SAVONIA

Tekoälyalustat (3op)
Kurssin esitysdiat 2/2

Tietotekniikan lehtori Ville Berg

ville.berg@savonia.fi

puh. 044 785 6054

Näissä dioissa käydään läpi

- Konenäkö ja kuvantunnistus
- Analytiikka ja datan analysointi
- Puheentunnistus ja ääniavustajat
- Tekoälyagentit (jos jää aikaa)

Konenäkö

- Konenäkö on tekoälyn osa-alue, joka pyrkii ymmärtämään ja tulkitsemaan kuvia ja videoita samalla tavalla kuin ihminen. Sen tavoitteena on esimerkiksi:
  - Havaita kohteita (esim. autot, ihmiset, eläimet)
  - Jäsentää kohtauksia (esim. missä järjestyksessä asiat ovat)
  - Seurata liikettä (esim. valvontakamerassa kulkevaa henkilöä)
  - Luoda kolmiulotteinen malli (esim. 3D-rekonstruktio kamerakuvista)
  - Se käyttää usein syväoppimista ja konvoluutiohermoverkkoja (CNN) kuvien analysointiin.

Miten konenäkö toimii

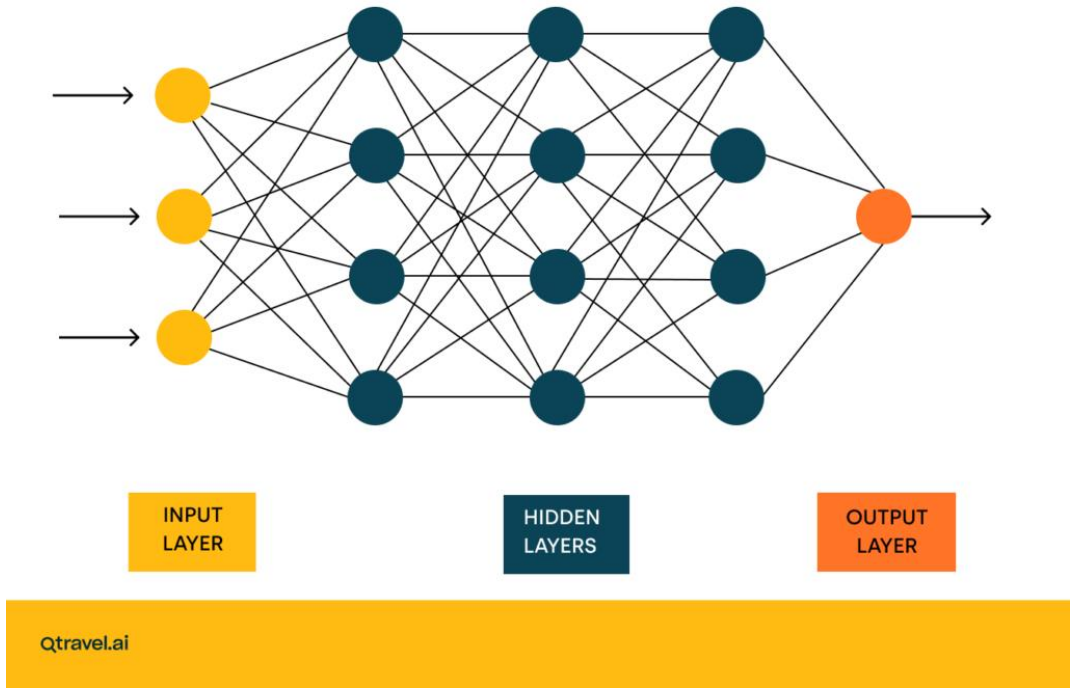
- Konenäkö toimii käyttämällä erilaisia laitteisto- ja ohjelmistokomponentteja visuaalisen tiedon tallentamiseen, käsittelyyn ja analysointiin. Konenäön prosessi noudattaa yleensä seuraavia tehtäviä:
  - Kuvan hankinta
  - Esikäsittely
  - Piirteiden tunnistus
  - Kuvion tunnistus
  - Päätöksenteko

Kuvantunnistus (engl. Image Recognition)

- Kuvantunnistus on yksi konenäön alatehtävistä, jossa pyritään tunnistamaan, mitä kuvassa on.
- Tyypillisiä tehtäviä ovat:
  - Kuvassa on koira vai kissa?
  - Tunnista tekstin fontti
  - Luokittele kuva roskaksi tai uudelleenkäytettäväksi
  - Lue numerokyltti auton kuvasta

Miten neuroverkot toimivat

- Neuroverkot vs ihmisen avihto
- Ensimmäiset solmut vastaanottavat dataa(input layer)
- Jokainen seuraavan tason solmu vastaanottaa edellisen tason solmujen tulokset ja keskittyy tiettyyn osa-alueen analysointiin
- Lopuksi saadaa tulos –esimerkiksi tieto kuvasta



Konenäön sovellukset: Kasvojen tunnistus

- Esimerkit
  - puhelimen lukitus voidaan avata kasvoilla
  - Valvontakamerat
- Algoritmi tunnistaa kasvojen piirteitä ja vertaa tietokantaan
- Käytetty teknologia
  - CNN eli konvoluutioverkko
  - Feature extraction(esim. Facenet)

Konenäköön ja kuvantunnistukseen kehittyneempi käyttö

- Kehittyneempi käyttö vaatii aina ohjelmoinnin osaamista
- Alustoja ovat mm.
  - TensorFlow/Keras – Google kehittämä, suosittu syväoppimiseen.
  - PyTorch – Meta (Facebook) kehittämä, joustavampi ja tutkimukseen suosittu.
  - OpenCV – Klassiseen kuvankäsittelyyn ja konenäköön.
  - YOLO (You Only Look Once) – Nopeaan objektintunnistukseen.
- Kehitysprosessi
  - Valitse projektiisi sopiva kirjasto
  - Kerää ja valmistele data
  - Kouluta malli
  - Optimoi ja testaa malli
  - Vie käyttöön

Konenäköön ja kuvantunnistukseen alustat ilman ohjelmointia

- Teachable Machine (Google)
  - Yksinkertainen selainpohjainen työkalu.
  - Voit ladata kuvia ja opettaa mallin tunnistamaan eri luokkia.
  - Voit viedä mallin käyttöön esim. verkkosivuille ilman koodausta.
  - Soveltuu: Nopea testaus, opetuskäyttö, yksinkertaiset sovellukset.
- Lobe (Microsoft)
  - Helppo "vedä ja pudota" -käyttöliittymä koneoppimismallien kouluttamiseen.
  - Tukee kuvantunnistusta ilman ohjelmointia.
  - Soveltuu: Aloittelijoille ja prototyyppien tekoon.
- Google AutoML Vision
  - Pilvipohjainen ratkaisu, joka kouluttaa kuvantunnistusmalleja automaattisesti.
  - Vaatii vain kuvien lataamisen ja mallin kouluttamisen ohjatusti.
  - Soveltuu: Yrityskäyttöön ja kehittyneempiin sovelluksiin.
- Clarifai ja IBM Watson Visual Recognition
  - Tarjoavat graafiset käyttöliittymät kuvien analysointiin ja mallien koulutukseen.
  - Soveltuu: Yrityksille ja laajempiin projekteihin.

Harjoitus

- Kokeillaan harjoituksena tutustua alustaan https://teachablemachine.withgoogle.com
- Valitse Image project -> Standard image project
- Valitse kaksi kohdetta, esimerkiksi banaani ja appelsiini
- Tee luokittelu ja tee webcam:lla taltiointi kummastakin kohteesta
- Klikkaa train model, jotta saat koulutettua mallin datallasi
- Nyt voit testata malliasi preview ikkunassa
- Seuraavaa harjoitusta varten lataa malli h5 muodossa koneellesi.

Harjoitus 2: pythonilla

- Tee hakemistorakenne kuten oikealla
- Aja seuraavat komennot ja asenna paketit:
  - pip install pillow
  - pip install numpy
  - pip install optree
  - pip install keras
  - pip install tensorflow==2.15.0
- Tämän jälkeen laita seuraavat sisällöt tiedostoihin

```
flower_app/
├── app.py              # Flask-sovellus
├── model.h5            # Tallennettu malli
├── templates/
│   └── index.html      # HTML-sivu
├── static/
│   └── uploads/        # Tallennetut kuvat
```

Harjoitus 2: index.html sisältö

```
<!DOCTYPE html>
<html>
<head>
<title>Kukkien tunnistus</title>
</head>
<body>
<h1>Lataa kuva kukasta</h1>
<form method="POST" enctype="multipart/form-data">
<input type="file" name="image">
<input type="submit" value="Tunnista">
</form>

{% if prediction %}
<h2>Tulos: {{ prediction }} ({{ confidence }} % varmuudella)</h2>
<img src="{{ user_image }}" alt="Kuva" style="max-width:300px;">
{% endif %}
</body>
</html>
```

Harjoitus 2: app.py sisältö

```python
from flask import Flask, render_template, request    Import
import tensorflow as tf    Import "tensorflow" could not be
import numpy as np    Import "numpy" could not be resolved
import os
import cv2    Import "cv2" could not be resolved

app = Flask(__name__)
UPLOAD_FOLDER = 'static/uploads'
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

# Lataa malli ja luokat
model = tf.keras.models.load_model("model.h5")
class_names = ['Luokka 1', 'luokka 2']  # Luokat

def predict_image(img_path):
    img = cv2.imread(img_path)
    img_resized = cv2.resize(img, (224, 224))
    img_array = tf.expand_dims(img_resized / 255.0, 0)
    predictions = model.predict(img_array)
    score = tf.nn.softmax(predictions[0])
```

Harjoitus 2: app.py sisältö

```
    result = class_names[np.argmax(score)]
    confidence = 100 * np.max(score)
    return result, confidence

@app.route("/", methods=["GET", "POST"])
def index():
    if request.method == "POST":
        file = request.files["image"]
        filepath = os.path.join(app.config['UPLOAD_FOLDER'], file.filename)
        file.save(filepath)
        result, confidence = predict_image(filepath)
        return render_template("index.html", prediction=result,
                               confidence=round(confidence, 2),
                               user_image=filepath)
    return render_template("index.html")

if __name__ == "__main__":
    app.run(debug=True)
```

Harjoitus 2: sovelluksen käynnistäminen

- Nyt voit käynnistää sovelluksen komennolla

```
python app.py
```

- Jos kaikki on mennyt niin kuin pitääkin, niin voit avata selaimen osoitteessa

  http://127.0.0.1:5000

- Nyt voit ladata yksittäisen kuvatiedoston koneellesi ja voit kokeilla edellisessä harjoituksessa kouluttamaasi mallia myös paikallisesti

Analytiikka ja datan analysointi

- Analytiikka tekoälyn näkökulmasta tarkoittaa datan keräämistä, käsittelyä ja analysointia tekoälyteknologioiden avulla
  - Ennakoiva analytiikka - Tekoäly voi ennustaa tulevia tapahtumia analysoimalla historiallista dataa ja tunnistamalla trendejä1
  - Generatiivinen analytiikka - Tekoäly voi luoda uusia tietoja ja sisältöjä, kuten tekstiä, kuvia tai musiikkia, perustuen olemassa olevaan dataan2
  - Reaaliaikainen analytiikka - Tekoäly voi analysoida dataa reaaliajassa, mikä mahdollistaa nopean päätöksenteon ja reagoinnin muuttuviin tilanteisiin

Ennakoiva analytiikka

- Ennakoiva analytiikka on kehittyneen analytiikan haara, joka keskittyy tulevien tapahtumien, käyttäytymisten ja tulosten ennustamiseen. Se käyttää tilastollisia tekniikoita, kuten koneoppimisalgoritmeja ja kehittynyttä ennustemallinnusta, analysoidakseen nykyisiä ja historiallisia tietoja ja arvioidakseen todennäköisyyttä, että jotain tapahtuu
- Vaiheet:
  - Datan keruu
  - Datan puhdistaminen
  - Mallinnus
  - Ennusteiden validointi
  - Tulosten tulkinta

Ennakoiva analytiikka

- Käyttökohteet
  - Liiketoiminta: Ennakoiva analytiikka auttaa yrityksiä ennustamaan kassavirtaa, optimoimaan hinnoittelua ja tunnistamaan potentiaalisia asiakkaita
  - Terveydenhuolto: Sairaalat käyttävät ennustemalleja riskien mittaamiseen, sairauksien seurausten ennustamiseen ja hoitolaitteiden toimitusketjujen hallintaan
  - Rahoitus: Pankit ja rahoituslaitokset käyttävät ennakoivaa analytiikkaa riskienhallintaan ja sijoitusstrategioiden kehittämiseen
- Hyödyt
  - Parempi päätöksenteko
  - Riskienhallinta
  - Tehokkuuden parantaminen

Generatiivinen analytiikka

- Generatiivinen analytiikka on kehittynyt analytiikan muoto, joka hyödyntää generatiivista tekoälyä (Gen AI) luodakseen uusia tietoja ja sisältöjä analysoidun datan perusteella. Tämä lähestymistapa eroaa perinteisestä analytiikasta, joka keskittyy pääasiassa olemassa olevan datan analysointiin ja tulkintaan.
- Vaiheet:
  - Datan keruu
  - Datan puhdistaminen
  - Mallinnus
  - Sisällön luominen
  - Tulosten validointi

Generatiivinen analytiikka

- Käyttökohteet
  - Markkinointi: Generatiivinen analytiikka voi luoda personoituja markkinointikampanjoita ja sisältöjä, jotka resonoivat paremmin kohdeyleisön kanssa
  - Luova teollisuus: Taiteilijat ja suunnittelijat voivat käyttää generatiivista analytiikkaa uusien teosten ja konseptien luomiseen
  - Terveysala: Generatiivinen analytiikka voi auttaa luomaan uusia lääketieteellisiä ratkaisuja ja hoitomenetelmiä analysoimalla potilastietoja ja lääketieteellisiä tutkimuksia
- Hyödyt
  - Innovaatio: Mahdollistaa uusien ja luovien ratkaisujen kehittämisen eri aloilla.
  - Personointi: Auttaa luomaan yksilöllisiä ja kohdennettuja sisältöjä ja palveluita.
  - Tehokkuus: Parantaa prosessien ja resurssien käyttöä luomalla automaattisesti uusia tietoja ja sisältöjä.

Reaaliaikainen analytiikka

- Reaaliaikainen analytiikka tarkoittaa prosessia, jossa dataa kerätään, analysoidaan ja esitetään välittömästi sen syntymisen jälkeen. Tämä mahdollistaa ajankohtaisen tiedon hyödyntämisen päätöksenteossa ilman viiveitä
- Vaiheet:
  - Datan kerääminen
  - Datan käsittely
  - Analyysi ja visualisointi
  - Päätöksenteko

Reaaliaikainen analytiikka

- Käyttökohteet
  - Liiketoiminta: Yritykset voivat seurata myyntiä, asiakaskäyttäytymistä ja markkinatrendejä reaaliajassa, mikä auttaa optimoimaan toimintaa ja parantamaan asiakaskokemusta
  - Terveydenhuolto: Sairaalat ja klinikat voivat seurata potilaiden terveydentilaa ja hoitoprosesseja reaaliajassa, mikä parantaa hoidon laatua ja tehokkuutta
  - Logistiikka: Kuljetusyritykset voivat seurata ajoneuvojen sijaintia ja toimitusten etenemistä reaaliajassa, mikä parantaa toimitusketjun hallintaa ja asiakastyytyväisyyttä
- Hyödyt
  - Nopeampi päätöksenteko
  - Parantunut tehokkuus
  - Parempi asiakaskokemus
- Haasteet
  - Datan laatu ja luotettavuusTeknologian ja infrastruktuurin vaatimukset:
  - Reaaliaikainen analytiikka vaatii kehittyneitä teknologioita ja infrastruktuuria

Analytiikkaan soveltuvat alustat

- No-code / Low-code -alustat
  - Power BI + Copilot
  - Tableau + Tableau GPT
  - Google Looker Studio
- ChatGPT tai muut LLM-pohjaiset työkalut
  - "Tunnista poikkeavat arvot tästä datasta."
  - "Luo yhteenveto myynnistä alueittain."
  - "Missä kuukaudessa oli eniten kasvua?"
  - Tämä ei vaadi mitään ohjelmointia – vain selkeän kysymyksen.
- AutoML-alustat
  - Google AutoML
  - DataRobot
  - Amazon SageMaker Canvas

Harjoitus. Datan analysointi generatiivisen tekoälyn avulla

- Seuraavassa harjoituksessa
  - Voit käyttää omaa mallidataasi tai
  - Saat opettajalta mallidataa tai
  - pyydä tekoälyltä mallidataa tällä syötteellä:
    - Voitko auttaa luomaan mallidataa excel-muodossa? Tarvitsisin PowerBi analysointia varten mallidataa ja ajattelin sopivan datan koostuvan verkkokauppadatasta. Tarvitsen dataa, jossa on tuote, hinta, myydyt kappaleet, ostohetki ja jos keksit sopivaa dataa, niin lisää se mukaan. Dataa voisi olla kalenterivuosilta 2022-2023.

Harjoitus. Datan analysointi generatiivisen tekoälyn avulla

- Tässä harjoituksessa tarvitaan henkilökohtaiset tunnukset. Mene osoitteeseen https://chatgpt.com ja kirjaudu esimerkiksi Googlen tunnuksilla sisään. Peruskäyttö ja tämän ohjeen kirjoitushetkellä uusin versio ChatGPT-4o on rajoitetulla käytöllä ilmainen. Sillä pystyy muutaman data-analyysin tekemään.
- Paina + näppäintä, lisää materiaali
- Jos käytät opettajalta saatua materiaalia, niin voit käyttää seuraavalla sivulla olevaa syötettä
- Jos käytät omaa materiaalia, niin kysy tekoälyltä, että millaisen analyysin se voi tehdä datallesi ja pyydä sitä tekemään analyysi
  - HUOM! ChatGPT antaa tarvittaessa myös koodin, jolla on tehnyt visualisoinnit

Harjoitus. Datan analysointi generatiivisen tekoälyn avulla syöte eli kopioi alla oleva syötteeksi

- Voitko tehdä perusanalyysin liitteenä olevalle datalle. Lisäksi haluan seuraavat analyysit:
  1. Yleiskatsaus myyntiin
  Kokonaismyynti
  Lasketaan kokonaismyynti jokaisesta tuotteesta: Myynti = Price × Units Sold
  Keskimääräinen myyntihinta, myyntimäärät ja tuoteryhmät
  2. Aikaan perustuva analyysi
  Milloin ostettiin eniten?
  Miten myynti kehittyi kuukausittain?
  Sesonkivaihtelut?
  3. Asiakasnäkökulma
  Kuinka monta asiakasta?
  Montako kertaa asiakkaat ostivat (jos tunnisteet toistuvat)?
  Isoimmat ostot per asiakas
  4. Tuoteanalyysi
  Mitkä tuotteet myyvät eniten?
  Mitkä tuottavat eniten rahaa?
  Korkein / matalin keskihinta
  5. Visualisoinnit graafeina
  Myynti per kuukausi
  Tuotekohtainen vertailu
  Top 5 -tuotteet liikevaihdon mukaan

Harjoitus. Datan analysointi generatiivisen tekoälyn avulla syöte eli kopioi alla oleva syötteeksi

- Analysoi edellisellä syötteellä saamasi analyysin. Vastaa seuraaviin kysymyksiin?
  - Onko analyysi käyttökelpoinen ja miksi?
  - Voisiko visualisointia parantaa jotenkin? Kokeile pyytää muutoksia tekoälyltä erilaisillä syötteillä
  - Kannattaisiko analyysiä tehdä vielä joillekin osa-alueille? Jos et keksi mille, niin kokeile kysyä tekoälyltä sitä

Harjoitus. Datan analysointi regressiomallilla

- Käytetään yksinkertaista ennustemallia, kuten lineaarista regressiota, datan analysoimiseksi ja ennustamiseksi.
- Esimerkiksi, jos datasetissä on aikasarjatietoa, voit yrittää ennustaa tulevia arvoja regressioanalyytiä käyttäen.
- Seuraavalla dialla esimerkki python-ohjelmointikielellä tehdystä regressiomallista
- Esimerkki käyttää numpy ja matplotlib kirjastoja, joita käytetään yleisesti datan käsittelyssä ja visualisoinnissa, sekä koneoppimiskirjastoa sklearn.

Harjoitus. Datan analysointi regressiomallilla

```
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
import numpy as np

np.random.seed(10)
X = np.random.rand(100, 1) * 10 # X on välillä 0 ja 10

# Lineaarinen suhde Y = 2 * X + 1 + satunnaisvirhe
y = 2 * X + 1 + np.random.randn(100, 1) # Lisätään satunnaista kohinaa

# Jos käytetään olemassaolevaa dataa, niin sen voi tuoda data_clean muuttujaan
# Oletetaan, että data_clean sisältää 'x_column' ja 'y_column'
# X = data_clean[['x_column']] # Tämä on ennustettavien muuttujien syöte
# y = data_clean['y_column'] # Tämä on ennustettava muuttuja

# Mallin luominen ja kouluttaminen
model = LinearRegression()
model.fit(X, y)

# Ennustaminen
y_pred = model.predict(X)

# Visualisoidaan ennuste
plt.scatter(X, y, color='blue') # Aito data
plt.plot(X, y_pred, color='red') # Ennustettu viiva
plt.title('Lineaarinen regressio')
plt.xlabel('X')
plt.ylabel('Y')
plt.show()
```

Harjoitus. Datan analysointi regressiomallilla

- Kokeile muuttaa edellisen harjoitukselle seuraavia muutoksia:
  - Kokeile muuttaa riviä y = 2 * X + 1 + np.random.randn(100, 1), erilaiseksi, että tunnistaako malli "oikean" kaavan
  - Tutki, että missä koodissa määritetään visualisointien värit ja kokeile muuttaa eri värejä visualisoinneissa
  - Kokeile käyttää jotain muuta lineaarista mallia, esimerkksi Ridge tai Lasso mallia. Tutki sitä varten dokumentaatiota osoitteessa https://scikit-learn.org/stable/api/sklearn.linear_model.html

Harjoitus. Datan analysointi klusterimallilla

- Jos datassa on monimutkaisempia ominaisuuksia, voit käyttää klusterointimenetelmää (kuten K-means) datan ryhmittelyyn.
- Tämä on erityisen hyödyllistä, jos haluat löytää piileviä rakenteita tai ryhmiä datassa ilman valmiita luokkia.
- Seuraavalla dialla on esimerkkikoodi K-means klusteroinnista
- Esimerkki käyttää numpy ja matplotlib kirjastoja, joita käytetään yleisesti datan käsittelyssä ja visualisoinnissa, sekä koneoppimiskirjastoa sklearn.

Harjoitus. Datan analysointi klusterimallilla

```
from sklearn.cluster import KMeans

from sklearn.datasets import make_blobs

import pandas as pd

import matplotlib.pyplot as plt


# Luodaan dataa kahdesta klusterista

X, y = make_blobs(n_samples=300, centers=2,
random_state=42)

data_clean = pd.DataFrame(X, columns=['x_column',
'y_column'])


# Oletetaan, että data_clean sisältää 'x_column'
ja 'y_column' ja haluamme ryhmitellä sen

X = data_clean[['x_column', 'y_column']] #
Muutetaan data kahteen muuttujaan
```

```
# Määritellään, että haluamme 3 klusteria

kmeans = KMeans(n_clusters=3,
random_state=0)

data_clean['cluster'] =
kmeans.fit_predict(X)


# Visualisoidaan klusterit

plt.scatter(data_clean['x_column'],
data_clean['y_column'],
c=data_clean['cluster'], cmap='viridis')

plt.title('K-means klusterointi')

plt.xlabel('X')

plt.ylabel('Y')

plt.show()
```

Harjoitus. Datan analysointi klusterimallilla

- Mallidataa luodessa luodaan kaksi keskittymää ja se pakottaa analyysin löytämään 3 klusteria
- Kokeile muuttaa edellisen harjoitukselle seuraavia muutoksia:
  - Kokeile muuttaa koodia niin, että KMeans malli löytää halutut kaksi klusteria
  - Kokeile lisätä keskuksia ja pakottaa koodi löytämään vain esimerkiksi 2 klusteria. Miten hyvin se toimii?

Harjoitus. Datasetin poikkeavien arvojen löytäminen

- Tekoälyä voi käyttää myös tunnistamaan poikkeavia arvoja (anomalia). Tämä voi olla hyödyllistä, jos datasetissä on virheellisiä tai epätavallisia arvoja.
- Seuraavassa esimerkissä näytetään miten tunnistetaan poikkeavia arvoja python-koodilla. Tästä käytetään myös nimitystä anomalien tunnistaminen.
- Koodi luo aluksi mallidataa ja lisää sinne anomaliapisteitä ja sen jälkeen se käyttää sklearn-kirjaston Isolation Forest –pakettia kyseisten anomaliapisteiden tunnistamiseen.

Harjoitus. Datasetin poikkeavien arvojen löytäminen

```
from sklearn.ensemble import IsolationForest

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd


# Luodaan tavallinen data

X = np.random.randn(200, 2) # Normaalijakauma


# Lisätään anomaliaa

outliers = np.random.uniform(low=-6, high=6, size=(10, 2))
# Poikkeavat havainnot


# Yhdistetään data

data = np.vstack([X, outliers])


# Muodostetaan DataFrame ja annetaan sarakkeille nimet

data_clean = pd.DataFrame(data, columns=['x_column',
'y_column’])


# Käytetään kaikkia numeerisia sarakkeita datasta

X = data_clean.select_dtypes(include=['float64', 'int64'])


# Luodaan malli, joka tunnistaa poikkeavat havainnot

model = IsolationForest(contamination=0.1) # Oletetaan,
että 10% tiedoista on poikkeavia

outliers = model.fit_predict(X)


# Merkitään poikkeavat havainnot

data_clean['outlier'] = outliers


# Visualisoidaan poikkeavat havainnot

plt.scatter(data_clean['x_column'],
data_clean['y_column'], c=data_clean['outlier'],
cmap='coolwarm')

plt.title('Anomalian tunnistus')

plt.xlabel('X')

plt.ylabel('Y')

plt.show()
```

Harjoitus. Datasetin poikkeavien arvojen löytäminen

- Kokeile lisätä poikkeavien arvojen joukkoa mallidatan luonnista. Mitä tapahtuu mallin tunnistukselle jos se on enemmän kuin 0.6?

Harjoitus. Logistinen regressio

- Jos datasetissä on selkeästi määritelty luokkia (esim. "hyvä" ja "huono" asiakas), voit käyttää luokittelumallia (kuten logistinen regressio) datan luokitteluun.
- Seuraavassa esimerkissä näytetään miten käytetään logistista regressiota datan lajitteluun.
-

Harjoitus. Logistinen regressio

```
from sklearn.linear_model import LogisticRegression

from sklearn.model_selection import train_test_split

from sklearn.metrics import accuracy_score

import numpy as np

import pandas as pd


# Luodaan dataa kahdesta luokasta

np.random.seed(0)

X = np.random.randn(100, 2)

y = (X[:, 0] + X[:, 1] > 0).astype(int) # Luokka 1, jos X[0] +
X[1] > 0


# Muodostetaan data_clean DataFrame

data_clean = pd.DataFrame(X, columns=['feature1', 'feature2'])

data_clean['feature3'] = data_clean['feature1'] *
data_clean['feature2'] # Lisätty ominaisuus

data_clean['target_column'] = y
```

```
# Oletetaan, että datasetissä on 'feature_columns' ja
'target_column'

X = data_clean[['feature1', 'feature2', 'feature3']] #
Ominaisuudet

y = data_clean['target_column'] # Kohde (luokka)


# Jaetaan data koulutus- ja testijoukkoon

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.3, random_state=0)


# Mallin kouluttaminen

model = LogisticRegression()

model.fit(X_train, y_train)


# Ennustaminen

y_pred = model.predict(X_test)


# Mallin tarkkuus

print(f'Tarkkuus: {accuracy_score(y_test, y_pred)}')
```

Harjoitus. Logistinen regressio

- Katso vielä datan analysointi regressiomallilla –harjoitusta ja yritä saada tulostetuksi näkyviin tieto siitä miten malli toimii
- Tutki miten logistinen regressio toimii ja miten sitä käytetään eri aloilla.

Puheentunnistus

- Puheentunnistus tarkoittaa puheen muuntamista tekstiksi. Kun ihminen puhuu mikrofonille, järjestelmä:
  - Tunnistaa ääniaallot (akustinen signaali).
  - Pilkkoo ne äänteiksi, sanoiksi ja lauseiksi.
  - Muuntaa puhutun sisällön tekstimuotoon, usein reaaliaikaisesti.
- Esimerkkejä:
  - Saneluohjelmat (esim. Google Docsin puhediktointi)
  - Puhe tekstiksi -toiminto älypuhelimessa
  - Automaattiset tekstitykset

Ääniavustajat

- Ääniavustajat ovat ohjelmistoja, jotka ymmärtävät puhuttuja komentoja ja suorittavat niihin liittyviä tehtäviä. Ne perustuvat puheentunnistuksen lisäksi myös luonnollisen kielen ymmärtämiseen (NLP) ja usein myös tekoälyagentteihin.
- Esimerkkejä:
  - Siri (Apple)
  - Google Assistant
  - Amazon Alexa

Puheentunnistuksen tekninen prosessi

1. **Äänen tallennus (Audio capture)**
   - Käyttäjän puhe otetaan talteen mikrofonilla.
   - Tuloksena on digitaalinen äänisignaali (yleensä WAV- tai MP3-muodossa).
2. **Esikäsittely (Preprocessing)**
   - Taustamelun poisto
   - Äänen normalisointi
   - Signaali pilkotaan **lyhyiksi pätkiksi** (esim. 20 ms "frameja").
3. **Ominaisuuksien poiminta (Feature Extraction)**
   - Puheesta lasketaan matemaattisia piirteitä kuten:
     - **MFCC** (Mel-Frequency Cepstral Coefficients)
     - **Spectrogrammit**
   - Nämä piirteet kertovat **millainen ääni** on kyseessä – esim. missä taajuusalueella ja millä voimakkuudella.

Puheentunnistuksen tekninen prosessi

4. Akustinen malli (Acoustic model)

- Malli oppii yhdistämään äänisignaalin **äänteisiin**.
- Yleisesti käytetään **neuroverkkoja** (esim. CNN, LSTM, Transformer).
- Esimerkki: tunnistaa, että tämä ääni vastaa "s"-äännettä.

5. Kielimalli (Language model)

- Määrittää **sanojen ja lauseiden todennäköisyyksiä**.
- Auttaa valitsemaan, sanoiko käyttäjä:
  - "Näin sen kissan" vai
  - "Näin sen kisan"
- Voidaan käyttää **n-gram-malleja, RNN:iä, tai nykyisin BERT/Transformer-malleja**.

6. Sanasto ja dekoodaus

- Äänteet muutetaan tekstiksi sanaston avulla.
- Käytetään dekooderia, joka yhdistää ääni- ja kielimallit optimaalisesti.

Ääniavustajan lisäprosessit

7. **Luonnollisen kielen ymmärrys(NLU)**
   - Ymmärtää intention(intent) ja avainsanat(entities)
   - Esimerkiksi “Avaa Spotify ja soita Slipknot”
     - Intentio “soita musiikkia”, entity “Slipknot”
8. **Toimintalogiikka (Dialog manager)**
   - Päätetään mitä tehdään
9. **Vastaus (Response generation)**
   - Jos tarvitaan vastaus, luodaan teksti esim GPT:tä käyttäen
10. **Tekstin muunto puheeksi (TTS –text to speech)**
    - Vastaus muutetaan puheeksi ja toistetaan käyttäjille

Teknologiat

- Äänen analysointi
  - Python librosa, pyaudio
- Puheentunnistusmallit
  - Google Speech API, Whisper, Vosk, Kaldi
- Kielimallit
  - GPT, BERT, n-gram, LSTM
- Ääniavustajan logiikka
  - Dialogflow, Rasa, Microsoft Bot Framework
- TTS
  - Google TTS, Amazon Polly, Coqui TTS

Harjoitus: Puheentunnistus

- Tutustutaan puheentunnistukseen ja nähdään, miten kone muuntaa puheen tekstiksi. Tähän harjoitukseen tarvitset puhelimen ja Google tilin
- Avaa Google Docs
- Luo uusi tyhjä asiakirja
- Valitse ylhäältä valikosta: Työkalut → Äänikirjoitus
- Valitse kieli: Suomi
- Klikkaa mikrofonikuvaketta ja puhu selkeästi esim. seuraava lause:
- "Tekoälyavustajat voivat auttaa arjen askareissa."
- Lopeta puhe ja katso, miten teksti ilmestyy dokumenttiin.
- Kokeile uudelleen eri lauseilla ja testaa, miten hyvin puhe tulkitaan.

Harjoitus 2: Oman ääniavustajan rakentaminen Voiceflow’lla

- Tutkitaan miten voisi luoda oma yksinkertainen ääniavustaja ilman ohjelmointia. Tähän harjoitukseen tarvitset selaimen ja ilmaisen Voiceflow-tilin osoitteessa https//www.voiceflow.com
- Rekisteröidy ja kirjaudu sisään Voiceflow’iin
- Luo uusi projekti → Valitse: Assistant
- Luo keskustelupolku esim. näin:
  - Start block
  - Speak block: Kirjoita esim. "Hei! Miten voin auttaa?"
  - Choice block: Luo valintoja kuten:
    - "Kerro säästä" → lisää vastaus "Tänään on aurinkoista!"
    - "Kerro vitsi" → lisää vastaus "Miksi tekoäly ei eksy? Koska sillä on aina reittioptimointi!"
- Klikkaa "Test" ja kokeile keskustella avustajan kanssa

Harjoitus 3: Puheentunnistus ja HTML-sivu

- Luo puheentunnistus.html tiedosto ja kirjoita sinne tämän ja seuraavan dian kuvissa oleva sisältö
- Avaa tiedosto selaimessa
- Paina “Aloita puheentunnistus” -nappia
- Puheesi kirjoitetaan html -sivulle

```
<!DOCTYPE html>
<html lang="fi">
<head>
  <meta charset="UTF-8">
  <title>Puheentunnistus Demo</title>
  <style>
    body {
      font-family: sans-serif;
      display: flex;
      flex-direction: column;
      align-items: center;
      padding: 2em;
    }
    button {
      padding: 1em 2em;
      font-size: 1.2em;
      margin-top: 1em;
    }
    #result {
      margin-top: 2em;
      font-size: 1.5em;
      color: #333;
    }
  </style>
</head>
```

Harjoitus 3: Puheentunnistus selaimessa

```
<body>
  <h1>🎤 Puheentunnistus Demo</h1>
  <p>Paina nappia ja puhu. Näet puheesi tekstinä.</p>
  <button onclick="startRecognition()">Aloita puheentunnistus</button>
  <div id="result"></div>

  <script>
    const resultDiv = document.getElementById('result');

    function startRecognition() {
      if (!('webkitSpeechRecognition' in window)) {
        alert("Tämä selain ei tue puheentunnistusta. Käytä Chromea.");
        return;
      }

      const recognition = new webkitSpeechRecognition();
      recognition.lang = 'fi-FI'; // Suomi
      recognition.interimResults = false;
      recognition.maxAlternatives = 1;
      recognition.start();

      recognition.onresult = function(event) {
        const transcript = event.results[0][0].transcript;
        resultDiv.textContent = `Tunnistettu teksti: "${transcript}"`;
      };

      recognition.onerror = function(event) {
        resultDiv.textContent = `Virhe: ${event.error}`;
      };

      recognition.onend = function() {
        console.log("Puheentunnistus päättyi.");
      };
    }
  </script>
</body>
</html>
```

Tekoälyagentit

- Tekoälyagentit (engl. AI agents) ovat ohjelmistoja tai järjestelmiä, jotka kykenevät toimimaan itsenäisesti ympäristössään jonkin tavoitteen saavuttamiseksi. Ne käyttävät tekoälymenetelmiä, kuten koneoppimista, päättelyä tai luonnollista kieltä, ja voivat tehdä päätöksiä, suunnitella toimia ja oppia kokemuksistaan.
- Peruspiirteet tekoälyagentille
  - Havaintokyky (perception
  - Päätöksenteko (reasoning/decision-making
  - Toiminta (action)
  - Oppiminen (learning)

Esimerkkejä tekoälyagenteista

- Verkkopohjaiset avustajat kuten ChatGPT tai Copilot, jotka suorittavat tehtäviä pyynnöstä ja voivat olla osa laajempaa agenttiverkostoa.
- Autonomiset robotit, kuten siivousrobotit tai itseajavat autot.
- Finanssialan agentit, jotka käyvät kauppaa markkinoilla automaattisesti.
- Multi-agentti-järjestelmät, joissa useat agentit kommunikoivat ja tekevät yhteistyötä (esim. logistiikkaverkostoissa tai simulaatioissa).

Harjoitus 1: Copilot agentti

- Tähän harjoitukseen tarvitaan Microsoft Copilot lisenssi
- Mene osoitteeseen https://copilot.microsoft.com
- Klikkaa oikeassa yläreunassa ”Luo agentti” tai ”create an agent” –nappia
- Siellä voit määrittää tekoälyavustimen kanssa, että millaisen agentin haluat luoda.
- Voit luoda vaikkapa keskusteluavustajan

Harjoitus 2: Pythonilla tehty tekoälyagentti

- Tässä harjoituksessa kokeillaan tehdä yksinkertainen tekoälyagenttiverkosto
- Tähän tarvitset
  - Python 3.11.10
  - Ollama (https://ollama.com/) ja esimerkiksi llama3.1 tekoälymalli
  - CrewAI
- Nyt kun käytetään jatkuvasti eri versioita pythonista, niin kannattaa olla kärryillä siitä, mitä milläkin versiolla tehdään.
  - Tähän voi asentaa pyenv työkalun https://github.com/pyenv/pyenv , jolla voi vaihtaa eri python versioiden välillä
  - Haasteena on se, että jos asennat työkalun väärällä pythonin versiolla, niin se yrittää käyttää väärää versiota jatkossakin.

Harjoitus 2: Pythonilla tehty tekoälyagentti

- Asenna CrewAI komennolla

```
pip install crewai
```

- Luo projekti crewAiTesti käyttämällä komentoa

```
crewai create crew crewAiTesti
```

- Jos edellä mainittu projekti ei toimi, niin kyseessä saattaa olla väärään python ympäristöön asennettu kirjasto. Tällöin pitää ottaa esim pythonin versio 3.11.10 käyttöön tätä varten.
- Se kysyy kaksi kysymystä
  - Toisessa kysyy, mitä tekoälymallien tarjoajaa käytetään. Valitaan tässä yhteydessä ollama
  - Sitten se kysyy mitä mallia käytetään. Valitse malli missä lukee llama3.1

Harjoitus 2: Pythonilla tehty tekoälyagentti

```
(base) villeberg@Mac agentit % crewai create crew latest-ai-development

Creating folder latest_ai_development...
Cache expired or not found. Fetching provider data from the web...
Downloading  [####################################]  434605/20872
Select a provider to set up:
1. openai
2. anthropic
3. gemini
4. nvidia_nim
5. groq
6. ollama
7. watson
8. bedrock
9. azure
10. cerebras
11. sambanova
12. other
q. Quit
Enter the number of your choice or 'q' to quit: 6
```

Harjoitus 2: Pythonilla tehty tekoälyagentti

```
Select a model to use for Ollama:
1. ollama/llama3.1
2. ollama/mixtral
q. Quit
Enter the number of your choice or 'q' to quit: 1
API keys and model saved to .env file
Selected model: ollama/llama3.1
  - Created latest_ai_development/.gitignore
  - Created latest_ai_development/pyproject.toml
  - Created latest_ai_development/README.md
  - Created latest_ai_development/knowledge/user_preference.txt
  - Created latest_ai_development/src/latest_ai_development/__init__.py
  - Created latest_ai_development/src/latest_ai_development/main.py
  - Created latest_ai_development/src/latest_ai_development/crew.py
  - Created latest_ai_development/src/latest_ai_development/tools/custom_tool.py
  - Created latest_ai_development/src/latest_ai_development/tools/__init__.py
  - Created latest_ai_development/src/latest_ai_development/config/agents.yaml
  - Created latest_ai_development/src/latest_ai_development/config/tasks.yaml
Crew latest-ai-development created successfully!
(base) villeberg@Mac agentit %
```

Harjoitus 2: Pythonilla tehty tekoälyagentti

- Nyt sinulla on hyvä pohja tehdä oma tekoälyagentti. Voit kokeilla ajaa projektin kansiosta komennolla

```
crewai run
```

- Projekti tuottaa sinulle report.md tiedoston. Tutki sen sisältöä
- Tutki määritystiedostoja
  - `src/testi/config/agents.yaml` agenttien määritys
  - `src/testi/config/tasks.yaml` tehtävien määritys
  - `src/testi/crew.py` oman logiigan, työkalujen ja määritteiden lisäys
  - `src/testi/main.py` sovelluksen sisään tuotavat määritteet agenteille ja tehtävät
- Em. Tiedostoja muokkaamalla voit mukauttaa agenttia toimimaan haluamallasi tavalla.