# Viikko 43 -tehtävät

## Tehtävä 1

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix, accuracy\_score, precision\_score, recall\_score

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.tree import export\_graphviz

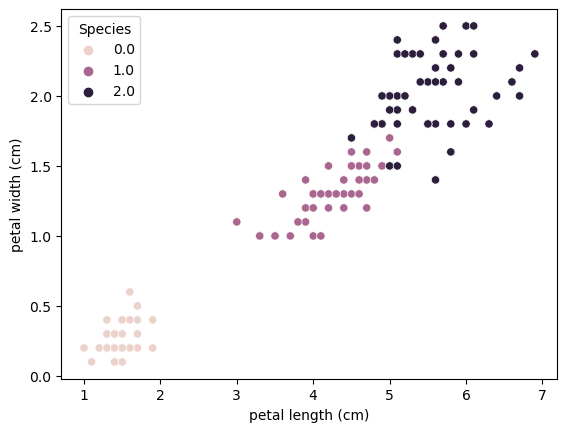
import graphviz

from sklearn import tree

df = pd.read\_csv('./work/viikko8/datasets/iris.csv')

sns.scatterplot(x='petal length (cm)', y='petal width (cm)', hue='Species', data=df)

plt.show()



X = df.iloc[:, 0:4]

y = df.iloc[:, [4]]

columns = X.columns

*# Splitting the dataset into the Training set and Test set*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 0)

*# Training the Decision Tree Classification model*

model = tree.DecisionTreeClassifier(max\_depth=4, criterion='gini')

model.fit(X\_train, y\_train)

mfi = model.feature\_importances\_

*# Predicting the Test set results*

y\_pred = model.predict(X\_test)

y\_pred\_pros = model.predict\_proba(X\_test)

*# Making the Confusion Matrix and accuracy\_score*

cm = confusion\_matrix(y\_test, y\_pred)

ax = plt.axes()

sns.heatmap(cm, annot=True, fmt='g', ax=ax)

ax.set\_title("DT")

plt.show()

*# Calculate accuracy score*

score = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy score: {score}")

Kuva, joka sisältää kohteen kuvakaappaus, teksti, neliö, Suorakaide

Kuvaus luotu automaattisesti

Accuracy score: 0.9736842105263158

*# Create dot file for graphviz visualization*

dot\_data = export\_graphviz(

            model,

            out\_file =  None,

            feature\_names = columns,

            class\_names = df['Class'].unique(),

            filled = True,

            rounded = True)

graph = graphviz.Source(dot\_data)

graph.render('./work/viikko8/iris')

Kuva, joka sisältää kohteen teksti, käsiala, kuvakaappaus, Tarralappu

Kuvaus luotu automaattisesti

*# predict new*

df\_new = pd.read\_csv('./work/viikko8/datasets/new-iris.csv')

predictions = model.predict(df\_new)

print(predictions)



## Tehtävä 2

Sama koodi kuin tehtävässä 1. Vaihdetaan vain malli:

*# Training the Decision Tree Classification model*

model = ensemble.RandomForestClassifier(max\_depth=5)

Kuva, joka sisältää kohteen kuvakaappaus, teksti, neliö, Suorakaide

Kuvaus luotu automaattisesti

Accuracy score: 0.9736842105263158

*# predict new*

df\_new = pd.read\_csv('./work/viikko8/datasets/new-iris.csv')

predictions = model.predict(df\_new)

print(predictions)



## Tehtävä 3

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import confusion\_matrix, accuracy\_score, precision\_score, recall\_score

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.compose import ColumnTransformer

from sklearn.preprocessing import OneHotEncoder

from sklearn.tree import export\_graphviz

import graphviz

from sklearn import tree

df = pd.read\_csv('./work/viikko8/datasets/titanic.csv')

X = df.iloc[:, 0:3]

y = df.iloc[:, [3]]

columns = ['PClass\_1', 'PClass\_2', 'Gender', 'Age']

print(columns)

*# dummies*

X\_org = X

ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(drop='first'), ['PClass', 'Gender'])], remainder='passthrough')

X = ct.fit\_transform(X)

*# Splitting the dataset into the Training set and Test set*

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25, random\_state = 0)

*# Training the Decision Tree Classification model*

model = tree.DecisionTreeClassifier(max\_depth=4, criterion='gini')

model.fit(X\_train, y\_train)

mfi = model.feature\_importances\_

*# Predicting the Test set results*

y\_pred = model.predict(X\_test)

y\_pred\_pros = model.predict\_proba(X\_test)

*# Making the Confusion Matrix and accuracy\_score*

cm = confusion\_matrix(y\_test, y\_pred)

ax = plt.axes()

sns.heatmap(cm, annot=True, fmt='g', ax=ax)

ax.set\_title("DT")

plt.show()

*# Calculate accuracy score*

score = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy score: {score}")

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, muotoilu

Kuvaus luotu automaattisesti

Accuracy score: 0.7957317073170732

*# Create dot file for graphviz visualization*

dot\_data = export\_graphviz(

            model,

            out\_file =  None,

            feature\_names = columns,

            class\_names = df['Survived'].astype(str).unique(),

            filled = True,

            rounded = True)

graph = graphviz.Source(dot\_data)

graph.render('./work/viikko8/titanic')

*# predict new*

df\_new = pd.read\_csv('./work/viikko8/datasets/titanic-new.csv')

df\_new = ct.transform(df\_new)

Kuva, joka sisältää kohteen teksti, diagrammi, kuvakaappaus, Suunnitelma

Kuvaus luotu automaattisesti

predictions = model.predict(df\_new)

print(predictions)



## Tehtävä 4

Sama koodi kuin tehtävässä 3. Vaihdetaan vain malli:

*# Training the Decision Tree Classification model*

model = ensemble.RandomForestClassifier(max\_depth=5)

*# Making the Confusion Matrix and accuracy\_score*

cm = confusion\_matrix(y\_test, y\_pred)

ax = plt.axes()

sns.heatmap(cm, annot=True, fmt='g', ax=ax)

ax.set\_title("DT")

plt.show()

*# Calculate accuracy score*

score = accuracy\_score(y\_test, y\_pred)

print(f"Accuracy score: {score}")

Kuva, joka sisältää kohteen teksti, kuvakaappaus, diagrammi, muotoilu

Kuvaus luotu automaattisesti

Accuracy score: 0.8201219512195121

*# predict new*

df\_new = pd.read\_csv('./work/viikko8/datasets/titanic-new.csv')

df\_new = ct.transform(df\_new)

predictions = model.predict(df\_new)

print(predictions)



Satunnaismetsällä saadaan hieman parempi accuracy score, mutta ennustukset ovat kuitenkin samat uudelle datalle.