UNIVERSITY OF ABERDEEN
DEVELOPMENT TRUST

Assessment 3

Applied Artificial Intelligence
CS5079

Yassin Dinana

52094480

1. Task 1:

Sentiment analysis is used widely in different applications and by different service suppliers. It is an automated process that analyses texts for polarity, after the text is processed the output is either positive or negative. Machine learning tools are used in this process where the system is trained to learn to detect sentiment without human interaction.

1.1- In order to create a sentiment analysis platform, the first important step is to retrieve and gather the correct data, the best way to train a model is to use real-world data, the given dataset is ideal as it includes real-world feedbacks by previous customers using a variety of vocabulary. After retrieving the correct data, different ways can be used to clean the dataset and apply feature extractions. As two datasets are given, the two datasets are combined and concatenated in one data frame and then apply abstraction skills and remove all irrelevant and redundant data if any. As the data is text, feature engineering skills can be applied to improve the training and accuracy, this process can be done by removing all HTML tags, remove accented characters, removing all special characters, remove stop words, and expand the contractions. Different models can be applied to train the model such as Bag of Words, TF IDF, and LSTM, according the chosen model to be used the pre-processing technique might be different and vary. As each implemented model will have different performance after evaluating it, the model that will be choose for deployment is the model with the best accuracy, lowest loss, and good runtime after implementing and comparing the models. When deploying the model online, different methods are considered such as using docket to containerize the flask application and then hosting it on a web service, another method could be pushing the trained model on GitHub and pushing the model to the web using Heroku and the git function.

1.2- The given dataset is consisted of two CSV files, where one file includes all the positive reviews which are labelled as “1” and the other file includes all negative feedbacks labelled as “0”. Looking at **figure 1** in the appendix, it can be seen that the datasets were importing using the Pandas Library function and then both datasets are concatenated into one data frame using the same library – it can also be seen that the first five rows of the dataset are printed including the reviews and their label. In order to visualise the shape of the dataset, it is also printed in **figure 1** where it is seen that the shape of the dataset is 50000 rows and 2 columns. By visualising the dataset after concatenating, it can be seen that the first 25000 reviews at positive and the last 25000 are negative, therefore, looking at **figure 2** in the appendix, it can be seen that the dataset is now shuffled which will improve the training process later on.

1.3- Using Exploratory Data Analysis, better known as EDA, to understand and analyses the raw data before applying any pre-processing to it. As the data is text, there are many ways to understand it. Looking at **figure 3** in the appendix, it can be seen that the data set does not contain any missing values which are represented as NaN or Null when checked, the information of the dataset can be also visualised, it can be seen that the data types of the review column are objects and as the label column is only consist of 0 and 1, therefore the data type in int64. It can be seen that there is a total of 50000 entries in the dataset.

Looking at **figure 4** in the appendix, it is important to check what are the most common words throughout the dataset, where after printing the 10 most common words, it can be seen that the word “The” is the most common with 156501 appearances. By looking at the dataset files individually, it can be seen that it includes HTML tags and special characters and other features which will tackled in the data pre-processing and features extractions later in this report.

1.4- Different pre-processing techniques are applied to the dataset in order to clean it from all characters and unwanted tags.

Removing All HTML tags: Looking at **figure 5** in the appendix, it can be seen that the Beautiful Soup package is used to remove and strip all HTML tags from the dataset, those HTML tags include (< >, /, \). Looking at **figure 6** it can be seen that a sentence from the dataset which includes HTML tags in printed which includes (<br /><br/>so) where the same sentence is re-printed after the pre-processing and it can be seen it does not include the HTML tags.

Removing All Special Characters: Looking at **figure 7** in the appendix, in can be seen that a function is implemented to remove all special characters from the dataset such as (=, ^, @, /). An example is also implemented in the same figure where a review pulled out from the dataset included many special characters and after running the function all the special characters are removed confirming that the function is working properly.

Expanding Contractions: In order to allow better performance, all the contractions in the dataset must be removed, an example of expanding contractions could be (can't = cannot or I'm = I am). Looking at **figure 8** in the appendix, a function is created to expand all contractions in the data set using a contraction dictionary which is called from the contractions library imported in the first cell of the code. After running the function and printing an example from the dataset, it can be seen in **figure 8** that the sentence used to include (I've been) and after cleaning it, it is expanded to be (I have been).

Lemmatizing Text: A function is implemented which is called Lemmatize Text, which is checking for a base word and all its forms. For example, the word "Love" can also be presented as Loves or Loving, when grouping and processing the lemmatization function it makes it easier when training the model on the lemmatized data as the words are processed faster.

Removing Stop Words: Another pre-processing technique which is important is to remove all stop words, stop words are words such as "The, is, in, for", the stop words need to be removed from the dataset as they might occur in abundance and affect the classification or clustering. Looking at **figure 9** in the appendix, a function is presented which removes all stop words, the function uses the English dictionary from the NLTK library which is imported earlier in the program. An example from the dataset is also printed in the same figure where it can be seen that the word "THE" is removed from the sentence after applying the function.

1.5- Different models are implemented in this task to build the sentiment analysis prediction platform, each model has different results that the other. This section will be covering the models implemented.

- Logistic Regression on BOW and TF-IDF features:

    The first step before building the models is to normalize the dataset after the pre-processing and save the normalized data in a different csv file. This is done and the new csv file including the normalized pre-processed data is used as an input for the model.

    Looking at **figure 10** in the appendix, it can be seen that the features for both the BOW and the TF IDF models are built with the help of existing packages such as the Count Vectorizer and the TFIDF Vectorizer, the model is also fitted in the cell. Where the Bag of Words model is simplified representation of all the reviews in the dataset, where it is a representation of the text describing the occurrence of words in a document. Where the TF-IDF is used to reflect how important a word is for the normalized data, where the TF is a score and IDF is another score that represents the weight of a word.

Looking at **figure 11**, the logistic regression on the Bag of Words is fitted and the presented table in the figure shows the output of the model where the accuracy reached 88% and the precision 87%. Where the linear regression model is fitted on the train features and labels, the result is presented as a confusion matrix that is printed in the output and the figure.

Figure 12 in the appendix shows the linear regression model on TF-IDF features output, where it can be seen that the accuracy is less than when running the linear regression model on the BOW features with 72% accuracy.

- Support Vector Machines on BOW and TF-IDF features:

Implementing another model on the BOW and TF-IDF features, this time we implement the Support Vector Machine and not the Linear Regression; the SVM tried to find the best margin between the line and support vectors which separates the classes from each other, where the classes are positive and negative. **Figure 13** in the appendix shows the implementation of the SVM model, it is very similar to how the linear regression was implemented, we use svm.fit and insert the training parameters created before. By looking at the confusion matrix in figure 13, it can be seen that the accuracy is high reaching 86% when implementing the SVM on the bag of words features. **Figure 14** shows the same SVM model implemented on the TF-IDF features, where the accuracy can be seen reaching 87% which is higher than the bag of words model, unlike the output confusion matrices of the logistic regression models.

- LSTM model with embedded layer:

The LSTM model stands for the *long-short term memory* is a RNN model which includes feedback connection and it always performs very well as it can process entire sequence of data at once. Many libraries are imported in order to implement the LSTM such as sklearn, TensorFlow, and Keras. The first step implemented is to tokenize the train and test data, which is a function used to split very large texts into small lines and words, it is imported from the NLTK library and eases the process of data processing and training when the data is passed to the LSTM Neural Network for training. Using the same Tokenizer function, a word to index is built to calculate the vocabulary size and create an index of the reviews and sentences by applying a number to each word.

Figure 15 in the appendix shows the LSTM model implementation after applying the features above using Tokenizer and Lab Encoder. Looking at the figure it can be seen that an Embedded Layer is specified in the Neural Network which is the first hidden layer in the network where the input of the size of the vocabulary which is given to be 27300. The embedded layer is given a value of 128 and the dimension of the LSTM units is given to be 64. Looking at **figure 16** in the appendix, after running the LSTM model on three epochs for evaluation, it can be seen that the accuracy kept increasing by every epoch reach 92% and the loss decreasing for 50% in the first epoch to 20% in the third one.

1.6- The models presented in the section above all run on data that is spit into train and testing, where the train data is 35000 and the test data is 15000. **Figure 17** in the appendix shows how the data is splitted before being fitted to the models. Where the number of reviews is given to be 35000 which is the training data, and it can be seen that we split the data in the Reviews and Labels columns, where in the splitting process using the iloc function **:**NumberOfReviews means the last 35000 data and NumberOfReviews**:** means the first 35000 data in the dataset. The same splitting is done before the feeding of the data for each model.

Figure 18 in the appendix shows that the reviews and the label columns is then split to train and test for each of them using the "Train_Test_split" function and the test size is specified to be 30% and the train is 70%.

1.7- After comparing all models, it is visualised that the SVM model had the best accuracy overall, therefore, the SVM model is chosen to be deployed on the web app using Flask.

Looking at **figure 19** in the appendix, it can be seen how we are downloading our model using pickle, the model is downloaded as binary and will be used in PyCharm for deployment on a web. Looking at **figure 20**, the flask app is being created where in the same directory our model is located as pickle file, there is an index HTML page as well which will help for deployment. Later we run the flask app using the terminal.

Looking at figure 21, it can be seen that the webpage runs on the browser showing two different boxes. The first box includes the "Review" for the user to implement and the second is the prediction of the model to predict either it is positive or negative.

2. Task 2:

2.1- In the medical sector, patient prioritization has always been an issue where the clinicians and nurses are sometimes obligated to take difficult decisions and choices when having to deal with a big number of patients, especially when different resources are limited such as technical facilities and area. A.I engineers, and Data Scientists are now able to develop patient prioritization systems, it looks each patient's medical records and the level of their illness develop the output.

At first, the idea of the patient prioritization system needs start the service in the United Kingdom before expanding to the United States and Europe, therefore, when presenting the idea to the National Health Insurance, better known as the NHS, different regulations that are presented by the Information Commissioner's Office, better known as the ICO, need to be satisfied before starting to sell and distribute the system to different hospitals in the UK. Those frameworks and regulations are intended to allow safety and data protections for all the users, nurses, and the patients. Regulations supplied by the ICO cover different areas such as accountability and fairness, data minimisation, and transparency. If all the requirements are met, the product will cover all the ethical aspects and will be ready to be launched in the United Kingdom.

In order to expand the A.I system to the EU, different regulation that are supplied by the European Commission must be met to ensure the safety and ethicality of the system before being launched in the European market. Those regulations could be different from the ones supplied by the ICO in the UK. The frameworks supplied by the EC cover safety and ethical aspects such as privacy, safety, and environmental wellbeing.

Finally, in order to expand the U.S, the process must proceed, the regulations and guidance applied by the Food and Drug Administration, better known as the FDA, must be met in order to work in the U.S. market, those regulations are similar to the ones given by the ICO and the EC as the main goal is to ensure privacy and safety for all of them, the frameworks provided by the FDA include, quality systems and good ML practice to allow fairness, as well as safety assurance and transparency.

Markets & Areas	*2.2(A)* *Requirements for each market*	*2.2(B)* *Understanding and Explaining the requirements*	*2.2(C)* *Solutions deployed to comply with frameworks*
ICO – United Kingdom	Accountability and Governance Implications Lawfulness, Fairness, and Transparency Assessing Security and Data Minimisation Ensuring Data Subject Rights. [1]	**Accountability and Governance Implications**: This regulates the companies to carry out the Data Protection Impact Assessment, also known as the DPIA, in order to document all the needs and General Data Protection Regulation (GDPR) requirements – it mainly focuses of the data protection and patient's privacy. **Lawfulness, Fairness, and Transparency:** This regulates the concept of validity, justice and openness when applying the patient prioritization system, it also explains how to define the acceptable goals when using A.I in healthcare and how it is mandatory to avoid bias and unfairness. **Assessing Security and Data Minimisation:** This has to be used by the company to consider the kinds of privacy attacks that are vulnerable to the A.I scheme and conform with the data minimisation concept. Before releasing the product, part of the regulations given by the ICO is for the organization to consider applying for risk management and security assessing [1]. **Ensuring Data Subject Rights:** This governs the company to satisfy the data subject rights in the sense of input and output of AI systems when it comes to automatic decision making in healthcare. The A.I system in patient prioritisation should avoid categorising data and this might lead to bias, the company should also design the AI system to facilitate the human review and provide the workers in the hospitals with adequate preparation to ensure that they are able to analyse the outcomes and appreciate the weaknesses of the AI system if the output is biased.	In order for the company to meet the given frameworks in the UK, different approaches must be taken, such as avoiding the bias in the data. In order to avoid the biasing in the data, we need to ensure that the correct model is chosen for training and testing, for example choosing whether to apply supervised or unsupervised learning. The second step to avoid bias is to do the correct pre-processing, by cleaning the data and make sure that there the dataset does not includes biasing – this can be done by removing or adding data until the equal; finally, another option could be to train different dataset with different classes on different models – this will avoid biasing as each dataset will include only one type of data. All data that will be collected needs to have approval from all patients, we have to make sure the data is safe and not shareable or likeable to be leaked, this can be done by implementing many backups and adding technical abilities to the system that will make it safe against attacks and hacking.
EC - Europe	Technical Robustness and safety Privacy and data governance Diversity, non-discrimination and fairness Societal and environmental wellbeing. [2]	**Technical Robustness and safety:** Those are a must in the European market where the main objective of this obligation is the prevention of harm and acquiring high level of safety both for the staff and safety on the system itself. Such robustness and safety include the resilience to attacks such as hacking, data poisoning, and model leakage. **Privacy and data governance:** This regulation is linked to the prevention of harm of privacy, it mainly covers the privacy and data protection schemes throughout the system's entire life cycle, for all the data that is imported and exported from the system such as the patient's private information. This regulation also covers how the data is accessed by different organizations and outlining who is accessing the data and under which circumstances. [2] **Transparency:** This regulation provided the European Commission focuses on the explicability and transparency of all the element related to the A.I system in the medical sector, those areas include the traceability such as the tracing of how the algorithms are implemented, used, and to what extent this algorithm can be developed. The explainability and the communication are also important parts, meaning the ability to present a document explaining how the A.I system operates and on what rules does it choose which patient to prioritise. Finally, the communication is very important, and it is the importance of how the medical team can communicate with the system [2]. **Diversity, non-discrimination and fairness:** Diversity must be included in the AI system to make sure the system in trustworthy. This includes that the system be avoid unfair biasing at all cost, the bias can occur due inclusion of historic bias. This bias might occur discrimination, pre-justice, and marginalisation. Another important point is the accessible to all people, the system should be diverse and able to be used by people, regardless of their age, gender, abilities, and characteristics. Finally, the system must be acceptable to all feedbacks by all users and workers. **Societal and environmental wellbeing:** This regulation is presented to assure that the A.I system is safe and presents no harm for the environment, it focuses on mainly three aspects which are the sustainable and environmentally friendly A.I not only in the real-time usage but also in the deployment, use -process and the entire supply chain. The second aspect is the social impact and how it should not affect the social agency or impact the social relationships. Finally, the last aspect is the society and democracy, taking intro affect avoiding the harm to the institutions, democracy, and society at large.	In order to meet all the regulations applied by the EC in Europe when applying A.I in the medical sector, focusing first on the data protection and safety. Like other information systems, this system should be shielded from bugs that will allow hackers to manipulate them and hack it, if the model is hacked then the data privacy is also hacked. As the medical sector relates to human's life, therefore, it is very sensitive, then it is very important to make sure that the output of the system is as accurate as possible, this can be done by training the model more often and using new data for testing and apply the appropriate pre-processing to the data. This will ensure that the accuracy of the prediction is high and will be able to prioritise the patients better, meeting the requirement of non-discrimination and fairness. As this system does not have any hardware or manufacturing, therefore, it is not affecting the environment. It is also important to supply all the correct documentation to the government to make sure all the requirements are met.

FDA – United States	Quality Systems and Good Machine Learning Practice Initial premarket assurance of safety and effectiveness Transparency monitoring A.I and ML-based Software as A Medical Device (SaMD) [3]	**Quality Systems and Good Machine Learning Practice**: Before the A.I system is launched in the U.S. market it needs to be geared towards developing, delivering, and maintaining high quality. **Initial premarket assurance of safety and effectiveness:** This framework is presented to allow the modifications in the system later when it is released, it focuses on the safety and effectiveness of the system by presenting a document which relays what the algorithm of the system is intended to learn and how the system is safe and effective to the users and patients even after it is updated and modified. This focuses on data management, performance evaluation, and update procedures [3]. **Transparency and real-world performance monitoring A.I and ML -based Software as A Medical Device (SaMD):** This framework implies that the system should be transparent to all users and patients by implemented the appropriate mechanisms in the algorithms, all the data that will be used and all the outputs from the system should be transparent and available, as well as how the data is being collected and processed – the manufacturers of the system must describe how the users will be notified with the system's embedded applications and with the updates – the communication is usually through letter, emails, and notifications).	In order to meet the regulations provided by FDA in the U.S., the approaches that be took to meet the quality systems and the good ML practice, three approaches can be used such as: Make sure there is a valid clinical association between the output of the system and target condition. Make sure the system processes the input data correctly to generate accurate, reliable, and precise output data. Finally, test and make sure that the use of the precise output data achieves the intended target in the context of patient prioritisation [3]. It is also mandatory to apply data collection protocols and quality assurance to make sure the data is pre-processed correctly. It is needed to re-train the models with different parameters and architectures. When upgrading the framework, ensuring that the validation of the system is in progress, we can also prepare when and how to enforce and arrange global and local updates, and eventually extend effective coordination, communication, and accountability to users to allow the perfect transparency.

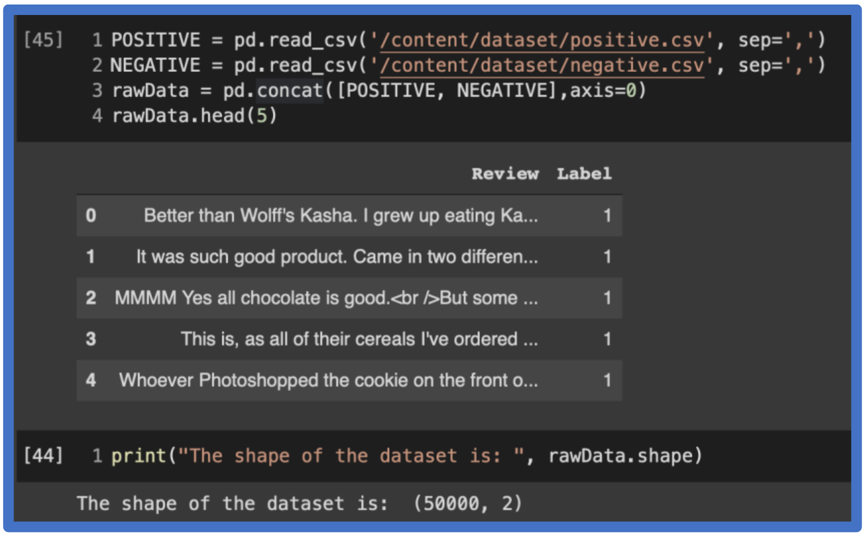
3. References:

1- ICO. "Guidance on the A.I Auditing framework". Available at: https://www.pdpjournals.com/docs/888032.pdf [Online]. Accessed 16 December 2020.

2- European Commission. 2020. "On Artificial Intelligence – A European approach to excellence and trust". Available at: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=COM:2020:65:FIN&qid=1522050991135&from=EN [Online]. Accessed 18 December 2020.

3- FDA. "Proposed Regulatory Framework for modifications to Artificial Intelligence/Machine Learning – Bases Software as Medical Device". Available at: https://www.fda.gov/files/medical%20devices/published/US-FDA-Artificial-Intelligence-and-Machine-Learning-Discussion-Paper.pdf [Online]. Accessed 19 December 2020.

4. Appendix:



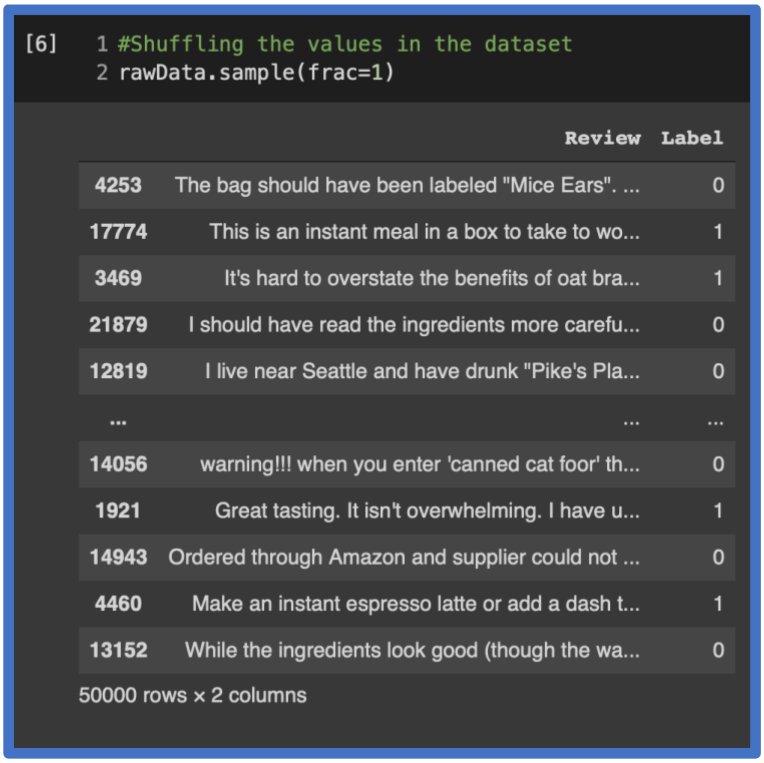
```
[45]  POSITIVE = pd.read_csv('/content/dataset/positive.csv', sep=',')
      NEGATIVE = pd.read_csv('/content/dataset/negative.csv', sep=',')
      rawData = pd.concat([POSITIVE, NEGATIVE],axis=0)
      rawData.head(5)
```

	Review	Label
0	Better than Wolff's Kasha. I grew up eating Ka...	1
1	It was such good product. Came in two differen...	1
2	MMMM Yes all chocolate is good.<br />But some ...	1
3	This is, as all of their cereals I've ordered ...	1
4	Whoever Photoshopped the cookie on the front o...	1

```
[44]  print("The shape of the dataset is: ", rawData.shape)
```

```
The shape of the dataset is:  (50000, 2)
```

Figure 1: Importing, Concatenating, and Printing the shape of the dataset.

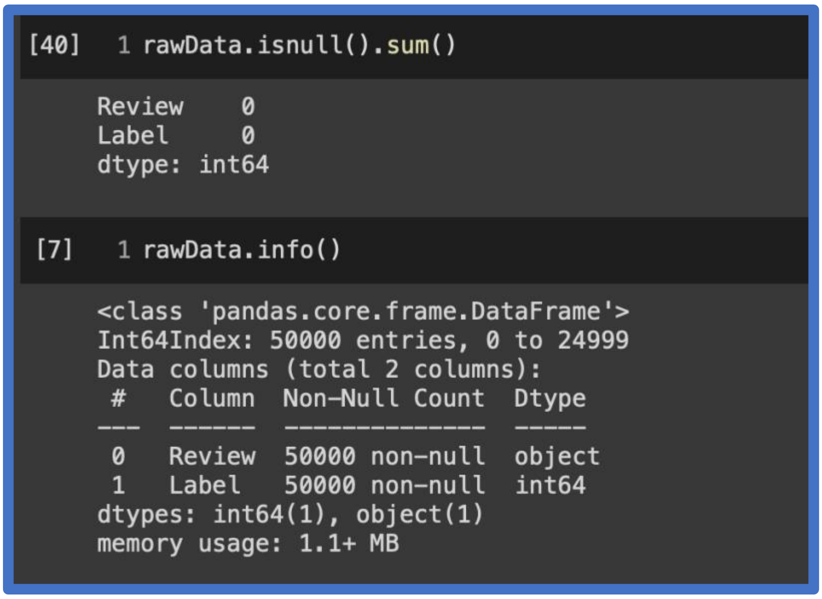


```
[6]  #Shuffling the values in the dataset
     rawData.sample(frac=1)
```

	Review	Label
4253	The bag should have been labeled "Mice Ears". ...	0
17774	This is an instant meal in a box to take to wo...	1
3469	It's hard to overstate the benefits of oat bra...	1
21879	I should have read the ingredients more carefu...	0
12819	I live near Seattle and have drunk "Pike's Pla...	0
...	...	...
14056	warning!!! when you enter 'canned cat foor' th...	0
1921	Great tasting. It isn't overwhelming. I have u...	1
14943	Ordered through Amazon and supplier could not ...	0
4460	Make an instant espresso latte or add a dash t...	1
13152	While the ingredients look good (though the wa...	0

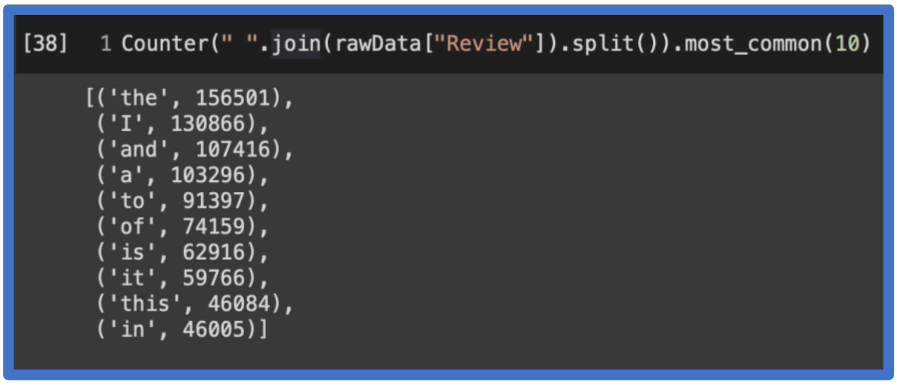
50000 rows × 2 columns

Figure 2: Shuffling the dataset



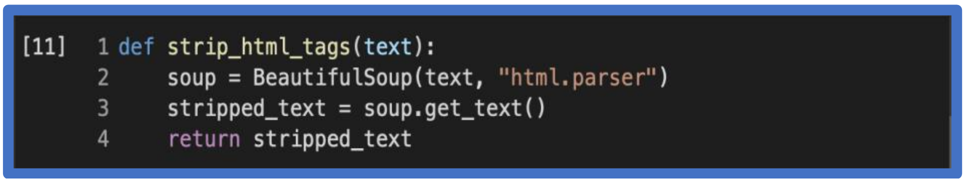
```
[40] rawData.isnull().sum()

     Review    0
     Label     0
     dtype: int64

[7]  rawData.info()

     <class 'pandas.core.frame.DataFrame'>
     Int64Index: 50000 entries, 0 to 24999
     Data columns (total 2 columns):
      #   Column  Non-Null Count  Dtype
     ---  ------  --------------  -----
      0   Review  50000 non-null  object
      1   Label   50000 non-null  int64
     dtypes: int64(1), object(1)
     memory usage: 1.1+ MB
```

Figure 3: Visualising the info of the dataset



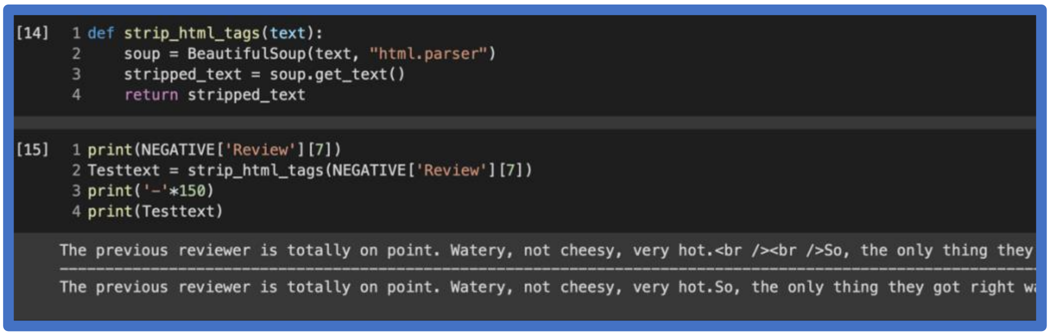
```
[38] Counter(" ".join(rawData["Review"]).split()).most_common(10)

     [('the', 156501),
      ('I', 130866),
      ('and', 107416),
      ('a', 103296),
      ('to', 91397),
      ('of', 74159),
      ('is', 62916),
      ('it', 59766),
      ('this', 46084),
      ('in', 46005)]
```

Figure 4: Visualising the most common words



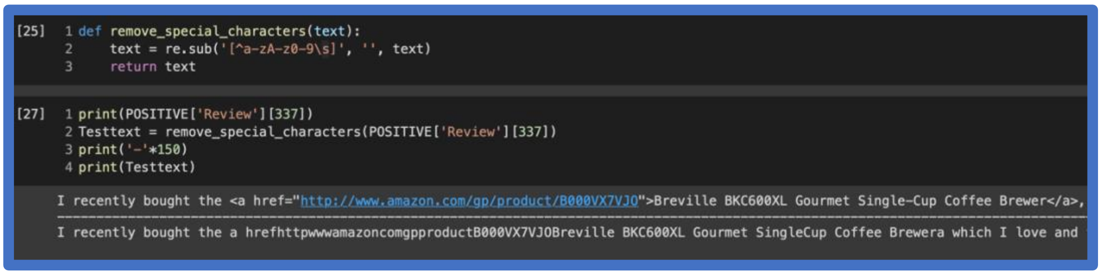
```
[11] def strip_html_tags(text):
         soup = BeautifulSoup(text, "html.parser")
         stripped_text = soup.get_text()
         return stripped_text
```

Figure 5: Removing all HTML tags from the dataset



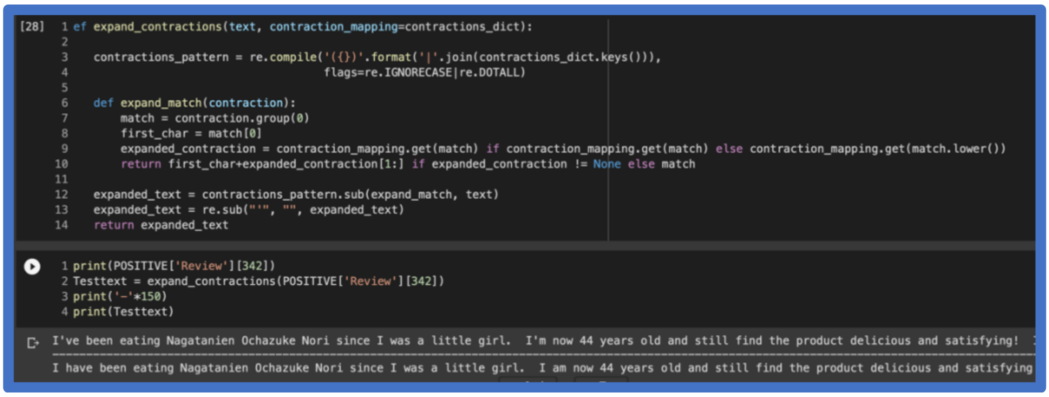
```
[14] def strip_html_tags(text):
         soup = BeautifulSoup(text, "html.parser")
         stripped_text = soup.get_text()
         return stripped_text

[15] print(NEGATIVE['Review'][7])
     Testtext = strip_html_tags(NEGATIVE['Review'][7])
     print('-'*150)
     print(Testtext)

     The previous reviewer is totally on point. Watery, not cheesy, very hot.<br /><br />So, the only thing they
     ----------------------------------------------------------------------------------------------------------------
     The previous reviewer is totally on point. Watery, not cheesy, very hot.So, the only thing they got right wa
```

Figure 6: Showing before and after HTML tags removed



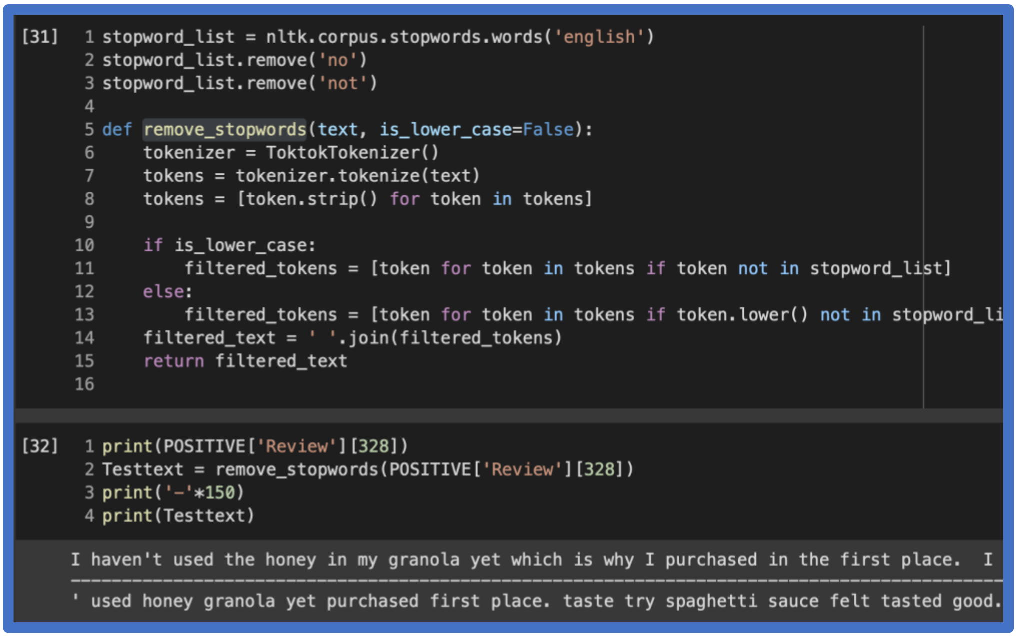
```
[25] def remove_special_characters(text):
         text = re.sub('[^a-zA-z0-9\s]', '', text)
         return text

[27] print(POSITIVE['Review'][337])
     Testtext = remove_special_characters(POSITIVE['Review'][337])
     print('-'*150)
     print(Testtext)

I recently bought the <a href="http://www.amazon.com/gp/product/B000VX7VJ0">Breville BKC600XL Gourmet Single-Cup Coffee Brewer</a>,
------------------------------------------------------------------------------------------------------------------------------------------------------
I recently bought the a hrefhttpwwwamazoncomgpproductB000VX7VJOBreville BKC600XL Gourmet SingleCup Coffee Brewera which I love and
```

Figure 7: Function to remove all special characters with example



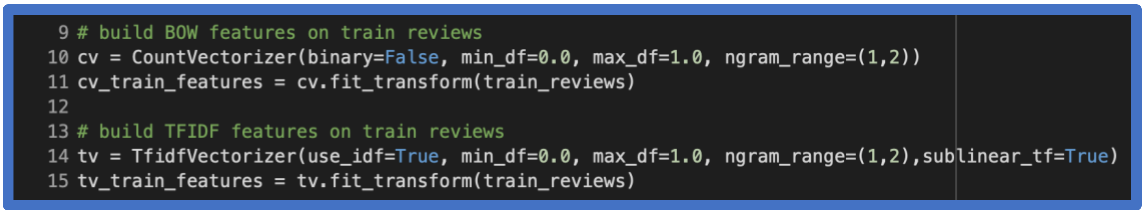
```
[28] ef expand_contractions(text, contraction_mapping=contractions_dict):

         contractions_pattern = re.compile('({})'.format('|'.join(contractions_dict.keys())),
                                           flags=re.IGNORECASE|re.DOTALL)

         def expand_match(contraction):
             match = contraction.group(0)
             first_char = match[0]
             expanded_contraction = contraction_mapping.get(match) if contraction_mapping.get(match) else contraction_mapping.get(match.lower())
             return first_char+expanded_contraction[1:] if expanded_contraction != None else match

         expanded_text = contractions_pattern.sub(expand_match, text)
         expanded_text = re.sub("'", "", expanded_text)
         return expanded_text

     print(POSITIVE['Review'][342])
     Testtext = expand_contractions(POSITIVE['Review'][342])
     print('-'*150)
     print(Testtext)

I've been eating Nagatanien Ochazuke Nori since I was a little girl.  I'm now 44 years old and still find the product delicious and satisfying!
------------------------------------------------------------------------------------------------------------------------------------------------------
I have been eating Nagatanien Ochazuke Nori since I was a little girl.  I am now 44 years old and still find the product delicious and satisfying
```

Figure 8: Expanding all contractions in the dataset with example



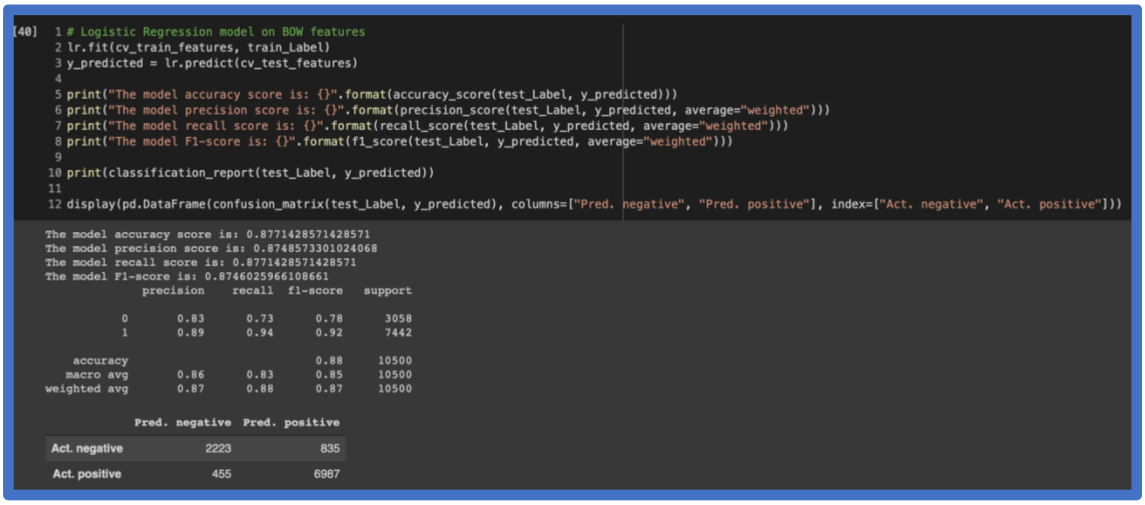
```
[31] stopword_list = nltk.corpus.stopwords.words('english')
     stopword_list.remove('no')
     stopword_list.remove('not')

     def remove_stopwords(text, is_lower_case=False):
         tokenizer = ToktokTokenizer()
         tokens = tokenizer.tokenize(text)
         tokens = [token.strip() for token in tokens]

         if is_lower_case:
             filtered_tokens = [token for token in tokens if token not in stopword_list]
         else:
             filtered_tokens = [token for token in tokens if token.lower() not in stopword_li
         filtered_text = ' '.join(filtered_tokens)
         return filtered_text


[32] print(POSITIVE['Review'][328])
     Testtext = remove_stopwords(POSITIVE['Review'][328])
     print('-'*150)
     print(Testtext)

I haven't used the honey in my granola yet which is why I purchased in the first place.  I
------------------------------------------------------------------------------------------------------------------------------------------------------
' used honey granola yet purchased first place. taste try spaghetti sauce felt tasted good.
```

Figure 9: Removing all stop words from the dataset with example



```
# build BOW features on train reviews
cv = CountVectorizer(binary=False, min_df=0.0, max_df=1.0, ngram_range=(1,2))
cv_train_features = cv.fit_transform(train_reviews)

# build TFIDF features on train reviews
tv = TfidfVectorizer(use_idf=True, min_df=0.0, max_df=1.0, ngram_range=(1,2),sublinear_tf=True)
tv_train_features = tv.fit_transform(train_reviews)
```

Figure 10: Building the features for BOW and TFIDF models

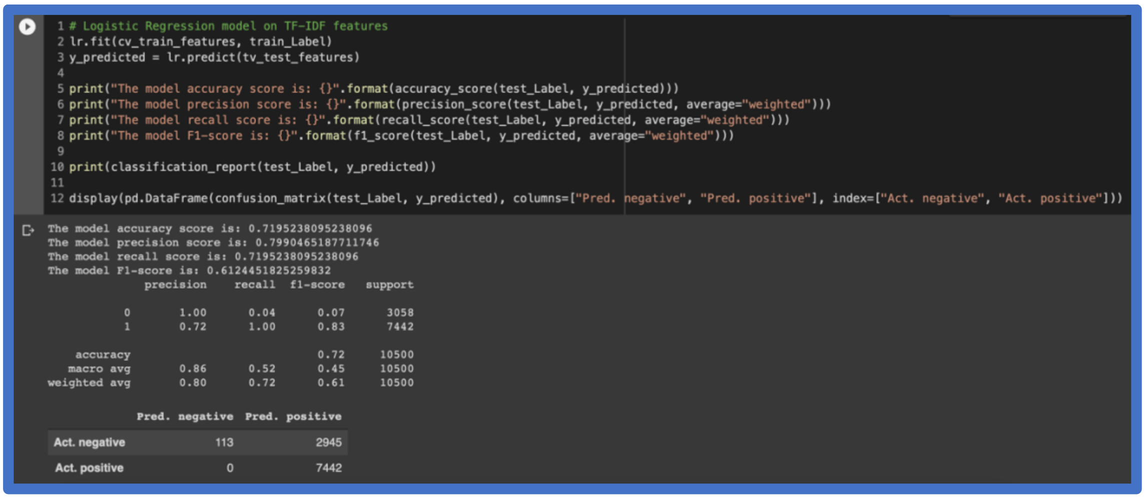


```python
# Logistic Regression model on BOW features
lr.fit(cv_train_features, train_Label)
y_predicted = lr.predict(cv_test_features)

print("The model accuracy score is: {}".format(accuracy_score(test_Label, y_predicted)))
print("The model precision score is: {}".format(precision_score(test_Label, y_predicted, average="weighted")))
print("The model recall score is: {}".format(recall_score(test_Label, y_predicted, average="weighted")))
print("The model F1-score is: {}".format(f1_score(test_Label, y_predicted, average="weighted")))

print(classification_report(test_Label, y_predicted))

display(pd.DataFrame(confusion_matrix(test_Label, y_predicted), columns=["Pred. negative", "Pred. positive"], index=["Act. negative", "Act. positive"]))
```

```
The model accuracy score is: 0.8771428571428571
The model precision score is: 0.8748573301024068
The model recall score is: 0.8771428571428571
The model F1-score is: 0.8746025966108661
              precision    recall  f1-score   support

           0       0.83      0.73      0.78      3058
           1       0.89      0.94      0.92      7442

    accuracy                           0.88     10500
   macro avg       0.86      0.83      0.85     10500
weighted avg       0.87      0.88      0.87     10500
```

	Pred. negative	Pred. positive
Act. negative	2223	835
Act. positive	455	6987

Figure 11: Output of the logistic regression model on BOW features.

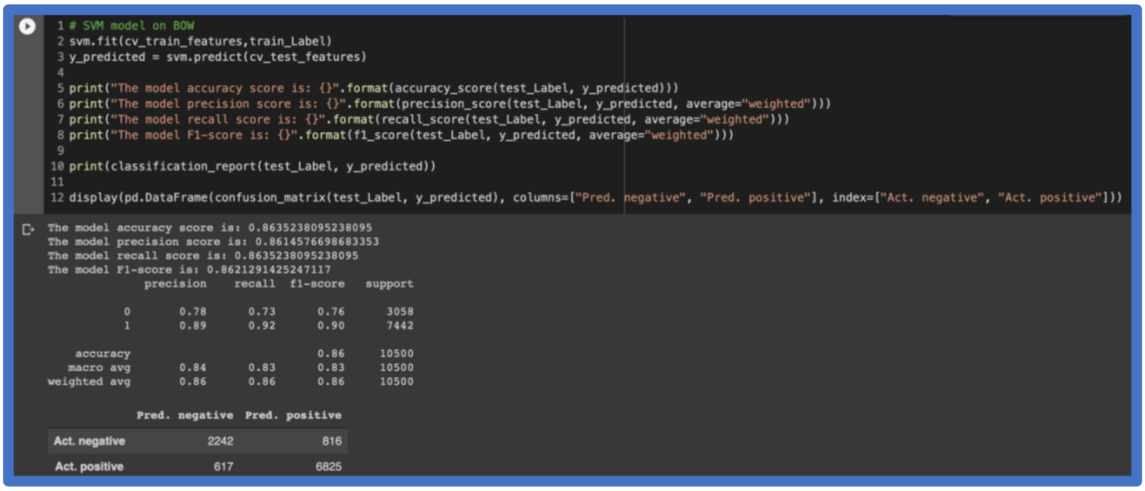


```python
# Logistic Regression model on TF-IDF features
lr.fit(cv_train_features, train_Label)
y_predicted = lr.predict(tv_test_features)

print("The model accuracy score is: {}".format(accuracy_score(test_Label, y_predicted)))
print("The model precision score is: {}".format(precision_score(test_Label, y_predicted, average="weighted")))
print("The model recall score is: {}".format(recall_score(test_Label, y_predicted, average="weighted")))
print("The model F1-score is: {}".format(f1_score(test_Label, y_predicted, average="weighted")))

print(classification_report(test_Label, y_predicted))

display(pd.DataFrame(confusion_matrix(test_Label, y_predicted), columns=["Pred. negative", "Pred. positive"], index=["Act. negative", "Act. positive"]))
```

```
The model accuracy score is: 0.7195238095238096
The model precision score is: 0.7990465187711746
The model recall score is: 0.7195238095238096
The model F1-score is: 0.6124451825259832
              precision    recall  f1-score   support

           0       1.00      0.04      0.07      3058
           1       0.72      1.00      0.83      7442

    accuracy                           0.72     10500
   macro avg       0.86      0.52      0.45     10500
weighted avg       0.80      0.72      0.61     10500
```

	Pred. negative	Pred. positive
Act. negative	113	2945
Act. positive	0	7442

Figure 12: Output of the logistic regression model on TF-IDF features

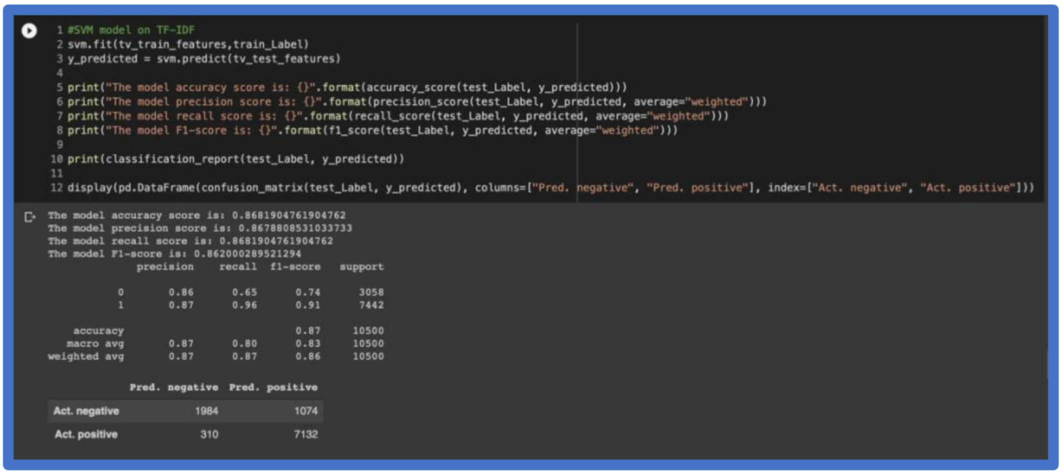


```python
# SVM model on BOW
svm.fit(cv_train_features,train_Label)
y_predicted = svm.predict(cv_test_features)

print("The model accuracy score is: {}".format(accuracy_score(test_Label, y_predicted)))
print("The model precision score is: {}".format(precision_score(test_Label, y_predicted, average="weighted")))
print("The model recall score is: {}".format(recall_score(test_Label, y_predicted, average="weighted")))
print("The model F1-score is: {}".format(f1_score(test_Label, y_predicted, average="weighted")))

print(classification_report(test_Label, y_predicted))

display(pd.DataFrame(confusion_matrix(test_Label, y_predicted), columns=["Pred. negative", "Pred. positive"], index=["Act. negative", "Act. positive"]))
```

```
The model accuracy score is: 0.8635238095238095
The model precision score is: 0.8614576698683353
The model recall score is: 0.8635238095238095
The model F1-score is: 0.8621291425247117
              precision    recall  f1-score   support

           0       0.78      0.73      0.76      3058
           1       0.89      0.92      0.90      7442

    accuracy                           0.86     10500
   macro avg       0.84      0.83      0.83     10500
weighted avg       0.86      0.86      0.86     10500
```

	Pred. negative	Pred. positive
Act. negative	2242	816
Act. positive	617	6825

Figure 13: Support Vector Machines model implementation of BOW

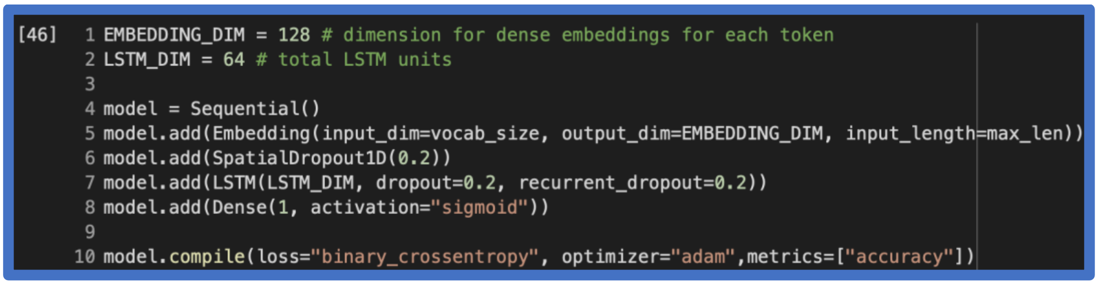


```
#SVM model on TF-IDF
svm.fit(tv_train_features,train_Label)
y_predicted = svm.predict(tv_test_features)

print("The model accuracy score is: {}".format(accuracy_score(test_Label, y_predicted)))
print("The model precision score is: {}".format(precision_score(test_Label, y_predicted, average="weighted")))
print("The model recall score is: {}".format(recall_score(test_Label, y_predicted, average="weighted")))
print("The model F1-score is: {}".format(f1_score(test_Label, y_predicted, average="weighted")))

print(classification_report(test_Label, y_predicted))

display(pd.DataFrame(confusion_matrix(test_Label, y_predicted), columns=["Pred. negative", "Pred. positive"], index=["Act. negative", "Act. positive"]))
```

```
The model accuracy score is: 0.8681904761904762
The model precision score is: 0.8678808531033733
The model recall score is: 0.8681904761904762
The model F1-score is: 0.862000289521294
              precision    recall  f1-score   support

           0       0.86      0.65      0.74      3058
           1       0.87      0.96      0.91      7442

    accuracy                           0.87     10500
   macro avg       0.87      0.80      0.83     10500
weighted avg       0.87      0.87      0.86     10500
```

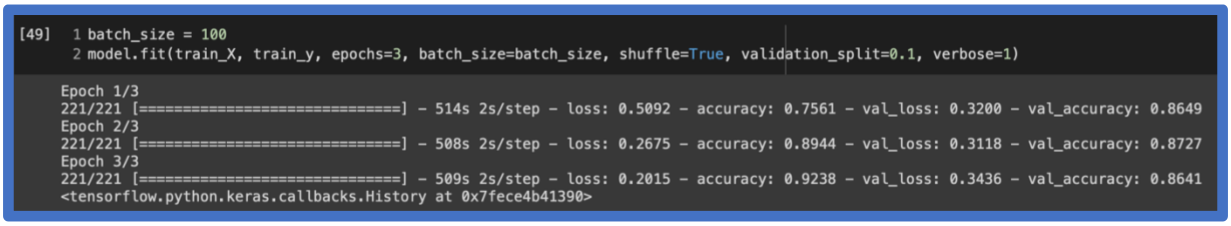
	Pred. negative	Pred. positive
Act. negative	1984	1074
Act. positive	310	7132

Figure 14: Support Vector Machines model implementation of TF-IDF



```
EMBEDDING_DIM = 128 # dimension for dense embeddings for each token
LSTM_DIM = 64 # total LSTM units

model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=EMBEDDING_DIM, input_length=max_len))
model.add(SpatialDropout1D(0.2))
model.add(LSTM(LSTM_DIM, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(1, activation="sigmoid"))

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

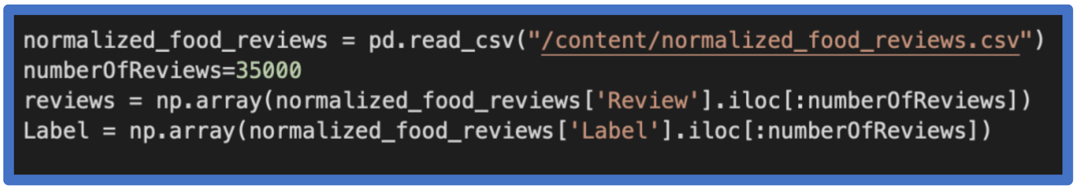
Figure 15: Implementation of the LSTM model.



```
batch_size = 100
model.fit(train_X, train_y, epochs=3, batch_size=batch_size, shuffle=True, validation_split=0.1, verbose=1)
```

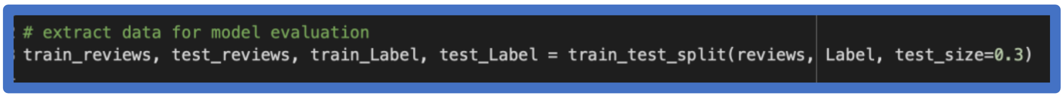
```
Epoch 1/3
221/221 [==============================] - 514s 2s/step - loss: 0.5092 - accuracy: 0.7561 - val_loss: 0.3200 - val_accuracy: 0.8649
Epoch 2/3
221/221 [==============================] - 508s 2s/step - loss: 0.2675 - accuracy: 0.8944 - val_loss: 0.3118 - val_accuracy: 0.8727
Epoch 3/3
221/221 [==============================] - 509s 2s/step - loss: 0.2015 - accuracy: 0.9238 - val_loss: 0.3436 - val_accuracy: 0.8641
<tensorflow.python.keras.callbacks.History at 0x7fece4b41390>
```

Figure 16: Training the LSTM model on three epochs.



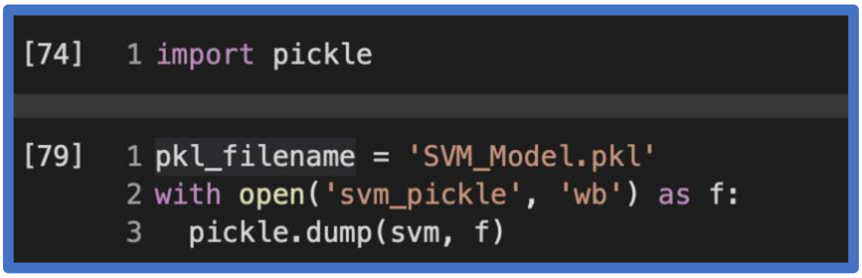
```
normalized_food_reviews = pd.read_csv("/content/normalized_food_reviews.csv")
numberOfReviews=35000
reviews = np.array(normalized_food_reviews['Review'].iloc[:numberOfReviews])
Label = np.array(normalized_food_reviews['Label'].iloc[:numberOfReviews])
```

Figure 17: Splitting the dataset to training and testing data



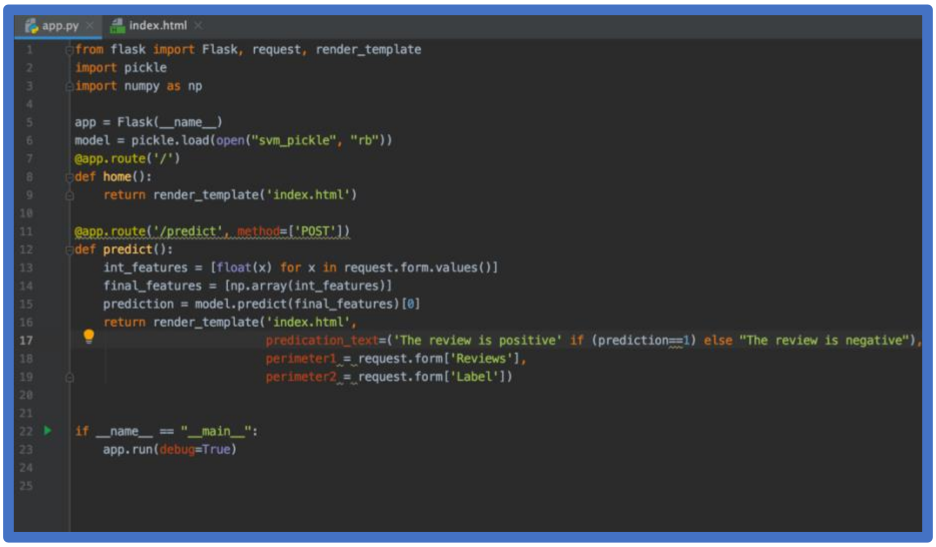
```
# extract data for model evaluation
train_reviews, test_reviews, train_Label, test_Label = train_test_split(reviews, Label, test_size=0.3)
```

Figure 18: Splitting the dataset and use Train_Test_Split function.



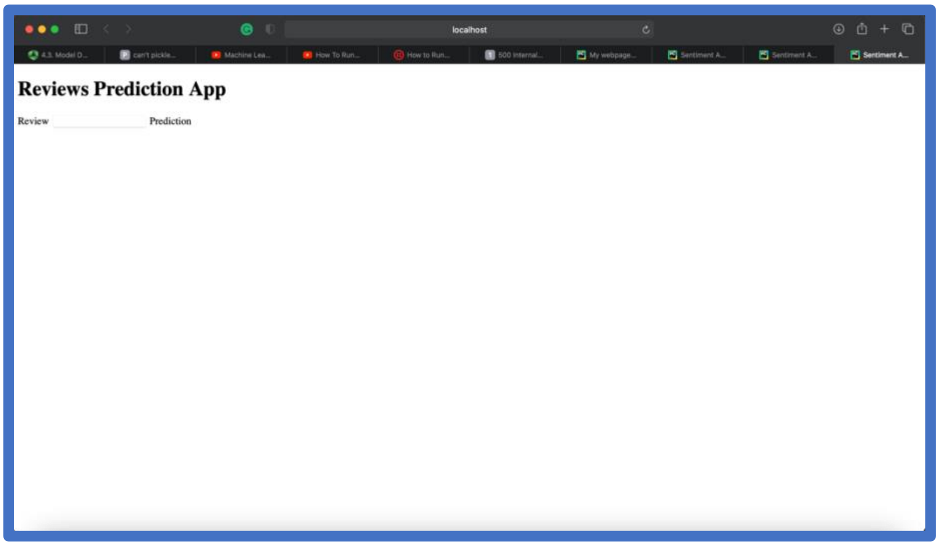
```
[74]  import pickle

[79]  pkl_filename = 'SVM_Model.pkl'
      with open('svm_pickle', 'wb') as f:
        pickle.dump(svm, f)
```

Figure 19: Downloading the SVM model using pickle.



```
from flask import Flask, request, render_template
import pickle
import numpy as np

app = Flask(__name__)
model = pickle.load(open("svm_pickle", "rb"))
@app.route('/')
def home():
    return render_template('index.html')

@app.route('/predict', method=['POST'])
def predict():
    int_features = [float(x) for x in request.form.values()]
    final_features = [np.array(int_features)]
    prediction = model.predict(final_features)[0]
    return render_template('index.html',
                           predication_text=('The review is positive' if (prediction==1) else "The review is negative"),
                           perimeter1 = request.form['Reviews'],
                           perimeter2 = request.form['Label'])


if __name__ == "__main__":
    app.run(debug=True)
```

Figure 20: Creating the FLASK App.



Reviews Prediction App

Review Prediction

Figure 21: Web Page deploying model on browser