Saemi Ramirez

011926418

D213 PA 2

Sentiment Analysis Using Neural Networks

10/25/2024

WGU

1. Purpose of this data analysis
   1. Research Question

Can we predict the customer’s sentiment based on their comments using the neural network model and NLP?

* 1. Define objectives or goals

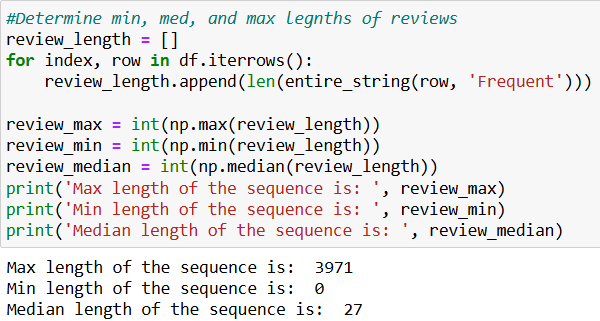
The goal is to build the neural network model and NLP and to analyze and predict if the comment was either positive or negative.

* 1. 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 LSTM (Long Short-Term Memory network) which is a type of a recurrent neural network is used for this analysis (Brownlee)

1. Summarize assumptions of a time series modeling including stationarity and autocorrelated data
   1. Perform exploratory data analysis on the chosen data set
      1. Presence of unusual characters – regex was used to identify any non-english characters in the review column  
         A close-up of a computer screen

         Description automatically generated
      2. Vocabulary size  
         The vocabulary size is 4284. First, the ‘Frequent’ column was created after cleaning up and filtering the ‘Review’ column. Tokenizer() and fit\_on\_texts functions were used then counted the length of the word with word\_index + 1  
         A screen shot of a computer code

         Description automatically generated
      3. Proposed word embedding length  
         Embedding length was calculated as 8 using the np.sqrt function with already calculated vocab\_size (4284)  
         A close-up of a computer code

         Description automatically generated
      4. Statistical justification for the chosen max sequence length  
         The max sequence length is 3971  
         
   2. Describe the goals of the tokenization process, including any code generated and packages that are used to normalize text during the tokenization process
      1. Nltk.word\_tokenizer() was used to break down customer’s reviews into single words  
         A computer screen shot of a program

         Description automatically generated

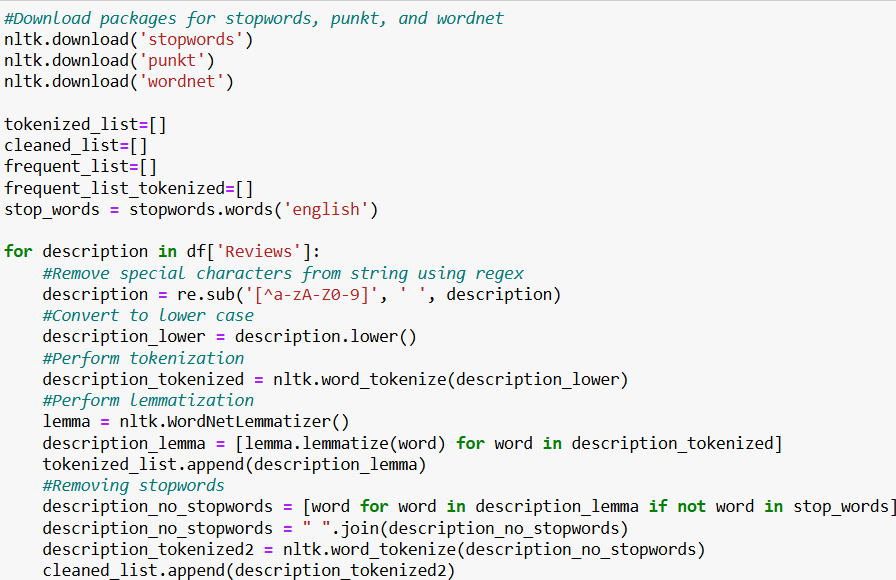
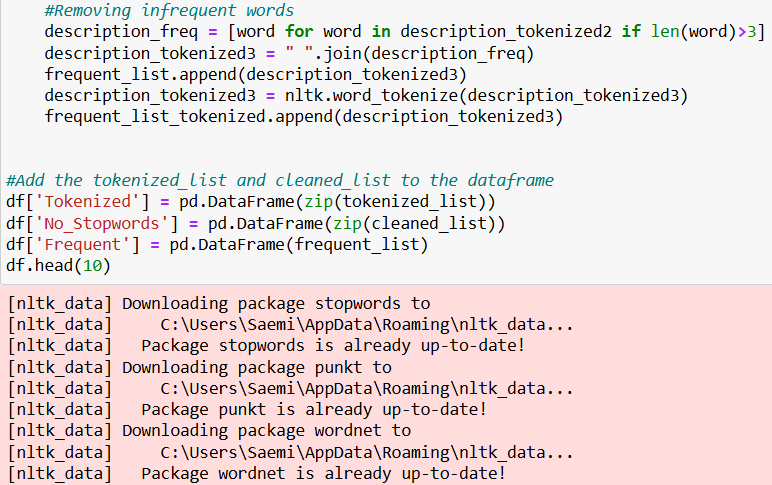
A screen shot of a computer code

Description automatically generated  
A screenshot of a computer

Description automatically generated

* 1. Explain the padding process
     1. If the padding occurs before or after the text sequence  
        The pad\_sequences was applied to the both X\_train and X\_test sets after the texts\_to\_sequences
     2. Screenshot of a single padded sequence  
        A screenshot of a computer code

        Description automatically generated
  2. Identify how many categories of sentiment will be used and an activation function for the final dense layer of the network  
     Two categories of sentiment which are positive or negative (0 or 1) will be used with ‘softmax’ as an activation function for the final dense layer of the network
  3. Explain the steps used to prepare the data for analysis including the size of the training, validation, and test set split
     1. Imported the data (Amazon, IMDB, and Yelp), added a third column (Source) and concatenated 3 data sources  
        A screen shot of a computer code

        Description automatically generated
     2. Used the regex, lower(), WordNetLemmatizer(), stopwords, remove infrequent words in one for loop  
          
          
        

A screenshot of a computer

Description automatically generated

* + 1. Saved the DataFrame to csv file  
       
    2. Created a function to make an one long string out of the ‘Frequent’ column  
       A close-up of a computer code

       Description automatically generated
    3. Calculated max, min, and median lengths of the result of entire\_string function with ‘Frequent’ column  
       A screenshot of a computer code

       Description automatically generated
    4. Split the data to train and test sets  
       A screen shot of a computer code

       Description automatically generated
    5. Padded on both of the train and the test sets using pad\_sequences after applying texts\_to\_sequences() function  
       A screenshot of a computer program

       Description automatically generated
    6. Converted the padded data to NumPy array to be used in the model  
       A close-up of text

       Description automatically generated
    7. Exported the NumPy arrayed sets to csv files  
       A computer code with text

       Description automatically generated with medium confidence
  1. Provide a copy of the prepared dataset  
     ‘cleaned\_data.csv’, ‘test\_label.csv’, ‘test\_padded.csv’, ‘training\_label.csv’, and ‘training\_padded.csv’ are submitted

1. Describe the type of network used
   1. Provide the output of the model summary of the function from TensorFlow  
      A screen shot of a computer code

      Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

* 1. Discuss the number of layers, the type of layers, and the total number of parameters  
     Total 3 different types of layers were used which are Embedding, GlobalAveragePooling1D, and the Dense.   
     Embedding layer is a method used to represent words as numerical vectors which means it transforms text into numbers. (Medewar)   
     GlobalAveragePooling1D outputs a matrix of batch x embedding\_size by averaging across the sequence dimension. “1D” refers to the fact that the averaging occurs over a single dimension. This process can handle varying sequence lengths. (Lathan)   
     Dense layers are essential components of neural networks, composed of neurons that are each connected to every neuron in the preceding layer. The term “dense” indicates that each neuron has a connection to all neurons in the previous layer. (Keras)  
     Out of numbers of parameters that the Embedding function has, 3 parameters, the input\_dim for the vocabulary size, the output\_dim for the embedding size, and the input\_length for the review\_max were used for this analysis.   
       
     A screen shot of a computer code

     Description automatically generated
  2. Justify the choice of hyperparameters
     1. Activation functions – ‘softmax’ was used for the activation parameter for the dense function to “convert(s) a vector of values to a probability distribution” (Layer)
     2. Number of nodes per layer  
        A screenshot of a computer

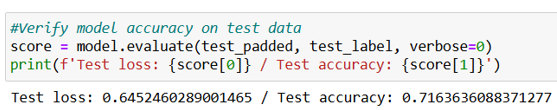
        Description automatically generated
     3. Loss function – ‘sparse\_categorical\_crossentropy’, which “computes the crossentropy loss between the labels and predictions” (tf.keras.losses.SparseCategorical.Crossentropy), was used for the loss in the compile function for the model
     4. Optimizer – ‘adam’ was used for the optimizer in the compile function for the model. Optimizer ‘adam’ is a stochastic gradient descent technique that relies on adaptive estimation of both first-order and second-order moments (Adam)
     5. Stopping criteria – EarlyStopping function with patience set to 2 was utilized for the callbacks parameter in the model.fit function. EarlyStopping callback allows monitoring a metric and halts training if no improvement is detected. (Early) Epochs was set to 20 as a starting point
     6. Evaluation metric – ‘Accuracy’ was used for the metric in the compile function for the model. Accuracy “calculates how often predictions equal labels” (Accuracy).

1. Evaluate the model training process and its relevant outcomes
   1. Discuss the impact of using stopping criteria to include defining the number of epochs including a screenshot showing the final training epoch  
      Stopping criteria stops the model training when there is no more to improve on the model. This helps preventing overfitting by halting the training process early, before the model becomes overly complex and starts memorizing the training data (Miseta) As the screenshot shown below, the model stopped training after the 9th iteration with the accuracy of 92.66%  
        
        
      A screenshot of a computer

      Description automatically generated
   2. Assess the fitness of the model and any actions taken to address overfitting  
      As the screenshot above, the model stopped training further after 9th iteration to prevent overfitting. Also EarlyStopping function was added on the callbacks parameter to avoid overfitting on the model.  
      The training model has the accuracy of approximately 88.54% on this analysis.   
      A screenshot of a computer code

      Description automatically generated
   3. Provide visualizations of the model’s training process including a line graph of the loss and chosen evaluation metric  
      A graph with lines and text

      Description automatically generated  
      A graph of a graph with blue and orange lines

      Description automatically generated
   4. Discuss the predictive accuracy of the trained network using the chosen evaluation metric  
      The accuracy on the test model is approximately 71.64%  
      
2. Provide the code  
   ‘Saemi Ramirez D213 PA2 – Sentiment Analysis Using Neural Networks.ipynb’ and ‘my\_model.keras’ are submitted  
   A screenshot of a computer code

   Description automatically generated
3. Discuss the functionality of your neural network including the impact of the network architecture  
   The neural network model was developed to classify sentiments either positive or negative. Model evaluation showed an accuracy of 88.5% on the training set and 71.6% on the test set. Overfitting was effectively mitigated by implementing an EarlyStopping function and limited training to the designated number of epochs, thus reducing the risk of memorization of the training data.  
   The model architecture comprises five layers, including one Embedding layer, one GlobalAveragePooling1D, and three Dense layers. The first Dense layer contains 100 neurons with 900 parameters using relu activation. The second Dense layer consists of 50 neurons with 5050 parameters and employes relu activation. The final Dense layer has 2 neurons with 102 parameters using softmax activation.
4. Recommend a course of action  
   Bigger size of samples is recommended for the better prediction rate for the test set.  
   I also recommend this model can be used for the gaming reviews such as Steam by analyzing the user’s reviews. It can be helpful for the players who play the game, but it will be more beneficial for the developers to analyze their weaknesses and apply it on their next game.
5. Show your neural network in an industry-relevant interactive development environment  
   ‘Saemi Ramirez (011926418) D213 PA 2 Sentiment Analysis Using Neural Networks.pdf’ is submitted
6. List 3rd party code
   1. DS Archives. *Part – 10 How to use FreqDist in NLTK with python (Natural Language Toolkit Tutorial)*. YouTube. (September 26, 2020). <https://www.youtube.com/watch?v=1K2ZQ5MYdnM>.
   2. *Implementing and Analyzing N-Grams in Python*. Open Library. <https://ecampusontario.pressbooks.pub/nudh3/chapter/implementing-and-analyzing-n-grams-in-python>.
   3. Mark. *How to convert list of lists into list – python.* Stack overflow. (December 1, 2019). <https://stackoverflow.com/questions/59130959/how-to-convert-list-of-lists-into-list-python>.
   4. Nithyashree. *What Are N-Grams and How to Implement Them in Python?* Analytics Vidhya. (September 17, 2024). <https://www.analyticsvidhya.com/blog/2021/09/what-are-n-grams-and-how-to-implement-them-in-python>.
   5. *Seaborn.*countplot. Seaborn. <https://seaborn.pydata.org/generated/seaborn.countplot.html>.
   6. Tim\_xyz. *How to install stop-words package for Anaconda*. StackOverflow. (January 22, 2018). <https://stackoverflow.com/questions/48385829/how-to-install-stop-words-package-for-anaconda>.
7. List 3rd party text citations or references
   1. *Accuracy metrics*. Keras. <https://keras.io/api/metrics/accuracy_metrics>.
   2. *Adam*. Keras. <https://keras.io/api/optimizers/adam>.
   3. Brownlee, Jason. *Time Series Prediction with LSTM Recurrent Neural Networks in Python with Keras*. Machine Learning Mastery. (August 7, 2022). <https://machinelearningmastery.com/time-series-prediction-lstm-recurrent-neural-networks-python-keras/>.
   4. *Early Stopping*. Lightning AI. <https://lightning.ai/docs/pytorch/stable/common/early_stopping.html>.
   5. *Keras dense layer*. Educative. (July 24, 2023). <https://www.educative.io/answers/keras-dense-layer>.
   6. Lathan. *What does GlobalAveragePooling1D do in keras?* Stack Overflow. (July 29, 2023). <https://stackoverflow.com/questions/75067335/what-does-globalaveragepooling1d-do-in-keras>.
   7. *Layer activation functions*. Keras. <https://keras.io/api/layers/activations>.
   8. Medewar, Soham. *Embedding Layers in Keras*. Code360. (March 27, 2024). <https://www.naukri.com/code360/library/embedding-layers-in-keras>.
   9. Miseta, Tamas. *Surpassing early stopping: A novel correlation-based stopping criterion for neural networks*. ScienceDirect. (January 28, 2024). <https://www.sciencedirect.com/science/article/pii/S0925231223011517>.
   10. *Tf.keras.losses.SparseCategoricalCrossentropy*. TensorFlow. <https://www.tensorflow.org/api_docs/python/tf/keras/losses/SparseCategoricalCrossentropy>.
8. Demonstrate professional communication in the content and presentation of your submission  
   Panopto Link: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=d7438a04-61c3-4125-b2fe-b214013ba01b>