**Harjoitukset**

Koneoppiminen, Syksy 2020

Tapani Alastalo

M1475

**Sisältö**

[1 Tehtävä 1 Lineaarinen regressio 2](#_Toc55579967)

[1.1 Lähdekoodit 3](#_Toc55579968)

[2 Tehtävä 2 4](#_Toc55579969)

[2.1 Lähdekoodit 5](#_Toc55579970)

[3 Tehtävä 3 7](#_Toc55579971)

[3.1 Lähdekoodi 7](#_Toc55579972)

[4 Tehtävät 4 Neuroverkot 8](#_Toc55579973)

[4.1 Lähdekoodi 8](#_Toc55579974)

[5 Tehtävä 5 10](#_Toc55579975)

[5.1 Lähdekoodi 11](#_Toc55579976)

[6 Tehtävä 6 12](#_Toc55579977)

[6.1 Lähdekoodi 13](#_Toc55579978)

[6.2 Lähdekoodi 15](#_Toc55579979)

[7 Tehtävä 7 17](#_Toc55579980)

[7.1 Lähdekoodi 17](#_Toc55579981)

[8 Tehtävä 8 18](#_Toc55579982)

[9 Tehtävä 9 19](#_Toc55579983)

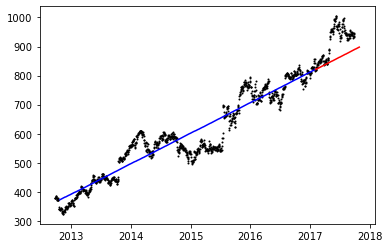
[10 Tehtävä 10 19](#_Toc55579984)

[11 Ss 19](#_Toc55579985)

[12 Tehtävä 20: Lähtevien asiakkaiden tunnistaminen 19](#_Toc55579986)

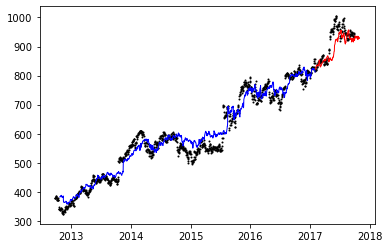
# Tehtävä 1 Lineaarinen regressio

a)



Ennusteen keskivirhe testidatassa on 70

b)



Ennusteen keskivirhe testidatassa on 35

## Lähdekoodit

import pandas as pd

import numpy as np

from sklearn import linear\_model

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

df = pd.read\_csv('data/Google\_Stock\_Price.csv')

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

df['Time'] = df.apply(lambda row: len(df) - row.name, axis=1)

df['CloseFuture'] = df['Close'].shift(30)

df\_test = df[:185]

df\_train = df[185:]

X = np.array(df\_train[['Time', 'Close']])

y = np.array(df\_train['CloseFuture'])

model = linear\_model.LinearRegression()

model.fit(X, y)

ennuste\_train = model.predict(X)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Time', 'Close']])

ennuste\_test = model.predict(X\_test)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Date'].values, df['Close'].values, color='black', s=1)

plt.plot((df\_train['Date'] + pd.DateOffset(days=30)).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Date'] + pd.DateOffset(days=30)).values, df\_test['Ennuste'].values, color='red')

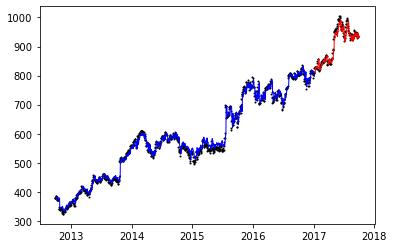
plt.show()

df\_validation = df\_test.dropna()

print("Ennusteen keskivirhe test datassa on %.f" % mean\_absolute\_error(df\_validation['CloseFuture'], df\_validation['Ennuste']))

# Tehtävä 2

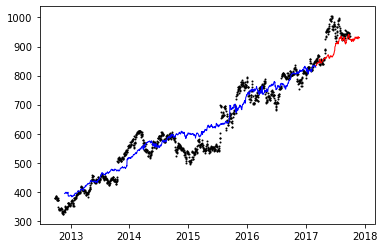
a) 7 päivän ennuste tulevaisuuteen



Ennusteen keskivirhe opetusdatassa on 16

Ennusteen keskivirhe testidatassa on 20

b) 60 päivän ennuste tulevaisuuteen



Ennusteen keskivirhe opetusdatassa on 34

Ennusteen keskivirhe testidatassa on 60

## Lähdekoodit

import pandas as pd

import numpy as np

from sklearn import linear\_model

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

days\_to\_forecast = 60

df = pd.read\_csv('data/Google\_Stock\_Price.csv')

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

df['Time'] = df.apply(lambda row: len(df) - row.name, axis=1)

df['CloseFuture'] = df['Close'].shift(days\_to\_forecast)

df\_test = df[:185]

df\_train = df[185:]

X = np.array(df\_train[['Time', 'Close']])

y = np.array(df\_train['CloseFuture'])

model = linear\_model.LinearRegression()

model.fit(X, y)

ennuste\_train = model.predict(X)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Time', 'Close']])

ennuste\_test = model.predict(X\_test)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Date'].values, df['Close'].values, color='black', s=1)

plt.plot((df\_train['Date'] + pd.DateOffset(days=days\_to\_forecast)).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Date'] + pd.DateOffset(days=days\_to\_forecast)).values, df\_test['Ennuste'].values, color='red')

plt.show()

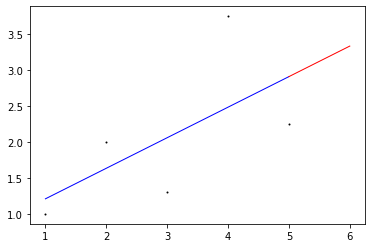
df\_train\_validation = df\_train.dropna()

df\_test\_validation = df\_test.dropna()

print("Ennusteen keskivirhe opetus datassa on %.f" % mean\_absolute\_error(df\_train\_validation['CloseFuture'], df\_train\_validation['Ennuste']))

print("Ennusteen keskivirhe test datassa on %.f" % mean\_absolute\_error(df\_test\_validation['CloseFuture'], df\_test\_validation['Ennuste']))

# Tehtävä 3



## Lähdekoodi

import pandas as pd

import numpy as np

from sklearn import linear\_model

import matplotlib.pyplot as plt

data = [[1.00,1.00],[2.00,2.00],[3.00,1.30],[4.00,3.75],[5.00,2.25],[6.00, None]]

df = pd.DataFrame(data, columns=['X', 'Y'])

df\_train = df[:5]

df\_test = df[4:]

X = np.array(df\_train['X'])

X = X.reshape(-1,1) # vain jos yksiulotteinen taulukko

y = np.array(df\_train['Y'])

model = linear\_model.LinearRegression()

model.fit(X, y)

df\_train['Ennuste'] = model.predict(X)

X\_test = np.array(df\_test['X'])

X\_test = X\_test.reshape(-1,1) # vain jos yksiulotteinen taulukko

df\_test['Ennuste'] = model.predict(X\_test)

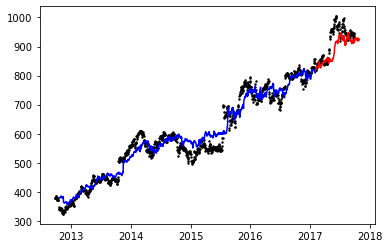
plt.scatter(df['X'].values, df['Y'].values, color='black', s=1)

plt.plot(df\_train['X'].values, df\_train['Ennuste'].values, color='blue', linewidth=1)

plt.plot(df\_test['X'].values, df\_test['Ennuste'].values, color='red', linewidth=1)

plt.show()

# Tehtävät 4 Neuroverkot



Ennusteen keskivirhe opetusdatassa on 30

Ennusteen keskivirhe testidatassa on 40

## Lähdekoodi

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

import tensorflow as tf

from tensorflow import keras

from sklearn import preprocessing

days\_to\_forecast = 30

df = pd.read\_csv('data/Google\_Stock\_Price.csv')

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

df['Time'] = df.apply(lambda row: len(df) - row.name, axis=1)

df['CloseFuture'] = df['Close'].shift(days\_to\_forecast)

df\_train = df[:185]

df\_test = df[185:]

X = np.array(df\_train[['Time', 'Close']])

scaler = preprocessing.MinMaxScaler()

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

y = np.array(df\_train['CloseFuture'])

# luodaan sequential tyyppinen neuroverkkomalli

model = tf.keras.Sequential([

# määritellään neuroverkon piilotettu kerros. 10 neuronia (2 inputs), activation funktio = rectified lineaarifunction, input kerros (input\_shape) = input arvojen lukumäärä

keras.layers.Dense(10, activation='relu', input\_shape=(2,)),

# 2. piilotettu kerros

keras.layers.Dense(10, activation='relu'),

# mallin output kerros. 1 ulostulo (output). Ei aktivointifunktiota # , activation='softmax'

keras.layers.Dense(1)])

# optimointialgoritmi Adam algoritmi (learning rate=0.001).

model.compile(optimizer=keras.optimizers.Adam(learning\_rate=0.001), #'adam', #tf.train.AdamOptimizer(0.001),

loss='mse', #'categorical\_crossentropy',

metrics=['mae']) # ['accuracy'])

# epochs = kuinka monta kertaa opetusdata käydään läpi training vaiheessa (painotus), batch\_size = kuinka monen data rivin jälkeen painokertoimia päivitetään (oppiminen)

model.fit(X\_scaled, y, epochs = 100, batch\_size = 10)

ennuste\_train = model.predict(X\_scaled)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Time', 'Close']])

X\_testscaled = scaler.transform(X\_test)

ennuste\_test = model.predict(X\_testscaled)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Date'].values, df['Close'].values, color='black', s=2)

plt.plot((df\_train['Date'] + pd.DateOffset(days=30)).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Date'] + pd.DateOffset(days=30)).values, df\_test['Ennuste'].values, color='red')

plt.show()

df\_train\_validation = df\_train.dropna()

df\_test\_validation = df\_test.dropna()

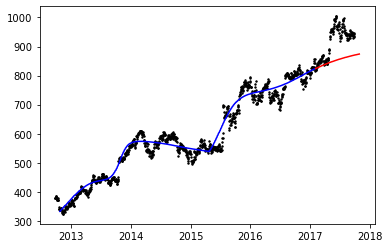
print("Ennusteen keskivirhe opetusdatassa on %.f" %

mean\_absolute\_error(df\_train\_validation['CloseFuture'], df\_train\_validation['Ennuste']))

print("Ennusteen keskivirhe testidatassa on %.f" %

mean\_absolute\_error(df\_test\_validation['CloseFuture'], df\_test\_validation['Ennuste']))

# Tehtävä 5



Ennusteen keskivirhe opetusdatassa on 19

Ennusteen keskivirhe testidatassa on 74

## Lähdekoodi

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

import tensorflow as tf

from tensorflow import keras

from sklearn import preprocessing

days\_to\_forecast = 30

df = pd.read\_csv('data/Google\_Stock\_Price.csv')

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

df['Time'] = df.apply(lambda row: len(df) - row.name, axis=1)

df['CloseFuture'] = df['Close'].shift(days\_to\_forecast)

df\_train = df[:185]

df\_test = df[185:]

X = np.array(df\_train[['Time']])

X = X.reshape(-1,1)

scaler = preprocessing.MinMaxScaler()

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

y = np.array(df\_train['CloseFuture'])

# luodaan sequential tyyppinen neuroverkkomalli

model = tf.keras.Sequential([

# määritellään neuroverkon piilotettu kerros. 10 neuronia (1 input), activation funktio = sigmoid, input kerros (input\_shape) = input arvojen lukumäärä

keras.layers.Dense(20, activation='sigmoid', input\_shape=(1,)),

# 2. piilotettu kerros

keras.layers.Dense(20, activation='sigmoid'),

# 3. piilotettu kerros

keras.layers.Dense(20, activation='relu'),

# mallin output kerros. 1 ulostulo (output). Ei aktivointifunktiota # , activation='softmax'

keras.layers.Dense(1)])

# optimointialgoritmi Adam algoritmi (learning rate=0.001).

model.compile(optimizer=keras.optimizers.Adam(learning\_rate=0.01), #'adam', #tf.train.AdamOptimizer(0.001),

loss='mse', #'categorical\_crossentropy',

metrics=['mae']) # ['accuracy'])

# epochs = kuinka monta kertaa opetusdata käydään läpi training vaiheessa (painotus), batch\_size = kuinka monen data rivin jälkeen painokertoimia päivitetään (oppiminen)

model.fit(X\_scaled, y, epochs = 100, batch\_size = 10)

ennuste\_train = model.predict(X\_scaled)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Time']])

X\_test = X\_test.reshape(-1,1)

X\_testscaled = scaler.transform(X\_test)

ennuste\_test = model.predict(X\_testscaled)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Date'].values, df['Close'].values, color='black', s=2)

plt.plot((df\_train['Date'] + pd.DateOffset(days=30)).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Date'] + pd.DateOffset(days=30)).values, df\_test['Ennuste'].values, color='red')

plt.show()

df\_train\_validation = df\_train.dropna()

df\_test\_validation = df\_test.dropna()

print("Ennusteen keskivirhe opetusdatassa on %.f" %

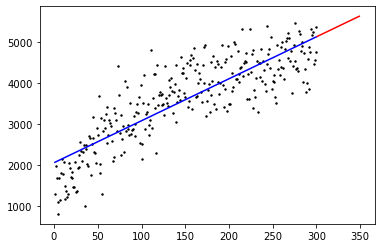
mean\_absolute\_error(df\_train\_validation['CloseFuture'], df\_train\_validation['Ennuste']))

print("Ennusteen keskivirhe testidatassa on %.f" %

mean\_absolute\_error(df\_test\_validation['CloseFuture'], df\_test\_validation['Ennuste']))ss

# Tehtävä 6

a) Lineaarista regressiomallia, jonka input-muuttujana on pelkkä aika.



Ennusteen keskivirhe opetusdatassa on 452

## Lähdekoodi

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

from sklearn import linear\_model

df = pd.read\_csv('data/Kysynta.csv', sep=';', encoding='latin\_1')

print(df)

df\_train = df[:300]

df\_test = df[300:]

for i in range(301, 350):

df\_test = df\_test.append({'Päivä': i}, ignore\_index=True)

X = np.array(df\_train[['Päivä']])

X = X.reshape(-1,1)

y = np.array(df\_train['Kysyntä'])

model = linear\_model.LinearRegression()

model.fit(X, y)

ennuste\_train = model.predict(X)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Päivä']])

X\_test = X\_test.reshape(-1,1)

ennuste\_test = model.predict(X\_test)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Päivä'].values, df['Kysyntä'].values, color='black', s=2)

plt.plot((df\_train['Päivä']).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Päivä']).values, df\_test['Ennuste'].values, color='red')

plt.show()

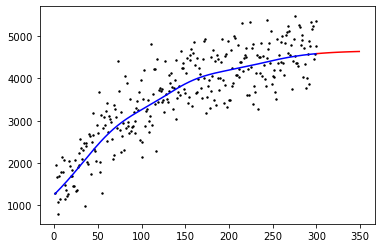
df\_train\_validation = df\_train.dropna()

df\_test\_validation = df\_test.dropna()

print("Ennusteen keskivirhe opetusdatassa on %.f" %

mean\_absolute\_error(df\_train\_validation['Kysyntä'], df\_train\_validation['Ennuste']))

b) MLP-neuroverkkoa, jonka input-muuttujana on pelkkä aika.



Ennusteen keskivirhe opetusdatassa on 405

Ennusteen keskivirhe testidatassa on 407

## Lähdekoodi

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn.metrics import mean\_absolute\_error

import tensorflow as tf

from tensorflow import keras

from sklearn import preprocessing

df = pd.read\_csv('data/Kysynta.csv', sep=';', encoding='latin\_1')

print(df)

df\_train = df[:300]

df\_test = df[300:]

for i in range(301, 350):

df\_test = df\_test.append({'Päivä': i}, ignore\_index=True)

X = np.array(df\_train[['Päivä']])

X = X.reshape(-1,1)

scaler = preprocessing.MinMaxScaler()

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

y = np.array(df\_train['Kysyntä'])

# luodaan sequential tyyppinen neuroverkkomalli

model = tf.keras.Sequential([

# määritellään neuroverkon piilotettu kerros. 10 neuronia (1 input), activation funktio = sigmoid, input kerros (input\_shape) = input arvojen lukumäärä

keras.layers.Dense(20, activation='sigmoid', input\_shape=(1,)),

# 2. piilotettu kerros

keras.layers.Dense(20, activation='tanh'),

# 3. piilotettu kerros

keras.layers.Dense(20, activation='relu'),

# mallin output kerros. 1 ulostulo (output). Ei aktivointifunktiota # , activation='softmax'

keras.layers.Dense(1)])

# optimointialgoritmi Adam algoritmi (learning rate=0.001).

model.compile(optimizer=keras.optimizers.Adam(learning\_rate=0.01),

loss='mse',

metrics=['mae'])

# epochs = kuinka monta kertaa opetusdata käydään läpi training vaiheessa (painotus), batch\_size = kuinka monen data rivin jälkeen painokertoimia päivitetään (oppiminen)

model.fit(X\_scaled, y, epochs = 100, batch\_size = 10)

ennuste\_train = model.predict(X\_scaled)

df\_train['Ennuste'] = ennuste\_train

X\_test = np.array(df\_test[['Päivä']])

X\_test = X\_test.reshape(-1,1)

X\_testscaled = scaler.transform(X\_test)

ennuste\_test = model.predict(X\_testscaled)

df\_test['Ennuste'] = ennuste\_test

plt.scatter(df['Päivä'].values, df['Kysyntä'].values, color='black', s=2)

plt.plot((df\_train['Päivä']).values, df\_train['Ennuste'].values, color='blue')

plt.plot((df\_test['Päivä']).values, df\_test['Ennuste'].values, color='red')

plt.show()

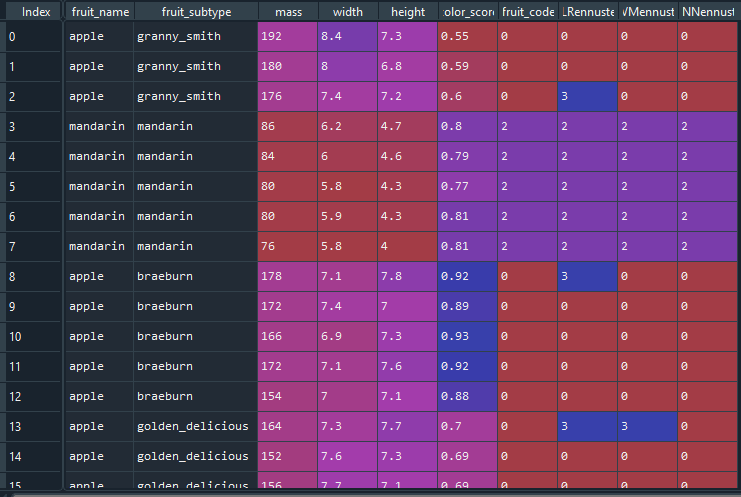
df\_train\_validation = df\_train.dropna()

df\_test\_validation = df\_test.dropna()

print("Ennusteen keskivirhe opetusdatassa on %.f" %

mean\_absolute\_error(df\_train\_validation['Kysyntä'], df\_train\_validation['Ennuste']))

# Tehtävä 7



Ennusteen keskivirhe logistisella regressiolla on 0.881.

Ennusteen keskivirhe SVM luokittelulla on 0.966.

Ennusteen keskivirhe K-Lähimmännaapurin luokittelulla on 0.983.

## Lähdekoodi

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from sklearn import preprocessing

from sklearn import linear\_model

from sklearn.metrics import accuracy\_score

from sklearn.svm import SVC

from sklearn.neighbors import KNeighborsClassifier

df = pd.read\_csv('data/fruit\_data.csv', sep=',', encoding='utf-8')

X = np.array(df[['mass', 'width', 'height', 'color\_score']])

fruit\_codes = {'apple':0, 'lemon':1, 'mandarin':2, 'orange':3}

df['fruit\_code'] = df['fruit\_name'].map(fruit\_codes)

y = np.array(df['fruit\_code'])

scaler = preprocessing.StandardScaler()

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

# LOGISTIC REGRESSION

model = linear\_model.LogisticRegression(multi\_class='multinomial', solver='newton-cg')

model.fit(X\_scaled, y)

ennuste = model.predict(X\_scaled)

print(accuracy\_score(y, ennuste))

df['LRennuste'] = ennuste

# SUPPORT VECTOR CLASSIFIER

model = SVC()

model.fit(X\_scaled, y)

ennuste = model.predict(X\_scaled)

print(accuracy\_score(y, ennuste))

df['SVMennuste'] = ennuste

# SUPPORT VECTOR CLASSIFIER

model = KNeighborsClassifier()

model.fit(X\_scaled, y)

ennuste = model.predict(X\_scaled)

print(accuracy\_score(y, ennuste))

df['KNNennuste'] = ennuste

# Tehtävä 8

Ss

# Tehtävä 9

# Tehtävä 10

# Ss

# Tehtävä 20: Lähtevien asiakkaiden tunnistaminen

Oheisessa datassa (Telco.csv) on teleoperaattorin asiakastietokanta. Muuttuja churn=1 ilmaisee, että asiakas on lopettanut sopimuksen. Tehtäväsi on ennustaa kunkin asiakkaan ”churn”-kentän arvo käyttämällä input-muuttujina muiden kenttien arvoja. Jätä datasta 100 satunnaista asiakasta test-dataksi, ja vssalitse loput training-dataksi, jonka avulla muodostat haluamasi koneoppimismallin. Raportoi vastaukseesi seuraavat asiat:

• Käyttämäsi koneoppimismalli ja sen tarkkuus training- ja test-datassa.

• Huolehdi siitä, ettei malli ole liikaa ylisovitettu.

• Listaus 20 satunnaisesta test-datan asiakkaasta, jossa näkyy kunkin asiakkaan todellinen churn-arvo sekä mallisi ennuste (churn-riski numerona välillä 0-1)