cirrhosis-prediction

Erru Torculas

2023-12-12

Cirrhosis Prediction

Cirrhosis is a late stage of scarring (fibrosis) of the liver caused by many forms of liver diseases and conditions, such as hepatitis and chronic alcoholism. The following data contains the information collected from the Mayo Clinic trial in primary biliary cirrhosis (PBC) of the liver conducted between 1974 and 1984. A description of the clinical background for the trial and the covariates recorded here is in Chapter 0, especially Section 0.2 of Fleming and Harrington, Counting Processes and Survival Analysis, Wiley, 1991. A more extended discussion can be found in Dickson, et al., Hepatology 10:1-7 (1989) and in Markus, et al., N Eng J of Med 320:1709-13 (1989).

A total of 424 PBC patients, referred to Mayo Clinic during that ten-year interval, met eligibility criteria for the randomized placebo-controlled trial of the drug D-penicillamine. The first 312 cases in the dataset participated in the randomized trial and contain largely complete data. The additional 112 cases did not participate in the clinical trial but consented to have basic measurements recorded and to be followed for survival. Six of those cases were lost to follow-up shortly after diagnosis, so the data here are on an additional 106 cases as well as the 312 randomized participants.

Objectives

1. To build a model that will classify images in detecting the Liver Cirrhosis using Liver Ultrasound Data.

2. To compare or trade off four algorithms such as Logistic Regression (LR), Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Networks (CNN) that is suitable in in detecting the Liver Cirrhosis using Liver Ultrasound Data.

3. To test the accuracy of four algorithms such as Logistic Regression (LR), Support Vector Machine (SVM), Multilayer Perceptron (MLP), and Convolutional Neural Networks (CNN) that will be fit in to detect Liver Cirrhosis Detection using Liver Ultrasound Data

1 - Packages

```r
#install.packages("dplyr")

library(dplyr)
```

```
##
## 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
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(gplots)
```

```
##
## Attaching package: 'gplots'
```

```
## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(tidyverse)
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats   1.0.0     ✔ readr     2.1.4
## ✔ ggplot2   3.4.4     ✔ stringr   1.5.1
## ✔ lubridate 1.9.3     ✔ tibble    3.2.1
## ✔ purrr     1.0.2     ✔ tidyr     1.3.0
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflic
ts to become errors
```

```
library(finalfit)
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:ggplot2':
##
##     margin
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(ggplot2)
library(e1071)
library(RSNNS)
```

```
## Loading required package: Rcpp
##
## Attaching package: 'RSNNS'
##
## The following objects are masked from 'package:caret':
##
##     confusionMatrix, train
```

```
library(tensorflow)
```

```
##
## Attaching package: 'tensorflow'
##
## The following object is masked from 'package:RSNNS':
##
##     train
##
## The following object is masked from 'package:caret':
##
##     train
```

```
library(keras)
library(smotefamily)

library(reticulate)
use_python("/usr/local/bin/python3")
```

2 - Data Preparation & EDA

```
cirrhosis_data <- read.csv("input/cirrhosis.csv")

print(head(cirrhosis_data))
```

```
##   ID N_Days Status            Drug   Age Sex Ascites Hepatomegaly Spiders Edema
## 1  1    400      D D-penicillamine 21464   F       Y            Y       Y     Y
## 2  2   4500      C D-penicillamine 20617   F       N            Y       Y     N
## 3  3   1012      D D-penicillamine 25594   M       N            N       N     S
## 4  4   1925      D D-penicillamine 19994   F       N            Y       Y     S
## 5  5   1504     CL         Placebo 13918   F       N            Y       Y     N
## 6  6   2503      D         Placebo 24201   F       N            Y       N     N
##   Bilirubin Cholesterol Albumin Copper Alk_Phos   SGOT Tryglicerides Platelets
## 1      14.5         261    2.60    156   1718.0 137.95           172       190
## 2       1.1         302    4.14     54   7394.8 113.52            88       221
## 3       1.4         176    3.48    210    516.0  96.10            55       151
## 4       1.8         244    2.54     64   6121.8  60.63            92       183
## 5       3.4         279    3.53    143    671.0 113.15            72       136
## 6       0.8         248    3.98     50    944.0  93.00            63        NA
##   Prothrombin Stage
## 1        12.2     4
## 2        10.6     3
## 3        12.0     4
## 4        10.3     4
## 5        10.9     3
## 6        11.0     3
```

```
cirrhosis_data <- cirrhosis_data %>%
  select(-ID)
```

```
str(cirrhosis_data)
```

```
## 'data.frame':    418 obs. of  19 variables:
##  $ N_Days       : int  400 4500 1012 1925 1504 2503 1832 2466 2400 51 ...
##  $ Status       : chr  "D" "C" "D" "D" ...
##  $ Drug         : chr  "D-penicillamine" "D-penicillamine" "D-penicillamine" "D-pe
nicillamine" ...
##  $ Age          : int  21464 20617 25594 19994 13918 24201 20284 19379 15526 25772
...
##  $ Sex          : chr  "F" "F" "M" "F" ...
##  $ Ascites      : chr  "Y" "N" "N" "N" ...
##  $ Hepatomegaly : chr  "Y" "Y" "N" "Y" ...
##  $ Spiders      : chr  "Y" "Y" "N" "Y" ...
##  $ Edema        : chr  "Y" "N" "S" "S" ...
##  $ Bilirubin    : num  14.5 1.1 1.4 1.8 3.4 0.8 1 0.3 3.2 12.6 ...
##  $ Cholesterol  : int  261 302 176 244 279 248 322 280 562 200 ...
##  $ Albumin      : num  2.6 4.14 3.48 2.54 3.53 3.98 4.09 4 3.08 2.74 ...
##  $ Copper       : int  156 54 210 64 143 50 52 52 79 140 ...
##  $ Alk_Phos     : num  1718 7395 516 6122 671 ...
##  $ SGOT         : num  137.9 113.5 96.1 60.6 113.2 ...
##  $ Tryglicerides: int  172 88 55 92 72 63 213 189 88 143 ...
##  $ Platelets    : int  190 221 151 183 136 NA 204 373 251 302 ...
##  $ Prothrombin  : num  12.2 10.6 12 10.3 10.9 11 9.7 11 11 11.5 ...
##  $ Stage        : int  4 3 4 4 3 3 3 3 2 4 ...
```

```
summary(cirrhosis_data)
```

```
##      N_Days        Status              Drug                Age
##  Min.   :  41   Length:418         Length:418         Min.   : 9598
##  1st Qu.:1093   Class :character   Class :character   1st Qu.:15644
##  Median :1730   Mode  :character   Mode  :character   Median :18628
##  Mean   :1918                                         Mean   :18533
##  3rd Qu.:2614                                         3rd Qu.:21272
##  Max.   :4795                                         Max.   :28650
##
##      Sex              Ascites          Hepatomegaly         Spiders
##  Length:418         Length:418         Length:418         Length:418
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##     Edema             Bilirubin       Cholesterol        Albumin
##  Length:418         Min.   : 0.300   Min.   : 120.0   Min.   :1.960
##  Class :character   1st Qu.: 0.800   1st Qu.: 249.5   1st Qu.:3.243
##  Mode  :character   Median : 1.400   Median : 309.5   Median :3.530
##                     Mean   : 3.221   Mean   : 369.5   Mean   :3.497
##                     3rd Qu.: 3.400   3rd Qu.: 400.0   3rd Qu.:3.770
##                     Max.   :28.000   Max.   :1775.0   Max.   :4.640
##                                      NA's   :134
##      Copper          Alk_Phos            SGOT        Tryglicerides
##  Min.   :  4.00   Min.   :  289.0   Min.   : 26.35   Min.   : 33.00
##  1st Qu.: 41.25   1st Qu.:  871.5   1st Qu.: 80.60   1st Qu.: 84.25
##  Median : 73.00   Median : 1259.0   Median :114.70   Median :108.00
##  Mean   : 97.65   Mean   : 1982.7   Mean   :122.56   Mean   :124.70
##  3rd Qu.:123.00   3rd Qu.: 1980.0   3rd Qu.:151.90   3rd Qu.:151.00
##  Max.   :588.00   Max.   :13862.4   Max.   :457.25   Max.   :598.00
##  NA's   :108      NA's   :106       NA's   :106      NA's   :136
##    Platelets      Prothrombin        Stage
##  Min.   : 62.0   Min.   : 9.00   Min.   :1.000
##  1st Qu.:188.5   1st Qu.:10.00   1st Qu.:2.000
##  Median :251.0   Median :10.60   Median :3.000
##  Mean   :257.0   Mean   :10.73   Mean   :3.024
##  3rd Qu.:318.0   3rd Qu.:11.10   3rd Qu.:4.000
##  Max.   :721.0   Max.   :18.00   Max.   :4.000
##  NA's   :11      NA's   :2       NA's   :6
```

Handling Missing Values

```
# drop the 6 rows with missing 'Stage'
cirrhosis_data <- cirrhosis_data[complete.cases(cirrhosis_data$Stage), ]

# Numerical columns --> Median
# Impute missing values in numerical columns with the median.
numerical_columns <- sapply(cirrhosis_data, is.numeric)

for (c in names(cirrhosis_data[, numerical_columns])) {
  cirrhosis_data[, c] <- ifelse(is.na(cirrhosis_data[, c]), median(cirrhosis_data[,
c], na.rm = TRUE), cirrhosis_data[, c])
}

# Categorical columns --> Most Frequent
# Impute missing values in categorical columns with the most frequent value
categorical_columns <- sapply(cirrhosis_data, is.factor)

for (c in names(cirrhosis_data[, categorical_columns])) {
  cirrhosis_data[, c] <- ifelse(is.na(cirrhosis_data[, c]), levels(cirrhosis_data[,
c])[which.max(table(cirrhosis_data[, c]))], cirrhosis_data[, c])
}

# Convert 'Stage' to integer
cirrhosis_data$Stage <- as.integer(cirrhosis_data$Stage)

cirrhosis_data <- na.omit(cirrhosis_data)
```

Missing Values

```
summary(cirrhosis_data)
```

```
##      N_Days         Status              Drug                Age       
##  Min.   :  41   Length:312         Length:312         Min.   : 9598  
##  1st Qu.:1191   Class :character   Class :character   1st Qu.:15428  
##  Median :1840   Mode  :character   Mode  :character   Median :18188  
##  Mean   :2006                                         Mean   :18269  
##  3rd Qu.:2697                                         3rd Qu.:20715  
##  Max.   :4556                                         Max.   :28650  
##      Sex              Ascites          Hepatomegaly         Spiders         
##  Length:312         Length:312         Length:312         Length:312        
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                             
##                                                                             
##                                                                             
##     Edema             Bilirubin       Cholesterol        Albumin     
##  Length:312         Min.   : 0.300   Min.   : 120.0   Min.   :1.96  
##  Class :character   1st Qu.: 0.800   1st Qu.: 255.8   1st Qu.:3.31  
##  Mode  :character   Median : 1.350   Median : 309.5   Median :3.55  
##                     Mean   : 3.256   Mean   : 364.1   Mean   :3.52  
##                     3rd Qu.: 3.425   3rd Qu.: 392.2   3rd Qu.:3.80  
##                     Max.   :28.000   Max.   :1775.0   Max.   :4.64  
##      Copper          Alk_Phos            SGOT        Tryglicerides  
##  Min.   :  4.00   Min.   :  289.0   Min.   : 26.35   Min.   : 33.0  
##  1st Qu.: 41.75   1st Qu.:  871.5   1st Qu.: 80.60   1st Qu.: 87.0  
##  Median : 73.00   Median : 1259.0   Median :114.70   Median :108.0  
##  Mean   : 97.49   Mean   : 1982.7   Mean   :122.56   Mean   :123.1  
##  3rd Qu.:123.00   3rd Qu.: 1980.0   3rd Qu.:151.90   3rd Qu.:145.2  
##  Max.   :588.00   Max.   :13862.4   Max.   :457.25   Max.   :598.0  
##    Platelets      Prothrombin        Stage      
##  Min.   : 62.0   Min.   : 9.00   Min.   :1.000  
##  1st Qu.:200.0   1st Qu.:10.00   1st Qu.:2.000  
##  Median :255.5   Median :10.60   Median :3.000  
##  Mean   :261.8   Mean   :10.73   Mean   :3.032  
##  3rd Qu.:322.0   3rd Qu.:11.10   3rd Qu.:4.000  
##  Max.   :563.0   Max.   :17.10   Max.   :4.000  
```

Recode and Mutate Data

```
cirrhosis_data <- cirrhosis_data %>%
  mutate(
    Sex = recode(Sex, 'M' = 0, 'F' = 1),
    Ascites = recode(Ascites, 'N' = 0, 'Y' = 1),
    Drug = recode(Drug, 'D-penicillamine' = 0, 'Placebo' = 1),
    Hepatomegaly = recode(Hepatomegaly, 'N' = 0, 'Y' = 1),
    Spiders = recode(Spiders, 'N' = 0, 'Y' = 1),
    Edema = recode(Edema, 'N' = 0, 'Y' = 1, 'S' = -1),
    Status = recode(Status, 'C' = 0, 'CL' = 1, 'D' = -1),
  )
```

```
str(cirrhosis_data)
```

```
## 'data.frame':    312 obs. of  19 variables:
##  $ N_Days       : int  400 4500 1012 1925 1504 2503 1832 2466 2400 51 ...
##  $ Status       : num  -1 0 -1 -1 1 -1 0 -1 -1 -1 ...
##  $ Drug         : num  0 0 0 0 1 1 1 1 0 1 ...
##  $ Age          : int  21464 20617 25594 19994 13918 24201 20284 19379 15526 25772
...
##  $ Sex          : num  1 1 0 1 1 1 1 1 1 1 ...
##  $ Ascites      : num  1 0 0 0 0 0 0 0 0 1 ...
##  $ Hepatomegaly : num  1 1 0 1 1 1 1 0 0 0 ...
##  $ Spiders      : num  1 1 0 1 1 0 0 0 1 1 ...
##  $ Edema        : num  1 0 -1 -1 0 0 0 0 0 1 ...
##  $ Bilirubin    : num  14.5 1.1 1.4 1.8 3.4 0.8 1 0.3 3.2 12.6 ...
##  $ Cholesterol  : num  261 302 176 244 279 248 322 280 562 200 ...
##  $ Albumin      : num  2.6 4.14 3.48 2.54 3.53 3.98 4.09 4 3.08 2.74 ...
##  $ Copper       : num  156 54 210 64 143 50 52 52 79 140 ...
##  $ Alk_Phos     : num  1718 7395 516 6122 671 ...
##  $ SGOT         : num  137.9 113.5 96.1 60.6 113.2 ...
##  $ Tryglicerides: num  172 88 55 92 72 63 213 189 88 143 ...
##  $ Platelets    : int  190 221 151 183 136 249 204 373 251 302 ...
##  $ Prothrombin  : num  12.2 10.6 12 10.3 10.9 11 9.7 11 11 11.5 ...
##  $ Stage        : int  4 3 4 4 3 3 3 3 2 4 ...
##  - attr(*, "na.action")= 'omit' Named int [1:100] 313 314 315 316 317 318 319 320
321 322 ...
##   ..- attr(*, "names")= chr [1:100] "314" "315" "316" "318" ...
```

3 - Preprocessing Data

```
#Upsampling
balancedData <- SMOTE(
  X = cirrhosis_data,
  target = cirrhosis_data$Stage,
  K = 3)

#Standardize
scaledData <- balancedData$data
```

```
X <- subset(scaledData, select = -c(Status, N_Days, Stage, class))
y <- scaledData$Stage

X_matrix <- as.matrix(sapply(X, as.numeric))
```

```
#Splitting the data 80/20
set.seed(23)
split <- createDataPartition(scaledData$Stage, p = 0.8, list = FALSE)
X_train <- X_matrix[split, ]
X_test <- X_matrix[-split, ]
y_train <- y[split]
y_test <- y[-split]
```

4 - Modelling

Random Forest

```
rf_model <- randomForest(x = X_train, y = factor(y_train), ntree = 500)

# Make predictions on the test set
predictions <- predict(rf_model, newdata = X_test)

# Evaluate the model
conf_matrix <- table(predictions, y_test)
print(conf_matrix)
```

```
##            y_test
## predictions  1  2  3  4
##           1 58  4  3  0
##           2  0  1  0  2
##           3  0  7 15  6
##           4  0  1  6 13
```

```
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
print(paste("Accuracy:", round(accuracy, 4)))
```

```
## [1] "Accuracy: 0.75"
```

```
# Feature importance
importance(rf_model)
```

```
##                MeanDecreaseGini
## Drug                  14.185748
## Age                   22.565158
## Sex                    6.064978
## Ascites                1.796973
## Hepatomegaly          36.645801
## Spiders               10.495472
## Edema                  2.446834
## Bilirubin             35.717158
## Cholesterol           20.481980
## Albumin               28.290631
## Copper                18.491163
## Alk_Phos              21.426696
## SGOT                  24.696566
## Tryglicerides         25.902124
## Platelets             21.301023
## Prothrombin           21.171215
```

SVM

```
svm_model = svm(x = X_train,
                y = y_train,
                type = 'C-classification',
                kernel = 'linear')

# Make predictions on the test set
predictions <- predict(svm_model, newdata = X_test)

# Evaluate the model
conf_matrix <- table(predictions, y_test)
conf_matrix
```

```
##            y_test
## predictions  1  2  3  4
##           1 58  7  4  1
##           2  0  0  0  1
##           3  0  3 15  6
##           4  0  3  5 13
```

```
accuracy <- (sum(diag(conf_matrix)) / sum(conf_matrix))
print(paste("Accuracy:", round(accuracy, 4)))
```

```
## [1] "Accuracy: 0.7414"
```

```
# dataframe to matrix

matrixX_train <- as.matrix(X_train)
matrixy_train <- as.matrix(y_train)
matrixX_test <- as.matrix(X_test)
matrixy_test <- as.matrix(y_test)
```

MLP

```
model_mlp <- keras_model_sequential()
input_shape <- dim(X_train)[2]

model_mlp %>%
  layer_dense(units = 32, activation = 'relu', input_shape = input_shape) %>%
  layer_dense(units = 64, activation = 'relu') %>%
  layer_dense(units = 32, activation = 'relu') %>%
  layer_dense(units = 5, activation = 'softmax') %>%
  compile(
    optimizer = 'adam',
    loss = 'sparse_categorical_crossentropy',
    metrics = c('accuracy'))

summary(model_mlp)
```

```
## Model: "sequential"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  dense_3 (Dense)                    (None, 32)                      544
##  dense_2 (Dense)                    (None, 64)                      2112
##  dense_1 (Dense)                    (None, 32)                      2080
##  dense (Dense)                      (None, 5)                       165
## ================================================================================
## Total params: 4901 (19.14 KB)
## Trainable params: 4901 (19.14 KB)
## Non-trainable params: 0 (0.00 Byte)
## ________________________________________________________________________________
```

```
model_mlp %>% fit(matrixX_train, matrixy_train, epochs = 20, validation_data = list(m
atrixX_test, matrixy_test))
```

```
## Epoch 1/20
## 15/15 - 0s - loss: 378.8661 - accuracy: 0.3141 - val_loss: 223.1509 - val_accurac
y: 0.5086 - 483ms/epoch - 32ms/step
## Epoch 2/20
## 15/15 - 0s - loss: 95.0060 - accuracy: 0.3205 - val_loss: 70.0251 - val_accuracy:
0.3103 - 70ms/epoch - 5ms/step
## Epoch 3/20
## 15/15 - 0s - loss: 51.6143 - accuracy: 0.3803 - val_loss: 52.1501 - val_accuracy:
0.4052 - 49ms/epoch - 3ms/step
## Epoch 4/20
## 15/15 - 0s - loss: 52.1113 - accuracy: 0.3312 - val_loss: 73.0927 - val_accuracy:
0.1724 - 50ms/epoch - 3ms/step
## Epoch 5/20
## 15/15 - 0s - loss: 45.1129 - accuracy: 0.3803 - val_loss: 57.0884 - val_accuracy:
0.2931 - 50ms/epoch - 3ms/step
## Epoch 6/20
## 15/15 - 0s - loss: 23.6788 - accuracy: 0.3868 - val_loss: 35.1046 - val_accuracy:
0.5000 - 48ms/epoch - 3ms/step
## Epoch 7/20
## 15/15 - 0s - loss: 33.0452 - accuracy: 0.3226 - val_loss: 76.1884 - val_accuracy:
0.5000 - 53ms/epoch - 4ms/step
## Epoch 8/20
## 15/15 - 0s - loss: 43.3555 - accuracy: 0.3462 - val_loss: 33.5154 - val_accuracy:
0.5431 - 49ms/epoch - 3ms/step
## Epoch 9/20
## 15/15 - 0s - loss: 36.5277 - accuracy: 0.3675 - val_loss: 41.8665 - val_accuracy:
0.1379 - 46ms/epoch - 3ms/step
## Epoch 10/20
## 15/15 - 0s - loss: 19.0711 - accuracy: 0.4017 - val_loss: 35.9004 - val_accuracy:
0.3276 - 48ms/epoch - 3ms/step
## Epoch 11/20
## 15/15 - 0s - loss: 24.9486 - accuracy: 0.3462 - val_loss: 43.3852 - val_accuracy:
0.5000 - 48ms/epoch - 3ms/step
## Epoch 12/20
## 15/15 - 0s - loss: 28.1007 - accuracy: 0.3739 - val_loss: 29.1176 - val_accuracy:
0.2241 - 48ms/epoch - 3ms/step
## Epoch 13/20
## 15/15 - 0s - loss: 23.3510 - accuracy: 0.4038 - val_loss: 39.1374 - val_accuracy:
0.2328 - 45ms/epoch - 3ms/step
## Epoch 14/20
## 15/15 - 0s - loss: 35.9711 - accuracy: 0.3419 - val_loss: 48.1245 - val_accuracy:
0.2586 - 48ms/epoch - 3ms/step
## Epoch 15/20
## 15/15 - 0s - loss: 36.2496 - accuracy: 0.4017 - val_loss: 29.7412 - val_accuracy:
0.2414 - 46ms/epoch - 3ms/step
## Epoch 16/20
## 15/15 - 0s - loss: 27.6098 - accuracy: 0.3504 - val_loss: 53.3075 - val_accuracy:
0.4828 - 46ms/epoch - 3ms/step
## Epoch 17/20
## 15/15 - 0s - loss: 32.7499 - accuracy: 0.3419 - val_loss: 39.1943 - val_accuracy:
0.2500 - 43ms/epoch - 3ms/step
## Epoch 18/20
## 15/15 - 0s - loss: 36.5610 - accuracy: 0.3440 - val_loss: 82.3010 - val_accuracy:
0.2500 - 46ms/epoch - 3ms/step
## Epoch 19/20
## 15/15 - 0s - loss: 48.4172 - accuracy: 0.3632 - val_loss: 44.0441 - val_accuracy:
0.2500 - 46ms/epoch - 3ms/step
## Epoch 20/20
## 15/15 - 0s - loss: 32.2664 - accuracy: 0.3547 - val_loss: 20.2731 - val_accuracy:
0.2414 - 44ms/epoch - 3ms/step
```

```
score <- model_mlp %>% evaluate(matrixX_test, matrixy_test)
```

```
## 4/4 - 0s - loss: 20.2731 - accuracy: 0.2414 - 24ms/epoch - 6ms/step
```

```
score
```

```
##       loss   accuracy 
## 20.2731247  0.2413793
```

CNN

```
CNNmodel <- keras_model_sequential()

CNNmodel %>%
  layer_conv_1d(filters = 32, kernel_size = 3, padding = 'same', activation = 'relu',
input_shape = c(16, 1)) %>%
  layer_max_pooling_1d(pool_size = 2) %>%
  layer_conv_1d(filters = 64, kernel_size = 3, padding = 'same', activation = 'relu')
%>%
  layer_max_pooling_1d(pool_size = 2) %>%
  layer_conv_1d(filters = 32, kernel_size = 3, padding = 'same', activation = 'relu')
%>%
  layer_max_pooling_1d(pool_size = 2) %>%
  layer_dropout(0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 16, activation = 'relu') %>%
  layer_dense(units = 5, activation = 'softmax') %>%
  compile(
    loss = 'sparse_categorical_crossentropy',
    optimizer = 'adam',
    metrics = 'accuracy')
summary(CNNmodel)
```

```
## Model: "sequential_1"
## ________________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## ================================================================================
##  conv1d_2 (Conv1D)                  (None, 16, 32)                  128
##  max_pooling1d_2 (MaxPooling1D)     (None, 8, 32)                   0
##  conv1d_1 (Conv1D)                  (None, 8, 64)                   6208
##  max_pooling1d_1 (MaxPooling1D)     (None, 4, 64)                   0
##  conv1d (Conv1D)                    (None, 4, 32)                   6176
##  max_pooling1d (MaxPooling1D)       (None, 2, 32)                   0
##  dropout (Dropout)                  (None, 2, 32)                   0
##  flatten (Flatten)                  (None, 64)                      0
##  dense_5 (Dense)                    (None, 16)                      1040
##  dense_4 (Dense)                    (None, 5)                       85
## ================================================================================
## Total params: 13637 (53.27 KB)
## Trainable params: 13637 (53.27 KB)
## Non-trainable params: 0 (0.00 Byte)
## ________________________________________________________________________________
```

```
CNNmodel %>%
  fit(
    x = matrixX_train,
    y = matrixy_train,
    epochs = 20,
    validation_data = list(matrixX_test, matrixy_test))
```

```
## Epoch 1/20
## 15/15 - 1s - loss: 259.8183 - accuracy: 0.3141 - val_loss: 67.8598 - val_accuracy:
0.5000 - 629ms/epoch - 42ms/step
## Epoch 2/20
## 15/15 - 0s - loss: 90.4796 - accuracy: 0.3504 - val_loss: 41.9381 - val_accuracy:
0.5086 - 57ms/epoch - 4ms/step
## Epoch 3/20
## 15/15 - 0s - loss: 32.0528 - accuracy: 0.2842 - val_loss: 3.6813 - val_accuracy:
0.2069 - 59ms/epoch - 4ms/step
## Epoch 4/20
## 15/15 - 0s - loss: 12.0261 - accuracy: 0.2286 - val_loss: 2.2647 - val_accuracy:
0.5086 - 65ms/epoch - 4ms/step
## Epoch 5/20
## 15/15 - 0s - loss: 6.8257 - accuracy: 0.4359 - val_loss: 1.5914 - val_accuracy: 0.
5000 - 60ms/epoch - 4ms/step
## Epoch 6/20
## 15/15 - 0s - loss: 5.0420 - accuracy: 0.4359 - val_loss: 1.5841 - val_accuracy: 0.
5000 - 65ms/epoch - 4ms/step
## Epoch 7/20
## 15/15 - 0s - loss: 4.7129 - accuracy: 0.4316 - val_loss: 1.5770 - val_accuracy: 0.
5000 - 64ms/epoch - 4ms/step
## Epoch 8/20
## 15/15 - 0s - loss: 4.5016 - accuracy: 0.4103 - val_loss: 1.5685 - val_accuracy: 0.
5000 - 64ms/epoch - 4ms/step
## Epoch 9/20
## 15/15 - 0s - loss: 3.2632 - accuracy: 0.4167 - val_loss: 1.5613 - val_accuracy: 0.
5000 - 62ms/epoch - 4ms/step
## Epoch 10/20
## 15/15 - 0s - loss: 3.3154 - accuracy: 0.4231 - val_loss: 1.5533 - val_accuracy: 0.
5000 - 61ms/epoch - 4ms/step
## Epoch 11/20
## 15/15 - 0s - loss: 2.7117 - accuracy: 0.3996 - val_loss: 1.5457 - val_accuracy: 0.
5000 - 63ms/epoch - 4ms/step
## Epoch 12/20
## 15/15 - 0s - loss: 2.7840 - accuracy: 0.4637 - val_loss: 1.5472 - val_accuracy: 0.
5000 - 62ms/epoch - 4ms/step
## Epoch 13/20
## 15/15 - 0s - loss: 2.3919 - accuracy: 0.4658 - val_loss: 1.5318 - val_accuracy: 0.
5000 - 63ms/epoch - 4ms/step
```

```
## Epoch 14/20
## 15/15 - 0s - loss: 2.2709 - accuracy: 0.4615 - val_loss: 1.5246 - val_accuracy: 0.
5000 - 62ms/epoch - 4ms/step
## Epoch 15/20
## 15/15 - 0s - loss: 2.4526 - accuracy: 0.4722 - val_loss: 1.5172 - val_accuracy: 0.
5000 - 61ms/epoch - 4ms/step
## Epoch 16/20
## 15/15 - 0s - loss: 2.1170 - accuracy: 0.4509 - val_loss: 1.5100 - val_accuracy: 0.
5000 - 61ms/epoch - 4ms/step
## Epoch 17/20
## 15/15 - 0s - loss: 2.2062 - accuracy: 0.4466 - val_loss: 1.5031 - val_accuracy: 0.
5000 - 60ms/epoch - 4ms/step
## Epoch 18/20
## 15/15 - 0s - loss: 2.4031 - accuracy: 0.4615 - val_loss: 1.4962 - val_accuracy: 0.
5000 - 62ms/epoch - 4ms/step
## Epoch 19/20
## 15/15 - 0s - loss: 2.4706 - accuracy: 0.4637 - val_loss: 1.4901 - val_accuracy: 0.
5000 - 61ms/epoch - 4ms/step
## Epoch 20/20
## 15/15 - 0s - loss: 2.0937 - accuracy: 0.4658 - val_loss: 1.4841 - val_accuracy: 0.
5000 - 85ms/epoch - 6ms/step
```

```
CNNmodel %>% evaluate(matrixX_test, matrixy_test)
```

```
## 4/4 - 0s - loss: 1.4841 - accuracy: 0.5000 - 26ms/epoch - 6ms/step
```

```
##     loss accuracy
##  1.48409  0.50000
```

```
predictionsTest <- CNNmodel %>% evaluate(matrixX_test, matrixy_test)
```

```
## 4/4 - 0s - loss: 1.4841 - accuracy: 0.5000 - 22ms/epoch - 5ms/step
```

```
predictionsTest
```

```
##     loss accuracy
##  1.48409  0.50000
```