**1. Johdanto**

**Esitellään projektin tavoitteet ja datan käsittelyprosessit.**

**Tavoitteena on analysoida kerättyä dataa ja saada hyödyllisiä näkemyksiä.**

**2. Datan lataaminen ja yhdistäminen**

**Ladataan Firestoresta data eri zoneille.**

**Data yhdistetään yhteen DataFrameen ja muokataan sopivaan muotoon.**

**Koodi:**

**# Haetaan data Firestoresta jokaiselle zonelle**

**all\_data = []**

**for zone, sub\_collection in zones.items():**

**sub\_collection\_ref = db.collection(collection).document(zone).collection(sub\_collection)**

**docs = sub\_collection\_ref.stream()**

**for doc in docs:**

**doc\_dict = doc.to\_dict()**

**doc\_dict['id'] = doc.id**

**doc\_dict['zone'] = zone**

**all\_data.append(doc\_dict)**

**# Muutetaan data pandas DataFrameksi**

**df = pd.DataFrame([{**

**'id': entry['id'],**

**'zone': entry['zone'],**

**'humidity': float(entry['humidity']),**

**'temperature': float(entry['temperature']),**

**'timestamp': entry['timestamp']**

**} for entry in all\_data])**

**3. Datan valmistelu ja visualisointi**

**Muokataan aikaleimat datetime-muotoon ja asetetaan ne indeksiksi.**

**Visualisoidaan lämpötilan ja kosteuden aikasarjat.**

**Koodi:**

**df['timestamp'] = pd.to\_datetime(df['timestamp'])**

**df.sort\_values('timestamp', inplace=True)**

**df.set\_index('timestamp', inplace=True)**

**plt.figure(figsize=(14, 6))**

**for zone in zones.keys():**

**subset = df[df['zone'] == zone]**

**plt.plot(subset.index, subset['humidity'], label=f'Kosteus {zone}')**

**plt.plot(subset.index, subset['temperature'], label=f'Lämpötila {zone}')**

**plt.show()**

**4. K-Means-klusterointi**

**Normalisoidaan data ja suoritetaan K-Means-klusterointi kolmelle klusterille.**

**Visualisoidaan klusterien jakaumat ja klusteriselitteet lisätään DataFrameen.**

**Koodi:**

**scaler = StandardScaler()**

**X\_scaled = scaler.fit\_transform(X)**

**kmeans = KMeans(n\_clusters=3, random\_state=0)**

**df['cluster'] = kmeans.fit\_predict(X\_scaled)**

**5. PCA (Principal Component Analysis)**

**Lasketaan pääkomponentit ja lisätään ne DataFrameen.**

**Visualisoidaan PCA-komponenttien hajontakaavio zonittain.**

**Koodi:**

**pca = PCA(n\_components=2)**

**X\_pca = pca.fit\_transform(X\_scaled)**

**df['pca\_comp1'] = X\_pca[:, 0]**

**df['pca\_comp2'] = X\_pca[:, 1]**

**plt.figure(figsize=(10, 6))**

**for zone, color in zone\_colors.items():**

**subset = df[df['zone'] == zone]**

**plt.scatter(subset['pca\_comp1'], subset['pca\_comp2'], color=color, label=zone)**

**plt.legend(title='Zone')**

**plt.show()**

**6. PLSRegression (Partial Least Squares)**

**Sovelletaan osittaista pienimmän neliösumman regressiota.**

**Visualisoidaan PLS-komponentit.**

**Koodi:**

**pls = PLSRegression(n\_components=2)**

**X\_pls = df[['humidity', 'temperature']]**

**Y\_pls = df[['humidity']]**

**pls.fit(X\_pls, Y\_pls)**

**df['pls\_comp1'] = pls.x\_scores\_[:, 0]**

**df['pls\_comp2'] = pls.x\_scores\_[:, 1]**

**plt.figure(figsize=(10, 6))**

**for zone, color in zone\_colors.items():**

**subset = df[df['zone'] == zone]**

**plt.scatter(subset['pls\_comp1'], subset['pls\_comp2'], color=color, label=zone)**

**plt.legend(title='Zone')**

**plt.show()**

**7. Histogrammit ja klusterit**

**Visualisoidaan histogrammit kosteudesta ja lämpötilasta zonittain.**

**Esitetään K-Means-klustereiden jakauma.**

**Koodi:**

**fig, axs = plt.subplots(2, 1, figsize=(14, 10))**

**for zone, color in zone\_colors.items():**

**subset = df[df['zone'] == zone]**

**axs[0].hist(subset['humidity'], bins=20, alpha=0.5, label=f'Kosteus ({zone})', color=color)**

**axs[0].set\_title('Kosteuden histogrammi')**

**axs[1].pie(cluster\_counts, labels=cluster\_counts.index, autopct='%1.1f%%')**

**axs[1].set\_title('Klusterien jakauma')**

**plt.show()**

**8. Lineaarinen regressio**

**Mallinnetaan lineaarinen yhteys kosteuden ja lämpötilan välillä.**

**Visualisoidaan regressiolinja ja lasketaan R²-arvo.**

**Koodi:**

**reg\_model = LinearRegression()**

**reg\_model.fit(X\_reg, y\_reg)**

**y\_pred = reg\_model.predict(X\_reg)**

**plt.scatter(X\_reg, y\_reg, label='Data')**

**plt.plot(X\_reg, y\_pred, color='red', label='Regressiolinja')**

**plt.legend()**

**plt.show()**

**9. ARIMA-mallinnus**

**Sovelletaan ARIMA-mallia aikasarjoihin.**

**Ennustetaan tulevia arvoja ja visualisoidaan ennusteet.**

**Koodi:**

**model = ARIMA(example\_data\_diff, order=(1, 1, 1))**

**model\_fit = model.fit()**

**forecast = model\_fit.forecast(steps=5)**

**plt.plot(example\_data.index, example\_data, label='Todellinen data')**

**plt.plot(forecast\_index, forecast, linestyle='--', label='Ennuste')**

**plt.legend()**

**plt.show()**