Homework 2

Adam Shelton

5/13/2019

Loading Data

```
congress_test = read_csv(here("data", "congress_test.csv"))
```

```
## Parsed with column specification:
## cols(
##   BillID = col_character(),
##   BillNum = col_double(),
##   Title = col_character(),
##   Major = col_double()
## )
```

```
congress_train = read_csv(here("data", "congress_train.csv"))
```

```
## Parsed with column specification:
## cols(
##   BillID = col_character(),
##   BillNum = col_double(),
##   Title = col_character(),
##   Major = col_double()
## )
```

```
congress_val = read_csv(here("data", "congress_val.csv"))
```

```
## Parsed with column specification:
## cols(
##   BillID = col_character(),
##   BillNum = col_double(),
##   Title = col_character(),
##   Major = col_double()
## )
```

Prepare Data

```
max_words = 10000
max_length = 100
x_train = congress_train$Title[1:270000]
y_train = congress_train$Major[1:270000] %>% as.numeric()
tokenizer_train = text_tokenizer(num_words = max_words) %>% fit_text_tokenizer(x_train)
sequences_train = texts_to_sequences(tokenizer_train, x_train)
data_train = pad_sequences(sequences_train, max_length)
```

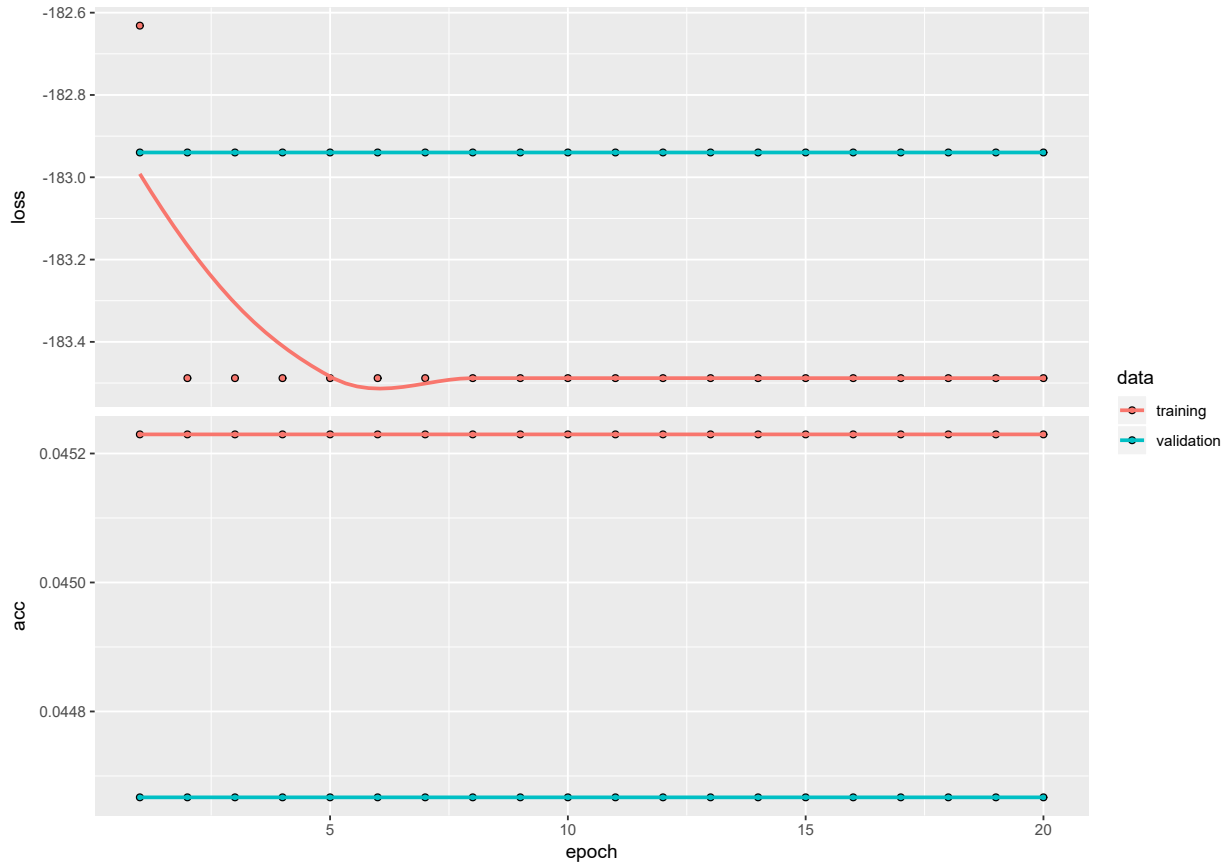
```
x_valid = congress_val$Title
y_valid = congress_val$Major %>% as.numeric()
tokenizer_valid = text_tokenizer(num_words = max_words) %>% fit_text_tokenizer(x_valid)
sequences_valid = texts_to_sequences(tokenizer_valid, x_valid)
data_valid = pad_sequences(sequences_valid, max_length)
```

Initial Model

```
init_model = keras_model_sequential() %>%
  layer_embedding(input_dim = max_words, output_dim = length(unique(congress_train$Major)), input_lengt
  layer_flatten() %>%
  layer_dense(units = 1, activation = "sigmoid")

init_model %>% compile(optimizer = "rmsprop",
                       loss = "binary_crossentropy",
                       metrics = c("acc")
)

summary(init_model)
```

```
## _________________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =========================================================================
## embedding (Embedding)        (None, 100, 21)           210000
## _________________________________________________________________________
## flatten (Flatten)            (None, 2100)              0
## _________________________________________________________________________
## dense (Dense)                (None, 1)                 2101
## =========================================================================
## Total params: 212,101
## Trainable params: 212,101
## Non-trainable params: 0
## _________________________________________________________________________
```

```
init_history = init_model %>% fit(data_train, y_train, epochs = 20, batch_size = 32, validation_data = 

plot(init_history)
```



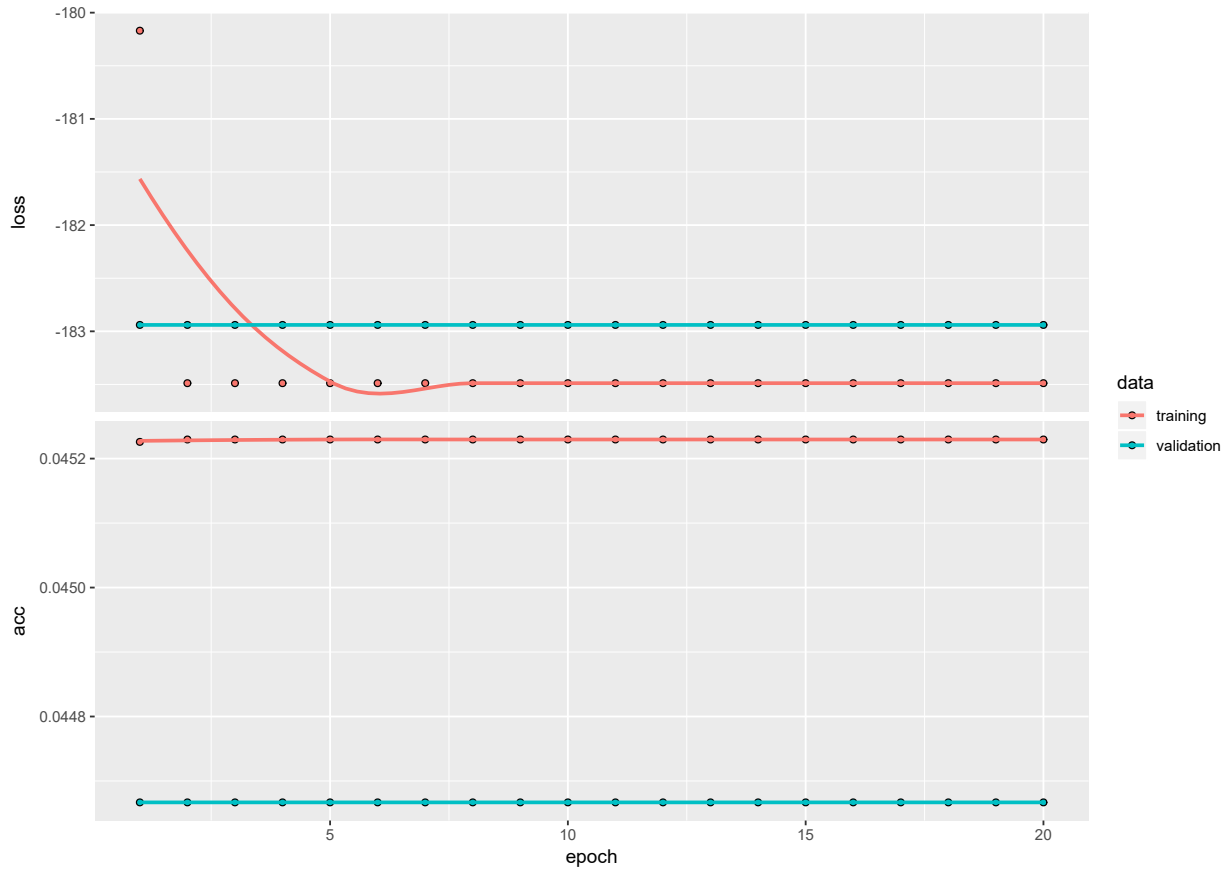
Simple RNN

```
simple_rnn = keras_model_sequential() %>%
  layer_embedding(input_dim = max_words, output_dim = 32, input_length = max_length) %>%
  layer_simple_rnn(units = 32) %>%
  layer_dense(units = 1, activation = "sigmoid")

simple_rnn %>% compile(optimizer = "rmsprop",
                       loss = "binary_crossentropy",
                       metrics = c("acc")
)

summary(simple_rnn)
```

```
## ___________________________________________________________________________
## Layer (type)                     Output Shape                  Param #
## ===========================================================================
## embedding_1 (Embedding)          (None, 100, 32)               320000
## ___________________________________________________________________________
## simple_rnn (SimpleRNN)           (None, 32)                    2080
## ___________________________________________________________________________
## dense_1 (Dense)                  (None, 1)                     33
## ===========================================================================
## Total params: 322,113
## Trainable params: 322,113
```

```
## Non-trainable params: 0
## _________________________________________________________________________
```

```
srnn_history = simple_rnn %>% fit(data_train, y_train, epochs = 20, batch_size = 32, validation_data = l

plot(srnn_history)
```



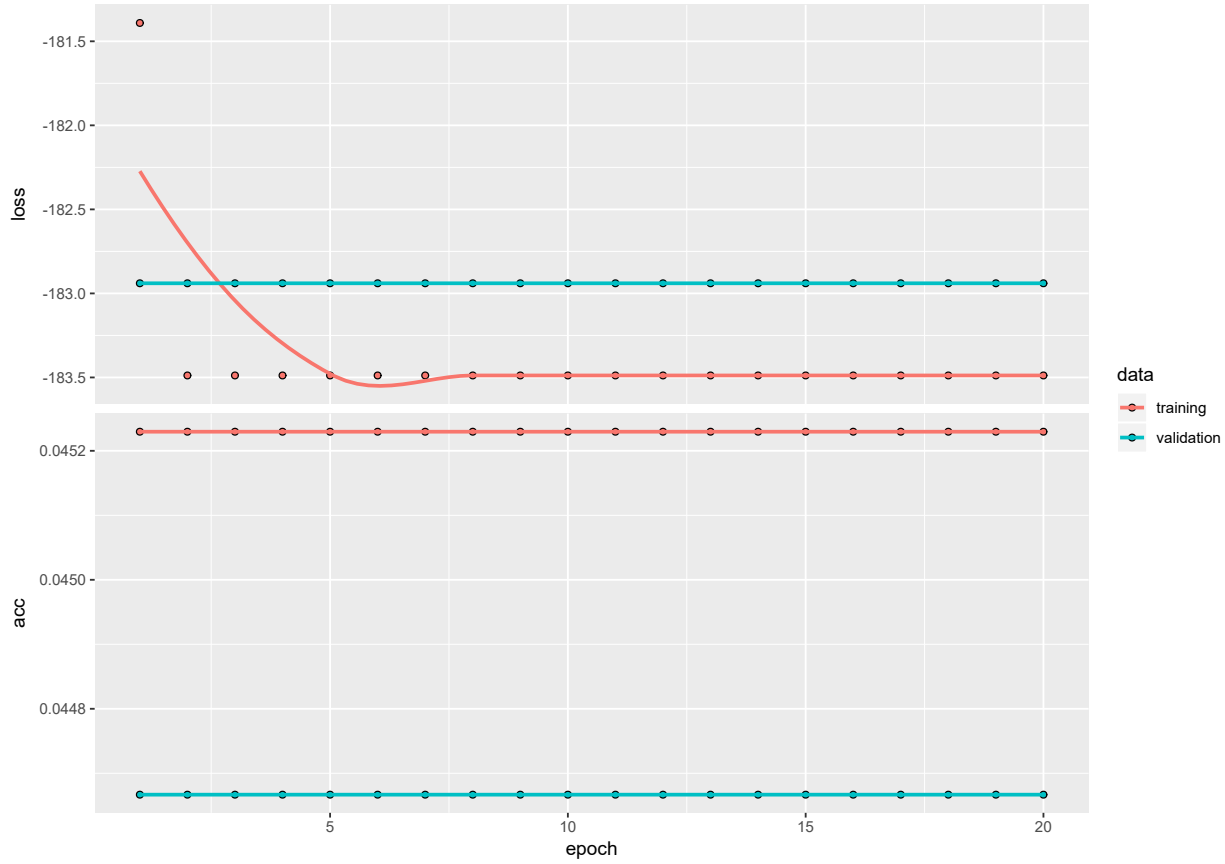
RNN with LSTM Layer

```
lstm_model = keras_model_sequential() %>%
  layer_embedding(input_dim = max_words, output_dim = 32, input_length = max_length) %>%
  layer_lstm(units = 32) %>%
  layer_dense(units = 1, activation = "sigmoid")

lstm_model %>% compile(optimizer = "rmsprop",
                       loss = "binary_crossentropy",
                       metrics = c("acc")
)

summary(lstm_model)
```

```
## _________________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =========================================================================
## embedding_2 (Embedding)      (None, 100, 32)           320000
## _________________________________________________________________________
```

```
## lstm (LSTM)                     (None, 32)                    8320
## ________________________________________________________________________________
## dense_2 (Dense)                 (None, 1)                     33
## ================================================================================
## Total params: 328,353
## Trainable params: 328,353
## Non-trainable params: 0
## ________________________________________________________________________________
```

```
lstm_history = lstm_model %>% fit(data_train, y_train, epochs = 20, batch_size = 32, validation_data = l

plot(lstm_history)
```



RNN with GRU Layer

Hyperparameter Tuning

```
tune_grid = expand.grid(output_dim = c(16, 32, 64), batch_size = c(32, 64)) %>% as_tibble()

for (index in 1:length(tune_grid)) {
  option = tune_grid[index, ]
  lstm_model = keras_model_sequential() %>%
  layer_embedding(input_dim = max_words, output_dim = option$output_dim, input_length = max_length) %>%
  layer_lstm(units = option$output_dim) %>%
  layer_dense(units = 1, activation = "sigmoid")
```

```
  lstm_model %>% compile(optimizer = "rmsprop",
                         loss = "binary_crossentropy",
                         metrics = c("acc")
  )

  summary(lstm_model)

  lstm_history = lstm_model %>% fit(data_train, y_train, epochs = 10, batch_size = option$batch_size, va

  plot(lstm_history)
}
```

```
## _________________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =========================================================================
## embedding (Embedding)        (None, 100, 16)           160000
## _________________________________________________________________________
## lstm (LSTM)                  (None, 16)                2112
## _________________________________________________________________________
## dense (Dense)                (None, 1)                 17
## =========================================================================
## Total params: 162,129
## Trainable params: 162,129
## Non-trainable params: 0
## _________________________________________________________________________
## _________________________________________________________________________
## Layer (type)                 Output Shape              Param #
## =========================================================================
## embedding_1 (Embedding)      (None, 100, 32)           320000
## _________________________________________________________________________
## lstm_1 (LSTM)                (None, 32)                8320
## _________________________________________________________________________
## dense_1 (Dense)              (None, 1)                 33
## =========================================================================
## Total params: 328,353
## Trainable params: 328,353
## Non-trainable params: 0
## _________________________________________________________________________
```