Data Science Final Project

MODEL 2: Network-based Classification Model.

Kjay O. Coca

2022-12-14

LOAD PACKAGES

```
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(keras)
library(tfruns)
library(rsample)
library(tfestimators)
```

```
## tfestimators is not recomended for new code. It is only compatible with Tensorflow version 1, and is
```

```
library(readr)
```

IMPORTING THE DATA

```
set.seed(123)
radiomics_data <- read_csv("D:/1 MASTERS/STAT225/FINAL PROJECT/STAT 325 _FINAL PROJECT_/Normalize Radio
```

```
## Rows: 197 Columns: 431
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (1): Institution
## dbl (430): Failure.binary, Failure, Entropy_cooc.W.ADC, GLNU_align.H.PET, Mi...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

SPLITTING FOR TRAINING AND TESTING

```
for_splitted  <-  sample(1:nrow(radiomics_data), round(nrow(radiomics_data) * 0.8))
radiomicsdata_train <- radiomics_data[for_splitted,]
radiomicsdata_test  <- radiomics_data[-for_splitted,]
```

In this case, I set 80 percent for training data and 20 percent for testing data. There are 39 observation for testing and 158 observation for training and both have 413 variables.

Set the X & Y Train and Test

```
X_train <- radiomicsdata_train[,-c(1,2)]%>%as.matrix.data.frame()
X_test <- radiomicsdata_test[,-c(1,2)]%>%as.matrix.data.frame()
y_train <- radiomicsdata_train$Failure.binary
y_test <- radiomicsdata_test$Failure.binary
```

Reshaping the dataset

```
X_train <- array_reshape(X_train, c(nrow(X_train), ncol(X_train)))
X_train <- X_train

X_test <- array_reshape(X_test, c(nrow(X_test), ncol(X_test)))
X_test <- X_test

y_train <- to_categorical(y_train, num_classes = 2)
```

```
## Loaded Tensorflow version 2.9.1
```

```
y_test <- to_categorical(y_test, num_classes = 2)
```

Five hidden layer

```
nbc_model <- keras_model_sequential() %>%
  layer_dense(units = 256, activation = "sigmoid", input_shape = c(ncol(X_train))) %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 128, activation = "sigmoid") %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 128, activation = "sigmoid") %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 64, activation = "sigmoid") %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 64, activation = "sigmoid") %>%
  layer_dropout(rate = 0.2) %>%
  layer_dense(units = 2, activation = "softmax")
```

Create five hidden layers with 256, 128, 128, 64 and 64 neurons, respectively with activation functions of Sigmoid. Create an output layer with two neurons respectively with activation functions of Softmax. Every layer is followed by a dropout to avoid overfitting.

Backpropagation

```
nbc_model %>%

compile(
    loss = "categorical_crossentropy",
    optimizer = optimizer_rmsprop(),
    metrics = c("accuracy")
  )
```

Compile the Model

```
 nbc_model %>% compile(
  loss = "categorical_crossentropy",
  optimizer = optimizer_adam(),
  metrics = c("accuracy")
)

history <- nbc_model %>%
  fit(X_train, y_train, epochs = 10, batch_size = 128, validation_split = 0.15)
```

Above is the training of the model with epoch = 10, batch size = 128 and validation split = 0.15. The accuracy is 0.6250 or 62.50 percent.

Evaluate the trained model

```
nbc_model %>%
  evaluate(X_test, y_test)
```

```
##      loss  accuracy
## 0.6939160 0.6153846
```

```
dim(X_test)
```

```
## [1]  39 429
```

```
dim(y_test)
```

```
## [1] 39  2
```

Upon evaluating the trained model using the testing dataset, the accuracy is 0.6154 or just 61.54 percent.

Model prediction

This are the model prediction.

```r
nbc_model   %>% predict(X_test) %>% `>`(0.8) %>% k_cast("int32")
```

```
## tf.Tensor(
## [[0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]
##  [0 0]], shape=(39, 2), dtype=int32)
```