Sentiment Analysis

Matthew E. Heino

Advanced Data Analytics

Background

The purpose of the assessment is to process text-based information. This assessment will make use of a few concepts. The first concept is natural language processing. This concept will be used to extract insights from text-based. In this text-based file, there will be reviews from product and service reviews. This information will be read from three files that prove reviews. The reviews were gathered from the following website.

- **https://archive.ics.uci.edu/dataset/331/sentiment+labelled+sentences**

The title of the dataset's website is "UCI Sentiment Labeled Sentences Data Set." This website contained three files. The files are the following:

- **amazon_cells_labelled.txt**
- **imdb_labelled.txt**
- **yelp_labelled.txt**

The data that is contained in these files was in a little different format than what has been given in previous assessments. These files were tab-delimited as opposed to previous files that were separated by ','. So there was a slight modification from how this data was read into the dataframe.

The files consisted of two columns. The first was composed of the text review. This text could be of almost any length. This review text will be tokenized in a later section. This process will be discussed in a subsequent section of the document. The next column that is "common" to all the files that will be used for the assessment is the "label" column. This column is composed of either a zero or a one. These stand for either a positive or a negative sentiment review. The one value indicates a positive sentiment review, while a zero indicates a negative review.

The contents of the file are composed of different amounts of rows for each of the files. The Amazon file contained 1000 rows. Based on research none of these rows contained null values. The next file is the IMDB file this file contained 748 rows and this file also had no null values in the observations. The last file, the Yelp file, contained 1000 rows and these rows all contained observations.

One other item to note is the data types that are found in these columns. The text column was a string or as recorded in the pandas data frame an object. The label column is an integer or as recorded in the pandas data frame int64 data type.

Part I: The Research Question

In this section, there will be a discussion about the research question that was proposed based on the given dataset. There will be a discussion of the objectives and goals that hope to be accomplished after creating the model. There will be an identification of the type of network that will be used to answer the question proposed in this section.

A1. The Research Question. The question that can answered using the data that has been provided is the following: " Is it possible to use the customer's unstructured data to gauge the sentiment of the customer's review based on the words that are in the review?" This question seeks to address the concept of natural language processing by applying it to data that is unstructured. Data in a review is data in a state that does not have any type of structure imposed upon it.

In previous assessments, the data always had some sort of structure. For example, in the medical data set, some numerics ranked the data for the medical surveys. There was an "ordinality" to this data. This is not present in the data set for this assessment. Part of the data

preparation will transform this data into a state that will make it possible to answer the question that was discussed earlier.

A2. Objectives and Goals. The objective of the model is to answer the question with a reasonable accuracy rate. This model's goal is to guess or predict the sentiment of the review based on previous observations that are given in the dataset. The model will successfully train using the training dataset and be able to predict the sentiment category based on this training.

The training and test datasets will be composed of the files that were stated in the Background section of this document. The data in these sets, as stated earlier, are composed of unstructured data. It is not possible to ascertain the sentiment using methods that have been covered in other r course content but will make use of a new type of model that will be discussed in the next section.

A3. Neural Network Identification. The neural network that will be used to answer the question is a sequential neural network. The model will be provided by the Keras Library. The neural network will be composed of three layers. The layers are the following:

- **Embedding layer** – This is a layer that is provided by the Keras library. This layer can be used on the type of data that we are looking to work on. This data is the text of the review. There are a few conditions that need to be met before this layer can be utilized. The text will be integer encoded before this layer is received. This process will be discussed in a subsequent section (Brownlee, 2021).
- **Flattening layer** – This layer will "flatten" the multi-dimensional array produced by the previous layer in the model. This flattening will yield a single or one-dimensional array that will be sent to the next layer (McLean, 2022).

- **Dense layer** – This will be an output layer of the model. There will be one output for this model.

There will be a **Dropout** component to the model. This helped prevent overfitting. This problem was overcome by another method that will be discussed in a future section of this paper (K. Team, n.d.).

The packages that were used to create the model are shown in the table below along with a brief description.

Python Library	Description
tensorflow	• Provided the following layers for the model: ◦ Embedding ◦ Flatten ◦ Dense ◦ Dropout

Part II: Data Preparation

This section will go over the mechanics of how the data was prepared to be used in the model. There will be a discussion on the goals of the tokenization process. A discussion of the padding process. An identification of the sentiment categories as well as the activation function that will be used in the final layer. An explanation of the data steps used to prepare the data.

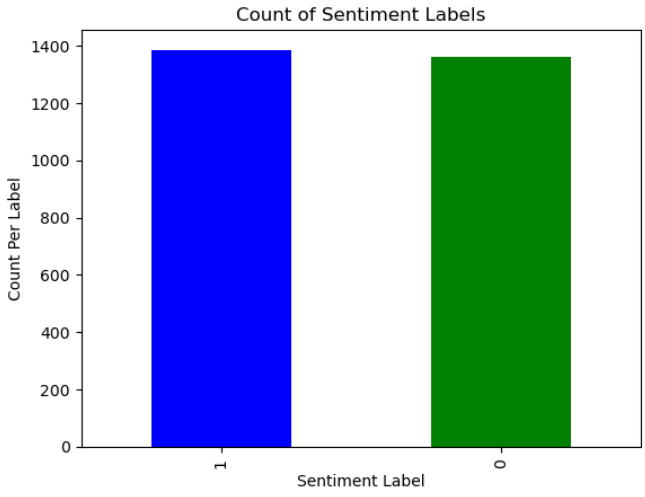
B1. Exploratory Data Analysis. The data that will be used for this assessment is composed of text and numeric data. The text is composed of reviews that have been concatenated from three data files that were discussed earlier. Since this is a data set that is

composed of text some values need to be handled before the text can be used in the model that will be created later. The text is composed of "unusual characters" in the text. The text itself contains punctuation that will need to be removed. The following will be removed from the text:

- “?", ".", ";",":", "!",' " ',","

The vocabulary size was based on the length of the word index and for this model, it will be 2026 as given by the **keras_tokenizer.word_index**. The maximum sequence length was determined through observing the contents of the data and it was concluded that the maximum length should be 64.

Other information while doing an exploration of the data is stated in the following paragraphs. Looking at the composition of the data that is contained in the master dataframe we can see that the data is split almost evenly between the two sentiment groups. There are 1386 observations in the positive group (label 1). This can be expressed in the plot that is shown below.



To help better understand the question it might be beneficial to show how words can have two different connotations based its usage. To explore this idea it was advantageous to see if this is

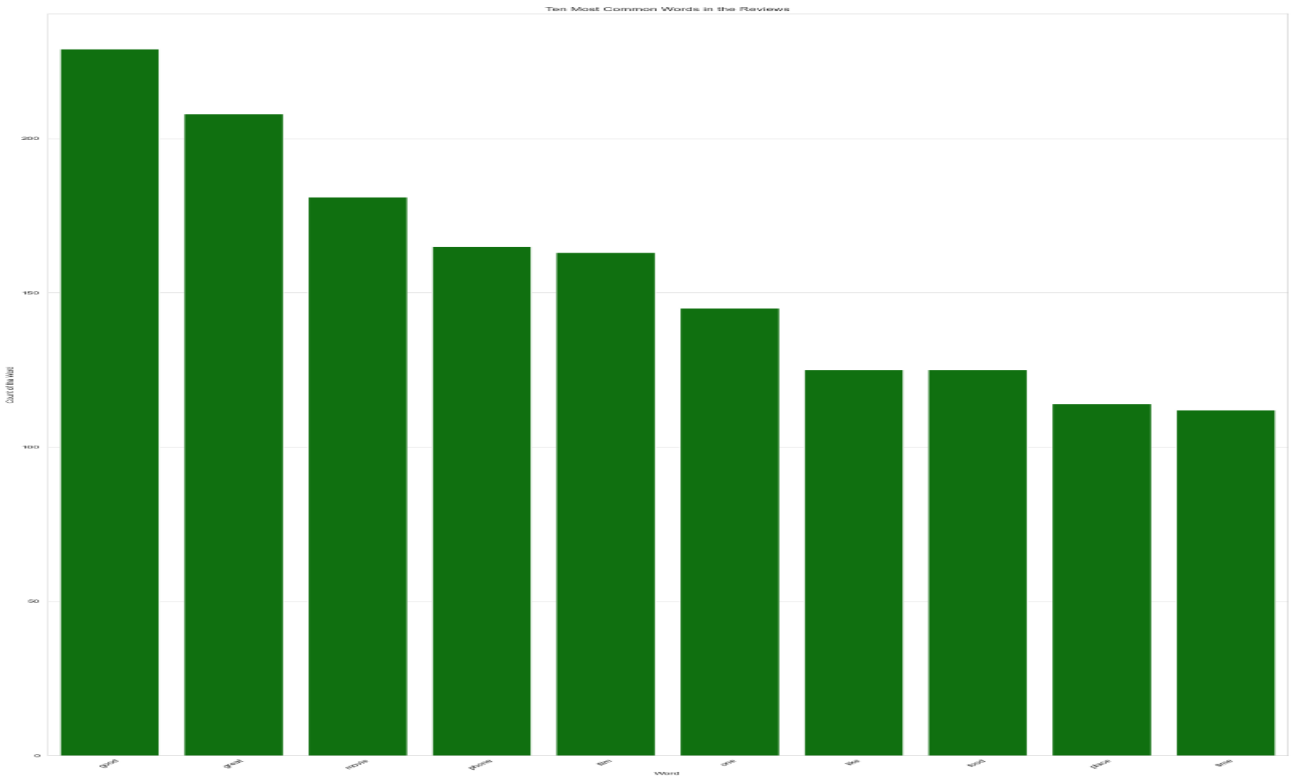
expressed in the data that is in the data frame. There is an output shown below that helps illustrate the idea.

```
Good Connotation:  good
                                                text  label
13                          Very good quality though      1
51  good protection and does not make phone too bu...      1

Bad Connotation:  good
                                                 text  label
81                                Not a good bargain.      0
374  Not a good item.. It worked for a while then s...      0

 ###################################


Bad Connotation:  bad
                                              text  label
59   The buttons for on and off are bad.       0
126  Basically the service was very bad.       0

 ########################################
```

If you review these words you can see that this problem is shown in the data. This dual connotation may cause a problem when using words as a way to predict the sentiment of the text review. This is something that should be kept in mind as the analysis and model creation advance in subsequent sections of the assessment.

When looking at the data it is beneficial to see what words are the most prevalent in the data. Keep in mind that the words that may appear could have dual meanings depending on their context. Creating a histogram can aid in discerning if any words may be detrimental to the model. Dual-connotation words could make it difficult to train the model.

The histogram below shows the top ten words that were found in the data.



(**Note**: For a better view of this histogram, please refer to the Jupyter Notebook.) In this histogram there are two words that may cause problems with the model. The words are good and great. Both of these words are used positively and negatively based on the labels for the associated text review. For the basis of this statement please refer to the cell 12 output of the notebook.

For a graphical representation of the words and their distribution, there is a word cloud produced to show the most prominent words in the data frame. This word cloud does not distinguish between positive and negative words but is used to illustrate the types of words and how they are distributed in the dataset.

B2. Goals of Tokenization. There will be two stages of tokenization employed in this assessment. The first tokenization will be undertaken to create a list of separate words that once composed the text string. This will break the sentence of the review into a list of individual words that will be later used in other sections of the assessment. An example of this first tokenization can be found in cell 15 of the Jupyter Notebook. Below is a screenshot of what an example of the tokenized will look like.

```
BEFORE: nice headphones for the price and they work great
AFTER : ['nice', 'headphones', 'for', 'the', 'price', 'and', 'they', 'work', 'great']
```

As you can observe the text has been broken up into individual words that can be processed later in the creation of data that will be suitable for use in the sequential model.

The next time the tokenized data will transformed into a numeric representation of the words that are found in the tokenized reviews. This application is shown in cells 24 through 25 of the Jupyter Notebook. This process takes the tokenized words and converts them into an array of numerics. There are a few methods that are employed here. The first is the **fit_on_texts** method and the other is **texts_to_sequences**. For further information about what each of these methods do please refer to the Jupyter Notebook.

B3. Padding Process. The padding process is a process in which the data is encoded to be of the same length. This is to help deal with data that may be of different lengths and to help ensure that the sequences of data are all of the same length (Understanding Masking & Padding, n.d.). There are a few options for padding the data sequences. The one that will be utilized in this assessment will be posted. This method will add the appropriate number of zeros.

To bring the length of the data to an appropriate length. The method that will be used is found in the Tensorflow library. The method is **pad_sequences** it has been given the alias "ps" in the application. The code for this can be found in cell 28 and the output can be found in cell 29 of the notebook. An example of the padding is shown below.

```
array([ 94,   8,  44, 358, 193,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,
          0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0,   0])
```

B4. Categories of Sentiment. The number of sentiment categories in this dataset is two. There is one for positive and this is represented by a one in the labels column of the dataset. The negative sentiment is represented by a zero in the labels column of the dataset.

The activation function chosen for the last Dense layer of the model will be the sigmoid function. This activation will make numbers that will be between the values of zero and one. We

are looking to create values that are either zero or one. The sigmoid is a simplified version of the **Softmax** function – the 2-element Softmax function (*tf.keras.activations.sigmoid*, n.d.). These results will match the desired sentiment outcome of the model. We want to have results that are sentiment numbers.

B5. Data Preparation Steps. The data that will be used in this assessment must be prepared in a way that can be used to create a model. In this section, the steps will be briefly discussed and where appropriate elaborated on. The steps that were used to prepare the data are the following:

1. The data currently exists in three different files as stated earlier in this document. The first step will be to concatenate this data into one master data frame. The method that was used to accomplish this was the pandas method **concat()**. This method will concatenate the separate data frames that were created from each of the individual files. To make it easier to reference the elements in the data frame the index for the master data frame will be reset to reflect the addition of these elements. The code for this can be found in cells 8 and 9 in the notebook or Part I.
2. The next step is to remove the punctuation from the reviews that are found in the master data frame. Using the **apply** function a method was created that would use a list of punctuation that will remove the punctuation from the string text that is the review. The method that was created is called **remove_function**. This function can be found in the function section of the notebook. The actual execution of the method can be found in cell 13.
3. To make sure that there is no distinction between upper can lower case that may mean the same thing the words in the sentence will be changed from the word's current case to that

of lowercase. The code for this can be found in cell 14 and an example of the result is shown below.

```
BEFORE being changed to lowercase: Nice headphones for the price and they work great

AFTER being changed to lowercase: nice headphones for the price and they work great
```

4. For the text review to be used in the model there is a need to break up the string into its component words. This is referred to as tokenization. The method that will be used in this assessment will be from the **nltk** library. The method that was executed on the text was the **reg_exp.tokenize**. This was used after creating a **RegexpTokenizer** object. This code can be found in cell 15. An example of the result of executing this method is shown below.

```
BEFORE: nice headphones for the price and they work great
AFTER : ['nice', 'headphones', 'for', 'the', 'price', 'and', 'they', 'work', 'great']
```

5. There are many words that are found in the review. Not all of these words are of use to the model. They do not offer any value and these will be removed. The common stopwords will be removed from the tokenized text that was created in the previous step. The stopwords will be downloaded from a repository and used to filter out the words that are match to ones that are in the review tokens. There will be a lambda function used to filter out the words that are in the stopwords list. Code for this step can be found in cell 16 and you will find an example of the results of executing this code. As you can see below some of the words have been removed from the list of tokens.

```
BEFORE revoving the stopwords: nice headphones for the price and they work great
AFTER removing the stopwords: ['nice', 'headphones', 'price', 'work', 'great']
```

6. The next step is to remove words that are short and infrequent. Any words that are too short will be removed as they may be of no real value. The code for this can be found in cell 17 of the notebook.
7. Words often have tenses that have the same meaning. For example, words like change and changing are just words with different tenses. These words should be considered one word and not considered two different words. It is possible to convert these words to a single tense by lemmatization of the word. This means that the words will be reduced to one tense of the word and this will be used to create the model in later steps. This was accomplished by using the **WordNetLemmatizer** object and using the **lemmatize** function that is a part of the class. This class can be found in nltk. The code to accomplish this can be found in cell 19.
8. The data was split into training and test sets. The splitting was based on an 80/20 split. Where the test data constituted 20% of the data while the training set was composed of 80% of the data. This seems to be the average for creating sets of this type.

These are the steps used to create the cleaned data set for the assessment.

B6. Cleaned Data.

A copy of the cleaned data can be found in the following file.

- **Heino D213 Task 2 Cleaned.csv**

The code can be found in the relevant section of the notebook.

Part III. Network Architecture

In this section, there will be a discussion of the model summary. A discussion of the number of layers that compose the model. A justification of the parameters that were used in the creation of the model.

C1. Model Summary. The model summary is shown below the summary was created by executing the summary method. A screenshot of the summary is shown below.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding (Embedding)       (None, 64, 1)             2026

 dropout (Dropout)           (None, 64, 1)             0

 flatten (Flatten)           (None, 64)                0

 dense (Dense)               (None, 1)                 65

=================================================================
Total params: 2,091
Trainable params: 2,091
Non-trainable params: 0
_________________________________________________________________
```

As stated in a previous section the model is composed of the following components:

- Embedding layer
- Dropout component
- Flatten (Flattening) layer
- Dense layer (the output layer)

Each of these layers and their purpose will be discussed in the next section of this part of the paper.

C2. The Layers and Parameters. The model as stated earlier is composed of quite a few layers. The layers can be described in the following manner.

- Embedding layer – the embedding layer will be used to convert each of the words in the data into a vector of a defined size (Brownlee, 2021).
- Flattening layer – This layer will be used to reduce the shape and dimension of the input layer (McLean, 2022).

- Dropout layer- The dropout layer will be the layer that will be applied to the model. This layer will be set at a specific value that selects a specific number of neurons to zero. The number chosen for this model was 0.05. This will set 5% of the neurons to a zero state.
- Dense layer – this will be the output layer. This layer will combine all the neurons that compose the model. This combination will become the output of the model.

The total number of parameters for this implementation of the model is the following:

- Embedding layer – The number of parameters is 2026.
- Dense layer – There were 65 parameters.

The total number of parameters was 2091 for all relevant layers. All these parameters are trainable. This information was derived from the model summary that was given in a previous section of the document.

C3. Justification of Hyperparameters. In this section, there will be a discussion of the activation function, the number of nodes per layer, the loss function, the optimizer, the stopping criteria, and the evaluation metric.

The activation function that was chosen for this model is the sigmoid function. This activation function is used for neural networks that are used to "classify" outputs. This function is specified in the Dense layer. (Please refer to the Functions section and cell 31 for the code.)

The number of nodes is the number of parameters that are associated with each layer. This information is given in the summary of the model.

The loss function is the function that helps to determine how good the neural network is at performing a certain task. In the case of this model, how well can it predict the sentiment given a list of review words? The loss function that was used in this model was "binary crossentropy." This loss function will measure the difference between the predicted binary

outcome and the actual outcome (Saxena, 2023). The loss function is specified when the model is compiled. Please refer to cell 32 for the code.

The optimizer used for this model is "adam." This optimizer can be used to minimize the loss function during the training portion of creating a neural network (Vishwakarma, 2023). This means that we are trying to find a value that minimizes the loss of data while training the data using the model.

The stopping criteria is a concept that can be employed to stop the model while it is still going through iterations of training. For a model to be effective we must stop training the model when there is no more advantage to continue to train the model. To stop the model there must be some criteria or metrics that can be used to halt the fitting of the model. There are several metrics to monitor. The one that was employed in this model was "val_loss."

This metric is applied to the test set and is akin to the loss that is calculated for the training set. We want to see a low value for the val_loss. When we see the value stop decreasing or the value starts to rise again we want to stop the model at this point. This is to make sure we do not begin to overfit the model.

The evaluation metric that will be used to gauge the performance of the model will be accuracy.

Part IV: Model Evaluation.

In this section of the assessment, there will be a discussion of the stopping criteria. There will be an assessment of the model. Visualizations of the model's training process. A discussion on the model's predictive accuracy.

D1. Stopping Criteria. When using stopping criteria there must be a metric chosen to ensure that the model performs well on data that the model has not seen before. The stopping criteria that was chosen was "val_loss." We want to stop the model when the value for these metrics ceases to go down. When the value for these metrics starts to rise it could mean that the model is beginning to overfit the data.

The number of epochs that was defined for the model was 300. This number of epochs was never reached using the stopping criteria. During multiple runs of the model, the highest epoch reached was 48 before early stopping prevented the model from running any more epochs.

Other parameters were used in the creation of the early stopping callback. The other parameter that was set was the patience parameter. This parameter is how many epochs to pass through before terminating the fitting of the model. It looks to see if there is any improvement in the val_loss metric before ending the fitting.

The screenshot shows the last epoch that was entered before the process was terminated.

```
Epoch 48/300
69/69 [==============================] - 0s 3ms/step - loss: 0.1820 - accuracy: 0.9381 - val_loss: 0.4339 - val_accuracy: 0.8036
Epoch 48: early stopping
```

To view the full output of the fitting of the model you can refer to cell 33's output. This will give a listing of all the loss/accuracy values through all the executed epochs.

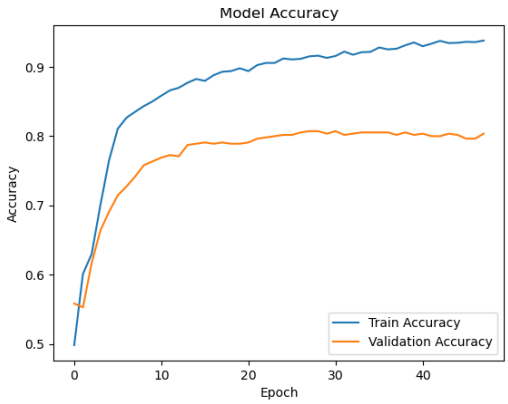
D2. Fitness of the Model. The model was stopped after running through 48 epochs. The model performed adequately it managed to obtain an 80.3% percent accuracy on the test data. This was done with the model as described in previous sections of the assessment. The model voided overfitting through the use of stopping criteria that were discussed in the previous section. This procedure of utilizing the dropout layer is to ensure that the neurons of the model do not develop an "interdependency" among the neurons of the model. This will allow the

network to develop a robust model and be able to generalize to the features of the dataset (Educative, n.d.).

This made sure that the model stopped when it was at its peak. While not a perfect model as it is only successful with 93.8% of the training data and only 80.3% of the test data. It is still a model that can be relied upon to be successful in predicting the sentiment around 80% of the time.

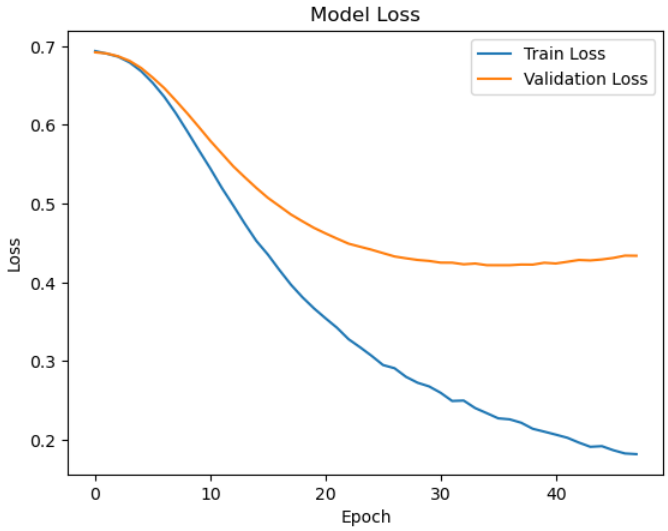
The model did exhibit a degree of overfitting, so there seems to be a problem with the stopping criteria. One method that could be employed is to reduce the size of the training. This is to avoid the model recognizing patterns that are only found in the training and it is not indicative of the dataset as a whole. Other methods would be to utilize a more complete training set that would encompass a wider and more diverse set of observations. This would better represent observations that would happen in the real world. The caveat with including a more diverse data set is that this new data would need to have more and different inputs that are not exhibited in the original dataset (*Overfit and Underfit*, n.d.).

D3. Visualization of the Model. In this section, you will see a visualization of the model as it pertains to the accuracy it achieved during each of the epochs. The first visualization shows the training process as it passes through each epoch.



Reviewing the graph above you can see that model training in regards to the accuracy of the model. It follows the training process through the 48 epochs that were completed before the stopping criteria were reached.

The metric that was used to stop the model from continuing was the val_loss metric this metric measures how loss occurs with the test data. The graph below shows the progression of the follows through 48 epochs of the model.



As you can see from the graph above the model val_loss reached a minimum of 43.3% This is the lowest value obtained using the parameters and the initial model. This finding will be discussed in a subsequent section of the assessment.

D4. Predictive Accuracy. The model as it is currently configured produced a model that is accurate with the test data 80.3% of the time. Using the **evaluate** function that is available also proved that the model is as accurate as it's stated previously. Output from the evaluation function is shown below.

```
18/18 [==============================] - 0s 2ms/step - loss: 0.4339 - accuracy: 0.8036

Test loss:  0.4339071810245514
Test accuracy:  0.803636372089386
```

Part V. Summary and Recommendations.

In this section, there will be references to the code that will be used to save the model. A discussion of the functionality of the neural network. A recommended course of action for the model.

E. Code to save the Model. The code that was used to save the model can be found in cell 38, Part V Summary and Recommendations. The model was saved using the save function. Please refer to this section of the Jupyter Notebook for this code.

F. Functionality of the Neural Network. The model was created with a select group of parameters and other settings. While the model was able to obtain modest results. The accuracy was approximately 80.3%. The model can make a few predictions for the sentiment of a customer's review. This is still a decent percentage of correct predictions. This could be further enhanced using other parameters and the addition of other layers that could improve the functionality and predictive accuracy of the model. This will be discussed in the next section.

G. Course of Action. Based on the results of the model there is room for improving the model. Avenues to pursue to increase the accuracy of the model could include the inclusion of other layers in the model. Changing the parameters that are in the current model to see if

modifying these could yield better results. While creating this model the author tried numerous different metrics to see what they accomplished in terms of see what could make the model better.

Changes to the number and types of layers that composed the model were tried and there were no better ones, but this does not mean that with more practice in creating the model these other configurations could not provide a better model than the one that was created in this assessment.

References

Part VI: Reporting

H. Industry Relevant IDE.

The industry-relevant IDE that was chosen for this assessment was a Jupyter Notebook. There will be a PDF of the notebook included with the submission of the assessment.

- File name: **Heino D213 Task 2 NLP V3.pdf**

I. Web Resources

This section will contain resources that were not provided by the university or the videos provided by DataCamp.

Brownlee, J. (2020, August 25). *Use early stopping to halt the training of neural networks at the right time*. MachineLearningMastery.com. Retrieved February 6, 2024, from https://machinelearningmastery.com/how-to-stop-training-deep-neural-networks-at-the-right-time-using-early-stopping/

tf.keras.preprocessing.text.Tokenizer. (n.d.). TensorFlow. Retrieved February 4, 2024, from https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/text/Tokenizer

Vu, D. (2023, February 23). *Generating WordClouds in Python Tutorial*. DataCamp. Retrieved February 7, 2024, from https://www.datacamp.com/tutorial/wordcloud-python

What does Keras Tokenizer method exactly do? (n.d.). Stack Overflow. Retrieved February 6, 2024, from https://stackoverflow.com/questions/51956000/what-does-keras-tokenizer-method-exactly-do

J. In-text Citations

This section will contain any in-text citations that were used to create or referenced in this document. All code references will be found in the previous section.

Brownlee, J. (2021, February 1). *How to Use Word Embedding Layers for Deep Learning with Keras*. MachineLearningMastery.com. https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/

Educative. (n.d.). *Educative Answers - trusted answers to developer questions*. Retrieved February 12, 2024, from https://www.educative.io/answers/keras-dropout-layer-implement-regularization

McLean, K. (2022, January 5). HOW TO USE Keras.layers.flatten() - Kevin McLean - Medium. *Medium*. Retrieved February 8, 2024, from https://kevinmlean.medium.com/how-to-use-keras-layers-flatten-c3f29ed1b686

Saxena, S. (2023, September 13). *Binary Cross Entropy/Log loss for binary classification*. Analytics Vidhya. Retrieved February 9, 2024, from https://www.analyticsvidhya.com/blog/2021/03/binary-cross-entropy-log-loss-for-binary-classification/

Team, K. (n.d.). *Keras documentation: Dropout layer*. Keras. Retrieved February 8, 2024, from https://keras.io/api/layers/regularization_layers/dropout/

tf.keras.activations.sigmoid. (n.d.). TensorFlow. Retrieved February 9, 2024, from https://www.tensorflow.org/api_docs/python/tf/keras/activations/sigmoid

Overfit and underfit. (n.d.). TensorFlow. Retrieved February 12, 2024, from https://www.tensorflow.org/tutorials/keras/overfit_and_underfit

Understanding masking & padding. (n.d.). TensorFlow. Retrieved February 9, 2024, from https://www.tensorflow.org/guide/keras/understanding_masking_and_padding

Vishwakarma, N. (2023, December 21). *What is Adam Optimizer?* Analytics Vidhya. Retrieved February 9, 2024, from https://www.analyticsvidhya.com/blog/2023/09/what-is-adam-optimizer/