Performance Assessment 2

D213 – Advanced Data Analytics

Jessica Hosey

MSDA, College of Information Technology

Western Governors University

January 19th, 2025

Part I: Research Question

A. 1. Summarize one research question you will answer using neural network models and NLP techniques. Be sure the research question is relevant to a real-world organizational situation, and that sentiment analysis is captured in your chosen data set(s).

How accurately can neural network models predict whether a reviewer would recommend a product based solely on the sentiment and content of their review?

The datasets contain text reviews and labels' 0' and '1', indicating an overall positive or negative review. No neutral responses are included in the dataset. If the reviewer seems positive about the product/experience, they would most likely recommend it to another person. If their review/experience were negative, they would likely not recommend the product to another person. We can use information from a neural network and NLP techniques to generally predict whether a customer would recommend (or not recommend) the product based on the type of words they use in their review.

A. 2. Define the objectives or goals of the data analysis. Be sure the objectives or goals are reasonable within the scope of the research question and are represented in the available data.

- Build a predictive model using NLP techniques to predict whether a reviewer would recommend a product (labeled '1') or not (labeled '0') based on the sentiment and text of their review.
- Understand key predictive features by analyzing the text to identify words, phrases, or patterns that contribute most to the model's predictions.
- Evaluate the model and assess its accuracy in predicting recommendations using precision, recall, a f-1 score, and accuracy to ensure the results are reliable and meaningful to the project's goals.
- Translate the model's outputs into insights/trends that the business can use to address product concerns to increase the likelihood of recommending products to another potential customer.

A. 3. Identify a type of neural network capable of performing a text classification task that can be trained to produce useful predictions on text sequences on the selected data set.

The neural network used for this classification task is a Long Short-Term Memory (LSTM) network known as Recurrent Neural Network (RNN).

Part II: Data Preparation

B. 1. Perform exploratory data analysis on the chosen data set and include an explanation of each of the following elements: the presence of unusual characters, vocabulary size, proposed word embedding length, and statistical justification for the chosen maximum sequence length.

The first step is to concat all three datasets into one large data frame. Then, look at the data available to see if there are any problems with column titles, structure of the tables, etc. Once everything looks okay, we will ensure no null or empty cells are in the table. If there is, we will deal with them appropriately.

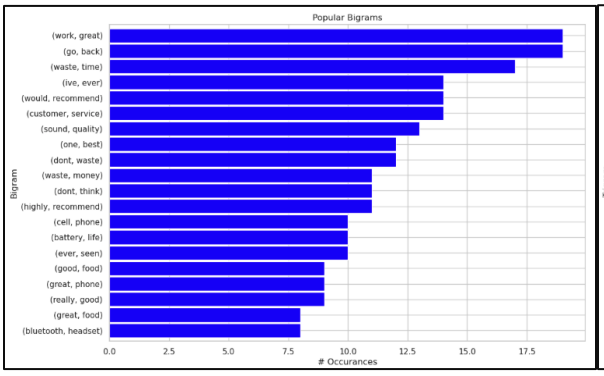
The second step is to prepare our reviews for neural network modeling and NLP techniques. Before we build the neural network model, we need to adjust the reviews and labels in the table. We used `regex` to

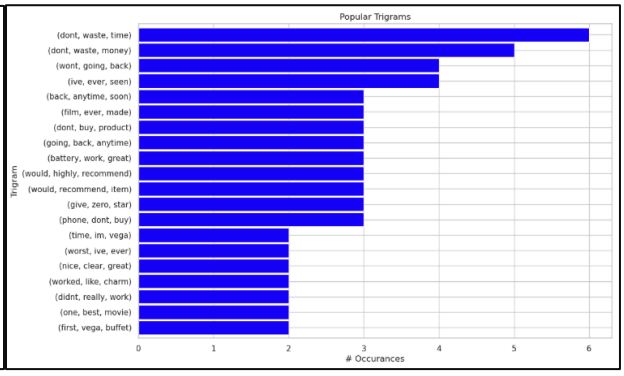
distinguish between English and non-English characters and remove unusual characters. This will separate the non-English characters from the English ones and remove them. After this, we separated the label column by '1' and '0' responses. Then, we need to get word counts and vocabulary size using `value_counts()`. Our vocabulary size in this combined data frame is 4,967 words. The next step was to see the most frequent words in the dataset and how many times they were used in sentiments (see first screenshot below). This led to looking into specific words that were always labeled with a positive sentiment (csat) and words always labeled with a negative sentiment (dsat) (see second screenshot below).

	all	csat	dsat
good	226	170	56
movie	208	101	107
great	205	188	17
film	182	96	86
phone	173	91	82
one	145	69	76
time	133	57	76
like	127	61	66
food	125	60	65
place	123	62	61

	all	csat	dsat
superb	7	7	0
priced	7	7	0
brilliant	7	7	0
joy	7	7	0
town	7	7	0
role	9	9	0
pleased	10	10	0
interesting	15	15	0
fantastic	18	18	0
delicious	23	23	0
unit	9	0	9
waited	10	0	10
wasted	10	0	10
mediocre	11	0	11
sucked	11	0	11
disappointment	11	0	11
unfortunately	11	0	11
disappointing	12	0	12
bland	13	0	13
poor	27	0	27

Next, we looked at what words were typically used together in the form of bigrams and trigrams. Below are the screenshots for popular bigrams and trigrams found within the dataset.





We then found the max sequence embedding of 8, a sequence length of 789, and the mean, minimum, and median sequence lengths of 7, 1, and 5, respectively (see screenshot below for code used).

```
max_sequence_embedding = int(round(np.sqrt(np.sqrt(vocab_size)), 0))
print('Max sequence embedding: ' + str(max_sequence_embedding))
```

✔ Last executed at 2025-01-19 07:31:22 in 19ms

```
Max sequence embedding: 8
```

```
list_of_lens = []
for index, row in df.iterrows():
    list_of_lens.append(len(clean_sentence(row['text'])))
sequence_length = max(list_of_lens)
print('Sequence length: ' + str(sequence_length))
```

✔ Last executed at 2025-01-19 07:31:31 in 1.39s

```
Sequence length: 789
```

```
mean_sequence_length = int(round(np.mean(list_of_lens), 0))
min_sequence_length = int(round(np.min(list_of_lens), 0))
median_sequence_length = int(round(np.median(list_of_lens), 0))

print('Mean sequence length: ' + str(mean_sequence_length))
print('Minimum sequence length: ' + str(min_sequence_length))
print('Median sequence length: ' + str(median_sequence_length))
```

✔ Last executed at 2025-01-19 07:31:33 in 15ms

```
Mean sequence length: 7
Minimum sequence length: 1
Median sequence length: 5
```

B. 2. Describe the goals of the tokenization process, including any code generated and packages used to normalize text during the tokenization process.

Here are the goals of the tokenization process used to normalize text before running and fitting our NLP model:

- Breaking text sentiment into "tokens" or short units.
- Standardize the text, ensuring no punctuation and that letters are lowercase, stemming, and lemmatization to keep the words consistent and reduce variability in each sentiment.
- Convert (map) the text into numeric terms. This is what is put into the NLP model, so we need to create a vocabulary of unique tokens from our dataset.
- Reduce model complexity by simplifying the sentiments and making it easier for the model to see the patterns.

After the text has been normalized, it is time to split our dataset into testing and training. For this project, we split the dataset into testing and training sentences and testing and training labels.

```
df['clean_text'] = df["text"].apply(clean_sentence)
```

✔ Last executed at 2025-01-19 07:31:37 in 1.10s

```
df.head()
```

✔ Last executed at 2025-01-19 07:31:39 in 21ms

	text	label	clean_text
0	so there is no way for me to plug it in here in the us unless i go by a converter.	0	[way, plug, u, unless, go, converter]
1	good case, excellent value.	1	[good, case, excellent, value]
2	great for the jawbone.	1	[great, jawbone]
3	tied to charger for conversations lasting more than 45 minutes.major problems!!	0	[tied, charger, conversation, lasting, 45, minutesmajor, problem]
4	the mic is great.	1	[mic, great]

```
def listToString(s):

    #initialize an empty string
    string = " "

    #return string
    return (string.join(s))
```

✔ Last executed at 2025-01-19 07:31:50 in 17ms

```
df['clean2_text'] = df['clean_text'].apply(listToString)
```

✔ Last executed at 2025-01-19 07:31:54 in 18ms

```
df.head()
```

✔ Last executed at 2025-01-19 07:31:57 in 20ms

	text	label	clean_text	clean2_text
0	so there is no way for me to plug it in here in the us unless i go by a converter.	0	[way, plug, u, unless, go, converter]	way plug u unless go converter
1	good case, excellent value.	1	[good, case, excellent, value]	good case excellent value
2	great for the jawbone.	1	[great, jawbone]	great jawbone
3	tied to charger for conversations lasting more than 45 minutes.major problems!!	0	[tied, charger, conversation, lasting, 45, minutesmajor, problem]	tied charger conversation lasting 45 minutesmajor problem
4	the mic is great.	1	[mic, great]	mic great

```
tokenizer = Tokenizer(oov_token="<OOV>")
```

✔ Last executed at 2025-01-19 07:32:00 in 11ms

B. 3. Explain the padding process used to standardize the length of sequences. Include the following in your explanation:

- **if the padding occurs before or after the text sequence**

We used post-padding to standardize the length of the sequences. If the max length of a sentiment is 7 words, then all the sentiments with less than 7 words will have zeros added to ensure they have 7 numbers in the sentiment, even if some of the numbers code for no words (like '0').

Code used:

```
vocab_size = 4967
embedding_dim = 8
max_length = 7
trunc_type = 'post'
oov_tok = '<OOV>'
padding_type = 'post'
```

✔ Last executed at 2025-01-19 07:32:48 in 17ms

```
tokenizer = Tokenizer(num_words=vocab_size, oov_token=oov_tok)
tokenizer.fit_on_texts(training_sentences)
word_index = tokenizer.word_index
print(word_index)
```

✔ Last executed at 2025-01-19 07:32:51 in 45ms

```
{'<OOV>': 1, 'movie': 2, 'film': 3, 'good': 4, 'phone': 5, 'great': 6, '0': 7, 'one': 8, '1': 9, 'like': 10, 'time': 11, 'work': 12, 'bad': 13, 'really': 14, 'well': 15, 'dont': 16, 'would': 17, 'place': 18, 'service': 19, 'best': 20, 'even': 21, 'ever': 22, 'quality': 23, 'character': 24, 'product': 25, 'also': 26, 'make': 27, 'food': 28, 'headset': 29, 'get': 30, 'love': 31, 'ive': 32, 'sound': 33, 'made': 34, 'use': 35, 'excellent': 36, 'battery': 37, 'could': 38, 'go': 39, 'better': 40, 'never': 41, 'thing': 42, 'recommend': 43, 'look': 44, 'im': 45, 'ear': 46, 'back': 47, 'acting': 48, 'see': 49, 'much': 50, 'way': 51, 'case': 52, 'first': 53, 'didnt': 54, 'think': 55, 'year': 56, 'nice': 57, 'price': 58, 'still': 59, 'scene': 60, 'worst': 61, 'story': 62, 'waste': 63, 'little': 64, '2': 65, 'every': 66, 'right': 67, 'people': 68, 'pretty': 69, 'everything': 70, 'got': 71, 'real': 72, 'problem': 73, 'say': 74, 'enough': 75, 'two': 76, 'doesnt': 77, 'know': 78, 'actor': 79, 'money': 80, 'piece': 81
```

```
sequences = tokenizer.texts_to_sequences(training_sentences)
padded = pad_sequences(sequences, maxlen=max_length, truncating=trunc_type)
testing_sentences = tokenizer.texts_to_sequences(testing_sentences)
testing_padded = pad_sequences(testing_sentences, maxlen=max_length)
```

✔ Last executed at 2025-01-19 07:32:55 in 44ms

- **a screenshot of a single padded sequence**

```
[  4  52  36 271   0   0   0]
```

B. 4. Identify how many sentiment categories will be used and an activation function for the final dense layer of the network.

There are 2 sentiment categories ('0' and '1'); therefore, binary_crossentropy will be used with sigmoid for activation.

```
model = tf.keras.Sequential([
    tf.keras.layers.Embedding(vocab_size, embedding_dim, input_length=max_length),
    tf.keras.layers.GlobalAveragePooling1D(),
    tf.keras.layers.Dense(100, activation='relu'),
    tf.keras.layers.Dense(50, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])
```

✔ Last executed at 2025-01-19 09:49:23 in 21ms

```
model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
```

✔ Last executed at 2025-01-19 09:50:58 in 26ms

B. 5. Explain the steps used to prepare the data for analysis, including the size of the training, validation, and test set split (based on the industry average).

- Load Dataset
- Clean and preprocess the text:
    - convert text to lowercase
    - remove punctuation, special characters, and extra spaces
    - remove stop words
    - tokenize text into short sequences.
- Split the dataset into training and testing; an 80-20 split was used for this project.
- Tokenizing and sequencing text using a tokenizer
- Converts words to numeric and creates a word_index
- Padding the sequences to ensure each sentence is the same length; we chose to post-pad the sequences.

B. 6. Provide a copy of the prepared data set.

See the files attached in the task submission.

Part III: Network Architecture

C. 1. Provide the output of the model summary of the function from TensorFlow.

```
model.summary()
```

✔ Last executed at 2025-01-19 07:33:38 in 28ms

Model: "sequential"

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 7, 8)	39,736
global_average_pooling1d (GlobalAveragePooling1D)	(None, 8)	0
dense (Dense)	(None, 100)	900
dense_1 (Dense)	(None, 50)	5,050
dense_2 (Dense)	(None, 1)	51

Total params: 45,737 (178.66 KB)
Trainable params: 45,737 (178.66 KB)
Non-trainable params: 0 (0.00 B)

C. 2. Discuss the number of layers, the type of layers, and the total number of parameters.

The model consists of 5 layers: an embedding layer, a global average pooling layer, a dense layer with 100 nodes, a dense layer with 50 nodes, and a dense layer with 1 node. The embedding layer maps sequences into dense vector representations at a fixed size. The global average pooling layer reduces the sequence dimension by averaging over the time steps. The dense layers with 100, 50, and 1 nodes are fully connected with the set number of nodes. The outputs for the dense layers will be at their set number of nodes (100, 50, and 1). Overall, the total number of parameters is 45737, the sum of all the trainable weights of the 5 different layers of the model.

C. 3. Justify the choice of hyperparameters, including the following elements:

- **activation functions**

We used the ReLU (rectified linear unit) in the hidden layers; it promotes non-linearity. We used sigmoid activation for the output layer, which is excellent for our binary model.

- **number of nodes per layer**

Using three dense layers with 100, 50, and 1 nodes helps capture patterns from the dataset, reduce complexity and dimensionality, and clarify whether the outputs are a single value of '1' or '0'.

- **loss function**

Binary Crossentropy was used to measure the difference between predicted probabilities and the actual labels.

- **optimizer**

We used the Adam optimizer, which has an adaptive learning rate to ensure faster convergence and efficiency with gradient updates.

- **stopping criteria**

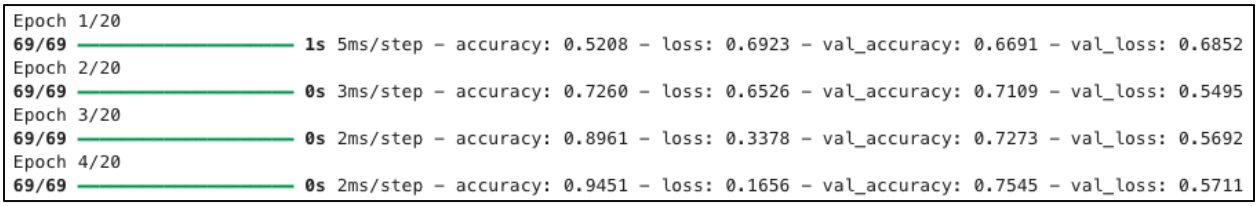
This will monitor validation loss and accuracy; the model stops training when its performance does not improve. This helps us from overfitting the model.

- **evaluation metric**

Accuracy is used as the evaluation metric; it measures the percentage of correct predictions the model makes.

Part IV: Model Evaluation

D. 1. Discuss the impact of using stopping criteria to define the number of epochs, including a screenshot showing the final training epoch.



The screenshot above shows the model's early stopping monitor stopped at 4 epochs. This is used to control the model's performance and to prevent overfitting the model. If the model starts to overfit, it will begin to memorize the training data rather than predict the unseen data. Overfitting would result in a high training accuracy but low validation performance. In addition, using an early stopping monitor will be more time efficient if using a larger dataset, essentially saving time and computational resources. If we had not used the early stopping monitor, the model would have continued through the 20 epochs it was set to and would have provided poor validation results.

D. 2. Assess the fitness of the model and any actions taken to address overfitting.

Training accuracy – 0.9451

Training loss – 0.1656

Validation accuracy – 0.7545

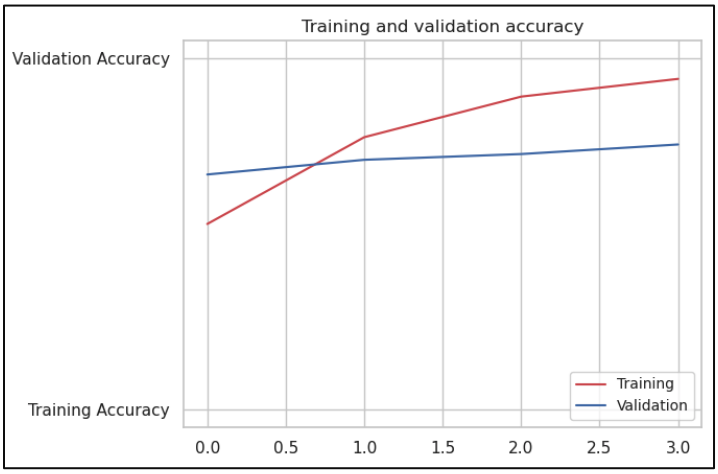
Validation loss – 0.5711

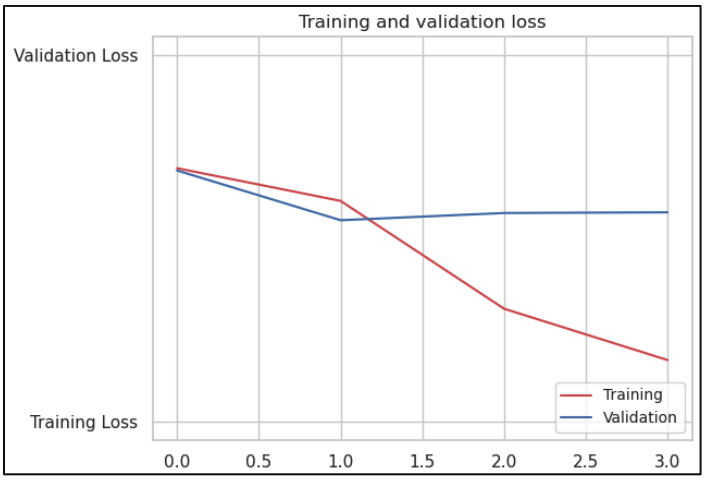
The model performed well on training data with a high prediction accuracy of 94.51%. This demonstrates that the model could effectively learn the patterns in the training dataset. A low training loss indicates that the model makes very few prediction errors. A validation accuracy of 75.45% means that the model

struggles to generalize well to new unseen data and is a sign of overfitting as the model fits well with the training data but fails to generalize on the validation (testing) data. A validation loss of 57.11% confirms that the model has overfitted the training data. This number should be close to the training loss value (16.56%) if overfitting did not occur.

One method was used to stop overfitting the model, early stopping with a patience of 20 epochs. The model stopped at 4 epochs, which suggests that even though the training accuracy was improving, the validation (testing) performance was not improving. Early stopping monitoring is a good strategy to prevent overfitting and stop training the model as soon as the evaluation metrics decline.

D. 3. Provide visualizations of the model's training process, including a line graph of the loss and chosen evaluation metric.





D. 4. Discuss the predictive accuracy of the trained network using the chosen evaluation metric from part D3.

Based on % training accuracy of 94.51%, the model correctly predicted above 90% of instances on the training dataset. Testing (validation) accuracy is at 75.45%, which means the model correctly predicted 75% of cases in the dataset. However, comparing validation and training accuracy, we see that both

numbers are pretty high. This may indicate that the model is overfitting and is memorizing the training data.

Part V: Summary and Recommendations

E. Provide the code you used to save the trained network within the neural network.

See the files attached in the task submission.

F. Discuss your neural network's functionality, including the impact of the network architecture.

The architecture of our model is stated above in part C3. Changing the number of layers may help the model validate (testing) data better.

As for the functionality of the network, it has some promising features, such as relatively high training accuracy. However, the validation accuracy is also high, which suggests that the model is overfitting and is memorizing the training data rather than predicting from unseen data sources. More data must be added to see if this changes the model's results.

G. Recommend a course of action based on your results.

There are a few actions we can take based on the results of the model. First, we can increase the amount of data provided. This dataset was only 10,000 rows; this is not a large dataset, which can lead the model to overfit. Another action could be simplifying the model by reducing the number of layers within the model. Simplifying the model will help it generalize to unseen data better, especially with a small dataset. Ultimately, adding more data to the model should help increase the prediction capabilities of testing data.

Part VI: Reporting

H. Show your neural network in an industry-relevant interactive development environment. Include a PDF or HTML document of your executed notebook presentation.

See the files attached in the task submission.

I. Sources

Aubergine Solutions. (2019, October 2). Scratching surface of RNN, GRU, and LSTM with example of sentiment analysis. Medium. Retrieved January 19, 2025, from https://medium.com/aubergine-solutions/scratching-surface-of-rnn-gru-and-lstm-with-example-of-sentiment-analysis-8dd4e748d426

Brownlee, J. (2018, August 17). How to configure the number of layers and nodes in a neural network. Machine Learning Mastery. Retrieved January 19, 2025, from https://machinelearningmastery.com/how-to-configure-the-number-of-layers-and-nodes-in-a-neural-network/

Brownlee, J. (2020, August 20). A Gentle Introduction to the Rectified Linear Unit (ReLU). Machine Learning Mastery. https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

Built In. (n.d.). ReLU activation function. Retrieved January 19, 2025, from https://builtin.com/machine-learning/relu-activation-function

GeeksforGeeks. (n.d.). Adam optimizer in TensorFlow. Retrieved January 19, 2025, from https://www.geeksforgeeks.org/adam-optimizer-in-tensorflow/

Kingma, D., and Ba, J. (2017, January 30). Adam: A method of stochastic optimization. Retrieved January 19th, 2025, from https://arxiv.org/abs/1412.6280

Sharma, A. (2019, July 30). A complete step-by-step tutorial on sentiment analysis in Keras and TensorFlow. Towards Data Science. Retrieved January 19, 2025, from https://towardsdatascience.com/a-complete-step-by-step-tutorial-on-sentiment-analysis-in-keras-and-tensorflow-ea420cc8913f

Stack Overflow. (n.d.). Detect strings with non-English characters in Python. Retrieved January 19, 2025, from https://stackoverflow.com/questions/27084617/detect-strings-with-non-english-characters-in-python

Team, K. (2020). Keras documentation: The Sequential model. K. Team. https://keras.io/guides/sequential_model/

UCI Machine Learning Repository. (n.d.). Sentiment labelled sentences. Retrieved January 19, 2025, from https://archive.ics.uci.edu/dataset/331/sentiment+labelled+sentences