National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Mahesh Kumar Uppalapati
Student ID: 22176373

School of Computing
National College of Ireland

Supervisor: Hicham Rifai

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……Mahesh Kumar Uppalapati……

Student ID: ……x22176373……

Programme: ……Data Analytics…… **Year:** ……2023……

Module: ……Research Project……

Lecturer: ……Hicham Rifai……

Submission Due Date: ……31/01/2024……

Project Title: Utilizing Deep Learning Techniques for Sentiment Analysis during disasters

Word Count: …487… **Page Count:** ……7……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Mahesh Kumar Uppalapati……

Date: ……30/01/2024……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Mahesh Kumar Uppalapati
Student ID: x22176373

1. Introduction

This configuration file explains the sequence of steps to execute the code and also show implementation details related to the document.

2. System Specification

The development of earthquake tweet disaster prediction was deployed on system with these following specifications.

- Processor: Intel Core i5-1235U
- Operating System: Microsoft Windows 11
- Ram: 16 GB
- Solid State Drive (SSD): 256GB

3. Software's Used:

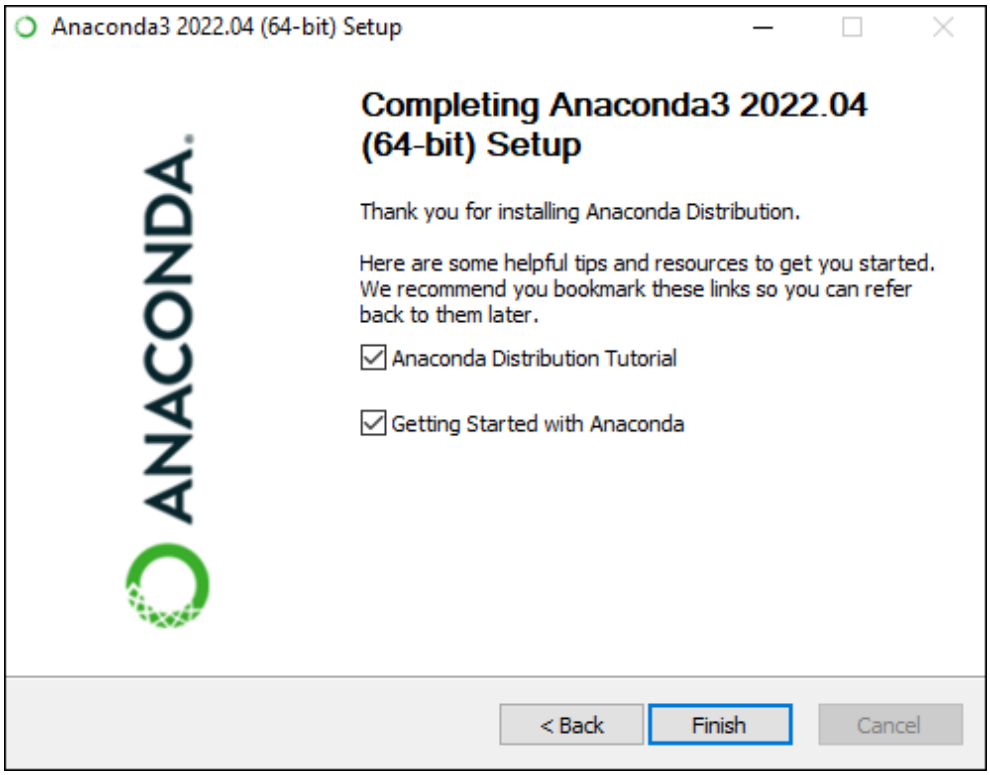
The following tools were used to executing the earthquake tweet disaster prediction

- Python 3.0.1
- Anaconda navigator
- Jupyter Notebook

4. Install the Software:

1) Install anaconda using below link:
https://docs.anaconda.com/free/anaconda/install/windows/

- Click on download and setup on system.



5. Dataset Source

1) Download dataset using below link:
   https://www.kaggle.com/datasets/kumari2000/turkey-syria-earthquake-tweets-dataset/

6. Execution of the Code

1) Install the libraries as show in below.

```
import numpy as np
import pandas as pd
import matplotlib
import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
import plotly.express as px
import missingno as msno
import re
import string

from wordcloud import WordCloud, STOPWORDS
from sklearn.decomposition import LatentDirichletAllocation
from collections import Counter
from nltk.sentiment import SentimentIntensityAnalyzer
from textblob import TextBlob

import warnings
warnings.simplefilter("ignore")
```

```
from sklearn import metrics
from sklearn.metrics import accuracy_score
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.metrics import precision_recall_curve, auc, roc_auc_score, roc_curve, recall_score
from sklearn.model_selection import train_test_split

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer

from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem.snowball import SnowballStemmer
from nltk.stem.porter import PorterStemmer
from nltk.stem import WordNetLemmatizer
from nltk import ngrams
import tensorflow as tf
from tensorflow.keras.layers import Input, Embedding, Bidirectional, LSTM, Dense, Attention, Concatenate
from tensorflow.keras.models import Model
from keras.preprocessing.sequence import pad_sequences
from keras.preprocessing.text import Tokenizer
from keras.models import Sequential, Model
from keras.layers import LSTM, Dense, TimeDistributed, Bidirectional, Embedding, Dropout, Flatten, Layer, Input
from keras.optimizers import SGD, Adam, Adagrad, Adadelta, RMSprop
from keras.callbacks import EarlyStopping, ModelCheckpoint
import keras.backend as K
```

2) Load the dataset

```
data=pd.read_excel('Earthquake datset.xlsx')
```

```
print("Data information:",data.shape)
```

```
Data information: (42650, 19)
```

3) Preprocess the data by removing breaks,punctuations,numbers links and etc.

```
# remove break symbol
def remove_break(text):
    return re.sub(r'<br />', '', text)
# remove punctuations
def remove_punct(text):
    nopunct = ''
    for c in text:
        if c not in string.punctuation:
            nopunct = nopunct + c
    return nopunct
# remove numbers
def remove_numbers(text):
    return re.sub(r'[0-9]', '', text)
# remove links
def remove_links(text):
    return re.sub(r'http\S+', '', text)
# remove stop words
def remove_stop_words(word_list):
    # stop words
    stopwords_list = set(stopwords.words('english'))
    # remove stop words
    word_list = [word for word in word_list if word not in stopwords_list]
    # stemming
    return ' '.join(word_list)
def get_root(word_list):
    ps = PorterStemmer()
    return [ps.stem(word) for word in word_list]
def clean_text(text):
    text = remove_break(text)
    text = remove_links(text)
```

4) Data before Pre-Processing

```
data['Tweet Content'].head(10)
```

```
0    "128 hours buried, rescued alive, checked in t...
1    "RICO from puppy to #rescuedog #USAR #PL and h...
2    "May Allah have mercy on the Muslim Ummah, Ame...
3    "استغفراللہ \n#TurkeySyriaEarthquake\n#Turkey\...
4    "کی رنگا رنگ تقریب اور #PSL8 اک طرف #TimHorton...
5    "128 hours buried, rescued alive, checked in t...
6    "کی رنگا رنگ تقریب اور #PSL8 اک طرف #TimHorton...
7    "کی رنگا رنگ تقریب اور #PSL8 اک طرف #TimHorton...
8    "128 hours buried, rescued alive, checked in t...
9    "128 hours buried, rescued alive, checked in t...
Name: Tweet Content, dtype: object
```

5) Data after Pre-processing

```
data['clean_content'].head(10)
```

```
0    hour buri rescu aliv check in the hospit fed a...
1    rico from puppi to rescuedog usar pl and hi ha...
2    may allah have merci on the muslim ummah ameen...
3    استغفراللہ turkeysyriaearthquak turkey syria t...
4    کی رنگا رنگ تقریب اور psl اک طرف timhorton پر ...
5    hour buri rescu aliv check in the hospit fed a...
6    کی رنگا رنگ تقریب اور psl اک طرف timhorton پر ...
7    کی رنگا رنگ تقریب اور psl اک طرف timhorton پر ...
8    hour buri rescu aliv check in the hospit fed a...
9    hour buri rescu aliv check in the hospit fed a...
Name: clean_content, dtype: object
```

6) After data is prepared intialise the models LSTM/SVM/Naïve-Bayes

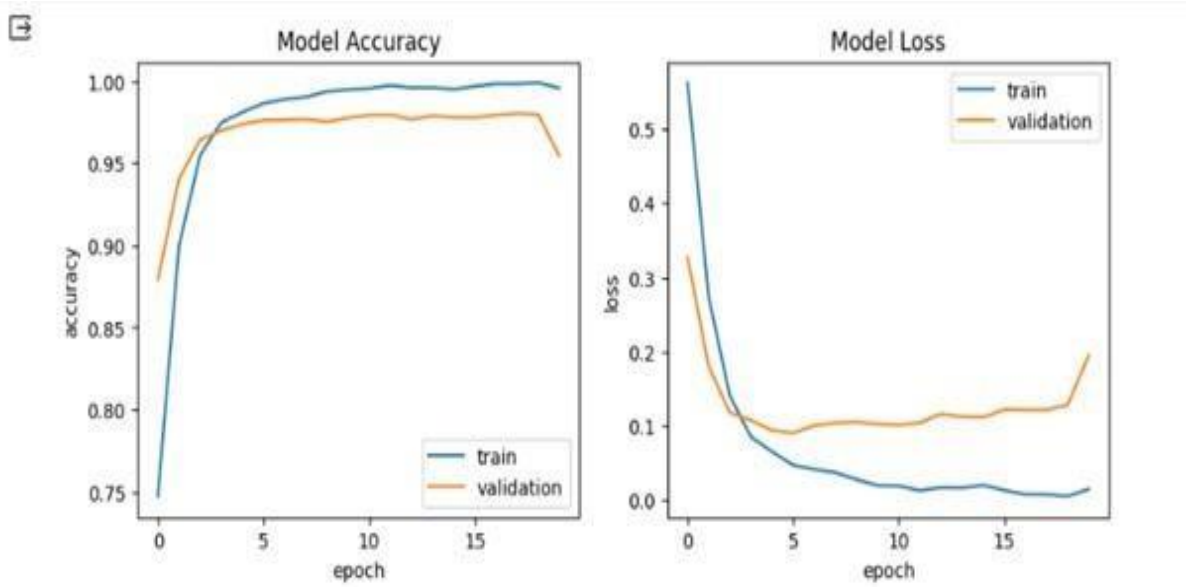
```
learning_rate = 0.1
decay_rate = learning_rate / epochs
momentum = 0.8

sgd = SGD(learning_rate=learning_rate, momentum=momentum)
# Build model
model= Sequential()
model.add(Embedding(vocab_size, embedding_size, input_length=max_len))
model.add(Conv1D(filters=32, kernel_size=3, padding='same', activation='relu'))
model.add(MaxPooling1D(pool_size=2))
model.add(Bidirectional(LSTM(32)))
model.add(Dropout(0.4))
model.add(Dense(3, activation='softmax'))
```

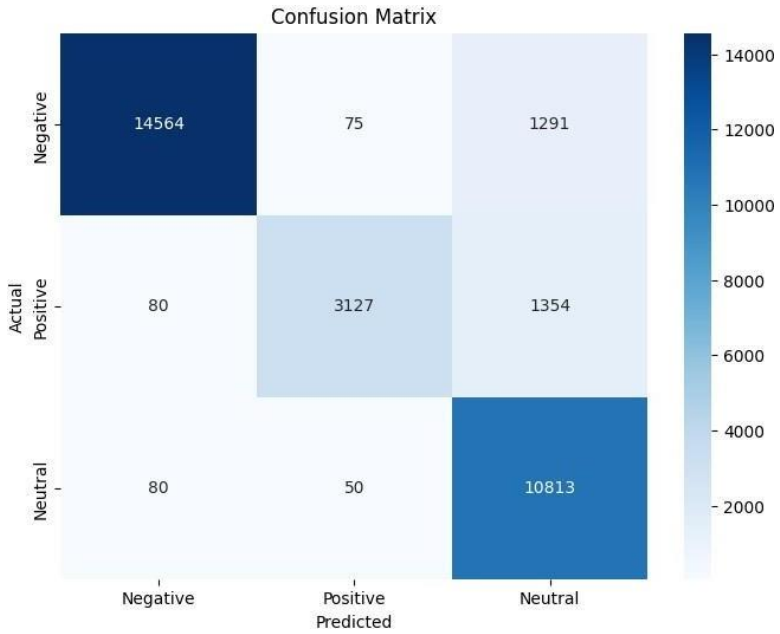
7) Perform the training by optimising the loss.

```
None
Epoch 1/20
295/295 [==============================] - 17s 44ms/step - loss: 0.5625 - accuracy: 0.7471 - precision: 0.8083 - recall: 0.6662 - val_loss: 0.3276 - val_accuracy: 0.8793 - val_precision: 0.8943 - val_recall: 0.8600
Epoch 2/20
295/295 [==============================] - 12s 41ms/step - loss: 0.2738 - accuracy: 0.8998 - precision: 0.9144 - recall: 0.8856 - val_loss: 0.1800 - val_accuracy: 0.9487 - val_precision: 0.9582 - val_recall: 0.9313
Epoch 3/20
295/295 [==============================] - 10s 33ms/step - loss: 0.1405 - accuracy: 0.9549 - precision: 0.9586 - recall: 0.9513 - val_loss: 0.1179 - val_accuracy: 0.9644 - val_precision: 0.9655 - val_recall: 0.9623
Epoch 4/20
295/295 [==============================] - 11s 38ms/step - loss: 0.0848 - accuracy: 0.9749 - precision: 0.9765 - recall: 0.9734 - val_loss: 0.1073 - val_accuracy: 0.9699 - val_precision: 0.9704 - val_recall: 0.9693
Epoch 5/20
295/295 [==============================] - 12s 40ms/step - loss: 0.0650 - accuracy: 0.9812 - precision: 0.9819 - recall: 0.9803 - val_loss: 0.0938 - val_accuracy: 0.9739 - val_precision: 0.9745 - val_recall: 0.9731
Epoch 6/20
295/295 [==============================] - 11s 39ms/step - loss: 0.0468 - accuracy: 0.9866 - precision: 0.9874 - recall: 0.9859 - val_loss: 0.0903 - val_accuracy: 0.9763 - val_precision: 0.9771 - val_recall: 0.9760
Epoch 7/20
295/295 [==============================] - 15s 50ms/step - loss: 0.0411 - accuracy: 0.9890 - precision: 0.9896 - recall: 0.9886 - val_loss: 0.1002 - val_accuracy: 0.9765 - val_precision: 0.9768 - val_recall: 0.9760
Epoch 8/20
295/295 [==============================] - 17s 58ms/step - loss: 0.0369 - accuracy: 0.9905 - precision: 0.9914 - recall: 0.9900 - val_loss: 0.1038 - val_accuracy: 0.9769 - val_precision: 0.9776 - val_recall: 0.9766
Epoch 9/20
295/295 [==============================] - 23s 77ms/step - loss: 0.0274 - accuracy: 0.9937 - precision: 0.9941 - recall: 0.9932 - val_loss: 0.1051 - val_accuracy: 0.9753 - val_precision: 0.9766 - val_recall: 0.9749
Epoch 10/20
295/295 [==============================] - 14s 47ms/step - loss: 0.0195 - accuracy: 0.9958 - precision: 0.9954 - recall: 0.9949 - val_loss: 0.1022 - val_accuracy: 0.9777 - val_precision: 0.9782 - val_recall: 0.9776
Epoch 11/20
295/295 [==============================] - 14s 48ms/step - loss: 0.0187 - accuracy: 0.9957 - precision: 0.9958 - recall: 0.9955 - val_loss: 0.1009 - val_accuracy: 0.9795 - val_precision: 0.9796 - val_recall: 0.9790
Epoch 12/20
295/295 [==============================] - 17s 59ms/step - loss: 0.0127 - accuracy: 0.9975 - precision: 0.9976 - recall: 0.9975 - val_loss: 0.1042 - val_accuracy: 0.9796 - val_precision: 0.9798 - val_recall: 0.9793
Epoch 13/20
295/295 [==============================] - 16s 54ms/step - loss: 0.0165 - accuracy: 0.9961 - precision: 0.9964 - recall: 0.9959 - val_loss: 0.1160 - val_accuracy: 0.9768 - val_precision: 0.9774 - val_recall: 0.9766
Epoch 14/20
295/295 [==============================] - 12s 40ms/step - loss: 0.0161 - accuracy: 0.9962 - precision: 0.9963 - recall: 0.9960 - val_loss: 0.1121 - val_accuracy: 0.9790 - val_precision: 0.9792 - val_recall: 0.9790
Epoch 15/20
295/295 [==============================] - 21s 70ms/step - loss: 0.0197 - accuracy: 0.9952 - precision: 0.9953 - recall: 0.9949 - val_loss: 0.1114 - val_accuracy: 0.9780 - val_precision: 0.9782 - val_recall: 0.9780
Epoch 16/20
295/295 [==============================] - 18s 62ms/step - loss: 0.0129 - accuracy: 0.9970 - precision: 0.9972 - recall: 0.9969 - val_loss: 0.1218 - val_accuracy: 0.9779 - val_precision: 0.9782 - val_recall: 0.9777
Epoch 17/20
295/295 [==============================] - 13s 45ms/step - loss: 0.0075 - accuracy: 0.9985 - precision: 0.9986 - recall: 0.9984 - val_loss: 0.1214 - val_accuracy: 0.9795 - val_precision: 0.9799 - val_recall: 0.9790
Epoch 18/20
295/295 [==============================] - 15s 52ms/step - loss: 0.0074 - accuracy: 0.9984 - precision: 0.9985 - recall: 0.9984 - val_loss: 0.1212 - val_accuracy: 0.9804 - val_precision: 0.9806 - val_recall: 0.9804
Epoch 19/20
295/295 [==============================] - 17s 57ms/step - loss: 0.0048 - accuracy: 0.9992 - precision: 0.9994 - recall: 0.9991 - val_loss: 0.1282 - val_accuracy: 0.9800 - val_precision: 0.9804 - val_recall: 0.9796
Epoch 20/20
295/295 [==============================] - 12s 40ms/step - loss: 0.0147 - accuracy: 0.9959 - precision: 0.9960 - recall: 0.9959 - val_loss: 0.1950 - val_accuracy: 0.9547 - val_precision: 0.9551 - val_recall: 0.9547
```

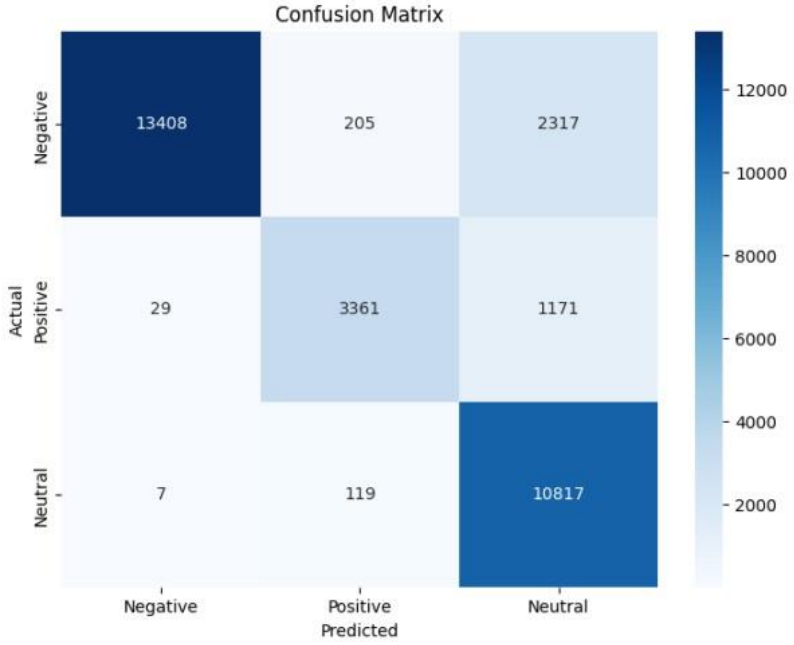
8) Check whether the mode loss is being opimised properly or not using loss plots



9) Then predict using trained models on test data and check confusion matrix for true positives.
SVM Confusion Matrix:



Navie Bayes Confusion Matrix:



References:

1) www.kaggle.com
2) www.anaconda..com