Tampereen ammattikorkeakoulu

Koneoppimisen hyödyntäminen opintojakson suorittamisen ennustamisessa

Marko Vahtola

OPINNÄYTETYÖ
Helmikuu 2023

Dataosaamisen ja tekoälyn ylempi tutkinto-ohjelma

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tekoälyn ja dataosaamisen ylempi tutkinto

MARKO VAHTOLA:
Koneoppimisen hyödyntäminen opintojaksojen suorittamisen ennustamisessa

Opinnäytetyö 54 sivua, joista liitteitä 1 sivua
Maaliskuu 2023

Tiedon määrä sekä laskentateho on kasvanut viime vuosina. Samaan aikaan pilvipalveluiden hyödyntäminen tiedon varastoinnissa ja prosessoinnissa on yleistynyt. Kehitys on osaltaan edesauttanut tekoälyn ja varsinkin sen alalajin koneoppimisen soveltamista erilaisten historiatietoon perustuvien ratkaisujen yleistymisessä. Koneoppivia algoritmeja hyödyntämällä datasta voidaan löytää säännönmukaisuuksia ja riippuvaisuuksia, joiden avulla on mahdollista luoda ennusteita.

Tämä opinnäytetyö toteutettiin suomalaiselle korkeakoululle. Korkeakoulun toiveena oli koneoppimisen hyödyntämismahdollisuuksien tutkiminen henkilökohtaisiin opintosuunnitelmiin valittujen ajoitettujen opintojen suorittamisten ennustamisessa. Opinnäytetyön ohessa konkreettisena kehittämistyönä toteutettiin koneoppimiseen perustuva ennustemalli. Ennusteen tarkoituksena on tukea opetukseen ja sen järjestämiseen liittyviä sidosryhmiä suunnittelemaan toimintaansa tehokkaammin. Samalla tutkittiin mahdollisuutta toteuttaa koneoppiminen ja siihen liittyvät prosessit Azure-pilviympäristössä sekä luoda sovellus ennusteen seuraamiseen.

Tutkimustyö toteutettiin konstruktiivisella menetelmällä, jossa tarkoituksena on löytää ratkaisu käytännön ongelmaan. Tutkimuksen tuloksia voidaan käyttää arvioitaessa koneoppimisen hyötyjä päätöksenteon tukena toiminnan suunnittelussa. Tämän lisäksi tuloksissa esitellään toimintatapa koneoppivan sovelluksen toteuttamiseen Microsoft Azure -pilviympäristöä ja Power BI -palvelua hyödyntäen.

Asiasanat: koneoppiminen, azure, pilvilaskenta

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Tekoälyn ja dataosaamisen ylempi tutkinto

MARKO VAHTOLA

Koneoppimisen hyödyntäminen opintojaksojen suorittamisen ennustamisessa

Masters thesis 54 pages, appendices 1 pages
February 2023

The aim of this thesis is to explore the potential of machine learning in enhancing the institutional resource planning process at a higher education institution. The focus is on using machine learning algorithms to produce predictions based on students' personal study plans. The data collected from the study plans will be used to train the machine learning models and to make predictions about the completion of courses.

The Azure cloud platform was chosen as the platform for technical development work related to machine learning in order to take advantage of its scalability and capabilities. The results of this thesis will provide insights into the potential of machine learning in the field of higher education institution resource planning and offer understanding for implementing machine learning in this context.

Overall, this thesis will contribute to the understanding of how machine learning can be utilized in the higher education institutions. The results of this study will provide valuable information for other universities and organizations who are looking to incorporate machine learning into their operations and processes.

Key words: machine learning, azure, cloud computing

SISÄLLYS

1 JOHDANTO ... 7
2 TAVOITE, RAJAUKSET JA TUTKIMUSMENETELMÄ ... 9
2.1 Tavoite ja tutkimuskysymykset ... 9
2.2 Rajaukset ... 9
2.3 Konstruktiivinen tutkimus ... 10
2.4 Konstruktiivisen tutkimukset vaiheet ja raportointi ... 10
3 TEKOÄLY JA KONEOPPIMINEN ... 12
3.1 Tekoäly ... 12
3.2 Koneoppiminen ... 13
3.3 Koneoppivat algoritmit ... 13
3.4 Piirteet ... 14
3.5 Algoritmien ryhmittely ... 14
3.6 Ohjattu oppiminen ja sen ryhmät ... 15
3.7 Ennustemallit ... 15
3.8 Ohjattuun oppimiseen perustuva matemaattinen malli ... 15
3.9 Päätöspuu ... 16
3.10 Päätösmetsä ... 17
3.11 Gradient-Boosted Tree ... 17
4 DATAN JAKAMINEN OSIIN JA KONEOPPIMISEN MITTARIT ... 19
4.1 Koulutus-, testaus- ja validointiryhmät ... 19
4.2 Mallin suoriutuminen ... 19
4.3 Ennusteen mittarit ... 20
4.3.1 Accuracy ... 20
4.3.2 Precision ... 20
4.3.3 Recall ... 21
4.3.4 Roc-kuvaaja ... 21
5 PILVILASKENTA JA OHJELMOINTI ... 22
5.1 Pilvilaskenta ... 22
5.2 Ohjelmointi ... 23
6 KONEOPPIMISEN PROSESSI JA TYÖTEHTÄVÄT ... 24
6.1 Koneoppimisen prosessi ... 24
6.2 Koneoppimiseen liittyvät tehtävänkuvat ... 25
7 KEHITYSTEHTÄVÄ-- ... 27
7.1 Tausta ... 27
7.2 Peppi-järjestelmä ... 27
7.3 Datan vaiheet ... 28

7.4 Tiedon siirtäminen ... 28
7.5 Tietoallas ja Databricks ... 29
7.6 Tietojoukon muodostaminen ... 29
7.7 Piirteiden rakentaminen ... 30
7.8 Datan analysointi ja tilastot ... 31
7.9 Datan käsittely ... 32
7.10 Datan jakauma ja korrelaatio ... 32
7.11 Datan jakaminen ... 34
8 KONEOPPIMISEN SOVELTAMINEN ... 35
8.1 Automl ... 35
8.2 Algoritmien vertailu ... 36
8.3 LightGBM ja piirteiden tärkeyden vertailu ... 38
8.4 Hyperparametrit. ... 39
8.5 Hyperopt ... 40
8.6 MLflow ... 41
8.7 Tuotantomalli ... 42
9 ENNUSTEEN JAKAMINEN ... 43
9.1 Mallin käyttö ... 43
9.2 Power BI ... 44
9.3 Ennusteen jakaminen ja visualisointi ... 44
10 TOIMINTAPROSESSI ... 46
11 POHDINTA JA JOHTOPÄÄTÖKSET ... 47
LÄHTEET ... 50
LIITTEET ... 54
Liite 1. Koneoppimiseen liittyvä kokonaisuus ja prosessin vaiheet ... 54

LYHENTEET JA TERMIT

Data	Tietoa, joka kerätty järjestelmään tai tiedostoon
Massadata	Valtava määrä tietoa, joka on yleensä kerätty useista eri lähteistä ja joka liian monimutkainen käsiteltäväksi perinteisillä tietokanta- ja analyysityökaluilla
Dataframe	Taulukkomainen rakenne, joka sisältää dataa
Matriisi	Riveihin ja sarakkeisiin jaettu taulukko
Vektori	Yksiulotteinen matriisi
Parametri	Arvo, joka välitetään ohjelmalle tai algoritmille ja jota käytetään ohjelman suorituksen määrittämiseen tai mukauttamiseen
Rajapinta	Mahdollistaa tietojen siirtämisen ohjelmistojen välillä
Saas	SaaS eli Software as a Service on ohjelmistojen tarjoamisen ja käytön malli, jossa ohjelmistot tarjotaan verkon yli pilvipalveluna
Parquet	Tiedostomuoto, joka on suunniteltu tallentamaan suuria määriä tietoja erittäin tehokkaasti
Korrelaatio	Osoittaa, kuinka vahvasti kahden muuttujan arvot ovat yhteydessä toisiinsa
Skaalaus	Muuttujien arvojen muuttaminen vertailukelpoiseksi
One-hot	Kategorisen muuttujan muuntaminen numeeriseen muotoon. Jokaiselle kategorialle luodaan oma sarake

1 JOHDANTO

Tiedon määrän sekä tietokoneiden laskentatehon kasvu on mahdollistanut tekoälyn ja varsinkin koneoppimisen kehittymisen viime vuosina (Niemi 2021,135). Samaan aikaan tiedon tallentaminen ja prosessointi eri menetelmillä on edistynyt varsinkin pilviympäristöissä. Yhdessä nämä ovat kasvattaneet dataan ja koneoppiviin algoritmeihin perustuvien ratkaisuiden yleisyyttä. Popkhadze (2021) mukaan tekoälyn kehittyminen ja käyttö on kasvanut nopeasti, mutta sen hyödyntäminen korkeakoulusektorilla ei ole vielä toistaiseksi yleistä (Popkhadze 2021, 254).

Alpaydin mukaan koneoppimisessa on tarkoituksena toteuttaa tietokoneohjelma, joka oppii. Oppimisen tuloksena kokemukseen pohjautuva suoritus paranee. (Alpaydin 2021, 40.) Oppimisen kannalta keskeinen resurssi on data. Korkeakouluissa dataa muodostuu erilaisissa järjestelmissä, joita käytetään operatiivisen toiminnan toteuttamisessa ja ohjaamisessa. Nämä järjestelmät on tyypillisesti suunniteltu suorittamaan ennalta määriteltyjä toimintoja, joihin liittyvät tiedot tallennetaan tietokantoihin. Siirtämällä tiedot datan jalostuksen ja tekoälyn hyödyntämisen mahdollistavaan ympäristöön niillä voidaan opettaa koneoppivia algoritmeja sekä ennustaa tulevaa.

Koneoppimista voidaan hyödyntää ratkaisemaan nopeasti ja tarkasti ihmiselle hitaita sekä tarkkuutta vaativia manuaalisia työtehtäviä. Se toimii konemaisesti, eikä sen keskittyminen katoa ihmisen tavoin yksitoikkoisissa tehtävissä. Koneoppivien mallien tuottamia tuloksia voidaan käyttää päätöksenteon tukena. Niiden avulla datasta voidaan löytää säännönmukaisuuksia, jotka muuten jäisivät huomaamatta.

Opinnäytetyö toteutettiin suomalaiselle korkeakoululle. Korkeakoulussa on tunnistettu tekoälyn potentiaali erilaisten ilmiöiden tunnistamisessa ja ongelmien ratkaisussa. Tämän lisäksi korkeakoulussa on suunniteltu pilviympäristöjen nykyistä laajempaa hyödyntämistä tiedon varastoinnissa ja koneoppimisen soveltamisessa. Tämän opinnäytetyön tutkimuksen aiheeksi valikoitui opiskelijoiden henkilökohtaisiin opintosuunnitelmiin valitsemien ajoitettujen opintojen suorittamisen

ennakointi. Ennakoinnilla pystytään paremmin arvioimaan tarvittavien opintojaksojen toteutuksien määrää ja tunnistamaan opintojaksoja, joiden suorittamisessa ilmenee ongelmia. Ennakoinnissa ei varsinaisesti olla kiinnostuneita yksittäisen opiskelijan suorituksesta vaan yhteenlasketuista ennusteista opintojaksokohtaisesti.

2 TAVOITE, RAJAUKSET JA TUTKIMUSMENETELMÄ

2.1 Tavoite ja tutkimuskysymykset

Opinnäytetyön laajempana tavoitteena on tutkia mahdollisuutta soveltaa koneoppimista koulutuksen toiminnanohjausjärjestelmään muodostuneisiin henkilökohtaisiin opintosuunnitelmiin ja niiden suorittamiseen liittyvään dataan. Tämän lisäksi selvitetään, pystytäänkö koneoppimiseen liittyvistä prosesseista rakentamaan automatisoitu ketju pilviympäristössä. Konkreettisen kehitystyön tavoitteena on toteuttaa koneoppivaan algoritmiin pohjautuva sovellus, josta seurataan ennustetta ajoitettujen opintojaksojen suoritusmääristä. Ennusteella luodaan läpinäkyvyyttä opiskelijoiden opintojen ajoituksiin ja niiden todelliseen suorittamiseen. Keskeisiä tutkimuskysymyksiä ovat:

- Soveltuuko lähdejärjestelmästä siirretty data koneoppimiseen?
- Voidaanko datasta löytää koneoppivalle algoritmille vaadittavia sidonnaisuuksia ja onko dataa riittävästi?
- Minkälaisilla koneoppivilla algoritmeilla voidaan tuottaa käyttötarkoitukseen soveltuva ennuste?
- Miten tekninen toimintamalli pilviympäristöön rakennetaan?

2.2 Rajaukset

Tekoälyn käyttöalue on suuri ja sen soveltaminen sisältää monenlaisia toimintamalleja. Tässä opinnäytetyössä keskitytään tekoäly termin alle kuuluvaan ohjattuun koneoppimiseen ja sen soveltamiseen erityisesti ennustemallien tuottamisessa. Opinnäytetyön tekninen toteuttaminen sekä siihen liittyvät ohjelmat ja järjestelmät ovat laajoja kokonaisuuksia. Sen vuoksi tässä opinnäytetyössä käsitellään käytännön toteutuksen kannalta olennaisimmat teknologiat, mutta niiden asennus, ylläpitäminen, hallinnointi ja kustannukset rajataan ulos tutkimuksesta. Yhtä lailla tekniseen toteuttamiseen liittyvästä ohjelmoinnista kuvataan koneoppimiseen liittyvä prosessi ja keskeisimmät toiminnot purkamatta auki jokaista

koodiriviä. Toteutuksessa hyödynnettiin pilvilaskentaa, jonka tehoa voidaan säädellä tarpeen mukaan. Sen vuoksi esimerkiksi algoritmien suoritusaikoja tai muita laskentatehoon liittyviä asioita ei analysoida. Tutkimuksessa käytettävä tekninen ympäristö ja työkalut rajautuvat korkeakoulussa käytettävään Microsoft Azure pilviympäristöön ja sinne asennettuihin maksullisiin työkaluihin. Teknisessä toteuttamisessa tutkitaan samalla niiden käyttöön liittyviä toimintatapoja ja parhaita käytänteitä.

2.3 Konstruktiivinen tutkimus

Konstruktiivinen tutkimus tuottaa ratkaisun organisaation käytännön ongelmiin. Tutkijan on opiskeltava tarvittava teoreettinen osaaminen ja sen jälkeen kehittävä organisaatiolle ratkaisu tai malli, jonka se voi ottaa käyttöön. (Virtanen 2006, 46-47) Tämän opinnäytetyön tavoitteena oli etsiä ratkaisuvaihtoehtoja ajoitettujen opintojen suorittamisten ennustamiseen sekä rakentaa käytännön ratkaisu ennusteen toteuttamiseen. Sen vuoksi konstruktiivinen menetelmä soveltui tutkimiseen.

2.4 Konstruktiivisen tutkimukset vaiheet ja raportointi

Kasanen, Lukka ja Siitosen (1991) mukaan konstruktiivinen tutkimus etenee seuraavien vaiheiden kautta:

1. Ensimmäiseksi etsitään mielenkiintoinen ongelma.
2. Toisessa vaiheessa rakennetaan riittävä käsitys tutkittavasta asiasta. Käsityksen muodostamisessa auttavat kirjallisuuteen perehtyminen sekä opinnoissa hankittu tieto.
3. Kolmantena luodaan ratkaisumalli, joka on lopputuloksen kannalta olennaista. Se saavutetaan innovatiivisella työskentelyllä sekä aktiivisella yhteydenpidolla toimeksiantajan kanssa.
4. Neljännessä vaiheessa varmistetaan testaamiseen avulla konstruktion oikeellisuus.
5. Esitetään ratkaisun tieteellinen uutuusarvo, jossa tuodaan esille sovellutetut teoriat ja niiden osuus.

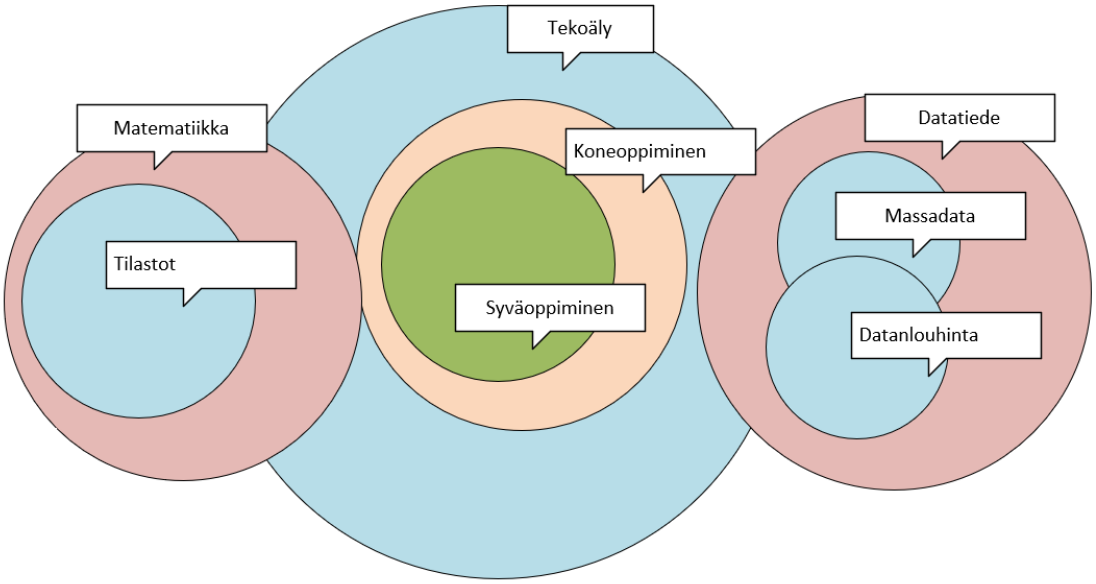
6. Lopuksi pohditaan yleistettävyyttä ja onko tuotettua ratkaisua mahdollista soveltaa muissa organisaatioissa. (Virtanen 2006, 50.)

Tutkimuksen raportoinnissa keskitytään kehitetyn ratkaisun esittelyyn, teoriakytkentöjen esittämiseen, uutuusarvon esittelyyn sekä yleistettävyyden toteamiseen (Virtanen 2006, 48). Varsinaisen opinnäytetyön raportoinnin tietoperusta esitetään osittain yhdessä kehitystyön kanssa. Näin ollen kehityksessä käytetyt teoriakytkennät tulevat selvemmin esille ja ne on helpompi ymmärtää.

3 TEKOÄLY JA KONEOPPIMINEN

3.1 Tekoäly

Tekoäly on tietojenkäsittelytieteenala, jonka tavoitteena on saada koneet tekemään älykkäitä asioita, esimerkiksi oppimaan ja ratkaisemaan ongelmia ihmisten ja eläinten tavoin. Toimiakseen tekoäly tarvitsee syötteitä ympäristöstään, joista se tekee laskelmia ja niihin perustuvia päätöksiä itsenäisesti. Tekoäly on laaja käsite ja se sisältää useita osa-alueita, jotka ovat liitoksissa toisiinsa. Matematiikka ja datatiede ovat olennainen osa tekoälyä, mutta silti erillisiä osakokonaisuuksia. Datatiede keskittyy dataan ja siihen kuuluvat massadata (eng. big data) sekä datan louhinta. Koneoppimista ja syväoppimista voidaan käyttää datatieteessä tiedon prosessoinnissa. (Xiao 2022, luku 1.) Eri osa-alueiden liitännäisyys tulee selkeästi esille tekoälyn käyttöönoton prosessissa. Käyttöönottoa käsitellään tässä opinnäytetyössä luvuissa 5, 6, 7 ja 8. Tekoälyn, koneoppimisen, syväoppimisen, datatieteen sekä matematiikan sidonnaisuus esitetään kuviossa 1.



KUVIO 1: Tekoälyn, koneoppimisen, syväoppimisen, datatieteen ja matematiikan sidonnaisuus (Xiao 2022, luku 1, muokattu).

3.2 Koneoppiminen

Koneoppimista voidaan kutsua tekoälyn alalajiksi. Se on systeemi, joka parantaa sille määritellyn tehtävän suorittamista perustuen kokemukseensa tai kasvavaan data määrään. (Elements of AI n.d.) Koneoppimisen ymmärtämisen kannalta tietoaineisto-, algoritmi- ja funktiotermien ymmärtäminen on tärkeää. Tietoaineisto voi yksinkertaisimmillaan olla taulukko, jonka vaakarivit sisältävät esimerkit ja pystyrivit sen piirteiden informaatiota. Prosessi tai ohjelma, jota tietokone noudattaa on algoritmi. Koneoppimisessa se määrittää tietoaineistoa tutkivan ja aineistoissa toistuvasti esiintyvät säännönmukaisuudet tunnistavan prosessin. Funktio on determistinen selitys joukolta syötteitä yhteen tai useampaan lopputulokseen. Determistisyys tarkoittaa, että funktion syöte arvojen joukko tuottaa aina saman lopputuloksen, esimerkiksi 2+2=4. (Kelleher 2020, 16.)

3.3 Koneoppivat algoritmit

Koneoppivat algoritmit löytävät säännönmukaisuuksia ja oppivat tekemään ennusteita dataa prosessoimalla. Niiden suoritus paranee ajan myötä uuden datan avulla, koska ne oppivat esimerkeistä. Tyypillisimmät käyttötapaukset ovat ennusteet (mitä tapahtuu), määrittelyt (mitä tulisi tehdä tavoitteiden saavuttamiseksi) ja kuvaukset (mitä tapahtui). (Chojecki 2020.) Koneoppiminen soveltuu ympäristöihin, joissa tapahtuu paljon muutoksia ja uutta dataa muodostuu koko ajan paljon. Sitä voidaan käyttää ratkaisemaan monimutkaisia ongelmia, joissa perinteiset ratkaisut eivät toimi tai ne vaativat suuren määrän hienosäätöä toimiakseen. (Géron 2022, luku 1.) Koneoppivat algoritmit on suunniteltu löytämään yhtäläisyyksiä ja samankaltaisia rakenteita niille syötetystä opetusdatasta. Ne perustavat oppimisisensa opetusdataan, jonka ne valjastavat koneoppivassa mallissa ennusteeksi uuden datan saapuessa. Sen vuoksi hyvä datan laatu on tärkeä osa koneoppimista. Hyvälaatuisesta datasta malli löytää oikeita korrelaatiota ja vivahteita, joista rakentuu osuvia ennusteita. Luonnollisesti huonolaatuinen ja virheitä sisältävä data tekee koneoppimisesta hyödyttömän. (Brink ym. 2017, 27.) Visualisoinnit ovat keskeinen osa koneoppimista ja siihen liittyvää päätöksentekoa. Lopputulosten esittämisen ohella niiden avulla pystytään kasvattamaan ymmärrystä datan sisällöstä, rakenteista sekä riippuvuuksista. (Gollapudi 2016, 78.)

3.4 Piirteet

Hyödynnyttäessä koneoppivia algoritmeja tarjolla on tyypillisesti lukuisia piirteitä, joilla voisi ennustaa lopputulosta. Rajoittavina tekijöinä piirteillä on kaksi tekijää. Piirteen arvo on oltava tiedossa ennustetta tehtäessä ja sen oltava numeerinen tai kategorinen tyypiltään. Datasta voidaan myös muodostaa piirteitä tilastollisilla menetelmillä esimerkiksi laskemalla jakaumia tai kokonaismääriä. Olennaista kuitenkin on, että koneoppivalle mallille syötetään vain piirteitä, jotka korreloivat kohdemuuttujan kanssa. Niiden valinta on haastava prosessi, koska liian suuri määrä sekoittaa algoritmin toiminnan eikä se pysty löytämään oikeanlaisia korrelaatioita niiden välillä. Toisaalta valitsemalla liian vähän piirteitä, jokin ennusteen kannalta olennainen asia jää ulos. Käytännöllinen tapa piirteiden valintaan on ensiksi kokeilla kaikkia piirteitä, joiden uskotaan olevan olennaisia ennusteen kannalta. Jos tulos on haluttu, lopetetaan. Muussa tapauksessa lisätään kokeiluun piirteitä, jotka saattaisivat vaikuttaa tulokseen. Tuloksen ollessa riittävä, lopetetaan. Muutoin ajetaan piirteiden valinta-algoritmi laajennettua piirrejoukkoa vasten, joka luokittelee ennustuksen kannalta tärkeimmät piirteet. (Brink ym. 2017, luku 2.) Kirjoittajien esille tuomat näkökulmat ovat olennaisia, mutta ne liittyvät osittain vain ohjattuun oppimiseen. Ohjaamattomassa oppimisessa ei määritellä kohdemuuttujaa ja näin ollen ei voida tietää piirteitä, jotka korreloivat kohdemuuttujan kanssa.

3.5 Algoritmien ryhmittely

Koneoppivat algoritmit jaetaan kolmeen ryhmään: ohjattuun oppimiseen, ohjaamattomaan oppimiseen ja vahvisteoppimiseen. Ohjatussa oppimisessa käytetään opetusaineistoa, jossa on määritelty syötteet ja niistä opittava lopputulos. Ohjaamattomassa oppimisessa koneelle annetaan data, mutta sille ei määritellä ennustettavaa lopputulosta. Vahvisteoppimisessa kone pyrkii parhaaseen mahdolliseen lopputulokseen ihmisen antamien vahvisteiden pohjalta. (Chojecki 2020.)

3.6 Ohjattu oppiminen ja sen ryhmät

Ohjatun oppimisen algoritmit jaetaan kahteen ryhmään: regressio ja klassifikaatio. Regressiossa ennustetaan lukua, esimerkiksi asunnon hintaa tai henkilön ikää. Klassifikaatioalgoritmit puolestaan ennustavat kategoriaa tai ryhmää, jotka ne oppivat opetusaineistoon määritellyistä luokista. Klassifikaatio voi olla binäärinen eli lopputuloksena on kaksi luokkaa, esimerkiksi onko luottokortin tapahtuma todellinen vai huijaus. Moniluokkaisessa klassifikaatiossa lopputuloksena on useita luokkia, jotka kuvaavat esimerkiksi mitä numeroa käsin kirjoitettu numero esittää. (Prosise 2022, luku 3.)

3.7 Ennustemallit

Regressio- ja klassifikaatioalgoritmeilla voidaan toteuttaa ennustemalleja. Kanasen ja Puolitaipaleen (2019) mukaan ennustemalleihin pohjautuvia koneoppimisenohjelmia voidaan käyttää avustamaan tai jopa korvaamaan asiantuntijoiden tekemiä päätöksiä. Ennustemalleja voidaan hyödyntää tarkasti rajatun ongelman tai tehtävän ratkaisussa. Niillä pystytään ennustamaan suurista datamääristä nopeasti asioita, jotka olisivat muutoin työläitä toteuttaa. Tämän lisäksi ne ovat käyttökustannuksiltaan edullisia. Ennustemallit pystyvät tuottamaan tarkkojakin tuloksia, mutta mallin käyttämistä itsenäisessä päätöksenteossa kuitenkin tulee harkita. Ne toimivat tilastollisten jakaumien perusteella ja ne ovat puolueettomia, eikä niillä ole ennakkoasenteita. (Kananen & Puolitaival 2019, 111.)

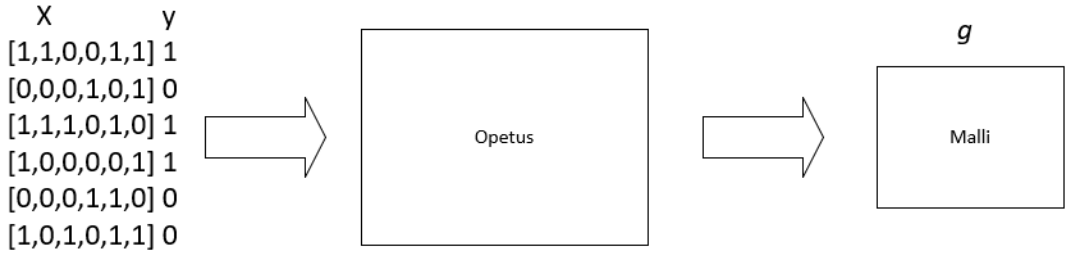
3.8 Ohjattuun oppimiseen perustuva matemaattinen malli

Grigorievin (2021) mukaan muodollisesti ohjattuun oppimiseen perustuva koneoppiva malli voidaan esittää matemaattisesti: $y \approx g(X)$

- *g* on funktio, joka koneoppimisella halutaan oppia.
- *X* on piirre matriisi, jonka rivi ovat piirre vektoreita.
- *y* kohdemuuttuja vektori.

Koneoppimisen tavoitteena on oppia funktio *g* siten, että kun malli saa *X* matriisiin se pystyy muodostamaan kohdemuuttujan *y*. Tätä prosessia kutsutaan mallin

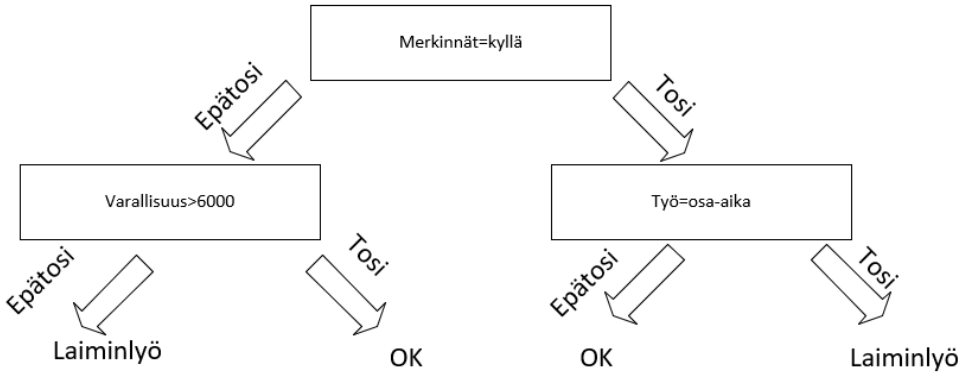
opettamiseksi. Käytännössä mallin opettamisessa algoritmille syötetään matriisiin *X* piirrevektorit ja lopputulos vektorissa *y*, jota mallin halutaan ennustavan. (Grigoriev 2021, luku 1.) Koneoppivan mallin opetus kuviossa 3.



KUVIO 2: Koneoppivan mallin opettaminen ja sen toiminta käytännössä (Grigoriev 2021, luku 1, muokattu).

3.9 Päätöspuu

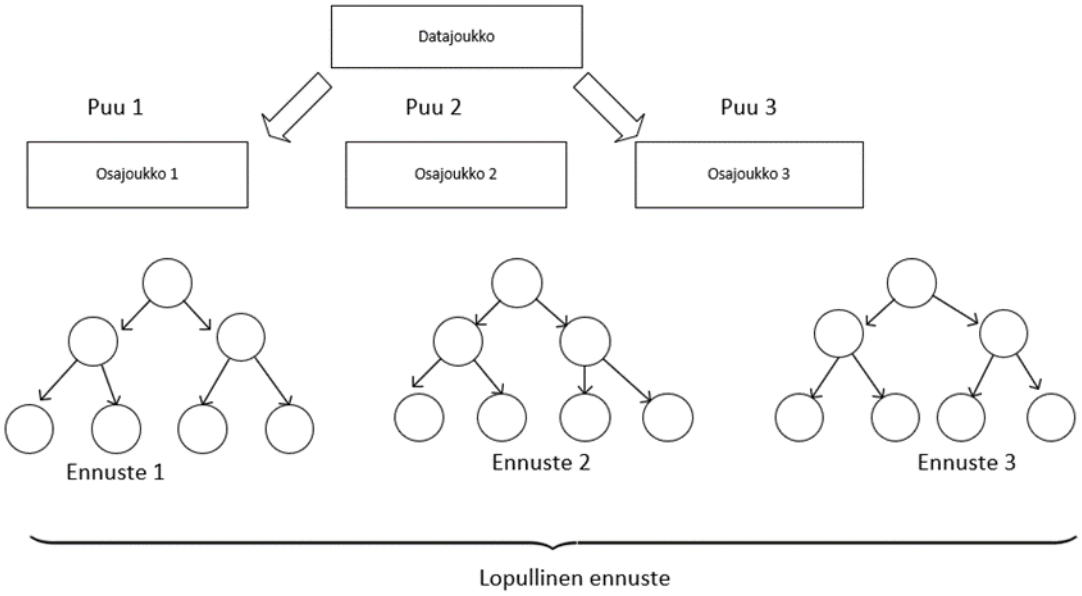
Päätöspuu (eng. decision tree) on yksinkertainen algoritmi, joka soveltuu numeerisen luvun tai luokan ennustamiseen. Sen rakenne on puumainen. (Tshepo 2021, luku 6.) Päätöspuun toiminta perustuu sarjaan jos-sitten-muutoin -sääntöjä. Puun jokainen solmu sisältää ehdon. Ehdon täyttyessä edetään oikealle, muutoin vasemmalle. Lopulta muodostuu lopputulos. Yksinkertaistettu päätöspuun toiminta lainahakemuksen käsittelyssä on esitetty kuviossa 3. (Grigoriev 2021, luku 6.)



KUVIO 3: Päätöspuun toiminta (Grigoriev 2021, luku 6, muokattu).

3.10 Päätösmetsä

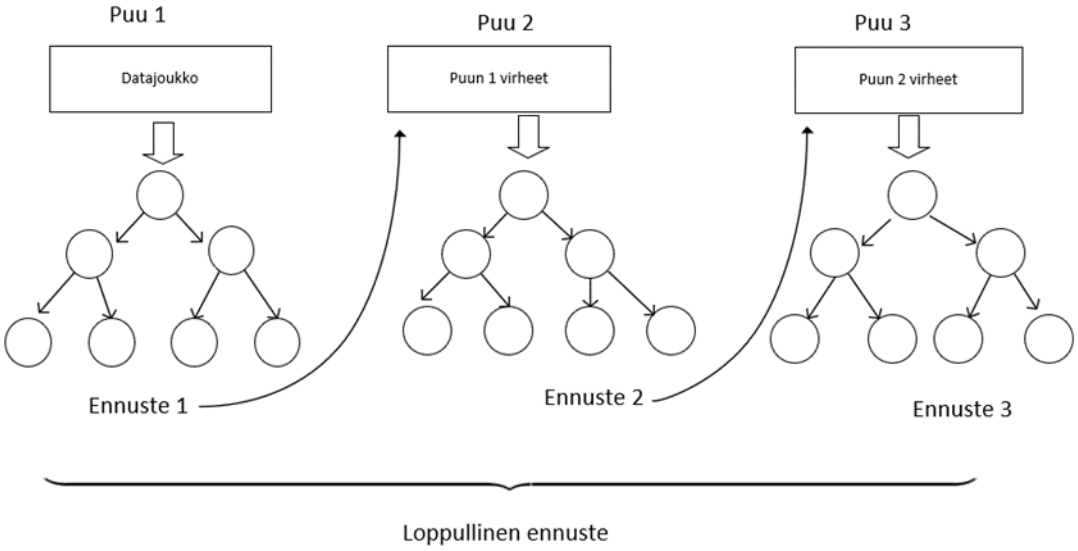
Päätöspuut ovat itsessään tehokkaita malleja, mutta niiden ennustevoima kasvaa yhdistelemällä yksittäisiä puita ryhmiksi. Päätösmetsämalissa (eng. random forest) joukko päätöspuita opetetaan siten, että jokainen yksittäinen puu opetetaan eri osajoukolla piirteitä. Lopputulos saadaan yhdistämällä yksittäisten puiden tulos yhdeksi lopputulokseksi. (Grigoriev 2021, luku 6.) Päätösmetsän toiminta esitetään kuviossa 4.



KUVIO 4: Päätösmetsän toiminta (Grigoriev 2021, luku 6, muokattu).

3.11 Gradient-Boosted Tree

Gradient-Boosted Tree (GBT) pohjautuu päätöspuurakenteeseen, mutta se yhdistää useiden puiden lopputuloksia. Päätösmetsässä puita koulutetaan yksittäin, mutta GBT mallissa puita opetetaan sidoksissa edeltävään. Algoritmi opettaa joukon päätöspuita peräkkäin siten, että seuraava puu pienentää edellisen puun virhettä. (Quinto 2020, luku 3.) Käytännössä GBT-mallissa opetetaan ensimmäinen puu ja selvitetään sen tekemät virheet. Seuraavaksi koulutettava puu korjaa edellisen virheet. Selvitetään virheet uudelleen ja koulutetaan seuraava puu korjaamaan taas virheet ja niin edelleen. (Grigoriev 2021, luku 6.) GBT-mallin toiminta esitetään kuviossa 5.



KUVIO 5. GBT mallin toiminta (Grigoriev 2021, luku 6, muokattu).

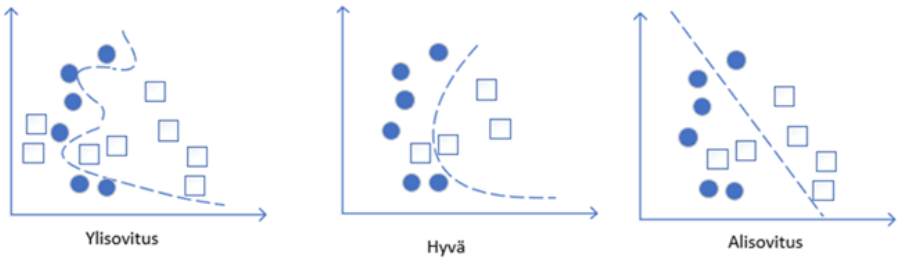
4 DATAN JAKAMINEN OSIIN JA KONEOPPIMISEN MITTARIT

4.1 Koulutus-, testaus- ja validointiryhmät

Ohjatussa koneoppimisessa data jaetaan kolmeen ryhmään: koulutus, testaus ja validointi. Malli opetetaan koulutusdatalla ja testataan testidatalla eli katsotaan kuinka tarkkoja ennustuksia opetetulla mallilla voi saada. Validointidatalla varmennetaan mallin toimivuus. Testaus- ja validointidatalla pystytään selvittämään algoritmin toiminta uudella datalla, jota se ei ole aiemmin nähnyt (Kananen & Puolitaival 2019, 46–47).

4.2 Mallin suoriutuminen

Mallin toimimista hyvin harjoitusdatalla, mutta huonoa kykyä yleistää aiemmin näkemättömään dataan kutsutaan ylisovitukseksi (eng. overfitting). Se on yleinen ongelma koneoppimisessa. Se saattaa johtua esimerkiksi liian monista parametreistä, jotka aiheuttavat liian monimutkaisen mallin. Alisovitus (eng. underfitting) tarkoittaa taas sitä, että malli ei ole tarpeeksi monimutkainen. Se ei opi tarvittavia sidonnaisuuksia harjoitusdatasta, mikä ilmenee huonona suorittamisena näkemättömään dataan. (Raschka, Liu, Mirjalili, Dzhulgakov 2022.) Kuviossa 6 havainnollistetaan yli- ja alisovitus sekä hyvän mallin toiminta. Olennaista on huomata, että tarkoituksena ei ole löytää kaikkia datapisteitä oikein jakavaa mallia, vaan hyvä yleistys. Hyvässä yleistyksessä on tyypillisesti mukana pieni määrä vääriä ennusteita.



KUVIO 6: Mallin suorittaminen (Raschka, Liu, Mirjalili, Dzhulgakov 2022, muokattu).

4.3 Ennusteen mittarit

Quinton (2020) mukaan luokitteluun perustuvassa koneoppimisessa mallin ennustetehoa voidaan arvioida vertaamalla tunnettua luokkaa ennustettuun. Tulokset jakautuvat neljään kategoriaan: 1) tosi positiivinen (TP), jossa sekä ennustettu muuttuja että kohdemuuttuja ovat positiivisia, 2) tosi negatiivinen (TN), jossa molemmat ovat negatiivisia, 3) väärä positiivinen (FP), jossa ennustettu muuttuja on positiivinen, mutta kohdemuuttuja negatiivinen sekä 4) väärä negatiivinen (FN), jossa ennustettu muuttuja on negatiivinen ja kohdemuuttuja positiivinen. Nämä muodostavat pohjan luokittelun algoritmien arviointiin. Mittareita arviotiin ovat esimerkiksi accurancy, presicion ja recall (Quinto 2020, luku 1). Tyypillisesti niiden tulokset esitetään taulukossa, jota kutsutaan konfuusiomatriisiksi.

4.3.1 Accuracy

Accuracy lasketaan jakamalla oikeiden ennusteiden määrä ennusteiden kokonaismäärällä. Se ei ole hyvä mittari, jos luokat eivät ole jakautuneet tasaisesti, esimerkiksi jos esiintyy 90% positiivisia ja 10% negatiivisia tapauksia. Ennustamalla 100 tapauksen datajoukosta kaikki positiivisiksi saadaan 90% accuracy (Quinto 2020, luku 1). Accuracy-kaava kuvassa 1.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

KUVA 1: Accuracy kaava.

4.3.2 Precision

Precision saadaan jakamalla oikeiden positiivisten määrä yhteenlaskettujen oikeiden positiivisten ja väärien positiivisten määrällä. Käytännössä se osoittaa kuinka malli onnistuu ennustamaan positiivisen tapauksen. Se on hyvä mittari tilanteissa, joissa väärien positiivisten ennusteiden hinta on suuri. (Quinto 2020, luku 1.) Precision-kaava kuvassa 2.

$$Precision = \frac{TP}{TP+FP}$$

KUVA 2: Precision kaava.

4.3.3 Recall

Recall lasketaan jakamalla oikeat positiiviset yhteenlasketuilla oikeiden positiivisten ja väärien negatiivisten määrällä. Recall on hyvä mittari, jos väärien negatiivisten ennusteiden hinta on korkea. (Quinto 2020, luku 1.) Recall-kaava kuvassa 3.

$$Recall = \frac{TP}{TP+FN}$$

KUVA 3: Recall-kaava.

4.3.4 Roc-kuvaaja

Binäärisessä luokittelussa voidaan käyttää ROC-kuvaajaa. Se kuvaa mallin ennustetarkkuutta. Sen avulla ymmärretään, kuinka paljon vääriä positiivisia aiheutuu, kun tietty määrä ennustetaan oikein. ROC-kuvaajassa alle jäävästä alueesta käytetään AUC-termiä eli Area Under Curve. Sen perusteella huomataan, kuinka hyvä luokittelija on kyseessä. Esimerkiksi 0,5 AUC-arvo vastaa kolikon heiton ennustetehoa. (Kananen & Puolitaival 2019, 179-181.)

5 PILVILASKENTA JA OHJELMOINTI

5.1 Pilvilaskenta

Teknologian ja digitalisaation kehitys on ollut viime vuosina nopeaa. Laskentateho on kasvanut ja sen hyödyntäminen on edullisempaa. Laitteistoresursseja on virtualisoitu eli fyysisten palvelimien sijasta käytetään virtuaalisia palvelimia. Virtualisoinnin avulla laitteet ja niihin liittyvät ohjelmistot suoritetaan suurissa laskentakeskuksissa tehokkaasti. Tätä kutsutaan pilvilaskennaksi. Tekoälymenetelmät ovat kehittyneet nopeasti kasvavan laskentakapasiteetin, algoritmien ja datan hyödyntämisen vuoksi. (Tarkoma 2021, 95.)

Andersonin (2023) mukaan pilvilaskenta on yksi merkittävämmistä teknologisista keksinnöistä tällä aikaudella. Sitä kutsutaan pilvilaskennaksi sen vuoksi, että virtualisoituja laitteistoresursseja käytetään internetin kautta. Pilvilaskennan termin alle kuuluu laaja joukko palveluita ja resursseja. Esimerkiksi palvelimet, tietokannat, tietosäilöt, verkkoinfrastruktuurit, ohjelmat ja monet muut IT-infrastruktuurit ovat osa pilvilaskentaa. (Anderson 2023, luku 1.)

Pilvilaskenta perustuu resurssien jakamiseen. Linthicum (2023) määrittelee pilvipalvelun tarjoajan vastuulle resurssien tarjonnasta huolehtimisen sekä niiden ylläpidon. Palvelun käyttäjien kustannukset rakentuvat resurssien käytön mukaan. Organisaatioiden näkökulmasta pilvilaskennassa on useita hyötyjä. Se on kustannustehokas tapa, koska pilvessä sijaitsevien resurssien hyödyntäminen vapauttaa organisaatiolta tarpeen investoida omiin laitteistoihin ja niiden ylläpitoon liittyviin toimintoihin sekä henkilökuntaan. Pilvilaskenta on tehokasta ja skaalautuvaa, eli resursseja voi tarpeen mukaan lisätä tai vähentää. Tämän lisäksi organisaation on helpompi pysyä kehityksen mukana, koska teknologinen kehitys tapahtuu nykyisin pilvessä. Tehokkaimmat, edullisimmat ja tietoturvallisimmat ratkaisut löytyvät usein julkisista pilvipalveluista kuten, AWS, Google ja Microsoft. (Linthicum 2023, luku 1.) Käytännössä suuria datamääriä sisältäviä monivaiheisia prosesseja voi olla vaikea suorittaa tehokkaasti ja automatisoidusti ilman pilvilaskentaa. Sen vuoksi nykypäivänä koneoppimisessa usein hyödynnetään pilvilaskentaa.

5.2 Ohjelmointi

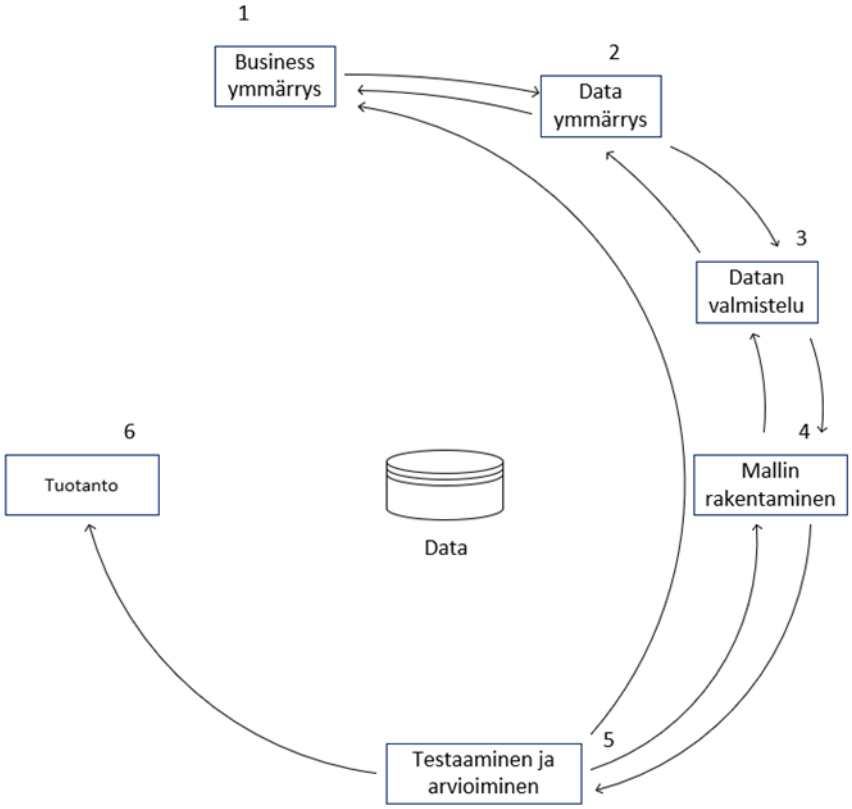
Tekoälyn hyödyntämiseen löytyy useita valmiita ja ilmaisia kehitystyökaluja. Sen kehittäminen ei ole sidottu yhteen ohjelmointikieleen. Yleisimmin käytetty kieli on Python. Se on kehitetty avoimen lähdekoodin pohjalta. Sitä voidaan käyttää vapaasti internetistä ja siihen löytyy runsaasti ohjelmistokirjastoja. Ohjelmistokirjastot ovat valmiita tietokoneohjelmia, jotka ratkaisevat tietyn tehtävän. Microsoftin, Googlen ja Amazonin kaltaiset yritykset kehittävät ja ylläpitävät kehitysympäristöjä tekoälyn hyödyntämiseen. Niitä on mahdollista hyödyntää osittain ilmaiseksi, mutta tuotantokäytössä kustannukset tyypillisesti rakentuvat tapahtumamääriin ja käyttötunteihin. Hyötynä kaupallisissa kehitysympäristöissä on niiden toimintavarmuus sekä skaalautuvuus tarpeiden mukaan. (Kananen & Puolitaival 2019, 188-189.) Kaupallisten ympäristöjen käytössä on tärkeää varmistaa, että ne otetaan käyttöön tietoturvallisesti ja tietosuojanäkökulmat huomioiden. Esimerkiksi selvitetään käytetäänkö tiedon prosessoinnissa tai tallentamisessa Euroopan Unionin ulkopuolella sijaitsevia laskentakeskuksia.

6 KONEOPPIMISEN PROSESSI JA TYÖTEHTÄVÄT

6.1 Koneoppimisen prosessi

Koneoppimiseen liittyvää prosessia voidaan selkeyttää ja organisoida toimintamalleilla. Yleisesti käytetty toimintamalli on Cross-Industry Standard Process for Data Mining (CRISP-DM). (Grigoriev 2021, luku 1.) Delen (2020) määritelmän mukaan sitä käytetään usein talousanalytiikkaan ja datatieteeseen liittyvissä projekteissa. Prosessi alkaa ratkaistavaan ongelmaan liittyvän tietämyksen keräämisellä ja tarpeen ymmärtämisen hahmottamisella. Toisessa vaiheessa selvitetään tarvittavat datalähteet ja niiden soveltuvuus projektiin. Kolmannessa vaiheessa data muokataan käyttötarkoitukseen sopivaan muotoon. Seuraavaksi dataa mallinnetaan erilaisilla algoritmeilla ja tutkitaan niiden potentiaalia ongelman ratkaisussa. Vaiheessa viisi testataan ja arvioidaan yhden tai useamman mallin luotettavuutta ja käyttökelpoisuutta. Lopputuloksena on käyttöönotettu ratkaisu, joka selvittää alkuperäisen ongelman ja palvelee alkuperäistä tarvetta. Vaikka CRISP-DM -prosessissa on numerojärjestyksessä etenevät askeleet, käytännössä niissä palataan usein aikaisempiin vaiheisiin. (Delen 2020, luku 3)

Kirjoittajan esittelemään malliin voisi lisätä nuolen vielä kohdasta 6 takaisin kohtaan 1, jolloin prosessista saadaan kokonainen kehä. Kehämäisellä ajattelulla mallin tuotanto käytöstä saatuja kokemuksia voi ajan kuluessa peilata takaisin alkuperäiseen tarpeeseen. Alkuperäinen tarve on saattanut muuttua esimerkiksi ympäristössä tapahtuneiden muutoksien vuoksi, jolloin tarvitaan kokonaisprosessin uutta arviointia. CRISP-DM -prosessin kuusi vaihetta on kuvattu kuviossa 7.



KUVIO 7: CRISP-DM -rakenne ja prosessi (Delen 2020, luku 3, muokattu).

6.2 Koneoppimiseen liittyvät tehtävänkuvat

Koneoppimisen mahdollistava prosessi sisältää useita vaiheita ja niiden suorittamiseen vaadittava osaamisen kirjo on laaja. Lakshmanan, Robinson & Munn (2020) määritelmän mukaan koneoppimiseen mahdollistavaan prosessiin organisaatiossa osallistuu tyypillisesti useita työntekijöitä. Yleisiä tehtäväkuvia koneoppimiseen liittyen ovat:

Datatieteilijä (eng. data scientist) keskittyy datan keräämiseen, analysointiin, prosessointiin ja koneoppivan mallin rakentamiseen.

Datainsinööri ylläpitää ympäristöjä ja hallinnoi tiedon siirtämiseen, tallentamiseen, valmisteluun ja prosessointiin liittyviä toimintoja.

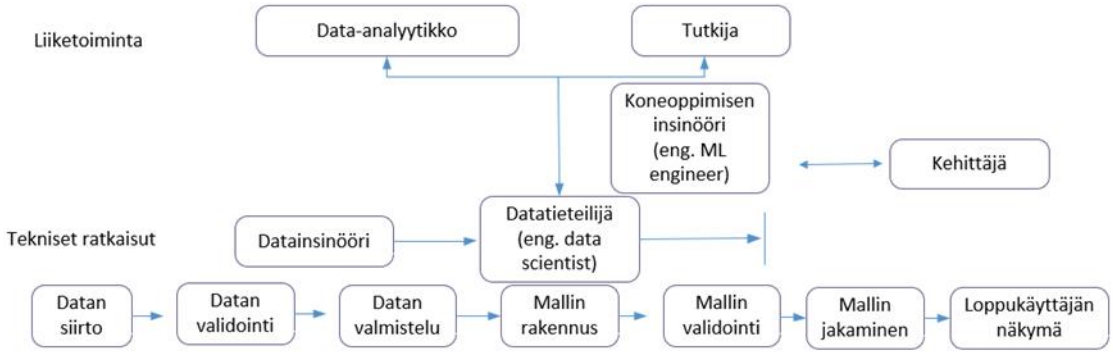
Koneoppimisen insinöörin (eng. ML engineer) tehtävät ovat samankaltaisia, kuin datainsinöörin, mutta keskittyvät koneoppivien mallien ylläpitoon, päivittämiseen, versionhallintaan sekä jakamiseen.

Tutkija etsii ja kehittää uusia algoritmeja. Tutkijan toimenkuva keskittyy tutkimaan uusia lähestymis- ja toimintatapoja koneoppimiseen liittyen, ei niinkään rakentamaan teknisiä ratkaisuja.

Data-analyytikko analysoi dataa ja koostaa tekemistään löydöksistä yhteenvetoja sekä visualisointeja. Data-analyytikko toimii lähellä liiketoimintaa ja pyrkii selvittämään miten dataan liittyvät löydökset voivat auttaa ratkaisemaan tietyn liiketoimintaan liittyvän ongelman.

Kehittäjä rakentaa tuotantosysteemejä ja sovelluksia, joilla loppukäyttäjät pystyvät seuraamaan koneoppivan mallin tuottamia tuloksia.

Pienissä organisaatioissa, joissa koneoppimista toteuttavia työntekijöitä on vähän, saman henkilön vastuulla on tyypillisesti useampia tehtäviä. (Lakshmanan ym. 2020, luku 1.) Kanasen ja Puolitaipaleen mukaan organisaatioissa, joissa tekoälyn tuotanto toteutetaan itse tarvitaan ainakin sovellusalueen asiantuntijaa, datatieteilijää ja datainsinööriä. (Kananen & Puolitaival 2019, 208.) Tekniset henkilöt ja tekoälyn erikoisosaajat ovat luonnollisesti tärkeitä, mutta sovellusalueen asiantuntijan rooli on keskeinen kokonaisuuden kannalta. Ilman toiminta- tai sovellusalueen riittävää käytännön tuntemusta koko koneoppimiseen liittyvä prosessi voi lähteä ohjautumaan väärään suuntaan. Kuviossa 8 on esitetty yleisiä koneoppimiseen liittyviä tehtävänkuvia ja niiden sidonnaisuuksia.



KUVIO 8: Koneoppimiseen liittyviä tehtävänkuvia ja vastuita (Lakshmanan ym. 2020, luku 1, muokattu).

7 KEHITYSTEHTÄVÄ

7.1 Tausta

Tämä kehitystehtävä tuotettiin korkeakoululle, jossa tapahtui vuoden 2022 alussa uudistus koulutusohjelmissa. Uudistuksen vuoksi koulutusohjelmat, opintosuunnitelmien sisällöt, opintojaksot sekä opiskelijoiden perusteet valita opintojaksoja muuttuivat. Opinnäytetyössä keskityttiin vuonna 2022 käyttöönotettuihin koulutusohjelmiin ja niihin sitoutuviin opintojaksoihin. Opinnäytetyöhön liittyvän kehitystyön kannalta olennainen kohderyhmä oli koulutuksen järjestämisen sidosryhmät. Heiltä puuttui ennustemalli koulutusohjelmien opintojaksojen kysyntään ja suorittamiseen liittyen.

Kehitystyöhön liittyvään dataan saatiin käyttö- ja tutkimuslupa korkeakoululta. Koneoppimisessa tietoja käsiteltiin anonymisoituina, siten että yksittäistä opiskelijaa ei pystytty tunnistamaan aineistosta. Kehitystyössä ei oltu kiinnostuneita yksittäisistä opiskelijoista vaan yhteenlasketusta ennusteesta opintojaksokohtaisesti. Datan käsittellyssä noudatettiin korkeakoulun Azure-hallintamallia, jossa on määritelty tiukat standardit datan hallintaan ja tietoturvaan liittyen.

Opinnäytetyön kirjoittajan aiemmissa projekteissa kertynyt tietämys käytetyn datan sisällöstä auttoi ymmärtämään ja käsittelemään tietoa kehitystyössä. Lisäksi kehitystyön eri vaiheissa pidetyissä lyhyissä kokouksissa korkeakoulun Peppi-asiantuntijoiden kanssa tehtiin tarkistuksia ja varmennuksia liittyen dataan. He auttoivat ymmärtämään ja valmistelemaan dataa sekä määrittelemään ennustemallin tarkoitusta ja tavoitteita.

7.2 Peppi-järjestelmä

Opinnäytetyön toteuttamisen kannalta keskeinen tietolähde oli Peppi-koulutuksen toiminnanohjausjärjestelmä. Peppi-konsortio, joka omistaa ja vastaa järjestelmän kehityksestä, määrittelee sen palvelukokonaisuudeksi, joka on tarkoitettu korkeakoulujen opintoihin, opetukseen ja niihin sidoksissa olevien palveluiden hallintaan. Pepin käyttöliittymää käytetään verkkoselaimen välityksellä. Se koos-

tuu käyttäjäryhmien roolien pohjalta rakennetuista työpöydistä: opettaja, opiskelija, suunnittelija, korkeakoulupalvelut ja pääkäyttäjä. (Peppi-konsortio n.d.) Järjestelmä on ollut käytössä kehitystehtävän kohteena olevassa korkeakoulussa elokuusta 2018. Järjestelmän taustalla toimivaan operatiivisen tietokantaan on muodostunut näin ollen käyttäjäryhmien tekemien toimintojen pohjalta dataa useiden vuosien ajalta.

7.3 Datan vaiheet

Opinnäytetyössä käytettävä data muodostui Peppi-järjestelmässä eri käyttäjäryhmien toimintojen tuloksena vaiheittain. Opiskelijan demografiset ja koulutukseen liittyvät tiedot tallentuivat järjestelmään opiskeluiden alussa. Näitä tietoja hallinnoivat ja päivittävät korkeakoulupalvelujen henkilökunta. Henkilökohtaisten opintosuunnitelmien tiedot muodostuivat opiskelijan tekemistä valinnoista. Ensin opiskelija valitsee henkilökohtaiseen opintosuunnitelmaansa opintojaksot sekä niiden ajoituksen ja lähettää opintosuunnitelmansa oppilaanohjaajalle hyväksyttäväksi. Opinto-ohjaaja tarkistaa ja hyväksyy suunnitelman, ellei siinä ilmene korjattavaa. Opiskelija ilmoittautuu valitsemilleen opintojaksoille ja saa niiden suorittamisesta tai suorittamatta jättämisestä myöhemmin merkinnät. Ajallisesti data muodostui hitaasti. Tärkeimpiä yksittäisiä ajankohtia olivat henkilökohtaisten opintosuunnitelmien lukukausikohtaiset viimeiset hyväksymispäivät.

7.4 Tiedon siirtäminen

Peppi-järjestelmä ei varsinaisesti tarjoa työkaluja tietojen koostamiseen tai raportointiin. Sen vuoksi Pepistä on tehty järjestelmän rajapintaa hyödyntämällä tiedonsiirto korkeakoulun ympäristössä sijaitsevan konesalin tietovarastoon. Kahaten mukaan (2004) tietovarastoja käytetään datan tallentamiseen alkuperäisten lähdejärjestelmien ulkopuolelle. Ne mahdollistavat datan uudelleen organisoinnin sekä tehokkaan käytön. (Kahate 2004, luku 9.) Organisaation omassa konesalissa sijaitseva data aiheuttaa rajoitetta, jos dataa halutaan prosessoida pilviympäristössä sijaitsevilla datan muokkauksen ja koneoppimisen mahdollistavilla työkaluilla. Sen vuoksi data siirrettiin tietovarastosta Microsoft Azuren pilviympäristöön. Microsoft määrittelee Azuren pilvessä toimivaksi alustaksi, jossa voi hyödyntää suurta joukkoa palveluja ja työkaluja. Ne mahdollistavat eri ympäristöissä

sijaitsevan datan jalostamisen ja sovellusten rakentamisen. (Microsoft n.d.) Käytännössä Azure pohjautuu pilvilaskentaan. Opinnäytetyön kannalta olennainen data siirrettiin ja tallennettiin Azure DataFactory -tiedonsiirto ohjelmalla tietoaltaaseen, jota kutsutaan nimellä Azure-datalake. Tiedonsiirron prosessi on kuvattu liitteessä 1.

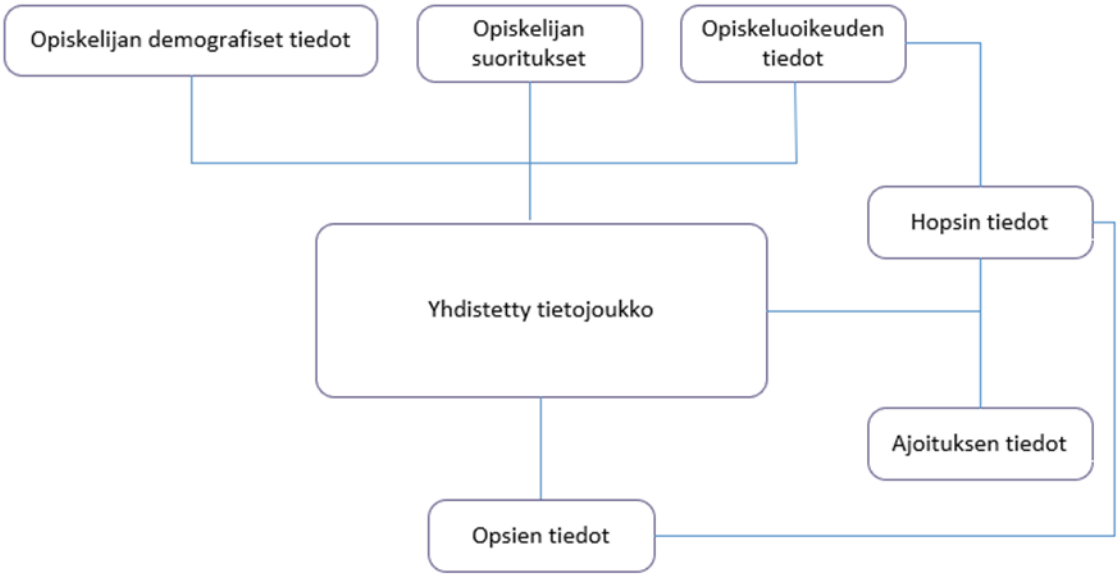
7.5 Tietoallas ja Databricks

Tietoallas on kapasiteetiltaan valtava tietosäilö, jonne on mahdollista tallentaa erimuotoista dataa strukturoituna tai strukturoimattomana. Dataa prosessoimalla niistä voidaan tuottaa analyysejä tai hyödyntää koneoppimista (Tomcy & Pankaj 2017). Azure tietoaltaassa sijaitsevien aineistojen käsittelyyn voidaan käyttää Databricks-työkalua. Databricks mahdollistaa tietojen siirtämisen eri lähdejärjestelmistä, niiden muokkaamisen ja tallentamisen. Tämän lisäksi sillä voidaan analysoida dataa ja soveltaa koneoppimista. Ohjelmointi toteutetaan verkkoselaimen notebookissa ja ohjelmointikielinä toimivat esimerkiksi SQL, Python ja Scala. Palacio (2021) mukaan Databricksin taustajärjestelmänä (eng. backend) toimii Apache Spark, joka luotiin alun perin suurta laskentatehoa vaativien tietomassojen käsittelyyn ja koneoppimiseen. Databricks käyttää prosessoinnissa kapasiteettinä pilvessä sijaitsevia klustereita (eng. cluster). Ne ovat joukko esikonfiguroituja virtuaalikoneita, jotka toimivat yhdessä resursseina toimintojen toteuttamiseen. (Palacio 2021, luku 1.) Databricks-klustereiden käyttöön ei liity ennakkomaksuja, sen sijaan kustannukset pohjautuvat niiden käyttöaikaan (Databricks 2022, n.d).

7.6 Tietojoukon muodostaminen

Tietoaltaaseen tallennetuista tiedoista jalostettiin koneoppimiseen soveltuva tietojoukko Databricks-työkalulla. Sillä pystytään hakemaan ja käsittelemään tietoaltaassa sijaitsevia tiedostoja suoraan, ilman manuaalista latausta. Tietoaltaan raakadatakansioon tallennetut parquet-tiedostot luettiin ensin spark dataframe -muotoon, minkä jälkeen niitä pystyttiin käsittelemään Python-ohjelmointikielellä.

Tutkimuksen kannalta kiinnostavat tiedot olivat jakautuneet eri tiedostoihin. Ensin valmisteltiin kolme osajoukkoa omiksi kokonaisuuksiksi tietoja yhdistämällä. Ensimmäinen osajoukko sisälsi opiskeluoikeuteen, suorituksiin sekä demografiaan liittyvät tiedot. Toinen henkilökohtaisten opintosuunnitelmien ja niihin liittyvien ajoituksien tiedot. Kolmannessa osajoukossa oli opintosuunnitelmien tiedot. Näistä rakennettiin iso yhdistetty tietojoukko siten, että henkilökohtaisiin opintosuunnitelmiin yhdistettiin muut osajoukot. Yhdistetystä tietojoukosta rajattiin ulos eronneet, valmistuneet ja menehtyneet opiskelijat. Mukaan otettiin ainoastaan vuonna 2022 hyväksytyt henkilökohtaiset opintosuunnitelmat ja niihin liittyvät ajoitukset sekä tieto suorituksen tilasta, hyväksytty tai hylätty. Tila laskettiin hyväksytyksi, jos opiskelija oli saanut opintojakson toteutuksesta arvosanan tai hyväksymismerkinnän. Hylätyksi laskettiin hylätty suoritusmerkintä tai opiskelijan opintojakson suorittamatta jättäminen. Opintosuunnitelmat rajattiin sisältämään vain vuonna 2022 käyttöönotetut korkeakoulututkintoon liittyvät opetussuunnitelmat ja niiden opintojaksot. Kuviossa 9 esitelty yhdistetyn tietojoukon muodostaminen.



KUVIO 9: Yhdistetyn tietojoukon muodostaminen.

7.7 Piirteiden rakentaminen

Tietojen yhdistämisen jälkeen koneoppimiseen oli tarjolla lukuisia mahdollisia piirteitä. Tavoitteena oli ennustaa binäärisen luokituksen avulla suorittaako opiskelija henkilökohtaisessa opintosuunnitelmassa ajoitetun opinnon. Kohdemuuttujan nimi oli Tila, jossa arvoina suorituksen tila: (hyväksytty/hylätty). Sen kannalta

olennaisia tietoja tutkittiin ja seuraavat piirteet jalostettiin yhdistetystä tietojoukosta:

studentEntitlementId: opiskelijan yksilöivätunniste

degreeProgramme: koulutusohjelma

educationType: koulutuksen muoto

location: koulutuksen toimipiste

tim_semester: ajoituksen lukukausi

language: koulutuksen kieli

name_1: opintojakso

attendingSemesterCount: paikallaololukukausien määrä

notAttendingSemesterCount: poissaololukukusien määrä

sex: sukupuoli

e_nat: kansallisuus (jaoteltu määrän perusteella yleisimpään 4 ryhmään)

e_le: koulusivistyksen kieli (jaoteltu määrän perusteella yleisimpään 3 ryhmään)

ratio: laskettu prosentuaalinen osuus edellisellä lukukaudella ajoitetuista ja suoritetuista opintojaksoista

Piirteiden nimet pidettiin samoina kuin raakadatassa. Samat nimet helpottivat muistamaan myöhemmissä vaiheissa, mitä tietoja ne vastaavat alkuperäisessä datassa.

7.8 Datan analysointi ja tilastot

Yhteensä dataa kertyi 29277 riviä. Se sisälsi 3014 opiskelijan ja 242 opintojakson tietoja. Tietojoukossa ei varsinaisesti ilmentynyt tyhjiä rivejä, mutta opintojen alussa olevilla opiskelijoilla puuttui ratiopiirteen arvo. Näille käsiteltiin 0-arvo. Tietojoukosta löytyi 82 tuplariviä. Syy tuplariveille oli tiettyjen opiskelijoiden opintojakson toistuva suunnitteleminen omaan opintosuunnitelmaan. He tyypillisesti kuitenkin keskeyttävät opintojakso suorittamisen. Vaikka rivit ilmenevät tietojoukossa useasti, ne ovat muodostuneet järjestelmän käytön näkökulmasta oikealla tavalla eivätkä sen vuoksi aiheuttaneet toimenpiteitä. Alkuvaiheessa tietotyyppeinä oli 12 numeerista ja 3 kategorista muuttujaa. Näihin kuitenkin tehtiin myöhemmässä vaiheessa muutoksia. Tietojoukon tilastot on esitetty kuvassa 4.

Dataset statistics	
Number of variables	15
Number of observations	29277
Missing cells	0
Missing cells (%)	0.0%
Duplicate rows	82
Duplicate rows (%)	0.3%
Total size in memory	2.8 MiB
Average record size in memory	100.0 B

Variable types	
Numeric	12
Categorical	3

KUVA 4: Tilastot.

7.9 Datan käsittely

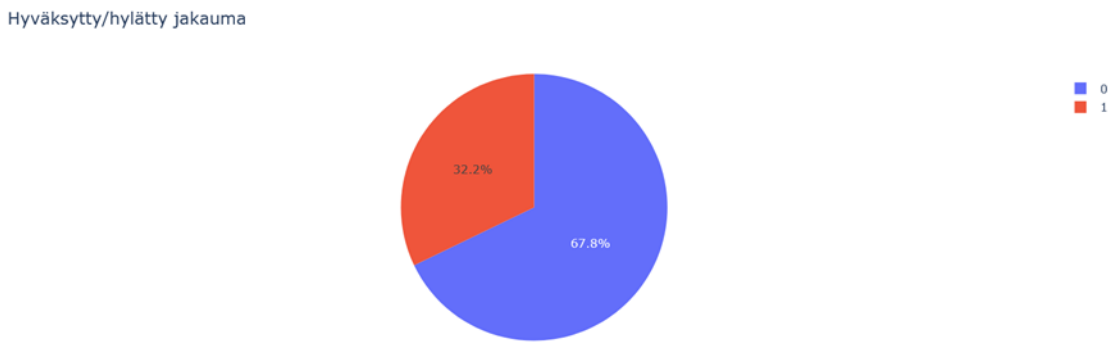
Python-ohjelmointikielen Pandas-kirjaston toiminnot ovat suosittuja koneoppivien algoritmien kehityksessä. Pandas-kirjaston ohella Scikit-learn -kirjastoa käytettiin laajasti kehitystyön eri vaiheissa. Useat Python-ohjelmointikielen data-analytiikkaan liittyvät toiminnot sekä koneoppivat algoritmit vaativat toimiakseen numeerisessa muodossa olevaa dataa. Sen vuoksi tietojoukon tekstimuotoiset ja kategoriset piirteet muunnettiin numeeriseksi Scikit-learn -kirjastoon kuuluvalla LabelEncoder-toiminolla. Kategorisia piirteitä olivat: studyEntitlementId, degreeProgramme, language, tim_semester, educationType, location, name_1, sex, e_nat, e_le ja tila. Kuvassa 5 esiteltynä lopullisen tietojoukon osajoukko. Tila kolumnissa hyväksytty sai muunnoksessa arvon 0 ja hylätty 1.

degree	educa	location	tim_seme	language	name_1	attending	notAttenc	sex	e_nat	e_le	age	ratio	tila
15	1	6	0	1	2	2	0	1	0	0	19	0	0
27	0	1	1	1	239	3	0	1	0	0	38	0	0
7	1	3	0	1	142	6	0	1	0	0	23	0	0
14	1	3	0	1	150	4	0	1	0	0	26	0	1
15	1	6	0	1	150	2	0	1	0	0	19	0	0
27	0	3	0	1	205	2	0	1	0	0	31	0	0
27	1	2	0	1	218	3	0	1	0	0	22	0.73	1

KUVA 5: Lopullisen tietojoukon osajoukko.

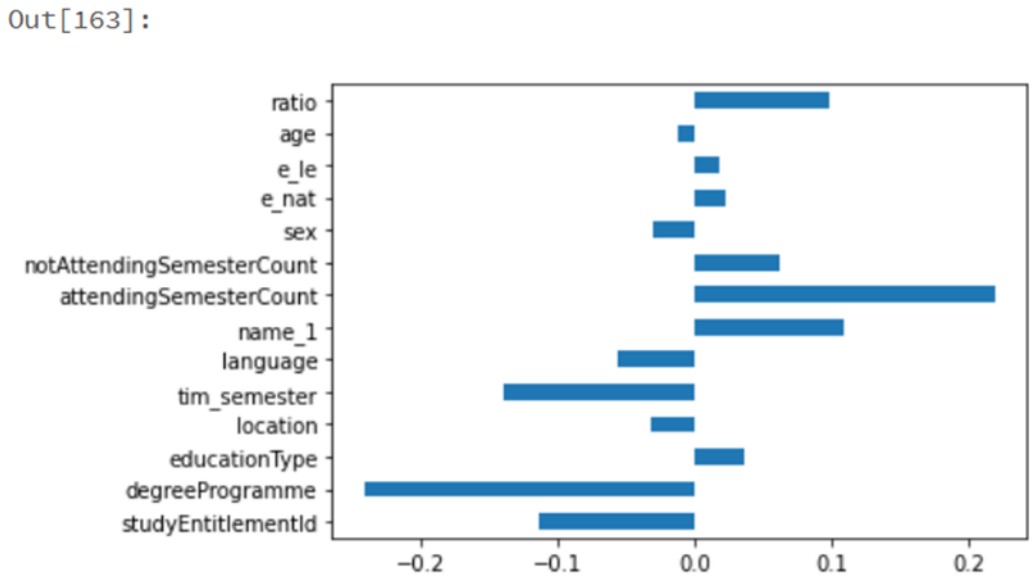
7.10 Datan jakauma ja korrelaatio

Koneoppimisen kannalta olennaista on kohdemuuttajan jakauman tarkastelu. Tila-kohdemuuttujan arvot jakautuivat 32,1% hylättyihin ja 67,9% hyväksyttyihin suorituksiin. Kohdemuuttujan jakauma on esitetty kuvassa 6.



KUVA 6: Kohdemuuttujan arvojen prosentuaalinen jakauma.

Georgen (2021) mukaan yksinkertainen tapa piirteiden sidonnaisuuksien etsimiseen kohdemuuttujaan nähden on käyttää korrelaatiota. Binaarisen luokittelun tapauksessa Pearsonin korrelaatio käyttäminen point biseral -menetelmällä on sopiva. Korrelaatioiden vahvuuden arviossa on useita tulkintoja, mutta vahvaksi korreloinniksi voidaan esimerkiksi mieltää 0.5 korkeammat arvot. (George 2021, luku 10.) Korrelaatioiden etsinnässä käytettiin Pandas-kirjaston corr-toimintoa. Erikoisen vahvaa korrelaatiota kohdemuuttujaan verrattuna ei historiadatasta löytynyt. Eniten riippuvuutta kohdemuuttujaan esiintyi attendingSemesterCount, name_1, tim_semester, degreeProgramme ja studyEntitlementId -piirteissä. Korrelaatiot ovat esiteltyinä kuvassa 7.



KUVA 7: Korrelaatioiden vertailu kohdemuuttajaan verrattuna (hyväksytty/hylätty).

7.11 Datan jakaminen

Lopullinen datajoukko jaettiin osiin koneoppimista varten. Jakamisessa käytettiin Scikit-learn -kirjaston train_test_split -toimintoa siten, että opetusdataksi käytettiin 80% koko datasta. Testi- ja validointidataksi säästettiin molempiin 10% datan kokonaismäärästä. Jakamisen jälkeen saatiin seuraavat dataframet:

x_trainn, opetus datajoukon piirteet

y_trainn, opetus datajoukon kohdemuuttuja

x_testt, testi datajoukon piirteet

y_testt, testi datajoukon kohdemuuttuja

x_valid, validointi datajoukon piirteet

y_valid, validointi datajoukon kohdemuuttuja

8 KONEOPPIMISEN SOVELTAMINEN

8.1 Automl

Binääriseen luokitteluun soveltuvia koneoppivia algoritmeja on lukuisia. Eri algoritmien suoriutumista voidaan kokeilla yksitellen tai koostetusti algoritmien vertailuun soveltuvalla Python-kirjastolla. Tässä tutkimuksessa algoritmien vertailuun käytettiin Databricks Automl -kirjastoa. Automl opettaa eri koneoppivia algoritmeja automaattisesti ja vertailee niiden suoriutumista. Tämän lisäksi se rakentaa valmiin koodiin mallien hyödyntämiseen. (Microsoft 2023.) Sen käyttö oli yksinkertaista. Import-komennolla otettiin käyttöön Automl-kirjasto. Mallille annettiin nimi ja koneoppimisen tavoitteeksi määriteltiin luokittelu. Tämän jälkeen syötettiin kohdemuuttujan sisältävä opetusdata, jonka se jaotteli itsenäisesti vielä opetus-, testi- ja validointiosiin. Lopuksi ilmoitettiin kohdemuuttuja sekä maksimiaika, jonka algoritmi voi toimia ennen automaattista keskeytystä. Automl-kirjaston käytön ohjelmointikoodi on esitetty kuvassa 7.

```
from databricks import automl

summary = automl.classify(trainn, target_col="tila", timeout_minutes=60)
```

KUVA 7: Automl-kirjaston käyttöönoton ohjelmointikoodi.

Ajon jälkeen algoritmi tulosti automaattisesti parhaan tuloksen saaneen mallin tulokset. Näiden ohella se tarjosi käyttöliittymät datan analysointiin ja koneoppivan mallin käyttöön sekä ajossa olleiden mallien vertailuun. Kuvassa 8 on esitelty Automl-ajon tulokset ja käyttöliittymät.

```
100%|██████████| 20/20 [02:00<00:00,  6.01s/trial, best loss: -0.7627964510691801]
```

For exploratory data analysis, open the data exploration notebook

To view the best performing model, open the best trial notebook

To view details about all trials, navigate to the MLflow experiment

Metrics for the best trial:

	Validation	Train
f1_score	0.763	0.850
score	0.774	0.855
recall_score	0.774	0.855
accuracy_score	0.774	0.855
log_loss	0.468	0.351
precision_score	0.766	0.855
roc_auc_score	0.827	0.923

KUVA 8: Automl-ajon tulokset ja sen rakentamat käyttöliittymät.

Kehitystyön kannalta olennaisin oli mallien vertailukäyttöliittymä. Sen sisältämien eri mittareiden tuloksia vertaamalla parhaiten menestyivät RandomForest-, LightGBM-, DecisionTree- ja XGBoost-algoritmit. Kuvassa 9 algoritmien vertailukäyttöliittymä.

Run Name	Created	Duration	Models	training_accurac	val_accuracy_scc
decision_tree	17 days ago	3.8s	sklearn	0.784	0.751
random_forest	17 days ago	8.8s	sklearn	0.833	0.754
xgboost	17 days ago	24.9s	sklearn	0.774	0.75
lightgbm	17 days ago	1.0min	sklearn	0.933	0.729

KUVA 9: Algoritmien vertailukäyttöliittymä.

8.2 Algoritmien vertailu

RandomForest-, LightGBM-, DecisionTree- ja XGBoost-algoritmit valittiin jatkokehitykseen. Kaikki neljä mallia perustuvat päätöspuurakenteeseen. Quinton (2020) mukaan ne eivät vaadi piirteiden skaalausta (eng. scaling) tai one-hot käsittelyä kategorisille muuttujille. One-hot -käsittelyä ei suositella, koska puurakenteesta voi tulla epätasapainoinen ja puut saattavat kasvaa erittäin suuriksi syvyyssuunnassa. (Quinto 2020, luku 1.) RandomForest-, LightGBM-, DecisionTree- ja XGBoost-algoritmien toimintaa kokeiltiin oletusasetuksilla opetus- ja testi-

dataa vasten. Niiden käytön ohjelmointi oli yksinkertaista. Import-komennolla tuotiin algoritmin sisältävä kirjasto sekä sen toiminnan arviointiin liittyvästä kirjastosta mittarit. Määritettiin malli ja koulutettiin se fit-komennolla opetusdataa vasten. Opetettua mallia käytettiin ennustamaan predict-komenolla testiaineistoa, jossa ei ollut mukana kohdemuuttujaa. Kuvassa 10 havainnollistettu mallin opetukseen liittyvä Python-ohjelmointikoodi.

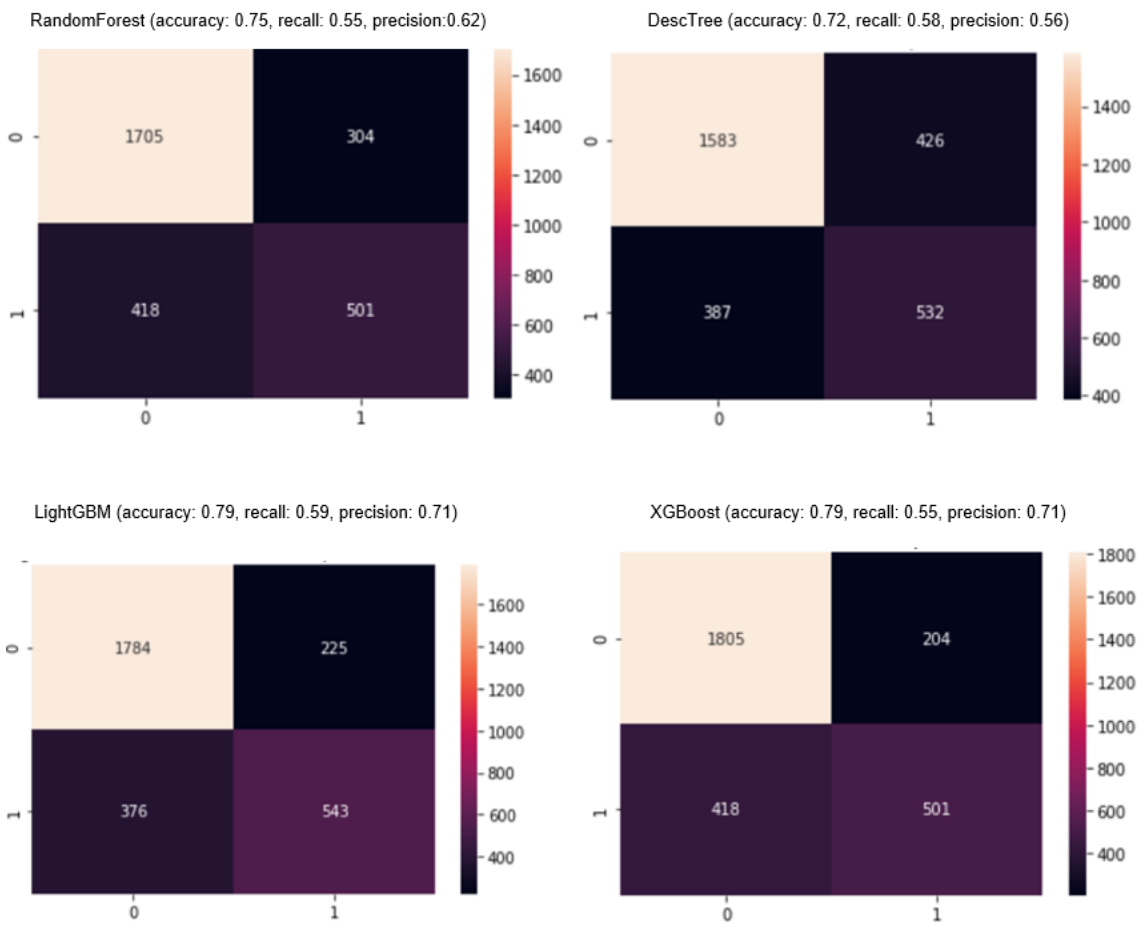
```
import lightgbm as ltb
from sklearn.metrics import auc, accuracy_score, confusion_matrix, mean_squared_error, recall_score, precision_score

model2= ltb.LGBMClassifier(objective= 'binary', boosting_type= 'gbdt')
model2.fit(x_trainn,y_trainn)
prediction2=model2.predict(x_testt)

confusion_matrix = confusion_matrix(y_testt, prediction2)
accuracy = accuracy_score(y_testt, prediction2)
recall = recall_score(y_testt, prediction2)
presicion = precision_score(y_testt, prediction2)
```

KUVA 10: Mallin opetuksen ohjelmointikoodi.

Mallin tuottamaa ennustetta verrattiin testiaineiston kohdemuuttajan arvoja vasten ja valituilla mittareilla, joista tulostettiin myöhemmin tässä luvussa esiteltävät konfuusiomatriisit. Parhaiten toimiviksi malleiksi osoittautuivat Gradient-Boosted Tree -mekanismiin perustuvat LightGBM ja XGBoost. Molemmat toimivat yhtä hyvin accuracy- ja precision-mittareilla, mutta LightGBM tuotti 4% paremman recall-tuloksen. Tästä syystä LightGBM valittiin kehitettäväksi malliksi. Algoritmien suoriutuminen on kuvattuna konfuusiomatriiseissa kuvassa 11.



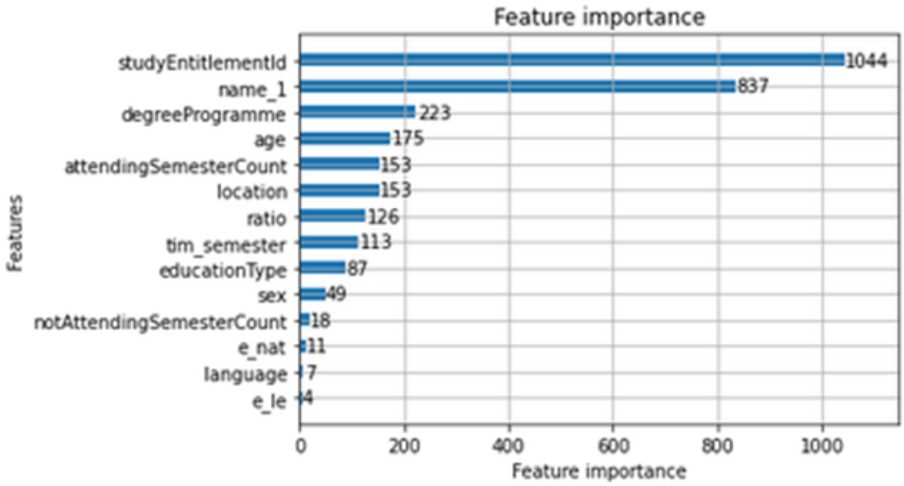
KUVA 11: Algoritmien vertailu.

8.3 LightGBM ja piirteiden tärkeyden vertailu

LightGBM on Microsoftin kehittämä suosittu GB-algoritmi ja soveltuu luokittelu- ja regressioennusteiden tuottamiseen. Se on suunniteltu toimimaan hajautetusti ja tehokkaasti, mikä nopeuttaa mallin koulutusaikaa sekä vähentää muistin käyttöä. (Quinto 2020, luku 1.) Seuraavaksi käsitellään algoritmin käyttöä, konfigurointia ja sen tuottamia tuloksia yksityiskohtaisemmin.

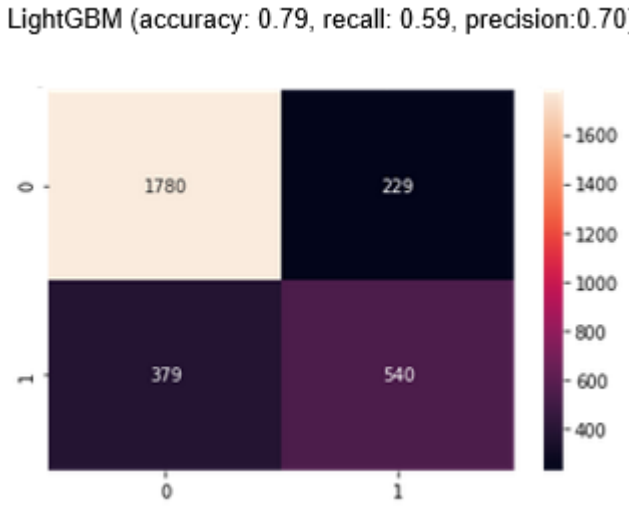
LightGBM-algoritmi sisältää toiminnon piirteiden tärkeyden arviointiin. Quinton (2020) mukaan se perustuu piirteiden tärkeyden pisteytykseen. Mitä korkeampi pistemäärä on, sitä suurempi merkitys sillä on mallin accuracyyn. (Quinto 2020, luku 1.) Tärkeimmäksi piirteiseksi oletusasetuksilla opetetulle mallille osoittautuivat: studentEntitlementId, name_1 ja degreeProgramme. Vähiten vaikutusta oli e_le, language ja e_nat piirteillä. Piirteiden tärkeyden pisteytys on kuvattuna kuvassa 12.

Out[174]:



KUVA 12: Piirteiden tärkeys.

Language- ja e_le-piirteet poistettiin datasta ja tarkistettiin, paranivatko tulokset. Accuracy- ja recall-arvot pysyivät samana alkuperäiseen piirrejoukkoon nähden, mutta precision-tulos huononi 1%. Näin ollen kehitystä jatkettiin alkuperäisellä piirrejoukolla. Konfuusio matriisi pienemällä piirrejoukolla kuvassa 13.



KUVA 13: Konfuusiomatriisi pienemmällä piirrejoukolla.

8.4 Hyperparametrit

Hyperparametrit kontrolloivat koneoppivan mallin toimintaa. Vaihtelemalla näiden parametrien arvoja voidaan vaikuttaa mallin suorituskykyyn. (Harrison 2019, luku 3.) GBT-mallit oppivat monimutkaisia sidonnaisuuksia, mutta ne ovat riskialttiita ylisovittamiselle. Sen vuoksi on tärkeää hallita niiden hyperparametrejä. (Jansen

2020, luku 12.) Algoritmi sisältää yli sata parametriä, joista Quinton (2020) mukaan tärkeimpiä ovat alla kuvatut parametrit.

Max_depth: säädetään puiden syvyyden kasvua. Syvyyden hallinnointi on tärkeää, koska matalat puut eivät yleisesti ylisovita mallia.

Num_leaves: kontrolloidaan puun monimutkaisuutta. Korkea arvo voi parantaa accuracya, mutta toisaalta aiheuttaa ylisovittamista.

Min_data_in_leaf: vaikutetaan mallin ali- ja ylisovittamiseen. Liian suuri arvo saattaa aiheuttaa alisuorittamisen.

Max_bin: algoritmi ryhmittelee piirteiden arvot ryhmiin. Ryhmien määrää säädetään lukumäärä arvolla. Pieni arvo voi parantaa mallin suoritusnopeutta ja ylisovittamista. Vastaavasti suuremmalla arvolla voidaan saavuttaa parempi accuracy.

Feature_fraction: LightGBM ottaa piirteistä alijoukkoja iteraatiohin. Tällä parametrillä säädetään alijoukon satunnaisia osuuksia iteraatioissa.

Bagging_fraction: Säätää datan määrä, joka otetaan mukaan jokaiseen iteraatioon.

Num_iterations: Määrittää boosting eli virheidenkorjaus iteraatioiden määrää.

(Quinto 2020, luvut 1, 3.)

8.5 Hyperopt

Sopivien hyperparametriasetusten etsimiseen on olemassa eri Python-kirjastoja ja menetelmiä. Tässä tutkimuksessa käytettiin Hyperopt-kirjastoa. Sen käytössä on määriteltävä avaruus (eng. space), josta parhaita arvoja etsitään, sekä tavoitefunktio. Tavoitefunktio ottaa hyperparametrit syötteenä ja antaa takaisin mittarin, esimerkiksi accuracy, jonka tulosta halutaan pienentää. (Agrawal 2020, luku 4.) Owenin määritelmään perustuen kirjaston tavoitteena on minimointi. Tästä

syystä luokittelussa, jossa tavoitteena on maksimointi perustuen esimerkiksi accuracyyn täytyy tavoitefunktioon lisätä negatiivinen merkintä. (Owen 2022, luku 8.) Hyperopt-kirjastoa käytetään opetus- ja testaustietojoukkoa vasten. Owenin mukaan (2021) tyypillisesti koneoppimisessa käytetään k-fold cross -validointia mallin opetuksessa. Hyperoptin kanssa sen käyttöä tulisi harkita, koska se pidentää algoritmin suoritusaikaa. Hyperopt-asetuksista löytyy toiminto, jolla määritettään lukumäärä hyperparametrikombinaatioille eli malleille, joita kokeillaan. Sen lukumäärä kasvattamalla voidaan saada vastaavuutta cross-validation -toiminnolle. (Owen 2021, Databricks.)

8.6 MLflow

Hyperoptin yhteydessä käytettiin MLflow-kirjastoa. Mlflow on rakennettu koneoppimisen liittyvien prosessien ja elinkaaren hallintaan. Sen avulla voidaan tallentaa ja vertailla mallien ajojen tuloksia. Tämän lisäksi se rakentaa helposti käytettävän rajapinnan, jolla malleja voidaan hyödyntää tuotantokäytössä. (MLflow Databricks, 2022 n.d.) Hyperoptia ajettiin eri hyperparametreilla ja niiden raja-arvoilla. Raja-arvojen määrittelyissä käytettiin perustana LightGBM-dokumentaatiossa (Parameters Tuning LightGBM, 2023 nd) esitettyjä suosituksia. Kuvassa 14 on esitetty kokeiltuja hyperparametrejä ja niiden arvojen vaihteluvälejä.

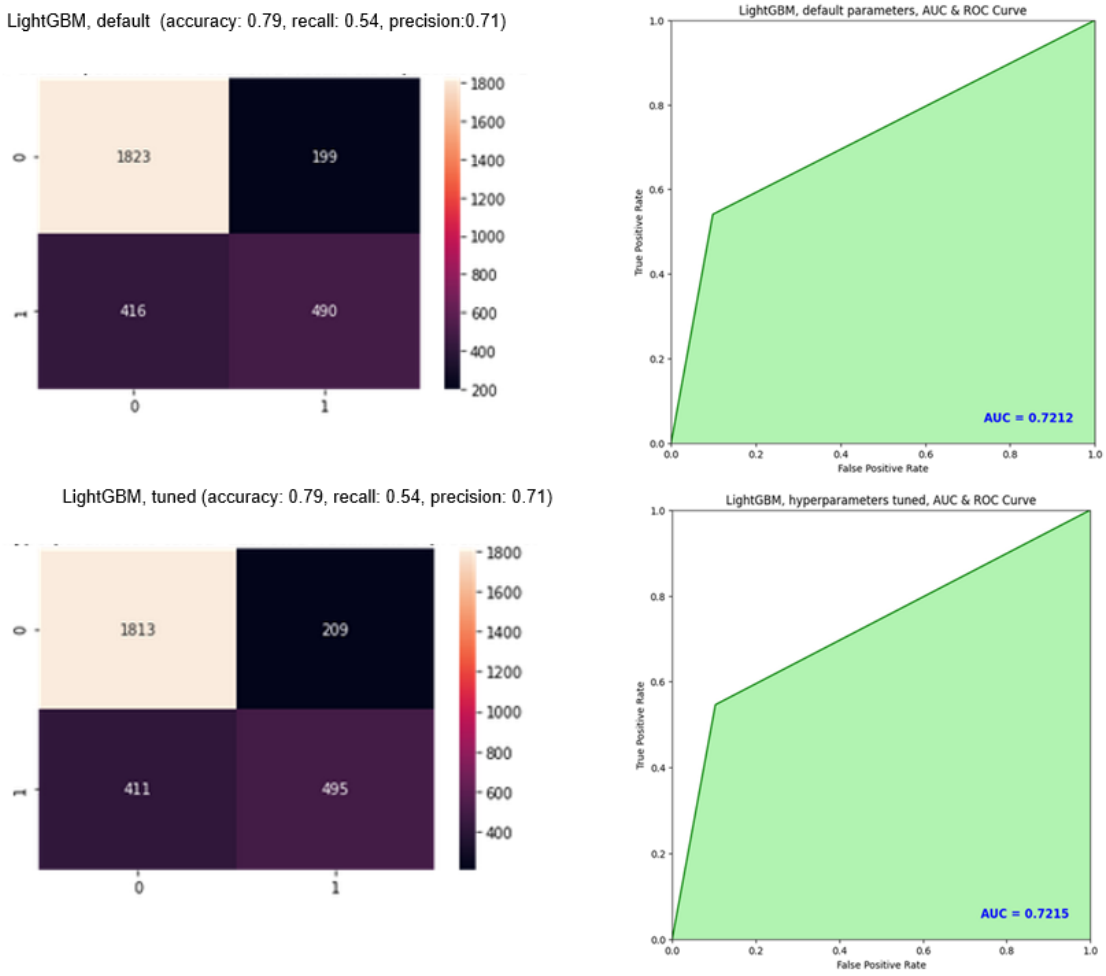
```
space = {
    'n_estimators': hyperopt.hp.quniform('n_estimators', 25, 500, 100),
    'colsample_bytree': hyperopt.hp.uniform('colsample_bytree', 0.5, 1.0),
    'learning_rate' : hyperopt.hp.uniform('learning_rate', 0.001, 0.1),
    'subsample': hyperopt.hp.uniform('subsample', 0.05, 1.0),
    'feature_fraction': hyperopt.hp.uniform('feature_fraction', 0.1, 1.0),
    'bagging_fraction': hyperopt.hp.uniform('bagging_fraction', 0.1, 1.0),
    'bagging_freq': scope.int(hyperopt.hp.quniform('bagging_freq', 0, 10, 1)),
    'max_depth':scope.int( hyperopt.hp.quniform('max_depth', 15, 100,1)),
    'min_data_in_leaf': scope.int(hyperopt.hp.quniform('min_data_in_leaf', 10, 200,1)),
     #The parameters below are cast to int using the scope.int() wrapper
    'num_iterations': scope.int(hyperopt.hp.quniform('num_iterations', 10, 200, 1)),
    'min_child_samples': scope.int(hyperopt.hp.quniform('min_child_samples', 10, 500, 1)),
    'num_leaves': scope.int(hyperopt.hp.quniform('num_leaves', 20, 50, 1))
}
```

KUVA 14: Hyperparametrit ja niiden arvot.

Mlflow:n tallentamien ajojen tuloksia vertailemalla huomattiin, että eri hyperparametriyhdistelmillä oli samankaltaiset tulokset. Paras Hyperoptin löytämä malli paransi noin 0.4% accuracya verrattuna oletusasetuksilla opetettuun malliin. Siinä käytettiin luvussa 8.4 kuvattuja hyperparametrejä.

8.7 Tuotantomalli

Mlflow-kirjaston tallentamaa hyperparametreillä optimoitua mallia sekä oletusasetuksilla opetettua mallia ajettiin aiemmin määriteltyä validation-dataa vasten. Molemmat mallit yliarvioivat hyväksytysti opintojakson toteutuksen suorittavien määrän. Osasyynä tähän on todennäköisesti hyväksytysti suorittavien suurempi määrä opetusdatassa. Mallientuottamat ennustukset olivat samat accuracy-, precision- ja recall-arvoja tutkittaessa. Hyperparametreillä optimoidun mallin AUC oli hieman parempi verrattuna oletusasetuksilla opetettuun malliin. Tästä syystä se valittiin tuotantomalliksi. Mallien suoriutuminen kuvattu kuvassa 15.



KUVA 15: Oletus asetuksilla ja hyperparametrein optimoidun mallin vertailu.

9 ENNUSTEEN JAKAMINEN

9.1 Mallin käyttö

Tuotantomallia hyödynnetään ottamalla käyttöön Mlflow-kirjasto ja kutsumalla sen hyperparametrien optimoinnin yhteydessä tallentamaa mallia. Malli ladattiin käyttöön mlflow.pync.load_model -komennolla, minkä jälkeen sillä pystyttiin ennustamaan predict-komennolla validointitietojoukkoa. Ennusteesta muodostettiin predict_df -niminen dataframe. Siihen kopioitiin yksilölliset tunnisteet validointitietojoukosta hyödyntämällä Pandas-dataframen indeksiä. Indeksin avulla yhdistettiin ennusteeseen validointitietojoukon kolumnit. Tuotantomallin käyttö ja tiedon yhdistely on kuvattu kuvassa 16 ja sen lopputulos kuvassa 17.

```
import mlflow
logged_model = 'runs:/xxxxxxxxxxxxxxxxxx/lightgbm-10_1_v2_23'

# Load model as a PyFuncModel.
loaded_model = mlflow.pyfunc.load_model(logged_model)

# Predict on a Pandas DataFrame.
import pandas as pd
ennuste=loaded_model.predict(pd.DataFrame(x_valid))

predicted_df = pd.DataFrame(data=ennuste, columns=['y_hat'],
                            index=x_valid.index.copy())
df_out= pd.merge(x_valid, predicted_df, how ='left', left_index=True,
                 right_index=True)
```

KUVA 16: Tuotantomallin käytön ohjelmointikoodi. Rajapinnan osoite muokattu.

degreePr	education	location	tim_seme	language	name_1	attending	notAttenc	sex	e_nat	e_le	age	ratio	y_hat
28	1	3	0	1	2	2	0	1	0	0	24	0	0
27	0	1	1	1	22	3	0	1	0	0	41	0	0
15	1	6	0	1	130	2	0	0	0	0	23	0	0
15	1	6	0	1	106	4	0	1	0	0	21	1	1
28	1	3	0	1	2	2	0	0	0	0	31	0	0
27	1	3	1	1	116	3	0	0	0	0	24	0	0
20	1	3	0	1	133	7	0	0	0	0	23	0	1
22	1	0	0	1	158	2	0	0	0	0	19	0	0
27	1	3	0	1	192	3	0	0	0	0	33	0.82	0

KUVA 17: Ennuste yhdistettynä validointi tietojoukkoon.

Ennusteessa olevat kategoriset tiedot olivat aiemman muunnoksen vuoksi numeerisessa muodossa. Niille yhdistettiin takaisin alkuperäiset nimet. Lopputuloksena syntynyt tietorakenne (eng. dataframe) tallennettiin parquet-muodossa tietoaltaaseen.

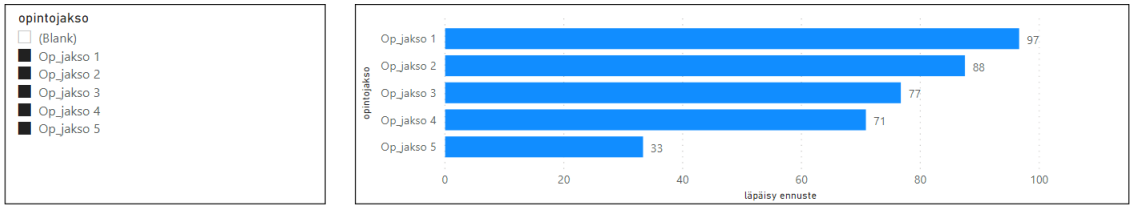
9.2 Power BI

Opinnäytetyön yhtenä tavoitteena oli rakentaa automaattisesti toimiva sovellus pilviympäristöön, josta sovelluksen käyttäjät pystyvät seuraamaan ennustetta reaaliajassa. Korkeakoulussa on käytössä Power BI -ohjelma, jota käytettiin rakennetun koneoppivan mallin tuottaman ennusteen jakamiseen. Arnold (2022) määrittelee Microsoft Power BI -ohjelman data-analytiikka ja visualisointi työkaluksi, joka on osa laajempaa ekosysteemiä. Se sisältää useampia komponentteja, joita hyväksikäyttämällä käyttäjät pystyvät rakentamaan ja jakamaan dataan liittyviä ratkaisuja eri tilanteissa. Keskeisimmät komponentit ovat Power BI Desktop ja Power BI service. Power BI Desktop -ohjelmalla muodostetaan yhteys dataan ja sen avulla muokataan sekä visualisoidaan data. Ohjelma on ilmainen ja se asennetaan omalle koneelle. Power BI service on verkossa toimiva SaaS palvelu, jolla Power BI Desktopilla rakennetut raportit voidaan jakaa organisaation sisällä. (Arnold 2022, luku 1.)

9.3 Ennusteen jakaminen ja visualisointi

Power BI Desktopissa pystyttiin muodostamaan yhteys Azure-tietoaltaassa sijaitsevaan ennusteeseen. Yhteyttä hyödyntämällä lopullisen ennusteen sisältävä parquet-tiedosto käsiteltiin ja visualisoitiin Power BI Desktopissa ennen julkaisemista Power BI serviceen. Ennuste sisälsi yksittäisen opiskelijan ennusteen henkilökohtaiseen opintosuunnitelmaan ajoitetusta opinnoista. Loppukäyttäjien kannalta kiinnostavampaa, oli saada tieto kokonaisten opintojaksojen läpäisytilanteesta. Sen vuoksi yksittäisten opiskelijoiden ennusteiden riveistä koostettiin laskennat, joiden lopputuloksena saatiin opintojaksoille yhteenlasketut ennusteet hyväksytysti ja hylätysti suoriutuville opiskelijoille. Tämän lisäksi laskettiin prosentuaalinen ennuste opintojakson hyväksytysti läpäiseville. Laskennat visualisoitiin siten, että ne esitettiin opintojakson nimen rinnalla taulukkomuotoisesti. Opinto-jaksojen prosentuaalinen läpäisyennuste esitettiin omana pylväsdiagrammina. Näkymään lisättiin osittaja, jolla loppukäyttäjät valitsevat haluamansa opintojaksot ennusteesta. Lopuksi visualisointi julkaistiin Power BI service -palvelussa. Power BI service palvellussa visualisoinnista pystyi helposti luomaan sovelluksen, johon määriteltiin kohderyhmään kuuluvat henkilöt ennen julkaisua.

Sovellus toimii Power BI servicen sisällä ja sitä voi seurata tietokoneelta ja mobiililaitteelta. Kuvassa 18 on esiteltynä näkymä julkaistusta sovelluksesta. Kuvassa opintojaksojen nimet muunnettu tietosuojasyiden vuoksi.



opintojakso	ennuste hyväksytty	ennuste hylätty	yhteensä	läpäisy ennuste %
Op_jakso 1	143	5	148	96,62
Op_jakso 2	49	7	56	87,50
Op_jakso 3	56	17	73	76,71
Op_jakso 4	102	42	144	70,83
Op_jakso 5	2	4	6	33,33
Total	**352**	**75**	**427**	**82,44**

KUVA 18: Power BI -näkymä ennusteesta kohderyhmälle.

10 TOIMINTAPROSESSI

Koneoppivan mallin toimintaan liittyvästä kokonaisuudesta saatiin rakennettua automaattisesti toimiva kokonaisuus. Peppi-järjestelmästä tietovarastoon tehty tiedonsiirto toimii itsenäisenä kokonaisuutena joka yö. Azure DataFactory -työkalulla toteutettu tiedonsiirto tietovarastosta Azure-tietoaltaaseen ajastettiin siirtämään halutut tiedostot kerran viikossa. Ennen tiedonsiirtämistä DataFactory poistaa aikaisemman tiedostot tietoaltaasta. Edellisen toiminnon suorittamisen jälkeen DataFactory-työkalulla laukaistiin Databricks-järjestelmällä tehty tiedon muokkaus ja koneoppivan mallin toiminta. Databricks poimii ensin uudet tiedostot tietoaltaasta ja sen jälkeen käsittelee datan koneoppivalle mallille soveltuvaan muotoon ja sen jälkeen suorittaa koneoppivan algoritmin. Lopuksi Databricks tallentaa ennusteen takaisin tietoaltaaseen. Power BI servicestä puolestaan asetettiin aikataulu päivitetyn ennusteen hakemiselle. Prosessi ja vaiheet ovat kuvattuina liitteessä 1.

11 POHDINTA JA JOHTOPÄÄTÖKSET

Peppi-järjestelmä sisältää arvokasta tietoa. Tiedonsiirtojen avulla raakadata tallennettiin Azure-pilviympäristöön. Peppi-järjestelmän raakadata sisältää monimutkaisia rakenteita ja tiedot olivat pirstaloituneet useisiin tiedostoihin. Azure-ympäristöön asennetulla Databricks-työkalulla tietoja pystyttiin käsittelemään, jalostamaan ja soveltamaan koneoppimista. Kehitystyön yhteydessä käytetyt Automl ja MLflow ovat kiinnostavia kokonaisuuksia. Ne sisältävät paljon toimintoja, joilla koneoppimisen kokeilemista ja käyttöönottoa voidaan helpottaa. Varsinkin MLflow tarjoaa hyvän perustan koneoppimisen käyttöönotolle ja elinkaaren hallinalle. Azure-ympäristö ja sinne asennetut työkalut osoittautuivat käyttökelpoiseksi. Niiden avulla kehitystyön tekeminen oli sujuvaa ja mielenkiintoista.

Koneoppivalle algoritmille syötettävän tietojoukon rakentaminen ja siihen liittyvien tietorakenteiden hahmottaminen sekä yhdistäminen oli haastava prosessi. Tämän jälkeinen tietojen muokkaaminen ja piirteiden jalostaminen oli jo helpompaa. Ajallisesti edellä mainitut vaiheet ottivat eniten aikaa opinnäytetyöhön liittyvän teknisen työn toteuttamisesta. Koneoppivien algoritmien kokeileminen hyvin valmisteltuun tietojoukkoon oli yksinkertaista, mutta niiden tuottamien ennusteiden tuloksien parantaminen osoittautui ennakkokäsityksen mukaisesti hitaaksi ja vaikeaksi prosessiksi. Uusien piirteiden jalostaminen on työlästä ja sopivien hyperparametriyhdistelmien löytäminen Hyperoptin kaltaisten kirjastojenkin avulla hidasta.

Opinnäytetyössä käytettyjen koneoppivien mallien tuloksista voidaan päätellä niiden tuovan lisäarvoa päätöksenteon tukemiseen. Tuotantomallin tuottamat tulokset olivat kohtalaiset. Kanasen ja Puolitaipaleen (2019) mukaan kolikon heiton AUC-arvo on 0,5 (Kananen & Puolitaival 2019, 181). Siihen verrattuna tuotanto mallin AUC 0.715 oli selvästi parempi. Opinnäytetyössä käytettiin vuoden 2022 alussa voimaan tulleen koulutusuudistuksen jälkeistä dataa, jota oli kertynyt kehitystyötä tehtäessä yhteensä vuoden ajalta. Tulokset todennäköisesti paranevat datamäärän lisääntyessä. Oppilasdatan hyödyntämisessä on kuitenkin muistettava, että se koostuu henkilöiden yksilöllisestä käyttäytymisestä. Näin ollen siihen liittyy sattumanvaraisuutta, joka heijastuu ennustuksen tuloksiin. Tuotantomallin

antamia tuloksia voi hyödyntää suunnittelun apuvälineenä ja niitä voi tulkita suuntaa-antavana pohjatietona.

Opinnäytetyössä käytettiin puurakenteisiin perustuvia algoritmeja. GBT-algoritmit sekä tuotantoon vallittu LightGBM osoittautuivat tehokkaiksi algoritmeiksi. LightGBM-hyperparametrejä optimoimalla saatiin pieni parannus tuloksiin. Hyperparametrejä ja niiden avaruuksia tutkimalla vielä tarkemmin olisi mahdollista todennäköisesti parantaa ennustetehoa ja vähentää ylisovitusta. Tämän lisäksi Hyperopt-funktion tavoitteena olisi voinut kokeilla laajemmin eri mittareita, kuten recall tai AUC. Toisaalta hyperparametrien optimoiminen vei aikaa ja mallin tuloksien paraneminen oli vähäistä. Todennäköisempi edistävä tekijä tuloksen paranemiseen olisi etsiä uusia piirteitä. Opinnäytetyötä tehdessä heräsikin kysymys, kuinka paljon työaikaa kannattaa käyttää koneoppivan mallin tulosten parantamiseen, jos saavutettu hyöty prosentuaalisesti mitattuna on esimerkiksi alle 1%? Datamäärän kasvaessa olisi mielenkiintoista kokeilla neuroverkkoa ja tutkia pystyisikö sen avulla rakentamaan tehokkaamman ennusteen.

Opinnäytetyön johdannossa esitettyihin tutkimuskysymyksiin saatiin vastaukset ja tutkimuksen liittyvissä kehitystyössä saatiin rakennettua automaattisesti toimiva prosessi sekä sovellus ennusteen jakamiseen kohderyhmällä palvilaskentaa hyödyntäen. Edellä mainittujen tavoitteiden saavuttamisessa koen onnistuneeni opinnäytetyössäni hyvin. Korkeakoulun kannalta saatiin kuvattua koneoppimiseen liittyvä prosessi ja sen vaiheet sekä tuotantotapa organisaation pilviympäristössä. Kehitystyössä toteutettu tuotantomalli on itsessään toimiva ja sitä voidaan hyödyntää päätöksenteon tukena suunniteltaessa tulevaa.

Tässä opinnäytetyössä kuvattu tekninen kehittäminen on kirjoittajan toteuttamaa. Osallistamalla useampia henkilöitä mukaan datan tutkimiseen ja tulkintaan sekä koneoppimiseen liittyvään tekniseen kehittämiseen saadaan enemmän näkökulmia, jolloin lopputuloksena syntyvä ennuste todennäköisesti paranee. Tämän lisäksi varsinkin datan tutkimisen ohessa syntyy tyypillisesti ideoita järjestelmien tehokkaampaan käyttöön sekä löydetään uusia riippuvaisuuksia tietojen välillä. Luvussa 5.2 esiteltiin koneoppimiseen liittyvää prosessia ja työtehtäviä. Koneop-

pivien tuotantomallien rakentaminen on työlästä ja yhden henkilön tekemänä raskasta. Jatkossa koneoppimiseen liittyvään tekniseen prosessiin korkeakoulun kannattaa hankkia lisäresursseja.

Opinnäytetyössä kuvattua toimintaprosessia sekä siihen liittyviä työkaluja ja algoritmeja voidaan hyödyntää toisissa organisaatiossa. On kuitenkin tärkeää ymmärtää, että kuvattujen työkalujen ja ohjelmien käyttöön liittyy kustannuksia. Tämän lisäksi Azure-ympäristön hyödyntäminen edellyttää organisaatiolta hallintamallia sen käyttöön, asiantuntemusta ympäristön konfigurointiin sekä huolellista perehtymistä tietosuoja- sekä tietoturvaseikkoihin.

LÄHTEET

Agrawal, T. 2020. Hyperparameter Optimization in Machine Learning: Make Your Machine Learning and Deep Learning Models More Efficient. Apress. Viitattu 25.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/hyperparameter-optimization-in/9781484265796/

Alpaydin, E. 2021. Koneoppiminen. Helsinki: Terra Cognita Oy.

Andersson, J C. 2023. Learning Microsoft Azure. O'Reilly Media, Inc. Viitattu 11.2.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/learning-microsoft-azure/9781098113315/

Arnold, J. 2022. Learning Microsoft Power BI. O'Reilly Media, Inc. Viitattu 28.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/learning-microsoft-power/9781098112837/

Brink, H., Richards J W. & Fetherolf, M. 2017. Real-world Machine Learning. Manning Publications. Viitattu 9.2.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/real-world-machine-learning/9781617291920/

Chojecki, P. 2020. Artificial Intelligence Business: How you can profit from AI. Packt Publishing. Viitattu 23.12.2022. Vaatii käyttöoikeuden.

Databricks. n.d. Databricks pricing. Verkkosivu. Viitattu. 11.12.2022. https://www.databricks.com/product/pricing

Databricks. n.d. Managed MLflow. Verkkosivu. Viitattu 25.1.2023. https://www.databricks.com/product/managed-mlflow

Delen, D. 2020. Predictive Analytics: Data Mining, Machine Learning and Data Science for Practitioners, 2nd Edition. Pearson FT Press. Viitatttu 20.11.2022. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/predictive-analytics-data/9780135946527/

Elements of AI. n.d. Related fields. Verkkosivu. Luettu 17.12.2022. https://course.elementsofai.com/1/2

George, N. 2021. Practical Data Science with Python. Packt Publishing Ltd. Viitattu 17.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/practical-data-science/9781801071970/

Géron, A. 2022. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 3rd Edition. O'Reilly Media, Inc. Viitattu 23.12.2022. Vaatii käyttöoikeuden.

https://learning.oreilly.com/library/view/hands-on-machine-learning/9781098125967/

Gollapudi, S. 2016. Practical Machine Learning. Packt Publishing Ltd.

Griegoriev, A. 2021. Machine Learning Bookcamp. Manning Publications. Viitattu 7.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/machine-learning-bookcamp/9781617296819/

Harrison, M. 2019. Machine Learning Pocket Reference. O'Reilly Media, Inc. Viitattu 24.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/machine-learning-pocket/9781492047537/

https://journal.fi/hallinnontutkimus/article/view/107896/65875. https://learning.oreilly.com/library/view/artificial-intelligence-business/9781800566514/

Jansen, S. 2020. Machine Learning for Algorithmic Trading. Packt Publishing. Viitattu 24.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/machine-learning-for/9781839217715/

Kahate, A. 2004. Introduction to Database Management Systems. Pearson India. Viitattu 27.11.2022. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/introduction-to-database/9788131700785/

Kananen, H. & Puolitaival, H. 2019. Tekoäly bisneksen uudet työkalut. Helsinki: Alma Talent ry.

Kelleher, J D. 2020. Syväoppiminen kuinka tekoäly toimii. Helsinki: Terra Cognita oy.

Lakshmanan, V., Robinson, S., Munn, M. 2020. Machine Learning Design Patterns. O'Reilly Media, Inc. Viitattu 10.2.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/machine-learning-design/9781098115777/

LightGBM. n.d. Parameters Tuning. Verkkosivu. Viitattu 26.1.2023. https://lightgbm.readthedocs.io/en/latest/Parameters-Tuning.html#

Linthicum, D. 2023. An Insider’s Guide to Cloud Computing. Addison-Wesley Professional. Viitattu 6.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/an-insiders-guide/9780137935819/

Microsoft. 2023. What is Automl? Verkkosivu. Viitattu 18.1.2023. https://learn.microsoft.com/en-us/azure/databricks/machine-learning/automl/

Microsot. n.d. What is Azure? Verkkosivu. Viitattu 27.11.2022. https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-azure/

Niemi, H. 2021. Miten tekoäly muuttaa oppimista ja koulutusta? Teoksessa Älykäs huominen. Helsinki: Gaudeamus.

Owen, S. 2021. Engineering blog. Verkkosivu. Viitattu 25.1.2023. https://www.databricks.com/blog/2021/04/15/how-not-to-tune-your-model-with-hyperopt.html

Owen, S. 2022. Hyperparameter Tuning with Python. Packt Publishing. Viitattu 25.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/hyperparameter-tuning-with/9781803235875/

Palacio, A B. 2021. Distributed Data Systems with Azure Databricks. Packt Publishing. Viitattu 10.12.2022. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/distributed-data-systems/9781838647216/

Peppi-konsortio. Käyttöliittymä. Verkkosivu. Viitattu 11.12.2022. https://www.peppi-konsortio.fi/kayttoliittyma/

Peppi-konsortio. n.d. Peppi voimaa ja vääntöä. Verkkosivu. Viitattu 26.11.2022. https://www.peppi-konso tio.fi/

Popkhadze, N. 2021. The Good, The Bad and The Ugly: AI in the higher education. Hallinnontutkimus 4/2021. Luettu 18.11.2022.

Prosise, J. 2022. Applied Machine Learning and AI for Engineers. O'Reilly Media, Inc. Viitattu 4.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/applied-machine-learning/9781492098041/

Quinto, B. 2020. Next-Generation Machine Learning with Spark. Apress. Viitattu 7.1.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/next-generation-machine-learning/9781484256695/

Raschka, S., Liu, Y., Mirjalili, V., Dzhulgakov, D. 2022. Machine Learning with PyTorch and Scikit-Learn. Packt Publishing. Viitatttu 3.2.2023. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/machine-learning-with/9781801819312/'

Tarkoma, S. 2021. Miten tekoäly vaikuttaa kokonaisturvallisuuteen? Teoksessa Älykäs huominen. Helsinki: Gaudeamus.

Tomcy, J. & Pankaj, M. 2017. Data Lake for Enterprises. Packt Publishing. Viitattu 27.11.2022. Vaatii käyttöoikeuden. https://learning.oreilly.com/library/view/data-lake-for/9781787281349/d9481f45-a45e-47d0-8e4a-56e18e54f4cf.xhtml

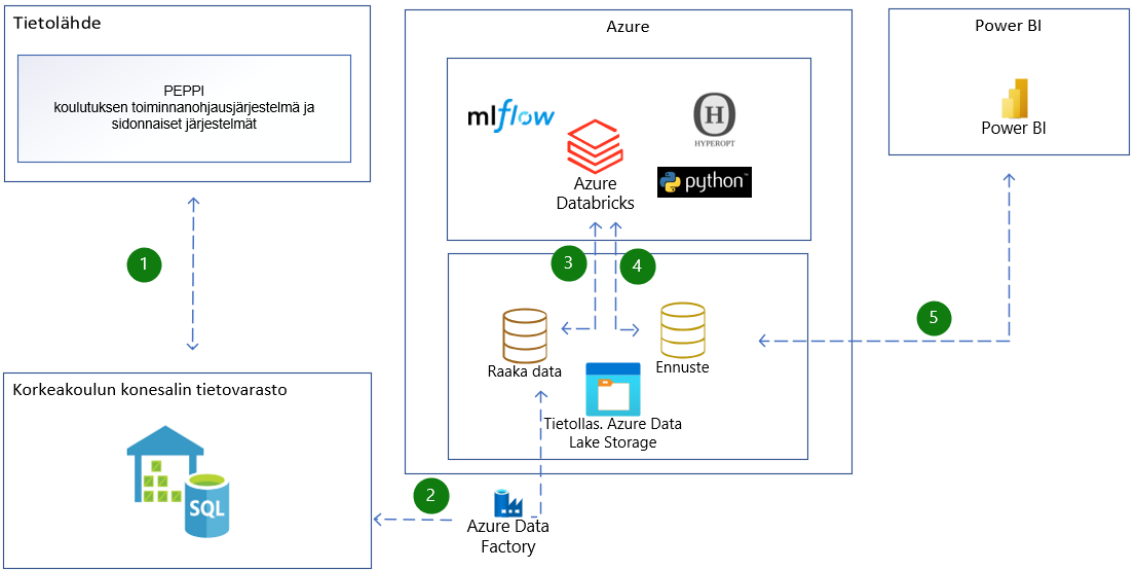
Tshepo, C N. 2021. Data Science Solutions with Python: Fast and Scalable Models Using Keras, PySpark MLlib, H2O, XGBoost, and Scikit-Learn. Apress. Viitattu 7.1.2023. Vaatii käyttöoikeuden.
https://learning.oreilly.com/library/view/data-science-solutions/9781484277621/

Virtanen, A. 2006. Konstruktiivinen tutkimusote. Miten koulutus ja elinkeinoelämän odotukset kohtaavat ammattikorkeakoulun opinnäytetöissä. Ammattikasvatuksen aikakauskirja, 8. Pdf-dokumentti. Viitattu 20.11.2022.
https://journal.fi/akakk/article/download/114874/67807/223935

Xiao, P. 2022. Artificial Intelligence Programming with Python. Wiley. Viitattu 11.12.2022. Vaatii käyttöoikeuden.
https://learning.oreilly.com/library/view/artificial-intelligence-programming/9781119820864/

LIITTEET

Liite 1. Koneoppimiseen liittyvä kokonaisuus ja prosessin vaiheet



1. Tiedonsiirto Pepistä tietovarastoon
2. Tiedonsiirto tietovarasto Azure tietoaltaaseen
3. Tietojen muokkaus ja koneoppiminen
4. Ennusteen tallennus tietoaltaseen
5. Ennusteen seuranta