Sentimental Analysis Using Neural Networks – D213

Gooden, Nina S. Gooden [ID #: 009823504]

Dr. William Sewell

Sentimental Analysis Using Neural Networks – D213

This analysis will explore the medical data of a theoretical real-world organization. By utilizing the Natural Language Toolkit (NLTK) I’ve learned throughout the duration of this course, I will analyze the positive and negative sentiment from a joined dataset from Amazon, IMDB, and Yelp. I will provide visualizations to support my assessments, in tandem with code for my models. As previously requested, I will also discuss the limitations and potential course of action this data supports.

## **Part I: The Research Question**

1. Research Question
   1. Can unstructured review data be accurately evaluated for sentiment?
2. Data Analysis Goals and Objective
   1. The goal for this analysis is to adequately evaluate positive and negative sentiment from an unstructured dataset.
3. Capable Network
   1. To accomplish the task, I will be utilizing the Keras framework. With Keras, I will be building a neural network (ANN) that will preprocess and load data, define a model, allow us to define a loss function and optimizer, and fit the model. (Tanwar, 2021).  
      In particular, I will be using the sequential model which allows for several CNN implementation layers, including but not limited to the following *(Building a Keras Sequential Model | Keras for Image Classification, n.d.)*:
      1. The convolution layer, which attempts to extract high-level features by replacing pixels with a filter
      2. The activation function, which control the thresholds for neurons
      3. The maxpooling layer, manipulates matrix maximums
      4. The dropout layer, that randomly ignores neurons to prevent overfitting
      5. The dense layer, a classifier
      6. The flattening “layer,” which is more of an operation or task than a set later, but creates a single vector for use by the softmax classifier
      7. Softmax classifiers, which give probabilities for each class label

## **Part II: Justification for Technique**

1. Exploratory Data Analysis
   1. In order to pass a clean, insightful data to the model, I had to prepare in a way that would result in actionable information. I began by importing quite a few libraries. After trying to use the .glob function, I ended up manually importing each of the datasets.

A screenshot of a computer

Description automatically generated

* 1. I ran a few checks to ensure the quality and quantity of the data.

A screenshot of a computer

Description automatically generated

* 1. The documentation for the datasets specified that there were no missing values, so I was comfortable moving to the next step of my evaluation. I used the .concat() function to combine them.

A screenshot of a computer

Description automatically generated

* 1. I noted that the index was off, so I also reset the index.

Graphical user interface, text

Description automatically generated

* 1. I checked the integrity of the combined data once more, since there was a discrepancy on how many values were available. I decided the information wasn’t too far off.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. Before I could work with the actual contents of the DataFrame, I needed to pull everything into single word snippets. To begin this process, I dropped case throughout.

Graphical user interface, text, application

Description automatically generated

* 1. Next, I tokenized to remove all special characters and break up sentences on white space. I did note that this method left contractions broken up, but since those are in the stopwords list, I wasn’t concerned.

Graphical user interface, text, application

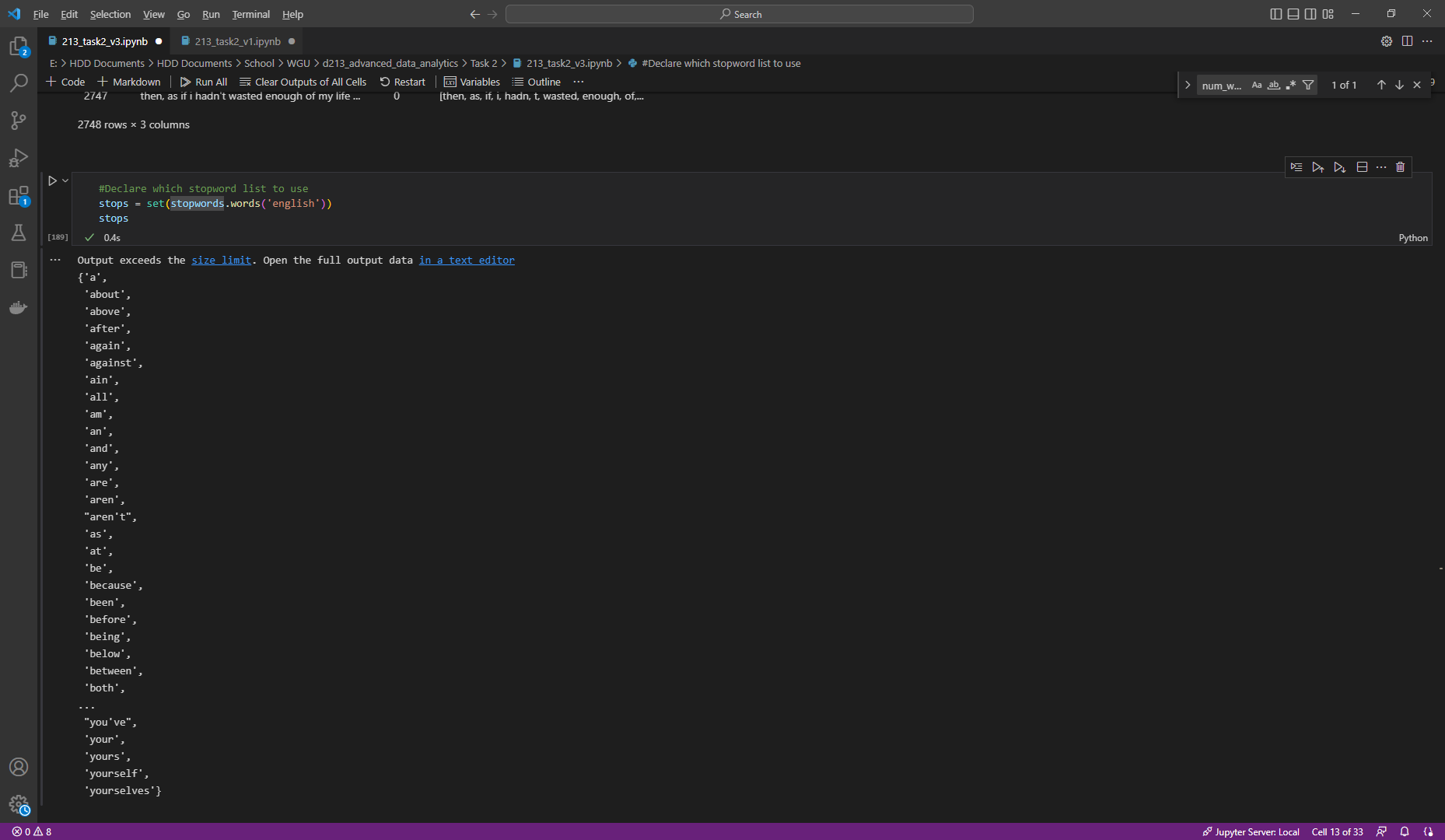
Description automatically generated

* 1. I evaluated the results.

Graphical user interface, text, application

Description automatically generated

* 1. Next, I identified which stopwords list I wanted to use.



* 1. And implemented that list.

Graphical user interface, text

Description automatically generated

* 1. I removed identified the most frequent words and put them in a list. Code credit (Library Carpentry, 2022)

Text

Description automatically generated

* 1. Then I removed words that appeared only once.

Text

Description automatically generated

* 1. I used lemmatization to group forms of similar words into a single item.

Text

Description automatically generated

* 1. I did a quick sanity check on all of the columns I was creating.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. Then I created a visualization for the top 25 words.

A screenshot of a computer

Description automatically generated with medium confidence

* 1. And finally, I created a wordcloud for those words.

Graphical user interface, text

Description automatically generated

* 1. I exported the dataset.

A screenshot of a computer

Description automatically generated

1. Tokenization process goals
   1. The goal for the tokenization process was to take the large quantities of text that were created by stripping out the sentence structure, to create clean and measurable “tokens,” and then converting those tokens into numeric arrays.
   2. By doing so, I am better able to segment the data into sentimental classifications.
   3. To begin, I created the training and test variables—again keeping the 4:1 ratio.

Graphical user interface, text

Description automatically generated

* 1. I then used the Tokenizer function. I selected 5,000 for the amount argument because I knew there were at least 4,880 words in the entire DataFrame. I also used the texts\_to\_sequences function.

Graphical user interface, text

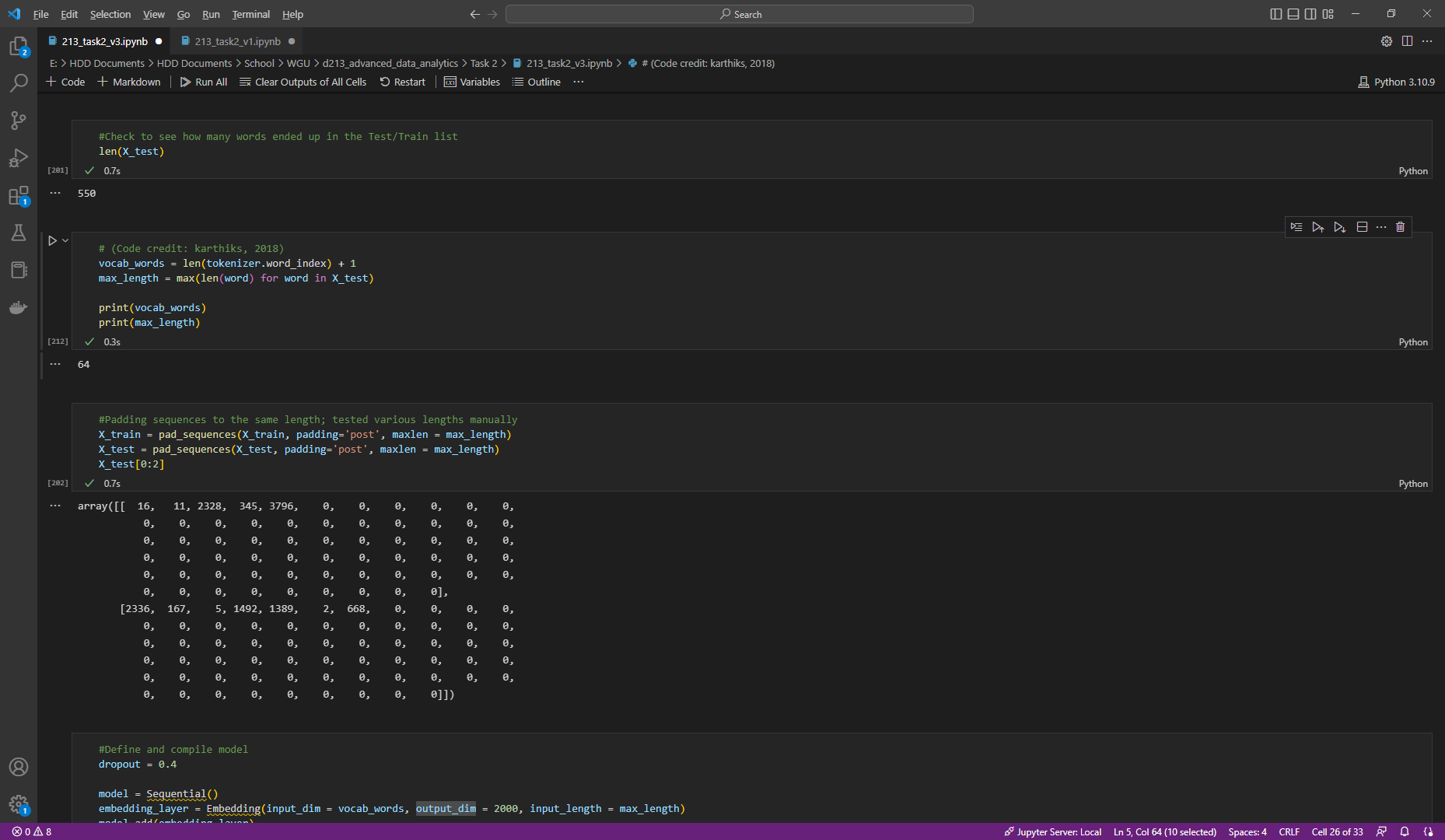
Description automatically generated

* 1. I checked the length of each of the X\_variables.

Graphical user interface, text

Description automatically generated

1. Padding process
   1. I did the padding process before the sequence, but after the second tokenization. I did this because I needed to know the best max\_length and vocabulary size after tokenization. (Paul, 2022). They were 4880 and 64, respectively. I also used the vocabulary size as my proposed word embedding length.



* 1. I created the padded sequences and printed a screenshot for this assignment.

A screenshot of a computer

Description automatically generated

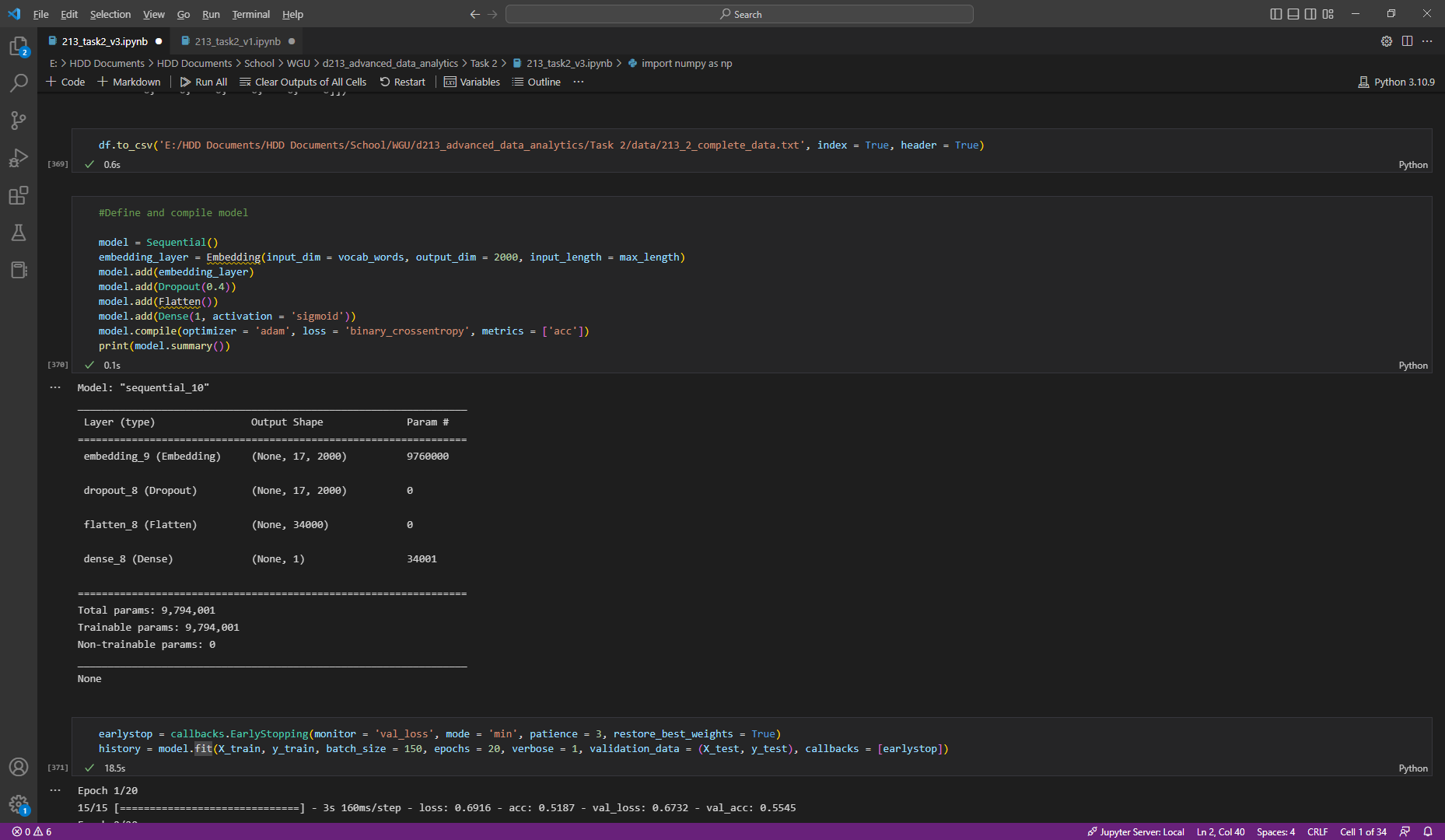
1. Categories of sentiment
   1. Per the documentation for the data sets, there are two sentiment categories. Negative and positive categories are noted as 0 and 1, respectively.
2. Steps
   1. Steps are covered by previous question #s and screenshots have been provided. They are as follows:
      1. Import data
      2. Drop case for data
      3. Remove punctuation and implement 1st tokenization (create individual words)
      4. Apply stopwords to remove common words from data
      5. Remove infrequent words from data
      6. Lemmatize
      7. Visualize top words
      8. Create training and test
      9. Second tokenization (convert words to numerical)
      10. Pad sequences
3. Copy of prepared dataset

Text

Description automatically generated

## **Part III: Network Architecture**

1. Model summary from TensorFlow
   1. Now that I have cleaned and processed my data, I was prepared to create my model.

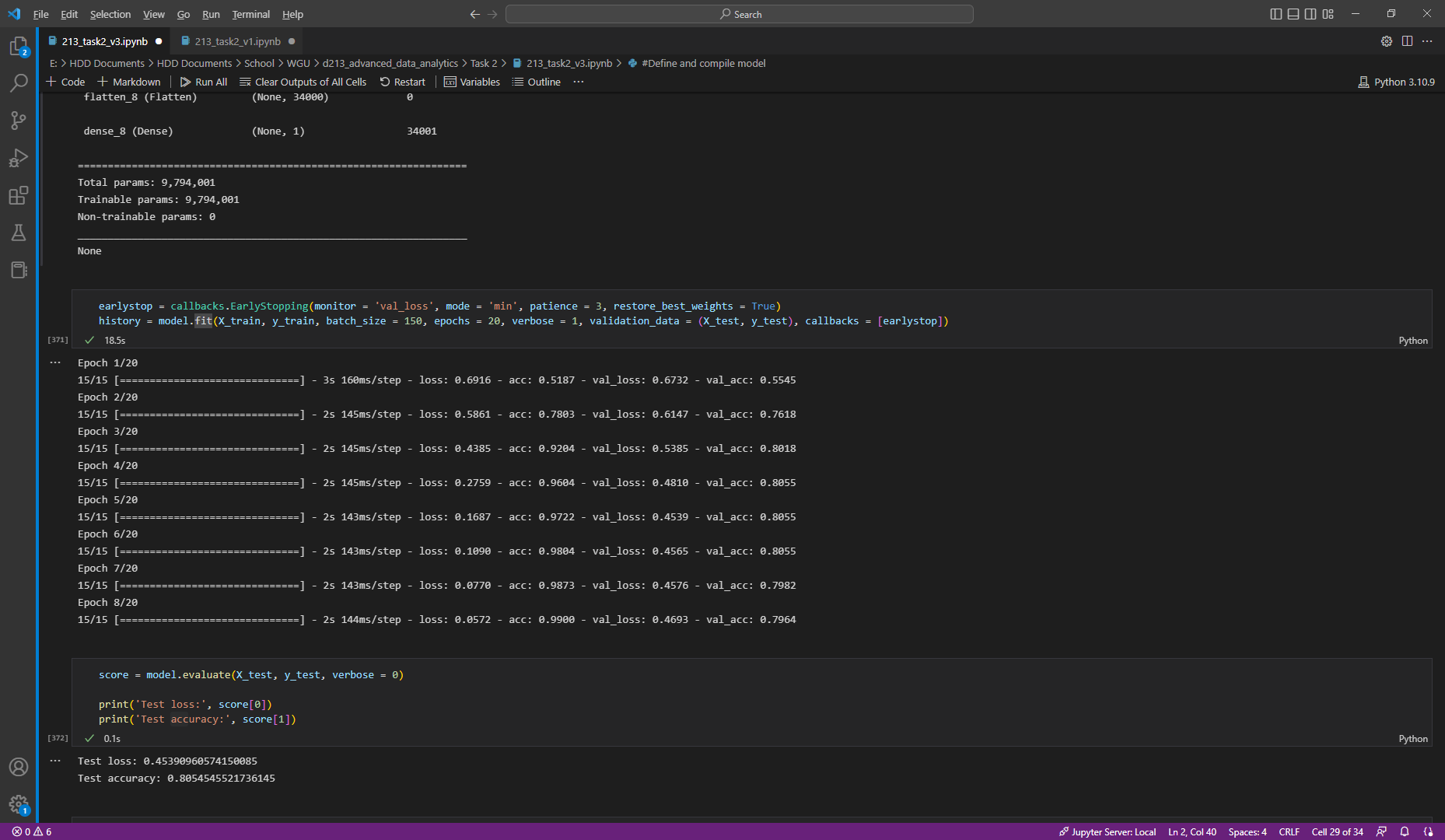


* 1. I began with defining and compiling the four layers of my model:
     1. Embedding: used to convert words to a fixed length; output shape (None, 17, 2000); 9,760,000 parameters
     2. Dropout: used to reduce the chance of overfitting; output shape (None, 17, 2000); 0 parameters
     3. Flatten: used to reduce dimension and shape of input layer; output shape (None, 34,000); 0 parameters
     4. Dense: used to combine all neurons and shape the final output; output shape (None, 1); 34,001

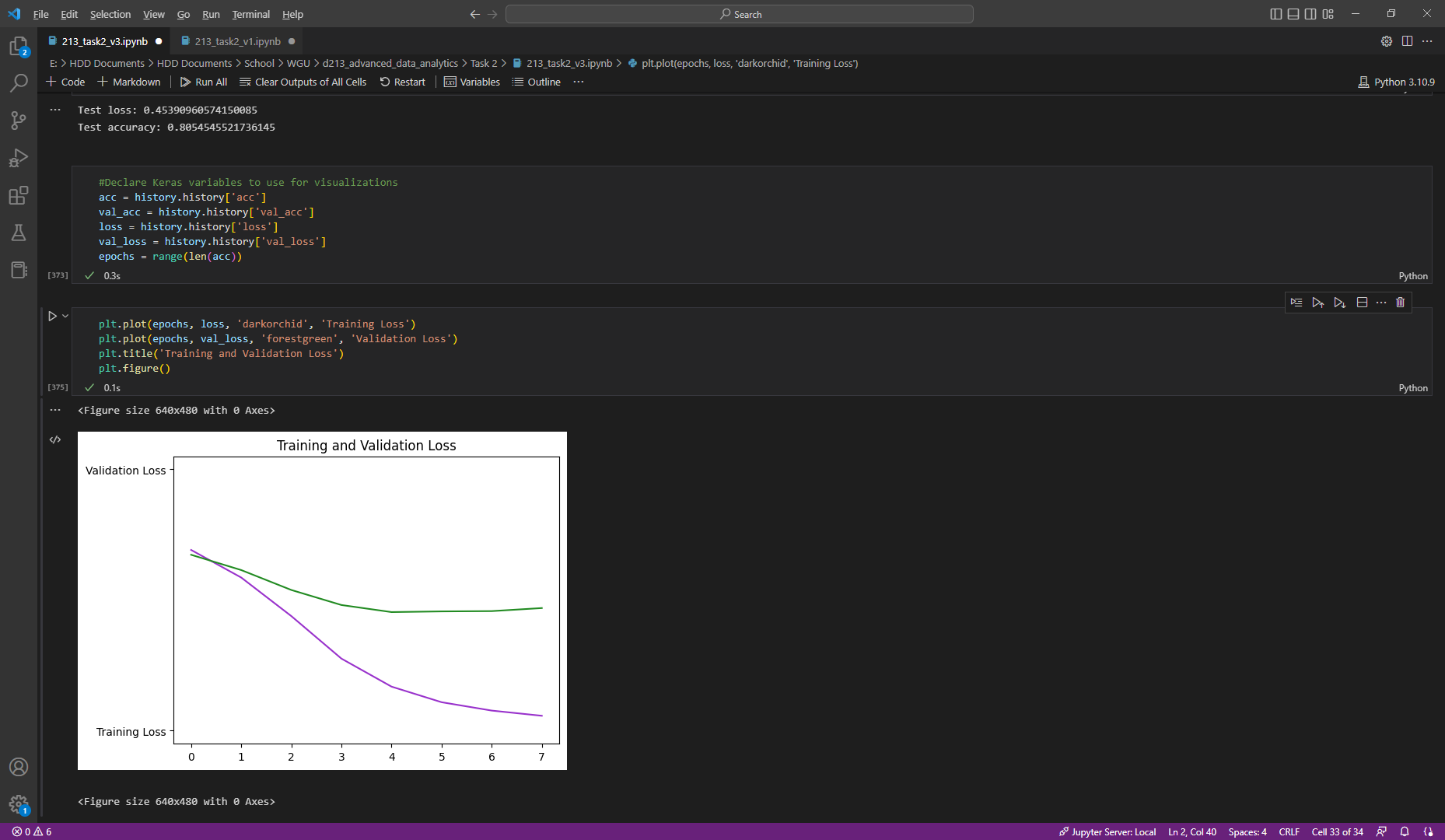
1. Justify hyperparameters
   1. Activation function: I used the sigmod activation function in the Dense layer because it is the function I’ve used professionally
   2. number of nodes per layer: answered in #1, but it is worth noting that due to the small size of the dataset, the number of nodes per layer would likely be less than the average model at < 10 for all layers.
   3. loss function: The loss function for this model is “binary\_crossentropy.” As far as I could tell throughout the course there is no “better” loss function, so I used the one that is used in the Adam documentation, which applies to probability loss.
   4. optimizer: I used the “Adam” optimizer because the Adam “is recommended as the default optimizer for most of the applications.” (Gupta, 2022)
   5. stopping criteria: I used the earlystopping function to help prevent under- and overfitting.
   6. evaluation metric: I used the [‘acc’] (accuracy) option to compare similarity between the data in the test and training set.

## **Part IV: Model Evaluation**

1. Using stopping criteria vs. epochs
   1. I opted to use a mixture of both options for my model. I set the epochs to a certain number, but depended on earlystopping to halt the model if necessary. It did so—stopping at epoch 8.



1. Model training visualization
   1. Line graph of loss



* 1. Chosen evaluation metric: I picked accuracy as my evaluation metric. This is noted in question #4.

Graphical user interface

Description automatically generated

1. Assess model fitness
   1. Measures taken to address overfitting: Implementing the earlystopping function, as well as ensuring that mutable variables were established based on dataset values.
2. Predictive accuracy of trained network
   1. I ran an evaluation of the model’s accuracy and found it to be a little less than 81%. This is within the acceptable range.

Graphical user interface

Description automatically generated

## **Part V: Summary and Reccomendations**

1. Code to save trained network: I used the default save.

Graphical user interface, text, website

Description automatically generated

1. Functionality of network
   1. My model returned an 81% accuracy, which is within the recommended range of “acceptability” to be classified as “Good.” (Allwright, 2022). As such, I’m confident that the network functions well enough to use in further evaluation.
2. Recommendation
   1. Though the model can be used in decision making as it is now, there is an opportunity to increase accuracy. I recommend that the analysis be rerun by additional teams before it is implemented into any business policy.
   2. In addition, the data set is very small. Additional nuance could be evaluated if there were more records running at the same time. The three companies in question should provide more up-to-date information for review.

## **Part VI: Reporting**

#### H. Interactive Development Environment

HTML file of my Jupyter Notebook attached.

#### I. Web Sources

DecisionForest. (2020, March 26). How To Remove StopWords, Punctuation, Emojis and HTML from Strings with Regex ❌NLP Tutorial Python [Video]. YouTube. https://www.youtube.com/watch?v=b9G78PxZtX8

GeeksforGeeks. (2021, May 9). How to Merge multiple CSV Files into a single Pandas dataframe ? https://www.geeksforgeeks.org/how-to-merge-multiple-csv-files-into-a-single-pandas-dataframe/

DataDaft. (2020, October 6). Python Regex: How To Remove Punctuation [Video]. YouTube. https://www.youtube.com/watch?v=hxU3HLX1W1k

*Counting Tokens in Text – Library Carpentry: Text & Data Mining*. (n.d.). Library Carpentry - Library Carpentry: Text & Data Mining. http://librarycarpentry.org/lc-tdm/08-counting-tokens/index.html

*Visualising Frequency Distributions – Library Carpentry: Text & Data Mining.* (n.d.). Library Carpentry - Library Carpentry: Text & Data Mining. http://librarycarpentry.org/lc-tdm/09-frequency-distributions/index.html

Menzli, A. (2022, November 14). *Tokenization in NLP: Types, Challenges, Examples, Tools*. neptune.ai. https://neptune.ai/blog/tokenization-in-nlp

Team, K. (n.d.). Keras documentation: The Sequential model. https://keras.io/guides/sequential\_model/

Saxena, S. (2021, December 16). Understanding Embedding Layer in Keras - Analytics Vidhya. Medium. https://medium.com/analytics-vidhya/understanding-embedding-layer-in-keras-bbe3ff1327ce

Real Python. (2022, September 1). Sentiment Analysis: First Steps With Python’s NLTK Library. https://realpython.com/python-nltk-sentiment-analysis/

NLTK :: Sample usage for sentiment. (n.d.). https://www.nltk.org/howto/sentiment.html

Wisdomml, W. (2022, August 10). Tokenization in Natural Language Processing. Wisdom ML. https://wisdomml.in/tokenization-in-natural-language-processing/

#### J. Acknowledge sources

Paul, S. (2022, December 6). *Variable-Length Sequences in TensorFlow Part 1: Optimizing Sequence Padding*. Carted Blog. https://www.carted.com/blog/variable-length-sequences-in-tensorflow-part-1/

Tanwar, S. (2022, January 7). Building our first neural network in keras - Towards Data Science. Medium. <https://towardsdatascience.com/building-our-first-neural-network-in-keras-bdc8abbc17f5>

Building a Keras Sequential Model | Keras for Image Classification. (n.d.). <https://makeschool.org/mediabook/oa/tutorials/keras-for-image-classification-pfw/building-a-keras-sequential-model/>

Karthiks. How to find “num\_words” or vocabulary size of Keras tokenizer when one is not assigned? (2018, November 28). Stack Overflow. <https://stackoverflow.com/questions/53525994/how-to-find-num-words-or-vocabulary-size-of-keras-tokenizer-when-one-is-not-as>

Gupta, A. (2022, December 2). A Comprehensive Guide on Deep Learning Optimizers. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/>

Allwright, S. (2022, August 10). What is a good accuracy score? Simply explained. Stephen Allwright. https://stephenallwright.com/good-accuracy-score/