Advanced Data Analytics - Sentiment Analysis – Performance Assessment

Steven Schindler

Advanced Data Analytics – D213

Sewell, Williams; PhD

College of I.T., Western Governors University

A1.

I used amazon reviews on magazine descriptions from the UCSD Recommender Systems Data Sets(McAuley, J. (n.d.).), the research question is thus:

Can a user's opinion on a magazine be predicted as positive or negative based on past reviews from other users?

A2.

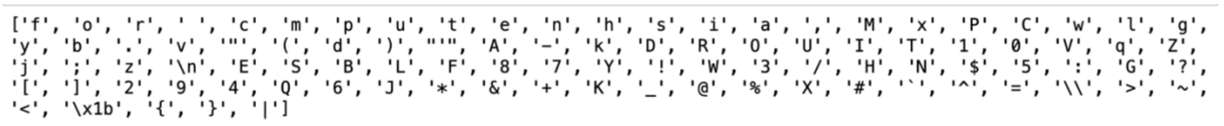
The goal of this analysis is to use sentiment analysis to understand magazine reader sentiment and predict a user's opinion on magazine subscriptions and if those opinion are positive or negative.

A3.

A recurrent neural network (RNN) is a type of artificial neural network which uses sequential data or time series data.(IBM. (n.d.).) It is a type of Deep learning algorithm that can be used for language translation, natural language processing(NLP), sentiment analysis and text classification. They take information from prior inputs to influence current inputs and outputs.(IBM. (n.d.).)

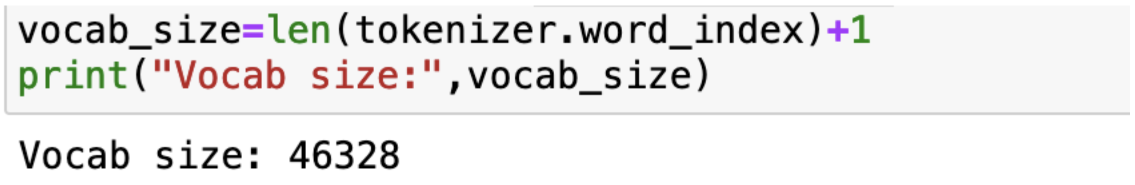
B1.

The presence of unusual characters can be shown by the picture below that has every character from the review_text from the data frame.



```
['f', 'o', 'r', ' ', 'c', 'm', 'p', 'u', 't', 'e', 'n', 'h', 's', 'i', 'a', ',', 'M', 'x', 'P', 'C', 'w', 'l', 'g',
'y', 'b', '.', 'v', '"', '(', 'd', ')', "'", 'A', '-', 'k', 'D', 'R', 'O', 'U', 'I', 'T', '1', '0', 'V', 'q', 'Z',
'j', ';', 'z', '\n', 'E', 'S', 'B', 'L', 'F', '8', '7', 'Y', '!', 'W', '3', '/', 'H', 'N', '$', '5', ':', 'G', '?',
'[', ']', '2', '9', '4', 'Q', '6', 'J', '*', '&', '+', 'K', '_', '@', '%', 'X', '#', '`', '^', '=', '\\', '>', '~',
'<', '\x1b', '{', '}', '|']
```

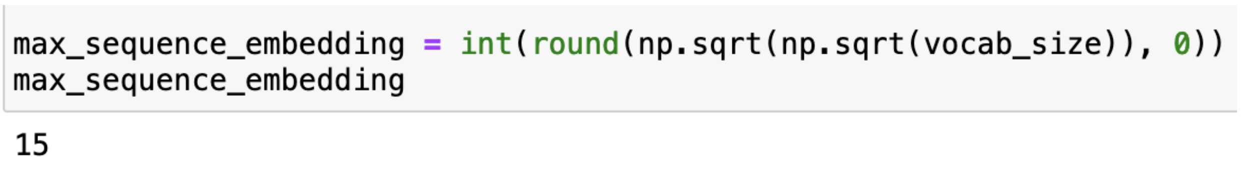
The vocabulary size is the number of unique words in my dataset and is shown in the picture below to be 46,328.



```
vocab_size=len(tokenizer.word_index)+1
print("Vocab size:",vocab_size)
```

```
Vocab size: 46328
```

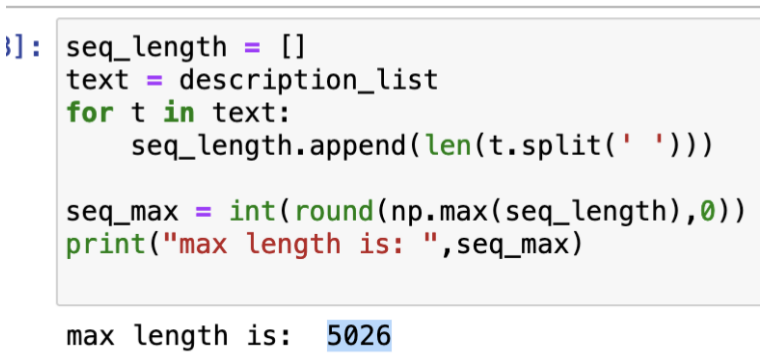
The proposed word embedding length is the square root of the square root of the vocabulary size which rounded to the nearest integer is 15.



```
max_sequence_embedding = int(round(np.sqrt(np.sqrt(vocab_size)), 0))
max_sequence_embedding
```

```
15
```

The sequence length is the length of the longest input sentence in the data set. The max sequence length is shown in the picture below as 5026. This will preserve the available input data so that the generated model can generalize properly and guard against input that does not generalize well.



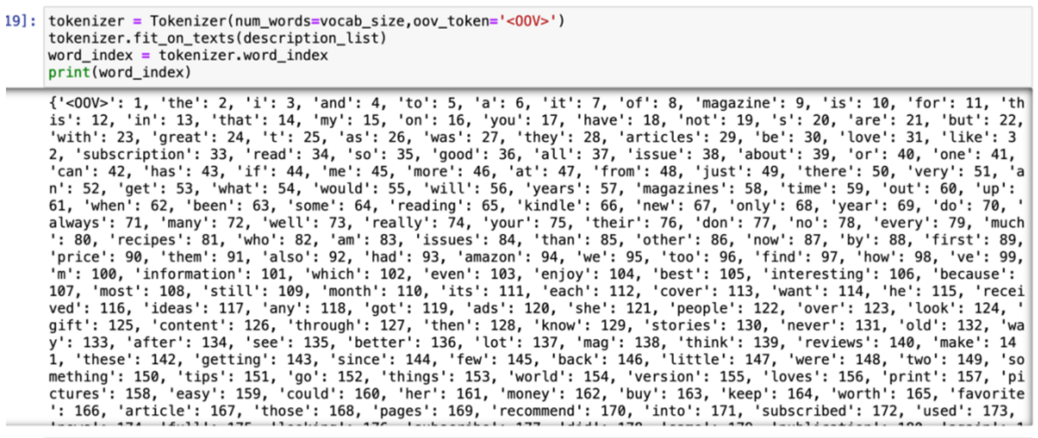
```
]: seq_length = []
   text = description_list
   for t in text:
       seq_length.append(len(t.split(' ')))

   seq_max = int(round(np.max(seq_length),0))
   print("max length is: ",seq_max)
```

```
max length is:  5026
```

B2.

The tokenization process is used to parse the text into smaller chunks also called tokens. These tokens consist of words, characters, and sub-words with a unique index assigned to them. The package Tokenizer is used and imported from tensorflow.keras. Below is a screenshot of the code and an example of the tokenization generated:

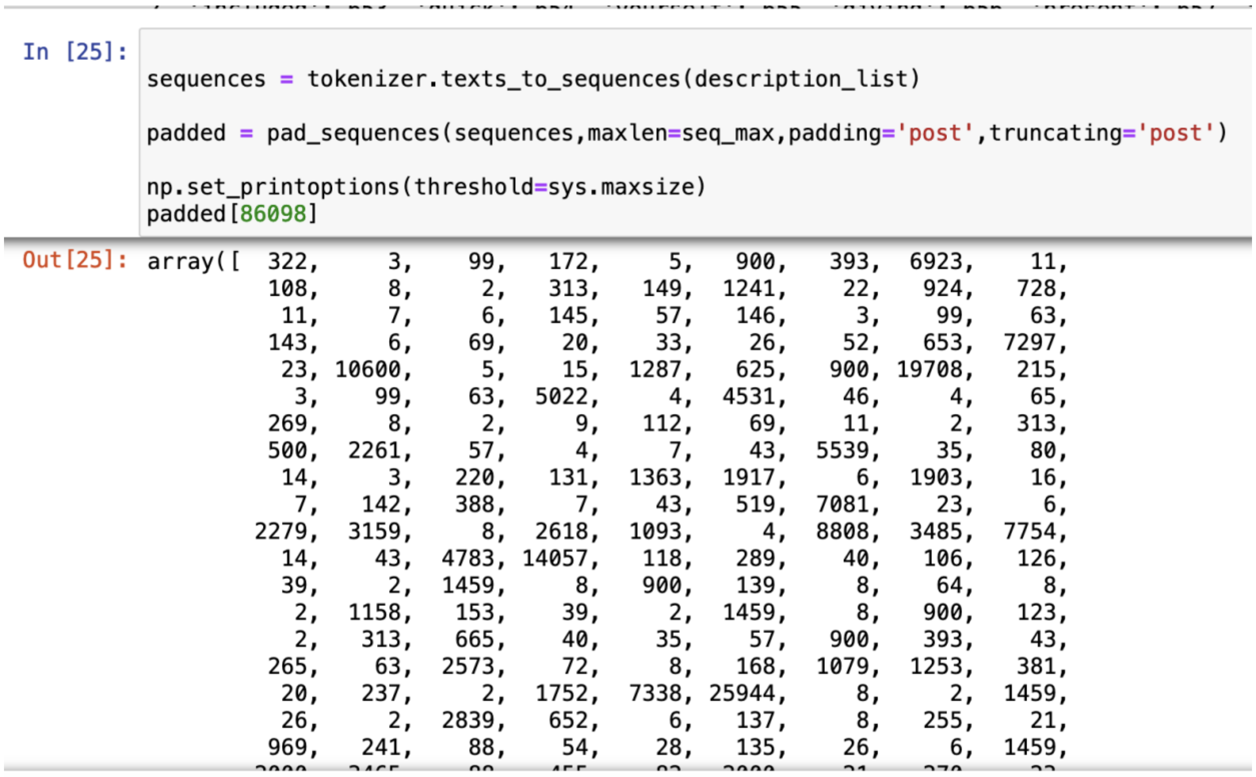


```
19]: tokenizer = Tokenizer(num_words=vocab_size,oov_token='<OOV>')
     tokenizer.fit_on_texts(description_list)
     word_index = tokenizer.word_index
     print(word_index)
```

```
{'<OOV>': 1, 'the': 2, 'i': 3, 'and': 4, 'to': 5, 'a': 6, 'it': 7, 'of': 8, 'magazine': 9, 'is': 10, 'for': 11, 'th
is': 12, 'in': 13, 'that': 14, 'my': 15, 'on': 16, 'you': 17, 'have': 18, 'not': 19, 's': 20, 'are': 21, 'but': 22,
'with': 23, 'great': 24, 't': 25, 'as': 26, 'was': 27, 'they': 28, 'articles': 29, 'be': 30, 'love': 31, 'like': 3
2, 'subscription': 33, 'read': 34, 'so': 35, 'good': 36, 'all': 37, 'issue': 38, 'about': 39, 'or': 40, 'one': 41,
'can': 42, 'has': 43, 'if': 44, 'me': 45, 'more': 46, 'at': 47, 'from': 48, 'just': 49, 'there': 50, 'very': 51, 'a
n': 52, 'get': 53, 'what': 54, 'would': 55, 'will': 56, 'years': 57, 'magazines': 58, 'time': 59, 'out': 60, 'up':
61, 'when': 62, 'been': 63, 'some': 64, 'reading': 65, 'kindle': 66, 'new': 67, 'only': 68, 'year': 69, 'do': 70, '
always': 71, 'many': 72, 'well': 73, 'really': 74, 'your': 75, 'their': 76, 'don': 77, 'no': 78, 'every': 79, 'much
': 80, 'recipes': 81, 'who': 82, 'am': 83, 'issues': 84, 'than': 85, 'other': 86, 'now': 87, 'by': 88, 'first': 89,
'price': 90, 'them': 91, 'also': 92, 'had': 93, 'amazon': 94, 'we': 95, 'too': 96, 'find': 97, 'how': 98, 've': 99,
'm': 100, 'information': 101, 'which': 102, 'even': 103, 'enjoy': 104, 'best': 105, 'interesting': 106, 'because':
107, 'most': 108, 'still': 109, 'month': 110, 'its': 111, 'each': 112, 'cover': 113, 'want': 114, 'he': 115, 'recei
ved': 116, 'ideas': 117, 'any': 118, 'got': 119, 'ads': 120, 'she': 121, 'people': 122, 'over': 123, 'look': 124, '
gift': 125, 'content': 126, 'through': 127, 'then': 128, 'know': 129, 'stories': 130, 'never': 131, 'old': 132, 'wa
y': 133, 'after': 134, 'see': 135, 'better': 136, 'lot': 137, 'mag': 138, 'think': 139, 'reviews': 140, 'make': 14
1, 'these': 142, 'getting': 143, 'since': 144, 'few': 145, 'back': 146, 'little': 147, 'were': 148, 'two': 149, 'so
mething': 150, 'tips': 151, 'go': 152, 'things': 153, 'world': 154, 'version': 155, 'loves': 156, 'print': 157, 'pi
ctures': 158, 'easy': 159, 'could': 160, 'her': 161, 'money': 162, 'buy': 163, 'keep': 164, 'worth': 165, 'favorite
': 166, 'article': 167, 'those': 168, 'pages': 169, 'recommend': 170, 'into': 171, 'subscribed': 172, 'used': 173,
```

B3.

The padding process is used so that every sentence imputed has the same size. Since every sentence could be a different length padding ensures a uniform shape for sentence size. For padding I used post padding and below is a screenshot of a single padded sequence:



```
In [25]:
sequences = tokenizer.texts_to_sequences(description_list)

padded = pad_sequences(sequences,maxlen=seq_max,padding='post',truncating='post')

np.set_printoptions(threshold=sys.maxsize)
padded[86098]
```

```
Out[25]: array([  322,     3,    99,   172,     5,   900,   393,  6923,    11,
                  108,     8,     2,   313,   149,  1241,    22,   924,   728,
                   11,     7,     6,   145,    57,   146,     3,    99,    63,
                  143,     6,    69,    20,    33,    26,    52,   653,  7297,
                   23, 10600,     5,    15,  1287,   625,   900, 19708,   215,
                    3,    99,    63,  5022,     4,  4531,    46,     4,    65,
                  269,     8,     2,     9,   112,    69,    11,     2,   313,
                  500,  2261,    57,     4,     7,    43,  5539,    35,    80,
                   14,     3,   220,   131,  1363,  1917,     6,  1903,    16,
                    7,   142,   388,     7,    43,   519,  7081,    23,     6,
                 2279,  3159,     8,  2618,  1093,     4,  8808,  3485,  7754,
                   14,    43,  4783, 14057,   118,   289,    40,   106,   126,
                   39,     2,  1459,     8,   900,   139,     8,    64,     8,
                    2,  1158,   153,    39,     2,  1459,     8,   900,   123,
                    2,   313,   665,    40,    35,    57,   900,   393,    43,
                  265,    63,  2573,    72,     8,   168,  1079,  1253,   381,
                   20,   237,     2,  1752,  7338, 25944,     8,     2,  1459,
                   26,     2,  2839,   652,     6,   137,     8,   255,    21,
                  969,   241,    88,    54,    28,   135,    26,     6,  1459,
```

B4.

There will be 5 categories of sentiment 1,2,3,4, and 5. A 1, 2, and 3 are used for a negative review while 4 and 5 are positive reviews. The softmax function which calculates the relative probabilities is used as the activation function for the final dense layer of the network.

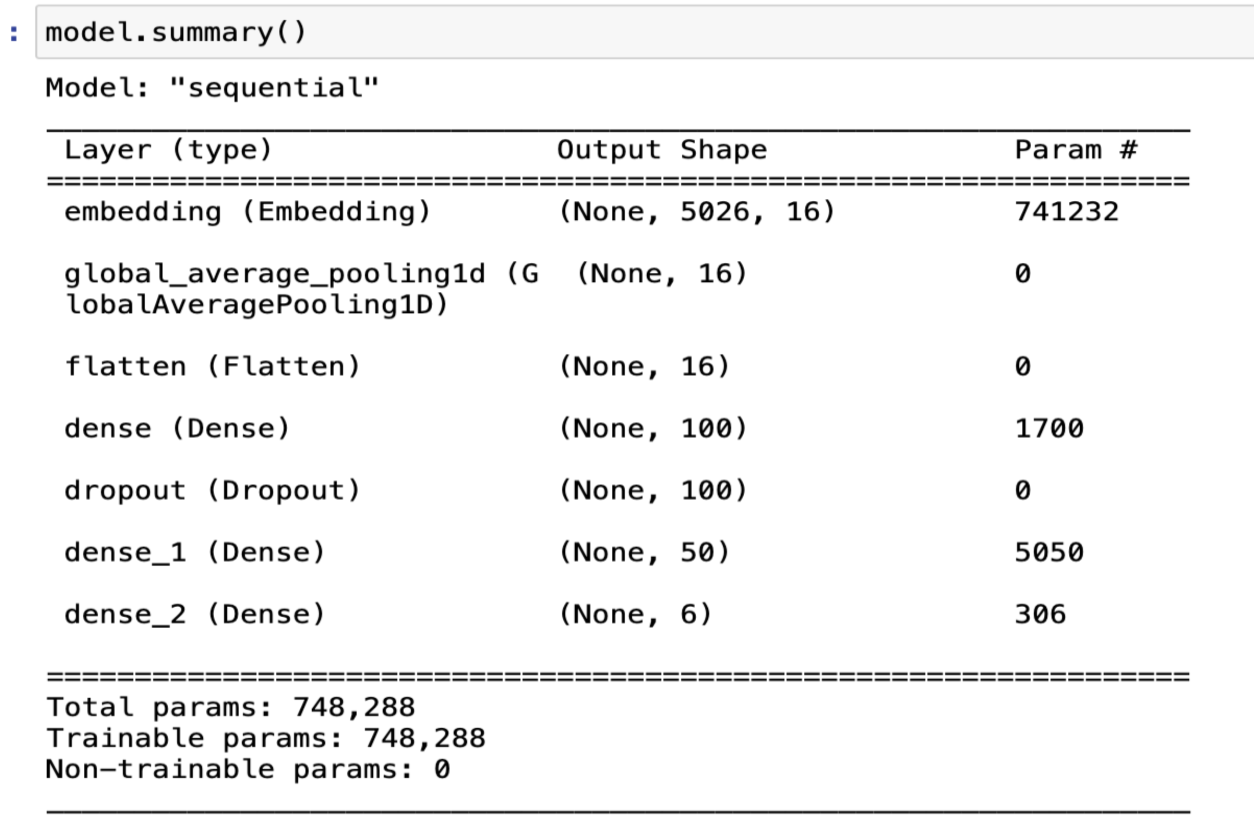
B5.

The first step was to read in the magazine reviews into a data frame. Then check for and remove unusual characters using regex and remove stop words. Explore the data by getting the vocab size, max sequence embedding and the max sequence length. Then split the data 80% and 20% into training and testing data respectively. Then apply the tokenizer onto the training set using the fit_on_text method. Then finally retrieve the word index and pad the sequence using post-padding to get the maximum length.

B6.

The prepared data sets are called padded_x_train.csv, padded_x_test.csv,y_train.csv and y_test.csv.

C1.



```
: model.summary()

Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 5026, 16)          741232

 global_average_pooling1d (G  (None, 16)               0
 lobalAveragePooling1D)

 flatten (Flatten)           (None, 16)                0

 dense (Dense)               (None, 100)               1700

 dropout (Dropout)           (None, 100)               0

 dense_1 (Dense)             (None, 50)                5050

 dense_2 (Dense)             (None, 6)                 306

=================================================================
Total params: 748,288
Trainable params: 748,288
Non-trainable params: 0
_________________________________________________________________
```

The output of the model summary is shown above.

C2.

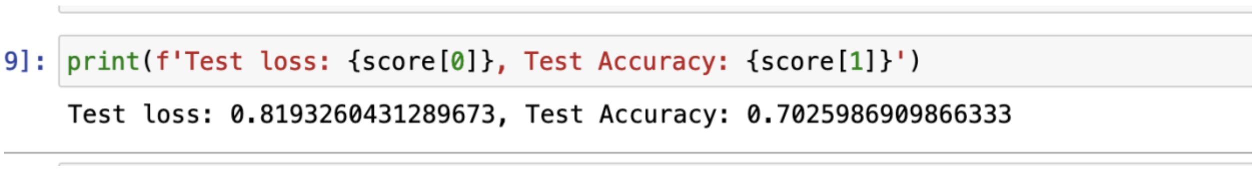
There are seven layers with a total of 748,288 parameters. The first layer embedding has 741,232 parameters. The global_average_pooling layer and the Flatten layer have zero parameters. Then there are three dense layers dense, dense_1, and dense_2 with 1700, 5050 and 306 parameters respectively. The dropout layer between dense and dense_1 has 0 parameters.

C3.

The activation method for dense and dense_1 is relu and for dense_2 it is softmax. Rectified linear activation(relu) is commonly used for hidden layers if the neural network must consider more than two types of sentiment. Softmax generalizes inputs in a logistically so that it yields results as probabilities between 0 and 1. The softmax function is often used as the last activation function to normalize output to a probability distribution between 0 and 1.(Brownlee, J. (2019, January 28).)

The number of nodes in the first dense layer is 100 and then 50 in dense_1. The final dense layer dense_2 has 6 nodes as it is [1:6) and there are 5 outputs 1-5. The loss function is sparse_categorical_crossentropy as sparse_categorical_crossentropy works on an integer target and produces a category index of the most likely matching category. The optimizer used is adam, The Adam algorithm is an extension of the stochastic gradient descent which will update network weights iteratively based on training data. (Cornell University, School of Chemical and Biomolecular Engineering. (n.d.).)

The stopping criteria is the early stopping monitor to stop the model from overfitting the training data and improve generalization. The monitor stops the training as soon as the validation score is not improving. The patience argument it takes is how many epochs the monitor can go without the score improving. The evaluation metric is accuracy which is how well the neural network can classify comments based on sentiment.

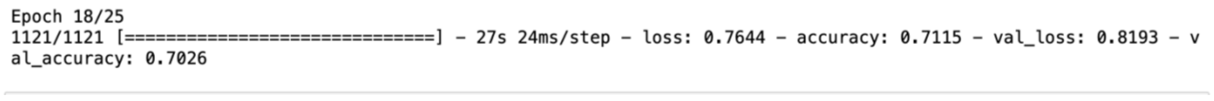


```
9]: print(f'Test loss: {score[0]}, Test Accuracy: {score[1]}')
    Test loss: 0.8193260431289673, Test Accuracy: 0.7025986909866333
```

The model has an accuracy of 70% as shown in the picture above.

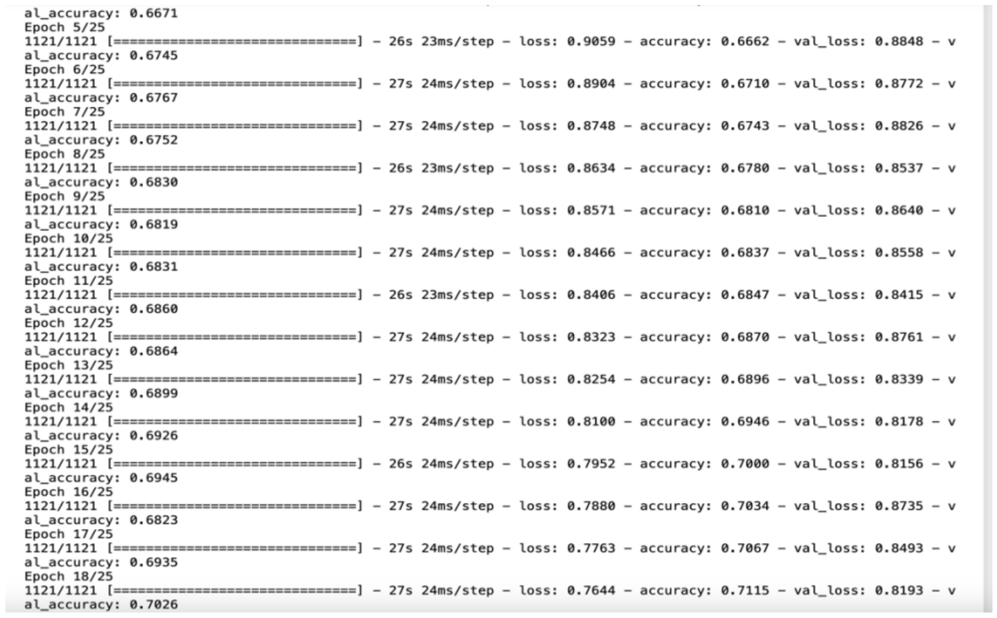
D1.

The stopping criteria has the number of epochs stopped at 18 instead of the setup of 25. The patience = 3 means that the early stop criteria waited 3 epochs and if the model did not improve it stopped the fitting early to prevent overfitting. The final epoch is pictured below:

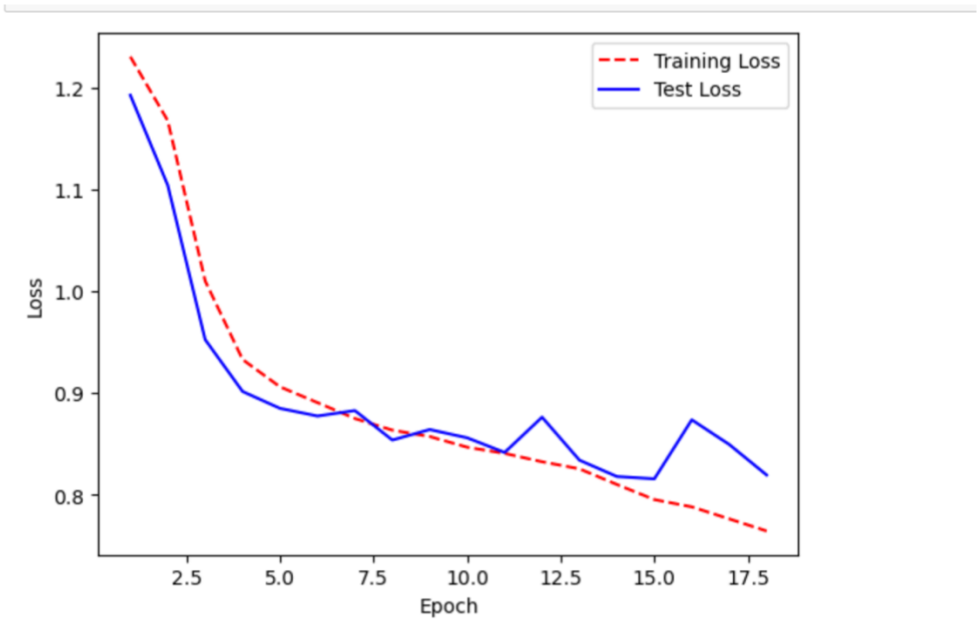


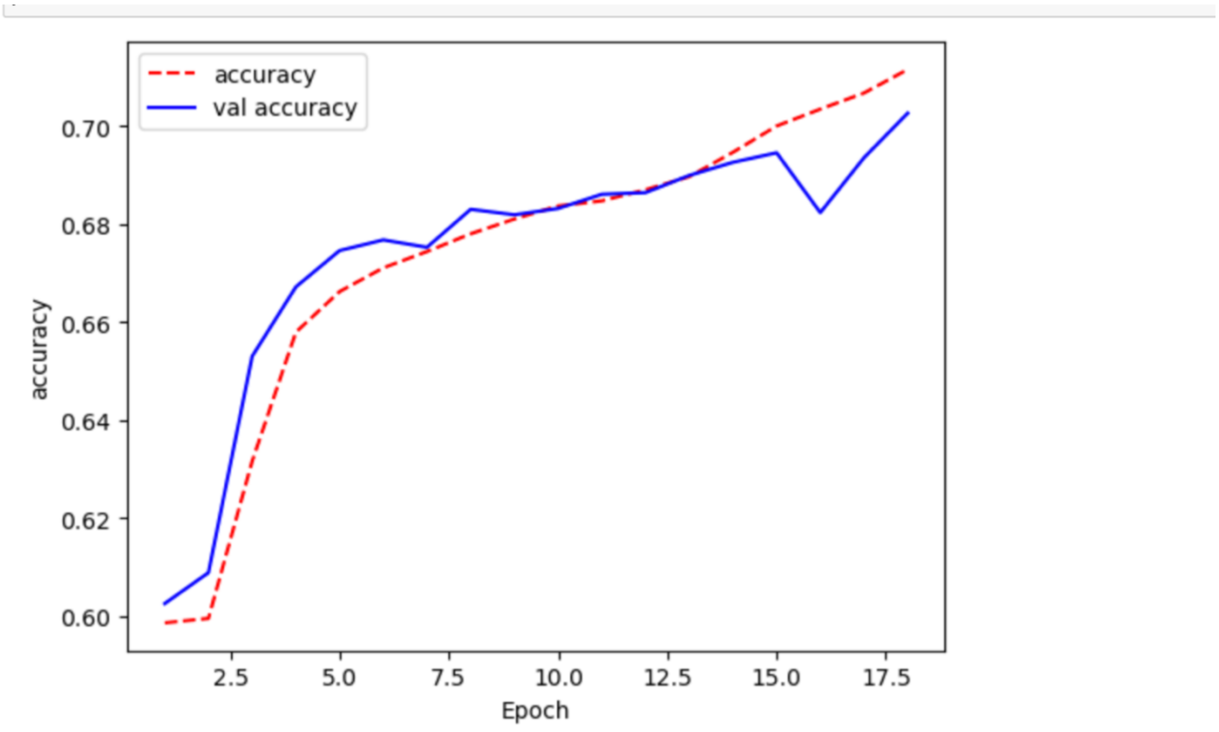
```
Epoch 18/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.7644 - accuracy: 0.7115 - val_loss: 0.8193 - v
al_accuracy: 0.7026
```

D2.



```
al_accuracy: 0.6671
Epoch 5/25
1121/1121 [==============================] - 26s 23ms/step - loss: 0.9059 - accuracy: 0.6662 - val_loss: 0.8848 - v
al_accuracy: 0.6745
Epoch 6/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8904 - accuracy: 0.6710 - val_loss: 0.8772 - v
al_accuracy: 0.6767
Epoch 7/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8748 - accuracy: 0.6743 - val_loss: 0.8826 - v
al_accuracy: 0.6752
Epoch 8/25
1121/1121 [==============================] - 26s 23ms/step - loss: 0.8634 - accuracy: 0.6780 - val_loss: 0.8537 - v
al_accuracy: 0.6830
Epoch 9/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8571 - accuracy: 0.6810 - val_loss: 0.8640 - v
al_accuracy: 0.6819
Epoch 10/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8466 - accuracy: 0.6837 - val_loss: 0.8558 - v
al_accuracy: 0.6831
Epoch 11/25
1121/1121 [==============================] - 26s 23ms/step - loss: 0.8406 - accuracy: 0.6847 - val_loss: 0.8415 - v
al_accuracy: 0.6860
Epoch 12/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8323 - accuracy: 0.6870 - val_loss: 0.8761 - v
al_accuracy: 0.6864
Epoch 13/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8254 - accuracy: 0.6896 - val_loss: 0.8339 - v
al_accuracy: 0.6899
Epoch 14/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.8100 - accuracy: 0.6946 - val_loss: 0.8178 - v
al_accuracy: 0.6926
Epoch 15/25
1121/1121 [==============================] - 26s 24ms/step - loss: 0.7952 - accuracy: 0.7000 - val_loss: 0.8156 - v
al_accuracy: 0.6945
Epoch 16/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.7880 - accuracy: 0.7034 - val_loss: 0.8735 - v
al_accuracy: 0.6823
Epoch 17/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.7763 - accuracy: 0.7067 - val_loss: 0.8493 - v
al_accuracy: 0.6935
Epoch 18/25
1121/1121 [==============================] - 27s 24ms/step - loss: 0.7644 - accuracy: 0.7115 - val_loss: 0.8193 - v
al_accuracy: 0.7026
```





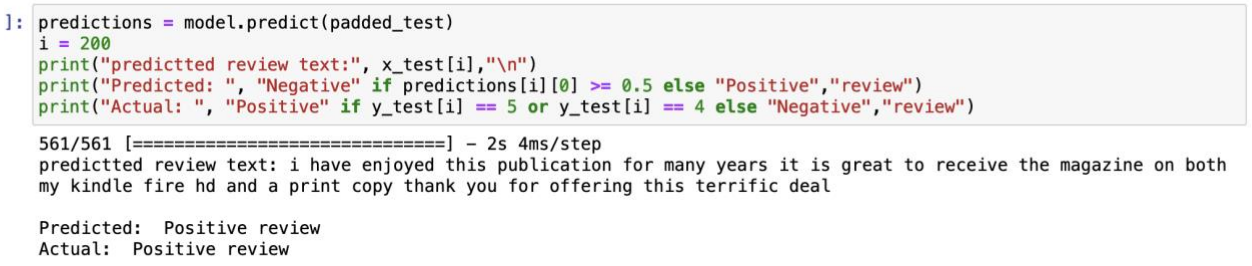
Above is the model's training process with a line graph of loss and the model's accuracy.

D3.

The use of early stopping prevents overfitting. When performance starts to degrade early stopping ends the training to reduce overfitting.(Brownlee, J. (n.d.).)

D4.

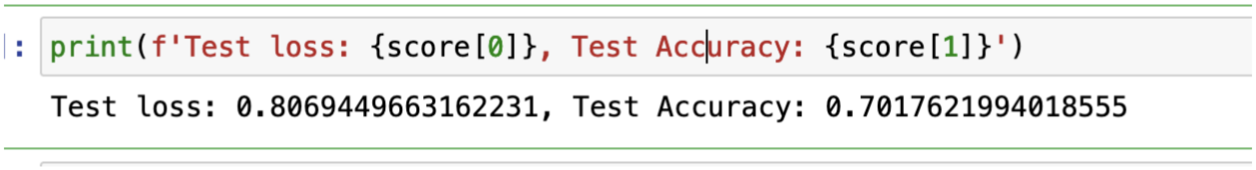
Below are pictures of a sample prediction and the model's accuracy which is 70%:



```
]: predictions = model.predict(padded_test)
   i = 200
   print("predictted review text:", x_test[i],"\n")
   print("Predicted: ", "Negative" if predictions[i][0] >= 0.5 else "Positive","review")
   print("Actual: ", "Positive" if y_test[i] == 5 or y_test[i] == 4 else "Negative","review")

   561/561 [==============================] - 2s 4ms/step
   predictted review text: i have enjoyed this publication for many years it is great to receive the magazine on both
   my kindle fire hd and a print copy thank you for offering this terrific deal

   Predicted:  Positive review
   Actual:  Positive review
```



```
]: print(f'Test loss: {score[0]}, Test Accuracy: {score[1]}')

   Test loss: 0.8069449663162231, Test Accuracy: 0.7017621994018555
```

The loss of the model is .80, the loss computes the distance between the current output of the algorithm and the expected output.(Singh, A. (2021, August 9).) Since my loss function is at .80 for category classification it is not ideal as this means my output is .8 units away from the expected output on an X,Y grid. However, with a model accuracy of 70% my output matches the expected output 70% of the time.

E.

model.save("sentimnent_analysis.h5")

F.

The neural network uses 71,724 reviews to train the data and 17,932 to test the data. The model was trained using the customer reviews and the ratings as labels. We can then use the model to predict sentiment based on reviews and if they will be positive or negative. The network architecture had an impact by giving my model an acceptable degree of accuracy of 70%.

G.

The goal was to predict if a magazine subscription would be positive or negative based on the review text. Amazon can use this information to gain meaningful insights on which magazine subscriptions will be positively reviewed in the future. They can then advertise these magazines to similarly themed magazine subscribers.(e.g., garden magazine to a different garden magazine)

References:

IBM. (n.d.). Recurrent Neural Networks. Retrieved July 3, 2023, from https://www.ibm.com/topics/recurrent-neural-networks

Brownlee, J. (n.d.). Early Stopping to Avoid Overtraining Neural Network Models. Retrieved July 3, 2023, from https://machinelearningmastery.com/early-stopping-to-avoid-overtraining-neural-network-models/

Brown, A. (2018, November 14). How to Increase the Accuracy of a Neural Network. Retrieved July 3, 2023, from https://towardsdatascience.com/how-to-increase-the-accuracy-of-a-neural-network-9f5d1c6f407d

Singh, A. (2021, August 9). What is Loss Function? Retrieved July 3, 2023, from https://towardsdatascience.com/what-is-loss-function-1e2605aeb904

Choudhary, D. (2021, October 11). A Complete Step-by-Step Tutorial on Sentiment Analysis in Keras and TensorFlow. Retrieved July 3, 2023, from https://towardsdatascience.com/a-complete-step-by-step-tutorial-on-sentiment-analysis-in-keras-and-tensorflow-ea420cc8913f

Brownlee, J. (2018, May 16). How to Use Word Embedding Layers for Deep Learning with Keras. Retrieved July 3, 2023, from https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/

Brownlee, J. (2019, January 28). Softmax Activation Function with Python. Retrieved July 3, 2023, from https://machinelearningmastery.com/softmax-activation-function-with-python/

Cornell University, School of Chemical and Biomolecular Engineering. (n.d.). Adam. Retrieved July 3, 2023, from https://optimization.cbe.cornell.edu/index.php?title=Adam

McAuley, J. (n.d.). Amazon product data. Retrieved July 3, 2023, from https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/index.html