Homework 7

DS Student

04, 14, 2022

```
library('keras')
library('ElemStatLearn')
library('nnet')
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
```

Question 1 Use the Keras library to re-implement the simple neural network discussed during lecture for the mixture data

```
# dataset
data <-  mixture.example
train_x <- mixture.example$x
train_y <- mixture.example$y
test_x <- mixture.example$xnew
```

```
model <- keras_model_sequential()
```

```
## Loaded Tensorflow version 2.6.0
```

```
model %>% layer_dense(units=10, activation = "relu", input_shape = c(2))%>% layer_dense(units =2, activa
model %>% compile(optimizer = "rmsprop",
                  loss = "sparse_categorical_crossentropy",
                  metric=c("accuracy"))
```

```
model %>% fit(train_x, train_y,
              epochs = 10,
              batch_size = 5)
```

```
fit_nnet <- nnet(x = train_x, y= train_y, size=10, entropy=TRUE, decay=0.02)
```

```
## # weights:  41
## initial  value 144.375889
## iter  10 value 103.739708
## iter  20 value 93.743236
## iter  30 value 89.948068
```

```
## iter  40 value 88.599596
## iter  50 value 87.573895
## iter  60 value 86.912060
## iter  70 value 86.732612
## iter  80 value 86.678093
## iter  90 value 86.659944
## iter 100 value 86.629076
## final  value 86.629076
## stopped after 100 iterations
```

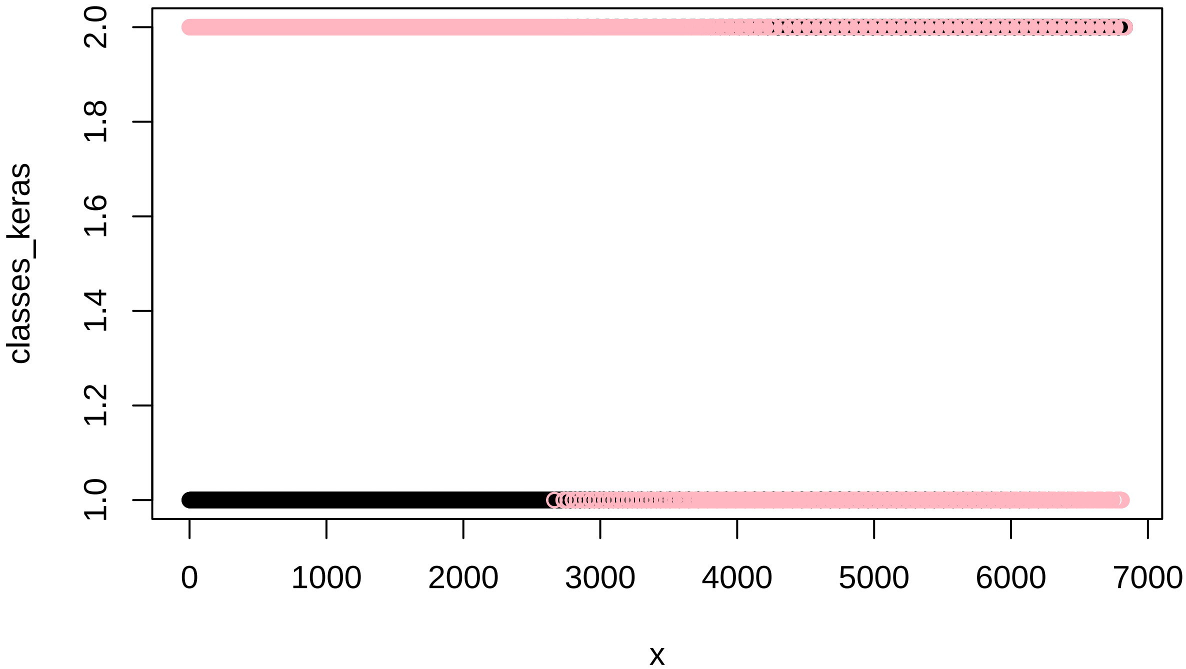
Create a figure to illustrate that the predictions are (or are not) similar using the 'nnet' function versus the Keras model.

```
prediction_keras <- model%>% predict(test_x)

classes_keras <- rep(0,1)
for (i in 1:6831){
  if (prediction_keras[i,1] > prediction_keras[i,2]){
    classes_keras[i] <- 1
  }else{
    classes_keras[i] <- 2
  }
}



prediction_nnet <- fit_nnet%>% predict(test_x)
classes_nnet<- rep(0,1)
for (i in 1:6831){
  if (prediction_nnet[i,1] > 0.5){
    classes_nnet[i]  <- 1
  }else{
    classes_nnet[i]  <- 2
  }
}
x <- seq(1,6831,1)
plot(x,classes_keras)
lines(x,classes_nnet,col = "lightpink",type = "p")
```



From the plot we can see that the predictions are not similar. The points don't have much overlap.

Convert the neural network into CNN

```
fashion_mnist <- dataset_fashion_mnist()
c(train_images, train_labels) %<-% fashion_mnist$train
c(test_images, test_labels) %<-% fashion_mnist$test

img_rows <- 28
img_cols <- 28

train_images<- array_reshape(train_images, c(nrow(train_images), img_rows, img_cols, 1))
test_images <- array_reshape(test_images, c(nrow(test_images), img_rows, img_cols, 1))

train_images <- train_images/255
test_images <- test_images/255

class_names = c('T-shirt/top',
                'Trouser',
                'Pullover',
                'Dress',
                'Coat',
                'Sandal',
                'Shirt',
                'Sneaker',
                'Bag',
                'Ankle boot')

train_labels <- to_categorical(train_labels, 10)
test_labels  <- to_categorical(test_labels, 10)


model1 <- keras_model_sequential() %>%
```

```
  layer_conv_2d(filters = 32, kernel_size = c(3,3), activation = 'relu', input_shape = c(28,28,1)) %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_conv_2d(filters = 64, kernel_size = c(3,3), activation = 'relu') %>%
  layer_max_pooling_2d(pool_size = c(2, 2)) %>%
  layer_dropout(rate = 0.25) %>%
  layer_flatten() %>%
  layer_dense(units = 128, activation = 'relu') %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 10, activation = 'softmax')
```

‘

```
model1 %>% compile(
  loss = loss_categorical_crossentropy,
  optimizer = optimizer_adadelta(),
  metrics = c('accuracy')
)

history <- model1 %>%
  fit(
    x = train_images, y = train_labels,
    epochs = 10
  )
```

```
plot(history)
```