja pystyttiin

korjaamaan. F1-tiimi analysoi gigatavuja harjoituksista saatua dataa Spotfiren avulla muutaman tunnin aikana ennen kunkin kilpailun alkua. (TIBCO Software Inc. [4], 2018)

Spotfire sisältää muutaman regressioon, luokitteluun, klusterointiin ja korrelaatioon liittyvän algoritmin normaalissa asennuksessa. Nämä ominaisuudet ovat helppokäyttöisiä, eivätkä vaadi käyttäjältä minkäänlaista ohjelmoinnin osaamista. Spotfireen voi lisätä käytännössä minkä tahansa data-analyysialgoritmin hyödyntämällä sisäänrakennettua TIBCO Enterprise Runtime for R (TERR) -moottoria. TERR kykenee hyödyntämään R-kielisiä kirjastoja, mikä tekee menetelmien sisällyttämisestä Spotfireen verrattain yksinkertaista. Ratkaisut eivät kuitenkaan ole täysin triviaaleja, vaan vähintään pieni määrä ohjelmointia vaaditaan kirjastofunktioiden hyödyntämiseen. Spotfire sisältää kirjastojen lataamiseen tarkoitetun toiminnon, joka hyödyntää Comprehensive R Archive Networkia (CRAN). CRAN sisälsi vuoden 2019 maaliskuussa lähes 14 000 algoritmikirjastoa. Kaikki CRAN:n tarjoamat kirjastot ovat ilmaisia.

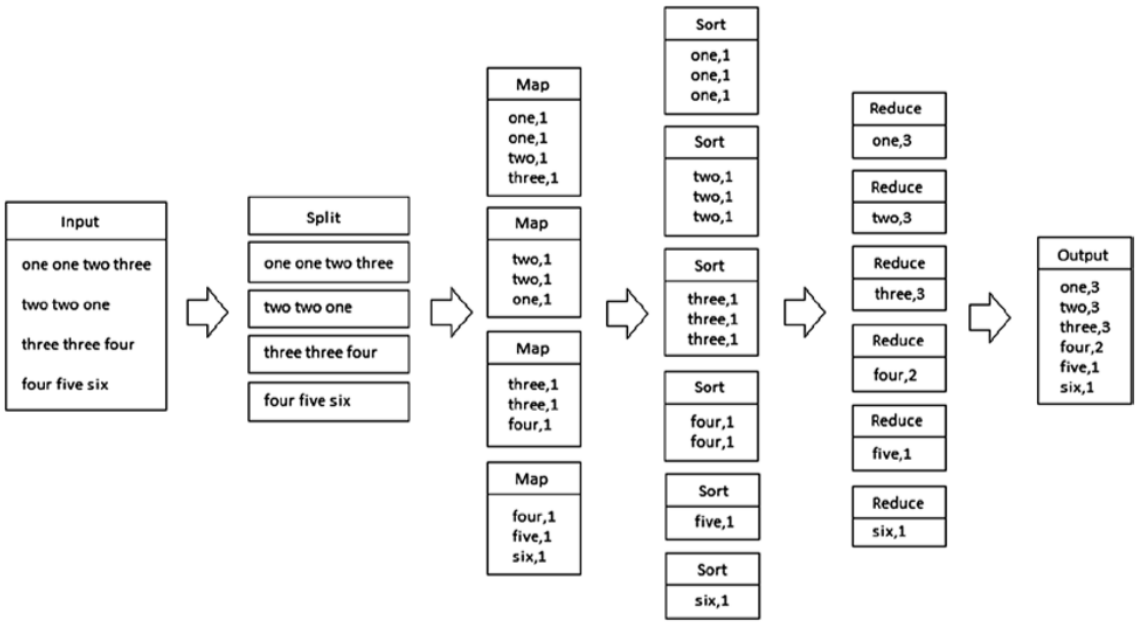
4.4 Apache Hadoop

Hadoop on Apachen tuottama Javapohjainen avoimen lähdekoodin ratkaisu Big datan hajautettuun käsittelyyn. Hadoop toimii pohjana monissa Big data -ratkaisuissa. Hadoop on rakennettu ottamaan huomioon mahdolliset laitteiston kaatumisen aiheuttamat ongelmat datan säilönnässä ja laskennassa hajauttamalla tallennus laskentaklusterin eri laitteille. Se on suunniteltu skaalautumaan erittäin suuriin datajoukkoihin, tiedostomääriin (jopa miljoonia) ja laskentayksikkömääriin, sekä toimimaan tavallisella kuluttajalaitteistolla. Hadoop on ottanut vaikutteita mm. Googlen MapReducesta. (Frampton, 2015) Hadoopille on suuri määrä sen toimintaa tukevia ratkaisuja, kuten Pig-ohjelmointikieli, joka on suunniteltu helpottamaan Map Reducen rakentamista.

Hadoop käyttää omaa **MapReduce** -algoritmiansa jakamaan ja käsittelemään (map) suuren datamäärään usealle eri laskentayksikölle/laskentaklusterille, minkä jälkeen käsitelty data tiivistetään (reduce) lopputulokseksi saatavaan muotoon. (Frampton, 2015) Esimerkki sanojen luokittelusta MapReduce -algoritmilla on esitetty kuvassa 14 (Frampton, 2015). Teollisuuden IIoT lähteistä saatavaa dataa voidaan periaatteessa käsitellä Map Reducen

avulla. MapReduce ei kuitenkaan kykene online-datan käyttöön, ja monimutkaisten algoritmien toteuttaminen on verrattain vaikeaa. (Guller, 2015).

MapReducen ydinidea on keventää suuren datamäärän kanssa työskentelevän työtaakkaa. MapReduce hoitaa laskennassa mukana olevien yksiköiden työnjaon, ongelmatilanteet ja kommunikaation. Näin ollen käyttäjän ei tarvitse huolehtia hajautuksen järjestämisestä, vaan hän voi keskittyä itse datan käsittelyyn. (Guller, 2015)



Kuva 14 Map Reduce -algoritmin toiminta sanojen lajittelussa ja laskennassa. (Frampton, 2015)

Hadoopin tiedostojenkäsittely tapahtuu mm. **HDFS**:n (Hadoop Distributed File System) avulla. Se jakaa tiedostoja usealle palvelimelle, ja siten sietää hyvin ongelmia ja on laajennettavissa. HDFS on tarkoitettu suurten datamäärien tallennukseen. Tiedostot jaetaan palasiin, joiden oletuskoko on 128 MB. Jokainen palanen löytyy useammalta systeemin tietokoneelta, joten yksittäisen yksikön kaatuminen ei ole suuri ongelma. HDFS sisältää kahdenlaisia yksiköitä: metatiedot sisältäviä nimiyksiköitä itse tiedot sisältäviä datayksiköitä. Yksiköt kommunikoivat ajoittain keskenään tietojen välittämiseksi ja toiminnan varmistamiseksi. Laskentaa varten haluttu tieto paikannetaan nimiyksikön

antamien metatietojen avulla, minkä jälkeen se pyydetään suoraan tiedot sisältävältä datayksiköltä. (Guller, 2015)

Hadoopin tutkiminen on hyvä lähtökohta Big dataa hyödyntävien ohjelmistojen ymmärtämiseen, sillä avoin lähdekoodi, suuri materiaalimäärä ja aktiivinen käyttäjäyhteisö antavat tutkimukseen hyvät edellytykset. Ratkaisun rakentaminen alusta loppuun vaatii kuitenkin huomattavasti aikaa ja tietotaitoa, mikä rajoittaa käytännön käyttömahdollisuuksia. Seuraavana esiteltävän Sparkin myös katsotaan yleisesti olevan modernimpi ja helppokäyttöisempi ratkaisu.

4.5 Apache Spark

Sparkia pidetään Hadoopin MapReducen seuraajana. Se toimii samassa roolissa Big datan prosessoinnissa, mutta omaa useita etuja verrattuna MapReduceen. Spark tarjoaa käyttäjälle ohjelmointirajapinnan, jonka avulla voi hyödyntää palvelinklusterin laskentatehoa suurien datamäärien käsittelyssä. Se on MapReducen tavoin skaalautuva ja sietää ongelmia, minkä lisäksi Spark on yksinkertaisempi käyttää. Se on kymmeniä, ellei satoja kertoja tehokkaampi kuin MapReduce, riippuen prosessoitavan datamäärän koosta. Nopeusero johtuu suureksi osaksi RAM-muistin tehokkaamasta hyödyntämisestä. Toisin kuin MapReduce, Spark kykenee myös online-datan analysointiin. (Guller, 2015)

Uusien yksiköiden lisääminen Sparkiin on helppoa, mikä takaa taloudellisen laskenta-kapasiteetin lisäämisen. Lisääminen ei vaadi ylimääräistä ohjelmointityötä, vaan sovellus tarjoaa vaihtoehdon automaattisesti. Spark myös käsittelee yksittäisten laskentayksiköiden kaatumisen esimerkiksi kovalevyn ongelmista johtuen ilman, että käyttäjän tarvitsee tehdä mitään. (Guller, 2015)

Sparkin asennuksen mukana tulee kirjastoja, joiden avulla Spark pystyy historiadatan prosessointiin, online-datan prosessointiin, koneoppimiseen, kuvaajien tekoon ja interaktiiviseen analyysiin. Mm. Sparkin MLlib -kirjasto sisältää useita koneoppimisalgoritmeja. Kirjasto siältää algoritmeja regressioon, luokitteluun, klusterointiin, muuttujien vähentämiseen, muuttujien muuttamiseen, yleisten kuvioiden selvittämiseen (frequent pattern mining) ja suositteluun liittyen. Näin ollen yhdellä laskentaklusterilla pystyy suorittamaan kaikki edellämainitut toimenpiteet, mikä helpottaa

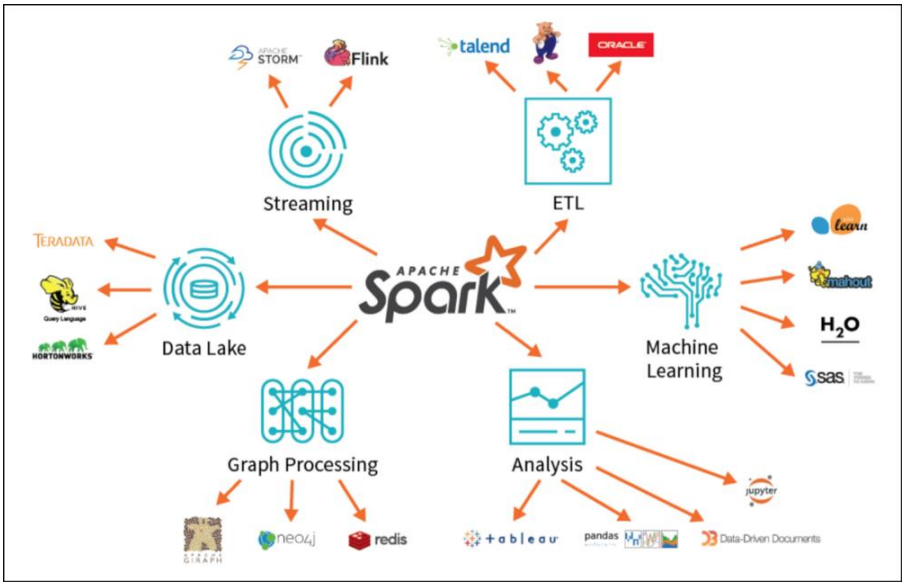
toimintaa ja vähentää koodin ja datan toistoa. Helppouden ja monikäyttöisyyden vuoksi perinteisesti MapReducelle tarkoitetut kirjastot ja sovellukset ovatkin nykyään saatavilla myös Sparkille. (Guller, 2015)

Spark on optimaalinen sovelluksille, joissa datamäärät ovat valtavia ja nopeus on tärkeää. Tämä tarkoittaa esimerkiksi iteratiivisia sovelluksia, joissa samaa dataa voidaan käsitellä kymmeniä tai satoja kertoja. Myös datan interaktiivinen analysointi on Sparkin vahvuuksia. (Guller, 2015)

Spark kykenee hyödyntämään mitä tahansa tiedonlähdettä, jota Hadoop kykenee käyttämään. Spark pystyy tämän lisäksi hyödyntämään monia muita datanlähteitä, sillä yhteisö on luonut yhteyksiä eri tietojärjestelmiin. (Guller, 2015)

Avoimen lähdekoodin ohjelmistona Spark on ilmainen, ja sille on tarjolla paljon opetusmateriaalia. Datan analysointi kuitenkin vaatii todennäköisesti huomattavasti enemmän osaamista kuin valmiiden ratkaisujen käyttö, sillä vaikka menetelmäkirjastoja on tarjolla, täytyy relevantit osat koota itse. Kokoaminen ja järjestelmän pystyttäminen vaativat todennäköisesti huomattavan määrän ohjelmointityötä. Lisäksi paikallisesti Sparkia suorittaessa tulee huolehtia fyysisen järjestelmän ylläpidosta ja hankinnasta, ellei ohjelmistoa pyöritä pilvipalvelussa.

Kuva 15 esittää laajasti Sparkin konseptin ja mahdollisia hyödyntämisen kohteita.



Kuva 15 Sparkin konsepti ja toiminnallisuuksia. (Rodriguez, 2019)

4.6 IBM Watson

IBM Watson IoT kykenee ennakoivaan analyysiin, jota voidaan hyödyntää ehkäisevässä kunnossapidossa. Esimerkiksi KONE käyttää Watsonia hissiensä kunnon analysoimiseen ja täten ehkäisevään kunnossapitoon. Koneen vastuulla on yli miljoonan hissin ja liukuportaiston kunnossapito. Watson ilmoittaa kunnossapitohenkilökunnalle, kun mahdollinen ongelma havaitaan. (IBM, 2017)

Watson sisältää monia toimintaa helpottavia työkaluja, kuten koneoppimismallien ja neuroverkkojen rakentamisen graafisesti. Analyysiyhtiö Forrester sijoitti IBM:n koneoppimis- ja ennakoivan analyysin palvelut johtavaan asemaan analyysissaan. (IBM [3], 2019)

Watsonin koneoppimistyökalut ovat skaalautuvia ja yksinkertaisia. Watson sallii mallien jatkuvan kouluttamisen ja niiden tuomisen muista data-analyysityökaluista. Watson käyttää Apache Sparkia laskennan ja koneoppimisen hajautukseen. Watsonilla voi myös automatisoida koneoppimisen hyperparametrien optimoinnin ja ominaisuuksien valitsemisen. (IBM [2], 2019)

IBM:n Predictive Maintenance and Optimization -paketti tarjoaa valmiita ennustavia malleja kunnossapitoon. Mallien avulla voi päätellä, mitkä laitteet ovat yli- tai alihuollettuja, ja missä laitteissa huolto on suoritettu optimaalisesti. Mallien avulla voi löytää suurimmat laitteiston toimintaan vaikuttavat tekijät ja käyttää tätä tietoa hyödyksi kunnossapitostrategiassa. (IBM [5], 2017)

Watsonin koneoppimisjärjestelmän hinnoittelu on esitelty IBM:n internetsivustolla. Lite-, Standard- ja professional -versioille on kullekin oma hinnoittelunsa, joka määritellään pääasiassa käytön mukaan. (IBM [4], 2019) IBM Watson Knowledge Studiosta on saatavilla rajoitettu ilmaisversio, jolla voi kokeilla viittä omaa koneoppimisprojektia. IBM:n erilliselle IoT Predictive Maintenance and Optimization -ratkaisulle ei ole ilmoitettu tarkkaa hintaa. Hinta neuvotellaan erikseen riippuen valittuun pakettiin kuuluvista komponenteista.

Kuva 16 esittää Watsonin analytiikkatoimintojen konseptin. Watson on laaja tuote, johon kuuluu konseptin tietojen lisäksi runsaasti eri toimintoja.

Kuva 16 IBM Watsonin analytiikan konsepti. (Ironside Group, 2019)

4.7 Umetrics Easy Analytics

Easy Analytics on Sartorius Stedim Biotechin omistaman Umetricsin analysointiohjelmisto. Umetrics on kehittänyt Easy Analyticsin lisäksi mm. kokeiden suunnitteluun tarkoitetun MODDEn. Easy Analyticsin mainostetaan sopivan monille aloille lääkekehityksestä paperiteollisuuteen. Easy Analyticsin korostetaan olevan peruskuluttajalle sopiva, eikä käyttö vaadi tiedonkäsittelytieteiden osaamista. (Sartorius Stedim Biotech, 2019)

Easy Analytics sisältää poikkeuksien havainnointiin, tilastolliseen prosessinhallintaan ja systeemin suorituskyvyn analysointiin liittyviä työkaluja. Datan käsittely Easy Analyticsilla on yksinkertaista ja ohjelmiston tuottamat tulokset ovat visuaalisesti esitettyjä ja luotettavia. Toiminnallisuudeltaan yksinkertaisempana Easy Analyticsin hinnoittelu on verrattain matalampi kuin monimutkaisempien ohjelmistojen. (Sartorius Stedim Biotech, 2019) Easy Analyticsin eri versioiden hinnat perustuvat ominaisuusvalikoimaan. Saatavilla on 32- ja 64-bittiset versiot ”työkalupakilla” ja ilman. Ilman työkalupakkia toimitettavat versiot maksavat 99 dollaria, työkalupakilla toimitettavat versiot 198 dollaria. (Sartorius Stedim Biotech [2], 2019)

4.8 KNIME

KNIME (Konstantz Information Miner) on KNIME AG:n ilmainen avoimen lähdekoodin ratkaisu tietojenkäsittelyyn. Pilvipalvelimille tiimityöskentelyyn tarkoitetun palvelinversion yhteydessä maksetaan lisensointimaksuja, mutta työpöytäkäyttö on ilmaista. KNIME kykenee tilastollisten ja koneoppimismallien laskemiseen, ja ratkaisujen rakentaminen onnistuu graafisesti ilman koodausta. Esimerkiksi Python-, Java- ja R-koodia on kuitenkin mahdollista sisällyttää sovelluksiin. KNIME sisältää lukuisia koneoppimismalleja liittyen luokitteluun ja klusterointiin.

Toisin kuin monissa muissa ilmaisissa ohjelmistoissa, KNIMEssä ei ole rajoitteita esimerkiksi datan rivimäärän tai prosessointiytimien suhteen. Yhtiössä myös painotetaan nimenomaan ohjelmistokehitystä. Yhtiö harjoittaa konsultointia vain harvoin. (KNIME AG, 2019)

Tietojen esikäsittely on KNIMEn avulla yksinkertaista, eikä dataa tarvitse esikäsitellä välttämättä lainkaan esim. Excelin avulla. Esikäsittelyn ominaisuuksiin kuuluvat mm. liikaa (määriteltävä määrä) tyhjiä datarivejä sisältävävien muuttujien poisto, vakiona tai lähes vakiona (määritettävissä varianssilla) pysyvien ominaisuuksien poisto, viiveen asettaminen muuttujille, muuttujien arvojen automaattinen lokerointi, SMOTE ja arvojen normalisointi.

KNIMEn voi yhdistää myös moniin muihin sovelluksiin, kuten Apache Sparkiin Big datan käsittelyssä ja Spotfireen datan visualisoinnissa. KNIMEn vahvuus on mallien rakentamisen helppous, mutta visualisoinnissa KNIME ei ole kaikkein vahvimmillaan. Erityisesti muodostettujen kuvien interaktiivisuus ja suodattimien määrä on heikompi kuin esimerkiksi Spotfiressa. Visualisointikyvyn puutetta voidaan paikata hyödyntämällä tukityökalua, kuten Spotfirea tulosten visualisointiin. Myös lisäosia asentamalla KNIMEen on mahdollista saada lisää visualisointiominaisuuksia. Graafisen mallin rakennuksen ja esikäsittelykykyjen ansiosta KNIMEllä tehdyn tiedonkäsittelyn eri vaiheet ovat erittäin helposti dokumentoitavissa, sillä kaikki välitulokset ja -vaiheet ovat saatavilla. Monien mallien testaus rinnan onnistuu myös KNIMEn graafisen mallin rakennuksen ansiosta helposti.

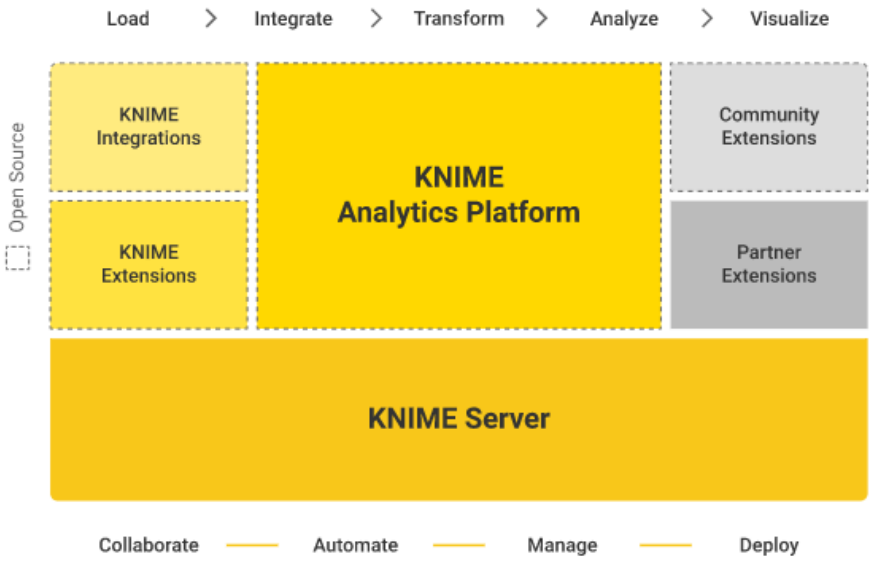
KNIMEn käyttöoikeussopimus on GNU General Public License V3, joka takaa ohjelmiston avoimuuden. Käytännössä tämä tarkoittaa samojen oikeuksien siirtymistä kehittäjältä asiakkaalle. Mikäli yritys kehittää ohjelmistoa eteenpäin, se voi myydä ratkaisun, mutta

asiakkaalle pitää toimittaa lähdekoodi samalla GNU GPL V3 lisenssillä. Näin ollen asiakas voi itse myydä ratkaisua eteenpäin ja jatkokehittää sitä, kunhan lisenssi kulkee mukana. (KNIME AG [3], 2019) Ohjelmiston hankkiminen eri lisenssillä on mahdollista ottamalla yhteys yritykseen. Vaikka ohjelmiston jakelijan tuleekin toimittaa lähdekoodi pyydettäessä, onnistuu ohjelmiston asentaminen yksinkertaisesti asennusohjelman avulla.

KNIME ei lähetä yritykselle mitään dataa, ellei käyttäjä siihen erikseen suostu. Suostumuksella kerätty data ei sisällä ohjelmistossa käsiteltävää dataa, asetuksia tai henkilökohtaista dataa, vaan pääasiassa metadataa eri yksiköiden käyttömääristä, käynnistymisistä, käyttöajasta jne. (KNIME AG [4], 2019)

Pilvipalvelun hinta perustuu ominaisuusvalikoimaan. Palvelu on tarjolla kolmessa eri kokoluokassa, Small, Medium ja Large. KNIME Server Small sisältää pienimmän ominaisuus- ja oikeusvalikoiman, ja hinta-arvio viiden hengen käyttäjätiimille on 7 500 € vuodessa. KNIME Server Mediumin hinnaksi arvioidaan samalla käyttäjämäärällä 25 000 € vuodessa. Parhaimman ominaisuus- ja oikeusvalikoiman sisältävän KNIME Server Largen hinnaksi arvioidaan samalla käyttäjämäärällä 45 500 € vuodessa. Tarkat ominaisuus- ja oikeustiedot ovat saatavilla osoitteessa https://www.knime.com/knime-software/knime-server-pricing. (KNIME AG [2], 2018)

Kuva 17 tiivistää KNIMEn konseptin. Konseptiin kuuluu avoimen lähdekoodin osiot ja lisäosat, sekä palvelinversioon liittyvä varsinainen mallien toimeenpano.



Kuva 17 KNIMEn konsepti tiivistettynä. (KNIME AG [5], 2018)

4.9 Ohjelmiston perusedellytykset ja harkinnan kohteet

Big dataa käsittelevällä ohjelmistokokonaisuudella on oltava monia perusedellytyksiä riippuen käyttökohteesta. Ohjelmiston tulee kyetä keräämään ja lajittelemaan dataa sekä siirtämään se käsittelyprosessiin. Dataa tulee kyetä säilyttämään useilla palvelimilla varmuuskopioineen, ja laitteiston kaatuminen ei saa aiheuttaa suuria ongelmia säilytyksessä tai laskennassa. (Frampton, 2015)

Ohjelmiston tulee myös tarjota käyttäjälleen riittävän hyvät työkalut ja tukipalvelut, sillä Big datan käyttö on aina käyttökohdetta varten räätälöityä. Online-datan tapauksessa ohjelmiston tulee myös kyetä tuottamaan käyttäjälle hyödyllisiä raportteja, sekä hyödyntämään reaaliaikaista dataa. Ohjelmiston tulisi myös hyödyntää laskentakapasiteettia datan tallennuskohteessa ylimääräisen siirtelyn välttämiseksi. (Frampton, 2015)

Parhaimman järjestelmän valintaan on useita ratkaisevia tekijöitä. Näistä kolme voidaan esitellä seuraavien kysymysten muodossa (Singh & Reddy, 2014):

1. Kuinka pian tuloksia tarvitaan?
2. Kuinka paljon dataa on prosessoitavana?
3. Vaatiiko mallien rakentaminen yhden vai useamman iteraatiokierroksen?

Puhtaasti teoreettisten kysymysten lisäksi on tärkeää käsitellä myös käytännön toteutusta. Tämä voidaan esitellä neljäntenä kysymyksenä:

4. Kuinka helppoa on ottaa järjestelmä käyttöön?

Kun näihin kysymyksiin on löydetty vastaukset, voidaan siirtyä tarkkailemaan systeemitason vaatimuksia. Näitä ovat mm. (Singh & Reddy, 2014):

1. Tarvitseeko datankäsittelykapasiteettia kasvattaa myöhemmin?
2. Onko datan liikkumisvauhti kriittinen sovellukselle?
3. Pitääkö sovelluksen pystyä käsittelemään laitteiston mahdollisia ongelmia?

Datankäsittelykapasiteettia voidaan kasvattaa kahdella tavalla, joko päivittämällä yksittäistä laskentayksikköä tai hajauttamalla laskenta usealle yksikölle. (Singh & Reddy, 2014)

Yksittäisen laskentayksikön päivittäminen on ohjelmistoille helpompaa, ja laitteiston hallinta on huomattavasti yksinkertaisempaa. Päivittäminen on kuitenkin rajallista, ja

tehokkaampien komponenttien hinta kasvaa nopeammin kuin suoritusteho. Yksittäistä laskentayksikköä päivittäessä tulee myös ottaa huomioon tulevaisuuden tarpeet, joten laskentakapasiteettia on yleensä ylimäärin. (Singh & Reddy, 2014)

Hajauttamalla laskentaa voidaan kapasiteetin päivityksiä tehdä tarpeen mukaan. Yksittäiset päivitykset antavat paremman hinta-laskentakapasiteetti suhteen ja laskentatehon kasvatusmahdollisuudet ovat periaatteessa rajattomat. Hajautettu järjestelmä on kuitenkin vaativampi ohjelmistolle ja ylläpidolle, sillä ohjelmiston pitää hallita datan hajautus ja rinnakkaisen laskennan aiheuttamat hankaluudet. Hajautettuun laskentaan itse kykeviä ohjelmistoja on rajallisesti. (Singh & Reddy, 2014) Monet yritykset ovat kuitenkin tehneet omat ratkaisunsa esimerkiksi Spark-yhteensopiviksi.

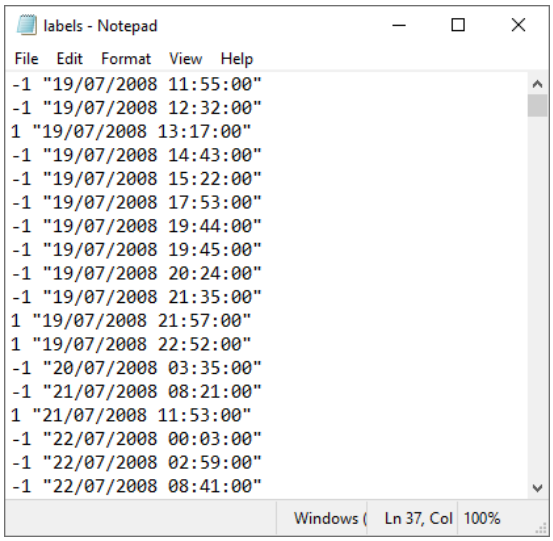
Käytännössä hajautuksen voi järjestää suorittamalla laskennan pilvipalvelussa. Pilvipalvelun käyttö voi kuitenkin olla ongelmallista tietoturvan kannalta, mikäli käsitelty data on arkaluontoista. Mm. käyttöoikeudet, fyysinen sijainti ja palveluntarjoajan oikeudet voivat olla ongelmallisia tietoturvan kannalta. Tietoturvaongelmia voi rajata salaamalla palveluun ladatun datan, mutta tämä puolestaan aiheuttaa ongelmia datan käytössä, esimerkiksi hauissa. Datan siirto pilveen vie myös aikaa ja resursseja. (Agarwal & Biswas, 2017)

4.10 Testi- ja tutkimusdata ohjelmistoille

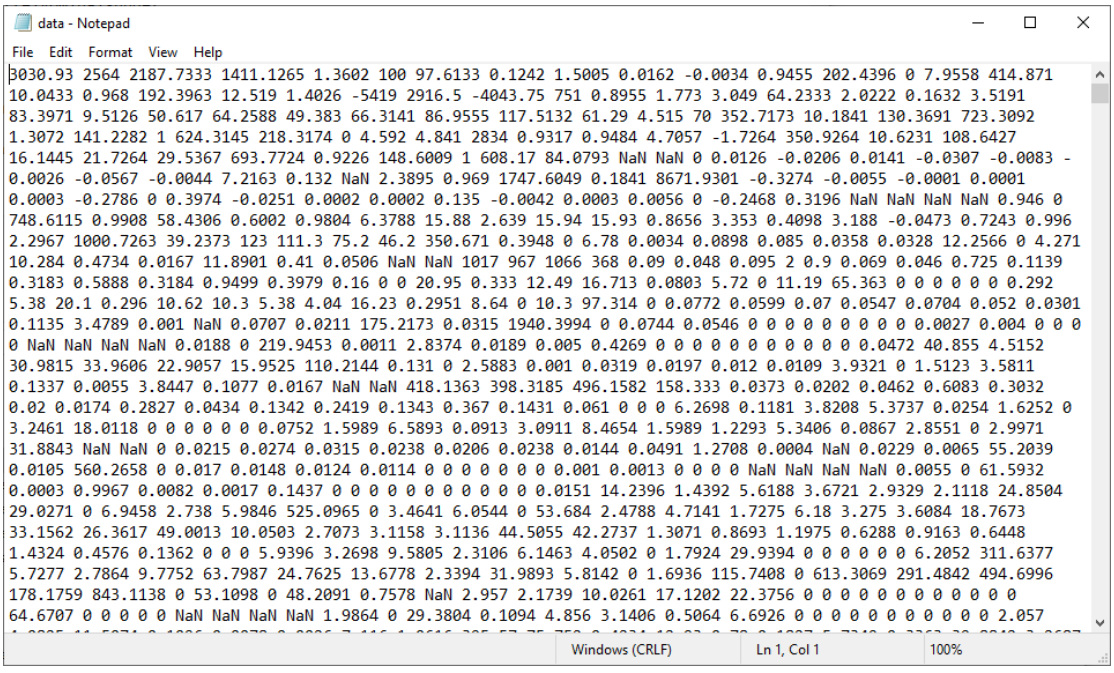
Ohjelmistojen ominaisuuksien- ja käytön helppouden testaamiseen käytetään SECOMin 19.11.2008 luovuttamaa puolijohteiden valmistusprosessista kerättyä avointa dataa (saatavilla osoitteessa http://archive.ics.uci.edu/ml/machine-learning-databases/secom/). Kansio sisältää kolme tiedostoa: yleistä tietoa datajoukosta kertovan names -tiedoston, aikaleimat ja ongelmatilannetiedot sisältävän labels -tiedoston, ja itse datan sisältävän data -tiedoston. Dataa analysoimalla ominaisuuksien toimintaa voidaan arvioida analyyttisesti, käytön helppoutta subjektiivisesti. (McCann & Johnston, 2008)

Datajoukko sisältää dataa yhteensä 591:stä muuttujasta 1 567:llä eri aikaleimalla, eli tiedot ovat 1597×591 kokoisessa matriisissa. Data on todellisesta valmistusprosessista, mutta SECOM ei anna selitteitä muuttujille. Dataa ei ole esikäsitelty, joten joukossa on tyhjiä arvoja, mittausvirheitä ja hyödyttömiä muuttujia. Datajoukko on tarkoitettu juurisyyanalyysiin ja luokitteluun. Ongelmatilanteita datajoukossa on yhteensä 104, ja nämä

on kuvattu labels -tiedostossa arvolla 1. Normaali toiminta on kuvattu arvolla -1. (McCann & Johnston, 2008) Kuvissa 18 ja 19 on kuvankaappaukset Notepadilla avatuista tiedostoista labels ja data.

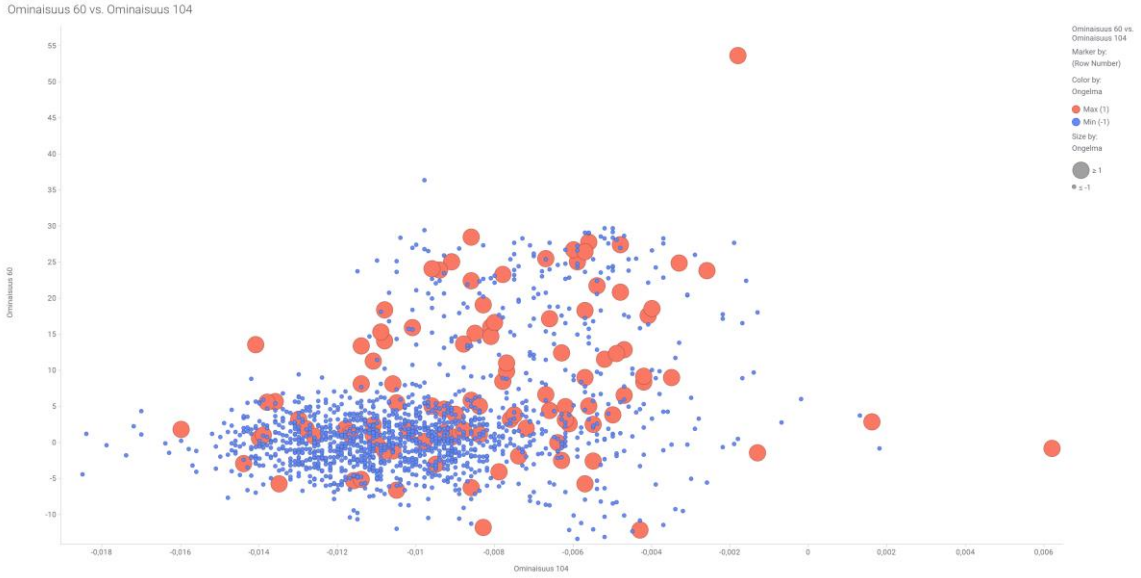


Kuva 18 labels -tiedoston sisältämät luokittelut ja aikaleimat.



Kuva 19 Esimerkkidataa tiedostosta data. Tieto ei järjesty Notepadissa järkevästi. Kaikki kuvassa näkyvä tieto kuuluu todellisuudessa labels –tiedoston ensimmäiselle aikaleimalle.

Kuva 20 sisältää SECOM-datan visualisoituna Spotfiressa. Kuvassa data on esitetty kahden suurimman ANOVA:n F-arvon omaavan ominaisuuden, eli ominaisuuksien 60 ja 104 avulla (Salem, et al., 2018).



Kuva 20 SECOM –data esitettynä ominaisuuksien 60 (pystyakseli) ja 104 (vaaka-akseli) suhteen. Suuremmat, punaiset merkit kuvaavat poikkeusdataa, pienemmät, siniset merkit kuvaavat ongelmatonta dataa.

Avoimen SECOM-datan lisäksi ohjelmistoilla käsitellään Empowerin yhdestä aikaisemmasta sellu- ja paperiteollisuuteen liittyvästä projektista saatua dataa. Tämän datan lähtötiedot ja muuttujien nimet ovat selvillä, mutta dataa käsitellään työssä anonyymisti. Kuvitteellisena esimerkkinä Sekoittimen nopeus [RPM] -muuttuja voisi esiintyä työssä nimeellä Ominaisuus 1. Myös kunnossapidon värähtelydataa käsitellään ohjelmistoilla.

5. Tyypillinen prosessiautomaatio ja kunnossapito

5.1 Kunnossapiton merkitys ja lajit

Tehokas kunnossapito on yksi toimivan tuotantoprosessin tärkeimmistä elementeistä. Se on yksi teollisuuslaitosten suurimmista kustannuseristä, kattaen jopa 30 %

käyttökustannuksista. Laitteiston elinjakson kustannuksista 60 – 75 % kuuluu kunnossapidolle. Tehokkaalla kunnossapidolla saadaan hyötyjä käyttövarmuuteen, energiankäyttöön sekä laitteiston elinjakson kuluihin. (O'Donovan, et al., 2015)

Erityisesti käyttövarmuuden parantaminen vaikuttaa yrityksen tulokseen tuotannon lisäämisen johdosta. Esimerkiksi sellu- ja paperitehtaiden seisokkiajan arvioitiin maksavan vuonna 2012 yli 20 000 dollaria tunnilta (AmericanAirFilter, 2012). Kunnossapidon kustannustehokkuus ja tuotantolaitteiston käyttövarmuus ovat suorassa yhteydessä kunnossapitotoiminnan suunnitteluun. Perinteisesti käytetyt mallit kunnossapidon päätöksen teon tukemisessa tuottavat toisinaan huonoja tai ristiriitaisia tuloksia. (Ruschel, et al., 2017)

Käytännössä kunnossapidon tehokkuutta laitehuollon suhteen voidaan mitata kahdella mittarilla: odottamattomien laiterikkojen yleisyydellä eli vikataajuudella ja hyödyntämättömän elinajan suuruudella. Hyödyntämättömällä elinajalla tarkoitetaan aikaa, jonka laite olisi kyennyt toimimaan normaalisti tehdyn ennakkohuollon ja vikaantumisen välillä. Mittarit ovat toisistaan riippuvaisia. Jos koko elinjakso hyödynnetään, laitteita vikaantuu enemmän. Vikaantumiset aiheuttavat yleisesti huomattavasti enemmän kustannuksia, kuin hyödyntämätön laitteen elinjakso. Kunnossapidon optimoinnissa nämä kaksi mittaria on saatava tasapainoon, jossa vikaantumisia tapahtuu mahdollisimman vähän, ja mahdollisimman paljon laitteiston komponenttien elinjaksoista hyödynnetään. Optimoinnin minimoitava yhtälö 30 on esitetty alla. (Susto, et al., 2015)

$$C_{tot} = f_{oh}c_{oh} + t_h c_h \tag{30}$$

jossa

C_{tot}	Kunnossapidon kustannukset laitteelle
f_{oh}	Laitteen vikataajuus
c_{oh}	Laitteen hajoamisen kustannukset
t_h	Laitteen huollettavan tai vaihdettavan komponentin hyödyntämätön elinaika
c_h	Hyödyntämättömän elinajan kustannukset

Kunnossapitolajeja ja -strategioita on erilaisia, ehkäisevästä ja laitteiston kuntoon perustuvasta strategiasta häiriötilanteisiin reagoivaan strategiaan. Suurien datamäärien käyttö soveltuu tukemaan ehkäiseviä kunnossapitolajeja. (O'Donovan, et al., 2015) Alla on esitelty lyhyesti neljä tyypillistä kunnossapitostrategiaa.

Reaktiivinen strategia

Reaktiivisessa strategiassa huoltotoimia suoritetaan vain tuotannon pysäyttävien ongelmatilanteiden yhteydessä, eli kyseessä on häiriökorjauksiin perustuva strategia. Reaktiivinen strategia ei vaadi aikataulutusta tai suunnittelua, mikä vähentää mm. työnjohdollisia tarpeita. Laitteisto on kuitenkin reaktiivisella strategialla toimivissa laitoksissa yleensä lyhytikäisempää, ja energiankulutus ja rejektin määrä prosessissa ovat mahdollisesti suurempia pienempien huoltamattomien vikojen vuoksi. Reaktiivinen strategia soveltuu käytännössä laitteistoille, jotka eivät ole prosessin kannalta keskeisiä tai joiden ennakkohuoltotoimenpiteet maksaisivat enemmän kuin mahdollinen vikaantuminen. (O'Donovan, et al., 2015)

Korjaava strategia

Korjaavassa strategiassa keskitytään pienempien vikojen korjaukseen niiden syntyessä suurempien vikojen välttämiseksi. Korjaavassa strategiassa laitteiston kunnon taso on paremmin tiedossa, ja useammat pienet huollot voivat pidentää laitteiston elinjaksoa. Laitteiston kunnon tarkkailuun tarvittava laitteisto ja työ maksavat enemmän kuin reaktiivisessa strategiassa. Korjaavan strategian käyttö soveltuu käytännössä kaikille, paitsi kaikkein tärkeimmille laitteille, joille on varattava enemmän resursseja. (O'Donovan, et al., 2015) (Bumblauskas, et al., 2017)

Ehkäisevä strategia

Ehkäisevässä strategiassa laitteistoa huolletaan joko tietyin aikavälein, jolloin kyseessä on jaksotettu kunnossapito, tai tietyn merkkinä toimivan suureen tarkkailun seurauksena (esim. elementin lämmitysaika), jolloin kyseessä on kuntoon perustuva kunnossapito (PSK Standardisointiyhdistys ry, 2011). Ehkäisevä kunnossapito nostaa käyttövarmuuden tasoa normaalitoiminnassa. Se kuitenkin kuluttaa mahdollisesti tarpeellista enemmän varaosia ja nostaa kunnossapitokustannuksia verrattuna kahteen edelliseen strategiaan. Strategiaa sovelletaan yleensä kriittiselle laitteistolle, jonka on toimittava aina. (O'Donovan, et al., 2015) (Bumblauskas, et al., 2017)

Ennustava strategia

Ennustava strategia kunnossapidossa kuuluu ehkäisevään, kuntoon perustuvaan kunnossapitolajiin. Ennustavassa strategiassa nimensä mukaisesti ennustetaan ongelmia ja laitteiston tai osien jäljellä olevaa elinaikaa. Onnistuneiden arvioiden avulla kyetään vähentämään ylimääräisiä ja ennenaikaisia huoltotoimenpiteitä. Ennustukset perustuvat kuitenkin dataan ja todennäköisyyksiin, minkä vuoksi myös ylimääräisiä huoltotoimenpiteitä saatetaan tehdä. Ennustavalla strategialla voidaan optimoida kunnossapitokustannuksia monissa laitteistoissa, joiden lyhyet käyttökatkot eivät ole kriittisiä prosessin kannalta. (O'Donovan, et al., 2015) Ennustavassa strategiassa on perinteisesti käytetty mm. Markovin päätösprosessiin perustuvia- (Markov Decision Process) ja muita analyyttisia malleja. (Bumblauskas, et al., 2017) Huollon tarpeen ennustamiseen käytetään dataa, jota saadaan esimerkiksi värähtely-, ääni-, öljy- ja lämpötilamittauksista. Värähtely on datanlähteistä yleisin. Tämän datan perusteella voidaan toimia suoraan, tai dataa voidaan käsitellä jonkinlaisella mallilla laitteiston kunnon määrittämiseksi. (Barga, et al., 2015)

5.2 Prosessiautomaatio ja anturointi

Prosessiautomaation perusperiaatteena on hallita laitteiden (prosessiteollisuudessa yleisimmin venttiilien, moottoreiden ja muiden toimilaitteiden) toimintaa tuotteen valmistusprosessissa. Hallintaan tarvitaan eri lähteistä saatavaa tietoa. Esimerkiksi jatkuvatoimisen reaktorin viipymäaikaa eli reaktioaikaa voidaan hallita sekä syöttöventtiilin että ulostuloventtiilin avulla. Lisäksi pumppujen ja reaktorin pinnan korkeuden aiheuttamat paineet vaikuttavat virtausnopeuksiin ja täten viipymäaikaan.

Anturit tuottavat jatkuvasti mittausdataa hallintatietokoneelle. Anturit lähettävät yleensä aikaleiman ja mitatun suureen tai suureiden arvot tietyin väliajoin. Tietokone käsittelee online-dataa ja tekee tarvittavat toimenpiteet, esimerkiksi antaa hälytyksen lämpötilan noustessa liian korkealle, tai avaa venttiilin nestetason noustessa liian korkealle. Data arkistoidaan käsittelyn jälkeen tietyin aikavälein yleensä joko relaatiotietokantaan tai yksinkertaisena lokitiedostona. Arkistoidusta datasta voidaan koostaa raportteja käyttäjälle. (O'Donovan, et al., 2015)

Mitatun suureen arvo esitetään suureen mittauksen mukaisessa muodossa. Muoto on yleensä totuusarvo, kokonaisluku tai reaaliluku. Mikäli mittauksen muoto on totuusarvo tai kokonaisluku, esitetään usein jonkin laitteen tilaa. Totuusarvotapauksessa tila on yleisesti päällä tai pois päältä (TRUE/FALSE). Mikäli relevantteja tiloja on useampia, kukin voidaan määritellä tietyksi kokonaisluvuksi. Reaalilukujen avulla käsitellään jatkuvia arvoja. Aikaleima voidaan esittää joko absoluuttisena aikana tai kuluneena aikana. (Love, 2007)

Varsinaiset mittaukset tehdään aina jonkinlaisesta signaalista. Signaalit voivat olla analogisia, diskreettejä tai pulsseja. (Love, 2007)

Analogisten signaalien arvot vaihtelevat jatkuvana syötteenä tietyllä alueella. Esimerkiksi nesteen sähkönjohtavuus voi toimia analogisena signaalina tietyn komponentin, esimerkiksi hydroksidi-ionin, pitoisuudesta. Signaali muutetaan yhtälöllä pitoisuudeksi ja pitoisuus ilmaistaan reealilukuna. (Love, 2007)

Diskreeteillä signaaleilla mitataan tilaa, kuten venttiilin asentoa. Asennolla voi olla kaksi tai useampia arvoja, joista yksinkertaisimmat ovat auki ja kiinni. (Love, 2007)

Pulssisignaaleilla tarkoitetaan jatkuvasti toistuvia diskreettejä signaaleja. Esimerkiksi pyörivien laitteiden tarkkailussa jokaisella kierroksella toistuu tietty määrä signaaleja, joiden perusteella pyörimisnopeus voidaan laskea. (Love, 2007)

Prosessiautomaation avulla kyetään hallitsemaan suoraan vain osaa prosessin muuttujista. Muuttujat voivat olla hallittuja (Controlled), villejä (Wild), kelluvia (Floating) tai määritettyjä (Determined). Hallitut muuttujat tarkoittavat prosessin toiminnan ohjauksessa käytettyjä muuttujia. Tämä voi tarkoittaa esimerkiksi tietyn putken virtausnopeutta, jota säädetään venttiilin avulla. Villit muuttujat riippuvat suoraan ylävirrassa tapahtuneista asioista, kuten kahden erikseen hallitun putken virtausnopeuksista. Kelluvilla muuttujilla tarkoitetaan muuttujia, joiden arvojen sallitaan muuttua tietyllä välillä. Määritetyt muuttujat ovat tulos aine- ja energiataseiden yhtäaikaisesta täyttymisestä. (Love, 2007)

KÄYTÄNNÖN OSA

6. Ohjelmistojen testaaminen: kriteerit ja metodiikka

Tässä kappaleessa pisteytetään järjestelmällisesti neljä kiinnostavaa, perusedellytykset täyttävää ohjelmistoa. Ohjelmistot ovat aiemmin työssä markkinointimateriaaliin perustuen esitellyt TIBCO Spotfire, KNIME, IBM Watson Studio ja Microsoft Azure Machine Learning Studio.

Koska testausaika ohjelmistoille oli melko rajallinen, on mahdollista, että tiettyjä ohjelmiston ominaisuuksia ei löydetty. Ohjelmistoja tarkkaillaan juurisyyanalyysin kannalta, joten online-datan hyödyntäminen ei ole otettuna huomioon. Testaamisen perusteella ohjelmistot pisteytetään alla esiteltyjen tarkkailun kohteena olevien asioiden ja painoarvojen mukaan. Painoarvoilla on suuri vaikutus vertailun lopputulokseen, joten niiden tarkka harkinta on tärkeää.

1. Yleinen helppokäyttöisyys, laatu ja tuki Painoarvo 1

Helppokäyttöisyys on ohjelmiston tärkein kriteeri. Eri menetelmien käytön tulee olla yksinkertaista, jotta tilanteeseen sopivin voidaan valita pisteytyksen perusteella. Graafinen mallien rakentaminen on helppokäyttöisyyden suhteen positiivinen asia. Myös tuen saanti joko ohjelmistolle tarkoitetuilta foorumeilta tai yritykseltä konsultointina on keskeistä helppokäyttöisyydelle.

Testausjärjestelyt:

Yleinen helppokäyttöisyys arvioidaan subjektiivisesti käyttökokemuksen perusteella viimeisenä, kun kaikki muut arvioinnit on tehty. Ominaisuuksien laadun arviointi kuuluu myös subjektiiviseen arviointiin, sillä osa ominaisuuksista ei välttämättä ole todellisuudessa käyttökelpoisia. Tuen saannin mahdollisuudet arvioidaan karkeasti testaamisessa syntyneiden ongelmatilanteiden ja niihin löytyneiden ratkaisujen perusteella. Dokumentaatiota ei erikseen tarkastella, vaan se katsotaan kuuluvaksi mukaan tukeen. Pääasia on ratkaisun löytämisen nopeus ja helppous. Konsultoinnin mahdollisuutta ja laatua ei arvioida, mutta mahdollisuudet konsultointiin esitetään.

2. Esikäsittelyominaisuudet ja tiedostojen luku Painoarvo 0.9
    a. Tietojen yhdistäminen
    b. Puuttuvat arvot ja vakioarvot
    c. Muuttujien valinta
    d. Viiveen asettaminen
    e. Datan osiin jakaminen
    f. Datan monistaminen

Tietojen esikäsittelyn helppous ja ominaisuusmäärä ovat kriittisiä ohjelmistoille, varsinkin kun käsitellään suuria datamääriä. Tietojen yhdistäminen on kriittinen ominaisuus esikäsittelyssä, sillä esimerkiksi juurisyyanalyysissa datanlähteitä voi olla valtava määrä.

Puuttuvien arvojen käsittely joko poistamalla kokonaisia sarakkeita tai korvaamalla puuttuvia arvoja automatisoidulla menetelmällä on kriittistä, sillä manuaalinen käsittely vie runsaasti aikaa. Sama pätee vakio- tai lähes-vakioriveihin.

Muuttujien valinta voi nopeuttaa mallien rakentamista huomattavasti. Jos rakentamiseen käytetään ylimäärin muuttujia, tapahtuu helposti ylisovitusta, mikä puolestaan vähentää luokittelutarkkuutta ja lisää mallin rakentamiseen käytettyä aikaa.

Aikaan sidotuissa käyttökohteissa, kuten prosessin juurisyyanalyysissa, viiveen asettaminen tietyille muuttujille on tärkeää tarkan analyysin tekemiseksi. Merkityksellisillä muuttujilla voi olla viivettä sekunteista tunteihin tutkittavan ongelman suhteen.

Luokittelijoiden toiminnan arviointia varten data tulee olla mahdollista jakaa osiin. Lisäksi tietyissä tilanteissa opetusdatan monistaminen esim. SMOTE:n avulla on hyödyllistä luokittelutuloksen parantamiseksi.

Testausjärjestelyt:

Esikäsittelyominaisuuksia testataan ohjelmistoissa SECOM-datan avulla. Data on runsaasti esikäsittelyä vaativaa, minkä lisäksi se on alkuperäisessä muodossaan kahdessa eri tekstitiedostossa. SECOM-datalla kyetään järkevästi testaamaan kaikki määritellyt esikäsittelyn ominaisuudet lukuun ottamatta viiveen asettamista. Viiveen asettaminen on kuitenkin triviaalia testata lähes millä tahansa aikasarjadatalla. Tiedostojen lukua voidaan testata syöttämällä eri muodossa olevaa dataa, esim. .txt, .csv ja .xlsx.

3. Menetelmien määrä Painoarvo 0.9

Sisäänrakennettujen menetelmien suuri määrä ohjelmistossa antaa hyvät mahdollisuudet rakentaa tehokkain malli luokittelulle ja merkitseville muuttujille. Eri menetelmiä kokeilemalla voidaan myös helpommin päätellä, onko esikäsittely epäonnistunut. Mikäli mikään menetelmä ei anna edes kohtalaisia tai heikkoja tuloksia, on esikäsittelyn epäonnistuneisuudella suuri todennäköisyys.

Testausjärjestelyt:

Menetelmien määrän arviointi perustuu tilastollisten- ja koneoppimismenetelmien karkeaan suuruusluokkaan.

4. Visualisointi- ja raportointiominaisuudet Painoarvo 0.5

Ohjelmiston visualisointi- ja raportointimahdollisuuksilla on kohtalainen merkitys. Itse analyysien kannalta nämä ominaisuudet eivät kuitenkaan ole kriittisimpiä. Riittävää on, että tulokset ymmärretään.

Testausjärjestelyt:

Visualisointi- ja raportointiominaisuuksissa arvioidaan ulos saatavien kuvien ja taulukoiden laatua. Hyvälaatuiset taulukot ovat kriittisiä itse analytiikassa, ja niiden avulla on mahdollista esittää tulokset halutulla tavalla missä vain ohjelmassa. Kuvien laatu on myös keskeinen asia raportoinnissa. Selkeä väritys ja muotoilu auttavat raportin intuitiivisessa ymmärtämisessä.

5. Menetelmien lisäysmahdollisuudet ja helppous Painoarvo 0.3

Ohjelmistoihin on mahdollista lisätä eri ominaisuuksia. Tämä vaatii yleisesti joko enemmän tai vähemmän ohjelmointia. Tässä työssä ei keskitytä menetelmien lisäämiseen ohjelmistoihin, joten arviointiperuste saa pienen painoarvon. Helppo menetelmien lisäys kuitenkin lisää ohjelmiston potentiaalia. Menetelmiä, jotka voidaan lisätä ohjelmistoon triviaalisti (esimerkiksi valitsemalla ohjelmiston sisäisestä valikosta menetelmä ja lataamalla se), arvioidaan mittarissa menetelmien määrä.

Testausjärjestelyt:

Lisäysmahdollisuuksia arvioidaan puhtaasti kunkin ohjelmiston dokumentaation perusteella.

6. Hinta Painoarvo 0.5

Ohjelmiston hinnalla on luonnollisesti kohtalainen painoarvo valinnassa. Asettamalla suhteellinen hinta ja sen painoarvo taulukkoon, pystytään arvioimaan sekä ominaisuuksia että hintaa samassa matriisissa. Vertailun lopputulos voidaan tällöin tulkita painoarvotettuna hinta-laatusuhteena. Jos halutaan ainoastaan ominaisuuksiltaan paras ohjelmisto, asetetaan hinnan painoarvoksi 0.

Pisteytys kullekin arvioitavalle ominaisuudelle tapahtuu välillä 1 – 10, jolloin parhaana arvona toimii 10. Ohjelmistoille annetut arvot ovat melko subjektiivisia ja melko lyhyen koekäytön jälkeen annettuja, minkä vuoksi arvioinnin loppupäätöksessä käytettävä arvojen summa on ainoastaan suuntaa antava.

Seuraavassa kappaleessa käydään ominaisuudet läpi ohjelmistoittain, kuvaten kunkin ohjelmiston kyvykkyys kyseisessä ominaisuudessa peräkkäin. Testaamisessa ei keskitytä vielä varsinaisiin tuloksiin, sillä tulokset ovat periaatteen tasolla menetelmäkohtaisia eivätkä ohjelmistokohtaisia. Varsinaiset pisteytykset on esitetty kappaleessa 6.7.

6.1 Yleinen helppokäyttöisyys ja saatavilla oleva tuki

Spotfire

Spotfiren sisäänrakennetut ominaisuudet ovat helppokäyttösiä. Erityisesti visualisointi on helppoa, ja vaihtoehtoja on monia. Malleja rakennettaessa on helppo valita halutut parametrit ja ominaisuudet. Datan saaminen oikeaan muotoon voi aiheuttaa hieman työtä. Johdetut arvot ovat laskettavissa insert calculated column -toiminnallisuuden avulla. Tämä tapahtuu kuitenkin sarake kerrallaan, mikä lisää työtä.

Tukea löytyy TIBCO:n wikistä, sekä yleisesti käyttäjiltä. Konsultointia on mahdollista hankkia yritykseltä.

KNIME

KNIME on helppokäyttöinen ja graafisen työnkulun rakentaminen on helppoa. Esimerkki työkulusta on annettu datan esikäsittelyn arvioinnissa. Ominaisuuksia löytyy runsaasti, ja ne ovat hyvin dokumentoituja ja pääosin helppokäyttöisiä.

Yksittäisten asioiden tekeminen on toisinaan haastavaa, mutta ohjelmistolla on aktiivinen yhteisö, minkä vuoksi ratkaisu yleensä löytyy melko nopealla haulla. KNIME AG ei panosta suuresti konsultointipalveluihin, joten yritykseltä tarkemman tuen saaminen on rajallista. Yksinkertaisiin kysymyksiin yrityksen edustajat vastaavat aktiivisesti KNIMEn foorumilla. Yleisen helppokäyttöisyyden vuoksi tukea ei kuitenkaan useimmissa tapauksissa tarvitse.

IBM Watson Studio Desktop/Cloud

Watson Studion työpöytä- ja pilvipalveluversioissa on periaatteessa samat toiminnallisuudet numeerisen datan analysointiin liittyvien Data Refineryn ja Modelerin suhteen, joten niitä käsitellään yhtenä kokonaisuutena. Pilvipalvelussa on omat etunsa laskennan ja mallien toimeenpanon suhteen, työpöytäversiossa datan yksityisyyden ja siirron suhteen.

Watsonin ominaisuudet ovat perusperiaatteeltaan helppokäyttöisiä, ja ohjelmiston ulkoasu on selkeä. Pienten ongelmien tullessa vastaan ei ohjelmisto kuitenkaan aina anna selkeitä tietoja virheen syystä, minkä vuoksi ongelmanratkaisu on toisinaan haastavaa.

Käytön hitaus on ongelma varsinkin ohjelmiston työpöytäversiossa. Tietokoneella, jonka prosessorina on Intel i5-6300U @ 2.5 GHz ja jossa on 8 GB RAM -muistia, osa yksinkertaisistakin operaatioista Data Refineryssä vei kymmeniä sekunteja, ellei minuutteja aikaa. Muut ohjelmistot suoriutuivat vastaavista operaatioista usein noin sekunnissa. Myös käynnistymisaika on Watson Studiolla huomattavasti pidempi kuin verrokkiohjelmistoilla.

IBM:n foorumeilta on mahdollista löytää tukea ongelmiin, ja konsultointia on mahdollista saada sekä IBM:ltä että kolmansilta osapuolilta.

Microsoft Azure Machine Learning Studio

Microsoft Azure Machine Learning Studion sisäänrakennetut ominaisuudet ovat helppokäyttöisiä. Tavallisessa käytössä ei tule huomattavia määriä virheitä, ja virheiden tapahtuessa annettu viesti ohjaa yleisesti oikeaan suuntaan.

Käyttöliittymä on selkeä, mutta kokemattoman käyttäjän voi olla haastavaa löytää yksikköä operaatiolle, jota ei ole aiemmin suorittanut. Esimerkiksi datan osiinjako tapahtuu useimmissa muissa ohjelmistoissa yksiköllä Partition, Azuren Machine Learning Studiossa tämä oli nimellä Split.

Palvelu toimii sujuvasti, eikä testikäytön aikana tullut vastaan tilanteita, joissa hitaus olisi haitannut toimintaa. Ainoastaan koneoppimismallien koulutuksessa kuluu huomattava määrä aikaa, kuten muissakin ohjelmistoissa.

Apua palvelun käyttöön löytää useilta eri foorumeilta, ja konsultointia on tarjolla Microsoftin arvioimilta kolmansilta osapuolilta.

6.2 Datan esikäsittely ja tiedostojen luku

Spotfire

Spotfire hyväksyy moitteetta .txt muodossa olevat SECOM-dataan liittyvät tiedostot ja parsii ne automaattisesti oikeisiin sarakkeisiin. Uusin Spotfiren versio myös ”arvaa” datatyypin hyvin, joten datatyypin valintaan ei kulu juurikaan aikaa. Mikäli jokin sarake on kuitenkin määritelty väärin, on tyypin vaihtaminen yksinkertaista dataa syöttäessä.

Tietojen yhdistäminen on Spotfiressa mahdollista melko yksinkertaisesti. Mikäli kahdella eri taulukolla on yhteinen tekijä, voidaan tätä hyödyntää yhdistämisessä. Jos yhteistä tekijää ei ole, kuten SECOM-datan tapauksessa, on yhdistäminen tehtävä lyhyen kiertoreitin kautta. Molemmille tauluille luodaan ensin rivitunniste. Tunnistetta käytetään tietojen yhdistämiseen vastaavasti kuin mitä vain muuta saraketta. Yhdistäminen voidaan toteuttaa Data Canvas -toiminnossa.

Spotfiresta ei testauksen aikana löytynyt yksinkertaisia suodattimia, joilla saisi esimerkiksi liikaa puuttuvia arvoja tai vakiona pysyviä arvoja suodatettua suuresta datajoukosta.

Ohjelma kuitenkin ottaa huomioon puuttuvat arvot malleja rakentaessa, mikä takaa helpon toiminnan.

Spotfiresta ei testiajanjaksolla löytynyt sisäänrakennettuja älykkäitä muuttujanvalintaominaisuuksia mallinrakennukseen, mutta manuaalinen valinta onnistuu. Lisäksi laskemalla ennen koneoppimismallien rakennusta esimerkiksi ANOVA:n F-arvot, voidaan nämä muuntaa pienellä työllä painoarvoiksi. Suodatusta on mahdollista tehdä kuitenkin esim. R-kielellä. Tiettyihin tarkasti määriteltyihin käyttötapauksiin löytyy valmiita esikäsittelypohjia.

Viiveen asettamiselle ei löydetty Spotfiressa suoraa ratkaisua. Tämäkin onnistuu kuitenkin periaatteessa kiertoreittiä käyttäen, jos laskee samalle taulukolle esim. RowID – 1...t arvot.

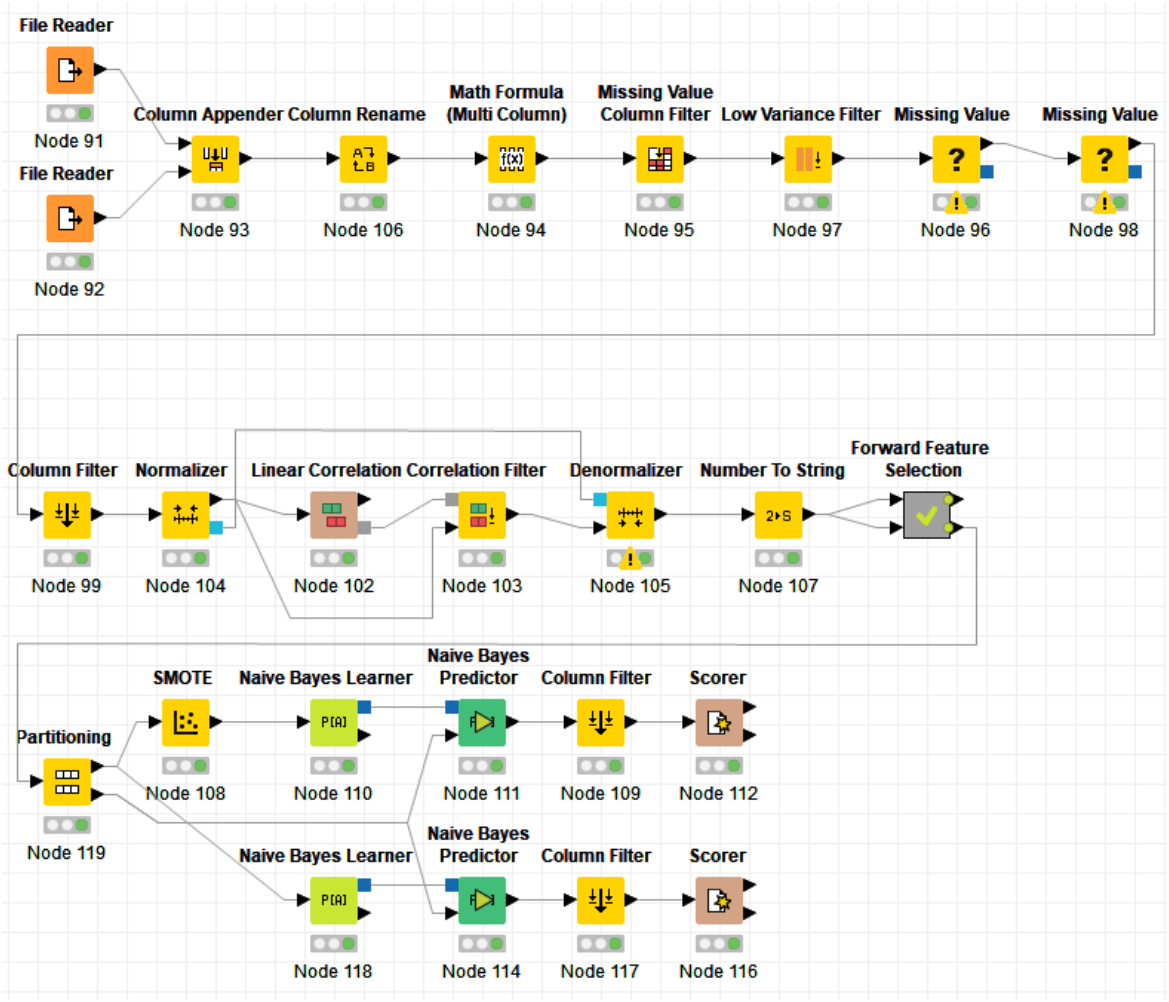
Datan osiin jakaminen tapahtuu luokittelualgoritmeja käyttäessä automaattisesti. Säädettävänä parametrina toimii validointijoukkojen määrä. Malli sovitetaan niin, että eri joukkoja käytetään opetuksessa ja testauksessa. Suodattaminen voidaan tehdä myös rajoittamalla esimerkiksi analyysissa mukana olevia rivejä. Ratkaisu ei kuitenkaan ole optimaalinen, sillä tarkka jako on haastavaa.

Datan monistamisen keinoja, kuten SMOTEa, ei ohjelmistossa ole suoriltaan saatavana.

Kaikki tapahtumat tehdään yleisesti yksittäin, eikä näkyvillä ole selkeää työnkulkua. Tämän vuoksi voi olla haastavaa korjata virheitä, joita on esikäsittelyssä tapahtunut.

KNIME

KNIME tarjoaa suuren määrän erilaisia datan esikäsittelyyn tarkoitettuja rakennuspalikoita, joita voidaan hyödyntää mallin graafisessa rakentamisessa. Luvussa 4.10 esitellylle SECOM-datalle rakennettu naiiviin Bayes luokitteluun johtava esikäsittelymalli on esitetty kuvassa 21. Kaikki esikäsittelykeinot, joita kuvassa on käytetty eivät välttämättä ole tilanteessa hyödyllisiä, mutta ne demonstroivat KNIMEn esikäsittelykykyä.



Kuva 21 Mahdollinen ratkaisu naiivin Bayes luokittelijan rakentamiseen ja pisteytykseen lähtien tiedostojen lukemisesta. Data etenee usean eri esikäsittelyvaiheen ja suodattimen läpi.

KNIMEn avulla voidaan virtaviivaistaa kaikki halutut datan esikäsittelyn vaiheet ja mallin käyttäminen. Käsittely alkaa eri tiedostojen lukemisesta, yhdistämisestä ja sarakkeiden uudelleenimeämisestä. Tämän jälkeen yksikössä Math Formula (Multi Column) muunnetaan SECOM-datan NaN arvot KNIMEn null muotoon. Yli 30 % null-arvoja sisältävät- ja matalan varianssin sarakkeet poistetaan. Jäljelle jääneet null-arvot korvataan ensisijaisesti edeltävällä arvolla, toissijaisesti seuraavalla arvolla. Yksikössä Column Filter poistetaan vielä nopeasti silmäiltävissä olevat hyödyttömät sarakkeet, sekä tapauksessa epäoleellinen aikaleimasarake. Tämän jälkeen kaikki muuttujien arvot normalisoidaan välille [0, 1] ja kaikkien muuttujien väliset lineaarikorrelaatiot lasketaan. Voimakkaasti toistensa kanssa korreloivista sarakkeista poistetaan määritettävä määrä, minkä jälkeen jäljelle jäänyt data denormalisoidaan alkuperäiseen muotoon. Luokkasarake muunnetaan

ohjelmiston vaatimaan string -muotoon ja data ajetaan naiivi Bayes -luokittelijan ominaisuusvalintaan, jossa eri yhdistelmiä kokeilemalla pyritään tarkimpaan luokittelutulokseen. Data jaetaan opetusdataan ja testidataan yksikössä Partitioning, minkä jälkeen malli rakennetaan ja pisteytetään sekä SMOTE:a hyödyntäen että ilman. KNIMEssä on myös mahdollista suorittaa erillinen pääkomponenttianalyysi PCA, joka muodostaa eniten varianssista selittävät lineaarikombinaatiot dataan perustuen.

Yllä oleva kuvaus sisältää kaikki arvioinnissa tarkkailtavat operaatiot, paitsi viiveen asettamisen. Viive on kuitenkin yksinkertaista asettaa Lag Column -yksiköllä silloin, kun sitä tarvitaan. Kaikkiaan KNIMEn esikäsittelyominaisuudet ovat erinomaisella tasolla.

IBM Watson Studio

IBM Watson Studio ei hyväksy SECOM -datan tekstitiedostoja. Työpöytäsovellus ei edes lue tiedostoja, pilvipalvelu sen sijaan lukee tiedostot, muttei hallitse muotoilua. Työpöytäsovellus vaatii luettavan tiedon olevan joko .csv tai .xls -muodossa.

Kun data on esikäsitelty .csv -muotoon esimerkiksi Excelillä, lukee Watson SECOM-datan muutamassa minuutissa. Tiedot eivät kuitenkaan automaattisesti ole oikeassa muodossa, vaan ohjelmisto esittää kaiken datan merkkijonomuodossa. Muodon pystyy muuttamaan joko vaihtamalla sarakkeet yksitellen, tai ajamalla verrattain yksinkertaisen R-kielisen komennon. Kun data on syötetty ja sarakkeet ovat oikeassa muodossa, onnistuu eri datalähteiden yhdistäminen helposti Modelerin Merge -yksiköllä.

Puuttuvien arvojen korvaaminen tapahtuu Data Refinery -osassa. Arvoja on mahdollista korvata joko tietyllä asetetulla arvolla tai käyttäen toista saraketta. Vakiosarakkeiden suodatukselle löytyy vaihtoehto Modeler-osion puolelta, nimellä Feature Selection. Kyseinen yksikkö suorittaa useampia suodatuksia, liittyen mm. puuttuvien arvojen määrään ja hajontaan.

Suorien viivesarakkeiden laskeminen ei onnistu ilman ohjelmointia, mutta Watson Studiosta löytyy suora ratkaisu aikasarjatiedon käsittelyyn nodesta Time Series. Ratkaisu sisältää suuren määrän eri vaihtoehtoja eri aikasarjaongelmien ratkaisuun.

Datan osiinjakaminen ja monistaminen onnistuvat ohjelmassa melko vaivattomasti käyttäen yksiköitä Partition ja SMOTE.

Microsoft Azure Machine Learning Studio

Azure Machine Learning Studio ei hyväksy tekstitiedostoja datanlähteenään. Suositusmuoto on .csv, ja näissä tiedostoissa datan tulee olla erotettu puolipisteellä. Jatkokäsittelyä varten SECOM-data syötettiin KNIMEllä muokatussa .csv -muodossa. Sarakkeiden yhdistäminen eri lähteistä onnistuu helposti Add Columns -yksiköllä.

Liikaa puuttuvia arvoja sisältävien sarakkeiden poistaminen onnistuu helposti Clean Missing Data -yksiköllä. Vakiona pysyvien arvojen poisto onnistuu todennäköisesti pienellä kiertoreitillä laskemalla ensin hajontaa tai varianssia, mutta testauksen aikana toimintoa ei löydetty.

Muuttujien valinnassa Azure Machine Learning Studio on hyvällä tasolla. Palvelussa voi suorittaa mm. PCA:n, ja valita muuttujia Fisherin lineeaarisella diskriminanttianalyysilla sekä muilla menetelmillä.

Viiveen asettamiselle ei löydetty suoraa ratkaisua palvelussa. Datan osiinjakaminen sen sijaan onnistuu Split Data -yksiköllä ja monistaminen SMOTE -yksiköllä.

6.3 Menetelmien määrä

Spotfire

Spotfire sisältää sisäänrakennettuja menetelmiä liittyen muuttujien välisiin yhteyksiin, klusterointiin, luokitteluun sekä regressioon. Muuttujien välisiä yhteyksiä voidaan tutkia viidellä eri menetelmällä. Klusterointi voidaan suorittaa K-means -algoritmilla tai hierarkisella algoritmilla, regressiomalli rakentaa lineaariregressiolla tai regressiopuulla sekä luokittelumallit joko luokittelupuulla tai logistisella regressiolla.

Kaikkiaan Spotfire siis sisältää melko pienen määrän varsinkin koneoppimisessa hyödynnettäviä malleja.

KNIME

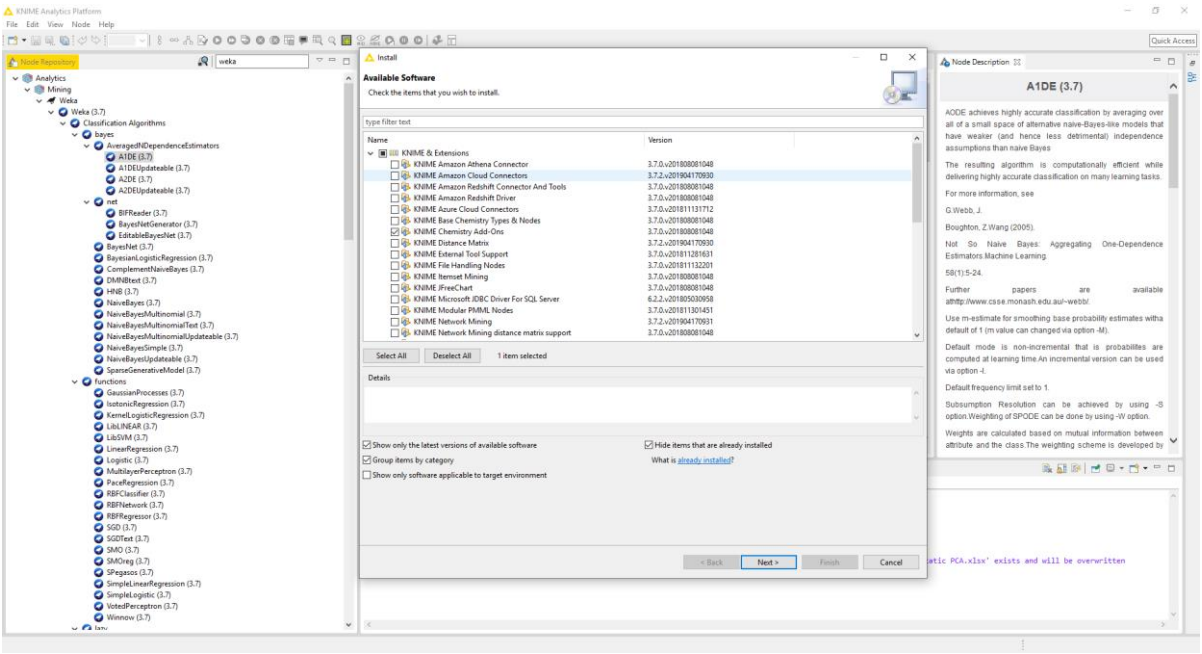
KNIME sisältää huomattavasti laajemman menetelmävalikoiman kuin Spotfire. Se sisältää käytännössä kaikki Spotfiren sisältämät menetelmät, minkä lisäksi se sisältää huomattavan määrän muita menetelmiä.

Luokittelun puolelta KNIME sisältää mm. naiivin Bayes -luokittelun, kaksi erilaista, säädettävää neuroverkkoa, päätöspuita ja metsiä (Random Forest, Gradient Boosting), lähimpään datapisteeseen perustuvan luokittelun, erilaisia regressioluokittelijoita ja support vector machinen. Myös muutamia klusterointialgoritmeja on saatavilla. Muuttujien välisiä yhteyksiä etsiviä algoritmeja on muutama enemmän kuin Spotfiressa.

KNIMEn Weka -lisäosalla saa yksinkertaisesti käyttöön valtavan määrän lisäalgoritmeja. Weka (Waikato Environment for Knowledge Analysis) on KNIMEä vastaava avoimen lähdekoodin ohjelmisto, joka on kehitetty Waikaton yliopistossa Uudessa-Seelannissa. (Witten, et al., 2016) Myös suuri määrä muita kirjastoja on saatavilla muutamalla napin painalluksella. Käytännössä mikä tahansa valmis lisäosa voidaan lisätä seuraavalla menettelyllä KNIMEn sisällä:

1. Avaa File -> Install KNIME Extensions
2. Hae tai selaa vaihtoehtoja, kunnes löydät haluamasi lisäosan
3. Valitse haluamasi lisäosa ja paina Next
4. Lue ja hyväksy käyttöoikeussopimus
5. KNIME lataa lisäosan ja asentaa sen
6. Lisäosa on käytettävissä KNIMEn uudelleenkäynnistyksen jälkeen

Kuva 22 sisältää kuvankaappauksen lisäosan lisäysprosessin kolmannesta vaiheesta. Kuvassa näkyy oikealla myös esimerkki WEKA -lisäosan sisältämän A1DE -algoritmin dokumentaatiosta



Kuva 22 Kuvankaappaus KNIMEstä lisäosan lisäysprosessin keskeltä. Oikeassa reunassa näkyy esimerkki WEKA -lisäosan sisältämän A1DE -algoritmin dokumentaatiosta.

IBM Watson Studio

Watsonin Modelerin Modeling -osio sisältää yhteensä 42 mallinnukseen tarkoitettua yksikköä. Joukossa on sekaisin luokitteluun, regressioon, klusterointiin, esikäsittelyyn ja muuttujien valintaan liittyviä yksiköitä. Eri tyyppeihin kuuluvia yksiköitä ei ole eritelty, mikä tekee tarkemman jaon haastavaksi. Menetelmien määrän suuruusluokka on samankaltainen kuin KNIMEssä ilman lisäosia.

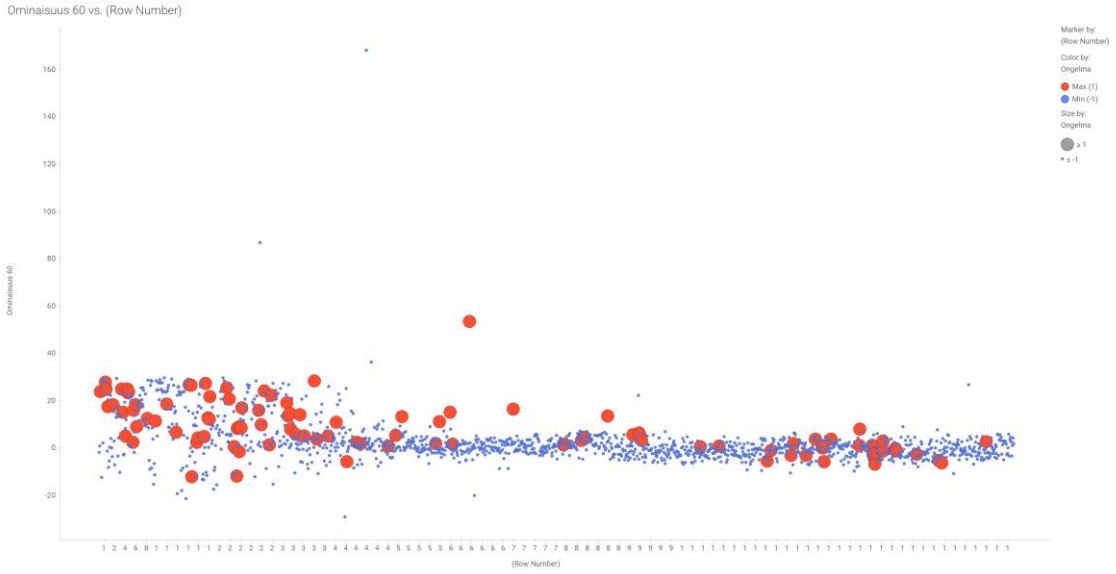
Microsoft Azure Machine Learning Studio

Azure Machine Learning studio sisältää koneoppimiseen liittyviä yksiköitä, jotka on lajiteltu poikkeuksien havainnointiin, luokitteluun, klusterointiin ja regressioon. Poikkeuksien havainnointiin liittyviä menetelmiä on 2, luokitteluun liittyviä menetelmiä 14, klusterointiin liittyviä menetelmiä 1 ja regressioon liittyviä menetelmiä 8. Azuren muuttujien valintaan ja esikäsittelyyn liittyvät menetelmät ovat eri osiossa. Karkealla tasolla menetelmien määrän voidaan sanoa olevan samaa luokkaa kuin Watsonissa tai lisäosattomassa KNIMEssä.

6.4 Visualisointi- ja raportointiominaisuudet

Spotfire

Spotfiren vahvuudet ovat sen visualisointiominaisuuksissa. Kaikilla Spotfiren lukuisilla visualisointiominaisuuksilla on suuri määrä helposti käytettäviä ominaisuuksia. Esimerkkinä toimii hajontakuvion (scatter plot) datapisteiden koon, -muodon ja -värin määrittäminen valittavien ominaisuuksien suhteen. Myös nopea suodattaminen tiettyjen arvojen suhteen on mahdollista. Alla olevassa esimerkkikuvassa luvussa 4.10 esitellyn SECOM-datan poikkeustilanteet on korostettu hajontakuviossa värin ja koon avulla.



Kuva 23 Poikkeustilannedata korostettuna Spotfiressa. Pystyakselina toimii yksi merkityksellisistä muuttujista, muuttuja 60, vaaka-akselina havainnon indeksi. Punaiset suuret datapisteet ovat poikkeusdataa, siniset pienet datapisteet normaalin toiminnan dataa.

Spotfire mahdollistaa myös ohjelmassa laskettujen tulosten muuntamisen esim. Excel-taulukoksi.

KNIME

KNIME mahdollistaa erittäin helpon ja tarkasti määriteltävän raportoinnin taulukkomuodossa. Myös tiettyjen kuvien, kuten korrelaatiokarttojen ja päätöspuiden raportointi on helppoa.

Erilaisten kuvaajien visualisointi on KNIMEssä verrattain rajallista. Tavallisten viivadiagrammien, hajontakuvioiden ym. tekeminen onnistuu, mutta muuttujien vaihtaminen ja erikoisten tapahtumien erittely ovat haastavampia operaatioita KNIMEssä kuin esimerkiksi Spotfiressa. Lisää visualisointiominaisuuksia saa lisäosia hyödyntämällä. Esimerkiksi R-visualisointeja voi hyödyntää KNIMEssä. Tukityökalun käyttäminen visuaalisessa esityksessä voi olla järkevä vaihtoehto, vaikka numeerisen osan analyysista tekisikin KNIMEllä, sillä interaktiivinen visuaalinen analyysi ei onnistu helposti. Hyödyntämällä esimerkiksi Spotfirea visualisoinnin työkaluna on mahdollista tulkita KNIMEllä saatuja tuloksia huomattavasti tehokkaammin ja nopeammin.

IBM Watson Studio

Watson Studio kykenee raportointiin hyvin sekä taulukko- että kuvamuodossa. Raportoinnin muoto valitaan Modelerissa käyttämällä haluttua yksikköä.

Visualisointiominaisuuksia Watsonissa on enemmän kuin KNIMEssä, mutta vähemmän kuin Spotfiressa. Spotfire voittaa myös responsiivisuudessaan ja helppokäyttöisyydessään Watsonin. Dataa pystyy visualisoimaan mm. viivadiagrammien, jakaumien ja histogrammien avulla.

Microsoft Azure Machine Learning Studio

Azure Machine Learning Studio sallii minkä tahansa työkulun vaiheen aikaisen datan lataamisen .csv -muodossa. Kunkin vaiheen muuttujien tiedot voidaan visualisoida histogrammeina tai laatikko-janakuviona (Box plot) sekä verrata esimerkiksi luokkamuuttujaan. Luokittelun laadusta kertova ROC-käyrä muodostetaan arvioinnin yhteydessä automaattisesti, ja kaksiluokkaluokittelun herkkyys on mahdollista säätää haluttuun arvoon.

6.5 Menetelmien lisäysmahdollisuudet ja helppous

Spotfire

Spotfiressa on mahdollista lisätä ominaisuuksia R-ohjelmointikielen avulla. Myös muilla kielillä ominaisuuksien lisääminen onnistuu, mutta sisäänrakennetun TERR –moottorin vuoksi R on yksinkertainen ratkaisu. R-kielisestä menetelmäkirjastosta, CRAN, löytyy menetelmiä laidasta laitaan. Menetelmien lisääminen ohjelmistoon ei kuitenkaan ole helppoa ilman kokemusta, vaan lisäyksessä tulee perehtyä kunkin menetelmän dokumentaatioon ja käyttöön. Aiemmasta kokemuksesta kielen parissa on varmasti hyötyä.

KNIME

KNIME mahdollistaa menetelmien ja toiminnallisuuden lisäyksen esimerkiksi Python- ja R-ohjelmoinnilla. Ohjelmisto kuitenkin sisältää valmiiksi todella laajan kirjaston, ja helposti saatavia menetelmiä on tarjolla runsaasti sisäänrakennetun lisäosapalvelun kautta. Koodin sisällyttäminen ohjelmistoon on helppoa, mutta sen kirjoittaminen vaatii toki taitoa kyseisessä kielessä.

IBM Watson Studio

Watson Studio on yhteensopiva mm. R:n ja Pythonin kanssa, mikä sallii menetelmien lisäämisen ohjelmointitaitoisille. Pilvipalvelussa on mahdollista hyödyntää mm. Sparkin MLlib -menetelmäkirjastoa ja TensorFlow’ta online-ennustuksessa. Rakentaminen ja datanlähteiden integroiminen vaativat työtä, kuten millä tahansa online-alustalla.

Microsoft Azure Machine Learning Studio

Azure Machine Learning Studio hyväksyy ohjelmakoodia Pythonilla ja R:llä, mikä takaa kohtalaisen helpon uusien menetelmien sisällyttämisen. Konenäkötoimintoihin liittyen palvelu sallii myös OpenCV -kirjaston hyödyntämisen.

6.6 Hinta

Spotfire

Spotfiren Analyst-lisenssin hinta pilvipalveluun on 1 250 dollaria vuodessa, mikä on verrattain halpa. Hinnoittelua omilla palvelimilla ajamiseen ei ole suoraan tarjolla. Ohjelmiston voi myös hankkia Amazonin AWS -pilvipalvelussa, jolloin hinta on 8 400 dollaria vuodessa oletusasetuksilla. Paremmat palveluasetukset maksavat enemmän ja pilvipalvelun käytöstä maksetaan erillinen tuntihinta. (TIBCO Software Inc. [3], 2019)

KNIME

KNIME on työpöytäkäytössä ilmainen ja sitä on mahdollista suorittaa esimerkiksi Microsoftin Azure- ja Amazonin AWS -pilvipalveluissa, joiden käytöllä on oma hinnoittelunsa.

KNIME AG tarjoaa myös mahdollisuuden tiimityöskentelyyn KNIME Server -palveluillaan, mikä sallii eri ominaisuuksia riippuen palvelun laajuudesta. Palvelun hinta on 7 500, 25 000 tai 45 500 euroa vuodessa, riippuen valitseeko pienen, keskisuuren vai suuren palvelun. Lista eri KNIME Server -versioiden toiminnallisuuksista on saatavilla https://www.knime.com/knime-software/knime-server-pricing. (KNIME AG [2], 2018) KNIME Server -palvelua voi suorittaa samoissa pilvipalveluissa kuin normaalia versiota tai omalla palvelimella.

IBM Watson Studio

Työpöytäversio IBM Watson Studiosta maksaa 200 euroa kuukaudessa eli 2 400 euroa vuodessa. (IBM [6], 2019)

Pilvipalvelun hinta vaihtelee. Tarjolla on ilmainen Lite-versio, johon sisältyy hyvin rajallinen määrä tallennustilaa ja prosessointitehoa. Toinen mahdollisuus on maksaa käytön mukaan, jolloin kustakin käytetystä komponentista maksetaan erillinen käyttötuntihinta. Eri komponentteja on tarjolla monia, joten tarkan hinta-arvion laskeminen on haastavaa. Kolmas

vaihtoehto on sopia mukautettu sopimus IBM:n kanssa, jolloin on mahdollista saada kiinteä kuukausihinta, tarjouksia ja mukautusvaihtoehtoja.

Microsoft Azure Machine Learning Studio

Azure Machine Learning Studion peruskäyttöoikeus maksaa 8,425 euroa kuukaudessa, minkä lisäksi laskutetaan kiinteä käytön tuntihinta, 0,844 euroa tunnissa. Hintaan ei sisälly tallennustilaa, vaan se täytyy hankkia erikseen. (Microsoft [9], 2019)

Studion käytön lisäksi on mahdollista hankkia ”tuotantoversio”, jonka avulla mallien suunnittelija voi antaa mallit kuluttajille käyttöön. Palvelun hinta on käyttömääräsidonnainen ja hinta vaihtelee nollan ja 8 433 euron välillä. (Microsoft [9], 2019)

6.7 Yhteenveto ohjelmistoista

Eri ohjelmistojen pisteytykset ja ominaisuuksien painoarvot on esitetty taulukossa 11. Taulukossa 12 painoarvot on kerrottu pisteytyksiin, ja summat on laskettu.

Taulukko 11 Eri ohjelmistojen pisteytykset ominaisuuksittain ja ominaisuuksien painoarvot.

	Painoarvo	Spotfire	KNIME	IBM Watson	MS Azure ML Studio
Yleinen käyttökokemus	1	7	9	5	7
Esikäsittelyominaisuudet	0.9	3	9	5	7
Menetelmien määrä	0.9	3	9	8	8
Visualisointi	0.5	10	4	7	6
Menetelmien lisäys	0.3	5	9	8	8
Hinta	0.5	8	10	5	5

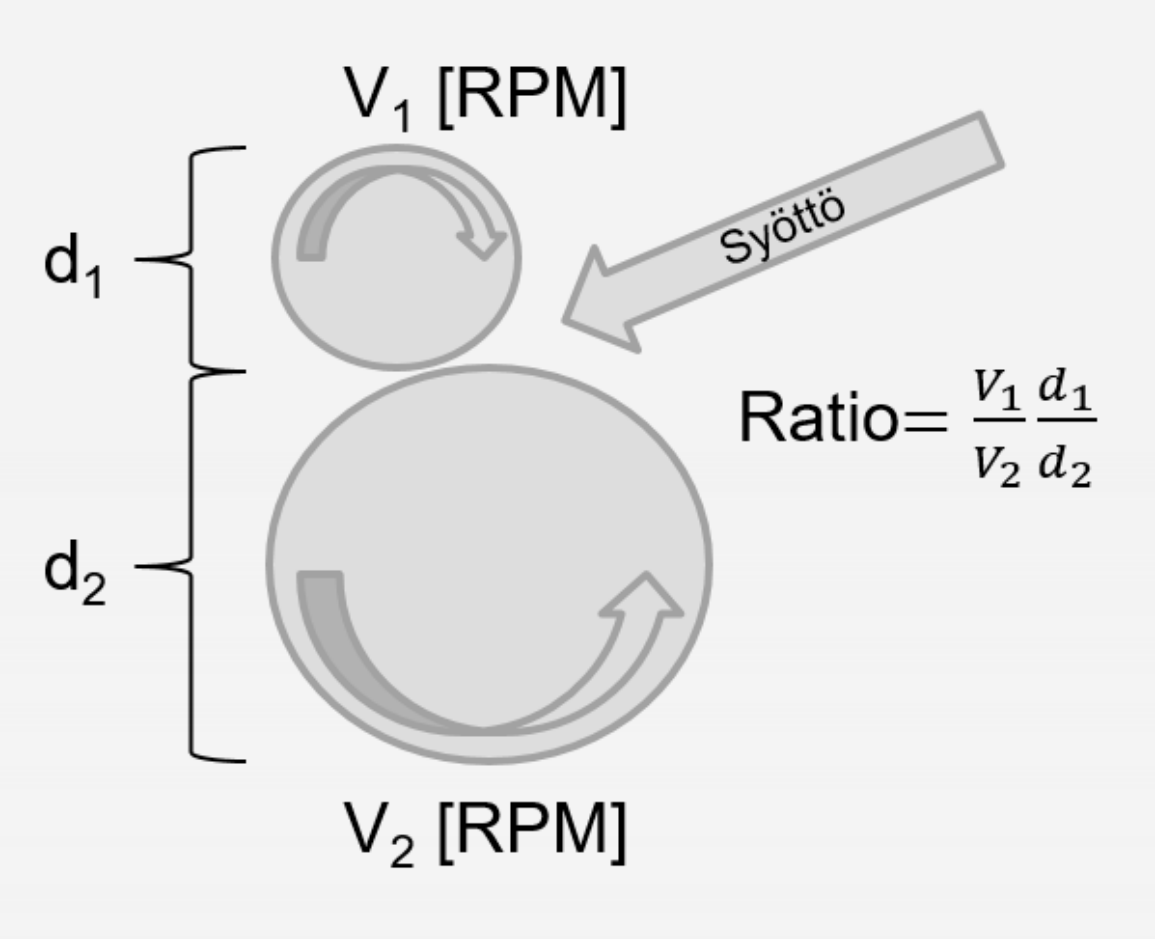
Taulukko 12 Ominaisuuksien pisteytykset painoarvojen kertomisen jälkeen. Teoreettinen maksimi on 41 pistettä.

	Spotfire	KNIME	IBM Watson	MS Azure ML Studio
Yleinen käyttökokemus	7	9	5	7
Esikäsittelyominaisuudet	2.7	8.1	4.5	6.3
Menetelmien määrä	2.7	8.1	7.2	7.2
Visualisointi	5	2	3.5	3
Menetelmien lisäys	1.5	2.7	2.4	2.4
Hinta	4	5	2.5	2.5
Yhteensä	22.9	34.9	25.1	28.4

Vertailussa KNIME saavuttaa korkeimmat yhteispisteet. Spotfire saa vertailussa heikoimmat pisteet, mutta kyseessä on pääasiassa ”Business Intelligence” -visualisointiin tarkoitettu ohjelmisto, eikä ohjelmistossa ole työnkulkuun perustuvaa mallinrakennusta. Vertailun perusteella tehdyllä valinnalla tässä työssä hyödynnetään analytiikassa KNIMEä, mutta visualisoinneissa pääasiassa Spotfirea.

7. Dataan pohjautuva juurisyyanalyysi teollisuustapauksessa

Teollisuuslaitoksella tuotanto pysähtyi ongelmatilanteiden vuoksi lyhyellä aikavälillä useaan otteeseen, aiheuttaen huomattavan määrän taloudellista haittaa tuotannon vähenemisen vuoksi. Ainakin osittainen juurisyy saatiin selville alkuperäisessä tutkimuksessa. Prosessin kahden vastakkain pyörivän sylinterin välinen nopeussuhde muuttui oleellisesti. Kuva 24 havainnollistaa kyseistä nopeussuhdetta.



Kuva 24 Prosessin ongelman juurisyynä toiminut kahden sylinterin välinen nopeussuhde.

Tässä työssä paneudutaan tapaukseen, ja pyritään löytämään tapauksen juurisyyt ja ennustamaan tuotannon ongelmatilanteita historiadatasta. Tässä työssä käytetty data on vain osa alkuperäisessä analyysissa käytetystä datasta.

Tulkittava data sisältää minuutin välein kerättyjä mittausarvoja yhteensä 61:stä eri prosessimuuttujasta, joista yksi on laitteen tuotanto. Tuotanto romahtaa ongelmatilanteen syntyessä nollaan, mikä sallii tilanteiden paikallistamisen datasta ilman ongelmien aikatietoa. Romahdustapauksia, joiden välissä tuotanto palaa normaalille tasolle on datassa yhteensä 30. Kaikki tuotannon alenemat oletetaan ongelmaperäisiksi. Osa prosessimuuttujista sisältää suuria määriä puuttuvia arvoja tai pysyy vakiona, joten olennainen osa ongelmanratkaisua on datan esikäsittely. Dataa on yhteensä noin 23 000 riviä. Kokonaisuudessaan datasoluja on siis noin 1 426 000. Data sisältää tietoa kemiallisista, mekaanisisista- ja prosessinohjausarvoista.

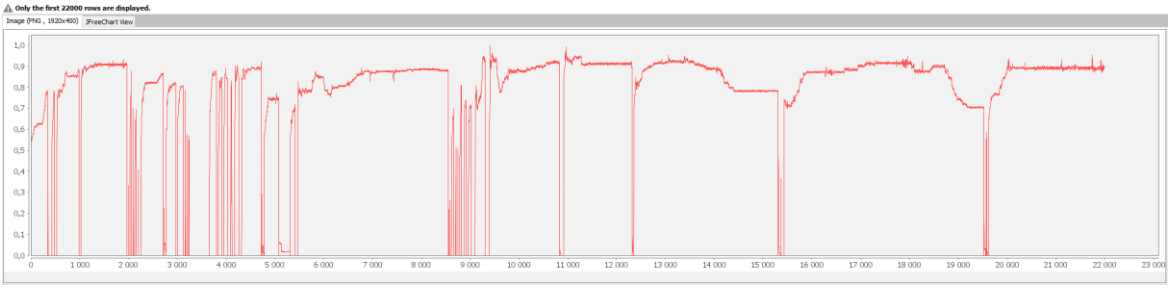
Yhden minuutin resoluution vuoksi kaikkia tilanteita ei voi välttämättä data-analyysin perusteella havaita. Prosessin ongelman tiedetään eskaloituneen erittäin nopeasti sen tapahtuessa, minkä vuoksi tarkempaa resoluutiota tarvittaisiin kaikkien tapauksien selvittämiseksi.

Ongelman lähestyminen aloitetaan ongelmatilanteiden ja normaalin toiminnan luokittelusta. Luokittelu aloitetaan luokittelemalla Ajotilanne -sarakkeeseen Excel -funktiolla normaali toiminta arvolla 0 ja nollatuotanto arvolla 2. Luokittelua jatketaan käsin, ja valittu määrä romahdusta välittömästi edeltäviä normaalin toiminnan minuutteja luokitellaan arvolla 1. Alustavasti luokkaan 1 sisällytetään vain viimeisin arvo, useampien arvojen lisäys tapahtuu tämän jälkeen helposti. Arvo 2 täytetään lisäksi matalan tuotannon ja ylösajojen luokaksi. Taulukossa 13 on esitetty esimerkki ongelmatilanteen luokittelusta. Kuva 25 sisältää välille [0, 1] normalisoidun tuotannon datankeräysajanjaksolta.

Taulukko esittää vain hyvin pienen osan tiedosta, ylivoimaisesti suurin osa datasta on normaalin toiminnan dataa.

Taulukko 13 Ongelmatilanteiden esimerkkiluokittelu tuotannon mukaan teollisuustapauksessa. Tuotanto on esitetty anonymisoinnin vuoksi vakiolla kerrotussa määrittelemättömässä yksikössä.

Ajotilanne	Tuotanto
0	2733.62
0	2728.73
0	2732.97
0	2734.00
0	2731.51
0	2736.52
0	2746.27
0	2740.10
0	2739.60
1	2746.71
2	1832.31
2	0.00
2	0.00
2	0.00
2	579.25
2	1158.50
2	0.00
2	0.00



Kuva 25 Teollisuusprosessin normalisoitu tuotantomuuttuja tutkittavalla ajanjaksolla. Visualisointi on suoritettu KNIMEssä.

Yllä määritellyn luokittelun perusteella pyritään löytämään datan avulla ongelmat aiheuttanut juurisyy. Juurisyyn etsimiseen käytetään kolmea tilastollista menetelmää.

Ensimmäisenä menetelmänä toimii eri ominaisuuksien väliset korrelaatiot, joiden perusteella voidaan tarkkailla korreloivatko tietyn prosessimuuttujan arvot positiivisesti tai negatiivisesti Ajotilanteen kanssa. Lisäksi korrelaatioita seuraamalla kyetään tulkitsemaan omituisesti käyttäytyvän muuttujan vuorovaikutuksia muihin muuttujiin.

Toisena menetelmänä toimii varianssianalyysi ANOVA. ANOVA lasketaan kullekin prosessimuuttujalle erikseen Ajotilanteen suhteen. Tuloksena saadut F- ja todennäköisyysarvot kertovat mitatunlaisten jakaumien esiintymisen todennäköisyydestä datassa satunnaisesti. Suuret F-arvot ja pienet todennäköisyysarvot omaavat prosessimuuttujat ovat siis tapauksessa todennäköisesti systemaattisesti normaalitilanteesta poikkeavalla tasolla, mikä antaa viitteitä kyseisten muuttujien osuudesta ongelman muodostumisessa.

Kolmantena menetelmänä toimii χ^2-testi. Testin periaate vastaa tapauksessa pitkälti ANOVA:a eli se kertoo prosessimuuttujien arvojen jakautumisen todennäköisyydestä mitatunlaisesti. χ^2-testi on kuitenkin etenkin epätasaisen luokkajaon tapauksessa ANOVAa herkempi voimakkaasti poikkeaville datapisteille vähemmistöluokassa.

Yhdistämällä näiden kolmen testin tulokset saadaan selville, mitkä prosessimuuttujat ovat poikenneet merkityksellisesti normaalitilanteesta juuri ennen ongelman syntymistä. Kun merkitykselliset muuttujat tunnetaan, on mahdollista prosessitiedon ja korrelaatioiden avulla

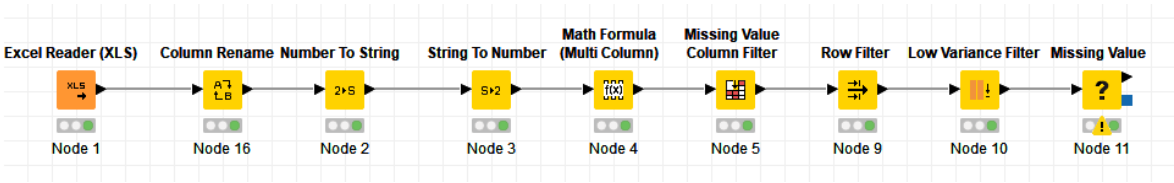
päästä varsinaiseen juurisyyhyn kiinni, mikäli se ei ole suoraan jokin testien merkittävimmistä muuttujista.

Tilastollisten menetelmien lisäksi ongelmaa pyritään mallintamaan eri luokittelualgoritmeilla. Mallinnuksen avulla pyritään havaitsemaan viimeinen minuutti ennen ongelmatilanteen syntyä. Nopeasti reagoivat operaattorit tai automaattinen järjestelmä olisi mallinnetun hälytyksen perusteella kykenevä esimerkiksi pienentämään prosessin syötön määrää ongelman lievittämiseksi. Esitetyssä tapauksessa juurisyy onnistuttiin korjaamaan muuttamalla prosessiparametreja, mutta tilanteessa, jossa selkeää juurisyytä ei löydetä tai juurisyytä ei kyetä korjaamaan, olisi mallintamisesta potentiaalisesti suuri hyöty.

7.1 Esikäsittely

Alkuperäinen data sisältää tuhansia väärään muotoon päätyneitä datapisteitä. Datapisteet olivat syystä tai toisesta aiemmassa käsittelyssä muuttuneet Excelin päivämäärämuotoon, mikä on tiedostossa fyysisesti ilmaistu kokonaislukuna päivästä 1.1.1900. Päivämääriksi muuttunut tieto poistettiin muuntamalla kaikki muodossa ”*.*.*” olleet solut muotoon NaN.

Excelillä tehdyn alustavan esikäsittelyn jälkeen tiedosto syötettiin KNIMEen. Kuvassa 26 on esitetty KNIMEssä tehty esikäsittelyn työnkulku.



Kuva 26 Datan esikäsittely KNIMEssä. Esikäsittely kattaa työnkulun vaiheet tiedoston lukemisesta puuttuvien arvojen täydentämiseen.

Kaikki prosessimuuttujien mittaussarakkeet muutettiin numeeriseen muotoon ja 2:lla merkattu koneen alhaallaolotieto suodatettiin pois. Näin ollen jäljelle jäi tietoa ainoastaan ajotilanteista, joissa tuotanto on normaalilla tasolla eli merkattuna ajotilanteena on 0 tai 1.

Tutkittavien rivien määrittämisen jälkeen suodatettiin sarakkeet, jotka sisältävät yli 30 % puuttuvia arvoja. Tämän jälkeen suodatettiin pienen varianssin omaavat sarakkeet,

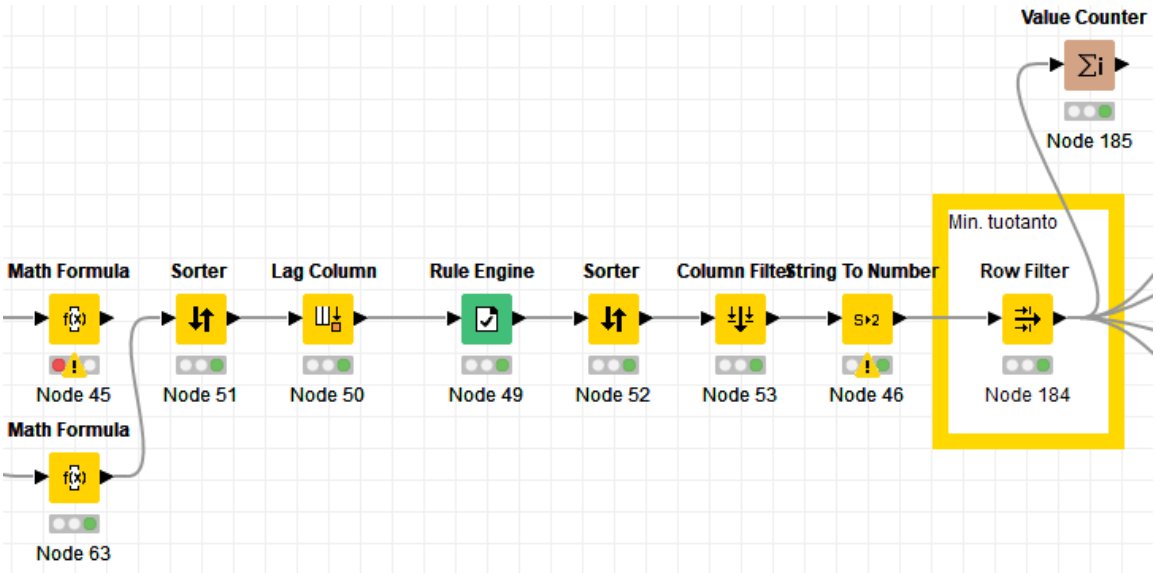
suodatusrajana toimi 0,1. Käsittelyt poistivat yhteensä kuusi saraketta, jättäen jäljelle 55 muuttujaa. Tilastolliset testit suoritettiin tälle datajoukolle.

Datan jäljelle jääneet puuttuvat arvot korvattiin kunkin muuttujan edellisellä arvolla. Edellistä arvoa käytettiin datan aikasarjamuodon vuoksi. Jatkuvan prosessin edellisen minuutin arvo on yleisesti normaalissa toiminnassa huomattavasti lähempänä todellista arvoa kuin esimerkiksi edellisen vuorokauden keskiarvo.

7.2 Tilastolliset analyysit

Käsiteltävälle datalle lasketaan eri tilastollisia arvoja liittyen asetettujen luokkien välisiin eroihin. Datalle lasketaan luokkien väliset tulokset erikseen, kun yksi, kaksi tai viisi ongelmatilannetta edeltävää arvoa on luokiteltu luokkaan Ajotilanne = 1.

Arvojen Ajotilanne = 1 lisääminen dataan onnistuu KNIMEssä kuvan 27 mukaisella työnkululla. Työnkulku on suoraa jatkoa kuvan 26 työnkululle, alkaen Low Variance Filter -yksiköstä.



Kuva 27 Työnkulku eri luokittelun 1 määrille. Kaksi ensimmäistä yksikköä ovat vaihtoehtoisia, riippuen käytetäänkö anonymisoitua dataa. Työnkulku kattaa nopeussuhteen laskemisen, valitun määrän Ajotilanne = 1 arvoja lisäämisen sekä datan suodatuksen tuotannon mukaan.

Ensimmäinen Math Formula -yksikkö laskee raakadatasta puuttuneen, luvussa 7 kuvassa 24 esitellyn, kahden sylinterin välisen nopeussuhteen. Kyseisen nopeuksien suhteen tiedetään olleen osasyynä tuotannon ongelmille aiemman datan analysoinnin perusteella.

Nopeussuhteen laskemisen jälkeen taulukko käännetään ympäri. Ajotilannetta kuvaava sarake viivästetään käännetyssä taulukossa askeleen välein yhteensä viisi askeletta (tai mikä tahansa määritelty määrä askeleita). Käytännössä tämä tarkoittaa, että Ajotilanteen kunkin hetkistä arvoa voidaan verrata sen viiteen edelliseen arvoon. Yksikössä Rule Engine annetaan ehdot arvon 1 lisäämiselle ajotilannesarakkeeseen. Mikäli arvo Ajotilanne(-t) = 1, sijoitetaan ajotilanteeseen arvo 1. Ajotilanne(–t) vastaa Ajotilanteen arvoa t minuuttia sitten. Määrittely on nopeasti muunnettavissa kommentoimalla tai poistamalla Rule Enginen riveistä kommentteja. Määrittely on esitelty alla kuvassa 28.

Kuva 28 Rule Engine -yksikön määrittely Ajotilannetiedolle.

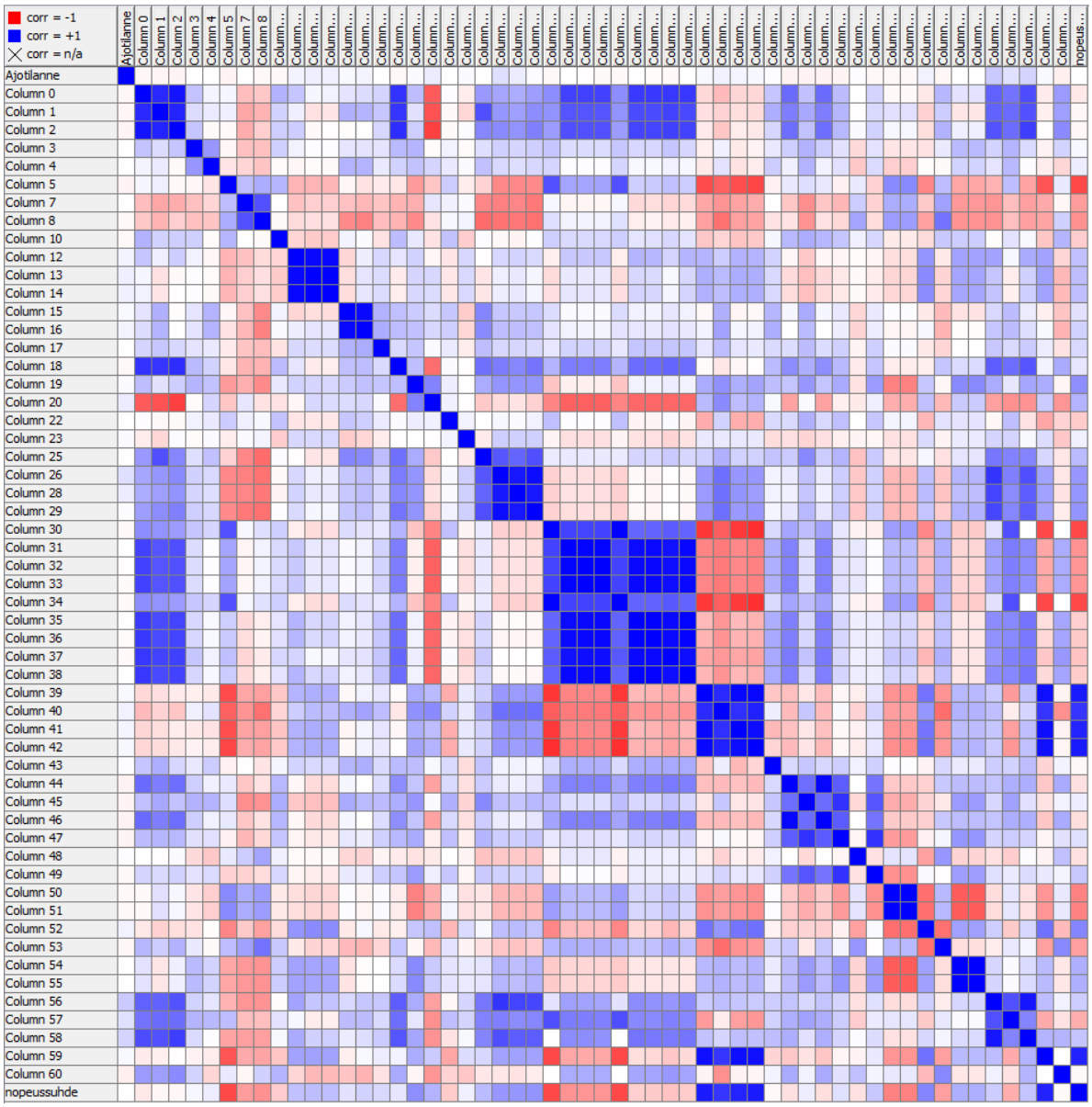
Halutun määrän ongelmatilannetta edeltävien arvojen määrittelyn jälkeen taulukko käännetään takaisin oikein päin, ja ylimääräiset ”Ajotilanne(-t)” sarakkeet poistetaan. Viimeisenä yksikkönä ennen testejä ja niiden yksittäisiä tarpeita on tuotannon määrittely. Yksikössä tuotannolle voidaan asettaa halutut ylä- ja alarajat tuotannon määrän mukaan, mikäli halutaan testata eri tuotantotilanteiden eroja. Ylimääräisessä haarassa oleva Value Counter -yksikkö kertoo Ajotilanteiden 1 ja 0 määrät eri tilanteissa. Vaihtehtoisesti voidaan käyttää Rule-Based Row Filter -yksikköä, jolla voidaan määritellä mukaan otettaviksi

riveiksi esimerkiksi vain lähellä ongelmatilanteiden tuotantoa olevat rivit. Tämä vähentää ongelmattomien ajonopeuksien merkitystä datassa. Testien tapauksessa tuotantotilanteiden rajauksena käytetään Ajotilannetiedon lisäksi tuotannon arvoja niin, että vain n. ±0.5 % mistä tahansa ongelmatilanteesta oleva tuotanto pyöristettynä tasalukuun huomioidaan. Tämä takaa käytettävien normaalin toiminnan arvojen verrannollisuuden ongelma-arvoihin nähden.

Tilastollisissa analyyseissa tulee ottaa huomioon Ajotilanteiden lukumääräsuhde. Analyysit ovat yleisesti tarkoitettuja tasaiselle luokkajaolle, minkä vuoksi osa tuloksien arvoista on suuntaa antavia eivätkä täydellisesti tilastollisesti päteviä. Tutkittavassa tapauksessa Ajotilanne 1 -tapauksia on 30 ja Ajotilanne 0 -tapauksia tuhansia. Näin ollen periaatteen tasolla Ajotilanne 1 -tapahtumien jakautuminen todellisuuteen on huomattavasti epätäydellisempää kuin Ajotilanne 0:n. Lisätessä Ajotilanne 1 -tapausten aikaikkunaa tasapainottuu ongelmatilanteiden määrä hiljalleen suhteessa normaaliin toimintaan. Tämän vuoksi tilastolliset ongelmat helpottuvat, mutta myös todellinen jakauma verrattuna vain viimeiseen minuuttiin muuttuu.

7.2.1 Ominaisuuksien korrelaatio ajotilanteen kanssa ja keskenään

Kaikille prosessimuuttujille voidaan laskea keskinäiset Pearsonin korrelaatiot ja esittää ne numeerisesti tai värikoodattuna. KNIMEssä tämä tapahtuu käyttämällä yksikköä Linear Correlation. Alla kuvassa 29 on esitetty värikoodattu korrelaatiomatriisi kaikkien prosessimuuttujien sekä Ajotilanteen välillä. Numeerinen data korrelaatiosta Ajotilanteen kanssa on liitteessä 4.



Kuva 29 Tilastollisessa analyysissa käytettyjen prosessimuuttujien ja Ajotilanteen korrelaatiomatriisi. Sininen väri kuvaa positiivista korrelaatiota, punainen negatiivista. Vaaleus tarkoittaa korrelaation puuttumista. Sana Column, eli sarake, vastaa prosessimuuttujaa.

Vaikka lineaarikorrelaatio ei olekaan kaikkein soveltuvin luokkien välisten erojen selvittämiseen, kuvasta nähdään prosessimuuttujien 26, 56, 57 ja 58 korreloivan eniten Ajotilanteen kanssa. Lineaarikorrelaatio ei kuitenkaan ole kovin voimakasta, sillä suurin korrelaatio Ajotilanteen kanssa on prosessimuuttujan 56 korrelaatio 0,207. Spearmanin korrelaatiolla saadaan vielä pienempiä korrelaatioarvoja.

Ominaisuudet keskenään sisältävät voimakkaitakin korrelaatioita, niin positiivisia kuin negatiivisia. Matriisin diagonaalilla esiintyy useampikin 2x2, 3x3 tai 4x4 voimakasta korrelaatiota sisältävä neliö. Nämä selittyvät kyseessä olevilla muuttujilla, koska lähes kaikissa tapauksissa kyseessä ovat mielivaltainen prosessimuuttuja, sekä tämän asetus-, minimi- ja maksimiarvot. Osa tapauksista ei kuitenkaan sisällä näin näkyviä syy-seuraussuhteita. Karttaa voidaan käyttää juurisyyn hakemisessa, kun merkittävimmät muuttujat ovat selvillä.

7.2.2 Ominaisuuksien varianssianalyysi Ajotilanteen suhteen

Varianssianalyysin suorittaminen datan esikäsittelyn jälkeen on yksinkertaista, eikä vaadi KNIMEssä muuta kuin One-way ANOVA -yksikön. Datan jälkikäsittely esitettävään muotoon vaatii jonkin verran työtä, muttei ole analyysin toiminnan kannalta keskeistä.

Varianssianalyysin laskemisessa käytetty yksikkö laskee myös Levenen testin (NIST/SEMATECH, 2012). Testin tarkoituksena on tarkastaa, onko eri luokilla sama varianssi. Saman varianssin oletus on useassa eri tilastollisessa menetelmässä, mukaan lukien ANOVA:ssa. Testin tarkoituksena on erotella epätasaisen luokkajaon tapauksissa muuttujat, jotka erottuvat ANOVA:ssa suuren varianssinsa vuoksi. (NIST/SEMATECH, 2012) Tilanteessa, jossa yhtä luokkaa on 30 ja toista tuhansia, ANOVA löytää käytännössä välttämättäkin joitakin merkityksellisiä muuttujia riippumatta valituista pisteistä. Merkityksellisyyden suuruusluokka kuitenkin vaihtuu pisteiden valinnan seurauksena, kuten myöhemmin esitetään.

ANOVA:n periaate teollisuuden ongelmien löytämisessä ei ole riippuvainen täydellisestä tilastollisesta paikkaansapitävyydestä. Päinvastoin, suuren varianssin havaitseminen ongelmatilanteiden välillä voi ohjata osassa tapauksia juurisyynä toimivan prosessimuuttujan suuntaan. Näin ollen sekä ANOVA:n F-arvon että Levenen testin W-arvon tarkkailu on kiinnostavaa juurisyyanalyysin kannalta. Näin saadaan tietoa sekä luokkien välisistä eroista että luokkien sisäisten varianssien välisistä eroista. Varsinkin nopeasti kehittyvien ongelmatilanteiden yhteydessä karkealla resoluutiolla kerätyllä datalla varianssi voinee antaa tietoa poikkeuksellisesti käyttäytyvistä prosessimuuttujista.

Alla taulukoissa 14 ja 15 on yhden minuutin Ajotilanne 1 -ikkunalla ajetun ANOVAn tulokset järjestettynä sekä suurimman F-arvon että suurimman Levenen testin arvon W mukaan. Data on suodatettu edellä esitetyn ongelmatilanne ±0.5 % tuotannon mukaan.

Taulukko 14 1 min ongelma-aikaikkunan suurimmat kymmenen ANOVAn F-arvoa omaavaa muuttujaa.

Prosessimuuttuja	F	W
56	572,97	1478,80
58	369,51	991,41
26	205,36	390,08
57	195,00	688,30
28	71,17	88,17
20	69,18	56,47
60	53,43	81,22
13	48,91	18,70
14	47,10	17,91
40	35,73	17,31

Taulukko 15 1 min ongelma-aikaikkunan suurimmat kymmenen Levenen testin arvoa W omaavaa muuttujaa.

Prosessimuuttuja	F	W
56	572,97	1478,80
58	369,51	991,41
57	195,00	688,30
26	205,36	390,08
Nopeussuhde	0,01	115,28
28	71,17	88,18
60	53,43	81,23
20	69,18	56,47
18	10,47	37,74
16	24,15	30,66

Testien tuloksien perusteella neljä tärkeintä muuttujaa molemmissa tapauksissa ovat samat, tosin kolmas ja neljäs vaihtavat paikkaa. Muuttujille voidaan tämän vuoksi tehdä seuraava tulkinta: sekä muuttujien välinen varianssi eri luokissa että muuttujien luokkien sisäisten varianssien ero ovat suuria. Näin ollen kyseisten ominaisuuksien voidaan todeta käyttäytyvän ongelmatilannetta edeltävällä minuutilla merkittävästi eri tavoin, kuin muun

normaalin toiminnan aikana. Tulkinnan perusteella ominaisuuksien voidaan katsoa olevan ongelman syntymisessä mukana joko suoraan tai välillisesti.

Nopeussuhteen havaitaan pysyvän luokkien välillä melko vakiona, mutta sen luokkien sisäinen varianssi muuttuu huomattavasti.

Vertailun vuoksi tarkkaillaan ainoastaan kahta minuuttia ongelmatilannetta edeltävää minuuttia. Ensimmäinen edeltävä minuutti on suodatettu datasta kokonaan pois. Tulokset testeille ovat esitetty taulukoissa 16 ja 17.

Taulukko 16 Vain toisen ongelman jälkeisen minuutin sisältävän ANOVAn suurimmat kymmenen F arvoa omaavaa muuttujaa.

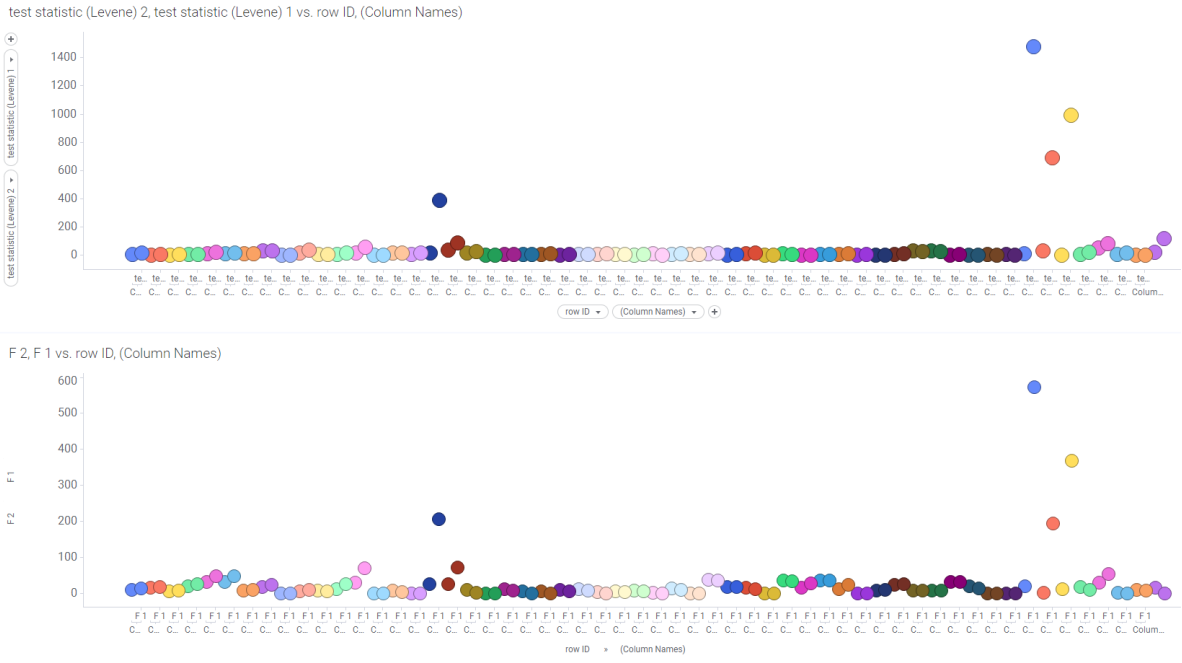
Prosessimuuttuja	F	W
40	37,12	9,87
46	36,99	6,63
44	36,26	11,61
13	32,75	11,45
52	32,40	0,20
14	31,58	10,77
20	30,72	15,97
60	29,68	51,04
28	26,95	34,13
26	26,27	12,87

Taulukko 17 Vain toisen ongelman jälkeisen minuutin sisältävän Levenen testin suurimmat kymmenen W arvoa omaavaa muuttujaa

Prosessimuuttuja	F	W
60	29,68	51,04
28	26,95	34,13
51	8,01	33,00
57	2,53	32,63
50	7,93	31,95
16	17,82	30,06
Nopeussuhde	15,66	19,73
18	5,80	17,55
20	30,72	15,97
23	8,06	14,83

Kun verrataan tuotannon romahdusta edeltäneen minuutin ja tätä edeltäneen minuutin tuloksia, havaitaan merkityksellisyyttä kuvaavien F arvojen voimakas lasku. Viimeisellä minuutilla neljä ylivoimaisesti merkityksellisintä prosessimuuttujaa eivät ole edes kymmenen suurimman F-arvon joukossa tätä edeltävällä minuutilla. Tulos vahvistaa ennakkotiedon ongelman nopeasta synnystä. Koska minuutin resoluutio riittää erottamaan näiden kahden luokittelutavan erot, on oletettavissa, että kaikki ongelmatilanteet eivät näy välttämättä viimeiselläkään minuutilla.

Kuvassa 30 on esitetty taulukoiden 16 ja 17 mukaiset Levenen testin W arvot ja F-arvot kaikille prosessimuuttujille. Kuva kertoo yhdellä silmäyksellä muuttujien muuttumisesta ongelman syntyhetkeä edeltävillä kahdella minuutilla. Vierekkäiset saman väriset merkit vastaavat muuttujien F-arvoja ja Levenen testin arvoja W viimeisellä ja sitä edeltävällä minuutilla. Taulukoissa 18 ja 19 on esitetty suurimman kymmenen F-arvon mukaan järjestetyt ANOVAn tulokset, kun aikaikkunana toimii 2- ja 5 minuuttia.



Kuva 30 Levenen testin W-arvot (yllä) ja ANOVAn F-arvot (alla) kullekin muuttujalle muuttujakohtaisesti värikoodattuna, kun käytetään ongelmatilannetta edeltävää minuuttia (oikean puoleinen samanvärisistä merkeistä) tai tätä edeltävän minuutin arvoa (vasemman puoleinen samanvärisistä merkeistä).

Taulukko 18 Kahden minuutin aikaikkunalla saadut ANOVAn tulokset F-arvon mukaan järjestettyinä.

Prosessimuuttuja	F	W
56	432,48	993,02
58	274,96	674,13
26	200,17	300,38
57	133,86	471,60
20	98,11	72,37
28	95,45	122,92
60	82,55	135,19
13	81,62	30,04
14	78,65	28,49
40	72,34	27,94

Taulukko 19 Viiden minuutin aikaikkunalla saadut ANOVAn tulokset F-arvon mukaan järjestettyinä.

Prosessimuuttuja	F	W
56	259,96	578,86
40	181,72	65,31
13	173,43	84,96
20	169,69	152,17
14	167,38	81,43
58	162,19	326,07
46	161,04	36,16
44	159,66	41,70
26	158,67	345,20
52	137,09	5,18

Tarkkailtaessa yhtä minuuttia suurempaa aikaikkunaa F-arvojen havaitaan pienenevän. Tämä on myös taulukoiden 16 ja 17 vertailun perusteella pääteltävissä, sillä merkitsevimpien prosessimuuttujien havaitaan katoavan ongelmatilanteen määrityksen muuttuessa. Kahden ja viiden minuutin arvot kuitenkin osoittavat, että viimeisellä minuutilla esiintyvät luokkien väliset erot näkyvät ANOVAssa huomattavasti voimakkaammin kuin muut erot.

Itse ANOVA ei havaitse merkittäväksi muuttujaksi tiedettyä nopeussuhdetta, mutta Levenen testi kertoo nopeussuhteen luokkien sisäisten varianssien eron olevan merkittävä. ANOVAn yhden minuutin aikaikkunan viisi ensimmäistä arvoa tosin liittyvät välillisesti nopeussuhteeseen, niin että prosessimuuttujat 56, 58 ja 57 ovat vaikuttavia tekijöitä

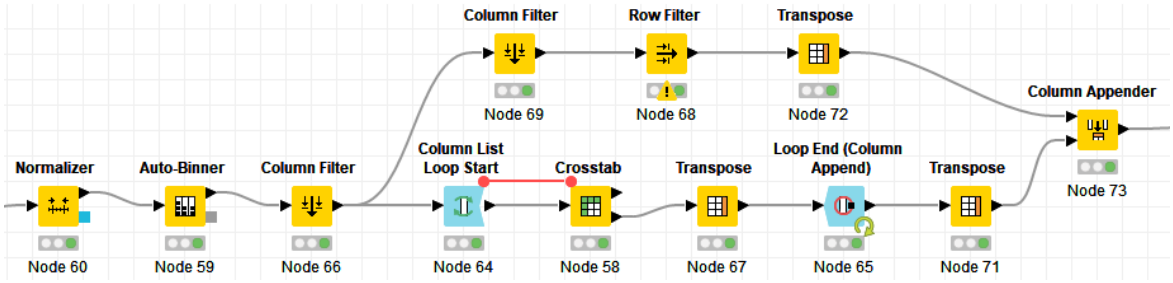
nopeussuhteessa, ja nopeussuhde puolestaan aiheuttaa osittain muuttujan 26 arvon. Muuttuja 28 on muuttujan 26 minuutin maksimiarvo.

ANOVAn ja Levenen testin voidaan todeta ohjaavan oikeaan suuntaan juurisyyanalyysissa. Testit ovat erittäin helposti toteutettavaissa valtavallekin määrälle muuttujia. Testin antamien tuloksien perusteella voidaan perehtyä merkitsevimpiin muuttujiin, ja tätä kautta päästä juurisyyn jäljille. Testien suurimmat arvot saaneet prosessimuuttujat siis käyttäytyvät käytännössä merkitsevästi eri tavoin viimeisellä minuutilla ennen ongelman syntymistä verrattuna muuhun normaaliin toimintaan. Tapauksesta riippuen tämä muutos voi kertoa joko muuttujan suorasta tai välillisestä vaikutuksesta ongelmatilanteen syntyyn tai ongelman synnyttävän muuttujan vaikutuksesta kyseiseen muuttujaan. Yhdistämällä henkilöstön prosessituntemus datasta saatuun tietoon pystytään tehokkaasti löytämään välillisten muuttujien perusteella todellinen juurisyy. Tutkitussa tapauksessa suuri osa juurisyytä oli kuvatun nopeussuhteen muuttuminen. Nopeussuhde ei kuitenkaan ole löydettävissä alkuperäisestä datasta, vaan yhteys nopeussuhteeseen saadaan yhdistämällä tieto prosessimuuttujien 56, 58, 57 ja 26 merkityksellisyydestä prosessituntemukseen.

Ajotilanteiden määrittely ”oikein” on kriittistä tulosten kannalta, kuten taulukoiden 16 ja 17 vertailu osoittaa. Nopeasti esiin tulevassa ongelmassa jo minuutilla eroava määritys voi ratkaista näkyykö ongelma tilastollisten testien perusteella vai ei.

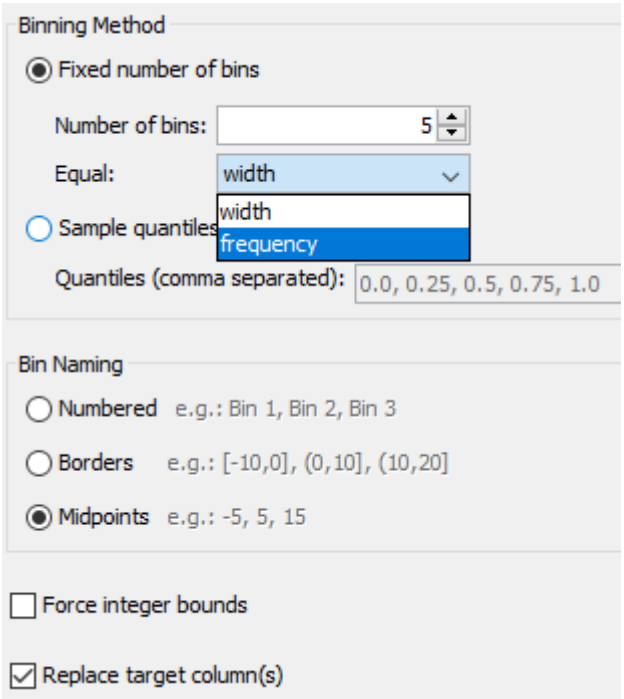
7.2.3 χ^2-testi Ajotilanteen ja ominaisuuksien välillä

χ^2-testin periaate tapauksen tulkinnassa on pitkälti sama kuin ANOVAlla. Testin tulokset kertovat luokkien välisistä eroista. Testin laskenta aloitetaan, kun kuvasta 27 yksiköstä Row Filter lähtevä tietovirta etenee χ^2 -testiin. χ^2 -testin laskentaan käytetty työnkulku on esitetty kuvassa 31.



Kuva 31 χ^2 -testin laskemiseen käytetty työnkulku. Työn kulku alkaa arvojen normalisoinnista ja lokeroinnista ja etenee itse testin kautta (Crosstab -yksikkö) tulosten keräämiseen.

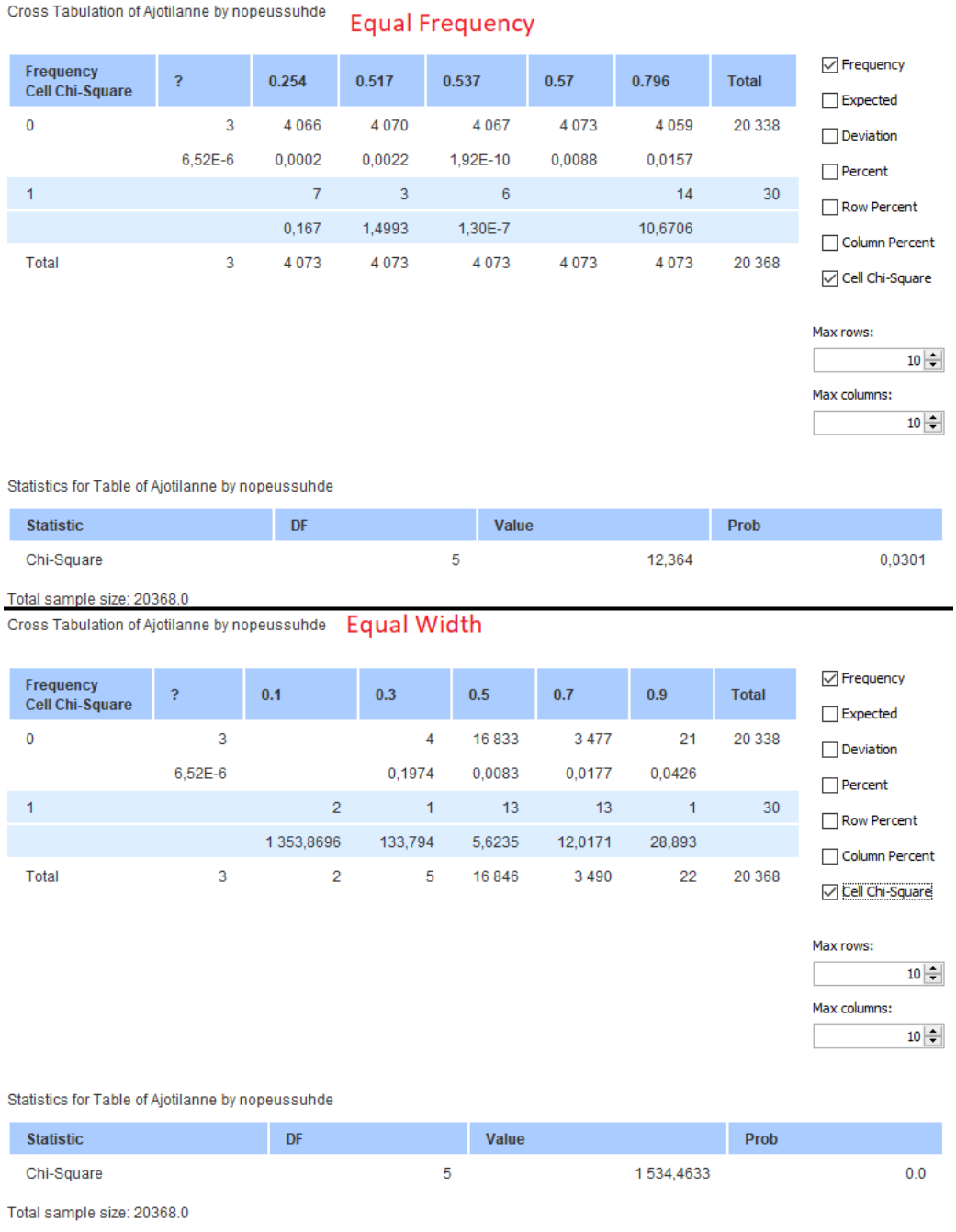
Työnkulun alussa kaikki tutkittavat prosessiparametrit normalisoidaan välille [0, 1] arvojen lokerointia varten. Lokerointi yksikössä Auto-Binner voidaan tehdä automaattisesti usealla eri perusteella, jotka on esitetty kuvassa 32.



Kuva 32 KNIMEn automaattisen lokeroinnin vaihtoehdot.

Lokerointi on kenties tärkein osa testin konfigurointia, sillä epätasainen luokkajako vaikuttaa testissä huomattavasti. Jos lokeroita valitaan liian paljon, korostuvat satunnaiserot Ajotilanne 1:ssä. Liian vähäinen lokerointi antaa kuitenkin vain karkeita tuloksia. Testien lokeroiden määräksi valittiin melko maltillinen arvo, 5 lokeroa.

Myös valinta tasaisen lokeroleveyden ja taajuuden välillä luo erittäin huomattavia eroja. Tasainen lokeron leveys antaa tilanteessa merkittävämpiä arvoja, joten se valitaan käyttöön. Vertailu nopeussuhteen merkityksellisyydestä yhden minuutin Ajotilanne 1 asetuksella tasaisella leveydellä ja taajuudella on esitetty kuvassa 33. Kuvan tuloksissa lähtödataa ei ole suodatettu tuotannon määrän perusteella.



Kuva 33 Ero tasaisen lokeroleveyden ja -taajuuden välillä χ^2 -testissä.

Lasketut χ^2 -arvot ovat tasaisella taajuudella 12,36 ja tasaisella leveydellä 1534,46. Tasaisen taajuuden tapauksessa nopeussuhde ei siis ole ilmiselvästi merkitsevä tekijä luokkien välillä. Jakauman merkityksellisyyden p-arvo on 0,03 eli vain hieman matalampi kuin yleisenä pidetty raja 0,05. Tasaisella leveydellä merkityksellisyys on sen sijaan ilmiselvää. Ero aiheutuu suureksi osaksi luokkasuhteesta, sillä Ajotilanne 1 sisältää huomattavan paljon äärimmäisiä arvoja, mutta lokerojen rajat määräytyvät tasaisella taajuudella pääasiassa vallitsevan luokan Ajotilanne 0 suhteen.

Lokeroinnin jälkeen valitaan testiin halutut prosessimuuttujat eli tämän testin tapauksessa kaikki. Työnkulku jakautuu valinnan jälkeen kahteen osaan, alapuolella tehtävään testiin ja yläpuolella muuttujien nimien muistiin ottamiseen. Yläosan toiminta ei ole laskujen kannalta relevanttia, vaan sen tarkoitus on yhdistää prosessimuuttujien nimet kerättyihin tuloksiin.

Alaosassa lasketaan χ^2 -testi yksikössä Crosstab. Testi ajetaan loopissa, joka käy läpi jokaisen sarakkeen ja laskee testin arvot Ajotilanteen suhteen. Loopin lopussa rivimuodossa annetuille testin arvoille suoritetaan transpoosi ja tiedot kerätään sarakkeena talteen. Loopin jälkeen arvot muutetaan takaisin riveihin ja yhdistetään vastaaviin muuttujien nimiin. Testitulokset muutetaan vielä anonyymeiksi keräämisen jälkeen numeroimalla ne.

Taulukot 20-22 sisältävät χ^2 -testin tuloksien yhteenvedon. Taulukot sisältävät kymmenen suurimman χ^2 -arvon omaavan prosessimuuttujan tulokset 1, 2 ja 5 minuutin Ajotilanne 1 -ikkunalla. Käytännössä nämä kymmenen muuttujaa ovat tilanteissa siis epätodennäköisimmin jakautuneet satunnaisesti mitatunlaisiin lokeroihin. Toisin sanoen nämä ominaisuudet muuttuvat testin mukaan eniten siirryttäessä viimeisistä ongelmaa edeltäneistä minuuteista muihin normaalin toiminnan arvoihin. Testissä on käytetty vain ongelmatilanteista maksimissaan ±0.5 % etäisyydellä olevia tuotantotilanteen arvoja.

Taulukko 20 Yhden minuutin Ajotilanne 1 -ikkunalla lasketut merkittävimmät prosessimuuttujat.

Prosessimuuttuja	χ^2
58	2880,46
56	2671,38
26	1878,25
57	1339,47
Nopeussuhde	854,28
28	400,37
47	300,49
45	285,52
25	273,29
31	143,66

Taulukko 21 Kahden minuutin Ajotilanne 1 -ikkunalla lasketut merkittävimmät prosessimuuttujat.

Prosessimuuttuja	χ^2
58	2311,47
56	1851,60
26	1165,36
57	814,26
Nopeussuhde	811,96
28	371,38
31	338,11
54	330,54
55	329,43
47	311,96

Taulukko 22 Viiden minuutin Ajotilanne 1 -ikkunalla lasketut merkittävimmät prosessimuuttujat.

Prosessimuuttuja	χ^2
58	1502,85
56	1271,44
26	824,12
25	571,68
13	505,49
14	467,20
0	437,43
28	435,50
57	420,19
Nopeussuhde	414,83

Kolmen merkitsivimmän muuttujan joukko pysyy samana läpi eri testien. Nopeussuhteen merkitys heikkenee suhteessa muihin muuttujiin tarkkailtessa pidemmällä kuin kahden

minuutin aikaikkunalla. χ^2 -arvo kuitenkin laskee kaikilla merkityksellisimmillä muuttujilla, kun suurempi aikaikkuna otetaan huomioon.

Aiemman tutkimuksen perusteella nopeussuhteen tiedetään olevan yksi merkitsevimmistä tekijöistä. Sille tehtyjen toimenpiteiden avulla prosessin alasajojen määrä väheni huomattavasti. Kolme ensimmäistä tekijää liittyvät kaikki nopeussuhteeseen välillisesti. Nopeussuhde aiheuttaa osittain muuttujan 26 arvon, ja muuttujat 58 ja 56 puolestaan aiheuttavat osittain nopeussuhteen.

Saadut χ^2 -arvot on mahdollista muuttaa todennäköisyyksiksi, joilla mitatunlaiset lokeroinnit saavutetaan satunnaisella jakautumisella. Arvojen muuntaminen ei tapauksessa kuitenkaan ole erityisen käytännöllistä tai havainnollistavaa, sillä merkittävimmät prosessimuuttujat saavat satunnaisen jakautumisen todennäköisyyden arvoja $p < 10^{-100}$. Koska lokerointi tehtiin kaikille muuttujille samalla tavoin, on testin vapausaste sama kaikilla muuttujilla. Näin ollen χ^2 -arvot ovat kääntäen verrannollisia satunnaiseen jakautumiseen, ja järjestys pysyy samana (joskin käänteisenä), vaikka χ^2 -arvot muunnettaisiin todennäköisyyksiksi.

Saatujen χ^2 -arvojen perusteella ei voida kuitenkaan periaatteen tasolla sanoa kyseisten muuttujien suoraan aiheuttavan ongelmia. Arvojen perusteella voidaan kuitenkin lähteä tutkimaan tiettyjä prosessimuuttujia ja niiden yhteyksiä ongelmatilanteisiin. Kun tarkkaillaan näihin muuttujiin liittyviä korrelaatio- ja kausaatiosuhteita, pystytään etenemään kohti todellista juurisyytä.

χ^2 -testi ei välttämättä ole hyödyllinen tapauksessa, sillä ANOVA ja muut menetelmät tuottavat käytännössä samaa tietoa pienemmällä vaivalla (ilman lokerointia). Jos tutkittava data kuitenkin sisältää valmiiksi lokeroituja arvoja tai kategorisia muuttujia, hyötyy testistä todennäköisemmin.

Testi on myös vähäisen ongelmadatan määrän vuoksi ongelmallinen. Esimerkiksi suodattamalla muut kuin lähellä (±0.5 %) ongelmatuotantoa olevat pisteet ja poistamalla viimeinen prosessin alasajoa edeltänyt arvo laskennasta ja luokittelemalla sitä edeltänyt minuutti arvolla 1 saadaan aikaan tilanne, jossa nopeussuhde on ylivoimaisesti merkittävin muuttuva tekijä χ^2 -arvolla 320,79. Yhden normaalin toiminnan ulkopuolella olevan pisteen nähdään kuitenkin aiheuttavan tästä arvosta 293,30 eli lähes kaiken.

7.2.4 Yhteenveto tilastollisista analyyseista

Kaikki käytetyt tilastolliset menetelmät viittaavat samaan, oikeaan suuntaan ongelman juurisyytä etsittäessä. Analyysit eivät välttämättä tilastollisessa mielessä ole valideja. Menetelmät perustuvat kuitenkin vastaavassa tapauksessa saatujen tilastollisten arvojen suuruusluokkiin ja järjestyksiin tarkkojen arvojen sijaan, minkä vuoksi menetelmät ovat tilanteessa hyödyllisiä ja antavat ongelmanratkaisijalle suuntaviivoja tai suoria vastauksia etenemiseen.

Käytettyjen menetelmien lisäksi vastaavaan tilanteeseen voitaisiin soveltaa useita muita menetelmiä. Suurimmassa osassa, ellei kaikissa muissa menetelmissä ilmenevät kuitenkin samat ongelmat epätasapainoisen luokkajaon vuoksi.

7.3 Ongelmatilanteiden luokittelu

Teollisuustapauksen ongelmatilanteiden ennustamiseen sovelletaan luokittelua, joka kuuluu koneoppimiseen. Ongelmatilannetta edeltävän minuutin havaitseminen luokittelulla mahdollistaisi automaattisen järjestelmän tai prosessinhoitajien reagoinnin syntyvään ongelmatilanteeseen.

Ongelmatilanteiden luokittelu ei ole tutkittavana olevan aikasarjadatan kanssa täysin yksiselitteistä. Vaikka ongelmatilanteissa onkin samoja piirteitä, joita koneoppimisella voidaan mallintaa, vaihtelee ongelmatilannetta edeltävien prosessimuuttujien muutosten esiintuloaika niin, että harva tapaus on havaittavissa kahta minuuttia ennen ongelmatilannetta ja useat tapaukset havaittavisssa joko yhtä minuuttia ennen tai ei lainkaan. Näin ollen luokittelijoiden tapauksessa ainoastaan viimeinen minuutti ennen tuotannon romahdusta luokitellaan ongelmatilanteeksi. Käytännössä tämä tarkoittaa mallin luovan yhden minuutin ennustuksen. Menettely takaa alustavan realistisen luokittelutuloksen, sillä luokista otetaan tasaisesti 80 % opetusdataan ja 20 % testidataan. Jos esimerkiksi jokaista ongelmatilannetta viisi edeltävää minuuttia olisi luokiteltu ongelmatilanteeseen, käsittelisi satunnaisnäytteenotolla opetusdata usein vain osaa näistä viidestä ja testidatassa luokittelija luokittelisi jäljelle jäävät ongelmatilanteisiin epärealistisella tarkkuudella.

Luokittelun tuloksia tarkkaillessa otetaan kuitenkin huomioon myös romahdusta edeltäneet kymmenen minuuttia. Tämä tehdään varsinaisen luokittelun jälkeen tarkastelemalla False Positive -luokittelujen todenmukaisuutta. Alla oleva alustavasta KNIMEllä tehdystä Random Forest -mallinnuksesta saatu sekaannusmatriisi taulukossa 23 ja tulokset taulukossa 24 demonstroivat menettelyn tarpeellisuutta.

Taulukko 23 Sekaannusmatriisi Random Forest -algoritmin alustaville tuloksille. Taulukko ja muut vastaavat taulukot on generoitu KNIMEssä.

	Ennustus: 0	Ennustus: 1
Todellisuus: 0	4062	6
Todellisuus: 1	4	2

Kuten matriisista nähdään, oikein ennustettuja ongelmatapauksia (TP) malli löysi kaksi. Samaan aikaan matriisin mukaan kuudessa tapauksessa ennustettiin ongelma, vaikka sitä ei todellisuudessa ollut (FP). Neljä tapausta jäi matriisin mukaan havaitsematta (FN). Taulukko 24, johon on poimittu kaikki ongelmatilanteet, ongelmatilanteeksi ennustetut arvot sekä tieto siitä, onko kymmenen minuutin sisällä tapahtumassa ongelmatilanne, kuitenkin kuvaa luokittelun onnistumista hieman eri näkökulmasta.

Taulukko 24 Ennustusten vertailu ajotilanteeseen sekä edeltäviin kymmeneen minuuttiin.

DateTime	Ajotilanne	Ennustettu ajotilanne	Ajotilanne (10 min)
X - 09:30	0	1	0
Y - 01:18	0	1	0
Y - 01:30	0	1	1
Y - 04:40	1	0	1
Y - 05:59	0	1	1
Y - 20:33	1	0	1
Z - 12:32	0	1	1
Z - 12:33	0	1	1
Z - 12:34	1	1	1
A - 03:05	1	0	1
A - 06:24	1	0	1
B - 15:07	1	1	1

Kun tulkitaan taulukkoa 24, havaitaan, että sekaannusmatriisin kuudesta FP arvosta vain kahta ei seurannut todellinen ongelma seuraavan kymmenen minuutin aikana. Näitä kahta

arvoa tarkemmin tulkitsemalla havaitaan, että Y – 01:18 aikaleimalla olevaa arvoa seurasi ongelmatilanne 13 minuutin kuluttua, mikä havaitaan myös seuraavan rivin arvoista. Aikaleimaa X – 09:30 sen sijaan ei seurannut ongelmatilanne lähiaikoina. Näin ollen kuudesta FP -arvosta vain yksi ei todellisuudessa edeltänyt ongelmaa. Käyttäen kymmentä minuuttia rajana, luku on kaksi. Todellisessa tarkkailussa on tietenkin käytettävä kymmenen minuutin kaltaista kovaa rajaa ”korjatun” matriisin rakentamisessa automaattisesti. Arvojen, jotka ovat taulukossa 24 tilanteessa [0, 1, 1] voitaisiin siis katsoa olevan TP -tapauksia, mikäli seuraavan kymmenen minuutin aikana on tapaus [1, 0, 1]. Näistä yksi luokiteltaisiin TP -kategoriaan, muut suodatettaisiin. Kyseisenlaista tapausta ei yllä olevassa esimerkissä ole, mutta menettelyä olisi mahdollista soveltaa käytännössä. Muissa tapauksissa [0, 1, 1] arvot ennustavat opetusdatan ongelmaa tai oikein havaittua ongelmaa, eli ne eivät oikeastaan täytä TP:n eivätkä FP:n määritelmää. Nämä arvot on siis käytännössä suodatettava pois.

Kun esitetyn mukaiset käsittelyt on tehty sekaannusmatriisille taulukossa 23, näyttää uusi matriisi taulukon 25 mukaiselta.

Taulukko 25 Käsitelty sekaannusmatriisi alustavalla tulosdatalla.

	Ennustus: 0	Ennustus: 1
Todellisuus: 0	4062	2
Todellisuus: 1	4	2

Kokeet tehdään kuitenkin ristivalidointia hyödyntäen, jolloin TP -arvojen lisääminen on tietyissä tapauksissa ongelmallista. Näin ollen kokeissa ajetaan tulokset kaikella datalla, minkä jälkeen ehdot [0, *, 1] täyttävät pisteet poistetaan ennen pisteytystä. Tämä tarkoittaisi yllä olevan matriisin tapauksessa TN -tapausten määrän pienenemistä – muut arvot pysyvät taulukon 25 mukaisina.

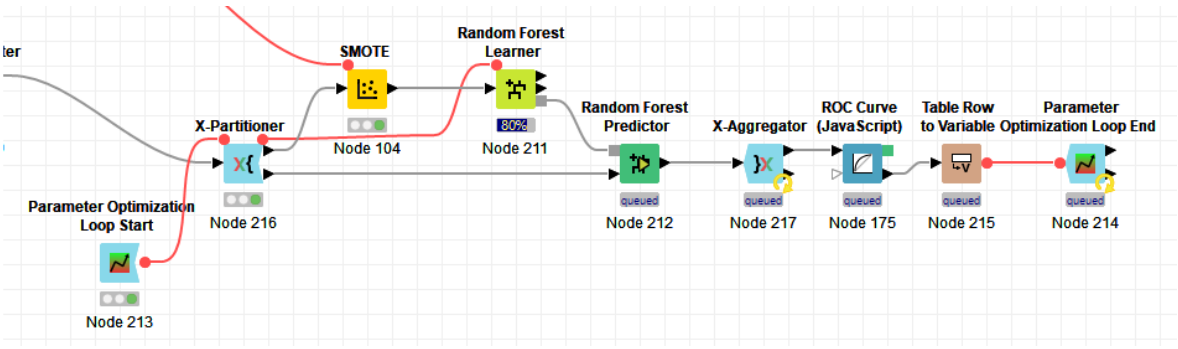
7.3.1 Testijärjestelyt

Luokittelua hyödyntäessä parametri-, muuttujanvalinta- ja esikäsittelyvaihtoehtoja on useita algoritmivalintojen lisäksi. Algoritmeista tarkkaillaan logistista regressiota, päätöspuuta, Random Forestia, Naiivia Bayesta ja neuroverkkoa.

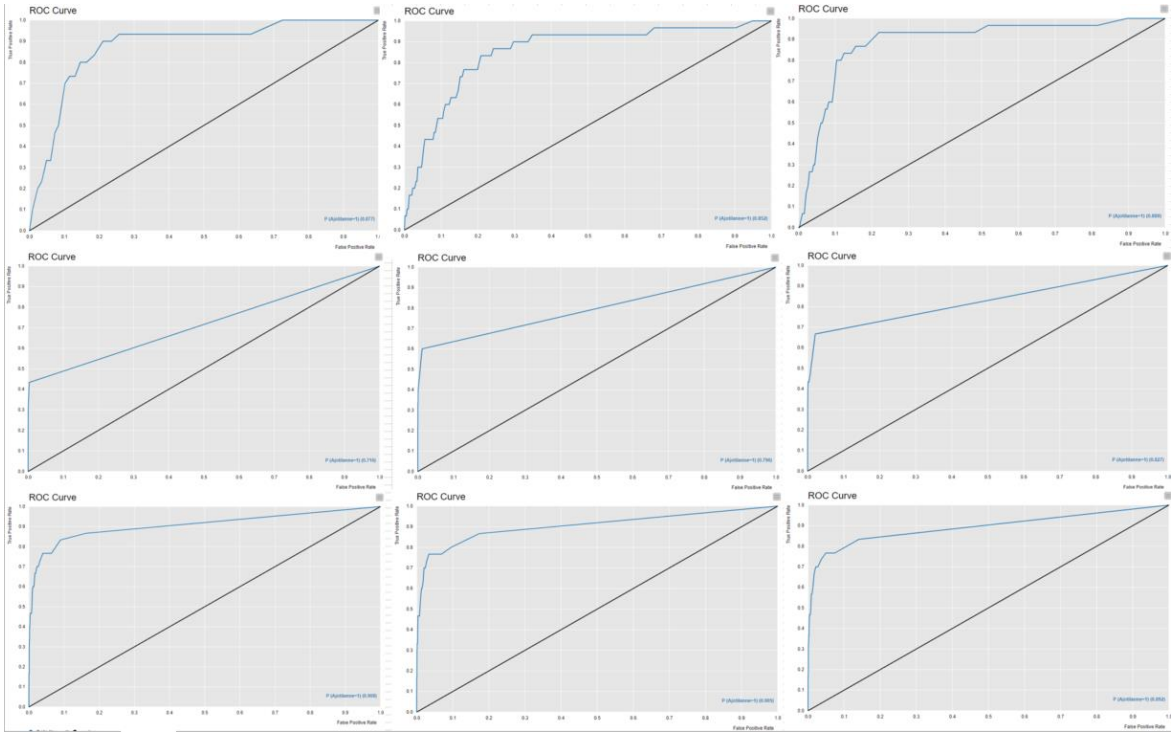
Muuttujanvalinta suoritetaan kolmella menetelmällä: valitsemalla kymmenen suurimman F- ja Levenen testin arvon omaavat prosessimuuttujat, redusoimalla muuttujat kymmeneen PCA-komponenttiin ja käyttämällä kaikkia muuttujia.

Kaikissa testeissä hyödynnetään SMOTE:a lähimpien naapurien määrää (kNN) lukuun ottamatta identtisillä asetuksilla opetusdatan suhteen. Ongelmatilanteiden määräksi generoidaan sama kuin normaalien tilanteiden määräksi käyttäen arvoa seed = 4606073459731165513.

SMOTEa hyödynnetään sen avulla parantuneen luokittelutuloksen vuoksi. Alustavissa testeissä ilman SMOTEa tehdyt luokittelut saivat optimoidut Area Under Curve (AUC) -arvot 0,859, 0,732 ja 0,811. Kymmenen lähimmän naapurin SMOTElla tehdyt luokittelut saivat optimoidut AUC -arvot 0,908, 0,904 ja 0,902. Ottamalla 24 näytettä molemmista luokista saavutetaan melko hyviä AUC -arvoja, mutta täsmällisyys on heikko verrattuna SMOTElla saatuihin tuloksiin. Alustavassa optimoinnissa muutettiin Random Forestin puiden määrää välillä 2-150 ja haarojen minimikokoa välillä 2-20. Optimoinnissa käytettiin ”Hill climbing” etsintästrategiaa, minkä vuoksi tulokset vaihtelevat hieman. Muuttujina käytettiin kymmenen suurinta F- ja Levenen testin arvoa omaavaa muuttujaa. Lopullisissa vertailuissa SMOTEn lähimpien naapurien määrä optimoidaan kunkin algoritmin tulosten avulla. Alustavien testien suorittamiseen käytetty työnkulku on esitetty kuvassa 34. Kolme esimerkkikäyrää kullakin metodilla (SMOTE, ei SMOTEa, 24/24) on esitetty kuvassa 35.



Kuva 34 SMOTEn hyödyllisyyden määrittämiseen käytetty työnkulku. SMOTE -yksikkö poistettiin ilman SMOTEa tehdyissä testeissä ja vaihdettiin Equal Size Sampling -yksikköön tarpeen vaatiessa. Työnkulku sisältää parametrien optimointiloopin sekä ristivalidointiloopin, minkä lisäksi työnkulussa rakennetaan Random Forest -malli sekä pisteytetään sen tulokset. Mallin rakennuksessa hyödynnetään SMOTEa.



Kuva 35 Esimerkki ROC -käyrät 24(0)/24(1) opetusdatalla (ylimpänä), ilman SMOTEa epätasapainoisella datalla (keskellä) ja SMOTEn avulla tasapainotetulla datalla (alimpana).

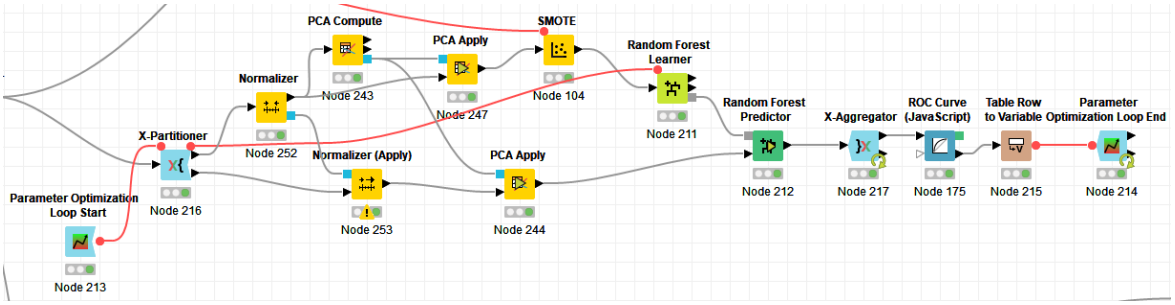
Kuvan 35 ROC -käyriä tulkitsemalla havaitaan sekä SMOTE:lla suoritettujen luokittelujen että esikäsittelemättömällä datalla tehtyjen luokittelujen ROC -käyrien nousevan alussa lähes pystysuoraan. Pystysuoraan nousu on aina ihanteellista, sillä tämä tarkoittaa suurta True Positive/False Positive arvojen suhdetta. 24(0)/24(1) näytteenotolla käyrä lähtee välittömästi viettämään oikealle, mikä on haitallista varsinkin epätasaisen luokkajaon tapauksessa. SMOTE:lla tehtyjen luokittelujen havaitaan olevan jatkuvasti alussa melko pystysuoria, minkä lisäksi ne nousevat keskimäärin korkeammalle kuin ilman esikäsittelyä tehtyjen luokittelujen tapauksessa.

Kaikki testit suoritetaan samoilla datanjakoasetuksilla: data jaetaan viiteen yhtäsuureen osaan molempien luokkien suhteen, ja tulokset lasketaan kullekin osalle ja summataan. Viittä osaa käytetään 80/20 jaon saavuttamiseksi. Testeissä käytetään vakiojakoa algoritmien välisen tasapuolisuuden vuoksi. Jako tehdään KNIMEn X-Partitioner -yksiköllä käyttäen asetuksia Stratified sampling ja seed = 1557749525429.

Parametrit optimoidaan kuvan 34 mukaisella järjestelyllä. Ensimmäisellä optimointikierroksella optimoidaan luokittelijan optimoitavat parametrit. Toisella

optimointikierroksella optimoidaan SMOTEn lähimpien naapurien määrä (tulevissa taulukoissa esitetty nimellä kNN, k-nearest neighbors) edellisen optimoinnin parhaimpien parametrien arvoilla. Optimointi ei ole täydellinen, mutta optimoidut arvot antavat huomattavasti parempia luokittelutuloksia (korkeampi AUC -arvo, parempi täsmällisyys) kuin perusarvoilla saadaan. Perusarvot ovat saatavilla KNIMEn versiossa 3.7.2 (18.4.2019) asettamalla haluttu yksikkö työnkulkuun ja tarkastamalla ne.

PCA:lla tehtävän muuttujanvalinnan tapauksessa työnkulkuun lisätään muutama yksikkö normalisointia ja PCA:n komponenttien laskemisesta varten kuvan 36 mukaisesti.



Kuva 36 Optimointityönkulku PCA:ta hyödynnettäessä. Kuvaan 34 työnkulkuun lisätyt yksiköt (Normalizer, PCA Compute, Normalizer Apply, PCA Apply) normalisoivat datan ja laskevat kymmenen ensimmäistä PCA-komponenttia, joita hyödynnetään mallinnuksessa.

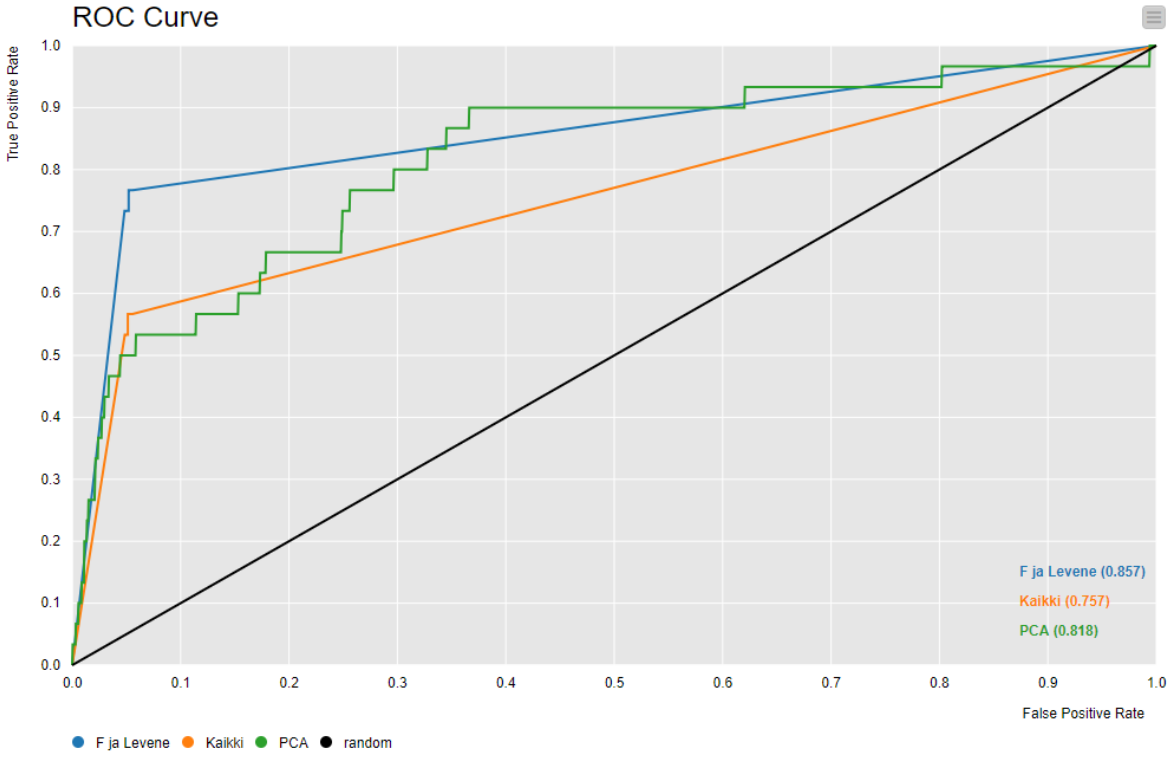
Seuraavissa luvuissa esitellään kuvien 34 ja 36 mukaisilla työnkuluilla optimoitujen luokittelijoiden tulokset eri algoritmeilla.

7.3.2 Logistinen regressio

Logistisen regression tapauksessa optimoitava parametri on oppimisnopeus, learning rate. Learning raten ja SMOTEn optimoinnin seurauksena saatiin kullekin muuttujatapaukselle optimoidut arvot, jotka on esitetty taulukossa 26. Kuvassa 37 on esitetty taulukon 26 mukaisilla parametreilla tehtyjen luokittelujen ROC-käyrät.

Taulukko 26 Logistisen regression optimoidut parametrit ja AUC -arvot eri muuttujan valinnoilla.

Muuttujat	Learning rate	kNN	AUC
Top 10 F/Levene	0,91	10	0,855
Kaikki ominaisuudet	0,01	10	0,754
10 PCA	0,41	2	0,814

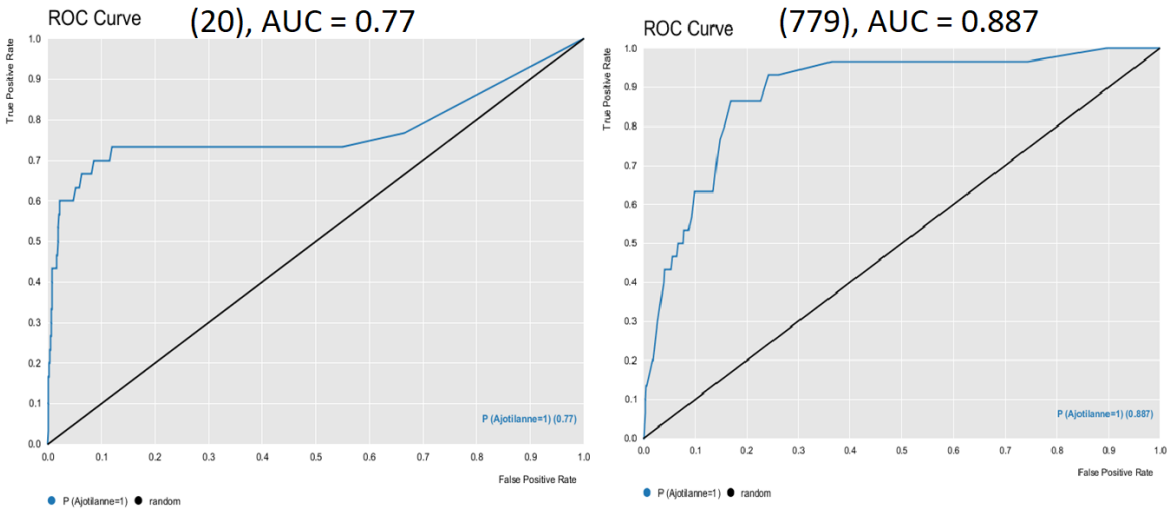


Kuva 37 Eri muuttujanvalinnoilla tehtyjen logistinen regressio -luokittelujen ROC -käyrät. Sininen käyrä kuvaa ANOVA:n F-arvoihin ja Levenen testin arvoihin perustuvan muuttujan valinnan avulla tehdyn luokittelun ROC -käyrää. Oranssin käyrä on kaikkiin muuttujiin perustuvan luokittelijan ROC -käyrä. Vihreän käyrän tapauksessa datasetille on suoritettu PCA (Principal Component Analysis), jonka kymmentä ensimmäistä komponenttia on käytetty luokittelijan rakentamiseen. Oikeassa alanurkassa näkyvät arvot kuvaavat kunkin luokittelijan AUC -arvoa.

Parhaiten rakennetuista luokittelijoista selviytyy F- ja Levenen testin W-arvojen perusteella tehdyn muuttujanvalinnan avulla rakennettu luokittelija. Mikään luokittelijoista ei kuitenkaan ole erityisen täsmällinen edes korkeilla todennäköisyyksillä. Tarkempi analysointi käyriin liittyen on esitetty luvussa 7.3.7.

7.3.3 Päätöspuu

Päätöspuun tapauksessa optimoidaan haarojen minimidatapistemäärää. Päätöspuun tapauksessa suurimmat AUC -arvot saadaan korkeilla minimidatapistemäärillä, mutta luokittelun täsmällisyys korkeammilla luottamusarvoilla kärsii huomattavasti, kuten kuvassa 38 on esitetty. Tämän vuoksi optimointi määritellään minimidatapistemäärävälille 2-20.

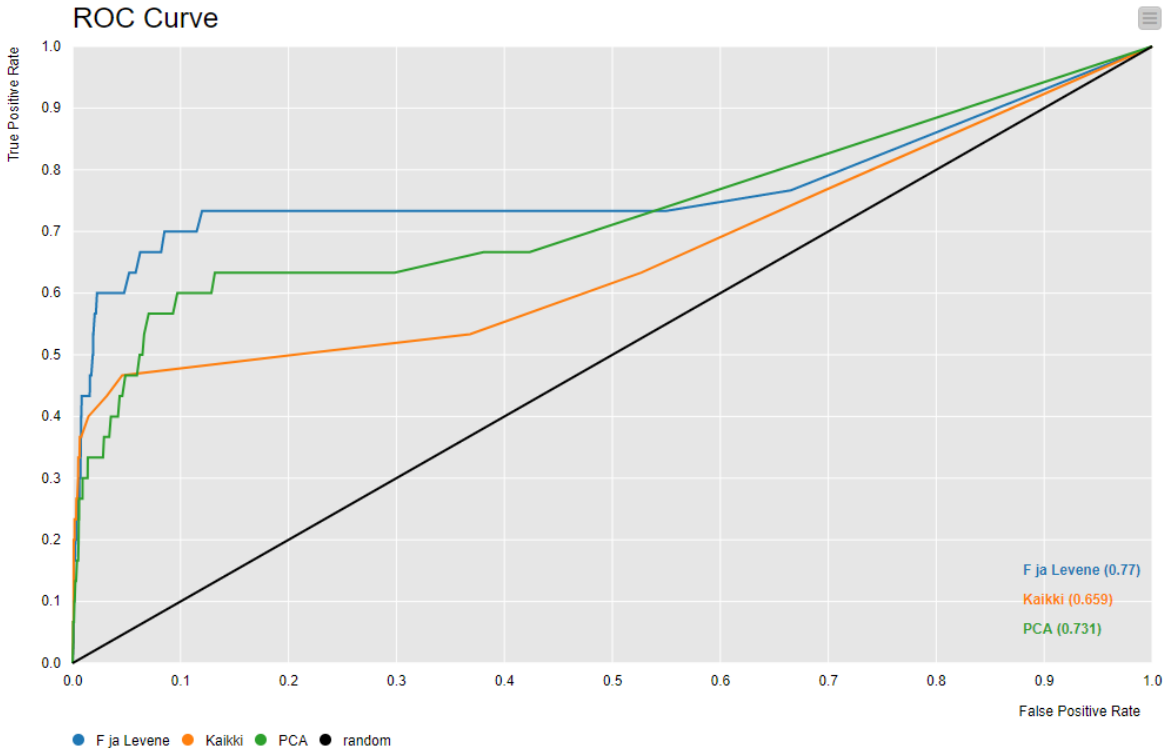


Kuva 38 Päätöspuuluokittelijan tulokset haarojen minimidatapistemäärän arvoilla 20 ja 779. Haarojen minimidatapistemäärän kasvaessa täsmällisyys kärsii korkeilla todennäköisyyksillä, vaikka AUC -arvo kasvaakin.

Optimoidut arvot eri muuttujanvalintatapauksissa on esitetty päätöspuulle taulukossa 27. Kuvassa 39 on esitetty taulukon 27 mukaisilla parametreilla tehtyjen luokittelujen ROC-käyrät.

Taulukko 27 Päätöspuun optimodut parametrit ja AUC -arvot eri muuttujan valinnoilla.

Muuttujat	Min. number of records per node	kNN	AUC
Top 10 F/Levene	15	17	0,769
Kaikki ominaisuudet	20	2	0,659
10 PCA	19	18	0,729



Kuva 39 Eri muuttujanvalinnoilla tehtyjen päätöspuuluokittelujen ROC -käyrät.

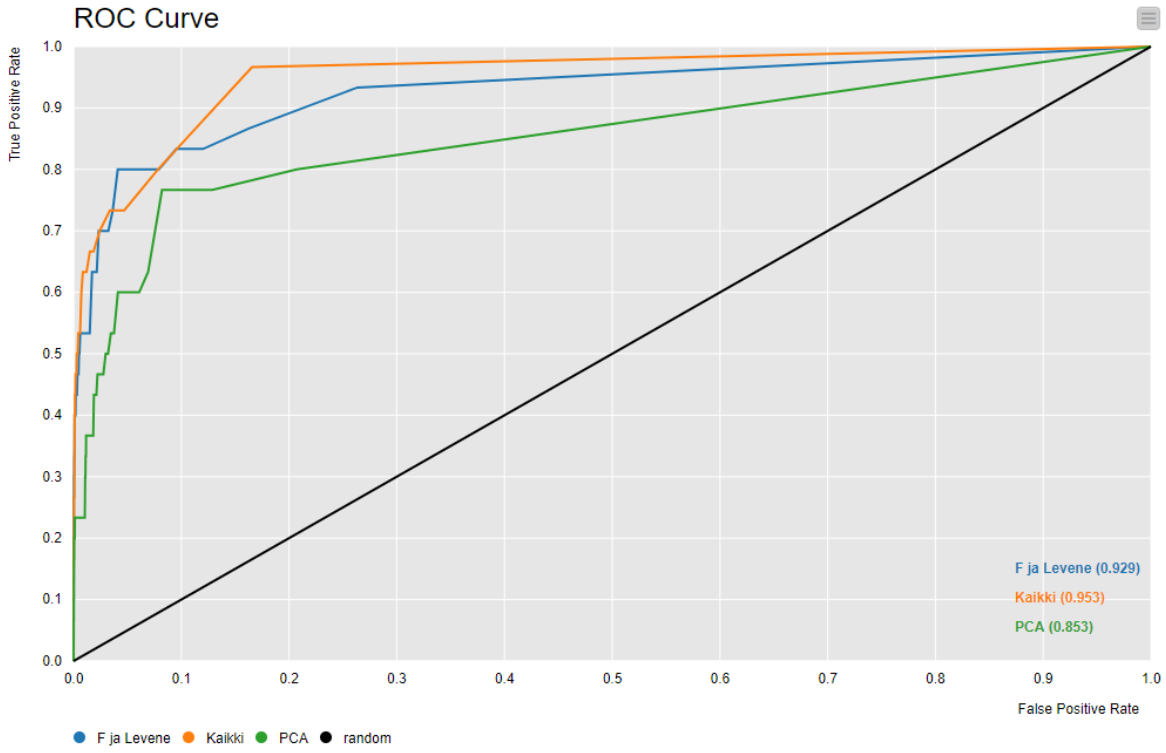
Käyriä ja niiden AUC -arvoja tarkkailtessa huomataan F- ja Levenen testin arvojen perusteella tehdyn muuttujanvalinnan tuottavan parhaita tuloksia. Tuotettu luokittelija on melko täsmällinen korkeilla todennäköisyyksillä, mutta täsmällisyys kärsii matalampiin todennäköisyyksiin mentäessä.

7.3.4 Random Forest

Random Forestin tapauksessa optimoidaan päätöspuun tavoin haarojen minimidatapistemäärää. Tämän lisäksi optimoidaan mallien eli puiden määrää. Optimoidut parametrit eri muuttujanvalintatapauksissa on esitetty taulukossa 28. Kuvassa 40 on esitetty taulukon 28 mukaisilla parametreilla tehtyjen luokittelujen ROC-käyrät.

Taulukko 28 Random Forest -algoritmin optimodut parametrit ja AUC -arvot eri muuttujan valinnoilla.

Muuttujat	NumTrees	MinNodeSize	kNN	AUC
Top 10 F/Levene	60	15	19	0,926
Kaikki ominaisuudet	102	6	18	0,949
10 PCA	142	5	24	0,881



Kuva 40 Eri muuttujanvalinnoilla tehtyjen Random Forest -luokittelujen ROC -käyrät.

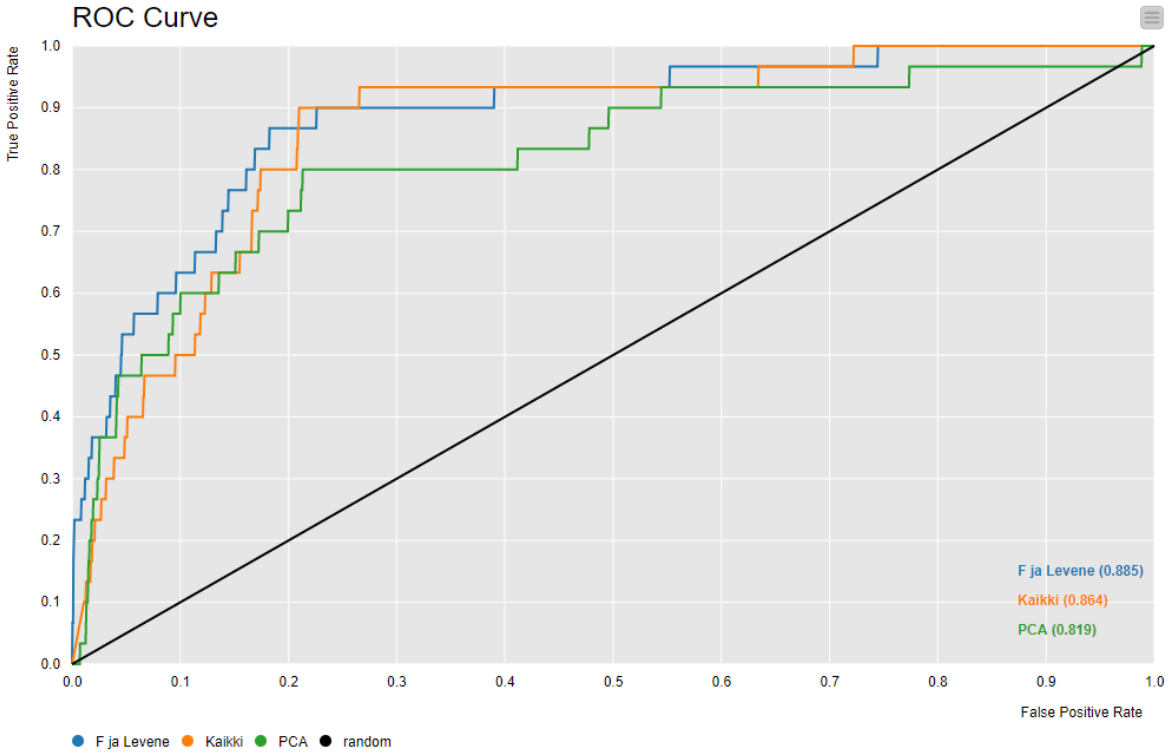
Kuvaa 40 ja sen AUC -arvoja tulkitsemalla voidaan todeta sekä kaikkien muuttujien käytön että F-arvon ja Levenen testin tuloksella tehdyn muuttujanvalinnan tuottavan hyviä luokittelutuloksia. Kaikkia muuttujia käyttämällä saadaan paras AUC -arvo. Täsmällisyys on hyvällä tasolla vielä melko suurellakin löytöasteella.

7.3.5 Naiivi Bayes

Naiivin Bayesilaisen luokittelijan yhteydessä ei optimoida itse algoritmin parametreja. Optimaaliset SMOTEn kNN arvot on esitetty taulukossa 29. Kuvassa 41 on esitetty taulukon 29 mukaisilla parametreilla tehtyjen luokittelujen ROC-käyrät.

Taulukko 29 Optimaaliset kNN arvot Naive Bayes -algoritmin eri muuttujanvalintatapauksissa.

Muuttujat	kNN	AUC
Top 10 F/Levene	19	0,882
Kaikki ominaisuudet	12	0,859
10 PCA	2	0,814



Kuva 41 Eri muuttujanvalinnoilla tehtyjen Naiivi Bayes -luokittelujen ROC -käyrät.

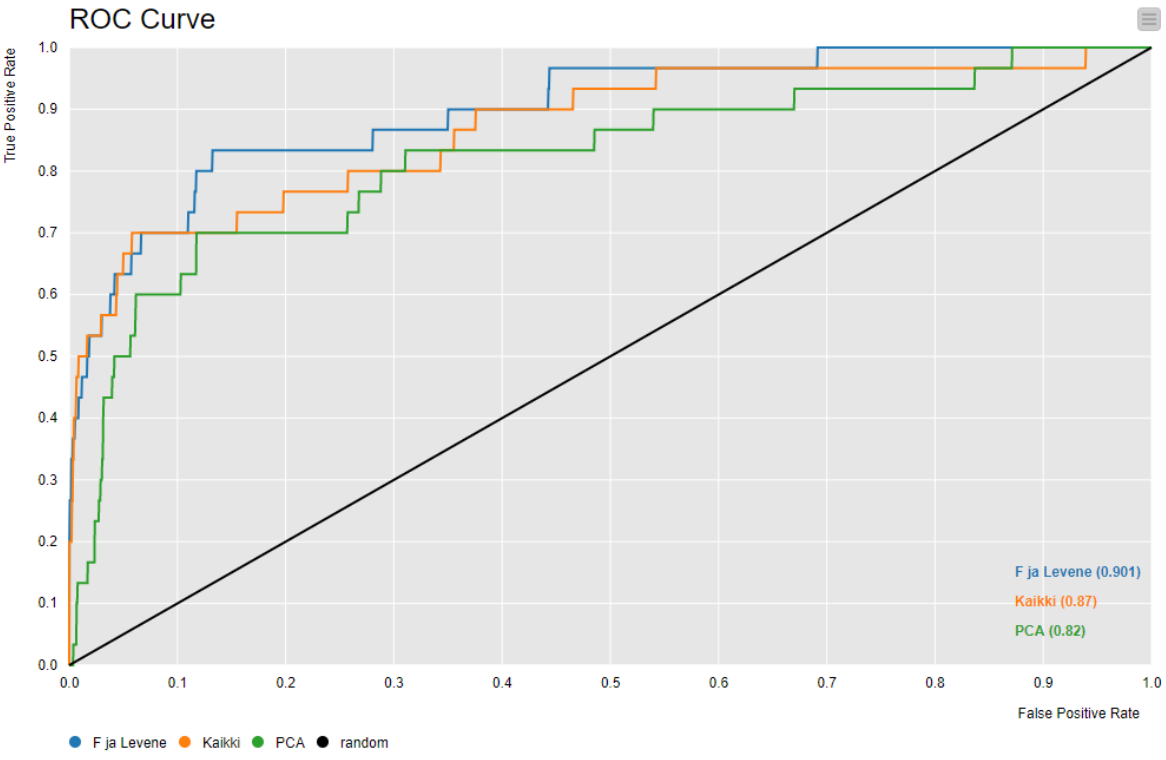
F-arvoilla ja Levenen testin arvoilla tehdyn muuttujanvalinnan havaitaan olevan paras vaihtoehto Naiivin Bayesilaisen luokittelijan tapauksessa. Täsmällisyys on hyvällä tasolla korkeilla todennäköisyyksillä, mutta kärsii nopeasti todennäköisyyksien pienetessä.

7.3.6 Neuroverkko

Neuroverkkona toimii testeissä KNIMEn RProp MLP (Multi-Layer Perceptron) Learner. Testeissä käytetään yhden piilotetun kerroksen verkkoa, jonka yksiköiden määrää optimoidaan. Taulukossa 30 on esitetty optimoidut yksiköiden määrät, SMOTEn kNN-arvot ja AUC -arvot neuroverkoille eri muuttujanvalinnoilla. Kuvassa 42 on esitetty taulukon 30 mukaisilla parametreilla tehtyjen luokittelujen ROC-käyrät.

Taulukko 30 Multi-Layer Perceptron -algoritmin optimodut parametrit ja AUC -arvot eri muuttujan valinnoilla.

Muuttujat	Num. Nodes	kNN	AUC
Top 10 F/Levene	5	20	0,881
Kaikki ominaisuudet	25	23	0,866
10 PCA	5	14	0,816



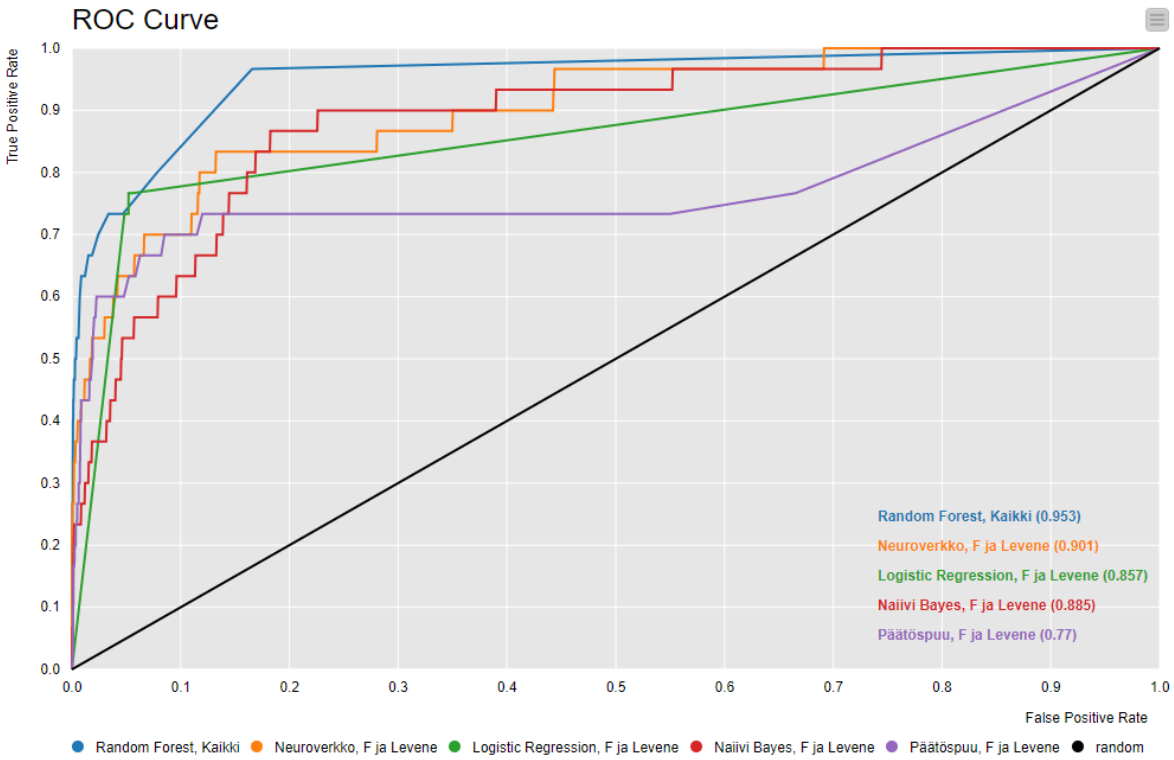
Kuva 42 Eri muuttujanvalinnoilla tehtyjen neuroverkkoluokittelujen ROC -käyrät.

Kaikkien muuttujien käyttäminen ja F-arvojen ja Levenen testin arvojen perusteella valittujen muuttujien käyttäminen tuottavat korkeilla todennäköisyyksillä samansuuntaisia tuloksia. Matalammilla todennäköisyyksillä kaikkien muuttujien käyttäminen kuitenkin laskee luokittelun täsmällisyyttä.

7.3.7 Yhteenveto ongelmatilanteiden luokitteluista

Ongelmatilanteiden luokittelu vastaa esitetynlaisessa tapauksessa yhden minuutin ennustusta ongelmatilanteesta. Testien perusteella kaikkia tilanteita ei voida havaita datasta käytetyillä algoritmeilla, mutta noin 70 % tapauksista on kohtalaisella täsmällisyydellä löydettävissä. Loputkin tapauksista olisi mahdollisesti löydettävissä erilaisella datan jaottelulla. Kaikissa testeissä käytettiin samalla luvulla alustettua viiteen osaan jakoa, minkä vuoksi kuhunkin osaan joutuneet 6 ongelmatapausta eivät ole hyödynnettävissä toistensa ennustamisessa.

Parhaiten testeissä pärjäsi Random Forest -algoritmi kaikkia muuttujia hyödyntäen. Toiseksi paras vaihtoehto oli Random Forest -algoritmi F-arvoihin ja Levenen testin arvoihin perustuvan muuttujanvalinnan avulla. Random Forestia lukuun ottamatta kaikki algoritmit tuottivat F-arvoihin ja Levenen testin arvoihin perustuvalla muuttujanvalinnalla parhaita tuloksiaan. PCA:n kymmenen ensimmäisen komponentin käyttäminen tuotti heikkoja tuloksia kautta linjan. Tämä johtunee selkeiden ongelmatilanteiden sumentumisesta muodostetuissa lineaarikombinaatioissa. Kuvassa 43 on esitetty kunkin algoritmin parhaan luokittelutuloksen ROC -käyrät algoritmien välisen vertailun vuoksi.



Kuva 43 Eri algoritmien tuottamien parhaiden luokittelijoiden ROC -käyrät.

Kuvasta 43 nähdään Random Forest -algoritmin tuottaman luokittelijan toimivan parhaiten lähes kaikissa pisteissä. Parhaan luokittelijan eri todennäköisyyksillä laskettuja sekaannusmatriiseja on esitetty taulukossa 31. Taulukossa kirjain E tarkoittaa ennustusta ja T todellisuutta.

Taulukko 31 Random Forest -algoritmin kaikilla muuttujilla suorittettujen luokittelujen sekaannusmatriisit eri luokittelutodennäköisyyksien arvoilla.

75 %	**E = 0**	**E = 1**
T = 0	20047	0
T = 1	30	0

50 %	**E = 0**	**E = 1**
T = 0	20046	1
T = 1	25	5

25 %	**E = 0**	**E = 1**
T = 0	20021	26
T = 1	17	13

20 %	**E = 0**	**E = 1**
T = 0	19997	50
T = 1	16	14

15 %	**E = 0**	**E = 1**
T = 0	19971	76
T = 1	15	15

10 %	**E = 0**	**E = 1**
T = 0	19902	145
T = 1	12	18

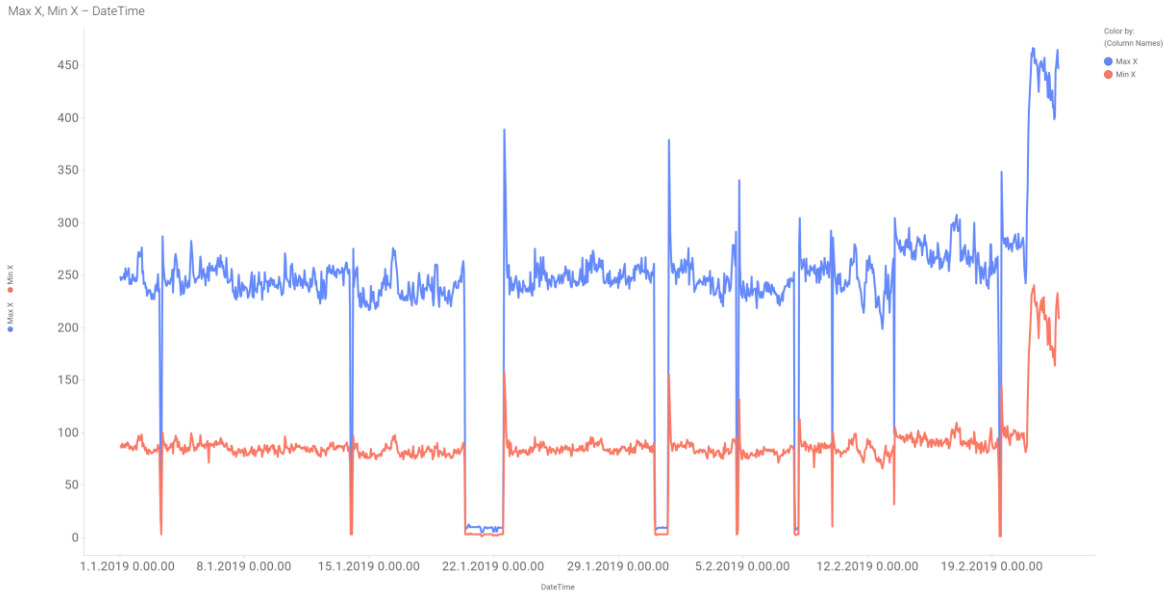
Matriisien perusteella voidaan sanoa, että noin puolet ongelmatilanteista olisi ollut havaittavissa noin minuuttia etukäteen järkevällä False Positive (FP) -taajuudella: 13 löydetyn ongelman tapauksessa FP arvoja tulisi kaksinkertainen määrä True Positive arvoihin nähden, 15 löydetyn ongelman tapauksessa viisinkertainen määrä. Kerrannaisuus kasvaa nopeasti pienempiin todennäköisyyksiin mentäessä.

Ongelmatilanteiden ennustamisen voidaan siis todeta olevan tilanteessa mahdollista, joskaan kaikkia tilanteita ei voida minuutin resoluutiolla havaita järkevästi ilman suurta määrää vääriä hälytyksiä. Ongelmatilanteiden ennustamisen sijaan tilanteessa onkin järkevä paneutua ongelman juurisyihin ja estää ongelman syntyminen. Mikäli ongelmaa ei pystyttäisi tai ei olisi taloudellisesti kannattavaa korjata, ennustus toisi huomattavasti lisäarvoa ongelmatilanteiden välttämisessä.

8. Värähtelytason poikkeuksien havainnointi pumpussa

Anjalankosken paperitehtaalla eräässä vakionopeudella toimivassa pumpussa havaittiin eri antureista kohonneita värähtelytasoja. Värähtelytasoja mitattiin kiihtyvyysanturin avulla. Mittausten resoluutio oli vaihteleva, kuitenkin niin, että tunnittaiset värähtelyn maksimi- ja minimiarvot kunkin akselin X, Y ja Z -suuntaisesti olivat saatavilla. Mittaukset antoivat värähtelyarvot yksittäisinä pisteinä, jotka kuvaavat yleistä tasoa. Spektrimittauksia kyseiset anturit eivät tehneet.

Värähtelytason nousu aiheutui ilmeisimmin jälkikäteen diagnosoidusta tiivisteveden kierron häiriöstä. Häiriön seurauksena noin puolentoista vuorokauden kuluttua pumpulla olleen massan havaittiin syttyneen tuleen. Tilanteen seurauksena ei aiheutunut tuotannon menetyksiä, mutta havaitsemalla tiivisteveden tukos ajoissa esimerkiksi värähtelydatan perusteella, oltaisiin vaaratilanteelta voitu välttyä. Kuvissa 44 ja 45 on esitetty värähtely- ja lämpötiladata syttymiseen johtaneelta ajanjaksolta.



Kuva 44 Pumpun erään anturin X-suuntaisen värähtelyn maksimi- ja minimiarvot vuoden 2019 alusta ongelmatilanteeseen asti. Keskiarvot eivät konfigurointien vuoksi ole saatavissa.

Kuva 45 Saman anturin mittaamat lämpötila-arvot samalta ajanjaksolta. Lämpötilan arvo jää palotilanteessakin melko matalaksi, sillä anturi sijaitsi suojatussa kotelossa paloa vastakkaisella puolella pumppua.

Värähtelytason muutos on kuvassa 44 silmin nähtävissä ja oikein säädetty yksinkertainen yläraja värähtelylle olisi riittänyt antamaan hälytyksen ennen ongelman eskaloitumista.

Yksinkertaista ylärajaa ei kuitenkaan ole aina mahdollista järkevästi asettaa, sillä laitteet, jotka toimivat esimerkiksi monilla eri kierrosnopeuksilla värähtelevät eri tavoin riippuen pyörimisnopeudesta. Myös ylösajojen aikana kohoavat värähtelytasot voivat aiheuttaa matalilla rajoilla vääriä hälytyksiä, minkä vuoksi myös vakionopeudella ajettavien laitteiden värähtelyn yläraja on usein liian korkealla havaitsemaan kehittyviä vikoja.

Värähtelytasoa tarkkaillaan tässä työssä eri menetelmillä, jotka ovat potentiaalisesti hyödynnettävissä tasaisilla ja vaihtelevilla nopeuksilla toimivissa laitteissa. Luvuissa 8.1, 8.2 ja 8.3 esitellään lyhyesti käytetyt menetelmät ennen niiden soveltamista pumpun tapaukseen. Esiteltävät ja hyödynnettävät kolme menetelmää ovat:

1. Autokorrelaatioon perustuva värähtelyn tarkkailu
2. (Auto)Regressioon perustuva värähtelyn tarkkailu
3. PCA-rekonstruktioon perustuva värähtelyn tarkkailu

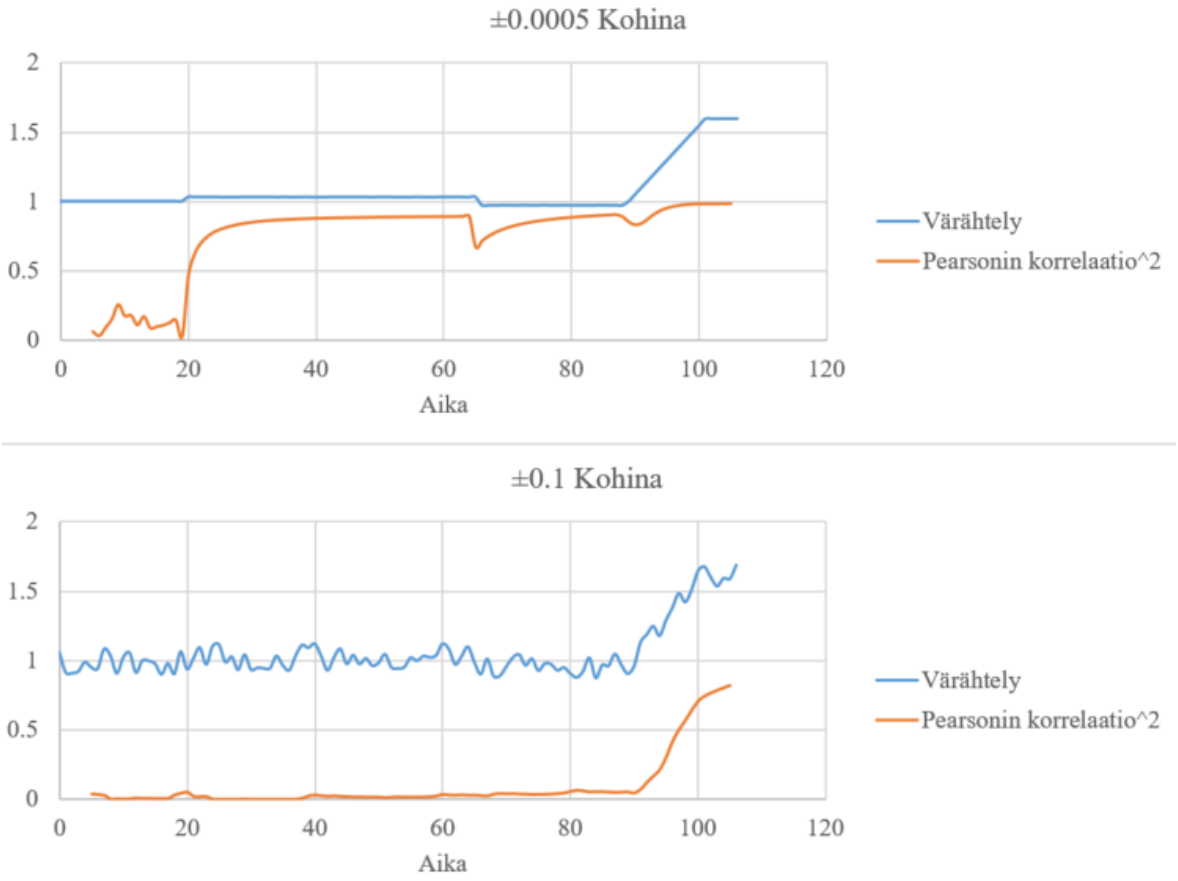
Värähtelyntasojen tarkkailu voi lisäksi perustua useaan muuhun menetelmään. Yksi näistä menetelmistä on yksiluokkaluokittelu. Yksiluokkaluokittelu on luokittelun erikoistapaus, jossa algoritmi pyrkii oppimaan yhteen luokkaan kuuluvien datapisteiden muuttujien rajat. Yksi yksiluokkaluokittelualgoritmi on yhden luokan Support Vector Classifier, joka yksiluokkaluokittelun tapauksessa muodostaa moniulotteisen tason sijaan moniulotteisen pallon. (Tax, 2001) Rajojen ulkopuolelle jäävät pisteet luokitellaan poikkeuksiksi, mikä voi tarkoittaa esimerkiksi liian korkeaa värähtelytasoa alhaisella pyörimisnopeudella. Myös monia muita algoritmeja voidaan hyödyntää yksiluokkaluokittelussa.

Yksiluokkaluokittelu on vaihtoehto poikkeuksellisen toiminnan erotteluun, mikäli saatavilla on laajasti ja edustavasti dataa useilta laitteen käyntinopeuksilta. Menetelmän luokittelutulos kärsii melko voimakkaasti käytettäessä vain osittaista dataa (Tax, 2001). Yksiluokkaluokittelua ei tässä työssä käytännössä testata. Algoritmien ja parametrien valinta yksiluokkaluokitteluun vaatii tarkkaa perehtymistä.

8.1 Autokorrelaatioon perustuva värähtelyn tarkkailu

Autokorrelaatiolla tarkoitetaan mitattavan suureen arvojen korrelaatiota edellisten arvojensa kanssa, esimerkiksi värähtelyn arvojen (t) korrelaatiota arvojen (t - 1) kanssa. Autokorrelaatioon perustuvan värähtelynseurannan tehokkuus perustuu peräkkäisten

värähtelymittausten riippumattomuuteen. Luonnollinen, pieni ja hiljattainen värähtelytason vaihtelu näkyy kohinattomassa datassa korrelaationa, vaikka merkittävää muutosta ei olisi tapahtunut. Normaalissa toiminnassa värähtely pysyy tyypillisesti pitkällä aikavälillä tarkkailtuna melko vakiona, mutta yksittäiset peräkkäiset pisteet vaihtelevat arvoltaan melko huomattavastikin. Tietämällä esimerkiksi edellisen minuutin värähtelyn taso ei siis pystytä sanomaan, onko värähtelyn taso seuraavan minuutin aikana hieman korkeampi vai matalampi. Kuva 46 havainnollistaa riippumattomuuden, eli käytännössä kohinan, merkityksellisyyttä autokorrelaatiossa.



Kuva 46 Excelissä luotu kuvitteellinen värähtelydata (siniset käyrät) pienellä (ylempi) ja kohtalaisella (alempi) kohinan määrällä. Ylemmän kuvan data alkaa tasolta 1, minkä jälkeen se nousee tasolle 1,03 ja laskee taas tasolle 0,97. Lopussa värähtelyn arvo nousee 0,05 askelein tasolle 1,60. Dataan on kaikissa vaiheissa lisätty värähtelytasoon ±0,0005*rand() minimaalisen kohinan luomiseksi. Alemmassa kuvassa ylemmän kuvan värähtelytasoon on lisätty ±0.1*rand() suuremman kohinan luomiseksi. Oranssit käyrät kuvaavat tilanteiden värähtelytason (t) Pearsonin korrelaatioita värähtelytasoon (t – 1) toiseen korotettuna koko edeltävältä aikaikkunalta.

Kuvassa 46 havaitaan autokorrelaation tason nousevan suhteellisesti todella pienienkin muutosten perusteella, kun data ei sisällä juurikaan kohinaa. Todellinen data kuitenkin sisältää aina kohinaa, kuten kuvan 46 alemman osan värähtelykäyrä. Kohinaa sisältävän datan tapauksessa ainoastaan suuremmat, systemaattiset muutokset näkyvät autokorrelaation arvoissa.

Vikaantumisen tai muun poikkeustilanteen edistyessä värähtelytaso alkaa tyypillisesti nousemaan. Tällöin on usein havaittavissa selkeä trendi, eli tarkkailtaessa esimerkiksi värähtelyn edellisen viiden tasatunnin arvoja, on jokainen arvo toistaan suurempi. Tämä trendi näkyy mitatun arvon ja näiden edellisten arvojen autokorrelaatioarvon nousuna tarkkailtaessa liikkuvaa aikaikkunaa. Tarkkailtavan aikaikkunan pituus voi olla minuutin välisillä mittauspisteillä esimerkiksi joitakin tunteja. Tuntien välisillä mittauspisteillä ikkuna voi olla kymmenistä tunneista satoihin tunteihin.

Autokorrelaatioon perustuvassa värähtelynseurannassa käsiteltävän datan suodattaminen on kriittistä. Tarkkailtavan laitteen alas- ja ylösajojen sekä nopeuden muutosten suodattaminen on välttämätöntä, sillä värähtely korreloi yleisesti laitteen nopeuden kanssa. Kun ajonopeudessa havaitaan muutos, on autokorrelaatioon perustuva tarkkailu pysäytettävä tarkkailun aikaikkunan mittaiseksi ajaksi, minkä jälkeen tarkkailua voidaan jatkaa.

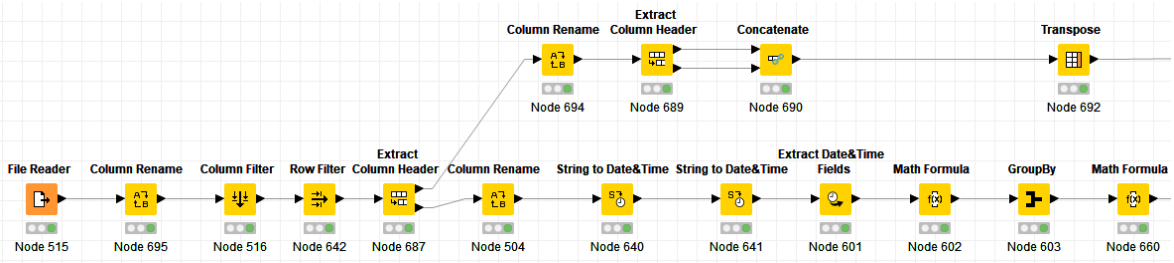
Autokorrelaatioon perustuva tarkkailu on siis potentiaalisesti tehokas vaihtoehto laitteille, jotka toimivat pitkiä aikoja vakionopeudella. Menetelmän havainnointikyky riippuu värähtelyarvojen muutosnopeuden ja hajonnan suhteesta. Menetelmä ei reagoi yksittäisiin, tavallisesti vääriä hälytyksiä aiheuttaviin piikkeihin, eikä menetelmän asettajan periaatteessa tarvitse tietää normaaleja värähtelyrajoja käyttöönotossa, mikä luo etuja verrattuna kiinteisiin rajoihin.

Hälytys autokorrelaatioon perustuvassa menetelmässä annetaan korrelaation noustessa tietylle tasolle. Taso voidaan määrittää automaattisesti esimerkiksi normaalin toiminnan historiallisen tason perusteella niin, että 2 kertaa korkeinta tasoa korkeammalle noussut autokorrelaatio laukaisee hälytyksen.

8.1.1 Autokorrelaation soveltaminen pumpun värähtelyn tarkkailussa

Autokorrelaatioon perustuvan tarkkailun rakentaminen historiadatalle on KNIMEllä melko yksinkertaista. Toteutus online-datalle ei kuitenkaan ole KNIMEllä mahdollista ja tarkkailu on käytännössä siirrettävä pilveen, jossa vastaava ratkaisu tulisi olla mahdollista ohjelmoida esimerkiksi Pythonilla. Ratkaisuun liittyvä matematiikka ja operaatiot eivät ole erityisen monimutkaisia, joten rajapinnat tuntevalle ohjelmoijalle itse ohjelmointi ei ole todennäköisesti ongelma.

Historiadatan analysoinnissa käytetään kuvissa 47 ja 49 esitettyä työnkulkua kunkin värähtelysuunnan [X,Y,Z] minimi- ja maksimiarvojen analysointiin. Työnkulku alkaa kuvalla 47.



Kuva 47 Värähtelydatan työnkulun analysoinnin alku autokorrelaatioon perustuvassa menetelmässä. Työnkulku sisältää työvaiheet datan lukemisesta laitteen käynnissäolon määrittämiseen.

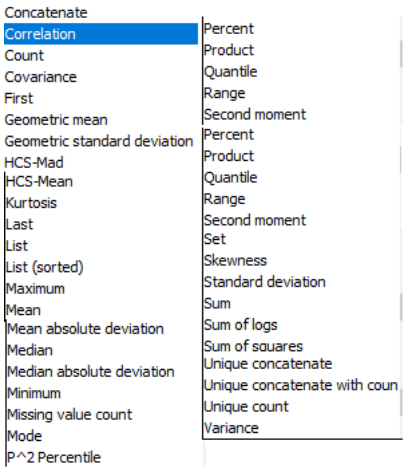
Työnkulku alkaa datan sisältävän tiedoston lukemisesta. Eri suuntien värähtelyt sisältävät sarakkeet nimetään esimerkiksi muotoon Min X. Column Filter -yksikössä poistetaan muut kuin analysoitava suunta ja aikatiedon sisältävät sarakkeet. Row Filter -yksikkö poistaa suodatuksen jäljiltä jääneet värähtelytietoa sisältämättömät rivit.

Extract Column Header -yksikkö muuntaa jäljelle jääneet sarakkeet standardimuotoon ja ottaa vanhat sarakkeiden nimet muistiin. Työnkulku jakautuu tämän jälkeen kahtia – ylempi osa kulkua kuljettaa mukanaan vanhat sarakenimet, eikä ole toiminnallisuuden kannalta merkittävä. Alempi osa analysoi itse datan.

Alemman virran Column Rename -yksikössä kolmas sarake, eli tutkittava värähtely, nimetään muotoon ”tutkittava”. Menettelyn ansiosta ensimmäisen Column Filter -yksikön

avulla valittu tarkkailtava värähtelysarake kulkee koko työnkulun läpi automaattisesti muita muutoksia vaatimatta. Uudelleennimeämisen jälkeen aika- ja päivämääräsarakkeet muunnetaan oikeaan muotoonsa yksiköissä String to Date&Time. Yksikössä Extract Date&Time Fields lisätään kullekin riville sarake, joka kertoo, mitä tasatuntia tarkkaillaan. Seuraava Math Formula -yksikkö sallisi jaottelun vielä valinnaisille minuuttiväleille tarpeen vaatiessa.

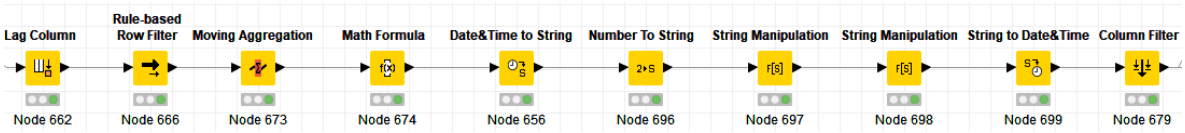
Yksikössä GroupBy määritellään värähtelylle yksi arvo jokaiselta tunnilta. Määrittelyn voi tehdä monella perusteella, jotka on listattu alla kuvassa 48.



Kuva 48 Vaihtoehdot Group By -yksikön laskentaperusteelle.

Jokaisen tunnin ensimmäinen arvo valitaan ryhmittelyperusteeksi. Myös muita vaihtoehtoja kokeiltiin, mutta esimerkiksi keskiarvoa käyttämällä pienetkin värähtelyn muutokset tuntien välillä ylikorostuvat autokorrelatiivisessa tarkkailussa, sillä värähtelyn satunnaisuus vähenee.

Tunnittaisen ryhmittämisen jälkeen datalle valitaan yksikössä Math Formula ”pudotusraja”. Täydellisemmässä ja etukäteen suunnitellussa tarkkailussa sisältäisi lähtödata tiedon esimerkiksi kierrosnopeudesta tai virrankulutuksesta, jonka perusteella laitteen poissaolo suodatettaisiin. Esimerkkitapauksessa tätä tietoa ei kuitenkaan ole, joten pisteet, jossa värähtelyn arvo on alle 1/5 keskiarvosta suodatetaan pois. Myös viimeisintä tällaista arvoa seuraavat viisi arvoa suodatetaan pois ylösajojen aiheuttaman normaalia suuremman värähtelytason vuoksi. Kuvassa 49 on esitetty työnkulun seuraava osa.

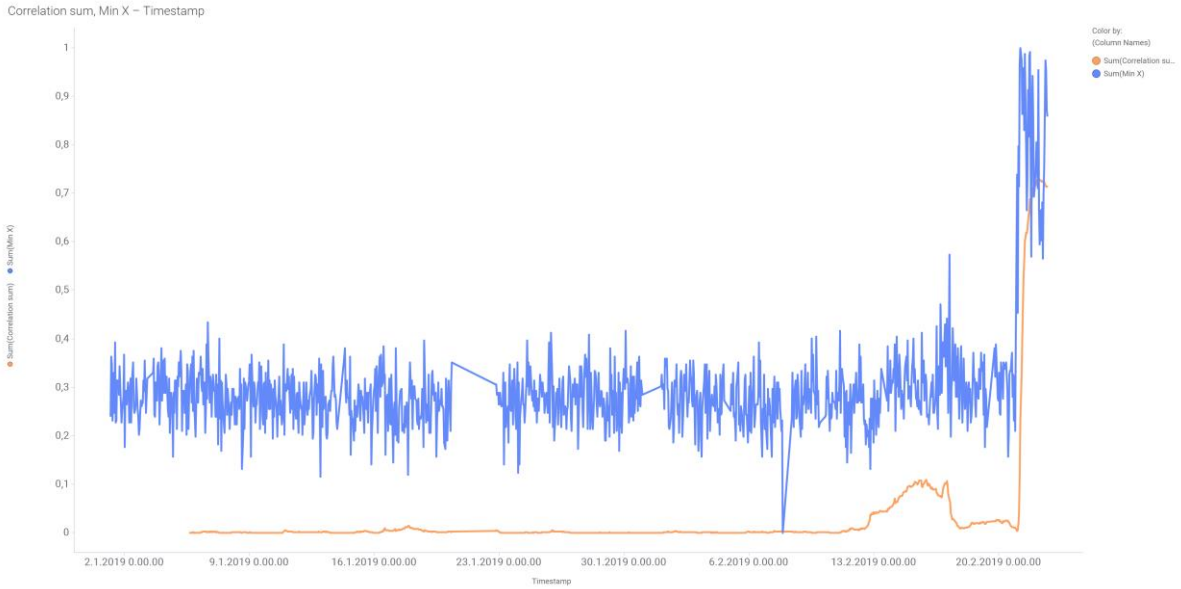


Kuva 49 Työnkulun jatko värähtelytasojen autokorrelaation tarkkailussa. Kuva sisältää työnkulun osat värähtelyarvojen viiveen asettamisesta ulostulon muotoiluun.

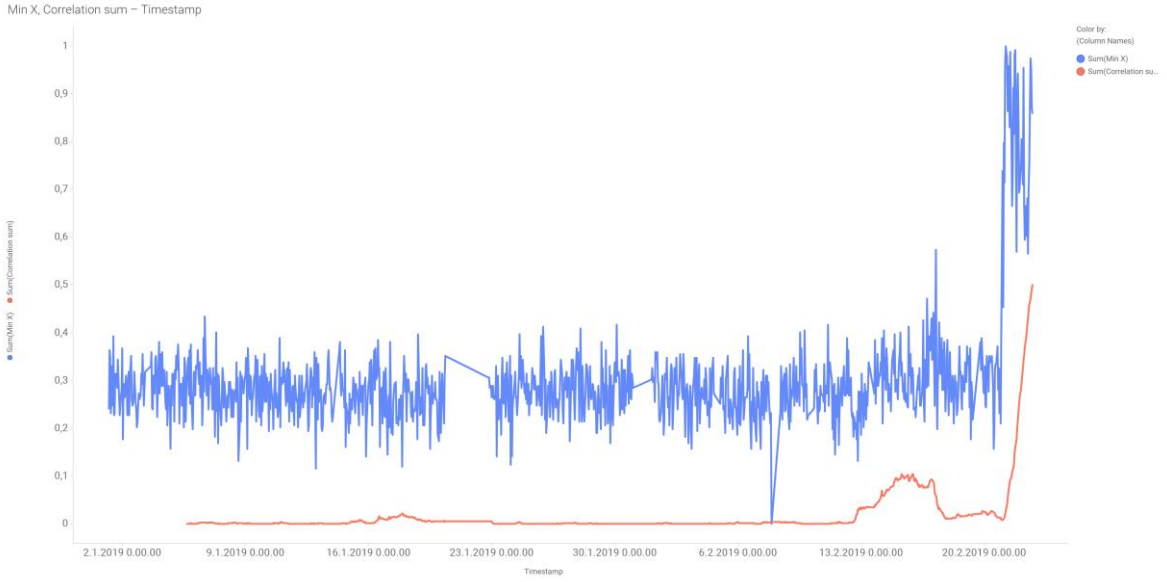
Matalien värähtelyarvojen pois suodattaminen ja viiden edellisen minuutin määrittäminen tapahtuvat yksiköissä Rule-based Row Filter ja Lag Column. Suodatuksen jälkeen lasketaan itse autokorrelaatioarvot yksikössä Moving Aggregation. Laskemisessa käytetään sadan edellisen arvon aikaikkunaa ja arvo autokorrelaatiolle lasketaan kullekin arvolle t(-1...-5) erikseen. Korrelaatioksi voidaan valita joko Pearsonin korrelaatio, Spearmanin korrelaatio tai Kendallin korrelaatio. Lasketuille korrelaatioarvoille lasketaan keskiarvo, joka korotetaan toiseen potenssiin yksikössä Math Formula. Toiseen korottaminen muuntaa negatiiviset arvot positiivisiksi ja alentaa autokorrelaation perustasoa. Vaikutus korostuu matalilla arvoilla, mikä saa voimakkaan autokorrelaation erottumaan selkeämmin.

Kaikki Math Column -yksikön jälkeiset yksiköt työnkulussa liittyvät ulostulon muotoseikkoihin, eivätkä täten ole merkityksellisiä itse analyysin kannalta.

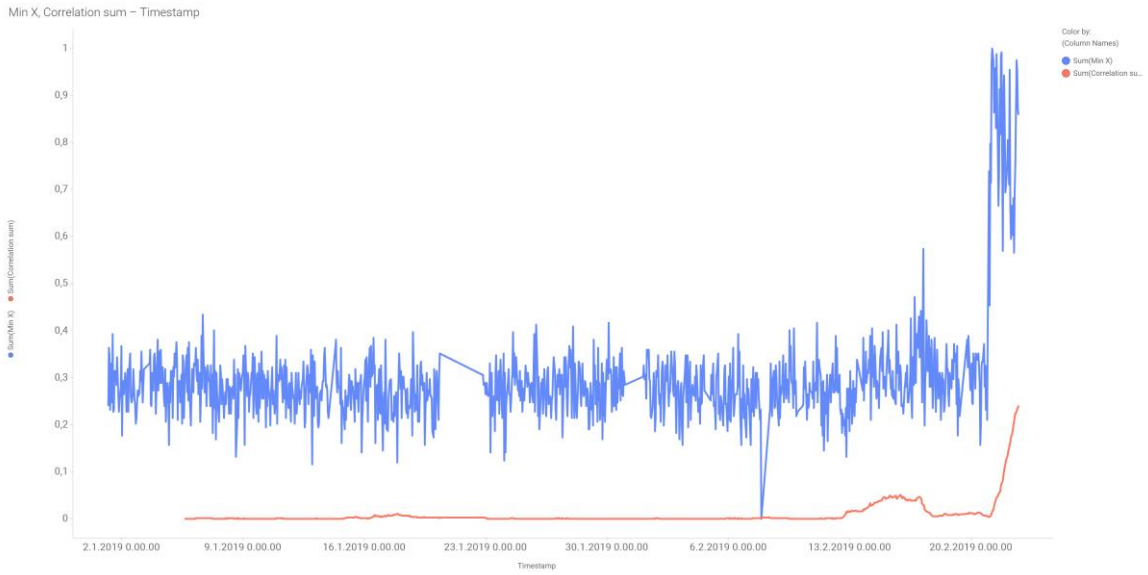
Kuvissa 50 - 52 on esitetty edeltävällä työnkululla lasketut autokorrelaatioarvot datajoukolle eri korrelaatiotyypeillä. Sininen, epätasainen viiva kuvaa välille [0, 1] normalisoituja X-suuntaisen värähtelyn minimiarvoja. Oranssi, tasainen viiva kuvaa autokorrelaation arvoja.



Kuva 50 X-akselin värähtelyn minimiarvojen tarkkailu 100 edellisen arvon liikkuvalla ikkunalla Pearsonin korrelaatiolla.



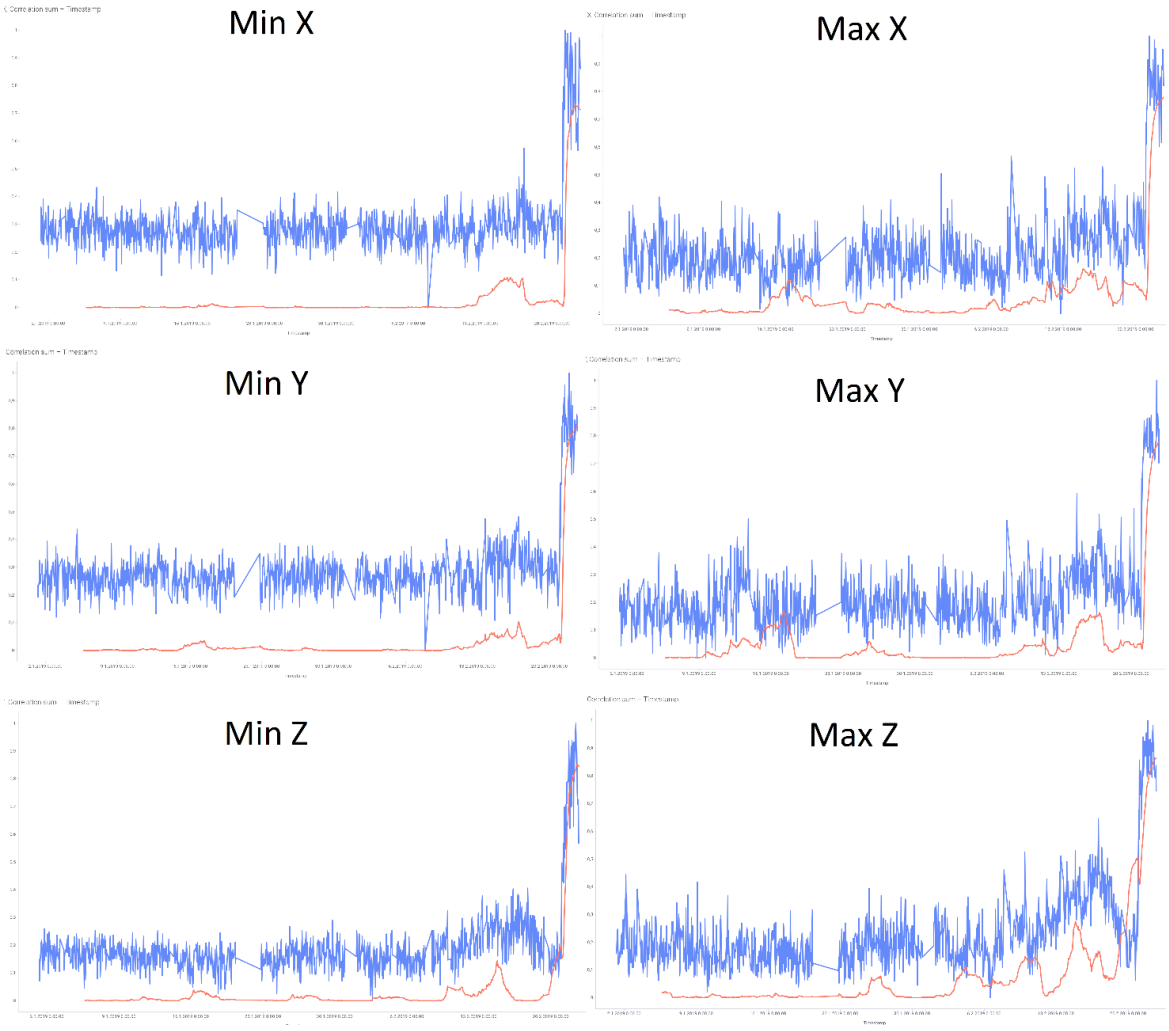
Kuva 51 X-akselin värähtelyn minimiarvojen tarkkailu 100 edellisen arvon liikkuvalla ikkunalla Spearmanin korrelaatiolla.



Kuva 52 X-akselin värähtelyn minimiarvojen tarkkailu 100 edellisen arvon liikkuvalla ikkunalla Kendallin korrelaatiolla.

Kuvista havaitaan Pearsonin lineaarikorrelaation reagoivan itse ongelmaan nopeiten suurimmalla kulmakertoimella. Kaikki korrelaatiotyypit kuitenkin reagoivat muutokseen. Parhaimman havainnointikyvyn vuoksi Pearsonin korrelaatiota käytetään jatkotarkasteluissa.

Korrelaatiokäyrissä näkyvä ongelmaa edeltävän nousun syy on avoin kysymys. On hyvinkin mahdollista, että kyseessä on aikaisempi lievempi ongelma, joka muutaman päivän jälkeen eskaloitui tukostilanteessa. Sama nousu näkyy myös muissa mittauksissa. Kuva 53 sisältää kaikkien akseleiden värähtelyjen minimi- ja maksimiarvojen Pearsonin autokorrelaatiokäyrät.



Kuva 53 Pumpun värähtelyn (sininen) ja autokorrelaation (oranssi) käyttäytyminen eri akseleilla. Värähtelyarvot on normalisoitu välille [0, 1].

Käyriä tulkitsemalla voidaan tehdä seuraavia johtopäätöksiä tarkkailtavasta kohteesta ja autokorrelaation toiminnasta kunnonvalvonnassa.

Varsinaista ongelmaa edeltävä värähtelyjen nousu näkyy erityisesti Z-akselin suuntaisessa värähtelyssä, korostuen maksimiarvoissa. Kyseessä on todennäköisesti jonkinlainen kehittyvä ongelma.

X- ja Y- akselien suuntaisissa värähtelyissä on havaittavissa yksittäinen poikkeama huomattavasti ennen aidon ongelman syntymistä. Poikkeama näkyy varsinkin Y-akselin suuntaisessa värähtelytasossa. Autokorrelaatio ei kuitenkaan nouse suuresti ja

korrelaatiokäyrässä tapahtuva nousukin on suurimmillaan värähtelyjen palatessa normaalitasolle.

Varsinaisen ongelmatilanteen aiheuttama värähtelyn nousu voidaan havaita autokorrelaatiolla helposti akselista riippumatta. Autokorrelaatio reagoikin kokeiden perusteella nimenomaan nopeisiin, selkeisiin muutoksiin paremmin kuin kohinaiseen hitaaseen muutokseen.

Minimikäyrien autokorrelaatiolle voitaisiin kaikilla akseleilla asettaa ylärajaksi esimerkiksi 0,07 ilman, että lainkaan vääriä hälytyksiä tulisi. Jos ainoastaan selkeimmät värähtelyn nousut haluttaisiin esille, autokorrelaation rajaksi voitaisiin asettaa esimerkiksi 0,2. Lisää dataa eri ongelmista tarvitaan yleisten rajaehdotusten antamiseksi.

Maksimikäyrät autokorreloivat yleisesti selkeästi enemmän kuin minimikäyrät. Korrelaation rajat on tällöin asetettava korkeammalle kuin minimikäyrien tapauksissa väärien hälytysten välttämiseksi.

Autokorrelaation voidaan todeta toimivan ongelmien havaitsemisessa värähtelydatasta. Menetelmän ideana ei ole löytää ongelmia, joita ihminen ei nopealla tarkastelulla havaitsisi, vaan olla normalisoitu ratkaisu hälytysrajojen asettamiselle. Autokorrelaatioon perustuva ratkaisu voidaan periaatteessa ottaa käyttöön heti anturin asentamisen jälkeen yleisestä värähtelytasosta riippumatta – koviin rajoihin perustuvat menetelmät tarvitsevat perustason ja hajonnan tietoja eli vähintään hieman alustavaa dataa. Autokorrelaation perusteella annetut hälytykset eivät myöskään laukeaisi yksittäisistä korkeista arvoista.

8.2 (Auto)regressioon perustuva värähtelyn tarkkailu

Regressioon perustuva värähtelyn tarkkailu kykenee ottamaan muuttuvat nopeudet huomioon paremmin kuin autokorrelaatioon perustuva tarkkailu. Menetelmä perustuu laitteen värähtelytason tai pyörimisnopeuden ennustamiseen, kun eri muuttujien arvoja tunnetaan. Regressiomallilla ennustetaan esimerkiksi pyörimisnopeuden ja värähtelyn edellisten arvojen perusteella tulevaa värähtelyarvoa tai nykyisten värähtelyarvojen

perusteella nykyistä pyörimisnopeutta. Ennustettua ja todellista arvoa verrataan toisiinsa, ja jos arvot eroavat toisistaan riittävästi, aletaan kumuloimaan hälytystasoa.

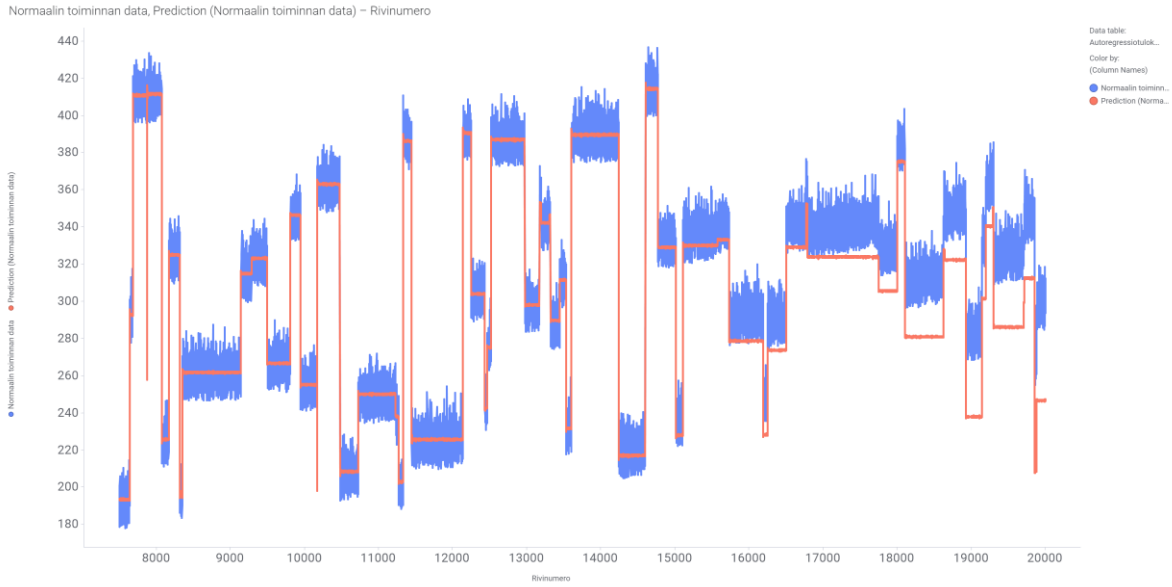
Kumuloituvaa hälytystasoa käytetään yksittäisten poikkeavien arvojen suodattamiseksi. Kumuloituminen voidaan tehdä esimerkiksi lisäämällä hälytystasoon 1 jokaisen huonon ennustuksen jälkeen, ja laskemalla tasoa 0,5:llä jokaisen hyvän ennustuksen jälkeen niin, että minimitaso on 0. Ennustuksen laatu voidaan määritellä automaattisesti keskihajontojen määränä validointidataan verrattuna. Esimerkiksi alle 3 keskihajontaa todellisuudesta eroavat ennustukset ovat hyviä, yli 3 keskihajontaa eroavat huonoja. Kun hälytystaso ylittää tietyn rajan, esimerkiksi 5, annetaan kunnossapitohenkilöstölle hälytys. Hälytystason rajalle voidaan myös asettaa datan perusteella automaattinen rajan luonti, esimerkiksi 2 tai 3 kertaisesti tunnetun normaalin toiminnan maksimi.

Potentiaalisesti hyödynnettävissä olevia regressiotyyppejä on erilaisia, esimerkiksi lineaarinen-, polynominen- ja päätöspuuregressio. Lineaarisella regressiolla voidaan mallintaa esimerkiksi vakionopeudella toimivia laitteita. Jos laitetta käytetään monilla eri nopeuksilla, voi lineaarinen regressio olla huono vaihtoehto, sillä värähtelyn taso ei todennäköisesti käyttäydy täysin lineaarisesti nopeuden suhteen. Polynominen regressio on hyvä vaihtoehto vaihtelevan nopeuden tapauksissa, mikäli yhteys nopeuden ja värähtelyn välillä on yksinkertaistettavissa polynomilla. Päätöspuuregressio ennustaa arvon perustuen sääntöihin, kuten pyörimisnopeuteen. Se siis toimii hyvin, jos opetusdataa on laajalti.

Polynominen regressio on hyvä vaihtoehto, mikäli ennustuksien laatu on kohtalaisen tasainen koko nopeusalueella. Menetelmä vaatii kuitenkin mielellään opetusdataa ainakin hitaimmasta ja nopeimmasta nopeudesta muutaman välinopeuden lisäksi kertoimien määrittämiseksi. Päätöspuuregressio vaatii dataa vielä laajemmin eri nopeusalueilta. Lineaarista regressiota voidaan hyödyntää pääasiassa yhdellä nopeudella toimivissa laitteissa. Tällöin yksittäisten mittaussuureiden tapauksessa on kyseessä autoregressio, sillä muuta hyödyllistä dataa, kuin edelliset arvot ja niiden muutokset ei ole saatavilla. Mallin ennustuskyky on värähtelyn luonteesta johtuen keskimäärin heikko, mutta ennustus heikkenee entisestään ongelmien ilmetessä.

Kuvassa 54 on esitetty Excelissä simuloidulla värähtelydatalla polynomisen regression toimintaa. Dataa simuloitiin 20 000 pistettä Excelissä käyttäen satunnaisia nopeuksia 0,6 ja 1 välillä. Värähtelyn määrä on simulaatiossa riippuvainen nopeuden 1,6 potenssista ja

perusarvosta 400. Hajontaa lisättiin lisäämällä kuhunkin arvoon 25 kertaa satunnaisluku välillä [0, 1] ja 25:n ja kolmen satunnaisluvun tulo. Dataan lisättiin alkava ongelma noin rivin 14 000 kohdalla, jolloin potenssin 1,6 arvo alkaa hiljalleen pienenemään. Ensimmäistä 7 500 riviä käytettiin regression opettamiseen.



Kuva 54 Simuloidut värähtelyarvot (sininen) ja pyörimisnopeuden, pyörimisnopeuden muutoksen ja viiden edellisen värähtelyarvon avulla ennustettu värähtely (oranssi).

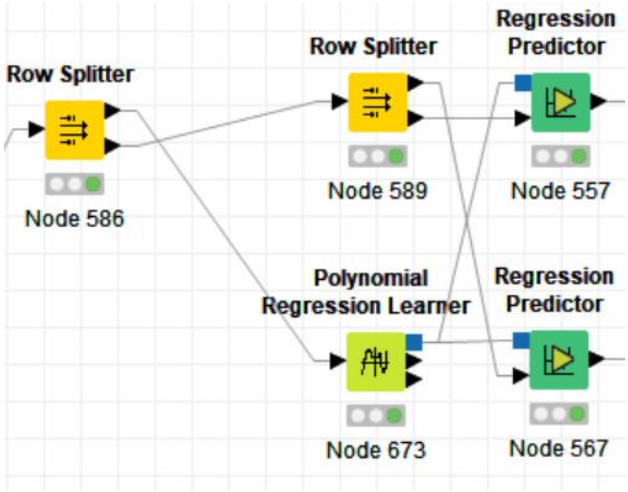
Kuvasta 54 nähdään, että jo pian simuloidun ongelman syntymisen jälkeen ennustuksen taso alkaa heikkenemään. Todellinen data ei tule koskaan olemaan yhtä selkeää ja hyvin ennustettavaa kuin simuloitu, mutta simuloitu data näyttää menetelmän toiminnan periaatteen.

Regression voidaan värähtelyn tarkkailussa ajatella olevan automaattinen muisti normaaleille arvoille. Tämän lisäksi regressio interpoloi ”normaalit” arvot nopeuksille, joita opetusdatassa ei ole.

8.2.1 Autoregression soveltaminen pumpun värähtelyn tarkkailussa

Historiadatan esikäsittely regressioon perustuvassa menetelmässä on pääosin samankaltainen kuin autokorrelaatioon perustuvassa menetelmässä. Merkittävä ero menetelmien välillä on Group By -yksikön ryhmittelyasetuksissa. Regressiossakin voidaan käyttää samaa tunnin resoluutiota, mutta arvon määrittämiseen voidaan käyttää ensimmäisen arvon sijaan esimerkiksi keskiarvoa. Keskiarvon käyttäminen lisää ennustettavuutta, mikä on hyvä asia regressioon perustuvassa tarkkailussa. Myös muiden tunnin ensimmäistä arvoa enemmän ennustettavien arvojen, kuten tunnin minimiarvon käyttäminen on mahdollista. Ylös- ja alasajot poistetaan autokorrelaatiomenetelmää vastaavalla tavalla ajonopeustiedon puuttumisen vuoksi.

Itse ennustuksessa käytetään värähtelyn kolmea edellistä arvoa sekä kolmea edellistä arvon muutosta. Mallin opetus, sekä validointi- ja testidatan ennustaminen tapahtuvat kuvassa 55 olevan työnkulun mukaisesti.

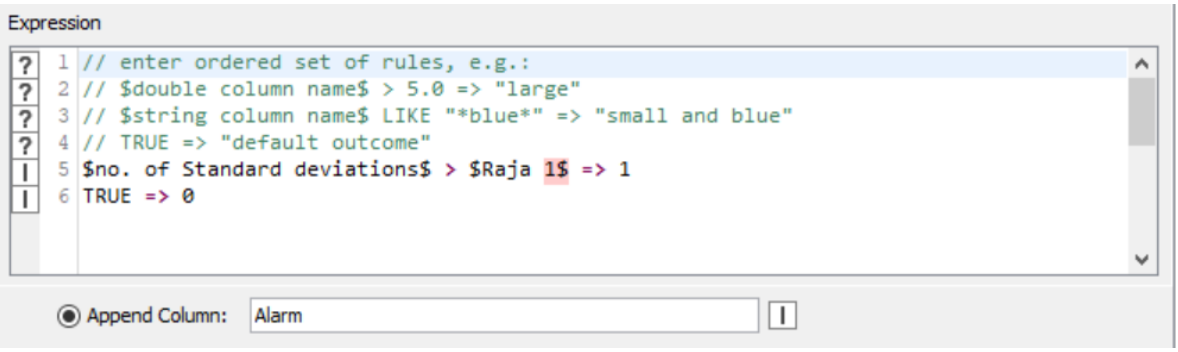


Kuva 55 Mallin opetus ja uusien arvojen ennustaminen KNIMEssä. Data jaetaan opetus-, validointi-, ja testidataan, joita hyödynnetään regressioyksiköissä.

Ensimmäinen Row Splitter -yksikkö jakaa datan opetus- (ylempi ulostulo) ja muuhun dataan. Toinen Row Splitter -yksikkö jakaa muun datan testi- (ylempi) ja validointidataan. Työnkulun Learner- ja Predictor -yksiköt voidaan vaihtaa haluttuun regressiomalliin. Menetelmää käytettäessä online-datan kanssa esimerkiksi ensimmäisen viikon data

ohjattaisiin mallin opetukseen ja tätä seuraavan viikon data validointiin. ”Testidata” on online-käytön tapauksessa todellinen data, jota tarkkaillaan.

Validointidatan ennustuksien perusteella lasketaan ennusteille keskihajonta. Testidatalle lasketaan kunkin pisteen hajonta erikseen ja tämä arvo jaetaan validointidatan keskihajonnalla, jolloin ennustuksen hajonta saadaan standardisoitua keskihajontojen määräksi. Keskihajontojen määrien perusteella Rule Engine -yksiköllä lisätään kuvan 56 mukainen uusi sarake, jonka kumuloitumista seurataan.



Kuva 56 Pistehälytyksien määrittely Rule Engine -yksiköllä. Lasketun sarakkeen kumuloitumista seurataan.

Historiadatan käyntitiedon puutteen vuoksi lasketaan jokaisen pisteen välinen aikaero. Aikaeroa käytetään nollaamaan hälytystaso, jos aikaero on yli kaksi kertaa keskimääräistä suurempi. Hälytystason muutos määritellään aiemman kuvauksen mukaiseksi – hälytys laskee puolella nollaan saakka hyvillä ennustuksilla ja nousee yhdellä huonoilla ennustuksilla. Hälytystason kumuloituminen on laskettu Python -koodilla, joka on esitetty kuvassa 57.

```python
from pandas import DataFrame
# Luodaan tyhjä taulukko skriptin ulostulolle
output_table = DataFrame()

# Napataan KNIMEstä sarakkeet listoiksi
alarmChange = list(input_table['Alarm Change'])
timeDiffX = list(input_table['Time difference X'])
timeDiffLimit = list(input_table['Aikaraja nollaukselle'])

# Luodaan kumulatiiviselle hälytysarvolle lista
# ja lisätään ensimmäiseksi arvoksi 0, jota
# tarvitaan loopin ensimmäisellä kierroksella
cumulativeAlarm = list()
cumulativeAlarm.append(0)

# Pyöritetään looppi kaikille riveille
for i in range(0, len(alarmChange)):
    # Määrittely nollalle minimiarvona
    if alarmChange[i] == -0.5 and float(cumulativeAlarm[i]) == 0:
        cumulativeAlarm.append(0)
    # Määritetään aikahypyt
    elif timeDiffX[i] > timeDiffLimit[0]:
        cumulativeAlarm.append(0)
    # Muuten lasketaan uusi arvo ja lisätään listaan
    else:
        newValue = cumulativeAlarm[i] + alarmChange[i]
        cumulativeAlarm.append(newValue)

# Poistetaan vielä listan ensimmäinen arvo, jota
# tarvittiin vain ensimmäiseen laskuun. Lista siirtyy
# askeleella taakse oikeaan paikkaan
cumulativeAlarm.remove(0)

# Ulostuloon kumulatiiviset arvot
output_table = DataFrame(cumulativeAlarm)
```

Kuva 57 Hälytyksien kumuloitumisen laskeminen. Laskeminen tapahtuisi Online-seurannassa erilaisella koodilla.

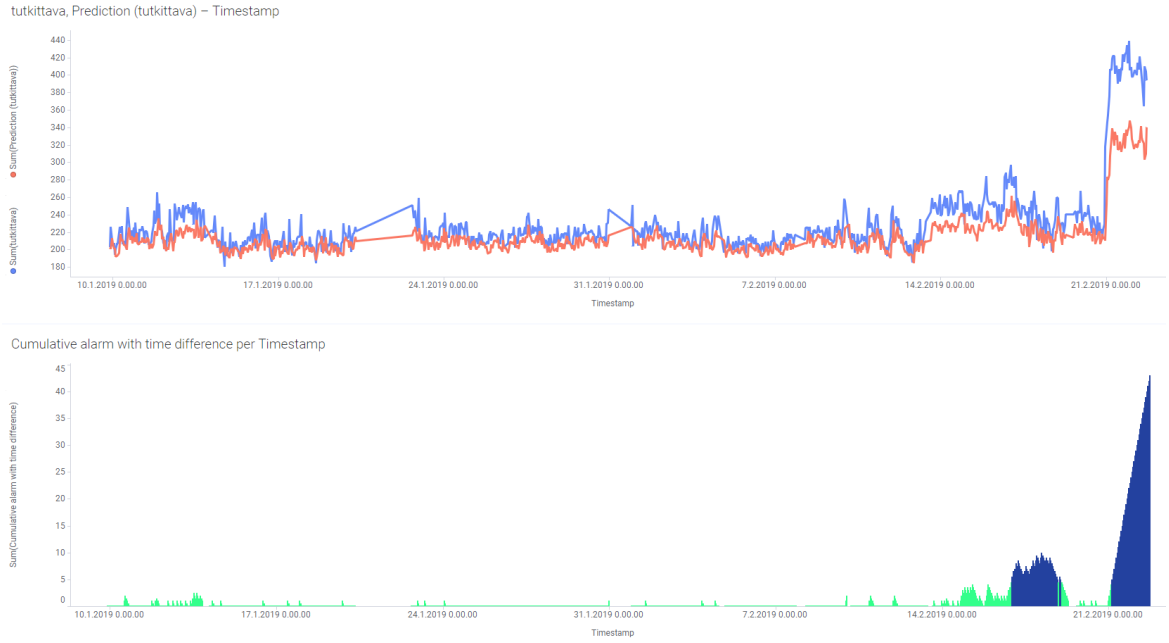
Kumuloituneiden hälytyksien avulla määritellään vielä Rule Engine -yksiköllä, milloin todellinen hälytys annettaisiin. Testien tapauksessa käytetään hälytysrajana kumulatiivisen hälytyksen arvoa 5, mutta automaattisen hälytysrajan asettaminen validointidatan avulla olisi ohjelmoimalla triviaalia.

Työnkulkua käytetään pumpun värähtelyn tarkkailussa muutamalla eri asetuksella. Testeissä käytetään lineaarista-, polynomista- ja päätöspuuregressiota tarkkailemaan Y-akselin suuntaisen värähtelyn määrää. Lisäksi lineaariregressiolla testataan Y-akselin

maksimivärähtelyllä Group By -yksikön keskiarvo- ja minimiasetuksia, joista paremmin ongelmat esittävä valitaan muihin testeihin. Hälytysrajat pidetään testeissä vakioina.

Kuvissa 58 ja 59 ovat lineaariregressiolla mallinnetut värähtelyt, todelliset värähtelyt sekä kumuloituva hälytystaso Y-akselin suuntaisen värähtelyn maksimiarvojen tunnittaisille keskiarvoille ja minimeille.

Kuva 58 Y-akselin suuntaisten värähtelyjen maksimiarvojen tunnittaiset keskiarvot (yllä, sininen) sekä lineaariregressiolla ennustetut arvot (yllä, punainen) ja kumuloitunut hälytysarvo (alla).

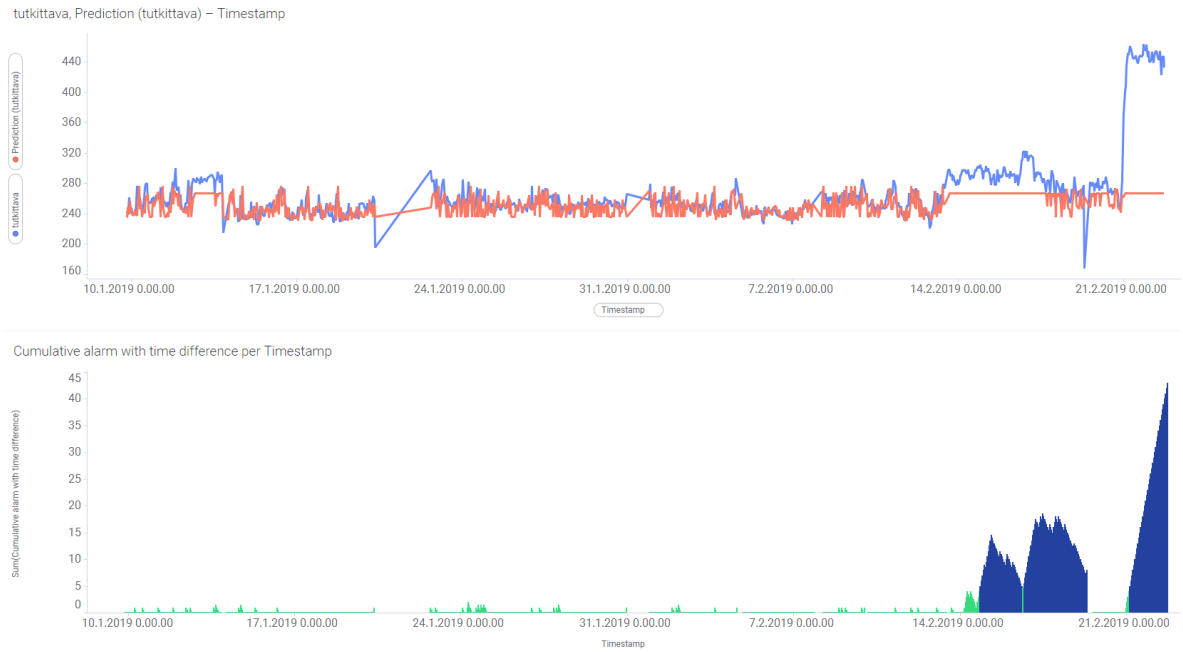


Kuva 59 Y-akselin suuntaisten värähtelyjen maksimiarvojen tunnittaiset minimiarvot (yllä, sininen) sekä lineaariregressiolla ennustetut arvot (yllä, punainen) ja kumuloitunut hälytysarvo (alla).

Kuvien 58 ja 59 perusteella sekä keskiarvoilla että minimiarvoilla muodostetun lineaariregressiomallin virheen avulla kyetään havaitsemaan ongelmatilanne sekä sitä edeltävä värähtelyn omituinen käyttäytyminen. Keskiarvoja käyttäessä myös tarkkailujakson alussa havaittu nousu aiheuttaisi hälytyksen, joskin vain lyhytaikaisesti. Keskiarvojen käyttäminen nostaa esimerkkitapauksessa tarkkailun herkkyyttä ja sitä käytetään muiden regressiomallien tutkimisessa. Myös minimiarvojen käyttäminen olisi täysin mahdollista, mutta rajauksen vuoksi kaikkia testejä ei tehdä molemmilla tavoilla.

Alla kuvissa 60 ja 61 on laskettu määritellyn mukaiset tulokset polynomisella- ja päätöspuuregressiolla.

Kuva 60 Värähtelyjen todelliset arvot (yllä, sininen), polynomisen regression avulla luotu ennustus (yllä, punainen) kullekin arvolle ja kumuloituva hälytystaso (alla) ennustusvirheen perusteella.



Kuva 61 Värähtelyjen todelliset arvot (yllä, sininen), päätöspuuregression avulla luotu ennustus (yllä, punainen) kullekin arvolle ja kumuloituva hälytystaso (alla) ennustusvirheen perusteella.

Polynomisen regression hälytyssignaalin voidaan todeta olevan hyvin pitkälti lineaarista regressiota vastaava, tosin polynomisessa tapauksessa tarkkailtavan ajanjakson alussa oleva värähtelytason verrattain lyhytaikainen nousu ei aiheuta väärää hälytystä. Polynomisen mallin havaitaan myös reagoivan huomattavasti voimakkaammin äkillisiin muutoksiin, ja varsinkin loppuajan voimakkaan nousun jälkeen arvo heittelee voimakkaasti.

Päätöspuuregression ennustukset eroavat lineaarisen- ja polynomisen regression ennusteista huomattavasti. Malli ei kykene lainkaan laajenemaan tuntemiensa alueiden ulkopuolelle, joten suuremmissa värähtelyn kohoamisen tapahtumissa ennuste antaa vaakasuoran viivan. Hälytyssignaali muodostuu samoissa tilanteissa kuin muissakin regressiotapauksissa, eikä vääriä hälytyksiä aiheudu.

Regressioon perustuvan värähtelyn tarkkailun voidaan todeta toimivan systeemin muutosten havaitsemisessa. Työn rajauksen ulkopuolelle jäävien vaihtelevalla nopeudella toimivien koneiden värähtelyn tarkkailuun mallit sopivat myös alustavien testien valossa. Tällöin esimerkiksi automaatiojärjestelmästä saatavien kahdeksan eri värähtelykomponentin avulla pyritään ennustamaan pyörimisnopeutta.

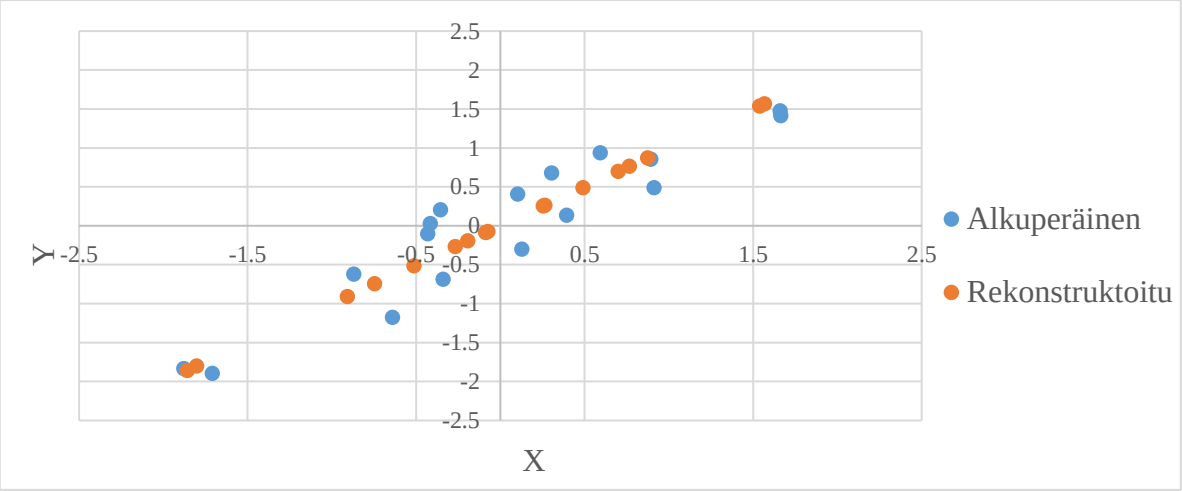
Menetelmän potentiaalisena heikkoutena on mallin sovitus. Regressiomallit sovitetaan automaattisesti parhaimmilla mahdollisilla kertoimilla, minkä vuoksi on mahdollista, että jokin mallia muodostava komponentti saa nollaa lähellä olevan mitättömän painoarvon. Tässä komponentissa näkyvät muutokset eivät tällöin vaikuta ennustukseen, minkä vuoksi komponentissa korostuvat ongelmat saattavat jäädä havaitsematta.

Testitapauksessa kaikki regressiomallit tuottivat samankaltaisia tuloksia. Eri tapauksissa ja jatkotutkimuksessa on siis syytä kokeilla laajemmin mallien välisiä eroja, esimerkiksi vaihtelevalla nopeudella toimivissa laitteissa.

8.3 PCA-rekonstruktioon perustuva värähtelyn tarkkailu

PCA-rekonstruktiota käytettäessä tutkittavat värähtelykomponentit redusoidaan pienempään määrään PCA-komponentteja, minkä jälkeen PCA-komponentit laajennetaan takaisin alkuperäisiksi värähtelyiksi. Tarkkailemalla alkuperäisten värähtelyjen ja laajennuksen suhteellista eroa keskihajontojen määränä voidaan havaita muutos systeemin toiminnassa. Kuvassa 62 on havainnollistamisen vuoksi esitetty kaksi kuvitteellista normalisoitua

värähtelykomponenttia ja näiden rekonstruktoidut arvot yhteen komponenttiin redusoinnin jälkeen.



Kuva 62 Alkuperäiset kuvitteelliset arvot kahdelle värähtelykomponentille ja niiden rekonstruktoidut arvot. Tässä tapauksessa rekonstruktioiden muodostaman suoran alapuolella olevat pisteet liikkuvat vasemmalle ja ylös, yllä olevat oikealle ja alas. Alkuperäisen pisteen ja rekonstruktion etäisyyttä N-ulotteisessa (tässä tapauksessa 2 ulotteisessa) avaruudessa verrataan poikkeuksien löytämiseksi.

PCA:n laskenta suoritetaan normaalin toiminnan normalisoidulle datalle, minkä jälkeen laskettuja normalisointi- ja PCA-malleja käytetään tulevan datan redusointiin ja laajentamiseen. Mallin laskemisen jälkeen tiettyä määrää dataa käytetään mallin keskihajonnan laskemiseen. Mallin toimeenpanossa lasketaan keskihajontojen määrää, joiden perusteella hälytystasoa voidaan nostaa. Alustavien testien perusteella normaalin toiminnan aikana tasaisella nopeudella pyörivän laitteen keskihajontojen määrä käyttäytyy yhdellä PCA-komponentilla jokseenkin normaalijakauman arvojen mukaisesti. Näin ollen tehtäessä oletus normaalin toiminnan normaalijakautumisesta, keskimäärin kolme pistettä tuhannesta ylittää kolmen keskihajonnan rajan. Jos hälytysrajan kumuloituminen suoritetaan regressioon perustuvan menetelmän tavoin, on kolmella keskihajonnalla, normaaliuden oletuksella ja kumuloituneella hälytysrajalla 5 tapahtuvan hälytyksen mahdollisuus täysin normaalissa toiminnassa mitätön.

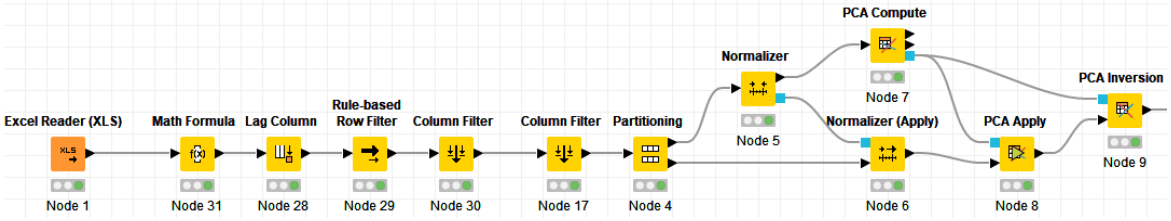
PCA:n rekonstruktiovirheeseen perustuvan värähtelyn tarkkailun etu verrattuna regressiovirheeseen perustuvaan tarkkailuun on sen varma kyky ottaa huomioon kaikki

värähtelykomponentit. Normalisoidun datan avulla jokainen komponentti on virheen suhteen merkityksellinen. Regression tapauksessa osalla komponenteista voi olla nollakertoimia, jotka poistavat kyseisen komponentin merkityksellisyyden.

PCA:han perustuva tarkkailu toimii periaatteessa niin tasaisesti pyörivissä kuin vaihtelevalla nopeudella pyörivissä laitteissa. PCA:n alkuperäisessä muodostuksessa tulee kuitenkin olla dataa laajasti eri pyörimisnopeuksilta, jotta optimaalinen konstruktio saavutetaan. Mikäli nopeuden vaihtelun aiheuttama värähtelytasojen muutos on huomattavan epälineaarista, voi PCA:n keskimääräinen rekonstruktiotarkkuus kärsiä huomattavasti.

8.3.1 PCA-rekonstruktion soveltaminen pumpun värähtelyn tarkkailussa

PCA-rekonstruktion tapauksessa käytetty työnkulku on esitetty alla kuvissa 63 - 65.

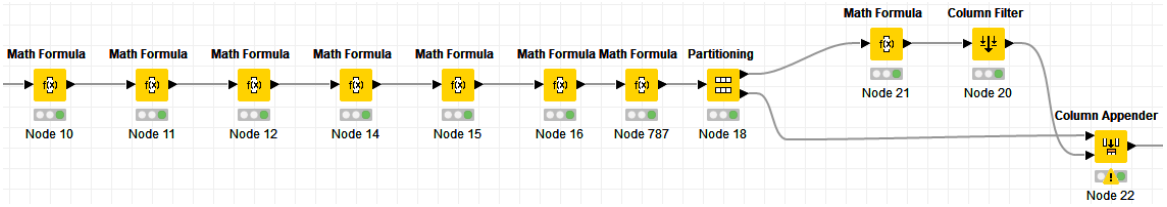


Kuva 63 PCA-rekonstruktiolla tehtävän kunnonvalvonnan KNIME työnkulun aloitus. Kuva sisältää työnkulun vaiheet datan lukemisesta PCA:n inversioon eli alkuperäisten pisteiden rekonstruktioon.

Työnkulku alkaa tunnittaisten värähtelyn keskiarvojen lukemisesta. Lukemisesta neljä seuraavaa yksikköä ovat pumpun poissaolon suodattamiseen tarkoitettuja. Math Formula -yksikössä lasketaan X-akselin maksimivärähtelyn keskiarvo ja jaetaan se kahdella. Lag Column -yksikössä X-akselin maksimivärähtelyn arvo viivästetään kolmeen edelliseen minuuttiin omiksi sarakkeikseen. Rule-based Row Filter -yksikössä viivästettyjä arvoja verrataan Math Formula -yksikössä laskettuun arvoon, ja jos jokin viivästetty tai todellinen arvo on alle yksikössä lasketun rajan, suodatetaan datapiste pois. Ensimmäinen Column Filter -yksikkö on varattu viivästettyjen arvojen ja Math Formulan arvon suodattamiseen. Toisessa Column Filter -yksikössä suodatetaan pois halutut komponentit, testitapauksessa anturin ilmoittama lämpötila.

Partitioning -yksikössä data jaetaan PCA:n laskemiseen käytettyyn dataan ja muuhun dataan. Yhteensä rivejä on 1 267, ja näistä sataa ensimmäistä käytetään mallin laskemiseen. Partitioning -yksiköstä lähtevä ylempi virta sisältää nämä 100 arvoa, jotka ensin normalisoidaan, minkä jälkeen PCA-malli lasketaan. Normalisointiin on mahdollista käyttää minimi – maksimi normalisointia välillä [0, 1] tai ns. Z-Score normalisointia, jonka tuloksena datan keskiarvo on 0 ja sen keskihajonta 1. Normalisoinnin vaikutus tuloksiin on kuitenkin minimaalinen, joten tapauksessa käytetään yleisesti PCA:n kanssa suositeltua Z-Score normalisointia.

Laskettujen normalisointimallin ja PCA-mallin avulla jäljelle jäänyt data Partitioning -yksikön alemmassa virrassa ensin normalisoidaan samalla mallilla kuin ensimmäiset 100 riviä, minkä jälkeen muodostettu PCA-malli ajetaan datalle. PCA-mallissa käytetyt kuusi värähtelykomponenttia (X, Y ja Z, minimi ja maksimi) redusoidaan yhteen PCA-komponenttiin, joka yksikön sisällä olevan tiedon mukaan sisältää 80 % kuudessa komponentissa olleesta informaatiosta. Tämä komponentti laajennetaan välittömästi PCA Inversion -yksikössä takaisin alkuperäisiksi komponenteiksi. Alkuperäiset arvot ovat pysyneet virrassa koko ajan mukana, joten tuloksena on kaikille komponenteille kaksi arvoa, esim. X min ja X min (#1). Työnkulun jatko on esitetty kuvassa 64.

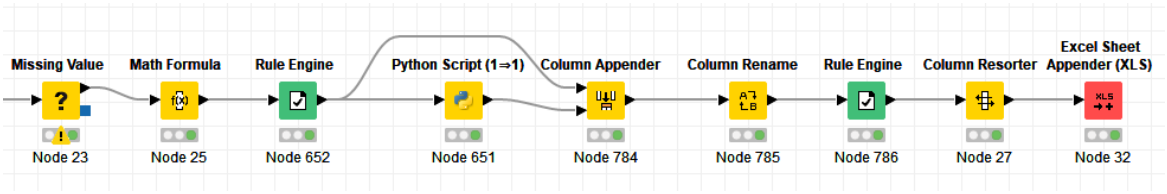


Kuva 64 PCA-rekonstruktiotyönkulun toinen osa. Kuva sisältää vaiheet rekonstruktiovirheiden laskemisesta keskihajonnan määritykseen.

Työnkulun toisen osan kuusi ensimmäistä Math Formula -yksikköä sisältävät kullekin komponentille alkuperäisten ja PCA:n läpikäyneiden arvojen erojen itseisarvot. Seitsemännessä Math Formula -yksikössä lasketaan näiden arvojen summa. Tämän jälkeen Partitioning -yksikössä data jaetaan sataan validointidatapisteeseen ja loppu- eli testidataan.

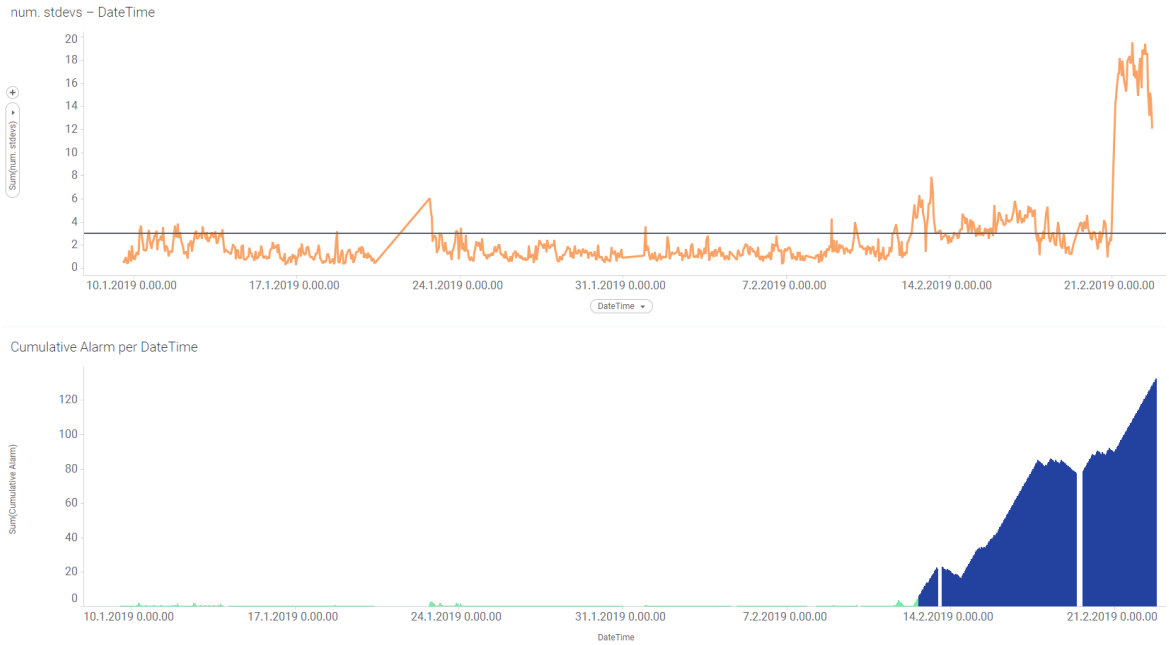
Ylempi virta sisältää validointidatan. Validointidatassa lasketaan Math Formula -yksikössä pisteiden erojen itseisarvojen summan keskiarvo eli PCA:n rekonstruktion

kokonaiskeskihajonta. Column Filter -yksikössä muut sarakkeet poistetaan, ja Column Appender -yksikössä keskihajonta liitetään testidatan sisältävään virtaan. Työnkulun jatko on esitetty kuvassa 65.



Kuva 65 PCA-rekonstruktioon perustuvan värähtelyn tarkkailun työnkulun viimeinen osa. Kuva sisältää työnkulun vaiheet puuttuvien arvojen täydentämisestä Excel-tiedoston kirjoittamiseen.

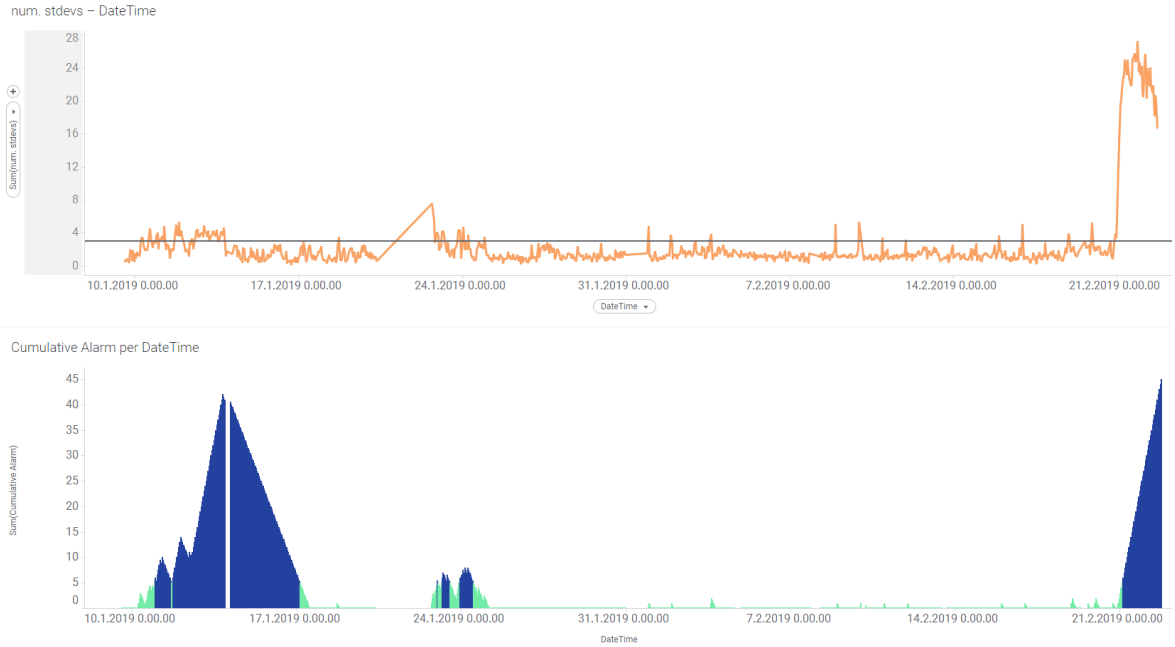
Työnkulun kolmas ja viimeinen osa alkaa keskihajontasarakkeessa olevien tyhjien arvojen täyttämisellä samalla vakioarvolla yksikössä Missing Value. Tämän jälkeisessä Math Formula -yksikössä lasketaan kullekin pisteelle sen rekonstruktion keskihajontojen määrä. Rule Engine -yksikössä määritellään regressiota vastaava kumulatiivisen hälytysarvon muutos, eli rekonstruktiovirheen ollessa yli 3 keskihajontaa lisätään kumulatiiviseen arvoon 1, muutoin lasketaan arvoa puoleella. Python Script -yksikkö sisältää regressioon nähden identtisen kumulatiivisten arvojen laskennan, tosin ilman aikahyppyjen huomioimista. Column Appender -yksikössä tämä tieto lisätään muuhun dataan, minkä jälkeen sarake uudelleennimetään yksikössä Column Rename. Rule Engine -yksikössä määritellään hälytyksen päälläoloksi arvot, jolloin kumulatiivinen hälytys on yli 5, minkä jälkeen sarakkeet uudelleenjärjestellään ja laskettu data kirjoitetaan Excel-tiedostoon. Kuva 66 sisältää Spotfiressa piirretyn rekonstruktiovirheen ja vastaavan hälytystason.



Kuva 66 Rekonstruktion virhe keskihajontoina (yllä) yhteen PCA-komponenttiin redusoinnin jälkeen sekä vastaava kumulatiivinen hälytystaso (alla). Ylemmän kuvan musta vaakaviiva vastaa kolmea keskihajontaa, mikä toimii hälytyksen nostorajana.

Rekonstruktion virheen nähdään reagoivan ongelmatilanteen lisäksi varsinaista ongelmatilannetta edeltäneeseen värähtelytason nousuun. Rekonstruktiovirhekäyrää tulkittaessa nähdään silmin myös tarkkailun kohteena olleen ajanjakson alussa tapahtunut värähtelyn nousu, mutta tällöin keskihajontojen määrä käy yli kolmen vain harvakseltaan. Keskivaiheilla näkyvän pitkän normaalin toiminnan jakson aikana hajonta on oletuksien mukaisesti suurin piirtein yhden tasolla.

Vertailun vuoksi tarkkaillaan vastaavaa tilannetta sillä erolla, että PCA-redusointi tehdään kahteen PCA-komponenttiin. KNIME ilmoittaa tämän säilyttävän tilanteessa 92 % informaatiosta yhden komponentin 80 % sijaan. Kahden komponentin tilanteen vastaavat arvot ovat esitettyinä kuvassa 67.



Kuva 67 Rekonstruktion virhe keskihajontoina (yllä) kahteen PCA-komponenttiin redusoinnin jälkeen sekä vastaava kumulatiivinen hälytystaso (alla). Ylemmän kuvan musta vaakaviiva vastaa kolmea keskihajontaa, mikä toimii hälytyksen nostorajana.

Kahteen komponenttiin redusoidun systeemin havaitaan käyttäytyvän huomattavan eri tavalla kuin yhteen komponenttiin redusoidun. Itse ongelmatilanne korostuu, mutta tämän lisäksi tarkkailujakson alussa oleva värähtelytason nousu näkyy selkeästi. Ongelmatilannetta edeltävä värähtelytason nousu ei sen sijaan ilmene lainkaan kahden komponentin tapauksessa.

Yksittäisten värähtelykomponettejen virheitä tarkastellessa sekä yhden- että kahden komponentin tapauksessa kyetään löytämään alkupään verrattain suuren rekonstruktiovirheen lähde. Virhe on kahden komponentin tapauksessa kuitenkin suhteellisesti suurempi, sillä kahden komponentin rekonstruktion keskihajonta on huomattavasti pienempi. Alkupään värähtelyn nousu näyttää lähes identtiseltä kaikilla värähtelykomponenteilla yhden ja kahden PCA-komponentin tapauksessa. Varsinaista ongelmatilannetta edeltävä värähtelytasojen muutos on sen sijaan kahdella komponentilla hyvin rekonstruktoitavissa, yhdellä komponentilla rekonstruktio on heikko. Tämä näkyy hälytystason nousuna yhden komponentin rekonstruktiossa.

PCA-rekonstruktion virheen voidaan todeta paikantavan poikkeustapauksia datasta. Komponenttien määrällä on suuri merkitys pienempien muutosten havainnointiin. Suuret

muutokset havaitaan komponenttien määrästä riippumatta, mutta riippuen PCA:ssa muodostuvista lineaarikombinaatioista useampaa kuin yhtä komponenttia käyttäessä kaikki erot eivät välttämättä näyttäydy tasaisesti. Yhdellä komponentilla normalisoidulla datalla toimittaessa jokaisella komponentilla on yhtä suuri teoreettinen merkitys, jolloin minkä tahansa värähtelykomponentin muutokset suhteessa toisiin värähtelykomponentteihin ovat havaittavissa rekonstruktiovirheenä. Toinen PCA-komponentti on ortogonaalinen (eli kohtisuorassa) ensimmäiseen komponenttiin nähden, sisältäen maksimaalisen varianssin kohtisuoruusehdolla, mikä jättää muut suunnat huomiotta ja parantaa komponentin suuntaisen hajonnan rekonstruktiota. Keskihajonta on siis käänteisesti verrannollinen komponenttien määrään, mutta ei näy kaiken suuntaisissa hajonnoissa tasaisesti.

8.4 Yhteenveto värähtelyn tarkkailusta pumpussa

Pumppujen ja muiden pyörivien laitteiden kunnonvalvonta värähtelyn perusteella on mahdollista monin eri tavoin. Kaikki työssä testatut menetelmät olisivat oikein määritettyinä kyenneet havaitsemaan tutkitun tapauksen tiivisteveden kierron tukoksen ja sitä seuranneen värähtelytason nousun ilman vääriä hälytyksiä.

Autokorrelatiivinen värähtelyn tarkkailu soveltuu yksittäisten värähtelymittausten tarkkailuun, mikäli halutaan välttää kovia rajoja. Menetelmä on riippuvainen datan hajonnasta. Hälytykset voidaan menetelmässä antaa suoraan autokorrelaatioarvon perusteella.

Regression virheeseen perustuvassa värähtelyn tarkkailussa on suurta potentiaalia sekä tasaisella- että vaihtelevalla nopeudella pyörivien laitteiden tarkkailussa. Ennustuksen kohteena oleva komponentti pitää valita ja ennustavien tekijöiden kertoimien on oltava jossain määrin merkityksellisiä, jotta kaikkien värähtelykomponenttien tarkkailu onnistuu. Hälytykset tulee menetelmässä antaa kumuloituvien huonojen ennustusten perusteella väärien hälytysten välttämiseksi.

PCA:n rekonstruktiovirheeseen perustuva värähtelyn tarkkailu soveltuu ainakin vakionopeudella toimivien laitteiden tarkkailuun ja periaatteen tasolla myös vaihtelevalla nopeudella toimiviin laitteisiin. Menetelmä vaatii useita eri värähtelyarvoja eri akseleilta

osittain riippumattomien ominaisuuksien tarpeen vuoksi. Testeissä kuuden värähtelyarvon redusointi yhteen PCA-komponenttiin antoi parhaat tulokset.

9. Johtopäätökset

KNIME valittiin työssä parhaiten soveltuvaksi ohjelmistoksi juurisyyanalyysiin. KNIME soveltuu hyvin offline -datan analysointiin, minkä lisäksi sillä on nopea rakentaa Proof of Concept -malleja, joilla voidaan testata mahdollisesti online-tarkkailussa käytettävien menetelmien käytännöllisyyttä. KNIMEn lisäksi TIBCO Spotfirea käytettiin työssä datan visualisointiin.

Teollisuuden ongelmien juurisyyanalyysissa monet menetelmät eivät ole tilastollisesti päteviä, vaan toimivat analyysissa suunnan antajina. Tilastollinen pätemättömyys aiheutuu pääasiassa pienestä ongelmatilanne – normaalitilanne suhteesta. 30:n ongelmatilanteen vertaaminen 20 000 normaalitilanteeseen aiheuttaa välttämättä ongelmia. Esimerkkitapauksessa suunnananto oli kuitenkin merkittävimpien ominaisuuksien osalta tarkkaa ja ohjasi nopeasti todellisen juurisyyn luokse. Varianssianalyysi ANOVA oli tutkitussa tapauksessa yksinkertaisin oikeaan suuntaan ohjaava menetelmä.

Ongelmatilanteiden luokittelun tapauksessa pyrittiin käytännössä ennustamaan tuotannon ongelmaa yhdellä minuutilla. Ennustaminen olisi ollut mahdollista noin puolessa tapauksista järkevällä False Positive (FP) -taajuudella, korkeampiin löytöasteisiin mennessä FP -taajuus kasvaa huomattavasti. Parhaiten testeissä pärjäsi Random Forest -algoritmin tuottama luokittelija, kun käytettiin kaikkia datassa olleita muuttujia. ANOVA:n F-arvoihin ja Levenen testin arvoihin perustuvalla muuttujanvalinnalla päästiin kuitenkin parhaisiin tuloksiin muilla, kuin Random Forest -algoritmilla, ja Random Forest -algoritmin tapauksessakaan ero näiden kahden muuttujanvalintamenetelmän välillä ei ollut erityisen suuri. Työssä tutkitussa tapauksessa juurisyiden korjaaminen olisi ennustusta hyödyllisempi vaihtoehto, mutta tilanteissa, joissa korjaaminen ei ole mahdollista voi ennustuksista olla hyötyä.

Pumpun värähtelytasoa tutkittiin kolmella menetelmällä. Varsinkin regressioon- ja PCA:han perustuvat menetelmät vaikuttavat mielenkiintoisilta normaalista poikkeavien tilanteiden

tarkkailussa. Molemmat menetelmät soveltuvat periaatteessa niin vakionopeudella kuin vaihtelevalla nopeudella toimiville laitteille, mutta tässä työssä tarkkailtiin ainoastaan vakionopeudella toimivaa laitetta. Jatkotutkimuksessa olisi mahdollista tarkkailla vaihtelevalla nopeudella toimivia laitteita ja tarkkailla yksiluokkaluokittelun toimivuutta kunnonvalvonnassa. Yksiluokkaluokittelun lisäksi Autoencoder -neuroverkkojen soveltaminen värähtelyn tarkkailuun on kiinnostava idea. Myös automaattinen poikkeuksien tunnistaminen spektrejä ja trendidataa yhdistämällä on mahdollinen tutkimuksen kohde.

Käytännössä automaattisen kunnonvalvonnan suurin haaste on eri datanlähteiden integrointi järjestelmään, johon päästään helposti käsiksi esimerkiksi Python-ohjelmoinnilla. Kun data on saatavilla, on kehitettävä helposti alustettava sovellus, johon voidaan valita ominaisuudet ja tarkkailtavat suureet. Sovelluksen tulee olla kykenevä lähettämään hälytyksiä esimerkiksi kunnonvalvonnan sähköpostilaatikkoon, minkä perusteella vastaanottaja tarkastaa kohteen automaatiojärjestelmästä ja tarpeen tullen fyysisesti. Vaihtoehtoisesti eri kohteiden hälytystasoista olisi mahdollista muodostaa dashboard-näkymä, jonka avulla mittaajat voisivat helposti silmäyksellä löytää kiinnostavimmat kohteet esimerkiksi värikoodeilla esitetyn hälytystason perusteella. Ohjelmistossa olisi hyvä olla helppo mahdollisuus uudelleenalustukseen esimerkiksi lähetetyn sähköpostin perusteella tai dashboardin kautta. Hälytyssähköposti voisi sisältää kirjallisen varoituksen, joka sisältää kohteen nimen ja sijainnin, sekä kuvaajat hälyttävästä laitteesta. Tämän lisäksi sähköpostin lopussa voisi olla linkki, jota klikkaamalla vastaanottajan on mahdollista alustaa malli uudelleen useiden väärien hälytysten tapauksessa.

Älykkäällä datan käytöllä voidaan parantaa teollisuuden kunnossapitoa ja käyttövarmuutta niin ongelmien juurisyitä löytämällä kuin kunnonvalvontaa parantamalla. Suurin este suuremmalle datan hyödyntämiselle teollisuudessa on dataan käsiksi pääsyn vaikeus ja eri tallennuspaikkojen suuri määrä. Yhtenäistämällä datansa sijainnin olisi yrityksien huomattavasti helpompi saada maksimaalinen arvo irti datastaan.

10. Kirjallisuus

Agarwal, P. & Biswas, S. S., 2017. Big Data on Cloud: Review. *International Journal of Advanced Research in Computer Science,* 8(2 (Special Issue)), pp. 49-51.

Alzghoul, A. & Löfstrand, M., 2011. Increasing availability of industrial systems through data stream mining. *Computers & Industrial Engineering,* Osa/vuosikerta 60, pp. 195-205.

AmericanAirFilter, 2012. *Case Study Pulp and Paper mills.* [Online] Available at: https://customers.aafintl.com/Portal%20News/~/media/product%20media/Case%20Studies/Paper%20Mill%20Uses%20SAAFShield%20to%20Ensure%20Productivity%20GPF%206%20104%20AUG12%20pdf.ashx
[Haettu 8 March 2019].

Barga, R., Fontama, V. & Tok, W. H., 2015. *Predictive Analytics with Microsoft Azure Machine Learning.* 2nd toim. s.l.:Apress.

Borchert, D. ym., 2018. Comparison of data science workflows for root cause analysis of bioprocesses. *Bioprocess and Biosystems Engineering.*

Bumblauskas, D., Gemmil, D., Igou, A. & Anzengruber, J., 2017. Smart Maintenance Decision Support Systems (SMDSS) based on corporate big data analytics. *Expert Systems With Applications,* Osa/vuosikerta 90, pp. 303-317.

Chiroma, H., Abdulhamid, S. M. & Liyana, S., 2018. Progress on Artificial Neural Networks for Big Data Analytics: A Survey. *IEEE Access.*

Dougherty, G., 2013. *Pattern Recognition and Classification: An Introduction.* New York: Springer Science.

Fahad, A. ym., 2014. A Survey of Clustering Algorithms for Big Data: Taxonomy and Empirical Analysis. *IEEE Transactions on Emerging Topics in Computing,* 2(3), pp. 267-279.

Fawcett, T., 2003. *ROC Graphs: Notes and Practical Considerations for Data Mining Researchers,* Palo Alto: HP Laboratories/Intelligent Enterprise Technologies Laboratory.

Frampton, M., 2015. *Big Data Made Easy: A Working Guide to the Complete Hadoop Toolset.* s.l.:Apress.

Fu, J. & Hart, P. W., 2016. Leveraging mill-wide big data sets for process and quality improvement in paperboard production. *Tappi Journal,* 15(5), pp. 309-319.

Furht, B. & Villanustre, F., 2016. *Big Data Technologies and Applications.* Cham: Springer International Publishing.

Gandhi, R., 2018. *Naive Bayes Classifier.* [Online] Available at: https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c
[Haettu 7 March 2019].

Gartner Inc., 2019. *Magic Quadrant for Data Science and Machine Learning Platforms.* [Online]
Available at: https://www.gartner.com/doc/reprints?id=1-65WC0O1&ct=190128&st=sb
[Haettu 18 February 2019].

GE Digital, 2019. *APM Reliability.* [Online]
Available at: https://www.ge.com/digital/sites/default/files/download_assets/APM-Reliability-from%20GE-Digital-brochure.pdf
[Haettu 13 February 2019].

Glosser.ca, 2013. *File:Colored neural network.svg.* [Online]
Available at: https://en.wikipedia.org/wiki/File:Colored_neural_network.svg
[Haettu 4 March 2019].

Guller, M., 2015. *Big Data Analytics with Spark: A Practitioner's Guide to Using Spark for Large Scale Data Analysis.* s.l.:Apress.

Haroon, D., 2017. *Python Machine Learning Case Studies.* Karachi: Apress.

IBM [2], 2019. *Watson Machine Learning.* [Online]
Available at: https://www.ibm.com/cloud/machine-learning
[Haettu 26 February 2019].

IBM [3], 2019. *IBM Watson Studio.* [Online]
Available at: https://www.ibm.com/cloud/watson-studio
[Haettu 26 February 2019].

IBM [4], 2019. *Watson Machine Learning: Pricing.* [Online]
Available at: https://www.ibm.com/cloud/machine-learning/pricing
[Haettu 26 February 2019].

IBM [5], 2017. *IBM Predictive Maintenance and Optimization.* [Online]
Available at: https://www.ibm.com/downloads/cas/7JELWLM8
[Haettu 26 February 2019].

IBM [6], 2019. *IBM Watson Studio Desktop: Pricing.* [Online]
Available at: https://www.ibm.com/products/watson-studio-desktop/pricing
[Haettu 30 April 2019].

IBM, 2017. *Watson IoT / Kone.* [Online]
Available at: https://www.ibm.com/watson/stories/kone/
[Haettu 26 February 2019].

Ironside Group, 2019. *Information for All: Watson Analytics and IBM's Self-Service Revolution.* [Online]
Available at: https://www.ironsidegroup.com/2015/01/07/information-for-all-watson-analytics-and-ibms-self-service-revolution/
[Haettu 24 June 2019].

Josefsson, T., 2017. *Root-cause analysis through machine learning in the cloud*, Uppsala: University of Uppsala.

Klein, S., 2017. *IoT Solutions in Microsoft's Azure IoT Suite.* Redmond: Apress.

KNIME AG [2], 2018. *KNIME Server.* [Online] Available at: https://www.knime.com/knime-software/knime-server-pricing [Haettu 28 March 2019].

KNIME AG [3], 2019. *KNIME License Terms and Conditions.* [Online] Available at: https://www.knime.com/downloads/full-license [Haettu 6 June 2019].

KNIME AG [4], 2019. *FAQ.* [Online] Available at: https://www.knime.com/faq#usage_data [Haettu 6 June 2019].

KNIME AG [5], 2018. *KNIME Software.* [Online] Available at: https://www.knime.com/knime-software [Haettu 24 June 2019].

KNIME AG, 2019. *KNIME Open Source Story.* [Online] Available at: https://www.knime.com/knime-open-source-story [Haettu 28 March 2019].

Lee, J. Y., Yoon, J. S. & Kim, B.-H., 2017. A big data analytics platform for smart factories in small and medium-sized manufacturing enterprises: An empirical case study of a die casting factory. *International Journal of Precision Engineering and manufacturing,* 18(10), pp. 1353-1361.

L'Heureux, A., Grolinger, K., Elyamany, H. F. & Capretz, M. A. M., 2017. Machine Learning With Big Data: Challenges and Approaches. *IEEE Access,* Osa/vuosikerta 5, pp. 7776-7797.

Love, J., 2007. *Process Automation Handbook: A Guide to Theory and Practice.* London: Springer.

McCann, M. & Johnston, A., 2008. *SECOM Data Set.* [Online] Available at: http://archive.ics.uci.edu/ml/machine-learning-databases/secom/secom.names [Haettu 7 March 2019].

Mechelucci, U., 2018. *Applied Deep Learning: A Case-Based Approach to Understanding Deep Neural Networks.* s.l.:Apress.

Microsoft [1], 2019. *Microsoft Azure.* [Online] Available at: https://azure.microsoft.com/en-us/ [Haettu 12 February 2019].

Microsoft [2], 2015. *Predictive Maintenance: Step 2C of 3, train and avaluation multi-class classification models.* [Online] Available at: https://gallery.azure.ai/Experiment/Predictive-Maintenance-Step-2C-of-3-train-and-evaluation-multi-class-classification-models-2 [Haettu 12 February 2019].

Microsoft [3], 2019. *Advanced analytics on big data.* [Online] Available at: https://azure.microsoft.com/en-us/solutions/architecture/advanced-analytics-on-big-data/
[Haettu 13 February 2019].

Microsoft [4], 2019. *Introducing the Cortana Analytics Suite.* [Online] Available at: https://azure.microsoft.com/en-us/resources/videos/azurecon-2015-introducing-the-cortana-analytics-suite/
[Haettu 20 February 2019].

Microsoft [5], 2015. *Introduction to Cortana Intelligence Suite.* [Online] Available at: https://social.technet.microsoft.com/wiki/contents/articles/36688.introduction-to-cortana-intelligence-suite.aspx
[Haettu 20 February 2019].

Microsoft [6], 2017. *Machine learning algorithm cheat sheet for Azure Machine Learning Studio.* [Online] Available at: https://docs.microsoft.com/en-us/azure/machine-learning/studio/algorithm-cheat-sheet
[Haettu 28 February 2019].

Microsoft [7], 2019. *Pricing calculator.* [Online] Available at: https://azure.microsoft.com/en-us/pricing/calculator/
[Haettu 28 February 2019].

Microsoft [8], 2019. *Daimler AG - digital transformation with big data and AI.* [Online] Available at: https://customers.microsoft.com/en-gb/story/daimler-extollo-azure-hdinsight-sql-database-key-vault-lake-store-analytics-automotive-germany-en
[Haettu 20 February 2019].

Microsoft [9], 2019. *Machine Learning Studio pricing.* [Online] Available at: https://azure.microsoft.com/en-us/pricing/details/machine-learning-studio/
[Haettu 30 April 2019].

Microsoft, 2019. *AI for Azure / Predictive maintenance.* [Online] Available at: https://azure.microsoft.com/en-us/use-cases/predictive-maintenance/
[Haettu 12 February 2019].

National Institute of Standards and Technology, 2013. *Upper Critical Values of the F Distribution.* [Online] Available at: https://www.itl.nist.gov/div898/handbook/eda/section3/eda3673.htm
[Haettu 15 March 2019].

Nazarathy, Y., 2019. *University of Queensland.* [Online] Available at: https://people.smp.uq.edu.au/YoniNazarathy/stat_models_B_course_spring_07/distributions/chisqtab.pdf
[Haettu 14 February 2019].

NIST/SEMATECH, 2012. *e-Handbook of Statistical Methods.* [Online] Available at: https://www.itl.nist.gov/div898/handbook/eda/section3/eda35a.htm [Haettu 10 May 2019].

Nykyri, M., 2018. *Data analytics for predictive maintenance in a pulp mill - case electric motors.* Lappeenranta: Lappeenranta University of Technology.

O'Donovan, P., Leahy, K., Bruton, K. & O'Sullivan, D. T. J., 2015. An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. *Journal of Big Data,* 2(25).

Prusty, M. R., Jayanthi, T. & Velusamy, K., 2017. Weighted-SMOTE: A modification to SMOTE for event classification in sodium cooled fast reactors. *Progress in Nuclear Energy,* Osa/vuosikerta 100, pp. 355-384.

PSK Standardisointiyhdistys ry, 2011. *PSK Standardisointi PSK 6201.* 3 toim. s.l.:s.n.

Pyne, S., Rao, P. B. L. S. & Rao, S. B. toim., 2016. *Big Data Analytics: Methods and Applications.* New Delhi: Springer India.

Rodriguez, R., 2019. *Threat Hunting with Jupyter Notebooks—Part 3: Querying Elasticsearch via Apache Spark.* [Online] Available at: https://posts.specterops.io/threat-hunting-with-jupyter-notebooks-part-3-querying-elasticsearch-via-apache-spark-670054cd9d47 [Haettu 24 June 2019].

Runkler, T. A., 2016. *Data Analytics: Models and Algorithms for Intelligent Data Analysis.* 2nd toim. s.l.:Springer Vieweg.

Ruschel, E., Santos, E. A. P. & Loures, E. d. F. R., 2017. Industrial maintenance decision-making: A systematic literature review. *Journal of Manufacturing Systems,* Osa/vuosikerta 45, pp. 180-194.

Ryżko, D., Gawrysiak, P., Kryszkiewicz, M. & Rybiński, H. toim., 2016. *Machine Intelligence and Big Data in Industry.* s.l.:Springer.

Sahai, H. & Ojenda, M. M., 2005. *Analysis of Variance for Random Models Volume II: Unbalanced Data.* Boston: Birkhäuser.

Salem, M., Taheri, S. & Yuan, J.-S., 2018. An Experimental Evaluation of Fault Diagnosis from Imbalanced and Incomplete Data for Smart Semiconductor Manufacturing. *Big Data and Cognitive Computing,* Osa/vuosikerta 2.

Sartorius Stedim Biotech [2], 2019. *Easy Analytics (pricing).* [Online] Available at: https://webshop.umetrics.com/collections/Easy-Analytics [Haettu 4 March 2019].

Sartorius Stedim Biotech, 2019. *Easy Analytics.* [Online] Available at: https://umetrics.com/product/umetrics-easy-analytics [Haettu 4 March 2019].

Sashruthi, S., 2018. *Linear Regression - Detailed View.* [Online] Available at: https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86 [Haettu 26 March 2019].

scikit-learn developers, 2018. *Choosing the right estimator.* [Online] Available at: https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html [Haettu 7 March 2019].

Sigma Plus Statistiek, 2019. *Chi-Square Independence Test - What and Why?.* [Online] Available at: https://www.spss-tutorials.com/chi-square-independence-test [Haettu 14 February 2019].

Silipo, R., Adä, I. & Winters, P., 2018. *Anomaly Detection in Predictive Maintenance.* [Online]
Available at: https://files.knime.com/sites/default/files/181212_Whitepaper_Anomaly_Detection_Predictive_Maintenance_KNIME.pdf
[Haettu 28 March 2019].

Singh, D. & Reddy, C. K., 2014. A survey on platforms for big data analytics. *Journal of Big Data.*

Skymind, 2019. *A Beginner's Guide to LSTMs and Recurrent Neural Networks.* [Online] Available at: https://skymind.ai/wiki/lstm#feedforward [Haettu 13 March 2019].

Spearman, C., 2010. (1904) The proof and measurement of association between two things. *International Journal of Epidemiology. Reprint,* Osa/vuosikerta 39, pp. 1137-1150.

Statistics How To, 2019. *What is Kendall's Tau?.* [Online] Available at: https://www.statisticshowto.datasciencecentral.com/kendalls-tau/ [Haettu 24 June 2019].

Susto, G. A. ym., 2015. Machine Learning for Predictive Maintenance: A Multiple Classifier Approach. *IEEE Transactions on Idustrial Infromatics,* 11(3), pp. 812-820.

Tax, D. M. J., 2001. *One-class classification - Concept-learning in the absence of counter-examples.* s.l.:Advanced School for Computing and Imaging.

TIBCO Software Inc. [2], 2019. *2018 Gartner Critical Capabilities for Data Science and Machine Learning Platforms.* [Online]
Available at: https://www.tibco.com/resources/analyst-report/gartner-critical-capabilities-data-science-and-machine-learning
[Haettu 18 February 2019].

TIBCO Software Inc. [3], 2019. *TIBCO Spotfire Editions and Pricing.* [Online] Available at: https://www.tibco.com/products/tibco-spotfire/editions-pricing [Haettu 28 February 2019].

TIBCO Software Inc. [4], 2018. *Mercedes-AMG Petronas Motorsport Handles Constant Change - Success Story.* [Online]

Available at: https://www.tibco.com/resources/success-story/mercedes-amg-petronas-motorsport-handles-constant-change-success-story
[Haettu 28 February 2019].

TIBCO Software Inc. [5], 2014. *Data Relationships Anova Algorithm.* [Online] Available at: https://docs.tibco.com/pub/spotfire/6.5.1/doc/html/intro/intro_introduction.htm
[Haettu 15 March 2019].

TIBCO Software Inc., 2019. *Big Data Analytics with TIBCO Spotfire.* [Online] Available at: https://www.tibco.com/products/tibco-spotfire/big-data-analytics
[Haettu 18 February 2019].

TIBCO Software Inc., 2019. *TIBCO Spotfire X.* [Online] Available at: https://www.tibco.com/products/tibco-spotfire
[Haettu 13 February 2019].

Tibshirani, R., 1996. Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, Osa/vuosikerta 58, pp. 267-288.

van Veen, F., 2016. *The Neural Network Zoo.* [Online] Available at: https://www.asimovinstitute.org/author/fjodorvanveen/
[Haettu 8 March 2019].

Wan, J. ym., 2017. A Manufacturing Big Data Solution for Active Preventive Maintenance. *IEEE Transactions on Industrial Informatics,* 13(4), pp. 2039-2047.

Vidal, R., Ma, Y. & Shankar Sastry, S., 2016. *Generalized Principal Component Analysis.* New York: Springer.

Witten, I. H., Frank, E., Hall, M. A. & Pal, C. J., 2016. *The WEKA Workbench: Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*, s.l.: Morgan Kaufmann.

Zhang, L., 2016. *Big Data Analytics for Fault Detection and its Application in Maintenance, Doctoral thesis.* Luleå: Luleå University of Technology.

11. Liitteet

LIITE I χ^2-jakauma -taulukko

LIITE II Algorithm Cheat Sheet, Microsoft

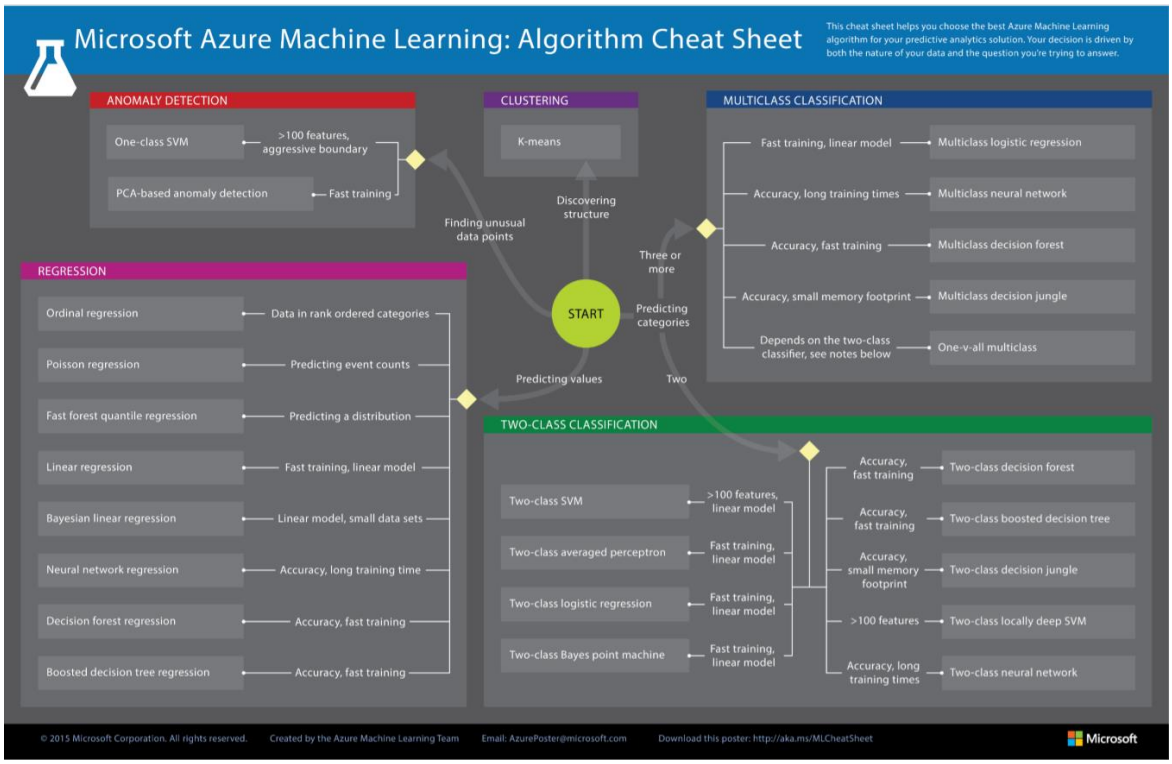
LIITE III Algorithm Cheat Sheet, scikit-learn

LIITE IV Ajotilanteen ja ominaisuuksien väliset Pearsonin korrelaatiot

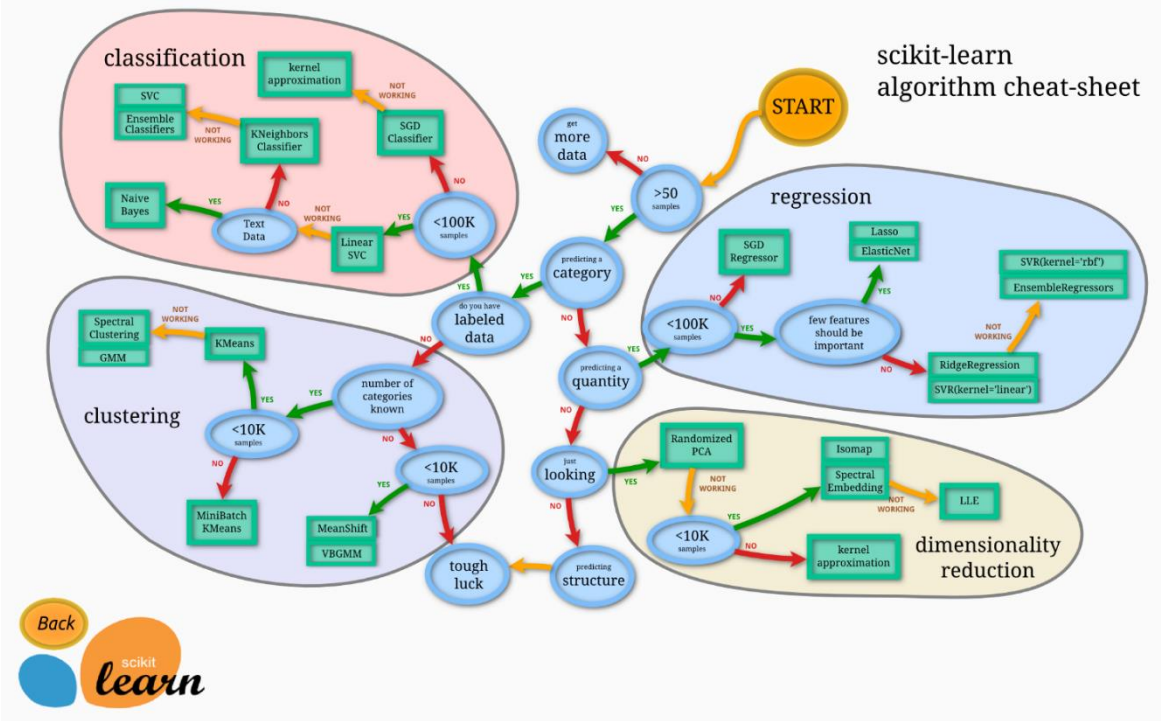
Chi-square Distribution Table

d.f.	.995	.99	.975	.95	.9	.1	.05	.025	.01
1	0.00	0.00	0.00	0.00	0.02	2.71	3.84	5.02	6.63
2	0.01	0.02	0.05	0.10	0.21	4.61	5.99	7.38	9.21
3	0.07	0.11	0.22	0.35	0.58	6.25	7.81	9.35	11.34
4	0.21	0.30	0.48	0.71	1.06	7.78	9.49	11.14	13.28
5	0.41	0.55	0.83	1.15	1.61	9.24	11.07	12.83	15.09
6	0.68	0.87	1.24	1.64	2.20	10.64	12.59	14.45	16.81
7	0.99	1.24	1.69	2.17	2.83	12.02	14.07	16.01	18.48
8	1.34	1.65	2.18	2.73	3.49	13.36	15.51	17.53	20.09
9	1.73	2.09	2.70	3.33	4.17	14.68	16.92	19.02	21.67
10	2.16	2.56	3.25	3.94	4.87	15.99	18.31	20.48	23.21
11	2.60	3.05	3.82	4.57	5.58	17.28	19.68	21.92	24.72
12	3.07	3.57	4.40	5.23	6.30	18.55	21.03	23.34	26.22
13	3.57	4.11	5.01	5.89	7.04	19.81	22.36	24.74	27.69
14	4.07	4.66	5.63	6.57	7.79	21.06	23.68	26.12	29.14
15	4.60	5.23	6.26	7.26	8.55	22.31	25.00	27.49	30.58
16	5.14	5.81	6.91	7.96	9.31	23.54	26.30	28.85	32.00
17	5.70	6.41	7.56	8.67	10.09	24.77	27.59	30.19	33.41
18	6.26	7.01	8.23	9.39	10.86	25.99	28.87	31.53	34.81
19	6.84	7.63	8.91	10.12	11.65	27.20	30.14	32.85	36.19
20	7.43	8.26	9.59	10.85	12.44	28.41	31.41	34.17	37.57
22	8.64	9.54	10.98	12.34	14.04	30.81	33.92	36.78	40.29
24	9.89	10.86	12.40	13.85	15.66	33.20	36.42	39.36	42.98
26	11.16	12.20	13.84	15.38	17.29	35.56	38.89	41.92	45.64
28	12.46	13.56	15.31	16.93	18.94	37.92	41.34	44.46	48.28
30	13.79	14.95	16.79	18.49	20.60	40.26	43.77	46.98	50.89
32	15.13	16.36	18.29	20.07	22.27	42.58	46.19	49.48	53.49
34	16.50	17.79	19.81	21.66	23.95	44.90	48.60	51.97	56.06
38	19.29	20.69	22.88	24.88	27.34	49.51	53.38	56.90	61.16
42	22.14	23.65	26.00	28.14	30.77	54.09	58.12	61.78	66.21
46	25.04	26.66	29.16	31.44	34.22	58.64	62.83	66.62	71.20
50	27.99	29.71	32.36	34.76	37.69	63.17	67.50	71.42	76.15
55	31.73	33.57	36.40	38.96	42.06	68.80	73.31	77.38	82.29
60	35.53	37.48	40.48	43.19	46.46	74.40	79.08	83.30	88.38
65	39.38	41.44	44.60	47.45	50.88	79.97	84.82	89.18	94.42
70	43.28	45.44	48.76	51.74	55.33	85.53	90.53	95.02	100.43
75	47.21	49.48	52.94	56.05	59.79	91.06	96.22	100.84	106.39
80	51.17	53.54	57.15	60.39	64.28	96.58	101.88	106.63	112.33
85	55.17	57.63	61.39	64.75	68.78	102.08	107.52	112.39	118.24
90	59.20	61.75	65.65	69.13	73.29	107.57	113.15	118.14	124.12
95	63.25	65.90	69.92	73.52	77.82	113.04	118.75	123.86	129.97
100	67.33	70.06	74.22	77.93	82.36	118.50	124.34	129.56	135.81

(Nazarathy, 2019)



(Microsoft [6], 2017)



(scikit-learn developers, 2018)

Prosessimuuttuja	Korrelaatio
Ajotilanne	1.000
0	-0.033
1	-0.038
2	-0.044
3	0.001
4	0.006
5	-0.045
7	-0.003
8	-0.026
10	-0.026
12	0.045
13	0.062
14	0.061
15	0.027
16	0.043
17	0.005
18	-0.029
19	0.021
20	0.073
22	0.003
23	0.019
25	0.004
26	0.126
28	0.074
29	0.012
30	-0.026
31	0.001
32	-0.009
33	-0.021

Prosessimuuttuja	Korrelaatio
34	-0.024
35	-0.003
36	-0.016
37	-0.023
38	-0.001
39	0.028
40	0.055
41	0.038
42	0.031
43	0.007
44	-0.052
45	-0.046
46	-0.052
47	-0.044
48	0.008
49	-0.027
50	-0.026
51	-0.026
52	0.050
53	-0.034
54	0.005
55	0.005
56	0.207
57	0.122
58	0.167
59	0.027
60	-0.064
Nopeussuhde	-0.001