Final Project

Julia Thacker

5/3/2022

```
library(keras)

## Warning: package 'keras' was built under R version 4.1.2

fashion <- dataset_fashion_mnist()

## Loaded Tensorflow version 2.8.0

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

Imported the dataset and separated the training and test data.

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

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

train_labels <- to_categorical(train_labels)
test_labels <- to_categorical(test_labels)
```

Converted the labels to categorical.

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

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

model %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Created a model with one layer and a 20% validation split.

```
model2 <- keras_model_sequential() %>%
  layer_dense(units = 512, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")
  layer_dense(units = 10, activation = "softmax")

## <keras.layers.core.dense.Dense>

model2 %>% compile(
  optimizer = "rmsprop",
  loss = "categorical_crossentropy",
  metrics = c("accuracy")
)

model2 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Created a model with an additional layer. This model did not outperform the first.

```
model3 <- keras_model_sequential() %>%
  layer_dense(units = 512, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")

model3 %>% compile(
  optimizer = "rmsprop",
  loss = "mse",
  metrics = c("accuracy")
)

model3 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Created a model using the mse loss function.

```
model4 <- keras_model_sequential() %>%
  layer_dense(units = 512, activation = "tanh", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")

model4 %>% compile(
  optimizer = "rmsprop",
  loss = "mse",
  metrics = c("accuracy")
)

model4 %>% fit(
  train_images, train_labels,
  epochs = 10,
```

```
  batch_size = 128,
  validation_split = 0.2)
```

Created a model that uses tanh activation.

```
model5<-keras_model_sequential() %>%
  layer_dense(units = 128, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")

model5 %>% compile(
  optimizer = "adam",
  loss = "mse",
  metrics = "accuracy"
)

model5 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Made a new model using Adam as the optimizer.

```
set.seed(1234)
initializer <- initializer_random_normal(seed = 1234)

model6 <- keras_model_sequential() %>%
  layer_dense(units = 32,
              activation = "relu", input_shape = ncol(train_images),
              kernel_initializer = initializer, bias_initializer =
initializer) %>%
  layer_dense(units = 10, activation = "softmax",
           kernel_initializer = initializer, bias_initializer = initializer)

model6 %>% compile(
  optimizer = optimizer_adam(learning_rate = 0.003),
  loss = "mse",
  metrics = "accuracy"
)

model6 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Created a model using adam as the optimizer with a learning rate of 0.003.

```
model7<-keras_model_sequential() %>%
  layer_dense(units = 256, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dropout(rate = 0.4) %>%
  layer_dense(units = 128, activation = "relu") %>%
```

```
  layer_dropout(rate = 0.4) %>%
  layer_dense(units = 10, activation = "softmax")

model7 %>% compile(
  optimizer = optimizer_rmsprop(),
  loss = "mse",
  metrics = "accuracy"
)

model7 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 128,
  validation_split = 0.2)
```

Created a model using dropout.

```
model8<-keras_model_sequential() %>%
  layer_dense(units = 128, activation = "relu", input_shape = c(28 * 28)) %>%
  layer_dense(units = 10, activation = "softmax")
  layer_dense(units = 10, activation = "softmax")

## <keras.layers.core.dense.Dense>

model8 %>% compile(
  optimizer = "adam",
  loss = "mse",
  metrics = "accuracy"
)

model8 %>% fit(
  train_images, train_labels,
  epochs = 10,
  batch_size = 500,
  validation_split = 0.4)
```

Created a final model using 3 layers, adam, mse loss function, a larger batch size of 500, and a validation split of 40%.