Lab 2

Adeline Casali

2024-01-28

Building the Model

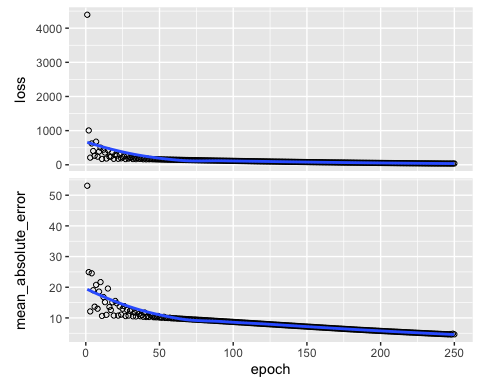
# Load packages  
library(reticulate)  
library(tensorflow)  
library(keras)  
  
# Preparing data  
mtcars <- mtcars  
mtcars\_x <- mtcars[, c("cyl", "disp", "hp")]  
mtcars\_x <- array(data = unlist(mtcars\_x),   
 dim = c(32, 3),   
 dimnames = list(rownames(mtcars\_x), colnames(mtcars\_x)))  
mtcars\_y <- mtcars[, "mpg"]  
  
# Building the model  
nn\_model <- keras\_model\_sequential() %>%  
 layer\_dense(units = 1, input\_shape = 3, activation = "linear")  
nn\_model

## Model: "sequential"  
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
## Layer (type) Output Shape Param #   
## ================================================================================  
## dense (Dense) (None, 1) 4   
## ================================================================================  
## Total params: 4 (16.00 Byte)  
## Trainable params: 4 (16.00 Byte)  
## Non-trainable params: 0 (0.00 Byte)  
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Adding hidden layers  
nn\_model <- keras\_model\_sequential() %>%   
 layer\_dense(units = 2, input\_shape = 3, activation = "relu") %>%   
 layer\_dense(units = 1, activation = "linear")  
nn\_model

## Model: "sequential\_1"  
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
## Layer (type) Output Shape Param #   
## ================================================================================  
## dense\_2 (Dense) (None, 2) 8   
## dense\_1 (Dense) (None, 1) 3   
## ================================================================================  
## Total params: 11 (44.00 Byte)  
## Trainable params: 11 (44.00 Byte)  
## Non-trainable params: 0 (0.00 Byte)  
## \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# Adding the loss function and optimization  
nn\_model %>% compile(optimizer = optimizer\_adam(learning\_rate = 0.2),   
 loss = "mean\_squared\_error",   
 metrics = "mean\_absolute\_error")  
  
# Training the neural network model  
nn\_model\_training <- nn\_model %>% fit(x = mtcars\_x,   
 y = mtcars\_y,   
 epochs = 250,   
 verbose = FALSE)  
  
# Plotting the process (learning curve)  
plot(nn\_model\_training)



1. The predicted miles per gallon of a vehicle with 8 cylinders, a displacement of 250, and horsepower of 200 is 23.26591 mpg.

prediction <- predict(nn\_model, array(c(8, 250, 200), dim = c(1, 3)))

## 1/1 - 0s - 413ms/epoch - 413ms/step

prediction

## [,1]  
## [1,] 23.26591

1. The loss function specified in the above model is mean squared error. This metric is particularly useful for the loss function in neural networks because it is a smooth and differentiable function, which is essential for optimization algorithms such as gradient descent.
2. The rank of a tensor refers to the number of axes or dimensions in the tensor. A matrix is a tensor of rank 2 as it has two dimensions (columns and rows).
3. Vector data is rank 1 tensor data, as it contains one axis.
4. This tensor is a rank 4 tensor, which is likely that of an image. The shape is (500, 256, 256, 3). In this case, the second dimension would be 256. If this tensor represented a set of color images, there would be 500 images.