SAVONIA
ammattikorkeakoulu

OPINNÄYTETYÖ - AMMATTIKORKEAKOULUTUTKINTO
TEKNIIKAN JA LIIKENTEEN ALA

TEKOÄLYKOODIN TUNNISTAMINEN BAYES-LUOKITTIMELLA

TEKIJÄ Markku Viertokangas

SAVONIA-AMMATTIKORKEAKOULU

OPINNÄYTETYÖ
Tiivistelmä

Koulutusala Tekniikan ja liikenteen ala	
Tutkinto-ohjelma Tietotekniikan tutkinto-ohjelma	
Työn tekijä Markku Viertokangas	
Työn nimi Tekoälykoodin tunnistaminen Bayes-luokittimella	
Päiväys 24.05.2024	Sivumäärä/Liitteet 26
Toimeksiantaja/Yhteistyökumppani(t) Savonia ammattikorkeakoulu	

Tiivistelmä

Tekoälyn yleistyttyä, sen hyödyntäminen ” oikotienä ” oppimisen sijaan on myös yleistynyt, eikä varsinaista tunnistusohjelmaa ole vielä olemassa siihen, onko tekoälyä hyödynnetty esimerkiksi koulutehtävien kanssa. Tämän opinnäytetyön tarkoituksena oli selvittää ja mahdollisesti rakentaa prototyyppi, joka kykenisi analysoimaan, onko sen käsittelemä data tuotettu tekoälyä hyödyntäen vai ei. Kehittämistyö toimi tilaajalleen, Savonian ammattikorkeakoululle, pohjana jatkokehittelyä varten.

Opinnäytetyö toteutettiin Python-ohjelmointikielellä, koska sillä on valmiina laaja valikoima erilaisia tekoälyyn ja koneoppimiseen tarkoitettuja kirjastoja. Työn tilaaja toimitti Bayes-luokittimen koulutusmateriaaliksi ohjelmointitehtäviä, joita Savonia-ammattikorkeakoulun tietotekniikan alan opiskelijat olivat tehneet. Luokittimen koulutusmateriaali on pääsääntöisesti tehty ennen vuotta 2022, jolloin ChatGPT julkaistiin maailmanlaajuisesti ihmisten käyttöön.

Lopputuloksena syntyi Bayes-luokitin, joka kykeni tunnistamaan lupaavasti tekoälyn luomaa koodia. Kyseessä on ns. proof of concept eli tulokset on rohkaisevia, mutta koska opetusaineisto oli suppea niin tulokset on enemmänkin suuntaa antavia jatkokehitys varten. Tämä opinnäytetyö osoitti että järjestelmä, joka osaa analysoida tekoälyn tuottamaa dataa on mahdollista toteuttaa.

Avainsanat

tekoäly, koneoppiminen, naiivi Bayes-luokitin, Python

SAVONIA UNIVERSITY OF APPLIED SCIENCES

THESIS
Abstract

Field of Study Technology, Communication and Transport	
Degree Programme Degree Programme in Information Technology	
Author Markku Viertokangas	
Title of Thesis Recognizing AI Code with a Bayes Classifier	
Date 24 May 2024	Pages/Appendices 26
Client Organisation /Partners Savonia University of Applied Sciences	

Abstract

With the increasing prevalence of artificial intelligence, its use as a "shortcut" for learning has also become more common. There is currently no proper detection software to determine if AI has been used, for example, in school assignments.

The purpose of this thesis was to explore and possibly build a prototype capable of analyzing whether the data it processes was produced using artificial intelligence or not. This development work would serve as a basis for further development for the client, Savonia University of Applied Sciences.

The thesis was implemented using Python programming language, because it offers a wide range of libraries for artificial intelligence and machine learning. The client provided training material for a Bayes classifier. This material consisted of programming assignments done by students in the field of information technology at Savonia University of Applied Sciences. Most of the training material was created before 2022, which is the year ChatGPT was launched for public use.

As a result, a Bayes classifier was developed that could identify AI-generated code with great promise. This is a proof of concept; the results are encouraging, but due to the limited training data, they are more indicative of further development. This thesis demonstrated that a system capable of analyzing AI-generated data is feasible to implement.

Keywords

artificial intelligence, machine learning, naive Bayes classifier, Python

SISÄLTÖ

KUVALUETTELO

1 JOHDANTO

Tehtävien tunnistaminen ja automaattinen arviointi ovat keskeisiä osa-alueita ohjelmoinnin opetuksessa ja arvioinnissa. Tulevaisuudessa tekoälypohjaiset järjestelmät voivat tarjota tehokkaan ja tarkemman tavan tunnistaa ja arvioida opiskelijoiden ohjelmointitehtäviä.

Tämän opinnäytetyön tarkoituksena oli selvittää ja mahdollisesti rakentaa prototyyppi, joka kykenisi analysoimaan, onko sen käsittelemä data tuotettu tekoälyä hyödyntäen vai ei. Kehittämistyö toimisi tilaajalleen, Savonian ammattikorkeakoululle, aihiona jatkokehittelyä varten. Tekoälyn yleistyttyä, sen hyödyntäminen ” oikotienä ” oppimisen sijaan on myös yleistynyt, eikä varsinaista tunnistusohjelmaa ole vielä olemassa siihen, onko tekoälyä hyödynnetty esimerkiksi koulutehtävien kanssa.

Opinnäytetyön tuotoksena syntyi lopulta Bayes-luokitin, joka pystyi tunnistamaan tekoälyn tuottaman koodin, joka toteutin Python-ohjelmointikielellä. Python valikoitui ohjelmointikieleksi, koska sillä on selkeä syntaksi ja laaja valikoima erilaisia kirjastoja valmiina. Työn tilaajalta sain luokittimen koulutusmateriaaliksi ohjelmointitehtäviä, joita tietotekniikan alan opiskelijat olivat tehneet.

Tässä opinnäytetyössä esitellään ensin tekoälyn peruskäsitteitä ja sen sovellusmahdollisuuksia ohjelmoinnin opetuksessa. Tämän jälkeen käydään läpi todennäköisyyslaskentaa ja Python-ohjelmointikielen ominaisuuksia ja sen käyttöä tekoälyratkaisujen kehittämisessä.

Varsinaisessa toteutusosassa esitellään Bayes-luokittimen työvaiheet ja keskeisimmät toimintaperiaatteet. Lopuksi kehittämistyössä esitellään järjestelmän testituloksia ja arvioidaan sen suorituskykyä ja tarkkuutta erilaisten ohjelmointitehtävien tunnistamisessa ja pohditaan järjestelmän mahdollisia jatkokehittämismahdollisuuksia.

2 TEKOÄLY

Tekoälyllä voidaan viitata koneiden kykyyn hyödyntää ihmisen älykkyyteen liittyviä taitoja, kuten päättelyä, oppimista, suunnittelua ja luovuutta. Tekoälyn avulla tekniset järjestelmät voivat havainnoida ympäristöään, käsitellä kerättyä tietoa ja ratkaista vaativiakin ongelmia saavuttaakseen tavoitteensa. Esimerkiksi tietokone kykenee vastaanottamaan tietoa, jota sen sensorit (esim. kamera), ovat keränneet ja saadun tiedon perusteella tiedon ennen vastaamista. Tekoälyjärjestelmät voivat näin ollen kehittää käyttäytymistään itsenäisesti analysoidessaan aiempien toimintojensa seurauksia ja toimimalla tämän pohjalta. (Euroopan parlamentti 2023.)

Tekoäly tekniikkana on pohjimmiltaan yhdistelmä ohjelmointia, matematiikkaa ja tilastotiedettä. Periaatteessa tekoälyn toimintaa määrittelevät perusasiat ovat yksinkertaisia ja helppoja matemaattisia käsitteitä. Toiminnan monimutkaisuus tulee esiin siinä, kun sitä aletaan soveltamaan käytäntöön tasolla toimivaksi, sillä koneet käsittelevät ihmistä nopeammin ja tehokkaammin moniulotteisia ja monitasoisia muuttujia. Tekoälylle ulottuvuuksia voi olla teorian tasolla rajaton määrä. (Kananen & Puolitaival 2019, luku 1.)

Nopeudesta ja tehokkuudesta huolimatta, kehittyneemmätkin tekoälyohjelmat ovat rajallisten resurssien, kuten tiedon, ajan ja energian varassa. Älykkäät ohjelmat pyrkivät löytämään tilanteeseen sopivan kompromissin sen sijaan, että se etsisi parasta mahdollista vaihtoehtoa yksittäisen halutun lopputuloksen mukaisesti. Esimerkiksi hätätilanteessa, kuten auton lähestyessä jalankulkijaa, ohjelman on kyettävä tekemään nopea päätös hätäjarrutuksesta. Epävarmassa tilanteessa on tarkoituksenmukaisempaa jarruttaa liikaa, kuin ottaa riski jalankulkijan yliajosta, eli tekoäly koittaa minimoida riskit yleisellä tasolla. Tekoälyn toiminnan tarkoituksenmukaisuus on näin ollen subjektiivista. (Toivonen 2023, luku 10.)

Toisinaan on helpompaa kertoa tekoälyohjelmalle, millaiseen lopputulokseen tehtävässä pitäisi päästä. Esimerkiksi koulun lukujärjestyksen laatimisessa voidaan käyttää yleiskäyttöistä algoritmia, joka pyrkii löytämään parhaiten annettuihin reunaehtoihin sopivan ratkaisun, kuten lukujärjestyksen, ilman että tarvittaisiin erillistä algoritmia jokaiseen mahdolliseen skenaarioon. Tehtävissä, joissa ratkaisun määrittely on vaikeaa, mutta joista on olemassa onnistuneita esimerkkejä. Tällaisiin tehtäviin soveltuu koneoppiminen, jossa ohjelma oppii sille annettujen esimerkkien avulla. Esimerkiksi puheentunnistuksessa kerätään ääninäytteitä ja niiden vastaavaa tekstiä, jonka avulla koneoppimisalgoritmi oppii tunnistamaan äänteitä ja muuttamaan ne tekstiksi. (Toivonen 2023, luku 11.)

Usein tekoälyohjelmissa yhdistetään näitä kaikkia lähestymistapoja. Suorat ohjeet perustuvat yleensä algoritmeihin, jotka huolehtivat kokonaisuudesta, mutta ne voivat hyödyntää myös koneoppimisen malleja osatehtävissä. Koneoppimisen avulla voidaan ratkaista osaongelmia, ja algoritmi valvoo näiden osatehtävien toimintaa varmistaen, että lopullinen ratkaisu on sekä järkevä että turvallinen. (Toivonen 2023, luku 11.)

2.1 Tekoälyn historia

Terminä ja tieteenalana tekoäly vakiintui vuonna 1956, kun tutkijat John McCarthy, Marvin Minsky, Claude Shannon ja Nathan Rochester järjestivät kesäseminaarin Dartmouth Collegessa Yhdysvalloissa, jonne he kutsuivat sinne muita aiheesta kiinnostuneita tutkijoita. Suurin osa osallistujista oli taustaltaan matemaatikkoja, sillä tietojenkäsittelytiede oli vasta alkamassa erottua omaksi tieteenalakseen. Dartmouthissa tutkimusalue sai nimekseen "artificial intelligence", eli "keinotekoinen älykkyys ", joka myöhemmin lyhennettiin "tekoälyksi" ja vielä lyhyemmäksi "AI:ksi". Vaikka tekoälyn tutkimusta oli tehty ennen seminaaria, sillä ei ollut vakiintunutta käsitettä. (Toivonen 2023, luku 9.)

Darthmountin Seminaarissa oletettiin, että oppiminen ja muu älykkyyttä vaativa toiminta voitaisiin teoreettisesti kuvata niin tarkasti, että niitä voitaisiin simuloida tietokoneella. Tämä käsitys on myöhemmin kuitenkin hylätty, sillä joidenkin tehtävien ratkaisuun ei päästä täsmällisten ohjeiden antamisella, mutta jotka voidaan ratkaista koneoppimisen avulla. Uusien käsitteiden muodostaminen, automaattinen ongelmien ratkaiseminen ja itseään kehittävät ohjelmat ovat edelleen keskeisiä teemoja tekoälyn tutkimuksessa. (Toivonen 2023, luku 9.)

1950-luvulla kehitetyt tekoälysovellukset toimivat tavallaan esi-isinä niille sovelluksille, joita käytetään tänä päivänä. Yksi sen ajan edelläkävijöistä oli tietokone, Ferranti Mark I, joka oli kykeneväinen pelaamaan shakkia ihmistä vastaan. Sen kyvyt eivät kuitenkaan yltäneet aivan mestaritasolle, sillä sen pelitaktiikka oli lähinnä reaktiivinen ja se kykeni ennustamaan ainoastaan kolmen siirron päähän. Ferranti Mark I voidaan pitää merkittävänä siinä mielessä, ettei se ollut ainoastaan laskukone, vaan se kykeni, tosin rajoitetusti, tekemään päätöksiä ja strategioimaan. (Salo 2023, luku 1.1.)

Toinen varhaisena tekoälysovelluksena pidetty ENIAC, suunniteltiin alun perin suorittamaan yksinkertaisia matemaattisia laskutoimituksia. Vaikka ihmisen ohjaaman ENIAC:n kyvyt olivat varsin rajalliset, se osoitti tietokoneiden voivan suorittaa tehtäviä, jotka aikaisemmin olivat olleet vain laskutaitoisten ihmisten ulottuvilla. (Salo 2023, luku 1.1.)

2.2 Koneoppiminen

Koneoppimisella (machine learning) tarkoitetaan tietokoneiden kykyä oppia ja parantaa suorituskykyään kokemuksen kautta. Se on tavallaan tekoälyn osa-alue, jonka perusperiaatteena on se, että kone oppii ilman suoraa ohjelmointia, eli analysoi itsenäisesti erilaisia rakenteita ja kuvioita pystyen myös tulkitsemaan niitä. Koneoppimisen myötä tekoälyt pystyvät siis oppimaan, tekemään päätöksiä ja päättelyä ja luomaan niiden pohjalta älykkäitä ratkaisuja ilman ihmisen väliintuloa. Koneoppimisen myötä tekoälylle voidaan syöttää valtava määrä tietoa, jonka pohjalta sitä voi kouluttaa antamalla sille testidataa ja saada lopulta automaattisen tuloksen. Jos saaduissa tuloksissa havaitaan virheellisyyttä, koneoppimisalgoritmi kykenee muutokseen ja käyttää uutta tietoa parempaan päätöksentekoon tulevaisuudessa. Koneoppimista voidaan soveltaa monipuolisesti eri aloilla, kuten terveydenhuollossa, liikenteessä, taloustieteessä ja markkinoinnissa. (Chopra & Khurana 2023, luku 2; Markkinoinnin trendit 2023.)

Koneoppiminen toimii yleensä kolmessa erilaisessa vaiheessa: syötteen esikäsittelyssä, mallin oppimisessa ja saadun datan ennustamisessa/päätöksenteossa. Esikäsittelyssä raakadata puhdistetaan, normalisoidaan ja muunnetaan sellaiseen muotoon, jonka tekoälyn koneoppimisalgoritmi ymmärtää.

Tällä vaiheella varmistetaan, että tekoälyn saama data on sopivaa mallin oppimiseen. Kun tekoälyn koneoppimisalgoritmi luo mallin saamallaan datalla, se luo mallin, joka kykenee tekemään ennusteita tai haluttuja päätöksiä uudelle syötteelle. Tässä vaiheessa käsitelty raakadata jaetaan kahteen osaan: opetusdataksi ja testidataksi. Opetusdataa apuna käyttäen algoritmi oppii tunnistamaan säännönmukaisuudet ja yleistämään niitä testidatan arviointia varten. Koneoppimisalgoritmi tuottaa ennusteen ja tekee päätöksen oppimansa datan perusteella. Tekoälynsuorituskykyä voidaan arvioida vertaamalla sen ennustamia tuloksia testidatan tuloksiin. (Markkinoinnin trendit 2023; Colliander 2022, 10.)

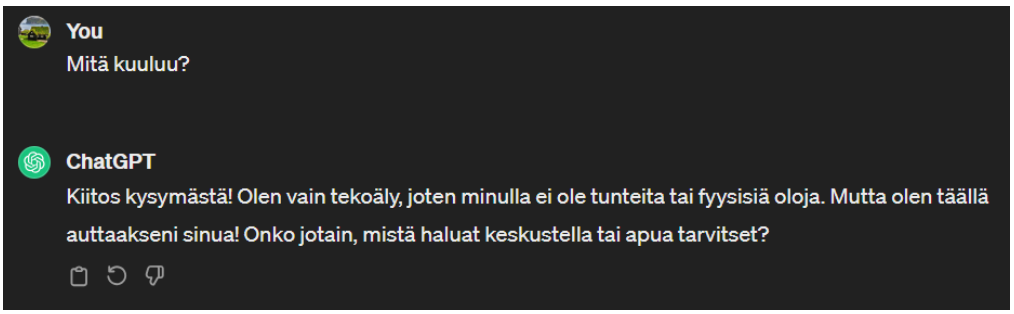
Ihmisten arjessa koneoppiminen näkyy monin eri tavoin, joista esimerkkinä voidaan mainita kohdistettu mainonta internetissä, jossa koneoppimista hyödynnetään laajalti. Kohdennetun mainonnan takana toimii tekoälyalgoritmi, joka kerää tietonsa käyttäjien IP-osoitteen perusteella. Tekoälyalgoritmi oppii tunnistamaan esimerkiksi kulutustottumuksia ja kykenee luomaan ennusteen siitä, millaisia mainoksia käyttäjälle kannattaa kohdistaa. (Klusaité 2023.)

2.3 ChatGPT

Yksi tunnetuimmista tekoälyistä on tekoäly- ja tutkimusyritys OpenAI:n kehittämä ChatGPT, joka julkaistiin yleiseen käyttöön marraskuussa 2022. ChatGPT:n GPT tulee sanoista Generative Pre-trained Transformer, jolla tarkoitetaan sitä, että se on koulutettu valtavalla määrällä luonnollisen kielen sisältöä. Tällä tarkoitetaan esimerkiksi internetistä löytyviä kirjoja, artikkeleita, verkkosivustoja ja sosiaalisen median sisältöä. (Microsoft julkaisuaika tuntematon.) Yksi ChatGPT:n merkittävimpinä pidettyinä ominaisuutena on sen kyky vuorovaikutukseen, jonka takia ohjelma on käyttäjäystävällinen ja suosittu (Koulutus 2023).

ChatGPT on siis valtava luonnollisen kielen käsittelytekniikka, joka kykenee vastaamaan sille esitettyihin kysymyksiin, käymään pitkiäkin keskusteluita hyödyntäen koneoppimista. Se on suunniteltu jäljittelemään mahdollisimman paljon ihmiskeskustelua. (Microsoft julkaisuaika tuntematon.). ChatGPT toimii keskustelubotin tai ns. chatbotin tavoin. Keskustelubotiksi kutsutaan ohjelmaa, joka tuottaa tekstiä reaaliajassa hyödyntäen tekoälyä. Se saadaan vaikuttamaan näin ihmismäiseltä ja sen kanssa voi keskustella lähes samalla lailla, kuin ihmisten kanssa. (Šimonélyté 2023). Salon (2023, luku 3.1) mukaan ChatGPT kykenee vastaamaan seurantakysymyksiin, myöntämään virheensä, haastamaan virheellisiä oletuksia ja hylkäämään sopimattomat pyynnöt.

ChatGPT on hyvin kehittynyt tekoäly ja on erinomainen apuväline erilaisten ideoiden keräämiseen, nopeiden tekstihahmotelmien luomiseen, tekstin kääntämiseen ja erilaisten ongelmien ratkaisuun laajan tietokantansa vuoksi (Koulutus 2023). Mitä enemmän tietoja annat sille käyttöön, sitä tarkempia vastauksia ChatGPT voi antaa (Microsoft julkaisuaika tuntematon).

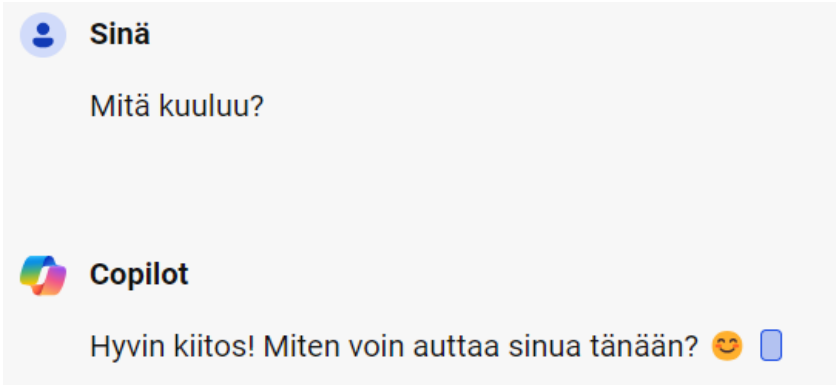


KUVA 1. Esimerkki keskustelu ChatGPT:n kanssa (Viertokangas 2024, CC BY-SA)

2.4 Microsoft Copilot

Microsoft Copilot on tekoälyä käyttävä digitaalinen avustaja, jonka tavoitteena on tarjota käyttäjille mukautettua apua erilaisiin tehtäviin ja toimintoihin (Microsoft julkaisuaika tuntematon). Microsoft Copilot, aiemmalta nimeltään Bing Chat Enterprise, on yksinkertaistettuna tekoälypohjainen chat-ohjelma verkossa, jota voi käyttää apuna, kun on tarve etsiä tietoa, suorittaa yksinkertainen tehtävä tai saada apua uusien ideoiden tutkimiseen. Copilot käyttää kehittynyttä luonnollisen kielen prosessointia ymmärtääkseen sille esitetyn kysymyksen ja kyetäkseen vastaamaan siihen. Copilotilta voi kysyä teoriassa mitä tahansa, yksinkertaisista faktoista aina monimutkaisiin kysymyksiin asti ja se antaa asiallisia ja tarkkoja vastauksia. (LAB julkaisuaika tuntematon.)

Copilot Microsoft 365 käyttää LLM-algoritmien yhdistelmää, eli tekoälyalgoritmia, joka käyttää syväoppimistekniikoita ja laajoja tietokantoja erilaisten sisältöjen ymmärtämiseen, yhteenvetojen luomiseen ja ennustamiseen. Nämä LLM:t sisältävät esimääritettyjä malleja, kuten generatiivisia koulutettuja muuntajia (GPT), kuten GPT-4, jotka on suunniteltu toimimaan erinomaisesti siltä vaadituilta tehtäviltä. (Microsoft 2024.)



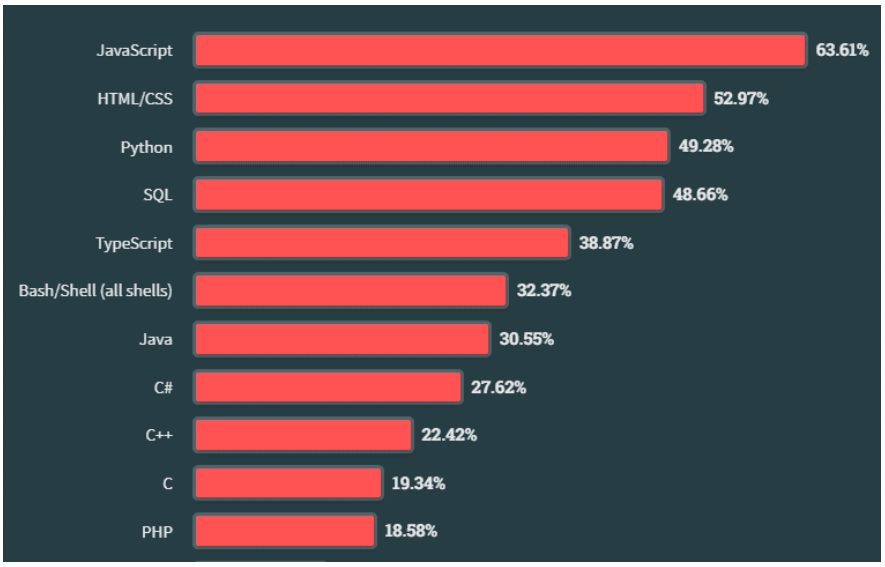
KUVA 2. Esimerkki keskustelu Copilotin kanssa (Viertokangas 2024, CC BY-SA)

3 PYTHON

Python on interaktiivinen olio-ohjelmointikieli, joka tukee moduuleja, poikkeuksia, dynaamista kirjoittamista, erittäin korkean tason dynaamisia tietotyyppejä ja luokkia. Python toimii monissa käyttöjärjestelmissä, kuten Unix-pohjaisissa (Linux, macOS) sekä Windowsissa, mikä tekee siitä erittäin monipuolisen vaihtoehdon erilaisiin kehitysympäristöihin. Vaikka Python on olio-ohjelmointikieli, se tukee useita ohjelmointiparadigmoja, sitä voidaan käyttää esimerkiksi laajennuskielenä sovelluksissa, jotka tarvitsevat ohjelmoitavan käyttöliittymän. (Python 2024.)

Pythonissa on valmiiksi sisäänrakennettu suuri standardikirjasto, joka kattaa esimerkiksi merkkijonojen käsittelyn, Internet-protokollat (HTTP, FTP, SMTP, XML-RPC, POP, IMAP), ohjelmistotekniikan (yksikkötestaus, profilointi, Python-koodin jäsentäminen) ja käyttöjärjestelmän rajapinnat (järjestelmäkutsut, tiedostojärjestelmät, TCP / IP-liitännät). (Python 2024.)

Toukokuussa 2023 yli 90 000 käyttäjää vastasi Stack Overflow-sivuston kyselyyn siitä, mitä ohjelmointikieltä he käyttivät, riippumatta siitä olivatko he alan ammattilaisia vai eivät (Stack Overflow 2023). Python oli ohjelmointikielenä kyselyssä kolmanneksi suosituin ohjelmointikieli (kuva 3).



KUVA 3. Kuvaleike ohjelmointikielten käytön yleisyydestä Stack Overflow:n teettämästä kyselystä (Stack Overflow 2023)

Pythonilla voidaan toteuttaa monimutkaisia statistisia laskelmia ja luoda koneoppimisalgoritmeja. Se sisältää myös runsaasti työkaluja datan visualisointiin, joka voisi selittää sen suosiota datatieteessä ja koneoppimisessa. Web-sovellusten kehittämiseen Python tarjoaa erilaisia kehityskehyksiä, kuten Django ja Flask, ja mahdollistaa myös dynaamisten ja skaalautuvien sovellusten rakentamisen. Pythonia käytetään laajasti automatisoinnissa ja skriptauksessa, kuten järjestelmänhallinnassa, tietokantakäsittelyssä ja tiedostojen käsittelyssä. Sen helppokäyttöisyys tekee siitä suositun valinnan myös ohjelmistotestauksessa ja prototyyppien luomisessa. (Dillemuth 2022.)

```python
def add(x,y):
    return x + y

if __name__ == '__main__':
    x = 5
    y = 10
    result = add(x,y)
    if result == 15:
        print('Test passed')
    else:
        print('Test failed')
        print(f'Expected: {x+y}, got: {result}')
```

KUVA 4. Python syntaksi (Viertokangas 2024, CC BY-SA)

3.1 Suositut python-kirjastot tekoälylle, kone- ja syväoppimisille

Scikit-Learn, joka tunnetaan myös nimellä sklearn, on koneoppimiskirjasto, joka tarjoaa valtavan valikoiman työkaluja erilaisiin ML-tehtäviin. Se rakennettiin useiden muiden suosittujen Python-kirjastojen päälle, mukaan lukien NumPy, SciPy ja Matplotlib, ja tarjoaa käyttäjälleen yhden käyttöliittymän ML-algoritmeille. (Developer 2023.)

Googlen kehittämä TensorFlow'n on avoimeen lähdekoodiin perustuva kone- ja syväoppimisen kirjasto, joka on erittäin monipuolinen ja helposti skaalautuva neuroverkoille ja syville neuroverkoille. Sen sisältämä laskentakaaviomalli sopii monimutkaisten neuroverkkojen määrittelyyn ja niiden kouluttamiseen. PyTorch on toinen laaja koneoppimisen kirjasto, joka on tunnettu dynaamisesta laskentakaaviostaan. Sen on kehittänyt Facebookin AI Research Lab (FAIR). (Developer 2023.)

OpenCV on Python-kirjasto, jota käytetään konenäkötehtäviin. Se sisältää valmiiksi suuren määrän työkaluja ja algoritmeja kuvan- ja videonkäsittelytehtäviin, minkä vuoksi sitä käytetään tekoäly- ja ML-ohjelmointiin, jossa tarvitaan visuaalisia elementtejä (kuten kasvojentunnistus). (Developer 2023.)

4 TODENNÄKÖISYYSLASKENTA

Todennäköisyyslaskennalla tarkoitetaan erilaisia matemaattisia menetelmiä, joilla pyritään ennustamaan tietynlaisten tapahtumien todennäköisyyttä. Sen luojina pidetään 1650-luvulla eläneitä matemaatikkoja, jotka kehittivät todennäköisyyslaskennan periaatteita pystyäkseen ymmärtämään ja sen myötä menestymään aikansa uhkapeleissä. (Henttonen, Peltomäki & Uusitalo 2006, luku 10; Matematiikkalehti Solmu 2002.)

Todennäköisyys mittaa kuinka yleistä tietyn ilmiön tapahtuminen on. Sen voi ilmaista joko sanallisesti tai numeroina, kuten reaalilukuina tai prosentteina. Termin määrittelyä hankaloittaa se, että todennäköisyys on ihmisen luoma omakohtainen käsite, jolle ei löydy luonnontieteellistä tai filosofista perustetta. Todennäköisyyslaskennan menetelmiä hyödynnetään monilla tieteenaloilla, kuten matematiikassa, luonnontieteessä, taloustieteessä, tietotekniikassa ja filosofiassa. Näistä menetelmistä käytetyimmät ovat klassisen todennäköisyyden määritelmä, tilastollinen todennäköisyystulkinta, bayesilainen todennäköisyysmääritelmä ja Kolmogorovin todennäköisyysteoria. (Matematiikkalehti Solmu 2002).

Stokastiset tapahtumina pidetään sellaisia tapahtumia, joiden lopputulos ole etukäteen määriteltävissä, vaan lopputulos riippuu aina jollain tavalla sattumasta. Esimerkiksi kolikon heitto on stokastinen tapahtuma: tiedetään lopputuloksen olevan joko kruuna tai klaava, muttei voida etukäteen ennustaa, kumpi puoli rahasta jää heitettäessä päällimmäiseksi. Vaikkei tiedetäkään tarkalleen, mikä heiton lopputulos on, voidaan kuitenkin määritellä ja laskea ennustettu todennäköisyys sille, että heiton tulos on klaava. (Nummenmaa, Holopainen & Pulkkinen 2019, luku 5.2.)

Deterministiset tapahtumat puolestaan ovat sellaisia, joiden lopputulos on ennakkoon tiedossa tai se voidaan määritellä laskennallisesti (Nummenmaa 2019, luku 5.2). Tuomenlehto, Holmlund, Huuskonen, Makkonen ja Surakka (2020, 301) mukaan tekniikan parissa on usein käytössä deterministinen ajattelutapa:

> kun kaikki vaikuttavat tekijät tunnetaan, lopputulos voidaan laskea yksikäsitteisesti etukäteen. Kaikkiin tilanteisiin tällainen tapa ajatella ei sovellu, sillä tarkasteltavan ilmiön lopputulos voi ennakolta ajatellen vaihdella. Lopputulokseen liittyy siis epävarmuutta. Todennäköisyyslaskenta antaa välineitä tällaisen epävarmuuden hallintaan.

Seuraavissa alaluvuissa esitellään yleisiä todennäköisyyslaskennan menetelmiä.

4.1 Kerto- ja yhteenlaskuperiaate

Jos valintatilanteessa on toisistaan riippumattomia, peräkkäisiksi tulkittavissa olevia vaiheita, kutsutaan tätä kertolaskuperiaatteeksi. Kaikkien erilaisten vaihtoehtojen lukumäärä lasketaan kertolaskulla. Jos vaiheessa 1 on m_1, erilaista vaihtoehtoa, vaiheessa 2 m_2 ja vaiheessa n m_n erilaista vaihtoehtoa, vaihtoehtoja on kaiken kaikkiaan $m_1 \cdot m_2 \cdot \ldots \cdot m_n$ kappaletta. (Tuomenlehto 2020, 301.)

Esimerkiksi veikkauksen jääkiekko kierroksella on 6 kohdetta, joissa jokaisessa on 3 eri vaihtoehtoa, 1, x, 2.

Koska jokaisessa kohteessa on 3 eri vaihtoehtoa, vaihtoehtojen kokonaismäärä on $3 \cdot 3 \cdot 3 \cdot 3 \cdot 3 \cdot 3 = 3^6 = 729$ kappaletta. On täytettävä 729 eri saraketta saadakseen varmasti kaikki 6 kohdetta oikein.

Yhteenlaskuperiaate tarkoittaa sitä, että jos valintatilanne jakautuu toisensa poissulkeviin vaihtoehtoihin, jotka eivät voi toteuta yhtä aikaa, lasketaan kaikkien erilaisten vaihtoehtojen lukumäärät yhteenlaskulla. Jos vaihtoehto 1 voi toteutua m_1 erilaisella tavalla, vaihtoehto 2 m_2 erilaisella tavalla ja vaihtoehto n m_n erilaisella tavalla, vaihtoehtoja on kaiken kaikkiaan $m_1 + m_2 + \cdots + m_n$ kappaletta. (Tuomenlehto 2020, 302.)

Esimerkiksi aakkostossa on 28 erilaista kirjainta. Korkeintaan kolmikirjaiminen sana voi olla joko yksi- kaksi- tai kolmikirjaiminen. Vaihtoehdot eivät voi toteutua samaan aikaan eli ne ovat toisensa poissulkevia.

Erilaisia kolmikirjaimisia sanoja on $28 \cdot 28 \cdot 28 = 28^3 = 21\,952$ kappaletta, erilaisia kaksikirjaimisia sanoja $28 \cdot 28 = 28^2 = 784$ kappaletta ja erilaisia yksikirjaimisia sanoja 28 kappaletta. Joten korkeintaan kolmikirjaimisia sanoja on $21\,952 + 784 + 28 = 22\,764$ kappaletta. (Tuomenlehto 2020, 302.)

4.2 Permutaatio ja variaatio

Kertomalla tai n-kertomalla tarkoitetaan n:n ensimmäisen positiivisen kokonaisluvun tuloa. Luvun n kertomalla tarkoitetaan tällöin lukua $n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 2 \cdot 1$, missä 0!=1. Merkintä $n!$ luetaan ”n kertoma”. Luvun kertoman avulla voidaan laskea, kuinka monella tapaa erilaisista alkioista koostuva joukko voidaan asettaa järjestykseen, jos kaikki alkiot otetaan mukaan. (Nummenmaa 2019, luku 5.)

Esimerkiksi jos on seitsemän eri ihmistä (Pekka, Pauli, Otto, Mikko, Kalle, Sandra, Vilma) ne voidaan laittaa $7! = 7 \cdot 6 \cdot 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1 =$ 5040:een eri järjestykseen. Eli ensimmäinen ihminen voidaan valita seitsemästä, seuraava kuudesta jne.

Usein perusjoukosta poimitaan ainoastaan osa alkioista, tällaisista järjestetyistä osajoukoista käytetään nimitystä variaatiot (Nummenmaa 2019, luku 5.1).

Tarkastellaan joukkoa, jossa on n kappaletta erilaisia alkioita. Otetaan näistä alkioista k kappaletta $(k < n)$ ja asetetaan ne jonoon. Jono voidaan muodostaa

$$n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot (n-k+1) = \frac{n!}{(n-k)!}$$

eri tavalla. (Tuomenlehto 2020, 303.)

Esimerkiksi jos ryhmässä on 7 henkilöä ja heistä tulisi valita autonkuljettaja ja tavarankantaja. Ensimmäinen ihminen voidaan valita 7:sta ja toinen 6:stä jäljelle jääneestä, eli eri vaihtoehtoja on:

$$7 \cdot 6 = \frac{(7 \cdot 6 \cdot 5 \ldots \cdot 1)}{(5 \cdot 4 \cdot 3 \ldots \cdot 1)} = \frac{7!}{(7-2)!} = \frac{7!}{5!} = \frac{7 \cdot 6 \cdot 5!}{5!} = 42\,.$$

4.3 Kombinaatio

n - alkioisesta joukosta muodostetaan osajoukkoja, joissa alkioiden lukumäärä kun $(k < n)$. Osajoukossa alkioiden järjestyksellä ei ole merkitystä, koska k kappaletta alkioita voidaan asettaa järjestykseen $k!$ eri tavalla, on vastaava variaatioiden lukumäärä $\frac{n!}{(n-k)!}$ jaettava luvulla $k!$.

Joukossa, jossa on n alkiota, voidaan siis muodostaa $\frac{n!}{k!\cdot(n-k)!}$ kappaletta k alkion osajoukkoa eli k-kombinaatiota. Lausekkeelle käytetään merkintää $\binom{n}{k}$, joka luetaan "n yli k". (Tuomenlehto 2020, 303.)

Esimerkiksi lotossa arvotaan 7 numeroa 40:stä. Erilaisia lottorivejä on $\binom{40}{7} = 18\,643\,560$ kappaletta.

4.4 Klassinen todennäköisyys

Klassisella todennäköisyydellä tarkoitetaan matemaatikko Jacob Bernoullin ja Pierre-Simon Laplacen teoriaa siitä, että erilaisten tapahtumien todennäköisyys liitetään tilanteisiin, joissa ajatellaan olevan äärellinen määrä keskenään yhtä todennäköisiä niin sanottuja alkeistapauksia. Tällöin on mielekästä määritellä tapahtuman A klassinen todennäköisyys kaavalla $P(A) = \frac{N(A)}{N}$, jossa N on kaikkien alkeistapausten lukumäärä ja $N(A)$ tapahtumalle A suotuisten alkeistapausten määrä. (Tuomenlehto 2020, 307; Matematiikkalehti Solmu 2002.)

Klassisen todennäköisyyden tärkeimpiä ominaisuuksia ovat:

- $P(\Omega) = 1$
- $0 \leq P(A) \leq 1$ $\quad A \subset \Omega$
- $P(A^c) = 1 - P(A)$ $\quad A^c \cup A = \Omega,\ A \cap A^c = \emptyset$
- $P(A \cup B) = P(A) + P(B) - P(A \cap B)$ $\quad A \subset \Omega, B \subset \Omega$
- $P(A \cup B) = P(A) + P(B)$ $\quad A \subset \Omega, B \subset \Omega\ ja\ A \cap B = \emptyset$
- $P(\bigcup_{i=1}^{n} A_i) = \sum_{i=1}^{n} P(A_i)$ $\quad A_i \subset \Omega, i = 1, \dots, n\ ja\ A_j \cap A_k = \emptyset, j \neq k,$

missä symbolilla Ω merkitään perusjoukkoa.

Esimerkki: sekoitetusta, tavallisesta 52 kortin korttipakasta nostetaan 2 korttia, missä korttien järjestyksellä ei ole väliä. Erilaisia, yhtä todennäköisiksi oletettavia vaihtoehtoja on $\binom{52}{2} = 1356$ kappaletta.

Kutsutaan tapahtumaa "nostetut kaksi korttia ovat herttoja" tapahtumaksi A. Tapahtumalle A suotuisia alkeistapauksia on $\binom{13}{2} = 78$ kappaletta. Tapahtuman A klassinen todennäköisyys on

$$P(A) = \frac{78}{1356} = \frac{1}{17}.$$

4.5 Tilastollinen todennäköisyys

Tilastollisen tutkimuksen peruslähtökohtana on määritellystä populaatiosta poimittu otos, jonka perusteella koetetaan tehdä populaatiota koskevia päätelmiä. Koska otos ei vastaa koskaan täysin koko populaatiota, tilastollisten menetelmien avulla pyritään määrittelemään, kuinka todennäköisesti otoksessa havaitut ilmiöt esiintyvät myös populaatiossa. Näitä päätelmiä varten tarvitaan kahdenlaisia työkaluja. Ensinnäkin tarvitaan kombinatorisia menetelmiä, joiden avulla voidaan tarkastella erilaisten joukkojen muodostumista ja niiden ominaisuuksia, jotka liittyvät keskeisesti otoksen ja populaation määrittämiseen. Toiseksi tarvitaan todennäköisyyslaskentaan liittyviä menetelmiä, joilla voidaan tarkastella erilaisiin tapahtumiin liittyviä todennäköisyyksiä ja niiden laskemista. (Nummenmaa 2019, luku 5.)

Todennäköisyyksiä voidaan laskea myös kokeellisten havaintojen perusteella. Yksinkertainen esimerkki tällaisesta kokeellisesta eli empiirisestä todennäköisyydestä on tilastollinen todennäköisyys. Siinä tapahtuman todennäköisyys on sama kuin vastaavan tilastomuuttujan arvon suhteellinen frekvenssi. (Kontkanen, Lehtonen & Luosto 2012, 71.)

Tämä siis tarkoittaa sitä, että toistaessa satunnaisilmiötä äärettömän monta kertaa, tapahtuman A osuus kaikista äärettömistä toistoista on tapahtuman A tilastollinen todennäköisyys. Käytännössä mitään satunnaisilmiötä ei voida toistaa äärettömän monta kertaa, mutta riittävän suurella toisto määrällä saadaan usein jo melko tarkka arvio tapahtuman tilastollisesta todennäköisyydestä. (Nummenmaa 2020, luku 5.2.)

Tapahtuman A todennäköisyys on

$$P(A) = \frac{f_A}{n},$$

missä f_A on tapahtuman A frekvenssi ja n havaintojen lukumäärä (Kontkanen 2012, 71).

Tulos	f	$p = \frac{f}{n}$
0	6	0,12
1	2	0,04
2	2	0,04
3	4	0,08
4	3	0,06
5	5	0,1
6	9	0,18
7	4	0,08
8	11	0,22
9	3	0,06
10	1	0,02
	50	1

KUVA 5. Pisteiden suhteelliset frekvenssit (Viertokangas 2024, CC BY-SA)

Esimerkiksi Mikko heitti tikkaa 50 kertaa ja kokosi tulokset taulukkoon (kuva 5). Voidaan laskea todennäköisyys sille, että heiton tulos on parillinen. Parillisten tulosten yhteenlaskettu frekvenssi on 6+2+3+9+11+1=32, joten

$$P(parillinen) = \frac{f_{parillinen}}{n} = \frac{32}{50} = \frac{16}{25} = 0{,}64\,.$$

(Kontkanen 2012, 71.)

4.6 Ehdollinen todennäköisyys

Ehdollinen todennäköisyys $P(B|A)$ tarkoittaa todennäköisyyttä sille, että tapahtuu tapahtuma tapahtuman B ehdolla, olettaen, että tapahtuma A on tapahtunut. Tällöin siis tapahtumat A ja B oletetaan toisistaan riippuviksi siten, että tapahtuman A lopputulos vaikuttaa jollain tavalla tapahtuman B todennäköisyyteen. (Nummenmaa 2020, luku 5.3.)

Jos jotkin satunnaisilmiön lopputulokset voidaan sulkea pois, päädytään ehdollisen todennäköisyyden käsitteeseen. Tapahtuman B ehdollinen todennäköisyys ehdolla A määritellään kaavalla

$$P(B|A) = \frac{P(A \cap B)}{P(A)}\,.$$

Tällöin tiedetään tai oletetaan tapauksen A tapahtuneen ja lasketaan tapauksen B todennäköisyys tämän tiedon tai oletuksen auttamana. Kertolaskulla määritelmästä seuraa kaava

$$P(A \cap B) = P(A) \cdot P(B|A)\,.$$

Tuomenlehdon (2020, 313-315) mukaan jos tunnetaan ositteiden A_i todennäköisyydet ja tapahtuman B ehdolliset todennäköisyydet ositteista A_i, tapauksen B todennäköisyys voidaan laskea kokonaistodennäköisyytenä

$$P(B) = \sum_{i=1}^{n} P(A_i) \cdot P(B|A_i)\,.$$

Esimerkiksi, kolmella koneella tuotetaan pulttia X. Koneiden prosenttiosuudet pulttien tuotannosta ovat 15 %, 30 % ja 55 %. Merkitään tapahtumaa A_i:llä tapahtumaan ”yksittäinen satunnaisesti valittu pultti on koneen i tuottama”. Tällöin $P(A_1) = 0{,}15; P(A_2) = 0{,}3; P(A_3) = 0{,}55$. Virheellisten pulttien osuus tuotetuista pulteista eri koneilla on puolestaan 2 %, 5 % ja 1 %. Merkitään V:llä tapahtumaa ”yksittäinen satunnaisesti valittu pultti on virheellinen”. Siispä $P(V|A_1) = 0{,}02; P(V|A_2) = 0{,}05; P(V|A_3) = 0{,}01$. Todennäköisyys sille, että satunnaisesti valittu pultti on virheellinen, voidaan laskea kokonaistodennäköisyytenä

$$P(V) = \sum_{i=1}^{3} P(A_i) \cdot P(V|A_i) = 0{,}15 \cdot 0{,}02 + 0{,}30 \cdot 0{,}05 + 0{,}55 \cdot 0{,}01 = 0{,}0235\,.$$

Voidaan siis todeta, että kaikista pulteista 2,35 % on virheellisiä. (Tuomenlehto 2020, 315.)

5 NAIIVI BAYES-LUOKITIN

Naiivi Bayes-luokitin on Bayesin lauseeseen (kaava 1) perustuva tilastollinen luokittelutekniikka, joka on nopea, tarkka ja luotettava algoritmi. Naiivi Bayes-luokittelija olettaa, että tietyn ominaisuuden vaikutus luokassa on riippumaton muista ominaisuuksista. Esimerkiksi lainanhakija on lainanmyöntäjän näkökulmasta haluttava tai ei, riippuen hänen tuloistaan, aiemmasta laina- ja tapahtumahistoriastaan, iästään ja sijainnistaan. Vaikka nämä ominaisuudet ovat käytännössä toisistaan riippuvaisia, näitä ominaisuuksia pidetään silti itsenäisinä luokkina. Oletus siitä, että tietty ominaisuus on muista riippumaton yksinkertaistaa laskentaa, ja siksi sitä pidetään naiivina. Tällaista oletusta voidaan kutsua luokan ehdolliseksi riippumattomuudeksi. (Datacamp 2023.)

Ehdollisen todennäköisyyden mukaisesti tapahtuman A todennäköisyys ehdolla B on

$$P(A|B) = \frac{P(A \cap B)}{P(B)}.$$

Tapahtuman B todennäköisyys ehdolla A on

$$P(B|A) = \frac{P(A \cap B)}{P(A)}.$$

Joista saadaan yhtälö

$$P(A|B)P(B) = P(A \cap B) = P(B|A)P(A).$$

Jakamalla yhtälön molemmat puolet tekijällä $P(B)$, saadaan Bayesin kaava

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)}, \qquad (1)$$

missä

- $P(A|B)$ on A:n todennäköisyys ehdolla B
- $P(B)$ on B:n priori-todennäköisyys
- $P(A)$ on reunatodennäköisyys, joka kuvaa tehtyjen havaintojen todennäköisyyttä
- $P(B|A)$ on B:n todennäköisyys ehdolla A ja tätä kutsutaan posterioritodennäköisyydeksi

(Wikipedia 2020.)

Lasketaan edellisessä kappaleessa esitetystä esimerkistä todennäköisyys sille, että virheellinen pultti on koneen 3 tuottama, missä

- A_3 on tapahtuma, että pultti on tuotettu koneella 3
- $P(A_3)$ on todennäköisyys, että pultti on tuotettu koneella 3 (0,55)
- $P(V|A_3)$ on todennäköisyys sille, että virheellinen pultti on tuotettu koneella 3 (0,01)
- $P(V)$ on kokonaistodennäköisyys sille, että pultti on virheellinen (0,0235)

käyttämällä Bayesin kaavaa

$$P(A_3|V) = \frac{P(A_3) \cdot P(V|A_3)}{P(V)} = \frac{0{,}55 \cdot 0{,}01}{0{,}0235} = \frac{0{,}0055}{0{,}0235} = 0{,}2340,$$

joten todennäköisyys sille, että virheellinen pultti on koneen 3 tuottama on 23,40 %.

Naiivi Bayes-luokitin, on yksi tehokkaimmista oppimisalgoritmeistä koneoppimiseen (Zhang, The Optimality of Naive Bayes 2004, 1). Nämä Bayesin lauseeseen perustuvat luokittimet eivät ole vain yksittäinen algoritmi vaan eräänlainen algoritmiperhe, jossa niillä kaikilla on yhteinen periaate, eli jokainen luokiteltava ominaisuuspari on toisistaan riippumaton. Toisin sanoen kullekin muuttujalle tarvitaan vain yksi todennäköisyys, mikä puolestaan helpottaa tarvittavan mallin laskentaa. (Scikit-learn julkaisuaika tuntematon.)

Naiivia Bayes-algoritmia käytetään paljon esimerkiksi erilaisten tekstien ja asiakirjojen luokittelussa ja roskapostin suodatuksessa, sillä ne vaativat pienen määrän koulutusdataa tarvittavien parametrien arvioimiseksi (Scikit-learn julkaisuaika tuntematon).

6 KEHITTÄMISTYÖ JA TULOKSET

Itse kehittämistyön toteutus ajoittui maalis- huhtikuun 2024 väliselle ajalle. Aloitin kehittämistyöni tutustumalla yllä olevan teoriapohjan käsitteisiin, jotta pystyin ymmärtämään, mihin valitsemani menetelmät perustuvat ja kuinka niitä pystyi parhaiten hyödyntämään kehittämistyössäni. Opinnäytetyössäni käytin Visual Studio Code tekstieditoria, python-ohjelmointikieltä sekä hyödynsin pythonin valmiita kirjastoja kuten Scikit-Learn.

Alkuperäinen suunnitelma kehittämistyön toimeksiantajan puolelta oli valita toteutustapa joko Bayes-luokittimen tai pienen neuroverkon väliltä. Nopeasti kuitenkin ilmeni, että pienen neuroverkon rakentaminen kehittämistyötäni varten olisi ollut prosessina liian hankala, aikaa vievä ja lisäksi opetusmateriaali oli neuroverkon luomiseen liian pieni, joten päädyin käyttämään Bayes-luokitinta, mikä löytyi valmiina Scikit-Learn-kirjastosta, joten sitä ei tarvinnut tehdä itse.

Sain kehittämistyön opetusmateriaalin työn tilaajalta ja se sisälsi Savonia Ammattikorkeakoulun käytössä olevien tietotekniikan kurssien materiaalia. Materiaali itsessään koostui kolmesta ennakkoon valitusta tehtävästä, joista koostui yhteensä 64 kappaletta opiskelijoiden tuottamia vastauksia, jotka olivat toteutettu C# ohjelmointikielellä. Henkilö- ja muut tunnistetiedot olivat saamastani materiaalista jo poistettu, joten opiskelijoiden anonymiteetti pysyi, eikä erillisiä lupia tehtävien käyttöön tarvittu.

Opinnäytetyöhöni valitsin kaksi tunnetuinta tekoälyä eli Microsoft Copilot ja ChatGPT. Tekoälyt tuottivat yhteensä 105 vastausta ja ne saivat samanlaisen tehtävänannon, kuin opiskelijat. Eli syötin samat tehtävänannot tekoälyille, otin niiden antamat vastaukset talteen, jonka jälkeen tekoälyjen tuottamille vastauksille suoritettiin yksikkötestaus. Jos tekoälyn antama vastaus ei suoriutunut testistä onnistuneesti, pyysin tekoälyä korjaamaan sen antamaa alkuperäistä koodia sen mukaan, missä virhe ilmeni. Testiohjelmat rakensin itse python-ohjelmointikielellä, jotka toteutin tehtäväkohtaisen tehtävänannon mukaisesti.

Yksikkötestauksen tarkoituksena oli antaa osviittaa siitä, miten tekoäly tuotti pyydetyn koodin verrattuna opiskelijoihin, mutta tässä tapauksessa haasteeksi muodostui se, ettei opiskelijoilla ollut mahdollisuutta testata omaa koodia testiohjelmalla, joten suurin osa myös näistä epäonnistuivat. Halusin myös selvittää, voisiko tekoälyn tuottamaa dataa erottaa ihmisen tekemästä silmämääräisesti.

Testivaiheen lopuksi minulla oli 64 kpl opiskelijoiden vastauksia ja 105 kpl tekoälyn tuottamaa vastausta omissa hakemistoissaan, jotka toimivat opetusaineistona Bayes-luokittimen kouluttamiseen. Ennen Bayes-luokittimen koulutusta esikäsittelin datan poistamalla siitä muun muassa yksi- ja moniriviset kommentit (kuva 6).

```
with open(file, 'r',encoding='utf-8') as fr:
        line:str = fr.read()
        line = re.sub(r'/\*(.|\n)*?\*/', '', line)
        line = re.sub(r'//.*$', '', line, flags=re.MULTILINE)
        line = re.sub(r'^//.*$', '', line, flags=re.MULTILINE)
        line = re.sub(r'\n{2,}', '\n', line, flags=re.MULTILINE)
        line = line.strip()
        line = re.sub(r'^[ \t]+','', line, flags=re.MULTILINE)
        line = re.sub(r' {2,}', ' ', line, flags=re.MULTILINE)
        line = re.sub(r'[{}]', '', line, flags=re.MULTILINE)
        line = re.sub(r'^.*\bclass\b.*$', '', line, flags=re.MULTILINE)
        line = re.sub(r'^.*\bstatic void Main\b.*$', '', line, flags=re.MULTILINE)
        line = line.replace('\ufeff','')

hlist = "\n".join([l for l in line.split("\n") if l.strip()])
flist = hlist.split('\n')
return flist
```

KUVA 6. Datan esikäsittely (Viertokangas 2024, CC BY-SA)

6.1 Bayes – luokittimen koulutus

Bayes-luokittimen koulutusta varten käytin apuna `CountVectorizer`-työkalua, jota käytetään yleisesti tekstin käsittelyyn ja vektorointiin koneoppimisalgoritmeja varten. Sen tarkoituksena oli muuntaa tekstikokoelman matriisiksi, jossa jokainen rivi vastaa yhtä dokumenttia ja jokainen sarake edustaa yhtä sanan tai sanajoukon esiintymää. `CountVectorizer`-työkalua hyödyntäen loin sanaluettelon, jonka avulla sain muutettua syöttämäni sanat vektoriksi, joka vastasi tietyn sanan esiintymisen lukumäärää. Esimerkiksi, jos vastauksissa esiintyi sana "ahven" kolme kertaa, "kuha" kerran ja "kiiski" kaksi kertaa, vastaava vektori olisi [3, 1, 2].

Lopulta yhdistin nämä vektorit luodakseni matriisin, jolloin luokitin pystyi käsittelemään tekstimuotoista dataa numeerisena muotona ja oppia piirteitä, jotka helpottivat sitä tunnistamaan ja ennustamaan tiettyjä asioita teksteistä, kuten luokkia tai aiheita.

```
vectorized = CountVectorizer()
X = vectorized.fit_transform(features)
X_train, X_test, y_train, y_test = train_test_split(X, label, test_size=0.2,random_state=40)
```

KUVA 7. Opetusaineiston jakaminen (Viertokangas 2024, CC BY-SA)

Luokittelussa oli kaksi vaihetta: oppimisvaihe ja arviointivaihe. Oppimisvaiheessa luokittelija kouluttaa mallinsa annetulle aineistolle ja arviointivaiheessa testaa luokittelijan suorituskykyä. Suorituskykyä arvioidaan eri parametrien, kuten tarkkuuden, virheen, tarkkuuden ja muistamisen, perusteella. (Datacamp 2023.) Jaoin opetusmateriaalin kahteen eri osaan: koulutusaineistoon (80 %) ja testiaineistoon (20 %). Koulutusaineistoa käytin mallin kouluttumiseen ja testiaineistoa käytin arvioimaan koulutetun mallin suorituskykyä. Tässä käytin apuna valmista Skicit-Learn-kirjaston ’train_test_split’ metodia, joka on yksi yleisimmistä menetelmistä koneoppimisen mallien kouluttamisessa ja testaamisessa (kuva 7). Tämä mahdollisti mallin kouluttamisen riittävällä datamäärällä ja sen suorituskyvyn arvioimisen uudella, ennennäkemättömällä datalla. Perusajatuksena oli, että koulutusdata opettaa mallin "oppimaan" datasta ja testidata tarjoaa objektiivisen tavan mitata sen suorituskykyä, kun sitä käytettiin uusiin, ennalta näkemättömiin tietoihin. Lisäksi käytin hyperparametrien (alpha arvon) määrittelyssä apuna valmista Skicit-Learn kirjastoa (GridSearchCV) eli kun alphan arvo on pieni se

lisää painoarvoa koulutusaineiston luokkien jakautumiselle, jos taas alpha on suuri se tasoittaa luokkien jakautumista. Eli alpha parametri auttaa estämään nollahavainto-ongelmia ja parantaa mallin yleistämistä uusiin havaintoihin.

Tässä vaiheessa luokitin laski jokaisen sanan esiintymistiheyden kussakin luokassa. Eli kyseessä on yleisesti tunnettu 'label' luokka, missä oli ihmisen sekä tekoälyn tuottamat koodit samassa: nolla (0) tarkoitti ihmisen tuottamaa ja ykkönen (1) tekoälyn tuottamaa koodia. Toisin sanoen luokitin, määritti saamansa informaation, tässä tapauksessa tietyt sanat, joko nollaksi tai ykköseksi tapauskohtaisesti. Nämä määrittelemäni sanat (features) toimivat ominaisuuksina ja muuttujina, jotka kuvaavat kunkin havainnon ominaisuuksia. Tässä tapauksessa muuttujia olivat tiedostosta luetut koodirivit (esimerkiksi int [] taulukko = new int [8]). Esimerkiksi tekoälyn tuottama koodi määräytyi tiedostopolun perusteella eli jos polussa oli ”copilot” tai ”chatgpt”.

6.2 Uusien havaintojen tunnistaminen

Uusien havaintojen tunnistamisessa käytin samoja kirjastoja, työkaluja ja esikäsittelyä datalle, mitä käytin koulutusvaiheessa. Bayes-luokitin arvioi aluksi uusien havaintojen piirteitä. Nämä piirteet ovat niitä ominaisuuksia tai muuttujia, jotka on määritetty koulutusdatassa ja joita käytetään havainnon luokittelun perusteena. Sen jälkeen luokitin käytti koulutettuja malleja ja sääntöjä arvioidakseen, mihin luokkaan uusi havainto todennäköisemmin kuului. Luokitin teki tämän laskemalla havainnon todennäköisyyden jokaiselle mahdolliselle luokalle annettujen piirteiden perusteella. Tämän jälkeen luokitin teki lopullisen päätöksen ennustamalla havainnon luokkaan, joka on saanut korkeimman todennäköisyyden. Lopuksi luokitin antoi ennusteen tai luokituksen uudelle havainnolle sen perusteella, miten se on arvioinut havainnon piirteitä ja soveltanut oppimiaan malleja ja viimeisenä uusien havaintojen tulokset kirjoitettiin tiedostoon.

6.3 Tulokset

Lopputulemana Bayes-luokitin pystyi tunnistamaan suhteellisen tarkasti, mitkä tehtävät on tehty pelkästään tekoälyä hyödyntäen. Tehtävät t1, t2, t8, t9, t15 ja t16 on toteutettu tekoälyä hyödyntäen (kuva 8). Toisaalta tehtävät t18 ja t19, jotka on toteutettu ihmisten toimesta, luokitin luokitteli nämä virheellisesti tekoälyn tuottamiksi. Tämä osoittaa, että vaikka luokitin näyttää lupaavalta, siinä on myös rajoituksia ja kehitystarpeita. Testasin Bayes-luokittimen tarkkuutta laittamalla kolme eri tapausta A, B ja C kaikki yhteen ja tutkin, miten se vaikutti tarkkuuteen (kuva 8). Kun tarkastellaan tapauksien A, B ja C yhdessä, luokittelun tarkkuus heikkenee, mikä korostaa tämän kehitystarvetta.

Luokittimen kyky erottaa tekoälyn ja ihmisten tuottama koodi on ensiaskel kohti luotettavien tunnistusjärjestelmien kehittämistä. Esimerkiksi tehtävien t1, t2, t8, t9, t15 ja t16 tarkka tunnistaminen osoittaa luokittimen potentiaalin tunnistaa tekoälylle tyypillisiä piirteitä. Nämä tehtävät sisälsivät piirteitä, kuten tiettyjä koodaustyylejä tai usein käytettyjä funktioita, jotka ovat yleisiä tekoälyn tuottamassa sisällössä.

Toisaalta tehtävien t18 ja t19 virheellinen luokittelu vahvistaa sen, että luokitin ei ole virheetön. Nämä ihmisten tekemät tehtävät luokiteltiin tekoälyn tuottamiksi todennäköisesti siksi, koska niissä oli koodauskaavoja tai -rakenteita, jotka luokitin yhdisti tekoälyyn. Tämä virheellinen luokittelu korostaa tarvetta parantaa luokittimen algoritmia ja koulutusdataa. Koulutusdatan monipuolistaminen

ja lisääminen voisi tulevaisuudessa auttaa luokitinta ymmärtämään paremmin ihmisten ja tekoälyn tuottaman koodin hienovaraisia eroja.

	Tapaus A	Tapaukset A,B ja C	
t1	81,08 %	89,19 %	AI
t2	69,57 %	65,22 %	AI
t3	16,67 %	26,67 %	
t4	5,26 %	18,42 %	
t5	15,79 %	21,05 %	
t6	22,86 %	28,57 %	
t7	15,62 %	12,50 %	
	Tapaus B		
t8	88,46 %	92,31 %	AI
t9	86,36 %	86,36 %	AI
t10	62,96 %	51,85 %	
t11	28,12 %	43,75 %	
t12	31,25 %	46,88 %	
t13	21,88 %	15,62 %	
t14	42,31 %	53,85 %	
	Tapaus C		
t15	60,00 %	66,67 %	AI
t16	92,31 %	92,31 %	AI
t17	52,63 %	47,37 %	
t18	92,86 %	85,71 %	
t19	88,89 %	83,33 %	
t20	38,46 %	38,46 %	

KUVA 8. Uusien havaintojen tulokset (Viertokangas 2024, CC BY-SA)

7 YHTEENVETO JA POHDINTA

Tämä kehittämistyö oli hyödyllinen oppimiskokemus. Aihe oli alusta alkaen mielenkiintoinen ja haluan myös tulevaisuudessa työskennellä sellaisissa projekteissa missä hyödynnetään tekoälyä. Henkilökohtaista kokemusta esimerkiksi Bayes-luokittimen toiminnasta ei alun perin ollut, joten tämä opinnäytetyön tekeminen tuntui ajoittain haastavalle mutta kykenin ratkaisemaan eteen tulleet ongelmatapaukset. Pidän tätä kehittämistyötä alalle hyödyllisenä, sillä sen jatkokehitysmahdollisuudet ovat laajat ja ovat mahdollisia toteuttaa tulevaisuudessa. Opinnäytetyö on saavutettava koska se on kielellisesti selkeä ja helposti ymmärrettävissä sekä kuvissa on selkeät otsikot ja ne on selitetty tekstissä.

Eettisesti tarkasteltuna tekoälyn hyödyntäminen tässä opinnäytetyössä oli vastuullista ja läpinäkyvää, sillä tähän kehittämistyöhön peilaten, sillä oli selkeä tarkoitus ja ilman sitä se ei olisi ollut mahdollista toteuttaa. Tekoälyä käytettiin luokittimen opetusaineiston hankkimiseen, jolloin saatiin haluttu vertailukelpoinen data, kielentarkistukseen ja tiivistelmän kääntämiseen englanniksi. Koin, että tässä opinnäytetyössä tekoälyn käyttäminen noudatti eettisiä periaatteita, huomioin tietosuojan sekä osasin arvioida ja tulkita tekoälyn tuloksia kriittisesti.

Kaiken kaikkiaan, vaikka nykyinen Bayes-luokittimen toteutus tarjoaa lupaavan ”proof of conceptin”, se myös selkiyttää jatkuvan kehityksen tarpeellisuutta. Luokittimen suorituskyky tekoälyn ja ihmisten tuottamien tehtävien erottelussa on rohkaiseva, mutta samalla se muistuttaa siitä, että tarvitaan vielä paljon lisää tutkimusta ja kehitystyötä. Tämä opinnäytetyö osoittaa onnistuneesti, että on mahdollista luoda järjestelmä, joka kykenee analysoimaan ja tunnistamaan tekoälyn tuottamaa dataa ja se luo mahdollisuuden jatkokehittämiselle.

Työssä on käytetty seuraavasti tekoälyä:

ChatGPT 2024. OpenAI. GPT-3.5. Käytetty kielentarkastukseen sekä tiivistelmän kääntämisessä englanniksi, toukokuu 2024. https://chatgpt.com

LÄHTEET

Chopra, Deepti & Khurana, Roopal 2023. Introduction to Machine Learning with Python. E-kirja. Doral: Bentham Science Publishers. Viitattu 02.04.2024.

Colliander, Jeremias 2022. Koneoppiminen ja massadata terveydenhuollossa. Kandidaatintutkielma. Tietojärjestelmätiede. Jyväskylän yliopisto. https://jyx.jyu.fi/bitstream/handle/123456789/81795/URN%3aNBN%3afi%3ajyu-202206163404.pdf?sequence=1. Viitattu 02.04.2024.

Datacamp 2023. Naive Byes Classification Tutorial using Scikit-learn. Verkkojulkaisu. Päivitetty 3/2023. https://www.datacamp.com/tutorial/naive-bayes-scikit-learn. Viitattu 04.04.2024

Developer 2023. 7 best python libraries for machine learning and AI. Verkkojulkaisu. Päivitetty 22.09.2023. https://www.developer.com/languages/python/python-libraries-for-machine-learning-ai/. Viitattu 07.04.2024.

Dillemuth, Jon 2022. Mihin python ohjelmointikieltä käytetään. Tieturi blogi. 5.12.2022. https://www.tieturi.fi/blogi/mihin-python-ohjelmointikielta-kaytetaan/. Viitattu 31.3.2024.

Euroopan parlamentti 2023. Mitä on tekoäly ja mihin sitä käytetään. Verkkojulkaisu. Päivitetty 20.06.2023. https://www.europarl.europa.eu/topics/fi/article/20200827STO85804/mita-tekoaly-on-ja-mihin-sita-kaytetaan. Viitattu 25.03.2024.

Henttonen, Juhani, Peltomäki, Jaana & Uusitalo, Seppo 2006. Tekniikan matematiikka 2. E-kirja. Helsinki: Edita. Viitattu 18.04.2024.

Kananen, Heidi & Puolitaival, Harri 2019. Tekoäly. Bisneksen uudet työkalut. E-kirja. Helsinki: Alma Talent. Viitattu 02.04.2024.

Klusaité, Laura 2023. Mitä on koneoppiminen. NordVPN blogi. 20.03.2023. https://nordvpn.com/fi/blog/koneoppiminen/. Viitattu 25.3.2024.

Kontkanen, Pekka, Lehtonen, Jukka & Luosto, Kerkko 2012. Pyramidi. Lukion pitkä matematiikka. 6, todennäköisyys ja tilastot. 1.painos. Helsinki: SanomaPro. Viitattu 07.04.2024.

Koulutus 2023. Mikä on ChatGPT ja mitä siitä pitäisi tietää. Verkkojulkaisu. Päivitetty 06.07.2023. https://www.koulutus.fi/artikkelit/mika-on-chatgpt-ja-mita-siita-pitaisi-tietaa-23286. Viitattu 25.3.2024.

LAB julkaisuaika tuntematon. Microsoft copilot. Verkkojulkaisu. https://elab.lab.fi/fi/it-ohjeet-ja-opiskelun-tyokalut/opiskelun-jarjestelmat/microsoft-copilot. Viitattu 02.04.2024.

Markkinoinnin trendit 2023. Mitä on koneoppiminen. Verkkojulkaisu. Päivitetty 23.05.2023. https://markkinoinnintrendit.fi/koneoppiminen/. Viitattu 25.03.2024.

Matematiikkalehti Solmu 2002. Sattuman matematiikkaa 1. Klassinen todennäköisyys. Verkkojulkaisu. Päivitetty 11.06.2002. https://matematiikkalehtisolmu.fi/2002/2/sattuma/. Viitattu 18.04.2024.

Microsoft julkaisuaika tuntematon. ChatGPT vs. Microsoft Copilot: What´s the difference. Verkkojulkaisu. https://support.microsoft.com/en-us/topic/chatgpt-vs-microsoft-copilot-what-s-the-difference-8fdec864-72b1-46e1-afcb-8c12280d712f. Viitattu 25.03.2024.

Microsoft 2024. Microsoft Copilot for Microsoft 365 -yleiskatsaus. Verkkojulkaisu. Päivitetty 03.02.2024. https://learn.microsoft.com/fi-fi/microsoft-365-copilot/microsoft-365-copilot-overview/. Viitattu 01.04.2024.

Nummenmaa, Lauri, Holopainen, Martti & Pulkkinen Pekka 2019. Tilastollisten menetelmien perusteet. E-kirja. Helsinki: SanomaPro. Viitattu 03.04.2024.

Porkamaa, Tuomas 2020. Naiivin Bayesin luokittelijan ilmaisukyvyn kasvattaminen attribuuttijoukkoa kasvattamalla. Kandidaatintutkielma. Tietojenkäsittelytieteiden tutkinto-ohjelma. Tampereen yliopisto. https://trepo.tuni.fi/bitstream/handle/10024/123278/PorkamaaTuomas.pdf?sequence=2. Viitattu 02.04.2024

Python 2024. General Python FAQ. Verkkojulkaisu. Päivitetty 03.04.2024. https://docs.python.org/3/faq/general.html#general-information.Viitattu 04.04.2024.

Salo, Immo 2023. Luova tekoäly mullistaa kaiken. ChatGPT näyttää tietä. E-kirja. Helsinki: Kauppakamari. Viitattu 02.04.2024.

Scikit Learn julkaisuaika tuntematon. Naive Bayes. Verkkojulkaisu. https://scikit-learn.org/stable/modules/naive_bayes.html. Viitattu 30.04.2024.

Šimoélyté, Miglé 2023. Mikä on chatbot. NordVPN blogi. 17.04.2023. https://nordvpn.com/fi/blog/mika-on-chatbot/. Viitattu 01.4.2024.

Stack Overflow 2023. 2023 Developer Survey. Verkkojulkaisu. Päivitetty 13.06.2023. https://survey.stackoverflow.co/2023/#technology-most-popular-technologies. Viitattu 08.04.2024.

Toivonen, Hannu 2023. Mitä tekoäly on. 100 kysymystä ja vastausta. E-kirja. Helsinki: Kustannusosakeyhtiö Teos. Viitattu 25.3.2024.

Tuomenlehto, Ari, Holmlund, Eero, Huuskonen, Maija, Makkonen, Heikki & Surakka, Jarkko 2020. Insinöörin Matematiikka. 6. painos. Helsinki: Edita. Viitattu 03.04.2024.

Wikipedia 2020. Bayesin teoreema. Verkkojulkaisu. Päivitetty 25.05.2020. https://fi.wikipedia.org/wiki/Bayesin_teoreema. Viitattu 07.04.2024.

Zhang, Harry 2004. The Optimality of Naïve Bayes. Pdf-tiedosto. Julkaistu 2004. https://courses.ischool.berkeley.edu/i290-dm/s11/SECURE/Optimality_of_Naive_Bayes.pdf. Viitattu 30.04.2024.