CI642 Report

Student Name: Eloise Derham

January 2021

**Accessing the code:**

Follow this link to the Google Colab folder in order to access the relevant code.

<https://colab.research.google.com/drive/1STehiJYZ1kTjj2GjztqceUNipCkD3kun?usp=sharing>

once the code as been accessed download the tweets.csv dataset file into the runtime environment.

Contents

[1. Introduction 4](#_Toc94261932)

[2. Task definition 4](#_Toc94261933)

[3. About my data 4](#_Toc94261934)

[4. Data cleaning process 4](#_Toc94261935)

[1. Convert strings to lower case 5](#_Toc94261936)

[2. Remove special characters apart from apostrophes 5](#_Toc94261937)

[3. Remove stop words 5](#_Toc94261938)

[4. Lemmatize and stem 6](#_Toc94261939)

[5. Use count-vectorizer to prepare data to be used as input into my model 6](#_Toc94261940)

[5. Model development 6](#_Toc94261941)

[Logistic regression model 6](#_Toc94261942)

[LSTM model 8](#_Toc94261943)

[6. Comparisons/ evaluation 10](#_Toc94261944)

[References 11](#_Toc94261945)

# Introduction

In this report I will outline the steps I took to create a simple NLP model. I will discus the collection and cleaning of the data, development of the model and compare my model to another classification model using multiple evaluation metrics.

# Task definition

I decided to investigate Natural Language Processing or NLP. I wanted to gather and clean my own data to feed into a machine learning model, with the goal of creating a model which could differentiate between 2 different genres of tweet.

NLP is used in many applications now from sentiment analysis to digital assistants and translators. “NLP makes it possible for computers to read text, hear speech, interpret it, measure sentiment and determine which parts are important” *(Sas.com, 2022)*.

The process I have gone through is similar to that used for sentiment analysis. This is often used to sift through thousands of reviews on websites and determine whether these reviews are positive or negative using machine learning. A Similar process called “Bayesian spam filtering” can be used to separate spam emails from regular ones by statistically identifying spam.

Sentiment analysis is often used with twitter to gather the publics opinion on certain topics, deciding which one the public is more favorable towards. With my project in particular it could be used to find and discover tweets of certain genres within a large dataset of tweets or to sort tweets into their respective genres.

# About my data

I chose to collect tweets about baking and sport, initially I collected 50 tweets but later increased this to 100. I stored my data in an excel table saved in csv format. My table consists of the tweet, the label (1 for sport and 0 for baking), and the number id of each tweet.

The reason I increased my data set to 100 entries is because I was getting 1.0 for all my evaluation metrics, this is common with a small dataset therefore for a more realistic representation of my model’s performance I increased the dataset.

By having a balanced dataset, I can be sure that if I received 90% accuracy it would suggest “a 90% accuracy on both the positively and negatively classed groups” *(stewart, 2020)* however many data sets are imbalanced and therefore further work must be done to correct imbalances before using the data on your model.

# Data cleaning process

Once I had collected my data, I completed several steps to clean it. I did this with the help of 2 sources the, IPython notebook *(juliandariomirandacalle, 2020)* and A python script to pre-process text (*262588213843476 (2019))*. I first experimented with different NLP cleaning processes and then chose the ones I believed to be the most helpful and important to my task. Below I describe the importance of each process and in my course work these methods can be seen in practice.

Overall cleaning data for an NLP model is important as it help to transform text data to the desired numeric state, by simplifying to end up with the most important words within the set.

## Convert strings to lower case

The first step is to convert all string to lower case, this ensures that all words can be identified. If the same word had one instance

Graphical user interface, text, application

Description automatically generated

Figure 1

## Remove special characters apart from apostrophes

When removing special characters, I first used \_ but I realised this also removes apostrophes, this makes the removal of stop-words not as efficient due to words such as “I’m” not being recognised without the apostrophe. For this reason, I researched a way to remove all punctuation except apostrophes so that the data would then work with stop word removal. This can be seen in the following figure.

Graphical user interface, text, application, email

Description automatically generated

Figure 2

## Remove stop words

Stop words are a list of words that are most common to the language being used. These are removed as they add no meaning to the data. NLTK offers a stop-word list for many languages.

Graphical user interface, text, application

Description automatically generated

Figure 3

Stop words can be removed as part of the cleaning process or they can be removed in the vectorization stage of your data. For the count-vectorization to work I had to remove them within the parameters of count-vectorization method.

Figure 4 shows the code used to remove stop words outside of vectorization.

Graphical user interface, text, application

Description automatically generated

Figure 4

## Lemmatize and stem

This process finds and extract the root words from the words in a sentence. Lemmatization typically returns more realistic words than stemming, but both often suffer from returning fake words such as “caring” to “care”.

For this reason, I decided not to use this pre-processing stem within my project as I didn’t believe it was necessary and could produce worse results.

The imports shown in figure 5 can be used if you decide to use lemmatization and stemming.

Graphical user interface, text, application, email

Description automatically generated

Figure 5

## Use count-vectorizer to prepare data to be used as input into my model

Models can’t take raw text as input, for this reason a vectorizer must be used to convert data to the desired format.

Graphical user interface, text, application

Description automatically generated

Figure 6

As input CountVectorizer takes, your desired tokenizer defined in code above, your desired list of stop words, defined in relation to the language being used and ngram\_range based on how many words you would like to look at together for example bigrams look at 2 words at a time. This can help with better interpretation of texts.

# Model development

I used 2 different models to predict whether a tweet was to do with sports 1 or baking 0, a Logistic regression model and an LSTM I then used performance metrics to compare both methods.

## Logistic regression model

Logistic regression is a simple classification model used with binary classification, for this reason it is perfect for our data which only offers 2 possible labels. It is often used in cases such as spam detection which could also present itself as a nlp problem.

Before setting up the model I created my X train, X test, and y train and test using the train\_test\_split function from sklearn. I used a test split of 20%.

Graphical user interface, application, Word

Description automatically generated

Figure 7

I defined my regression model with 200 iterations and then fit it to the values for my X and Y train variables.

Graphical user interface, application, Word

Description automatically generated

Figure 8

Once I had run the model, I evaluated 4 different metrics offered by sklearn, accuracy, precision, recall and f1 score.

Graphical user interface, application, Word

Description automatically generated

Figure 9

Accuracy relates to the how accurate your models’ predictions have been compared to the true values whilst precision evaluates on how many values tested positive were actually positive. Recall focuses on false negative and the f1 score is a measure of balance between precision and Recall.

We can see the algorithm is working the way desired by showing the weights given to certain words therefore showing which features are most important in defining the genre of each tweet.

I also used code from IPython notebook *(juliandariomirandacalle, 2020)* to evaluate which features were most important in classifying the tweet (seen in figure 10). Unsurprisingly “sport” and “baking” where in within the words with the highest weights. By visualising features weights, it helps with the explain ability of the algorithm as it helps users to understand what section soft text the model basing its decisions on.

Graphical user interface, application, Word

Description automatically generated

Figure 10

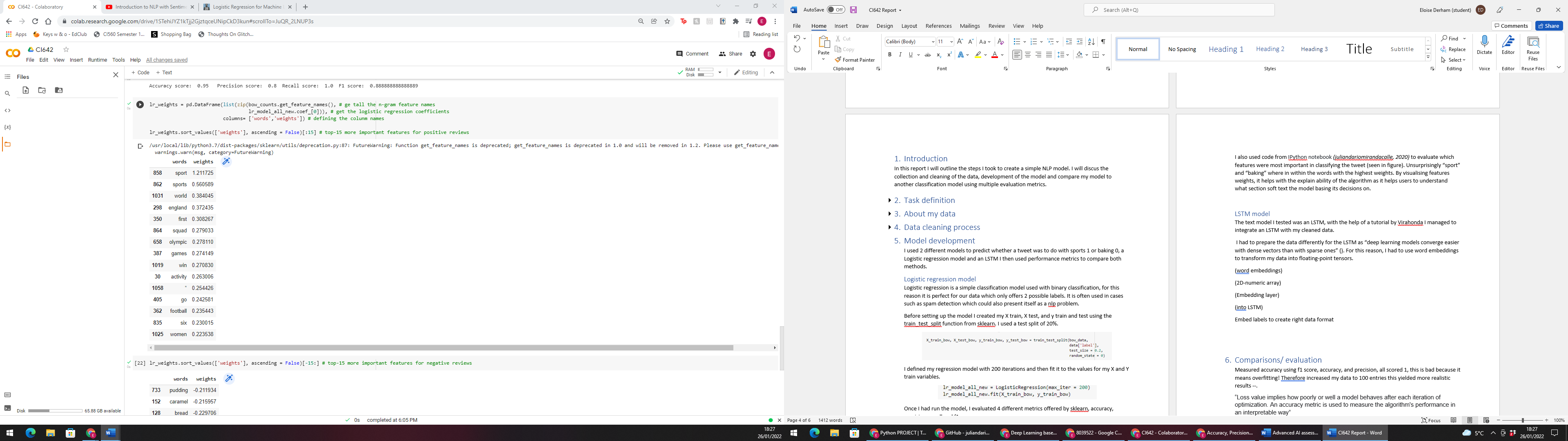


Figure 11

Graphical user interface, application, Word

Description automatically generated

Figure 12

## LSTM model

The text model I tested was an LSTM, with the help of a tutorial by Virahonda I managed to integrate an LSTM network with my cleaned data.

A Long Short Term Memory network is a neural network designed to recognize patterns in data taking time sequence into account. They are therefore good for NLP problems as to understand the context of language it is important to look at sentences as a whole, using long short term memory makes this possible.

I had to prepare the data differently for the LSTM as “deep learning models converge easier with dense vectors than with sparse ones” *(Virahonda, 2020).* For this reason, I had to use word embeddings to transform my data into floating-point tensors.

Graphical user interface, application, Word

Description automatically generated

Figure 13

I also had to encode the labels to match the data type, so that they could be used in the train and test data split.

Graphical user interface, application, Word

Description automatically generated

Figure 14

I then defined my train and test variables with a 20% test split to match the linear regression model.

Graphical user interface, application, Word

Description automatically generated

Figure 15

Next, I defined and compiled my LSTM model. I created a simple model with 3 layers. An embedding layer to fit the neural net to text data by representing word data as a dense vector representation, a LSTM layer and a final dense layer for the output of predictions. I used the sigmoid activation function with the dense layer as this is the best function for binary classification problems.

When compiling the model, I used an adam optimizer and a binary\_crossentropy loss function as the nature of my problem is binary classification. “The optimization algorithm is used to train the network and the loss function is used to evaluate the network that is minimized by the optimization algorithm”. *(https://www.facebook.com/MachineLearningMastery, 2017)*

Graphical user interface, text, application

Description automatically generated

Figure 16

I evaluated my model based on accuracy metrics as well as loss. I displayed these results using matplot library as displayed in figures 17 and 18.

Graphical user interface, application

Description automatically generated

Figure 17

“The training loss indicates how well the model is fitting the training data, while the validation loss indicates how well the model fits new data” *(baeldung, 2020)*

If our graph did show a clear decrease in loss, then our model could be underfitting our data. Underfitting is defined as such “When the algorithm is not able to model either training data or new data, consistently obtaining high error values that don’t decrease over time” *(baeldung, 2020).*

Graphical user interface, application

Description automatically generated

Figure 18

Accuracy is shown based on the validation and training sets of our model, here we can see that the training accuracy peaks at 1 whilst the validation accuracy peaks at 0.95, if there was a large gap between both values it could mean that our model was overfitting on the data. If the model was found to be overfitting drop out layers can be used to try to increase validation accuracy.

# Comparisons/ evaluation

At first my metrics for the regression model all showed a score of one, this was because my data set was too small, for this reason I increased my data to 100 inputs to get a more realistic feel for the use of my models.

LSTM models in general are able to gather a greater understanding of the patterns within data and therefore are better for more complex NLP problems however logistic regression algos are easier to understand and explain, using them for NLP makes it easier to identify what your algorithm is using to come to its conclusion. Explain ability can help to validate the accuracy of your model.

Overall, I received the same accuracy between both models. Therefore, it is hard to say which model is better suited for this task. However, for more complex NLP problems not focusing on binary classification, where more context needs to be understood such as personal assistants LSTM networks are the better option.

‌

‌

‌

‌

# References

Ebrahim, M. (2021). *Remove punctuation using Python*. [online] Like Geeks. Available at: https://likegeeks.com/python-remove-punctuation/ [Accessed 25 Jan. 2022].

https://www.facebook.com/MachineLearningMastery (2017). *The 5 Step Life-Cycle for Long Short-Term Memory Models in Keras*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/5-step-life-cycle-long-short-term-memory-models-keras/ [Accessed 27 Jan. 2022].

Miranda-Calle, Julián Darío, IPython notebook: Introduction to NLP with Sentiment Analysis, July 2020, URL: <https://github.com/juliandariomirandacalle/NLP_Notebooks>

Sas.com. (2022). *What is Natural Language Processing?* [online] Available at: https://www.sas.com/en\_us/insights/analytics/what-is-natural-language-processing-nlp.html [Accessed 28 Jan. 2022].

Stewart, M. (2020). *Guide to Classification on Imbalanced Datasets - Towards Data Science*. [online] Medium. Available at: https://towardsdatascience.com/guide-to-classification-on-imbalanced-datasets-d6653aa5fa23 [Accessed 21 Jan. 2022].

262588213843476 (2019). *A python script to preprocess text (remove URL, lowercase, tokenize, etc..)*. [online] Gist. Available at: <https://gist.github.com/MrEliptik/b3f16179aa2f530781ef8ca9a16499af> [Accessed 19 Jan. 2022].

‌

‌Bibliography

Virahonda, S. (2020). *An easy tutorial about Sentiment Analysis with Deep Learning and Keras*. [online] Medium. Available at: https://towardsdatascience.com/an-easy-tutorial-about-sentiment-analysis-with-deep-learning-and-keras-2bf52b9cba91 [Accessed 25 Jan. 2022].

https://www.facebook.com/MachineLearningMastery (2019). *How to Calculate Precision, Recall, F1, and More for Deep Learning Models*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/how-to-calculate-precision-recall-f1-and-more-for-deep-learning-models/ [Accessed 26 Jan. 2022].

Analytics Vidhya. (2020). *Text Classification in Natural Language Processing*. [online] Available at: https://www.analyticsvidhya.com/blog/2020/12/understanding-text-classification-in-nlp-with-movie-review-example-example/ [Accessed 26 Jan. 2022].

https://www.facebook.com/MachineLearningMastery (2017). *How to Use Word Embedding Layers for Deep Learning with Keras*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/use-word-embedding-layers-deep-learning-keras/ [Accessed 27 Jan. 2022].

user8512104 (2018). *classification metrics can’t handle a mix of continuous-multioutput and multi-label-indicator targets*. [online] Stack Overflow. Available at: https://stackoverflow.com/questions/48987959/classification-metrics-cant-handle-a-mix-of-continuous-multioutput-and-multi-la/48991515 [Accessed 26 Jan. 2022].

Data Pagla (2016). *Plotting learning curve in keras gives KeyError: “val\_acc.”* [online] Stack Overflow. Available at: https://stackoverflow.com/questions/39883331/plotting-learning-curve-in-keras-gives-keyerror-val-acc [Accessed 26 Jan. 2022].

Pluralsight.com. (2019). *Deep Learning Model Data Visualization using Matplotlib | Pluralsight*. [online] Available at: https://www.pluralsight.com/guides/data-visualization-deep-learning-model-using-matplotlib [Accessed 26 Jan. 2022].

https://facebook.com/vitalflux and http://facebook.com/vitalflux (2022). *Accuracy, Precision, Recall & F1-Score - Python Examples - Data Analytics*. [online] Data Analytics. Available at: https://vitalflux.com/accuracy-precision-recall-f1-score-python-example/ [Accessed 26 Jan. 2022].

Team, K. (2022). *Keras documentation: Metrics*. [online] Keras.io. Available at: https://keras.io/api/metrics/ [Accessed 26 Jan. 2022].