HW6

2024-12-09

```
#

library(reticulate)
```

```
## Warning:     'reticulate'          R     4.4.2
```

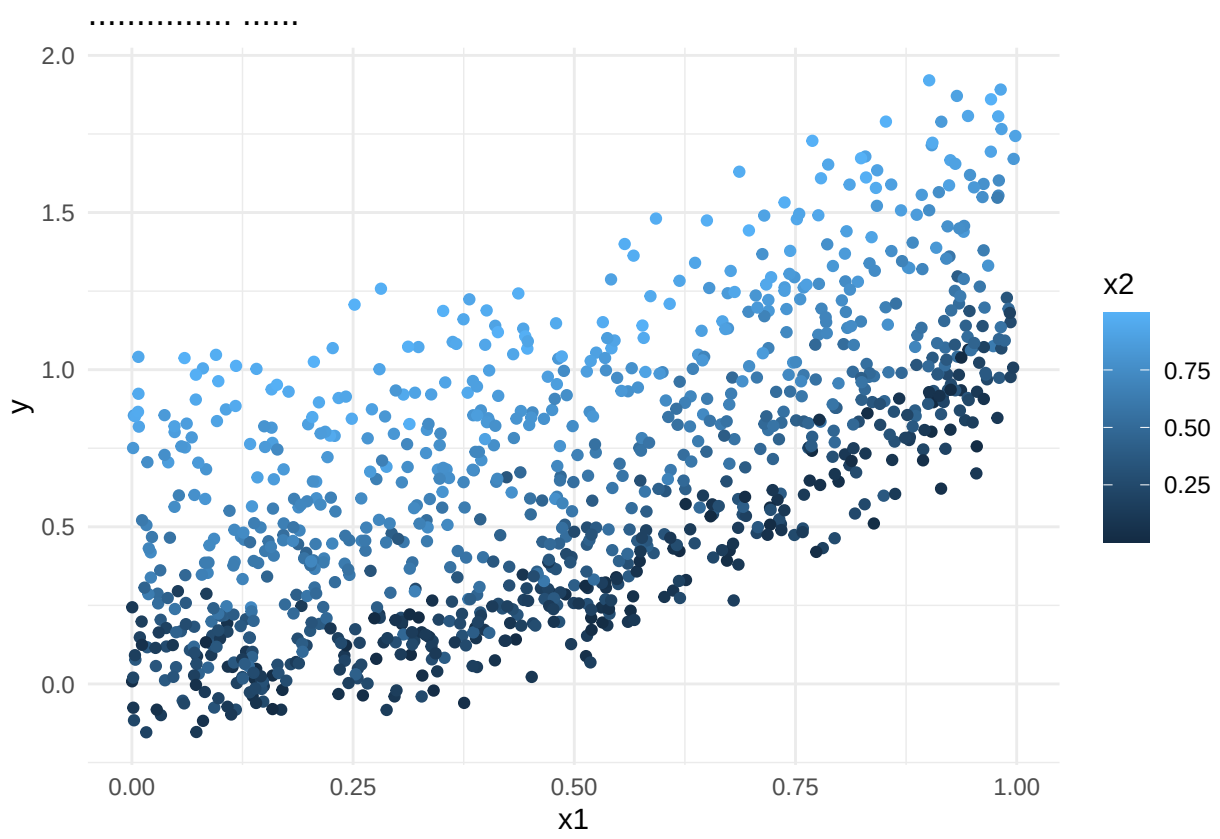
```
py_config()
```

```
## python:         C:/Users/Lena/Documents/.virtualenvs/r-reticulate/Scripts/python.exe
## libpython:      C:/Users/Lena/AppData/Local/Programs/Python/Python311/python311.dll
## pythonhome:     C:/Users/Lena/Documents/.virtualenvs/r-reticulate
## version:        3.11.0 (main, Oct 24 2022, 18:26:48) [MSC v.1933 64 bit (AMD64)]
## Architecture:   64bit
## numpy:          C:/Users/Lena/Documents/.virtualenvs/r-reticulate/Lib/site-packages/numpy
## numpy_version:  2.0.2
##
## NOTE: Python version was forced by VIRTUAL_ENV
```

```
set.seed(42)
n <- 1000

#           x1, x2         y
x1 <- runif(n)
x2 <- runif(n)
epsilon <- rnorm(n, mean = 0, sd = 0.1)
y <- x1^2 + x2^2 + epsilon

#
data <- data.frame(x1 = x1, x2 = x2, y = y)

#
library(ggplot2)
```

```
## Warning:     'ggplot2'          R     4.4.2
```

```
ggplot(data, aes(x = x1, y = y, color = x2)) +
  geom_point() +
  labs(title = "            ", x = "x1", y = "y") +
  theme_minimal()
```




```r
#
sigmoid <- function(x) {
  1 / (1 + exp(-x))
}

sigmoid_derivative <- function(x) {
  sig <- sigmoid(x)
  sig * (1 - sig)
}

#
input_size <- 2   # x1, x2
hidden_size <- 5  #
output_size <- 1  # y

#
set.seed(42)
W1 <- matrix(runif(input_size * hidden_size, -1, 1), nrow = input_size)
b1 <- rep(0, hidden_size)
W2 <- matrix(runif(hidden_size * output_size, -1, 1), nrow = hidden_size)
b2 <- rep(0, output_size)

#
feed_forward <- function(X) {
  Z1 <- X %*% W1 + matrix(rep(b1, each = nrow(X)), nrow = nrow(X), byrow = TRUE)
  A1 <- sigmoid(Z1)
```

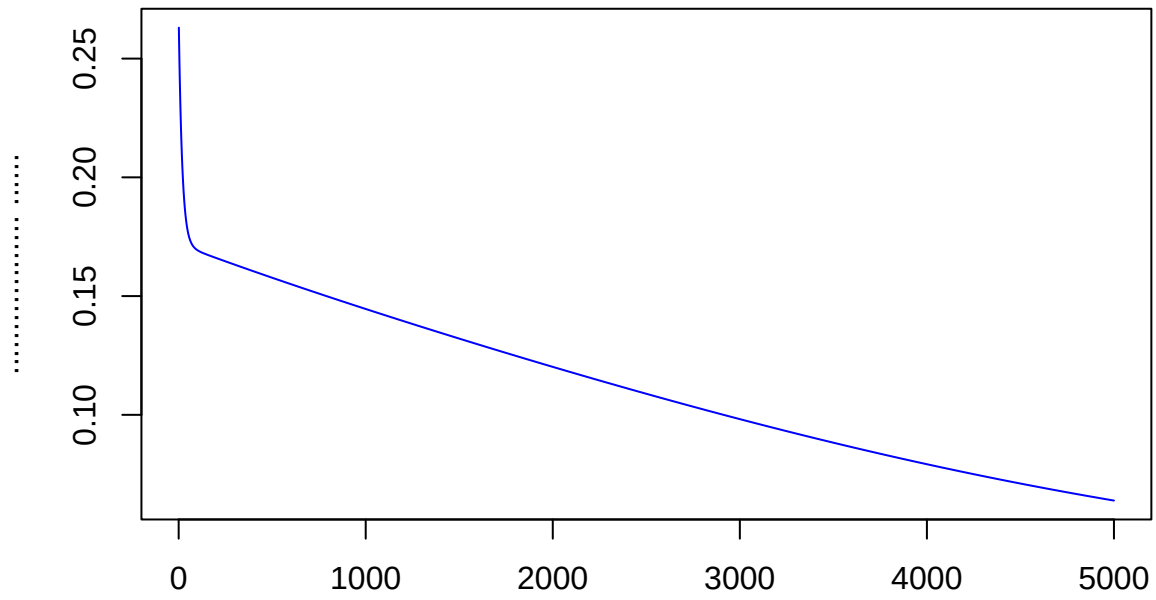
```
  Z2 <- A1 %*% W2 + matrix(rep(b2, each = nrow(A1)), nrow = nrow(A1), byrow = TRUE)
  A2 <- Z2  #
  list(A1 = A1, A2 = A2)
}

#
back_propagation <- function(X, y, A1, A2, learning_rate = 0.01) {
  m <- nrow(X)

  #
  dA2 <- A2 - y
  dW2 <- t(A1) %*% dA2 / m
  db2 <- colSums(dA2) / m

  #
  dA1 <- dA2 %*% t(W2) * sigmoid_derivative(A1)
  dW1 <- t(X) %*% dA1 / m
  db1 <- colSums(dA1) / m

  #
  W1 <<- W1 - learning_rate * dW1
  b1 <<- b1 - learning_rate * db1
  W2 <<- W2 - learning_rate * dW2
  b2 <<- b2 - learning_rate * db2
}

#
num_iterations <- 5000
learning_rate <- 0.01
losses <- numeric(num_iterations)

for (i in 1:num_iterations) {
  #
  res <- feed_forward(cbind(x1, x2))
  A1 <- res$A1
  A2 <- res$A2

  #           (             )
  loss <- mean((A2 - y)^2)
  losses[i] <- loss

  #
  back_propagation(cbind(x1, x2), y, A1, A2, learning_rate)

  if (i %% 500 == 0) {
    cat("Iteration:", i, "Loss:", loss, "\n")
  }
}
```

```
## Iteration: 500 Loss: 0.1577271
## Iteration: 1000 Loss: 0.1445747
## Iteration: 1500 Loss: 0.1321024
## Iteration: 2000 Loss: 0.1201996
```

```
## Iteration: 2500 Loss: 0.1088729
## Iteration: 3000 Loss: 0.0981975
## Iteration: 3500 Loss: 0.08827696
## Iteration: 4000 Loss: 0.0792116
## Iteration: 4500 Loss: 0.0710766
## Iteration: 5000 Loss: 0.06390998
```

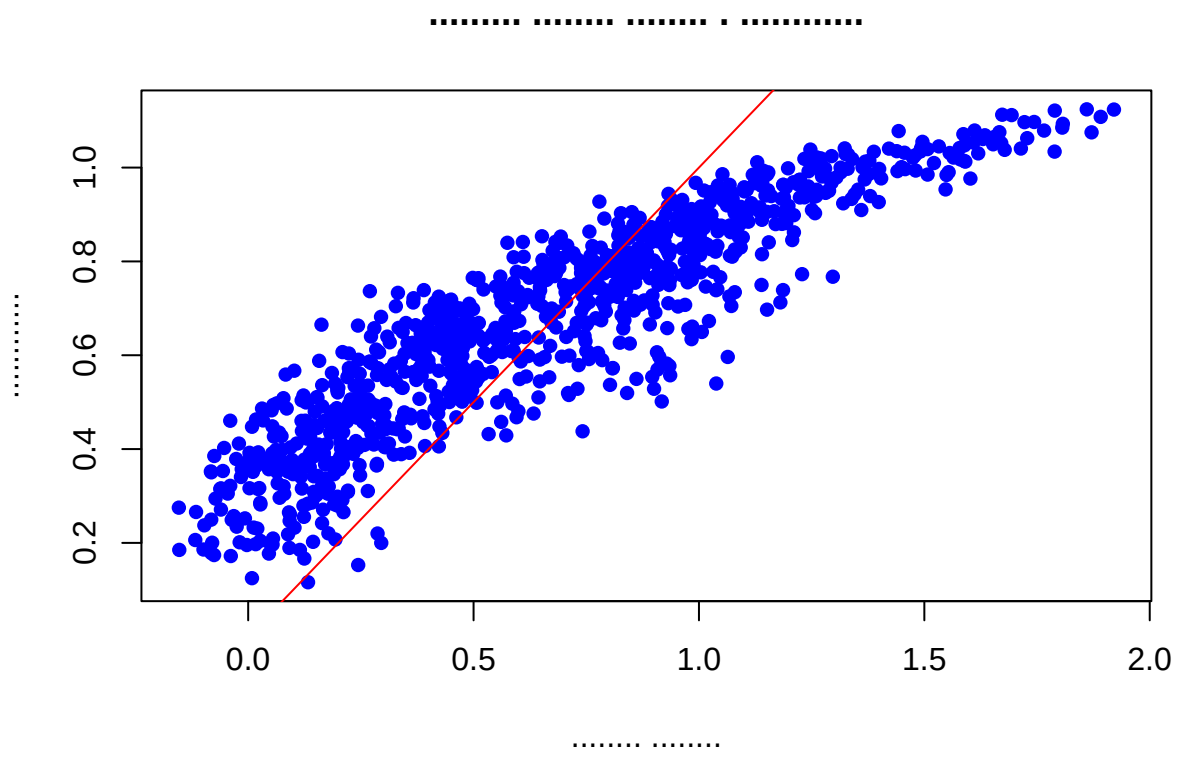
```r
#
plot(1:num_iterations, losses, type = "l", col = "blue",
     xlab = "     ", ylab = "              ",
     main = "                    ")
```

....



.....

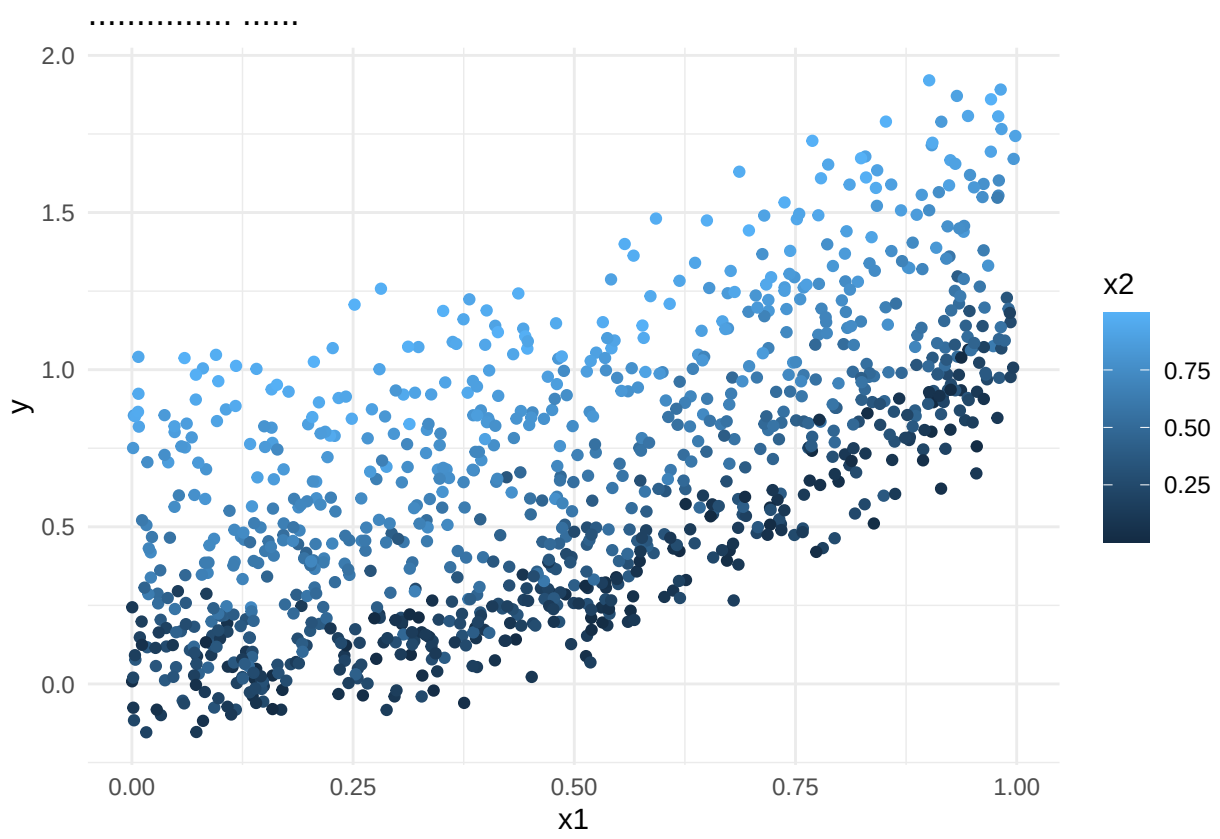
```r
#        y
predictions <- feed_forward(cbind(x1, x2))$A2

#
plot(y, predictions, col = "blue", pch = 16,
     xlab = "          ", ylab = "       ",
     main = "                          ")
abline(0, 1, col = "red")  #
```




```
library(keras)
```

```
## Warning:     'keras'          R     4.4.2
```

```
library(ggplot2)

set.seed(42)
n <- 1000

#           x1, x2          y
x1 <- runif(n)
x2 <- runif(n)
epsilon <- rnorm(n, mean = 0, sd = 0.1)
y <- x1^2 + x2^2 + epsilon

#
data <- data.frame(x1 = x1, x2 = x2, y = y)

#
ggplot(data, aes(x = x1, y = y, color = x2)) +
  geom_point() +
  labs(title = "             ", x = "x1", y = "y") +
  theme_minimal()
```




```r
#
X <- cbind(x1, x2)  #
Y <- y  #

#
X_tensor <- array_reshape(X, c(n, 2))  #
```