Untitled

Jay Zhu

2022-12-06

Model 2 - DNN

```
# Helper packages
library(tensorflow)
library(keras)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice

##
## Attaching package: 'caret'

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

```
library(plyr)
```

Import data

```
radio_df <- read.csv("radiomics_completedata.csv")
radio_df$Institution <- revalue(radio_df$Institution, c(A = "1", B = "2", C="3", D="4"))
```

Split dataset

```
# Set seeds
seeds = set.seed(200)

#splitting the data into training and testing
index<-createDataPartition(radio_df$Failure.binary,p=0.7,list=F)

#Test labels in the Species column (column 5)
```

```
Train_Features <- data.matrix(radio_df[index,-2])
Train_Labels <- radio_df[index,2]

Train_Labels
```

```
##   [1] 1 1 0 0 0 1 1 1 1 0 1 0 1 1 1 1 0 0 1 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1
##  [38] 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0 1 1 0 0 0 1 0 1 0 0 0 1 0 0 0 0
##  [75] 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 1 0 1 0 0 1 1 1 0 1 0 0 1 0 0 0 1 1 0 0 0
## [112] 0 0 0 0 1 0 0 0 1 0 1 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 0 0
```

```
Test_Features <- data.matrix(radio_df[-index,-2])
Test_Labels <- radio_df[-index,2]
Test_Labels
```

```
##  [1] 0 0 1 1 1 0 1 1 0 1 0 0 0 0 0 0 0 0 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 1 1
## [39] 0 1 0 0 1 0 0 1 0 1 1 1 0 1 0 0 1 0 1 0 0
```

```
# Converting the labels into categorical
to_categorical(as.numeric(Train_Labels))[,c(-1)] -> Train_Labels
```

```
## Loaded Tensorflow version 2.9.2
```

```
to_categorical(as.numeric(Test_Labels))[,c(-1)] -> Test_Labels
```

Data structure

```
#Structure the dataaset
str(Train_Features)
```

```
##  num [1:138, 1:430] 1 1 1 1 1 1 1 1 1 1 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:138] "2" "4" "5" "7" ...
##   ..$ : chr [1:430] "Institution" "Failure" "Entropy_cooc.W.ADC" "GLNU_align.H.PET" ...
```

Create the matrix

```
#converting the features into matrix
as.matrix(apply(Train_Features, 2, function(x) (x-min(x))/(max(x) - min(x)))) -> Train_Features
as.matrix(apply(Test_Features, 2, function(x) (x-min(x))/(max(x) - min(x)))) -> Test_Features
```

Create the model

```r
#model training
model <- keras_model_sequential() %>%
  layer_dense(units = 256, activation = "sigmoid", input_shape= ncol(Train_Features)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 128, activation = "sigmoid") %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 128, activation = "sigmoid") %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 64, activation = "sigmoid") %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 64, activation = "sigmoid") %>%
  layer_dropout(rate = 0.25) %>%
  layer_dense(units = 2, activation = "softmax") %>%

  compile(
    loss = "sparse_categorical_crossentropy",
    optimizer = optimizer_rmsprop(),
    metrics = c('accuracy')
  )

summary(model)
```

```
## Model: "sequential"
## _________________________________________________________________________
##  Layer (type)                       Output Shape                    Param #
## =========================================================================
##  dense_5 (Dense)                    (None, 256)                     110336
##  dropout_4 (Dropout)                (None, 256)                     0
##  dense_4 (Dense)                    (None, 128)                     32896
##  dropout_3 (Dropout)                (None, 128)                     0
##  dense_3 (Dense)                    (None, 128)                     16512
##  dropout_2 (Dropout)                (None, 128)                     0
##  dense_2 (Dense)                    (None, 64)                      8256
##  dropout_1 (Dropout)                (None, 64)                      0
##  dense_1 (Dense)                    (None, 64)                      4160
##  dropout (Dropout)                  (None, 64)                      0
##  dense (Dense)                      (None, 2)                       130
## =========================================================================
## Total params: 172,290
## Trainable params: 172,290
## Non-trainable params: 0
## _________________________________________________________________________
```

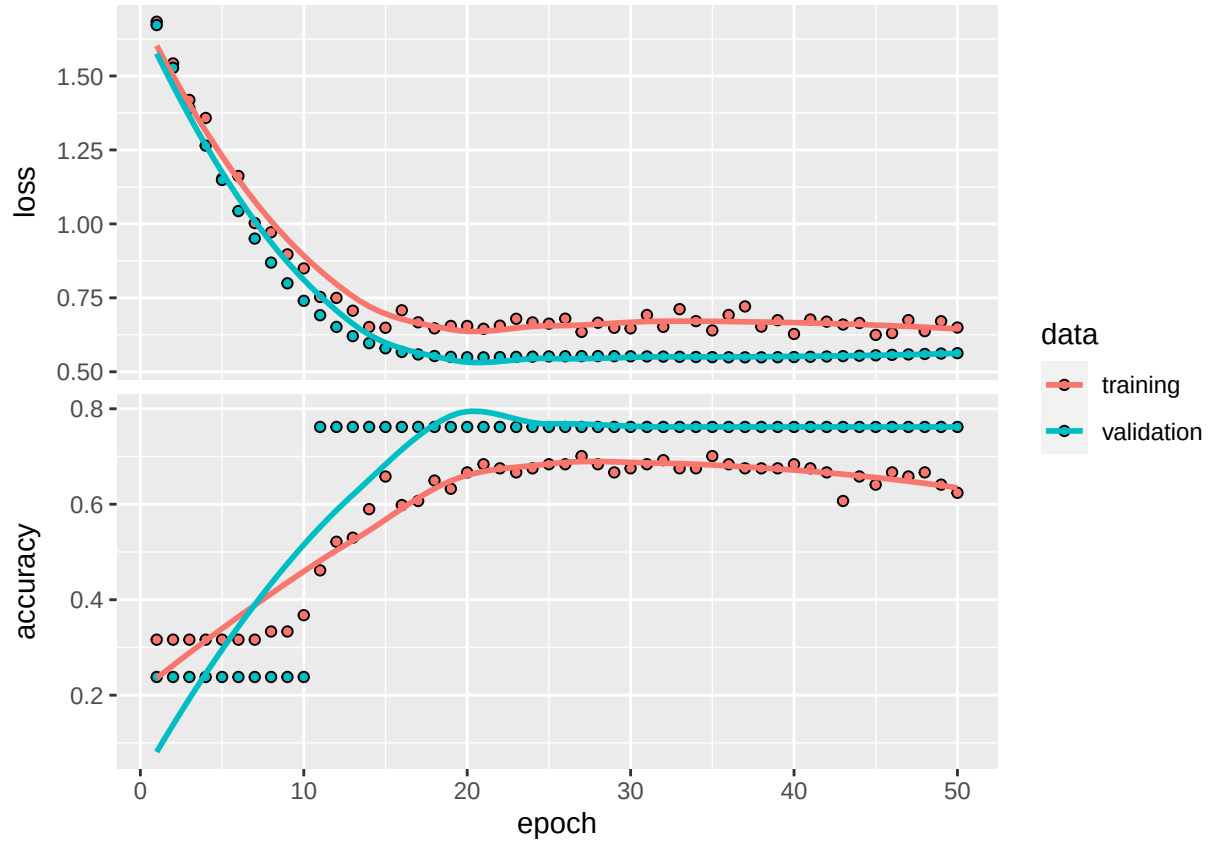
Model compiling

```r
model %>% compile(
  loss = "sparse_categorical_crossentropy",
  optimizer = optimizer_adam(),
  metrics = c("accuracy")
)
```

#trained model history

```
history <- model %>%
  fit(Train_Features, Train_Labels, epochs = 50, batch_size = 128, validation_split = 0.15)

plot(history)
```



#model evaluation

```
model %>%
  evaluate(Test_Features, Test_Labels)
```

```
##      loss  accuracy
## 0.7079821 0.5762712
```

#model prediction

```
model %>% predict(Test_Features)
```

```
##            [,1]      [,2]
##  [1,] 0.6859435 0.3140565
##  [2,] 0.6859446 0.3140554
##  [3,] 0.6859085 0.3140915
##  [4,] 0.6859536 0.3140464
##  [5,] 0.6859142 0.3140858
##  [6,] 0.6859336 0.3140664
```

```
##  [7,] 0.6859210 0.3140790
##  [8,] 0.6859062 0.3140937
##  [9,] 0.6859336 0.3140664
## [10,] 0.6859159 0.3140841
## [11,] 0.6859540 0.3140460
## [12,] 0.6859496 0.3140504
## [13,] 0.6859389 0.3140611
## [14,] 0.6859417 0.3140583
## [15,] 0.6859531 0.3140469
## [16,] 0.6859525 0.3140475
## [17,] 0.6859844 0.3140156
## [18,] 0.6859428 0.3140572
## [19,] 0.6859145 0.3140855
## [20,] 0.6860054 0.3139946
## [21,] 0.6858994 0.3141006
## [22,] 0.6859259 0.3140741
## [23,] 0.6859409 0.3140591
## [24,] 0.6859040 0.3140960
## [25,] 0.6858804 0.3141197
## [26,] 0.6859009 0.3140991
## [27,] 0.6859632 0.3140368
## [28,] 0.6859370 0.3140630
## [29,] 0.6859456 0.3140544
## [30,] 0.6859379 0.3140621
## [31,] 0.6859195 0.3140804
## [32,] 0.6857887 0.3142113
## [33,] 0.6857521 0.3142479
## [34,] 0.6857660 0.3142340
## [35,] 0.6857595 0.3142404
## [36,] 0.6858144 0.3141855
## [37,] 0.6858751 0.3141249
## [38,] 0.6858475 0.3141525
## [39,] 0.6858795 0.3141205
## [40,] 0.6857926 0.3142075
## [41,] 0.6858017 0.3141983
## [42,] 0.6858179 0.3141821
## [43,] 0.6857367 0.3142633
## [44,] 0.6860642 0.3139357
## [45,] 0.6861153 0.3138847
## [46,] 0.6860570 0.3139430
## [47,] 0.6861143 0.3138857
## [48,] 0.6860547 0.3139452
## [49,] 0.6860523 0.3139478
## [50,] 0.6860242 0.3139757
## [51,] 0.6860902 0.3139097
## [52,] 0.6860783 0.3139217
## [53,] 0.6861038 0.3138962
## [54,] 0.6860935 0.3139065
## [55,] 0.6860797 0.3139203
## [56,] 0.6860369 0.3139631
## [57,] 0.6860878 0.3139122
## [58,] 0.6860718 0.3139282
## [59,] 0.6860722 0.3139277
```