OULUN AMMATTIKORKEAKOULU

Ville-Veikko Komulainen

TUKIASEMAN OHJELMISTOTESTAUKSEN AIKAISEN SUORITINKÄYTTÖDATAN PROFILOINTI JA POIKKEAVUUKSIEN HAVAITSEMINEN

TUKIASEMAN OHJELMISTOTESTAUKSEN AIKAISEN SUORITINKÄYTTÖDATAN PROFILOINTI JA POIKKEAVUUKSIEN HAVAITSEMINEN

Ville-Veikko Komulainen
Opinnäytetyö
Syksy 2019
Tietotekniikan tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan tutkinto-ohjelma, ohjelmistokehityksen suuntautumisvaihtoehto

---

Tekijä: Ville-Veikko Komulainen
Opinnäytetyön nimi suomeksi: Tukiaseman ohjelmistotestauksen aikaisen suoritinkäyttödatan profilointi ja poikkeavuuksien havaitseminen
Opinnäytetyön nimi englanniksi: Data profiling and anomaly detection of Base Transceiver Station software testing CPU data
Työn ohjaajat: Sami Aho, Kari Jyrkkä
Työn valmistumislukukausi ja -vuosi: Syksy 2019
Sivumäärä: 73

Opinnäytetyön tavoitteena oli profiloida tukiasemaohjelmiston testauksen aikana tuotettua suoritinkäyttödataa ja perehtyä poikkeavuuksien havaitsemisessa käytettäviin algoritmeihin ja menetelmiin. Niiden käyttökelpoisuutta tutkittiin ja niiden käytettävyyttä muiden metriikoiden analysoinnissa arvioitiin.

Työ toteutettiin koostamalla aineistot käyttäen sekä hyväksytyksi että hylätyiksi luokiteltujen testien suoritinkäyttödataa. Datan pohjalta luotiin yksinkertainen dataprofiili, joka kuvaa tukiaseman suorittimen keskimääräistä käyttäytymistä testauksen aikana. Hyväksyttyjä ja hylättyjä testejä vertailtiin keskiarvotestin pohjalta luotuun viuhkaprofiiliin. Edistyneemmistä poikkeavuuksien havaitsemisessa käytetyistä menetelmistä tutkittiin alipäästösuodatukseen perustuvia menetelmiä, ARIMA-mallia, LSTM-neuroverkkoa, K-Means-klusterointia ja matriisiprofiilia.

Yksinkertainen vertailu keskiarvotestiin osoittautui menetelmistä tehokkaimmaksi. Opinnäytetyön avulla saatiin paljon pohjustavaa tietoa erilaisten poikkeavuuksien havaitsemisessa käytettyjen menetelmien käyttökelpoisuudesta. Tuloksia voidaan hyödyntää Nokian sisäisten lokianalyysijärjestelmien ja -käytäntöjen jatkojalostamisessa ja kehittämisessä. Työn konkreettinen lopputulos on yli 90-sivuinen Jupyter Notebook -dokumentti. Dokumenttiin on koottu kaavioiden, selitysten ja hyödyllisten linkkien lisäksi toimivaksi havaittuja koodiesimerkkejä, joita voidaan yrittää integroida nykyisiin ja tuleviin lokien analysoinnissa käytettäviin järjestelmiin.

Asiasanat: dataprofilointi, poikkeavuuksien havaitseminen, data-analyysi, tilastolliset menetelmät, ohjelmistotestaus, aikasarja-analyysi

ABSTRACT

Oulu University of Applied Sciences
Degree Programme of Information Technology, Software Development

---

Author: Ville-Veikko Komulainen
Title of thesis: Data profiling and anomaly detection of Base Transceiver Station software testing CPU data
Supervisors: Sami Aho, Kari Jyrkkä
Term and year when the thesis was submitted: Autumn 2019
Pages: 73

The thesis was written for Nokia Solutions and Networks. The goal was to explore methods and techniques used in data profiling and anomaly detection using base transceiver CPU usage data collected during software testing phase. Work done on the thesis was a part of Nokia's ongoing effort to further improve and automate software testing analysis using collected logs.

A dataset comprised of both passed and failed test runs was collected. Based on the data profiles of passed tests, an averaged data profile was created. Tests were compared against the averaged data profile to judge whether it can discern between failed and passed tests. Other methods used in time series anomaly detection were also explored. These methods included low-pass filter based techniques, ARIMA-models, LSTM neural networks, K-Means-clustering and a novel matrix profile-based technique.

The simplest method based on averaged data profile was determined to be the most accurate in determining correctly whether a test is passed or failed. The final product of the thesis was a Jupyter Notebook with concrete code examples, useful links, figures and explanations and results on the used techniques. The Jupyter Notebook can be used as a basis for future improvements as well as useful repository of working Python code to be used in data clean up and analysis.

Keywords: Data profiling, anomaly detection, data analysis, statistical methods, software testing, time series analysis

ALKULAUSE

Tämän opinnäytetyön tilaajana toimi Nokia Solutions and Networks OY. Oulun ammattikorkeakoulun ohjaavana opettajana toimi tietotekniikan lehtori Kari Jyrkkä. Yrityksen puolesta ohjaajana toimi Technical Leader Sami Aho. Kiitokset ansaitsevat Eija Viirret, Jarkko Koskela ja Katri Leppälä vastauksista kiperiin tukiaseman toimintaan liittyviin kysymyksiin. Kiitokset myös Henri Vehkaojalle avusta matemaattisten käsitteiden ja neuroverkon toiminnan selventämisessä. Erityiskiitos koko PET6-tiimille uskomattomasta hurmoksesta.

Oulussa 8.9.2019

Ville-Veikko Komulainen

SISÄLLYS

SANASTO

ARIMA	Autoregressive Integrated Moving Average. Aikasarjan mallintamisessa ja ennustamisessa käytetty menetelmä
CPU	Central Processing Unit eli suoritin on tietoteknisen laitteen konekielisiä käskyjä suorittava osa.
CSV	Comma Separated Values, taulukkomaisen tiedon tallennusmuoto, jossa taulukon kukin sarake on erotettu toisistaan pilkulla tai muulla välimerkillä. Rivit on erotettu toisistaan rivinvaihdolla.
KPI	Key Performance Indicator. Mitattava tai laskettava arvo, jonka perusteella voidaan arvioida järjestelmän suorituskykyä.
LSTM	Long Short-Term Memory. Eräänlainen takaisinkytkeytyvä neuroverkko, joka soveltuu erityisesti aikasarjadatan käsittelyyn.
MAE	Mean Absolute Error. Absoluuttinen keskivirhe
MSE	Mean Squared Error. Keskineliövirhe
Pistoyksikkö	Rakenne, joka sisältää elektroniikkakomponentteja ja joka voidaan liittää kasettiin, kehikkoon tai muuhun mekaaniseen rakenteeseen. Tutkittava data on peräisin näiden pistoyksiköiden suorittimista.
Python	Monikäyttöinen tulkattava ohjelmointikieli
RMSE	Root Mean Squared Error. Keskineliövirheen neliöjuuri
RNN	Recurrent Neural Network. Takaisinkytkeytyvä neuroverkko

1 JOHDANTO

Tukiasemaohjelmiston testauksessa syntyy valtava määrä dataa, jonka perusteella ohjelmistotestaajan on arvioitava ohjelmiston laatua ja toimivuutta. Dataprofiloinnilla pyritään tutkimaan dataa ja tekemään siitä käyttökelpoisempaa. Sillä voidaan tunnistaa ohjelmistotestauksessa tärkeiden metriikoiden tyypillistä käyttäytymistä. Dataprofiili on raakadatan pohjalta luotu esitys, joka kuvastaa jotain datan analysoinnin kannalta merkittävää ominaisuutta. Tukiasemaohjelmiston testauksessa tällainen hyödyllinen profiili voisi esimerkiksi olla pylväskaavio, joka kuvaa eri prosessien suhteellista suoritinkäyttöä. Jo pelkästään hyvin profiloitu data voi tarjota arvokasta tietoa päätöksenteon tueksi, mutta tyypillisesti dataa täytyy vielä analysoida. Merkittävä osa datan analysointia on poikkeavuuksien havaitseminen.

Tämän opinnäytetyön tavoitteena on profiloida tukiasemaohjelmiston testauksen aikana tuotettua suoritinkäyttödataa ja perehtyä poikkeavuuksien havaitsemisessa käytettäviin algoritmeihin ja menetelmiin. Suoritinkäyttö on vain yksi metriikka, jota testauksen aikana tarkkaillaan. Pyrkimyksenä on perehtyä dataprofilointiin ja poikkeavuuksien havaitsemisessa käytettyihin menetelmiin, tutkia menetelmien käyttökelpoisuutta ja arvioida niiden käytettävyyttä myös muiden metriikoiden analysoinnissa.

Työ aloitetaan tutustumalla saatavilla olevaan dataan ja sen erityispiirteisiin. Datasta valitaan jokin mielekäs piirre tutkittavaksi. Piirteelle luodaan sille tyypillinen profiili, johon vertaamalla pyritään löytämään poikkeavuuksia. Tämän jälkeen suoritetaan poikkeavuuksien havaitsemisessa käytettävien algoritmien ja menetelmien vertailua ja valitaan näistä jokin tarkemmin tutkittavaksi. Tutkinnan tulokset analysoidaan ja dokumentoidaan.

Opinnäytetyö on osa Oulun ammattikorkeakoulun tieto- ja viestintätekniikan tutkinto-ohjelman opintoja ja sen tilaajana toimii Nokia Solutions and Networks OY. Työ on osana jatkuvasti Nokiassa tehtävää tukiasemaohjelmistotestauksen aikana syntyvää lokien analysoinnin automaatio- ja kehittämistyötä. Työ tehtiin kesän 2019 aikana.

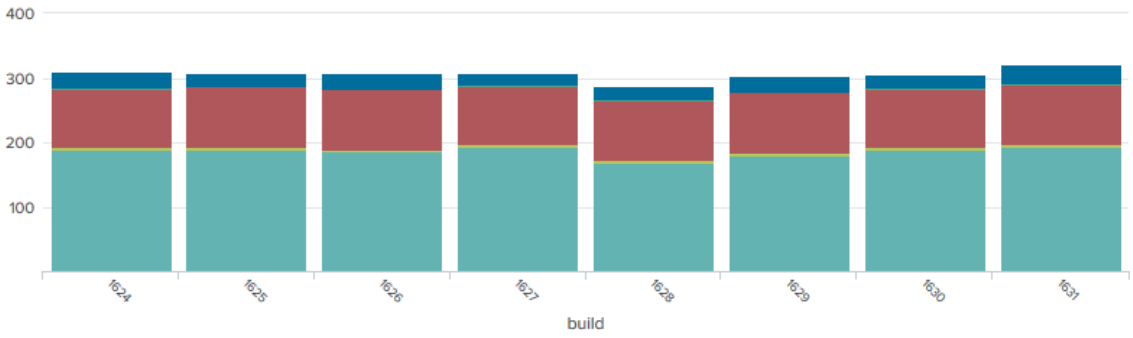
2 KÄSITTEET, MENETELMÄT JA TYÖKALUT

Tässä luvussa tutustutaan dataprofiloinnin ja poikkeavuuksien havaitsemisen käsitteisiin sekä käytettäviin menetelmiin ja työkaluihin.

2.1 Dataprofilointi

Dataprofilointi on prosessi, jonka avulla pyritään luonnehtimaan raakadatan sisäisiä ominaisuuksia (1, s. 131). Dataprofilointi sisältää esimerkiksi dataa kuvaavien statistiikkojen, kuten minimin, maksimin ja summan, laskemista, siinä esiintyvien säännönmukaisuuksien tutkimista, sen ryhmittelyä ja kuvailua sekä metadatan koostamista sen perusteella (2).

Dataprofiili on raakadatan pohjalta luotu esitys, joka kuvaa jotain datan analysoinnin kannalta tärkeää ominaisuutta. Kuvassa 1 on profiloituna tukiasemaohjelmiston testauksen aikana kerätyn datan perusteella koostettu esitys eri ohjelmistopakettien välisestä suorittimen eli CPU:n käytöstä. Kuvassa jokainen värillinen palkki vastaa tietyn prosessin suoritinkäytön keskiarvoa prosentteina. Dataprofiilin luomiseksi on siis summattu tukiaseman muodostavilta pistoyksiköiltä jokaisen CPU:n tietyn prosessin prosentuaalinen suoritinkäyttö tietyn ajan sisällä. Tämän summatun luvun ja mittapisteiden määrän perusteella on laskettu keskiarvo. Kuvasta pystytään jo silmämääräisesti profiileja vertailemalla päättelemään ainakin ohjelmistopaketin 1628 olevan poikkeava.



KUVA 1. Tukiasemaohjelmiston ohjelmistopakettien välinen suorittimen käyttö

Data, jonka perusteella tämä profiili on luotu, sisältää tyypillisesti lähes miljoona datapistettä noin 100 minuutin ajalta. Miljoonan datapisteen sisältämä tieto on

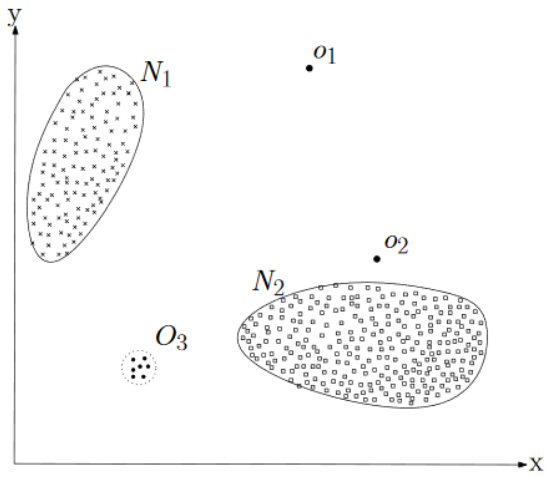
siis tiivistetty murto-osaan alkuperäisestä, mutta nyt sen perusteella pystytään jo arvioimaan tukiasemaohjelmiston laatua ja tekemään päätöksiä sen toimituskelpoisuudesta. Luonnollisesti tukiasemaohjelmiston laatua ja suorituskykyä arvioidessa ei tarkastella pelkästään suoritinkäyttöä. Yhdessä metriikat kuten muistin käyttö ja erilaiset KPI-arvot muodostavat kokonaisuuden, joka saattaa sisältää teratavuja raakadataa. KPI-arvot (engl. Key Performance Indicator) ovat joko mitattavia tai mittausten perusteella laskettavia arvoja, joilla voidaan arvioida tukiasemaohjelmiston suorituskykyä.

Tilan säästämisen lisäksi dataprofiloinnilla on monia muitakin hyötyjä. Se tekee saatavilla olevasta datasta korkealaatuisempaa ja sen avulla pystytään havainnollistamaan erilaisten tietoaineistojen ja -lähteiden välisiä riippuvuuksia. Tämä johtaa tarkempaan ennakoivaan analytiikkaan ja päätöksentekoon. Dataprofiloinnin avulla yrityksessä tuotettu tieto on organisoidumpaa ja sen avulla voidaan saada näkemyksiä riskeistä, mahdollisuuksista ja trendeistä. Myöskin datan tuottamisessa havaitut virheet, kuten korruptoituminen tai käyttäjästä johtuvat syöttövirheet, tulevat hyvin esille dataa profiloitaessa. (3.) Dataprofilointi voi siis auttaa olemassa olevien datankeräysprosessien kehittämisessä.

2.2 Poikkeavuuksien havaitseminen

Poikkeavuuksien havaitseminen (engl. anomaly detection, outlier detection) tarkoittaa merkittävällä tavalla poikkeavien datapisteiden, tapahtumien tai havaintojen etsimistä datasta (4). Kuten dataprofiloinnilla, myös poikkeavuuksien havaitsemisella saadaan raakadatasta esille merkittävää ja usein kriittistä tietoa, jota voidaan hyödyntää päätöksenteossa. Poikkeavuuksien havaitsemisen voidaan ajatella olevan dataprofiloinnista seuraava askel. Kun datasta on luotu halutunlainen esitys dataprofiloinnin avulla, käytetään poikkeavuuksien havaitsemismenetelmiä osoittamaan poikkeamat luodusta profiilista. Ohjelmistotestauksessa poikkeavuuksien havaitsemista voidaan käyttää ohjelmiston laadun tutkimisessa tai vikojen etsimisessä. Tarkalla, automatisoidulla ja luotettavalla poikkeavuuksien havaitsemisella voidaan toimittaa laadukkaampia ohjelmistoja lyhyemmässä ajassa.

Jotta havainto voidaan todeta poikkeavaksi, täytyy tarkasteltavalle asialla olla hyvin määritellyt normaalin käyttäytymisen rajat. Mikäli havainto ei vastaa normaalia se voidaan luokitella poikkeavuudeksi. Kuvassa 2 on ryhmitelty datapisteitä kahteen normaaliksi tulkittavaan klusteriin N_1 ja N_2. Niiden ulkopuolelle jäävät pisteet O_1 ja O_2 sekä pienempi klusteri O_3 ovat tässä tapauksessa poikkeavuuksia. (5, s. 2.)



KUVA 2. Yksinkertainen esimerkki poikkeavuuksista kaksiulotteisessa data-aineistossa (5, s. 2.)

2.2.1 Haasteita

Näennäisesti poikkeavuuksien havaitseminen vaikuttaisi siis yksinkertaiselta: määritellään normaalin rajat ja kaikki näiden rajojen ulkopuolella olevat havainnot ovat poikkeavuuksia. Todellisuudessa moni asia tekee tästä prosessista haasteellista. Normaalin ja epänormaalin välinen raja voi olla häilyvä ja se voi muuttua ajan kuluessa. Poikkeavuuden määritelmä on myös hyvin alakohtaista. Esimerkiksi pienet vaihtelut ihmisen ruumiinlämmössä voidaan tulkita poikkeaviksi kun taas vastaavan kokoiset heilahtelut osakemarkkinoilla ovat yleisiä. (5, s. 3.)

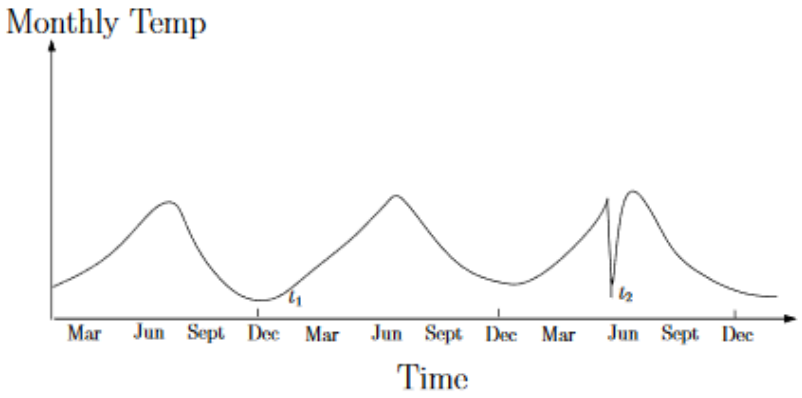
Käsiteltävä data voi olla kohinaista tavalla, joka muistuttaa varsinaisia poikkeamia. Tällaisen kohinan tunnistaminen ja poistaminen on vaikeaa. Saatavilla olevan datan määrä tai sen laatu voivat myös olla esteinä. Epäkelvolla datalla on vaikea kouluttaa ja todentaa käytettäviä malleja. Tyypillisesti poikkeavuuksien

havaitsemissa käytettävät tekniikat ja menetelmät ovat hyvin alakohtaisia. Kaikkein perustavanlaatuisessa muodossaan poikkeavuuksien havaitseminen on siis hyvin haastava ongelma ratkaistavaksi. (5, s. 3.)

2.2.2 Poikkeavuustyyppejä

Poikkeavuudet voidaan luokitella kolmeen erilaiseen tyyppiin. Yksinkertaisimmillaan ne voivat olla pistepoikkeavuuksia (engl. point anomaly). Ne ovat yksittäisiä datapisteitä, joiden arvo poikkeaa merkittävällä tavalla oletetusta. Esimerkiksi luottokorttihuijauksia tunnistettaessa, henkilön kulutustottumuksista poikkeava suuri kertaostos voidaan tulkita poikkeavuudeksi. (5, s. 7–10.)

Asiayhteydestä riippuvat poikkeavuudet (engl. contextual anomalies) puolestaan ovat poikkeavuuksia vain tietyssä kontekstissa. Jos esimerkiksi seurataan vuosittaista lämpötilan vaihtelua, ovat pakkasen puolelle menevät lukemat normaaleita talvella, mutta vastaavat lukemat keskikesällä ovat poikkeavuuksia. Kuvassa 3 havainnollistetaan tätä esimerkkiä. Lämpötilat t_1 ja t_2 saavat täsmälleen samat arvot, mutta ainoastaan t_2 on poikkeava. (5, s. 7–10.)



KUVA 3. Esimerkki asiayhteydestä riippuvasta poikkeavuudesta (5, s. 8.)

Yhteispoikkeavuudet (engl. collective anomalies) muodostuvat yksittäisistä datapisteistä, jotka eivät itsessään välttämättä ole poikkeavia, mutta yhdessä tai tietyssä järjestyksessä esiintyessään ne muodostavat poikkeavuuden. Jos esimerkiksi seurataan verkkoon liitetyn tietokoneen tietoliikennettä protokolla-analysaat-

torilla, voidaan muodostettujen yhteyksien järjestyksen perusteella tunnistaa tietomurtoja. Tilannetta, jossa tietokoneeseen muodostetaan SSH-yhteys, voidaan pitää itsessään tavanomaisena asiana, mutta jos yhteyden muodostusta seuraa puskuriylivuoto ja tiedostojen siirtäminen FTP-protokollaa käyttäen, on syytä epäillä tietomurtoa. (5, s. 7–10.)

Erilaiset poikkeavuustyypit eivät ole toisiaan poissulkevia. Yhteis- tai pistepoikkeavuus voi samalla olla myös asiayhteydestä riippuva poikkeavuus, mikäli ne analysoidaan liitettyinä osaksi asiayhteyttään. (5, s. 7–10.)

2.2.3 Ohjaamaton ja ohjattu poikkeavuuksien havaitseminen

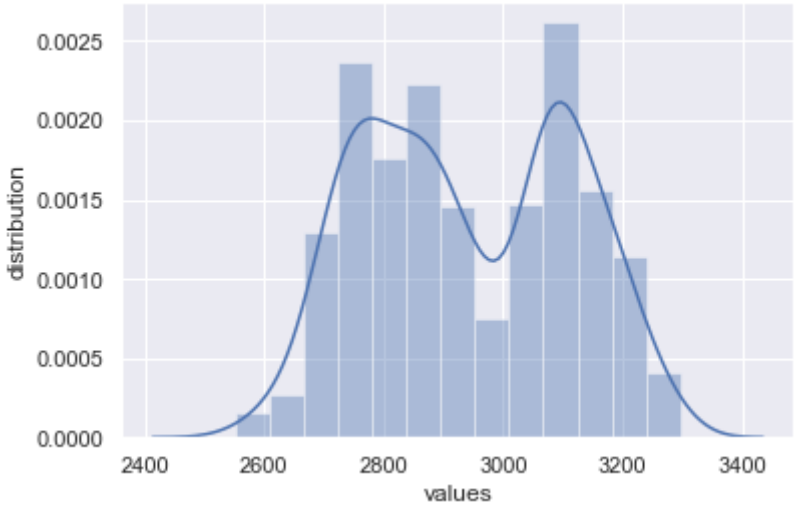
Ohjaamattoman ja ohjatun poikkeavuuksien havaitsemisen erot pelkistyvät siihen, onko tarkasteltavaan dataan merkitty poikkeamat valmiiksi vai ei. Ohjaamattomassa (engl. unsupervised) poikkeavuuksien havaitsemisessa tarkasteltavasta datasta ei tiedetä ennakkoon, onko se poikkeavaa vai ei. Vastaavasti ohjatussa (engl. supervised) poikkeavuuksien havaitsemisessa tarkasteltava data on valmiiksi merkitty poikkeavaksi tai normaaliksi. (5, s. 10–11.) Ohjatuissa menetelmissä pyritään siis luomaan valmiin datan pohjalta jonkinlainen malli, jota voidaan käyttää tuntemattomaan dataan.

2.3 Poikkeavuuksien havaitsemisessa käytettyjä menetelmiä

Riippumatta siitä, minkälaista menetelmää poikkeavuuksien havaitsemiseen käytetään, täytyy menetelmällä olla jonkinlainen ulostulo, jonka perusteella voidaan päätellä, onko poikkeavuuksia havaittu. Testidatan jokaiselle tarkasteltavalle tietoalkiolle voidaan määrittää jonkinlainen poikkeavuuspistemäärä, joka kertoo, kuinka paljon alkio poikkeaa normaalista esimerkiksi prosentuaalisesti. Alkio voidaan myös yksiselitteisesti erottaa joko poikkeavaksi tai normaaliksi. (5, s. 11.)

Seuraavaksi esitellään poikkeavuuksien tunnistamisessa käytettäviä menetelmiä yleisellä tasolla. Menetelmien valinta on riippuvainen tutkittavan datan muodosta eli kaikki näistä menetelmistä eivät sovellu tässä opinnäytetyössä tarkasteltavan suoritinkäyttödatan poikkeavuuksien havaitsemiseen. Esimerkiksi tilastollisia menetelmiä valitessa osa menetelmistä on käyttökelpoisia vain, jos tarkasteltava tie-

toaineisto noudattaa normaalijakaumaa. Kuvassa 4 on esitettynä työssä tarkasteltavan tietoaineiston yhden ominaisuuden saamien arvojen jakauma. Jakauma ei noudata Gaussin kellokäyrälle tyypillistä muotoa, joten esimerkiksi Grubbsin poikkeavuustesti karsittiin jo työn alkuvaiheessa pois testattavista menetelmistä. Työssä käytettyjä tekniikoita esitellään ja vertaillaan tarkemmin luvussa 4.



KUVA 4. Tarkasteltavien arvojen jakauma

2.3.1 Luokitteluun perustuvat menetelmät

Luokittelussa luodaan valmiiksi poikkeavaksi tai normaaliksi erotellusta datasta yleinen malli, jota kutsutaan luokittelijaksi. Olettamus näissä menetelmissä on, että tarpeeksi suurella koulutusdatan määrällä pystytään oppimaan sen erityispiirteet, jonka jälkeen syötteinä saadut alkiot pystytään jaottelemaan joko normaaleiksi tai poikkeaviksi. Luokitteluun perustuvia menetelmiä ovat esimerkiksi osa neuroverkoista, Bayes-verkot, tukivektorikoneet ja sääntöpohjaiset menetelmät kuten päätöspuut. (5, s. 20–25.)

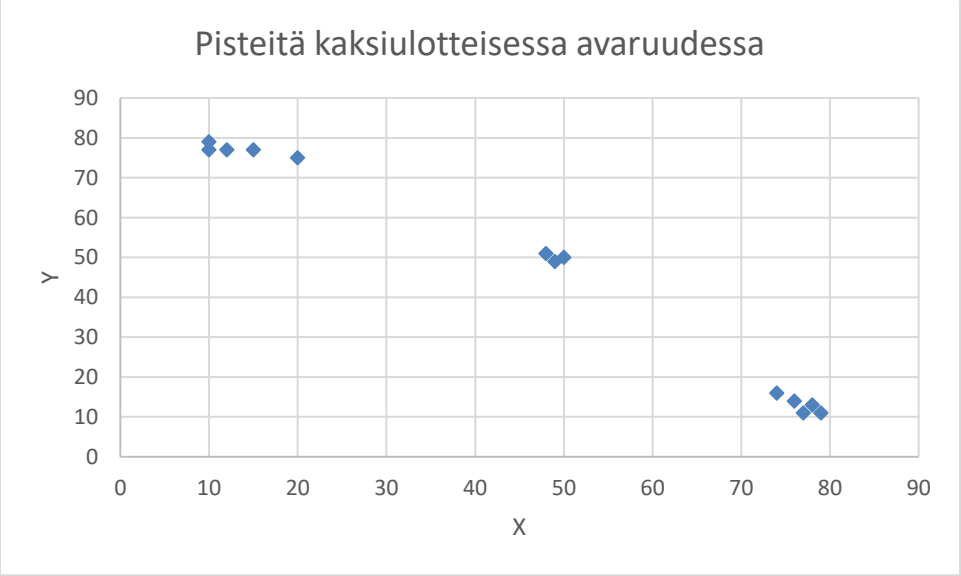
Erityisesti useampaan luokkaan alkioita jaottelevissa menetelmissä voidaan hyödyntää useita julkisesti saatavilla olevia tehokkaita algoritmeja. Menetelmien testaus on nopeaa, sillä jokaista testattavaa alkiota voidaan verrata ennalta määriteltyyn malliin. Luokittelupohjaisten menetelmien hyödyntämisen edellytyksenä on, että koulutusdata on valmiiksi luokiteltua. Osassa menetelmistä ulostulona on

poikkeavuuden todennäköisyyteen perustuva lukuarvo, mutta valtaosassa voidaan vain jaotella testattavat alkiot joko poikkeaviin tai normaaleihin. (5, s. 20–25.)

2.3.2 Lähinaapurimenetelmät

Lähinaapurimenetelmissä perusolettamuksena on, että normaalit datapisteet muistuttavat tai ovat lähellä toisiaan. Näissä menetelmissä lasketaan yleensä kahden pisteen välistä etäisyyttä n-ulotteisessa avaruudessa tai niiden samankaltaisuutta. (5, s. 25.)

Lähinaapurimenetelmät voidaan jakaa karkeasti kahteen kategoriaan naapureiden etäisyyden tai naapuruston tiheyden perusteella. Tarkasteltaville alkioille voidaan antaa poikkeavuuspistemäärä niiden k:nnen lähimmän naapurin etäisyyden perusteella. Symboli k merkitsee, monenneksiko lähimmän naapurin etäisyyttä tarkastellaan. (5, s. 25.) Kuvassa 5 havainnollistetaan asiaa luomalla satunnaisesti määriteltyjen rajojen sisällä kolme erillistä ryhmää pisteitä. Oletetaan, että keskellä olevat pisteet ovat poikkeamia. Mikäli $k = 1$, ei näitä pisteitä tunnistettaisi poikkeaviksi, sillä niiden etäisyys lähimpään naapuriin on suurin piirtein sama kuin normaaleidenkin pisteiden. Jos puolestaan $k = 3$, poikkeamat havaitaan, koska jokaisen poikkeavan pisteen etäisyys kolmanneksi lähimpään naapuriin on selvästi suurempi kuin normaaleilla pisteillä.



KUVA 5. Lähinaapurimenetelmän havainnollistaminen

Alkioille voidaan laskea myös niiden ympäröimän avaruuden tiheys. Tällöin etäisyyttä k:nneksi lähimpään naapuriin käytetään tarkasteltavan alkion ympärille muodostuvan hyperpallon säteenä. Normaaleiksi tulkittavilla pisteillä tämän pallon säde on pieni, joten sen suhteellinen tiheys on suuri. Vastaavasti poikkeavilla pisteillä säde kasvaa suureksi ja pisteiden tiheys pienenee. (5, s. 27.)

Lähinaapurimenetelmien etuina on niiden suoraviivainen muokattavuus erityyppiselle datalle. Tyypillisesti vain k:n arvoa täytyy muuttaa, vaikkakin sen määrittäminen voi olla vaikeaa. Ne ovat myös täysin ohjaamattomia luonteeltaan eli niille syöttävän datan ei tarvitse olla valmiiksi luokiteltua eivätkä ne välitä datan sisäisestä hajonnasta. Laskennallisesti ne ovat kuitenkin raskaampia toteuttaa, sillä niiden täytyy laskea jokaisen alkion etäisyys jokaiseen muuhun alkioon.

2.3.3 Klusterointimenetelmät

Klusteroinnilla tarkoitetaan aineellisten tai abstraktien alkioiden ryhmittelyä niin, että keskenään samankaltaiset alkiot kuuluvat samaan ryhmään (6). Poikkeavuuksien havaitsemisessa klusteroidusta datasta voidaan käyttää kolmea erilaista tekniikkaa. Voidaan tarkastella kuuluvatko ne johonkin klusteriin, kuinka kaukana ne ovat klusterinsa keskustasta tai kuinka suuria niiden muodostamat klusterit ovat. Yksinkertaisimmillaan voidaan siis olettaa normaaleiden datapisteiden kuuluvan johonkin klusteriin, kun taas poikkeamat eivät kuulu mihinkään. Nämä tekniikat on kuitenkin yleensä optimoitu klustereiden löytämiseksi, joten poikkeamien löytäminen niillä on epävarmaa. (5, s. 30.)

Datapisteiden etäisyyttä niitä lähinnä olevan klusterin keskustaan voidaan myös käyttää poikkeavuuksien havaitsemiseen. Normaalien datapisteiden etäisyys lähimpien klustereidensa keskustasta on pieni, kun taas poikkeamat sijaitsevat kaukana. Jos poikkeavuudet muodostavat oman klusterinsa, näitä tekniikoita ei voida käyttää niiden havaitsemiseen. Tämän ongelman ratkaisussa voidaan käyttää olettamusta, jonka mukaan normaalit datapisteet muodostavat suuria tai tiheitä klustereita, kun taas poikkeamat kuuluvat joko pieniin tai harvoihin klustereihin. (5, s. 30–31.)

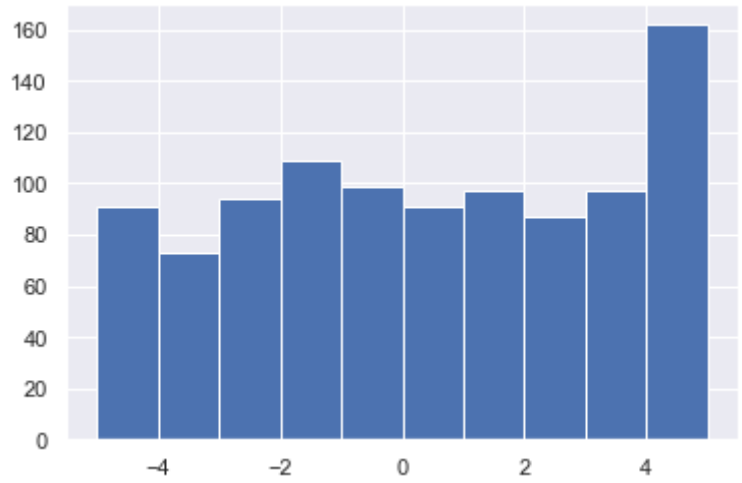
Kuten lähinaapurimenetelmät, myös klusterointimenetelmät toimivat ilman ohjausta ja niitä voidaan muokata käytettäväksi erityyppisellä datalla helposti. Poikkeavuuksien havaitseminen on kuitenkin monen algoritmin sivutuote eli niitä ei ole varsinaisesti optimoitu tähän tehtävään. (5, s. 32-33.)

2.3.4 Tilastolliset menetelmät

Tilastolliset menetelmät sovittavat niille syötetyn datan johonkin tilastolliseen malliin, jonka jälkeen tietoalkioita testataan tätä mallia vasten käyttäen tilastollista päättelyä. Mikäli tilastollisen mallin on epätodennäköistä tuottaa testatun kaltainen alkio, se voidaan todeta poikkeavaksi. Menetelmät voidaan jakaa parametrillisiin tai parametrittomiin. (5, s. 33.)

Parametrilliset menetelmät olettavat normaalin datan noudattavan jonkinlaista hajontaa. Parametrilliset menetelmät voidaan jakaa edelleen kahteen eri tyyppiin sen mukaan minkälaista hajontaa käsiteltävä data noudattaa. Normaalijakaumamenetelmissä datan hajonnan oletetaan noudattavan Gaussin kellokäyrää. Yleisesti näissä menetelmissä tarkasteltavan alkion etäisyyttä arvioidusta keskiarvosta voidaan pitää sen poikkeavuuspistemääränä. Mikäli tarkasteltava data noudattaa jonkinlaista muuta jakaumaa, voidaan käyttää regressiomalleja. Näissä menetelmissä regressiomalli sovitetaan käsiteltävän datan perusteella, jonka jälkeen alkioita testataan sitä vasten. Testattavan alkion jäännösosa on se osa datasta, jota malli ei pysty selittämään. Tätä jäännösosaa voidaan käyttää tarkasteltavan alkion poikkeavuuspistemääränä. (5, s. 33–37.)

Parametrittomissa menetelmissä käytettävien mallien rakennetta ei määritellä etukäteen, vaan se muodostetaan syötteenä olleen datan perusteella. Parametrittomia menetelmiä ovat esimerkiksi histogrammi- ja kernelifunktiopohjaiset menetelmät. (5, s. 37–39.) Kuvassa 6 on esitettynä tuhannesta satunnaisluvusta välillä −5…5 luotu histogrammi. Histogrammissa on esitettynä x-akselilla aineistosta löytävät arvot ja y-akselilla näiden määrä. Histogrammi siis muodostaa datan pohjalta profiilin, johon tuntemattomia tietoalkioita voidaan verrata. Tässä yksinkertaisessa esimerkissä arvot, jotka eivät kuulu mihinkään esitetyistä pylväistä, ovat poikkeamia.



KUVA 6. Satunnaisluvuista välillä −5...5 muodostettu histogrammi

Mikäli aineiston jakauma arvioidaan hyvin, tilastolliset menetelmät voivat toimia ohjaamattomasti. Niiden tuottamiin poikkeavuuspistemääriin liittyy myös luottamusväli, jota voidaan käyttää lisätietona testattavan alkion poikkeavuudesta. Niiden avulla saadaan myös tilastollisesti perusteltuja ratkaisuja, mikäli oletukset aineiston jakaumasta pitävät paikkansa. Niiden käyttökelpoisuus kuitenkin pienenee sitä mukaa, kun tarkasteltavan datan ulottuvuudet kasvavat. Moniulotteisen tietoalkion yksittäiset ominaisuudet voivat noudattaa hyvinkin tarkasti tiettyä jakaumaa, mutta niiden yhdistelmät käyttäytyvät epäsäännönmukaisesti. Tästä johtuen sopivan mallin löytäminen voi olla liian työlästä. (5, s. 39.)

2.3.5 Informaatioteoriaan perustuvat menetelmät

Informaatioteoriaan perustuvat menetelmät analysoivat käsiteltävän datan tietosisältöä. Niiden mukaan poikkeavuudet lisäävät data-aineiston epäsäännönmukaisuuksia. Myös nämä menetelmät toimivat ohjaamattomasti ilman olettamuksia datan tilastollisesta jakaumasta. Niillä on kuitenkin vaikea pisteyttää havaittuja poikkeavuuksia. Niiden suorituskyky on myös suuresti riippuvainen valitusta informaatioteoreettisesta mittauksesta. (5, s. 39–41.)

2.3.6 Spektriin liittyvät menetelmät

Spektriin liittyvillä menetelmillä pyritään likiarvoistamaan käsiteltävää dataa niin, että käytetään datasta vain sellaisia ominaisuuksia, jotka selittävät suurimman

osan sen vaihtelusta. Data pyritään siis esittämään käyttäen vähemmän ulottuvuuksia, jolloin normaalit ja poikkeavat tietoalkiot näyttävät merkittävästi erilaisilta. Monet näistä menetelmistä hyödyntävät jollain tavalla pääkomponenttianalyysiä (PCA, Principal Component Analysis) datan ulottuvuuksien vähentämiseen. Spektriin liittyvät menetelmät suorittavat automaattisesti ulottuvuuksien vähentämistä eli ne sopivat tilastollisia menetelmiä paremmin suuriulotteisen datan käsittelyyn. Ne myös toimivat ohjaamattomasti. Nämä menetelmät ovat kuitenkin laskennallisesti monimutkaisia ja ne ovat käyttökelpoisia vain, jos poikkeamat pystytään erottelemaan normaalista pienempiulotteisessa avaruudessa. (5, s. 41–42.)

2.4 Käsiteltävä data

Työssä tarkasteltava data on tukiasemaohjelmiston testauksen aikana kerättyä suoritinkäyttödataa. Data kuvaa prosentteina, kuinka suuren osan ajasta tunnetun mittaisella tarkastelujaksolla tietty prosessityyppi on käyttänyt tietyn pistoyksikön suoritinta. Pistoyksiköllä tarkoitetaan tukiasemaan liitettävää elektroniikkakomponentteja sisältävää korttimaista piirilevyä. Pistoyksiköillä voi olla tukiaseman toiminnan kannalta erilaisia tehtäviä. Esimerkiksi tukiasemassa, josta työssä käsitelty data on peräisin, yksi pistoyksikkö toimii muiden pistoyksiköiden pääkontrolliyksikkönä ja loput toimivat tukiaseman kapasiteettiyksiköinä. Tällaisia kapasiteettiyksiköitä tarkasteltavassa tukiasemassa on kahdeksan kappaletta. Kapasiteettiyksiköiden määrä vaihtelee tukiasemakohtaisesti. Mitä enemmän tukiasemassa on kapasiteettiyksiköitä, sen enemmän se kykenee muodostamaan soluja ja käsittelemään mobiilitietoliikennettä. Tukiasemaohjelmisto määrittelee solujen maksimimäärän eli kapasiteettiyksiköitä ei voida lisätä tukiasemaan loputtomasti.

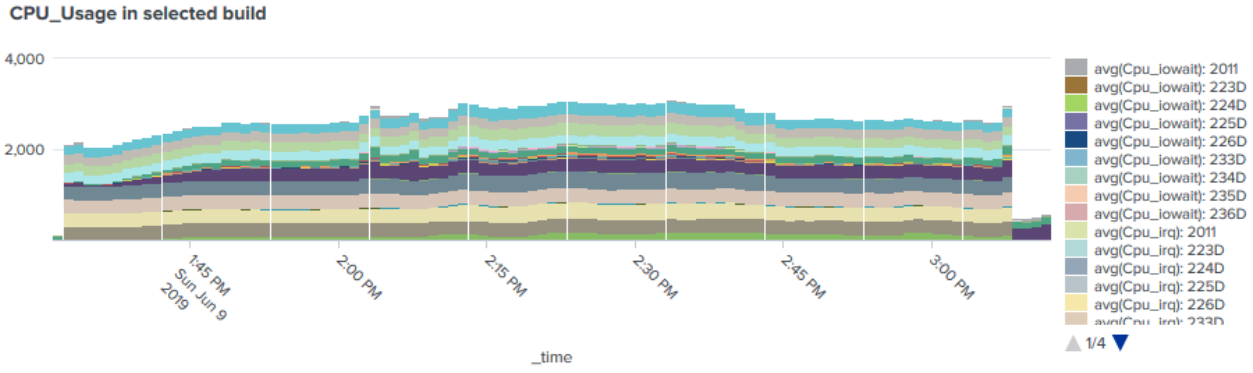
Yksittäisen pistoyksikön käyttöjärjestelmänä toimii erityisesti tukiasemakäyttöön tarkoitettu Linux-käyttöjärjestelmä. Prosessityyppimittaukset suoritetaan Linuxin top-komennolla, jonka avulla voidaan tarkastella Linux-pohjaisen järjestelmän suoritin- ja muistikäyttöä. Kuvassa 7 on esitelty CentOS-pohjaisen palvelimen suoritinkäyttötyypit tietyllä ajanhetkellä. Pistoyksikölle suoritettu top-komento palauttaa tismalleen samat mittaukset. Käsiteltävässä datassa näistä id-arvo on jä-

tetty pois. Kyseessä on mittaus, joka kertoo, kuinka kauan prosentuaalisesti suoritin on ollut tekemättä mitään tarkasteltavalla ajanjaksolla. Yksittäisen prosessityypin pistoyksikkökohtaisen suoritinkäyttö voi tarkasteltavalla ajanjaksolla nousta yli 100 prosentin. Tämä merkitsee suorittimen käyttäneen useampaa kuin yhtä ydintä prosessointiin.

```
%Cpu(s):  4,6 us,  3,1 sy,  0,0 ni, 92,0 id,  0,0 wa,  0,0 hi,  0,3 si,  0,0 st
```

KUVA 7. CentOS-palvelimen top-komennon palauttama prosentuaalinen suoritinkäyttö prosessityypeittäin

Testauksen aikana prosessityyppimittauksia suoritetaan viiden sekunnin välein, mutta tämä tukiaseman tuottama raakadata on tiivistetty laskemalla saatujen arvojen keskiarvoja minuutin ajalta. Kuvassa 8 on esitettynä tällainen raakadatan pohjalta luotu dataprofiili, jossa on eroteltuna eri värein yksittäisen pistoyksikön yksittäisen prosessin suoritinkäyttö ajan suhteen. Kuvan selitteestä oikeassa laidassa nähdään noin neljäsosa eri pistoyksikkö- ja prosessiyhdistelmistä. Numeroarvo on pistoyksikön yksilöivä tunnus. Tässä tapauksessa 2011 toimii kontrolliyksikkönä ja loput ovat kapasiteettiyksiköitä.



KUVA 8. Yhden testin aikainen suoritinkäyttöprofiili

Testaus, jonka ajalta käsittelemämme data on peräisin, on niin sanottua penkkitestausta (engl. bench testing, benchmarking). Sen tavoitteena on testata tukiasemaohjelmistoa laboratoriossa normaaleja käyttöoloja jäljitellen. Testaus suoritetaan generoimalla kuormalaitteella tukiasemalle tietoliikennettä, joka jäljittelee todellista mobiilitietoliikennettä liikennemallien avulla. Tukiasema siis käsittelee

testauksen aikana puheluita, mobiilidataa, hakuja, solunvaihtoja yms. kuten todellisessa maailmassa.

Käsittelemämme tietoaineisto on aikasarjamuotoista. Tämä tarkoittaa tarkasteltavien havaintojen olevan aikajärjestyksessä ja niiden olevan tyypillisesti tasaisin aikavälein (7). Aikasarjassa aikakomponentti lisää yhden ulottuvuuden tarkasteltavaan tietoaineistoon ja se täytyy ottaa huomioon poikkeavuuksien havaitsemismenetelmiä valitessa. Aikasarjaa käsiteltäessä puolestaan täytyy ottaa huomioon siitä mahdollisesti löytyvät ominaisuudet kuten trendi, stationäärisyys, kausittaisuus ja muuttujien autokorrelaatio ajan suhteen. (8.)

2.5 Käytettävät työkalut

Työssä käytettävä data esikäsiteltiin Splunk-ohjelmistoalustalla, josta se tuotiin CSV-tiedostoina käsiteltäväksi. Data-analyysissä käytettiin Python-ohjelmointikieltä ja sen Keras-rajapintaa sekä Matplotlib-, Pandas-, scikit-learn-, Statsmodels- ja STUMPY-kirjastoja. Ohjelmointiympäristönä toimi Jupyter Notebook.

2.5.1 Splunk-ohjelmistoalusta

Splunk on samannimisen yrityksen kehittämä ohjelmistoalusta, joka on tarkoitettu suurten data-aineistojen analysointiin ja indeksointiin. Se on tarkoitettu erityisesti erilaisten laitteiden ja järjestelmien tuottamien lokiviestien käsittelyyn. Splunk tunnistaa automaattisesti useita tiedostoformaatteja ja kykenee poimimaan käsiteltävästä datasta muuttuja-arvo-parit eli kentät automaattisesti. Käyttäjä pystyy suorittamaan hakuja käsiteltävälle datalle käyttäen Splunkin omaa SPL-hakukieltä ja visualisoimaan sitä tulosten perusteella. (9.)

2.5.2 Jupyter Notebook -ohjelmointiympäristö

Jupyter Notebook on Project Jupyterin tuottama avoimen lähdekoodin ohjelmointiympäristö, joka on tarkoitettu yhdistämään ajettavaa koodia, kuvia, taulukoita ja tekstiä interaktiiviseksi dokumenttimaiseksi kokonaisuudeksi. Varsinainen Jupyter Notebook -ohjelmisto käynnistää palvelimen, johon voidaan ottaa yhteys etänä tai paikallisesti verkkoselaimella, jolloin sillä luotuja dokumentteja pääsee muokkaamaan. Koodia se suorittaa muistikirjakerneleiksi kutsuttujen laskenta-

moottoreiden avulla. Ohjelmisto sisältää viralliset kernelit Julia-, R- ja Python-ohjelmointikielille, joiden lisäksi Jupyter-yhteisö on kehittänyt kernelit lähes kaikille kuviteltavissa oleville ohjelmointikielille. (10.)

2.5.3 Käytetyt Python-kirjastot

CSV-tiedostoista tuotu data tallennettiin Pandas-ohjelmointikirjaston DataFrame-tietorakenteeseen. Pandas on kirjasto, jonka avulla tietoa voidaan analysoida ja sitä voidaan tallentaa helppokäyttöisiin tietorakenteisiin. DataFrame on tähän kirjastoon kuuluva tietorakenne, joka muistuttaa Excel-taulukkoa tai SQL-tietokannan taulua. (11.)

Matplotlib-kirjastoa käytettiin kaavioiden luomiseen käsiteltävästä datasta. Matplotlib on 2D-grafiikkakirjasto, jota voidaan käyttää sovelluskehitykseen, interaktiiviseen skriptaukseen ja julkaisukelpoisten kuvien luomiseen (12).

Scikit-learn-kirjastoa käytettiin virhemetriikoiden laskemiseen, klusterointiin ja tarkasteltavan datan skaalaamiseen halutulle välille. Scikit-learn on monipuolinen koneoppimisessa ja data-analyysissä käytettävä kirjasto, joka toteuttaa luokittelussa, regressiossa, klusteroinnissa ja ulottuvuuksien vähentämisessä käytettäviä algoritmeja. Sen avulla dataa voidaan myös esikäsitellä ja sitä voidaan käyttää käytettävien mallien valitsemisen apuna. (13.)

Keras on ohjelmointirajapinta, jonka avulla voidaan rakentaa neuroverkkoja (14). Sen avulla luotiin ja testattiin LSTM-neuroverkkoa.

Statsmodels-kirjastolla voidaan luoda ja arvioida tilastollisia malleja, suorittaa tilastollisia testejä ja tutkia käsiteltävän datan tilastollisia ominaisuuksia (15). Sen avulla käsiteltävää dataa tasoitettiin eksponentiaalisesti ja siitä luotiin ARIMA-malli. Lisäksi kirjastoa käytettiin datan tilastollisten ominaisuuksien tutkimiseen ja testaamiseen.

STUMPY-kirjastoa käytettiin matriisiprofiilin luomiseksi käsiteltävästä datasta. Kirjasto toteuttaa matriisiprofiilin luomisessa vaadittavat laskennallisesti tehokkaat algoritmit Python-ohjelmointikielellä (16).

3 DATAN ESIKÄSITTELY JA OMINAISUUKSIEN TUTKIMINEN

Työ aloitettiin kokoamalla menetelmien testauksessa käytettävä tietoaineisto. Kerätyn tietoaineiston tilastollisia ominaisuuksia testattiin ja sitä siistittiin. Aineiston pohjalta luotiin keskiarvoistettu suoritinkäytön dataprofiili, johon yksittäisiä testejä verrattiin. Aineiston pohjalta luotiin dataprofiilit myös tavallista ja painotettua liukuvaa keskiarvoa sekä eksponentiaalista tasoitusta käyttäen. Näistä parhaiten toimivan menetelmän tulokset valittiin pohjatasoksi, johon edistyneempiä menetelmiä verrattiin. Yksittäisistä testeistä luotiin myös yhteen ketjutettu aikasarja, jota käytettiin ARIMA-mallin, K-means-klusteroinnin, LSTM-neuroverkon ja matriisiprofiilin toteuttamisessa ja testauksessa. Ketjutetulla aikasarjalla saatiin sitä hyödyntäville menetelmille suurempi määrä datapisteitä käsiteltäväksi. Esimerkiksi ARIMA-malli tai LSTM-neuroverkko eivät kykene muodostamaan mielekkäitä ennustuksia yksittäisen testin noin 75 datapisteen perusteella.

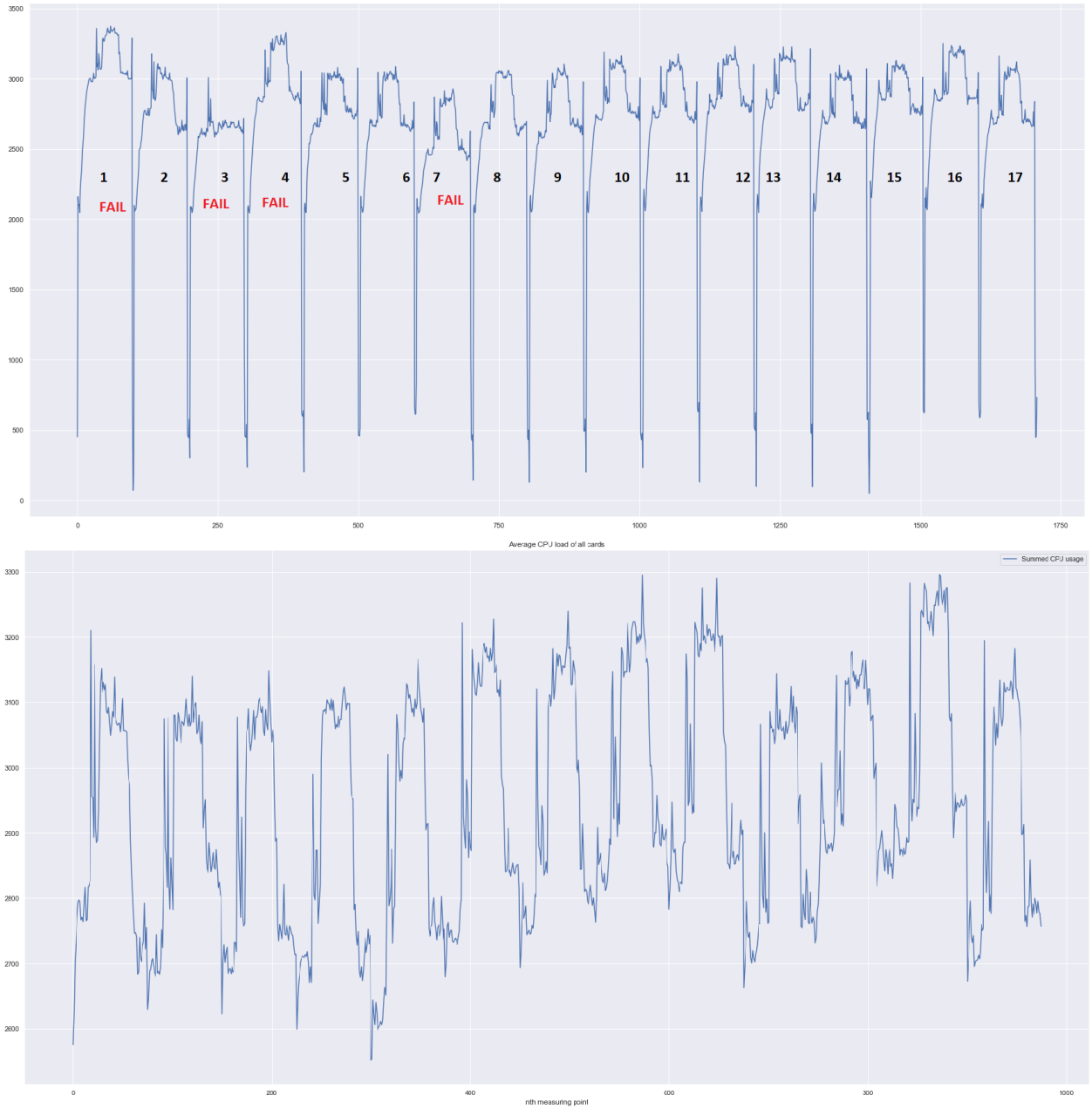
3.1 Data-aineiston valinta, kerääminen ja siistiminen

Data-aineiston valinnassa tarkasteltiin jo valmiiksi onnistuneiksi tai epäonnistuneiksi luokiteltuja testejä. Luokittelu oli suoritettu vertaamalla testiä yksittäisen verrokkitestin metriikkoihin. Onnistuneista testeistä valikoitiin yhdeksi tietoaineistoksi sellaisia, joiden kaikki suorituskykyä mittaavat metriikat oli arvioitu hyviksi. Epäonnistuneista testeistä valikoitiin sellaisia, jotka oli arvioitu epäonnistuneiksi vähintään suoritinkäytön osalta. Myös datan eheyttä käytettiin valintakriteerinä. Menetelmien tulosten todentamisessa käytettiin vielä erillistä hyväksytyistä testeistä koostettua tietoaineistoa. Aineistoihin valittiin vain sellaisia testejä, joiden ajalta saatavilla oleva data oli yhtenäisen mittaista ja jossa ei ollut tyhjiä arvoja keskellä.

Kuvassa 7 esiteltiin tyypillinen suoritinkäytön dataprofiili. Tämä dataprofiili on luotu Sami Ahon toteuttaman SPL-hakukielisen kyselyn pohjalta. Kysely laskee raakadatasta, jossa suoritinkäyttömittauksia on 5 sekunnin välein, kunkin pistoyksikön kunkin prosessityypin keskimääräisen suoritinkäytön prosentteina minuutin ajalta. Profiilin luomiseen käytetty tieto tuotiin Splunkista ulos CSV-muodossa.

Noudetut CSV-tiedostot luettiin erillisiin Pandasin DataFrame-tietorakenteeseen, ja jokaisesta testistä karsittiin alusta 15 ja lopusta 10 minuutin ajalta rivejä pois. Nämä ovat testin aloitus- ja lopetusajankohtia, jolloin suoritinkäytössä on paljon normaaliksi tulkittavaa vaihtelua. Tällä karsinnalla saatiin aikaiseksi tarkempia malleja.

Kullekin riville laskettiin vielä kaikkien tarkasteltavan rivin arvojen summa, jota käytettiin suoritinkäytön yleisen tason profiilina. Yksittäiset testit myös ketjutettiin yhteen, jolloin pystyttiin silmämääräisesti vertailemaan testien suoritinkäyttöä keskenään. Yksittäiset testit on piirretty kuvaan jatkuvaksi kokonaisuudeksi poistamalla merkityksetön välivaihe. Tällä yksinkertaisella dataprofiloinnilla kyettiin tunnistamaan aikaisemmin hyväksytyiksi luokitelluista testeistä neljä sellaista, joiden yleinen suoritinkäytön taso poikkesi huomattavasti odotetusta. Nämä testit siirrettiin osaksi menetelmien testauksessa käytettävien hylättyjen testien tietoaineistoa. Kuvassa ylhäällä yksittäiset testit on numeroitu ja selvästi poikkeavat testit on eritelty punaisella FAIL-tunnisteella. Alempana nämä poikkeavat testit on poistettu tietoaineistosta.

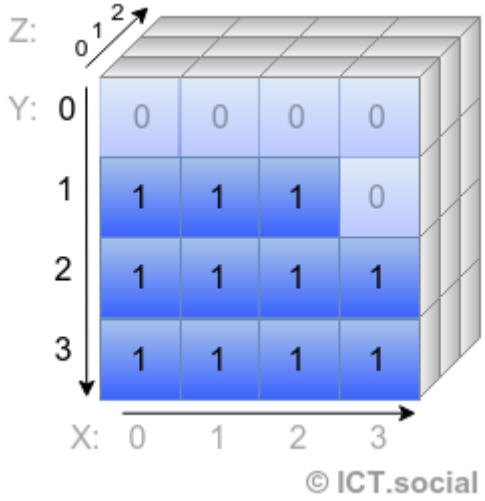


KUVA 9. Hyväksytyiksi luokiteltujen testien ketjutettu summakäyrä. Yllä testit ennen poikkeavien testien poistamista, alla jälkeen

3.2 Keskiarvotestin luominen

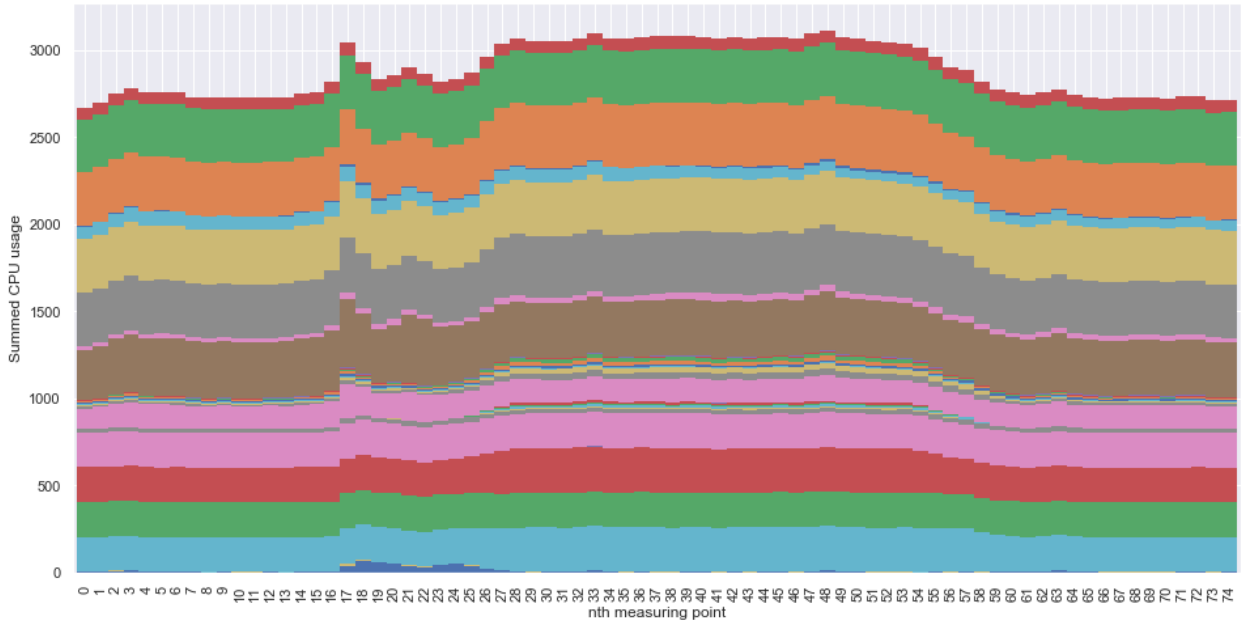
Jokainen yksittäisestä testistä luotu DataFrame on 2-ulotteinen taulukko, jossa sarakkeina ovat pistoyksikkö-prosessiyhdistelmät ja rivit ovat minuutin välein toistuvia aikapisteitä. Yhtä saraketta ylhäältä alas seuraamalla voidaan siis tarkastella tietyn pistoyksikön tietyn prosessin suoritinkäyttöä aikajärjestyksessä. Kun nämä DataFramet ladotaan päällekkäin kolmiulotteiseksi taulukoksi Pandasin

Panel-tietorakenteen avulla, voidaan yksittäistä kolmatta akselia seuraamalla tarkastella tietyn pistoyksikön tietyn prosessin suoritinkäyttöä tietyllä ajanhetkellä testien välillä. Kuvassa 10 on havainnollistettu 3-ulotteisen taulukon rakennetta.



KUVA 10. Kolmiulotteinen taulukko (17)

Z-akselia seuraamalla voidaan siis suorittaa akselille erilaisia laskutoimituksia ja koostaa näistä tuloksista uuden kaksiulotteisen taulukon. Tällä menetelmällä luotiin erilliset DataFramet, joihin on laskettu suoritinkäytön keskiarvo ja -hajonta tietyn pistoyksikön tietylle prosessille tietyllä ajanhetkellä. Kuvassa 11 on keskiarvotestin pohjalta luotu Matplotlib-kirjastoa käyttäen alkuperäistä dataa vastaava dataprofiili. Valtaosa menetelmistä testattiin käyttäen joko tämän keskiarvoprofiilin summakäyrää, eli profiilin ylätasoa, tai ketjutettua testien summakäyrää.



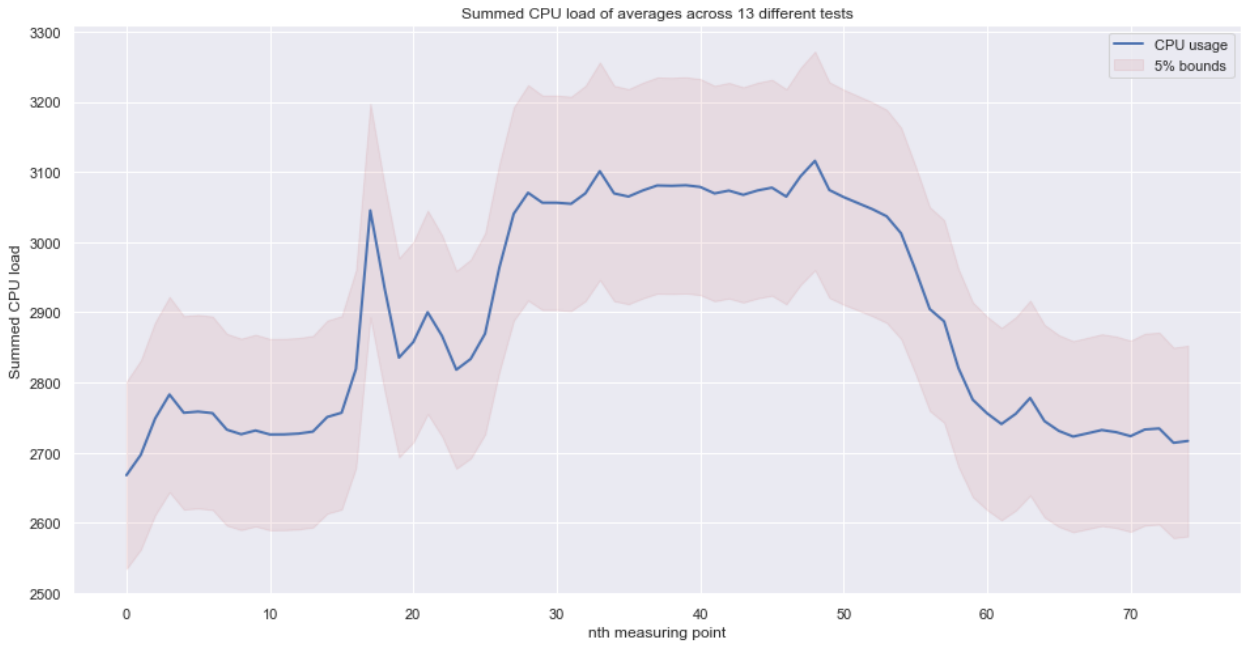
KUVA 11. Keskiarvotestin dataprofiili

3.3 Yksittäisten testien vertaaminen keskiarvotestiin

Ajatus yksittäisten testien vertailemisesta keskiarvotestiin syntyi yksinkertaisen intuition perusteella. Hyväksytyiksi luokitellut testit ovat dataprofiileiltaan hyvin saman näköisiä. Dataprofiilin muodostamissa arvoissa kuitenkin oli jonkin verran vaihtelua. Keskiarvotestin luomisella nämä vaihtelut pyrittiin asettamaan keskimääräiselle tasolle ja arvojen keskihajonnan pääteltiin määrittelevän, paljonko arvot tietyllä ajanhetkellä vaihtelevat. Aluksi tietyllä ajanhetkellä tapahtuvaa prosessien suoritinkäytön keskihajontaa käytettiin viuhkaprofiilin luomiseen. Lopulta viuhkaprofiilin rajoiksi asetettiin 5 % keskiarvon saamasta arvosta tietyllä ajanhetkellä. Tulokset keskihajonnan ja 5 %:n keskiarvon käyttämisen välillä olivat lähes samanlaiset sillä erotuksella, että jälkimmäisellä menetelmällä ei havaita todentamisessa käytetystä hyväksyttyjen testien aineistosta yhtään vääriä positiivisia.

Viuhkaprofiili määrittelee suoritinkäytön yleisen tason ylä- ja alarajan keskiarvotestin summakäyrän ympärille. Asiaa on havainnollistettu kuvassa 11. Sininen käyrä kuvaa yleistä suoritinkäytöntasoa ja sen ympärillä oleva punainen alue

viuhkaprofiilia. Ajanhetkellä t viuhkaprofiilin ylä- ja alataso lasketaan siis kaavalla 1.



KUVA 12. Keskiarvotestin summakäyrä ja viuhkaprofiili

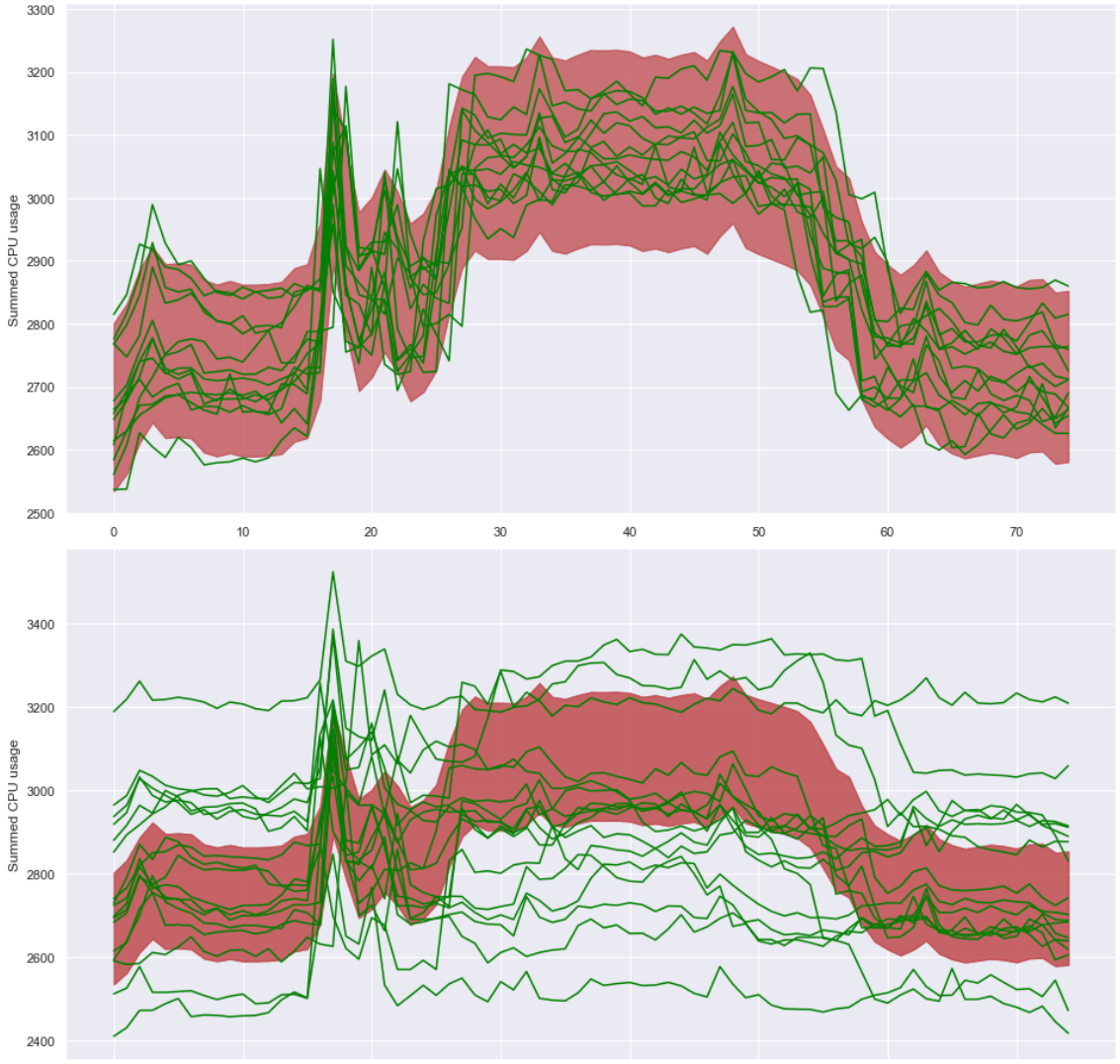
$$X_t = \mu_t \pm 0{,}05\mu_t$$ KAAVA 1

X_t = ylä- tai alataso ajanhetkellä t

μ_t = keskiarvo ajanhetkellä t

5 % keskiarvosta todettiin asettavan rajat, joiden sisälle hyväksytyiksi luokiteltujen testien suoritinkäytön summakäyrä pääsääntöisesti asettuu. Sillä havaittiin myös suurin määrä oikeita positiivisia epäonnistuneiden testien aineistoa tarkasteltaessa. Kaikki arvot, jotka eivät pysy näiden rajojen sisällä, luokiteltiin poikkeamiksi. Kuvassa 13 on esitetty viuhkaprofiili ja sekä hyväksyttyjen että hylättyjen testien summakäyrät. Kuvasta nähdään hyväksyttyjen testien tuottavan poik-

keavuuksia. Kaikki arvot, jotka jäävät viuhkaprofiilin ulkopuolelle, tulkitaan poikkeavuuksiksi. Hyväksytyissä testeissä esiintyvien poikkeavuuksien lukumäärän perusteella laskettiin poikkeamien määrän keskiarvo- ja hajonta.



KUVA 13. Yksittäiset testit ja viuhkaprofiili. Yllä onnistuneet ja alla epäonnistuneet testit

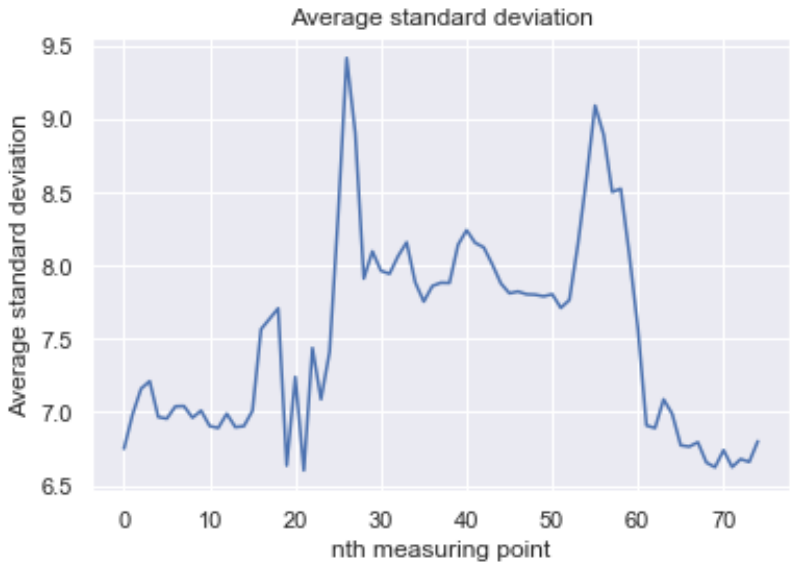
Sallituksi poikkeamien määräksi asetettiin poikkeaminen määrän keskiarvon ja keskihajonnan summa. Hyväksytyiksi luokitelluista testeistä kaikki jäävät tämän rajan alle eli vääriä positiivisia ei havaita yhtään. Epäonnistuneissa testeissä esiintyvien poikkeamien määrää tähän rajaan verratessa noin 88 % jää rajan yläpuolelle eli 12 % näistä testeistä jää havaitsematta. Tällä yksinkertaisella menetelmällä saatua tulosta pidettiin pohjatasona, joihin kaikkia muita menetelmiä verrattiin.

3.4 Tietoaineiston tilastollisten ominaisuuksien tutkiminen

Käytettyjen tietoaineistojen tilastollisia ominaisuuksia täytyi tutkia, jotta voitiin tehdä päätelmiä testattavista poikkeavuuksien havaitsemisessa käytetyistä menetelmistä, ja siitä tarvitseeko tietoaineistoja jatkokäsitellä ennen niiden sovittamista valittuihin menetelmiin. Tietoaineiston arvojen jakaumaa tarkastelemalla pystyttiin karsimaan pois tilastollisista poikkeavuudenhavaitsemismenetelmistä normaalijakaumaan perustuvat menetelmät. Luvun 2 kuvassa 4 esitettiin ketjutetun summakäyrän jakauma, joka ei siis noudata Gaussin kellokäyrää.

ARIMA-mallia ja LSTM-neuroverkkoa varten tietoaineiston stationäärisyyttä täytyi testata, sillä niiden käyttö olettaa syötetyn datan olevan stationääristä. Aikasarjadatan sanotaan olevan stationääristä, mikäli sen tilastolliset ominaisuudet kuten keskihajonta tai varianssi eivät muutu ajan suhteen. Tyypillisesti aikasarjassa esiintyvä trendi tai kausittaisuus vähentää sen stationäärisyyttä. (18.) Prosesseja, jotka tuottavat stationääristä aikasarjadataa, on helpompi analysoida. Niiden käyttäytymistä pystytään ennustamaan tarkemmin sillä tapa, jolla ne muuttuvat, on ennakoitavissa. (19.) Kuvasta 14 pystytään testien keskimääräistä keskihajontaa ajan suhteen tarkastelemalla päättelemään, ettei aikasarjadatamme ole stationääristä sillä keskihajonta vaihtelee suuresti ajan

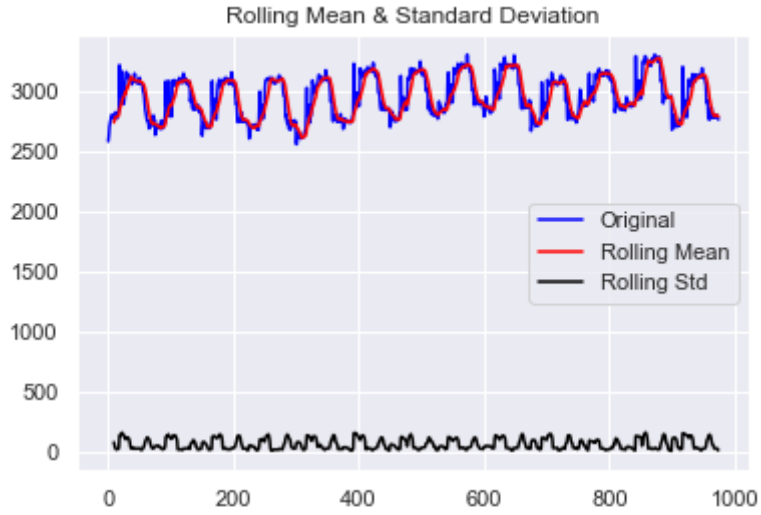
suhteen. Kuvasta stationäärisyyden havaitseminen on melko suoraviivaista. Pidempää aikasarjaa tarkasteltaessa se voi kuitenkin olla haastavaa.



KUVA 14. Testien keskimääräinen keskihajonta ajan suhteen

Stationäärisyyttä pystytään testaamaan Dickey-Fullerin testillä. Dickey-Fullerin testin nollahypoteesina on, että tarkasteltava aikasarja sisältää yksikköjuuren eli stationäärisyyttä vähentävän komponentin. Kuvassa 15 on esitetty ketjutetun aikasarjadatan Dickey-Fuller-testin tulokset. Tuloksia tulkitaan tarkastelemalla Test Statistic -arvoa ja vertaamalla sitä kriittisiin arvoihin (Critical Value). Mikäli Test Statistic -arvo on pienempi kuin kriittiset arvot, voidaan hylätä nollahypoteesin ja todeta aikasarjan olevan stationääristä. Kuvassa Test Statistic -arvo on pienempi

kuin yksikään kriittisistä arvoista, joten voimme päätellä sen olevan yli 99 % luottamuksella stationääristä. (20.)

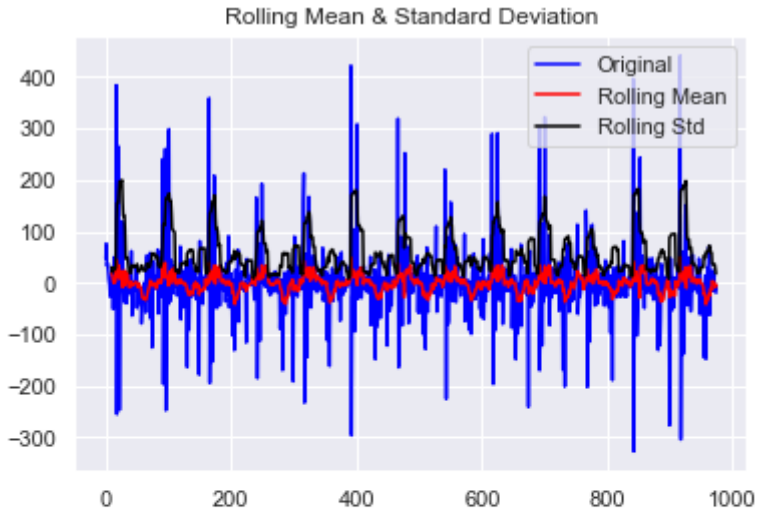



```
Results of Dickey-Fuller Test:
Test Statistic                -8.421065e+00
p-value                        1.982978e-13
#Lags Used                     2.200000e+01
Number of Observations Used    9.520000e+02
Critical Value (1%)           -3.437238e+00
Critical Value (5%)           -2.864581e+00
Critical Value (10%)          -2.568389e+00
dtype: float64
```

KUVA 15. Ketjutetun aikasarjadatan Dickey-Fuller-testin tulokset

Dickey-Fullerin testi suoritettiin myös käsittelemättömälle aikasarjadatalle, josta ei ollut poistettu testien alku- ja loppuvaiheen dataa. Tällä aineistolla Test Statistic -arvo oli suurempi kuin 10 %:n kriittinen arvo eli ei voitu sanoa edes 90 % varmuudella sen olevan stationääristä.

Datan stationäärisyyden lisäämiseksi on olemassa monenlaisia keinoja. Näistä yleisin on datan muuntaminen niin, että tarkastellaankin kahden aikapisteen välistä erotusta eli tarkastellaan todellisten arvojen sijaan muutoksen suuruutta (20). Tällä menetelmällä käsittelemättömästä datasta saatiin stationääristä, mutta käsitellyn datan stationäärisyys itse asiassa pieneni. Kuvassa 16 on esitetty käsitellyn erotusdatan Dickey-Fuller-testin tulokset. Data on silti edelleen yli 99 % varmuudella stationääristä, mutta Test Statistic -arvo on suurempi kuin suoritinkäytön summakäyrää tarkasteltaessa.




```
Results of Dickey-Fuller Test:
Test Statistic                -6.272788e+00
p-value                        3.966665e-08
#Lags Used                     1.600000e+01
Number of Observations Used    9.570000e+02
Critical Value (1%)           -3.437202e+00
Critical Value (5%)           -2.864565e+00
Critical Value (10%)          -2.568381e+00
dtype: float64
```

KUVA 16. Käsitellyn erotusdatan Dickey-Fuller-testin tulokset

Testejä, joista alku- ja loppuosat on poistettu, voidaan käyttää sellaisenaan käyttämiemme menetelmien testaamiseen. Datan stationäärisyyden tutkiminen on silti syytä pitää mielessä, kun menetelmiä aletaan soveltamaan uuteen dataan. Dickey-Fullerin testillä voidaan tarkastaa, vaatiiko data jatkokäsittelyä, ennen kuin sitä voidaan hyödyntää.

3.5 Ennustusmenetelmien virheen vertailutason tuottaminen

ARIMA-malli ja LSTM-neuroverkko perustuvat ennustuksen luomiseen opitun aikasarjan perusteella. Jotta voimme arvioida, kuinka hyvin ennustaminen toimii, täytyy ensin tuottaa jonkinlainen yksinkertainen pohjataso ennustusvirheelle. Eräs hyvä menetelmä tämän pohjatason tuottamiseen on ns. jatkuvuusalgoritmi eli naiivi ennustaminen. Tämän menetelmän mukaan ajanhetkellä $t + 1$ tutkittavan arvon tulisi olla ajanhetkeä t vastaava arvo. Ennustuksena käytetään siis alkuperäistä aikasarjaa, mutta sitä on siirretty yhden pykälän verran. (21.) Naiivin

ennustamisen tuottamien virhemetriikoiden arvoja verrataan tarkasteltavan menetelmän tuottamiin arvoihin. Tutkittavan menetelmän tuottaman virheen tulisi olla pienempi kuin naiivin ennustamisen tuottama virhe, jotta sen voidaan todeta olevan jatkotutkimuksen arvoinen.

Virheen laskemiseksi on olemassa useita metriikoita. Näistä hyvin yleisesti käytössä on keskineliövirhe eli MSE (engl. mean squared error) ja sen neliöjuuri eli RMSE (engl. root mean squared error). Keskineliövirhe lasketaan vertailemalla todellisten arvojen ja ennustuksen erotuksen neliötä ja se lasketaan kaavalla 2. (22.)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_t - \tilde{y}_t)^2$$ KAAVA 2. (22.)

MSE = keskineliövirhe

n = tarkasteltavien pisteiden määrä

y_t = todellinen arvo ajanhetkellä i

$\tilde{y}_t$ = ennustus ajanhetkellä i

Toiseen potenssiin korottaminen varmistaa virheen arvon säilyvän positiivisena. Sen myötä MSE on myös herkkä suurille virheille. Jos virhettä halutaan tarkastella samassa yksikössä kuin tarkasteltavat arvot, käytetään sen neliöjuurta eli RMSE:tä. RMSE:n laskeminen voidaan esittää kaavalla 3. (22.)

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_t - \tilde{y}_t)^2}$$ KAAVA 3. (22.)

RMSE = keskineliövirheen neliöjuuri

n = tarkasteltavien pisteiden määrä

y_t = todellinen arvo ajanhetkellä t

$\tilde{y}_t$ = ennustus ajanhetkellä t

MSE:n ja RMSE:n käyttäminen johti suuriin virhelukemiin, sillä käytetyt mallit eivät kyenneet varautumaan ketjutettua summakäyrää käytettäessä testien vaihtokohdissa tapahtuvaan suoritinkäytön laskuun. Tämän vuoksi virheen vertailussa käytettiin myös absoluuttista keskivirhettä eli MAE:tä (engl. mean absolute error). Absoluuttinen keskivirhe on siis virheen itseisarvon keskiarvo. Se voidaan laskea kaavalla 4. (22.)

$$MAE = \frac{1}{n}\sum_{i=1}^{n} | y_t - \tilde{y}_t |$$ KAAVA 4. (22.)

MAE = absoluuttinen keskivirhe

n = tarkasteltavien pisteiden määrä

y_t = todellinen arvo ajanhetkellä t

$\tilde{y}_t$ = ennustus ajanhetkellä t

Virhemetriikoiden laskemisessa käytettiin scikit-learn-kirjaston funktioita. Taulukossa 1 on esitetty naiivin ennustuksen tuottamat ennustusvirhemetriikat.

TAULUKKO 1. Naiivin ennustuksen virhemetriikat

Metriikka	**Arvo**
MSE	5110,95
RMSE	71,49
MAE	38,78

4 TESTATTAVAT MENETELMÄT

Työssä testattiin useita aikasarjadatan poikkeavuuksien havaitsemisessa käytettäviä menetelmiä. Näistä yksinkertaisimpia ovat alipäästösuodatukseen perustuvat menetelmät. Näistä menetelmistä testattiin liukuvan keskiarvon menetelmää ja eksponentiaalista tasoitusta, joita harkittiin myös sopiviksi pohjatason määrittämiseen. ARIMA-mallia ja LSTM-neuroverkkoa käytettiin ennustusten luomiseen käsiteltävän datan pohjalta. K-Means-klusteroinnilla pyrittiin aikasarjadatasta löytämään sille tyypillisiä suoritinkäyttösekvenssejä. Matriisiprofiililla tarkasteltiin miten samanlaisia suoritinkäyttödatasta pilkotut segmentit ovat keskenään. Menetelmiin päädyttiin kattavien verkkohakujen perusteella. Internetistä haettiin tietoa yleisesti käytettävistä aikasarjadatan poikkeavuuksien havaitsemisessa käytetyistä menetelmistä. Menetelmät valittiin sen perusteella, kuinka nopeasti menetelmistä kyettiin luomaan toimiva prototyyppi testausta varten ja kuinka hyvin ne soveltuvat käsiteltävään dataan. Valtaosaan menetelmistä löydettiin hyvin käytännönläheiset ohjeet, joita noudattamalla menetelmien testaaminen oli suoraviivaista. Alustavia tuloksia saatiin nopeasti, mutta usein menetelmien vaatimien parametrien hienosäätö ja syvällinen ymmärtäminen olisi vaatinut kohtuuttoman paljon aikaa.

4.1 Alipäästösuodatukseen perustuvat menetelmät

Liukuvan keskiarvon menetelmät ja eksponentiaalinen tasoitus ovat matemaattisessa mielessä aikasarjadatan alipäästösuodatusta. Ne poistavat kohinaa käsiteltävästä aikasarjasta, tasoittavat lyhytaikaisia vaihteluita ja tuovat pitkän aikavälin trendin esille. (23.) Näitä menetelmiä käytettäessä poikkeavuuksien havaitsemiseen tarkoituksena on luoda alkuperäisestä aikasarjasta alipäästösuodatettu versio, johon alkuperäistä dataa verrataan. Jokaista menetelmää testatessa luotiin ensin menetelmän mukainen malli ketjutetusta aikasarjadatasta, yksittäisistä testeistä ja lopulta keskiarvotestistä. Viuhkaprofiili luotiin asettamalla sen rajat samalla tavalla kuin keskiarvotestiin vertailtaessa. Jokaisessa menetelmässä poikkeavuuksia havaittiin myös hyväksytyiksi luokitelluissa testeissä, joten havaittujen poikkeamien keskiarvo ja -hajonta laskettiin ja epäonnistuneiksi luokiteltujen testien poikkeavuusmääriä vertailtiin näihin lukemiin.

4.1.1 Yksinkertainen liukuva keskiarvo

Yksinkertaista liukuva keskiarvoa varten täytyy määrittää, kuinka pitkältä aikaväliltä keskiarvoa lasketaan. Jos esimerkiksi liukuvan keskiarvon ikkunan koko $n = 5$, lasketaan aikasarjan jokaiselle ajanhetkelle t keskiarvo alkaen itsestään ja $n - 1$ edellisestä arvosta. (24.) Taulukossa 2 on esitelty yksinkertaisen liukuvan keskiarvon menetelmällä saatuja tuloksia. Ketjutettua summakäyrää tai testejä itsenäisesti tarkastellessa ei voida vielä todeta menetelmän havaitsevan merkittävästi enemmän poikkeavuuksia hyväksytyissä testeissä kuin hylätyissä. Keskiarvotestistä luotuun käyrään vertailtaessa puolestaan ero on selvä, mutta tulokset ovat hieman heikompia verrattuna keskiarvotestin pohjalta luotuun viuhkaprofiiliin.

TAULUKKO 2. Yksinkertaisen liukuvan keskiarvon avulla saatuja tuloksia

Tarkasteltava data	**Poikkeamia, hyväksytyt testit**	**Poikkeamia, hylätyt testit**
Pohjataso yksinkertaisella viuhkaprofiililla	0 kpl / 0 %	15 kpl / 88,24 %
Ketjutettu summakäyrä	246 kpl / 25,23 %	345 kpl / 27,06 %
Yksittäiset testit itsenäisesti	3 kpl / 23,08 %	5 kpl / 29,41 %
Yksittäiset testit verrattuna keskiarvotestistä luotuun alipäästösuodatettuun käyrään	0 kpl / 0 %	14 kpl / 82,35 %

4.1.2 Painotettu liukuva keskiarvo

Painotetussa liukuvassa keskiarvossa tarkasteltavien pisteiden arvot saavat erilaisen painotuksen keskiarvon laskennassa. Tyypillisesti uudemmat aikapisteet

saavat suuremman painotuksen kuin myöhemmät. Painokertoimet voidaan laskea tasaisesti aikapisteiden välille tai niiden arvon kasvaminen tai pieneneminen voi olla eksponentiaalista. (24.) Työssä käytettiin eksponentiaalisesti painotettua liukuvaa keskiarvoa Pandas-kirjastosta löytyvää funktiota käyttäen. Taulukossa 3 on esitelty testauksen tulokset. Ketjutetuissa testeissä tulokset olivat hyvin samankaltaisia yksinkertaisen liukuvan keskiarvon kanssa. Itsenäisiä hylättyjä testejä tarkastellessa oikein poikkeaviksi tunnistettiin nyt yli puolet. Keskiarvotestistä luodun eksponentiaalisesti painotetun ja tavallisen liukuvan keskiarvon välillä tuloksissa ei ollut eroja.

TAULUKKO 3. Eksponentiaalisesti painotetun liukuvan keskiarvon tulokset

Tarkasteltava data	**Poikkeamia, hyväksytyt testit**	**Poikkeamia, hylätyt testit**
Pohjataso yksinkertaisella viuhkaprofiililla	0 kpl / 0 %	15 kpl / 88,24 %
Ketjutettu summakäyrä	155 kpl / 15,90 %	216 kpl / 16,94 %
Yksittäiset testit itsenäisesti	2 kpl / 15,38 %	10 kpl / 58,82 %
Yksittäiset testit verrattuna keskiarvotestistä luotuun alipäästösuodatettuun käyrään	0 kpl / 0 %	14 kpl / 82,35 %

4.1.3 Eksponentiaalinen tasoitus

Eksponentiaalinen tasoitus muistuttaa hyvin paljon eksponentiaalisesti painotetun liukuvan keskiarvon laskemista sillä erotuksella, että tätä menetelmää soveltaessa ei tarkastella aikasarjaa tietyn kokoisissa segmenteissä. Sen sijaan ajanhetkellä t lasketaan eksponentiaalisesti painotetun keskiarvon käyttäen kaikkia

edeltäviä aikapisteitä. Yksinkertainen eksponentiaalinen tasoitus lasketaan kaavalla 5.

$$\check{y}_t = \alpha \cdot y_t + (1 - \alpha) \cdot \check{y}_{t-1} \qquad \text{KAAVA 5}$$

$\check{y}_t$ = arvo ajanhetkellä t

α = tasoituskerroin, jonka arvo on $0 < \alpha < 1$

Aikasarjadatan käsittelyssä yleisesti ovat käytössä myös kaksinkertainen eksponentiaalinen tasoitus, joka ottaa huomioon aikasarjan trendin, ja kolminkertainen eksponentiaalinen tasoitus eli Holt-Wintersin-metodi, joka ottaa huomioon edellisen lisäksi myös kausittaisuuden. (25.) Työssä käytettiin yksinkertaista eksponentiaalista tasoitusta. Taulukossa 4 esitellään tulokset. Ketjutettua dataa tarkasteltaessa poikkeavuuksia havaittiin vähemmän kuin edellisillä metodeilla. Itsenäisesti hylättyjä testejä tarkasteltaessa poikkeamia havaittiin oikein heikommin kuin eksponentiaalisesti painotettua liukuvaa keskiarvoa käytettäessä. Keskiarvotestin pohjalta luodun eksponentiaalisen tasoituksen tulokset olivat samanlaiset kuin yksinkertaisen keskiarvotestivertailun. Suuremmalla tietoaineistolla voitaisiin näiden välillä huomata jonkinlainen ero.

TAULUKKO 4. Eksponentiaalisen tasoituksen tulokset

Tarkasteltava data	**Poikkeamia, hyväksytyt testit**	**Poikkeamia, hylätyt testit**
Pohjataso yksinkertaisella viuhkaprofiililla	0 kpl / 0 %	15 kpl / 88,24 %
Ketjutettu summakäyrä	139 kpl / 14,26 %	196 kpl / 20,10 %
Yksittäiset testit itsenäisesti	3 kpl / 23,08 %	8 kpl / 47,06 %
Yksittäiset testit verrattuna keskiarvotestistä luotuun alipäästösuodatettuun käyrään	0 kpl / 0 %	14 kpl / 88,24 %

4.2 ARIMA-malli

ARIMA-mallit ovat ryhmä erilaisia tilastollisia malleja, joilla voidaan analysoida ja ennustaa aikasarjadataa. Se on lyhenne englanninkielisestä termistä Autoregressive Integrated Moving Average eli autoregressiivinen integroitu liukuva keskiarvo. ARIMA-malleja ei ole olemassa vain yhtä, vaan niistä puhuttaessa käytetään tyypillisesti merkintätapaa ARIMA(p, d, q), jossa mallin saamat kirjainparametrit ovat kokonaislukuja. (26.) Mallin ominaisuuksista johtuen tietyt mallit ovat matemaattisesti vastaavia muiden ennustusmenetelmien kanssa. Esimerkiksi ARIMA(0, 1, 0) tuottaa satunnaiskulkua vastaavia ennustuksia. ARIMA(0, 1, 1) puolestaan on yksinkertaista ja ARIMA(0, 2, 2) kaksinkertaista eksponentiaalista tasoitusta. (27.) Mikäli jokin mallin parametreista saa arvon 0, tarkoittaa tämä käytännössä, ettei kyseistä ominaisuutta käytetä. Näin esimerkiksi ARIMA(1, 0, 1)-mallia voitaisiin kutsua myös ARMA-malliksi. (26.) ARIMA-mallin parametrien

valinta on yritystä ja erehdystä vaativa prosessi, jossa kokenut menetelmän soveltaja voi kuitenkin tehdä valistuneita arvauksia mahdollisesti toimivista parametriyhdistelmistä.

4.2.1 Mallin parametrit

ARIMA-mallin parametri p viittaa autoregressioon. Autoregressiossa oletetaan aikasarjan aikaisempien arvojen vaikuttavan nykyhetkeen ja tulevaisuuteen. Parametrin arvo pyrkii selittämään p:n aikaisemman arvon vaikutusta nykyhetkeen ja tulevaisuuteen yhdistettynä riippumattomaan ja identtisesti jakautuneeseen satunnaislukujen joukkoon. (26, 28.)

Parametri d viittaa edellisten arvojen erotuksen tarkasteluun. Sen arvo määrittelee, kuinka monesti erotuksia suoritetaan. ARIMA-malliin kuuluvan integroinnin tavoitteena on käsiteltävän datan stationäärisyyden vähentäminen. (26; 28.)

Liukuvan keskiarvon parametri q viittaa liukuvan keskiarvon ikkunan kokoon eli siihen, kuinka monelta edelliseltä havainnolta lasketaan liukuva keskiarvo. Todellisia arvoja ja liukuvaa keskiarvoa verrataan keskenään ja näiden kahden jäännösvirhettä käytetään mallin luomiseen. (26, 28)

4.2.2 Parametrien optimointi

ARIMA-mallin parametrien määrittämiseen voidaan käyttää Box-Jenkinsin metodia (29). Sen soveltaminen voi kuitenkin olla hyvin haastavaa ja aikaa vievää ilman syvällistä ymmärrystä metodin ja ARIMA-mallin matemaattisista taustoista. Parametrien optimoimiseksi voidaan käyttää koneoppimisessa yleisesti käytettyä hyperparametrien optimointiin tarkoitettua taulukkoetsintää. Taulukkoetsinnässä mallin erilaisia parametriyhdistelmiä käydään läpi järjestelmällisesti. Jokaisesta parametriyhdistelmästä muodostetaan sen mukainen ARIMA(p, d, q)-malli, joka sovitetaan tarkasteltavaan dataan. Menetelmässä ensimmäiset 66 % ketjutetun aikasarjan datapisteistä valittiin mallin kouluttamisessa käytettäviksi arvoiksi ja loppua 34 %:a käytettiin ennustamiseen. Ennustuksen ja todellisten arvojen välistä ennustusvirhettä vertailtiin keskenään. Pienimmän ennustusvirheen tuottama malli valittiin. Ennustusvirhettä mittaavaksi metriikaksi valittiin MSE. (30.)

Noin tuhannen datapisteen ja 63 erilaisen parametriyhdistelmän läpikäymisessä kestää yrityskäyttöön tarkoitetulla kannettavalla tietokoneella noin 3,5 tuntia eli menetelmä ei soveltuisi esimerkiksi tuotantokäytössä dynaamiseen ARIMA-mallin parametrien määrittämiseen. Taulukossa 5 on esitetty taulukkoetsinnän tulokset. Punaisella merkityt solut ovat saaneet suurempia arvoja kuin naiivin ennustamisen mallilla saadut tulokset. Vihreät arvot puolestaan ovat saaneet pienempi arvoja kuin naiivin ennustamisen tulokset. Pienimmän arvon saa malli ARIMA(10, 0, 0). Tämä viittaisi integroinnin ja liukuvan keskiarvon olevan tarpeettomia ennustusten tekemiseen käsiteltävän datan pohjalta. Autoregression vaikutukset puolestaan täytyy huomioida kymmeneltä edelliseltä aikapisteeltä.

TAULUKKO 5. ARIMA-mallin parametriyhdistelmien keskineliövirhe

Malli	MSE
ARIMA(0, 0, 0)	28610,297
ARIMA(0, 0, 1)	12516,961
ARIMA(0, 0, 2)	8892,462
ARIMA(0, 1, 0)	5119,006
ARIMA(0, 1, 1)	4873,348
ARIMA(0, 1, 2)	4803,663
ARIMA(0, 2, 0)	12155,328
ARIMA(0, 2, 1)	5133,111
ARIMA(1, 0, 0)	4890,738
ARIMA(1, 0, 1)	4747,685
ARIMA(1, 0, 2)	4698,734
ARIMA(1, 1, 0)	4959,801
ARIMA(1, 1, 1)	4831,337
ARIMA(1, 1, 2)	4798,758
ARIMA(1, 2, 0)	8988,996
ARIMA(2, 0, 0)	4797,529
ARIMA(2, 0, 1)	4721,573
ARIMA(2, 0, 2)	4690,875
ARIMA(2, 1, 0)	4775,78
ARIMA(2, 1, 1)	4779,26
ARIMA(2, 1, 2)	4799,436
ARIMA(2, 2, 0)	7115,257
ARIMA(2, 2, 1)	4788,712
ARIMA(4, 0, 0)	4678,166
ARIMA(4, 0, 1)	4693,668
ARIMA(4, 1, 0)	4812,139
ARIMA(4, 1, 1)	4812,188
ARIMA(4, 1, 2)	4819,833
ARIMA(4, 2, 0)	6006,807
ARIMA(4, 2, 1)	4826,044
ARIMA(6, 0, 0)	4704,213
ARIMA(6, 0, 1)	4700,334
ARIMA(6, 1, 0)	4813,574
ARIMA(6, 1, 1)	4813,373
ARIMA(6, 2, 0)	5577,615
ARIMA(6, 2, 1)	4826,463
ARIMA(8, 1, 0)	4788,991
ARIMA(8, 1, 1)	4793,279
ARIMA(8, 2, 0)	5415,27
ARIMA(8, 2, 1)	4943,246
ARIMA(8, 2, 2)	4958,841
ARIMA(10, 0, 0)	4675,525
ARIMA(10, 1, 0)	4846,703
ARIMA(10, 1, 1)	4825,365
ARIMA(10, 1, 2)	4824,582
ARIMA(10, 2, 0)	5186,985
ARIMA(10, 2, 1)	4980,992
ARIMA(10, 2, 2)	4975,742

Osa parametriyhdistelmistä johtaa ajonaikaiseen virheeseen. Näiltä malleilta ei näin ollen ole saatu tuloksia. Koodin suoritusvaiheessa tähän varauduttiin yksinkertaisella try-catch-rakenteella, jotta optimointiprosessi ei keskeytyisi.

4.2.3 Menetelmän soveltaminen poikkeavuuksien havaitsemisessa

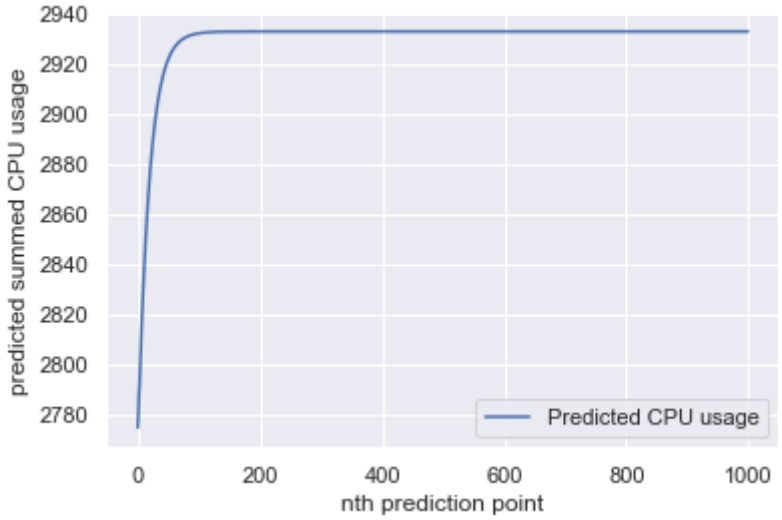
Taulukosta 5 nähdään, ettei yksikään ARIMA-malli pääse virheettömään ennustukseen. Kuvassa 17 on esitetty ARIMA(10, 0, 0)-mallilla toteutettu ennustuksen ja todellisten arvojen vertailu. Punainen käyrä on ARIMA-mallin tuottama ennustus, sininen todelliset arvot. Tyypillisesti ARIMA-malli reagoi suuriin muutoksiin pienellä viiveellä eivätkä sen tuottamat piikit ja laskut saavuta aivan samanlaisia lukemia kuin todelliset arvot. Koska ARIMA-malli ei kykene reagoimaan yllättäviin tilanteisiin kovin luotettavasti, poikkeavan datan pohjalta luotujen mallien täytyisi ainakin teoriassa tuottaa suurempi ennustusvirhe kuin normaalin datan pohjalta luotujen mallien.

KUVA 17. ARIMA-mallin mukainen ennustus ja todelliset arvot

Hyväksyttyjen ja hylättyjen ajojen vertailu ARIMA-mallia käyttäen aloitettiin muuttamalla koulutus- ja testausdatapisteiden määrän suhdetta. Malli sovitettiin ensin

koko ketjutettuun dataan, jonka jälkeen mallin tehtäväksi annettiin ennustaa yksittäisen hyväksytyn testin käyttäytyminen. Tämä toistettiin jokaiselle hyväksytylle testille erikseen. Ennustusten virhemetriikoiden keskiarvo ja -hajonta laskettiin. Virhemetriikoina käytettiin MSE:tä ja MAE:tä. Tämän jälkeen hylätyiksi luokiteltujen testien pohjalta luotiin malli käyttäen samaa menetelmää eli yksittäiset testit liitettiin erikseen osaksi hyväksyttyjen testien ketjutettua dataa. Hylättyjen testien ennustusvirhe laskettiin ja sitä vertailtiin hyväksyttyjen testien keskimääräiseen virheeseen. Ylärajana pidettiin hyväksyttyjen ajojen keskiarvon ja -hajonnan summaa. Testi tulkittiin hylätyksi, mikäli sen ennustuksen MSE tai MAE oli suurempi kuin yläraja. Malleja testatessa havaittiin ARIMA(10, 0, 0):n tuottavan tietyillä testeillä ajonaikaisia varoituksia, joiden vuoksi hyväksyttyjen testien virhemetriikat nousivat kohtuuttoman suuriksi. Myös parametrien optimoinnin perusteella toiseksi parhaaksi arvioitu malli tuotti virheitä. Virheettömään suoritukseen päästiin mallilla ARIMA(2, 0, 2). Yksinkertaisemman mallin käyttämisessä etuna oli myös testauksen suoritusajan pienentyminen tunneista noin kymmeneen minuuttiin. Epäonnistuneista ajoista menetelmällä tunnistettiin oikein ainoastaan 23,53 %, kun taas puolestaan onnistuneista ajoista saatiin vääriä positiivisia 27,27 %.

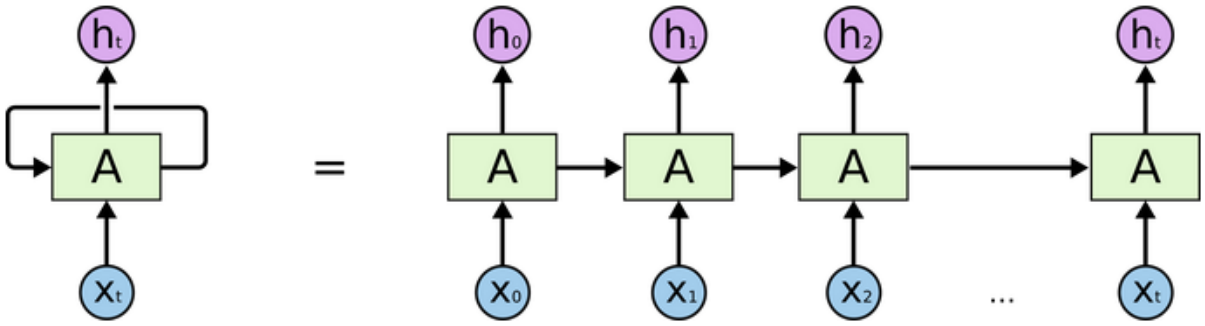
Mallia luotaessa se sovitetaan ensin koko koulutusaineistolle. Tämän jälkeen testausaineistoa ennustetaan iteroiden datapiste kerrallaan. Jokaisella iteraatiolla todellinen havainto lisätään koulutusaineiston viimeiseksi osaksi ja malli luodaan uudestaan. Tämä on välttämätöntä nimenomaan tietoaineiston autoregressiivisestä luonteesta johtuen. Aiemmat datapisteet vaikuttavat ennustuksiin ja tämä täytyy ottaa huomioon. (31.) Kuvassa 18 ARIMA-malli on ensin sovitettu koko ketjutettuun suoritinkäyttödataan ja tämän jälkeen sille on annettu tehtäväksi ennustaa seuraavat tuhat datapistettä ilman, että mallia sovitetaan uudestaan jokaisen datapisteen jälkeen. Huomataan mallin ennustusten asettuvan tietylle tasolle hyvin pian, jonka jälkeen se ei ennusta aikasarjassa tapahtuvan minkäänlaisia muutoksia. Iteroiva lähestymistapa, jolla hyödyllisiä ennustuksia saadaan, on puolestaan laskennallisesti hyvin raskasta. Yksittäistä mallia testattaessa testausaineiston ollessa noin 330 datapistettä ennustuksen luomisessa kestää noin 11 minuuttia käyttäen ARIMA(10, 0, 0)-mallia. Tästä johtuen mallin testaaminen koko data-aineistolla saattoi kestää jopa tunteja.



KUVA 18. Tuhannen datapisteen ennustus ilman iterointia

4.3 LSTM-neuroverkko

LSTM-neuroverkot (engl. Long Short-Term Memory Network) ovat eräänlaisia takaisinkytkeytyviä neuroverkkoja (engl. RNN, Recurring Neural Network). Rekursiivisen luonteensa vuoksi takaisinkytkeytyvät neuroverkot soveltuvat erityisen hyvin sekvenssimäisen datan käsittelyyn, jossa aikaisempien datapisteiden vaikutus nykyhetkeen ja tulevaisuuteen tulee ottaa huomioon. Ne siis kykenevät muistamaan ja hyödyntämään aikaisempaa tilaansa. Takaisinkytkeytyviä neuroverkkoja onkin käytetty esimerkiksi tekstin- ja puheentunnistukseen, jossa aikaisemmat sanat muodostavat kontekstin, johon käsiteltävää dataa verrataan. (32.) Kuvassa 19 on yksinkertaistettu takaisinkytkeytyvän neuroverkon rakennetta. Syötteen x_t perusteella neuroverkon osa A tuottaa ulostulon h_t. Tämä ulostulo syötetään takaisin neuroverkkoon, jossa sen vaikutus otetaan huomioon seuraavassa vaiheessa. (33.)

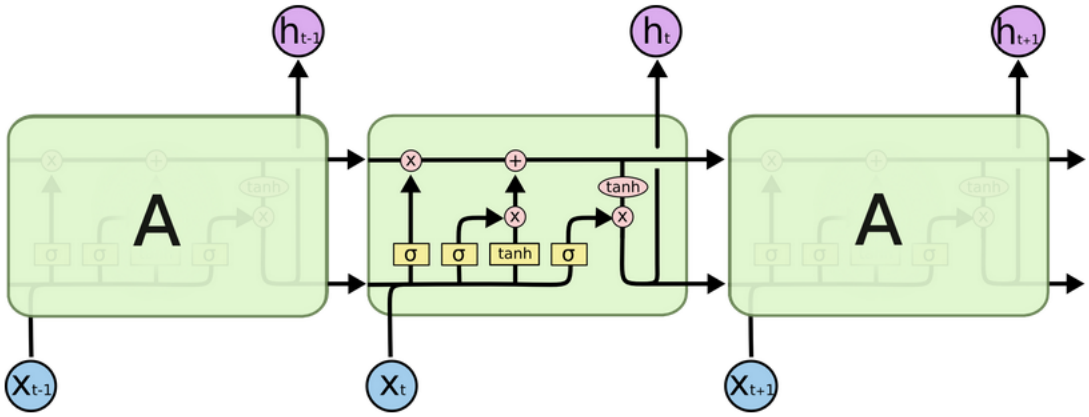


KUVA 19. Takaisinkytkeytyvän neuroverkon toimintaperiaate yksinkertaistettuna (33)

Yksinkertaisten takaisinkytkeytyvien neuroverkkojen on kuitenkin sitä vaikeampi yhdistää syötteitään ulostuloihin, mitä kauempana ne ovat toisistaan. LSTM-neuroverkoilla tämä ongelma pyritään välttämään. (33.)

4.3.1 Verkon yksittäisen solun rakenne

LSTM-neuroverkon ja sen yksittäisen solun rakenne on esitetty kuvassa 20. Keltaiset laatikot ovat neuroverkon kerroksia ja vaaleanpunaiset ympyrät syötteille suoritettavia pisteoperaatiota. Hyvin paljon yksinkertaistettuna neuroverkon solun voidaan ajatella toimivan kuten linjaston. Mustat viivat kuljettavat nuolten osoittamaan suuntaan n-ulotteisia vektoreita. Viivojen yhtymiskohdissa kaksi eri vektoria lasketaan yhteen ja erkanemiskohdissa vektori kopioidaan. Ylin vaakatasossa kulkeva viiva, jonka alkuarvo saadaan edelliseltä solulta syötteenä, kuvaa solun tilaa tietyllä ajanhetkellä. Alemmassa tasossa kulkeva viiva laskee yhteen edellisen solun lopetustilan h_{t-1} ja uuden syötteen X_t. Tämä yhdistetty vektori kuljetetaan neuroverkon solun eri kerrosten läpi, joissa sen arvoja päivitetään. Kerrosten läpi kuljetetuilla uusilla vektoreilla puolestaan päivitetään solun senhetkistä tilaa pisteoperaatioiden avulla. (33.)



KUVA 20. LSTM-neuroverkon ja sen solun rakenne (33)

Pelkistetysti voidaan sanoa symbolilla σ merkittyjen sigmoidifunktiokerrosten kontrolloivan solun tilaa. Tanh-kerroksella puolestaan pakotetaan vektoreiden arvot välille −1…1. Vastaavasti tanh-pisteoperaatiolla pakotetaan solun tila tälle välille. Solun tehtyä tehtävänsä sen tila välitetään seuraavalle solulle ja prosessia toistetaan haluttuun rajaan asti. (33.)

4.3.2 Neuroverkon rakentaminen ja testaus

Käytännössä neuroverkkoa sovitettaessa kehittäjällä ei tarvitse olla syvällistä ymmärrystä sen toiminnasta. Keras-rajapinta on abstrahoitu siten, että kehittäjän tarvitsee tietää vain neuroverkon solujen, koko tietoaineiston läpikäyntien ja kerralla käsiteltävien näytteiden määrä. Koulutusaineistoa läpikäydessään neuroverkko yrittää ennustaa aineiston käyttäytymistä ja minimoimaan ennustusvirheensä vaihtelemalla eri kerrosten käyttämiä arvoja. Ennustusvaiheessa neuroverkko soveltaa oppimaansa ja valitsee syötteidensä perusteella kerrosten saamat arvot tavalla, joka tuottaa pienimmän ennustusvirheen. (34.)

LSTM-neuroverkon oletettiin havaitsevan poikkeavuuksia samalla periaatteella kuin ARIMA-mallin. Sen tuottaman ennustusvirheen pitäisi olla suurempi hylätyillä testeillä kuin hyväksytyillä. Neuroverkon toimintaa testattiin ketjutetulla summakäyrällä. Käytettävää dataa täytyy esikäsitellä muotoilemalla ratkaistava ongelma valvotuksi koneoppimiseksi. Valvotussa koneoppimisessa syötteen x ja ulostulon y sisäinen riippuvuus pyritään etsimään. Kun sisäinen riippuvuus pystytään arvioimaan mahdollisimman tarkasti, voidaan ulostuloa yrittää ennustaa

syötteiden perusteella. LSTM-neuroverkon rakentaminen Keras-rajapinnan avulla vaatii datan olevan jaottelua syöte- ja ulostulokomponentteihin. Datasta voidaan muotoilla valvotun koneoppimisen ongelma käyttämällä havaintoa ajanhetkellä $t - 1$ syötteenä ja havaintoa ajanhetkellä t ulostulona. (34.)

Neuroverkolle syötettävän datan täytyy olla samassa skaalassa kuin sen käyttämän aktivointifunktion. LSTM-neuroverkko käyttää aktivointifunktionaan hyperbolista tangenttia, jonka arvot ovat välillä −1…1. Syötettävä data skaalataan tälle välille käyttämällä scikit-learn-kirjaston MinMaxScaler-moduulia. Syötettävän datan tulee olla myös mahdollisimman stationääristä. Helposti tähän vaatimukseen päästään käyttämällä alkuperäisten arvojen sijaan kahden perättäisen arvon erotusta. Neuroverkkoa koulutettaessa sen tappiofunktiona käytetään MSE:tä. Sen arvo siis pyritään minimoimaan. (34.)

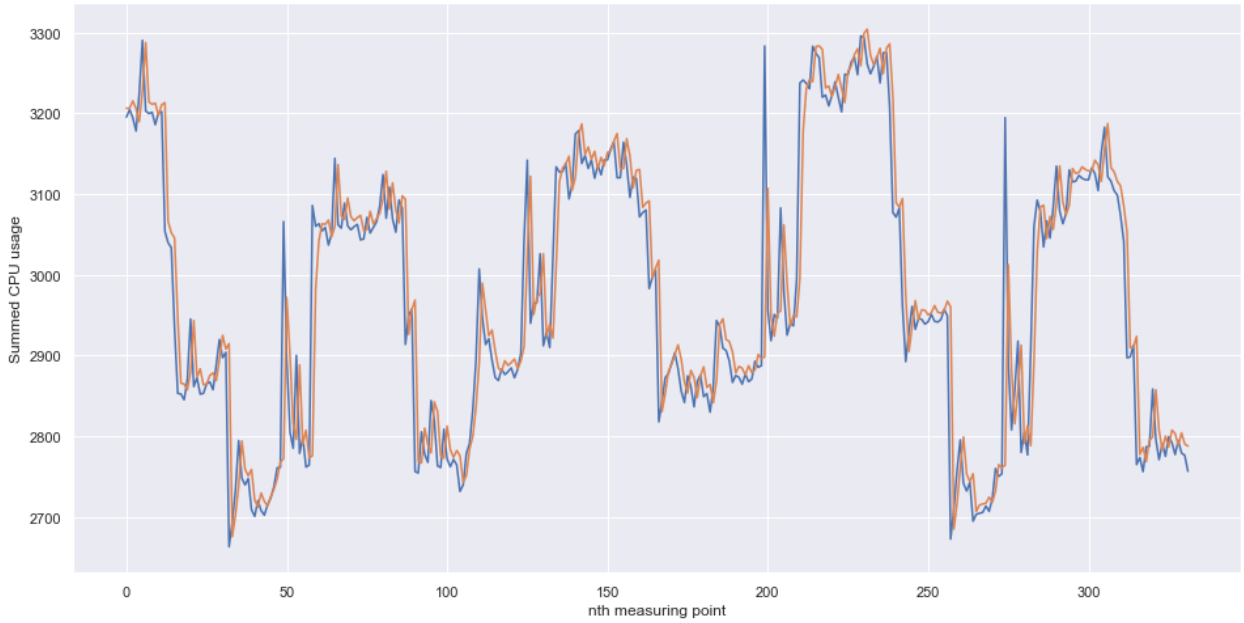
Kuvassa 21 on esitetty sovituksessa käytetty funktio. Parametreikseen se ottaa koulutuksessa käytettävän tietoaineiston, kerralla käsiteltävien näytteiden määrän, koko aineiston läpikäymiskerrat sekä solujen määrän neuroverkossa. Tietoaineistosta erotellaan syötekomponentti X ja ulostulokomponentti y ja se muotoillaan neuroverkolle sopivaan muotoon. Annetuilla parametreilla neuroverkko ensin käännetään, jonka jälkeen sitä aletaan sovittamaan. Käytetty lähde suositteli käsiteltävien näytteiden määräksi 1, aineiston läpikäymiskerroiksi 3000 ja solujen määräksi 4. (34)

```
def fit_lstm(train, batch_size, nb_epoch, neurons):
    X, y = train[:, 0:-1], train[:, -1]
    X = X.reshape(X.shape[0], 1, X.shape[1])
    model = Sequential()
    model.add(LSTM(neurons, batch_input_shape=(batch_size, X.shape[1], X.shape[2]), stateful=True))
    model.add(Dense(1))
    model.compile(loss='mean_squared_error', optimizer='adam')
    for i in range(nb_epoch):
        model.fit(X, y, epochs=1, batch_size=batch_size, verbose=0, shuffle=False)
        model.reset_states()
    return model
```

KUVA 21. Neuroverkon sovituksessa käytetty funktio (34)

Koulutuksessa käytettiin jälleen ensimmäistä 66 %:a ketjutetun summakäyrän arvoista, jonka jälkeen neuroverkolle annettiin tehtäväksi ennustaa loput 34 %. Neuroverkkoa koulutusdataan sovitettaessa lähestymistavan kääntöpuolet tulivat ilmeiseksi. Sovituksessa kestää 644 datapisteellä noin kolme tuntia. Tästä syystä

neuroverkkoa ei pystytty käyttämään yksittäisten ajojen vertailussa keskenään samalla tavalla kuin ARIMA-mallia. Sen tuottamat ennustukset datalla, josta ei ollut poistettu testien alku- ja loppupäistä tietoa, olivat kuitenkin tarkempia kuin yhdelläkään ARIMA-mallilla. Käsitellyllä datalla tulokset olivat samansuuntaisia kuin erilaisilla ARIMA-malleilla. Kuvassa 22 on esitetty neuroverkon tuottama ennustus esikäsitellyllä datalla. Oranssi käyrä kuvaa neuroverkon ennustusta ja sininen todellisia arvoja. Päällisin puolin ennustus vaikuttaisi olevan melko tarkka, mutta virhemetriikoita laskettaessa huomataan sen olevan vain hieman parempi kuin naiivi ennustaminen.



KUVA 22. Neuroverkon ennustus ja todelliset arvot käyttäen käsiteltyä dataa

Taulukossa 6 on esitelty LSTM-neuroverkon ja naiivin ennustuksen virhemetriikat. MSE ja RMSE ovat vain hieman pienempiä kuin taulukossa 1 esitellyt naiivin ennustuksen vastaavat metriikat. MAE puolestaan on naiivin ennustamisen kanssa lähes sama.

TAULUKKO 6. LSTM-neuroverkon tuottaman ennustuksen virhemetriikat

Virhemetriikka	**LSTM:n virhe**	**Naiivin ennustuksen virhe**
MSE	4601,90	5110,95
RMSE	67,84	71,49
MAE	38,70	38,78

Nämä arvot vaihtelevat hieman eri testikertojen välillä, vaikka parametreihin tai dataan ei tehtäisikään minkäänlaisia säätöjä. Suorituskyky ei kuitenkaan radikaalisti muutu testauskertojen välillä.

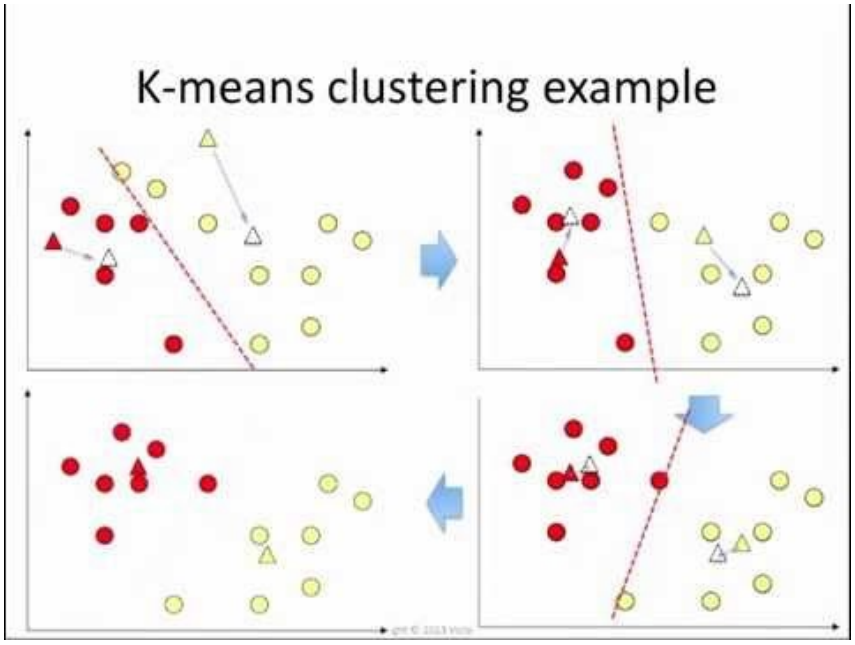
4.4 K-Means-klusterointi

K-Means on klusterointialgoritmi, joka pyrkii klusteroimaan käsiteltävän datan k kappaleeseen erillisiä klustereita. Jokaiselle erilliselle datapisteelle määritellään klusteri, johon se kuuluu, lähimmän klusterikeskustan perusteella. Datapisteet pyritään klusteroimaan niin, että keskenään samankaltaiset datapisteet kuuluvat samoihin klustereihin. Tyypillisesti datapisteen ja klusterin etäisyys lasketaan euklidisena etäisyytenä. (35)

4.4.1 Algoritmin toiminta

Klusterointi aloitetaan määrittelemällä k kappaletta klustereiden keskustoja sattumanvaraisesti. Tämän jälkeen määritellään, mihin klusteriin kukin datapiste kuuluu laskemalla niiden euklidinen etäisyys kuhunkin klusterin keskustaan. Datapiste määritellään kuuluvaksi siihen klusteriin, jonka keskusta on sitä lähinnä. Kun kaikille datapisteille on määritelty klusteri, lasketaan klustereille uudet keskustat. Uusi keskusta on kaikkien klusteriin kuuluvien datapisteiden keskiarvo. Tätä prosessia toistetaan, kunnes klusterin keskusta ei enää muutu, eli datapisteiden klusterit eivät vaihdu, tai kunnes haluttu määrä iteraatioita ollaan suoritettu.

(35.) Prosessia on havainnollistettu kuvassa 23. Värilliset kolmiot ovat klustereiden keskustoja ja ympyrät datapisteitä kaksiulotteisessa aineistossa. Keskustat on aluksi määritelty sattumanvaraisesti. Kun datapisteet on määrätty klustereihin, lasketaan keskustat uudestaan. Havaitaan kahden datapisteen itseasiassa kuuluvan punaiseen klusteriin, joten keskipisteet lasketaan uudestaan. Jälleen huomataan yhden datapisteen kuuluvan eri klusteriin, kuin mihin se oli alun perin määrätty ja keskipisteet lasketaan uusiksi. Lopulta klustereiden keskustat vaihtavat hieman paikkaa, mutta yksikään datapiste ei enää vaihda klusteria. Algoritmi on saavuttanut päätepisteensä ja klusterointi lopetetaan. (36)



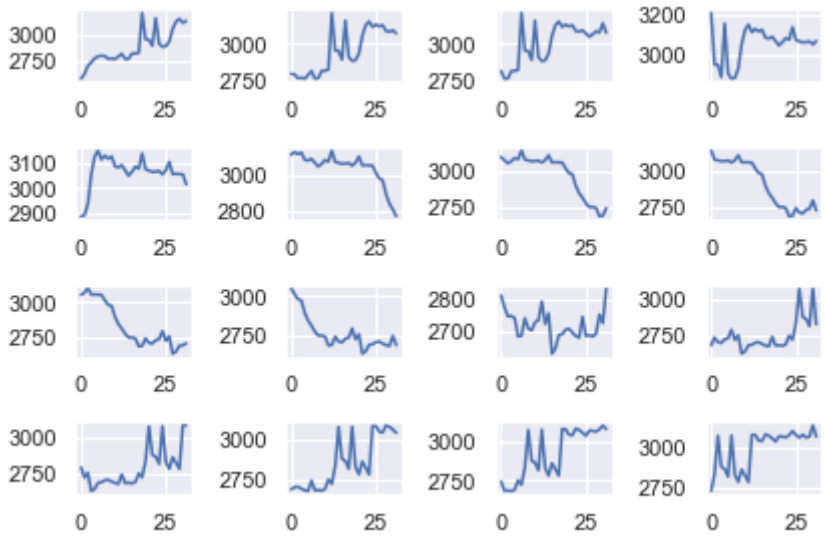
KUVA 23. Esimerkki K-means-klusteroinnista (36)

4.4.2 Menetelmän soveltaminen poikkeavuuksien havaitsemisessa

Aikasarjadatasta poikkeavuuksia voidaan yrittää havaita käyttäen k-means-klusterointia. Data pilkotaan tietynkokoisiin segmentteihin ja segmentit klusteroidaan. Segmentin pituus määrittelee, kuinka monessa ulottuvuudessa pisteiden etäisyyttä klusterinsa keskustaan lasketaan. Jos segmentin pituudeksi määritellään esimerkiksi 5, käsitellään pisteitä viisiulotteisessa avaruudessa. Klustereiden keskipisteitä käytetään ikään kuin rakennuspalikoina, joista aikasarja koostuu. Alkuperäinen aikasarja rakennetaan uudestaan klustereiden keskustojen perusteella ja tätä rekonstruoitua aikasarjaa verrataan alkuperäiseen laskemalla sille

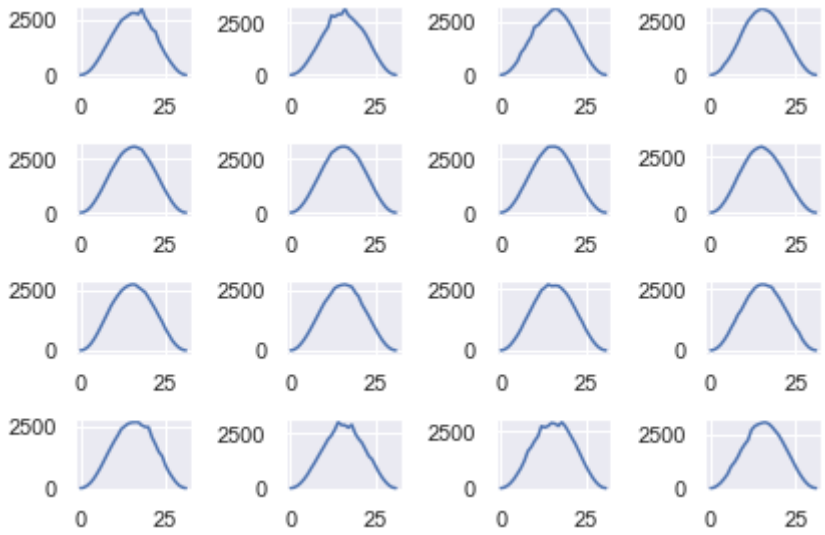
rekonstruointivirhe. Poikkeavien testien oletetaan koostuvan sellaisista aikasarjasegmenteistä, jotka ovat kaukana klustereidensa keskustoista. Niiden rekonstruoitujen versioiden täytyisi siis tuottaa suurempi virhe kuin hyväksyttyjen testien. (37.)

Datan segmentoinnissa määritellään ensin segmentin ja liukuvan ikkunan pituudet. Liukuva ikkuna määrittelee, kuinka suuri hyppy segmenttien aloituskohtien välillä on. Esimerkiksi jos segmentin koko olisi 5 ja liukuvan ikkunan koko 3, otetaan ensin DataFramen indeksistä 0 alkaen 5 näytettä, tämän jälkeen indeksistä 3 alkaen 5 näytettä ja jälleen indeksistä 6 alkaen 5 näytettä jatkaen, kunnes saavutetaan loppu. Mikäli lopusta ei saada segmentin pituista näytettä, sitä ei käytetä. Kuvassa 24 on esitetään pieni otanta aikasarjasta leikattuja 32 pisteen mittaisia segmenttejä.



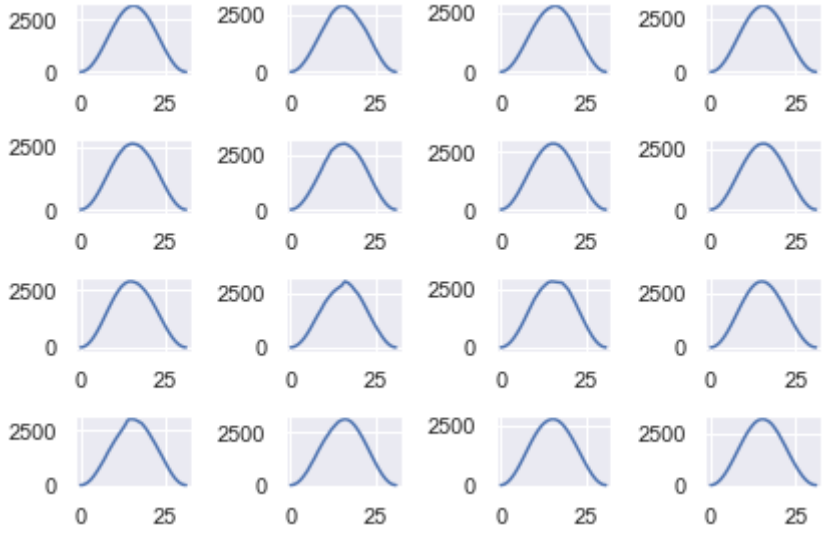
KUVA 24. Aikasarjasta leikattuja 32 pisteen mittaisia segmenttejä

Segmentit eivät ala ja lopu nollasta. Aikasarjadataa rekonstruoidessa tämä voi johtaa aikasarjan katkeilemiseen. Siispä jokainen segmentti pakotetaan alkamaan ja loppumaan nollaan kertomalla ne ikkunafunktiolla. Kuvassa 25 kuvan 23 segmentit on kerrottu ikkunafunktiolla.



KUVA 25. Ikkunafunktiolla kerrotut segmentit

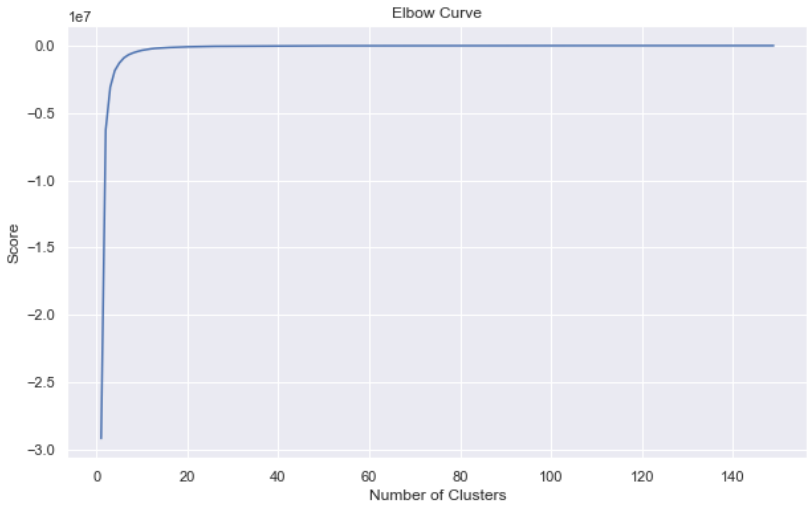
Klusteroija luodaan scikit-learn-moduulin KMeans-moduulia käyttäen antamalla sille parametriksi haluttu klustereiden määrä. Klusteroija sovitetaan ikkunafunktiolla kerrotuilla segmenteillä. Kuvassa 26 on esitetty löydetyt klustereiden keskustat, kun klustereiden määräksi on määritelty 16.



KUVA 26. Klustereiden keskustat

Sopiva klustereiden määrä voidaan määrittää niin sanotulla kyynärpäämetodilla. Metodissa käsiteltävä data sovitetaan eri määrälle klustereita ja tarkastellaan, kuinka paljon klustereiden lisääminen selittää datan varianssia. (38.) Kuvassa 27

tarkastellaan ketjutetun datan kyynärpäämetodin tuloksia. Kohta, jossa käyrä tasaantuu, on kyynärpäämetodin mukaan se piste, jonka jälkeen klustereiden lisääminen ei selitä enää datan varianssia. Käyrän perusteella sopiva klustereiden määrä olisi siis noin 15–20.



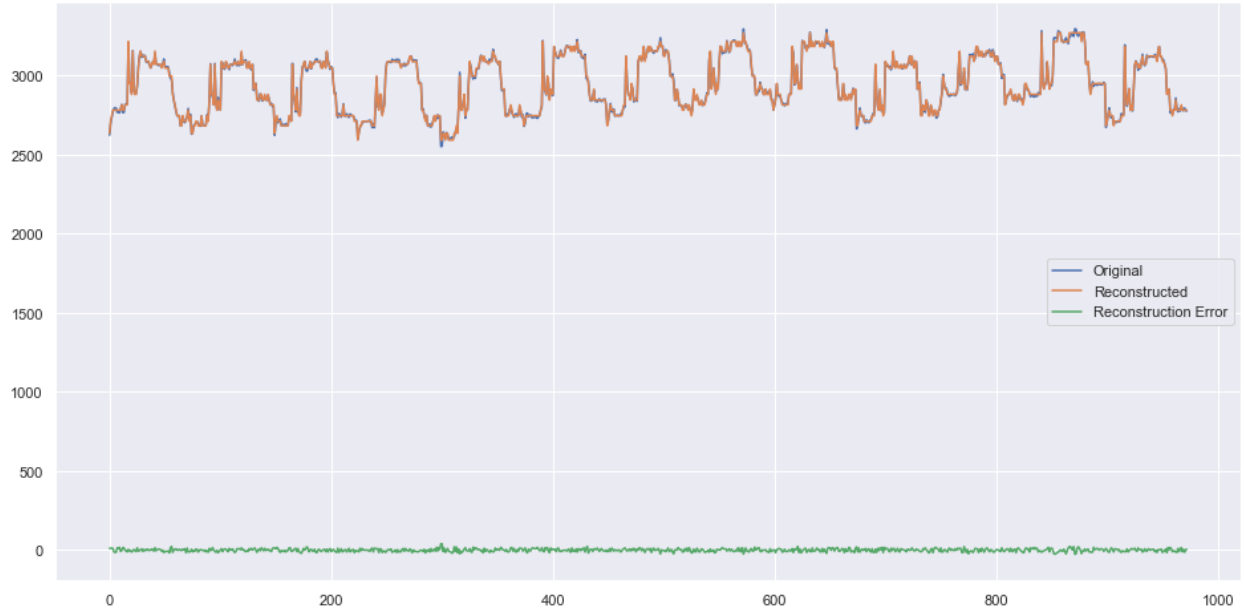
KUVA 27. Ketjutetun datan kyynärpäämetodin tulokset

Sopivaa segmentin pituutta ja liukuvan ikkunan kokoa määriteltäessä käytettiin samaa taulukkoetsintämetodia kuin ARIMA-parametrien määrittämisessä. Menetelmällä pyrittiin saamaan aikaiseksi mahdollisimman pieni rekonstruointivirhe. Parametriyhdistelmiä oli huomattavasti enemmän, kuin ARIMA-mallin parametreja määritettäessä. 12 000 eri yhdistelmällä jokaisen yhdistelmän läpikäymisessä kesti kuitenkin vain 35 minuuttia kun ARIMA-mallin taulukkoetsinnässä 63 yhdistelmällä kesti 3,5 tuntia.

Pienimpään rekonstruointivirheeseen päästiin aina segmentin koolla 3 ja liukuvan ikkunan koolla 2. Klustereiden maksimimäärää vaihdeltiin 40:n, 50:n ja 60:n välillä. Suurin arvo tuotti aina pienimmän rekonstruointivirheen. Tämä on loogista: mitä enemmän erilaisia ryhmiä, joihin segmenttejä voidaan luokitella, sitä helpommin rekonstruoidessa löytyy oikean muotoinen segmentti.

Liian suuri segmenttien määrä voi kuitenkin johtaa ongelmiin virheellistä dataa rekonstruoidessa. Esimerkiksi jos testien keskivaiheilla nähtävä piikki saapuu liian aikaisin tai myöhään, mutta on oikean muotoinen, ei tämä menetelmä osaa ottaa kantaa sen poikkeavuuteen. Lopullisiksi testattaviksi parametreiksi valittiin

segmentin pituudeksi 3, liukuvan ikkunan kooksi 2 ja klustereiden määräksi 20. Kuvassa 28 esitetään ketjutetun datan rekonstruoinnin tulokset edellä mainittuja parametreja käyttäen. Alkuperäinen ja rekonstruoitu käyrä ovat lähes päällekkäin eli rekonstruointi onnistuu lähes täydellisesti. Myös yksittäisten datapisteiden rekonstruointivirhe säilyy tasaisena.



KUVA 28. Ketjutettu data, rekonstruoitu data ja absoluuttinen keskivirhe

Taulukossa 7 on esitelty rekonstruoinnin virhemetriikat. Vertailua näiden arvojen ja ennustamisessa tuotettujen arvojen välillä ei ole mielekästä tehdä, sillä ne mittaavat eri asiaa. Pienistä arvoista voidaan kuitenkin päätellä rekonstruoinnin toimivan tarkasti.

TAULUKKO 7. Rekonstruoinnin virhemetriikat

Virhemetriikka	**Tulos**
MSE	95,27
RMSE	9,76
MAE	8,14

Hyväksyttyjen ja hylättyjen testien vertailussa käytettiin samanlaista perusperiaatetta kuin ARIMA-mallia testatessa. Rekonstruointikirjasto luotiin ensin koko ketjutetun datan pohjalta, jonka jälkeen yksittäisten testien rekonstruointivirhettä vertailtiin. Hyväksytyistä testeistä luotiin virheen pohjataso laskemalla virheiden keskiarvo ja -hajonta. Testi määriteltiin poikkeavaksi, mikäli sen virhe ylitti keskiarvon ja -hajonnan summan. Suuri virhe viittaa siis siihen, että hyväksytyistä testeistä luodun kirjaston perusteella ei pystytty rakentamaan tarkasti suoritinkäytön summakäyrää, joten tarkasteltavan testin täytyy olla jollain tavalla poikkeava. Epäonnistuneiksi määritellyistä testeistä oikein tunnistettiin 58,82 %, vääriä positiivisia puolestaan ei havaittu yhtään Tulokset olivat parempia kuin ARIMA-mallin tulokset, mutta silti kaukana yksinkertaisesta keskiarvotestiin vertailusta.

Aikasarjadatan segmenttien klusteroinnin merkityksellisyydestä on esitetty kritiikkiä. Ollakseen merkityksellinen algoritmin tulisi tuottaa aina samanlaiset tulokset, mikäli sille syötettävä tietoaineisto ei muutu. K-means-klusteroinnin tapauksessa kritiikki kohdistuu sen alussa tehtävään summittaiseen klusterien keskustojen asetteluun, joka johtaa tulosten epäluotettavuuteen. (39.) Menetelmää testatessa havaittiin virhemetriikoiden arvojen vaihtelevan hieman. Tulokset olivat testikertojen välillä samanlaisia, mutta niihin tulisi silti suhtautua varauksella.

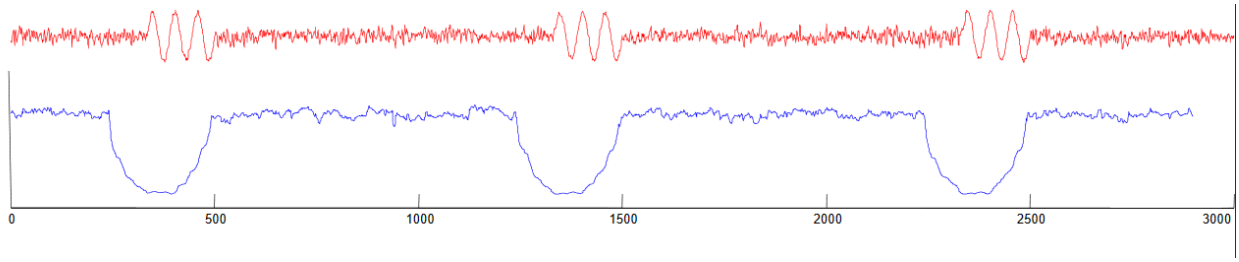
4.5 Matriisiprofiili

Matriisiprofiili on aikasarjadatan perusteella luotu kuvaus ajanhetkellä t alkavan m:n datapisteen euklidisesta etäisyydestä lähimpään naapuriinsa. Sen avulla siis pystytään tarkastelemaan, kuinka paljon aikasarjan muodostamat segmentit muistuttavat toisiaan. Sitä voidaan hyödyntää poikkeamien havaitsemisen lisäksi esimerkiksi klusteroinnissa, luokittelussa ja usein toistuvien ilmiöiden löytämisessä. Se ei ole riippuvainen tutkittavasta datasta eli sitä voidaan soveltaa monenlaiseen sekvenssimäiseen dataan, kuten esimerkiksi ääninäytteiden, genomin tai sydänsähkökäyrän analysointiin. (40.) Menetelmän kehittäjien mukaan sen käyttäminen tekee suurimmasta osasta aikasarjadatan louhintatehtävistä triviaaleja, se voidaan laskea erittäin tehokkaasti ja algoritmit, jotka on rakennettu sen varaan, perivät kaikki sen hyvät ominaisuudet. Matriisiprofiili on verrattain uusi konsepti. Ensimmäiset julkaistut artikkelit ovat vuodelta 2016. Sillä on kui-

tenkin saatu jo erittäin lupaavia, konkreettisia tuloksia esimerkiksi tietoverkkoliikenteen analysoinnissa. (41.) Vaikka matriisiprofiilin tekijät eivät kutsukaan menetelmäänsä suoraan dataprofiloinniksi, se kuitenkin selvästi täyttää dataprofiloinnin määritelmän. Sen avulla käsiteltävästä tietoaineistosta saadaan esille sellaisia datan sisäisiä riippuvuuksia, joita ei pysty pelkkää raakadataa tarkkailemalla huomaamaan.

4.5.1 Matriisiprofiilin luominen

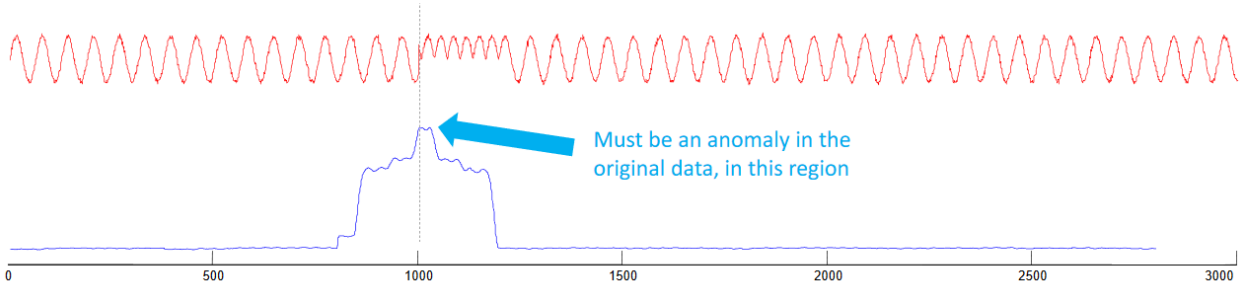
Matriisiprofiilissa data pilkotaan samankokoisiin palasiin ja palasten etäisyyttä toisiinsa verrataan. Ajanhetkellä t matriisiprofiili saa siis arvoksi kyseiseltä hetkeltä alkavan segmentin etäisyyden lähimpään naapuriinsa m-ulotteisessa avaruudessa käyttäen z-normalisoitua euklidista etäisyyttä. Symboli m merkitsee siis tässä tapauksessa myös segmenttien pituutta. Kuvassa 29 on luotu keinotekoisesti tuotetusta raakadatasta matriisiprofiili. Punainen käyrä merkitsee raakadataa ja sininen puolestaan matriisiprofiilia. Käsiteltävän aineiston pituus on 3000 datapistettä ja segmentin pituus 100 datapistettä. Raakadata on pääosin tasaista kohinaa, jolloin myös matriisiprofiilin arvot saavat saman tyyppisiä arvoja. Dataan on lisätty kolme erillistä ja keskenään samankaltaista kohtaa, jotka muistuttavat siniaaltoa. Näiden kohdalla matriisiprofiilin arvot ovat satunnaista kohinaa pienempiä. Segmentit, jotka sisältävät siniaaltoa muistuttavia datapisteitä, ovat siis keskenään hyvin samankaltaisia, joten niiden arvot matriisiprofiilissa ovat pieniä. (40.)



KUVA 29. Keinotekoisesti tuotettu raakadata ja sen pohjalta laskettu matriisiprofiili (40)

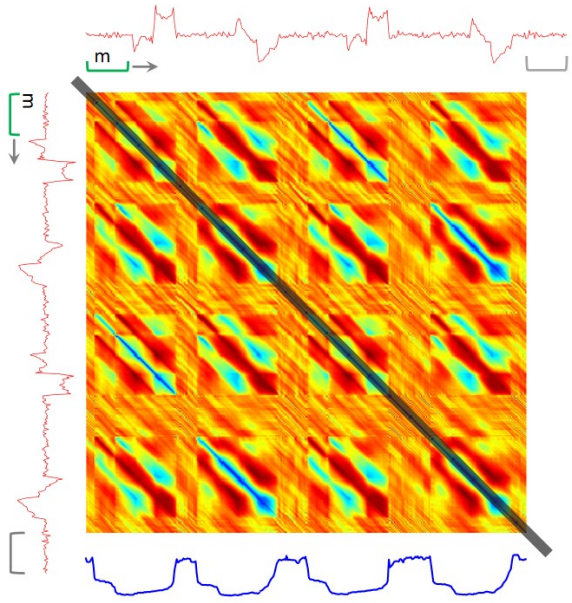
Kuvassa 30 puolestaan on luotu jälleen keinotekoisesti aikasarjadataa, joka suurimmalta osin muistuttaa siniaaltoa. Ajanhetkeltä 1000 siihen on luotu synteetti-

sesti poikkeama. Matriisiprofiilista poikkeama havaitaan suurena arvona. Segmentit, joihin poikkeama sisältyy, eivät muistuta läheisesti muita aikasarjasta pilkottuja segmenttejä, joten niiden etäisyys lähimpään naapuriinsa on suuri. (40.)



KUVA 30. Keinotekoinen aikasarja, johon on luotu poikkeama (40)

Kuvassa 31 on esitetty matriisiprofiilin laskeminen visuaalisesti intuitiivisella tavalla. Jokaista aikasarjasta luotua m:n mittaista segmenttiä verrataan siis keskenään ja niiden etäisyys toisiinsa lasketaan. Värillisessä matriisissa punainen väri merkitsee segmenttien olevan kaukana toisistaan, sininen puolestaan viittaa segmenttien olevan lähellä toisiaan. Harmaalla olevaa aluetta ei oteta huomioon laskuissa, sillä kahden samasta ajanhetkestä luotavan segmentin etäisyys toisistaan on luonnollisesti 0. Pohjalla oleva sininen käyrä merkitsee aikasarjan perusteella luotua matriisiprofiilia. (40.)

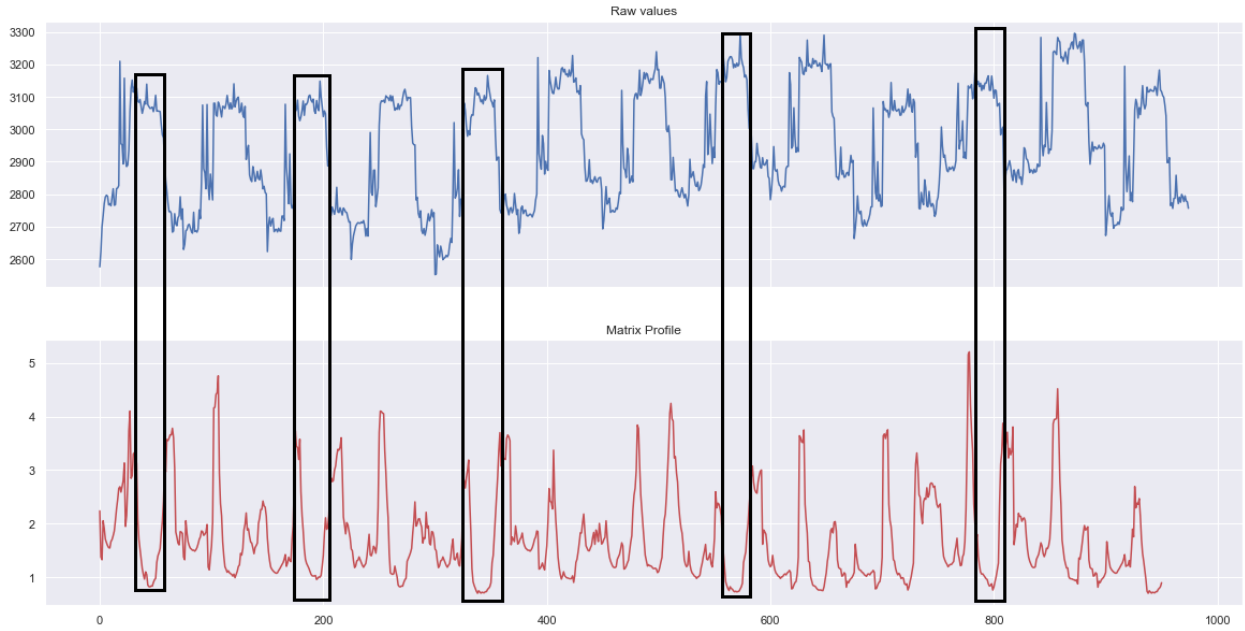


KUVA 31. Matriisiprofiilin laskemisen visualisointi (40)

Matriisiprofiili voitaisiin periaatteessa luoda naiivisti vertaamalla jokaista segmenttiä keskenään. Tätä lähestymistapaa käytettäessä matriisiprofiilin laskemiseen kuluva aika kuitenkin kasvaa sitä pidemmäksi, mitä pidempi käsiteltävä aikasarja on. Sen laskemiseksi onkin kehitetty useita algoritmeja, jotka pienentävät käsittelyaikaa merkittävästi. Merkittävimpiä matriisiprofiilin luomiseksi kehitettyjä algoritmeja ovat STAMP, STOMP, GPU-STOMP ja SCRIMP. (40.)

4.5.2 Matriisiprofiilin soveltaminen poikkeavuuksien havaitsemisessa

Matriisiprofiilin luomiseen käytettiin avoimen lähdekoodin STUMPY-kirjastoa. Kirjasto toteuttaa STUMP-nimisen algoritmin, joka pohjautuu GPU-STOMP-algoritmiin. (16.) Kuvassa 32 on luotu ketjutetun aikasarjadatan pohjalta matriisiprofiili käyttäen segmentin kokona 25 datapistettä. Mitä pienempi segmentin koko on, sitä kohinaisemmalta matriisiprofiili vaikuttaa. Kuvassa on mustareunaisella laatikolla korostettu muutamia kohtia, joissa matriisiprofiili saa pieniä arvoja. Nämä pienet arvot viittaisivat siihen, että juuri tämänkaltaista käyttäytymistä suoritinkäytön summakäyrällä esiintyy paljon.



KUVA 32. Ketjutettu aikasarjadata ja sen pohjalta luotu matriisiprofiili

Myös matriisiprofiilia käytettiin poikkeavuuksien havaitsemisessa noudattaen samanlaista periaatetta kuin ARIMA-mallia ja K-Means-klusterointia testattaessa.

Ketjutetun aikasarjan matriisiprofiilin saamien arvojen keskiarvo ja -hajonta laskettiin, jonka jälkeen yksittäiset testit liitettiin aikasarjan jatkoksi. Mikäli yksittäisen testin matriisiprofiilin keskiarvo oli suurempi kuin koko ketjutetun aikasarjan keskiarvon ja -hajonnan summa, pidettiin testiä poikkeavana. Segmentin pituudella havaittiin olevan vaikutusta testauksen tuloksiin. Taulukossa 8 on dokumentoitu tuloksia segmenttien koolla 5–50 kasvattaen arvoa viidellä testien välillä. Huomion arvoista on, ettei erillisistä todentamisessa käytetyistä hyväksytyistä testeistä poikkeaviksi tunnisteta yhtäkään.

TAULUKKO 8. Poikkeavat testit segmentin pituuden perusteella

Segmentin pituus	**Poikkeaviksi määritellyt hylätyt testit**	**Poikkeaviksi määritellyt hyväksytyt testit**
5	0/17	0/11
10	0/17	0/11
15	2/17	0/11
20	8/17	0/11
25	11/17	0/11
30	11/17	0/11
35	12/17	0/11
40	12/17	0/11
45	12/17	0/11
50	12/17	0/11

4.6 Yhteenveto menetelmien tuloksista

Taulukossa 9 on esitetty yhteenveto kaikkien menetelmien tuloksista. Alipäästösuodatusmenetelmissä on otettu huomioon vain keskiarvotestin summakäyrän (kuva 12, sininen käyrä) alipäästösuodattaminen. Oikeat positiiviset ovat hylätyiksi luokiteltuja testejä, jotka tunnistettiin oikein hylätyiksi. Väärät positiiviset puolestaan ovat hyväksytyiksi luokiteltuja testejä, jotka tunnistettiin virheellisesti hylätyiksi.

TAULUKKO 9. Yhteenveto menetelmien tuloksista

Menetelmä	**Oikeita positiivisia**	**Vääriä positiivisia**
Viuhkaprofiili	88,24 %	0,00 %
Yksinkertainen liukuva keskiarvo	82,35 %	0,00 %
Painotettu liukuva keskiarvo	82,35 %	0,00 %
Eksponentiaalinen tasoitus	88,24 %	0,00 %
ARIMA	23,53 %	27,27 %
LSTM	Ei sovellettu	Ei sovellettu
K-Means	58,82 %	0,00 %
Matriisiprofiili	70,59 %	0,00 %

Ainoastaan eksponentiaalisella tasoituksella päästiin samanlaisiin lukemiin kuin yksinkertaisella viuhkaprofiililla. Parannusta viuhkaprofiiliin verrattuna ei kuitenkaan tapahtunut, joten menetelmän soveltaminen vain lisäisi ylimääräisiä työvaiheita analysointiin ilman merkittävää parannusta. Ennustukseen perustuvat menetelmät eli ARIMA-malli ja LSTM-neuroverkko voisivat toimia paremmin pidempää aikasarjaa tarkasteltaessa. K-Means-klusteroinnin tulokset voisivat parantua suuremmalla tietoaineistolla, mutta menetelmää aikasarjadataan soveltaessa tulee myös huomioida sitä kohtaan esitetty kritiikki. Matriisiprofiilia puolestaan voitaisiin yrittää hyödyntää muussakin kuin pelkässä poikkeavuuksien havaitsemisessa. Se oli myös testatuista edistyneemmistä menetelmistä helpoin toteuttaa

niin datan esikäsittelyn, parametrien valinnan kuin myös laskentaan kuluvan ajan osalta.

5 POHDINTA

Opinnäytetyön tehtävänä oli perehtyä dataprofilointiin ja poikkeavuuksien havaitsemisessa käytettyihin menetelmiin, tutkia menetelmien käyttökelpoisuutta ja arvioida niiden käytettävyyttä myös muiden metriikoiden analysoinnissa. Useiden hyväksyttyjen testien pohjalta luotiin keskiarvotesti, johon muita testejä verrattiin. Lisäksi edistyneemmistä menetelmistä testattiin alipäästösuodatukseen perustuvia menetelmiä, ARIMA-mallia, LSTM-neuroverkkoa, K-Means-klusterointia ja matriisiprofiilin hyödyntämistä.

Yksinkertainen keskiarvotestiin vertailu osoittautui menetelmistä tehokkaimmaksi niin tulostensa kuin helpon implementoinnin puolesta. Menetelmää voisi suoraan hyödyntää esimerkiksi keskiarvoprofiilin luomiseen tukiaseman muistinkäytön tutkimisessa. Menetelmän haittana on kuitenkin sen tukiasema- ja testikohtaisuus. Yhtä luotua profiilia ei voida suoraan hyödyntää tukiasemien, joiden pistoyksikkökonfiguraatiot ovat erilaisia, eikä erilaisten testiolosuhteiden välillä. Tukiasemaohjelmiston uudet ominaisuudet voivat myös muuttaa normaalia profiilia erilaiseksi, jolloin vaaditaan riittävä määrä mahdollisimman tarkasti onnistuneeksi luokiteltuja testejä uuden profiilin luomiseen. Tulevaisuudessa keskiarvotestin ja tutkittavan testien välistä ristikorrelaatiota voitaisiin myös yrittää hyödyntää tulosten parantamiseksi.

Edistyneemmillä menetelmillä ei saavutettu kovin hyviä tuloksia osittain niiden parametrien optimoinnin vaikeuden vuoksi. Penkkitestauksen lyhyt kesto voi myös vaikuttaa erityisesti ARIMA-mallin ja LSTM:n käytettävyyteen. Nämä menetelmät voisivat olla hyödyllisempiä esimerkiksi useamman päivän kestävien tukiaseman stabiilisuustestauksen yhteydessä.

Vaikka matriisiprofiilin käyttö poikkeavuuksien havaitsemisessa ei tuottanutkaan aivan yhtä hyviä tuloksia kuin yksinkertainen keskiarvotestiin vertailu, sen hyödyllisyyttä dataprofiloinnin kannalta kannattaa tutkia tarkemmin. Menetelmä on perusperiaatteeltaan yksinkertainen, mutta sen avulla käsiteltävästä datasta voidaan saada esille paljon tietoa, jota ei suoraan voida nähdä. Sillä voitaisiin esimerkiksi tutkia, kuinka tarkasti erilaisten tutkittavien metriikoiden käyttäytyminen tukiasemaohjelmiston eri koontiversioiden välillä muistuttaa toisiaan.

Opinnäytetyön avulla saatiin paljon pohjustavaa tietoa erilaisten poikkeavuuksien havaitsemisessa käytettyjen menetelmien käyttökelpoisuudesta. Tuloksia voidaan hyödyntää Nokian sisäisten lokianalyysijärjestelmien ja -käytäntöjen jatkojalostamisessa ja kehittämisessä. Työn konkreettinen lopputulos on yli 90-sivuinen Jupyter Notebook -dokumentti. Dokumenttiin on koottu kaavioiden, selitysten ja hyödyllisten linkkien lisäksi toimivaksi havaittuja koodiesimerkkejä, joita voidaan yrittää integroida nykyisiin ja tuleviin lokien analysoinnissa käytettäviin järjestelmiin.

LÄHTEET

1. Loshin, David 2012. Business Intelligence: The Savvy Manager's Guide. Burlington: Morgan Kaufmann.

2. What Is Data Profiling? Process, Best Practices and Tools. Panoply.io. Saatavissa: https://panoply.io/analytics-stack-guide/data-profiling-best-practices/. Hakupäivä 28.6.2019.

3. Lewis, Sarah – Rouse, Margaret 2019. Data profling. TechTarget. Saatavissa: https://searchdatamanagement.techtarget.com/definition/data-profiling. Hakupäivä 2.7.2019.

4. Flovik, Vegard 2018. How to use machine learning for anomaly detection and condition monitoring. Towards Data Science. Saatavissa: https://towardsdatascience.com/how-to-use-machine-learning-for-anomaly-detection-and-condition-monitoring-6742f82900d7. Hakupäivä 9.7.2019.

5. Chandola, Varun – Banerjee, Arindam – Kumar, Vipin 2009. Anomaly Detection: A Survey. ACM Computing Surveys vol 41, nro 3, artikkeli nro 15. Saatavissa: http://cucis.ece.northwestern.edu/projects/DMS/publications/AnomalyDetection.pdf. Hakupäivä 9.7.2019.

6. Tuononen, Marko 2005. Klusterointimenetelmät. Joensuun yliopisto, tietojenkäsittelytiede. Laudaturseminaariesityksen kirjallinen tukimateriaali. Saatavissa: http://cs.joensuu.fi/~mtuonon/Klusterointimenetelmat.pdf. Hakupäivä 16.7.2019.

7. NIST/SEMATECH e-Handbook of Statistical Methods. 2013. 6.4.1 Definitions, Applications and Techniques. USA: National Institute of Standards and Technology. Saatavissa: https://www.itl.nist.gov/div898/handbook/pmc/section4/pmc41.htm. Hakupäivä 24.7.2019.

8. Peixeiro, Marco 2019. Almost Everything You Need to Know About Time Series. Towards Data Science. Saatavissa: https://towardsdatascience.com/almost-everything-you-need-to-know-about-time-series-860241bdc578. Hakupäivä 24.7.2019.

9. Klein, Helge 2014. What is Splunk and How Does it Work? Helge Klein. Saatavissa: https://helgeklein.com/blog/2014/09/splunk-work/. Hakupäivä 29.7.2019.

10. Jupyter/IPython Notebook Quick Start Guide. 2015. Read the Docs. Saatavissa: https://jupyter-notebook-beginner-guide.readthedocs.io/en/latest/. Hakupäivä 29.7.2019.

11. Pandas. Saatavissa: https://pandas.pydata.org/. Hakupäivä 30.7.2019.

12. Hunter, J.D. 2007. Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering vol. 9, nro 3, sivut 90–95. Saatavissa: https://ieeexplore.ieee.org/document/4160265?ALU=LU1049036. Hakupäivä 30.7.2019.

13. Scikit-learn. Saatavissa: https://scikit-learn.org/stable/. Hakupäivä 30.7.2019.

14. Keras. Saatavissa: https://keras.io/. Hakupäivä 30.7.2019.

15. Statsmodels. Saatavissa: https://www.statsmodels.org/stable/index.html. Hakupäivä 30.7.2019.

16. Law, Sean M. 2019. STUMPY: A Powerful and Scalable Python Library for Time Series Data Mining. Journal of Open Source Software vol 4, nro 39. Saatavissa: https://github.com/TDAmeritrade/stumpy. Hakupäivä 30.7.2019.

17. Capka, David 2019. Lesson 11 - Multidimensional arrays in C# .NET. ICT.social. Saatavissa: https://www.ict.social/csharp/basics/multidimensional-arrays-in-csharp-net. Hakupäivä 30.7.2019.

18. Hyndman, Ray J. – Athanasopoulos, George 2018. Forecasting: principles and practice, 2. painos. Melbourne: OTexts. Saatavilla: https://otexts.com/fpp2/. Hakupäivä 31.7.2019.

19. Palachy, Shay 2019. Stationarity in time series analysis. Towards Data Science. Saatavissa: https://towardsdatascience.com/stationarity-in-time-series-analysis-90c94f27322. Hakupäivä: 31.7.2019.

20. Analytics Vidhya Content Team. 2016. A comprehensive beginner's guide to create a Time Series Forecast (with Codes in Python and R). Analytics Vidhya. Saatavilla: https://www.analyticsvidhya.com/blog/2016/02/time-series-forecasting-codes-python/. Hakupäivä 31.7.2019.

21. Brownlee, Jason. 2016. How to Make Baseline Predictions for Time Series Forecasting with Python. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/persistence-time-series-forecasting-with-python/. Hakupäivä 31.7.2019.

22. Drakos, Georgios. 2018. How to select the Right Evaluation Metric for Machine Learning Models: Part 1 Regression Metrics. Towards Data Science. Saatavissa: https://towardsdatascience.com/how-to-select-the-right-evaluation-metric-for-machine-learning-models-part-1-regrression-metrics-3606e25beae0 . Hakupäivä 3.9.2019.

23. Choudhary, Pramit. 2017. Introduction to Anomaly Detection. Oracle. Saatavissa: https://www.datascience.com/blog/python-anomaly-detection. Hakupäivä 2.8.2019.

24. Banton, Caroline. 2019. Moving Average, Weighted Moving Average, and Exponential Moving Average. Investopedia. Saatavilla: https://www.investopedia.com/ask/answers/071414/whats-difference-between-moving-average-and-weighted-moving-average.asp.Hakupäivä 2.8.2019.

25. Sergeev, Dmitriy. 2018. Open Machine Learning Course. Topic 9. Part 1. Time series analysis in Python. Medium. Saatavissa: https://medium.com/open-machine-learning-course/open-machine-learning-course-topic-9-time-series-analysis-in-python-a270cb05e0b3. Hakupäivä 2.8.2019.

26. Brownlee, Jason. 2017. How to Create an ARIMA Model for Time Series Forecasting in Python. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. Hakupäivä 2.8.2019.

27. Nau, Robert. Introduction to ARIMA: nonseasonal models. Fuqua School of Business. Saatavissa: http://people.duke.edu/~rnau/411arim.htm. Hakupäivä 2.8.2019.

28. Malik, Farhad. 2018. Understanding Auto Regressive Moving Average Model — ARIMA. Medium. Saatavissa: https://medium.com/fintechexplained/understanding-auto-regressive-model-arima-4bd463b7a1bb . Hakupäivä 2.8.2019.

29. Brownlee, Jason. 2017. A Gentle Introduction to the Box-Jenkins Method for Time Series Forecasting. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/gentle-introduction-box-jenkins-method-time-series-forecasting/. Hakupäivä 2.8.2019.

30. Brownlee, Jason. 2017. How to Grid Search ARIMA Model Hyperparameters with Python. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/grid-search-arima-hyperparameters-with-python/. Hakupäivä 2.8.2019.

31. Brownlee, Jason. 2017. How to Create an ARIMA Model for Time Series Forecasting in Python. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/arima-for-time-series-forecasting-with-python/. Hakupäivä 5.8.2019.

32. Nicholson, Chris. 2019. A Beginner's Guide to LSTMs and Recurrent Neural Networks. Skymind. Saatavissa: https://skymind.ai/wiki/lstm. Hakupäivä 6.8.2019.

33. Olah, Christopher. 2015. Understanding LSTM Networks. Colah’s blog. Saatavissa: https://colah.github.io/posts/2015-08-Understanding-LSTMs/. Hakupäivä 6.8.2019.

34. Brownlee, Jason. 2017. Time Series Forecasting with the Long Short-Term Memory Network in Python. Machine Learning Mastery. Saatavissa: https://machinelearningmastery.com/time-series-forecasting-long-short-term-memory-network-python/. Hakupäivä 7.8.2019.

35. Dabbura, Imad. 2018. K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks. Towards Data Science. Saatavissa: https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a. Hakupäivä 7.8.2019.

36. Lavrenko, Victor. 2013. K-means clustering: how it works. Video. Edinburghin yliopisto. Saatavissa: https://www.youtube.com/watch?v=_aWzGGNrcic. Hakupäivä 7.8.2019.

37. Rahtz, Matthew. 2015. Anomaly Detection with K-Means Clustering. Amid Fish. Saatavissa: http://amid.fish/anomaly-detection-with-k-means-clustering. Hakupäivä 7.8.2019.

38. Li, Susan. 2019. Time Series of Price Anomaly Detection. Towards Data Science. Saatavissa: https://towardsdatascience.com/time-series-of-price-anomaly-detection-13586cd5ff46. Hakupäivä 20.8.2019.

39. Keogh, Eamonn – Lin, Jessica. 2005. Clustering of time-series subsequences is meaningless: implications for previous and future research. Knowledge and Information Systems, vol 8, nro 2, sivut 154–177. Saatavissa: https://www.cs.ucr.edu/~eamonn/meaningless.pdf. Hakupäivä 8.8.2019.

40. Keogh, Eamonn – Mueen, Abdullah. 2017. Time Series Data Mining Using the Matrix Profile: A Unifying View of Motif Discovery, Anomaly Detection, Segmentation, Classification, Clustering and Similarity Joins Part 1. KDD2017-konferenssissa pidetyn luennon diat. Saatavissa: http://www.cs.ucr.edu/~eamonn/Matrix_Profile_Tutorial_Part1.pdf. Hakupäivä 9.8.2019.

41. Keogh, Eamonn. 2019. The UCR Matrix Profile Page. University of California – Riverside. Saatavissa: https://www.cs.ucr.edu/~eamonn/MatrixProfile.html. Hakupäivä 9.8.2019.

42. Mueen, Abdullah – Keogh, Eamonn. 2017. Time Series Data Mining Using the Matrix Profile: A Unifying View of Motif Discovery, Anomaly Detection, Segmentation, Classification, Clustering and Similarity Joins Part 2. KDD2017-konferenssissa pidetyn luennon diat. Saatavissa: https://www.cs.ucr.edu/~eamonn/Matrix_Profile_Tutorial_Part2.pdf. Haku-päivä 9.8.2019.