A
Minor Project Report on

"Fake Disaster Tweet Detection Web-App"

In partial fulfilment of requirements for the degree of

Bachelor of Technology (B. Tech.)

In

Computer Science and Engineering

Submitted by

Ms. Harshita Dadhich
(170297)

Under the Guidance of

Dr. Sunil Jangir

Department of Computer Science and Engineering

SCHOOL OF ENGINEERING AND TECHNOLOGY

Mody University and Science and Technology
Lakshmangarh, Distt. Sikar-332311

December 2020

A C K N O W L E D G E M E N T

Apart from the efforts of me, the success of this project depends largely on the encouragement and guidance of many others. I take this opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project.

I am profoundly grateful to Dr. Sunil Jangir for this support, guidance and continuous encouragement throughout to see that this project rights its target since its commencement to its completion. I would also like to express my deepest appreciation towards Dr. A. Senthil HOD Computer Science Department and Asst. Dean of SET whose invaluable guidance helped me in completing this project. At last, I must express my sincere heartfelt gratitude to all the members of Computer Science Engineering Department who helped me directly or indirectly during this course of work.

Harshita Dadhich

170297

CERTIFICATE

This is to certify that the minor project report entitled "Fake Disaster Tweet Web-App " submitted by Ms. Harshita Dadhich, as a partial fulfilment for the requirement of B. Tech. VII Semester examination of the School of Engineering and Technology, Mody University of Science and Technology, Lakshmangarh for the academic session 2019-2020 is an original project work carried out under the supervision and guidance of Dr. Sunil Jangir my has undergone the requisite duration as prescribed by the institution for the project work.

PROJECT GUIDE:	**HEAD OF DEPARTMENT**
Approval Code: AUT_20_CSE_F12_08	**Signature:**
Name: Dr. Sunil Jangir	**Name:** Dr. A. Senthil
Date: 26/12/2020	**Date:**

EXAMINER-I:	**EXAMINER-II**
Name:	**Name:**
Dept:	**Dept:**

ABSTRACT

This project "Fake Disaster Tweet Detection" aims to help predict, tweet weather it is fake or real. It uses Multinomial Naïve Bayes approach for detecting fake or real tweet from existing dataset available on Kaggle. The classifier will be trained only on text data. Traditionally text analysis is performed using Natural Language Processing also known as NLP. Natural language processing is a field which comes under Artificial Intelligence. Its main focus is on letting computers understand human language and process it. NLP helps recognize and predict diseases using speech, it helps in sentiment analysis, cognitive assistant, spam detection, healthcare industry, etc. In this project Training Data is pre-processed, then sent to the classifier, then and the classifier predicts weather the tweet is real or fake.

This project is made on Jupyter Notebook which is a part of Anaconda Navigator. This project ran successful on Jupyter Notebook. The dataset was successfully loaded into the notebook. All the extra python packages which were required for project completion also loaded into the notebook. The model is also deployed successfully using html, CSS, python and flask.

The accuracy score on test data is 77.977%. average recall value is 0.775 and average precision score is 0.775. Precision is used to calculate number of correct positive predictions made by the model. Recall is used to calculate number of correct positive prediction made out of all the positive predictions that could have been made.

TABLE OF CONTENTS

Chapter 1: Introduction

1. INTRODUCTION

Nowadays, social media plays an important role for spreading information especially Twitter. A tweet is a message or any short information posted on twitter that contains any kind of text, video, a GIF, or images. It is displayed on profile page of the person who is sending and Home Timeline of the person who is following the person posting the tweet. Twitter has become the biggest source of misinformation especially during disasters like earthquakes, floods, wildfires, etc. People post on twitter about disaster preparation, its development, and recovery which are not always true [1]. Daily, many verified and un-verified users post tweet on twitter and not all tweets posted on tweeter are genuine. The important point in this matter lies in classification of the fake and real tweets.

The problem of detecting fake news on disaster spreading on twitter has become more and more important these days [2]. Most papers published till now uses methods which had generalized detection of disaster tweets [3] or are mainly focused on tweets related to disaster [4]. In [5] author uses hybrid approach for detecting fake news based on node2vec to extract features and graph embedding.

This project "Fake Disaster Tweet Detection" aims to help predict, tweet weather it is fake or real. It uses Multinomial Naïve Bayes approach for detecting fake or real tweet from existing dataset available on Kaggle (https://www.kaggle.com/c/nlp-getting-started/data). The classifier will be trained only on text data. Traditionally text analysis is performed using Natural Language Processing also known as NLP.

Natural language processing is a field which comes under Artificial Intelligence. Its main focus is on letting computers understand human language and process it. NLP helps recognize and predict diseases using speech, it helps in sentiment analysis, cognitive assistant, spam detection, healthcare industry, etc. [6]. In this project Training Data is preprocessed, then sent to the classifier, then and the classifier predicts weather the tweet is real or fake. The limitation of the classifier "Multinomial Naïve Bayes Classifier algorithm" assumes that each attribute is mutually independent of each other which is almost impossible in real life [7]. Another limitation of this project is if a categorical value is present in test dataset but not in training dataset than the model will give zero for that.

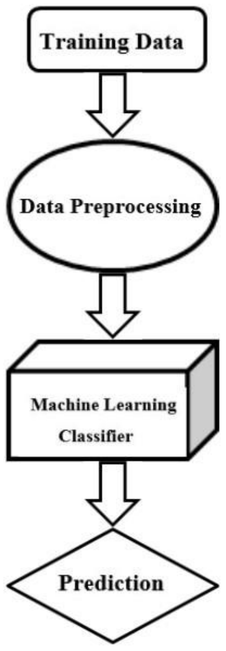


Figure 1.1: Prediction Model

Chapter 2: System Design

2. System Design

2.1 System Flowchart

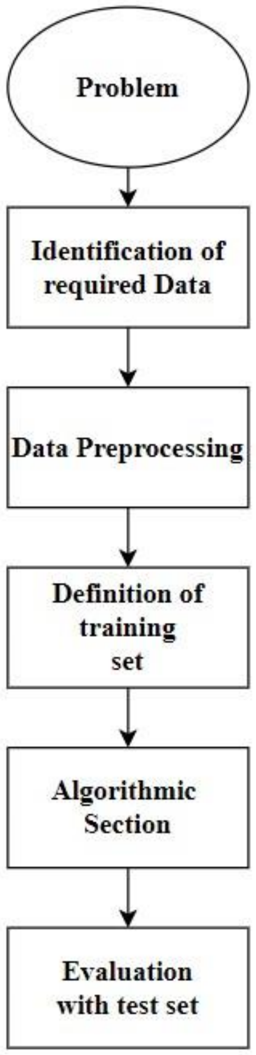


Figure 2.1 System Flowchart

Problem: To detect disaster tweet whether its fake or real using machine learning algorithm. In this the concept of Natural language Processing is used.

Identification of data: In this project I have used dataset available on Kaggle competition based on Natural language processing. This project works only on text data. It has five columns:

1. Id: It tells unique identification of each tweet
2. Text: It tells the tweet in text form
3. Location: It tells the place from where tweet was sent and it can be blank
4. Keyword: It tells a particular word in the tweet and it can be blank
5. Target: It tells the actual value of the tweet weather it's a real tweet or fake.

Data-preprocessing: First the preprocessing is done in the dataset which include removal of punctuations, then removal of URLs, digits, non-alphabets, contractions, then tokenization and removing Stopwords and removing Unicode. Then lemmatization is done on dataset. After preprocessing Countvectorizer is used to convert text data into numerical data as the classifier only works for numerical data. The dataset is then split into 70% training data and 30% test data.

Definition of Training Data: The training dataset which contains the 70% of the whole dataset is used for training the model.

Algorithm Section: In this project Multinomial Naïve Bayes classifier algorithm us used for detection disaster tweet whether they are fake or real.

Evaluation with test set: Several text samples are passed through the model to check whether the classification algorithm gives correct result or not.

Chapter 3: Hardware and Software Details

3. Software used

3.1 Anaconda Navigator

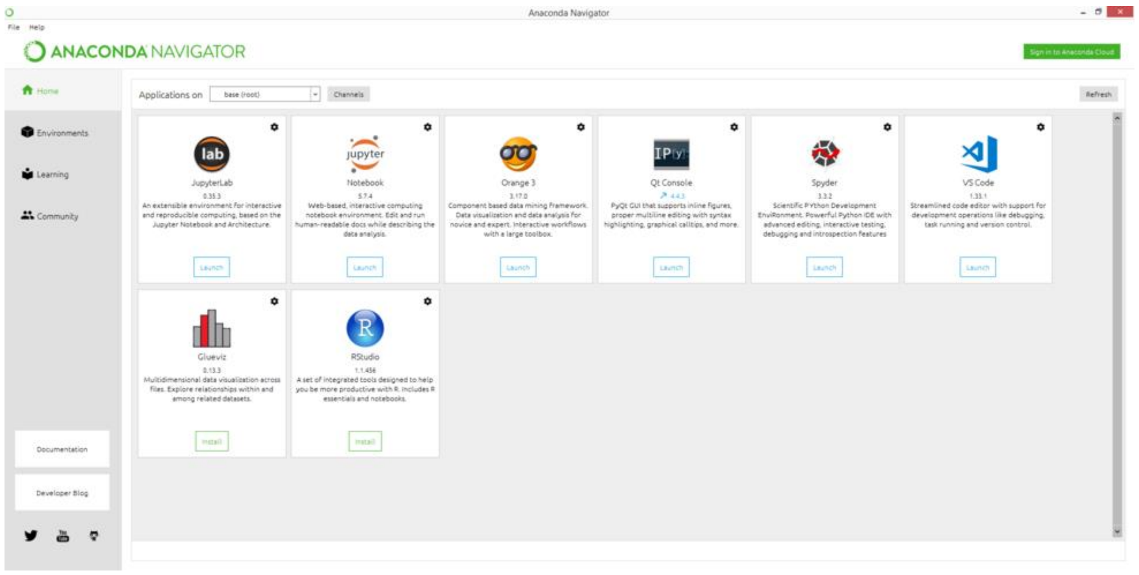
Anaconda is a platform for scientific computing. It is made for Python and R programming languages. Various tasks can be performed on this distribution like machine learning application, predictive analysis, data science, large-scale data processing. Its main aim is to make package management and deployment simpler. It includes packages used in data science projects suitable for all the three MacOS, Linux and Windows. Anaconda was developed in 2012 by Peter Wang and Travis Oliphant.

Figure 3.1. Anaconda Navigator Logo

Anaconda Distribution has 250 automatically installed packages and additional 7500 packages which are open source and can be installed using conda package and PyPI. Anaconda Distributer also has a GUI, Anaconda Navigator which is a graphical command line interface. From Anaconda Repository, or Anaconda Cloud, or user's private repository open-source packages can be installed. Using conda build command, custom packages can be made.

By default, there are many applications present in anaconda navigator: -

- Jupyter Notebook
- Jupyter Lab
- Spyder
- RStudio
- Orange 3 App



JupyterLab
0.35.3
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Notebook
5.7.4
Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

Orange 3
3.17.0
Component based data mining Framework. Data visualization and data analysis for novice and expert. Interactive workflows with a large toolbox.

Qt Console
4.4.3
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.

Spyder
3.3.2
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features

VS Code
1.33.1
Streamlined code editor with support for development operations like debugging, task running and version control.

Glueviz
0.13.3
Multidimensional data visualization across files. Explore relationships within and among related datasets.

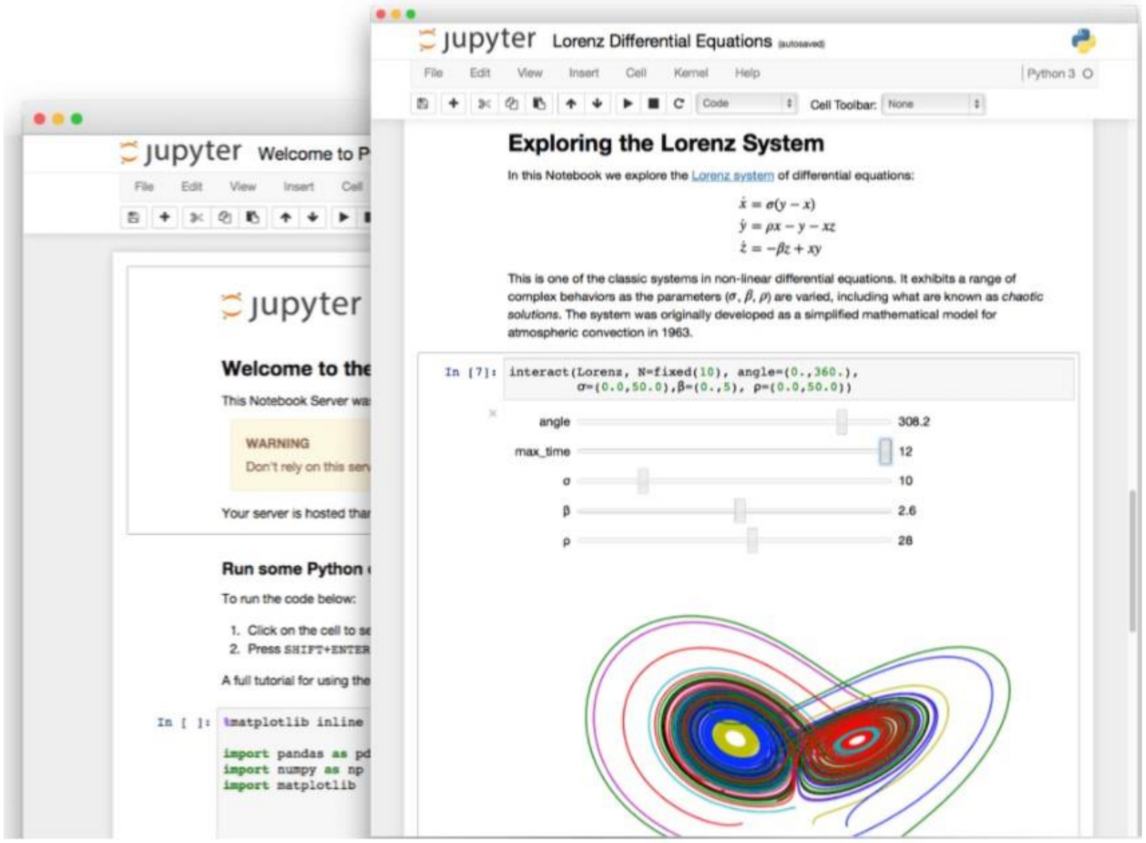
RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.

Figure 3.2 Anaconda Navigator

3.2 Jupyter Notebook

Jupyter Notebook is a web-based application. It is open source and allows us to create documents and share them. The documents contain codes, mathematical equations, graphs and text. Jupyter Notebook is used for various purposes like model training for Artificial Intelligence projects, stimulation of mathematical equations, plotting graphs for statistical data, etc. The Notebook also has console based interactive approach for computation. The Jupyter Notebook has two parts, one is web-based application and another is documentation part.

Figure 3.3. Jupyter Notebook Logo



jupyter

Welcome to the

This Notebook Server wa

WARNING

Don't rely on this serv

Your server is hosted thar

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

```
angle      308.2
max_time   12
σ          10
β          2.6
ρ          28
```

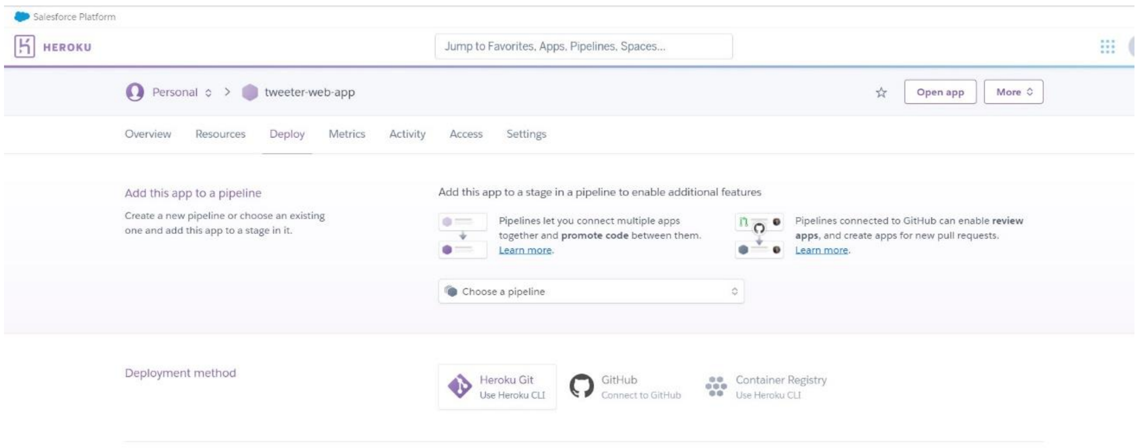
Figure 3.4. Jupyter Notebook

3.3 Heroku

Heroku is a web-based application. It supports various languages of programming. Heroku is used for deployment of various apps. Heroku helps developers to focus more on coding part rather than infrastructure. It is integrated with salesforce to give high performance. It provides a very strong dashboard. It is friendly for beginners and the companies which are at initially phase. It is suitable for less computation. It allows us to make a new server in less time with the help of command line interface.

HEROKU

Figure 3.5. Heroku Logo



Add this app to a pipeline

Create a new pipeline or choose an existing one and add this app to a stage in it.

Add this app to a stage in a pipeline to enable additional features

Pipelines let you connect multiple apps together and **promote code** between them. Learn more.

Pipelines connected to GitHub can enable **review apps**, and create apps for new pull requests. Learn more.

Deployment method

Figure 3.6. Heroku Cloud Platform

3.4 Python

Python is a programming language used for general purpose. It is an interpreted language which means it executes instructions directly, without previous compilation the program into machine language instructions. It is a high-level language which means it uses elements of natural language, which is easier to understand by humans. It is an object-oriented programming language. Python was first released in 1991 by Guido van Rossum and python 2.0 was released in 2000. It can easily be integrated with other languages like C, C++ etc. Python is very widely used in Machine learning and Deep learning applications.

Figure 3.7 Python logo

3.5 Flask

Flask is a web framework which is written in python. It does not require particular type of tool or libraries. It provides various tools and libraries to develop web application.

Once the flask server is started, we can go to browser and open localhost: 5000. Here the flask sever has rendered the default template. Applications that use the Flask framework include Pinterest and LinkedIn.

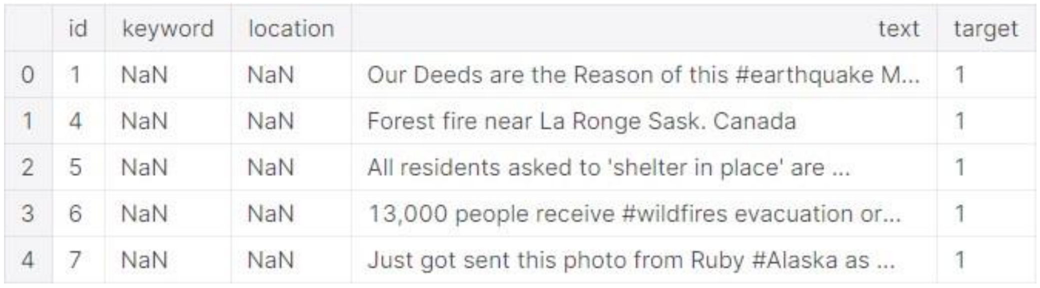
Figure 3.8 Flask logo

Chapter 4: Implementation Work Details

4. Implementation Work Details

The data-set which is used in this project "Fake disaster tweet detection" is taken from Kaggle competition "Natural Language Processing with Disaster Tweets". The data-set contains 7613 samples. This project works only on text data. It has five columns:

6. Id: It tells unique identification of each tweet
7. Text: It tells the tweet in text form
8. Location: It tells the place from where tweet was sent and it can be blank
9. Keyword: It tells a particular word in the tweet and it can be blank
10. Target: It tells the actual value of the tweet weather it's a real tweet or fake.



	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1

Figure 4.1 Dataset

Step 1: Exploratory Data Analysis

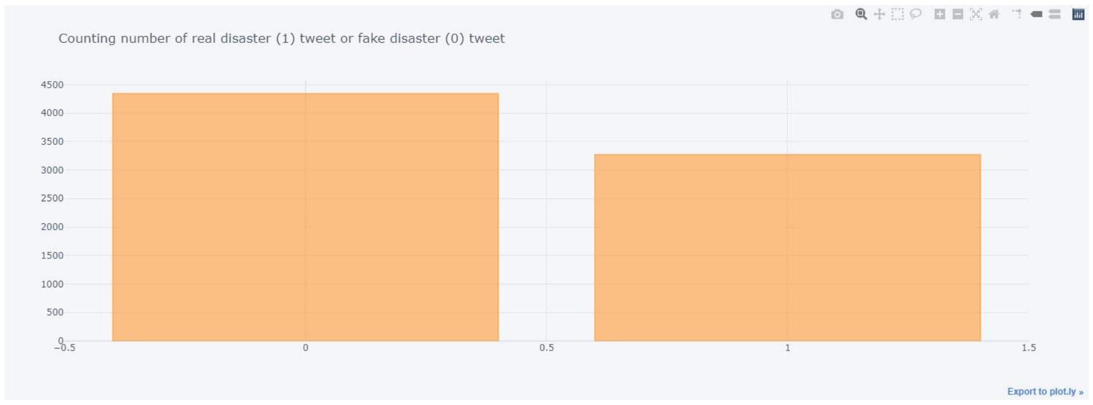


Figure 4.2 Number of real and fake tweet

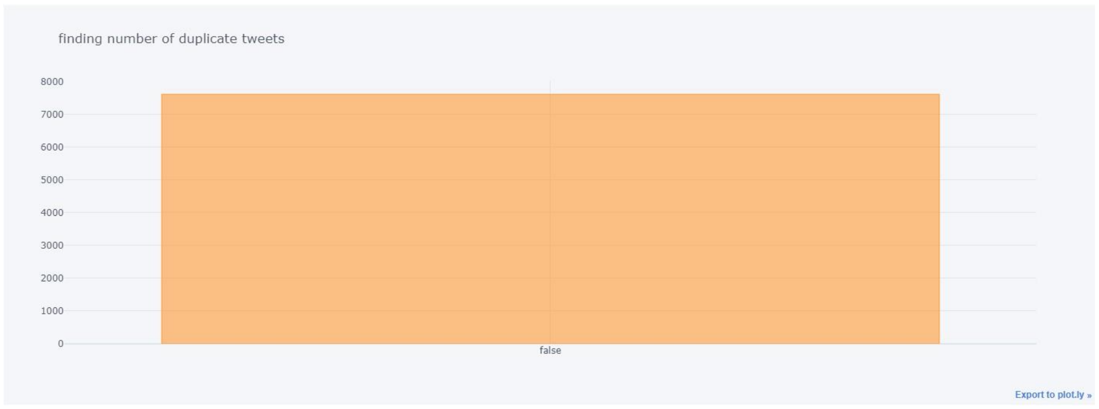


Figure 4.3 Number of Duplicate values

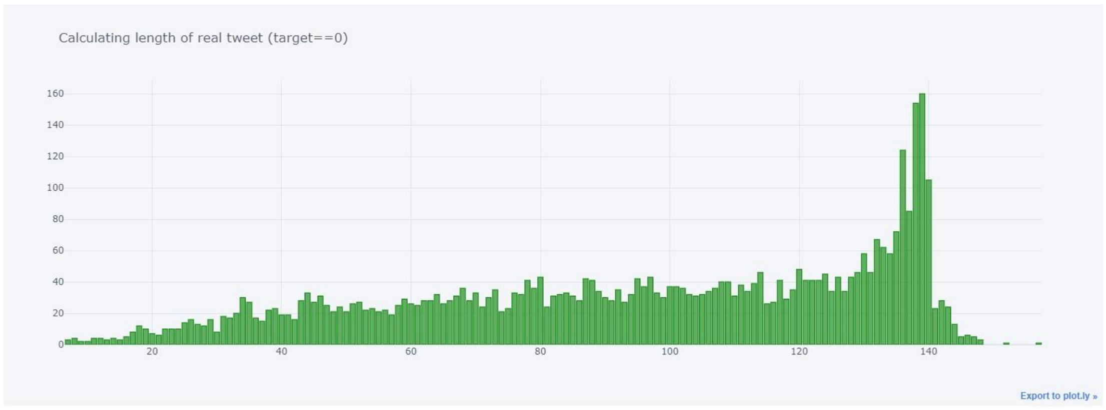


Figure 4.4 Length of Real Tweets

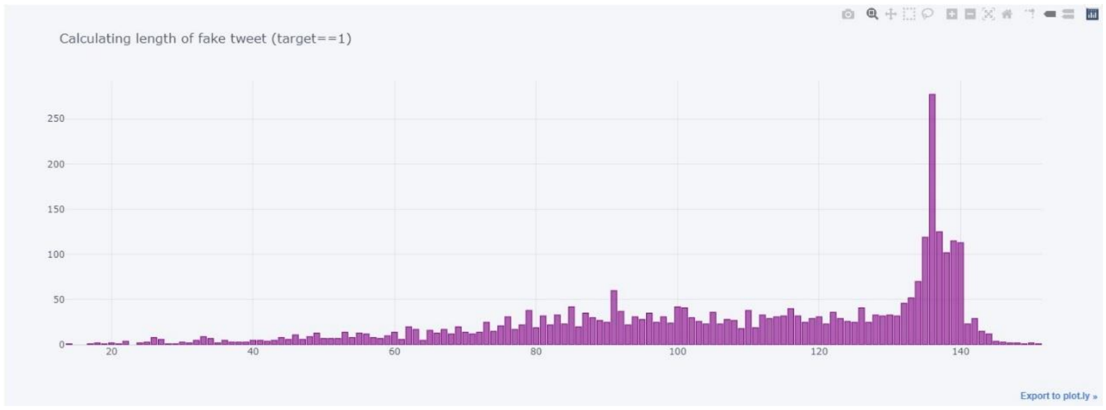


Figure 4.5 Length of Fake Tweets

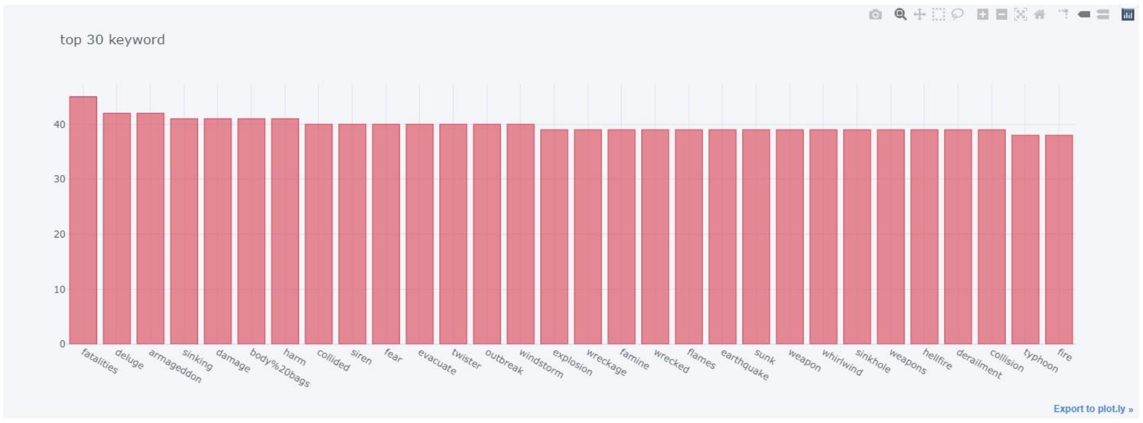


Figure 4.5 Top 30 Keywords

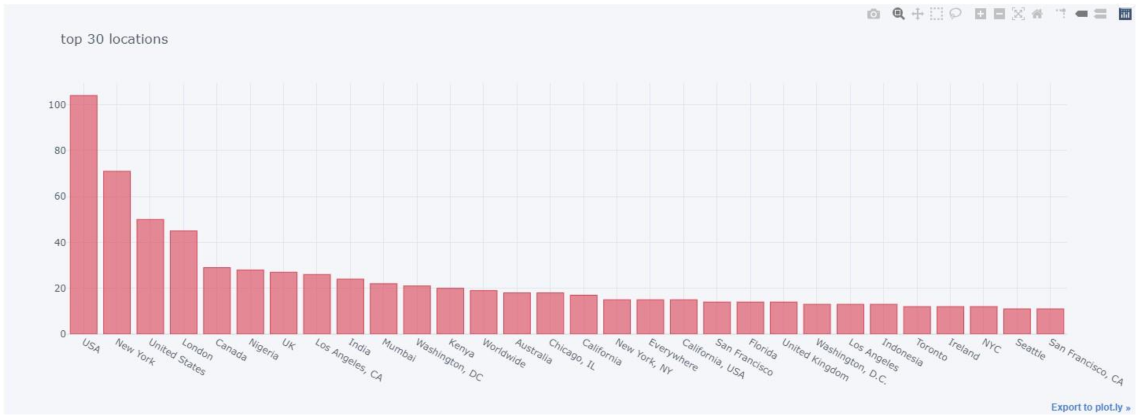


Figure 4.6 Top 30 Locations

Step 2: Data-Preprocessing

1. Removing Punctuations:

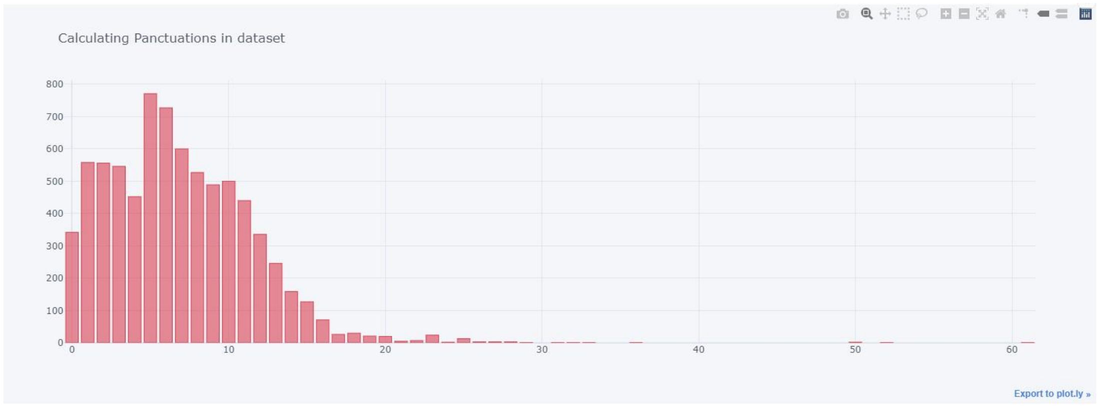
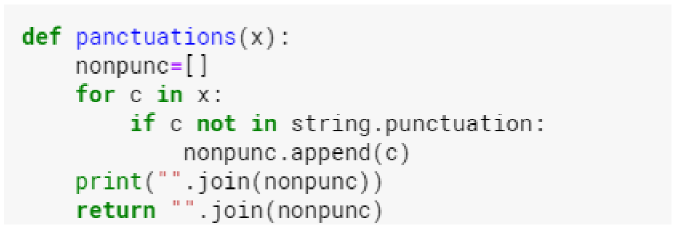


Figure 4.7 Punctuations in dataset

Punctuations are removed with the help of following python code:



```
def panctuations(x):
    nonpunc=[]
    for c in x:
        if c not in string.punctuation:
            nonpunc.append(c)
    print("".join(nonpunc))
    return "".join(nonpunc)
```

Figure 4.8 Code to remove Punctations

2. Removing URLs, digits, non-alphabets, _:
   True means it has HTTP and False means it does not have HTTP

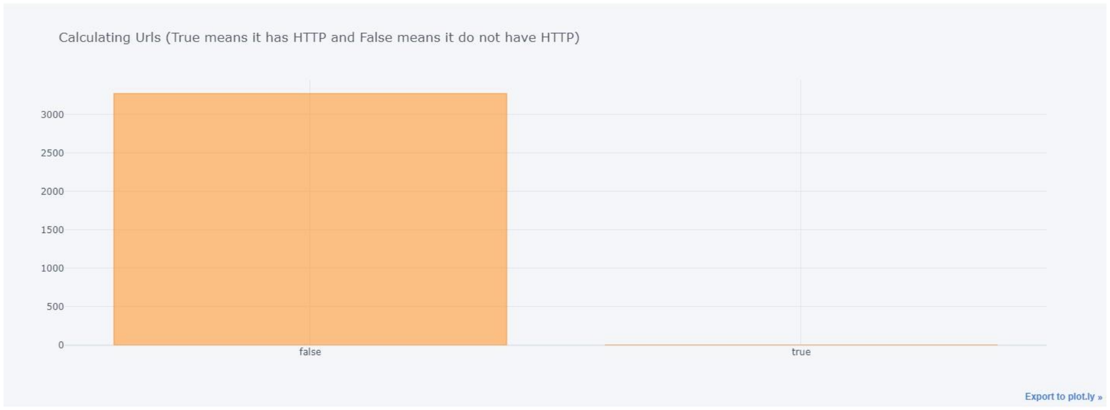
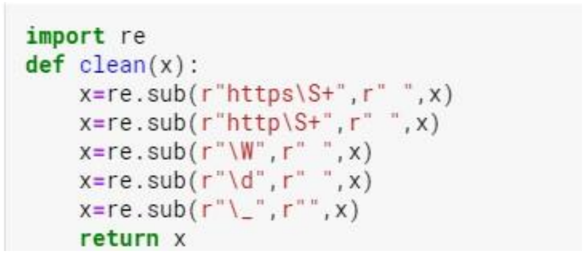


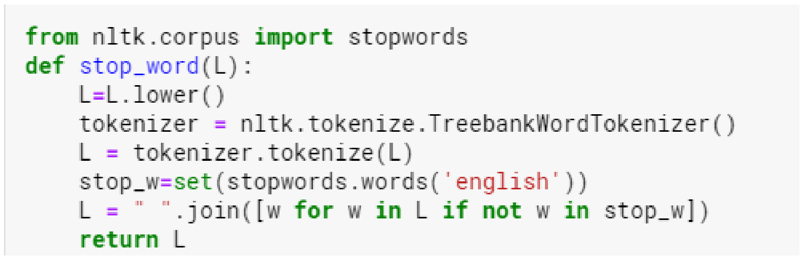
Figure 4.9 Tweets with URLs



```
import re
def clean(x):
    x=re.sub(r"https\S+",r" ",x)
    x=re.sub(r"http\S+",r" ",x)
    x=re.sub(r"\W",r" ",x)
    x=re.sub(r"\d",r" ",x)
    x=re.sub(r"\_",r"",x)
    return x
```

Figure 4.10 Code to remove URLs, digits, non-alphabets,_

3. Removing Contraction:
   It expands the words which are written in short form like can’t is expanded into cannot, I’ll is expanded into I will, etc.
4. Lowercase the text, tokenize them and remove Stopwords:
   Tokenizing means splitting the text into list of tokens. Stopwords are the words in text which does not provide additional meaning to the text.



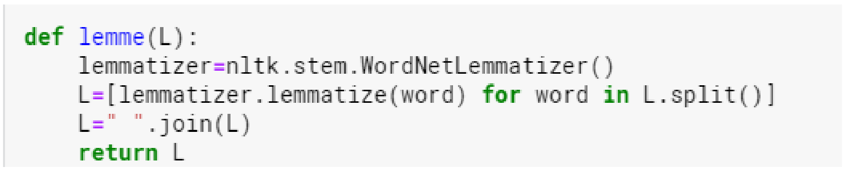
```python
from nltk.corpus import stopwords
def stop_word(L):
    L=L.lower()
    tokenizer = nltk.tokenize.TreebankWordTokenizer()
    L = tokenizer.tokenize(L)
    stop_w=set(stopwords.words('english'))
    L = " ".join([w for w in L if not w in stop_w])
    return L
```

Figure 4.11 Code for Stopwords

Figure 4.11 Code for removing stopwords, lowercasing and tokenizing the test

5. Lemmatizing:

   It converts any word into its root form like running, ran into run.



```python
def lemme(L):
    lemmatizer=nltk.stem.WordNetLemmatizer()
    L=[lemmatizer.lemmatize(word) for word in L.split()]
    L=" ".join(L)
    return L
```

Figure 4.12 Code for Lemmatization

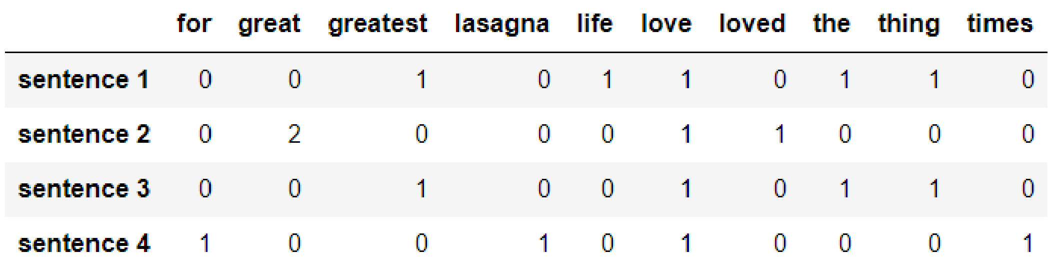
6. Countvectorizer:

   Text cannot be used to train our model, it has to be converted into number which our computer can understand, so for in this project Countvectorizer is used. Countvectorizer counts the number of times each word appears in a document. Countvectorizer works as:

   Step1: It first identifies unique words in complete dataset.

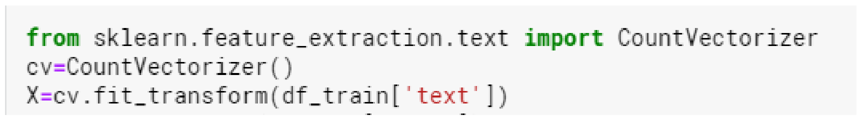
   Step 2: then it will create an array of zeros for each sample of same length as above

   Step 3: It then take each word at a time and find its occurrence in each sample in the dataset. The number of times the word appears in the sample will replace the zero positioned at the word in the list. This will repeat for every word.



	for	great	greatest	lasagna	life	love	loved	the	thing	times
sentence 1	0	0	1	0	1	1	0	1	1	0
sentence 2	0	2	0	0	0	1	1	0	0	0
sentence 3	0	0	1	0	0	1	0	1	1	0
sentence 4	1	0	0	1	0	1	0	0	0	1

Figure 4.13 Countvectorizer

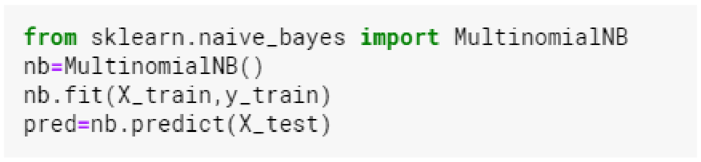


```
from sklearn.feature_extraction.text import CountVectorizer
cv=CountVectorizer()
X=cv.fit_transform(df_train['text'])
```

Figure 4.14 Code for Countvectorization

Step 3: Model Used:

In this project Multinomial Naïve Bayes approach is used for detecting fake or real tweet from existing dataset available on Kaggle. Naïve Bayes classifier is a based on probability theorem "Bayes Theorem" and also ha s an assumption of conditional independence among the every pairs.



```
from sklearn.naive_bayes import MultinomialNB
nb=MultinomialNB()
nb.fit(X_train,y_train)
pred=nb.predict(X_test)
```

Figure 4.15 Code for Model Training

Chapter 5: Source Code

5. Source Code

5.1 app.py

```
from flask import Flask
from flask import render_template
from flask import request
from flask import jsonify
from flask import redirect
from flask import url_for
from nltk.stem import WordNetLemmatizer
import re
import nltk
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
import numpy as np
import string
from sklearn.externals import joblib
from sklearn.feature_extraction.text import CountVectorizer
from unidecode import unidecode
import contractions
nltk.download('stopwords')
nltk.download('wordcount')

app = Flask(__name__)

@app.route('/')
def index():
    return render_template("index.html" )
```

```
@app.route("/api", methods=["GET","POST"])
def api():
    if request.method == "POST":
        cv = joblib.load('count.pkl')
        model = joblib.load('tweet.pkl')
        pstem = PorterStemmer()

        tweet = request.form["tweet"]
        text = tweet

        text = text.lower()
        text = text.split()
        p=[]
        for word in text:
            p.append(contractions.fix(word))
        t=[' '.join(p)]
        text=t[0]

        text = re.sub("[^a-zA-Z]", ' ', text)
        text=re.sub("https\S+"," ",text)
        text=re.sub("http\S+"," ",text)
        text=re.sub("\W"," ",text)
        text=re.sub("\d"," ",text)
        text=re.sub("\_","",text)
        text = unidecode(text)

        text = text.split()
        nonpunc=[]
        for c in text:
            if c not in string.punctuation:
                nonpunc.append(c)
        text = nonpunc

        lemmatizer=WordNetLemmatizer()
        L=[lemmatizer.lemmatize(word) for word in text]

        text = [' '.join(text)]

        X=cv.transform(text)

        prediction = model.predict(X)

        if prediction == 1:
            msg = "Your tweet is Not Fake (Real tweet)!!"
            return render_template("index.html", msg=msg, tweet=tweet)
        else:
            error = "Your tweet is Fake!!"
            return render_template("index.html", error=error, tweet=tweet)
    else:
        return redirect(url_for("index"))

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

5.2 Index.Html

```
<html>
<head>
<link rel="stylesheet" href="{{ url_for('static', filename = 'css/bootstrap.min.css') }}">
<link rel="stylesheet" href="{{ url_for('static', filename = 'css/style.css') }}">
<title>Fake DisasterTweet Detection</title>
</head>
<body>

<div class='container'>

{% if error %}
<div class="alert alert-danger">{{ error }}</div>
{% endif %}

{% if msg %}
<div class="alert alert-success">{{ msg }}</div>
{% endif %}

<h1>Fake Disaster Tweet Detection</h1>
<p class="lead">Simpel web api/service to detect fake <span>Disaster Tweet</span>
based on Multinomial Naive Bayes machine learning Classifier!!!!!!!!!!!</p>
<div class='image'>
<img src="{{ url_for('static', filename = 'img/tweet.png') }}">
</div>
<form method='post' action='/api' class="form-group">
<textarea id='tweet' placeholder='Write your tweet here ...' name='tweet' class="form-
control">{% if tweet%}{{ tweet }}{% endif %}
</textarea>
<input type='submit' value='test your tweet' class="btn btn-primary">
<!--<button id='clearTweet' class="btn btn-danger">Clear Tweet</button>-->
```

```
</form>
</div>

<script>
    clearBtn = document.getElementById("clearTweet")
    tweet = document.getElementById("tweet")
    clearBtn.onclick = function(e){
        e.preventDefault()
        tweet.value = "";
    }
</script>
</body>
</html>
```

5.3 style.css

```
body{
    background-image: url("../img/bg.jpg");
    background-size : cover;
}
.container{
    width: 735px;
    margin: 10vh auto;
    background-color: #FFF;
    padding: 15px;
    border-radius: 9px;
    box-shadow: 0px 0px 0px 5px #000000 inset;
}

h1{
text-shadow: #9bb4c3 0px 3px 4px;
```

```
    color: black;
    font-size: 40px;
    text-align: center;
    margin-bottom: 15px;
}

p{
    text-align: center;
    color: #a79e9e;
    line-height: 1.5;
}

span{
color: #00aced;
}

.image{
    margin: auto;
    padding: 6px;
    background: #1c536d;
    width: 128px;
    height: 128px;
    box-shadow: -1px 8px 10px #111215;
    overflow: hidden;
    border-radius: 50%;
    position: relative;
}

img{
    position: absolute;
    width: 100%;
    height: 100%;
    border-radius: 50%;
}

textarea{
    margin: 30px auto 15px;
    height: 127px;
    box-shadow:0px 0px 0px 0.5px #a3a3a3;
    padding: 15px;
}

input[type='submit']{
    text-transform: capitalize;
}
.alert{
    text-align: center;
}
```

Chapter 6: Project Screen

6. Project Screen

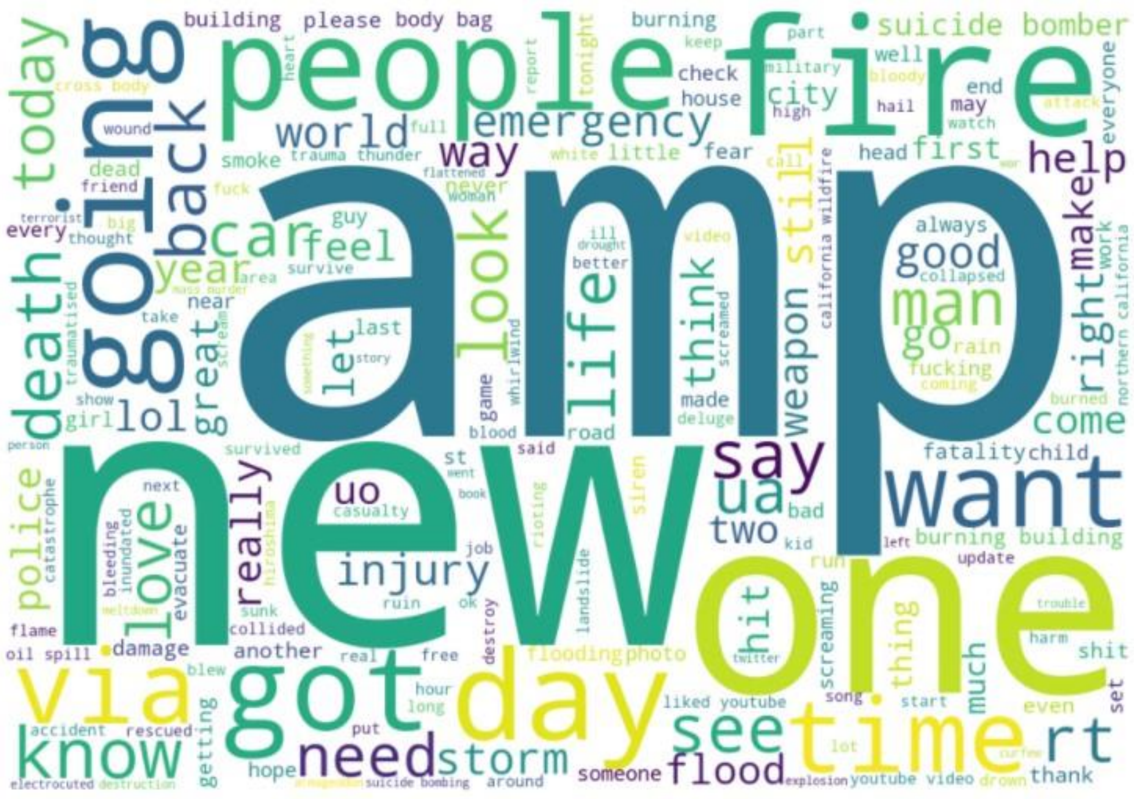
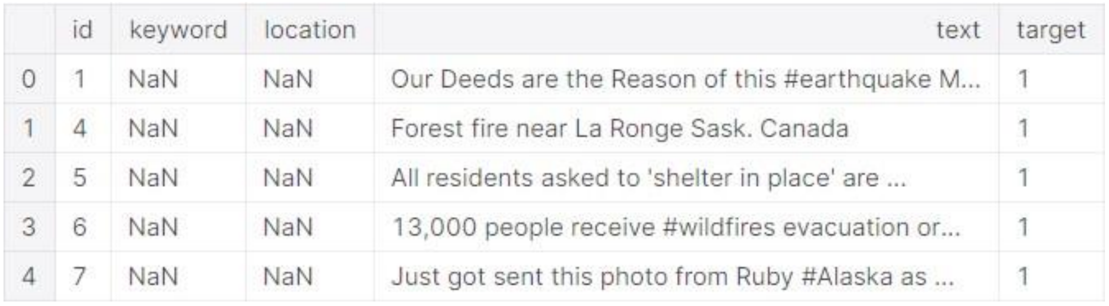
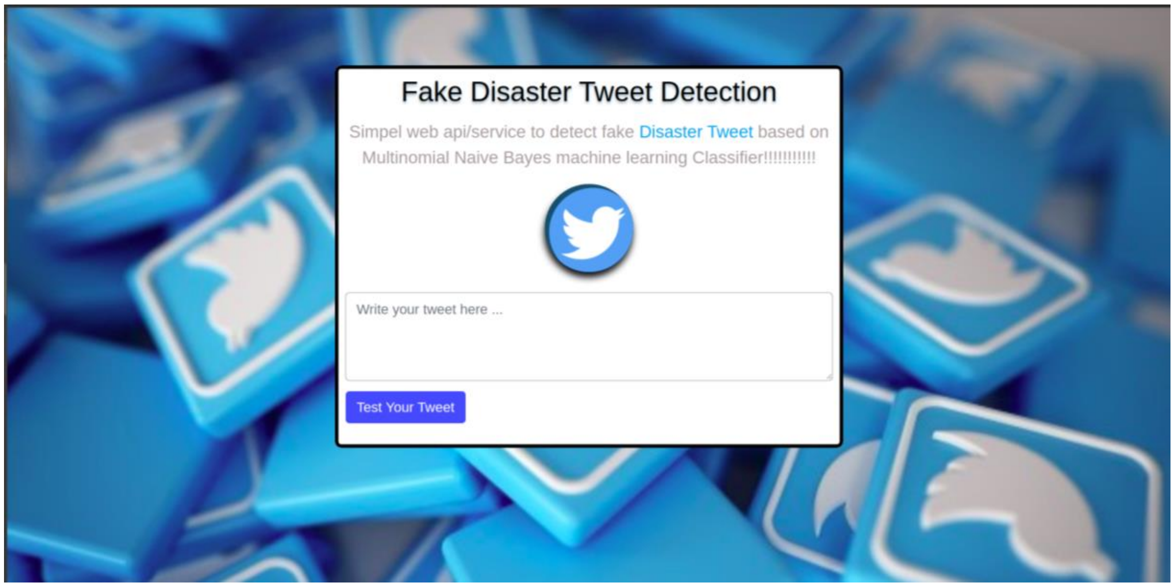


Figure 6.1 Most occurring words in Dataset



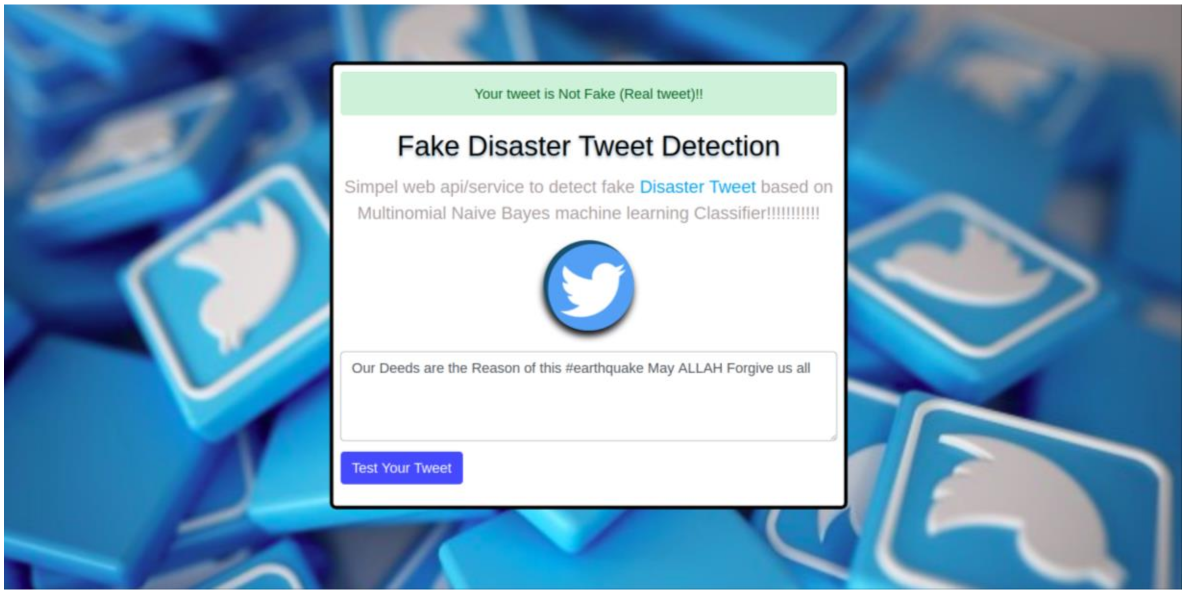
	id	keyword	location	text	target
0	1	NaN	NaN	Our Deeds are the Reason of this #earthquake M...	1
1	4	NaN	NaN	Forest fire near La Ronge Sask. Canada	1
2	5	NaN	NaN	All residents asked to 'shelter in place' are ...	1
3	6	NaN	NaN	13,000 people receive #wildfires evacuation or...	1
4	7	NaN	NaN	Just got sent this photo from Ruby #Alaska as ...	1

Figure 6.2 Dataset


Fake Disaster Tweet Detection

Simpel web api/service to detect fake Disaster Tweet based on Multinomial Naive Bayes machine learning Classifier!!!!!!!!!!

Figure 6.3 Webapp

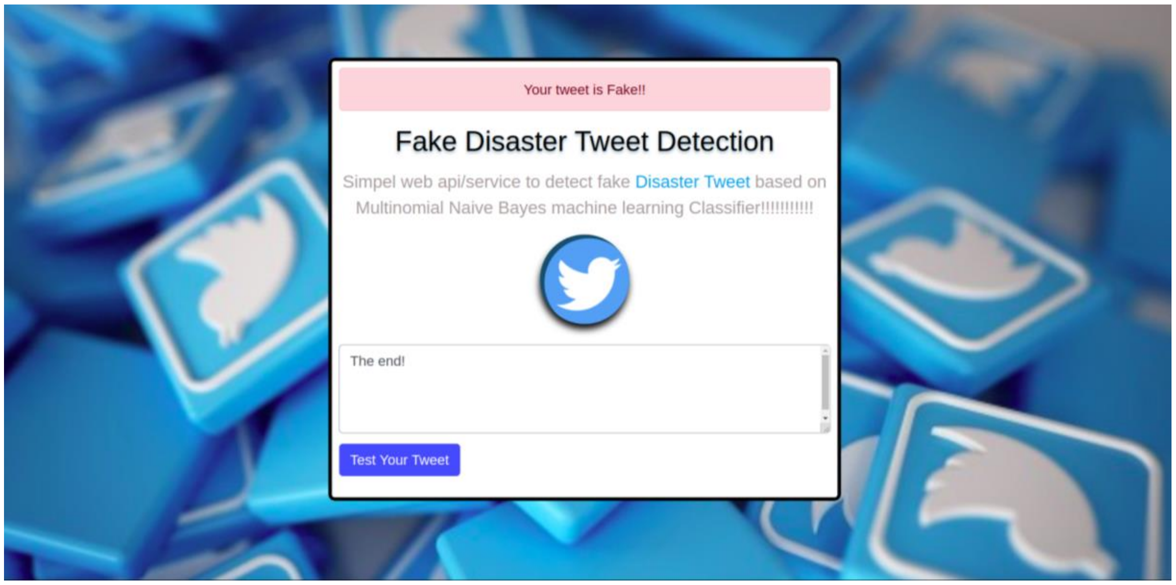

Your tweet is Not Fake (Real tweet)!!

Fake Disaster Tweet Detection

Simpel web api/service to detect fake Disaster Tweet based on Multinomial Naive Bayes machine learning Classifier!!!!!!!!!!

Our Deeds are the Reason of this #earthquake May ALLAH Forgive us all

Figure 6.4 Detecting Real Tweet


Your tweet is Fake!!

Fake Disaster Tweet Detection

Simpel web api/service to detect fake Disaster Tweet based on Multinomial Naive Bayes machine learning Classifier!!!!!!!!!!

The end!

Figure 6.5 Detecting Fake Tweet

Chapter 7: System Testing

7. System Testing

This project is made on Jupyter Notebook which is a part of Anaconda Navigator. This project ran successful on Jupyter Notebook. The dataset was successfully loaded into the notebook. All the extra python packages which were required for project completion also loaded into the notebook. The model is also deployed successfully using html, CSS, python and flask.

Machine learning model is evaluated we normally use classification accuracy which is number of correct predictions divided by total number of predictions.

$$Accuracy = \frac{Number\ of\ Correct\ predictions}{Total\ number\ of\ predictions\ made}$$

This accuracy measuring technique works well when there are equal number of samples in dataset belonging to each class. The accuracy score on test data is 77.977%. average recall value is 0.775 and average precision score is 0.775. Precision is used to calculate number of correct positive predictions made by the model. Recall is used to calculate number of correct positive prediction made out of all the positive predictions that could have been made.

- **Precision** = True Positives / (True Positives + False Positives)
- **Recall** = True Positives / (True Positives + False Negatives)

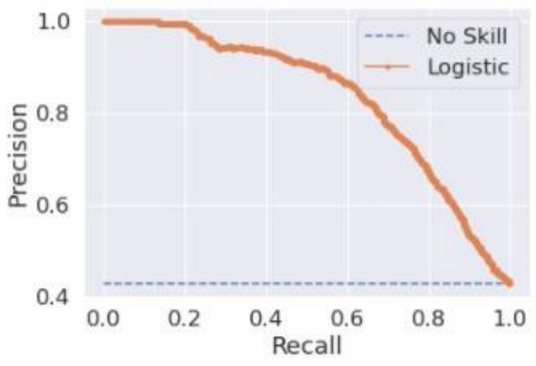


Figure 7.1 Precision-Recall Curve

Chapter 8: Conclusion

8. Conclusion

In this project only one classification algorithm is used which is Multinomial Naïve Bayes. First the preprocessing is done in the dataset which include removal of punctuations, then removal of URLs, digits, non-alphabets, contractions, then tokenization and removing Stopwords and removing Unicode. Then lemmatization is done on dataset. After preprocessing Countvectorizer is used to convert text data into numerical data as the classifier only works for numerical data. The dataset is then split into 70% training data and 30% test data. The accuracy score on test data is 77.977%. average recall value is 0.775 and average f1 score is 0.775.

8.1 Future Scope

In the future, some other classification algorithm can also be tried on this dataset like KNN, Support vector machine (SVM), Logistic Regression, even Deep learning algorithm can also be used which gives very high accuracy. Vectorizing can be done using other methods like word2vec, Tf-Idf vectorizer etc.

Chapter 9: Bibliography

9. Bibliography

1. M. T. Niles, B. F. Emery, A. J. Reagan, P. S. Dodds, and C. M. Danforth. Social media usagepatterns during natural hazards.PLOS ONE, 14(2):1–16, 02 2019

2. Bakhteev, O., Ogaltsov, A., & Ostroukhov, P. (2020). Fake News Spreader Detection using Neural Tweet Aggregation. CLEF.

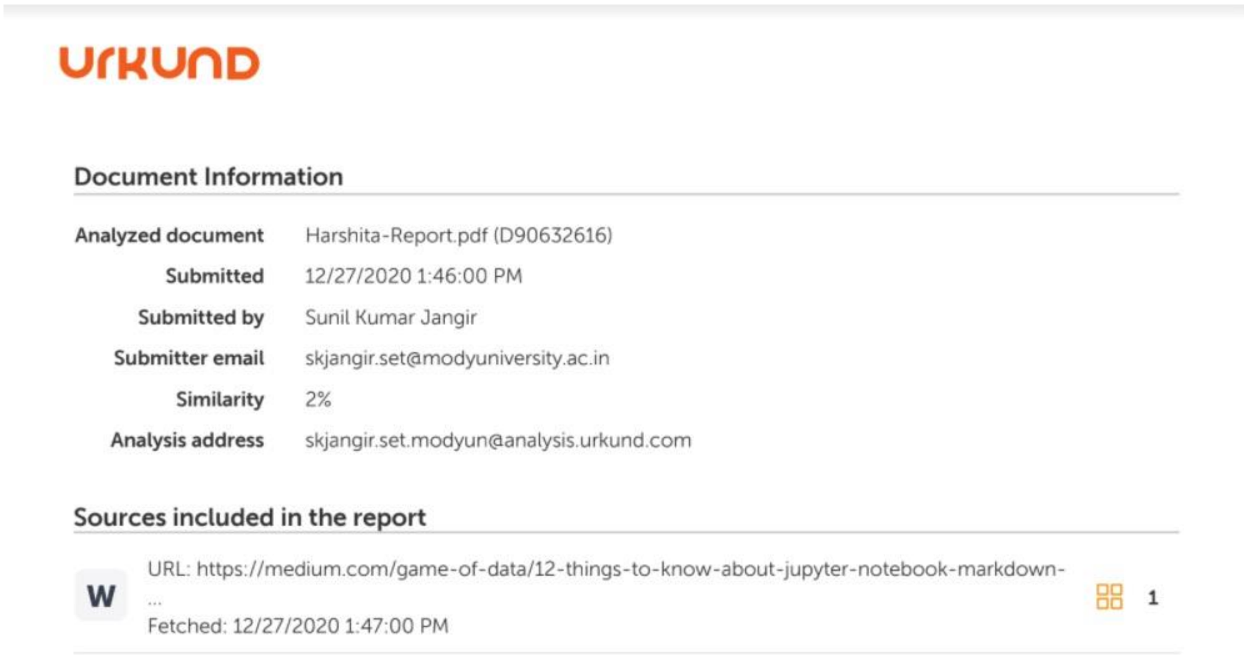
3. G. Burel and H. Alani. Crisis Event Extraction Service (CREES) - Automatic Detection andClassification of Crisis-related Content on Social Media. In15th International Conference onInformation Systems for Crisis Response and Management (ISCRAM), Rochester, NY, USA,May 2018.

4. C. Caragea, A. Silvescu, and A. H. Tapia. Identifying informative messages in disaster eventsusing convolutional neural networks. In13th International Conference on Information Systemsfor Crisis Response and Management (ISCRAM), Rio de Janeiro, Brazil, May 2016.

5. Hamdi, T., Slimi, H., Bounhas, I., & Slimani, Y. (2020, January). A Hybrid Approach for Fake News Detection in Twitter Based on User Features and Graph Embedding. In *International Conference on Distributed Computing and Internet Technology* (pp. 266-280). Springer, Cham.

6. Adam Geitgey (2018,Jul,18). *Natural Language Processing is Fun* [Online]. Available: https://medium.com/@ageitgey/natural-language-processing-is-fun-9a0bff37854e.

7. Srishti Sawla. (2018,Jun,9). *Introduction to Naive Bayes for Classification* [Online]. Available:https://medium.com/@srishtisawla/introduction-to-naive-bayes-for classification-baefefb43a2d.



URKUND

Document Information

Analyzed document	Harshita-Report.pdf (D90632616)
Submitted	12/27/2020 1:46:00 PM
Submitted by	Sunil Kumar Jangir
Submitter email	skjangir.set@modyuniversity.ac.in
Similarity	2%
Analysis address	skjangir.set.modyun@analysis.urkund.com

Sources included in the report

W	URL: https://medium.com/game-of-data/12-things-to-know-about-jupyter-notebook-markdown-... Fetched: 12/27/2020 1:47:00 PM	1