R ja koneoppiminen

[Company name] | [Company address]

Asiantuntijaluennot

Bragge Janne TTM23SAI

[Year]

Sisällysluettelo

[Logistinen Regressio 3](#_Toc184468377)

[Datan valmistelu 3](#_Toc184468378)

[Datan visualisointi 4](#_Toc184468379)

[Mallin rakentaminen 5](#_Toc184468380)

[Ennusteet ja visualisointi 6](#_Toc184468381)

[Mallin arviointi 7](#_Toc184468382)

[Päätöspuu 8](#_Toc184468383)

[Datan valmistelu 8](#_Toc184468384)

[Datan jako treeni- ja testijoukkoihin 9](#_Toc184468385)

[Päätöspuun rakentaminen 10](#_Toc184468386)

[Päätöspuun visualisointi 11](#_Toc184468387)

[Ennusteiden tekeminen 12](#_Toc184468388)

[Mallin arviointi 12](#_Toc184468389)

[Visualisointi 13](#_Toc184468390)

[Kommentit ja jatkokehitys 13](#_Toc184468391)

[KNN 14](#_Toc184468392)

[Datan valmistelu 14](#_Toc184468393)

[Datan skaalaus 14](#_Toc184468394)

[Datan jako treeni- ja testijoukkoihin 16](#_Toc184468395)

[KNN-mallin rakentaminen 16](#_Toc184468396)

[Mallin arviointi 17](#_Toc184468397)

[Visualisointi 18](#_Toc184468398)

[Hyperparametrien säätäminen 19](#_Toc184468399)

[Arviointi 20](#_Toc184468400)

[K-Means 21](#_Toc184468401)

[Datan valmistelu 21](#_Toc184468402)

[Datan visualisointi 22](#_Toc184468403)

[Datan skaalaus 23](#_Toc184468404)

[Optimaalisen klusterimäärän valinta 24](#_Toc184468405)

[K-Means Klusterointi 25](#_Toc184468406)

[Klusteroinnin visualisointi 26](#_Toc184468407)

[Tulosten tulkinta 27](#_Toc184468408)

[Jatkokehitys 27](#_Toc184468409)

[Naive Bayes 28](#_Toc184468410)

[Datan valmistelu 28](#_Toc184468411)

[Datan jako treeni- ja testijoukkoihin 29](#_Toc184468412)

[Naive Bayes -mallin rakentaminen 30](#_Toc184468413)

[Ennusteiden tekeminen 31](#_Toc184468414)

[Mallin suorituskyvyn arviointi 32](#_Toc184468415)

[Visualisointi 33](#_Toc184468416)

[Tulosten tulkinta 34](#_Toc184468417)

[Jatkokehitys 34](#_Toc184468418)

# Logistinen Regressio

Tässä on yksinkertainen esimerkki lineaarisesta regressiosta R-kielellä, jossa selitetään vaihe vaiheelta, miten se toteutetaan. Projekti analysoi kuvitteellista dataa, jossa yritetään ennustaa myyntiä mainosbudjetin perusteella.

## Datan valmistelu

Ensiksi luodaan esimerkkidata, joka sisältää mainosbudjetin (Advertising\_Budget) ja myynnin (Sales).

# Luo esimerkkidata

set.seed(123) # Aseta satunnaissiementä toistettavuuden varmistamiseksi

data <- data.frame(

Advertising\_Budget = runif(50, 10, 100), # Satunnaisia mainosbudjetteja välillä 10-100

Sales = runif(50, 5, 50) # Satunnaisia myyntilukuja välillä 5-50

)

# Tarkastele dataa

head(data)

A number of numbers and a number of numbers

Description automatically generated with medium confidence

## Datan visualisointi

Piirretään scatterplot, jotta nähdään mahdollinen yhteys mainosbudjetin ja myynnin välillä.

# Asenna ggplot2, jos sitä ei ole

if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)

# Scatterplot

ggplot(data, aes(x = Advertising\_Budget, y = Sales)) +

geom\_point() +

labs(title = "Mainosbudjetti ja Myynti",

x = "Mainosbudjetti (€)",

y = "Myynti (€)")

A graph of black dots

Description automatically generated

## Mallin rakentaminen

Käytetään lm()-funktiota luodaksemme lineaarisen regressiomallin.

# Lineaarinen regressiomalli

model <- lm(Sales ~ Advertising\_Budget, data = data)

# Tulosta mallin yhteenveto

summary(model)

A screenshot of a computer

Description automatically generated

## Ennusteet ja visualisointi

Lisätään ennustettu trendiviiva scatterplottiin.

# Lisää mallin ennusteet dataan

data$Predicted\_Sales <- predict(model)

# Visualisoi malli

ggplot(data, aes(x = Advertising\_Budget, y = Sales)) +

geom\_point() +

geom\_line(aes(y = Predicted\_Sales), color = "blue") +

labs(title = "Lineaarinen Regressio: Mainosbudjetti ja Myynti",

x = "Mainosbudjetti (€)",

y = "Myynti (€)")

A graph with black dots and a line

Description automatically generated

## Mallin arviointi

Lasketaan mallin suorituskykymetriikoita, kuten R² ja RMSE.

# R-squared ja RMSE

r\_squared <- summary(model)$r.squared

rmse <- sqrt(mean((data$Sales - data$Predicted\_Sales)^2))

# Tulosta metriikat

cat("R²:", round(r\_squared, 3), "\n")

cat("RMSE:", round(rmse, 3), "\n")

A white rectangular object with black lines

Description automatically generated

# Päätöspuu

## Datan valmistelu

Luodaan esimerkkidata, kuten lineaarisen regressioesimerkin kohdalla.

# Luo esimerkkidata

set.seed(123)

data <- data.frame(

Advertising\_Budget = runif(50, 10, 100), # Satunnaisia mainosbudjetteja

Sales = runif(50, 5, 50) # Satunnaisia myyntilukuja

)

# Tarkastele dataa

head(data)

A screenshot of a computer

Description automatically generated

## Datan jako treeni- ja testijoukkoihin

Jaetaan data osiin mallin kouluttamista ja testaamista varten.

# Asenna caret-paketti, jos sitä ei ole

if (!require("caret")) install.packages("caret")

library(caret)

# Jaa data treeni- (70%) ja testijoukkoon (30%)

set.seed(123)

index <- createDataPartition(data$Sales, p = 0.7, list = FALSE)

train\_data <- data[index, ]

test\_data <- data[-index, ]

A close-up of a computer screen

Description automatically generated

## Päätöspuun rakentaminen

Käytetään rpart-pakettia päätöspuun luomiseen.

# Asenna rpart-paketti, jos sitä ei ole

if (!require("rpart")) install.packages("rpart")

library(rpart)

# Luo päätöspuumalli

tree\_model <- rpart(Sales ~ Advertising\_Budget, data = train\_data, method = "anova")

# Tulosta puun yhteenveto

summary(tree\_model)

A computer code with numbers and letters

Description automatically generated with medium confidence

## Päätöspuun visualisointi

Visualisoidaan päätöspuu, jotta ymmärretään sen rakenne.

# Asenna rpart.plot-paketti, jos sitä ei ole

if (!require("rpart.plot")) install.packages("rpart.plot")

library(rpart.plot)

# Piirrä päätöspuu

rpart.plot(tree\_model, type = 3, fallen.leaves = TRUE, main = "Päätöspuu")

A screenshot of a computer

Description automatically generated

## Ennusteiden tekeminen

Käytetään testidataa ennusteiden tekemiseen.

# Tee ennusteet testidatalle

test\_data$Predicted\_Sales <- predict(tree\_model, newdata = test\_data)

# Tarkastele testidatan ennusteita

head(test\_data)

A screenshot of a computer

Description automatically generated

## Mallin arviointi

Arvioidaan päätöspuumallin suorituskykyä metriikoilla, kuten RMSE.

# Laske RMSE

rmse <- sqrt(mean((test\_data$Sales - test\_data$Predicted\_Sales)^2))

# Tulosta RMSE

cat("RMSE päätöspuulle:", round(rmse, 3), "\n")

A close-up of a computer code

Description automatically generated

## Visualisointi

Visualisoidaan ennusteet vs. toteutuneet arvot.

# Scatterplot todellisten ja ennustettujen arvojen välillä

if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)

ggplot(test\_data, aes(x = Sales, y = Predicted\_Sales)) +

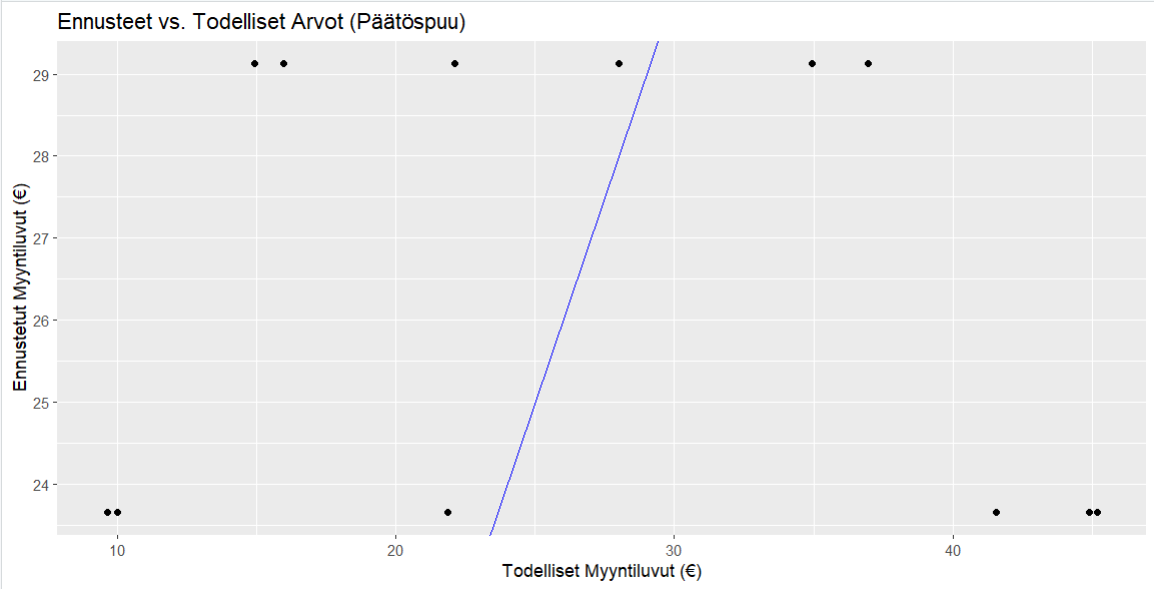
geom\_point() +

geom\_abline(slope = 1, intercept = 0, color = "blue") +

labs(title = "Ennusteet vs. Todelliset Arvot (Päätöspuu)",

x = "Todelliset Myyntiluvut (€)",

y = "Ennustetut Myyntiluvut (€)")



## Kommentit ja jatkokehitys

1. **Päätöspuun selitys**: Päätöspuu jakaa datan yksinkertaisiin sääntöihin perustuen ja toimii hyvin, kun data sisältää epälineaarisia yhteyksiä.
2. **Jatkokehitys**: Voit kokeilla muita menetelmiä, kuten satunnaismetsiä (random forest), mikäli päätöspuun suorituskyky ei ole riittävä.
3. **Tuning**: Päätöspuun monimutkaisuutta voi säätää rpart-funktiossa esimerkiksi cp-parametrilla (complexity parameter).

KNN

## Datan valmistelu

Luodaan esimerkkidata, jossa on kaksi muuttujaa: Advertising\_Budget ja Sales.

# Luo esimerkkidata

set.seed(123)

data <- data.frame(

Advertising\_Budget = runif(100, 10, 100), # Mainosbudjetti välillä 10-100

Sales = runif(100, 5, 50) # Myynti välillä 5-50

)

# Tarkastele dataa

head(data)

A white text with black numbers

Description automatically generated with medium confidence

## Datan skaalaus

KNN-menetelmä on herkkä muuttujien mittayksiköille, joten data on hyvä skaalata.

# Skaalaa muuttujat välillä 0-1

if (!require("caret")) install.packages("caret")

library(caret)

# Skaalaus

scaled\_data <- data

scaled\_data[, 1:2] <- scale(data[, 1:2])

# Tarkasta skaalattu data

head(scaled\_data)

A screenshot of a computer code

Description automatically generated

## Datan jako treeni- ja testijoukkoihin

Jaetaan data treeni- ja testijoukkoihin (70% treeni, 30% testi).

# Jaa data

set.seed(123)

index <- createDataPartition(scaled\_data$Sales, p = 0.7, list = FALSE)

train\_data <- scaled\_data[index, ]

test\_data <- scaled\_data[-index, ]

## KNN-mallin rakentaminen

Käytetään kknn-pakettia KNN-mallin rakentamiseen.

# Asenna kknn-paketti, jos sitä ei ole

if (!require("kknn")) install.packages("kknn")

library(kknn)

# Rakenna KNN-malli

knn\_model <- kknn(Sales ~ Advertising\_Budget,

train = train\_data,

test = test\_data,

k = 5) # Käytetään 5 lähintä naapuria

# Ennusteet testidatalle

test\_data$Predicted\_Sales <- knn\_model$fitted.values

## Mallin arviointi

Lasketaan KNN-mallin suorituskykymetriikat, kuten RMSE.

# RMSE (Root Mean Squared Error)

rmse <- sqrt(mean((test\_data$Sales - test\_data$Predicted\_Sales)^2))

# Tulosta RMSE

cat("RMSE KNN-mallille:", round(rmse, 3), "\n")

A computer code with blue text

Description automatically generated

## Visualisointi

Visualisoidaan ennusteiden ja toteutuneiden arvojen välinen suhde.

if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)

# Scatterplot: Todelliset vs. Ennustetut arvot

ggplot(test\_data, aes(x = Sales, y = Predicted\_Sales)) +

geom\_point(color = "blue") +

geom\_abline(slope = 1, intercept = 0, color = "red", linetype = "dashed") +

labs(title = "KNN: Todelliset vs. Ennustetut Myyntiarvot",

x = "Todelliset Myyntiarvot",

y = "Ennustetut Myyntiarvot")

A graph with a red line

Description automatically generated

## Hyperparametrien säätäminen

KNN-mallin suorituskyky riippuu suuresti lähimpien naapureiden lukumäärästä (k). Voimme etsiä parhaan arvon käyttämällä esimerkiksi ristiinvalidaatiota.

# Kokeile eri k-arvoja ja valitse paras

k\_values <- 1:20

rmse\_values <- sapply(k\_values, function(k) {

model <- kknn(Sales ~ Advertising\_Budget, train = train\_data, test = test\_data, k = k)

A graph with numbers and lines

Description automatically generated

sqrt(mean((test\_data$Sales - model$fitted.values)^2))

})

# Parhaan k-arvon valinta

best\_k <- k\_values[which.min(rmse\_values)]

cat("Paras k:", best\_k, "\n")

# Visualisoi k:n vaikutus RMSE:hen

plot(k\_values, rmse\_values, type = "b", main = "k-arvon vaikutus RMSE:hen",

xlab = "k (Lähimpien naapureiden lukumäärä)", ylab = "RMSE")

## Arviointi

1. **KNN-menetelmä** on tehokas, mutta se vaatii datan skaalauksen ja sopivan k:n valinnan.
2. **Visualisoinnit** auttavat ymmärtämään, kuinka hyvin malli toimii.
3. **Jatkokehitys**: Voit lisätä muuttujia tai kokeilla muita ennustemalleja, kuten regressiota tai päätöspuita, ja vertailla niiden tuloksia.

# K-Means

## Datan valmistelu

Luodaan esimerkkidata, jossa on kaksi muuttujaa: Advertising\_Budget ja Sales.

# Luo esimerkkidata

set.seed(123)

data <- data.frame(

Advertising\_Budget = runif(100, 10, 100), # Mainosbudjetti välillä 10-100

Sales = runif(100, 5, 50) # Myynti välillä 5-50

)

# Tarkastele dataa

head(data)

A screenshot of a computer screen

Description automatically generated

## Datan visualisointi

Visualisoidaan data, jotta nähdään, miten asiakkaat jakautuvat.

if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)

# Scatterplot

ggplot(data, aes(x = Advertising\_Budget, y = Sales)) +

geom\_point() +

labs(title = "Asiakastiedot: Mainosbudjetti ja Myynti",

x = "Mainosbudjetti (€)",

y = "Myynti (€)")

A graph with black dots

Description automatically generated

## Datan skaalaus

K-Means on etäisyyksiin perustuva menetelmä, joten muuttujat on hyvä skaalata ennen klusterointia.

# Skaalaa muuttujat

data\_scaled <- scale(data)

# Tarkastele skaalattua dataa

head(data\_scaled)

A screenshot of a computer code

Description automatically generated

## Optimaalisen klusterimäärän valinta

Käytetään **"Elbow Method"** -menetelmää optimaalisen klusterimäärän arvioimiseen.

# Laske WCSS (Within-Cluster Sum of Squares) eri klusterimäärille

wcss <- sapply(1:10, function(k) {

kmeans(data\_scaled, centers = k, nstart = 25)$tot.withinss

})

# Visualisoi WCSS

plot(1:10, wcss, type = "b", pch = 19, frame = FALSE,

xlab = "Klusterien määrä",

ylab = "WCSS (Within-Cluster Sum of Squares)",

main = "Elbow Method")

Optimaalinen klusterimäärä valitaan kohdasta, jossa WCSS:n lasku hidastuu ("kyynärpää").

A graph of a line graph

Description automatically generated with medium confidence

## K-Means Klusterointi

Valitaan esimerkiksi **3 klusteria** ja suoritetaan K-Means.

# Suorita K-Means klusterointi

set.seed(123)

kmeans\_model <- kmeans(data\_scaled, centers = 3, nstart = 25)

# Lisää klusterit alkuperäiseen dataan

data$Cluster <- as.factor(kmeans\_model$cluster)

# Tarkastele klusteroitua dataa

head(data)

A screenshot of a computer code

Description automatically generated

## Klusteroinnin visualisointi

Visualisoidaan klusterit eri väreillä.

# Scatterplot klustereille

ggplot(data, aes(x = Advertising\_Budget, y = Sales, color = Cluster)) +

geom\_point(size = 3) +

labs(title = "K-Means Klusterointi: Asiakassegmentit",

x = "Mainosbudjetti (€)",

y = "Myynti (€)") +

scale\_color\_manual(values = c("red", "blue", "green"))

A graph with many colored dots

Description automatically generated

## Tulosten tulkinta

* **Klusterit**: Jokainen klusteri edustaa ryhmää, jolla on samanlaisia ominaisuuksia. Esimerkiksi:
  + Klusteri 1: Asiakkaat, joilla on pieni mainosbudjetti ja pieni myynti.
  + Klusteri 2: Asiakkaat, joilla on suuri mainosbudjetti ja korkea myynti.
  + Klusteri 3: Asiakkaat, joilla on keskitasoinen budjetti ja myynti.
* **Sovellus**: Klusterien perusteella yritys voi kohdentaa markkinointistrategioita eri asiakasryhmille.

## Jatkokehitys

* **Hyperparametrien säätäminen**: Testaa erilaisia klusterimääriä ja valitse optimaalinen.
* **Lisää muuttujia**: Voit lisätä muita muuttujia, kuten asiakastyytyväisyyden tai tuoteryhmän.
* **Vertailu muihin klusterointimenetelmiin**: Kokeile esim. hierarkkista klusterointia.

# Naive Bayes

## Datan valmistelu

Luodaan esimerkkidata, jossa asiakkaat jaetaan kahteen ryhmään (High ja Low) myyntilukujen perusteella.

# Luo esimerkkidata

set.seed(123)

data <- data.frame(

Advertising\_Budget = runif(100, 10, 100), # Mainosbudjetti välillä 10-100

Sales = runif(100, 5, 50) # Myynti välillä 5-50

)

# Luo luokkamuuttuja (High tai Low myynnin perusteella)

data$Category <- ifelse(data$Sales > median(data$Sales), "High", "Low")

data$Category <- as.factor(data$Category)

# Tarkastele dataa

head(data)

A screenshot of a computer

Description automatically generated

## Datan jako treeni- ja testijoukkoihin

Jaetaan data treeni- (70%) ja testijoukkoon (30%).

if (!require("caret")) install.packages("caret")

library(caret)

# Jaa data

set.seed(123)

index <- createDataPartition(data$Category, p = 0.7, list = FALSE)

train\_data <- data[index, ]

test\_data <- data[-index, ]



## Naive Bayes -mallin rakentaminen

Käytetään **e1071**-pakettia Naive Bayes -mallin rakentamiseen.

# Asenna e1071-paketti, jos sitä ei ole

if (!require("e1071")) install.packages("e1071")

library(e1071)

# Rakenna Naive Bayes -malli

nb\_model <- naiveBayes(Category ~ Advertising\_Budget + Sales, data = train\_data)

# Tarkastele mallia

print(nb\_model)

A screenshot of a computer

Description automatically generated

## Ennusteiden tekeminen

Käytetään testijoukkoa ennusteiden tekemiseen ja lisätään ennusteet dataan.

# Tee ennusteet testidatalle

test\_data$Predicted\_Category <- predict(nb\_model, test\_data)

# Tarkastele ennusteita

head(test\_data)

A screenshot of a computer

Description automatically generated

## Mallin suorituskyvyn arviointi

Arvioidaan mallin tarkkuutta vertaamalla todellisia luokkia ennustettuihin.

# Confusion Matrix

conf\_matrix <- confusionMatrix(test\_data$Predicted\_Category, test\_data$Category)

# Tulosta tulokset

print(conf\_matrix)

A screenshot of a computer

Description automatically generated

## Visualisointi

Visualisoidaan, miten luokat jakautuvat päätösrajalla.

# Scatterplot ennustetuille luokille

if (!require("ggplot2")) install.packages("ggplot2")

library(ggplot2)

ggplot(test\_data, aes(x = Advertising\_Budget, y = Sales, color = Predicted\_Category)) +

geom\_point(size = 3) +

labs(title = "Naive Bayes: Luokittelu Asiakasryhmiin",

x = "Mainosbudjetti (€)",

y = "Myynti (€)") +

scale\_color\_manual(values = c("red", "blue"))

A graph with red and blue dots

Description automatically generated

## Tulosten tulkinta

1. **Confusion Matrix**:
   * Näyttää mallin tarkkuuden, herkkyyden ja spesifisyyden.
   * Esim. "Accuracy" kertoo, kuinka suuri osa ennusteista on oikein.
2. **Visualisointi**:
   * Visualisointi auttaa näkemään, miten mainosbudjetti ja myynti vaikuttavat asiakkaan kategoriaan.

## Jatkokehitys

* **Hyperparametrien säätö**: Säädä mallin parametreja, kuten prioreja, parantaaksesi suorituskykyä.
* **Lisää muuttujia**: Kokeile lisätä datan muita piirteitä, kuten asiakastyytyväisyys tai alue.
* **Vertaa muihin malleihin**: Kokeile esim. logistista regressiota tai SVM:ää ja vertaa tuloksia.