max(average\_acc\_history$validation\_acc)

[1] 0.8075095

> # Evaluate on Testset

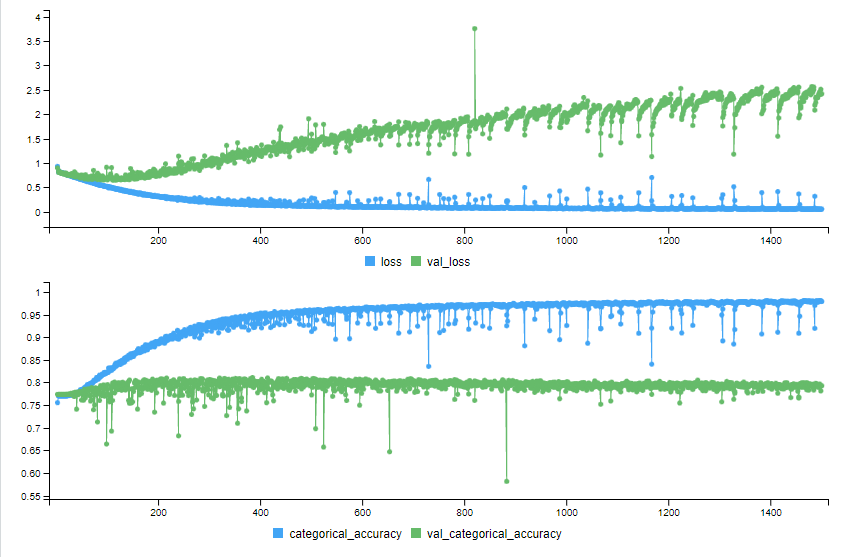
> eval <- evaluate(model, test\_data, test\_targets, verbose = 1)

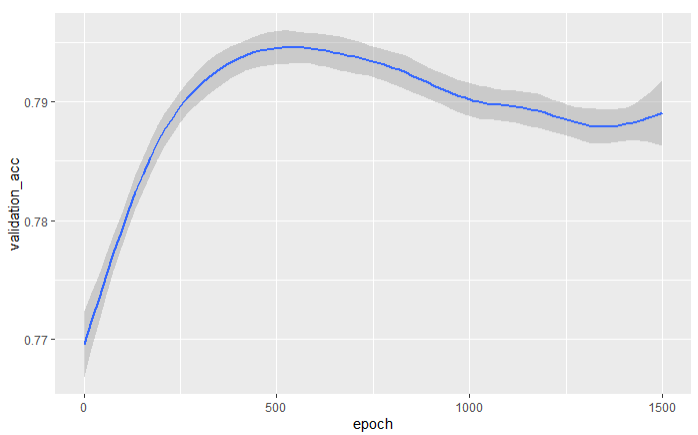
423/423 [==============================] - 1s 2ms/step - loss: 3.1921 - categorical\_accuracy: 0.7380

> eval

loss categorical\_accuracy

3.1921480 0.7380019





#train\_data <- matrix\_data

train\_data <- data.matrix(matrix\_data)

test\_data <- data.matrix(matrix\_data\_test)

train\_targets <- data.matrix(matrix\_targets)

test\_targets <- data.matrix(matrix\_targets\_test)

# Function to build the model

build\_model <- function() {

model <- keras\_model\_sequential() %>%

#layer\_batch\_normalization(axis = -1L, input\_shape = dim(train\_data)[[2]]) %>%

layer\_dense(units = 64, activation = "relu", input\_shape = dim(train\_data)[[2]]) %>%

layer\_dense(units = 64, activation = "relu") %>%

#layer\_dropout(0.3) %>%

layer\_dense(units = 64, activation = "relu") %>%

layer\_dense(units = 64, activation = "relu") %>%

layer\_dense(units = 64, activation = "relu") %>%

layer\_dense(units = 8, activation = "softmax")

model %>% compile(

optimizer = optimizer\_sgd(learning\_rate = 0.2),

loss = "categorical\_crossentropy",

metrics = "categorical\_accuracy"

)

}