jamk

Koneenpiirustusprosessin tehostaminen Catia-ympäristössä koneoppimistyökalun avulla

Jesse Hiltunen

Opinnäytetyö, AMK
Toukokuu 2024
Konetekniikan tutkinto-ohjelma

jamk | Jyväskylän ammattikorkeakoulu
University of Applied Sciences

jamk

Kuvailulehti

Hiltunen, Jesse

Koneenpiirustusprosessin tehostaminen Catia-ympäristössä koneoppimistyökalun avulla.

Jyväskylä: Jyväskylän ammattikorkeakoulu. Toukokuu 2024, 91 sivua.

Konetekniikan tutkinto-ohjelma. Opinnäytetyö AMK.

Julkaisun kieli: suomi

Julkaisulupa avoimessa verkossa: kyllä

Tiivistelmä

Koneensuunnittelun olennainen osa on piirustusten teko, joiden avulla erilaiset työkappaleet ja kokoonpanot voidaan valmistaa. Projektit aloitetaan usein tarkistamalla, kuinka vastaavia on aiemmin tehty. Tällaisessa tilanteessa piirustuksien osalta kappaleiden mitoitus ja toleranssien määrittely vievät paljon aikaa, sillä suunnittelijan tulee ensin selvittää vanhan projektin tiedot, etsiä oikeat piirustukset sekä referoida niitä luodakseen uutta. Parhaimmillaan vanhojen tietojen etsintä ja avaaminen voi viedä tunteja suunnittelijan aikaa työpäivän aikana. Prosessin tehostaminen voi tuoda niin säästöjä kuin lisätä suunnittelijoiden tehokkuutta.

Työn tavoite on saada toimeksiantajayrityksen mekaniikkasuunnittelijoiden käyttöön koneoppimista hyödyntävä työkalu, mikä ehdottaa suunnittelijoille vanhojen piirustusten pohjalta käyttäjän Catiassa valitsemaan pituus- tai halkaisijamittaan kolme todennäköisintä toleranssia. Työ toteutetaan kehitystutkimuksena. Työn aikana selvitetään, millainen työmäärä on koneoppimismallin kehityksessä sekä sen integroinnissa tuotantoympäristöön.

Valmiin työkalun tarkkuus kolmen ensimmäisen ennustuksen osalta on 93,4 %, kun se koulutettiin ja testattiin neljän eri yhteistyötilan aineistolla. Koneoppimismallin avulla luodun työkalun avulla on mahdollista tehostaa tuotantoa vähentämällä vanhojen kuvien tarkasteluun käytettyä aikaa sekä se vähentää inhimillisten virheiden mahdollisuutta. Työkalun tulokset näyttävät alustavasti lupaavilta, joten tulevaisuudessa koneoppimiselle löytyy varmasti lisää hyödyllisiä käyttökohteita.

Lopulta on kuitenkin pidettävä mielessä, että matemaattiset mallit pohjautuvat aina menneisyyteen sekä oletukseen, että siellä on toistuva kaava taustalla. Työtä tehdessä koneoppimismalleilla ei ole ihmisten lailla kykyä kehittää täysin uutta, joten opetetun materiaalin ulkopuolelta se ei kykene tekemään uutta. Vaikka työkalu olisi testiympäristössä täysin tarkka, voi reaalimaailmassa tulla vastaan muutoksia, joihin se ei kykene mukautumaan. Suunnittelija on loppujen lopuksi vastuussa päätöksistä, ja työkalun on vain tarkoitus tehostaa tämän työntekoa vähentämällä toistuvien työtehtävien määrää.

Avainsanat (asiasanat)

Koneoppiminen, Catia V6, 3DExperience, koneensuunnittelu, tekoäly

Muut tiedot (salassa pidettävät liitteet)

Liite 4 on salassa pidettävä, ja se on poistettu julkisesta työstä. Salassapidon perusteena on viranomaisten toiminnan julkisuudesta annetun lain (621/1999) 24§:n kohta 21, yrityksen liike- tai ammattisalaisuus. Salassapitoaika on viisi (5) vuotta, salassapito päättyy 6/2029.

jamk

Description

Hiltunen, Jesse

Streamlining the machine drawing process in a Catia environment with a machine learning tool

Jyväskylä: JAMK University of Applied Sciences, May 2024, 91 pages.

Degree Programme in Mechanical Engineering. Bachelor´s thesis.

Permission for open access publication: Yes

Language of publication: Finnish

Abstract

An essential part of machine design is creating the drawings, which are used to produce various workpieces and assemblies. Projects often start by checking how similar projects have been done before. In such a situation, the designer must spend a lot of time on the drawings to dimension the parts and define the tolerances, as they must first find the information from the old project, locate the correct drawings and reference them to create a new one. At best, searching and opening old data can take hours of the designer's time during the working day. Streamlining the process can both save money and increase the efficiency of designers.

The goal of this work is to provide the mechanical designers of the client company with a machine learning tool that based on old drawings, suggests for the user three most likely tolerances for the selected length or diameter dimension in Catia. The work will be carried out as a development study. The thesis will determine the workload involved in developing a machine learning model and integrating it into a production environment.

The accuracy of the final tool for the first three predictions is 93.4% when trained and tested with data from four different collaboration spaces. The machine learning tool makes improving production efficiency possible by reducing the time spent on reviewing old drawings and by reducing the possibility of human errors. The initial results of the tool look promising, so there are bound to be more useful applications for machine learning in the future.

Ultimately, however, it must be kept in mind that mathematical models are always based on the past and the assumption that there is a recurring pattern in the background. Machine learning models do not have the ability to develop something completely new like humans, so they cannot create something outside the material they have learned. Even if a tool is perfectly accurate in a testing environment, it may encounter changes in the real world to which it cannot adapt. In the end, the designer is responsible for the decisions, and the tool is only intended to make their work more efficient by reducing the number of repetitive tasks.

Keywords/tags (subjects)

Machine Learning, Catia V6, 3DExperience, machine design, artificial intelligence

Miscellaneous (Confidential information)

Attachment 4 is confidential and has been removed from the public thesis. Secrecy is based on part 21 of section 24 of the law of Publicity. The confidentiality period is five (5) years, and it ends on 6/2029.

Sisältö

Kuviot

Taulukot

Opinnäytetyön termistö

Algoritmi	Ohjelmoinnissa tietokoneelle annetaan toteutettavaksi käskyjä. Näitä käskyjä yhdistelemällä saa rakennettua algoritmin.
Alisovitus	Koneoppimismalli ei löydä tietoaineistosta tarpeeksi yhteyksiä, jonka myötä malli ei pysty tekemään tarkkoja ennustuksia mihinkään dataan.
Arkkitehtuurisuunnittelu	Ohjelmiston kokonaisuuden suunnittelu. Näihin lukeutuu käytettävän ohjelmointikielen, tietokannan ja ohjelmointiympäristön valinta sekä suunnitelma kuinka lopullinen tuote saadaan tuotantoon.
Avoin lähdekoodi	Ohjelmisto tai alusta, minkä lähdekoodiin kaikilla on vapaa pääsy. Näiden ympärillä on usein oma yhteisö, joka kehittää ohjelmistoa eteenpäin.
Catia	Ranskalaisen ohjelmistoyrityksen Dassault Systèmesin valmistama CAD-suunnitteluohjelmisto. Nykyään integroitu 3DExperiencen kylkeen.
Data-analytiikka	Suuren tietomäärän analysointia, minkä tavoite on saada ymmärrystä aiheesta. Tähän yleensä kuuluu tietoaineiston visualisointi ihmisille luettavaan muotoon, trendien muodostaminen sekä syy-seuraussuhteiden löytäminen.
ECT	Toimeksiantajayrityksen valmistama ohjelmistotyökalu, joka toimii Catian päällä. Työkalu on tehty sujuvoittamaan suunnittelijoiden työtä.
Funktio	Tietotekniikassa käskysarja tai aliohjelma, joka palauttaa arvon. Matematiikassa kahden olion tai suureen riippuvuussuhde toisiinsa. Funktio palauttaa aina saman arvon, mikäli muuttujan arvo pysyy samana.
Kirjasto	Ohjelmoinnissa kokoelma valmiita aliohjelmia, mitä voi käyttää hyödyksi oman ohjelman rakentamisessa, jottei kaikkea tarvitse rakentaa alusta.
Koneenpiirustus	Koneensuunnittelussa yksiselitteinen piirustus tuotteesta tai kappaleesta. Voi olla esimerkiksi kappaleen valmistusta varten oleva piirustus, missä on kappaleen mitat ja toleranssit, joiden lisäksi voi olla erillisiä merkintöjä.
Koneoppiminen	Koneoppiminen on tekoälyn osa-alue. Koneoppimismallit saadaan löytämään korrelaatioita suurista tietojoukoista algoritmien avulla. Ehtolauseiden ja silmukoiden sijaan se muodostaa tilastollisia malleja koulutusaineistosta.
ML.NET	Microsoftin ylläpitämä avoimen lähdekoodin koneoppimisalusta. Alustaa voi laajentaa muilla koneoppimiskehyksillä, kuten TensorFlow, PyTorch, ONNX tai Infer.NET. ML.NET:iä ohjelmoidaan C# kielellä.
Neuroverkko	Laskentamalli, joka ottaa mallia aivojen rakenteesta. Käytetään esimerkiksi kuvan- ja objektintunnistusta vaativissa tehtävissä.
Nimellismitta	Virheettömän elementin mitta piirustuksessa.
Ohjelma	Ohjelma on tietokoneelle syötetty loogisten käskyjen kokonaisuus, mitä se suorittaa.

Ohjelmisto	Ohjelmien, tietoaineiston sekä eri kirjastojen yhdistelmä.
Ohjelmistokehys	Englanniksi software framework. Ohjelmiston runko tai alusta, mistä ohjelmoija voi omien laajennusten avulla rakentaa toimivan ohjelmiston.
Ohjelmointirajapinta	Application Programming Interface, eli lyhyemmin API. Rajapinta, jonka avulla ohjelmat pystyvät keskustelemaan keskenään. Rajapinnan avulla toiset ohjelmat pääsevät käsiksi ohjelmiston palveluihin sekä voivat syöttää sille parametrejä.
Ohjelmointiympäristö	IDE, Integrated Development Environment. Ohjelma, joka sisältää työkaluja ohjelmoinnin helpottamiseen. Voi sisältää esimerkiksi versionhallintaan, dokumentaatioon sekä virheiden korjaamiseen liittyviä työkaluja.
Ominaisuus	Koneoppimismallin koulutuksessa käytetty syöte, jota malli käyttää sille määritellyn tehtävän suorittamiseksi. Esimerkiksi omakotitalon ominaisuuksia voisi olla sijainti, väri, koko, kerrosten määrä ja hinta. Suomenkielisessä aineistossa käytetään myös termejä piirre, muuttuja tai attribuutti.
Palvelu	Kokonaisuus, mihin sisältyy ohjelmisto, laitteisto, niiden ylläpito sekä mahdolliset tukipalvelut. Ohjelmistot voivat tarjota palveluita myös toisilleen.
Parametri	Tieto, joka välitetään aliohjelmalle. Esimerkiksi ohjelmalle, mikä tekee kahden luvun yhteenlaskua, annetaan parametreinä yhteenlaskettavat luvut.
PLM	Catian uusimman V6 version käyttämä palvelin, jota ohjelma käyttää tiedon hakuun ja tallennukseen. PLM sisältää muutakin, mutta Dassaultin 3DExperiencen automaatio-ohjeessa
Päätöspuu	Koneoppimisalgoritmi, joka on omiaan luokittelutehtäviin. Toimii hyvin, kun data on taulukkomuodossa.
Ristiinvalidointi	Koneoppimismallia kouluttaessa tietoaineisto jaetaan pienempiin osiin, minkä avulla koulutetaan omat mallit ja tämän jälkeen näiden suorituskykyä verrataan keskenään. Jokainen jaettu joukko toimii vuorollaan validointiaineistona koulutusten aikana.
Sovellus	Käyttäjää jollain tavalla hyödyttävä ohjelmisto. Tietokoneen käyttöjärjestelmä voi olla ohjelmisto, minkä avulla voidaan ajaa sovelluksia, kuten nettiselain.
Sovite	Kahden kappaleen yhteensovittaminen käyttäen tietynlaista välystä taikka ahdistusta. Voidaan määritellä kokemuspohjaisesti taikka laskennallisesti mikäli toiminnallisuus niin vaatii.
Suoritusketju	Englannin kielen pipeline ilmaisusta vapaa suomennos. Tarkoittaa kaikkia toimenpiteitä, jotka koneoppimismallin kehityksessä tehdään.
Syntaksi	Ohjelmointikielen sanasto ja kielioppi. Määrittelee, miten ohjelmointikieltä koodataan.
Tietoaineisto	Aineistojoukko vaaka- ja pystyrivejä. Vaakariveillä on yleensä yksittäiset tietueet ja pystyriveiltä löytyy riviin liittyvät ominaisuudet.
Tietue	Yksi vaakarivi ominaisuuksia tietoaineistossa.

Toleranssi	Sallitut rajat tietylle tuotteen ominaisuudelle. Mittojen kohdalla on yleensä ylä- ja alaeromitat, mutta toleransseja on myös muunlaisia.
Tunniste	Koneoppimismallille syötetty luokka, joka on mallille annettujen ominaisuuksien summa. Esimerkkinä malli, joka luokittelee vaatteita: Tunnisteina voi olla paita, villapaita ja housut. Ominaisuuksia voivat olla muoto, onko hihoja, hihojen mitta ja materiaali. Näiden avulla malli luokittelee mikä tunniste on kyseessä.
Yhteistyötila	Catian Collaborative Space. 3DEXPERIENCE alustan pilvipohjainen osa, jossa voi hallita projekteja sekä dataa.
Ylisovitus	Koulutettu malli oppii koulutusdatan liian tarkasti, eikä osaa tehdä ennustuksia sen ulkopuolelle.

1 Johdanto

1.1 Lähtötilanne

Muutaman viimeisen vuoden aikana koneoppiminen ja tekoäly ovat nousseet termeinä suuren yleisön tietouteen. Tekoäly on tunnettu terminä jo 1950-luvulta lähtien, mutta vasta viime vuosina tietokoneiden laskentateho on päässyt sille asteelle, että koneoppimismallien kouluttamisessa on saatu huomattavaa kehitystä. (Roser, 2022.) Tunnetuin edistysaskel on varmasti OpenAI:n marraskuussa 2022 julkaisema laaja kielimalli ChatGPT. Moni yritys on saanut tämän jälkeen innostuksen panostaa erilaisiin koneoppimissovelluksiin, ja markkinoille onkin tullut runsaasti erilaisia malleja hyödyntäviä työkaluja.

Koneoppiminen on tekoälyn osa-alue. Nykyiset kone- tai syväoppimisen avulla koulutetut mallit ovat usein luotu yhteen työtehtävään, jossa ne loistavat. Mutta niiden kääntöpuolena on, etteivät ne kykene tekemään tehtäviä niille koulutetun alueen ulkopuolelta. Siinä missä tekoäly koettaa jäljitellä ihmisen käytöstä, koneoppimismallit erikoistuvat ennalta määritettyihin tehtäviin hyvällä suorituskyvyllä. (Artificial Intelligence (AI) vs. Machine Learning, n.d.)

Erilaiset koneoppimissovellukset ja niiden suorituskyky herättivät mielenkiinnon opetella aihetta itse. Toimeksiantajayrityksessä pidetyn pienen aivoriihen jälkeen syntyi idea testata, millainen työ on kouluttaa koneoppimisen avulla yrityksen käyttämään Catia-lisäosaan työkalu, joka auttaa suunnittelijaa piirustuksen mittoihin liittyvien toleranssien kanssa. Tämän avulla voisi olla mahdollista vähentää virheiden mahdollisuutta piirustuksien kanssa sekä parantaa tuottavuutta sujuvoittamalla mekaniikkasuunnittelijan työtä.

Toimeksiantajayrityksenä työlle toimii suomalainen insinööritoimisto, jonka asiakkaina toimii suuria koneita ja laitteita valmistavia yrityksiä. Työkalun on tarkoitus tulla lisäosaksi yrityksellä jo käytössä olevan Catia-ympäristössä toimivaan, työntekoa tehostavaan ECT ohjelmistoon, mikäli sen suorituskyky mahdollistaa suunnitteluprosessin tehostamisen.

1.2 Tehtävä

Työn tavoite on selvittää koneoppimisella varustetun työkalun mahdollisuuksia, jotka tehostavat mekaniikkasuunnittelijoiden työntekoa. Toimeksiantajayrityksen asiakkaille valmistettavat projektit sisältävät runsaasti tuotteita, mihin tehdään kokonaisuuteen nähden maltillisesti projektikohtaisia muutoksia. Tällöin projektit noudattavat pitkälti samaa kaavaa, ja kuten Palo (2023, 10) omassa opinnäytetyössään mainitsee, on uusien mallien revisiointi helpointa aloittaa vanhoja malleja referoimalla. Koska piirustusten laadinta alkaa usein aiempien töiden pohjalta, koko suunnitteluprosessia ei tarvitse aloittaa tyhjästä. Jokaiselle asiakkaalle tehdään kuitenkin omat piirustukset tuotteista, jolloin suunnittelija aloittaa työnsä etsimällä vanhoja piirustuksia taikka malleja tietokannasta, joista voi ottaa mallia uuteen piirustukseen.

Piirustusten ja vanhojen projektien etsiminen sekä suurien mallien avaaminen Catian PLM:ssä voi kestää jopa tunteja. Suunnittelijan aika maksaa työnantajalle kymmeniä euroja jokaista odotteluun mennyttä työtuntia kohti, ja tämä luku kertaantuu jokaisen suunnittelijan kohdalla. Mikäli työkalun avulla voi vähentää tätä vanhan aineiston penkomista, tuo se pitkällä ajanjaksolla huomattavia säästöjä. Mikäli aihetta tarkastelee pienemmällä kunnianhimolla, työkalu vähintäänkin nopeuttaa yksittäisten toleranssien asettamista mittoihin ja pienentää inhimillisten virheiden mahdollisuutta, joka voi virtaviivaistaa esimerkiksi piirustusten tarkastusprosessia.

Koulutetuissa koneoppimismalleissa on potentiaalia tehostaa työntekoa huomattavasti monenlaisissa tapauksissa. Koska tämä on ensimmäinen koneoppimismalli, mikä toimeksiantajayrityksessä on tarkoitus lisätä ECT:n toiminnallisuuksiin, työ rajattiin suhteellisen yksinkertaiseksi, jottei työ kaadu ratkaisun monimutkaisuuteen.

Tavoitteena on, että kouluttamalla malli suurella määrällä vanhoja piirustuksia sen on mahdollista ehdottaa vanhojen piirustusten pohjalta mittoihin sopivia toleransseja ilman, että suunnittelijan täytyy etsiä ja avata ne tietokannasta tarkastelua varten.

1.3 Tutkimustyö

Työssä on käytetään tieteellistä tutkimusta. Tämä tarkoittaa, että työn teossa hyödynnetään menetelmiä, mitkä ovat todettu hyväksi niin aineiston käsittelyn kuin tiedonkeruun osalta. Näiden pohjalta pyritään pääsemään luotettavaan lopputulokseen. Tietoperustassa käydään läpi aiheesta jo olemassa olevaa tietoa ja tutkimuksessa pyritään lisäämään aihealueesta uutta tietoa. Työn tulosten tulee olla perusteltavissa. (Kananen 2010, 24–25.)

Kun tutkitaan ilmiöitä, taustalta löytyy aina joku ratkaistava ongelma. Tieteen avulla koetetaan parantaa ja edistää yhteiskunnan hyvinvointia ja oloja. (Kananen 2010, 18). Tutkimuksista saadun tieteellisen tiedon avulla voidaan täsmentää erilaisiin ongelmiin liittyviä kysymyksiä, joiden avulla on mahdollista saada uusia näkökulmia sekä ideoita oman työn kehitystä varten. Kysymyksissä tulee olla jotain, mikä voidaan tutkimuksen avulla selvittää ja siihen saada ratkaisu. Tieteen teossa on otettava huomioon, että tekijällä on ennestään tarpeeksi tietoa tieteenalasta ja hän osaa käyttää ennestään hyväksyttyjä tutkimustapoja sekä metodisia keinoja kysymyksien selvittämiseen. (Hirsjärvi, Remes, Sajavaara, & Sinivuori, 18–22.)

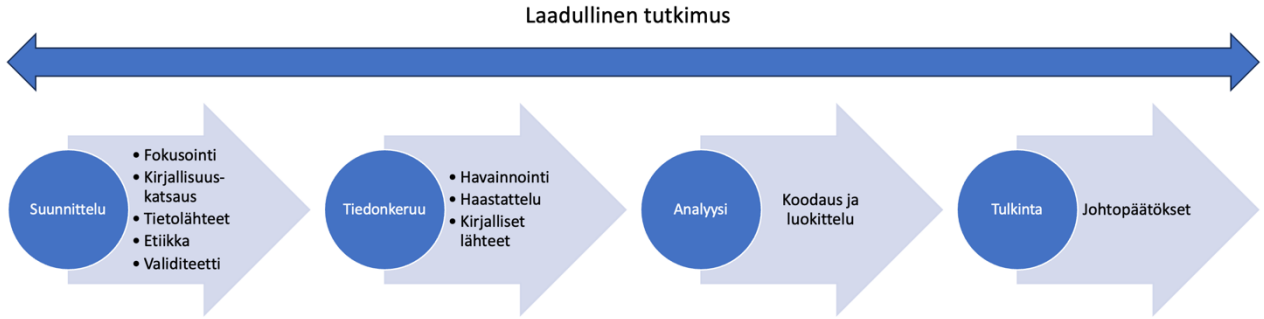
Kun tutkija on selvittänyt ongelmaan liittyvät kysymykset, on niihin aika etsiä ratkaisuja. Suunnitelmallinen, järjestelmällinen ja perusteluihin nojaava tutkimustyö pitää huolen, että tutkimustyön tulokset kestävät tarkastelun. Tarkastelemalla aihepiiriin liittyviä aiempia tutkimuksia ja niiden lopputulemia tutkija saa käsityksen, kuinka ongelman ympärillä liittyviä aiheita on aiemmin ratkottu. Perehdyttyään tarkasti tutkittavaan kohteeseen ja sen metodologiaan voi tutkija luoda itselleen tutkimusta varten suunnitelman tutkimukselleen. Tulosten tulee olla tieteellisesti perusteltavissa. (Hirsjärvi ym., 22.)

Kysymyksiä voi olla yksi tai useampia, joihin haetaan saman muotoista ratkaisua. Kysymyksen muoto voi olla mitä, kuinka, miten, paljonko tai miksi. Kysymyksiin saadaan tieteen avulla saadaan tietyn muotoinen ratkaisu avaamalla ensin ilmiötä, tutkimalla sen sisältöä ja yhdistämällä kuinka ne vaikuttavat tähän ilmiöön. Mikäli ilmiöstä ei ole aiempaa tutkimustietoa joko tutkimuksen puutteen tai uuden näkökulman vuoksi, on pyrkimys ensin määrittää tutkittava ilmiö. (Kananen 2010, 19–20.)

1.4 Kvalitatiivinen tutkimus

Kvalitatiivisessa tutkimuksessa pyritään saamaan mahdollisimman kokonaisvaltainen ymmärrys tutkittavasta kohteesta. Siinä koetetaan saada ymmärrys kohteen ominaisuuksista, laadusta sekä merkityksistä. Kvalitatiivista tutkimusta on mahdollista tehdä esimerkiksi haastatteluiden tai kyselyiden muodossa. (Hirsjärvi ym., 160–163.) Kvalitatiivisen tutkimuksen avulla saadaan hahmoteltua tutkimukselle tutkittava ilmiö, jonka pohjalta voidaan tehdä kvantitatiivinen tutkimus. Kvalitatiivinen tutkimus on pohja kaikelle tutkimukselle, sillä sen avulla saadaan määritettyä tutkimuskysymykset. Kvalitatiivinen tutkimus etenee usein kuvion yksi mukaisen prosessikaavion mukaisesti (Kananen 2010, 37.)

Ihminen kykenee joustamaan erilaisissa tilanteissa, minkä vuoksi kvalitatiivisessa tutkimuksessa tutkija luottaa enemmän omiin havaintoihinsa esimerkiksi haastattelutilanteessa, kuin erilaisilla mittausvälineillä hankittuun tietoon. Aineistoa tulee tarkastella yksityiskohtaisesti eri näkökulmista kokonaisvaltaisen kuvan saamiseksi. Tutkimukselle on tyypillistä, että sen menetelmät muotoutuvat tutkimuksen edetessä. Koska kaikkiin kysymyksiin ei ole mahdollista vastata määrällisesti, on tutkimuksen tukena käytettävä kvalitatiivisia tutkimuskeinoja. (Hirsjärvi ym., 160–164.)



Kuvio 1. Kvalitatiivisen tutkimuksen prosessikaavio (Kananen 2010, muokattu)

1.5 Kvantitatiivinen tutkimus

Ihmisillä on tapana tarkastella asioita aina omanlaisella näkökulmalla. Määritelmä hyvälle tai laadukkaalle on subjektiivista. Aihetta on mietitty antiikin kreikan ajoilta asti, kun Aristoteles pohti

kuinka laadun merkitys vaihtelee jokaisen yksilön kohdalla (Ross, 168.) Tämän vuoksi tarvitsemme usein tutkimuksen tueksi myös kvantitatiivisia tietoja.

Kvantitatiivinen tutkimus tarkoittaa, että tutkimuksessa käytetään hyödyksi erilaisia laskennallisia ja täsmällisiä tilastollisia menetelmiä. Siinä keskeisiä ominaisuuksia on aiempien teorioiden sekä tutkimusten johtopäätökset, uutta tutkimusta ajatellen käsitteiden määrittely sekä hypoteesin esittäminen. Aineiston keruuta varten on hyvä tehdä etukäteen suunnitelma, jotta siihen voidaan soveltaa tilastollisia analyysityökaluja. (Hirsjärvi ym., 140.)

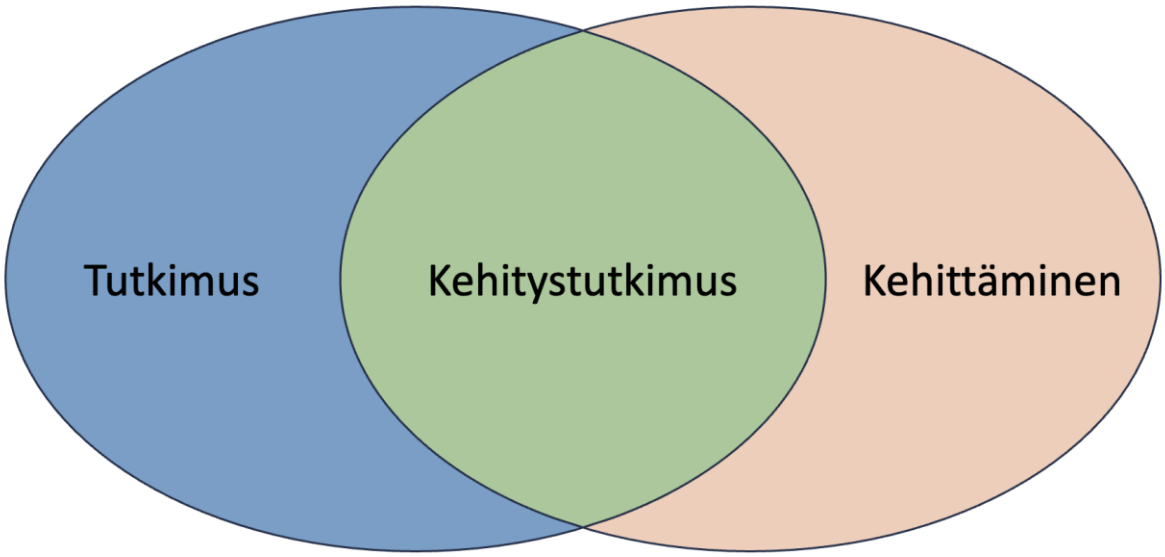
Kvantitatiivinen tutkimus tarvitsee tunnetun ilmiön ja teoriapohjan sen tutkimisen tueksi. Ongelmaan haetaan tutkimuskysymys, jonka pohjalta voidaan kerätä aineistoa tutkimuksen tueksi. Tiedonkeruun jälkeen tulkitaan tuloksia tilastotieteen avulla. Kvantitatiivisessa tutkimuksessa prosessi kulkee jatkuvasti eteenpäin, jolloin alussa tehtyjä virheitä ei voi korjata helposti vaan prosessi on aloitettava alusta. Kun tutkija on saanut kokoon laadukkaan tietoaineiston, hän voi tutkia lukujen välisiä suhteita käyttämällä hyödykseen erilaisia laskuoperaatioita. Tutkija voi käyttää prosessin aikana kuvion kaksi mukaista kaaviota hyödykseen. (Kananen 2010, 74–77.)

Kuvio 2. Kvantitatiivisen tutkimuksen prosessikaavio (Kananen 2010, muokattu)

1.6 Kehittämistutkimus

Tutkimusten taustalta löytyy usein ongelma, joka halutaan ratkaista. Kehittämistutkimuksien aiheet ovat yleensä ammattiin sidottuja, ja niillä koetetaan parantaa tutkimuksen taustalla olevaa ilmiötä tai prosessia. Kun ongelmaan on löydetty ratkaisu, se on tarkoitus viedä käytäntöön. Jokai-

nen askel matkalla ongelmasta ratkaisuun tulee olla perusteltu ja dokumentoitu, jolloin kuka tahansa ulkopuolinen taho voi ymmärtää, kuinka lopputulokseen on päästy. Kehittämistutkimuksessa on tarkoitus löytää työelämän kannalta ongelmaan ratkaisu, joka on myös varmistettu toimivaksi. (Kananen 2012, 12–16.)



Kuvio 3. Kehitystutkimuksessa on tarkoitus yhdistää tutkimus ja kehitys

Ongelman ratkaisuun on määritetty erilaiset menetelmät niin tiedonkeruuseen, analysointiin kuin ongelmanratkaisuun. Nämä tulee työssä kuvata ja perustella. Tiedonkeruun lähteenä on mahdollista käyttää monia erilaisia tietolähteitä. Aiemmista tutkimuksista ja kirjallisuuskatsauksista on mahdollista löytää hyviä mittareita ja työkaluja oman työn toteuttamisen avuksi. Kehittämistutkimuksessa yhdistellään eri tutkimusmenetelmiä, joiden avulla pyritään tuottamaan ongelmaan käytännöllinen ratkaisu. (Kananen 2012, 16–19.) Kehittämistutkimus eroaa perinteisestä tutkimuksesta siinä, että siihen kuvion kolme tavoin liitetään tutkimuksen rinnalle kehittämisprosessi.

Kuten Kananen (2012, 46) mainitsee, ei pelkästään esimerkiksi järjestelmien käyttöönotto taikka jonkin asian toteuttaminen ole tutkimukseksi luokiteltavaa. Jos järjestelmään tehdään muutoksia, voi olla mahdollisesti kyse kehitystutkimuksesta. Tutkimus vaatii toteutuksen rinnalle jonkin tutkittavan ongelman. Kehittämistutkimuksessa itse tutkimus tapahtuu lähinnä prosessin alussa ja lopussa, kun prosessin keskellä tapahtuu itse tuotteen, palvelun, prosessin taikka toiminnan kehittäminen (Kananen 2012, 45).

1.7 Hyvä tieteellinen käytäntö ja eettisyys

Hyvän tieteellisen käytännön perusperiaatteiden mukaisesti työssä käytetään alusta loppuun niin tieteellistä käytäntöä kuin menettelytapoja (Hyvä tieteellinen käytäntö… 2023, 11), joiden avulla varmistetaan toiminnan laatu työtä suunnitellessa sekä menetelmiä ja analyyseja käyttäessä.

Avoimen tieteen periaatteita kunnioittaen tässä työssä otetaan huomioon aiempaa tutkimustietoa sekä muiden tekemästä työstä annetaan niille kuuluva arvo asianmukaisesti. Muiden tulkinnat, tutkimusaineistot sekä ideat tulee ilmaista selkeästi. Tutkimusaineiston keräämisestä ja säilytyksestä sovitaan asianomaisten kanssa ja tarvittaessa sopimuksia tarkennetaan työn edetessä. Aineiston avoimuutta edistetään niiltä osin, miltä se on mahdollista. (Hyvä tieteellinen käytäntö… 2023, 13–14.)

Työssä käytetyssä aineistossa ei ole tietoja, joiden avulla olisi mahdollista tunnistaa suoraan taikka välillisesti henkilöitä tai yhdistää tietueita kenenkään yksilön tekemäksi. Aineistosta ei löydy henkilötietoja Aineistoon ei ulkopuoliset pääse käsiksi eikä se ole tarkasteltavissa. Koneoppimismalliin ei tallennu yksittäisiä tietueita, ainoastaan painotuksia erilaisille ominaisuuksille.

1.8 Tutkimuskysymykset ja työn rajaus

Kuten Hirsjärvi ja muut mainitsevat, voi kysymysten laatiminen ongelmien pohjalta olla hankalampaa kuin itse tutkimustyön toteuttaminen. (Hirsjärvi ym., 125). Työn pohjimmaisen tavoitteen pohjalta mielessä oli selkeä struktuuri, jonka mukaan ongelmaa voisi lähteä ratkaisemaan. Tämän ajatuksen pohjalta on herännyt kysymyksiä, joihin olisi tavoite on saada vastaus:

- Mitä kaikkea toimivan koneoppimismallin koulutus vaatii?
- Kuinka suuri työ on saada integroitua edellä mainittu malli tuotantoympäristöön?
- Millä tavalla työkalun tuottavuutta voidaan mitata?

Koska tällaisella työkalulla on runsaasti erilaisia mahdollisuuksia ja potentiaalia skaalautua suuriinkin työtehtäviin, tässä työssä työkalu on rajattu koskemaan Catian Drafting työtilassa olevia piirustusten mittoja ja niiden mittatoleransseja. Catian API:ssa on mahdollisuus tunnistaa erilaiset pituus- sekä halkaisijamitat, joten molempia toleransseja varten laaditaan omat mallit. Halkaisijamitoista ei nykyisellään pysty päättelemään onko kyseessä reikä- vai akselimitoitus, joten malli on koulutettava ja toivottava että se kykenee ehdottamaan mittaan sopivaa toleranssia. Mikäli yksinkertainen työkalu saadaan toimimaan, kehityksen ajalta saadun kokemuksen avulla on mahdollista lähteä jatkokehittämään vaativampia järjestelmiä.

2 Tietoperusta

2.1 Tekoälyä vai koneoppimista?

Viimeisen parin vuoden aikana termien tekoäly ja AI käyttö ovat yleistyneet, ja ne ovat tulleet mukaan arkikieleen. Tekoälyllä viitataan tietokoneisiin, mitkä jäljittelevät ihmisten kognitiivisia toimintoja sekä älykkyyttä. Tarkempi määrittely on hankalaa, sillä älykkyys on määritelmänä hyvin laaja. Kaplan ja Haenlein (2019) ovat aikoinaan tulkinneet tekoälyn määritelmäksi ”järjestelmän kyvyksi tulkita ulkoista tietoa oikein, oppia tällaisesta tiedosta ja käyttää oppimaansa tiettyjen tavoitteiden ja tehtävien saavuttamiseksi joustavan mukauttamisen avulla”. Kuten Karaali (2023, 6) julkaisussaan mainitsee, tämän hetken tekoälysovelluksilta puuttuu kyvykkään ja kriittisen ajattelun taidot, joita ihmisiltä löytyy.

Tekoäly luokitellaan nykyään kolmeen luokkaan. Yleisimpänä kapea tai erikoistunut tekoäly (ANI, Artificial Narrow Intelligence), mihin kaikki nykyiset tekoälysovellukset kuuluvat. Sen lisäksi tekoälyn voi luokitella yleiseen / ihmisen kaltaiseen (AGI, Artificial General Intelligence) taikka ihmisen suorituskyvyn ylittäviin tekoälyihin (ASI, Artificial Super Intelligence). (Tekoäly on hyvä renki... 2020.) Näistä kapean tekoälyn sovelluksista voi mainita esimerkkeinä erilaiset konenäön sovellukset tai esimerkiksi monelle tutun luonnollisen kielen käsittelyyn tehdyn laajan kielimallin ChatGPT:n. (AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What’s the difference? 2023.)

Koneoppiminen on tekoälyn alalaji, minkä avulla kone tai järjestelmä oppii ja parantamaan itseään kokemuksen perusteella. Kapean tekoälyn sovellus on tehty suoriutumaan tietystä tehtävästä hyvin, ja koneoppiminen tarkoittaa menetelmiä, joilla se sovellus saadaan aikaan. Tekoälyä voi ajatella sateenvarjona, minkä alle koneoppiminen kuuluu. Koneoppimisen lisäksi varjon alle kuuluu luonnollisen kielen käsittely, syväoppiminen ja robotiikka. (Artificial intelligence (AI) vs. machine learning (ML) n.d.)

Ohjelmistot kehitettiin aikoinaan erilaisten yritysprosessien sujuvoittamiseksi. Ajan saatossa ohjelmistot ovat levinneet jokapäiväiseen elämäämme, ja myös niiden käytettävyyteen on alettu kiinnittämään erityistä huomiota. Nykyään ohjelmistoilla on kolme pääasiallista tavoitetta; säästää ihmisiä toistuvien työtehtävien tekemiseltä, peilata prosesseja kuin ne reaalimaailmassa tapahtuu sekä auttaa ja voimaannuttaa ihmisiä. (Esposito & Esposito 2022.)

Tekoäly on samalla lailla ohjelmistoa missä muutkin tähän asti tehdyt tietokoneohjelmistot. Siinä missä aiemmin tiedostojen tallennus ja tiedonhaku olivat automaatiota, nykyään tekoälyn kykyä lukea ja tiivistää pitkiä tekstejä voidaan pitää samanlaisena edistysaskeleena. Tekoäly ei ole enää vain tutkimuskohde, vaan se alkaa olla osa nykyaikaista ohjelmistoa. Haaste onkin kehittää reaalimaailman ongelmiin sopivia ratkaisuja. Parhaimmatkaan tekoälymallit eivät ole hyödyllisiä, ellei niille ole toimivaa käyttöliittymää. (Esposito & Esposito 2022.)

Kuvio 4. Koneoppimismalli toimii kuten muutkin ohjelmat

Yksinkertaistettuna jokainen tekoälymalli on ohjelma, mikä pohjautuu matematiikkaan. Kuvion neljä mukaisesti ohjelmalle annetaan syöte, jonka se prosessoi matematiikan ja algoritmien avulla muuntaen sen ulostuloksi. Ulostulona voi toimia esimerkiksi luokittelun tulokset todennäköisyyksineen tai ennuste jostain luvuista, kuten lämpötilasta. Luonnollisen kielenkäsittelyn ollessa kyseessä malli ennustaa annetun syötteen pohjalta tekstikokonaisuuksia laskemalla todennäköisyyksiä sanojen muodostamille sarjoille (Introduction to Large Language Models, n.d.)

2.2 Koneoppimisesta yleisesti

Koneoppimisen tarkoitus on kouluttaa tietokone tekemään ennustuksia koulutustietoaineiston avulla. Koneoppimisessa tietokone etsii sille syötetyistä tietojoukoista korrelaatioita algoritmien avulla, joiden pohjalta koulutettu malli pystyy tekemään päätöksiä ja ennustuksia. (Machine learning definition in detail, n.d.) Koneoppimismallien tarkkuus riippuu pitkälti niille syötetyn laadukkaan tietoaineiston määrästä. Osa algoritmeista tukee mallin uudelleenkoulutusta, kun taas joidenkin algoritmien kohdalla malli on aina koulutettava alusta asti uudelleen.

Koneoppimisessa on kolme päätyyppiä, jotka ovat ohjattu, vahvistettu ja ohjaamaton oppiminen. Jokainen niistä toimii käyttäen aineistoa ja algoritmeja, joiden pohjalta ne tekevät joko ennustuksia, etsivät aineistosta säännönmukaisuuksia, taikka ne oppivat tunnistamaan erilaisia luokkia. (3 Types of Machine Learning You Should Know, 2023.)

2.2.1 Ohjattu oppiminen

Ohjatussa oppimisessa mallille annetaan koulutusaineistoa sekä ennalta määritetty tunniste. Tästä esimerkkinä toimii malli, joka koettaa tunnistaa kuvasta eläimen. Tällöin koulutusdatassa on kuvia eri eläimistä, ja kuvien mukana on tunniste, joka kertoo mallille mikä eläin kuvasta löytyy. Näin malli oppii ohjatusti tunnistamaan kuvista eri eläimet.

Tätä koneoppimisen tyyppiä kutsutaan ohjatuksi sen vuoksi, että algoritmille syötetään valmiiksi tunnistettua dataa, joiden avulla se voi päätellä sille syötetyistä ominaisuuksista lopputuloksen.

Näiden pohjalta mallin on mahdollista löytää ne korrelaatiot, joiden avulla päästään annettuun lopputulokseen. (3 Types of Machine Learning You Should Know 2023.)

Huonona puolena ohjatun oppimisen mallit yleensä tarvitsevat runsaasti laadukasta korvamerkittyä aineistoa mallin koulutukseen. Mallin koulutuksessa voi myös mennä ylisovittamisen puolelle, jolloin mallin osaaminen kohdistuu liikaa koulutusaineistoon eikä se pysty ennustamaan sen ulkopuolelta mitään oikein.

2.2.2 Vahvistettu oppiminen

Vahvistettu oppiminen tarkoittaa, että algoritmi oppii olemalla vuorovaikutuksessa sille syötettyjen ominaisuuksien kanssa ja se tekee esimerkiksi luokittelun tai ennustuksen niiden perusteella, riippuen millaista mallia ollaan kouluttamassa. Malli saa oikein ja väärin menneiden ennustuksien pohjalta joko positiivista tai negatiivista palautetta, joka ohjaa algoritmia oikeaan suuntaan.

Vahvistettu oppiminen toimii myös silloin, kun käytettävä tietoaineisto on merkitsemätöntä. Ilman tunnisteita malli voi tehdä ennustuksia, jolloin malli oppii ongelmasta lisää jokaisen ennustuksen ja vahvistuksen kautta. (3 Types of Machine Learning You Should Know 2023.) Esimerkkinä tällaisesta mallista voisi olla videopelin tekoäly, joka oppii pelaajan liikkeistä ja muokkaa toimintaansa sen mukaan.

2.2.3 Ohjaamaton oppiminen

Ohjaamattomassa oppimisessa mallille ei anneta laisinkaan merkittyjä tietueita. Tällöin mallille annetaan vain suuri määrä aineistoa, mistä se etsii rakenteita ja ominaisuuksia. Ohjaamaton oppiminen ei ole järin tehokas tunnistamaan aineistosta rakenteita, joiden avulla se voisi tehdä ennustuksia. Myös mallin suorituskyvyn arviointi voi olla hankalaa.

Ohjaamattoman oppimisen vahvuutena on sen kyky löytää sekalaisesta aineistosta erilaisia korrelaatioita ja tehdä sen mukaan klustereita. Esimerkkinä ohjaamattoman oppimisen mallista voisi olla asiakassegmentointi, jolloin se osaa luokitella asiakkaita erilaisiin ryhmiin ostosten perusteella. Ohjaamattoman oppimisen avulla malli voi löytää ominaisuuksia, joita ei perinteisellä data-analyysilla ole mahdollista löytää.

2.3 Koneoppimisjärjestelmän teon eri vaiheet

Koneoppimismallin koulutuksessa on useita vaiheita. Ensimmäisenä tulee selvittää millaiseen ongelmaan on malli on tarkoitus kouluttaa, ja kuinka se voi ratkaista ongelman. Tämän jälkeen on selvitettävä millaista aineistoa ongelman ratkaisu vaatii ja kuinka paljon sitä on saatavilla. Tämän jälkeen aineisto tulee esikäsitellä muotoon, joka tukee mallin koulutusta. (Sakhuja 2022.) Data-analyysista tarkemmin kappaleessa 2.8.

Viimeistään aineiston esikäsittelyn jälkeen selvitetään ennakkoedellytykset koulutusta varten. Koulutusalgoritmeja on runsaasti erilaisia tehtäviä varten. Näistä tehtävistä esimerkkeinä voi mainita:

- binääriluokittelun sähköpostien roskapostien tunnistamiseen,
- regressioalgoritmin hintojen ennustamiseen,
- tuotteiden suositus asiakkaille aiempien ostosten perusteella,
- objektintunnistus kuvasta tai videosta,
- aineiston ja syötteiden luokittelu eri kategorioihin.

Koulutusalgoritmin ja käytettävien funktioiden valinta vaihteleekin ratkaistavan ongelman mukaan. Yhtä algoritmia voi käyttää moneen eri tehtävään, jolloin samaan ongelmaan voi kouluttaa useamman mallin käyttäen eri algoritmeja ja vertailla näiden suorituskykyä parhaan löytämiseksi.

Koulutusaineisto jaetaan yleensä vähintäänkin kahteen osaan, koulutus- ja testausaineistoon. Joissain tapauksissa tämän lisäksi osaa käytetään validointiaineistona, jonka avulla koulutuksen aikana malli säätää parametrejään tämän avulla.

Koulutusvaiheessa malli käyttää sille syötettyjä ominaisuuksia valittua algoritmia hyödyntäen itsensä koulutukseen, jotta se pääsisi toivottuun tulokseen. Tätä tavoitetta yleensä mitataan jollain mittarilla, joista lisää kohdassa 4.5.2. Koulutuksen jälkeen on mallia testataan, minkä avulla selvitetään mallin soveltuvuutta ja suorituskykyä jonkin mittarin avulla. Esimerkki yleisestä mittarista on tarkkuus, millä tarkastellaan oikein menneiden veikkausten määrää verrattuna testattujen tietueiden kokonaismäärään. Kahdeksan oikein mennyttä veikkausta kymmenen tietueen joukosta tarkoittaa 80 % tarkkuutta. Tarkkuus ei kuitenkaan ole kaikissa tilanteissa luotettavin mahdollinen mittari, joten suorituskyvyn mittausta varten mittarit tulee valita tapauskohtaisesti.

Mikäli tässä vaiheessa malli ei täytä sille annettuja vaatimuksia, voidaan sen koulutusparametrejä muuttaa. Usean iteraation jälkeen mallin tulisi olla mahdollisimman tehokas sille annetun ongelman ratkaisemisessa. Tällöin malli on valmis tekemään luotettavasti ennustuksia tuotantoympäristössä, mutta mallin suorituskykyä olisi silti hyvä tarkkailla ja säätää mallia tarvittaessa.

2.4 MLOps – käytäntöjä elinkaaren automatisointiin

Koneoppimisjärjestelmään koulutuksen ja käyttöönoton lisäksi on tärkeää suunnitella käyttöönoton jälkeinen aika. Tuotantoympäristöön integroinnin ohella tulee ottaa huomioon mallin elinkaari ja ylläpito käyttöönoton jälkeen. Kuvioon viisi on Googlen toimesta kerätty tiivis paketti huomioitavia asioita. Mallit voivat toimia hyvin yrityksen omissa tiloissa ja testeissä, mutta integroitaessa toisenlaiseen ympäristöön on mahdollista, että jotain menee jossain vaiheessa pieleen mallin ollessa kykenemätön mukautumaan uuden ympäristön tuomiin muutoksiin. (MLOps: Continuous delivery… 2023.)

Kuvio 5. Koneoppimisjärjestelmän elementtejä Googlen esittämänä (cloud.google.com, muokattu)

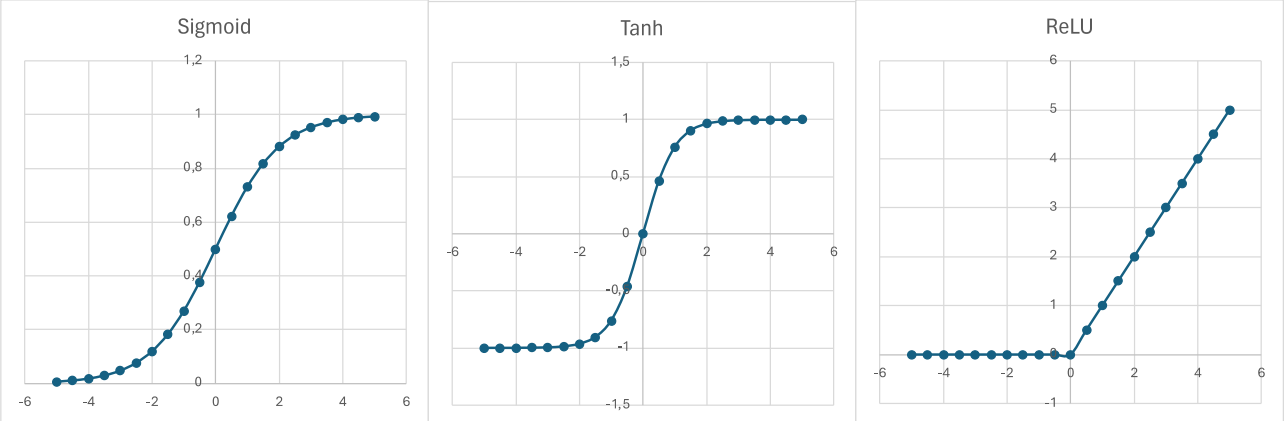
Koneoppimisjärjestelmien monimutkaisuus tuo omat ongelmansa elinkaaren automatisoimista ajatellen. Ympäristön muutokset voivat saada mallit toimimaan odottamattomasti, jolloin niiden ylläpitokustannukset voivat nousta. (Chaudhary, Davydov, Ebner, Golovin, Holt, Phillips, Sculley, Young n.d., 1–2.) Koneoppimista varten on saatavilla erilaisiin ongelmiin rutkasti valmiita ohjelmistopaketteja ja -kirjastoja. Järjestelmää suunnitellessa kannattaa kuitenkin huomioida kokonaisuus siltä osin, ettei koodiin lisätä erillisiä kirjastoja liikaa. Tietyissä tapauksissa valmiiden ohjelmistokirjastojen sisällyttäminen järjestelmään houkuttaisi, mutta voi olla järkevämpää implementoida ominaisuudet itse, etenkin jos kirjastot on kirjoitettu toisella kielellä. Tällöin ajan mittaan järjestelmä voi pysyä selkeämpänä ylläpitää. (Chaudhary ym., N.d. 5–6.)

Järjestelmän työvaiheet tietoaineiston kaapimisesta tuotantoon vientiin on mahdollista automatisoida. Keräämällä jatkuvasti uutta tietoaineistoa ja todentamalla sen luotettavuus voidaan uudelleenkouluttaa malli säännöllisesti tai silloin, kun mallin suorituskyky vaikuttaa heikentyneen tuotannossa. Uuden ja vanhan mallin suorituskykyä voi vertailla mittareiden avulla, jolloin parempi tulee tuotannon käyttöön. Tarkkailemalla mallin suorituskykyä voidaan ennakoida muutosten tai uudelleenkoulutuksen tarve. Suunnittelemalla aineiston kaavinta ja käsittely, virheiden korjaus ja testaus, sekä mallin koulutus modulaarisiksi ohjelmistoikseen voidaan helpottaa järjestelmän ylläpitoa pitkällä aikavälillä. (MLOps: Continuous delivery… 2023.)

2.5 Tensoripohjaiset, neuroverkkojen luomista varten parhaat alustat

Alla esitellyt TensorFlow sekä PyTorch tukevat neuroverkkojen koulutusta ja käyttävät tensoreita apuna laskennassa. Verkot muodostuvat tasoista, jotka ovat yhteydessä toisiinsa. Tasoina toimii syöte, ulostulo ja niiden keskellä verkon mukaan vaihteleva määrä piilotettuja tasoja. Näiden välillä neuronit laskevat aktivointifunktioita käyttäen ulostuloa.

Yleisimpiä aktivointifunktioita neuroverkkojen osalta ovat Sigmoid, Tanh, ReLU ja Softmax. Näistä kolme ensimmäistä on visualisoitu kuviossa kuusi. Sigmoid on differentioituva ja monotoninen funktio, tarkoittaen että käyrälle voidaan löytää kaltevuus käyttäen mitä tahansa kahta pistettä ja käyrä ei ole täysin laskeva taikka nouseva. Arvot sijoittuvat nollan ja yhden välille. Tanh funktio toimii sigmoidin tavoin, mutta arvot sijoittuvat välille miinus yksi – yksi. ReLU, eli rectified linear unit on nykyään yleinen neuroverkoissa, koska se on kärsii epätodennäköisemmin häviävän gradientin ongelmasta. Siinä kaikki negatiiviset arvot saavat arvon nolla, kun positiiviset arvot pysyvät muuttumattomina. (Pykes 2022.)



Kuvio 6. Sigmoid, Tanh ja ReLU aktivointifunktiot visualisoituna Excelissä

Softmax aktivointifunktio toimii pääsääntöisesti viimeisenä ulostulotasona, sillä sitä käytetään usein luokittelussa todennäköisyyksien laskemiseen eri luokkien välille. (Pykes 2022.) Softmax funktio toimii alla olevan kaavan mukaisesti:

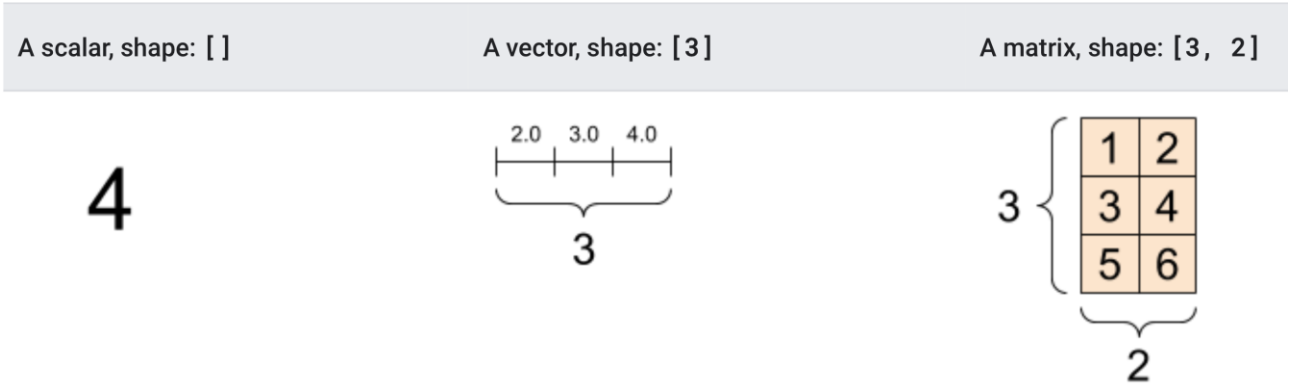
$$\begin{bmatrix} -1 \\ 0 \\ 2 \\ 5 \end{bmatrix} \rightarrow \frac{e^{z_i}}{\sum_{j=1}^{n} e^{z_j}} \rightarrow \begin{bmatrix} 0{,}002 \\ 0{,}006 \\ 0{,}047 \\ 0{,}944 \end{bmatrix}$$

Missä e on Neperin luku, ja z on syötteenä toimiva luku. Kaava tarkoittaa, että ensin jokainen syötematriisin luku korotetaan luonnollisen eksponenttifunktion potenssiin, minkä jälkeen nämä tulokset jaetaan kaikkien eksponenttien summalla. Tuloksena on syötteen kokoinen matriisi lukuja nollan ja yhden väliltä, eli toisin sanoen todennäköisyysjakauma.

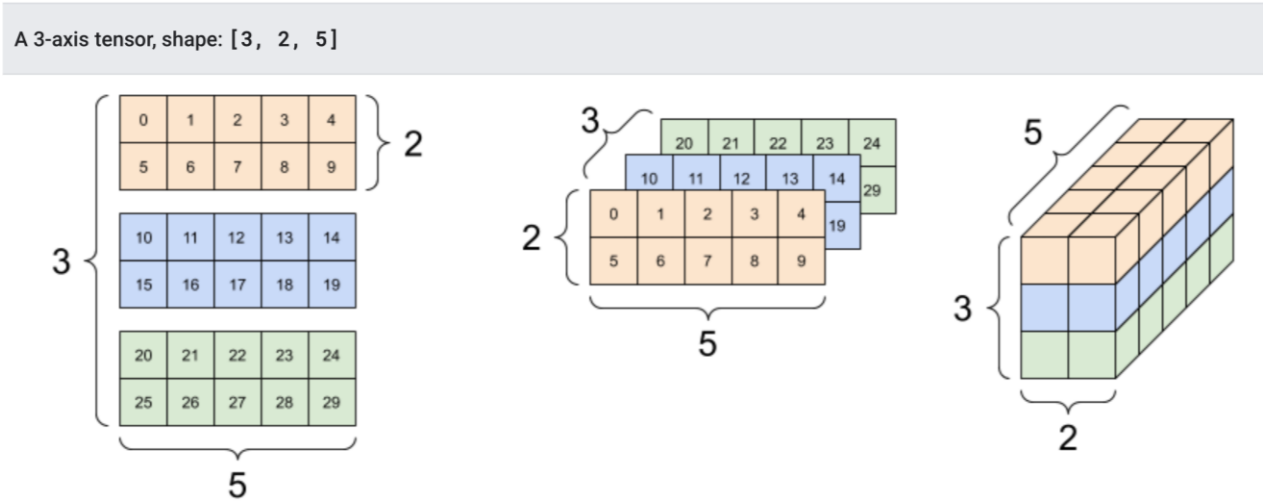
Nämä aktivointifunktiot toimivat myös muussa käytössä kuin neuroverkkojen koulutuksessa, mutta niissä näitä käytetään melko yleisesti.

2.5.1 Mikä on tensori?

Tensori on moniulotteinen matriisi, millä on tietty tiedostomuoto. Käytännössä niitä voi kutsua tietorakenteiksi, mitkä voivat säilyttää erimuotoista tietoaineistoa. Ne skaalautuvat hyvin erilaisiin laskentatehtäviin. Tensorit ovat muuttumattomia, eli niiden sisältöä ei voi jälkikäteen muuttaa. Tämä tarkoittaa, ettei niiden sisältöä voi muuttaa niiden luonnin jälkeen. Molemmat TensorFlow sekä PyTorch käyttävät näitä tensoreita laskennassaan. Tensorit voivat olla erilaisia kertolukuja, kuten skalaari (0. kertaluku), vektori (1. kertaluku), matriisi (2. kertaluku), jotka ovat kuvattu kuviossa seitsemän. Tällaisen tensorin lisäksi ne voivat olla useamman kertaluvun omaavia, kuten esimerkki kuviossa kahdeksan. (Introduction to tensors n.d.). Käytännössä kaikista matriiseista, joissa on enemmän kuin kaksi ulottuvuutta käytetään nimitystä tensori (Introduction to PyTorch tensors n.d.).



Kuvio 7. Tensoreiden kolme ensimmäistä kertalukua. (Lähde: Tensorflow.org)



Kuvio 8. Kolmiulotteisen tensorin [3, 2, 5] muoto (Tensorflow.org)

Tensoreiden määrä, muoto ja tietotyyppi riippuvat mallille syötetyistä ominaisuuksista. Tietokoneet käsittelevät tietoa numeroiden muodossa, minkä vuoksi esimerkiksi tekstimuotoiset tietotyypit tulee muuttaa ensin numeraaliseen muotoon käsittelyä varten.

2.5.2 TensorFlow – Googlen koneoppimisalusta

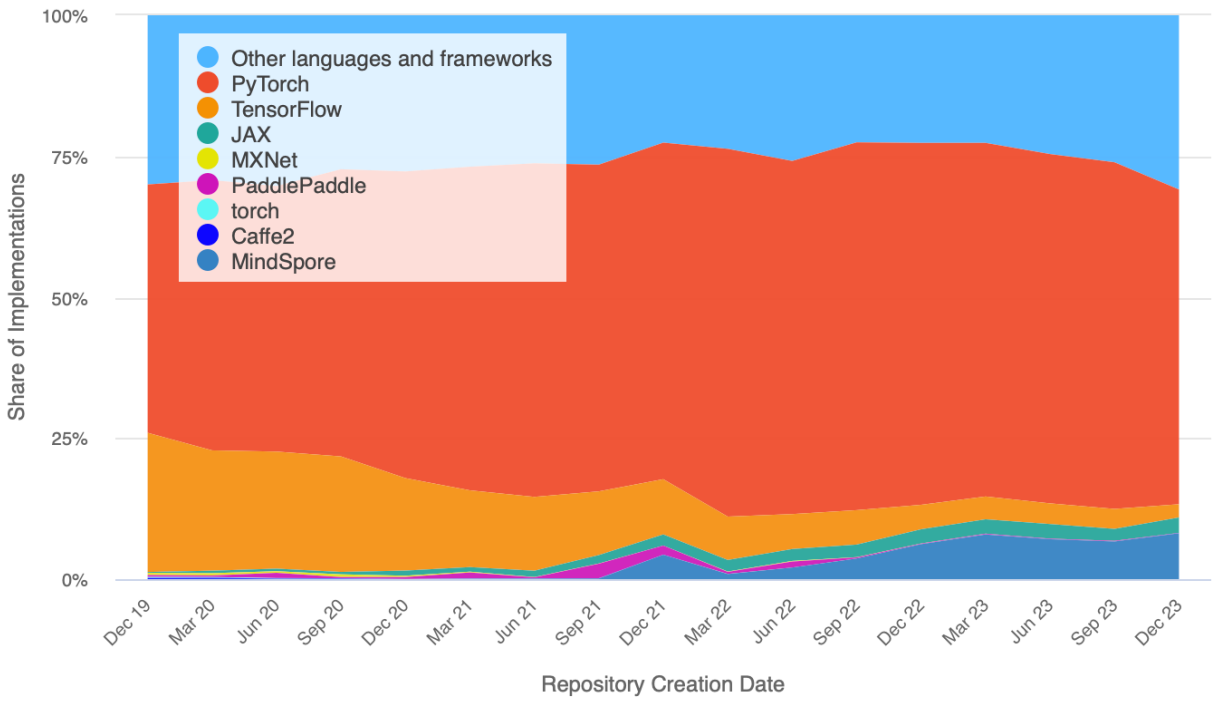
TensorFlow on Googlen koneoppimista ja tekoälyn kehitystä varten tekemä avoimen lähdekoodin ohjelmointikirjasto. Se on kokonaisvaltainen alusta, millä voi hoitaa koko suoritusketjun tuotantoon asti. TensorFlown 2.0 versiossa lisättiin integrointi korkean tason ohjelmointirajapinnan Kerasin kanssa. TensorFlow on hyvä erilaisten koneoppimis- sekä syväoppimissovellusten tekoon. Ohjelmointikielenä TensorFlowissa toimii Python, mutta Keras tuo tähän päälle oman syntaksinsa. (Keras: The high-level API for TensorFlow n.d.)

Keras mahdollistaa mallien esikäsittelyn, koulutuksen ja tuotantoon viennin hieman lego palikoiden tavoin (Challet 2015). Siinä on käyttäjäystävällinen rajapinta, minkä avulla saa hoidettua niin datan tuomisen, esikäsittelyn, mallien kouluttamisen ja testaamisen, sekä lopuksi mallin tallentamisen tuotantoympäristöön integrointia varten. Kaikki esikäsittely saadaan sisällytettyä malliin, jolloin esimerkiksi koulutuksessa ja tuotannossa ominaisuudet käsitellään samalla tapaa. (Keras: The high-level API for TensorFlow n.d.)

2.5.3 PyTorch – Metan vastine koneoppimiselle

TensorFlown tapaan PyTorch on Pythonin päälle rakennettu ohjelmistokirjasto erilaisten syväoppimissovellusten sekä neuroverkkojen koulutukseen. PyTorch on Metan (entinen Facebook) kehittämä (Yegulalp 2017) ja käyttää TensorFlown tavoin tensoreita moniulotteisten matriisien laskennassa. PyTorch muistuttaa enemmän tavallista Pythonia kuin esimerkiksi TensorFlow. (Yasar & Lewis 2022.)

Vahvuuksina PyTorchille voi lukea sen virheiden jäljittämisen helppouden ja integroinnin ONNX:n kanssa. Lisäksi sen syntaksi muistuttaa perus Pythonia muita koneoppimisalustoja enemmän. Kuten kuviosta yhdeksän näkyy, se on selvästi suosituin vaihtoehto tutkijoiden keskuudessa. Kääntöpuolena PyTorch on hankalampi saada integroitua tuotantoympäristöön kuin esimerkiksi TensorFlow. (Wadawadagi 2023.)



Kuvio 9. Eri koodauskielien suosio tutkimuspapereissa 3.1.2024. (Paperswithcode.com)

2.6 Algoritmipainotteiset Scikit ja ML.NET koneoppimista varten

Scikit-learn ja ML.NET tukevat pääasiassa algoritmeja koneoppimissovelluksia varten. Algoritmilla tarkoitetaan jonkin asian tekemistä tiettyjen määritettyjen vaiheiden mukaan (Louridas 2020, 5). Tällä voidaan tarkoittaa prosessia, mikä suoritetaan yksityiskohtaisen ohjeiden avulla. Tietokoneet ovat huomattavasti ihmistä nopeampia suorittamaan tällaisia tehtäviä. Ohjelmointia voi kutsua tavaksi saada tietokone ymmärtämään nämä vaiheet. (Louridas 2020, 23.) Tietyt ongelmat ovat niin monimutkaisia, että optimaalisen ratkaisun hakeminen algoritmin avulla ei ole käytännöllistä, mutta monilla approksimaatioalgoritmeilla on mahdollista saada ratkaisu, mikä on tarpeeksi lähellä ihanteellista tulosta ollakseen käytännöllinen (Louridas 2020, 40–41.)

2.6.1 scikit-learn – Koneoppimista Pythonilla

Scikit-learn on edellisten tavoin Python pohjainen koneoppimisalusta. Se eroaa aiemmista siinä, ettei sillä ole tapana kouluttaa neuroverkkoja, vaan se pohjautuu valmiisiin algoritmeihin, joita käyttäjä voi hyödyntää mallien koulutuksessa. Scikitistä löytyy neuroverkoilla tehdyt implementaatiot luokittelua ja regressiota varten, mutta ne eivät ole yhtä suosittuja tai joustavia muokkaamisen suhteen kuin esimerkiksi TensorFlown tai PyTorchin vastaavat. Scikit-learnin vahvuutena on,

että se perustuu monelle Pythonin käyttäjälle tuttuun NumPy kirjastoon. Se myös integroituu kätevästi muihin usein käytettyihin Python kirjastoihin kuten Pandas, Matplotlib sekä SciPy. (scikit-learn dokumentaatio n.d.)

Scikitin API koostuu viidestä ohjaavasta periaatteesta, joiden myötä alustan käyttö on pyritty pitää helppokäyttöisenä. Kaikki scikit-learnin algoritmit on toteutettu Estimator API:n avulla, mikä huolehtii yhtenäisestä käyttöliittymästä alustalla tehdyille sovelluksille. Tämän myötä kaikissa malleissa koulutuksen kaava on sama, estimaattorityypistä riippumatta. (VanderPlas 2016.)

2.6.2 ML.NET – Microsoftin C# pohjainen koneoppimisalusta

ML.NET on Microsoftin kehittämä alusta koneoppimistyökaluja ja malleja varten. Suurin ero aiempiin alustoihin on siinä, että se on suunniteltu helposti integroitavaksi .NET alustalla tehtyihin ohjelmiin. Ohjelmointikielenä ML.NET:ssä toimii C# tai F#. ML.NET:iin on mahdollista integroida malleja muista koneoppimisalustoista, kuten Infer.NET, TensorFlow tai ONNX. Osa malleista tukee myös vientiä ONNX muotoon. (ML.NET – An open source and cross-platform machine learning framework n.d.)

ML.NET:n vahvuus on sen .NET alusta, minkä vuoksi se on helppo integroida samalla alustalla tehtyihin sovelluksiin. ML.NET tarjoaa myös edellisistä alustoista poiketen AutoML ja Model Builder työkalut. ML.NET ei tällä hetkellä tue omien neuroverkkojen luomista, vaan se pohjautuu valmiiden algoritmien käyttöön mallien koulutuksessa. Alusta kuitenkin tukee esimerkiksi TensorFlown avulla koulutettuja neuroverkkomalleja osana suoritusketjua.

Aiemmin mainitut AutoML ja Model Builder ovat kaksi mallien koulutusta helpottavaa työkalua niille, joilla ei ole runsaasti kokemusta ohjelmoinnista. Näistä ensimmäinen on ML.NET:n ohjelmointirajapinta, joka käy automaattisesti ongelmaan sopivia algoritmeja läpi. Käytännössä suoritusketju rakennetaan samalla tavalla kuin muulloinkin, mutta tietyn algoritmin asettamisen sijaan rajapinnalle annetaan aika-arvo, minkä ajan se kokeilee automaattisesti eri algoritmeja. Ajan loputtua ohjelma voi tallentaa mittareihin nähden parhaiten suoriutuneen mallin.

Model Builder on graafisen käyttöliittymän kautta käytettävä liitännäinen Microsoftin ohjelmointiympäristö Visual Studioon. Sen avulla käyttäjän on mahdollista tehdä koneoppimismalleja ilman aiempaa ohjelmointikokemusta. Työkalu tukee AutoML:ää koulutusprosessissaan. (ML.NET Model Builder n.d.) Sen avulla on mahdollista tehdä monen laisia koneoppimismalleja, mutta kaikkiin tilanteisiin se ei sovi. Esimerkiksi tilanteessa, jossa on paljon esikäsiteltävää aineistoa ennen mallille syöttöä, on helpompaa tehdä esikäsittely itse koodin puolella, sillä sitä Model Builder ei tue.

2.7 Koulutusprosessi ML.NET alustalla

Koulutusprosessin vaiheisiin kuuluu tietoaineiston tuominen mallin koulutusta varten, niiden muuntaminen oikeanlaiseen muotoon, ominaisuuksien valitseminen ja lopulta koulutus. Tätä prosessia kutsutaan työssä suoritusketjuksi. Jokaisella koneoppimisalustalla on oma tapansa toteuttaa suoritusketjunsa, mutta kaikki noudattavat jollain tasolla samaa kaavaa.

- Tunnistetaan ongelma, mihin etsitään ratkaisua
- Hankitaan ratkaisua varten tarpeeksi kattava laadukas tietoaineisto
- Tietoaineiston ja ominaisuuksien esikäsittely
- Koneoppimismallin koulutus, testaus ja arviointi
- Mallin tallennus
- Tuotantoympäristöön integrointi

Kun on ensiksi tunnistettu ongelma minkä ratkaisemiseksi tarvitaan koneoppimismalli, on hankittava tietoaineisto. Aineiston ollessa kasassa, sen voi ladata ML.NET:iin. Aineisto voi olla eri muodoissa; yhdessä tai useammassa tiedostossa, tietokannassa tai tietokoneen muistissa. ML.NET:n kohdalla aineisto ladataan koulutusta varten tietokannasta IDataView rajapintaan. Rajapintaa hyödyntäen aineistoa voi pilkkoa osiin, hoitaa puuttuvien arvojen tai tietueiden käsittelyä taikka suodattaa tietoja. Myös muut esikäsittelyt, kuten aineiston normalisointi tai kategorisen aineiston muuttaminen numeraaliseksi, hoidetaan viimeistään tässä vaiheessa.

Kun aineisto on valmis käytettäväksi, sen avulla koulutetaan malli. Koulutusta varten valitaan ongelmaan parhaiten sopeutuva algoritmi. Oikean algoritmin löytäminen onkin oleellinen osa ongelman ratkaisua. Mallin suorituskykyä tarkastellaan eri mittareilla ja tarvittaessa hyperparametreja muutetaan koulutuksen aikana, jotta mallin suorituskyky saadaan toivotulle tasolle. Tämän jälkeen mallin voi tallentaa tuotantoympäristöön sisällyttämistä varten.

2.8 Data-analyysi

Laadukas tietoaineisto on perusta jokaiselle koneoppimis- ja tekoälysovellukselle. Kuten Brownlee (2020a, x) kirjansa esipuheessa mainitsee, tietoaineiston valmistelu vie yleensä eniten aikaa koko projektista, vaikka se onkin vähiten keskusteltu aihe koneoppimismallien koulutuksessa. Monien lähteiden mukaan tiedonhankintaan ja aineiston esikäsittelyyn menee koko prosessista noin 80 % (What is Data Preparation? n.d., Ghosh 2023, Kurama 2020). Dimensional Researchin tekemän tutkimuksen mukaan 96 % yrityksistä on kohdannut ongelmia joko tietoaineiston laadun tai tunnisteiden kanssa (Artificial Intelligence and Machine Learning Projects Are Obstructed by Data Issues 2019). Samassa raportissa mainitaan 72 % yrityksistä maininneen, että tuotantoympäristöön soveltuvan mallin koulutukseen tarvitsisi vähintään 100 000 tietueen aineiston ennen käyttöönottoa.

Suuren tietomäärän lisäksi aineiston tulee olla tarpeeksi laadukasta. Todellisesta maailmasta saadussa aineistossa on usein erilaisia virheitä, poikkeamia tai muuta kohinaa, joka vääristää kokonaisuutta. Osa tiedosta voi olla puutteellista, jolloin joku ominaisuus on esimerkiksi jäänyt tyhjäksi. (Ghosh 2023.)

Raaka tietoaineisto sisältää hyvin todennäköisesti kohinaa tai se on muuten soveltumatonta mallin koulutukseen. Mallin voi toki kouluttaa raakadatalla, mutta tulokset eivät ole yhtä hyviä kuin käsitellyn tietoaineiston kanssa. Kun tietoaineisto on hankittu, tulee se muokata koulutusta varten sopivampaan muotoon esikäsittelyn ja ominaisuuksien suunnitteluprosessin avulla. Aineistoa voi myös olla niin paljon, että oleellisen tiedon poiminta nopeuttaa prosessia huomattavasti (Data Preparation and Feature Engineering in ML n.d.).

2.8.1 Tietoaineiston hankinta ja rakenne

Kuten edellä mainitaan, laadukkaan tietoaineiston hankinta on tärkeä askel jokaisen mallin koulutuksessa. Sen hankinta vaihtelee jokaisen mallin kohdalla, sillä ongelma, jota mallilla pyritään ratkaisemaan, luonnollisestikin vaihtelee. Tietoaineistoa suunnitellessa tulee kuitenkin kiinnittää huomiota minkälaisia ominaisuuksia sitä varten on mahdollista kerätä, saako aineistoa riittävästi luotettavaa mallia varten sekä millä tavoin aineisto hankitaan. Kerätäänkö aineistoa kaapimalla sitä eri tietolähteistä, kyselyiden tai haastatteluiden avulla tai tuotetaanko aineistoa itse käsin?

Tulevaisuudessa EU:n datasäädös avaa valmistajien tekemien tuotteiden ja palveluiden keräämää tietoa muiden käyttöön (Deike, Kempe-Mueller & Righini 2023). Tämä voi avata uusia markkinoita avoimille tietoaineistoille ja sitä myötä raivata tietä innovatiivisille ratkaisuille koneoppimisen saralla. Avaamalla teollisuuden tietoaineistojen käyttöä yrityksille ja kuluttajille saadaan suuria määriä dataa yleiseen käyttöön, minkä avulla voidaan kehittää entistä parempia digitaalisia palveluita (Datasäädös: Komissio ehdottaa… 2022.)

Tietoaineistoa on tietyissä tapauksissa mahdollista tuottaa myös synteettisesti. Esimerkiksi ennakoivaa kunnossapitoa varten Albaryak, Jomâa, Selçuk, & Ünal (2021) ovat tehneet digitaalisen kaksosen avulla synteettistä aineistoa koneoppimista varten, tarkoituksenaan tehdä malli ennakoivaa kunnossapitoa varten. Digitaalisen kaksosen avulla tehtiin fysiikkamallinnuksia, mistä kerättiin aineistoa talteen. Digitaalisen mallin sensoreiden tuottamaa tietoa verrattiin oikean maailman verrokkiin, ja digimallia iteroitiin, kunnes tulokset olivat tarpeeksi lähellä toisiaan. Tämän jälkeen erilaisia vikatilanteita saatiin simuloitua digitaalisesti ilman tarvetta rikkoa oikeaa konetta. Synteettisen aineiston avulla koulutettu mallin tarkkuus testeissä oli 90–99 % välillä. Tulevaisuudessa EU: uuden datasäädöksen vuoksi laitteiden keräämää dataa tulee antaa asiakasyritysten ja kuluttajien käyttöön, jolloin esimerkiksi ostettuun koneeseen voinee tehdä itse mallin ennakoivaa kunnossapitoa varten. (Datasäädös: Komissio ehdottaa… 2022.)

2.8.2 Tietotyypit

Tietoaineistossa oleva ominaisuus on yleensä muodoltaan joko numeerista, nominaalista taikka ordinaalista. Nominaaliset ja ordinaaliset luokitellaan kategorisiksi muodoiksi. Numeerista ominaisuutta voidaan mitata jonkinlaisen väli- tai suhdeasteikon avulla ja ne esitetään reaali- tai kokonaislukuina. Suhdanneasteikon origo on aina nollassa ja sille voidaan tehdä aritmeettisia operaatioita. Väliasteikossa ominaisuuksia mitataan asteikolla, mikä on kiinteä mutta mielivaltaisella origolla sekä välillä, mutta niille ei sovellu kerto- ja jakolaskut. Esimerkkinä näistä on lämpötila; Celsius ja Fahrenheit asteikoita voi laskea ja verrata muttei voida esimerkiksi sanoa 10C olevan kaksi kertaa kylmempi kuin 20C. Kun taas Kelvin asteikko alkaa nollasta, jolloin sen origo on määritelty nollaksi. Tässä tapauksessa Celsius ja Fahrenheit edustavat väliasteikkoa ja Kelvin suhdanneasteikkoa. (Kelleher & Tierney 2018, 41–43.)

Nominaalinen eli kategorinen tietue on jonkun luokan nimi. Niitä ei voi järjestellä arvojärjestykseen kuten numeerisia. Nominaalinen tietue voi olla, vaikka lemmikki, jolloin sen luokkia ovat esimerkiksi kissa, koira, hamsteri ja rotta. Nominaalinen tietue voi olla myös binäärinen, kuten roskaposti, joka joko on roskapostia tai ei ole. (Kelleher & Tierney 2018, 43–44.)

Ordinaalisia piirteitä voi luokitella arvon mukaan, vaikka ne muuten muistuttavat nominaalisia. Ero edellisiin on, ettei arvojen välistä voi olla varma. Esimerkkinä ordinaalisesta tietueesta toimii mielipidekyselyt, joissa on asteikko yhdestä viiteen. Kognitiivinen ero huonoimman ja toisiksi huonoimman arvosanan välillä voi olla eri kuin parhaan ja toisiksi parhaan, jolloin tätä väliä ei voi mitata. (Kelleher & Tierney 2018, 44.) Ominaisuuksien määrä, muoto sekä lukumäärä vaikuttavat tietoaineiston laatuun ja sen käsittelyyn.

2.8.3 Tietoaineiston esikäsittely

Aineiston esikäsittely on tärkeää ennen mallin koulutusta. Esikäsittelyn aikana tietoaineistoa puhdistetaan poistamalla kohinaa aiheuttavia tietueita sekä tietoa, joka ei ole oleellista koulutettavan mallin kannalta. Iso ongelma tietoaineistojen kanssa on puuttuvat tiedot. Niiden tunnistaminen ja käsittely on olennaista, jotta vältytään tulosten vääristymiseltä. Samoin erilaisia poikkeamia voi olla, jotka tulee joko poistaa taikka muokata niin, että tietue ei ole enää poikkeama. Esimerkkinä tästä kirjoitusvirheet syötetyssä tiedossa. (Omardonia 2023.)

Aineiston ollessa ehyt, tietoaineistoa voi myös skaalata ja normalisoida (Omardonia 2023). Yleinen tapa normalisoinnissa on muuttaa numeraaliset arvot lukujen 0 ja 1 välille liukuluvuksi. Myös tekstimuotoinen aineisto tulee esikäsitellä ja muuttaa numeraaliseksi arvoksi ennen mallin koulutusta. Tähän on useita tapoja. Mikäli tekstimuotoisia ominaisuuksia on vähän (esimerkiksi viikonpäivät), One-Hot-Encoding on hyvä valinta. Mutta aineiston kasvaessa on hyvä muuttaa teksti pienempiin osiin kuvion 10 mukaisesti. (Prepare data for building a model 2023.) Osa koneoppimialgoritmeista vaatii, että aineisto on skaalattu, normalisoitu ja tunnisteet nimetty, jotta ne voivat kouluttaa toimivan mallin (Omardonia 2023).

Transform	Description	Result
1. NormalizeText	Converts all letters to lowercase by default	this is a good product
2. TokenizeWords	Splits string into individual words	["this","is","a","good","product"]
3. RemoveDefaultStopWords	Removes stop words like *is* and *a*.	["good","product"]
4. MapValueToKey	Maps the values to keys (categories) based on the input data	[1,2]
5. ProduceNGrams	Transforms text into sequence of consecutive words	[1,1,1,0,0]
6. NormalizeLpNorm	Scale inputs by their lp-norm	[0.577350529, 0.577350529, 0.577350529, 0, 0]

Kuvio 10. Esimerkki lauseen esikäsittelystä koneoppimista varten. (learn.microsoft.com)

Tietoaineisto voi olla myös epätasapainoista. Aineistoa hankkiessa esimerkiksi luokittelutehtäviin on mahdollista, ettei kaikkia luokkia ole saatu kerättyä suunnilleen yhtä paljon. Tällöin tämä aineiston epätasainen jakauma tulee ottaa huomioon mallin koulutuksessa. Pienet erot määrissä eivät haittaa, mutta mikäli tiettyä tunnistetta on huomattavasti enemmän kuin muita, malli todennäköi-

sesti ylisovittaa itsensä tätä tunnistetta kohti. Tällöin mallin tarkastelu tarkkuuden kannalta voi vaikuttaa hyvältä, mutta tositilanteessa se ei pysty tunnistamaan muita kuin tuon yhden tunnisteen. Tämän vuoksi mallin suorituskykyä tulee mitata useammilla mittareilla. (Nabi 2018.)

2.8.4 Ominaisuuksien suunnitteluprosessi

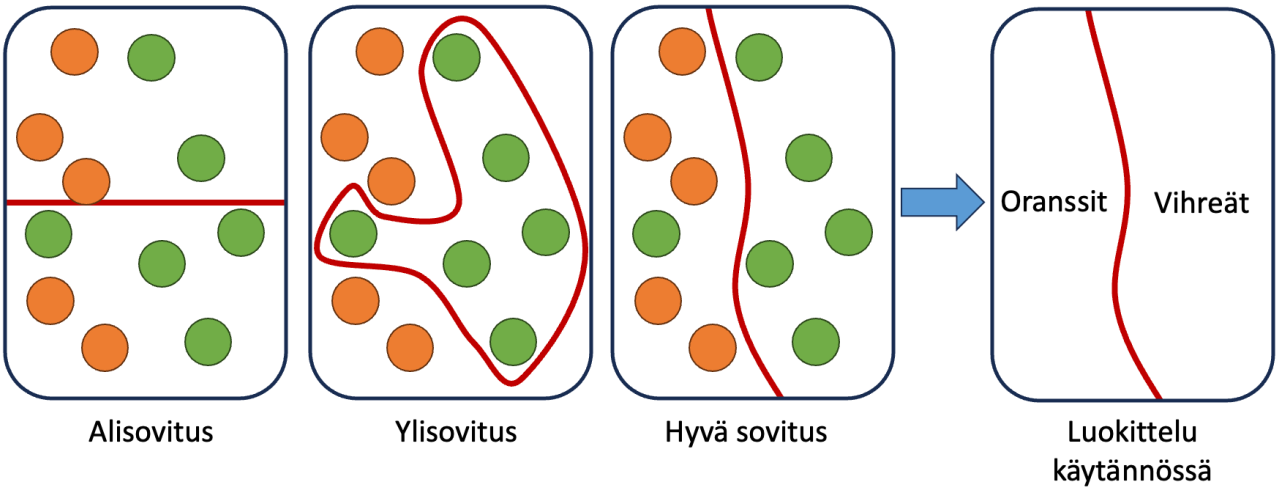
Esikäsittelyn rinnalla kulkeva tehtävä on ominaisuuksien suunnittelu. Tässä vaiheessa ominaisuuksia joko tuotetaan, muunnetaan, valitaan, analysoidaan, arvioidaan ja / tai erotellaan (Dong & Liu 2018, 3). Tärkeimpiä työn tavoitetta silmällä pitäen ovat ominaisuuksien muuntaminen, valinta ja analysointi.

Yksi tapa hoitaa tyhjien tietueiden täyttö on ottaa ominaisuuden kaikista numeerisista arvoista keskiarvo, joka sijoitetaan tyhjien tietueiden paikalle. Tällöin se ei vaikuta kovin paljoa mallin käytökseen. Mikäli taulukkomuotoisessa tietoaineistossa on sarakkeita, joiden tietoja emme halua koulutusmateriaaliin, voimme pudottaa ne pois. Myös eri ominaisuuksien yhdistäminen uudeksi voi olla joissain tapauksissa järkevää. (Omardonia 2023.) Laadukkaan tietoaineiston avulla malli kouluttuu tarkemmaksi ja se tekee vähemmän virheitä ennustuksissaan. Esikäsittely sekä ominaisuuksien muuntaminen on mahdollista integroida mallin suoritusketjuun, jolloin uudet tietueet käyvät läpi saman prosessoinnin suorituskyvyn maksimoimiseksi.

2.8.5 Koulutus, validointi- ja testausaineisto

Tietoaineisto on tapana jakaa vähintäänkin kahteen osaan, koulutus- ja testausaineistoon. Monesti näiden jakosuhde on vakiona 80 / 20. Syy aineiston osiin jakamiseen on se, että mallin suorituskykyä voidaan testata koulutuksen jälkeen aineistolla, jota se ei ole koulutuksen aikana nähnyt. Muuten mallin kohdalla on mahdollisuus yli- tai alisovitukseen. (Prepare data for building a model 2023.)

Microsoftin artikkelissa on oiva vertauskuva aiheeseen liittyen: ylisovitusta vastaisi se, että opiskelija tietää tentin kysymykset ja vastaukset etukäteen, ja saisi täten siitä täydet pisteet. Mutta mikäli kysymyksiä hieman muutettaisi, pisteytys romahtaisi. Eli opiskelija olisi oppinut vastaukset ulkoa. Alisovituksen ollessa kyseessä tilanne menisi niin, ettei opiskelijalla ole alkujaan tarpeeksi materiaalia opiskella tentin kysymyksiä varten. (Prepare data for building a model 2023.) Kuvion 11 tapauksessa hyvin sovitettu malli löytää pallojen väreistä trendin mutta jättää satunnaiset poikkeamat huomioimatta.



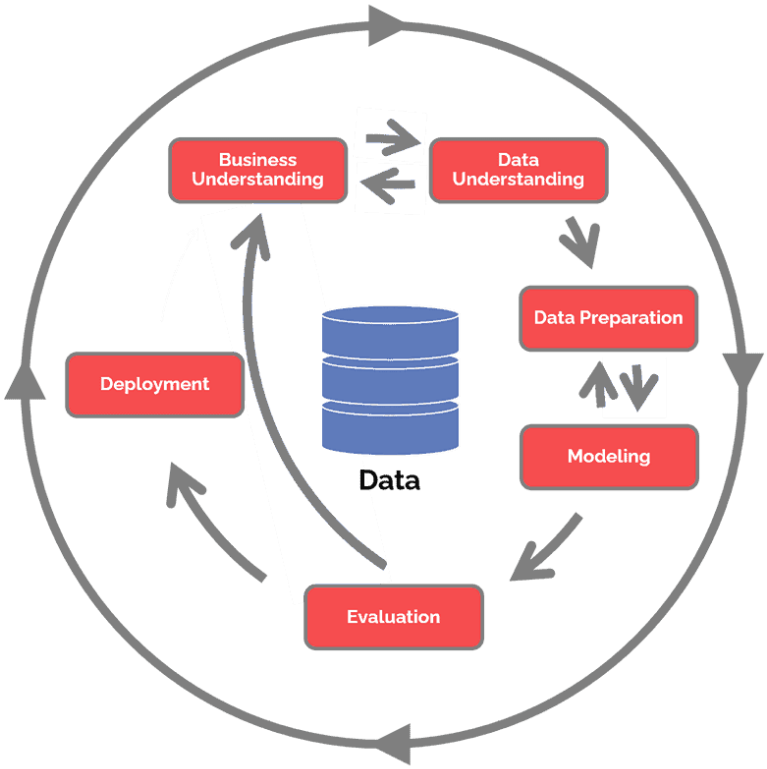
Kuvio 11. Luokittelun esimerkki erivärisillä palloilla

Joidenkin mallien kohdalla on mahdollista käyttää myös validointiaineistoa. Se on osa koulutusaineistosta, jota ei suoraan käytetä koulutukseen, vaan koulutuksen aikana mallin hyperparametrien säätämiseen. Tämä eroaa testausaineistosta siinä, että testiaineistoon malli ei pääse koulutuksen aikaan käsiksi, kun taas validointia käytetään koulutuksessa. Vasta kun malli on koulutettu, sen suorituskykyä testataan tietueilla, joita se ei ole aiemmin nähnyt. (Brownlee 2020b.)

2.8.6 CRISP-DM eli standardiprosessi tiedonlouhinnan toimialoille

Yleinen datatieteilijöiden käyttämä malli liiketoiminnan tehostamiseksi on CRISP-DM, joka tulee sanoista Cross-industry standard process for data mining. Kuviossa 12 tietoaineisto on kaiken keskipisteenä, ja nuolet näyttävät kuinka prosessissa lähtökohtaisesti edetään, mutta mikäli tieteilijä kokee, ettei tietylle vaiheelle ole tarvetta, voi sen ohittaa. Prosessi aloitetaan usein liiketoiminnan sekä aineiston ymmärtämisestä.

Liiketoimintaan liittyvän ongelman tunnistamiseksi ja ratkaisemiseksi toimialan henkilöillä on paras tietämys omasta aineistostaan, mutta tieteilijöillä on paras ammattitaito sen hyödyntämiseen. Mikäli aineiston avulla on mahdollista saada ratkaisu ongelmaan, siirtyy tieteilijä aineiston käsittelyyn ja mallien tekoon. (Kelleher ja Tierney 2018, 56–60.)



Kuvio 12. CRISP-DM malli. (Lähde: Data Science Process Alliance)

Aineiston käsittelystä on kerrottu aiemmissa kappaleissa. Mallinnuksen aikana aineiston avulla koulutetaan erilaisia koneoppimismalleja testaten eri algoritmeja tavoitteena löytää parhaiten suoriutuva malli. Tässä vaiheessa on hankala tietää mikä algoritmi suoriutuu tehtävästä parhaiten ja aineistosta voi löytyä jotain odottamatonta, joten yleensä useampi iteraatio koulutuksessa on tarpeen. Koulutetun mallin avulla voidaan lopuksi tehdä ohjelmisto yrityksen käyttöön liiketoimintaa tehostamaan. (Kelleher ja Tierney 2018, 60–62.)

Lopulta mallin toimintaa arvioidaan ja se otetaan käyttöön yrityksessä. Mallit, jotka istuvat yrityksen nykyisiin käytäntöihin ja infrastruktuuriin ovat hyviä toiminnan tehostamista ajatellen. Kuviossa 12 ulommaiset nuolet tarkoittavat prosessin olevan iteratiivinen, joten mallin suorituskykyä tulee tarkkailla jatkossa. Esimerkiksi kunnossapitoa varten tehdyn mallin kohdalla koneessa olevan sensorin vaihto uudempaan malliin saattaa pilata suorituskyvyn täysin, kun se saa sille opetetusta poikkeavaa aineistoa käyttöönsä. (Kelleher ja Tierney 2018, 62–64). Tämä suorituskyvyn tarkkailuvälin tarve vaihtelee käyttökohteittain, joten se olisi hyvä suunnitella ennen käyttöönottoa.

2.9 Koneoppimismallien musta laatikko

Uusia malleja kouluttaessa olisi hyvä koettaa selvittää miten ne päätyvät johtopäätöksiinsä. Malleille annetaan koulutusaineisto ja tavoitteet, jonka myötä ne oppivat itsenäisesti parametrit, joilla päästään toivottuun tulokseen. Mutta tämä välivaihe on koneoppimisen musta laatikko; ei ole helppoa keinoa selvittää miksi jokin malli päätyy tekemiinsä ratkaisuihin.

Mustalla laatikolla tarkoitetaan koneoppimismallien monimutkaisuutta siihen pisteeseen, ettei edes niiden kehittäjät voi olla varmoja niiden toiminnasta. Suurissa neuroverkoissa solmukohtia voi olla miljoonia, jolloin kaikkien yhteyksien ymmärtäminen menee pitkälti mahdottomaksi. (Zewe 2023b).

Esimerkkinä tästä tutkimus lääketieteen puolelta. Tutkijat kouluttivat neuroverkkoa tunnistamaan ihosyöpiä kuvista. Neuroverkko pääsi hyvälle tasolle erilaisten ihovaurioiden tunnistamisessa. Valitettavasti lähempi tarkastelu paljasti, että malli käytti kuva-aineistossa olevia viivoittimia yhtenä parametrinä sairauksien todennäköisyydelle, sillä lääketieteellisissä syöpämuunnoksia sisältävissä kuvissa on usein viivoitin tuomassa mittakaavaa. Tämän myötä malli oppi tunnistamaan viivoittimen läheisyyden pahanlaatuisen ihovaurion merkkinä erilaisten ihomuutosten sijaan. Tällöin tosimaailmassa mallin suorituskyky jäisi huomattavasti heikommaksi, sillä malli teki päätökset aiheelle merkityksettömien ominaisuuksien mukaan. (When AI flags the ruler... 2021.)

Tämän työn kohdalla mallin päätökset eri toleransseihin päätymisestä voi jäädä pimentoon. Tätä laatikkoa voidaan yrittää avata selvittämällä ominaisuuksien tärkeyttä mallin suorituskykyä varten permutaation avulla. Riippuen käytettävästä alustasta on mahdollista tarkkailla mallin päätöksiä visuaalisesti siihen myös tarkoitettujen työkalujen, kuten Netronin, avulla.

3 Koneensuunnittelu

3.1 Suunnitteluprosessi lyhyesti

Suunnitteluprosessi alkaa yleensä määrittelemällä millaiseen ongelmaan tai tarkoitukseen tuotetta suunnitellaan. Erilaisia tuotekehitysprosesseja sekä -tapoja on runsaasti, mutta niitä ei ole tarkoitus käydä kovin tarkasti tässä työssä läpi. Prosessissa usein edetään konseptin kautta alustaviin suunnitelmiin, josta edetään tarkempaan suunnitteluun. Ennen valmista tuotetta on tehtävä useita prototyyppejä sekä paljon erilaista testausta.

Tällainen prosessi on kuitenkin aikaa vievä etenkin suurien tuotteiden kohdalla. Domun (2016) esittää artikkelissaan oivana esimerkkinä Airbusin 320 tuoteperheen. Koska uuden lentokoneen suunnittelu tyhjältä pohjalta kestää jopa vuosikymmeniä, on järkevämpää valmistaa vanhojen pohjalta uusia ominaisuuksia sisältäviä päivitettyjä malleja kuin aloittaa aina alusta.

Kuten Barber, Sole & Turner (n.d, 182) toteavat paperissaan, useat nykyajan suunnittelutehtävät 6eivät olekaan täysin uusien tuotteiden tai keksintöjen keksimistä vaan pikemminkin aikaisempien parantamista ja niiden pohjalta innovointia. Tämän vuoksi prosessissa tukeudutaan vanhoihin malleihin ja piirustuksiin, kun tarkastetaan, miten asiat on aiemmin tehty. Tämän prosessin tehostaminen on aina tavoiteltavaa, ja sen vuoksi työssä testataan koneoppimisen mahdollisuuksia piirustusympäristössä.

3.2 Suunnitelmasta toteutukseen

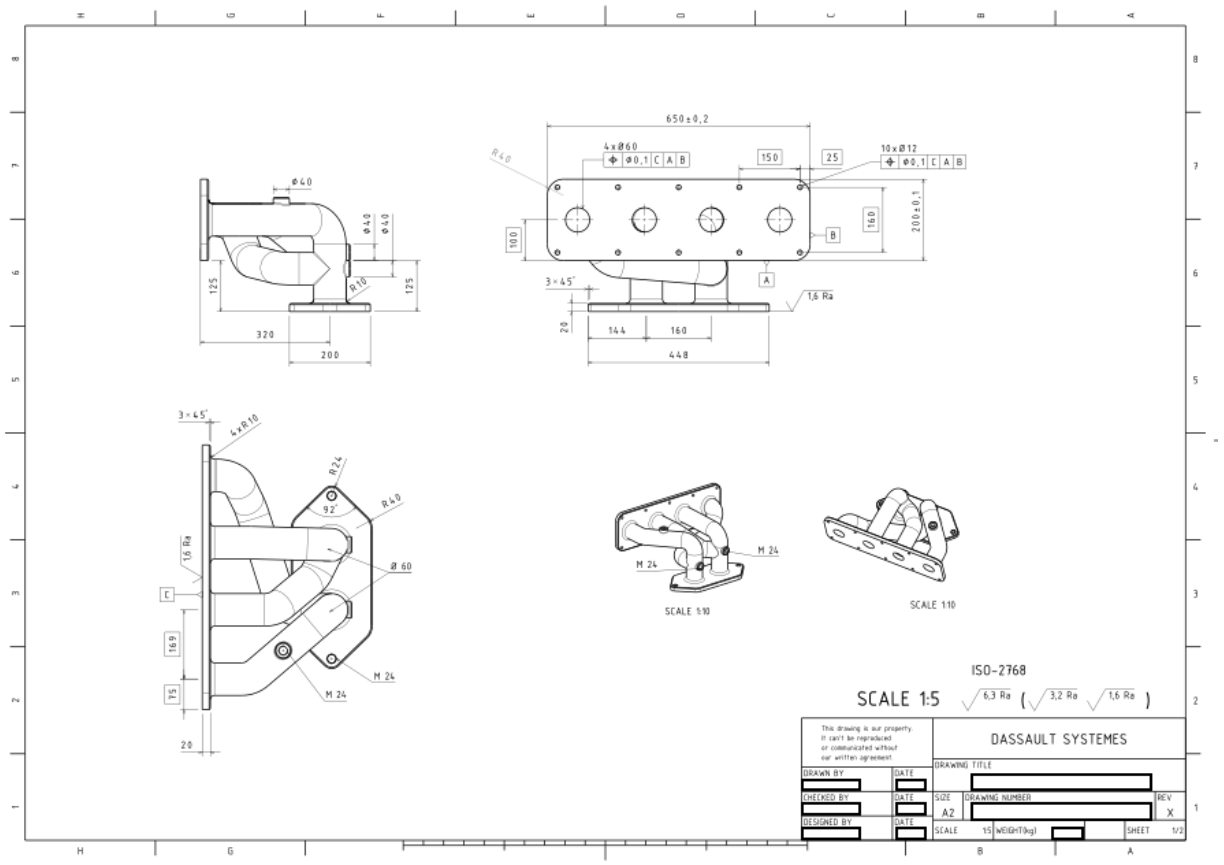
Koneensuunnittelussa tuotteita mallinnetaan nykyisin CAD-ohjelmistojen avulla. Näistä tehdään usein koneenpiirustuksia, joiden avulla tuotteet valmistetaan. Piirustuksiin tuodaan vähintään yksi kuvanto mallinnetusta kappaleesta tai tuotteesta ja siihen lisätään erilaisia mittoja, toleransseja sekä mahdollisesti muita ohjeistuksia valmistusta varten. Suomessa piirustuksia tuotetaan eniten ISO standardin mukaisesti.

Koneensuunnittelussa käytetään työstettävien työkappaleiden ominaisuuksien kuvaamiseen geometrista tuotemäärittelyä, ISO GPS-järjestelmää. Koska olemme rajanneet työn koskemaan pituusmittoja, käytämme työssä tukena SFS-EN ISO 286-1 sekä SFS-EN ISO 286-2 standardeja. Ensimmäinen koskee enemmän ISO-merkintäjärjestelmän toleransseja, sovitteita sekä eromittoja, kun taas jälkimmäisestä löytyvät reikien ja akseleiden perustoleranssiluokat sekä rajaeromitat taulukoituna.

3.3 Koneenpiirustus

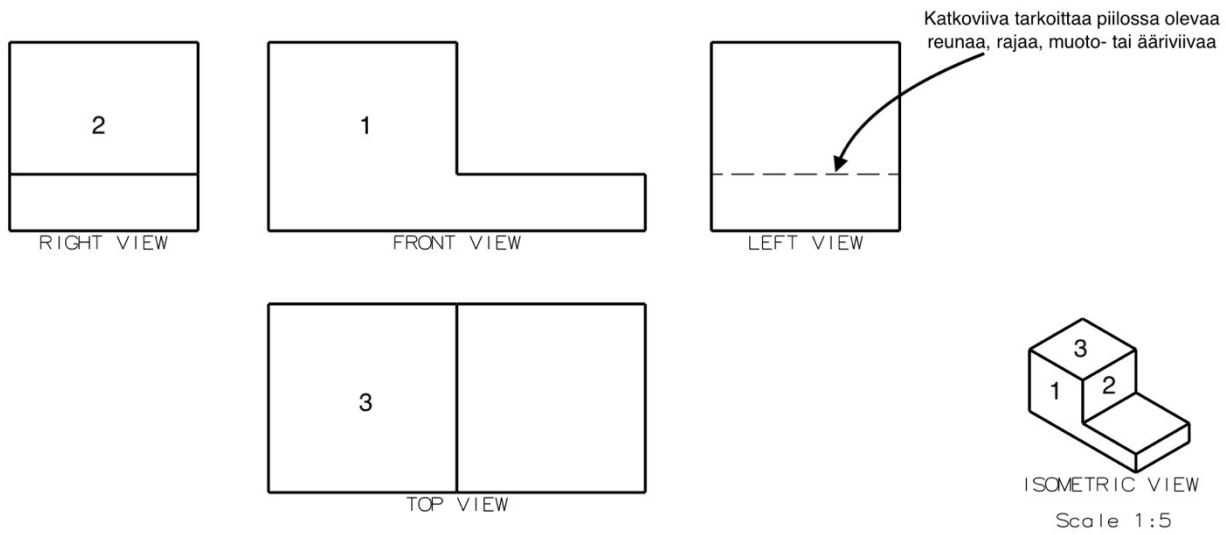
Koneenpiirustuksilla on tarkoitus esittää erilaisia kappaleita yksiselitteisesti ja täsmällisesti. Niitä on monenlaisia: kokoonpanokuvissa voidaan näyttää tuote ja siihen kuuluvat osat, kun jokaiselle osalle on taasen valmistusta varten työkuvat. Työkuvat sisältävät kappaleista riippuen erilaisia tietoja, kuten projektioita, leikkauksia, pinta- ja hitsausmerkkejä, geometrisiä toleransseja, mittoja ja mittakaavoja. Kuviossa 13 ovat esimerkkeinä otsikkotaulun yläpuolella esillä skaala, yleistoleranssi sekä pinnanlaatuun viittaavat merkinnät.

Piirustuksia laatiessa tekijän tulee ymmärtää millaisilla valmistustekniikoilla kappale voidaan valmistaa. Myös eri käyttötarkoituksille on omia piirustuksia: kuljetusta, pakkausta, erilaisia lastuavan työstömenetelmän työvaiheita varten tai käyttäjille käyttöohjeeksi. (Pere 2021, luvut 1.1–1.4).



Kuvio 13. Esimerkki pakosarjan piirustuksesta

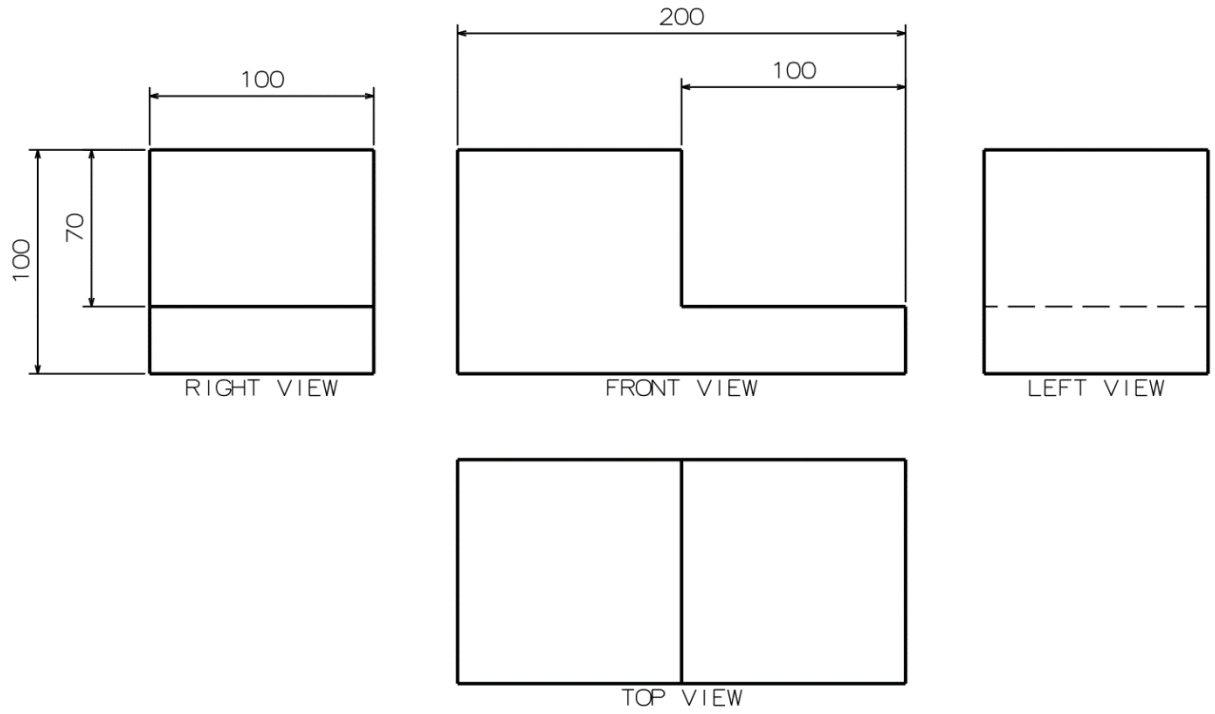
Piirustuksissa koetetaan kuvata kaksiulotteisen kuvan avulla kolmiulotteista kappaletta. Tämän vuoksi kappale kuvataan projektioiden avulla useasta kulmasta, josta karkea esimerkki kuviossa 14. Keskellä on pääprojektio 1, josta muut projektiot on kuvannettu. Kuviossa on käytössä yhden käännön menetelmä, mikä tarkoittaa, että jokaista projektiota varten kappaletta on käännetty 90°. Pääprojektioksi valitaan kappaleesta sellainen projektio, mikä antaa siitä eniten tietoa. (Pere 2021, luvut 4.1–4.5).



Kuvio 14. Yksinkertaisesta kappaleesta projektioita, numeromerkinnät lisätty selvittämään mistä suunnasta projektio on

Työkuvissa projektioihin lisätään kappaleesta olennaisia mittoja, joiden avulla se voidaan valmistaa. Koneenpiirustuksessa nämä mitat ovat yleisimmin millimetrejä. On huomioitavaa, että mittojen ketjutusta koko kappaleen matkalta tulisi välttää ja sinne tulisi jättää valmistuksen epätarkkuuksien vuoksi mahdollisuus vaihtelulle (Valtanen 2019, 513). Kuviossa 15 on pääprojektiolle annettu mitat 200 mm koko kappaleen pituudeksi ja 100 mm loven pituudeksi. Tämä tarkoittaa, että tämän kappaleen kohdalla loven mitta on tärkeämpi kuin jäljelle jäävän osan, jolloin jäljelle jäävään osaan saa kohdistua valmistusepätarkkuudet. Piirustukseen tulee kappaleen lopulliset mitat kaikkine tarvittavine tietoineen (Pere 2021, luku 7.1). Piirustuksista tulee löytyä tuotteen tai kappaleen toiminnan ja valmistuksen kannalta olennaiset ominaisuudet, joita on lueteltuna alla:

- Projektiot
- Päämitat
- Muut mitat
- Materiaali
- Pintamerkit
- Geometriset toleranssit
- Sovitteet



Kuvio 15. Edelliseen esimerkkiin lisättyjä mittoja

Työkuvien perimmäinen tarkoitus on antaa valmistajalle tarvittavat tiedot kappaleiden tai tuotteiden valmistusta varten. Piirustuksissa noudatetaan kansainvälisiä standardeja, jotta valmistus onnistuu missä vain samoilla säännöillä. Suunnitteluprosessin tarkempi läpikäynti ei kuulu tämän työn piiriin, vaan jatkossa pääpainona on mitoitus, toleranssit sekä niihin liittyvän prosessin tehostamiseen.

3.4 Toleranssit

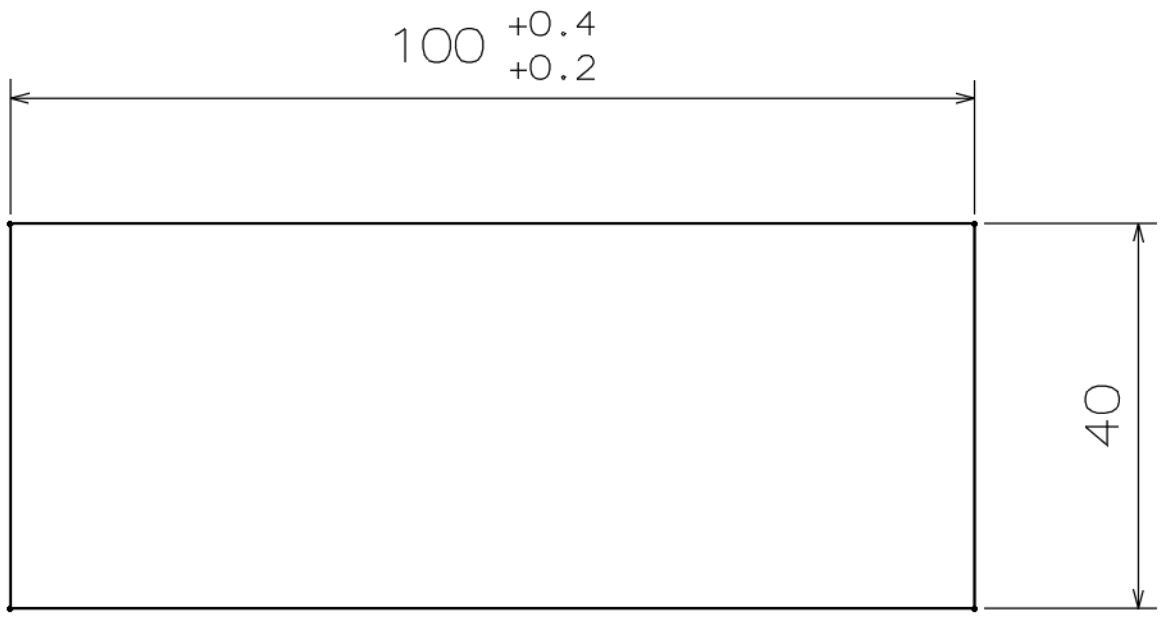
Koneensuunnittelussa toleranssit määräävät rajoja tuotteen teknisille ominaisuuksille. On olemassa geometrisia toleransseja, mittatoleransseja sekä yleistoleransseja. Geometrisillä toleransseilla rajoitetaan elementtien poikkeamia ihanteellisesta. Esimerkiksi tilanteet, missä kahden pinnan tulee olla yhdensuuntaisia tai kappaleiden olla kohtisuorassa toisiaan kohti. Mittatoleranssit ilmaisevat mitan poikkeaman ylä- ja alapäätä. ISO 2768- yleistoleranssin avulla on standardisoitu yleisesti toleranssien määrittely teknisen piirustuksen osalta, jottei jokaiseen mittaan tarvitse erikseen asettaa toleranssia valmistuksen epätarkkuuksien vuoksi. ISO 2768 sisältää lisäksi neljä toleranssiluokkaa f, m, c ja v, mitkä määräävät toleranssien tarkkuutta. Tässä työssä keskitytään mittatoleransseihin pituus- ja halkaisijamitoille.

Pituusmitoissa toleranssit vaihtelevat monen ominaisuuden mukaan, olennaisimpina mitan pituus sekä kuinka tarkka toleranssi sille halutaan asettaa, ottaen huomioon toiminnallisuus ja käytettävät valmistusmenetelmät. Sylinterin muotoisille elementeille on olemassa omat taulukot sekä

rei'ille että akselimaisille kappaleille. Näihin on työtä ajatellen mahdollista hankkia tiedot käyttäen esimerkiksi Catiaa, etsimällä SFS: sivuilta ISO 286 mukaiset toleranssit tai käyttämällä Valtasen taulukkokirjaa hyödyksi, sillä toleranssit ovat taulukoitu näihin kaikkiin valmiiksi.

Työkappaleille annetaan valmistusta varten nimellismitta ja toleranssi. Nimellismitta tarkoittaa elementin virheetöntä mittaa, mutta valmistusmenetelmien vuoksi reaalimaailmassa on mahdotonta valmistaa absoluuttisesti tarkkoja kappaleita. Valmistusmenetelmien tarkkuus korreloi hinnan kanssa, joten mitä tarkemmat toleranssit kappaleelle määritellään, sitä kalliimpaa on sen valmistus ja tarkistus. Kaikille työkappaleille ei ole tarvetta määritellä tarkkaa toleranssia niiden toiminnallisuuden vuoksi, mutta esimerkiksi sovitteiden kohdalla kappaleen mittojen tulee olla annettujen raja-arvojen välissä, jotta sovite toimii.

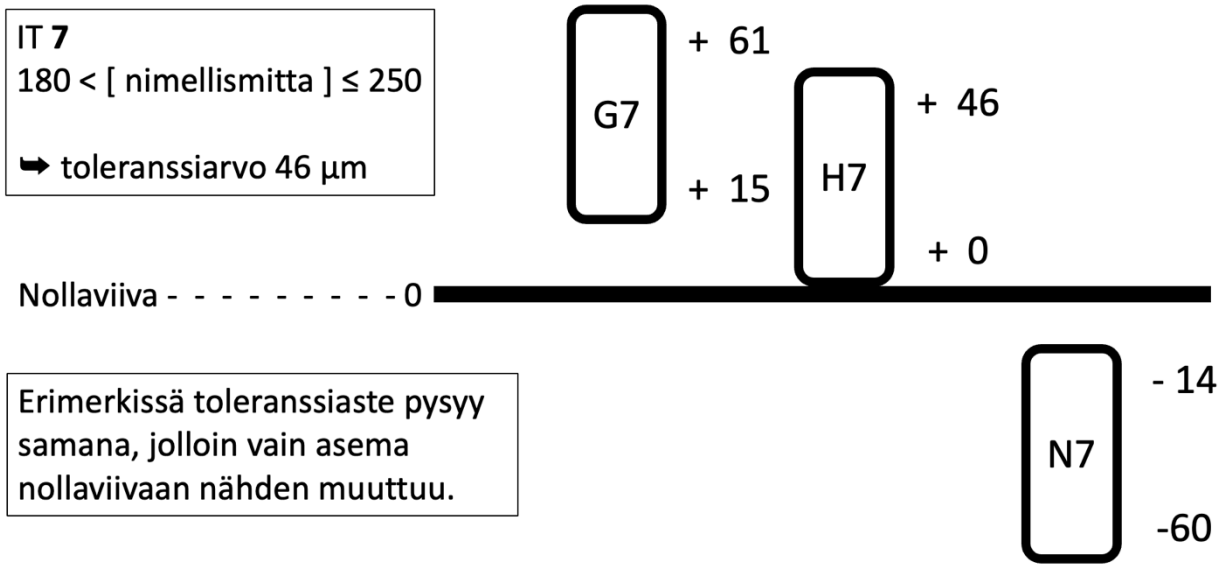
Pituusmitta on määritelty tarkoittamaan kahden vastakkaisen yhdensuuntaisen pinnan väliä. Pinnat voivat olla joko vastakkain tai poissuunnattuja toisiinsa nähden. Kuvion 16 esimerkissä vaakasuuntainen poissuunnattu mitta on määritelty 100 millimetriksi, ja pystysuuntainen 40 millimetriksi. Vaakasuuntaiselle mitalle on annettu toleranssiväli, minkä alarajamitta on +0,2 mm ja ylärajamitta +0,4 mm. Tämä tarkoittaa, että valmiin valmistetun työkappaleen vaakamitan tulisi olla 100,2 mm ja 100,4 mm väliltä. Piirustuksiin voidaan lisätä merkintä yleisille toleransseille valmistusta ajatellen työtapakohtainen ISO 2768 otsikkotauluun, jolloin piirustuksen jokaista mittaa ei ole pakko toleroida.



Kuvio 16. Esimerkki pituusmittatoleranssista

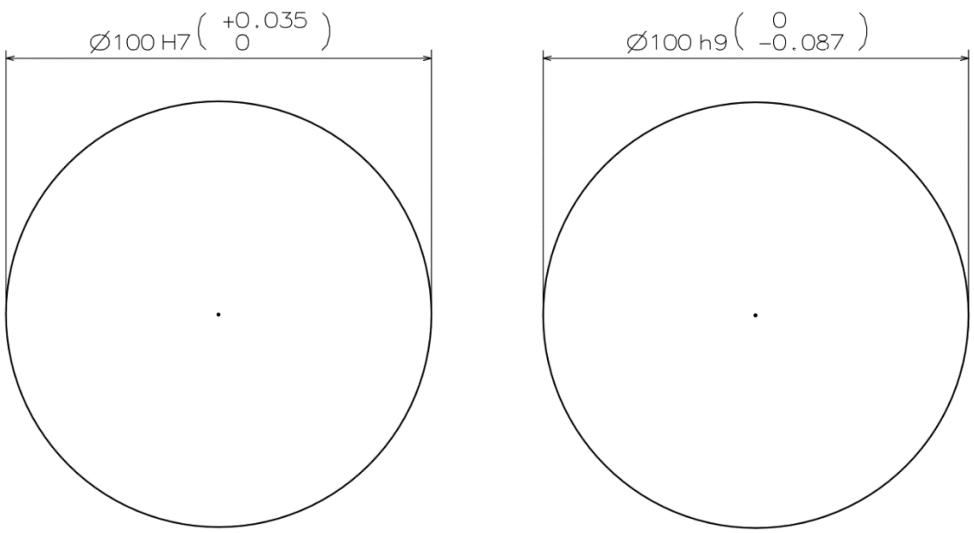
Toleranssit sylinterimäisille halkaisijamitoille on lajiteltu reikä- ja akselimaisille kappaleille erikseen. Nämä toleranssit koostuvat kirjaimesta sekä toleranssiluokasta. Alkukirjain määrittelee peruseromitan ja numero perustoleranssiasteen (SFS-EN ISO 268-1, 34). Reikätoleranssit on kerrottu

isolla alkukirjaimella ja akselitoleranssit pienellä. Perustoleranssiasteita on 20 ja ne vaihtelevat IT01–IT18 välillä, missä pienempi numero tarkoittaa tarkempaa toleranssia. Toleranssin suuruus on ylä- ja alaeromitan erotus. Kirjain toleranssissa tarkoittaa toleranssialueen asemaa niin sanottuun nollaviivaan nähden. Sekä rei'ille että akseleille löytyy standardeista omat taulukot, jotta niitä ei tarvitse osata ulkoa. Alla olevassa kuviossa 17 on havainnollistettu visuaalisesti kolmen eri toleranssin asemaa nollaviivaan nähden.



Kuvio 17. Esimerkki toleranssialueiden asemista

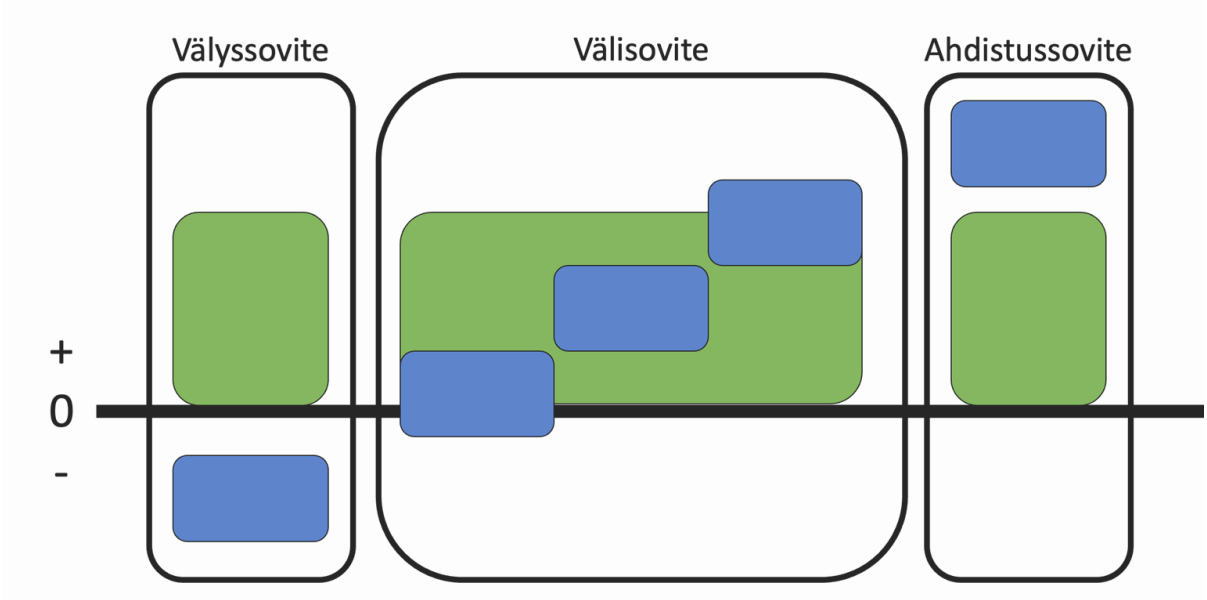
Toleransseilla on useita merkintötapoja (Valtanen 2019, 572). Toimeksiantajayrityksessä on käytössä näistä Valtasen mukaan suositeltavin, eli toleranssit näytetään järjestyksessä nimellismitta, toleranssiluokka sekä ylempi- sekä alempi rajaeromitta suluissa toleranssiluokan jälkeen. Tällainen merkitsemistapa on tuotteen suunnittelijaa sekä valmistajaa ajatellen paras, sillä kaikki tarvittava informaatio molemmille on selkeästi esillä. Alla olevassa kuviossa 18 näkyy Catian esitys kahdesta halkaisijamitasta ja niiden toleransseista. Catiassa ei ole ohjelmallisesti mahdollisuutta tarkistaa, onko kyseessä reikä- vai akselimitta, vaan molemmat näkyvät koodin puolella halkaisijana.



Kuvio 18. Vasemmalla on reikätoleranssi ja oikealla akselitoleranssi

3.5 Sovitteet

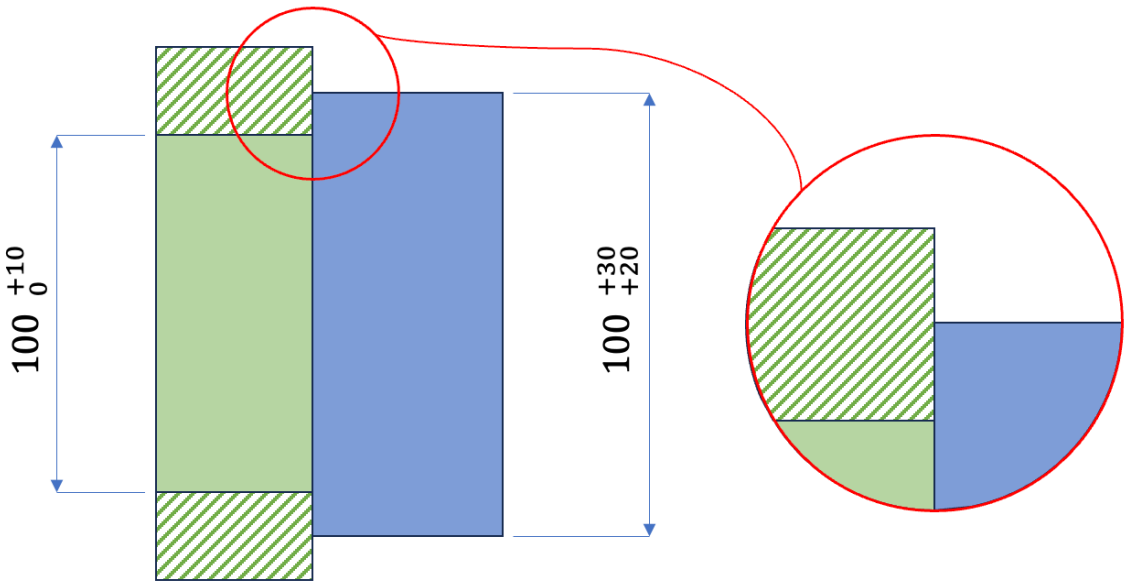
Sovite tarkoittaa kahden eri elementin mittojen eroja ennen kuin elementit liitetään toisiinsa. Esimerkiksi akselin ja laakerin välisen sovitteen edellytyksenä on, että molempien nimellismitat ovat samat, jotta sovitteen voi valmistaa (SFS-EN ISO 268-2 2010, 8). Koneensuunnittelussa yleisiä sovitteita ovat välys, väli- ja puristussovitteet, joita määritellään työkappaleisiin käyttökohteen mukaan. Näitä on havainnollistettu kuviossa 19. Kappaleisiin määritellyt toleranssialueet kertovat millainen sovite on kyseessä. Sovitteita voi laittaa kokemuksen mukaan tai laskemalla toiminnallisten vaatimusten mukaisesti.



Kuvio 19. Reikäkantajärjestelmän mukaisia sovitteita

Sovitteisiin käytetään joko akseli- tai reikäkantajärjestelmää. Reikäkantajärjestelmän sovitteiden asettumisesta esimerkki kuviossa 19, missä vihreällä pohjalla on reiän toleranssialue ja sinisellä akselin toleranssialue. (ISO-toleranssit n.d.)

Toiminnallisuudesta riippuen sovitteen luonne muuttuu. Mikäli akseli ja laakeri halutaan liittää tiukasti toisiinsa käyttäen sovitetta, voidaan liitoksessa käyttää ahdistussovitetta. Tällöin akselin halkaisija on hieman suurempi kuin reiän halkaisija, joten kappaleiden välille tulee ahdistus. Tällöin olisi kyseessä kuvion 20 mukainen ahdistussovite.



Kuvio 20. Ahdistussovite. Akseli sinisellä, reikää kuvaa vihreä alue. Viivoitettu alue tarkoittaa laakeria

3.6 Suunnittelija määrittelee kulut

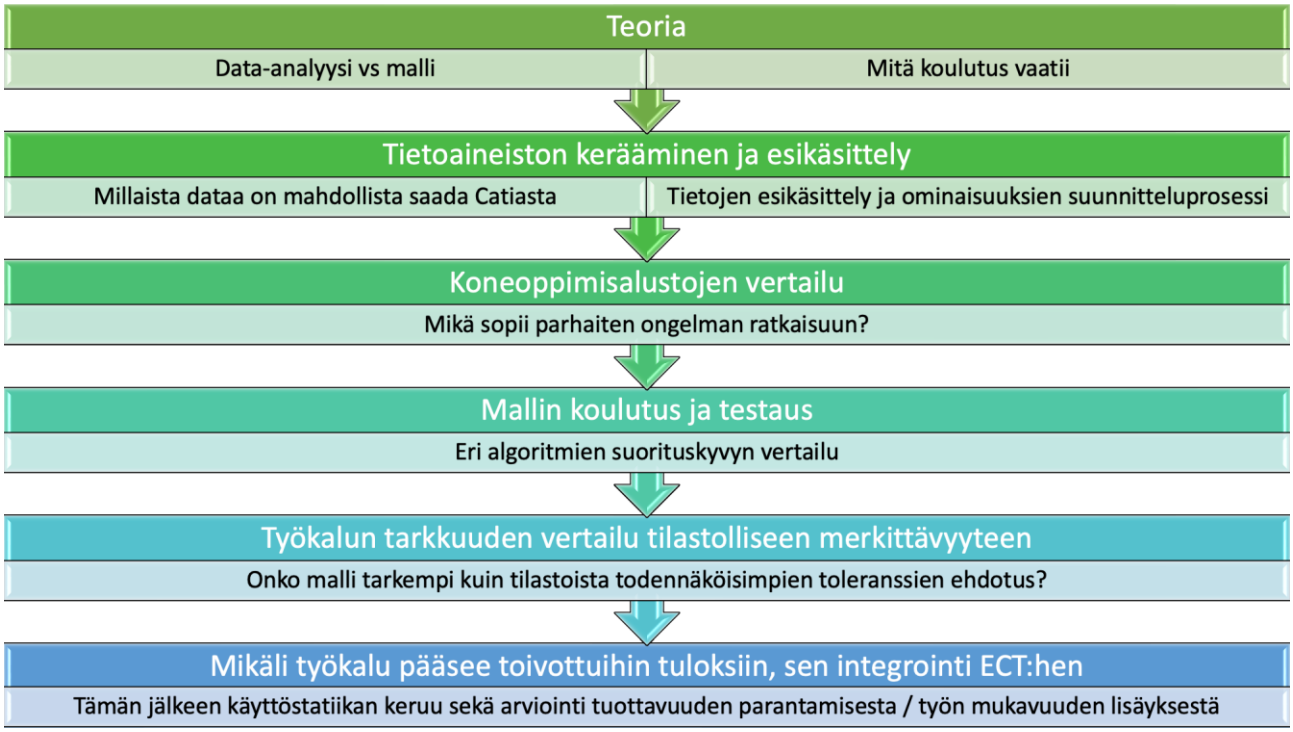
Monesti kerrotaan suunnittelijan päättävän 70–80 % tuotteen elinkaaren kuluista (Holden 2019, Whitney 1988). Näihin kuluihin lukeutuu monia eri asioita, joista esimerkkejä lueteltu alla. Idea tämän ajattelutavan taustalla on, että muutoksien teko jo valmistuksessa olevaan tuotteeseen on huomattavasti kalliimpaa tehdä kuin suunnittelun aikana. Lisäksi suunnittelun aikana tehdyt päätökset vaikuttavat myös tuotteen elinkaaren kuluihin. (Pearson & Ulrich 1993, 2–6.)

- Tuotteen materiaali
- Valmistustavat
- Miten tarkat toleranssit tuotteella on
- Millaisia kiinnikkeitä tuotteessa käytetään
- Kuinka monesta osasta tuote valmistetaan
- Millaista tuotteen kunnossapito on elinkaarta ajatellen

Mutta myös itse suunnittelusta tulee kuluja. Kauppalehden artikkelin (Aaltonen 2023) mukaan AMK taustaisen konetekniikan suunnittelijan palkka on vuositasolla keskimäärin 44 299 € ja diplomi-insinöörillä 50 139 €. Oletetaan esimerkkilaskelmassa insinöörin palkaksi 3500 €/kk, työnantajalle kulut ovat 1,6 kertoimella noin 5600 €/kk. Työviikko on 37,5 h, jolloin yhden tunnin odottelulle tulee hintaa hieman päälle 37,30 €. Kun tätä lukua kertoo suunnittelijoiden lukumäärällä ja miten usein odottelua tulee, summa kasvaa vuositasolla huomattavaksi. Kuluihin voi palkan lisäksi laskea myös eri suunnitteluohjelmistojen lisenssit, työasemat ja muut tarvittavat välineet työn tekoon. Mikäli työkalun saa tarpeeksi tarkaksi, voi työajan tehostamisesta saada vuositasolla varteenotettavia säästöjä.

4 Työn toteutus

Työkalun suunnittelu ja laatiminen on moniosainen prosessi, mikä eteni työssä jokseenkin kuvion 21 mukaisesti. Koneoppimisen ollessa tuntematon rajaseutu konetekniikan taustalla, ensimmäisenä tuli hankkia kattava teoriapohja työn kannalta.



Kuvio 21. Työkalun valmistuksen prosessi

4.1 Tietoaineiston kerääminen

Ennen koneoppimismallin koulutusta tulee kysyä, tarvitseeko ongelmaan kouluttaa mallia vai selviääkö ongelma perinteistä data-analyysia käyttäen. Joskus ongelmaan saa ratkaisun selvittämällä tietoainestoa ja analysoimalla sitä. Koneoppimisen hyvä puoli on, että se voi löytää suurista tietoaineistoista korrelaatioita sekä malleja, joita ihminen ei kykenisi hahmottamaan.

Tämän työn kohdalla edellä olevaan kysymykseen vastaaminen vaatii vähintäänkin tietoaineiston hankkimisen tarpeeksi pitkältä aikaväliltä. Koska koneoppiminen perustuu menneisyyden tietoaineistosta oppimiseen, tarvitsee toleranssien hankintaa varten kaapia paljon tietoa vanhoista piirustuksista. Usean koneoppimismallin kohdalla suorituskyky paranee sen mukaan, mitä enemmän aineistoa niillä on käytettävissä. Tällainen kaapimistyö olisi käsivoimin todella aikaa vievää sekä erilaisten virheiden mahdollisuus olisi suuri, joten kaavintaan etsitään ratkaisua Catian API:n kautta.

4.1.1 Koulutus- ja testausaineiston kaavinta

Catian dokumentoinnista löytyy tarvittavat menetelmät Drafting-työtilan tietojen kaavintaan, joten tätä kaavintaa varten luodaan oma ohjelmansa. Kaavitut tiedot tallennetaan SQL tietokantaan. Koulutusaineistona työkalulle on tarkoitus kaapia tiedot Catian PLM:stä löytyvistä vanhoista piirustuksista, mitkä on asetettu ”Released” tilaan. Syy tälle on, että tällöin piirustuksista löytyvät mitat ja toleranssit ovat tarkastettuja ja hyväksyttyjä, jolloin minimoidaan virheellisen aineiston keruun riski. Piirustuksista saisi kaavittua runsaasti tietoa, mutta alla lueteltuna mallin koulutusta varten olennaisimmat:

- Project, eli Catian yhteistyötila, missä piirustukset ovat.
- Title, eli piirustuksen nimi.
- SheetName, eli piirustuksesta löytyvän arkin nimi.
- ViewScale, eli piirustuksen mittakaava.
- ViewType, eli mikä kuvanto on kyseessä.
- DimensionValue, eli mitan arvo.
- DimensionType, joka on mitan tyyppi. Esimerkiksi pituus- / halkaisijamitta.
- Toleranssien ylä- ja alamitat.

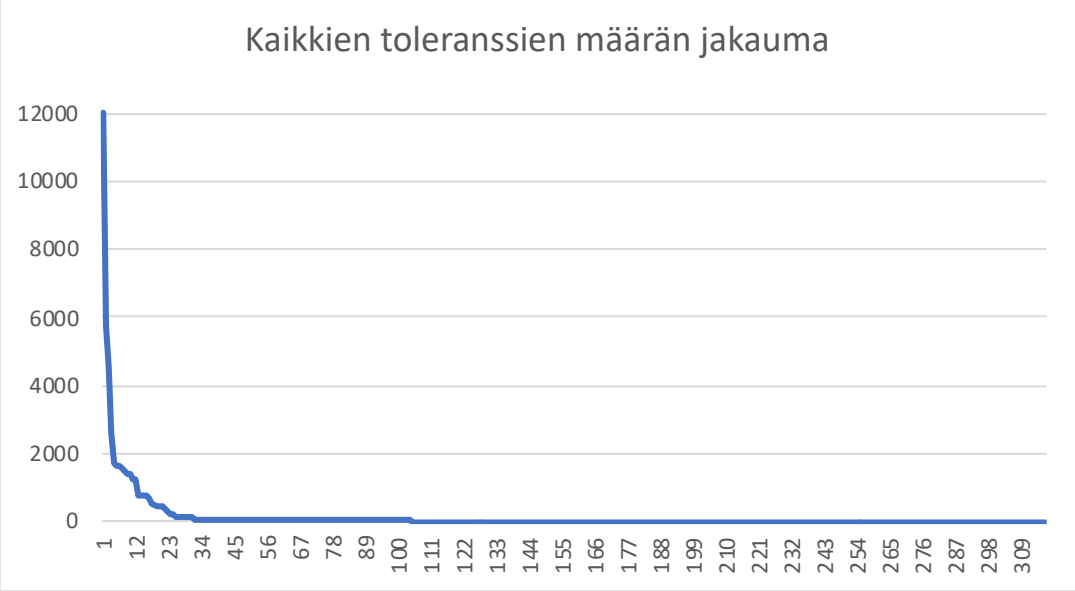
Nämä tiedot ovat saatavissa jokaiselle toleranssille, mitkä on asetettu mittoihin. Tietokannan avulla on mahdollista tarkastella eri toleranssien esiintymistä ja olisiko niistä mahdollista tehdä hyödyllistä työkalua ilman koneoppimistyökalua.

Työkalua varten kaavittiin tietoja kahdeksasta eri yhteistyötilasta. Toleroituja mittoja löytyi pituusmitoille vähän päälle 46 000, halkasijamitoille valitettavasti vain noin 2200. Tästä eteenpäin työssä esiintyvät kuviot, taulukot ja teksti viittaavat oletuksena pituusmittoihin, ellei muuten mainita. Halkaisijamitoille kaaviot ja tilastot noudattavat samanlaista linjaa mutta pienemmällä määrällä.

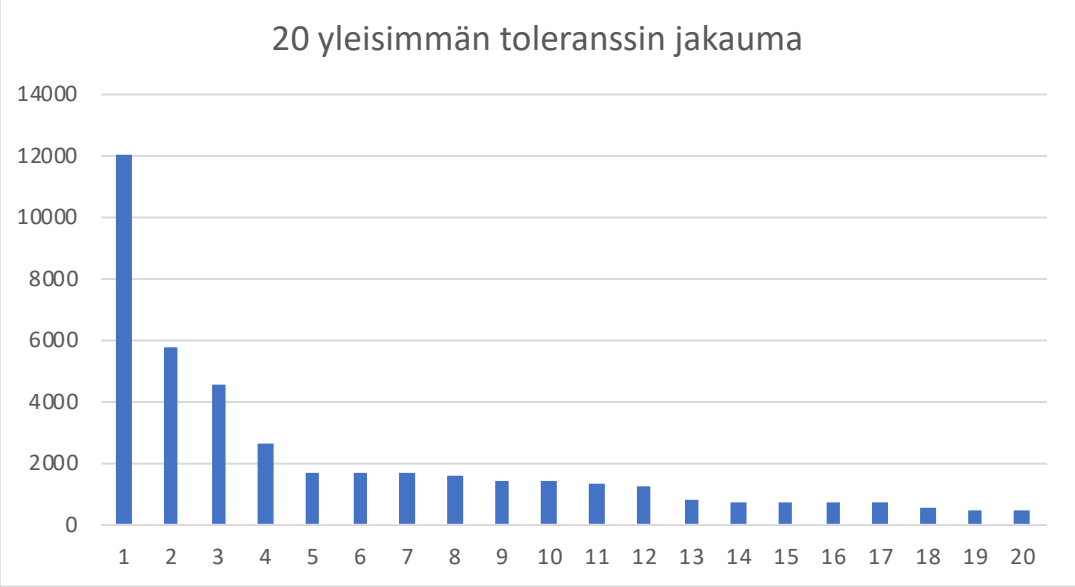
Neljä yhteistyötilaa sisälsivät alle tuhat tietuetta, joten ne jätettiin kuvion 22 vertailusta pois, sillä niiden määrät todennäköisesti aiheuttaisivat vain kohinaa mallin koulutuksessa heikentäen sen suorituskykyä. Kohinalla tarkoitetaan tässä tapauksessa ennalta-arvaamatonta tai satunnaista vaihtelua aineistossa, minkä vuoksi ne häiritsevät mallin tavoitetta löytää kaavoja tai yhteyksiä aineistosta. Esimerkiksi toleranssi, joka ilmenee aineistossa vain kerran aiheuttaa kohinaa, sillä yhden rivin pohjalta on mahdotonta löytää kaavaa tämän toleranssin käytölle. Lopullinen määrä toleroiduille mitoille oli 46 016 tietuetta. Kuten kuviosta 22 näkyy, tietyt toleranssit ovat

huomattavasti suuremmassa käytössä kuin toiset. Kuviossa 23 on havainnollistettu miten jyrkkä lasku yksittäisten toleranssien määrässä on jo kahdenkymmenen ensimmäisen kohdalla.

Jokaisesta yhteistyötilasta on kaavittu erilaiset toleranssiyhdistelmät (plus ja miinus toleranssi) ja jokaisesta ainutlaatuisesta parista on tehty oma merkkijononsa, mille on kohdistettu oma Id numero. Näin niistä on saatu helposti käsiteltäviä kenttiä. Halkaisijamittojen kohdalla tunnisteena toimii kirjain + numero yhdistelmä.

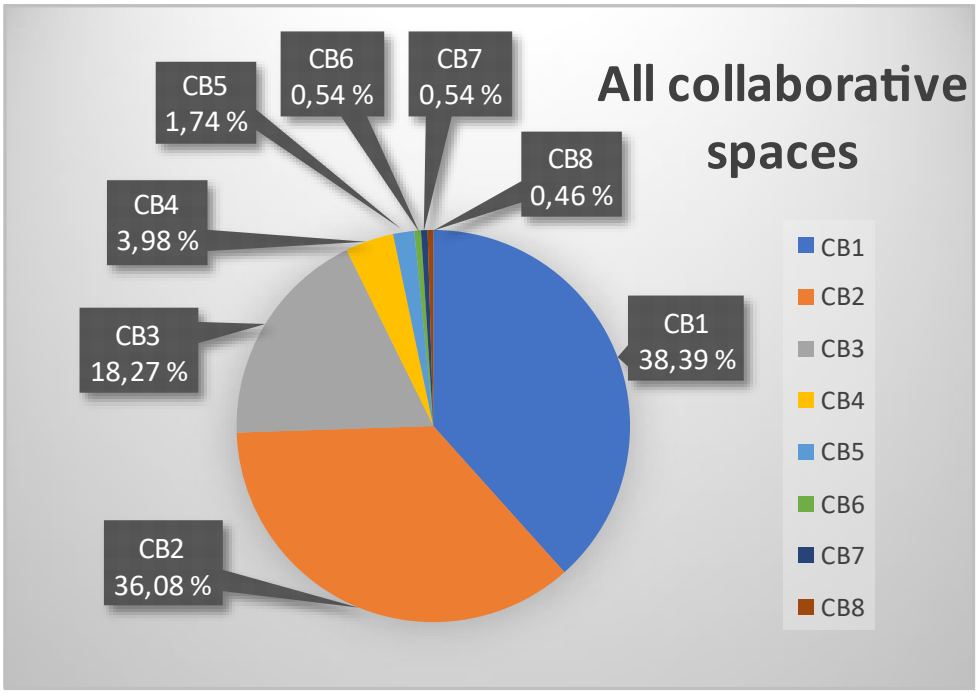


Kuvio 22. Kuvaaja yksittäisten pituustoleranssien lukumäärästä tietokannassa



Kuvio 23. Kun tarkastellaan 20 lukumäärältään suurinta pituustoleranssia, nähdään kuinka lukumäärät lähtevät nopeasti laskuun

Teoriassa kaavitusta tiedosta olisi mahdollista tehdä työkalu, joka ehdottaa käyttäjälle todennäköisimpiä toleransseja tietokannasta. Kolme yleisintä kaavittua toleranssia ovat 48,6 % koko kaavitusta tietoaineistosta. Täten koneoppimisella koulutetun työkalun tulee yltää tarkkuudessa vähintään samaan, jotta siitä olisi mitään hyötyä. Kuviossa 24 piirakkakaaviosta voi nähdä, kuinka eri yhteistyötilojen toleranssien määrät jakautuvat. Samoin suuri määrä toleransseja ilmenee aineistossa vain muutaman kerran, jolloin malli ei kykene löytämään kaavaa niille.



Kuvio 24. Toleranssien määrän jakautuminen eri yhteistyötiloissa

Kuten yllä olevasta kuviosta 24 voimme nähdä, tietoaineisto jakautuu epätasaisesti ja puolet kaavituista yhteistyötiloista sisältävät niin vähän tietoa, että ne kannattaa jättää kohinan muodostumisen vuoksi aineistosta pois. Myöhemmässä kappaleessa on testattu lopullisen mallin suorituskykyä kouluttaen toinen malli aineistoilla, missä on ollut kaikki yhteistyötilat mukana, minkä jälkeen vertailemalla tuloksia selviää tämän hypoteesin paikkansapitävyys.

Kerätty tietoaineisto on sanalla sanoen epäsuhtaista. Kaikissa kerätyissä ominaisuuksissa on lähes logaritmia vastaava pudotus havaittavissa. Tämä tietysti vaikuttaa omalta osaltaan koulutettavan mallin tarkkuuteen, mutta siitä lisää myöhemmin. Lisää kaavioita eri ominaisuuksien esiintymisestä on löydettävissä liitteestä kolme.

4.1.2 Standardinmukaiset toleranssit

Koulutus- ja testausaineiston lisäksi työkalu tarvitsee halkaisijamittoja varten standardien mukaiset toleranssit, mitkä on taulukoitu. Tarkoitus on kaapia tiedot jokaiselle toleranssiasteelle ja -alueelle, joista voi tehdä omat tiedostot, mitkä voi lukea nopeasti muistiin. Tällöin kaiken ei ole pakko

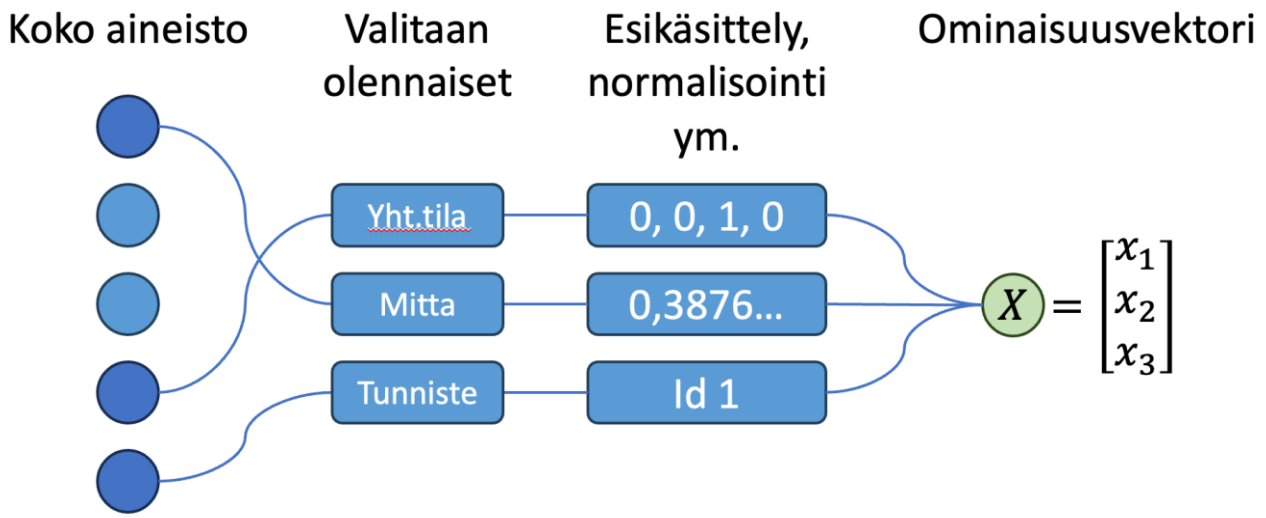
olla muistissa, sillä kuten tietoaineiston visualisoinnista on nähtävissä, suurin osa toleransseista on vain harvoin käytössä. Eli esimerkkinä, toleranssialueelle A tehdään tiedostot A9–A13, joissa on omat taulukot perusmitalle sekä ylä- ja alaeromitoille. Käyttäjän valitessa mittaan A10 toleranssi, ohjelma hakee tiedoston A10, vertaa valittua mittaa peruseromittaan ja asettaa tämän jälkeen oikeat toleranssit mittaan.

Toleranssitaulukoiden vertailuissa eri lähteiden välillä oli pieniä eroavaisuuksia, mistä tarkemmin kappaleessa 6.3. Pääasiassa erot olivat mitättömiä ja selittyivät inhimillisillä virheillä. Kerätty aineisto testattiin vielä ennen työkalun tuotantoon vientiä varmistamalla yksikkötestauksen avulla, että työkalun tiedostoista hakemat toleranssin vastasivat Catian asettamia samanlaisessa halkaisijamitassa.

4.2 Tietoaineiston käsittely

Kerätty tietoaineisto on tallennettu SQL tietokantaan, mistä Visual Studioon tehty ohjelma hakee aina ennen koulutusta uusimmat tiedot. Esikäsittelyä aineistolle tapahtuu jo tietokannan päässä, kun esimerkiksi tuumamitoituksella tehdyt toleranssit suodatetaan pois. Myös yhteistyötilat, missä on alle tuhat tietuetta, jätetään koulutuksen ulkopuolelle aineiston vähyyden vuoksi.

Visual Studiossa aineistoa tuodessa piirustusten nimet käsitellään omalla esikäsittelyohjelmalla, mikä karsii niistä turhia sanoja, merkkejä sekä yhtenäistää samankaltaisia mutta pienillä muutoksilla olevia nimikkeitä. Tämän jälkeen aineisto tuodaan ML.NET:n IDataView rajapintaan ja sille tehdään ML.NET:n suoritusketjun avulla lisää esikäsittelyä. Kaikessa lyhkäisyydessään, toleranssien Id:t korvamerkittiin tunnisteiksi, nominaaliset ja ordinaaliset tietueet muunnettiin numeraaliseksi käyttäen joko OneHotEncoding taikka FeaturizeText metodeja sekä osa numeraalisista tietueista muunnettiin Single- tietotyypiksi, sillä ML.NET: kouluttajat odottavat usein liukulukuja kokonaislukujen sijaan. Lopuksi kuvion 25 mukaisesti kaikki ketjutetaan yhteen vektoriin, minkä jälkeen malli voidaan kouluttaa.



Kuvio 25. Yksinkertainen esitys käsittelyn vaiheista

Tietoaineistoa kerätessä voi hankkia niin paljon tietoa kuin mahdollista, mutta kouluttaessa on olennaista valita mitkä ominaisuudet ovat olennaisia koulutuksen kannalta. Jotkut ominaisuudet voivat olla yhdistettäviä, tai niiden pohjalta voi luoda uusia. Mikäli mallilla on liian monta ominaisuutta, voi tunnisteen ennustaminen olla hankalaa koska kaikki ominaisuudet kilpailevat keskenään tärkeydestä. (Eledath n.d.)

Mikäli tällä käsittelyllä ja mallin parametreja muuttamalla ei päästä hyväksyttäviin tuloksiin, aineistoa voi esikäsitellä vielä muilla tavoin. Esimerkiksi tietyille ominaisuuksille voi asettaa ylä- ja alarajat, kuinka paljon tietueita kyseisellä ominaisuudella tulisi löytyä. (Data Transformation and Feature Engineering in Python 2021.) Aineisto on silti valitettavan epätasaista. Aineiston tasaamiseksi voisi yrittää esimerkiksi logaritmista muunnosta. Tällaista ominaisuutta ei kuitenkaan ML.NET:stä löydy sellaisenaan, joten se tulisi implementoida itse, eikä tähän nähty työn aikana tarvetta.

4.3 Koneoppimisalustojen vertailu

Edellä esitellyt koneoppimisalustat olivat kaikki vaihtoehtoina työkalua suunnitellessa. Python pohjaiset alustat vaikuttivat alkuun parhailta vaihtoehdoilta, sillä Pythonista ohjelmointikielenä löytyi eniten kokemusta. Näihin alustoihin löytyi myös internetistä helpoiten tietoa, koska ne ovat olleet jo useamman vuoden laajalti käytössä. ML.NET on tapauksista tuorein, ja muista poiketen se käyttää C# kieltä.

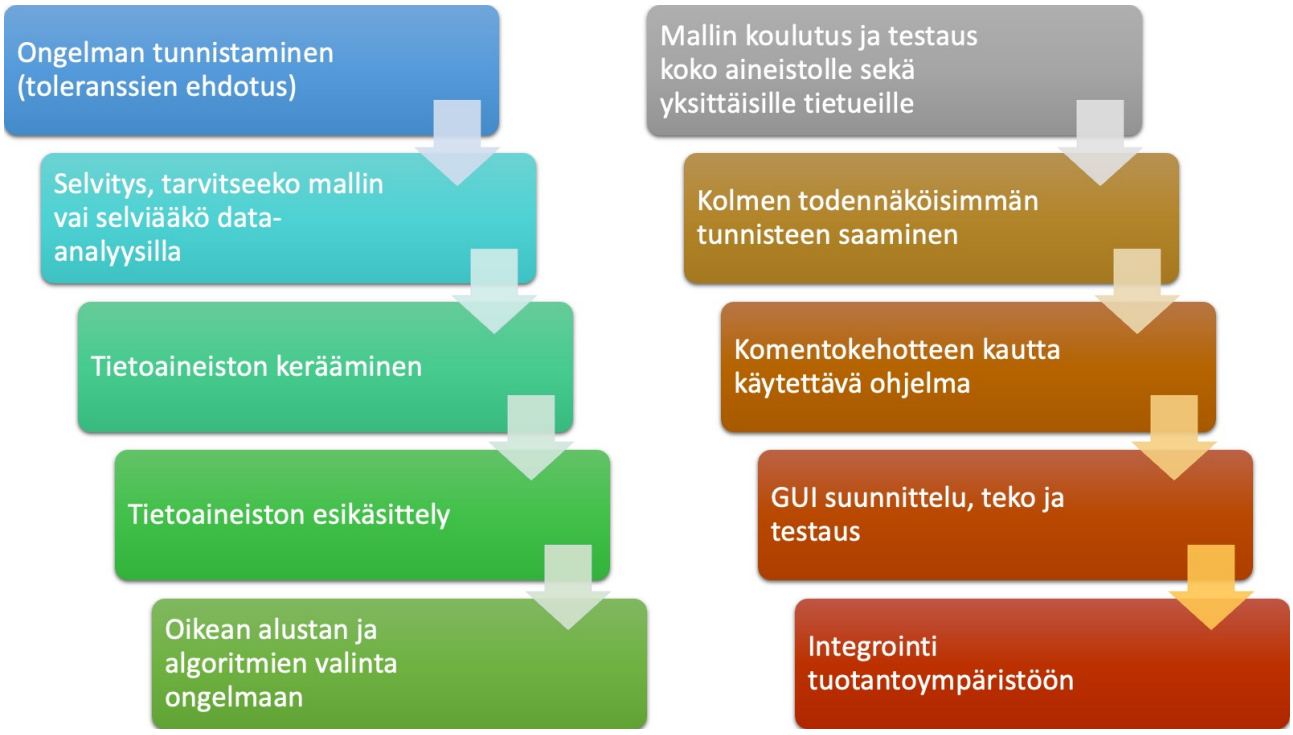
Kaikkia alustoja tuli testattua parin-kolmen viikon jakson ajan pienellä testidatalla. Näin jokaisesta alustasta sai käsityksen, kuinka asiat toimivat ja millaista niiden käyttö isommassa kaavassa on. Vaikka Python pohjaisten kielien avulla sai tehtyä nopeammin toimivat koneoppimismallit, joilla

sai konsoliapplikaation avulla tehtyä testejä, koitui ML.NET valinnaksi. ECT työkalu toimii .NET alustalla ja Python mallien integroiminen esikäsittelyineen alustan päälle osoittautui hankalammaksi kuin mallin koulutus alusta alkaen ML.NET avulla.

ML.NET:n integraatio .NET alustaan helpottaa työkalun tekoa myös muista näkökulmista. Koska kaikki toimii saman ohjelmointikielen päällä, voi aineistoa esikäsitellessä helposti tehdä omia esikäsittelyvaiheita, joita ei valmiina ML.NET:n puolelta löydy. Myös aineiston tuominen valitusta tietokannasta koulutusta varten oli helpompaa implementoida tätä kautta.

4.4 Työkalun suunnittelu

Tämän työn prosessin voi jakaa pitkälti kahteen osaan kuvion 26 tavoin. Tähän mennessä on suoritettu vasemmanpuoleisen osion vaiheet, missä keskityttiin tunnistamaan ongelma, hankkimaan sitä varten aineistoa sekä löytämään tarvittavat työvälineet ongelman ratkaisua varten. Tämän jälkeen itse työkalun teko lähti etenemään oikeanpuoleisen vaiheistuksen mukaisesti.



Kuvio 26. Järjestys, missä työkalun valmistamisen prosessi eteni

4.4.1 Työkalun toiminta käyttöliittymän takana

Alkuun tuli suunnitella työkalun arkkitehtuuri. Ensimmäisessä vaiheessa on päätetty millaiset ominaisuudet mallille annetaan koulutusaineistossa, mutta ennen koulutusta on tunnistettava millä algoritmeilla ja kouluttajilla mallia lähdetään kouluttamaan. ML.NET:n etusivulla kerrotaan alustan

pystyvän muun muassa tunneanalyysiin, tuotesuositusten tekoon, hintojen ennustamiseen, asiakassegmentoinnin tekoon, kohteiden tunnistukseen, petosten ja myyntipiikkien havaitsemiseen, kuvien luokitteluun sekä myynnin ennustukseen. (ML.NET – An open source and cross-platform machine learning framework n.d.) Näistä löytyi tarpeeksi vaihtoehtoja erilaisten ongelmien ratkaisuun, jotta alustalla kannatti jatkaa eteenpäin.

Ensimmäisenä ajatuksena oli yrittää tuotesuositusten mukaista mallia, jolloin se olisi suositellut mittaan sopivimpia toleransseja samaan tapaan kuin verkkokaupat suosittelevat asiakkailleen sopivia tuotteita. Alla oleviin algoritmeihin perehtymisen jälkeen kävi selväksi, ettei idea sovellukaan tähän käyttökohteeseen. Tutkiskelun jälkeen ML.NET:n mainostaman kuvantunnistuksen alta löytyi käyttötarkoitukseen sopiva työkalu, nimittäin luokittelu.

Usein koneoppimisen esimerkeissä kerrotaan kuinka malli on esimerkiksi opetettu tunnistamaan kuvasta joko kissa tai koira. Tällöin kyseessä on binääriluokittelu, jolloin vastaus on joko kissa tai koira. Kun luokkia on useampi, puhutaan moniluokkaisesta luokittelusta (Multiclass Classification). Tällöin kahden vaihtoehdon sijaan tunnisteita, eli vaihtoehtoja, on useampia, mistä malli koettaa tunnistaa sille opetetun materiaalin avulla todennäköisimmän vaihtoehdon.

Kuvien sijasta tämän työkalun tapauksessa keskitymme luokittelemaan toleransseja annettujen ominaisuuksien avulla. Koska meillä on tietokannassa jokaiselle ainutlaatuiselle pituusmittojen ylä- ja alaerotoleranssille nämä tiedot sisältävä tietue, voimme tehdä luokittelun tämän tietueen id-tunnistetta käyttäen. Alustava idea on, että käyttäjän valittua mitta ja avattua ECT työkalu, ohjelma tekee ennustuksen ja ehdottaa todennäköisimpiä vaihtoehtoja Id:n mukaan.

- Toleranssin ainutlaatuinen Id numero, luokittelua varten
- Tietuetta vastaava yläeromitta
- Tietuetta vastaava alaeromitta

Koska Catia erittelee pituus- ja halkaisijamitat omikseen, koulutamme halkaisijamitoille oman mallin. Tähän käytämme tunnisteena toleranssitunnisteen ja -asteen yhdistelmää. Myös koodin puolella rakenne on hieman erilainen, sillä kaikkea tietoa ei kannata pitää yhdessä tietueessa.

Käyttäjän valittua halkaisijamitan, ohjelma tekee ennustuksen. Ohjelma järjestelee ennustukset laskevaan järjestykseen todennäköisyyden mukaan ja tarkistaa, että alla olevan listan kohdat yksi ja kaksi toteutuvat. Tämän jälkeen se joko palauttaa kohdan 3 mukaisen listan, tai ohittaa epäsopivan ehdotuksen.

1. Valittu mitta on suurempi kuin minimimitta johon toleranssin voi sijoittaa (esimerkkinä tunniste T:n taulukot alkavat 24 mm ylöspäin)
2. Valittu mitta on pienempi kuin maksimimitta mihin asteikko yltää (useassa tapauksessa 3250 mm)

3. Lista sopivia toleransseja
    a. Mitan yläraja, minkä avulla haetaan seuraavat:
    b. Yläeromitta
    c. Alaeromitta

Listan kohta 3a tarkoittaa ohjelman tarkistavan valittuna olevan mitan, minkä avulla se hakee oikean toleranssin JSON tiedostosta yhteensopivat rajaeromitat siihen. Tarkennettuna, käyttäjä valitsee mitan, mikä on 200 mm ja työkalu ehdottaa toleranssia H7. Työkalu etsii oikean rivin, mikä on 180 < [mitta] ≤ 250. Tämän avulla työkalu pystyy laittamaan halkaisijaan oikeat rajaeromitat.

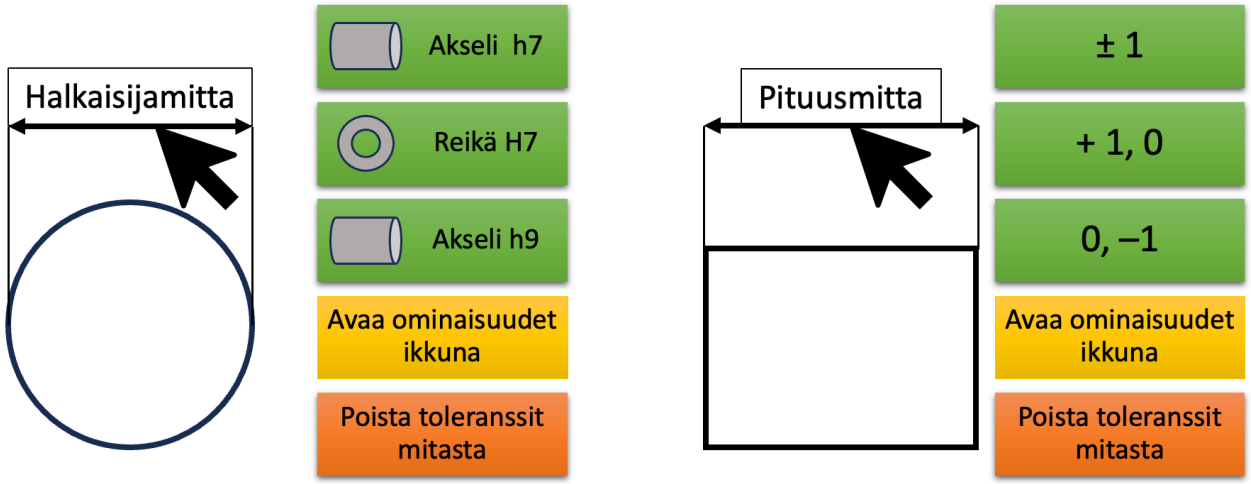
Idea olisi, että prototyypin asteelle päässeen työkalun toimintaa testataan niin koneoppimisalustaan sisäänrakennettujen mittareiden kuin itse tehdyn testausprosessin avulla. Kun työkalu on hyväksytyllä tasolla, työkalu annetaan suunnittelijoiden käyttöön, jolloin sen käytöstä voidaan kerätä tilastotietoja. Kerätyn tiedon pohjalta voi tehdä päätelmiä työkalun kannattavuudesta ja tuottavuuden tehostamisesta.

4.4.2 Työkalun käyttöliittymä

Käyttäjälle tarjotaan kolme vaihtoehtoa mistä valita. Tämä vaikutti sopivalta määrältä siksi, että nopealla vilkaisulla näkee löytyykö suunnittelijan mielestä sopivaa toleranssia eikä kolmen vaihtoehdon kohdalla tule vielä valinnanvaikeutta. Eri tutkimukset monivalintakysymysten kohdalla ovat tulleet siihen tulokseen, että kolme vaihtoehtoa on myös hyvä kompromissi valintojen määrässä (Supe & Vyas 2007, 132).

Näiden kolmen vaihtoehdon alla on vielä kaksi painiketta, joista toisen avulla voi avata Catian ominaisuudet-ikkunan ja toisella saa poistettua mitasta toleranssit. Nämä ovat siitä syystä, että käyttäjä voi halutessaan poistaa toleranssit nopeasti tai mikäli työkalu ei ole suositellut käyttäjän mielestä oikeaa toleranssia, hän voi helposti avata ikkunan, missä niiden laittaminen onnistuu.

Käyttöliittymän ulkoasu pyritään pitämään yksinkertaisena, sillä sen on tarkoitus olla työkalu. Ylimääräinen silmäkarkki ei tehosta työkalun suorituskykyä millään tapaa. Lopullisella ulkoasulla jäljitellään Catian omaa käyttöliittymää. Kuviossa 27 näkyy alustava hahmotelma, miltä valmis työkalun sommitelma voisi näyttää.



Kuvio 27. Yksinkertainen esitys työkalun visuaalisesta ilmeestä

4.5 ML.NET työkalun runkona

ML.NET:stä löytyy iso määrä valmiita algoritmeja erilaisten koneoppimisongelmien ratkaisuun. Moniluokkaista luokittelua varten löytyy esimerkiksi alla luetellut algoritmit:

- LightGBM
- SdcaMaximumEntropy
- SdcaNonCalibrated
- LbfgsMaximumEntropy
- OneVersusAll
- PairwiseCoupling
- NaiveBayes

Näistä PairwiseCoupling sekä OneVersusAll kouluttavat mallin verraten binääritunnistuksen avulla jokaista tunnistetta toisiinsa. Nämä jätin pois tunnisteiden suuren määrän vuoksi. NaiveBayes toimii ML.NET dokumentaation mukaan lähinnä pienten tunnistemäärien kanssa, joten se myös putoaa pois vaihtoehdoista.

Jäljelle jääneet LightGBM, Lbfgs sekä molemmat Sdca sekä algoritmit pääsivät testiin mallin koulutuksessa. Algoritmien toimintaan syvempi perehtyminen ei kuulu tämän asiakirjan piiriin, joten tässä huomioidaan vain testien tulokset. Algoritmien suorituskykyä testausta varten koulutettiin kaikilla neljällä algoritmilla omat koneoppimismallit. Näiden kohdalla vertailu tapahtui täysin ML.NET:n omien mittareiden avulla, joista lisää kappaleessa 3.4.2. Näiden tuloksena LightGBM päätyi voittajaksi MicroAccuracyn kohdalla, mitä testissä käytettiin mittarina, joten työkalun kehitys jatkui tämän algoritmin toimiessa runkona koneen koulutukselle.

Kuten Kelleher ja Tierney (2018, 100) ovat kirjassaan maininneet, koneoppimisessa suuri haaste on löytää omaan aineistoon parhaiten sopiva algoritmi. Jokaisella algoritmilla on oma oppimisvinouma, tarkoittaen niiden suosivan tai etsivän ratkaisua tietynlaisten funktioiden avulla, ja etenkin

kirjavalle aineistojoukolle erilaisia yhdistelmiä on niin paljon, ettei yksittäinen algoritmi voi käydä jokaista vaihtoehtoa läpi.

4.5.1 Mitä eri mittarit sisältävät?

Työkalua kouluttaessa sen suorituskykyä kuvaa erilaiset koulutukseen sisäänrakennetut mittarit. Työssä ei käydä tarkemmin läpi matematiikkaa jokaisen erillisen mittarin takana, vaan tärkeintä on ymmärtää, mitä niiden avulla mitataan ja millaista tulosta mittarille tavoitellaan.

LogLoss on yksi yleinen tapa mitata mallin suorituskykyä luokitteluongelmissa. Se laskee jokaisen syötteen kohdalla todennäköisyyden tunnisteelle. Binääriluokituksen tapauksessa LogLoss kertoisi kuinka lähellä ennustuksen todennäköisyys vastaisi todellista arvoa, joka olisi 0 tai 1 (ei / kyllä). Mitä lähempänä nollaa LogLoss arvo on, sitä tarkempi malli on ennustuksissaan. (Dembla 2020).

LogLossReduction mittari vertaa mallin suorituskykyä satunnaisennusteeseen nähden, jolloin esimerkkinä tulos 0.2 tarkoittaa mallin olevan 20 % tarkempi kuin satunnainen ennuste. Mittari voi näyttää myös negatiivista arvoa, mikäli mallin suorituskyky on heikompi kuin satunnainen ennustus tai valinta.

Micro- ja MacroAccuracy molemmat mittaavat mallin tarkkuutta oikein menneiden ennustusten osalta. Eroa niillä on, että MicroAccuracy ei ota huomioon eri luokkiin kuulumista. Se on hyvä valinta, kun tunnisteissa on selvää epätasaisuutta määrän osalta. MacroAccuracy laskee tarkkuutta jokaisen luokan tasolla erikseen, jolloin jokaisen luokan tulisi periaatteessa vaikuttaa lopputulokseen yhtä paljon. Eli jälkimmäisen osalla tunnisteiden lukumäärä ei vaikuta tarkkuuden painotukseen.

Molemmissa tarkkuuden mittareissa mitä lähemmäksi ykköstä päästään, sen parempi tarkkuus. Näistä kahdesta MicroAccuracy on yleisimmin varma valinta tarkkuuden selvitykseen. (Kershaw, Kirch, Ormont, Schonning, Victor, Quintatilla 2023.) Mikäli käyttäjä haluaa tietää esimerkiksi kolmen todennäköisimmän ennusteen tuoman tarkkuuden, hän voi käyttää TopKAccuracy nimistä mittaria, johon K:n tilalle määritellään tässä tapauksessa kolmonen. Tällöin malli laskee automaattisesti yhteen kolmen ennusteen yhteenlasketun tarkkuuden.

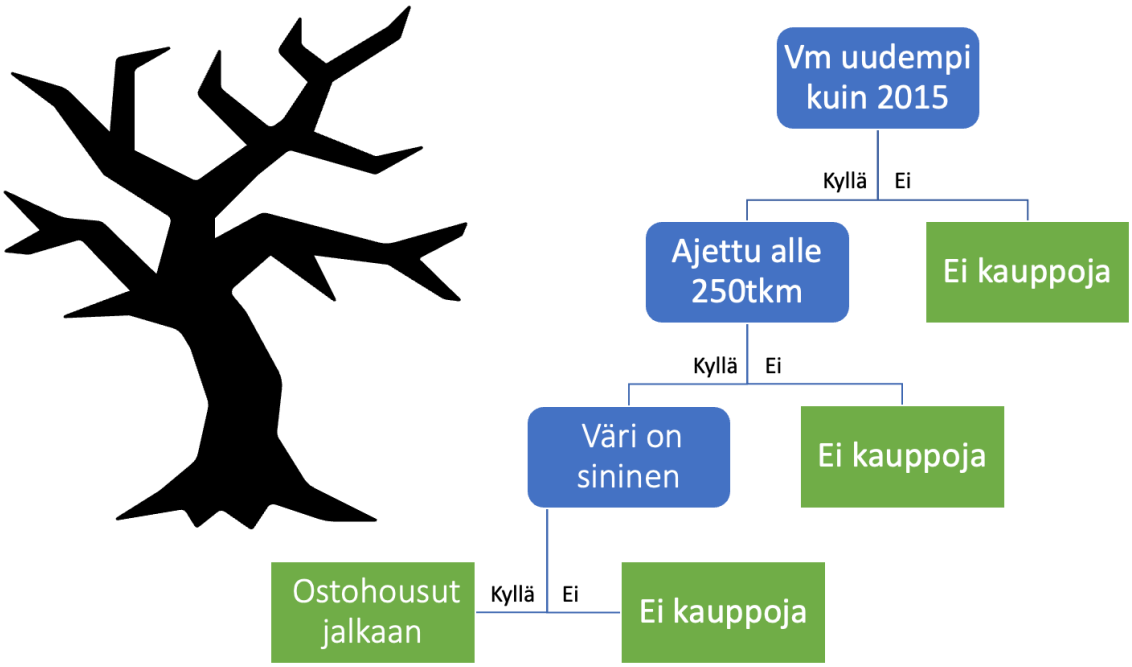
4.6 Neuroverkon ja päätöspuun vertailua

Usein koneoppimismalleista tai tekoälystä puhuttaessa ihmisille tulee mieleen neuroverkko. Luokittelua varten meillä on ML.NET:ssä valmiina erilaisia algoritmeja, joista löytyy myös päätöspuualgoritmeja. Päätöspuut ja neuroverkot toimivat hieman samalla periaatteella jakaen ominaisuuksia ja rakentaen mallia niiden välisten yhteyksien pohjalta. Päätöspuita voikin pitää yksinkertaistettuina versioina neuroverkoista. (Ye 2020.)

ML.NET:ssä ei työtä tehdessä ollut samalla tapaa neuroverkkojen luomiseen tukea kuin esimerkiksi TensorFlowissa tai PyTorchissa. Siinä on kuitenkin mahdollisuus tuoda ulkopuolinen malli suoritusketjuun, mitä käyttää ennusteissa. Tämän vuoksi tuli tutkia valmiina löytyvien algoritmien vaihtoehdoksi, miten toisella alustalla koulutettu neuroverkko soveltuisi tehtävään.

4.6.1 Päätöspuiden toiminta

Toisin kuin esimerkiksi regressiomallien kohdalla, päättelypuita hyödyntävien mallien rakenne voi olla jokaisella koulutuskerralla erilainen (Dissanayake 2023). Asiaa voi ajatella näin: Meillä on tietty määrä ominaisuuksia, ja joku niistä toimii puun juurina. Puusta lähtee eriämään ominaisuuksia omina oksinaan, ja niiden paikat voivat vaihdella jokaisella koulutuskerralla. Kuviossa 28 on yksinkertainen esimerkki päätöspuusta; Kuvitellaan, että ensimmäisellä koulutuksella lähtökohta on kuvion mukaisesti vuosimalli. Mikäli toisella koulutuskerralla lähtökohtana olisi väri, lopputulos voi olla erilainen. Siniset laatikot kuvaavat ominaisuuksiin kohdistuvia testejä, kun vihreät osoittavat testin jälkeistä päätöstä.



Kuvio 28. Yksinkertainen päätöspuu visualisoituna

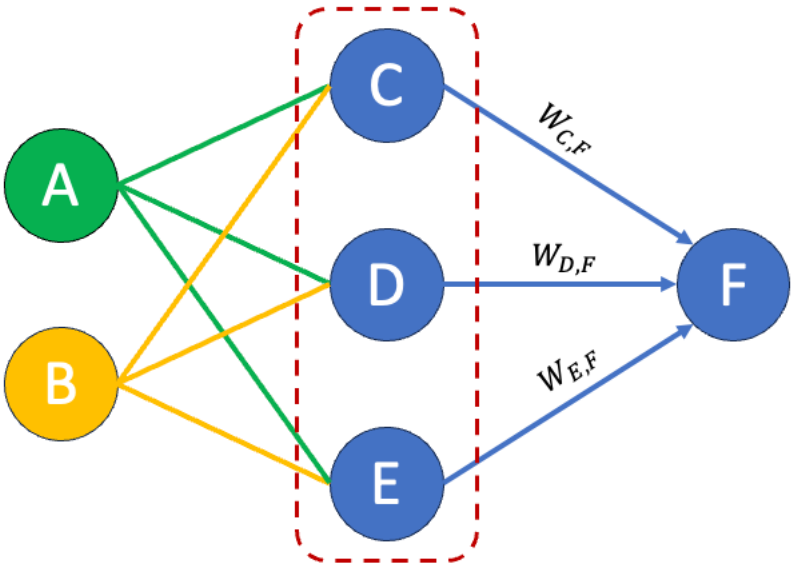
Päätöspuut muistuttavatkin hieman vuokaaviota, missä jokainen haarautumiskohta on jonkinlainen suodatuskriteeri. Puuta tehdessä voi määritellä vähimmäisjoukon tietueita, joista yksittäinen lehti tai oksa muodostuu, ja tarvittaessa puun koon voi rajoittaa haluttuun syvyyteen. Päätöspuuta tehdessä tulee kuitenkin pitää huoli tarpeeksi suuresta tietoaineistosta, sillä liian pienellä aineistolla puun ennustuksista tulee epätarkkoja. (Nelson 2020.) Päätöspuiden hyvä suorituskyky nominaalisen ja ordinaalisen aineiston kanssa pohjautuukin osaltaan niiden tapaan käsitellä niille syötettyä aineistoa. Puut voivat käyttää aktivointifunktion sijaan solmukohdassa esimerkiksi

ehtolauseita, jolloin valinta nominaalisten tietueiden välillä on sujuvaa. Lisäksi niiden tulkinta on helpompaa kuin neuroverkoissa niiden visuaalisen esitystavan vuoksi, mikäli malli tukee tätä vaihtoehtoa.

Puussa käytetyn koulutusalgoritmin tavoitteena on löytää luokittelua varten sääntöjä, joiden avulla ominaisuuksia voidaan erotella omiin sarjoihinsa. Kun aineistosta on saatu eroteltua sarjoja, joiden tavoitearvot ovat samat, todennäköisyyksien avulla on mahdollista ennustaa myös uusien ominaisuuksien kuuluvan tiettyyn joukkoon. (Kelleher ja Tierney 2018,136–137.)

4.6.2 Neuroverkkojen toiminta

Kuviossa 29 on visualisoituna yksinkertainen yhdistetty neuroverkko. Vasemmassa laidassa on syöte, missä on kaksi neuronia A ja B. Yhdistämällä näitä neuroneita rinnakkain ja peräkkäin saadaan muodostettua neuroverkko. Kuvion verkossa informaatio virtaa vasemmalta oikealle. Kukin neuroni oppii koulutuksen aikana yksinkertaisen funktion, minkä avulla se muokkaa ulostuloa. Verkossa jokainen neuroni on yhteydessä jokaiseen neuroniin, joka on seuraavalla tasolla. Neuroverkoissa on vaihtelevasti piilotettuja tasoja, joita kuviossa 29 on yksi ja se kattaa neuronit C, D ja E. Näiden syötteet eivät tule aineistosta suoraan, minkä vuoksi niitä kutsutaan piilotetuiksi.



Kuvio 29. Visualisoitu yksinkertainen yhdistetty neuroverkko

Neuroverkko on kaikessa yksinkertaisuudessaan funktio, joka muodostuu yhdistämällä yksittäisten neuronien yksinkertaisia funktioita yhteen. Funktio kuvaa deterministisesti verkon syötteiden muodostumista aina samanlaiseksi ulostuloksi. Tämä tarkoittaa käytännössä sitä, ettei ulostulo muutu, mikäli syöte pysyy samana. (Kelleher 2019, 18.)

Jokainen neuroni saa syötteenä paino- ja sisääntuloarvojen matriisin W * X, joka muodostuu Kelleherin (2019, 72) oppien mukaan seuraavasti:

$$z = (w_1 x_1) + (w_2 x_2) + \cdots + (w_n x_n)$$

missä w on yksittäisen neuronin painoarvo ja x on neuronin saama sisääntulo. Kaava voidaan esittää myös yksinkertaisemmassa muodossa:

$$z = \sum_{i=1}^{n} x_i * w_i$$

Neuronin arvo lasketaan ottamalla syötteiden matriisien W ja X tulos ja lisäämällä niihin bias-termi b, minkä jälkeen ne kerrotaan aktivointifunktion A avulla.

$$y = A(W * X + b)$$

Yllä olevassa kaavassa y on neuronin arvo, A on aktivointifunktio, W painoarvojen matriisi, X on neuronin sisääntulojen matriisi ja b viittaa nollasta eroavaan vakioon bias-termiin, mikä määritetään koulutuksen alussa satunnaisesti ja se määräytyy koulutuksen aikana lopulliseen arvoonsa. Tämän myötä kuvion 29 neuronissa F lasketun ulostulon laskukaava on seuraavanlainen:

$$Neuronin\ F\ ulostulo = A(W_{C,F}C + W_{D,F}D + W_{E,F}E)$$

Virheiden laskennassa tavoite on saada se mahdollisimman pieneksi. Mikäli se on nolla, ei ole tarvetta muuttaa verkon neuroneiden painotuksia. Mikäli virhe on joko positiivinen tai negatiivinen, saadaan sitä pienennettyä säätämällä neuroneiden painotuksia oikeaan suuntaan. Yleensä tässä käytetään apuna vastavirta-algoritmia. (Kelleher ja Tierney 2018, 128–130.)

Muuttamalla verkon leveyttä ja syvyyttä, eli neuronien määrää yhdessä tasossa sekä näiden tasojen määrää, saadaan rakennettua tunnistettua ongelmaa varten malli. Luokitteluongelman ollessa kyseessä aktivointifunktion valinta vaikuttaa suuresti verkon suorituskykyyn. (Zewe 2023a.) Luokitellessa meillä on viimeisellä tasolla neuroni jokaista tunnistetta kohden, ja jokaisen ulostulona on todennäköisyys, jolla syöte kuuluu luokkaan (Louridas 2020, 211–222.)

4.6.3 Kumpi on parempi työn tavoitteita ajatellen?

Molemmissa tavoissa kouluttaa malli on puolensa. Eräiden testien mukaan päätöspuu on suorituskykyisempi strukturoidun, tietokantamuotoisen aineiston kanssa, kuin neuroverkot. Tätä tukee IEEE:n julkaisema tutkimus aiheesta (Borisov, Haug, Kasneci, Leemann, Pawelczyk & Seßler 2022,

12), missä puupohjaisia algoritmeja käyttävä LightGBM oli nopein kolmessa ja toisiksi nopein kahdessa testissä yhdeksästä. Päätöspuita on myös yleensä nopeampaa kouluttaa kuin neuroverkkoja.

Neuroverkot toimivatkin paremmin jäsentelemättömän tiedon, kuten kuvien, videon ja äänen kanssa (Nabil 2023). Koulutusaineistomme on tallennettu SQL tietokantaan, joten päätöspuu vaikuttaisi paremmalta vaihtoehdolta. Tämän lisäksi aineistossamme on paljon nominaalista ja ordinaalista aineistoa, minkä käsittelyyn päätöspuu on parempi vaihtoehto (Kelleher ja Tierney 2018, 136).

Päätöspuiden tapa luokitella ehtolausein kuulostaa järkevältä vaihtoehdolta, kun määrittelemällä puun juuriksi yhteistyötilan voi malli lähteä tekemään päätöksiä sen perusteella sen sijaan, että malli kävisi kaikkien tilojen mahdolliset vaihtoehdot läpi. Tällöin käyttäjän valitessa mitan jo ensimmäisessä vaiheessa vaihtoehtojen määrää saadaan kutistettua huomattavasti, mikä tekee ennustuksesta tehokkaampaa.

Päätöspuuta kouluttaessa tulee kuitenkin muistaa, ettei siitä tee liian monimutkaista. Puun monimutkaisuus vaikuttaa sen suorituskykyyn, joten liian monimutkainen puu on suorituskyvyltään heikompi (Maimon & Rokach 2008, 14–15). Työssä käytettävässä aineistossa on tietueita ja ominaisuuksia suhteellisen vähän, joten päätöspuuta ajatellen malli ei pääse kasvamaan liian monimutkaiseksi.

Edellä mainituista syistä mallin koulutukseen käytetään ML.NET alustalta löytyvää LightGBM algoritmia. Se on rakennettu perinteisen päätöspuualgoritmien pohjalta, mutta käyttää suorituskyvyn edistämiseksi lisänä eksklusiivisten ominaisuuksien niputtamista (Exclusive Feature Bundling, EFB) ja gradienttipohjaista yksipuolista näytteenottoa (Gradient-based One-Side Sampling, GOSS). LightGBM:n on todettu voittavan useimmat perinteiset päätöspuualgoritmit laskentanopeudessa ja muistin käytössä (Ke, Finley, Wang, Chen, Ma, Ye, Liu, 2017, 8), mikä tekee siitä oivan valinnan työkalua varten.

4.7 Mallin koulutus ja testaus

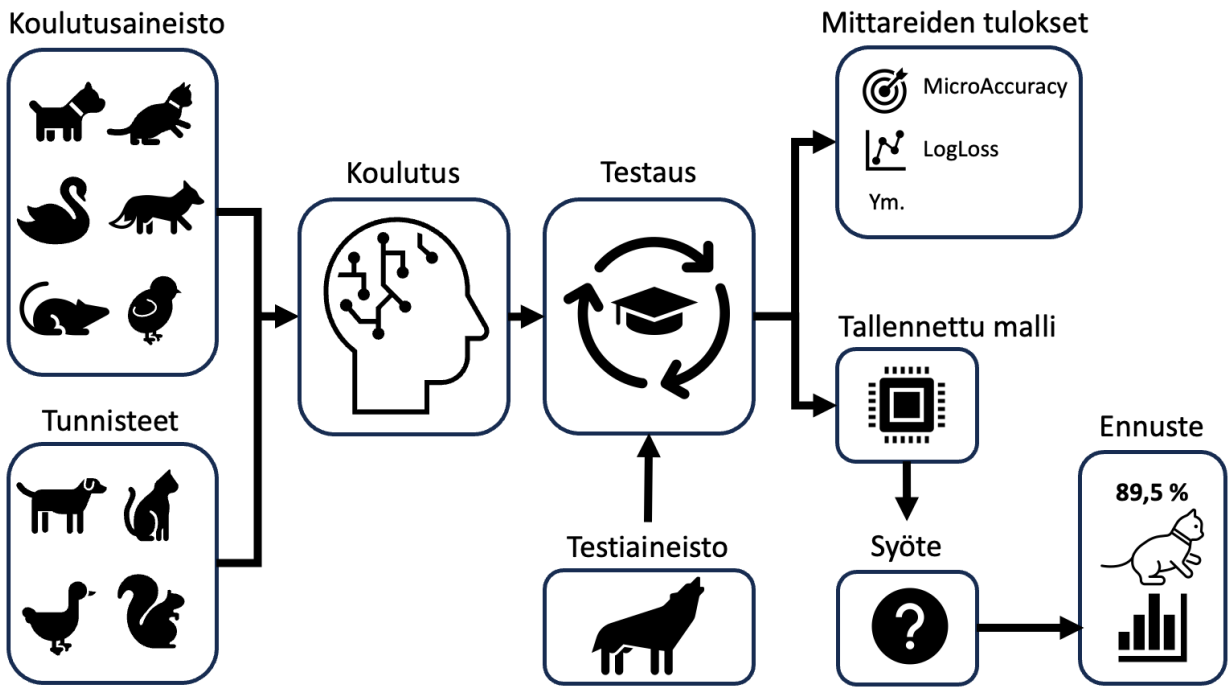
4.7.1 Mallin koulutusprosessi

Työtä tehdessä koulutus eteni monen eri iteraation lävitse ja malli muuttui matkalla useasti. Malleja koulutettiin samalla kun tietoaineistoa kerättiin ja tällöin huomasi selkeän korrelaation mallin suorituskyvyn ja tietoaineiston määrän kanssa. Kun aineistossa päästiin yli 10 000 tietueen, mallia kouluttaessa päätöspuun avulla mittarit ilmaisivat tasaisempaa vaihtelua yksittäisten koulutuskertojen välillä. Pienemmällä aineistomäärällä jokaisen koulutuskierron jälkeen mittarit vaihtelivat runsaasti.

Tietoaineistoa on työtä tehdessä hieman päälle 46 000 tietuetta, minkä myötä on malli koulutettu käyttäen ristiinvalidointia. Tämän avulla malli pääsi parempiin tuloksiin niin mittareiden kuin oman testiohjelman tuloksissa. Yksittäinen koulutuskierros tällä aineistolla ei vie paria minuuttia kauempaa, mutta koska kyseessä on päätöspuu, parhaan mallin löytämiseen menee oma aikansa.

Selkeyden vuoksi koulutusprosessia on visualisoitu alla kuviossa 30. Kuviossa aineisto on korvattu eläinten kuvilla, koska ne toimivat paremmin tuhansien tietueiden taulukkoa paremmin. Tietoaineistona toimii Catiasta löytyvien vanhojen piirustusten avulla koottu SQL-tietokanta. Tämä on jaettu koulutus- ja testiaineistoiksi. Pituustoleransseista on koottu kaikki ainutlaatuiset yhdistelmät ja niille on annettu tunnistetta varten Id. Mallia kouluttaessa tietokannasta otetaan nämä toleranssit ulos ja tehdään JSON tiedosto, missä on Id:tä vastaavat toleranssit. Halkaisijatoleranssien kohdalla tunnisteena toimii kirjaimen ja numeron yhdistelmä, josta tarkastetaan valitun mitan mukaan oikea toleranssi.

Koulutuksen jälkeen mallin suorituskykyä testataan testiaineiston avulla, jolloin saadaan ulos eri mittarit. Koulutusohjelmaan on määritetty vaatimukset MicroAccuracyn sekä LogLossin osalta. Aluksi tarkkuuden on oltava yli 70 % ja LogLossin alle 2.0, ja rajat päivittyvät uusien mallien mukaan, jolloin tallennetaan näillä kahdella mittarilla mitattuna paras malli. Mikäli malli läpäisee nämä, se tallennetaan myöhempää testausta varten. Tallennettu malli sijoitetaan lopulta työkaluun, jossa se ennustaa käyttäjän valitsemasta mitasta todennäköisyydet toleransseille. Mikäli käyttäjä valitsee näistä jonkun, ohjelma asettaa toleranssit Catiassa valittuun mittaan ja muuttaa samalla mitan tarkkuuden vastaamaan toleranssien tarkkuutta.



Kuvio 30. Visuaalinen esitys mallin koulutusprosessista tuotantoon

4.7.2 Pituustoleransseille koulutetun mallin suorituskyvyn testaaminen

Kouluttaessa mallia saadaan suorituskyvystä tietoa mittareiden avulla. Näiden tueksi valmistettiin testausta varten oman testiohjelma, joka käy testiaineiston läpi rivi kerrallaan ja tekee ennustukset jokaiselle riville erikseen. Sen jälkeen ohjelma ottaa talteen montako kutakin toleranssin Id:tä testiaineistosta löytyi, sekä kuinka mones veikkaus ohjelmalla on osunut oikeaan, ja muodostaa oman tiedoston näistä tiedoista. Näitä tietoja voi myöhemmin käyttää mallin tarkempaan tarkasteluun.

Mallia testattiin kouluttaa itse ohjelmoidun ohjelman lisäksi AutoML:n avulla. Muutaman kerran kyseisen koulutuksen aikana saadut mittarit näyttivät mallin olevan todella tarkka testiaineiston kohdalla. Tarkastellessa itse tehdyn testiohjelman tulostetta kävi kuitenkin ilmi, ettei mallin ennustukset osuneet kertaakaan oikeaan. Tarkkuus oli lähes 98 %, muttei tarkemmin tarkasteltuna yksikään vastannut oikeita tietueita.

Tähän voi vaikuttaa moni asia, joista todennäköisimpänä epätasainen aineisto. Tässä tapauksessa malli oppii veikkaamaan ns. ”väärää” tulosta aina, mutta sen mielestä tarkkuus on tällöin korkea. Selkeyden vuoksi, ajatellaan 1000 tietueen aineistoa missä on 990 kissa- ja 10 koirakuvaa. Ideana on kouluttaa aineiston avulla malli, mikä tunnistaa eläimet toisistaan. Mutta mikäli mallin ajavana mittarina on tarkkuus, se saavuttaa helposti 99,9 % tarkkuuden veikkaamalla kissaa jokaisen kuvan kohdalla, vaikkei se olisikaan toivottu tulos.

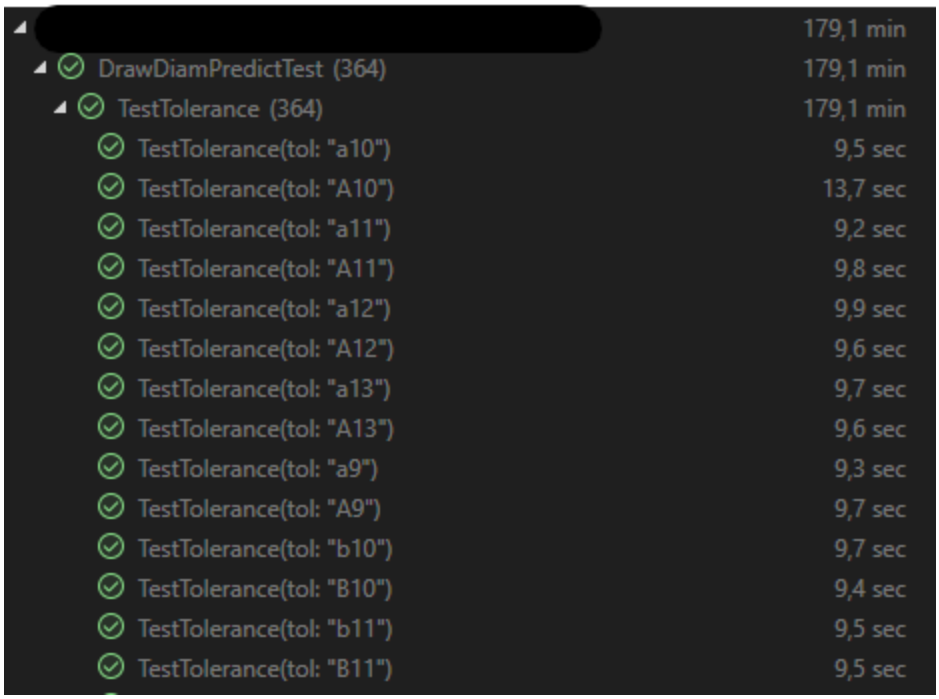
Kuten Kämäräinen (2023, 148) kirjassaan mainitsee, päätöspuut ovat niin tehokkaita oppimaan aineistoa, että ylisovitusta tapahtuu helposti. Tähän ratkaisukeinona olisi joko parametrien muutos, AutoML:n kohdalla koulutusajan lyhentäminen tai mikäli mahdollista, validointiaineiston käyttäminen koulutuksessa. Päätöspuumalleista voidaan myös karsia solmukohtia, jotka eivät vaikuta luokittelun tulokseen.

Koska AutoML:n tulokset olivat vähemmän kuin optimaalisia, käytettiin työkalussa mallia, minkä suoritusketju oli kasattu itse. ML.NET:stä löytyi kattavat ohjeet tätä varten. Samalla aiheeseen sai huomattavasti paremman ymmärryksen, kun ei antanut automaation hoitaa kaikkea. Mallin suoritusketjun esikäsittely toimii hyvin pitkälle ML.NET:n tarjoamilla työkaluilla, mutta piirustusten nimiä varten tehtiin erillinen ohjelma parantamaan niiden käsittelyä.

4.7.3 Toleranssien testaaminen työkalun osalta

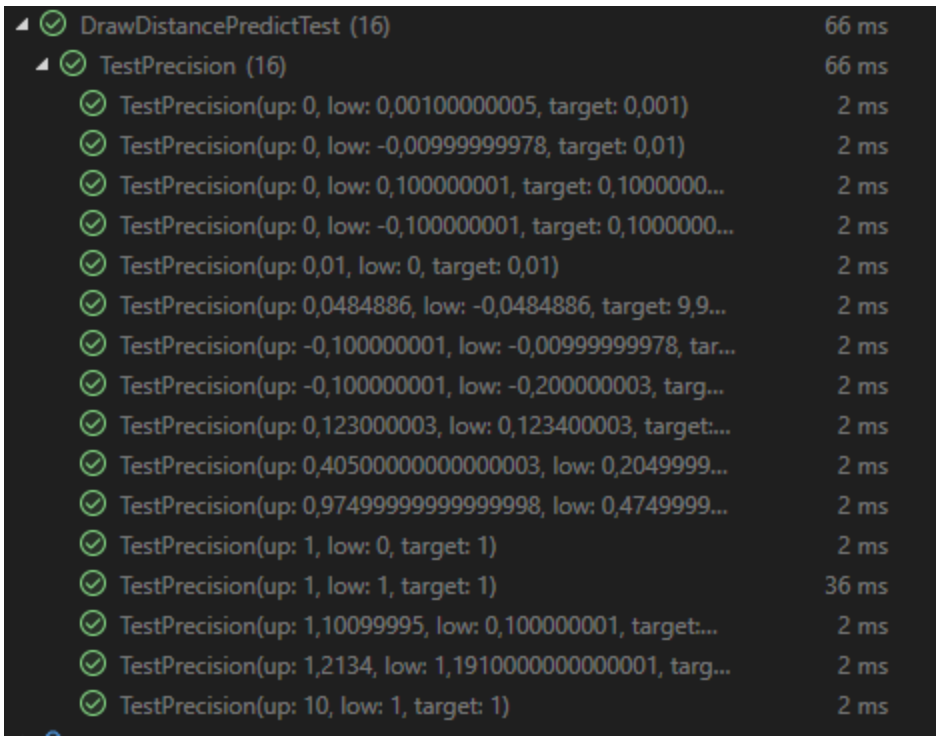
Koska halkaisijamittojen toleranssit löytyvät standardeista, tuli niitä varten tehdä ohjelma, mikä testaa työkalun asettamien toleranssien olevan samat kuin mitä Catiassa. Tällä ehkäistään erilaiset inhimilliset virheet, mitä tietoja viedessä työkaluun olisi mahdollista tulla. Jokaisesta toleranssista valmistettiin oma Excel tiedosto, mistä luotiin ohjelmallisesti omat JSON tiedostot nopeaa käsittelyä varten. Tämän jälkeen testiohjelma piirsi Catiaan millin välein kaksi ympyrää, mihin asetettiin

toleranssit. Toiseen ympyrään tuli Catian asettama toleranssi, ja toiseen koneoppimismallin työkalun vastaava. Mikäli kumpikin toleranssi vastasivat toisiaan, testi meni sen toleranssin ja halkaisijan osalta läpi kuvion 31 mukaisesti. Jälkiviisaana tämän testiohjelman avulla olisi voinut automaattisesti kaapia kaikki toleranssit taulukoihin, joista olisi tehnyt JSON tiedostot toleransseille.



Kuvio 31. Testien läpiajo halkaisijamitoille

Seuraavaksi tuli tehdä pituustoleransseja ajatellen testityökalu. Koska tietokoneet käsittelevät desimaalilukuja erikoisesti, toleranssien tarkkuuden selvitystä varten tuli tehdä oma testinsä. Kuvion 32 mukaisesti mahdollisimman monta erilaista vaihtoehtoa toleranssin ylä- ja alamitoille tuli testata niin, että kaikista saadaan oikea tarkkuus, mitä voidaan käyttää toleranssia asettaessa mittaan.



Kuvio 32. Testien läpiajo pituustoleransseille

4.8 Ominaisuuksien tärkeyden permutaatio

Ominaisuuksien tärkeyden selvittäminen (englanniksi Permutation Feature Importance) on koneoppimista varten tärkeää, sillä se avaa mallin mustaa laatikkoa. Sen avulla saadaan selville, kuinka paljon kukin ominaisuus vaikuttaa mallin suorituskykyyn ja mihin suuntaan. Analyysi ei toisaalta ota kantaa siihen, kuinka eri ominaisuuksien yhdistelmät vaikuttavat siihen. Analyysi toimii käytännössä sekoittamalla satunnaisesti aina yhtä ominaisuutta tietoaineistossa kerrallaan, minkä jälkeen se vertaa tämän sekoitetun mallin suorituskykyä alkuperäiseen malliin. Mitä suurempi muutos, sitä tärkeämpi ominaisuus on mallin suorituskykyä ajatellen. (Kershaw, Ormont, Schonning, Victor, Tabladillo, Quintatilla 2022.)

Kuten Breimanin (2001, 18–23) paperissa mainitaan, eri ominaisuuksien korrelaatio voi vaikuttaa analyysin tekemään tärkeyteen. Kahdessa ominaisuudessa voi olla samoja tietoja, jolloin niiden testaaminen erikseen tuo yhtä paljon eroa mallin mittareita ajatellen. Mutta mikäli niitä käytetään mallissa yhtä aikaa ja puussa nämä kaksi tulevat peräkkäin, ominaisuudet eivät paranna mallin suorituskykyä huomattavasti.

Ilman tätä analyysia mallin tekemät päätökset jäisivät varjoon. Kuten kappaleessa 2.9 mainittiin, koneoppimismallit saattavat oppia reaalimaailmaa ajatellen käyttämään turhia ominaisuuksia ennusteissaan. Tämän työkalun kohdalla mielestäni tällaiselle ei kovin suurta vaaraa ole ollut, sillä kaikki ominaisuudet löytyvät Catian piirustusten ketjusta. Tällä tarkoitetaan sitä, että kun Catian

puuta lähtee kiipeämään mitasta ylöspäin, pääsee lopulta piirustuksen ja yhteistyötilan tasolle. Mikäli kyseessä olisi vaikkapa konenäköön perustuva malli, pitäisin analyysia tärkeämpänä. Tähän vaikuttaa tietysti mallin tarkoitus ja tietoaineisto.

Taulukko 1. Ominaisuuksien tärkeyden permutaation jälkeiset tulokset valmiin mallin osalta, lajiteltuna LogLossin mukaan suurimmasta pienimpään.

Feature	MacroAccuracy	MicroAccuracy	LogLoss	LogLossReduction
FeatTitle	-1,432101612	-0,713239142	5,947749296	-2,048062692
FeatSN	-0,15840941	-0,289508111	3,389709865	-1,167221073
DimensionValue	-0,502112095	-0,277341706	2,075482851	-0,714676894
ViewScale	-0,298371316	-0,221088435	1,475558704	-0,508097531
FeatProject	-0,116064504	-0,136839351	0,763082152	-0,262761594
VTSingle	-0,236073641	-0,079277865	0,51795572	-0,178354153
DTSingle	-0,169149079	-0,043956044	0,297360771	-0,102393943

Taulukossa yksi on mallille tehdyn PFI-analyysin tulokset. Taulukkoa tarkastellessa käy selväksi, että piirustuksen nimi vaikuttaa tulokseen eniten. Tähän todennäköisesti vaikuttaa se, että tietynlaisissa piirustuksissa käytetään tietyn skaalan toleransseja. Tällaisia piirustuksia voi olla esimerkiksi tukipalkin, kiinnikkeiden tai pohjalevyjen piirustukset, joiden toleranssit voivat erota toisistaan toiminnallisuuden perusteella.

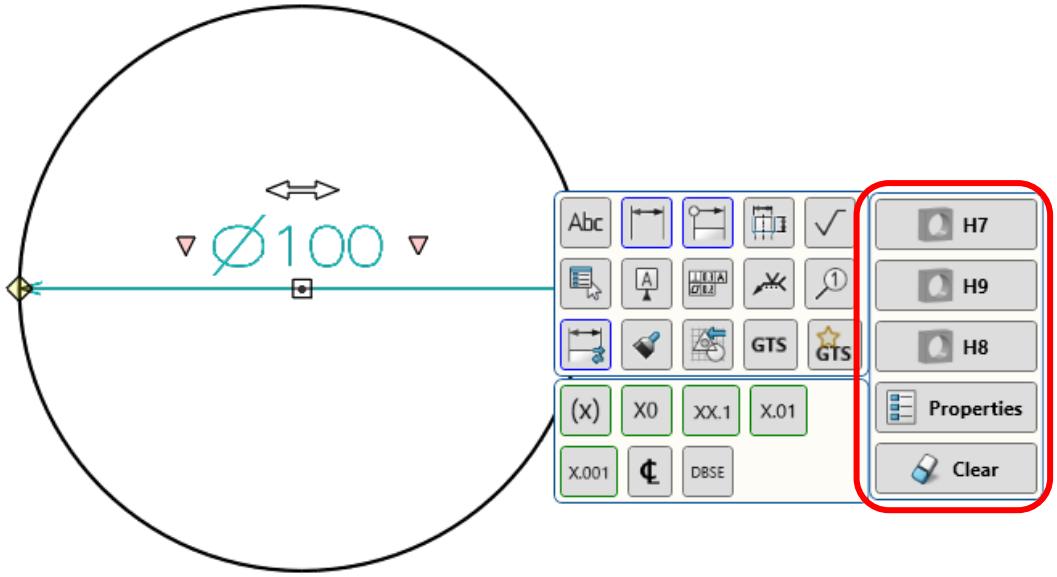
Liitteessä kaksi on visualisoituna tietoaineistoa ja ominaisuuksien jakautumista. Jos ei lasketa mukaan piirustuksen nimeä tai nimellismittaan viittaavia ominaisuuksia, ainutlaatuiset vaihtoehdot tietueissa liikkui 10–50 kappaleen välillä. Esimerkiksi erilaisia piirustuksen skaaloja oli 37 kpl, piirustusarkin nimiä 46 kpl ja kuvantoja 12 kpl. Erilaisia piirustuksen nimiä oli yli 2000.

Ottaen huomioon miten epätasaisesti aineiston määrä jakautuu jokaisen ominaisuuden kohdalla, piirustusten nimien määrä vaikuttanee positiivisesti mallin suorituskykyyn puhtaasti vaihtoehtojen määrän muodossa. Samoin niiden kappalemäärä jakautuu tasaisemmin, kuin monen muun ominaisuuden kohdalla. Esimerkiksi ainutlaatuisia mittoja aineistosta löytyi 7743 kappaletta. Mikäli aineistosta jätettäisiin pois mitat, mitä löytyi alle 40 kappaletta, olisi meillä vain 178 erilaista mittaa. Nämä 2,3 % mitoista kattavat lukumäärällisesti noin 45,2 % aineistosta. Tämän vuoksi suuri osa mitoista on sellaisia, että niitä on lukumäärältään liian vähän, jotta malli pystyisi löytämään korrelaatioita muiden ominaisuuksien ja mittojen välille.

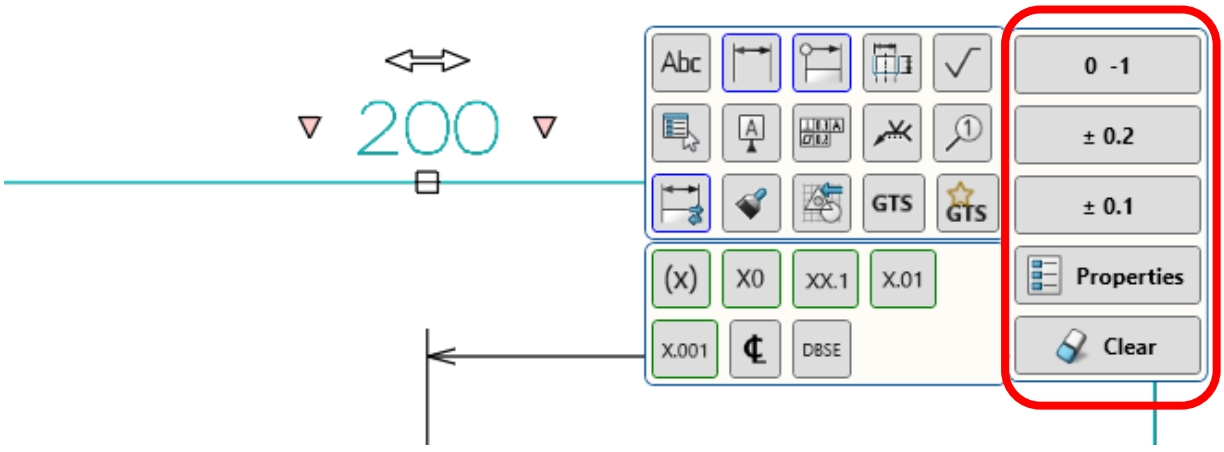
4.9 Työkalun lopullinen ulkoasu käyttäjiä varten

Työkalun ulkoasun tulee olla yksinkertainen, jotta siitä näkee nopealla vilkaisulla kaiken oleellisen. Käyttäjän valitessa mitta Catiassa ja avatessa ECT, kellukkeen oikealle puolelle ilmestyy automaattisesti työkalun tekemät ennusteet. Työkalussa oli valmiiksi ominaisuus, mikä tekee viisi vaihtoehtoa kellukkeen oikealle puolelle, joten tätä käytettiin myös tämän työkalun pohjana. Muuttamalla

ainoastaan painikkeiden sisältöä työkalun yleisilme pysyy samana, ja painikkeet noudattavat käyttäjille tuttua kaavaa. Koska kyseessä on työkalu, jota suunnittelijat käyttävät työn tehostamiseen, on sen toiminnallisuus tärkeämpää kuin komea ulkoasu.



Kuvio 33. Työkalun toiminta valitun halkaisijamitan kohdalla



Kuvio 34. Työkalun toiminta valitun pituusmitan kohdalla

Kuvioissa 33 ja 34 näkyy miten erilaisten mittojen osalta työkalun ulkoasu muuttuu hieman. Halkaisijamitan ollessa kyseessä työkalu ilmaisee toleranssin lisäksi kuvalla onko kyseessä reikä- vai akselitoleranssi selkeyden vuoksi, jotta ne erottavat toisistaan silmäyksellä.

5 Työn tulokset

5.1 Suorituskyky

Työkalun pituusmittojen toleransseja varten koulutettu malli pääsi valmiiden mittareiden osalta alla olevan taulukon kaksi mukaisiin tuloksiin. Mallin saavuttama 93,4 % tarkkuus ennustuksissa kolmen ensimmäisen vaihtoehdon kohdalla on hyväksyttävällä tasolla. Kuten kappaleessa 4.8. kävimme läpi, yhteistyötilojen tietoaineiston määrässä on selkeitä epätasaisuuksia, muuta siitä huolimatta tarkkuus on kohtuullinen.

Liitetiedostosta kaksi löytyy vertailu kahden mallin välillä. Toinen malli on koulutettu kaikella aineistolla mitä tietokannasta löytyy ja toinen neljän suurimman yhteistyötilan aineistolla. Alla olevat tulokset ovat jälkimmäisen mallin tulokset. Neljän suurimman yhteistyötilan malli suoriutui paremmin niin koko aineiston kuin vain neljän suurimman yhteistyötilan aineistojen osalta. Verrokkina näille liitteessä on testiaineistosta löytyvien toleranssien määrä, jolloin kolmen eniten löytyvän toleranssin avulla tarkkuus jää molemmissa tapauksissa alle 50 %. Työkalun koulutuksesta on siis ollut hyötyä verrattuna vain tilastollisesti suurimpien vaihtoehtojen valintaan.

Taulukko 2. Työkalun pituustoleransseja varten koulutetun mallin mittarit.

LogLoss	MicroAccuracy	LogLossReduction	Top 3 Accuracy
0,844	**80,6 %**	**0,709**	**93,4 %**

Ideaalina ratkaisuna voisi ajatella, että kouluttaessa jokaiselle yhteistyötilalle oma malli päästäisi parhaimpaan tulokseen, mutta aineiston määrän vuoksi se on hankalaa. Työkalussa käytetty LightGBM algoritmi ylisovittaa helposti pienillä, alle 10 000 tietueen aineistoilla (Bahmani 2023), ja neljän pienimmän yhteistyötilan kohdalla kaavituista tietueista löytyi yhteensä alle 1600 toleranssia. Aineistoa voisi kaapia pidemmältä ajalta, mutta herää kysymys, milloin kaavittu aineisto menee liian vanhaksi eikä se ole enää käyttökelpoista uusia projekteja ajatellen. Voi myös ajatella, ettei työtilassa, josta ei aineistoa löydy, tehdä kovin usein piirustuksia, joissa työkalusta olisi huomattavaa hyötyä. Nämä neljä pienintä yhteistyötilaa muodostavat koko aineistosta vain 3,28 prosenttia.

Vertailua varten koulutettiin neljälle suurimmalle yhteistyötilalle kullekin oma malli, joiden tulokset löytyvät liitetiedostosta yksi. Jokainen pääsi kolmen ensimmäisen ennustuksen osalta yli 90 % tarkkuuteen. Tarkastellessa näiden mallien testiaineistosta löytyvien toleranssien määrää tulee selväksi, että jokaisen yhteistyötilan kohdalla käytetyt toleranssit vaihtelevat jonkin verran mutta myös päällekkäisyyttä on havaittavissa. Voisi kuvitella, että koko aineistolla koulutettu LightGBM algoritmia hyödyntävä malli tekee päätökset valitun yhteistyötilan mukaan jotakuinkin samalla suorituskyvyllä kuin jokaiselle yhteistyötilalle erikseen koulutettu malli, jolloin ylläpidon kannalta on järkevämpää kouluttaa yksi iso malli monen pienen sijaan.

Tietokoneen resurssien kannalta yksi iso malli voi viedä enemmän muistia käyttäjän koneelta kuin pieni, mikäli käyttäjä tekee vain yhdessä yhteistyötilassa töitä. Koulutettujen mallien koko on kuitenkin sen verran pieni, ettei CAD-mallinnukseen kykenevä työasema sellaisesta hidastu. Monesta erillisestä mallista koostuvan järjestelmän ylläpito on lisäksi hankalampaa kuin yhden mallin. Pienimpien yhteistyötilojen mallit kärsivät kaiken päälle tarkkuudesta, kun niiden koulutukseen ei löydy tarpeeksi ajantasaista materiaalia.

Liitetiedostojen yksi ja kaksi taulukoista voi nähdä, että mallien ennustuksen tarkkuus ei kasva huomattavasti kolmen ensimmäisen ennusteen jälkeen. Aiemmassa kappaleessa 4.4.2 pohdittiin vaihtoja työkalun ehdottamien ennustuksien määrille, ja kuvioiden näyttämät lukemat tukevat tätä kolmen vaihtoehdon määrää myös käytännöllisyyden puolelta. Mikäli mallin antamat ennusteet olisivat neljännen ja viidennen vaihtoehdon kohdalla oikein vain pienellä todennäköisyydellä, tuottaisi niiden antamat vaihtoehdot lähinnä turhaa valinnanvaikeutta.

5.1.1 Halkaisijatoleransseille koulutetun mallin suorituskyky

Koska halkaisijamitoille oli oma aineistonsa, mittarit tietysti olivat hieman eriävät, mutta pitkälti samaa luokkaa. Tärkeimpinä mittareina LogLoss 0,93, MicroAccuracy 0,79 ja Top3K 0,90. Aineistoa on huomattavasti vähemmän, mikä vaikuttaa omalta osaltaan tuloksien luotettavuuteen.

Siinä missä pituustoleransseilla oli 317 tunnistetta aineistossa, halkaisijatoleransseja oli vain 47 erilaista, joista kymmenen ensimmäistä kattavat 89,1 % koko aineistosta. Jäljelle jääneitä toleransseja oli kappalemäärältään kutakin alle 30. Aineiston toleransseista h7, H8 ja h9 ovat kolme yleisintä ja ne kattavat 66,3 % aineistosta. Malli on siis kuitenkin tarkempi kuin kolme eniten käytettyä toleranssia.

5.1.2 Pieleen menneet ennustukset

Vaikka malli pääsee tarkkuutta ajatellen yli 93 % tarkkuuteen, tarkoittaa se, että noin seitsemällä kerralla sadasta tulee eteen tilanne, jolloin yksikään ehdotetuista toleransseista ei sovi mittaan. Tutkimalla näitä pieleen menneitä on mahdollista saada tietoa mallin heikkouksista ja mahdollisesti parantaa mallia niiltä osin uusien revisioiden kohdalla. Tarkastelemalla taulukkoa kolme selviää miten epäonnistuneet ennustukset jakautuvat, ja ensimmäinen sekä kolmas yhteistyötila sisältävät eniten vääriä ennustuksia.

Taulukko 3. Väärin menneiden ennustusten määrät yhteistyötiloittain.

Yhteistyötila	Väärien määrä	Väärät, %	Tol. lkm. koko aineisto	% koko aineisto
CB 1	346	57,19 %	18264	39,69 %
CB 2	34	5,62 %	17165	37,30 %
CB 3	191	31,57 %	8693	18,89 %
CB 4	34	5,62 %	1893	4,11 %

Väärin menneet ennustukset jakautuvat pitkälti suurimman ja kolmanneksi suurimman yhteistyötilan välille. Osittain tätä selittää toleranssien määrä eri tiloissa, mutta väärien ennusteiden määrä ei korreloi täysin aineiston määrän kanssa. Toiseksi suurin yhteistyötila sisältää kahdesta muusta poikkeavia töitä, joten mallin on todennäköisesti helpompi kohdistaa toleranssit siellä oikein. Mikäli tätä ei oteta huomioon, korreloi virheiden määrä taulukon neljä mukaisesti paremmin aineiston määrän kanssa.

Taulukko 4. Väärin menneet ennustukset ilman toiseksi suurinta yhteistyötilaa.

Yhteistyötila	Väärien määrä	Väärät, %	Tol. lkm. koko aineisto	% koko aineisto
CB 1	346	57,19 %	18264	63,31 %
-	-	-	-	-
CB 3	191	31,57 %	8693	30,13 %
CB 4	34	5,62 %	1893	6,56 %

Alla olevaan taulukkoon viisi kerättiin tieto, mikä kertoo minkä toleranssien kohdalla malli on veikannut eniten väärin. Kolmas sarake kertoo toleranssin lukumäärän verrattuna koko aineistoon. Kuten taulukosta voi nähdä, toleranssien määrä aineistossa ei korreloi suoraan virheiden lukumäärän kanssa. Tämän vuoksi virheellistä aineistoa tarkasteltiin vielä tarkemmin.

Taulukko 5. Eniten vääriä ennustuksia aiheuttaneet toleranssit.

NumTolId	Lkm virhe	Toleranssin määrä koko aineistossa	Määrä vääristä	Lkm koko aineisto
38	48	1,56 %	7,93 %	719
3	43	3,67 %	7,11 %	1688
6	33	2,62 %	5,45 %	1205
17	33	1,56 %	5,45 %	720
18	33	1,21 %	5,45 %	559
5	31	5,64 %	5,12 %	2595
40	28	1,02 %	4,63 %	471
32	18	9,85 %	2,98 %	4531
9	17	12,55 %	2,81 %	5774
164	17	1,61 %	2,81 %	741
44	15	0,83 %	2,48 %	384
16	15	0,52 %	2,48 %	238
24	12	3,59 %	1,98 %	1652
27	12	0,58 %	1,98 %	267
31	11	1,03 %	1,82 %	476

Taulukko 6.Viidentoista yleisimmän virheellisiä ennustuksia aiheuttavan toleranssin vertailu yhteistyötiloittain.

Collab Space / NumTolID	CB 1	CB 2	CB 3	CB 4
38	14	0	28	6
3	18	3	21	1
6	22	0	8	3
17	22	0	11	0
18	24	0	7	2
5	16	0	14	1
40	20	0	7	1
32	10	3	5	0
9	10	0	6	1
164	0	17	0	0
44	14	1	0	0
16	6	1	4	4
24	5	0	6	1
27	4	0	8	0
31	9	0	1	1

Taulukossa kuusi on jaettu virheet jokaiseen yhteistyötilaan erikseen. Kuten aiemmin näimme, yhteistyötila kaksi sisältää suhteellisen vähän virheitä aineiston määrään nähden. Tiloissa yksi ja kolme on havaittavissa eniten virheitä. Taulukkoa tarkastellessa on myös nähtävissä pientä korrelaatiota näiden tilojen välillä. Todennäköisesti näissä yhteistyötiloissa käytetään samoja toleransseja, mikä osaltaan aiheuttaa mallille hankaluuksia ennustusten osalta.

Taulukko 7. Korrelaatio virheiden jakautumisessa eri yhteistyötilojen välillä.

CB	**1**	**2**	**3**	**4**
1	0,0 %	-48,5 %	37,2 %	15,9 %
2	-48,5 %	0,0 %	-27,3 %	-26,6 %
3	37,2 %	-27,3 %	0,0 %	54,3 %
4	15,9 %	-26,6 %	54,3 %	0,0 %

Taulukkoon seitsemän on laskettu eri yhteistyötilojen väliltä löytyvä korrelaatio virheiden ilmentymän osalta. Suurempi luku tarkoittaa, että virheiden sijainti ja määrä korreloi lineaarisesti keskenään kahden yhteistyötilan välillä.

Taulukko 8. Eniten virheitä aiheuttaneiden toleranssien määrät koko aineistossa.

Collab Space / NumTolID	CB 1	CB 2	CB 3	CB 4
38	176	0	449	94
3	1172	57	371	88
6	1038	7	107	53
17	413	14	293	0
18	361	27	139	32
5	2370	0	215	10
40	275	0	176	20
32	345	4044	96	46
9	2122	3565	80	7
164	8	725	8	0
44	286	63	35	0
16	145	8	60	25
24	67	30	1542	13
27	108	8	148	3
31	397	57	12	10

Lisäselvityksen vuoksi taulukkoon kahdeksan kerättiin eniten virheitä aiheuttaneiden toleranssien määrät jokaisesta yhteistyötilasta, jolloin tiedon avulla toi tarkistaa eri toleranssien sekä virheiden esiintymisen yhteistyötiloittain.

Näiden tietojen pohjalta voi tehdä johtopäätöksen, että eniten virhe-ennustuksia tulee, kun samat toleranssit ovat yleisesti käytössä useammassa kuin yhdessä yhteistyötilassa. Toleransseja, joiden kohdalla tehtiin virheellisiä ennustuksia, oli yhteensä 110 erilaista. Taulukoissa esiintyvät 15 eniten virheistä löytynyttä toleranssia muodostavat kaikista virhe-ennusteista 60,5 %. Loppujen virheellisten joukosta suurin osa oli sellaisia toleransseja, mitä löytyi aineistosta todella niukasti, jolloin malli ei ole todennäköisesti löytänyt säännönmukaisuuksia ominaisuuksien ja toleranssien välille.

5.2 Työkalun tulevaisuus ja uudet mahdollisuudet

Seuraava vaihe olisi muokata nykyinen työkalun koulutusputki modulaariseksi ja automaattiseksi. Tällöin se voisi hakea automaattisesti valitun aikavälin sykleissä uusimmat piirustukset ja niiden avulla kouluttaa uuden mallin. Tätä uutta mallia verrattaisi suorituskyvyltään vanhaan, jolloin näistä suorituskykyisempi jää työkalun käyttöön. Tietoaineiston kohdalla voisi testata logaritmista muutosta ja katsoa parantaako aineiston tasaaminen eri ominaisuuksien välillä suorituskykyä. Koulutusputkeen voisi lisätä tietueiden suodattamisen lukumäärän avulla, jolloin kohinaa aiheuttavat tietueet voisi poistaa aineistosta. Mikäli tiettyä toleranssia käytetään vain todella harvoin, se on turhaan mukana heikentämässä mallin suorituskykyä. Aineistoa voisi esikäsitellä kattavammin muillakin tavoin, kun tästä työvaiheesta on kertynyt työn myötä lisää ymmärrystä.

Uusia mahdollisuuksia on käyty läpi tarkemmin liitteessä neljä.

5.3 Huomioitavaa työkalun käytöstä

Sille on syynsä, miksi piirustuksilla tulee olla tekijä, tarkastaja ja hyväksyjä. Useamman silmäparin avulla on tarkoitus minimoida inhimillisten virheiden mahdollisuus. Mitoitus ja oikeiden toleranssien valinta käyttötarkoituksen mukaan on suunnittelijan vastuulla, eikä tätä tehtävää tule ulkoistaa täysin tietokoneelle.

Työkalun on tarkoitus toimia suunnittelijan tukena, eikä sitä ole tarkoitus käyttää painikkeena, mistä valitaan aina ensimmäinen ehdotus koska se on ollut aikaisemmin valitun mitan kaltaisissa tilanteissa se todennäköisin. Se ei ota huomioon esimerkiksi kappaleeseen käytettäviä valmistusmenetelmiä, sen toiminnallisuutta tai esimerkiksi toisesta piirustuksesta selviäviä sovitteita, joiden avulla kappaleet tulisi liittää toisiinsa. Toisin sanoen, konteksti kokonaisuudesta puuttuu. Malli ei myöskään voi luoda täysin uutta, sillä tämänhetkiset tekoälysovellukset eivät sellaiseen pysty. Mikäli jotain vaihtoehtoa, esimerkiksi tiettyä pituustoleranssia ei löydy aineistosta, työkalu ei voi sellaista luoda lennosta.

Suurin osa piirustusten mitoista ei sisällä toleransseja, vaikka työkalu niitä niihin voi ehdottaakin. Kaavitusta aineistosta löytyy yli 2,36 miljoonaa mittaa, mutta esimerkiksi pituustoleranssin omaavia mittoja on hieman päälle 46 000. Tämä tarkoittaa, että kaikista mitoista noin 1,9 prosenttia sisältää pituustoleranssin. Kaavitut piirustukset sisältävät monenlaisia muitakin piirustuksia kuten työ-, osa- tai kokoonpanopiirustuksia, joihin kaikkiin ei toleransseja merkitä. Osa piirustuksista ei tarvitse kuin yleistoleranssimerkinnän.

6 Pohdinta

6.1 Tutkimuskysymysten vastaukset

Idea oli kerätä työkalun valmistuttua tilastotietoa sen käytöstä. Tämän myötä työhön pystyi käyttämään kvantitatiivista lähestymistapaa. Tilastoja käyttöstatiikasta kerättiin muun muassa siitä, onko käyttäjä valinnut useammasta ehdotuksesta toleranssin vaiko jonkun muun vaihtoehdon ja kuinka usein mitäkin ominaisuutta suunnittelijat käyttävät.

Tutkimustyötä varten kehitimme kolme kysymystä, mihin haettiin ratkaisua. Työn aikana kävi selväksi, ettei kysymyksiin ei ole yksiselitteistä vastausta, sillä jokainen malli ja sen kehitysprosessi on ainutlaatuinen. Kysymyksiin löytyi silti vastauksia, joiden avulla uusien mallien kehitys tulee olemaan huomattavasti nopeampaa ja helpompaa.

Mitä kaikkea toimivan koneoppimismallin koulutus vaatii?

Koneoppimismallia rakentaessa tiettyyn ongelmaan tärkeintä on ymmärtää ratkaistava ongelma sekä ratkaisuun vaadittava tietoaineisto sekä sen laatu. Mikäli mallille syöttää heikkolaatuista aineistoa, ei voi odottaa luotettavia tuloksia. Aineiston analysointiin ja esikäsittelyyn käytetty aika ei ole hukkaan heitettyä.

Mallin koulutus vaatii ymmärrystä ongelmasta, joka halutaan ratkoa, jotta siihen on mahdollista löytää ratkaisuun sopivia keinoja. Mikäli ongelmaan ei saada ratkaisua perinteisen data-analyysin menetelmin, voi ongelmaa koettaa ratkaista valmiiden algoritmien ja mallien avulla. Mikäli tämä ei ratkaise ongelmaa, on mahdollista yrittää kehittää omia algoritmeja mallin tueksi.

Valmiin mallin tekemien päätöksien ymmärtäminen on tärkeää, ja erilaiset analyysit auttavat tässä. On mahdollista käyttää ohjelmistoista valmiiksi löytyviä analyysityökaluja tai keksiä jokaisen mallin kohdalla juuri sen toimintaa avaavia ratkaisuja.

Kuinka suuri työ on saada integroitua edellä mainittu malli tuotantoympäristöön?

Mallien käyttöönottoprosessi vaihtelee jokaisen työn kohdalla riippuen siitä, millä alustalla malli on koulutettu sekä millaiseen ympäristöön se on tarkoitus integroida. Tämän työn kohdalla molemmat, niin malli kuin kohdeympäristö ovat kirjoitettu C#:llä, mikä teki käyttöönotosta helppoa. Tämän lisäksi mallit ovat tiedostokooltaan pieniä, joten ne sai helposti integroitua ECT:n asennuspaketin mukaan. Mikäli olisi kyse monimutkaisesta mallista, mikä on tallennettu pilvipalvelimelle, olisi käyttöönotto todennäköisesti hankalampaa.

Suunnitellessa mallin integrointia tulee huomioida mallin elinkaari. Tätä varten voi kysyä kysymyksiä, jotka rajaavat kehitystä oikeaan suuntaan. Mikäli mallia on tarvetta uudelleenkouluttaa, tehdäänkö uuden aineiston keräämistä varten automaattinen ohjelma vai kerätäänkö sitä jostain käsin? Tukeeko algoritmi uudelleenkoulutusta vai pitääkö se kouluttaa alusta asti uudelleen? Kuinka mallin suorituskykyä analysoidaan ajan kuluessa, jotta se pysyy oleellisena myös käyttöönoton jälkeen? Tämän vuoksi esimerkiksi MLOps:n kaltaisten käytänteiden tunteminen on hyvä apu.

Millä tavalla työkalun tuottavuutta voidaan mitata?

Työkalun luotettavuutta ja suorituskykyä voidaan mitata joko alustojen omilla mittareilla tai tekemällä omia testejä alkuperäistä ongelmaa silmällä pitäen. Tämän työn kohdalla oman testiohjelman teko oli tarpeen, jotta sai kolmen ensimmäisen vaihtoehdon määrät ja todennäköisyydet. Sen avulla saatiin myös tieto, ettei AutoML:n tekemä malli ollut niin hyvä kuin mittarit antoivat kertoa. Työkalun tarkkuuden ollessa hyväksyttävällä tasolla voi olettaa, että sen käyttö säästää suunnittelijan aikaa.

Ajansäästöä ajatellen mallin hyötyä on hankala arvioida. Mikäli tätä tarkastelee puhtaasti sen kannalta, kauanko vaikka halkaisijatoleranssin sijoitus mittaan vie perinteisin keinoin verrattuna työkaluun, ajansäästö on joitain sekunteja yhtä käyttökertaa kohden. Toisaalta työkalu voi ehdottaa toleranssia, minkä vuoksi suunnittelija olisi joutunut etsimään vanhoista piirustuksista tietoa ja täten säästää huomattavasti enemmän.

Sekuntimääräisesti ajansäästöä voi laskea käyttöstatiikan avulla. Aina suunnittelijan käyttäessä työkalun ominaisuuksia tietokantaan tulee tästä merkintä, joiden pohjalta on muodostettu kuvaajia ominaisuuksien käyttöstatiikasta.

6.2 Työn luotettavuuden ja eettisyyden arviointi

Työhön käytetyt lähteet vaihtelevat blogipostauksista tieteellisiin papereihin. Tieteelliset, vertaisarvioidut tutkimukset ovat oletettavasti luotettavia lähteitä. Niidenkin kohdalla tulee kiinnittää huomiota tutkimusten rahoittajiin ja miettiä, vaikuttaako se mahdollisesti tutkimustuloksiin. Osa verkosta löytyvistä tekstilähteistä viittaa työssään lähteisiin, mutta osassa on vain kirjoittajan omat kerronnat ja tulkinnat aiheesta. Tällaisten kirjoitusten kohdalla sisältö on ollut kuitenkin linjassa esimerkiksi kirjalähteiden kanssa, joten pidän lähteitä luotettavina.

Työkalun suorituskykyä on vertailtu useaan otteeseen erikokoisilla aineistoilla. Suorituskykyä mittaavat mittarit on selitetty ML.NET:n dokumentoinnissa kattavasti auki ja esimerkiksi kaikkien algoritmien toimintaan on perustellut lähteet. Tämän lisäksi testausta varten on tehty oma ohjelma, minkä avulla saa aineiston sekä työkalun ennusteiden tiedot kattavasti talteen.

Eettisyyden kannalta työkalun tekoon käytetyt aineistot eivät ole yksiöitävissä millään tapaa. Tietokantaan tallennetut tiedot eivät ole yhdistettävissä kehenkään eikä niihin ole mahdollista päästä mallin kautta käsiksi, sillä siihen tallentuu ainoastaan painotuksia mitan eri ominaisuuksille.

6.3 Käyttäjästatiikka

Käyttäjästatiikkaa on niukasti riippuen monesta eri syystä, päällimmäisenä tällä hetkellä yrityksessä Catialla tehtävien projektien vähäinen määrä. Tämän niukkuuden vuoksi ajansäästöllisiä vaikutuksia suuressa mittakaavassa on hankala spekuloida. Parhaillaan tehokkuuden nosto ja ajankäytön tehokkuus moninkertaistuu, mutta pahimmassa skenaariossa työkalu jää ainoastaan käyttämättä, mikä ei muuta nykytilannetta heikompaan suuntaan.

Kuvion 35 käyttöstatiikasta voi nähdä, kuinka työkalun käyttö on lähtenyt nousuun vuodenvaihteen jälkeen. Työkalun kohdalla ei ole tämän mukaan havaittavissa runsasta muutosvastarintaa, vaan käyttäjät ovat ottaneet työkalun hyvin vastaan. Työkalusta on koetettu suunnitella mahdollisimman käyttäjäystävällinen, jotta suunnittelijoiden ei tarvitse epäröidä sen käyttöönottoa. Työkalua varten on myös laadittu ohjevideo, mikä kertoo työkalun toiminnasta ja käytöstä.

Edellä mainitussa kuviossa kuvataan pituusmittojen sijoitusta mittoihin, ja työajansäästönä on käytetty maltillisesti muutamaa sekuntia per toleranssin asetus. Tähän perusteluna se, että toleranssin valinta työkalusta on hiukan nopeampaa kuin Catian omilla työkaluilla. Mahdollista ajansäästöä vanhojen piirustuksien etsimisen sijaan on mahdotonta arvioida, koska se vaihtelee jokaisen piirustuksen kohdalla. Toisaalta henkilötasolla kertyvät pienetkin tehostukset voivat olla organisaatiotasolla jopa henkilötyövuosia.

Kuvio 35. Käyttöstatiikasta tuotu arvio työajansäästöstä ja käyttökerroista.

6.4 Ristiriitoja toleransseissa lähteiden välillä

6.4.1 Catian ja SFS:n ristiriita N9 toleranssin kohdalla

Testatessa halkaisijamittojen toleranssia todettiin toleranssin N9 kohdalla yli 500 mm halkaisijalla ristiriitaa. SFS:n mukaan N9 toleranssit ovat ensimmäistä riviä (nimellismitta ≤ 3) lukuun ottamatta nollan ja ylärajamitan välillä. Yli 500 mm halkaisijalla alarajamitat muuttuvat negatiivisiksi, jolloin

kumpikin toleranssiarvo on negatiivinen. Catialla yli 500 mm halkaisijalla toleranssien alarajamitta pysyi nollassa.

Kummassakin lähteessä toleranssiarvo pysyi samana, mutta peruseromitta muuttui. Tutkittua koulutusaineistoa kävi selväksi, ettei sieltä löydy yhtään kertaa N9 toleranssia käytettynä yli 500 mm halkaisijalla, joten ne jätettiin työkalun suosituksista pois. Toleranssin ollessa harvinainen työkalu epätodennäköisesti sitä koskaan suosittelisi halkaisijoihin, joten sellaisessa tapauksessa, jossa sille on tarvetta, suunnittelija joutuisi sen joka tapauksessa laittamaan käsin.

6.4.2 Catian ja SFS:n & Valtasen pieniä ristiriitoja

Vertaillessa toleranssitaulukoita muutaman toleranssin kohdalta löytyi pieniä eroavaisuuksia. Catia todennäköisesti laskee toleranssit kaavan avulla taulukkoarvon sijaan tiettyjen toleranssien kohdalla. Esimerkkeinä H17 & H18 toleransseilla Catia antaa arvot +1 mm, 0 mm ja +1,4 mm, 0 mm, kun halkaisija on kaksi millimetriä. Valtasen taulukkokirjan sivulla 668 ja SFS-EN ISO 268-2 sivulla 32 tämän kyseisen toleranssin kohdalla on tyhjää.

SFS:n standardin sivulla 16 lukee, että tarvittaessa paikat, mitkä on jätetty tyhjiksi, voidaan laskea standardin SFS-EN ISO 268-1 taulukoiden avulla (SFS-EN ISO 268-2, 16). Todennäköisesti Catia laskee nämä toleranssit täysin käyttäen kaavoja, jolloin jokaiselle mitalle löytyy omat toleranssit. Halkaisija on niin pieni, ettei siinä ole kappaleen mittasuhteisiin nähden oikein järkeä, minkä vuoksi sitä ei muista lähteistä todennäköisesti löydy.

Toinen ristiriitojen lähde on melko varmasti näppäilyvirhe taulukoita tehdessä. SFS 286-2 asiakirjassa toleranssin A9 arvot ovat +775 ja +60 µm, kun sen Catian ja Valtasen mukaan arvojen kuuluisi olla +775 ja +660 µm. (SFS-EN ISO 268-2, 24, Valtanen 2019, 666.)

6.5 Tekoäly: Missä ollaan ja mihin ollaan menossa?

ML.NET ja muut alustat tuovat koneoppimisen laajan käyttäjäkunnan ulottuville tekemällä siitä mahdollista ilman syvempää kokemusta data-analyysista. Valmiit algoritmit toimivat ohjelmoijille tuttujen funktioiden ja metodien tapaan, jolloin niitä on helppo käyttää. Kaikissa tilanteissa valmiit algoritmit tai aktivointifunktiot eivät ole päteviä ongelman ratkaisuun, mutta niitä on kuitenkin kehitetty laajalti erilaisiin tilanteisiin. Näistä syistä uskon erilaisten tekoälysovellusten yleistyvän runsaalla tahdilla lähitulevaisuudessa.

6.5.1 Mikä on nykytilanne?

Erilaisten generatiivisten tekoälymallien käyttö on kasvanut viime aikoina, ja niitä sovelletaan laajasti useilla eri aloilla. CES 24 tapahtumassa tekoälyä oli lisätty arkipäiväisiin tuotteisiin, kuten tyynyihin ja hammasharjoihin (Elliot & Khan 2024). Suurin osa näistä tekoälyistä vaikuttaa päälle liimatuilta markkinointitermeiltä, muistuttaen 2010-luvulla vallinnutta algoritmien ylistystä.

Nimestä poiketen nykyisiltä tekoälyiltä ei löydy älykkyyttä. Ihmiset sopeutuvat melko hyvin erilaisista muutostilanteista, mutta koneoppimismalleille tämä on vielä hankalaa. Jos esimerkiksi koulutettaisi koneoppimismalli teollisuuteen jonkun koneen ennakoivaa kunnossapitoa varten, siihen kerätään tietoa toiminnasta erilaisten sensoreiden avulla. Jos joku sensori vaihdetaan vastaavaan mutta hieman erilaiseen tuotteeseen, voi mallin joutua kouluttamaan uusiksi, koska sensoreilta saatava tieto on hiukan erilaista.

Mutta mallit kehittyvät ripeää tahtia: Markkinoille on alkanut ilmestyä multimodaalisia tekoälymalleja, jotka osaavat käsitellä niin tekstiä, kuvia, ääntä ja videota. Mallien kehitys on itse asiassa niin ripeää, että kirjoitusprosessin aikana pohdintaa on kirjoitettu useaan otteeseen uusiksi generatiivisten mallien ottaessa suuria harppauksia eteenpäin. Näitä malleja on alettu integroida jopa ihmistä muistuttaviin robotteihin, jotka suoriutuvat erilaisista hienomotoriikkaa vaativista tehtävistä.

Näiden generatiivisten mallien kohdalla on silti havaittavissa yksi ongelma; vaikka ne tekevätkin laadukasta jälkeä monelta osin, lopputulos on harvoin sellainen kuin alkuperäinen näkemys on ollut. Moneen käyttöön tällainen tarpeeksi hyvä on varmasti riittävää. Mutta oli kyseessä kuvat, teksti taikka koodi, tekoäly on vielä kaukana ihmisen korvaamisesta.

6.5.2 Minne ollaan menossa?

Erilaisten tekoälysovellusten käyttö tulee kasvamaan. Koneoppimiseen liittyvien markkinoiden on ennustettu kohoavan maailmanlaajuisesti yli 771,38 miljardiin dollariin vuoteen 2032 mennessä (Machine Learning Market Size To Surpass… n.d.). Ajatustyötä vaativien töiden osalta tällä hetkellä näyttää siltä, ettei työpaikat katoa. Mutta työnteko voi tehostua niin, että työpaikkojen määrä tulee vähenemään. Tällä hetkellä generatiiviset mallit on rakennettu tuottamaan materiaalia muttei tarkistamaan niiden faktapohjaa. Lisäksi nykyiset mallit alkavat hallusinoida helposti, eli esittämään vääriä väitteitä faktoina. Tämän vuoksi mallit voivat toimivat parhaiten ihmisen tukena työtehtäviä tehdessä, jolloin asiantuntija voi tarkistaa tulokset, mutta mallien kehittyessä tilanne voi muuttua.

Mallien koulutuksessa tulisi ottaa huomioon erilaiset kognitiiviset vinoumat. Koska tekoälysovellukset ovat pohjimmiltaan ihmisen tekemiä ja kouluttamia, tulee pitää mielessä heidän tarkoitusperänsä (Valtiala 2022). Erilaiset pahansuopia aikeita varten kehitetyt huijaukset ja syväväärennökset ovat alkaneet levitä maailmalla (Leprince-Ringuet 2020), joten lähdekriittisyyden merkitys korostuu. Tietoa tulee olemaan saatavilla enemmän kuin koskaan, mutta tästä massasta luotettavan tiedon erottaminen tulee olemaan entistä suuremmassa roolissa.

Teollisuutta ajatellen koneälyä aletaan hyödyntämään laajalti prosessien optimoinnin työkaluna. Erilaiset tekoälyä käyttävät robotit tulevat olemaan tulevaisuudessa iso osa tuotantoa ja esimer-

kiksi laaduntarkkailussa konenäköä hyödyntävät sovellukset tulevat tehostamaan toimintaa. Tulevaisuudessa myös koneensuunnittelua varten tulee varmasti erilaisia sovelluksia, jotka tehostavat toimintaa tavoilla, mitä ei vielä osata ennustaa.

6.5.3 Luova ajattelu ja tekoäly

Shead (2022) on maininnut artikkelissaan IBM:llä olleen jo vuonna 1979 puhe kuinka tietokoneita ei voi koskaan pitää vastuussa teoistaan ja sen vuoksi ne eivät koskaan saisi tehdä johtopäätöksiä. Koneoppimismallit pohjautuvat matematiikkaan ja todennäköisyyksiin. Kuten O'Neil (2017, 38) mainitsee teoksessaan, koneoppimismallit käyttävät koulutuksessaan menneisyyteen pohjautuvaa aineistoa, minkä pohjalta ne oppivat tunnistamaan kaavoja.

Kuinka erilaiset tekoälysovellukset vaikuttavat pitkällä aikavälillä luovaan ajatteluun? Tällä hetkellä ne ovat hyviä jokapäiväisten ja tylsien tehtävien automatisointiin, mutta ne vievät myös paljon ajatustyötä pois ihmisiltä. Uutta projektia aloittaessa luonnosten teko kuvien tai tekstin avulla on helppo ulkoistaa mallille antamalla käsky tehdä luonnos aiheesta, vaikka sen tekeminen itse voisi olla parempi oman luovuuden kannalta. Tällöin pääsisi käyttämään ajatustyötä uusiin näkökulmiin ja mitä tulisi priorisoida. Toisaalta malli voi luoda jotain, mitä itse ei olisi koskaan tullut ajatelleeksi.

Sama koskee esimerkiksi oman ajankäytön priorisointia. Mikäli pyytää mallia tiivistämään pitkän artikkelin tai tekstin niin ajatustyö jää välistä pois, kun ei tarvitse miettiä artikkelin kirjoittajan näkökulmaa, tekstin pääasioita taikka tutkimuksen kohdalla käytettyjen metodien luotettavuutta. Tällöin luottaa sokeasti algoritmien määrittelemiin todennäköisyyksiin. Mallit voivat tehdä asioista paljon helpompaa mutta joutuuko siinä pidemmän päälle epäedulliseen asemaan, kun on ulkoistanut oman elämän hallintaa ja ajatustyön koneelle. Voiko tällainen lyhyen aikajakson tehokkuuden parannus vaikuttaa pitkällä aikavälillä negatiivisesti, kun ihminen antaa tekoälylle vapaat kädet oman luovuuden suhteen?

Lähteet

3 Types of Machine Learning You Should Know. 2023. Artikkeli Coursera.com sivustolle. Viitattu 2.1.2024. https://www.coursera.org/articles/types-of-machine-learning

Aaltonen, R. 2023. Palkka jopa 5600 e/kk ja varma työpaikka viiden vuoden päästä – Näitä aloja opiskelevan ei tarvitse pelätä tulevaa. Artikkeli kauppalehden nettisivuilla. Viitattu 25.1.2024. https://www.kauppalehti.fi/uutiset/palkka-jopa-5600-e-kk-ja-varma-tyopaikka-viiden-vuoden-paasta-naita-aloja-opiskelevan-ei-tarvitse-pelata-tulevaa/447a6436-eb66-4b70-8969-ffa701676c0b

AI vs. Machine Learning vs. Deep Learning vs. Neural Networks: What's the difference?. 2023. IBM: n data ja AI tiimin tekemä artikkeli eri tekniikoista. Viitattu 4.1.2024. https://www.ibm.com/blog/ai-vs-machine-learning-vs-deep-learning-vs-neural-networks/

Albaryak, Ö. Jomâa, M. Selçuk, S. Ünal, P. 2021. A Workflow for Synthetic Data Generation and Predictive Maintenance for Vibration Data. Julkaisu synteettisen aineiston käytöstä mallien koulutuksessa Research Gaten sivustolla. Viitattu 16.1.2024. https://www.researchgate.net/publication/354865626_A_Workflow_for_Synthetic_Data_Generation_and_Predictive_Maintenance_for_Vibration_Data

Artificial intelligence (AI) vs. machine learning (ML). N.d. Artikkeli AI:n ja ML:n eroista Google Cloudin sivustolla. Viitattu 9.1.2024. https://cloud.google.com/learn/artificial-intelligence-vs-machine-learning

Artificial Intelligence and Machine Learning Projects Are Obstructed by Data Issues. 2019. Dimensional Research yrityksen tekemä selvitys tekoälyn ja koneoppimisen koulutuksesta yrityksissä. Viitattu 4.1.2024. https://cdn2.hubspot.net/hubfs/3971219/Survey%20Assets%201905/Dimensional%20Research%20Machine%20Learning%20PPT%20Report%20FINAL.pdf

Artificial Intelligence (AI) vs. Machine Learning. N.d. Artikkeli AI:n ja ML:n eroista Columbia Universityn verkkosivuilla. Viitattu 2.1.2024. https://ai.engineering.columbia.edu/ai-vs-machine-learning/

Bahmani, MJ. 2023. Understanding LigthGBM Parameters (and How to Tune Them). Artikkeli Neptune.ai sivustolla. Viitattu 11.1.2024. https://neptune.ai/blog/lightgbm-parameters-guide

Barber, P. Sole, M. E. Turner, I. N.D. Mechanical engineering design, learning from the past to design better future? Ote Design and Technology Education: An International Journal kokoelmasta. Viitattu 19.3.2024. https://files.eric.ed.gov/fulltext/EJ1323752.pdf

Borisov, V. Haug, J. Kasneci, G. Leemann, T. Pawelczyk, M ja Seßler, K. 2022. Deep Neuran Networks and Tabular Data: A Survey. IEEE:n tekemä tutkimus neuroverkkojen suoriutumisesta tabulaarisella aineistolla. Viitattu 16.2.2024. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9998482

Breiman, L. 2001. Random Forest. Tieteellinen paperi liittyen satunnaispuualgoritmiin. Viitattu 16.1.2024. https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf

Brownlee, J. 2020. Data Preparation for Machine Learning. v. 1.2. Viitattu 8.1.2024. https://www.google.fi/books/edition/Data_Preparation_for_Machine_Learning/uAPuDwAAQBAJ

Brownlee, J. 2020. What is the Difference Between Test and Validation Datasets? Artikkeli Machine Learning Masteryn sivuilla. Viitattu 8.1.2024. https://machinelearningmastery.com/difference-test-validation-datasets/

Chaudhary, V. Davydov, E. Ebner, D. Golovin, D. Holt, G. Phillips, T. Sculley, D & Young, M. N.d. Machine Learning: The High-Interest Credit Card of Technical Debt. Viitattu 16.1.2024. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/43146.pdf

Data Preparation and Feature Engineering in ML, N.d. Artikkeli Google Developer sivustolla. Viitattu 5.1.2024. https://developers.google.com/machine-learning/data-prep/

Datasäädös: Komissio ehdottaa toimenpiteitä oikeudenmukaisen ja innovatiivisen datatalouden edistämiseksi. 2022. Lehdistötiedote. Viitattu 17.1.2024. https://ec.europa.eu/commission/presscorner/detail/fi/ip_22_1113

Data Transformation and Feature Engineering in Python. 2021. Artikkeli Visual-Design.net sivustolla aineiston käsittelystä. Viitattu 16.1.2024 https://www.visual-design.net/post/data-transformation-and-feature-engineering-in-python

Dembla, G. 2020. Intuition behind Log-Loss score. Artikkeli, joka avaa LogLoss mittaria towardsdatascience.com sivustolla. Viitattu 15.1.2024. https://towardsdatascience.com/intuition-behind-log-loss-score-4e0c9979680a

Dissanayake, K. 2023. Machine Learning Algorithms(8) – Decision Tree Algorithm. Artikkeli towardsdatascience.com verkkosivulla. Viitattu 16.1.2024. https://towardsdev.com/machine-learning-algorithms-8-decision-tree-algorithm-533b6926ddbb

Domun, Y. 2016. Aircraft Design Process Overview. Artikkeli lentokoneen suunnitteluprosessiin liittyen engineerclick.com sivustolla. Viitattu 19.3.2024. https://www.engineeringclicks.com/aircraft-design-process/

Dong, G. Liu, H. 2018. Feature Engineering for Machine Learning and Data Analytics. Boca Raton: CRC Press.

Eledath, B. N.d. Intro to Feature Engineering for Machine Learning with Python. Viitattu 16.1.2024. https://www.learndatasci.com/tutorials/intro-feature-engineering-machine-learning-python/

Elliot, M. Kahn, I. 2024. AI at CES 2024: Take a Look at the Coolest Tech From the Show. Viitattu 6.2.2024. https://www.cnet.com/pictures/coolest-ai-tech-ces-2024-weve-seen-so-far/13/

Esposito, D. Esposito, F. 2022. Programming ML.NET. E-kirja. Pearson Education, Inc. Viitattu 15.1.2024. https://books.google.fi/books?id=HtfPE-AAAQBAJ&pg=PT89&lpg=.PT89#v=onepage&q&f=false

Ghosh, S. 2023. A Comprehensive Guide to Data Preprocessing. Artikkeli Neptune.Ai sivustolla. Viitattu 5.1.2024. https://neptune.ai/blog/data-preprocessing-guide

Haenlein, M. Kaplan, A. 2019. Siri, Siri, in my hand: Who's the fairest in the land? On the interpretations, illustrations, and implications of artificial intelligence. Artikkeli sciencedirect sivustolla. Viitattu 11.1.2024. https://www.sciencedirect.com/science/article/abs/pii/S0007681318301393

Hirsjärvi, S., Remes, P., Sajavaara, P., & Sinivuori, E. 2009. Tutki ja kirjoita. 15. uud. p. Helsinki: Tammi.

Holden, H. 2019. Design for Manufacturing and Assembly (DFM/A) or Predictive Engineering. Artikkeli Altiumin sivuilla. Viitattu 30.1.2024. https://resources.altium.com/p/design-manufacturing-and-assembly-dfma-or-predictive-engineering

Hyvä tieteellinen käytäntö ja sen loukkausepäilyjen käsitteleminen Suomessa 2023. Tutkimuseettisen neuvottelukunnan julkaisuja 2/2023. Viitattu 18.1.2024

Introduction to Large Language Models. N.d. Google Developers resurssi, jossa kerrotaan suurien kielimallien toiminnasta. Viitattu 23.2.2024. https://developers.google.com/machine-learning/resources/intro-llms

Introduction to tensors. N.d. TensorFlow dokumentaatio, jossa selitetään mitä tensorit ovat. Viitattu 3.1.2024. https://www.tensorflow.org/guide/tensor

Introduction to PyTorch tensors. N.d. PyTorch dokumentaatio, jossa selitetään PyTorchin tensoreista. Viitattu 3.1.2024. https://pytorch.org/tutorials/beginner/introyt/tensors_deeper_tutorial.html

ISO-toleranssit. N.d. Artikkeli Camcut.fi sivustolla, missä kerrotaan ISO toleransseista. Viitattu 22.1.2024. https://www.camcut.fi/tuki/iso-toleranssit/

Kananen, J. 2010. Opinnäytetyön kirjoittamisen käytännön opas. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Kananen, J. 2012. Kehittämistutkimus opinnäytetyönä. Jyväskylä: Jyväskylän ammattikorkeakoulu.

Karaali, G. 2023. Artificial Intelligence, Basic Skills, and Quantitative Literacy. Numeracy 16, Iss. 1. Article 9. Viitattu 15.1.2024. https://digitalcommons.usf.edu/cgi/viewcontent.cgi?article=1438&context=numeracy

Ke, G. Finley, T. Wang, T. Chen, W. Ma, W. Ye, Q. Liu, T-Y. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. Tieteellinen paperi. Viitattu 9.1.2024. https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf

Kelleher, J.D. Tierney, B. 2018. Data Science. Cambridge, Massachusetts: MIT Press.

Kelleher, J. D. 2019. Deep Learning. Cambridge, Massachusetts: MIT Press.

Deike, T. Kempe-Mueller, S. Righini, E. 2023. 5 Things To Know About The European Data Act. Artikkeli littyen Euroopan Unionin datasäädös, joka käsittelee dataa. Viitattu 23.1.2024. https://www.lw.com/en/people/admin/upload/SiteAttachments/Five-Things-To-Know-About-the-European-Data-Act.pdf

Keras: The high-level API for TensorFlow. N.d. TensorFlow ohjelmointirajapinnan dokumentaatio. Viitattu 3.1.2024. https://www.tensorflow.org/guide/keras

Kershaw, N. Ormont, J. Schonning, N. Victor, Tabladillo, M. Y. Quintatilla, L .2022. Interpret model predictions using Permutation Feature Importance. Artikkeli Microsoftin ML.NET koneoppimisalustan opastussivuilla. Viitattu 16.1.2024. https://learn.microsoft.com/en-us/dotnet/machine-learning/how-to-guides/explain-machine-learning-model-permutation-feature-importance-ml-net

Kershaw, N. Kirch, S. Ormont, J. Schonning, N. Victor, Y. Quintatilla, L .2023. Evaluate your ML.NET model with metrics. ML.NET koneoppimisalustalta löytyviä mittareita mallin suorituskyvyn arviointiin. Viitattu 9.1.2024. https://learn.microsoft.com/en-us/dotnet/machine-learning/resources/metrics

Kurama, V. 2020. A Comprehensive Guide to the DataLoader Class and Abstractions in PyTorch. Viitattu 5.1.2024. https://blog.paperspace.com/dataloaders-abstractions-pytorch/

Kämäräinen, J. 2023. Koneoppimisen perusteet. Tallinna: Printon Tükikoda.

Leprince-Ringuet, D. 2020. Evil AI: These 20 are the most 20 dangerous crimes that artificial intelligence will create. Artikkeli zdnet sivustolla. Viitattu 8.2.2024. https://www.zdnet.com/article/evil-ai-these-are-the-20-most-dangerous-crimes-that-artificial-intelligence-will-create/

Louridas, P. 2020. Algorithms. The MIT Press essential knowledge series. Cambridge, Massachusetts: MIT Press.

Machine learning definition in detail. N.d. Artikkeli SAP:n verkkosivuilla. Viitattu 2.1.2024. https://www.sap.com/products/artificial-intelligence/what-is-machine-learning.html

Machine Learning Market Size To Surpass USD 771.38 Bn By 2032. N.d. Artikkeli Precedence Researchin verkkosivuilla. Viitattu 16.1.2024. https://www.precedenceresearch.com/machine-learning-market

Maimon, O. Rokach, L. 2008. Data Mining with Decision Trees: Theory and Applications. World Scientific Publishing. Viitattu 15.1.2024. https://doc.lagout.org/Others/Data%20Mining/Data%20Mining%20with%20Decision%20Trees_%20Theory%20and%20Applications%20%282nd%20ed.%29%20%5BRokach%20%26%20Maimon%202014-10-23%5D.pdf

ML.NET – An open source and cross-platform machine learning framework. N.d. Microsoftin ML.NET etusivu. Viitattu 11.1.2024. https://dotnet.microsoft.com/en-us/apps/machinelearning-ai/ml-dotnet

ML.NET Model Builder. N.d. Microsoftin dotnet dokumentaatio. Viitattu 4.1.2024. https://dotnet.microsoft.com/en-us/apps/machinelearning-ai/ml-dotnet/model-builder

MLOps: Continuous delivery and automation pipelines in machine learning. 2023. Artikkeli Googlen pilvipalveluiden arkkitehtuurikeskuksen sivuilla. Viitattu 16.1.2024. https://cloud.google.com/architecture/mlops-continuous-delivery-and-automation-pipelines-in-machine-learning

Nabi, J. 2018. Machine Learning – Multiclass Classification with Imbalanced Dataset. Artikkeli towardsdatascience.com verkkosivuilla. Viitattu 15.1.2024. https://towardsdatascience.com/machine-learning-multiclass-classification-with-imbalanced-data-set-29f6a177c1a

Nabil, M. 2023. Decision Trees vs. Neural Networks. Artikkeli medium.com verkkosivuilla. Viitattu 15.1.2024. https://medium.com/@navarai/decision-trees-vs-neural-networks-ff46f47ce0a0

Nelson, D. 2020. Mikä on päätöspuu? Artikkeli Uniute.ai sivustolla. Viitattu 15.1.2024. https://www.unite.ai/fi/mikä-on-päätöspuu/

Omardonia. 2023. The Data Wizard's Guide to Preprocessing and Feature Engineering. Artikkeli Mediumin verkkosivuilla. Viitattu 5.1.2024. https://levelup.gitconnected.com/the-data-wizards-guide-to-preprocessing-and-feature-engineering-e44ac2268739

O'Neil, C. 2017. Weapons of Math Destruction. Ensimmäinen painos. Iso-Britannia: Clays Ltd.

Palo, L. 2023. Koneensuunnittelun opas. Opinnäytetyö, AMK. LAB-ammattikorkeakoulu, Insinööri (AMK), konetekniikka. Viitattu 2.1.2024. https://www.theseus.fi/bitstream/handle/10024/791261/Palo_Leevi.pdf

Pearson, S. A. Ulrich, K. T. 1993. Does Product Design Really. Determine 80 % of Manufacturing Cost? Massachusetts Institute of Technologyn tekemä työasiakirja. https://dspace.mit.edu/bitstream/handle/1721.1/47202/doesproductdesig00ulri.pdf?sequence=1&isAllowed=y

Pere, A. 2021. Koneenpiirustus 1 & 2. 13. p. Espoo: Kirpe.

Prepare data for building a model. 2023. Artikkeli Microsoft Learn sivustolla. Viitattu 5.1.2024. https://learn.microsoft.com/en-us/dotnet/machine-learning/how-to-guides/prepare-data-ml-net

Pykes, K. 2022. PyTorch Tutorial: Building a Simple Neural Network From Scratch. Artikkeli Datacamp.com sivustolla. Viitattu 16.1.2024. https://www.datacamp.com/tutorial/pytorch-tutorial-building-a-simple-neural-network-from-scratch

Roser, M. 2022. The brief history of artificial intelligence: The world has changed fast – what might be next? Artikkeli ourworldindata.com sivustolla. Viitattu 22.12.2023. https://ourworldindata.org/brief-history-of-ai

Ross, W. D. 1995. Aristotle. 2. p. Taylor & Francis Group. ISBN 9780415120685.

Sakhuja, V. 2022. Learn how to build a Machine Learning Model in 8 steps. Artikkeli Medium.com sivustolla. Viitattu 2.1.2024. https://medium.com/@varun.sja/learn-how-to-build-a-machine-learning-model-in-8-steps-7e62a06526e3

scikit-learn dokumentaatio. N.d. Avoimen lähdekoodin koneoppimisalustan dokumentaatio. Viitattu 4.1.2024. https://scikit-learn.org/stable/getting_started.html

SFS-EN ISO 268-1/AC: Geometrinen tuotemäärittely (GPS). Pituusmittojen ja toleranssien ISO-merkintäjärjestelmä. Osa 1: Toleranssien, eromittojen ja sovitteiden perusteet. Helsinki: Suomen Standardisoimisliitto SFS. Korjattu 8.4.2013. Viitattu 22.1.2024. https://janet.finna.fi/ , SFS Online.

SFS-EN ISO 268-2: Geometrinen tuotemäärittely (GPS). Pituusmittojen toleranssien ISO-merkintäjärjestelmä. Osa 2: Reikien ja akselien perustoleranssiluokkien ja rajaeromittojen taulukot. Helsinki: Suomen Standardisoimisliitto SFS. Vahvistettu 11.10.2010. Viitattu 22.1.2024. https://janet.finna.fi/ , SFS Online.

Shead, S. 2022. Machines are getting better at writing their own code. But human-level is 'light years away'. Artikkeli CNBC: n verkkosivuilla. Viitattu 11.1.2024. https://www.cnbc.com/2022/02/08/deepmind-openai-machines-better-at-writing-their-own-code.html

Supe, A. Vyas, R. 2007. Multiple choice questions: A literature review on the optimal number of options. Intian kansallisessa lääketieteellisessä aikakausilehdessä julkaistu tutkimus. Viitattu 24.1.2024. https://www.researchgate.net/publication/23468587_Multiple_choice_questions_A_literature_review_on_the_optimal_number_of_options

Tekoäly on hyvä renki, muttei isäntä – Maavoimien tutkimuskeskus pohti tekoälyä. 2020. Artikkeli maanpuolustuskorkeakoulun sivuilla. Viitattu 9.1.2024. https://maanpuolustuskorkeakoulu.fi/-//1950813/tekoaly-on-hyva-renki-muttei-isanta-maavoimien-tutkimuskeskus-pohti-tekoalya

Toleranssit ja sovitteet. N.d. Kouvolan Ammattiopiston koulutusaineistoa lentokoneasentajan koulutukseen. Viitattu 23.1.2024. https://peda.net/ksao/oppimisympäristö/koulutusalat/tjla/alv/lentokoneasentaja/lph/m7hjkl/7sjv/7tjs:file/download/66f33b283b632433e8538c841b656dfd6910dad7/7.6%20Toleranssit%20ja%20sovitteet.pdf

Valtanen, E. 2019. Tekniikan taulukkokirja. 22. p. Mikkeli: Genesis-Kirjat.

Valtiala, M. 2022. Tekoälystandardien suomenkieliset termit. Viitattu 15.1.2024. https://sfs.fi/tekoalyn-suomenkieliset-termit/

VanderPlas, J. 2016. Python Data Science Handbook. Viitattu 15.1.2024. https://jakevdp.github.io/PythonDataScienceHandbook/

Wadawadagi, V. 2023. TensorFlow vs PyTorch: Deep Learning Frameworks [2024]. Artikkeli knowledgehut.com sivustolla. Viitattu 3.1.2024. https://www.knowledgehut.com/blog/data-science/pytorch-vs-tensorflow#advantages-and-disadvantages-of%C2%A0tensorflow%C2%A0

What is Data Preparation? N.d. Amazonin tietopankki koneoppimis- ja tekoäly-ympäristöistä. Viitattu 5.1.2024. https://aws.amazon.com/what-is/data-preparation/

When AI flags the ruler, not the tumor – and other arguments for abolishing the black box (VB Live). 2021. Artikkeli VentureBeatin verkkosivuilla. Viitattu 8.1.2024. https://venturebeat.com/business/when-ai-flags-the-ruler-not-the-tumor-and-other-arguments-for-abolishing-the-black-box-vb-live/

Whitney, D. E. 1988. Manufacturing by Design. Kirjaan perustuva artikkeli Harvard Business Review sivulla. Viitattu 30.1.2024. https://hbr.org/1988/07/manufacturing-by-design

Yasar, K. Lewis, S. 2022. What is PyTorch? Artikkeli techtaget.com sivustolla. Viitattu 3.1.20224. https://www.techtarget.com/searchenterpriseai/definition/PyTorch

Ye, A. 2020. When and Why Tree-Based Models (Often) Outperform Neural Networks. Artikkeli Towards Data Sciencen verkkosivuilla. Viitattu 15.1.2024. https://towardsdatascience.com/when-and-why-tree-based-models-often-outperform-neural-networks-ceba9ecd0fd8

Yegulalp, S. 2017. Facebook brings GPU-powered machine learning to Python. Uutinen InfoWorld sivustolla. Viitattu 3.1.2024. https://www.infoworld.com/article/3159120/facebook-brings-gpu-powered-machine-learning-to-python.html

Zewe, A. 2023. A method for designing neural networks optimally suited for certain tasks. Uutinen Massachusetts Institute of Technologyn sivuilla. Viitattu 2.1.2024. https://news.mit.edu/2023/method-designing-neural-networks-optimally-suited-certain-tasks-0330

Zewe, A. 2023. Unpacking the “black box” to build better AI models. Uutinen Massachusetts Institute of Technologyn sivuilla. Viitattu 4.3.2024. https://news.mit.edu/2023/stefanie-jegelka-machine-learning-0108

Liitteet

Liite 1. Yksittäisten yhteistyötilojen avulla koulutettujen mallien suorituskyky.

CB 1			
PredictionIndex	Amount		
1	2700	73,91 %	73,91 %
2	511	13,99 %	87,90 %
3	106	2,90 %	90,80 %
4	50	1,37 %	92,17 %
5	41	1,12 %	93,29 %
6	24	0,66 %	93,95 %
7	20	0,55 %	94,50 %
0	19	0,52 %	95,02 %
8	16	0,44 %	95,46 %
11	15	0,41 %	95,87 %
NumToId	Amount		
4	1221	33,42 %	33,42 %
5	461	12,62 %	46,04 %
9	430	11,77 %	57,81 %
3	255	6,98 %	64,79 %
6	218	5,97 %	70,76 %
17	95	2,60 %	73,36 %
31	77	2,11 %	75,47 %
32	77	2,11 %	77,58 %
18	68	1,86 %	79,44 %
22	61	1,67 %	81,11 %

CB 2		top 3	98,36 %
PredictionIndex	Amount		
1	3259	94,93 %	94,93 %
2	100	2,91 %	97,84 %
4	22	0,64 %	98,48 %
3	18	0,52 %	99,00 %
5	7	0,20 %	99,20 %
6	4	0,12 %	99,32 %
8	4	0,12 %	99,44 %
9	4	0,12 %	99,56 %
0	3	0,09 %	99,65 %
20	2	0,06 %	99,71 %
NumToId	Amount		
32	806	23,48 %	23,48 %
9	713	20,77 %	44,25 %
170	332	9,67 %	53,92 %
165	279	8,13 %	62,05 %
136	272	7,92 %	69,97 %
7	253	7,37 %	77,34 %
163	230	6,70 %	84,04 %
172	174	5,07 %	89,11 %
164	153	4,46 %	93,57 %
23	91	2,65 %	96,22 %

CB 3			
PredictionIndex	Amount		
1	1285	73,89 %	73,89 %
2	242	13,92 %	87,81 %
3	51	2,93 %	90,74 %
4	22	1,27 %	92,01 %
0	19	1,09 %	93,10 %
6	19	1,09 %	94,19 %
5	13	0,75 %	94,94 %
8	11	0,63 %	95,57 %
7	10	0,58 %	96,15 %
12	8	0,46 %	96,61 %
NumToId	Amount		
4	869	49,97 %	49,97 %
24	300	17,25 %	67,22 %
38	85	4,89 %	72,11 %
3	76	4,37 %	76,48 %
17	62	3,57 %	80,05 %
5	47	2,70 %	82,75 %
18	32	1,84 %	84,59 %
40	31	1,78 %	86,37 %
32	28	1,61 %	87,98 %
27	26	1,50 %	89,48 %

CB 4			
PredictionIndex	Amount		
1	305	80,47 %	80,47 %
2	31	8,18 %	88,65 %
3	9	2,37 %	91,02 %
8	5	1,32 %	92,34 %
10	4	1,06 %	93,40 %
0	3	0,79 %	94,19 %
7	3	0,79 %	94,98 %
4	3	0,79 %	95,77 %
5	3	0,79 %	96,56 %
9	2	0,53 %	97,09 %
NumToId	Amount		
4	267	70,45 %	70,45 %
38	15	3,96 %	74,41 %
3	13	3,43 %	77,84 %
6	11	2,90 %	80,74 %
18	8	2,11 %	82,85 %
32	7	1,85 %	84,70 %
40	7	1,85 %	86,55 %
22	7	1,85 %	88,40 %
16	5	1,32 %	89,72 %
75	4	1,06 %	90,78 %

Liite 2. Työkalun tarkkuuden vertailu koulutusaineiston mukaan.

Training set	----------	All data	
Testing set from	----------	All data	
PredictionIndex	Amount	%	Sum
1	7464	78,44 %	78,44 %
2	955	10,04 %	88,48 %
3	259	2,72 %	91,20 %
4	127	1,33 %	92,53 %
5	113	1,19 %	93,72 %
6	51	0,54 %	94,26 %
7	50	0,53 %	94,79 %
0	38	0,40 %	95,19 %
8	32	0,34 %	95,53 %
9	29	0,30 %	95,83 %
Top 3 preds amount	8678		

Training set	----------	4 big CB	
Testing set from	----------	All data	
PredictionIndex	Amount	%	Sum
1	7700	80,92 %	80,92 %
2	988	10,38 %	91,30 %
3	207	2,18 %	93,48 %
4	105	1,10 %	94,58 %
5	65	0,68 %	95,26 %
6	41	0,43 %	95,69 %
7	37	0,39 %	96,08 %
9	27	0,28 %	96,36 %
8	27	0,28 %	96,64 %
0	27	0,28 %	96,92 %
Top 3 preds amount	8895		

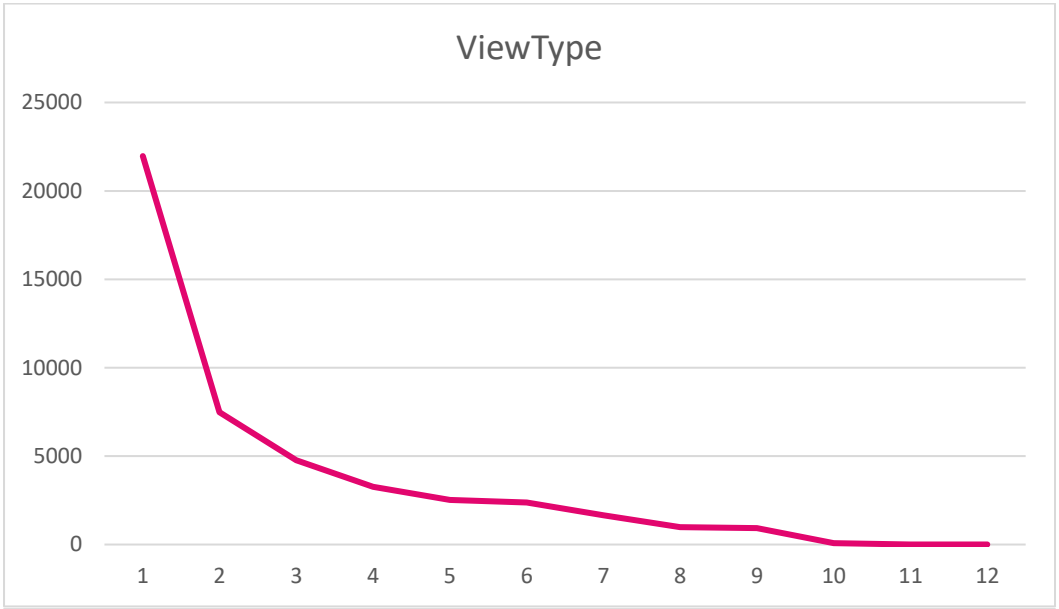
Only tolerances			
4 biggest CB	----------		All data
NumTolId	Amount		
4	2434	25,58 %	25,58 %
9	1176	12,36 %	37,94 %
32	904	9,50 %	47,44 %
5	537	5,64 %	53,08 %
3	389	4,09 %	57,17 %
7	322	3,38 %	60,55 %
170	311	3,27 %	63,82 %
24	311	3,27 %	67,09 %
165	288	3,03 %	70,12 %
6	268	2,82 %	72,94 %
Top 3 amount	4514		

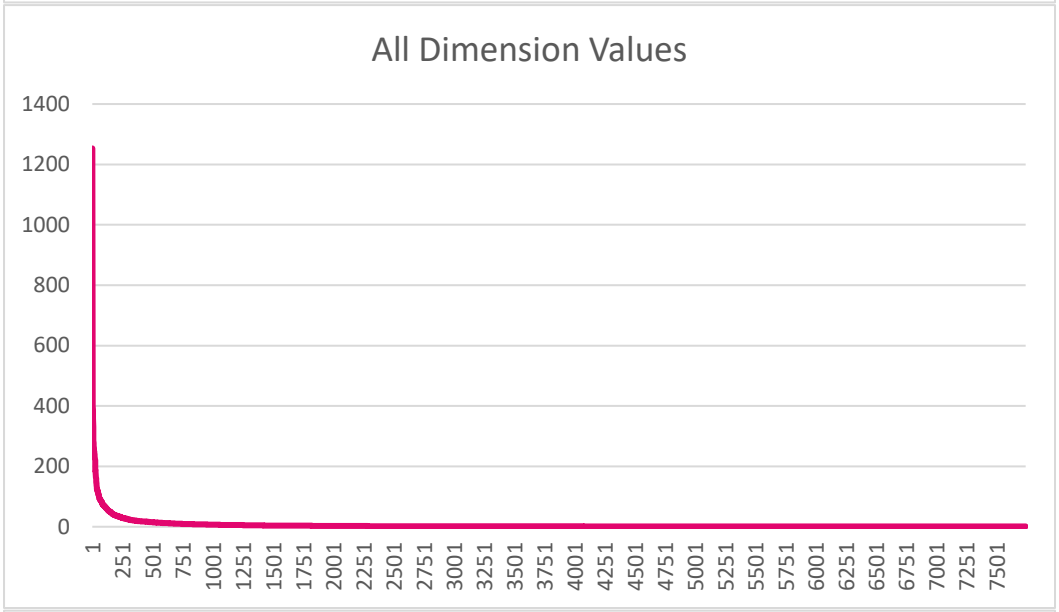
Training set	----------	All data	
Testing set from	----------	4 big CB	
PredictionIndex	Amount	%	Sum
1	7490	81,39 %	81,39 %
2	991	10,77 %	92,16 %
3	219	2,38 %	94,54 %
4	94	1,02 %	95,56 %
5	61	0,66 %	96,22 %
7	43	0,47 %	96,69 %
6	39	0,42 %	97,11 %
8	23	0,25 %	97,36 %
9	20	0,22 %	97,58 %
10	15	0,16 %	97,74 %
Top 3 preds amount	8700		

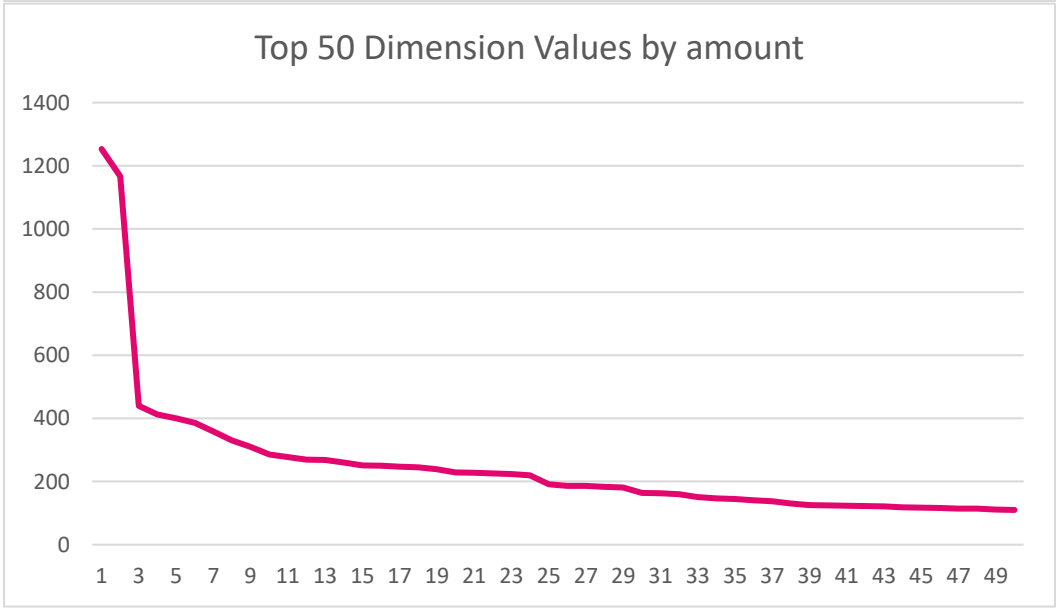
Training set	----------	4 big CB	
Testing set from	----------	4 big CB	
PredictionIndex	Amount	%	Sum
1	7614	82,73 %	82,73 %
2	965	10,49 %	93,22 %
3	196	2,13 %	95,35 %
4	85	0,92 %	96,27 %
5	42	0,46 %	96,73 %
6	40	0,43 %	97,16 %
7	39	0,42 %	97,58 %
8	27	0,29 %	97,87 %
9	18	0,20 %	98,07 %
14	15	0,16 %	98,23 %
Top 3 preds amount	8775		

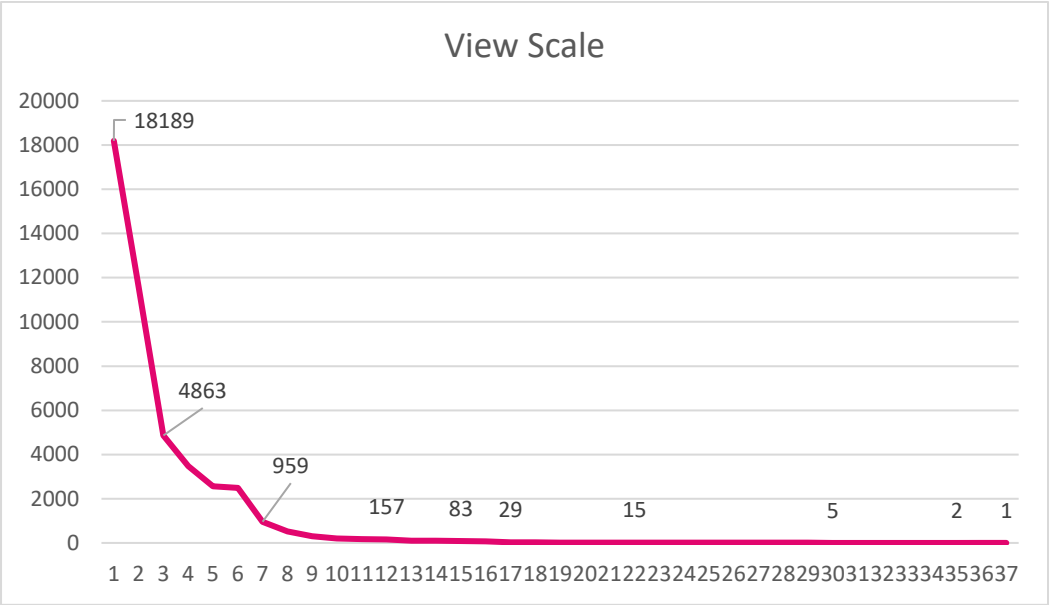
Only tolerances			
4 biggest CB	----------		4 big CB
NumTolId	Amount		
4	2444	26,56 %	26,56 %
9	1142	12,41 %	38,97 %
32	884	9,61 %	48,58 %
5	559	6,07 %	54,65 %
3	325	3,53 %	58,18 %
170	318	3,46 %	61,64 %
24	311	3,38 %	65,02 %
7	310	3,37 %	68,39 %
165	281	3,05 %	71,44 %
136	266	2,89 %	74,33 %
Top 3 amount	4470		

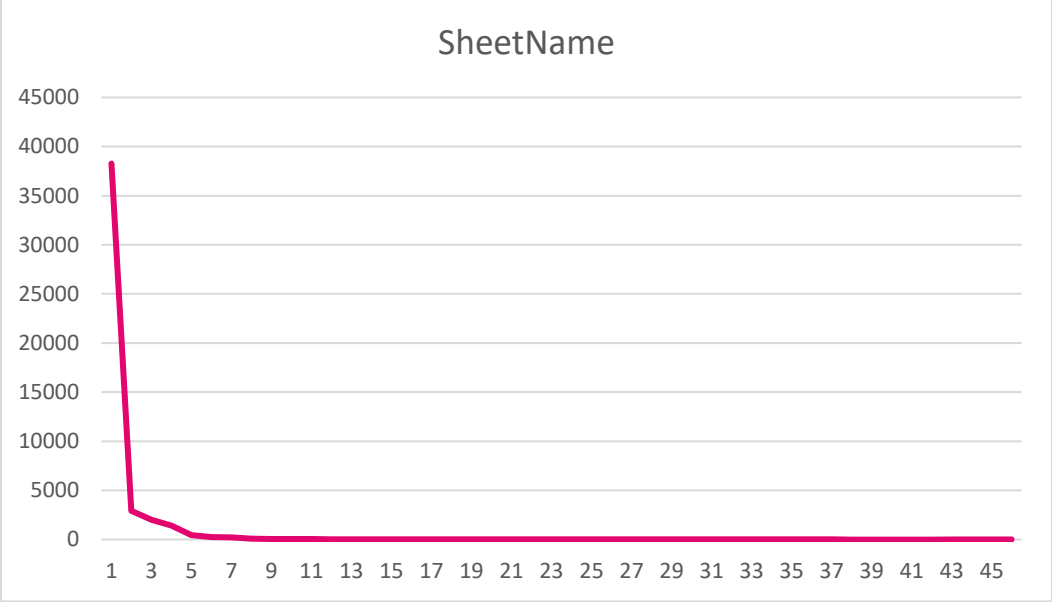
Liite 3. Tietoaineiston visualisointi

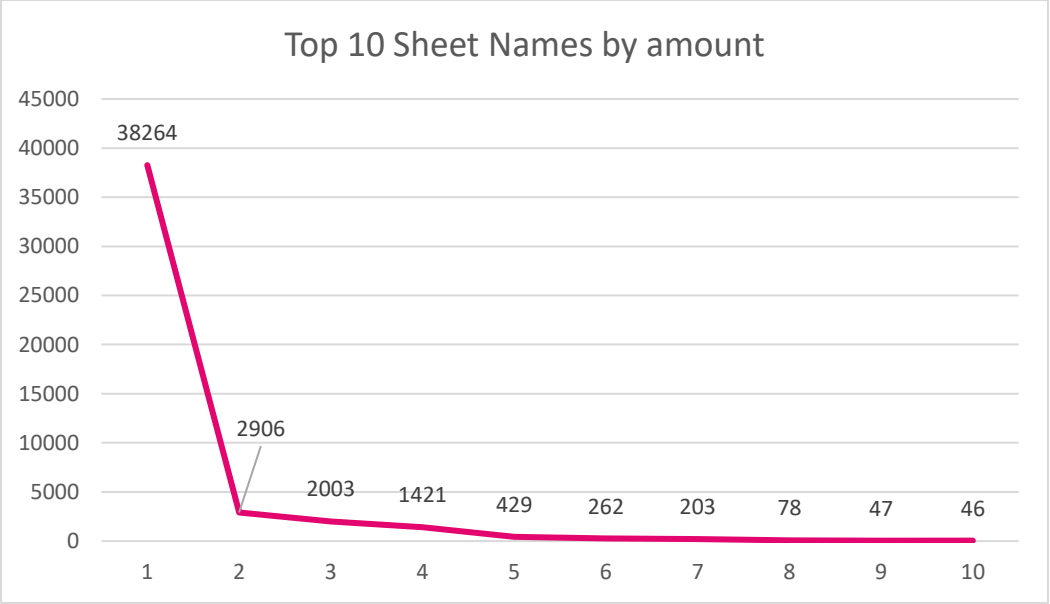


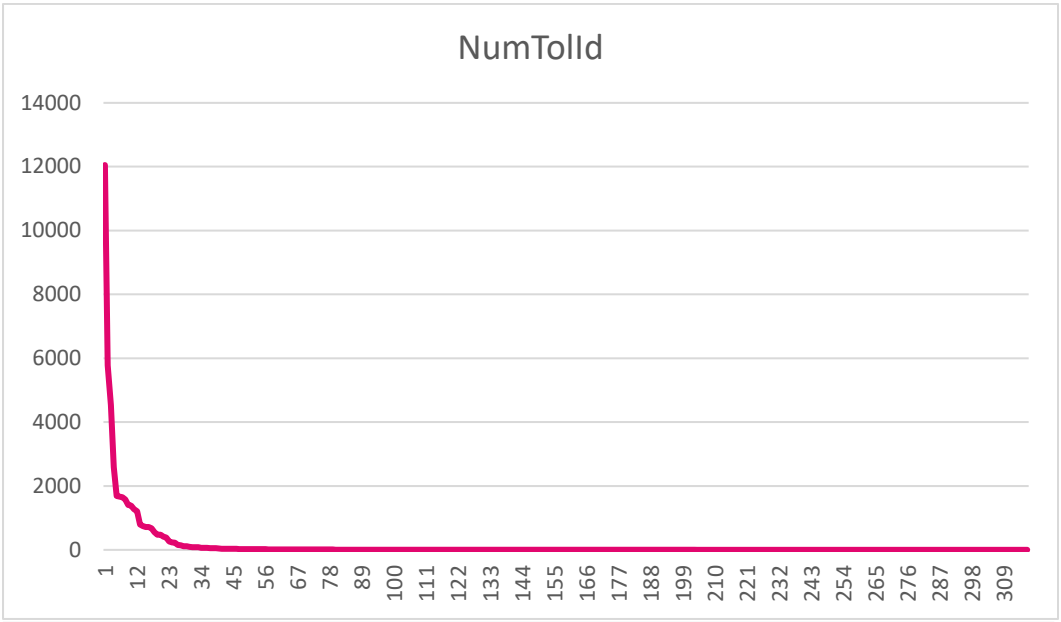


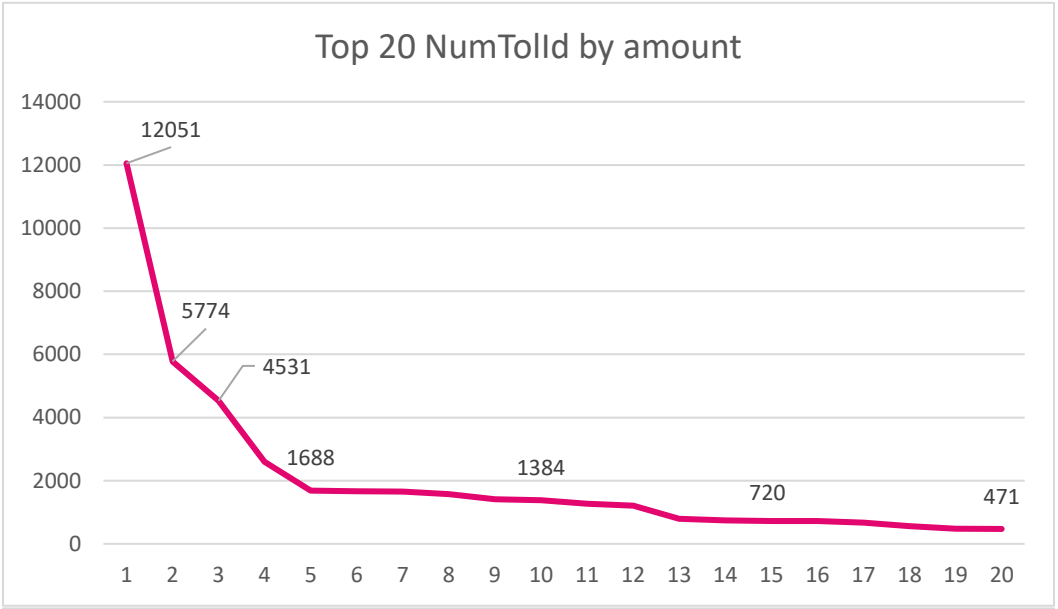


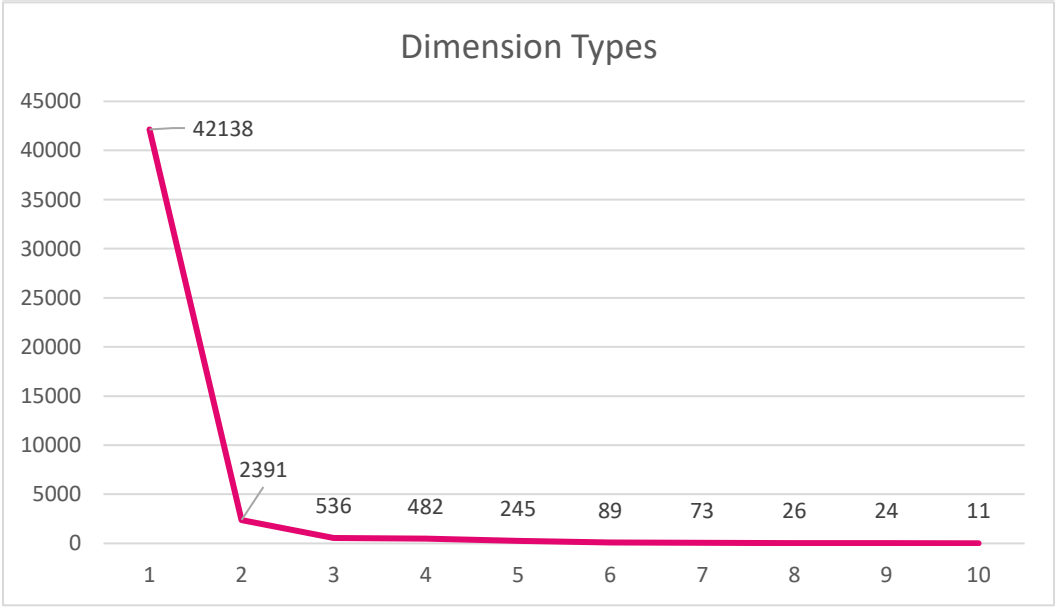

View Scale
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
18189
4863
959
157
83
29
15
5
2
1
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37

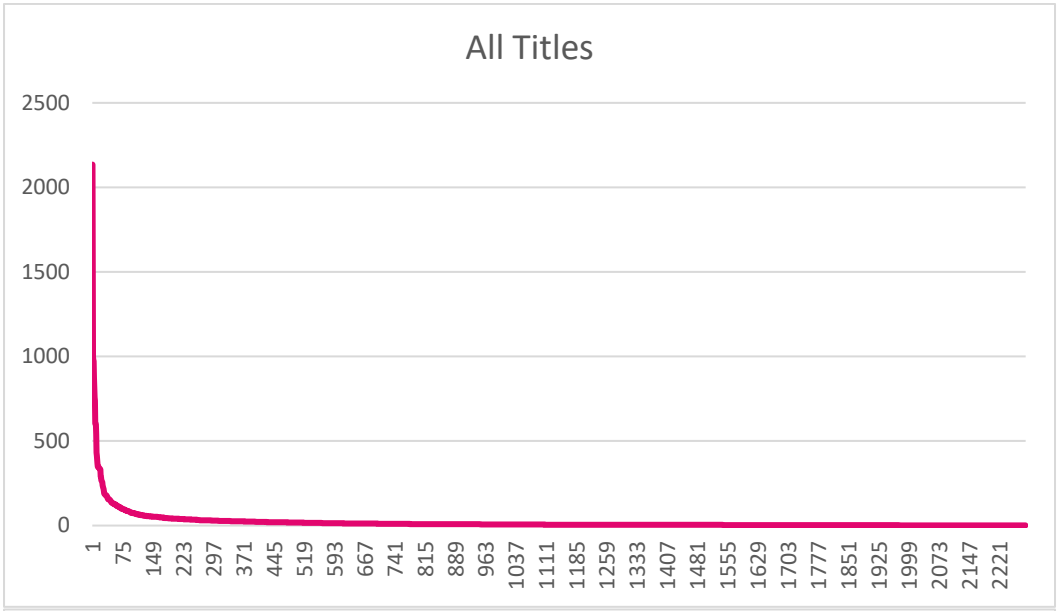

SheetName
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45

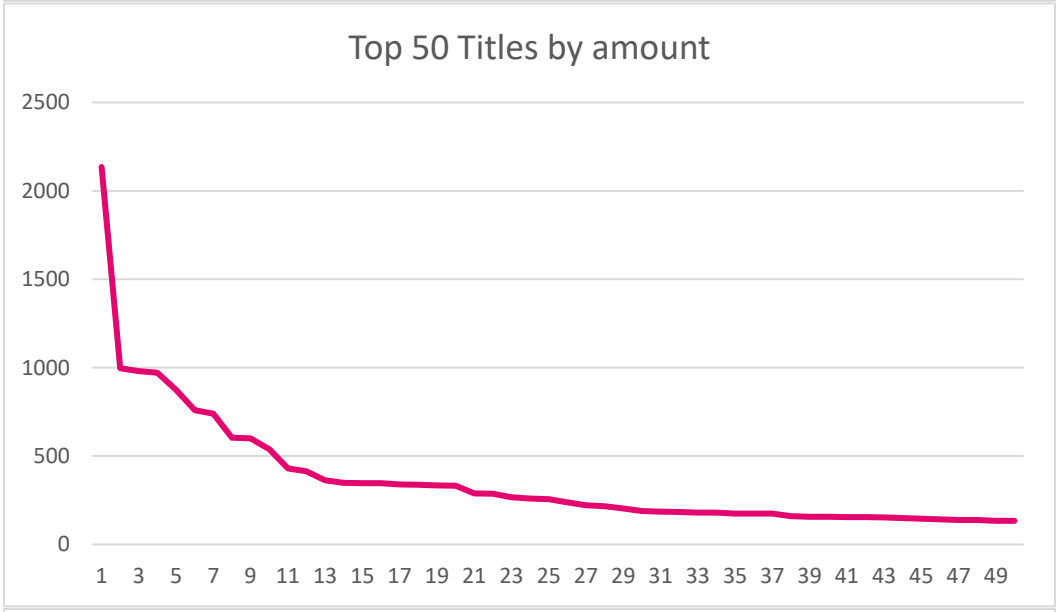

Top 10 Sheet Names by amount
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
38264
2906
2003
1421
429
262
203
78
47
46
1 2 3 4 5 6 7 8 9 10

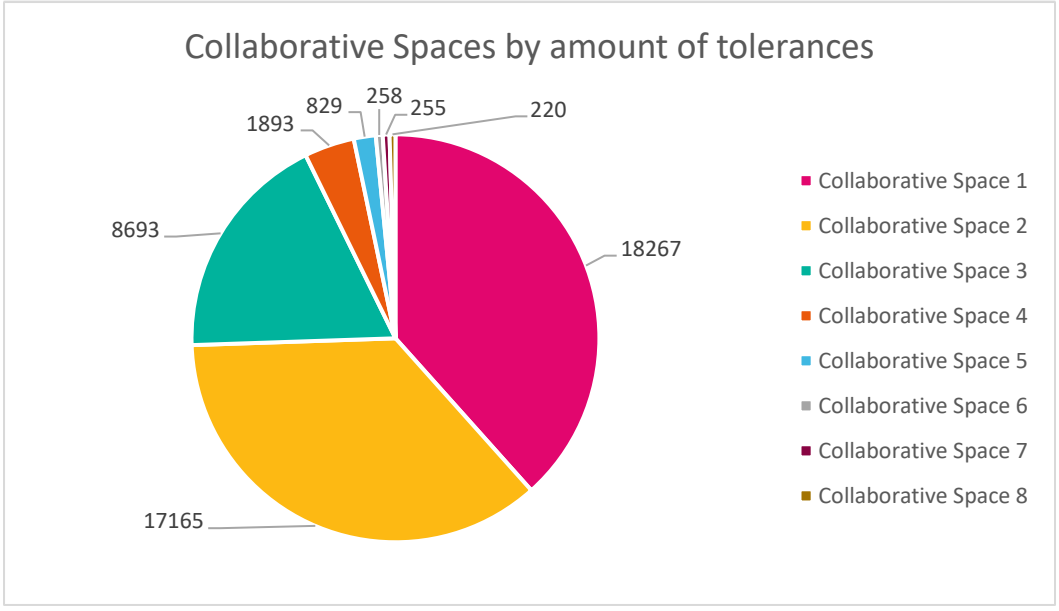

NumTollId
14000
12000
10000
8000
6000
4000
2000
0
1
12
23
34
45
56
67
78
89
100
111
122
133
144
155
166
177
188
199
210
221
232
243
254
265
276
287
298
309


Top 20 NumTollId by amount
14000
12000
10000
8000
6000
4000
2000
0
12051
5774
4531
1688
1384
720
471
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20


Dimension Types
45000
40000
35000
30000
25000
20000
15000
10000
5000
0
42138
2391
536
482
245
89
73
26
24
11
1
2
3
4
5
6
7
8
9
10


All Titles
2500
2000
1500
1000
500
0
1
75
149
223
297
371
445
519
593
667
741
815
889
963
1037
1111
1185
1259
1333
1407
1481
1555
1629
1703
1777
1851
1925
1999
2073
2147
2221


Top 50 Titles by amount
2500
2000
1500
1000
500
0
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49


Collaborative Spaces by amount of tolerances
829
258
255
220
1893
8693
18267
17165
Collaborative Space 1
Collaborative Space 2
Collaborative Space 3
Collaborative Space 4
Collaborative Space 5
Collaborative Space 6
Collaborative Space 7
Collaborative Space 8

Liite 4. Työkalun tulevaisuuden mahdollisuuksia (salassa pidettävä).