Initial Practice: Parkinson's Disease

Statistical Learning with Deep Artificial Neural Networks

Alexander J Ohrt

22 februar, 2022

Contents

1 Introduction

We are working on a dataset describing Parkinson's disease. Click here for more information regarding the dataset. In short, the dataset is composed of a range of biomedical voice measurements from 42 people with early-stage Parkinson's disease recruited to a six-month trial of a telemonitoring device for remote symptom progression monitoring.

The main objective is to predict the severity of Parkinson's disease based on the data. More details are given in the following.

2 Load the Parkinsons Data

```
data <- read.csv("parkinsons_updrs.data")
str(data)
```

```
#> 'data.frame':    5875 obs. of  22 variables:
#>  $ subject.     : int  1 1 1 1 1 1 1 1 1 1 ...
#>  $ age          : int  72 72 72 72 72 72 72 72 72 72 ...
#>  $ sex          : int  0 0 0 0 0 0 0 0 0 0 ...
#>  $ test_time    : num  5.64 12.67 19.68 25.65 33.64 ...
#>  $ motor_UPDRS  : num  28.2 28.4 28.7 28.9 29.2 ...
#>  $ total_UPDRS  : num  34.4 34.9 35.4 35.8 36.4 ...
#>  $ Jitter...    : num  0.00662 0.003 0.00481 0.00528 0.00335 0.00353 0.00422 0.00476 0.00432 0.00496
#>  $ Jitter.Abs.  : num  3.38e-05 1.68e-05 2.46e-05 2.66e-05 2.01e-05 ...
#>  $ Jitter.RAP   : num  0.00401 0.00132 0.00205 0.00191 0.00093 0.00119 0.00212 0.00226 0.00156 0.002
#>  $ Jitter.PPQ5  : num  0.00317 0.0015 0.00208 0.00264 0.0013 0.00159 0.00221 0.00259 0.00207 0.00253
#>  $ Jitter.DDP   : num  0.01204 0.00395 0.00616 0.00573 0.00278 ...
#>  $ Shimmer      : num  0.0256 0.0202 0.0168 0.0231 0.017 ...
#>  $ Shimmer.dB.  : num  0.23 0.179 0.181 0.327 0.176 0.214 0.445 0.212 0.371 0.31 ...
#>  $ Shimmer.APQ3 : num  0.01438 0.00994 0.00734 0.01106 0.00679 ...
#>  $ Shimmer.APQ5 : num  0.01309 0.01072 0.00844 0.01265 0.00929 ...
#>  $ Shimmer.APQ11: num  0.0166 0.0169 0.0146 0.0196 0.0182 ...
#>  $ Shimmer.DDA  : num  0.0431 0.0298 0.022 0.0332 0.0204 ...
#>  $ NHR          : num  0.0143 0.0111 0.0202 0.0278 0.0116 ...
#>  $ HNR          : num  21.6 27.2 23 24.4 26.1 ...
#>  $ RPDE         : num  0.419 0.435 0.462 0.487 0.472 ...
#>  $ DFA          : num  0.548 0.565 0.544 0.578 0.561 ...
#>  $ PPE          : num  0.16 0.108 0.21 0.333 0.194 ...
```

```
summary(data)
```

```
#>     subject.          age            sex           test_time
#>  Min.   : 1.00   Min.   :36.0   Min.   :0.0000   Min.   : -4.263
#>  1st Qu.:10.00   1st Qu.:58.0   1st Qu.:0.0000   1st Qu.: 46.847
#>  Median :22.00   Median :65.0   Median :0.0000   Median : 91.523
#>  Mean   :21.49   Mean   :64.8   Mean   :0.3178   Mean   : 92.864
#>  3rd Qu.:33.00   3rd Qu.:72.0   3rd Qu.:1.0000   3rd Qu.:138.445
#>  Max.   :42.00   Max.   :85.0   Max.   :1.0000   Max.   :215.490
#>   motor_UPDRS      total_UPDRS      Jitter...         Jitter.Abs.
#>  Min.   : 5.038   Min.   : 7.00   Min.   :0.000830   Min.   :2.250e-06
#>  1st Qu.:15.000   1st Qu.:21.37   1st Qu.:0.003580   1st Qu.:2.244e-05
#>  Median :20.871   Median :27.58   Median :0.004900   Median :3.453e-05
#>  Mean   :21.296   Mean   :29.02   Mean   :0.006154   Mean   :4.403e-05
#>  3rd Qu.:27.596   3rd Qu.:36.40   3rd Qu.:0.006800   3rd Qu.:5.333e-05
#>  Max.   :39.511   Max.   :54.99   Max.   :0.099990   Max.   :4.456e-04
```

```
#>    Jitter.RAP        Jitter.PPQ5         Jitter.DDP          Shimmer
#>  Min.   :0.000330   Min.   :0.000430   Min.   :0.000980   Min.   :0.00306
#>  1st Qu.:0.001580   1st Qu.:0.001820   1st Qu.:0.004730   1st Qu.:0.01912
#>  Median :0.002250   Median :0.002490   Median :0.006750   Median :0.02751
#>  Mean   :0.002987   Mean   :0.003277   Mean   :0.008962   Mean   :0.03404
#>  3rd Qu.:0.003290   3rd Qu.:0.003460   3rd Qu.:0.009870   3rd Qu.:0.03975
#>  Max.   :0.057540   Max.   :0.069560   Max.   :0.172630   Max.   :0.26863
#>   Shimmer.dB.     Shimmer.APQ3      Shimmer.APQ5     Shimmer.APQ11
#>  Min.   :0.026   Min.   :0.00161   Min.   :0.00194   Min.   :0.00249
#>  1st Qu.:0.175   1st Qu.:0.00928   1st Qu.:0.01079   1st Qu.:0.01566
#>  Median :0.253   Median :0.01370   Median :0.01594   Median :0.02271
#>  Mean   :0.311   Mean   :0.01716   Mean   :0.02014   Mean   :0.02748
#>  3rd Qu.:0.365   3rd Qu.:0.02057   3rd Qu.:0.02375   3rd Qu.:0.03272
#>  Max.   :2.107   Max.   :0.16267   Max.   :0.16702   Max.   :0.27546
#>   Shimmer.DDA           NHR                HNR              RPDE
#>  Min.   :0.00484   Min.   :0.000286   Min.   : 1.659   Min.   :0.1510
#>  1st Qu.:0.02783   1st Qu.:0.010955   1st Qu.:19.406   1st Qu.:0.4698
#>  Median :0.04111   Median :0.018448   Median :21.920   Median :0.5423
#>  Mean   :0.05147   Mean   :0.032120   Mean   :21.680   Mean   :0.5415
#>  3rd Qu.:0.06173   3rd Qu.:0.031463   3rd Qu.:24.444   3rd Qu.:0.6140
#>  Max.   :0.48802   Max.   :0.748260   Max.   :37.875   Max.   :0.9661
#>       DFA              PPE
#>  Min.   :0.5140   Min.   :0.02198
#>  1st Qu.:0.5962   1st Qu.:0.15634
#>  Median :0.6436   Median :0.20550
#>  Mean   :0.6532   Mean   :0.21959
#>  3rd Qu.:0.7113   3rd Qu.:0.26449
#>  Max.   :0.8656   Max.   :0.73173
```

3 Description of the Variables

As we have seen above, the dataset contains 5875 rows, i.e. 5875 measurements. The columns consist of **patient ID**, **age**, **sex**, **time interval since enrollment date**, **motor_UPDRS**, **total_UPDRS** and 16 voice biomedical measurements. The variables are the following

- subject. - The patient ID. Integer that uniquely identifies each subject.
- age - Age of each subject.
- sex - Gender of the subject; '0' = male and '1' = female.
- test_time - Time since recruitment into the trial. The integer part is the number of days since recruitment.
- motor_UPDRS - Clinician's motor UPDRS score, linearly interpolated.
- ctotal_UPDRS - Clinician's total UPDRS score, linearly interpolated.
- Jitter(%), Jitter(Abs), Jitter:RAP, Jitter:PPQ5, Jitter:DDP - Several measures of variation in fundamental frequency.
- Shimmer, Shimmer(dB), Shimmer:APQ3, Shimmer:APQ5, Shimmer:APQ11, Shimmer:DDA - Several measures of variation in amplitude.
- NHR, HNR - Two measures of ratio of noise to tonal components in the voice.
- RPDE - A nonlinear dynamical complexity measure.
- DFA - Signal fractal scaling exponent.
- PPE - A nonlinear measure of fundamental frequency variation.

As noted, the objective is to predict the severity of the disease, where severity is defined based on the variable **total_UPDRS**: The disease is severe if **total_UPDRS > 25**. This variable is created below.

BESKRIV LITT MER KRING HVA DE 16 BIOMEDISINSKE VARIABLENE ER (IKKE BARE KOPIERE

FRA NETTSIDEN SLIK JEG HAR GJORT OVER!)

4 Create the Binary Variable of Parkinson's Severity

```
data$severity <- data$total_UPDRS > 25
dim(data)
```

```
#> [1] 5875   23
```

```
summary(data$severity)
```

```
#>    Mode   FALSE    TRUE
#> logical    2188    3687
```

5 Normalization

The variables of the 16 voice measurements are normalized by means of the min-max transformation.

```
normalize <- function(x) {
    return((x- min(x))/(max(x)-min(x)))
}

for (i in 1:16){
  data[, 6+i] <- normalize(data[,6+i])
}

summary(data)
```

```
#>     subject.          age             sex           test_time
#>  Min.   : 1.00   Min.   :36.0   Min.   :0.0000   Min.   : -4.263
#>  1st Qu.:10.00   1st Qu.:58.0   1st Qu.:0.0000   1st Qu.: 46.847
#>  Median :22.00   Median :65.0   Median :0.0000   Median : 91.523
#>  Mean   :21.49   Mean   :64.8   Mean   :0.3178   Mean   : 92.864
#>  3rd Qu.:33.00   3rd Qu.:72.0   3rd Qu.:1.0000   3rd Qu.:138.445
#>  Max.   :42.00   Max.   :85.0   Max.   :1.0000   Max.   :215.490
#>   motor_UPDRS      total_UPDRS      Jitter...         Jitter.Abs.
#>  Min.   : 5.038   Min.   : 7.00   Min.   :0.00000   Min.   :0.00000
#>  1st Qu.:15.000   1st Qu.:21.37   1st Qu.:0.02773   1st Qu.:0.04553
#>  Median :20.871   Median :27.58   Median :0.04104   Median :0.07281
#>  Mean   :21.296   Mean   :29.02   Mean   :0.05369   Mean   :0.09423
#>  3rd Qu.:27.596   3rd Qu.:36.40   3rd Qu.:0.06021   3rd Qu.:0.11523
#>  Max.   :39.511   Max.   :54.99   Max.   :1.00000   Max.   :1.00000
#>    Jitter.RAP       Jitter.PPQ5        Jitter.DDP         Shimmer
#>  Min.   :0.00000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
#>  1st Qu.:0.02185   1st Qu.:0.02011   1st Qu.:0.02185   1st Qu.:0.06047
#>  Median :0.03356   Median :0.02980   Median :0.03361   Median :0.09207
#>  Mean   :0.04645   Mean   :0.04118   Mean   :0.04650   Mean   :0.11664
#>  3rd Qu.:0.05174   3rd Qu.:0.04383   3rd Qu.:0.05179   3rd Qu.:0.13816
#>  Max.   :1.00000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
#>   Shimmer.dB.      Shimmer.APQ3      Shimmer.APQ5     Shimmer.APQ11
#>  Min.   :0.0000   Min.   :0.00000   Min.   :0.00000   Min.   :0.00000
#>  1st Qu.:0.0716   1st Qu.:0.04762   1st Qu.:0.05361   1st Qu.:0.04827
#>  Median :0.1091   Median :0.07507   Median :0.08481   Median :0.07407
#>  Mean   :0.1369   Mean   :0.09652   Mean   :0.11027   Mean   :0.09155
```

```
#>  3rd Qu.:0.1629   3rd Qu.:0.11775   3rd Qu.:0.13215   3rd Qu.:0.11073
#>  Max.   :1.0000   Max.   :1.00000   Max.   :1.00000   Max.   :1.00000
#>   Shimmer.DDA           NHR               HNR              RPDE
#>  Min.   :0.00000   Min.   :0.00000   Min.   :0.0000   Min.   :0.0000
#>  1st Qu.:0.04758   1st Qu.:0.01426   1st Qu.:0.4900   1st Qu.:0.3911
#>  Median :0.07507   Median :0.02428   Median :0.5594   Median :0.4800
#>  Mean   :0.09650   Mean   :0.04256   Mean   :0.5528   Mean   :0.4790
#>  3rd Qu.:0.11775   3rd Qu.:0.04168   3rd Qu.:0.6291   3rd Qu.:0.5681
#>  Max.   :1.00000   Max.   :1.00000   Max.   :1.0000   Max.   :1.0000
#>       DFA              PPE          severity
#>  Min.   :0.0000   Min.   :0.0000   Mode :logical
#>  1st Qu.:0.2336   1st Qu.:0.1893   FALSE:2188
#>  Median :0.3685   Median :0.2586   TRUE :3687
#>  Mean   :0.3959   Mean   :0.2784
#>  3rd Qu.:0.5612   3rd Qu.:0.3417
#>  Max.   :1.0000   Max.   :1.0000
```

6 Separation into Train and Test Data

I will use (pseudo-) random sampling to separate the data into a training and test set.

```
set.seed(1)
ratio <- 0.9
sample.size <- floor(nrow(data) * ratio)
train.indices <- sample(1:nrow(data), size = sample.size)
train <- data[train.indices, ]
test <- data[-train.indices, ]

x_train <- data.matrix(train[,-23])
y_train <- to_categorical(train[, 23], num_classes = 2)
```

```
#> Loaded Tensorflow version 2.7.1
```

```
x_test <- data.matrix(test[,-23])
y_test <- to_categorical(test[, 23], num_classes = 2)
```

7 Implementation of a Dense DNN

A dense deep neural network (DNN) for severity prediction is made. It has two hidden layers, with 10 nodes in each hidden layer. I have implemented two variants of this DNN; variant A has two output nodes, while variant B has only one output node.

7.1 Variant A: Two Output Nodes

Since we have two output nodes in this variant, we should use the *softmax* activation function in the output and the *categorical_crossentropy* loss function.

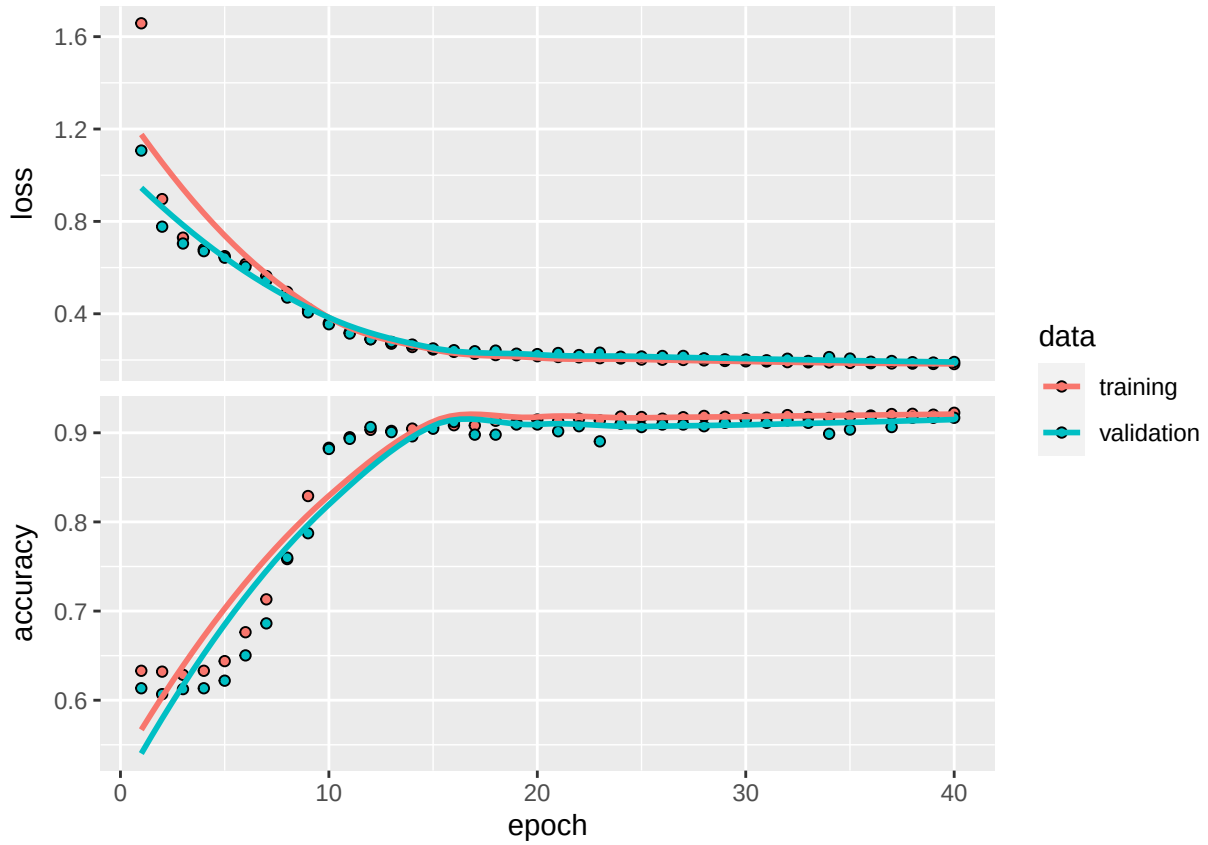
```
set.seed(1)
# defining the model and layers
model <- keras_model_sequential() %>%
  layer_dense(units = 10, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dense(units = 10, activation = 'relu') %>%
  layer_dense(units = ncol(y_train), activation = 'softmax')
```

```
summary(model)
```

```
#> Model: "sequential"
#> _________________________________________________________________________
#>  Layer (type)                       Output Shape                    Param #
#> =========================================================================
#>  dense_2 (Dense)                    (None, 10)                      230
#>
#>  dense_1 (Dense)                    (None, 10)                      110
#>
#>  dense (Dense)                      (None, 2)                       22
#>
#> =========================================================================
#> Total params: 362
#> Trainable params: 362
#> Non-trainable params: 0
#> _________________________________________________________________________
```

```
# compile (define loss and optimizer)
model %>% compile(loss = 'categorical_crossentropy',
                  optimizer = optimizer_rmsprop(),
                  metrics = c('accuracy'))
# train (fit)
history <- model %>% fit(data.matrix(x_train), y_train, epochs = 40,
              batch_size = 256, validation_split = 0.2)
# plot
plot(history)
```

```
#> `geom_smooth()` using formula 'y ~ x'
```




```
# evaluate on training data.
model %>% evaluate(x_train, y_train)
```

```
#>      loss  accuracy
#> 0.1847276 0.9175336
```

```
# evaluate on test data.
model %>% evaluate(x_test, y_test)
```

```
#>      loss  accuracy
#> 0.1473642 0.9438776
```

7.2 Variant B

Since we have one output nodes in this variant, we should use the *sigmoid* activation function in the output and the *binary_crossentropy* loss function.

```
set.seed(1)
# defining the model and layers
model2 <- keras_model_sequential() %>%
  layer_dense(units = 10, activation = 'relu', input_shape = c(ncol(x_train))) %>%
  layer_dense(units = 10, activation = 'relu') %>%
  layer_dense(units = 1, activation = 'sigmoid')

summary(model2)
```

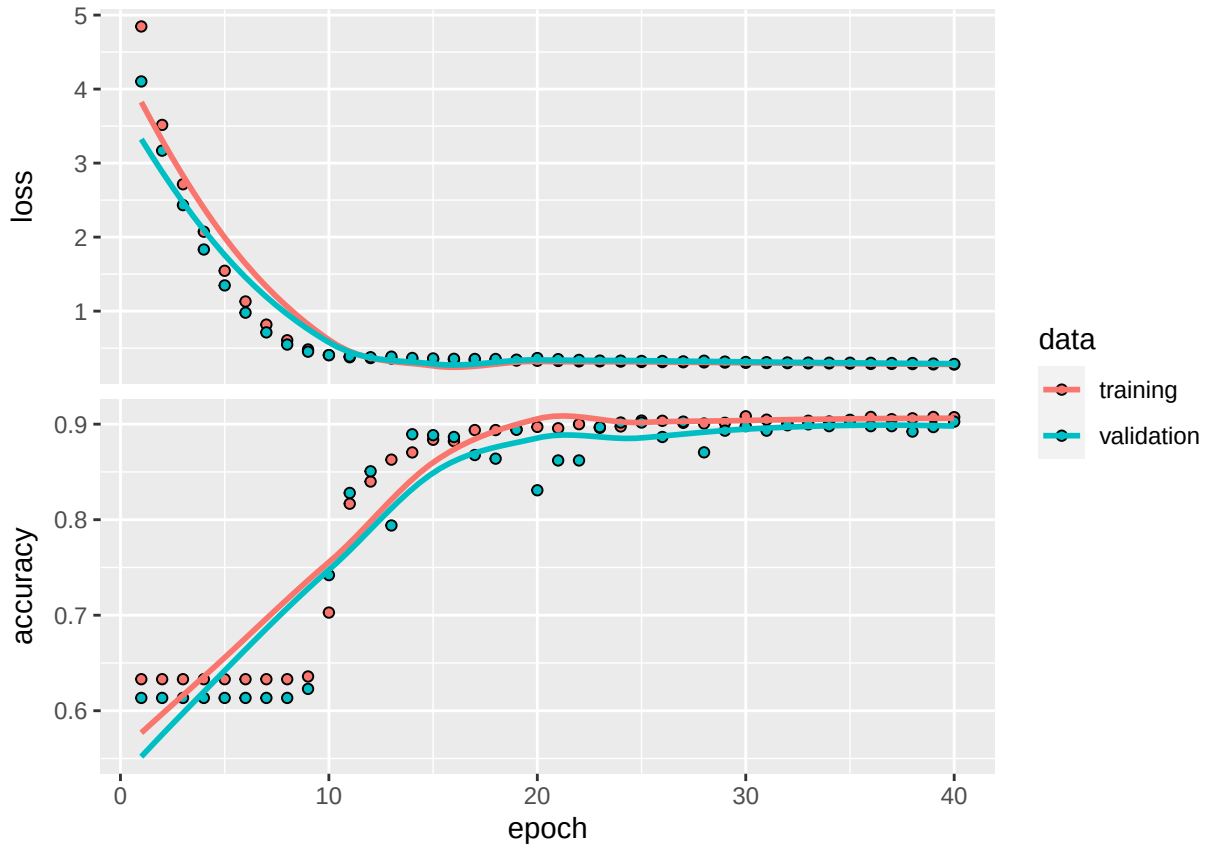
```
#> Model: "sequential_1"
#> ________________________________________________________________________________
#>  Layer (type)                        Output Shape                      Param #
```

```
#> ================================================================================
#>  dense_5 (Dense)                    (None, 10)                    230
#>
#>  dense_4 (Dense)                    (None, 10)                    110
#>
#>  dense_3 (Dense)                    (None, 1)                     11
#>
#> ================================================================================
#> Total params: 351
#> Trainable params: 351
#> Non-trainable params: 0
#> --------------------------------------------------------------------------------
```

```
y_train2 <- as.numeric(data.matrix(train[,23]))
y_test2 <- as.numeric(data.matrix(test[,23]))

# compile (define loss and optimizer)
model2 %>% compile(loss = 'binary_crossentropy',
                   optimizer = optimizer_rmsprop(),
                   metrics = c('accuracy'))
# train (fit)
history2 <- model2 %>% fit(data.matrix(x_train), y_train2, epochs = 40,
              batch_size = 256, validation_split = 0.2)
# plot
plot(history2)
```

```
#> `geom_smooth()` using formula 'y ~ x'
```



```
# evaluate on training data.
model2 %>% evaluate(x_train, y_train2)
```

```
#>      loss  accuracy
#> 0.2774411 0.9103462
```

```
# evaluate on testing data.
model2 %>% evaluate(x_test, y_test2)
```

```
#>      loss  accuracy
#> 0.2632757 0.9251701
```

8 Predictions

8.1 Predictions for Variant A

```
y_pred <- model %>% predict(x_test) %>% k_argmax()
y_pred <- as.array(y_pred)
(tab <- table("Predictions" = y_pred, "Labels" = test[, 23]))
```

```
#>            Labels
#> Predictions FALSE TRUE
#>           0   215   21
#>           1    12  340
```

```
# accuracy in predictions (as shown with the "evaluate" above).
(tab[1]+tab[4])/sum(tab)
```

```
#> [1] 0.9438776
```

8.2 Predictions for Variant B

```
# Predictions for one output node
y_pred2 <- model2 %>% predict(x_test) %>% `>`(0.5) %>% k_cast("int32")
y_pred2 <- as.array(y_pred2)
(tab2 <- table("Predictions" = y_pred2, "Labels" = test[, 23]))
```

```
#>            Labels
#> Predictions FALSE TRUE
#>           0   206   23
#>           1    21  338
```

```
# accuracy in predictions (as shown with the "evaluate" above).
(tab2[1]+tab2[4])/sum(tab2)
```

```
#> [1] 0.9251701
```