Week2_HomeworkSimonsen

Steven Simonsen

2024-09-08

```
library(reticulate)
library(tensorflow)
library(keras3)
```

```
##
## Attaching package: 'keras3'

## The following objects are masked from 'package:tensorflow':
##
##     set_random_seed, shape
```

```
use_virtualenv("my_tf_workspace", required = TRUE)


mtcars <- mtcars
mtcars_x <- mtcars[, c("cyl", "disp", "hp")]

mtcars_x <- array(data = unlist(mtcars_x), dim = c(nrow(mtcars), 3),
                  dimnames = list(rownames(mtcars),
                                  colnames(mtcars_x)))

mtcars_y <- mtcars[, "mpg"]


set.seed(42)
nn_model <- keras_model_sequential() %>%
  layer_dense(units = 1, input_shape = 3, activation = "linear")

nn_model
```

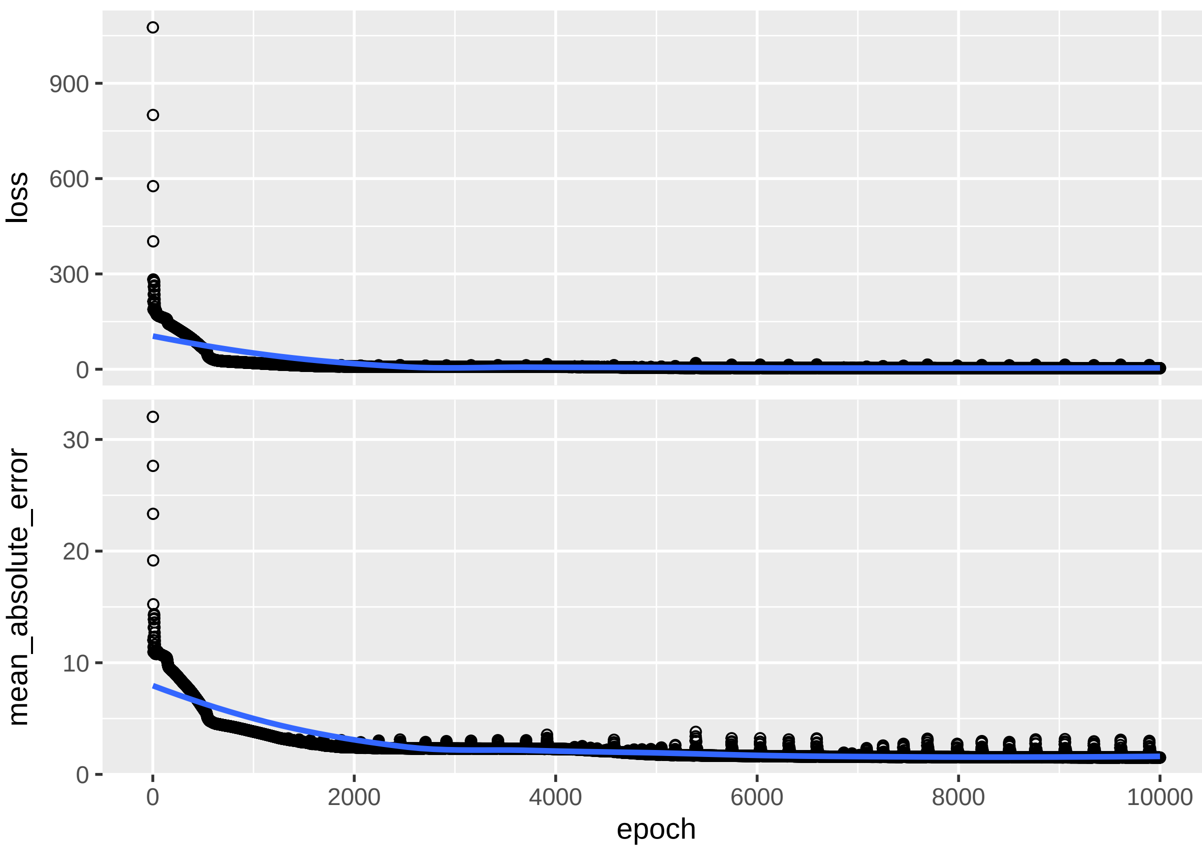
```
## Model: "sequential"
##
##   Layer (type)                    Output Shape                  Param #
##
##   dense (Dense)                   (None, 1)                           4
##
##  Total params: 4 (16.00 B)
##  Trainable params: 4 (16.00 B)
##  Non-trainable params: 0 (0.00 B)
```

```
#clear_session(free_memory = TRUE)

nn_model <- keras_model_sequential() %>%
  layer_dense(units = 4, input_shape = 3, activation = "relu") %>%
```

```r
  layer_dense(units = 1, activation = "linear")

nn_model
```

```
## Model: "sequential_1"
##
##  Layer (type)                       Output Shape                    Param #
##
##  dense_1 (Dense)                    (None, 4)                            16
##
##  dense_2 (Dense)                    (None, 1)                             5
##
## Total params: 21 (84.00 B)
## Trainable params: 21 (84.00 B)
## Non-trainable params: 0 (0.00 B)
```

```r
nn_model %>% compile(optimizer = optimizer_adam(learning_rate = 0.01),
                     loss = "mean_squared_error",
                     metrics = "mean_absolute_error")

nn_model_training <- nn_model %>% fit(x = mtcars_x,
                                      y = mtcars_y,
                                      epochs = 10000,
                                      verbose = FALSE)

plot(nn_model_training)
```



```r
get_weights(nn_model)
```

```
## [[1]]
##             [,1]       [,2]       [,3]        [,4]
## [1,] -4.01050234  2.0503147  0.7121634 -0.77634412
## [2,]  0.35241047  0.5790529 -0.8115038  0.29124552
## [3,] -0.09168075 -0.2686783 -0.6352078 -0.09791051
##
## [[2]]
## [1] -12.01632  13.10728   0.00000 -11.19092
##
## [[3]]
##            [,1]
## [1,]  1.0062258
## [2,]  0.5841576
## [3,] -0.3201422
## [4,] -2.4225204
##
## [[4]]
## [1] 6.969387
```

```r
prediction1 <- predict(nn_model, array(c(6, 350, 125), dim = c(1, 3)))
```

```
## 1/1 - 0s - 32ms/step
```

```r
prediction2 <- predict(nn_model, array(c(8, 250, 200), dim = c(1, 3)))
```

```
## 1/1 - 0s - 17ms/step
```

```r
print(prediction1)
```

```
##          [,1]
## [1,] 17.96146
```

```r
print(prediction2)
```

```
##          [,1]
## [1,] 16.41502
```

1) Using the neural network above, what is the predicted miles per gallon of a vehicle with 8 cylinders, a displacement of 250, and a horsepower of 200?

To accurately model this behavior, I made a few modifications to the code above. First, I added a few layers to the neural network to add some additional non-linear behavior, and identify complex relationships in the data. Second, I decreased the learning rate to a value of .01 to encourage slower convergence to update the weights more slowly, and also lead to more stable behavior. Third, I increased the number of epochs to 10,000 to extend the number of passes through the training set. Overall, the predicted miles per gallon of a vehicle with 8 cylinders, a displacement of 250, and a horsepower of 200 is 16.95697.

2) What loss function did we specify when building the neural network model above?

The loss function specified above is the mean squared error, which takes the average the average squared difference between the predicted values and the actual values.

3) Define the rank of a tensor. What is the rank of a matrix (remember, a matrix is a tensor of a particular rank)?

The rank of a tensor refers to the number of dimensions, or the number of axis' required to describe the tensor. So, in this context, a matrix is a tensor of rank 2, as it has two dimensions (i.e. rows and columns) to describe it.

4) The data we will be working with throughout the course, including in this lab, is vector data. What is the tensor rank of vector data?

The tensor rank of vector data is rank 1.

5) Consider the following tensor of ones: array(1, dim = c(500, 256, 256, 3)). What is its rank? What is its shape? What is the dimension of the second axis? If this tensor represented a collection of color images, how many images are there?

The rank of this tensor is 4, and the shape is represented as [500, 256, 256, 3]. The dimension of the second axis is 256. If this tensor represented a collection of color images, there would be 500 images.