TARC
TUNKU ABDUL RAHMAN UNIVERSITY COLLEGE
BEYOND EDUCATION

# BMCS2003 ARTIFICIAL INTELLIGENCE

**202205 Session, Year 2022/23**

**Assignment Documentation**

| |
|---|
| **Full Name: CHOW WENG YONG** |
| **Student ID: 21WMR04874** |
| **Programme: Bachelor of Information Technology (Honours) in Information Security** |
| **Tutorial Class: 3** |
| **Project Title: Natural Language Processing** |
| **Module In-Charged:** - Web Crawling - Preprocessing - Long Short Term Memory - K-Nearest Neighbors - User Interface |

**Other team members' data**

| No | Student Name | Module In Charge |
|---|---|---|
| 1 | CHOW WENG YONG | - Web Crawling - Preprocessing - Long Short Term Memory - K-Nearest Neighbor - User Interface |

| Lecturer: Dr. Goh Ching Pang | Tutor: Dr. Ho Chuk Fong |
|---|---|

# 1. Introduction

## 1.1. Problem Background

A community forms by individuals. An individual exists as a distinct entity. Each individual has own individuality that possessing own needs and rights. This makes each individual have different thoughts, mindsets, perspectives, etc. Sentiment analysis comes with the aims to monitor and understand sentiment of an individual such as the feelings. Generally, sentiment analysis is often applied in the business industry. The businesses use sentiment analysis to understand their customers deeper in order to enhance their customer experience. It works through insights generation from their customers' perspective such as on the products or services reviews of customers, social media shares of customers, etc. These insights craft a unique, cognitive, compassionate empathy that build understanding with customers thus offering the businesses chances to draw closer the gap between the businesses and their customers. An understanding towards customers with the data from the past helps in predicting their purchasing behaviors in the future. That's the common purpose of sentiment analysis (Robinson, 2021).

Individuals are sensitive. They are tied with emotions for most of the time. Understanding feelings of others offers chances in getting closer as they will feel connected with the similar existences. Sentiment analysis applies by detecting the emotion of an individual at certain state through their text or speech made. An emotion such as happy, angry, sad, etc. represents the subconscious of an individual without any doubt. Individuals sometimes don't even know what they want, but their speech shows it out. Thus, sentiment analysis works here. For example, customers approach customer service and explain on their frustration through text. Analysis on the text sent, detects the emotion of customers being frustrated thus the business may make appropriate response to it such as prioritizing the customers' request. This will make an empathetic response which can be felt by the customers that the businesses care about them and pulling the relationship between the businesses and the customers to another level of closeness. There are several commonly known sentiment analysis services in the market offering services mentioned to the businesses as outsources for the businesses to not need to develop by themselves. These services offer the businesses opportunities in building empathy relationship with their customers by having a more precise business planning such as how to increase customers' user experience, retain existing customers, spread out the brand name to potential customers, etc. (Robinson, 2021).

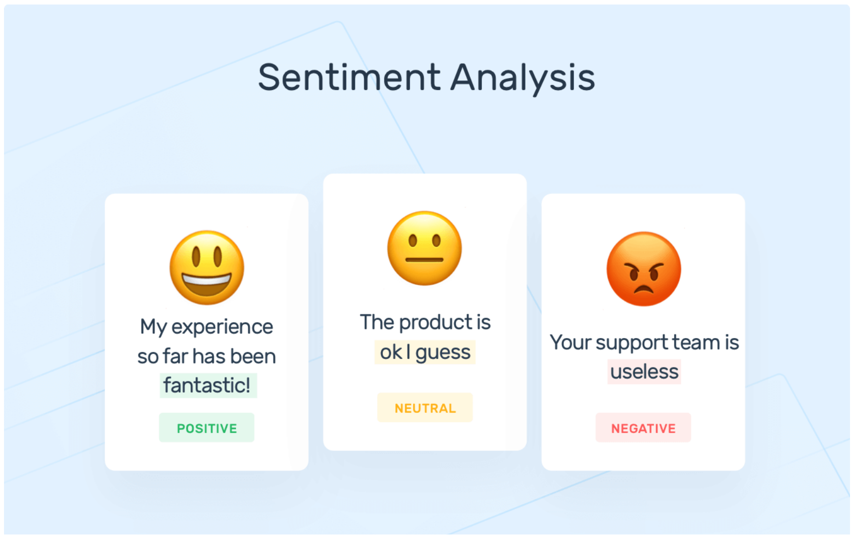

Sentiment Analysis
My experience so far has been fantastic!
POSITIVE
The product is ok I guess
NEUTRAL
Your support team is useless
NEGATIVE

*Figure 1.1 Sentiment Analysis Classification (MonkeyLearn, 2022)*

Commonly, sentiment analysis appears in the Customer Relationship Management (CRM) system. In managing customer relationship, it can be a crucial issue. There are several sentiment analysis relevant applications for businesses. Firstly, Voice of Customer (VoC) Programs that helps in understanding customers' feelings to improve customer experience by identifying and fixing any issues occurs. One of the commonly applies VoC is Net Promoter Score (NPS) that usually ask about customers' rating from certain scale such as 0-10 on certain issues. It often used by restaurant businesses in knowing how their foods quality is, services quality is, etc. However, NPS itself is purely about scoring which businesses are not able to know more precise opinions of the customers. Besides, there is brand sentiment analysis help in monitoring customers' feelings about the brands. It analyzes the customers' community to keep an eye on the businesses' reputation and even competitive advantages compared to their competitors'. Thus, making the businesses able to launch a suitable branding marketing campaign in line with the needs of their customers. On the other hand, there is social media sentiment analysis which keep increasing year by year as the widen usage of social media nowadays. Social media is considered as one of the powerful platforms in getting closer to the customers. Customers often share their thoughts, mindsets, perspectives, etc. through social media by posting it out. This will help in not only understanding customers but in improving and maintaining brand reputation as well (Thematic, 2022).

In a nutshell, the implementation of sentiment analysis is to help in solving relations issues such as for businesses knowing customers better. The businesses may improve customer service via sentiment analysis such that reviewing on customers' reviews to identify positive or negatives manner they are in at that moment thus allowing customer service to handle the customers well. Next, the businesses may then boost the products and services as they are able to stand from customers' perspective in understanding issues with their current products and services. Besides, the business may implement strategic marketing plan. Then, the businesses may monitor the brand name from building it to maintain it by keeping an eye on their customers. Moreover, the businesses may keep in touch with customers' perspective in real time with sentiment analysis by understanding them always. According to research by Apex Global Learning, every one star given by customers increases revenue by 5%~9% and a three-star rated business with five-star rated business have a different of 18% in their revenue. Thus, this is how and why sentiment analysis will work in achieving an empathy relationship between businesses and customers (Thoughts, 2021).

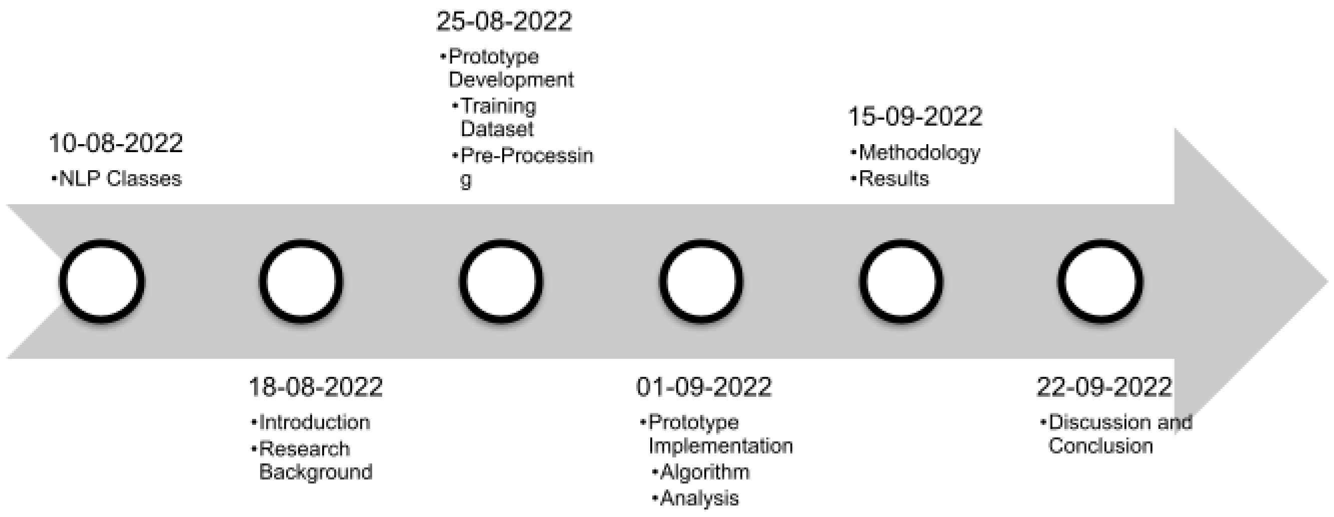
## 1.2. Objectives / Aims

As in line with sentiment analysis environment, the system is expected to be able to work with individuals' emotions through text by classifying them to positive and negative classes in order to understand their thought, mindsets, perspectives, etc. The system will first be inserted with training dataset that initially will go through the text pre-processing to eliminate unnecessary elements. Next, the clear training dataset will go through process of classification into positive and negative classes via several algorithms. Then, the algorithms will be compared in different terms such as accuracy, precision, recall, etc. to determine the effectiveness of the algorithm in working with sentiment analysis. At the end of the day, the system will be able to receive any put of statements and analyze on it to determine the state of the statement. In a nutshell, the input is used to solely differentiate sentences in general of either positive or negative such as a sentence relevant to someone personality, a sentence relevant to product review, etc. as there is no limitation of only certain statements can be processed.

## 1.3. Motivation

Sentiment analysis is common seen everywhere nowadays as it is closely related to individuals' daily life. Thus, sentiment analysis may help in different ways for different industries. First and foremost, sentiment analysis may help in retainment on loyalty of customers. The analysis learns on nature of customers such as whether customers have a good experience or a bad experience on the products or services offered by analyzing the customers' reviews of that particular product or service. If customers have a positive review, it means they are satisfied with the products or services thus there will be higher possibilities of the customers to become loyal customers then purchase again; If customers have a negative review, it means they are unsatisfied with the products or services thus there will be lower possibilities of the customers to become loyal customers then would not purchase again. In addition, the analysis may lead to spreading of brand name as well by enhancing on the analyzed positive review to a better one and improving on the analyzed negative review if any issues. Besides, apart from the main motivation of helping business in analyzing the customer review to enhance the relationship between businesses and customers from aspects of customer service, brand reputation etc., sentiment analysis may be implemented in other fields as well. For instance, sentiment analysis of either positive or negative state the statement is may help in determination on worthiness of individuals. The analysis learns on quality of individuals such as whether individuals have a good impression or a bad impression in others' eyes by analyzing the bystanders' comments on that particular individual. If bystanders have a positive comment, it means the individuals have a positive personality trait thus the individual is worthy to make friends with or worthy to learn from; If by standers have a negative comment, it means the individuals have a negative personality trait thus the individual is unworthy to make friends with or unworthy to learn from.

## 1.4. Timeline / Milestone



# 2. Research Background

## 2.1. Background of the Applications

Sentiment analysis is about text mining which identify polarity of an opinion in order to learn the nature of the opinion. The polarity of an opinion means to be the state that being completely opposite from each other such as it can be classified into classes of either positive or negative. However, the result of polarity may differ on the accuracy, precision, recall, etc. depends on the approaches apply such as the dataset used to train on, the algorithm used to analyze on, etc. (Jacob, 2010) It defines its purposes into four types such that fine-grained, emotion detection, aspect-based, and multilingual. Firstly, fine-grained analysis often used to interpret the review in form of rating between scales such as from one of unsatisfied to ten of satisfied. Next, emotion detection analysis offers analysis on emotion of individuals such as happy, angry, sad, etc. Then, aspect-based analysis analyzes the inputs through certain aspects of the inputs to focus more on it. Lastly, multilingual analysis involves numerous steps which could be complex and there are several on market examples such as sentiment lexicons and it could be coded as well such as corpora translation, noise detection, etc. (MonkeyLearn, 2022).

Sentiment analysis can be built via different programming languages such as Python, JavaScript, etc. In Python, there are several choices of tools such as Scikit-learn, NLTK, SpaCy, TensorFlow, Keras, PyTorch, etc. Meanwhile, in JavaScript, there are several choices of tools such as OpenNLP, Standford CoreNLP, Lingpipe, Weka, etc. (MonkeyLearn, 2022). For instance, Natural Language Toolkit (NLTK) is used in building the algorithm for analysis. NLTK is built distinctively for the works related to Natural Language Processing (NLP) on Python platform. It consists of over 50 corpora and lexical resources that comes along with a set of text processing libraries. These libraries help in the processes of classification, tokenization, stemming, tagging, parsing, and semantic reasoning on the inputs of the algorithm (NLTK, 2022).

Sentiment analysis defines its algorithms into three types such that rule-based, automation and hybrid. First and foremost, a rule-based algorithm performs the analysis based on pre-defined conditions of rules. For example, there are two polarities of positive and negative in the algorithm, when the number of positive polarities is higher than negative polarity, then returns a positive result else when the number of negative polarities is higher than positive polarity, then returns a negative result. However, most of the time, the result could not be taken into consideration as it solely based on the pre-defined rules and more rules will increase the complexity of the algorithm to be understood. However, a limitation of rule-based is it don't count a sentence as whole thus any complexity in NLP will causes errors in the results if the algorithm not well built. On the other hand, an automation algorithm performs the analysis based on training model of machine learning. An algorithm can be classified into supervised learning and unsupervised learning which the only different is that supervised learning give training data with label to the machine to learn on what will the output be of respective data while unsupervised learning has no label which the machine only learns about feature and do the classification by itself. In an automation algorithm, there will be several processes that makes the machine to learn in order to produce result such as training and prediction process which training let the machine to learn about an input with its corresponding output and prediction let the machine to assign an input to its corresponding output, feature extraction from text by using bag-of-word model, classification algorithms such as Naïve Bayes Classification, Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machines (SVM), Long Short Term Memory

(LSTM) etc. Last but not least, a hybrid system is a combination rule-based algorithm and automation algorithm (MonkeyLearn, 2022; Thematic, 2022).

There are two types of learning approaches may be applied to let a machine learn about the sentiment of individuals such as supervised learning and unsupervised learning, First and foremost, supervised learning is a learning approach which the input is labelled with output to let the machine learns on and classify data then predict outcomes accurately by yielding it. The algorithm will be in an iterative mode which it adjusts until the error minimized to the most possible state. Moreover, supervised learning has two different areas of classification and regression. A classification is when an algorithm is able to classify the input data into correct output category while a regression is when an algorithm is able to understand a relationship exist between dependent and independent variables thus making a conclusion on their relationship. For instance, supervised learning is often implemented with image and object recognition, spam detection, customer sentiment analysis, etc. (Education, 2020). On the other hand, unsupervised learning is a learning approach which the input is not labelled thus making the machine need to discover the patterns of the input to cluster it into right class. Moreover, unsupervised learning has two different areas of clustering and association. A clustering is when the machine is able to discover the inherent between raw and unclassified data thus clustering them in a class. The clustering may work in ways such as exclusive clustering where the stipulation of a data point can exist only in one cluster, overlapping clustering which data points may belong to multiple clusters, etc. Besides, an association is when the machine is able to discover the relationship of rules between data such as a large portion of data ruling the others. For instance, unsupervised learning is often implemented with news section, computer vision, anomaly detection, customer personas, etc. (Education, 2020)

In Natural Language Processing (NLP), there are several essential stages for the input to go through in order increase the effectiveness of the machine in resulting output. Firstly, lexical analysis, the initial stage in processing input where input is break into series of tokens by removing unnecessary blanks of the input such as by using Tokenization. Secondly, syntactic analysis, the stage of learning how words in the input are arranged to ensure it make sense in order as well as the grammatically correctness and conversion of root form such as by using stemming or lemmatization. Thirdly, semantic analysis, the stage of checking out the meaning of words from predefined dictionary and. Fourthly, discourse integration, the stage of analyzing the relationship between presence of immediate words and next words to check whether it means differently as paragraph in real word depends on the context. Fifthly, pragmatic analysis, the stage that derive an insight from the textual data (T, 2022).

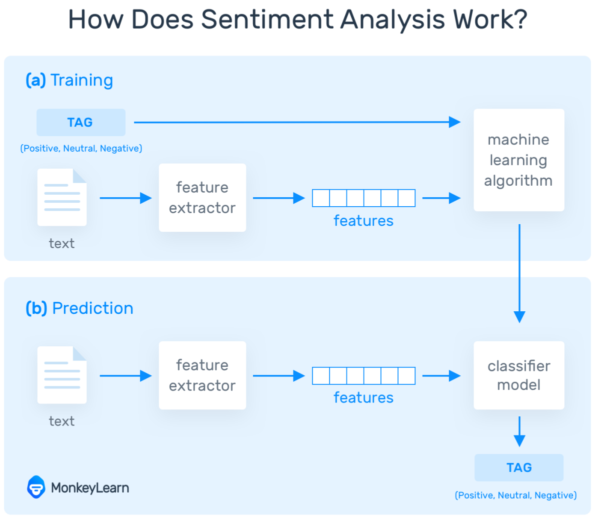

How Does Sentiment Analysis Work?
(a) Training
TAG
(Positive, Neutral, Negative)
text
feature extractor
features
machine learning algorithm
(b) Prediction
text
feature extractor
features
classifier model
TAG
(Positive, Neutral, Negative)
MonkeyLearn

*Figure 2.2 Automated Sentiment Analysis Supervised Approach (MonkeyLearn, 2022)*

As for supervised learning, there are several algorithms works within. Firstly, logistic Regression is classification in framing binary output of probability on 0<=x<=1 to predict the binary 0 or 1 or in the mode of true or else and yes or no. It calculates the linear output ten stashing over it to produce regression output which is either on the positive side or negative side but not any in the middle. Next, K-Nearest Neighbors (KNN) is a non-parametric classification that calculate distance between each data. It based on the data proximity and association to other data which it assumes that the data nearby with each other is considered as one. It usually uses the Euclidean distance to determine the distance between each other to classify them. Then, Naïve Bayes Classification is classification based on Bayes' Theorem that calculate probability. The presence of one data does not impact others in the probability which they have an effect on the result equally (Education, 2020). Moreover, Recurrent Neural Network (RNN) is one of deep learning algorithms for a supervised learning approach. In RNN, neurons are interconnected with each other such as how human brains interconnecting each nerve. The neurons store information and pass to each other as input for further analysis. However, RNN has shortage of vanishing gradient which occurs when the weights are updated through error calculation and back-propagation needed. Thus, another algorithm of Long Short Term Memory (LSTM) overcome the issue with RNN. LSTM has a memory cell at the top thus making the information from previous instance can be pass to the next one in an efficient manner as it can remember the information from previous one (T, 2022).

Most of the approaches in sentiment analysis works similarly aside from the algorithms applied. For instance, one of the Naïve Bayes Classification implemented works as, first and foremost, a model known as bag-of-words that is to simplify NLP representation is used. The model processes the input such as word, sentence, documents, etc. into a form of bag that store each word individually with a value attached. Next, inputs that act as training set and test set, is processed into a list of tokens in the form of [(feat, label)]. The feat is a feature dictionary such as True and label is the classification label such as Positive, Negative, etc. Then, the result of polarity on the classifier that evaluate its effectiveness may issue on several metrics such as accuracy, precision, recall, etc. and these metrics maybe affected by stopwords, low information features, etc. (Jacob, 2010). When inputs are given in a large format, the possibilities of the features for including numbers of stopwords and for being as low information features are high. Thus, elimination of stopwords and low information features helps in increasing the performances on the metrics of the classifier such as by removing noisy data that causes overfitting and the curse of dimensionality. A high information feature is determined by calculating information gain from each word such as comparing the frequency of the feature between classes. For example, a word that exists frequently in positive manner but infrequently in negative manner would be considered as high information feature (Jacob, 2010; Thematic, 2022).

## 2.2. Analysis of Tools

| **Tools comparison** | **Remark** | **Jupyter Notebook** | **Visual Studio Code (VS Code)** | **PyCharm** | **ATOM** | **Sublime Text** | **IDLE** |
|---|---|---|---|---|---|---|---|
| Type of license and open source license | State all types of license | Berkeley Software Distribution (BSD) | Open Source | ● Professional<br>● Community | Open Source | Proprietary | Python Software Foundation License |
| Year founded | When is this tool being introduced? | 2014 | 2015 | 2010 | 2014 | 2008 | 1998 |
| Founding company | Owner | Fernando Perez<br>Brian Granger | Microsoft Corporation | JetBrains | GitHub<br>Microsoft Corporation | Jon Skinner<br>Sublime HQ | Guido van Rossum |
| License Pricing | Compare the prices if the license is used for development and business/commercialization | ● Not Applicable | ● Not Applicable | ● US $9.90 per month for Professional Edition<br>● Free for Community Edition | ● Not Applicable | ● US $99 for three years of updates for personal<br>● US $65 per seat per year for business | ● Not Applicable |
| Supported features | What features that it offers? | ● Web-based interactive computational environment for creating browser-based REPL documents<br>● Integrate live code, equations, computational output, visualizations, and multimedia resources | ● Meet IntelliSense in syntax highlighting and autocomplete<br>● Debug code right from the editor by launching with break points, call stacks, and interactive console<br>● Git commands built-in<br>● Extensible and customizable | ● Intelligent python assistance<br>● Web development frameworks<br>● Scientific tools<br>● Cross-technology development<br>● Remote development capabilities<br>● Built-in developers' tools | ● Teletype<br>● GitHub<br>● Cross-Platform Editing<br>● Built-in Package Manager<br>● Smart Autocompletion<br>● File System Browser<br>● Multiple Panes<br>● Find and Replace<br>● Packages<br>● Themes<br>● Customization<br>● Under the Hood | ● GPU Rendering<br>● Apple Silicon and Linux ARM64<br>● Tab Multi Select<br>● Context-Aware Auto Complete<br>● Refreshed UI<br>● Typescript, JSX, TSX Support<br>● Superpowered syntax definitions<br>● Updated Python API | ● Multi-window text editor with syntax editing, autocompletion, and smart indent<br>● Cross-platform<br>● Python shell windows with colorizing of code input, output, and error message<br>● Debugger with persistent breakpoints, stepping, and viewing global and local namespaces<br>● Configurations, browsers, and other dialogs |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Common applications | In what areas this tool is usually used? | • Data Cleaning and Transformation<br>• Numerical Simulation<br>• Exploratory Data Analysis<br>• Data Visualization<br>• Statistical Modeling<br>• Machine Learning<br>• Deep Learning | • Develop computer programs<br>• Develop websites<br>• Develop web apps<br>• Develop web services<br>• Develop mobile apps | • Data Analysis<br>• Web Development<br>• DevOps<br>• Scrapers<br>• Software Testing<br>• Education Purpose | • Text and source code editor | • Text and source code editor | • Create desktop applications<br>• Create web applications |
| Customer support | How the customer support is given, e.g., proprietary, online community, etc. | Jupyter Discourse Forum | • Microsoft FAQ<br>• Developer Community | • Headquarters at Czech Republic<br>• Sales and Technical Support | • GitHub<br>• Twitter<br>• Discussions<br>• Stuff<br>• RSS Feed | • Technical Support Forum<br>• Sales FAQ<br>• Email at sales@sublimetext.com | • Python Community |
| Limitations | The drawbacks of the software | • No drag and drop functionality<br>• Each notebook assigned with own kernel<br>• No single window view for overall document | • No drag and drop functionality<br>• Unlimited choices of languages causing difficulty in managing plugins | • High memory usage thus degrading functionality of code<br>• No other programming languages than Python for community version | • Easy crash due to non-core package misbehaving<br>• No remote pairing<br>• No non-GUI environment<br>• No block selection for easy navigation | • Annoying pop up for free edition<br>• Need to run code by using another software | • File cannot be accessed another device unless copied<br>• not as advanced as its contemporaries<br>• changes are not saved automatically |

## 2.3. Justification of Selected Tool

As a beginner in Python development, Jupyter Notebook is suitable to be used as it is the easiest tool implemented for a simple and direct understanding delivery to the user through streamlined and document-centric approach. It is a web-based interactive computational environment for creating notebook documents. Besides, Jupyter Notebook is a free open source tool. Next, it is integrated with the Anaconda environment which is a platform with thousands of open-source packages and libraries for Python developers to develop in a convenient environment.

On the other hand, as Jupyter Notebook is document-centric approach, it offers environment where it is easier to open any other relevant documents at the same time for reference while developing it without the need to open it elsewhere. Moreover, with streamlined approach, it offers environment in running codes line by line and one by one which as if the line is incorrect, it will show up errors as a result and the relevant code behind would not be able to run if the errors not solved while the errors are precise as well which will be able to be solved within time instead of those that needs to figure out the errors without explanation given. Thus, each section may be act as code, text of Markdown, mathematics, plots, and rich media.

.

# 3. Methodology

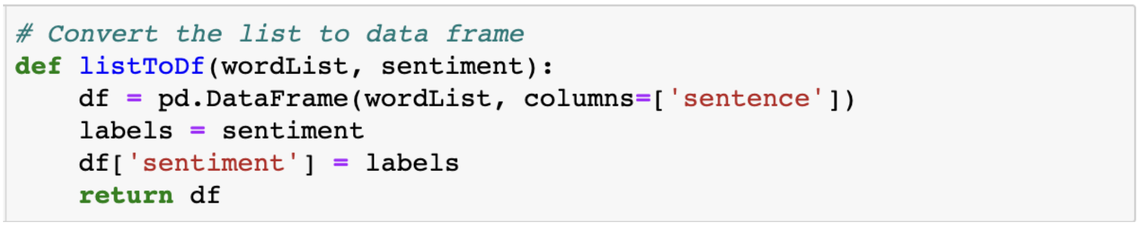
## 3.1. Description of Dataset

There are several datasets included in the system as shown below:

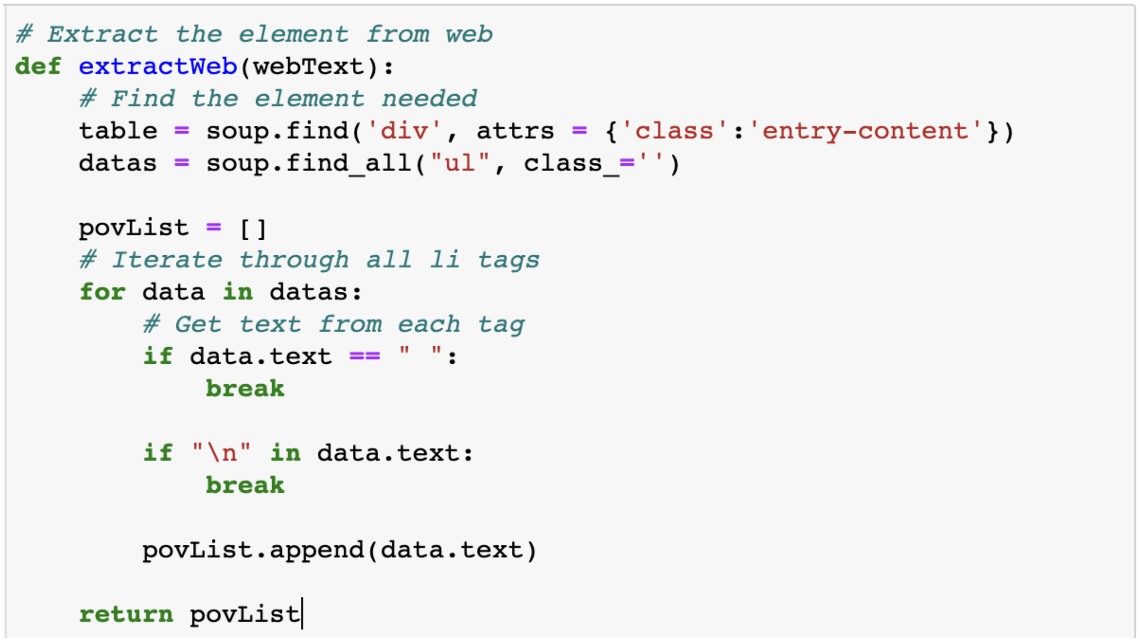
| **Data Structures / Data Dictionary** |
|---|
| 150 Positive Sentences in English<br>Source:https://www.wordscoach.com/blog/examples-of-positive-sentences/ |
| 100 Negative Sentences in English<br>Source:https://www.wordscoach.com/blog/examples-of-negative-sentences/ |
| 50, 000 Positive and Negative Movie Reviews<br>Source:https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews<br>Remark: Spitted into 5 CSV files; Uploaded to GitHub; Accessed through GitHub Raw URL |
| 2, 000 Positive Words in English<br>Source: https://ptrckprry.com/course/ssd/data/positive-words.txt |
| 4, 700 Negative Words in English<br>Source: https://ptrckprry.com/course/ssd/data/negative-words.txt |

### 3.1.1. Web Crawling – Chow Weng Yong_Ainisha Ting

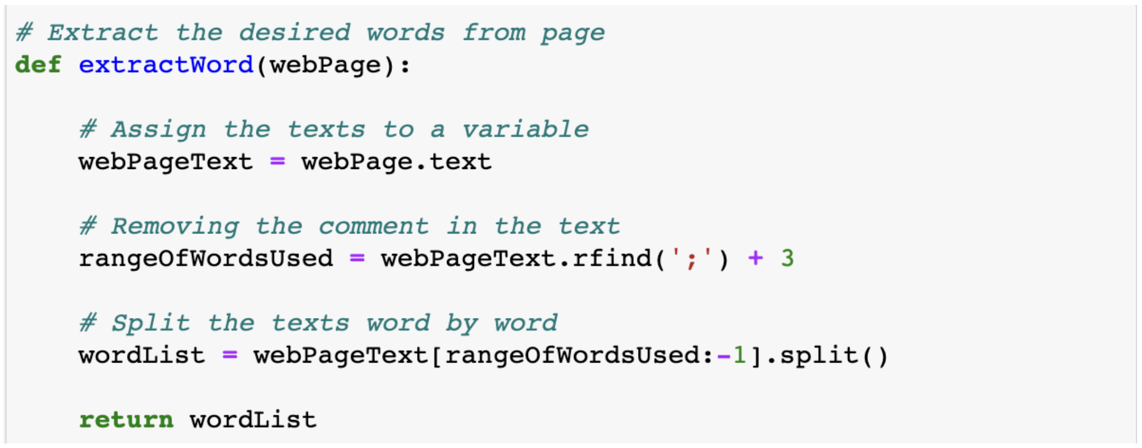
**Web Crawling**



```python
# Convert the list to data frame
def listToDf(wordList, sentiment):
    df = pd.DataFrame(wordList, columns=['sentence'])
    labels = sentiment
    df['sentiment'] = labels
    return df
```



```python
# Extract the element from web
def extractWeb(webText):
    # Find the element needed
    table = soup.find('div', attrs = {'class':'entry-content'})
    datas = soup.find_all("ul", class_='')

    povList = []
    # Iterate through all li tags
    for data in datas:
        # Get text from each tag
        if data.text == " ":
            break

        if "\n" in data.text:
            break

        povList.append(data.text)

    return povList
```



```python
# Extract the desired words from page
def extractWord(webPage):

    # Assign the texts to a variable
    webPageText = webPage.text

    # Removing the comment in the text
    rangeOfWordsUsed = webPageText.rfind(';') + 3

    # Split the texts word by word
    wordList = webPageText[rangeOfWordsUsed:-1].split()

    return wordList
```

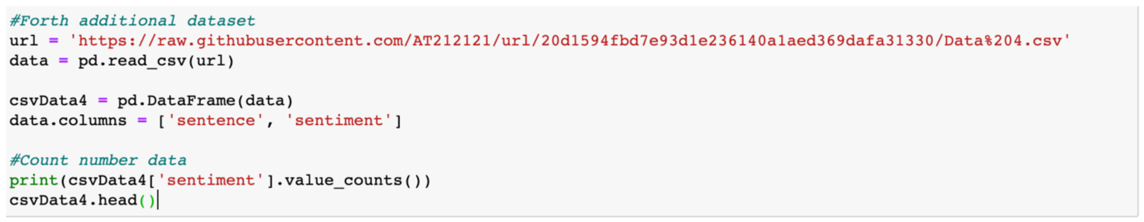
```python
# Get access to the website
url = "https://www.wordscoach.com/blog/examples-of-positive-sentences/"
resp = requests.get(url)
soup = BeautifulSoup(resp.text, 'html.parser')

# Check if the website is reachable
if resp.status_code == 200:
    povList = extractWeb(soup)
    povDf = listToDf(povList, 'positive')
    print('Sentence is Crawled. ')
else:
    print('The Website is Unreachable !')
```

```python
# Get access to the website
url = "https://www.wordscoach.com/blog/examples-of-negative-sentences/"
resp = requests.get(url)
soup = BeautifulSoup(resp.text, 'html.parser')
# Check if the website is reachable
if resp.status_code == 200:
    negList = extractWeb(soup)
    negDf = listToDf(negList, 'negative')
    print('Sentence is Crawled. ')
else:
    print('The Website is Unreachable !')
```

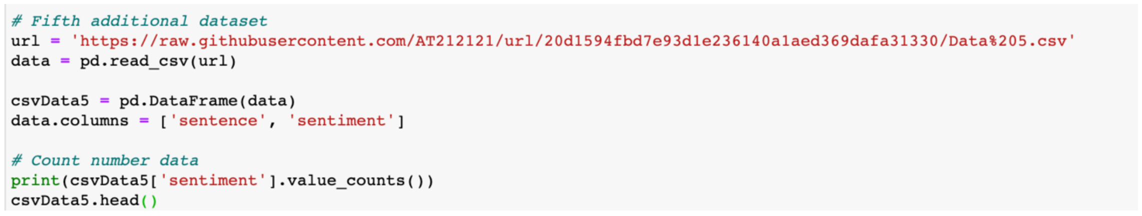
```python
# First additional dataset
url = 'https://raw.githubusercontent.com/AT212121/url/20d1594fbd7e93d1e236140a1aed369dafa31330/Data%201.csv'
data = pd.read_csv(url)

csvData = pd.DataFrame(data)
data.columns = ['sentence', 'sentiment']

# Count number data
print(csvData['sentiment'].value_counts())
csvData1.head()
```

```python
# Second additional dataset
url = 'https://raw.githubusercontent.com/AT212121/url/20d1594fbd7e93d1e236140a1aed369dafa31330/Data%202.csv'
data = pd.read_csv(url)

csvData2 = pd.DataFrame(data)
data.columns = ['sentence', 'sentiment']

# Count number data
print(csvData2['sentiment'].value_counts())
csvData2.head()
```

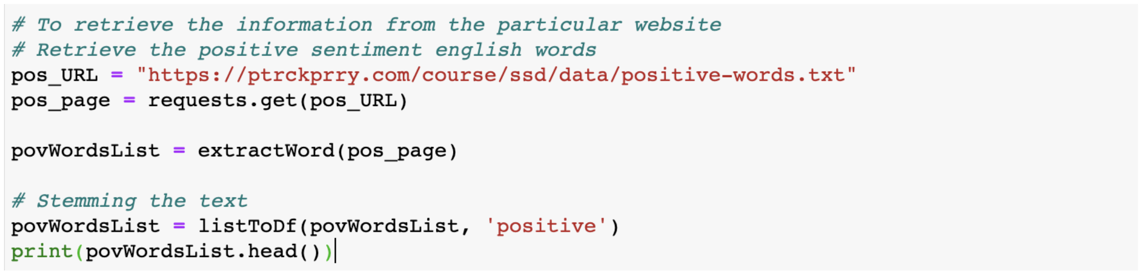
```python
# Third additional dataset
url = 'https://raw.githubusercontent.com/AT212121/url/20d1594fbd7e93d1e236140a1aed369dafa31330/Data%203.csv'
data = pd.read_csv(url)

csvData3 = pd.DataFrame(data)
data.columns = ['sentence', 'sentiment']

# Count number data
print(csvData3['sentiment'].value_counts())
csvData3.head()
```



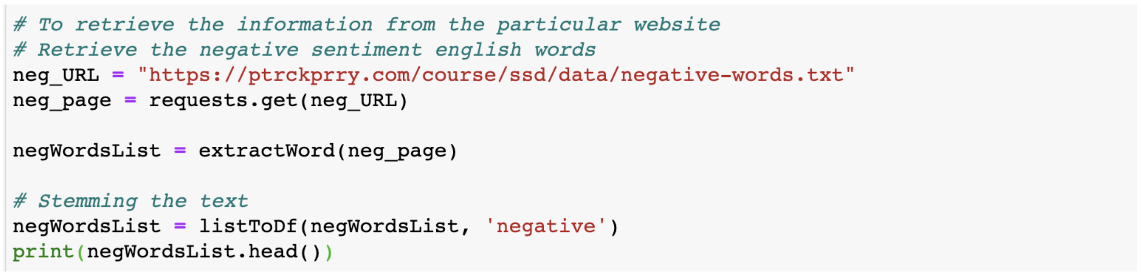
```python
#Forth additional dataset
url = 'https://raw.githubusercontent.com/AT212121/url/20d1594fbd7e93d1e236140a1aed369dafa31330/Data%204.csv'
data = pd.read_csv(url)

csvData4 = pd.DataFrame(data)
data.columns = ['sentence', 'sentiment']

#Count number data
print(csvData4['sentiment'].value_counts())
csvData4.head()
```



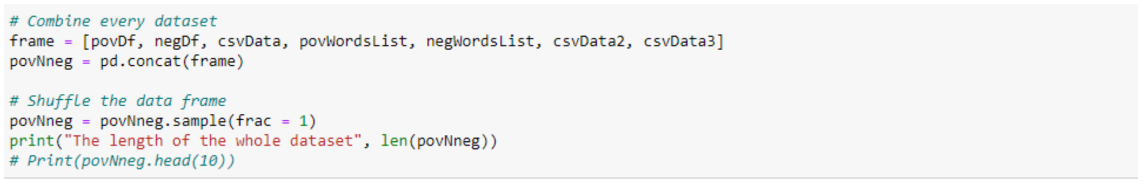
```python
# Fifth additional dataset
url = 'https://raw.githubusercontent.com/AT212121/url/20d1594fbd7e93d1e236140a1aed369dafa31330/Data%205.csv'
data = pd.read_csv(url)

csvData5 = pd.DataFrame(data)
data.columns = ['sentence', 'sentiment']

# Count number data
print(csvData5['sentiment'].value_counts())
csvData5.head()
```



```python
# To retrieve the information from the particular website
# Retrieve the positive sentiment english words
pos_URL = "https://ptrckprry.com/course/ssd/data/positive-words.txt"
pos_page = requests.get(pos_URL)

povWordsList = extractWord(pos_page)

# Stemming the text
povWordsList = listToDf(povWordsList, 'positive')
print(povWordsList.head())
```



```python
# To retrieve the information from the particular website
# Retrieve the negative sentiment english words
neg_URL = "https://ptrckprry.com/course/ssd/data/negative-words.txt"
neg_page = requests.get(neg_URL)

negWordsList = extractWord(neg_page)

# Stemming the text
negWordsList = listToDf(negWordsList, 'negative')
print(negWordsList.head())
```



```python
# Combine every dataset
frame = [povDf, negDf, csvData, povWordsList, negWordsList, csvData2, csvData3]
povNneg = pd.concat(frame)

# Shuffle the data frame
povNneg = povNneg.sample(frac = 1)
print("The length of the whole dataset", len(povNneg))
# Print(povNneg.head(10))
```

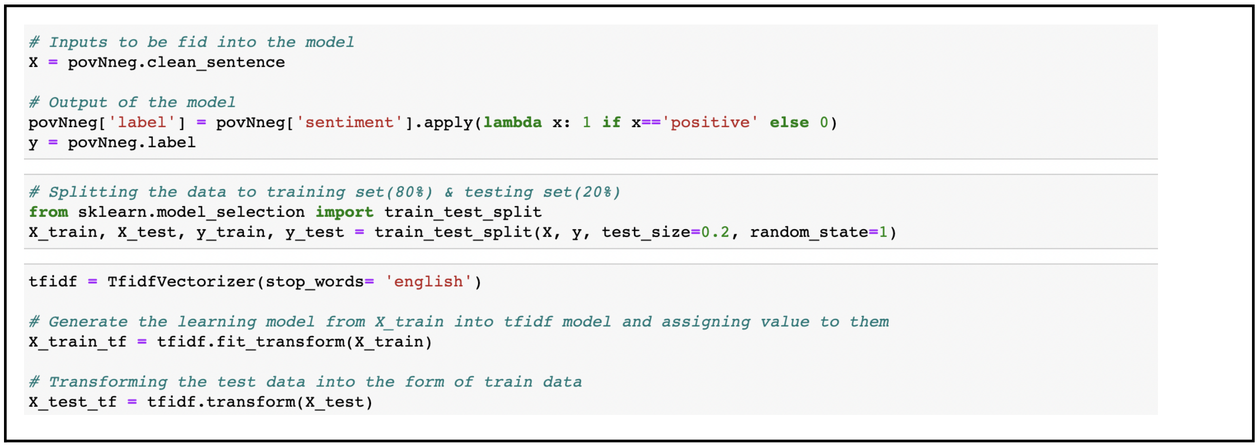
## 3.1.2. Data Preprocessing – Chow Weng Yong

**Data Preprocessing**

```
# Function for removing punctuation
def remove_punctuation(text):
    punctuationfree = "".join([i for i in text if i not in string.punctuation])
    return punctuationfree

# Function for removing numbers
def remove_number(text):
    numberFree = ''.join(i for i in text if not i.isdigit())
    return numberFree

# Function for removing stopwords from tokenized columns
def remove_stopwords(text):
    # Stop words we using
    stopwords = nltk.corpus.stopwords.words('english')
    new_Stop = 'br'
    stopwords.extend(new_Stop)
    output= [i for i in text if i not in stopwords]
    return output

# Function for lemmatization
def lemmatizerFunction(text):
    # Object for Lemmatization
    lemmatizer = WordNetLemmatizer()

    lemm_text = [lemmatizer.lemmatize(word) for word in text]
    return lemm_text

# Function for tokenization
def tokenization(text):
    tokens = nltk.word_tokenize(text)
    return tokens
```

```
# Storing the sentence without punctuation in another column (clean_sentence)
povNneg['clean_sentence']= povNneg['sentence'].apply(lambda x:remove_punctuation(x))

# Removing numbers from clean_sentence
povNneg['clean_sentence']= povNneg['clean_sentence'].apply(lambda x:remove_number(x))

# Convert all the element in (clean_sentence) to lower case
povNneg['clean_sentence'] = povNneg['clean_sentence'].apply(lambda x: x.lower())

# Storing the tokenized clean_sentence into another column(tokenized)
povNneg['tokenized'] = povNneg['clean_sentence'].apply(lambda x: tokenization(x))

# Removing the stop words from tokenized columns
povNneg['tokenized'] = povNneg['tokenized'].apply(lambda x:remove_stopwords(x))

# Lemmatizing texts
povNneg['tokenized'] = povNneg['tokenized'].apply(lambda x:lemmatizerFunction(x))
```

```
povNneg['clean_sentence'] = povNneg['tokenized'].apply(lambda x:" ".join(x))
```



```python
# Inputs to be fid into the model
X = povNneg.clean_sentence

# Output of the model
povNneg['label'] = povNneg['sentiment'].apply(lambda x: 1 if x=='positive' else 0)
y = povNneg.label
```

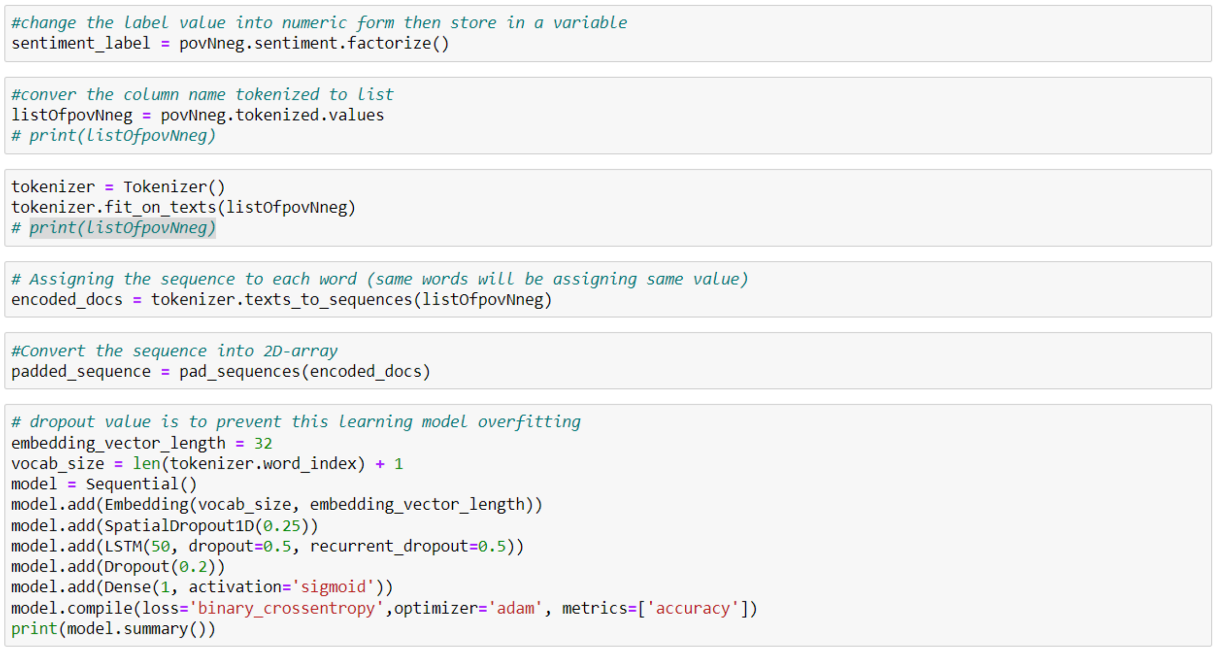
```python
# Splitting the data to training set(80%) & testing set(20%)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)
```

```python
tfidf = TfidfVectorizer(stop_words= 'english')

# Generate the learning model from X_train into tfidf model and assigning value to them
X_train_tf = tfidf.fit_transform(X_train)

# Transforming the test data into the form of train data
X_test_tf = tfidf.transform(X_test)
```

## 3.2. Applications of the Algorithms

**Algorithms**

Long Short Term Memory (LSTM) – Chow Weng Yong



```
#change the label value into numeric form then store in a variable
sentiment_label = povNneg.sentiment.factorize()
```

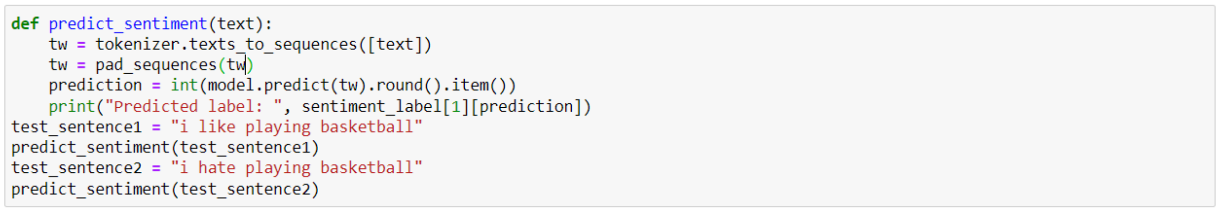
```
#conver the column name tokenized to list
listOfpovNneg = povNneg.tokenized.values
# print(listOfpovNneg)
```

```
tokenizer = Tokenizer()
tokenizer.fit_on_texts(listOfpovNneg)
# print(listOfpovNneg)
```

```
# Assigning the sequence to each word (same words will be assigning same value)
encoded_docs = tokenizer.texts_to_sequences(listOfpovNneg)
```

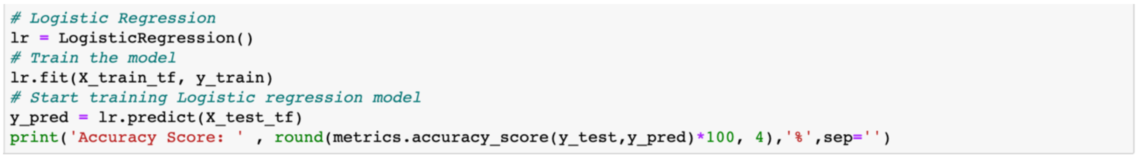
```
#Convert the sequence into 2D-array
padded_sequence = pad_sequences(encoded_docs)
```

```
# dropout value is to prevent this learning model overfitting
embedding_vector_length = 32
vocab_size = len(tokenizer.word_index) + 1
model = Sequential()
model.add(Embedding(vocab_size, embedding_vector_length))
model.add(SpatialDropout1D(0.25))
model.add(LSTM(50, dropout=0.5, recurrent_dropout=0.5))
model.add(Dropout(0.2))
model.add(Dense(1, activation='sigmoid'))
model.compile(loss='binary_crossentropy',optimizer='adam', metrics=['accuracy'])
print(model.summary())
```

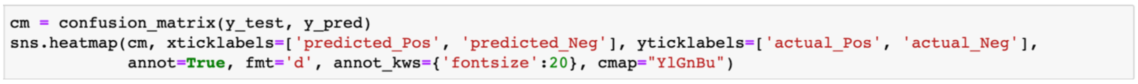


```
def predict_sentiment(text):
    tw = tokenizer.texts_to_sequences([text])
    tw = pad_sequences(tw)
    prediction = int(model.predict(tw).round().item())
    print("Predicted label: ", sentiment_label[1][prediction])
test_sentence1 = "i like playing basketball"
predict_sentiment(test_sentence1)
test_sentence2 = "i hate playing basketball"
predict_sentiment(test_sentence2)
```

## Logistic Regression – Ainisha Ting

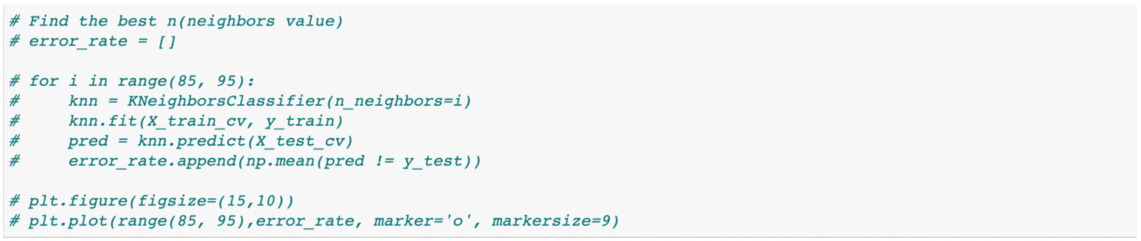


```
# Logistic Regression
lr = LogisticRegression()
# Train the model
lr.fit(X_train_tf, y_train)
# Start training Logistic regression model
y_pred = lr.predict(X_test_tf)
print('Accuracy Score: ' , round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

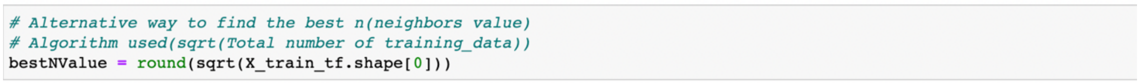


```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

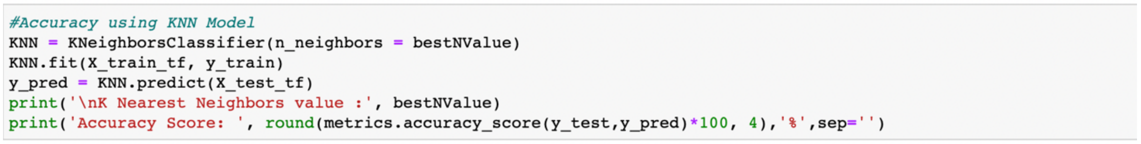
## K-Nearest Neighbors (KNN) – Chow Weng Yong



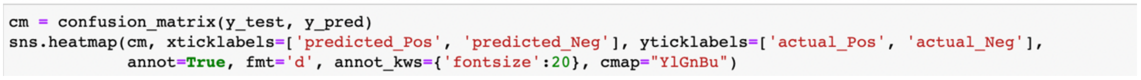
```
# Find the best n(neighbors value)
# error_rate = []

# for i in range(85, 95):
#     knn = KNeighborsClassifier(n_neighbors=i)
#     knn.fit(X_train_cv, y_train)
#     pred = knn.predict(X_test_cv)
#     error_rate.append(np.mean(pred != y_test))

# plt.figure(figsize=(15,10))
# plt.plot(range(85, 95),error_rate, marker='o', markersize=9)
```



```
# Alternative way to find the best n(neighbors value)
# Algorithm used(sqrt(Total number of training_data))
bestNValue = round(sqrt(X_train_tf.shape[0]))
```

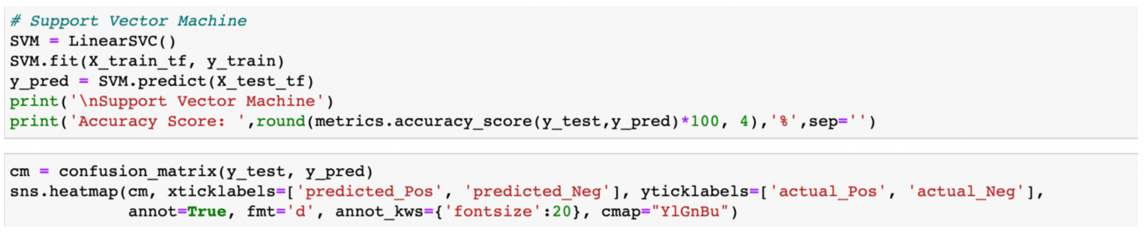


```
#Accuracy using KNN Model
KNN = KNeighborsClassifier(n_neighbors = bestNValue)
KNN.fit(X_train_tf, y_train)
y_pred = KNN.predict(X_test_tf)
print('\nK Nearest Neighbors value :', bestNValue)
print('Accuracy Score: ', round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```



```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

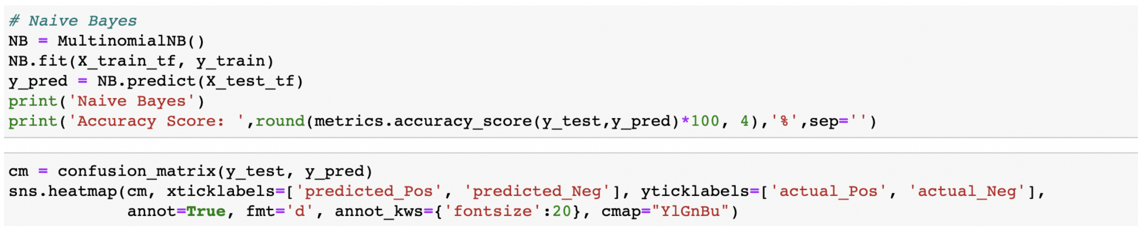
## Support Vector Machine (SVM) – Ainisha Ting



```
# Support Vector Machine
SVM = LinearSVC()
SVM.fit(X_train_tf, y_train)
y_pred = SVM.predict(X_test_tf)
print('\nSupport Vector Machine')
print('Accuracy Score: ',round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

## Naïve Bayes – Ainisha Ting



```
# Naive Bayes
NB = MultinomialNB()
NB.fit(X_train_tf, y_train)
y_pred = NB.predict(X_test_tf)
print('Naive Bayes')
print('Accuracy Score: ',round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

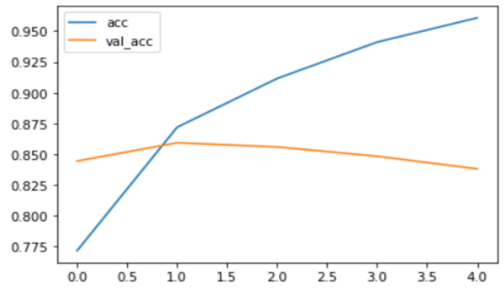
## Results

### Long Short Term Memory (LSTM) – Chow Weng Yong

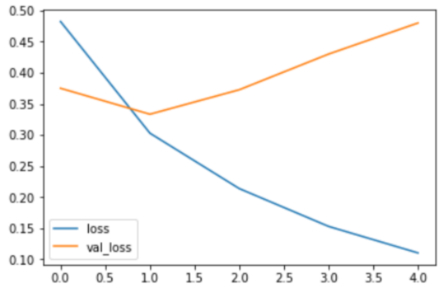
```python
#validation_split is splitting the data to use 20% for validation
history = model.fit(padded_sequence,sentiment_label[0],validation_split=0.2, epochs=5, batch_size=32)
```

```
Epoch 1/5
927/927 [==============================] - 789s 846ms/step - loss: 0.4824 - accuracy: 0.7716 - val_loss: 0.3750 - val_accuracy:
0.8444
Epoch 2/5
927/927 [==============================] - 780s 841ms/step - loss: 0.3028 - accuracy: 0.8719 - val_loss: 0.3333 - val_accuracy:
0.8593
Epoch 3/5
927/927 [==============================] - 783s 845ms/step - loss: 0.2136 - accuracy: 0.9115 - val_loss: 0.3727 - val_accuracy:
0.8559
Epoch 4/5
927/927 [==============================] - 719s 775ms/step - loss: 0.1528 - accuracy: 0.9411 - val_loss: 0.4301 - val_accuracy:
0.8484
Epoch 5/5
927/927 [==============================] - 777s 838ms/step - loss: 0.1101 - accuracy: 0.9608 - val_loss: 0.4801 - val_accuracy:
0.8381
```

```python
plt.plot(history.history['accuracy'], label='acc')
plt.plot(history.history['val_accuracy'], label='val_acc')
plt.legend()
plt.show()
```



```python
plt.plot(history.history['loss'], label='loss')
plt.plot(history.history['val_loss'], label='val_loss')
plt.legend()
plt.show()
```



```python
def predict_sentiment(text):
    tw = tokenizer.texts_to_sequences([text])
    tw = pad_sequences(tw,maxlen=200)
    prediction = int(model.predict(tw).round().item())
    print("Predicted label: ", sentiment_label[1][prediction])
test_sentence1 = "I Love Basketball"
predict_sentiment(test_sentence1)
test_sentence2 = "I Hate Basketball"
predict_sentiment(test_sentence2)
```

```
1/1 [==============================] - 1s 687ms/step
Predicted label:  negative
1/1 [==============================] - 0s 47ms/step
Predicted label:  negative
```

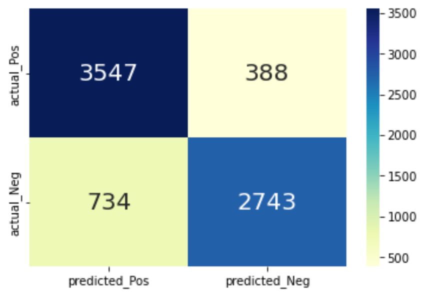
## Logistic Regression – Ainisha Ting

```
# Logistic Regression
lr = LogisticRegression()
# Train the model
lr.fit(X_train_tf, y_train)
# Start training Logistic regression model
y_pred = lr.predict(X_test_tf)
print('Accuracy Score: ' , round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
Accuracy Score: 84.8624%
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

```
<AxesSubplot:>
```



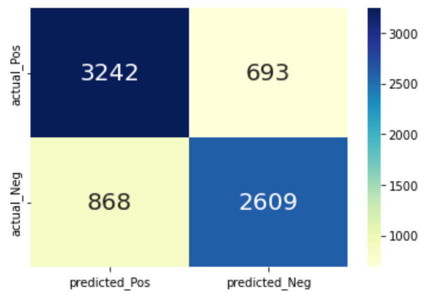
## K-Nearest Neighbors (KNN) – Chow Weng Yong

```
#Accuracy using KNN Model
KNN = KNeighborsClassifier(n_neighbors = bestNValue)
KNN.fit(X_train_tf, y_train)
y_pred = KNN.predict(X_test_tf)
print('K Nearest Neighbors value :', bestNValue)
print('Accuracy Score: ', round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
K Nearest Neighbors value : 172
Accuracy Score: 78.9396%
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

```
<AxesSubplot:>
```



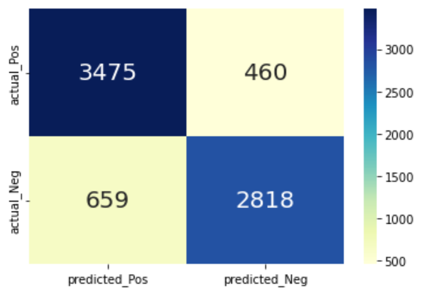
## Support Vector Machine (SVM) – Ainisha Ting

```
# Support Vector Machine
SVM = LinearSVC()
SVM.fit(X_train_tf, y_train)
y_pred = SVM.predict(X_test_tf)
print('Support Vector Machine')
print('Accuracy Score: ',round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
Support Vector Machine
Accuracy Score: 84.9029%
```

```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

```
<AxesSubplot:>
```



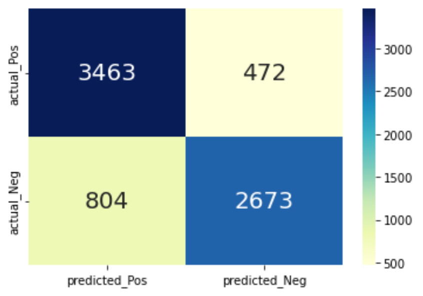
## Naïve Bayes – Ainisha Ting

```
# Naive Bayes
NB = MultinomialNB()
NB.fit(X_train_tf, y_train)
y_pred = NB.predict(X_test_tf)
print('Naive Bayes')
print('Accuracy Score: ',round(metrics.accuracy_score(y_test,y_pred)*100, 4),'%',sep='')
```

```
Naive Bayes
Accuracy Score: 82.7847%
```

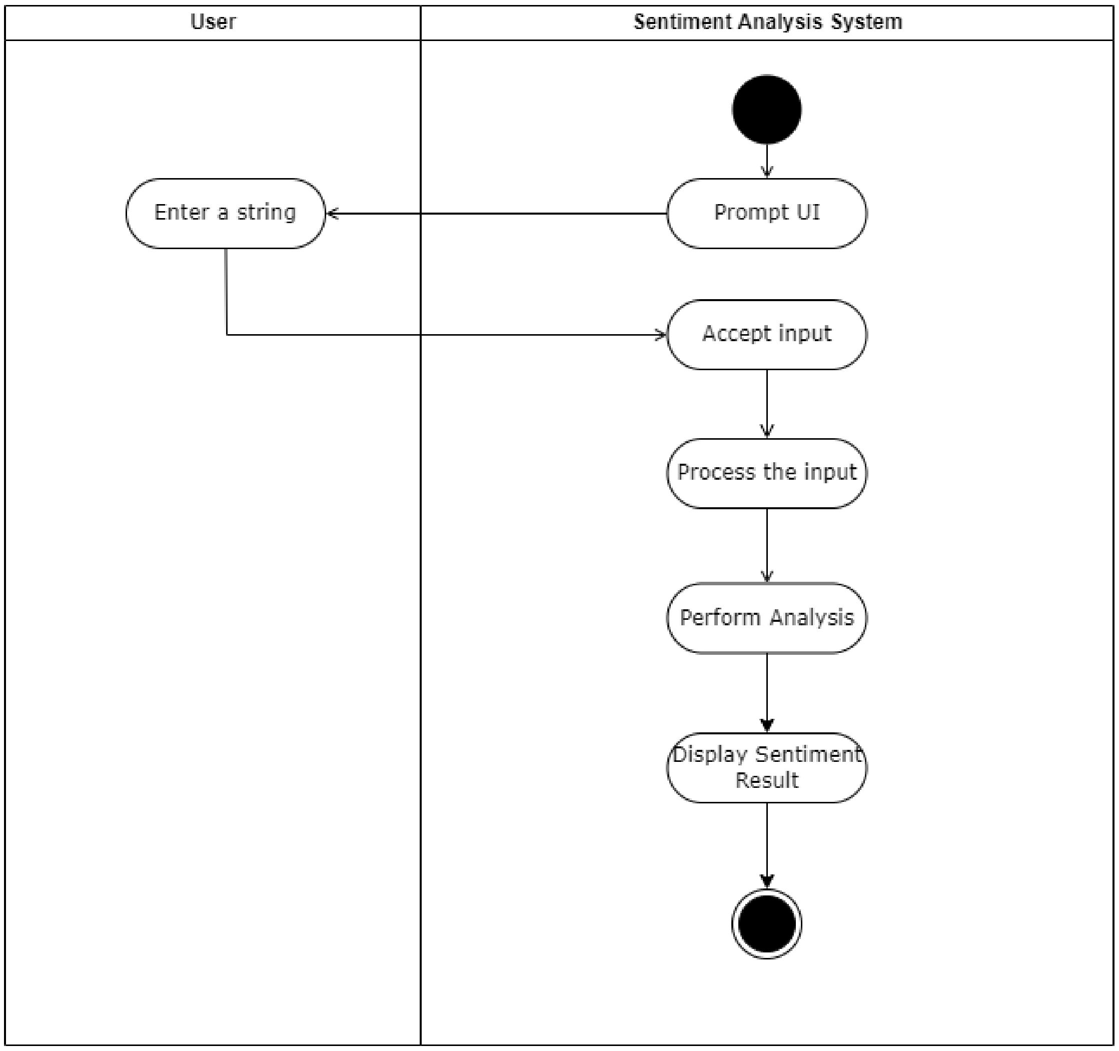
```
cm = confusion_matrix(y_test, y_pred)
sns.heatmap(cm, xticklabels=['predicted_Pos', 'predicted_Neg'], yticklabels=['actual_Pos', 'actual_Neg'],
            annot=True, fmt='d', annot_kws={'fontsize':20}, cmap="YlGnBu")
```

```
<AxesSubplot:>
```



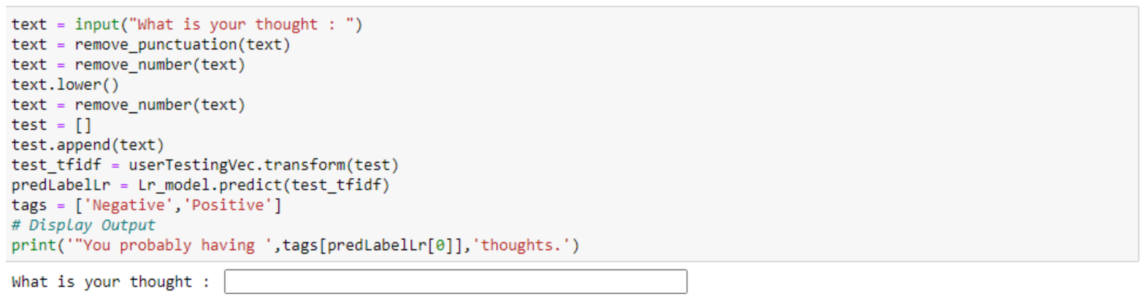
## 3.3. System Flowchart

### 3.3.1. Activity diagram



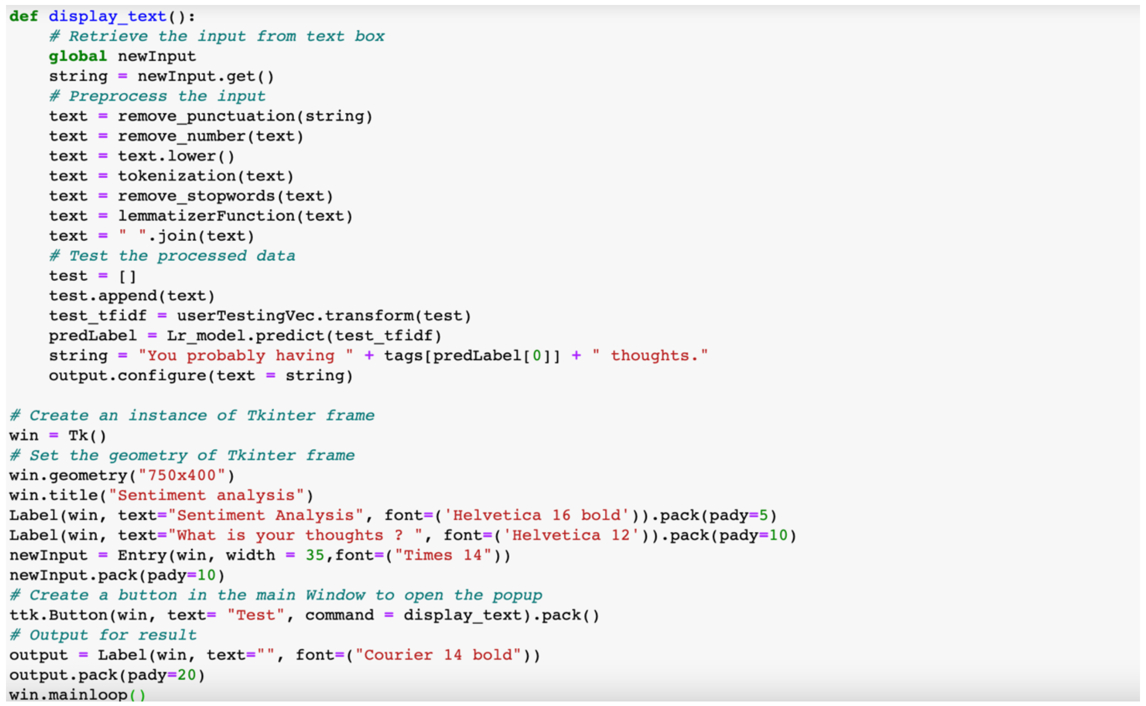
### 3.3.2. User Interface

**User Interface**

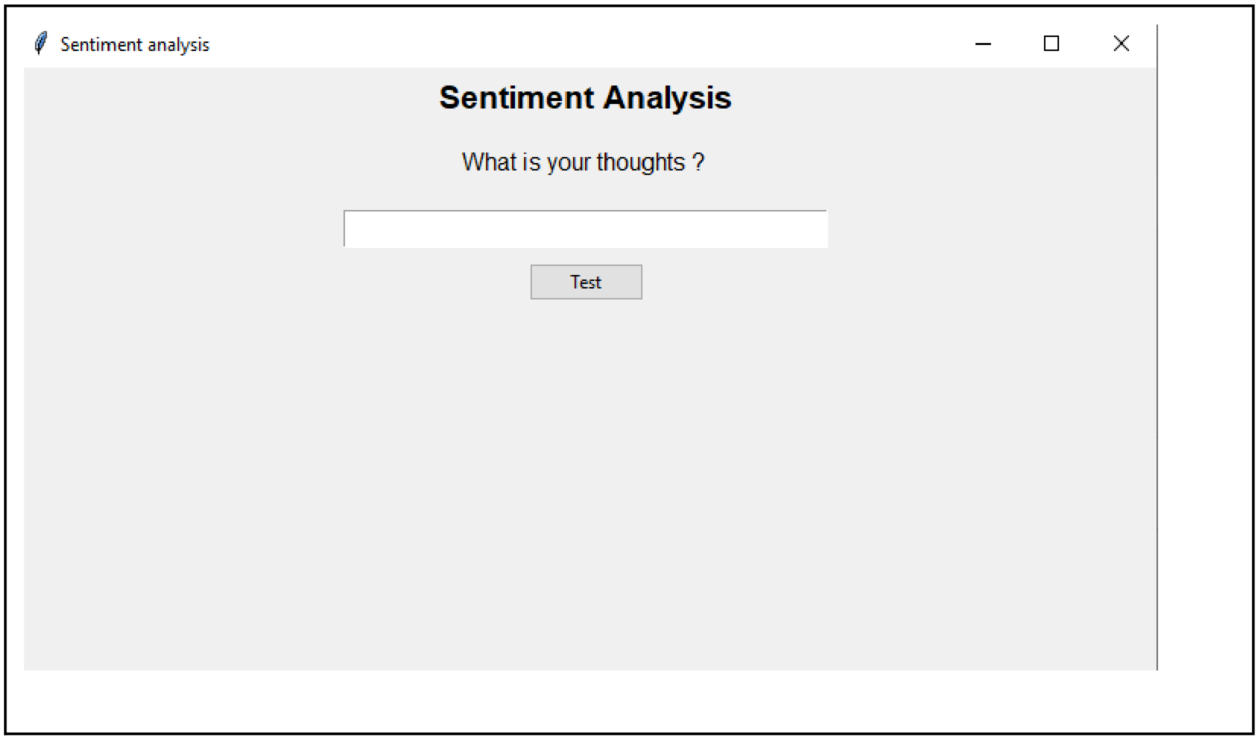


```
text = input("What is your thought : ")
text = remove_punctuation(text)
text = remove_number(text)
text.lower()
text = remove_number(text)
test = []
test.append(text)
test_tfidf = userTestingVec.transform(test)
predLabelLr = Lr_model.predict(test_tfidf)
tags = ['Negative','Positive']
# Display Output
print('"You probably having ',tags[predLabelLr[0]],'thoughts.')
```

```
What is your thought :
```



```
def display_text():
    # Retrieve the input from text box
    global newInput
    string = newInput.get()
    # Preprocess the input
    text = remove_punctuation(string)
    text = remove_number(text)
    text = text.lower()
    text = tokenization(text)
    text = remove_stopwords(text)
    text = lemmatizerFunction(text)
    text = " ".join(text)
    # Test the processed data
    test = []
    test.append(text)
    test_tfidf = userTestingVec.transform(test)
    predLabel = Lr_model.predict(test_tfidf)
    string = "You probably having " + tags[predLabel[0]] + " thoughts."
    output.configure(text = string)

# Create an instance of Tkinter frame
win = Tk()
# Set the geometry of Tkinter frame
win.geometry("750x400")
win.title("Sentiment analysis")
Label(win, text="Sentiment Analysis", font=('Helvetica 16 bold')).pack(pady=5)
Label(win, text="What is your thoughts ? ", font=('Helvetica 12')).pack(pady=10)
newInput = Entry(win, width = 35,font=("Times 14"))
newInput.pack(pady=10)
# Create a button in the main Window to open the popup
ttk.Button(win, text= "Test", command = display_text).pack()
# Output for result
output = Label(win, text="", font=("Courier 14 bold"))
output.pack(pady=20)
win.mainloop()
```


Sentiment analysis

Sentiment Analysis

What is your thoughts ?

## 3.4. Proposed Test Plan / Hypothesis

| **Test Plan** |
|---|
| Input: (From Dataset 5)<br><br>Hickory Dickory Dock was a good Poirot mystery. I confess I have not read the book, despite being an avid Agatha Christie fan. The adaptation isn't without its problems, there were times when the humour, and there were valiant attempts to get it right, was a little overdone, and the events leading up to the final solution were rather rushed. I also thought there were some slow moments so some of the mystery felt padded. However, I loved how Hickory Dickory Dock was filmed, it had a very similar visual style to the brilliant ABC Murders, and it really set the atmosphere, what with the dark camera work and dark lighting. The darker moments were somewhat creepy, this was helped by one of the most haunting music scores in a Poirot adaptation, maybe not as disturbing as the one in One Two Buckle My Shoe, which gave me nightmares. The plot is complex, with all the essential ingredients, though not as convoluted as Buckle My Shoe,and in some way that is a good thing. The acting was very good, David Suchet is impeccable(I know I can't use this word forever but I can't think of a better word to describe his performance in the series) as Poirot, and Phillip Jackson and Pauline Moran do justice to their integral characters brilliantly. And the students had great personalities and well developed on the whole, particularly Damian Lewis as Leonard. All in all, solid mystery but doesn't rank along the best. 7.5/10 Bethany Cox |
| Output:<br><br>You probably have positive thoughts. |

**Test Plan**

Input: (From Dataset 5)

I have to confess that I am severely disappointed.<br /><br />This version can in no way compete with the version of 1995. The reason why I watched it was that I wasn't entirely happy with Ciaran Hinds as Captain Wentworth and thought that Rupert Penry-Jones looked much more like the Captain I had imagined when I read the book. And he was too.<br /><br />Unfortunately that is the only redeeming quality of the film. The rest is as un-Austen-like as possible.<br /><br />Miss Elliot would NEVER have run through the streets of Bath like this. It wasn't in her character and it just wasn't done by a lady of the those times. The Anne Elliot of the book was a lady and she had dignity. There are other painful anachronisms but this was the worst.<br /><br />Although there are 3 important quotes from the book, they are at entirely inappropriate moments, warning those who know the book that yet another important part of the book will either be missing or completely changed.<br /><br />And although this version is not much shorter than the other one, it feels like everything is rushed. Very little care was taken to introduce the characters, show their dispositions and motives. Important scenes were omitted. How could they possibly have butchered the final scenes in this way ? A disaster ! And it was by far not as beautifully photographed as the other one.<br /><br />No, no, no. If you love Austen, then don't waste your time with this.

Output:

You probably have negative thoughts.

# 4. Result

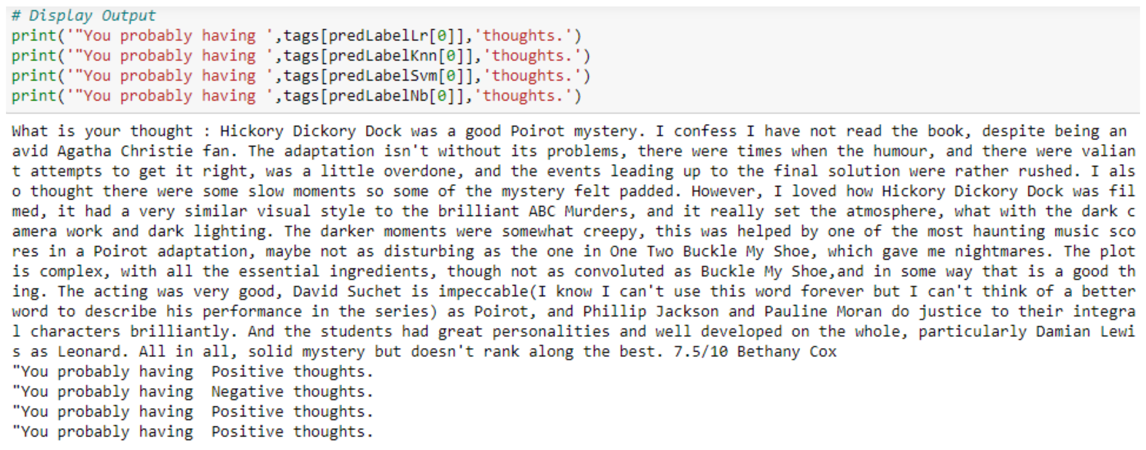
## 4.1. Results

**Test Result**

Input: (From Dataset 5)

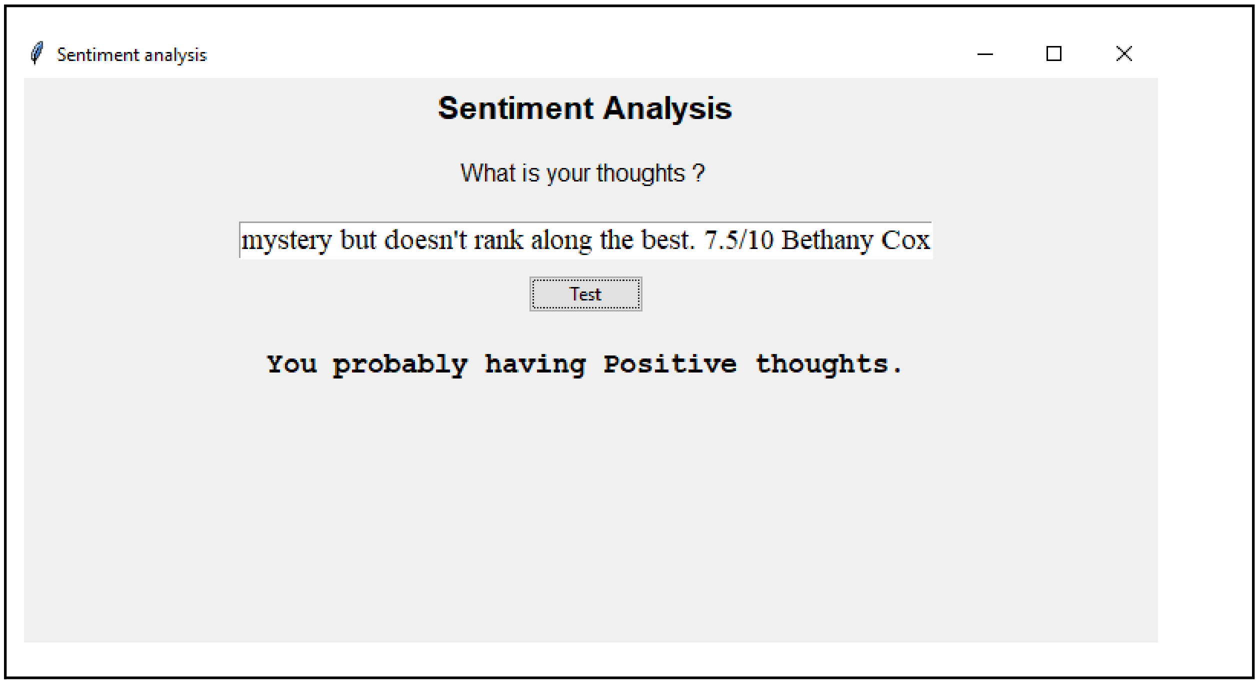
Hickory Dickory Dock was a good Poirot mystery. I confess I have not read the book, despite being an avid Agatha Christie fan. The adaptation isn't without its problems, there were times when the humour, and there were valiant attempts to get it right, was a little overdone, and the events leading up to the final solution were rather rushed. I also thought there were some slow moments so some of the mystery felt padded. However, I loved how Hickory Dickory Dock was filmed, it had a very similar visual style to the brilliant ABC Murders, and it really set the atmosphere, what with the dark camera work and dark lighting. The darker moments were somewhat creepy, this was helped by one of the most haunting music scores in a Poirot adaptation, maybe not as disturbing as the one in One Two Buckle My Shoe, which gave me nightmares. The plot is complex, with all the essential ingredients, though not as convoluted as Buckle My Shoe,and in some way that is a good thing. The acting was very good, David Suchet is impeccable(I know I can't use this word forever but I can't think of a better word to describe his performance in the series) as Poirot, and Phillip Jackson and Pauline Moran do justice to their integral characters brilliantly. And the students had great personalities and well developed on the whole, particularly Damian Lewis as Leonard. All in all, solid mystery but doesn't rank along the best. 7.5/10 Bethany Cox

Output:



```
# Display Output
print('"You probably having ',tags[predLabelLr[0]],'thoughts.')
print('"You probably having ',tags[predLabelKnn[0]],'thoughts.')
print('"You probably having ',tags[predLabelSvm[0]],'thoughts.')
print('"You probably having ',tags[predLabelNb[0]],'thoughts.')
```

```
What is your thought : Hickory Dickory Dock was a good Poirot mystery. I confess I have not read the book, despite being an
avid Agatha Christie fan. The adaptation isn't without its problems, there were times when the humour, and there were valian
t attempts to get it right, was a little overdone, and the events leading up to the final solution were rather rushed. I als
o thought there were some slow moments so some of the mystery felt padded. However, I loved how Hickory Dickory Dock was fil
med, it had a very similar visual style to the brilliant ABC Murders, and it really set the atmosphere, what with the dark c
amera work and dark lighting. The darker moments were somewhat creepy, this was helped by one of the most haunting music sco
res in a Poirot adaptation, maybe not as disturbing as the one in One Two Buckle My Shoe, which gave me nightmares. The plot
is complex, with all the essential ingredients, though not as convoluted as Buckle My Shoe,and in some way that is a good th
ing. The acting was very good, David Suchet is impeccable(I know I can't use this word forever but I can't think of a better
word to describe his performance in the series) as Poirot, and Phillip Jackson and Pauline Moran do justice to their integra
l characters brilliantly. And the students had great personalities and well developed on the whole, particularly Damian Lewi
s as Leonard. All in all, solid mystery but doesn't rank along the best. 7.5/10 Bethany Cox
"You probably having  Positive thoughts.
"You probably having  Negative thoughts.
"You probably having  Positive thoughts.
"You probably having  Positive thoughts.
```



Sentiment Analysis

What is your thoughts ?

mystery but doesn't rank along the best. 7.5/10 Bethany Cox

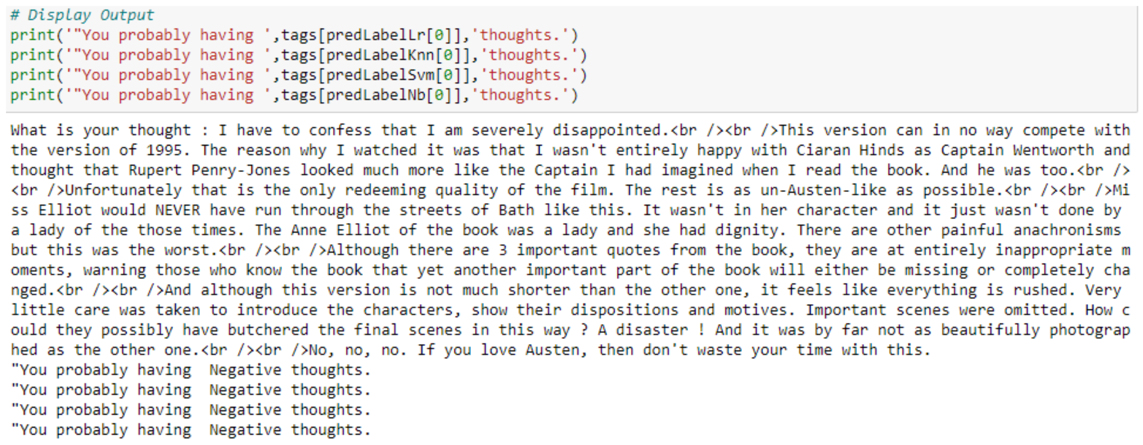
```
You probably having Positive thoughts.
```

**Test Result**

Input: (From Dataset 5)

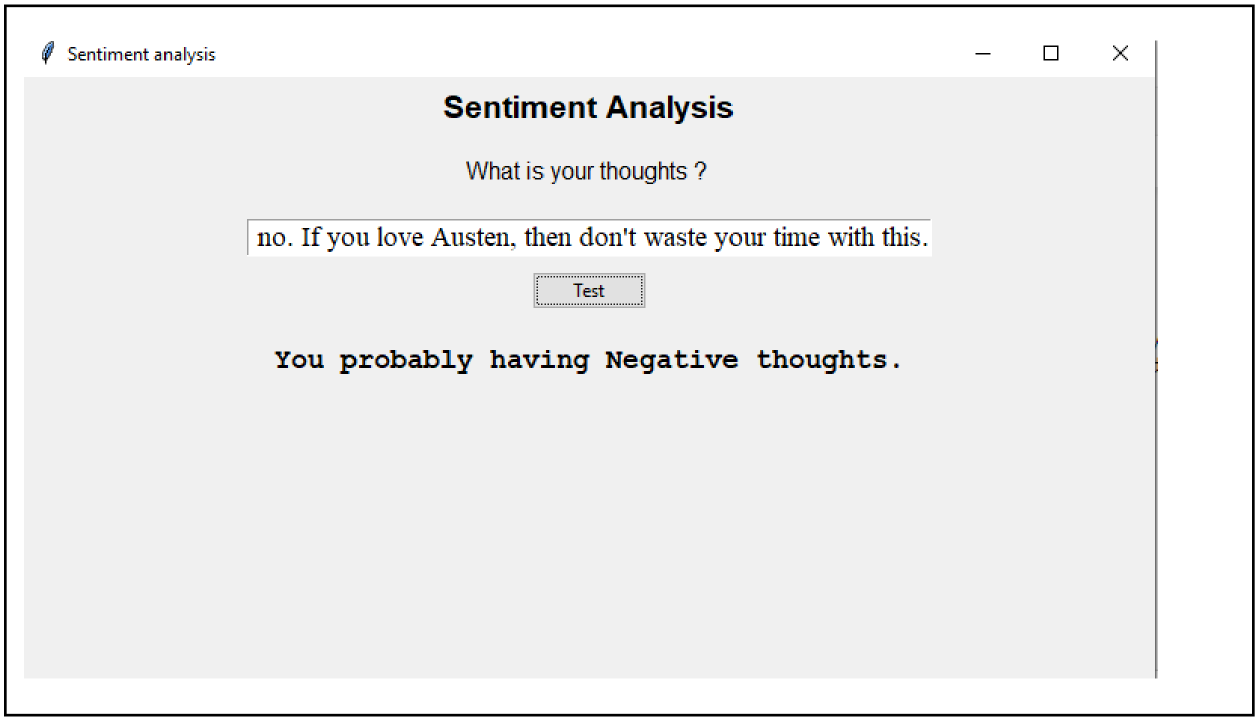
I have to confess that I am severely disappointed.<br /><br />This version can in no way compete with the version of 1995. The reason why I watched it was that I wasn't entirely happy with Ciaran Hinds as Captain Wentworth and thought that Rupert Penry-Jones looked much more like the Captain I had imagined when I read the book. And he was too.<br /><br />Unfortunately that is the only redeeming quality of the film. The rest is as un-Austen-like as possible.<br /><br />Miss Elliot would NEVER have run through the streets of Bath like this. It wasn't in her character and it just wasn't done by a lady of the those times. The Anne Elliot of the book was a lady and she had dignity. There are other painful anachronisms but this was the worst.<br /><br />Although there are 3 important quotes from the book, they are at entirely inappropriate moments, warning those who know the book that yet another important part of the book will either be missing or completely changed.<br /><br />And although this version is not much shorter than the other one, it feels like everything is rushed. Very little care was taken to introduce the characters, show their dispositions and motives. Important scenes were omitted. How could they possibly have butchered the final scenes in this way ? A disaster ! And it was by far not as beautifully photographed as the other one.<br /><br />No, no, no. If you love Austen, then don't waste your time with this.

Output:



```
# Display Output
print('"You probably having ',tags[predLabelLr[0]],'thoughts.')
print('"You probably having ',tags[predLabelKnn[0]],'thoughts.')
print('"You probably having ',tags[predLabelSvm[0]],'thoughts.')
print('"You probably having ',tags[predLabelNb[0]],'thoughts.')
```

```
What is your thought : I have to confess that I am severely disappointed.<br /><br />This version can in no way compete with
the version of 1995. The reason why I watched it was that I wasn't entirely happy with Ciaran Hinds as Captain Wentworth and
thought that Rupert Penry-Jones looked much more like the Captain I had imagined when I read the book. And he was too.<br />
<br />Unfortunately that is the only redeeming quality of the film. The rest is as un-Austen-like as possible.<br /><br />Mi
ss Elliot would NEVER have run through the streets of Bath like this. It wasn't in her character and it just wasn't done by
a lady of the those times. The Anne Elliot of the book was a lady and she had dignity. There are other painful anachronisms
but this was the worst.<br /><br />Although there are 3 important quotes from the book, they are at entirely inappropriate m
oments, warning those who know the book that yet another important part of the book will either be missing or completely cha
nged.<br /><br />And although this version is not much shorter than the other one, it feels like everything is rushed. Very
little care was taken to introduce the characters, show their dispositions and motives. Important scenes were omitted. How c
ould they possibly have butchered the final scenes in this way ? A disaster ! And it was by far not as beautifully photograp
hed as the other one.<br /><br />No, no, no. If you love Austen, then don't waste your time with this.
"You probably having  Negative thoughts.
"You probably having  Negative thoughts.
"You probably having  Negative thoughts.
"You probably having  Negative thoughts.
```



Sentiment Analysis

What is your thoughts ?

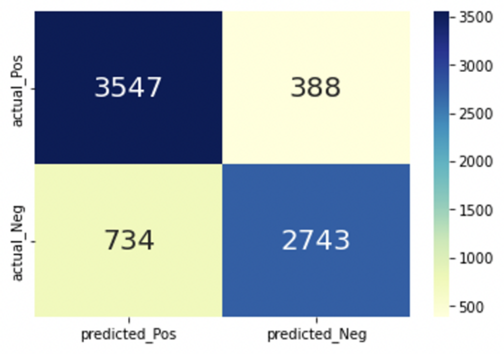
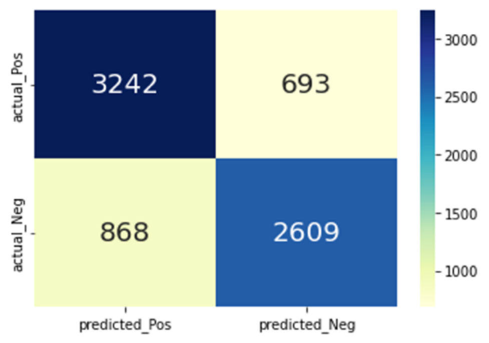
no. If you love Austen, then don't waste your time with this.

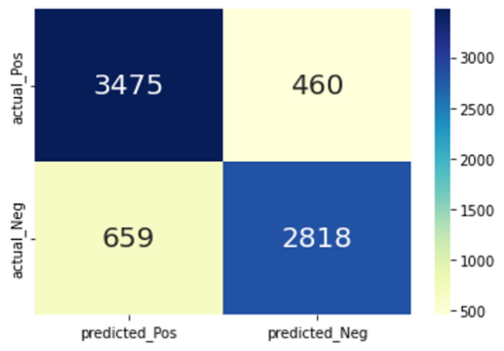
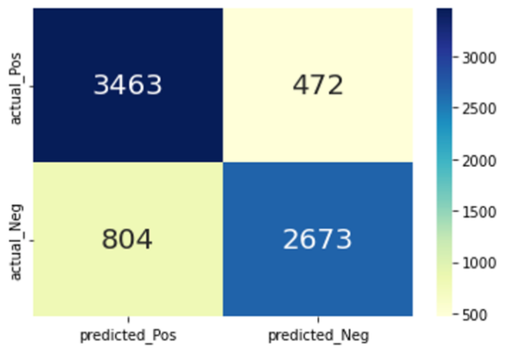
```
You probably having Negative thoughts.
```

## 4.2. Discussion/Interpretation

First and foremost, from the result of positive test data, Logistic Regression, Support Vector Machine (SVM), and Naïve Bayes Classification show results of Positive which as predicted but K-Nearest Neighbors (KNN) show results of Negative which not as predicted. This show that only one algorithm is unable to detect positive statements accurately. However, as the accuracy of KNN is lower than others, it is understandable that the result might be inaccurate as others. There might be issues with the algorithm which need improvement but there might be issues with the dataset of training and test as well which only can be detected if the algorithm is deeply being explored in the future. On the other hand, from the result of negative test data, Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machine (SVM), and Naïve Bayes Classification show results of Negative which as predicted. This show that the algorithms are able to detect negative statements accurately. As compared to positive statements, the negative statements might have higher accuracy.

Table below show the accuracy, precision, and recall of each algorithm

| | Plot Graph |
|---|---|
| Logistic Regression |  |
| K-Nearest Neighbors (KNN) |  |

| | |
|---|---|
| Support Vector Machine (SVM) |  |
| Naïve Bayes Classification |  |

# 5. Discussion and Conclusion

## 5.1. Achievements

In the project, the algorithms of Long Short Term Memory (LSTM), Logistic Regression, K-Nearest Neighbors (KNN), Support Vector Machines (SVM) and Naïve Bayes Classification are deployed successfully. The accuracy of each algorithm is tabulated as shown:

| |
|---|
| Long Short Term Memory (LSTM): 77.42% / 87.19% |
| Logistic Regression: 84.86% |
| K-Nearest Neighbors (KNN): 78.94% |
| Support Vector Machines (SVM): 84.90% |
| Naïve Bayes Classification: 82.78% |

From the accuracy of each algorithm, is shown that each of them reached accuracy above 75%. There is space for improvement for each algorithm as the accuracy could be higher. However, the development is still achieving the goal which able to classify the statements.

## 5.2. Limitations and Future Works

Sentiment analysis may perform several analyses such as morphology, prosody, phonology, syntactic ambiguity, semantic ambiguity, etc. to find out the true meaning of a sentence, However, as sentiment analysis is closely related to individuals', it can be challenging and causing issues in various ways such as subjectivity and tone, context and polarity, irony and sarcasm, comparisons, emojis, defining neutral, and human annotator accuracy (MonkeyLearn, 2022).

Firstly, subjectivity and tone occur as the could be classified into subject and object which subject is more relevant to sentiment while object is not. For example, when describing a car as powerful and colored, powerful will have a positive sentiment while colored will have a neutral sentiment. This is due to the subjectivity and tone of powerful is more than colored. Next, context and polarity occur as different contexts with same inputs may results differently. For example, inputs of 'Everything about it' and 'Nothing about it' for contexts of encouragement and criticism will differ as one is positive and one is negative. Thus, it is challenging as the algorithm is unable to learn the context by itself unless it is specified.

Besides, irony and sarcasm occur as individuals tend to use a word in an oppositive situation. For example, 'Yeah. For sure!', the exclamation mark may mean to be in an irony or sarcasm way of criticizing but the yeah may means to be in a positive manner. This is due to individuals' expression are differ. Then, comparisons and emojis occur due to each individual's expression are differ. For example, the comparisons between two useful things or between a new invention with useless existed one and emojis between social cultures or personal opinions. Moreover, defining neural occurs as things can be between positive and negative. For example, the word of 'maybe' means to be yes but no at the same time. Lastly, human annotator accuracy occurs as there is different mindsets between each individual. Therefore, developers of algorithms may have different opinions in the analysis thus making the results to be differ as the algorithm are developers by each different individual.

On the other hand, as the environment of developing the sentiment analysis is limited to a study area of students and the developers are having limited knowledges of sentiment analysis, the project is expected to be imperfect with several areas of improvements could be made. For instances, dataset could be prepared with more specific one, data preprocessing could be more precise to eliminate more noises, algorithms could be improved with more knowledge on the algorithms in the future, results could be more accurate if everything is enhanced respectively. The project is expected to be continued in the near future with more knowledge from the developers in the field of sentiment analysis.

# Reference & Source

Robinson, S., 2021. *Sentiment analysis: Why it's necessary and how it improves CX.* [Online]
Available at:
https://www.techtarget.com/searchcustomerexperience/tip/Sentiment-analysis-Why-its-necessary-and-how-it-improves-CX
[Accessed 18 08 2022].

Thematic, 2022. *Sentiment Analysis: Comprehensive Beginners Guide.* [Online]
Available at: https://getthematic.com/sentiment-analysis/
[Accessed 18 08 2022].

Thoughts, P., 2021. *5 Reasons Why Sentiment Analysis is Important.* [Online]
Available at: https://www.pyoneer.io/post/5-reasons-why-sentiment-analysis-is-important
[Accessed 18 08 2022].

Jacob, 2010. *TEXT CLASSIFICATION FOR SENTIMENT ANALYSIS – NAIVE BAYES CLASSIFIER.* [Online]
Available at:
https://streamhacker.com/2010/05/10/text-classification-sentiment-analysis-naive-bayes-classifier/
[Accessed 18 08 2022].

MonkeyLearn, 2022. *Sentiment Analysis: A Definitive Guide.* [Online]
Available at: https://monkeylearn.com/sentiment-analysis/
[Accessed 18 08 2022].

NLTK, 2022. *Natural Language Toolkit.* [Online]
Available at: https://www.nltk.org/
[Accessed 18 08 2022].

Education, I. C., 2020. *Supervised Learning.* [Online]
Available at: https://www.ibm.com/cloud/learn/supervised-learning
[Accessed 18 08 2022].

Education, I. C., 2020. *Unsupervised Learning.* [Online]
Available at: https://www.ibm.com/cloud/learn/unsupervised-learning
[Accessed 18 08 2022].

T, S. K., 2022. *Natural Language Processing – Sentiment Analysis using LSTM.* [Online]
Available at:
https://www.analyticsvidhya.com/blog/2021/06/natural-language-processing-sentiment-analysis-using-lstm/
[Accessed 18 08 2022].

Jacob, 2010. *TEXT CLASSIFICATION FOR SENTIMENT ANALYSIS – ELIMINATE LOW INFORMATION FEATURES.* [Online]
Available at:
https://streamhacker.com/2010/06/16/text-classification-sentiment-analysis-eliminate-low-inform

ation-features/
[Accessed 18 08 2022].