**Fake News Detection**

**A PROJECT REPORT**

*In partial fulfillment of the requirements for the award of the degree*

**BACHELOR OF TECHNOLOGY**

**IN**

**INFORMATION TECHNOLOGY**

*Under the guidance of*

**MR. MAHENDRA DATTA**

**BY**

**HIMANSHU KUMAR**

**DEEP CHOUDHURY**

**ARINDAM GHOSH**

**RAHUL KUMAR**

****

**SILIGURI INSTITUTE OF TECHNOLOGY**

**SILIG, WEST-BENGAL, INDIA**

**In association with**

**TITLE OF THE PROJECT : -**

**FAKE NEWS DETECTION:**

1. **PROJECT MEMBERS : -**

HIMANSHU KUMAR,

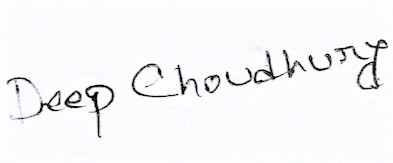
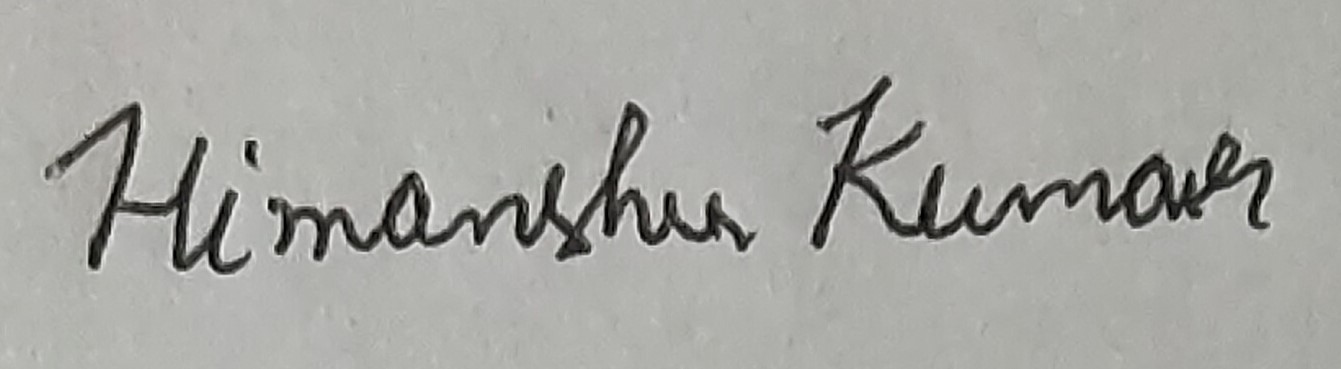
DEEP CHOUDHURY,

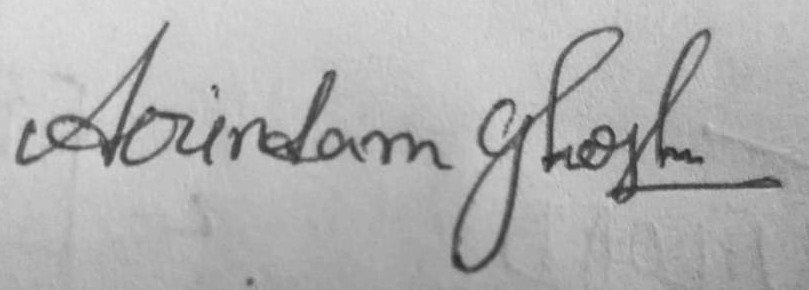
ARINDAM GHOSH

RAHUL KUMAR

1. **NAME OF THE GUIDE : -** MAHENDRA DATTA.
2. **PROJECT VERSION CONTROL: -**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Primary Author** | **Description Of Version** | **Date Completed** |
| Final | HIMANSHU KUMAR,  DEEP CHOUDHURY,  ARINDAM GHOSH,  RAHUL KUMAR | Project Report | **24-08-2021** |

****

****

Signature of the Student Signature of the Supervisor

Date: 15-08-2021 Date: 24-08-2021

For Office Use Only

****

**MAHENDRA DATTA**

**APPROVED**

Project proposal / Evaluator

### NOT APPROVED

**DECLARATION**

We hereby declare that the project work being presented in the project proposal entitled **“FAKE NEWS DETECTION”** in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF** **TECHNOLOGY** at **ARDENT COMPUTECH PVT LTD, SALTLAKE, KOLKATA, WEST BENGAL,** is an authentic work carried out under the guidance of **MR. MAHENDRA DATTA**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

**Date: 15-08-2021**

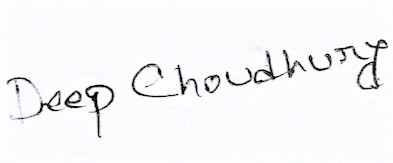
**Name of the Student**:

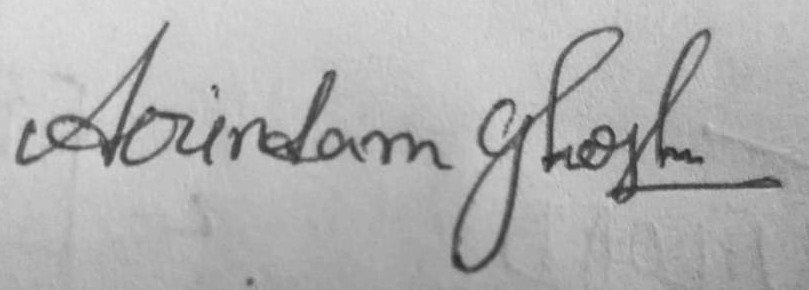
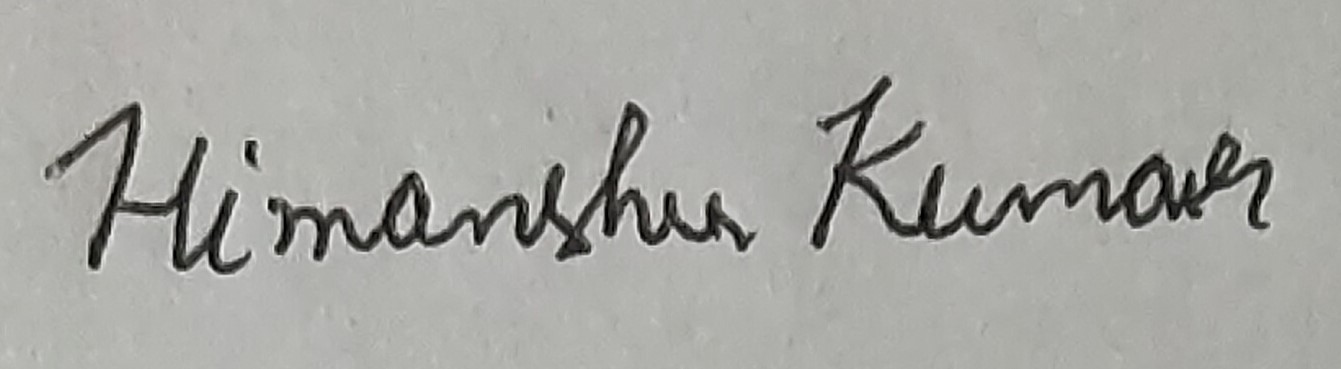
HIMANSHU KUMAR,

DEEP CHOUDHURY,

ARINDAM GHOSH

RAHUL KUMAR

**Signature of the student**:

****



**CERTIFICATE**

This is to certify that this proposal of minor project entitled **“CREDIT CARD FRAUD DETECTION ANALYSIS”** is a record of bonafide work, carried out by **ALOK RANJAN TIWARI, SHIVANI SINGH, SURENDRA YADAV, and SHIVAM KUMAR**under my guidance at **ARDENT COMPUTECH PVT LTD**. In my opinion, the report in its present form is in partial fulfillment ofthe requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** and as per regulations of the **ARDENT COMPUTECH PRIVATE LIMITED*.*** To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

**Guide / Supervisor**

****

------------------------------------------------

**MR. MAHENDRA DATTA**

Project Engineer

ARDENT COMPUTECH PVT. LTD.

**ACKNOWLEDGEMENT**

Success of any project depends largely on the encouragement and guidelines of many others. I take this sincere opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project work.

I would like to show our greatest appreciation to ***Mr. Mahendra Datta***, Project Engineer at ARDENT COMPUTECH PRIVATE LIMITED, Kolkata. I always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

Words are inadequate in offering our thanks to the other trainees, project assistants and other members at Ardent computech pvt.ltd. for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

**CONTENTS:-**

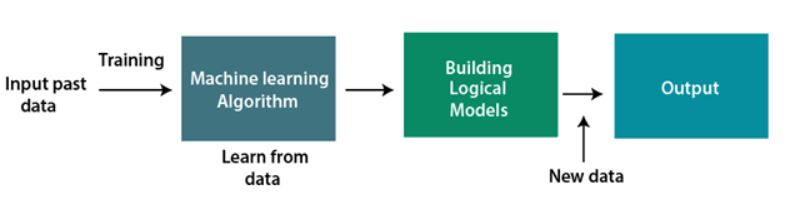
1. **ABSTRACT**
2. **LIST OF COMMOM MACHINE LEARNING ALGORITHM**
3. **PROBLEM STATEMENT**
4. **INTRODUCTION**
5. **DETAILS OF THE PROJECT**
6. **Data Collection**
7. **About The Data**
8. **SYSTEM REQUIREMENTS**
9. **Software Requirements**
10. **Hardware Requirements**
11. **MODULES USED IN THE PROJECT(with explanation)**
12. **CODES WITH OUTPUTS**
13. **CONCLUSION**
14. **UPLODED PROJECT AND DOCUMENT IN OPENSORSE AND GOOGLE COLAB LINK**
15. **BIBLIOGRAPHY**

**ABSTRACT**

**Machine Learning**

Machine learning is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning focuses on the development of computer programs** that can access data and use it learn for themselves.

The process of learning begins with observations or data, such as examples, direct experience, or instruction, in order to look for patterns in data and make better decisions in the future based on the examples that we provide. **The primary aim is to allow the computers learn automatically** without human intervention or assistance and adjust actions accordingly.



**Classification of Machine Learning**

1. Supervised Learning.
2. Unsupervised Learning.
3. Reinforcement Learning.
4. **Supervised Learning:**

Supervised learning, as the name indicates, has the presence of a supervisor as a teacher. Basically supervised learning is when we teach or train the machine using data that is well labeled. Which means some data is already tagged with the correct answer. After that, the machine is provided with a new set of examples(data) so that the supervised learning algorithm analyses the training data(set of training examples) and produces a correct outcome from labeled data.

**Types:-**

Regression

Logistic Regression

Classification

Naive Bayes Classifiers

K-NN (k nearest neighbors)

Decision Trees

Support Vector Machine

**Advantages:-**

Supervised learning allows collecting data and produces data output from previous experiences.

Helps to optimize performance criteria with the help of experience.

Supervised machine learning helps to solve various types of real-world computation problems.

**Disadvantages:-**

Classifying big data can be challenging.

Training for supervised learning needs a lot of computation time. So, it requires a lot of time.

1. **Unsupervised learning :**

Clustering: A clustering problem is where you want to discover the inherent groupings in the data, such as grouping customers by purchasing behavior.

Association: An association rule learning problem is where you want to discover rules that describe large portions of your data, such as people that buy X also tend to buy Y.

**Types of Unsupervised Learning:-**

Clustering

Exclusive (partitioning)

Agglomerative

Overlapping

Probabilistic

**Advantages:-**

Advantages of Unsupervised Learning

There are some reasons why we sometimes choose unsupervised learning in place of supervised learning. Here are some of the advantages:

Labeling of data demands a lot of manual work and expenses. Unsupervised learning solves the problem by learning the data and classifying it without any labels.

The labels can be added after the data has been classified which is much easier.

It is very helpful in finding patterns in data, which are not possible to find using normal methods.

Dimensionality reduction can be easily accomplished using unsupervised learning.

**Disadvantages:-**

Disadvantages of Unsupervised Learning

Now, let’s have a look at some cons of unsupervised learning algorithm:

The result might be less accurate as we do not have any input data to train from.

The model is learning from raw data without any prior knowledge.

It is also a time-consuming process. The learning phase of the algorithm might take a lot of time, as it analyses and calculates all possibilities.

1. **Reinforcement learning :**

Reinforcement learning is an area of Machine Learning. It is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behavior or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of a training dataset, it is bound to learn from its experience.

**Types of Reinforcement: There are two types of Reinforcement:**

**Positive –**

Positive Reinforcement is defined as when an event, occurs due to a particular behavior, increases the strength and the frequency of the behavior. In other words, it has a positive effect on behavior.

**Advantages of reinforcement learning are:**

Maximizes Performance

Sustain Change for a long period of time

Disadvantages of reinforcement learning:

Too much Reinforcement can lead to overload of states which can diminish the results

**Negative –**

Negative Reinforcement is defined as strengthening of a behavior because a negative condition is stopped or avoided.

**Advantages of reinforcement learning:**

Increases Behavior

Provide defiance to minimum standard of performance

Disadvantages of reinforcement learning:

It Only provides enough to meet up the minimum behavior

## LIST OF COMMON MACHINE LEARNING ALGORITHMS:

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

1. Linear Regression
2. Logistic Regression
3. Decision Tree
4. SVM
5. Naive Bayes
6. KNN (K-Nearest Neighbours)
7. K-Means
8. Random Forest
9. Dimensionality Reduction Algorithms.

There are also Gradient Boosting Algorithms:

1.GBM

2.XGBoost

3.Light GBM

4.CatBoost

## 1. Linear Regression

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation Y= a \*X + b.

The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life! The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

* Y – Dependent Variable
* a – Slope
* X – Independent variable
* b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Look at the below example. Here we have identified the best fit line having linear equation **y=0.2811x+13.9**. Now using this equation, we can find the weight, knowing the height of a person.



Linear Regression is mainly of two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent variable. And, Multiple Linear Regression(as the name suggests) is characterized by multiple (more than 1) independent variables. While finding the best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.

## 2. Logistic Regression

It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variables. In simple words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as **logit regression**. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

odds= p/ (1-p) = probability of event occurrence / probability of not event occurrences

ln(odds) = ln(p/(1-p))

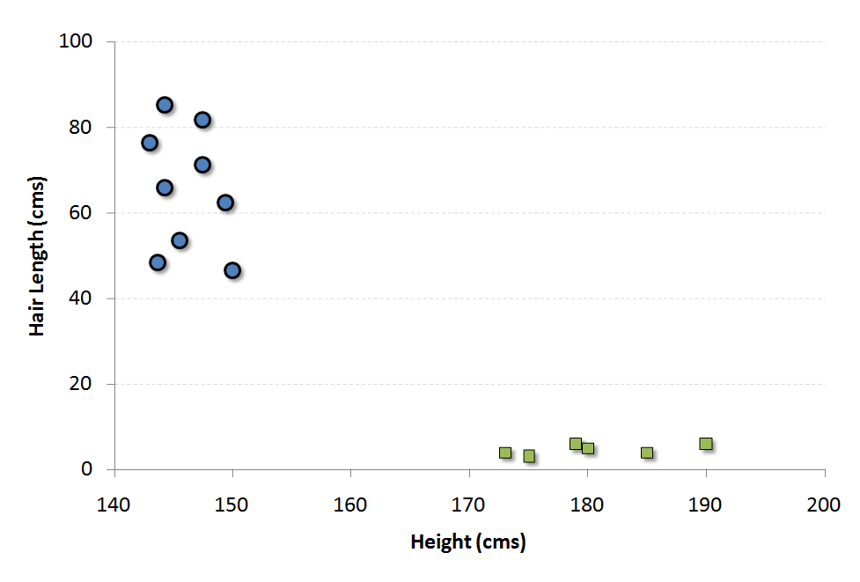
logit(p) = ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk

Above, p is the probability of presence of the characteristic of interest. It chooses parameters that maximize the likelihood of observing the sample values rather than that minimize the sum of squared errors (like in ordinary regression).

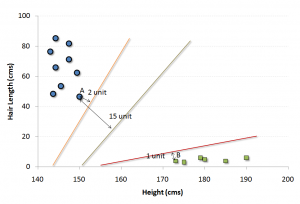
## 3. SVM (Support Vector Machine)

It is a classification method. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

For example, if we only had two features like Height and Hair length of an individual, we’d first plot these two variables in two-dimensional space where each point has two co-ordinates (these co-ordinates are known as **Support Vectors**)



Now, we will find some line that splits the data between the two differently classified groups of data. This will be the line such that the distances from the closest point in each of the two groups will be farthest away.



In the example shown above, the line which splits the data into two differently classified groups is the black line, since the two closest points are the farthest apart from the line. This line is our classifier. Then, depending on where the testing data lands on either side of the line, that’s what class we can classify the new data as.

## 4. kNN (k- Nearest Neighbors)

It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.

These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNN modelling.



KNN can easily be mapped to our real lives. If you want to learn about a person, of whom you have no information, you might like to find out about his close friends and the circles he moves in and gain access to his/her information!

**Things to consider before selecting kNN:**

* KNN is computationally expensive
* Variables should be normalized else higher range variables can bias it
* Works on pre-processing stage more before going for kNN like an outlier, noise removal

## 5. Random Forest

Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we’ve collection of decision trees (so known as “Forest”). To classify a new object based on attributes, each tree gives a classification and we say the tree “votes” for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is planted & grown as follows:

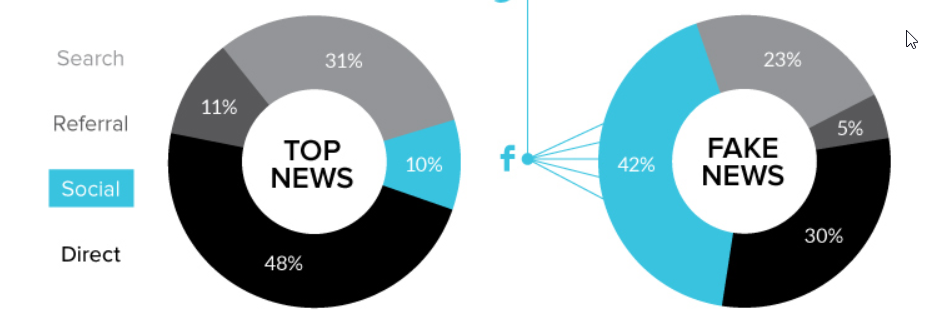
1. If the number of cases in the training set is N, then sample of N cases is taken at random but with replacement. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

**PROBLEM STATEMENT**

**FAKE NEWS DETECTION**

**INTRODUCTION**

## What is Fake News?



How much of what we read on social media and on supposedly “credible” news sites is trustworthy? It is extremely easy for anyone to post what they desire and although that can be acceptable, there is the notion of taking it a step too far, such as posting false information online in order to cause a panic, using lies to manipulate another person’s decision, or essentially anything else that can have lasting repercussions. There is so much information online that it is becoming impossible to decipher the true from the false. Thus, this leads to the problem of fake news

What is fake news? Fake news is the deliberate spread of misinformation via traditional news media or via social media. False information spreads extraordinarily fast. This is demonstrated by the fact that, when one fake news site is taken down, another will promptly take its place. In addition, fake news can become indistinguishable from accurate reporting since it spreads so fast. People can download articles from sites, share the information, re-share from others and by the end of the day the false information has gone so far from its original site that it becomes indistinguishable from real news

**DETAILS OF THE PROJECT**

1. **Data Collection.**

* There was 2 dataset

1. Fake.csv II) True.csv

required for this project was taken from *“kaggle.com”*.

* Kaggle allows users to find and publish datasets to solve challenges or web-based data science projects*.*

1. **About the Data.**

* The data-set consists of 23482 Rows and 4 Columns for Fake.csv and 21418 Rows and 4 Columns for True.csv.
* The description of each Columns is as follows:-

**Content**

1. Title: Consist news title
2. Text: Consist of the details news
3. Subject: Which type of news it is (eg. Political news , sports news , etc.)
4. Date: News Publishing date

**FOR ONLINE USE:**

**1.SOFTWARE REQUIREMENTS**

* **OS: Windows or linux**
* **Web: chrome**
* **Website google Colab (https://colab.research.google.com/)**
* **language python**

**FOR OFFLINE USE:**

**1.HARDWARE REQUIREMENTS**

* **RAM:4GB and higher**
* **processor :intel i3 and above**
* **hard disk:500GB: minimum**

**2.SOFTWARE REQUIREMENTS**

* **os: windows or linux**
* **python IDE:python 2.7.x and above**
* **jupyter notebook**
* **setup tools and pin to be installed for 3.6 and above**
* **language python**

**Modules used in the Project:**

* NumPy :- It is a general purpose array processing package.it provides a high performance multidimensional array object, and tools for working with these arrays .its also an efficient multidimensional container of generic data.
* Pandas :- It is the most popular python library that is used for data analysis.it provides highly optimized performance with back-end source code is purely written is **C** or **python .**subdivided into two parts it is :- **series and dataframe.**
* Matplotlib :- It is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environment across platforms .it tries to make easy things easy and hard things possible. you can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code.
* Seaborn :- It is a python data visualization library based on matplotlib.it provides a high level interface for drawing attractive and informative statistical graphics.it is closely integrated with pandas datastructures.
* Tokenization :- Tokenization is the process of turning sensitive data into nonsensitive data called "tokens" that can be used in a database or internal system without bringing it into scope. Tokenization can be used to secure sensitive data by replacing the original data with an unrelated value of the same length and format. The tokens are then sent to an organization’s internal systems for use, and the original data is stored in a secure token vault.
* Pad sequence:- Padding As we know all the neural networks needs to have the inputs that should be in similar shape and size. When we pre-process the texts and use the texts as an inputs for our Model. Note that not all the sequences have the same length, as we can say naturally some of the sequences are long in lengths and some are short. Where we know that we need to have the inputs with the same size, now here padding comes into picture. The inputs should be in same size at that time padding is necessary.
* Word Cloud:- Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. ... The dataset used for generating word cloud is collected from UCI Machine Learning Repository.
* Sequence Model:- Sequence models are the machine learning models that input or output sequences of data. Sequential data includes text streams, audio clips, video clips, time-series data and etc. Recurrent Neural Networks (RNNs) is a popular algorithm used in sequence models.

**ACTUAL CODES WITH OUTPUT**

**Importing necessary libraries**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

import nltk

import re

from wordcloud import WordCloud

Explanation:-

We import the numpy, pandas, matplotlib, seaborn, Word Cloud(provides high-level interface for attractive and informative statistical graphics).

**Importing models:**

from tensorflow.keras.preprocessing.text import Tokenizer

from tensorflow.keras.preprocessing.sequence import pad\_sequences

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense,Embedding,LSTM,Conv1D,MaxPool1D

from sklearn.model\_selection import train\_test\_split

from sklearn.metrics import classification\_report,accuracy\_score

Explanation:-

We import the models Tokenizer, pad\_sequence .

**Importing the Fake dataset**

fake= pd.read\_csv('https://raw.githubusercontent.com/ML-Deep-Learning/Fake\_Real\_news\_dataset/main/Fake.csv')

Explanation:-

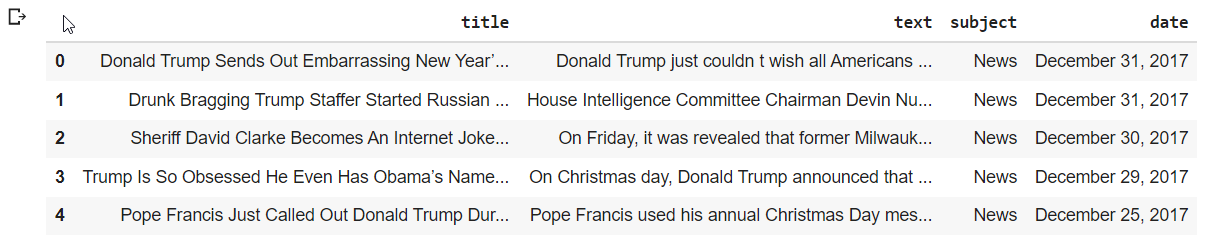
The csv file(dataset) is read using the “pd.read\_csv(‘filename or link)” command.

**Understanding the Fake Data**

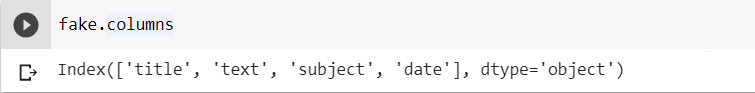
fake.head()

Explanation:-

head(), gives us a quick look at our data set.

Output :-

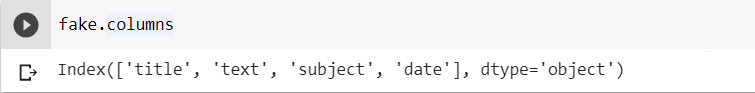
**Reading the Fake column subjects**



Explanation:-

* Column the Columns of the data set.

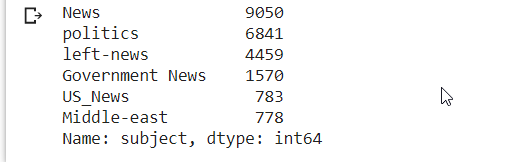
Output :-

****

**Count subject values**

fake['subject'].value\_counts()

Output :-



Explanation:-

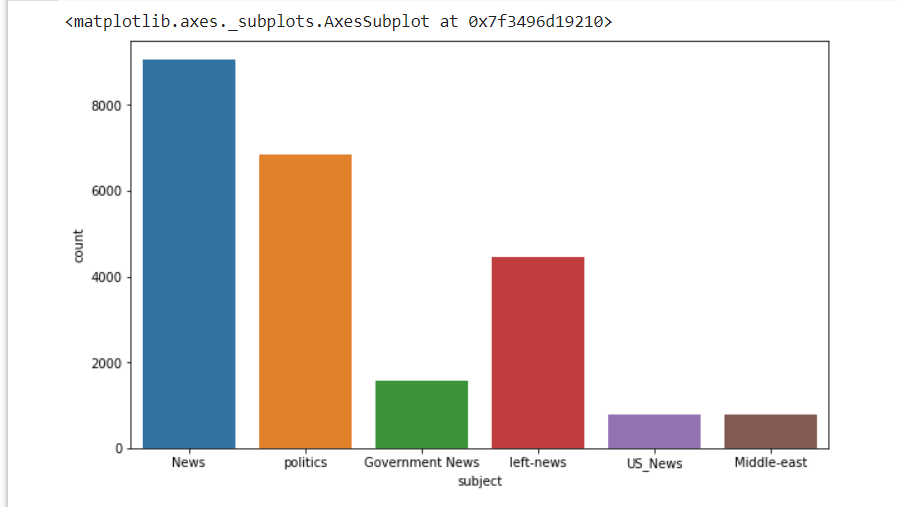
Count the Subject values

Plot in Figure

plt.figure(figsize=(10,6))

sns.countplot(x='subject',data=fake)

Output :-



Explanation :-

Print the data set in a bar graph figure

**Put the data in text**

text=' '.join(fake['text'].tolist())

Explanation:-

Put the text data as a list in the text

**Use WordCloud to**

wordcloud = WordCloud(width=2000,height=1100,margin=10).generate(text)

fig = plt.figure(figsize=(10,10))

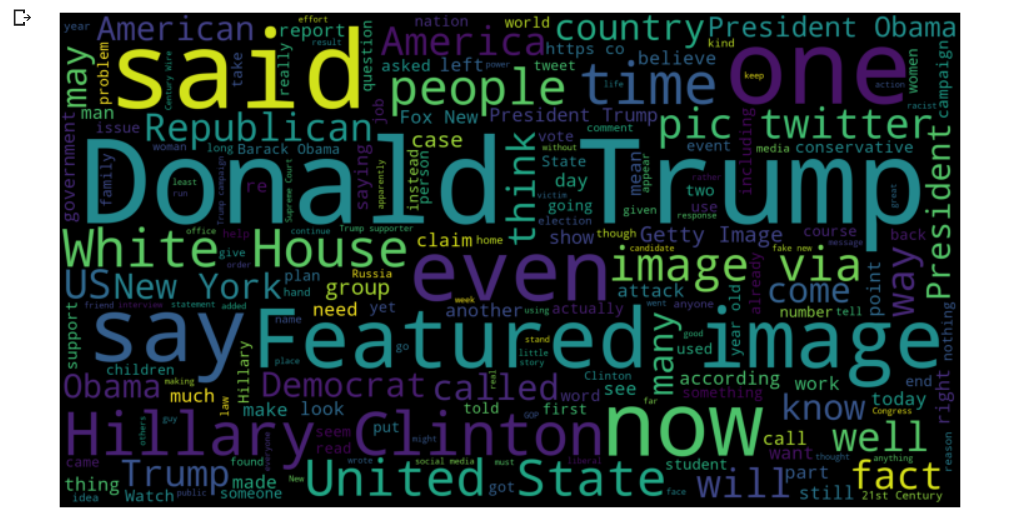
plt.imshow(wordcloud)

plt.axis('off')

plt.tight\_layout(pad=0)

plt.show()

**Output :-**



Explanation:-

Use wordcloud to show the text use in Fake data as a wordcloud banner.

**Importing the Fake data-set**

real = pd.read\_csv('https://raw.githubusercontent.com/ML-Deep-Learning/Fake\_Real\_news\_dataset/main/True.csv')

Explanation:-

The csv file(dataset) is read using the “pd.read\_csv(‘filename or link)” command.

**Put the data in text**

text=' '.join(fake['text'].tolist())

Explanation:-

Put the text data as a list in the text

**Use Wordcloud to**

wordcloud = WordCloud(width=2000,height=1100,margin=10).generate(text)

fig = plt.figure(figsize=(10,10))

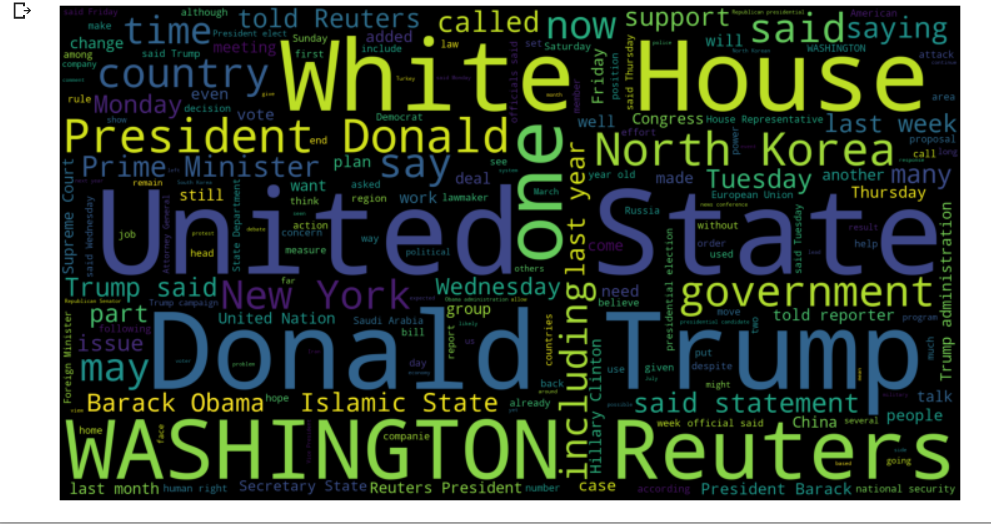
plt.imshow(wordcloud)

plt.axis('off')

plt.tight\_layout(pad=0)

plt.show()

**Output :-**

****

Explanation:-

Use word-cloud to show the text use in Fake data as a word-cloud banner.

**Detect the unknown publisher**

unknown\_publishers = []

for index , row in enumerate(real.text.values):

  try:

    record = row.split('-', maxsplit=1)

    record[1]

    assert(len(record[0])<120)

  except:

      unknown\_publishers.append(index)

Explanation:-

The above code will find try to find the unknown publisher data and add into the unknown\_publisher list.

**Print the Unknown Publisher length**

len(unknown\_publishers)

Output:-

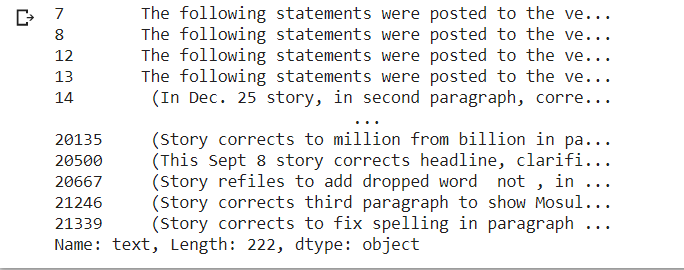
222

Explanation:-

Print the Unknown Publisher length

**Print the unknown publisher list as iloc style**

real.iloc[unknown\_publishers].text

Output:-

Explanation:-

We will print the unknown publisher list as the iloc style

**Separate the news as per publisher**

publisher = []

tmp\_text = []

for index , row in enumerate(real.text.values):

  if index in unknown\_publishers:

    tmp\_text.append(row)

    publisher.append('Unknown')

  else:

    record = row.split('-' , maxsplit=1)

    publisher.append(record[0].strip())

    tmp\_text.append(record[1].strip())

Explanation:-

Separate the news as fake and true as per the publisher is verified or not

**Store the data**

real['publisher']= publisher

real['text'] = tmp\_text

Explanation:-

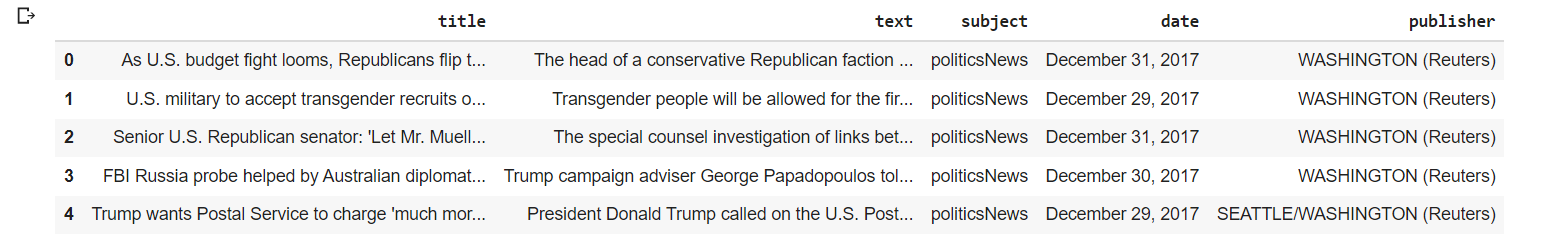
Store the publisher data in real at publisher sub-field and tmp\_text in real at text sub-field

**Understanding the True Data**

true.head()

Explanation:- head(), gives us a quick look at our data set.

Output :-

****

**Print the shape of the real data set**

real.shape

Output :-

(21417, 5)

Explanation:- Print the shape of the real data set

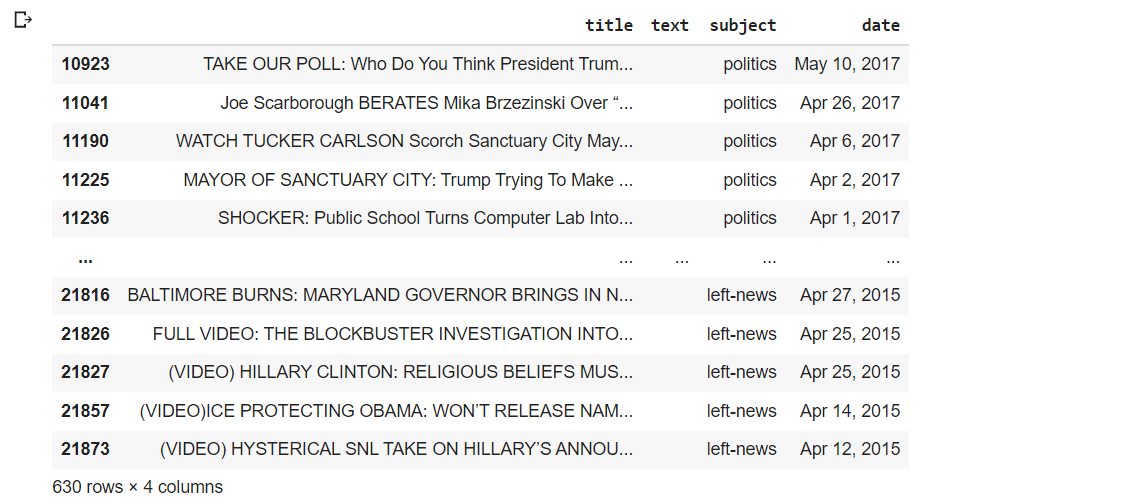
**Add the index of empty fake data**

empty\_fake\_index = [index for index, text in enumerate(fake.text.tolist()) if str(text).strip()==""]

Explanation:- Which fake data are empty their index will added to empty\_fake\_index list

**Print the fake list as iloc style**

fake.iloc[empty\_fake\_index]

Output:-

Explanation:-

We will print the fake list as the iloc style

**Add the real and fake data**

real['text'] = real['title'] + " " + real['text']

fake['text'] = fake['title'] + " " + fake['text']

Explanation:- Adding the fake and the real data in real and fake at text field.

**Make it in lower case**

real['text'] = real['text'].apply(lambda x: str(x).lower())

fake['text'] = fake['text'].apply(lambda x: str(x).lower())

Explanation:- make the text sub-field of real data set in lower case.

**Make fake and real in boolean**

real['class'] = 1

fake['class'] = 0

Explanation:- Set the real’s sub-field class as 1

And set the fake’s sub-field class as 0

**Store the real list in real**

real = real[['text', 'class' ]]

Explanation:- Store the real data list in real variable

**Store the fake list in fake**

fake = fake[['text', 'class' ]]

Explanation:- Store the fake data list in fake variable

**Add the real data in data**

data = real.append(fake, ignore\_index=True)

Explanation:- in data add the real data

**Install some modules**

!pip install spacy==2.2.3

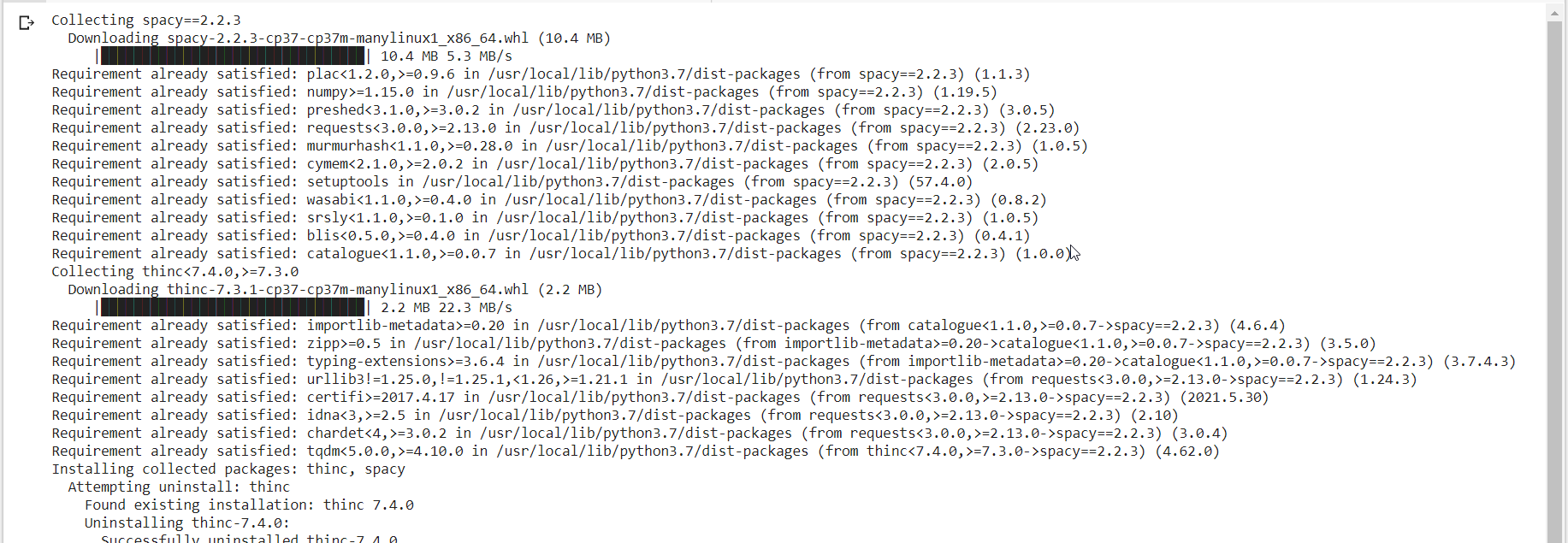
!python -m spacy download en\_core\_web\_sm

!pip install beautifulsoup4==4.9.1

!pip install textblob==0.15.3

!pip install git+https://github.com/laxmimerit/preprocess\_kgptalkie.git --upgrade --force-reinstall

Output :-



Explanation:- Install some python modules.

**Import preprocess\_kgptalkie**

import preprocess\_kgptalkie as ps

Explanation:-

import preprocess\_kgptalkie

**Removing the spacial Char**

data['text'] = data['text'].apply(lambda x: ps.remove\_special\_chars(x))

Explanation:-

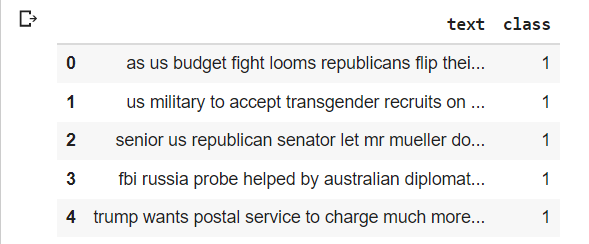
Remove the spacial character from the sub-field text in data

**Understanding the data Data**

data.head()

Explanation:- head(), gives us a quick look at our data set.

Output :-

****

**Import gensim**

import gensim

Explanation:-

Importing gensim

**Set the data class value**

y = data['class'].values

Explanation:-

Set the subfield class value of data in y

**Split the text**

X = [d.split() for d in data['text'].tolist()]

Explanation:-

Split the text and store in x list type variable

**Print the type**

type(X[0])

Output:-

List

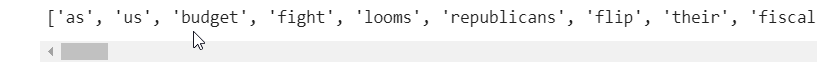
Explanation:-

Print the type of the variable x

**Print the element in x**

print(X[0])

Output:-



Explanation:-

Print the element of the variable x

**Convert word to vector**

DIM=100

w2v\_model = gensim.models.Word2Vec(sentences=X , size=DIM, window=10, min\_count=1)

Explanation:-

Word2vec used to map each word to a vector of typically several hundred elements

**Print the length of the vector data set w2v\_model**

len(w2v\_model.wv.vocab)

Output:-

231911

Explanation:-

Print the length of the vector data set of w2v\_model

**Print the vector data set of w2v\_model**

w2v\_model.wv.vocab

Output:-

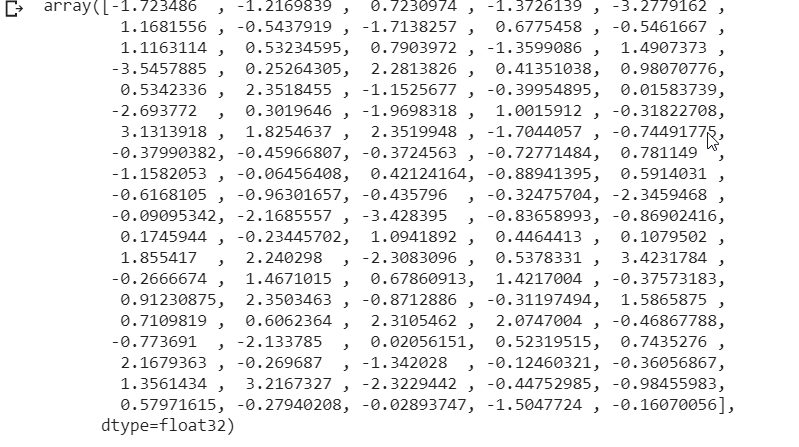
Explanation:-

Print the vector data set of w2v\_model

**Print the vector data set with word usa**

w2v\_model.wv['usa']

Output:-



Explanation:-

Print the vector data set which contains the word usa

**Print the vector data set which most similar to India**

w2v\_model.wv.most\_similar('india')

Output:-

[('pakistan', 0.732241153717041), ('malaysia', 0.7033170461654663), ('china', 0.6538301706314087), ('beijings', 0.6493744850158691), ('narendra', 0.6394171714782715), ('indias', 0.6343130469322205),

('modi', 0.6325240135192871),

('thailand', 0.6314331293106079), ('australia', 0.6197332739830017), ('indian', 0.6178133487701416)]

Explanation:-

Print the vector data set which is most similar with the word India

**Split the word with tokenizer**

tokeniser = Tokenizer()

tokeniser.fit\_on\_texts(X)

Explanation:-

Split the words based on regular expression

**Set the text to sequences to**

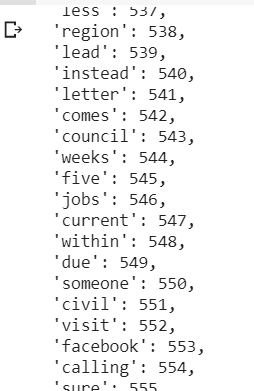
X= tokeniser.texts\_to\_sequences(X)

Explanation:-

Set the text to sequences to x

**Print the word\_index**

tokeniser.word\_index

Output:-

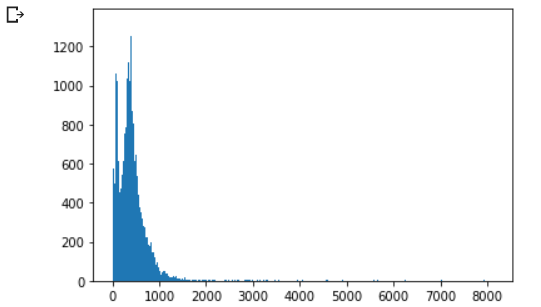
Explanation:- Print the word\_index of tokeniser

**Plot a histogram of x (**tokeniser.texts\_to\_sequences(X)**)**

plt.hist([len(x) for x in X], bins= 700)

plt.show()

Output:-



Explanation:-

Show or plot the data of x in a histogram

**Add a numpy array contains the length of x**

nos = np.array([len(x) for x in X])

len(nos[nos>1000])

Output:-

1584

Explanation:-

Add a numpy array contains the length of length of x in nos and put the count the of len of nos where nos is greater than 1000

**Set the preferred length to x**

maxlen = 1000

X = pad\_sequences(X, maxlen=maxlen)

Explanation:-

Set a preffered length which longer than x observed sequences to x.

**Print the length of x at index 101**

len(X[101])

Output:-

1000

Explanation:-

Print the length of x in index 101

**Set the tokeniser word\_index to variable**

vocab\_size = len(tokeniser.word\_index) + 1

vocab = tokeniser.word\_index

Explanation:-

Set the length of the tokeniser.word\_index +1 in vocab\_size and

Tokeniser.word\_index in vocab.

**Store the vector of model in an array**

def get\_weight\_matrix(model):

  weight\_matrix = np.zeros((vocab\_size,DIM))

  for word, i in vocab.items():

    weight\_matrix[i] = model.wv[word]

  return weight\_matrix

Explanation:- Store the vector of model in an array weight\_matrix

**Store the matrix**

embedding\_vectors = get\_weight\_matrix(w2v\_model)

Explanation:-

Store the get\_weight\_matrix(w2v\_model) in embedding\_vectors.

**Shape the matrix**

embedding\_vectors.shape

Output:-

(231912, 100)

Explanation:-

Shape the matrix with the help of .shape method.

**Compile the model**

model = Sequential()

model.add(Embedding(vocab\_size, output\_dim=DIM, weights = [embedding\_vectors],input\_length=maxlen,trainable= True))

model.add(LSTM(units=128))

model.add(Dense(1, activation='sigmoid'))

model.compile(optimizer='adam',loss= 'binary\_crossentropy', metrics='accuracy')

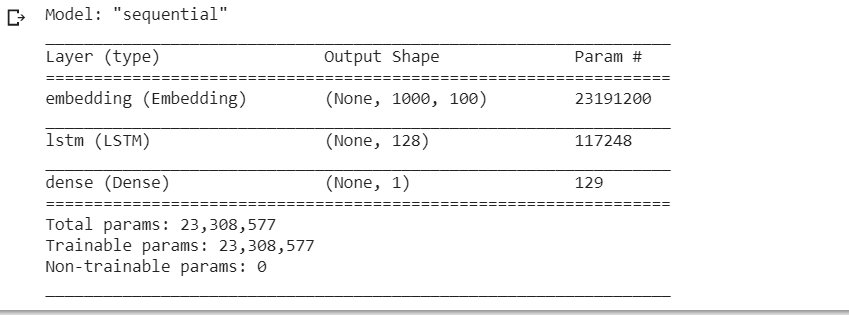
Explanation:-

Compile the model

**Print the model summary**

model.summary()

Output:-



Explanation:-

the summary of the trained model

**Train the x and y of the given data set**

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X,y)

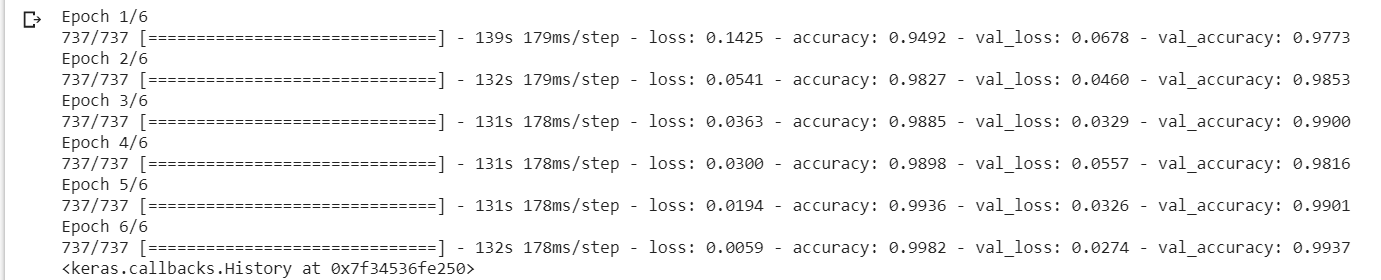
Explanation:-

Train the x and y from the given data set by just simply spiting them

**Fit the model**

model.fit(X\_train,y\_train,validation\_split=0.3, epochs=6)

Output:-



Explanation:-

Fit the given the model

**Find the model prediction**

y\_pred = (model.predict(X\_test) >=0.5).astype(int)

Explanation:-

Predict the model from x\_test as a int type and store in y\_predict

**Print the accuracy score**

accuracy\_score(y\_test , y\_pred)

Output:-

0.9923385300668152

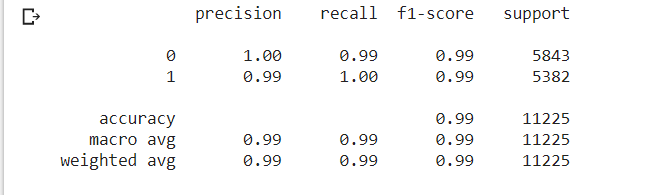
Explanation:-

Printing the accuracy score of the trained model

**Print the Classification Result**

print(classification\_report(y\_test,y\_pred))

Output:-



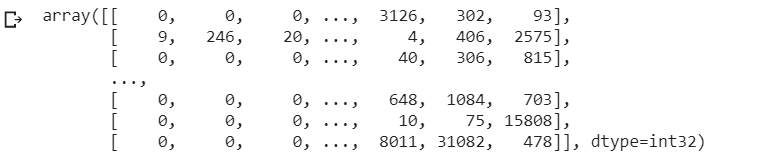
Explanation:-

Printing the Classification report of the trained model

**Print the X\_test result**

X\_test

Output:-



Explanation:-

Printing the X\_test of the trained model

**Example::**

x=['Taliban declare war is over as Afghan President Ashraf Ghani, top diplomats flee Kabul']

x= tokeniser.texts\_to\_sequences(x)

x= pad\_sequences(x, maxlen=maxlen)

(model.predict(x) >=0.5).astype(int)

Output:-

array([[0]])

Explanation:-

We put a news which which is false in x=[ ‘ YOUR NEWS’] and we get the output array([[0]]) means false as per the example news.

**Status: Success**

**CONCLUSION**

*The task of classifying news manually requires in-depth knowledge of the domain and expertise to identify anomalies in the text. In this research, we discussed the problem of classifying fake news articles using machine learning models and ensemble techniques. The data we used in our work is collected from the World Wide Web and contains news articles from various domains to cover most of the news rather than specifically classifying political news. The primary aim of the research is to identify patterns in text that differentiate fake articles from true news. We extracted different textual features from the articles using an LIWC tool and used the feature set as an input to the models. The learning models were trained and parameter-tuned to obtain optimal accuracy. Some models have achieved comparatively higher accuracy than others. We used multiple performance metrics to compare the results for each algorithm. The ensemble learners have shown an overall better score on all performance metrics as compared to the individual learners*

---------------------------------------------------------------------------

**UPLODED PROJECT AND DOCUMENT IN OPENSORSE AND GOOGLE COLAB LINK**

* Opensourse Code link:

<https://github.com/ML-Deep-Learning/Fake_Real_ML_Project>

* Google Colab Code link: [https://colab.research.google.com/drive/1ejLV7IuZtWza3G\_PgqVatjyOXnB4ex9z#scrollTo=-RhMboGEE6oi](https://colab.research.google.com/drive/1ejLV7IuZtWza3G_PgqVatjyOXnB4ex9z" \l "scrollTo=-RhMboGEE6oi)

**BIBLIOGRAPHY**

* <https://www.tensorflow.org/api_docs/python/tf/all_symbols>
* [www.kaggle.com](http://www.kaggle.com/)
* <https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761>
* <https://scikit-learn.org/>
* [https://www.visualcapitalist.com/fake-news-problem-one-chart/](●https:/www.visualcapitalist.com/fake-news-problem-one-chart/)

**THANK YOU**