Homework7

Steven Black

4/5/2019

IS 677: Introduction to Data Science Spring 2019

Homework Assignment 7 (Due: April 7, 2019, midnight EST)

Use the fashion_mnist data that comes with Keras to answer the questions below. The fashion_mnist dataset has 60,000 training images and 10,000 test images. All images are 28*28 arrays and are grey scale images. The images are individual articles of clothing labeled as 0-9 as follows:

- 0: T-shirt/top,
- 1: Trouser,
- 2: Pullover,
- 3: Dress,
- 4: Coat,
- 5: Sandal,
- 6: Shirt,
- 7: Sneaker,
- 8: Bag,
- 9: Ankle boot.

You can follow the example from last week to answer questions 1-6.

```
library(keras)
```

1. Load the data into a variable called fashion. (5 points)

```
fashion <- dataset_fashion_mnist()
```

2. Load the train_images, train_labels, test_images, and test_labels into appropriate variables. (5 points)

```
train_images <- fashion$train$x
train_labels <- fashion$train$y
test_images <- fashion$test$x
test_labels <- fashion$test$y
```

3. Build the network with one hidden layer with 512 units, and "relu" activation function. Compile the network with "rmsprop" as the optimizer, "categorical_crossentropy" as the loss function, and "accuracy" as the metric. (10 points)

```
build_model <- function() {
  network <- keras_model_sequential() %>%
    layer_dense(units = 512, activation = "relu", input_shape = c(28 * 28)) %>%
    layer_dense(units = 10, activation = "softmax")

  network %>% compile(
    optimizer = "rmsprop",
    loss = "categorical_crossentropy",
    metrics = c("accuracy")
  )
}
```

4. Train the network with 5 epochs. Remember that you will have to reshape the images and categorically encode the labels first. (10 points)

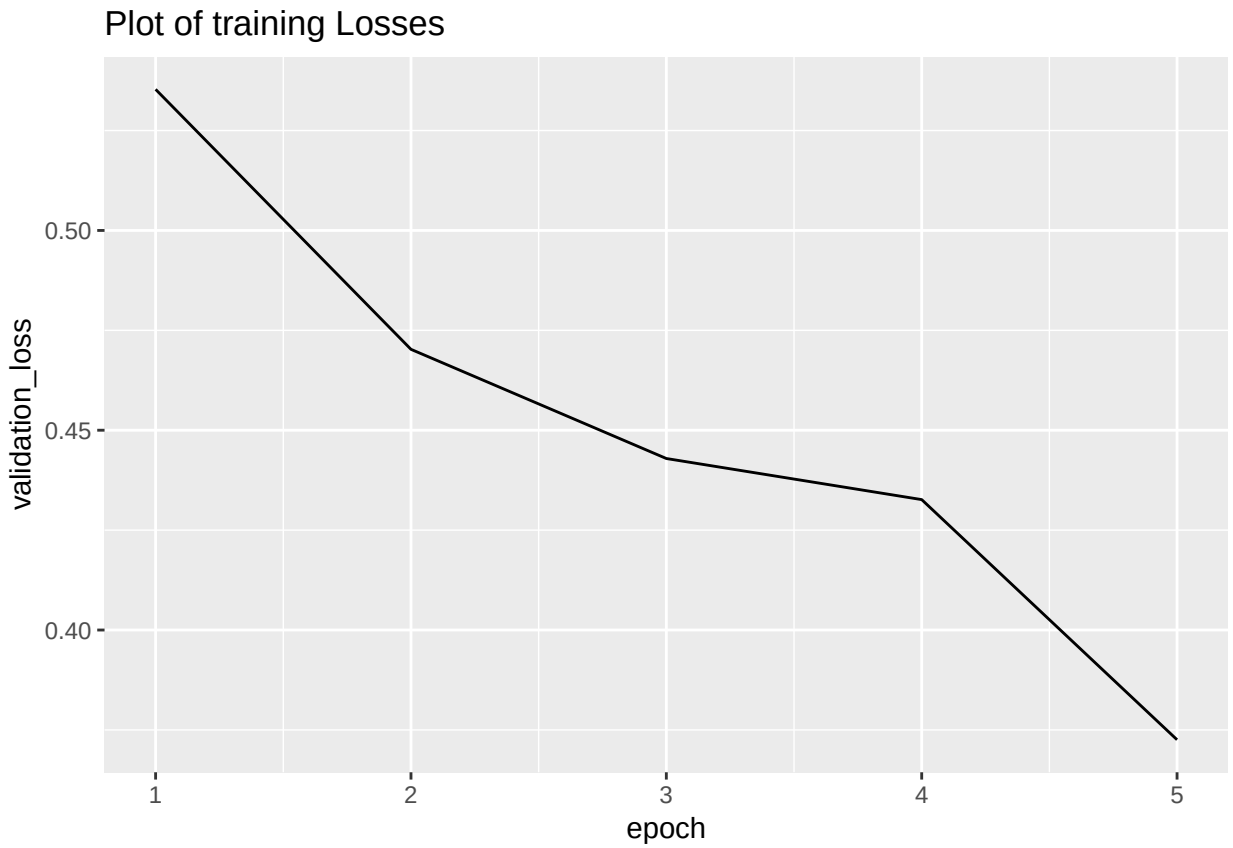
```
network <- build_model()
train_images_reshaped <- array_reshape(train_images, c(60000, 28 * 28))
train_images_reshaped <- train_images_reshaped / 255

test_images_reshaped <- array_reshape(test_images, c(10000, 28 * 28))
test_images_reshaped <- test_images_reshaped / 255

train_labels_categorical <- to_categorical(train_labels)
test_labels_categorical <- to_categorical(test_labels)

network %>% fit(train_images_reshaped, train_labels_categorical, epochs = 5, batch_size = 128)
```

5. Validate the model with a validation split of 0.3. Plot the training and validation losses. Is the model overfitting or underfitting? Why? (10 points)

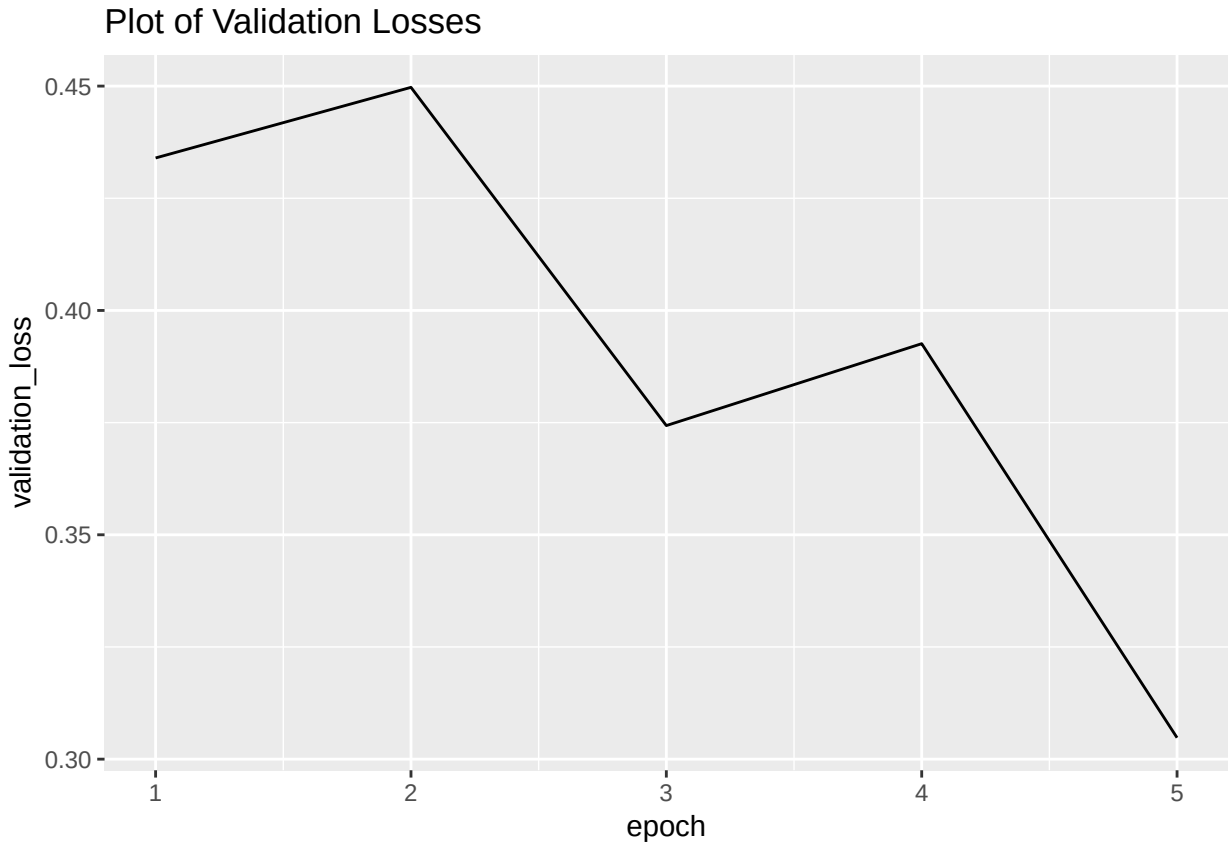
It's hard to say because with a sample size of only five epochs there is too much variability to see a greater trend. It does seem that there is overfitting because loss starts to increase toward the end of the graph.

```
indices <- sample(1:nrow(train_images_reshaped), size = 0.3 * nrow(train_images_reshaped))
evaluation_data  <- train_images_reshaped[-indices, ]
evaluation_targets <- train_labels_categorical[-indices, ]
partial_train_data <- train_images_reshaped[indices, ]
partial_train_targets <- train_labels_categorical[indices, ]

model <- build_model()

num_epochs <- 5

train_history <- model %>% fit(
  partial_train_data,
  partial_train_targets,
  validation_data = list(evaluation_data, evaluation_targets),
  epochs = num_epochs, batch_size = 128
)

validation_history <- model %>% fit(
  evaluation_data,
  evaluation_targets,
  validation_data = list(partial_train_data, partial_train_targets),
  epochs = num_epochs, batch_size = 128
)
```

```
library(ggplot2)
train_loss_obj <- data.frame(
  epoch = seq(1:length(train_history$metrics$val_loss)),
  validation_loss = train_history$metrics$val_loss
)
ggplot(train_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of training Losses")
```




```
library(ggplot2)
validation_loss_obj <- data.frame(
  epoch = seq(1:length(validation_history$metrics$val_loss)),
  validation_loss = validation_history$metrics$val_loss
)
ggplot(validation_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of Validation Losses")
```



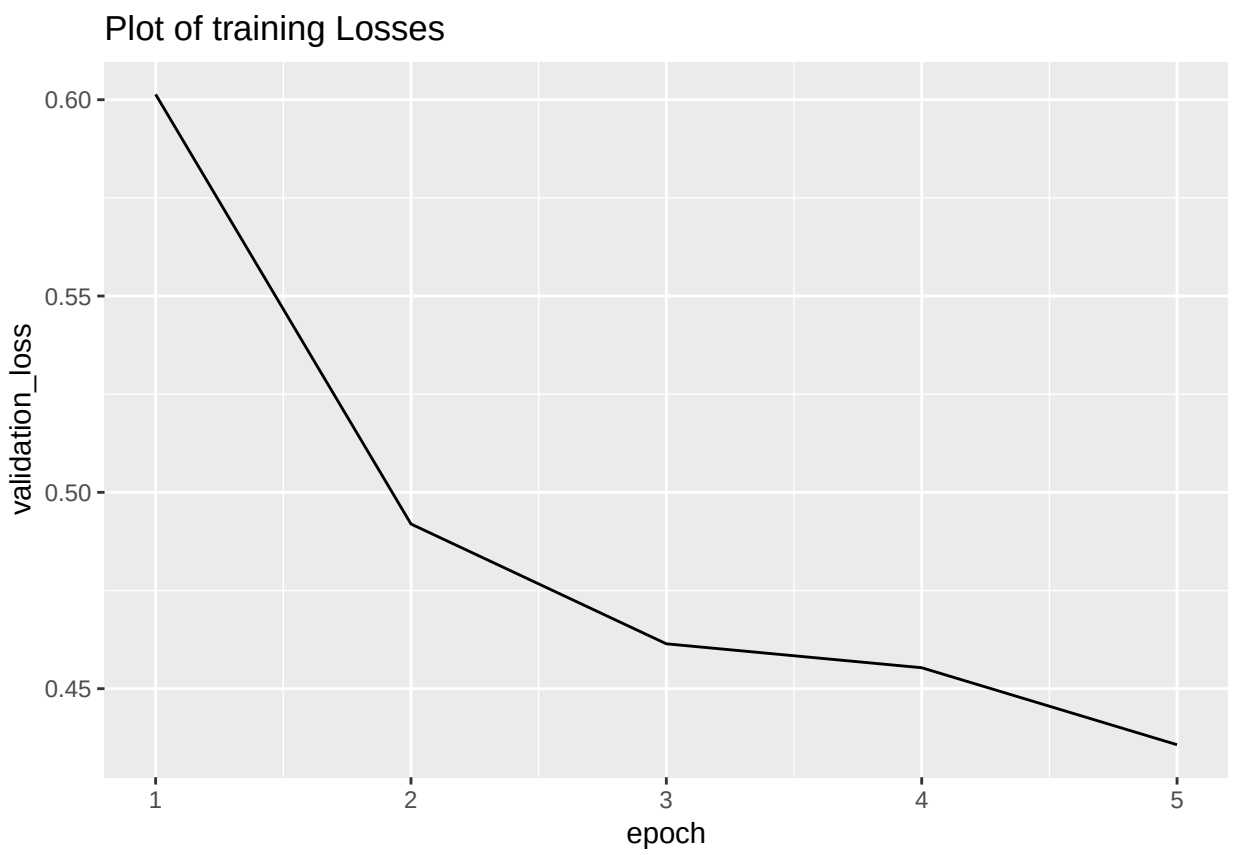
6. Clear the previous session. Reduce the model size. Train, validate with the same split and plot the losses. Is the model overfitting or underfitting? (20 points)

The small sample size still makes it difficult to see a larger trend in validation loss. It does seem to still be increasing as the plot progresses which suggests overfitting.

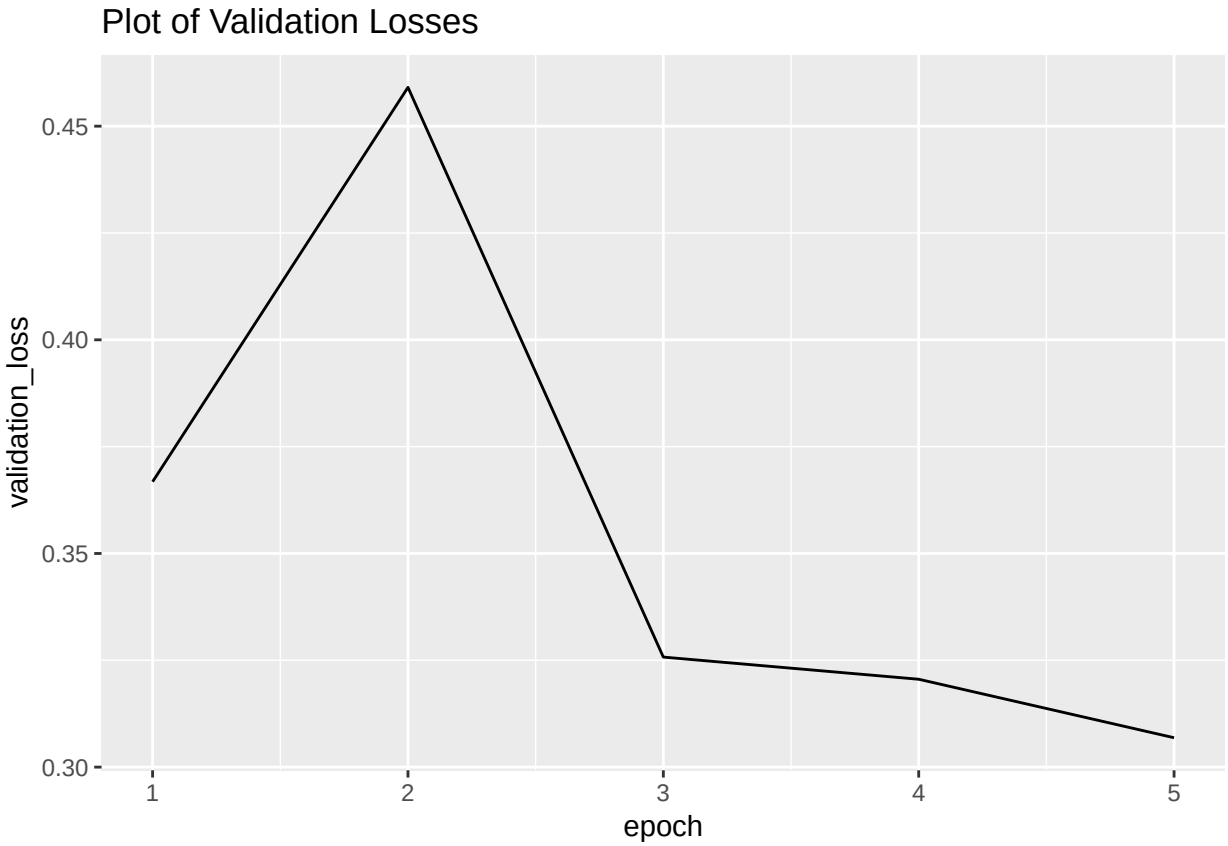
```
k_clear_session()

network <- keras_model_sequential() %>%
  layer_dense(units = 256, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")

network %>% compile(
  optimizer = "rmsprop",
  loss = "categorical_crossentropy",
  metrics = c("accuracy")
)

train_history <- network %>% fit(
  partial_train_data,
  partial_train_targets,
  validation_data = list(evaluation_data, evaluation_targets),
  epochs = num_epochs, batch_size = 128
)

validation_history <- network %>% fit(
  evaluation_data,
  evaluation_targets,
```

```
  validation_data = list(partial_train_data, partial_train_targets),
  epochs = num_epochs, batch_size = 128
)
```

```
library(ggplot2)
train_loss_obj <- data.frame(
  epoch = seq(1:length(train_history$metrics$val_loss)),
  validation_loss = train_history$metrics$val_loss
)
ggplot(train_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of training Losses")
```



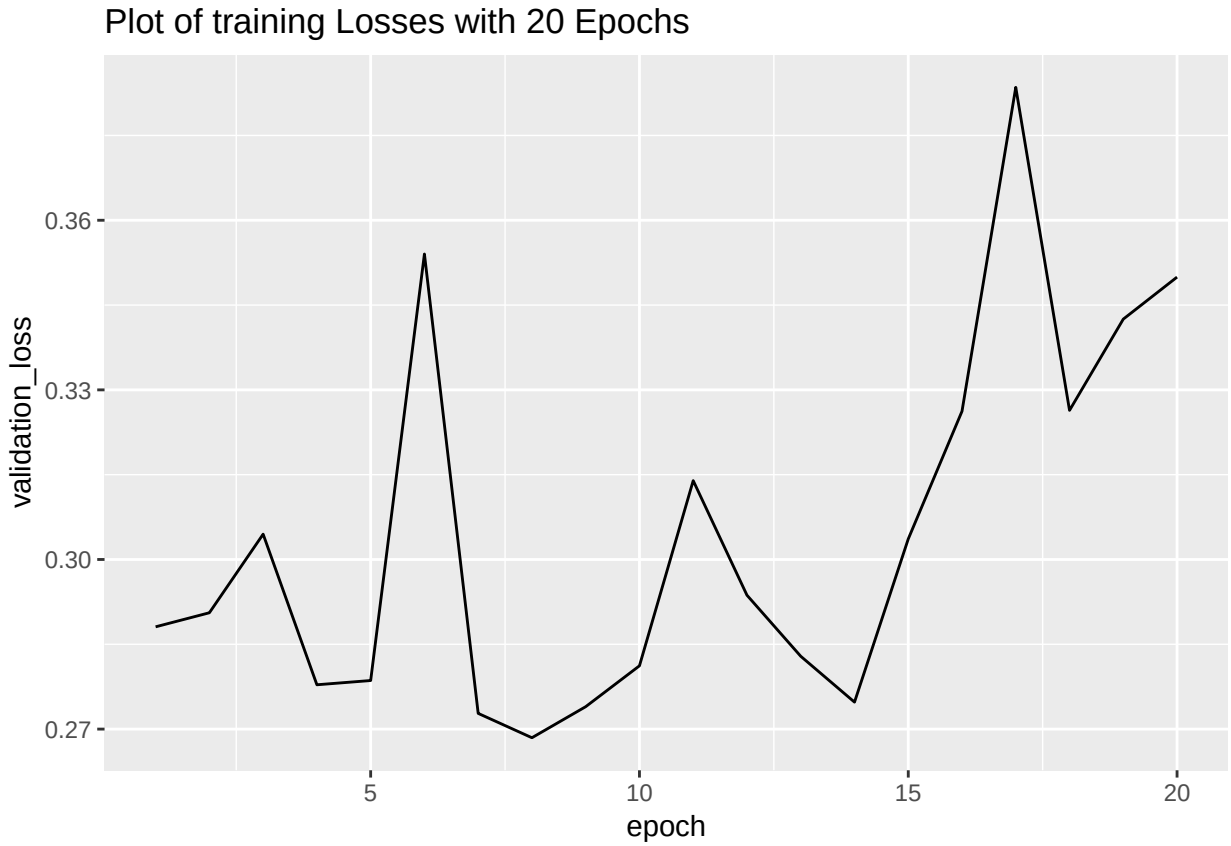

```
library(ggplot2)
validation_loss_obj <- data.frame(
  epoch = seq(1:length(validation_history$metrics$val_loss)),
  validation_loss = validation_history$metrics$val_loss
)
ggplot(validation_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of Validation Losses")
```



7. Fit the model in question 6 with 20 epochs. Is it overfitting? (20 points)

There is a definate upward trend as the plot progressess suggesting that this model is overfitting.

```
num_epochs <- 20

train_history <- network %>% fit(
  partial_train_data,
  partial_train_targets,
  validation_data = list(evaluation_data, evaluation_targets),
  epochs = num_epochs, batch_size = 128
)

train_loss_obj <- data.frame(
  epoch = seq(1:length(train_history$metrics$val_loss)),
  validation_loss = train_history$metrics$val_loss
)
ggplot(train_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of training Losses with 20 Epochs")
```



8. Clear the previous session. Add a dropout layer to the model in question 7 and retrain it with 20 epochs. Comment of model overfitting or underfttting. (20 points)

There is a definate upward trend as the plot progressess suggesting that this model is overfitting.

```
k_clear_session()

network <- keras_model_sequential() %>%
  layer_dense(units = 512, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dropout(rate = 0.5) %>%
  layer_dense(units = 10, activation = "softmax")

network %>% compile(
  optimizer = "rmsprop",
  loss = "categorical_crossentropy",
  metrics = c("accuracy")
)
num_epochs <- 20

train_history <- network %>% fit(
  partial_train_data,
  partial_train_targets,
  validation_data = list(evaluation_data, evaluation_targets),
  epochs = num_epochs, batch_size = 128
)

train_loss_obj <- data.frame(
  epoch = seq(1:length(train_history$metrics$val_loss)),
```

```
  validation_loss = train_history$metrics$val_loss
)
ggplot(train_loss_obj,
       aes(x = epoch, y = validation_loss)
       ) + geom_line() + ggtitle("Plot of Training Losses with Dropout")
```