DSC_520_week10_Assignment00

Reenie Christudass

2022-08-13

Contents

Load libraries as needed

```
if(!require('factoextra')) {
  install.packages("factoextra", repos="http://cran.us.r-project.org")
  library('factoextra')
}
```

```
## Loading required package: factoextra

## Warning: package 'factoextra' was built under R version 4.2.1

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.2.1

## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
if(!require('cluster')) {
  install.packages("cluster", repos="http://cran.us.r-project.org")
  library('cluster')
}
```

```
## Loading required package: cluster

## Warning: package 'cluster' was built under R version 4.2.1
```

```
if(!require('NbClust')) {
  install.packages("NbClust", repos="http://cran.us.r-project.org")
  library('NbClust')
}
```

```
## Loading required package: NbClust
```

```
if(!require('e1071')) {
  install.packages("e1071", repos="http://cran.us.r-project.org")
  library('e1071')
}
```

```
## Loading required package: e1071

## Warning: package 'e1071' was built under R version 4.2.1
```

```
if(!require('caTools')) {
  install.packages("caTools", repos="http://cran.us.r-project.org")
  library('caTools')
}
```

```
## Loading required package: caTools

## Warning: package 'caTools' was built under R version 4.2.1
```

```
if(!require('class')) {
  install.packages("class", repos="http://cran.us.r-project.org")
  library('class')
}
```

```
## Loading required package: class
```

```
## Warning: package 'class' was built under R version 4.2.1
```

```
if(!require('dplyr')) {
  install.packages("dplyr", repos="http://cran.us.r-project.org")
  library('dplyr')
}
```

```
## Loading required package: dplyr
```

```
## Warning: package 'dplyr' was built under R version 4.2.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
rm(list = ls())
## Set the working directory to the root of your DSC 520 directory
setwd("C:/Users/chris/dsc520/data")
```

Read the binary-classifier-data

```
df_binary <- read.csv("C:/Users/chris/dsc520/data/binary-classifier-data.csv")
head(df_binary)
```

```
##   label        x        y
## 1     0 70.88469 83.17702
## 2     0 74.97176 87.92922
## 3     0 73.78333 92.20325
## 4     0 66.40747 81.10617
## 5     0 69.07399 84.53739
## 6     0 72.23616 86.38403
```

```
df_binary_copy <- df_binary
```

Euclidean distance of binary-classifier-data

```
#calculate Euclidean distance between columns a and d
euclidean <- function(x, y) sqrt(sum((x - y)^2))

print("Euclidean distance (binary-classifier-data) between two columns x and y in a dataframe is: ")
```

```
## [1] "Euclidean distance (binary-classifier-data) between two columns x and y in a dataframe is: "
```

```
euclidean(df_binary$x, df_binary$y)
```

```
## [1] 1411.959
```

Convert y to be a factor

```
is.factor(df_binary$y)
```

```
## [1] FALSE
```

```
df_binary$y <- as.factor(df_binary$y)
head(df_binary)
```

```
##   label        x                y
## 1     0 70.88469 83.1770155691132
## 2     0 74.97176 87.9292174061897
## 3     0 73.78333 92.2032543690917
## 4     0 66.40747 81.1061732248732
## 5     0 69.07399  84.537386045772
## 6     0 72.23616 86.3840256689234
```

Build the model for binary-classifier-data

```
binary_model <- glm(y~x, data=df_binary,family="binomial")
summary (binary_model)
```

```
##
## Call:
## glm(formula = y ~ x, family = "binomial", data = df_binary)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8229   0.0352   0.0367   0.0380   0.0396
```

```
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) 7.166938   1.783161   4.019 5.84e-05 ***
## x           0.003303   0.034893   0.095    0.925
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 16.623  on 1497  degrees of freedom
## Residual deviance: 16.614  on 1496  degrees of freedom
## AIC: 20.614
##
## Number of Fisher Scoring iterations: 10
```

```
#abline(binary_model, col='blue')
```

Predict using model

```
df_binary$Predicted <- predict(binary_model, df_binary, type='response')
head(df_binary$Predicted )
```

```
## [1] 0.9993898 0.9993980 0.9993956 0.9993807 0.9993861 0.9993925
```

convert predicted values back to true/false

```
df_binary$Predicted_T_F  <- ifelse (df_binary$Predicted >= 0.5, "true", "false")
head(df_binary)
```

```
##   label        x                y Predicted Predicted_T_F
## 1     0 70.88469 83.1770155691132 0.9993898          true
## 2     0 74.97176 87.9292174061897 0.9993980          true
## 3     0 73.78333 92.2032543690917 0.9993956          true
## 4     0 66.40747 81.1061732248732 0.9993807          true
## 5     0 69.07399  84.537386045772 0.9993861          true
## 6     0 72.23616 86.3840256689234 0.9993925          true
```

Determine accuracy of the model

```
accuracy <- mean(df_binary$Predicted == df_binary$y)
print(accuracy)
```

```
## [1] 0
```

```
## ACCURACY is 0
```

Plot

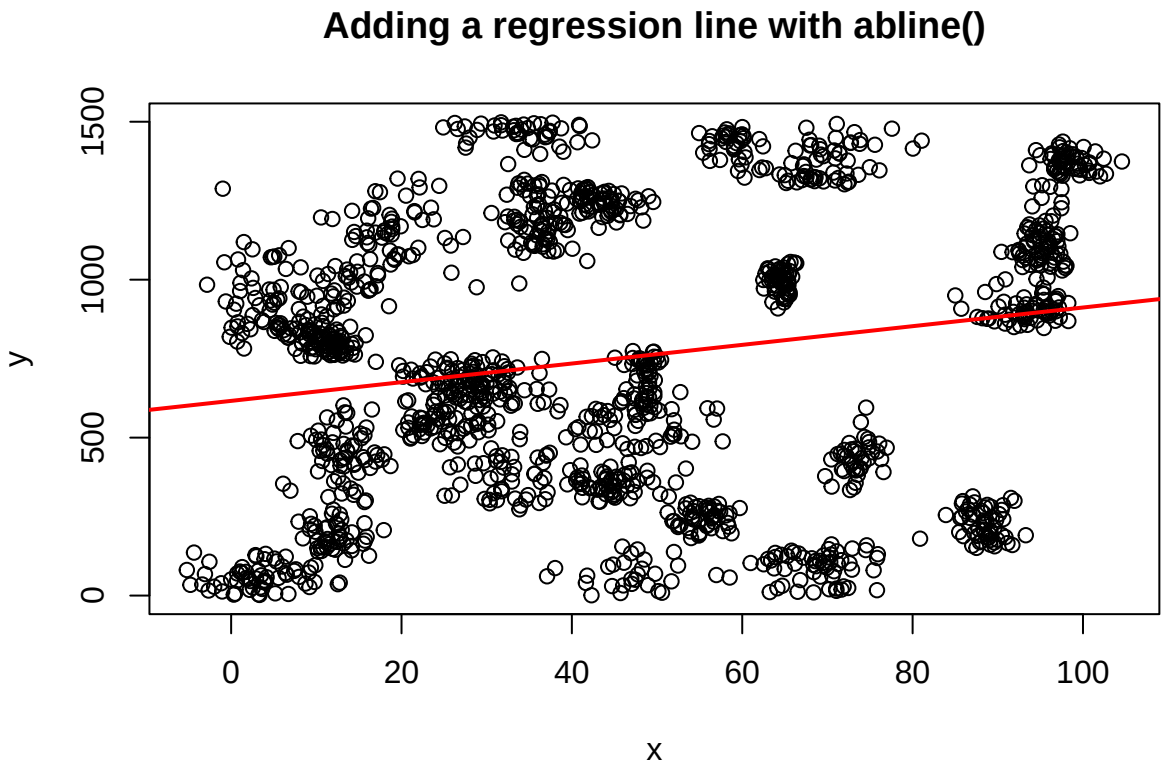
```
# Scatterplot
plot(x = df_binary$x,
     y = df_binary$y,
     xlab = "x",
     ylab = "y",
     main = "Adding a regression line with abline()"
     )

# Calculate regression model
model <- lm(formula = y ~ x,
                  data = df_binary)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
# Add regression line
abline(model,
       col = "red", lwd = 2)
```



Read the trinary-classifier-data

```
df_trinary <- read.csv("C:/Users/chris/dsc520/data/trinary-classifier-data.csv")
head(df_trinary)
```

```
##   label        x        y
## 1     0 30.08387 39.63094
## 2     0 31.27613 51.77511
## 3     0 34.12138 49.27575
## 4     0 32.58222 41.23300
## 5     0 34.65069 45.47956
## 6     0 33.80513 44.24656
```

Euclidean distance of trinary-classifier-data

```
#calculate Euclidean distance between columns a and d
euclidean <- function(x, y) sqrt(sum((x - y)^2))

print("Euclidean distance (trinary-classifier-data) between two columns x and y in a dataframe is: ")
```

```
## [1] "Euclidean distance (trinary-classifier-data) between two columns x and y in a dataframe is: "
```

```
euclidean(df_trinary$x, df_trinary$y)
```

```
## [1] 1357.734
```

Convert y to be a factor

```
is.factor(df_trinary$y)
```

```
## [1] FALSE
```

```
df_trinary$y <- as.factor(df_trinary$y)
head(df_trinary)
```

```
##   label        x               y
## 1     0 30.08387 39.6309356367601
## 2     0 31.27613 51.7751073869153
## 3     0 34.12138 49.2757480671664
## 4     0 32.58222  41.232996587068
## 5     0 34.65069  45.479561540635
## 6     0 33.80513  44.246560844561
```

Build the model for binary-classifier-data

```
trinary_model <- glm(y~x, data=df_trinary,family="binomial")
summary (trinary_model)
```

```
##
## Call:
## glm(formula = y ~ x, family = "binomial", data = df_trinary)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.8220   0.0313   0.0341   0.0386   0.0498
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  7.99960    2.34864   3.406 0.000659 ***
## x           -0.01206    0.03680  -0.328 0.743109
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 16.714  on 1567  degrees of freedom
## Residual deviance: 16.607  on 1566  degrees of freedom
## AIC: 20.607
##
## Number of Fisher Scoring iterations: 10
```

```
#abline(binary_model, col='blue')
```

Predict using model

```
df_trinary$Predicted <- predict(trinary_model, df_trinary, type='response')
head(df_trinary)
```

```
##   label        x                y Predicted
## 1     0 30.08387 39.6309356367601 0.9995178
## 2     0 31.27613 51.7751073869153 0.9995109
## 3     0 34.12138 49.2757480671664 0.9994938
## 4     0 32.58222  41.232996587068 0.9995031
## 5     0 34.65069  45.479561540635 0.9994906
## 6     0 33.80513  44.246560844561 0.9994957
```

convert predicted values back to true/false

```
df_trinary$predict_T_F <- ifelse (df_trinary$Predicted >= 0.5, "true", "false")
head(df_trinary)
```

```
##   label        x                y Predicted predict_T_F
```

```
## 1     0 30.08387 39.6309356367601 0.9995178        true
## 2     0 31.27613 51.7751073869153 0.9995109        true
## 3     0 34.12138 49.2757480671664 0.9994938        true
## 4     0 32.58222  41.232996587068 0.9995031        true
## 5     0 34.65069  45.479561540635 0.9994906        true
## 6     0 33.80513  44.246560844561 0.9994957        true
```

Determine accuracy of the model

```
accuracy <- mean(df_trinary$y == df_trinary$Predicted)
print(accuracy)
```

```
## [1] 0
```

```
## ACCURACY is 0
```

Plot

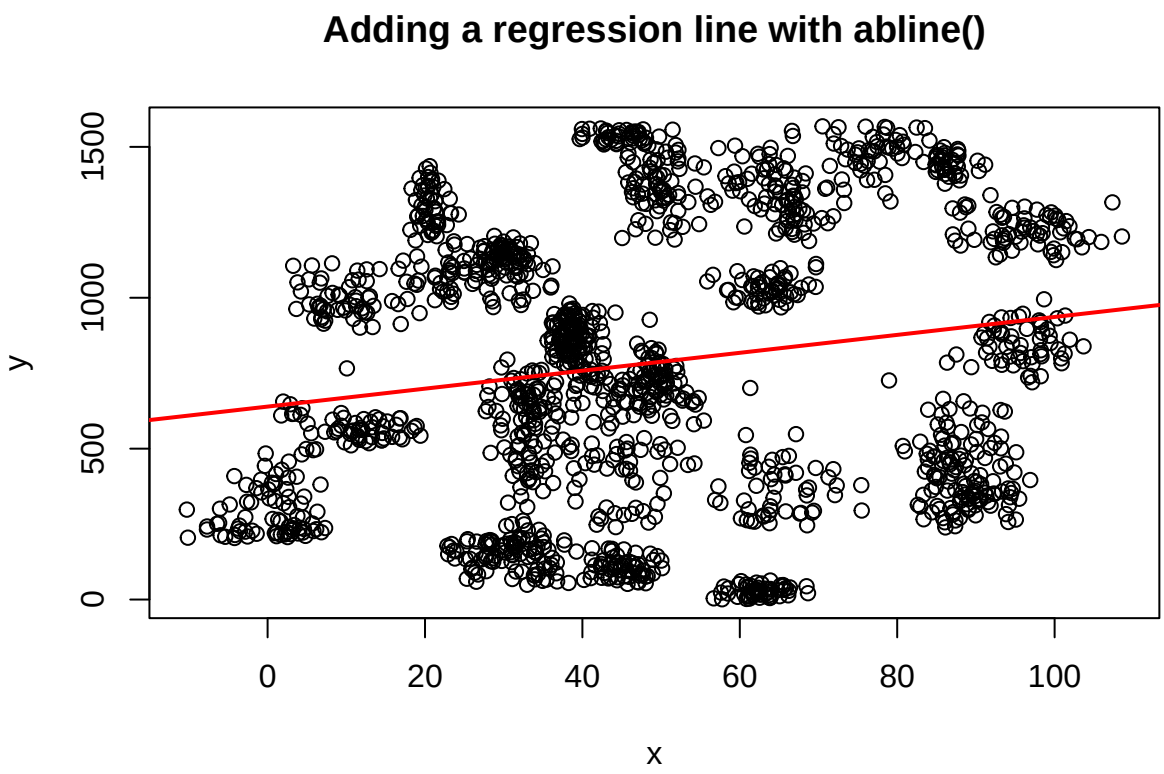
```
# Scatterplot
plot(x = df_trinary$x,
     y = df_trinary$y,
     xlab = "x",
     ylab = "y",
     main = "Adding a regression line with abline()"
     )

# Calculate regression model
model <- lm(formula = y ~ x,
                  data = df_trinary)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored
```

```
## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
# Add regression line
abline(model,
       col = "red", lwd = 2)
```


Adding a regression line with abline()
y
x
0
500
1000
1500
0
20
40
60
80
100