Fake News Detection in Social Media

Project Thesis/Software Project-1/Software Project-2

Submitted By

17-35518-3	Alam, Md. Iftakhar
17-35530-3	Bristy, Mushfika Jannat
17-35617-3	Hossain, Mosharaf
18-36072-1	Chowdhury, Md. Samiul Hauque

(Remove rows from the above table if you have less number of group members)

Department of Computer Science

Faculty of Science & IT

American International University Bangladesh

November 25, 2021

Declaration

We declare that this thesis is our original work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

(Please remove the signing blocks if you have less number of group members)

_____

Alam, Md. Iftakhar

17-35518-3

CSE

_____

Bristy, Mushfika Jannat

17-35530-3

CSE

_____

Hossain, Mosharaf

17-35617-3

CSE

_____

Chowdhury, Md. Samiul Hauque

18-36072-1

CSE

Approval

The thesis titled "Fake News Detection in Social Media" has been submitted to the following respected members of the board of examiners of the department of computer science in partial fulfilment of the requirements for the degree of Bachelor of Science in Computer Science on (date of defence) and has been accepted as satisfactory.

(Please remove the signing blocks if you do not have any external or co-supervisor)

_____

Md. Asiful Islam

Lecturer & Supervisor

Department of Computer Science

American International University-Bangladesh

_____

Md. Al-Amin

Lecturer & External

Department of Computer Science

American International University-Bangladesh

_____

Dr. Md. Mahbub Chowdhury Mishu

Assistant Professor & Head (Undergraduate)

Department of Computer Science

American International University-Bangladesh

_____

Professor Dr. Tafazzal Hossain

Dean

Faculty of Science & Information Technology

American International University-Bangladesh

_____

Dr. Carmen Z. Lamagna

Vice Chancellor

American International University-Bangladesh

Acknowledgement

We would to like start by thanking the almighty (May all the praises be upon him) for giving us this opportunity to through this thesis program.

We were assigned to do this thesis under the supervision of Md. Asiful Islam sir. We would like to express our humble gratitude towards him. He has taught and provided us with invaluable lessons about how a thesis is conducted, how can we constantly improve our research, etc. We would also like thank the numerous authors who have built the very foundation upon which our research stands on.

Abstract

With the widespread use of social media, an additional plague has emerged: the spread of fake news. There have been numerous researches conducted on Twitter and other social media sites. Despite being one of the most popular social media sites, Facebook has undergone microscopic studies to detect fake news. The limited research that has been done on Facebook has not been able to come up with a viable answer to this problem. We collected user interactions data of Facebook posts and utilized them on machine learning algorithms to obtain more reliable and accurate results. We were able to achieve a 96.414 percent accuracy by using classifier algorithms.

Table of Contents

List of Tables

List of Figures

Chapter 1: Introduction

1.1 Fake news detection in social media

The idea or concept behind fake news is not novel. It has been in existence before the inception of the internet but not at this massive scale. With an ever-increasing user base, social media platforms have become the go-to site for a quick injection of news or headlines from a variety of sources in a short period of time. According to a 2017 research, social media platforms were used by nearly two-thirds of adults in the United States to get their news [1].

"Articles or contents that are purposefully produced by publishers to deceive or mislead the readers" is a generally accepted definition of fake news [2]. On a theoretical level, we can divide fake news into three categories [3]. Those are as follows:

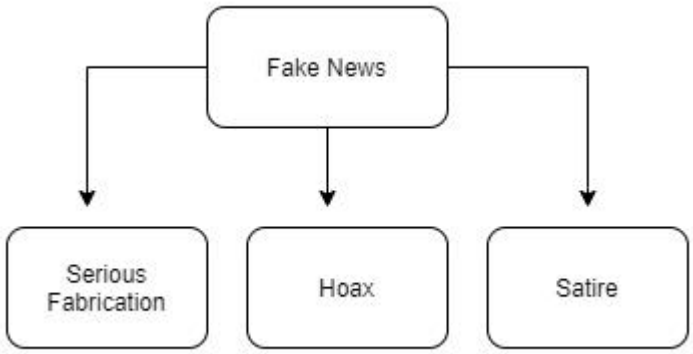


Fig. 1-1: Types of fake news.

1. Serious fabrications, i.e., deliberate fabrication or creation of an event that never took place. It can be further described as the news that was made out of thin air to aid the perpetrator's motive. This type of fake news has the most devastating effect out of the three. It is used to defame a particular person, organization, or a specific community.

2. Hoaxes define the news that contains misinformation about an event with the intent to be picked up by trustworthy publishers. These types of news rely on a current event to build their foundation. Their primary aim is to spread misinformation regarding a recent incident to create riots, protests, or social instability in society. Unlike serious fabrication, they require much more effort to successfully detect a piece of news.

3. Satires are created solely for the sake of amusement. This kind of news usually originates from the pages that were created with the sole intent of entertaining the masses. As such, when a post is shared from these types of pages due to the preconceived notion people have regarding them, they do not give them much importance, and often time the tone used in their post dismisses the seriousness and instead generates humor.

The major aim of this research is to detect serious fabrications and hoax news on Facebook. Due to the significance of these two types of news and the consequences that they can have on society, they were selected. The latter one or satires were eliminated since the weight of these types of news is much less than the other types.

The Dataset was built with the sole intent of distinguishing these two types of fake news and true news. Each row of the Dataset contains a post collected from a public page on Facebook and later on, by the usage of a thorough analysis of the attributes of each post (Number of comments, reactions, etc.) and a classifier, distinction between fake and true news were made.

The widespread dissemination of fake news can have severe consequences for people and society. Fake news undermines the credibility of news organizations. It has a negative impact on politics, as well as other fields such as health, science, sports, and financial markets [4,5]. In the 2016 US election, the widespread fake news mostly supported Donald Trump over Hilary Clinton [6]. It had a large-scale implication on the outcome of the election. Fake news has the potential to disrupt financial markets [7]. Fake news on medical science can be detrimental to public health, affecting health care and non-compliance with health requirements [8,9]. For example, on October 14, 2021, a Facebook post stated that the "Corona PCR' test' is implanting a microchip," [10] which was fake, but the fake news was widely accepted. "During this coronavirus pandemic, 'fake news is putting lives at risk": UNESCO**.** So, it is essential to recognize fake news on social media because it negatively impacts our personal and social lives.

Twitter and other social media platforms were used in the majority of the study on spotting fake news. A few publications focused on Facebook, but their Dataset was never disclosed.

The studies conducted in this domain can be divided into four major categories. Among them, two approaches (Text and sentiment analysis) heavily rely on language to distinguish fake news. Thus, failing to perform when the news contains media files. The latter two approaches (Feature-based) require a large quantity of data to function proficiently and are computationally expensive. The other one (Propagation path) needs an adequate number of characteristics.

Facebook is one of the most prominent online platforms in the world, and according to a study, Facebook is the most used platform for disseminating fake news [11]. Our research exclusively uses a dataset that was acquired from Facebook utilizing fact-checking websites as a reference, and we were able to reach an accuracy of over 90% using a classification model.

Chapter 2: Related work

There has been a surge in detecting fake or misleading news on social media platforms in recent times. Researchers used various methods ranging from text and sentiment analysis to the propagation and feature-based approaches to distinguish fake news on social media platforms, websites, blogs, and other places. The end goal of these studies is the same, but the methods used to get there are vastly different. The existing fake news detection methods can be divided into four main categories.

2.1 Text Analysis

One of the predominant methods is analyzing the text in the headlines or article body to find linguistic features such as n-gram matching lemmatized input using CoreNLP Lemmatizer [12]. Another method involves counting the number of pronouns while also looking at the writing style and checking the author's previous history, such as spreading false information [13]. To recognize fake news, [14] examines the text in the article body for qualities (n-grams, punctuation, psycholinguistic traits, readability, and syntax).
Text analysis is very dependent on the type of content; therefore, it cannot detect content devoid of textual elements or comprises audio, video, or photos, among other things. Photos, videos, and audio files can readily be fabricated to advance the perpetrators' desire to deceive readers to achieve a specific aim. It is further hindered by the nature of language, which allows it to detect only one type of language. The inherent nature of this strategy necessitates a far more efficient and powerful system capable of accurately detecting any fake news.

2.2 Propagation & Hybrid Approach

A novel way is based on user characteristics analysis who is accountable for propagating the fake news, e.g., number of friends, followers, number of previous posts, location, registration date, etc. This approach can detect fake news at the early propagation stage, e.g., within the first five minutes [15]. One significant issue is authors did not investigate whether these eight user characteristics are adequate for machine learning algorithms, furthermore whether the same approach is appropriate for Facebook since Facebook repudiates to share users' data due to data security issues.

Hybrid approaches can detect fake news, e.g., by analyzing text content, users' characteristics, propagation path, and element-based [16]. [17] analyzed both content and propagation information using a supervised classifier and got outperformed. These approaches are time-consuming; instead, machine learning algorithms may come up with malfunction results if one or more features fail in the initial stage. To overcome this problem, we counted the total comments count, share count, reaction counts of a Facebook post to predict whether the post is fake or authentic.

2.3 Features Based Approach

It is one of the leading strategies to detect fake news on social platforms. For diverse collections, feature-based methods were used to categorize problems and define themes.
Numerous features assess the credibility of news such as topic, message (message provide in the news), source, headline, body, media contents, propagation path, user's information (age, account verification, number of followers, number of friends, date of account creation, etc.) [16,18,19]. These features were built using data from Twitter and various news sources. 3-fold cross-validation and learning schemes were used to normalize the traits [16]. To find out the accuracy in detecting fake news, SVM, Binary Classification Problem, and Deep Network Models (CNN, RNN, Geometric Deep Learning) were utilized [16,18,19].
Though deep network models (CNN, Geometric Deep Learning, RNN) have been used to detect fake news in recent years, they have certain downsides, such as requiring a large quantity of data to get higher performance than other methods and being very computationally expensive due to large and complicated data models.

2.4 Sentiment based approach

Extraction of emotions or sentiments from Twitter [20] and other blogs such as Weibo [21] has been the subject of research. Using the EFN framework to detect fake news on Weibo, emotions were categorized and assessed based on several characteristics such as emotional intensity, category, and expression [21]. In the case of Twitter, the PHEME dataset was used for the analysis & LIWC (Linguistic Inquiry and Word Count) corpus to score a sentiment [20]. DNN (Deep Neural Networks) trained with the Flair Library was another way for detecting fake news using sentiment analysis [22].

Most recent research has relied on pre-built public datasets gathered from Twitter, although a few others have attempted to detect fake news on websites, blogs, and other platforms. Text and sentiment analysis depends on a specific language and fails to deliver when the news contains media files. The large number of characteristics required to attain increased accuracy is difficult to determine in the feature-based approach, making it computationally demanding. In the propagation method, Inadequate features during verdict a user as a fake news carrier.

Despite the fact that it is the most widely used social media platform regarding user count [23], few academics used Facebook to analyze fake news. The "Like" reaction on Facebook was utilized in a study to detect fake news [24]. Apart from "Like," Facebook included five more reactions to allow users to share their thoughts on a post [25].
As a result, a new horizon in detecting fake news on Facebook has opened up.

Chapter 3: Problem Definition

We'll go through the problem statement in more detail in this section, detecting fake news on social media. We assume D= {d1, d2, d3……, dn} is the number of total data that denotes a set of n news pieces. In addition to that, each news in dataset D has a label L = 0 or 1 associated with it. The letter L is assigned to each piece of news (dn). L(dn)=0 signifies false news, whereas L(dn)=1 signals true news. By mapping D to L, Model M will assess the Dataset to establish the label of fake and true news.

Chapter 4: Data collection method

Our data collection method was completed by a large number of steps. A picture is given below represents our data collection method flowchart which make our procedure more comprehensible.

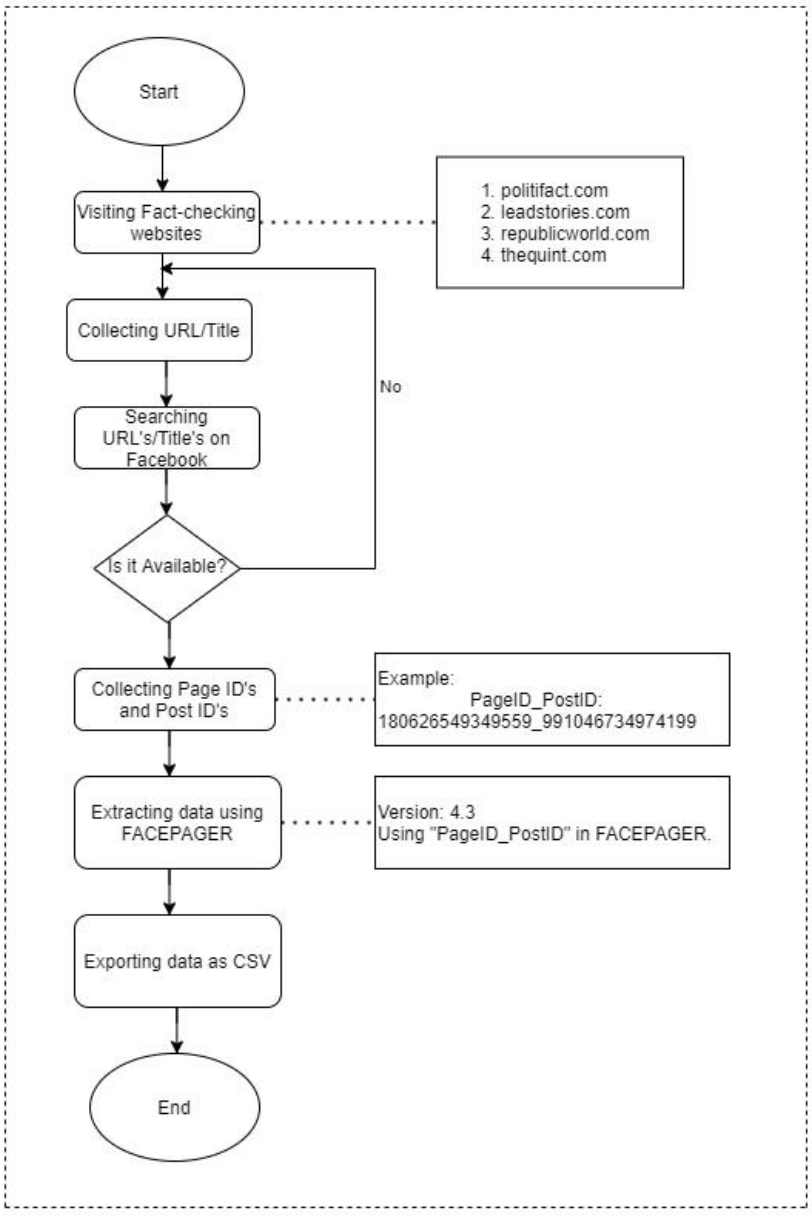


Fig. 4-1: Flowchart of data collection method.

4.1 Fact-checking and true news pages:

Fact-checking is one kind of journalism where news is categorized into true, fake, satire, or humor. Our data (fake news) was collected by using several fact-checking websites [26,27, 28, 29].

Social media plays a significant role in spreading fake news. Facebook is one of the prominent social media sites where fake news is disseminated faster than others. Wherefore fake news detection is being crucial for social media sites. Recently Facebook has been using fact-checking sites, e.g., independent fact-checker. According to press books, some renowned fact-checking websites are:

- Politifact
- Factchecker.org
- Snopes
- Truth Be Told
- Hoax-Slayer

Another reputed website named "Duke Reporters" shared a huge number of fact-checking sites and sorted by country. According to these sites, fact-checking websites are transparent while analyzing and verifying fake news.

Based on these sites, in our study, we collected fake news from the websites below:

1. Politifact
2. Factcheker.org
3. lead stories
4. The quint

Now when it comes to true news and news transparency, there is a lot for true news sites, and according to Forbes, they are:

- The New York times
- The Washington Post
- BBC
- The Economist
- Politico
- CNN
- TIME magazine
- NBC News
- USA Today
- ABC News
- Bloomberg business new

In our study, we collected true news from the Facebook pages of these reputed sites. Posts with a maximum number of engagements were focused while collecting information from this page.

4.2 Manual fact-checking:

In order to collect fake news, we have gone through the fact-checking websites and observed the articles that were published by the sites. The sites publish a number of news and verdict, whether they are fake, satire, hoax, fabricated, or partially fabricated.

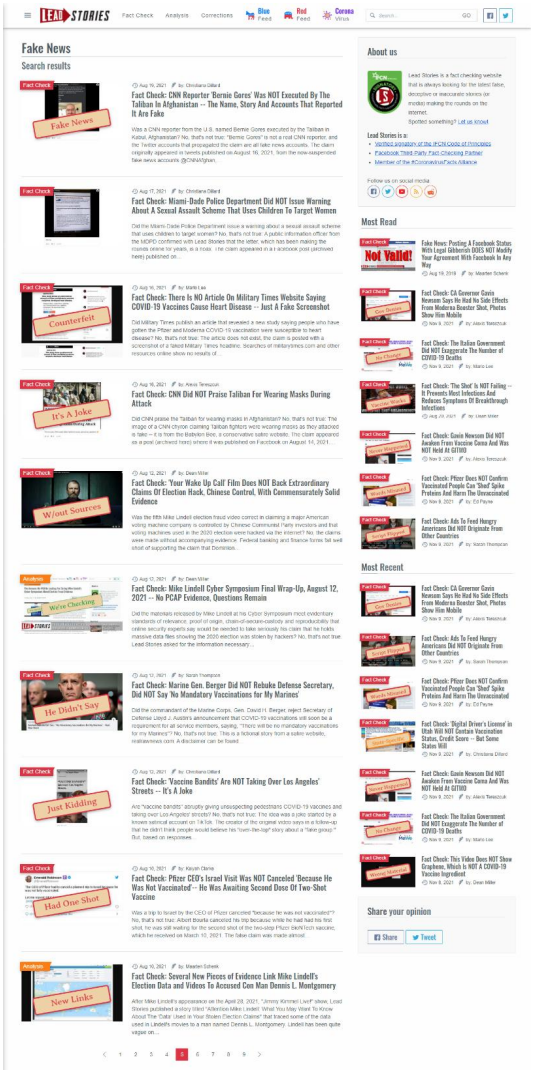


Fig. 4-2: Image of a fact checking website.

This is how fact checker sites publish their categorized article. Thenceforth we analyzed each of the articles and categorized whether it is satire, fake, hoax, or something else, and then we found whether it was published on Facebook or not.

4.3 Collecting URL:

Fig. 4-3: Article of a fake new.

This picture elucidates how the fact-checker website publishes fake news on its site. Here they are claiming that the post is totally fake, and they provided a link where those posts were published. In the picture, the post link is mentioned in the red box at the top. Consequently, we clicked on the link and completed further steps.

4.4 Searching URL's/tittle on Facebook:

After analyzing the articles and clicking/ collecting the link, we searched it on Facebook. Now, if the post is available, we cross-checked again whether it is labeled by Facebook or not. One significant point here is that we only collected posts published by the public pages as posts from users or groups cannot be extracted after the Cambridge Analytica scandal.

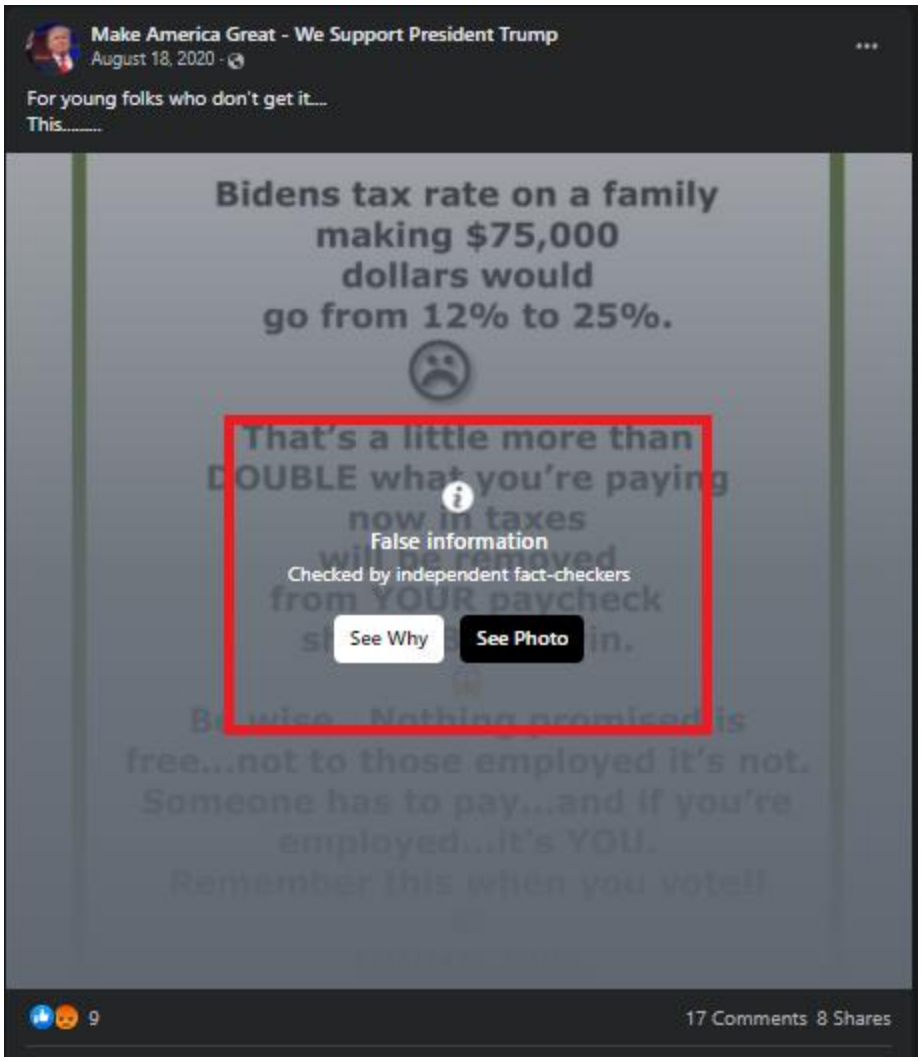


Fig. 4-4: Fake post on Facebook (Checked by independent fact-checker).

Here in the picture red box shows that the post was verified by a fake checker named independent fact-checker, and Facebook labeled them as false information.

4.5 Collecting Page ID and post ID:

After verifying posts on Facebook by an independent fact-checker, we collect the page id and post id in order to extract the data. The PageID_PostID format was used. We compiled a list of page and post id.

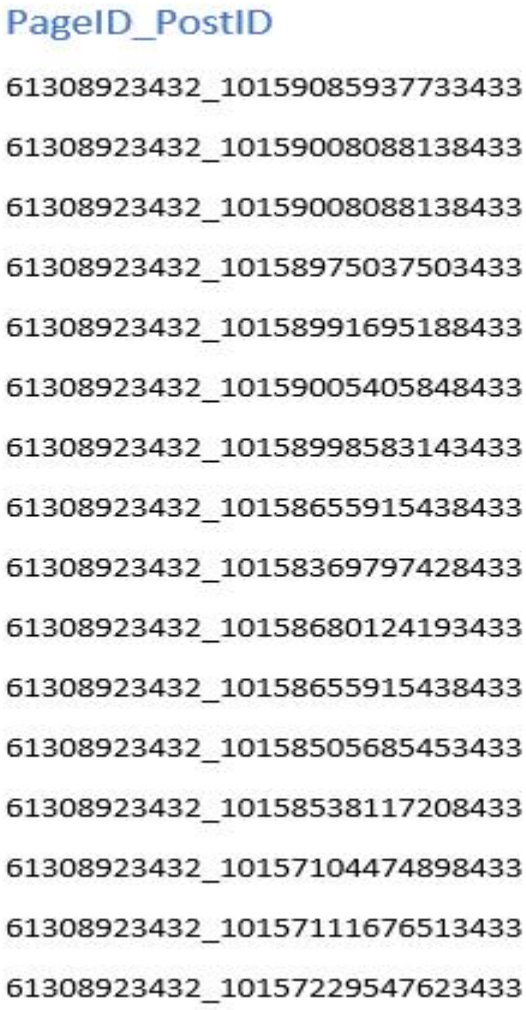


Fig. 4-5: The picture depicts the list we created to extract data.

4.6 Extracting data using Facepager:

Facepager is a web-based data retrieval application [30]. The tool developed by Jakob Jünger and Till Keyling (2019). The tool can retrieve data from social media sites such as Facebook, Twitter, YouTube, and Amazon. The tool is compatible with some presets that can be used to extract the data users want. We used Facepager (version 4.3) to retrieve data from Facebook.

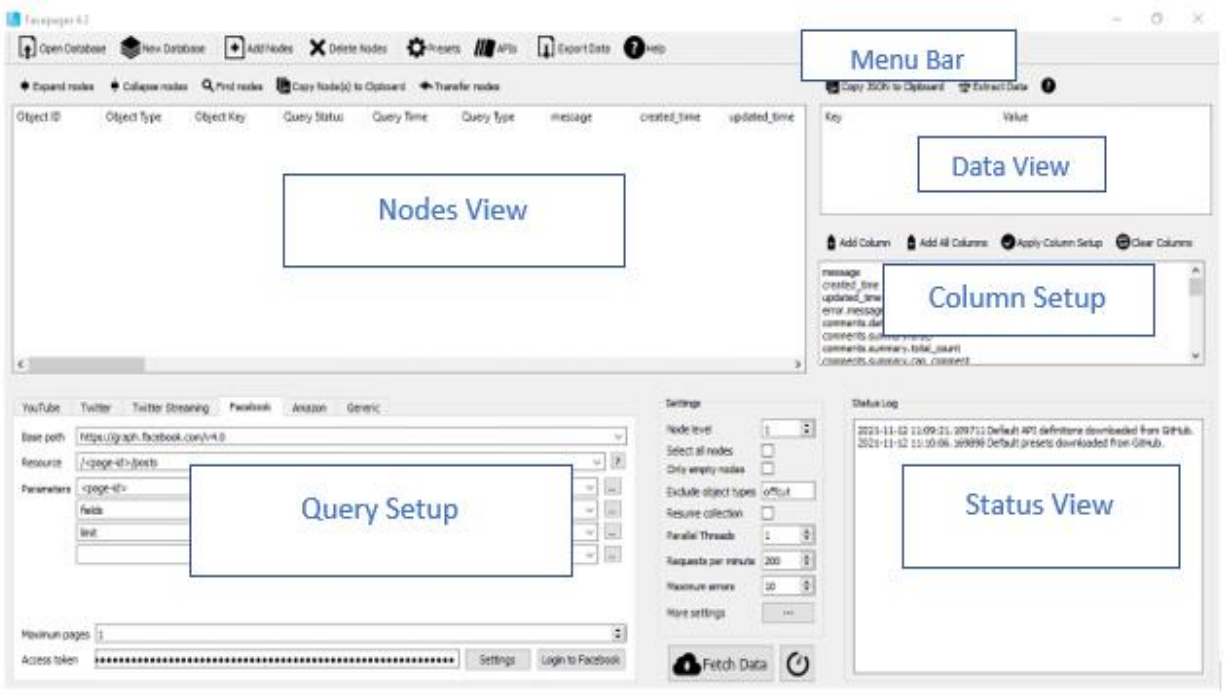


Fig. 4-6: The picture shows the basic layout of Facepager.

Menu Bar- The section of available operations with a drop-down menu.

Nodes View – The data collection objects of Facebook feed.

Data View -Displays all data associated with the selected object or row in Nodes View.

Column Setup – Users can set columns as their desired result.

Query View – It shows the begging point for retrieving data from the API.

Status View – It shows the error message of the query.

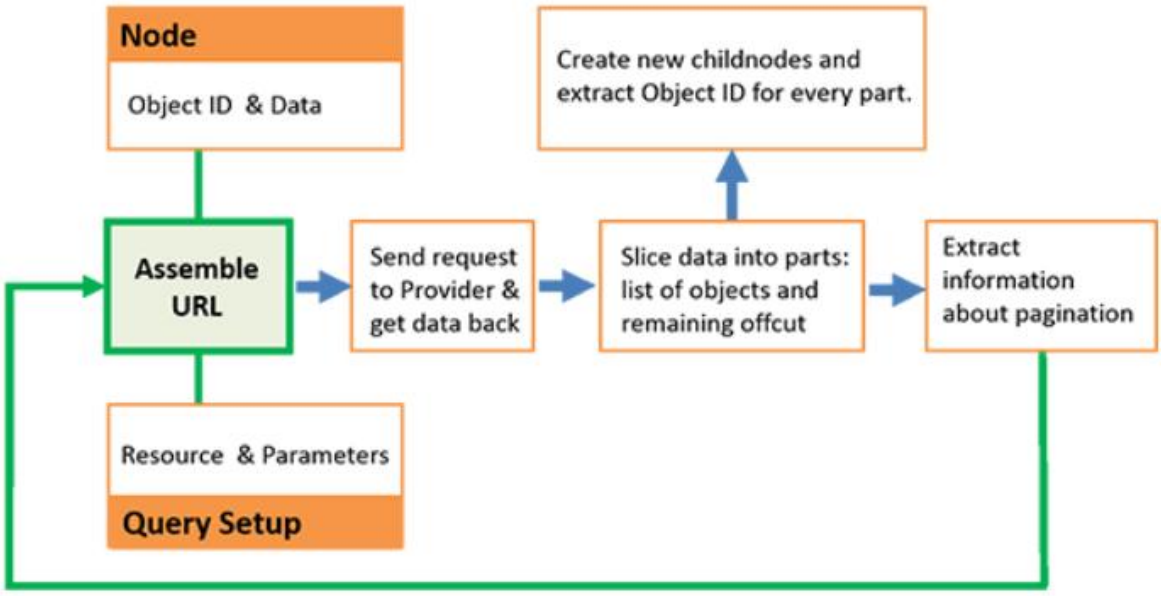


Fig. 4-7: Facepager Process Model.

To retrieve data, we choose a Facebook option from Facepager then we log in to access Facebook. After that, in Facepager, we created two databases: one for fake news and the other for true news.

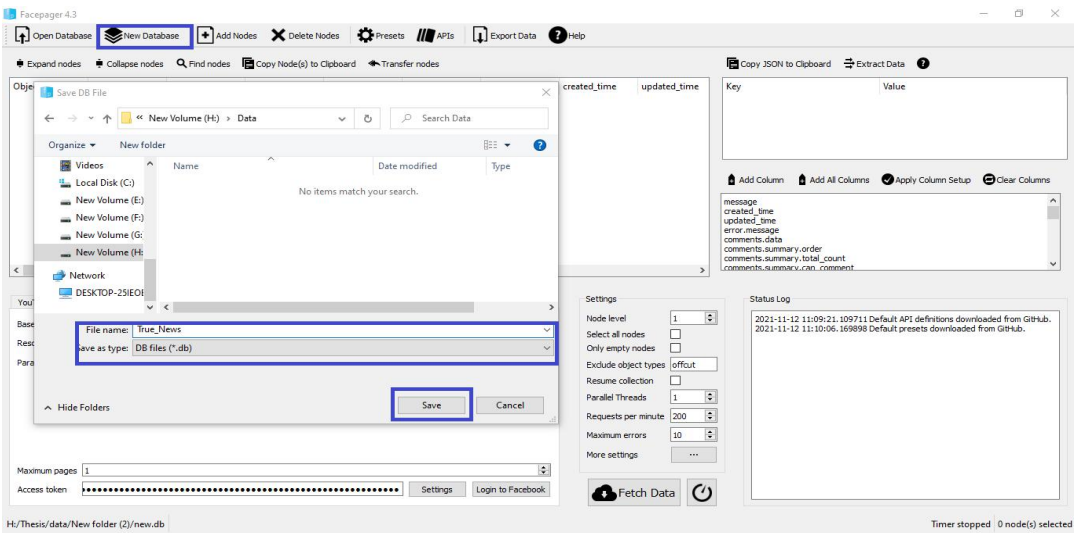


Fig. 4-8: Creating Database.

Then, in add notes, pageID_postID was given (as a starting point for further data collection), and presets were selected to obtain the desired data.

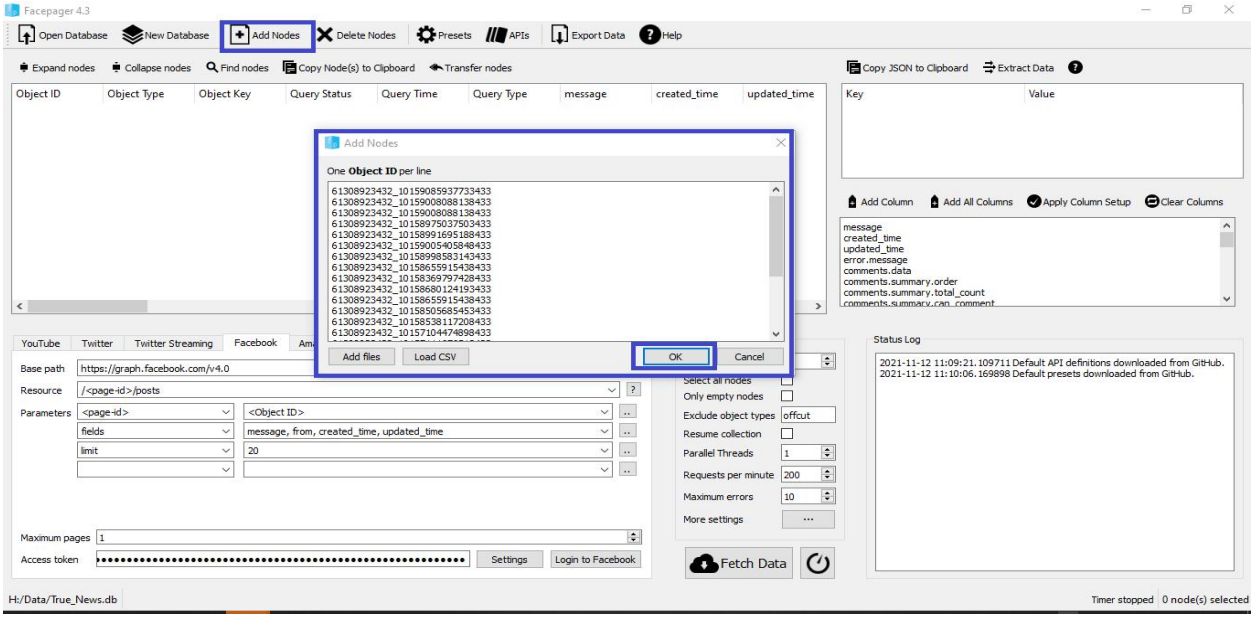


Fig. 4-9: Adding notes.

There are eight options in presets; we selected three options they are-Detail about posts, Get reactions and Get Facebook posts.

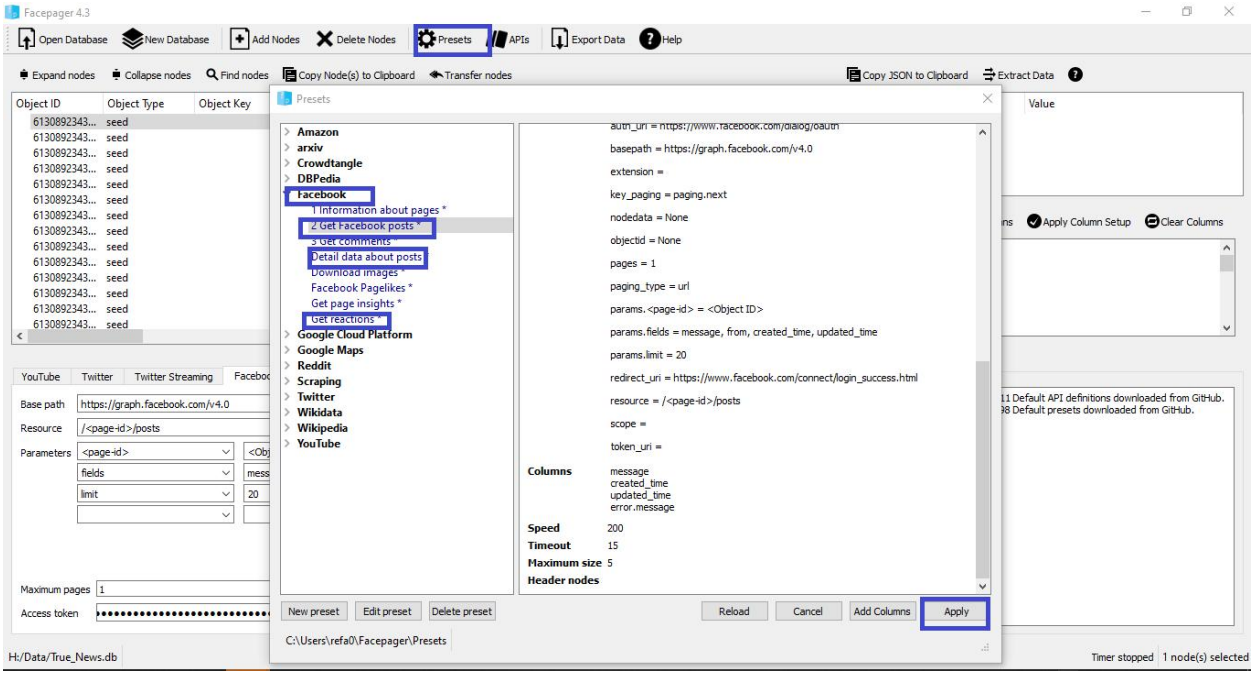


Fig. 4-10: Selecting Presets.

After configuring the presets, the fetch data button was pressed to extract data from a given address.

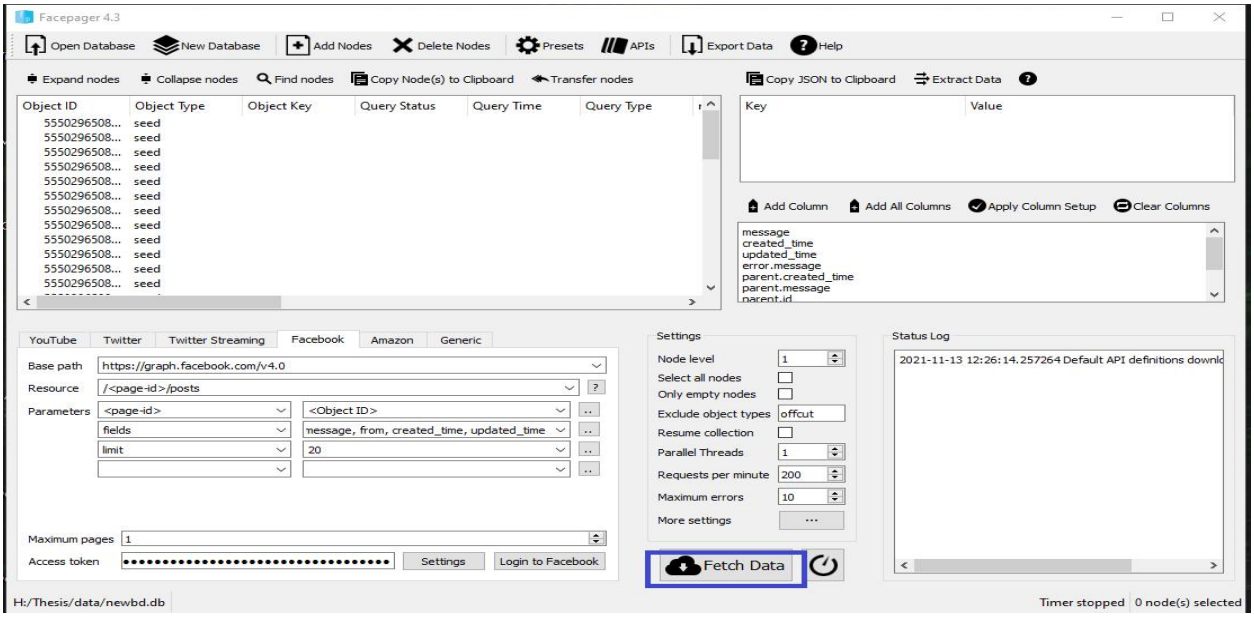


Fig. 4-11: Fetching Data.

The tools then return our data in JSON format. The JSON file that was collected is displayed inside the Data View tab.

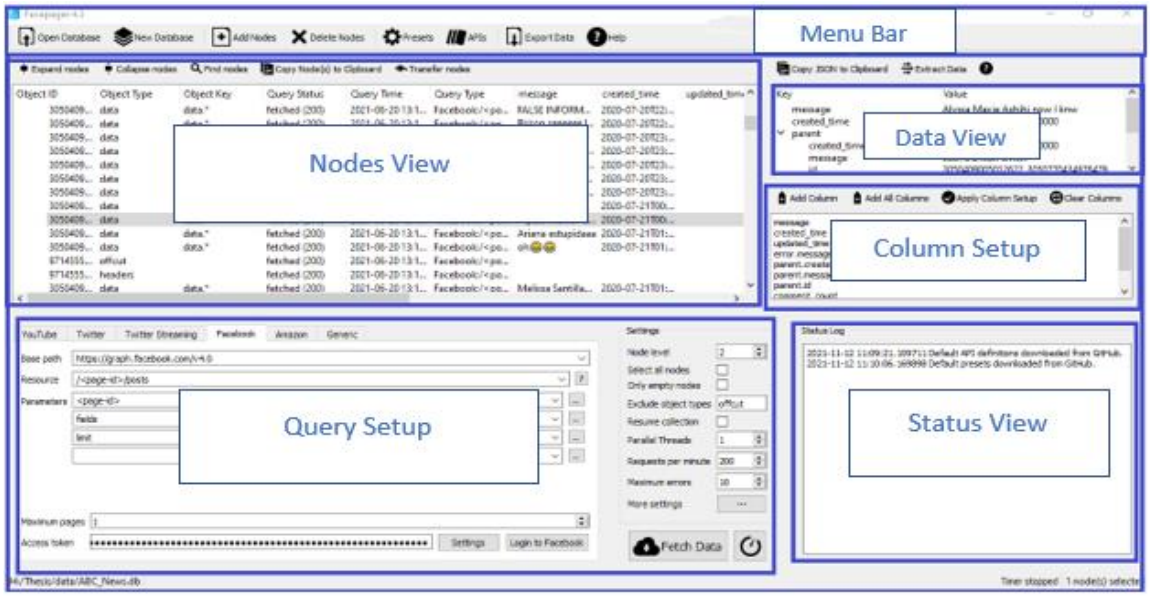


Fig. 4-12: Interface view after extracting data.

After extracting, we export data in two CSV files.

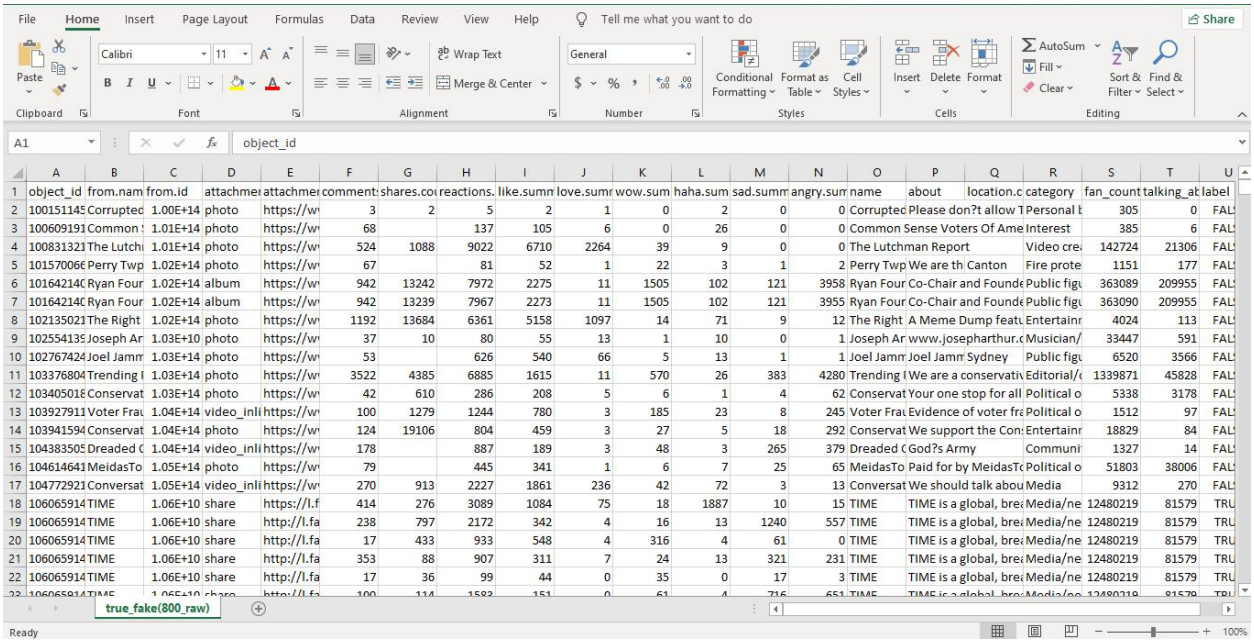


Fig. 4-13: CSV File.

Chapter 5: Dataset description

5.1 Dataset name

The name of our Dataset is "Dataset for fake news detection on Facebook." Our entire Dataset was collected from Facebook, and the intention was to detect a Facebook post, whether it is fake or not, using machine learning algorithms. The entire Dataset was stored in excel files from the beginning to the end.

5.2 Column Description

The columns name and descriptions are below-

1. **Page Information:**

- object_id: It contains the post id and page id, which were used in Facepager to extract data.
- name: Represents the page name.
- category: In this section, it represents page category, e.g., personal blog, public figure, musician/band, editorial, political, entertainment, interest, community center, media news, just for fun, podcast, etc.
- fan_count: It represents how many people are following this page.
- talking_about_count: It's an important feature as it shows the total number of shares, reactions, followers, etc.
- about: About row contains a short description of the page. On the about section page, the authority shares their agenda.

2. **Post Information:**

- form.id: Shows the post id.
- attachments.data.0.type: Elucidates whether the post contains an image or a video, or other type of content.
- attachments.data.0.url: Contains the URL of the post.
- comment: In this column, it shows the total number of comments in an individual post.
- shares.count: It has the total count of shares, e.g., how many people shared this post.
- reactions.summary.total_count: It is the summation of total reaction (like, haha, wow, angry, love, sad) count. This is a picture of a Facebook post. The marked red portion shows all the seven reactions by Facebook.

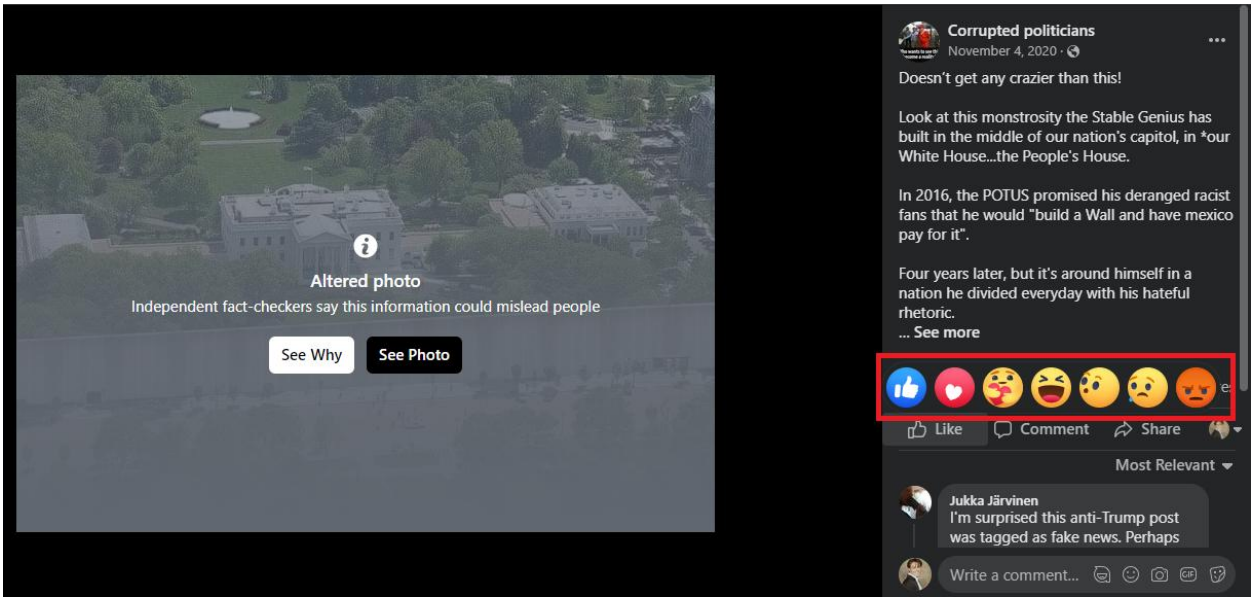


Fig. 5-1: Reactions of a Facebook post.

- like.summary.total_count: Represents like counts.
- love.summary.total_count: Represents share counts.
- wow.summary.total_count: Represents wow counts.
- haha.summary.total_count: Represents haha counts.
- sad.summary.total_count: Represents sad counts.
- angry.summary.total_count: Represents angry counts.
- label: It shows whether this row is containing fake news or true news.

3. Comment Information:

- Comment Id: The id represents a comment within a specific post.
- Parent: For comment replies, this is the comment that this is a reply to.

A screenshot below represents our initial Dataset which has 21 columns.

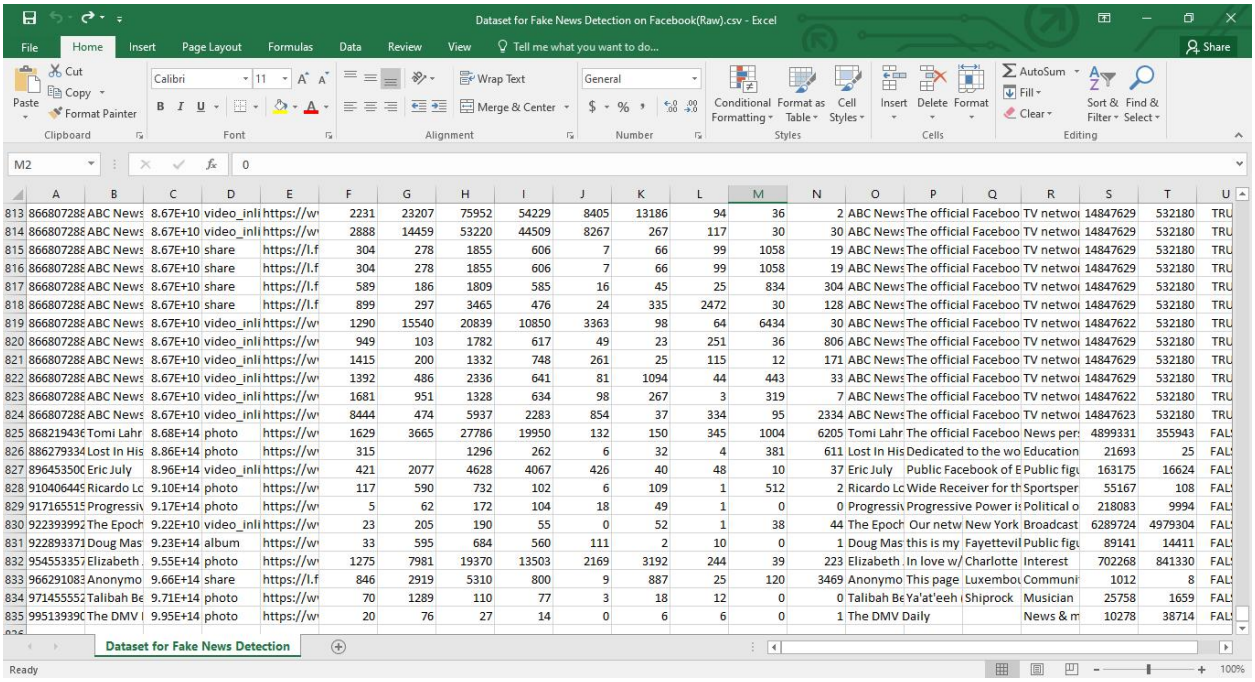


Fig. 5-2: Raw dataset.

Another significant part of our initial Dataset is the comments part. We have collected comments on a fake post. The below screenshot elucidates the comments file of our screenshot.

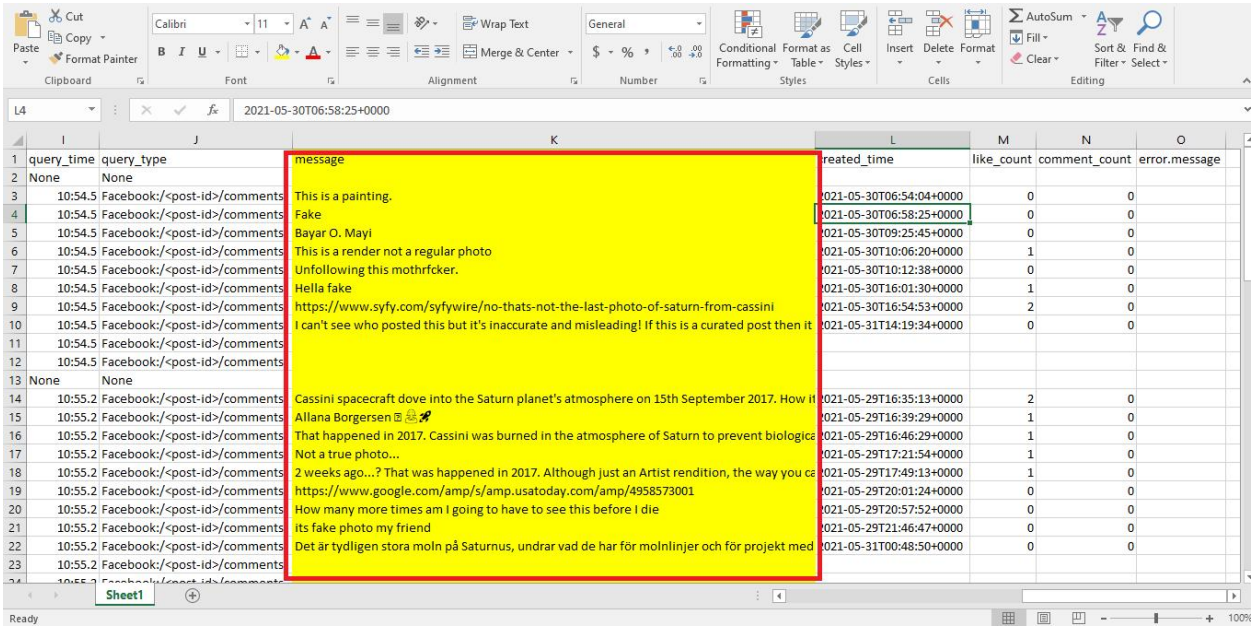


Fig. 5-3: Comments of raw dataset.

Here the message column shows the comments that were posted by the users. Comments part can be significant values for sentiment analysis. But here, our prime intention is to detect fake news using a machine learning algorithm. That's why we avoided this table.

5.3 Raw to final

It is already mentioned that our initial Dataset has 834 rows and 21 columns. As our target is to detect fake news from the Dataset so this huge number of data might not be necessary for our model rather, it may cause an issue in the machine learning algorithm. That's why it is necessary to deduct unnecessary columns from the Dataset.

5.4 Data Reduction

Two columns named "name" and "location.city" were deleted from the initial Dataset. The "name" column had a redundancy while other columns had no significance for our model. On the other hand, Facepager was unable to provide appropriate information "location.city" columns. That's why these two columns were dropped from the excel file to create the final Dataset.

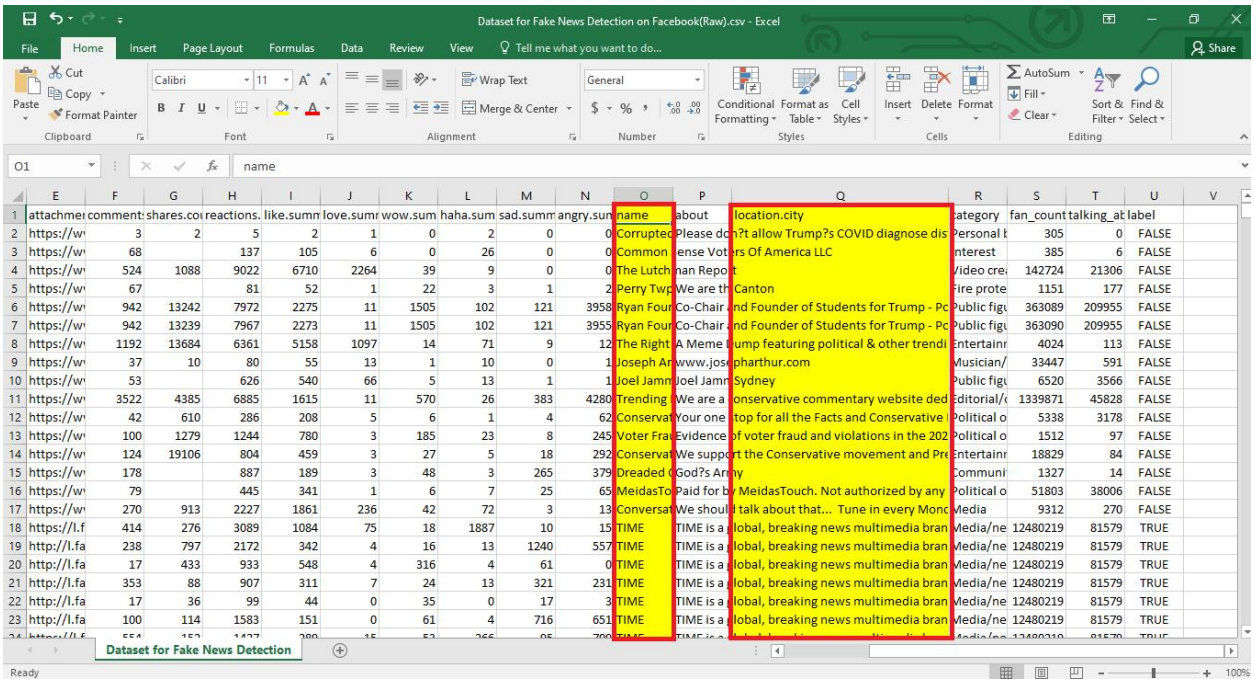


Fig. 5-4: Deleted columns.

On the other hand, some cells have null values, and they may cause a problem as well. In this case, the cell was replaced by 0. After applying all the necessary steps, our final Dataset looks like the picture below.

5.5 Amount of data

Our initial Dataset has a large number of data, including user comments, reactions, and other information, e.g., page id post id. It encompasses 834 labeled data points, 614 of which are "true" and 220 "false." There are a total of 19 attributes, which are grouped into two groups: post information and page information.

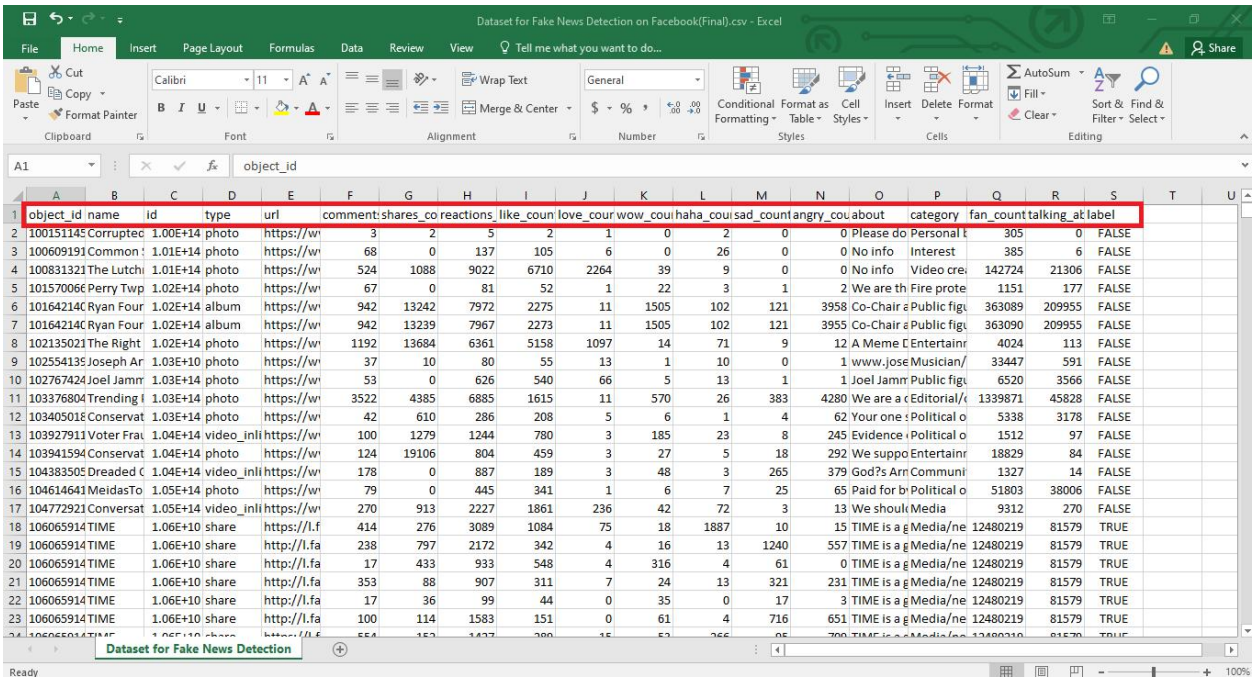


Fig. 5-5: Initial columns name.

In our final Dataset there is a total of 834 rows and 19 columns, and all the null cells are replaced with 0.

5.6 Naming

The column's name had been changed for convenience. It helped while the Dataset was implemented in our model.

Table 5-A: Columns naming.

Columns Name (Before Naming)	Columns Name (After Naming)
object_id	object_id
from.name	name
from.id	id
attachments.data.0.type	type
attachments.data.0.url	url
comments. summary. total_count	comments_count
shares. Count	shares_count
reactions.summary.total_count	reactions_count
like.summary.total_count	like _count
love.summary.total_count	love _count
wow.summary.total_count	wow _count
haha.summary.total_count	haha _count
sad. Summary.total_count	sad_count
angry. Summary.total_count	angry_count
about	about
category	category
fan _count	fan _count
talking_about_count	talking_about_count

5.7 Dataset group

We divided our datasets into two groups: post information and page information. Page information includes the data regarding a page, i.e., name, about, fan count, followers. The Post information segment contains general information about a post, i.e., type, URL, label, comment, share and reaction count, and count for six individual reactions. Both of the section is identified by a unique identifier named Object_id and id, respectively.

Table 5-B: Attributes name.

Group	Attribute
Page Information	Object_id
	name
	about
	category
	fan_count
	talking_about_count
Post Information	id
	type
	url
	Comment_count
	Share_count
	label
	reactions_count
	like_count
	love_count
	wow_count
	haha_count
	sad_count
	angry_count

5.8 Data Types

In our Dataset, there are three types of data string, integer, and Boolean.

Table 5-C: Data Types.

Attribute	Data Types
Object_id	String
name	String
about	String
category	String
fan_count	Int64
talking_about_count	Int64
id	Int64
type	String
url	String
Comment_count	Int64
Share_count	Int64
label	bool
reactions_count	Int64
like_count	Int64
love_count	Int64
wow_count	Int64
haha_count	Int64
sad_count	Int64
angry_count	Int64

Chapter 6: Experiment

6.1 Experiments Outline

This procedure consists of four parts that describe the model from data source to selecting the best model for the dataset and evaluating a hypothesis of the applied models.

Such as data collection, data preprocessing, training model, validation [**Fig. 6-1**].

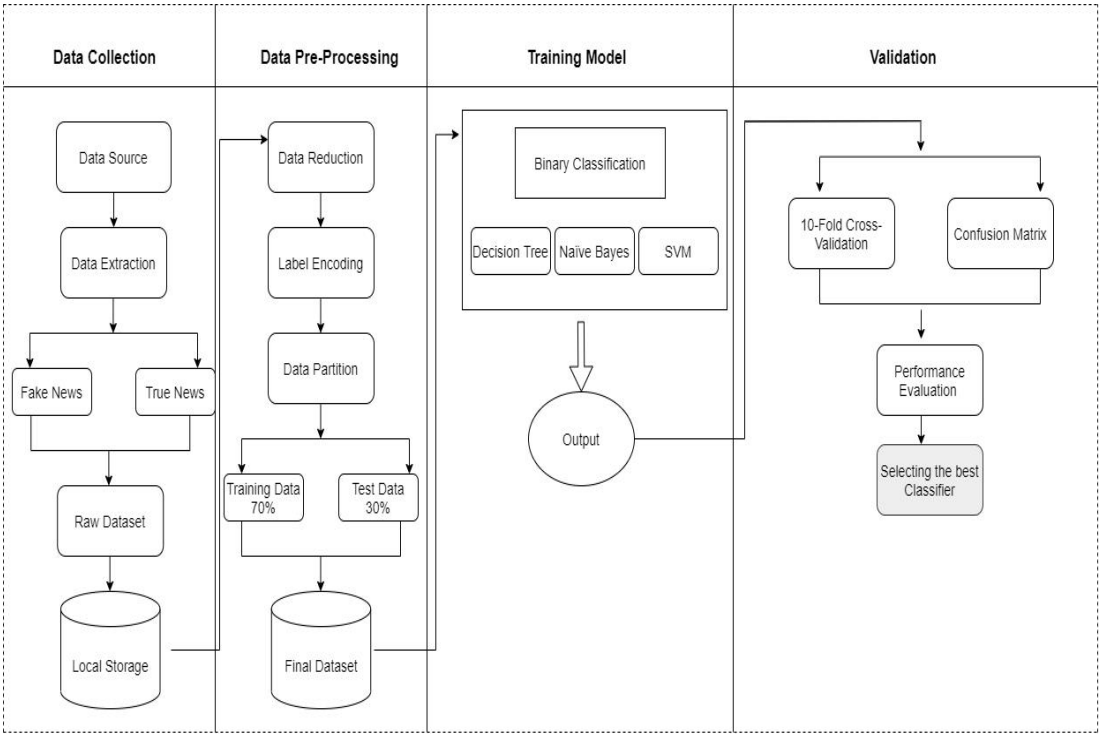


Fig. 6-1: Experimental procedure.

6.2 Data Collection

Fact-checking websites and famous news pages on Facebook were utilized in combination to obtain both fake and true content. They were retrieved using Facepager (Version 4.3). They were combined to generate a raw dataset for later processing. In this section 5, called "Data Collection Method," it is briefly discussed how to acquire data for a new dataset of Facebook social media.

6.3 Data Pre-processing

After combining the raw dataset, there were 22 columns; some were unnecessary, such as information not linked to the posts or pages but associated with the application. They were erased subsequently. The remaining 19 columns' names were renamed for better comprehension. There was a limited number of rows (5-7) that had some missing values without any content files like videos or images, which were replaced with suitable values like names like "video," "image" etc.

6.3.1 Label encoding

Many of the machine learning algorithms are able to comprehend numeric data instead of categorical data to measure accuracy and other metrics and to model for training and testing. Categorical data must be encoded into numbers using encoding terms such as LabelEncoder, OneHotEncoder, and so on. Some algorithms in ML can perform well with both numeric and categorical data. The ideal technique in any data science project is to change categorical data to a numeric value.

Label encoding is being used to change categorical data to numeric data so that it can be turned into a machine-readable form. It is the fundamental preprocessing stage for the structured dataset in supervised learning. Some examples of it are described below:

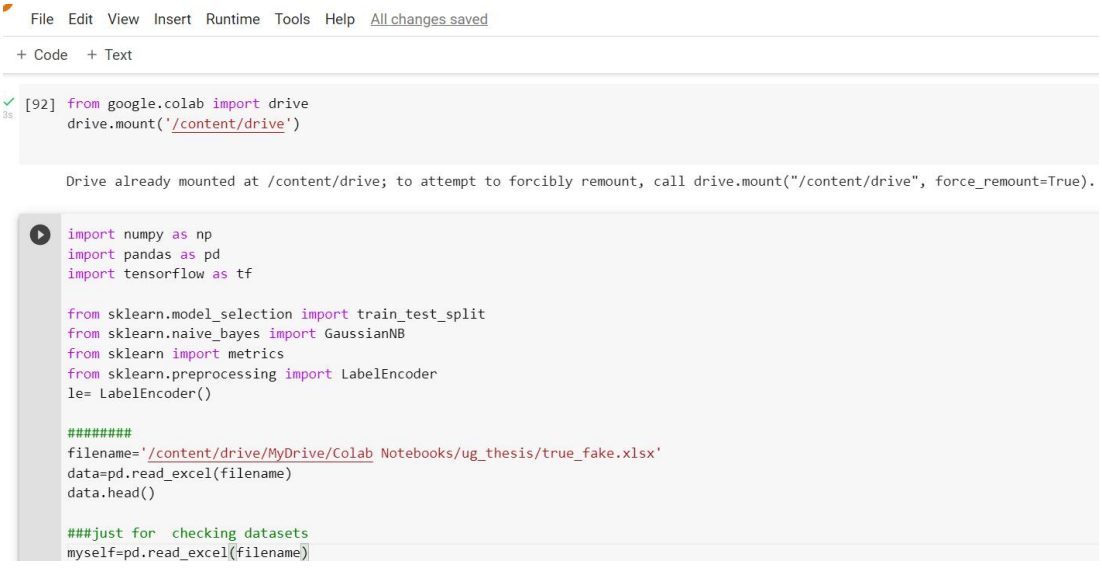


Fig. 6-2: Importing Dataset using pandas.

According to Google Research, the Google Colab environment is used to do Python coding through the browser, including Python code and Python machine learning techniques, utilizing the Google cloud.

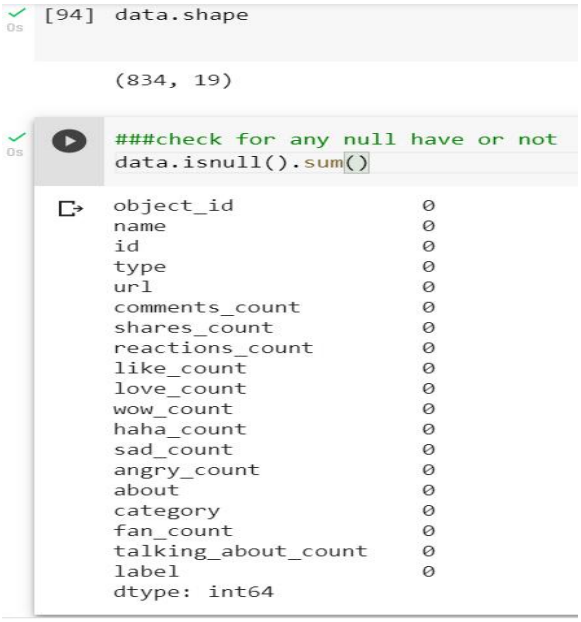
First, the disk mount is finished by granting access to the Google Cloud Service and using pandas to import the dataset from the Google Cloud uploaded files. As a result, several libraries must be imported in order to complete the work. Because the database file is an excel spreadsheet, Python pandas codes are used to import it [**Fig. 6-2**].

```
###To know data type names

data.dtypes
```

```
object_id              object
name                   object
id                      int64
type                   object
url                    object
comments_count          int64
shares_count            int64
reactions_count         int64
like_count              int64
love_count              int64
wow_count               int64
haha_count              int64
sad_count               int64
angry_count             int64
about                  object
category               object
fan_count               int64
talking_about_count     int64
label                    bool
dtype: object
```

Fig. 6-3: Datatypes from the actual Dataset.

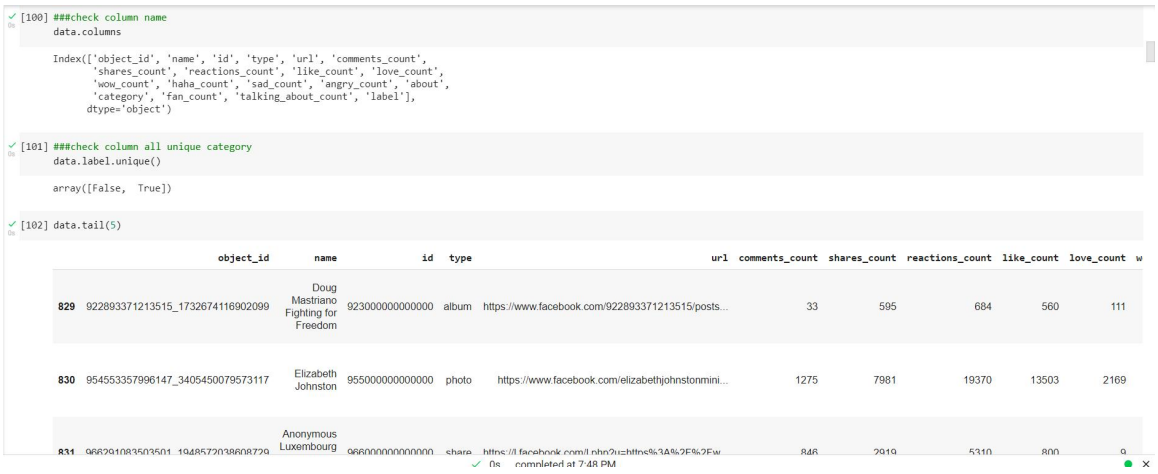
To preprocess data, look at the actual data types to see whether columns should be preprocessed or not, such as converting to numeric values, as illustrated in figure [Fig. 6-3]. Some are numeric values, while others are mixed-type or string-type objects, as well as Boolean types.



```
[94] data.shape

(834, 19)

###check for any null have or not
data.isnull().sum()

object_id               0
name                    0
id                      0
type                    0
url                     0
comments_count          0
shares_count            0
reactions_count         0
like_count              0
love_count              0
wow_count               0
haha_count              0
sad_count               0
angry_count             0
about                   0
category                0
fan_count               0
talking_about_count     0
label                   0
dtype: int64
```

Fig. 6-4: NULL values checking.

The dataset was produced with 834 rows and 19 columns of data values, but there are actually 835 rows in total, including one named row in the first row. Before, only a limited number of cells (less than 5 to 10 rows in each column) had their values modified. Those are video content and picture files, which were simply named file names. Those files aren't going to be uploaded. This isn't primarily a content-based strategy. The data is correctly gathered with care, and there are no null values in the dataset because the resultant response is zero in every column [Fig. 6-4].



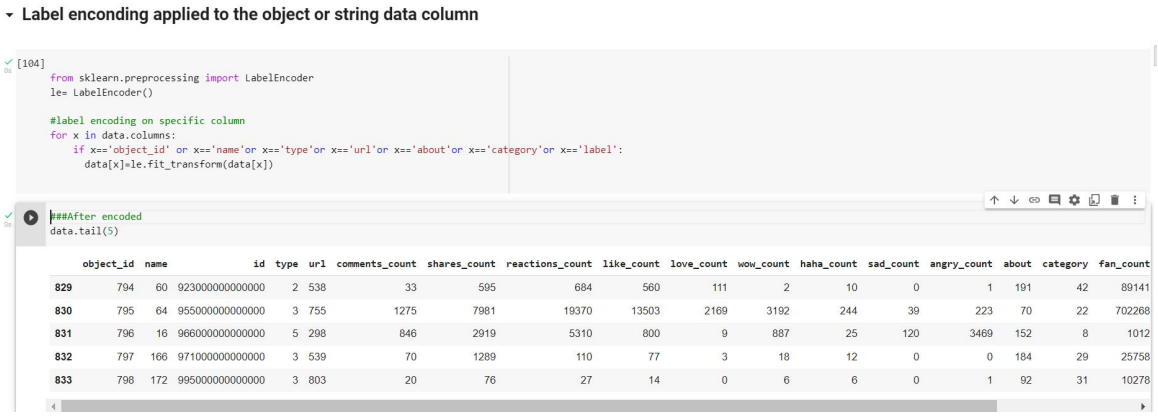
```
[100] ###check column name
      data.columns

      Index(['object_id', 'name', 'id', 'type', 'url', 'comments_count',
             'shares_count', 'reactions_count', 'like_count', 'love_count',
             'wow_count', 'haha_count', 'sad_count', 'angry_count', 'about',
             'category', 'fan_count', 'talking_about_count', 'label'],
            dtype='object')

[101] ###check column all unique category
      data.label.unique()

      array([False,  True])

[102] data.tail(5)
```

	object_id	name	id	type	url	comments_count	shares_count	reactions_count	like_count	love_count	w
829	922893371213515_1732674116902099	Doug Mastriano Fighting for Freedom	923000000000000	album	https://www.facebook.com/922893371213515/posts...	33	595	684	560	111	
830	954553357996147_3405450079573117	Elizabeth Johnston	955000000000000	photo	https://www.facebook.com/elizabethjohnstonmini...	1275	7981	19370	13503	2169	
831	966291083503501_1948572038608729	Anonymous Luxembourg	966000000000000	share	https://l.facebook.com/l.php?u=https%3A%2F%2Fw...	846	2919	5310	800	9	

0s completed at 7:48 PM

Fig. 6-5: Analyzing data from the Dataset.

The columns in the Dataset are collected and compared to [**Fig**. 6-5], which are object and boolean types. For preprocessing, verify whether class label values are unique or not [**Fig.** 6-5].



	object_id	name	id	type	url	comments_count	shares_count	reactions_count	like_count	love_count	wow_count	haha_count	sad_count	angry_count	about	category	fan_count
829	794	60	923000000000000	2	538	33	595	684	560	111	2	10	0	1	191	42	89141
830	795	64	955000000000000	3	755	1275	7981	19370	13503	2169	3192	244	39	223	70	22	702268
831	796	16	966000000000000	5	298	846	2919	5310	800	9	887	25	120	3469	152	8	1012
832	797	166	971000000000000	3	539	70	1289	110	77	3	18	12	0	0	184	29	25758
833	798	172	995000000000000	3	803	20	76	27	14	0	6	6	0	1	92	31	10278

Fig. 6-6: Label encoding.

Some Python packages are imported to apply label encoding. To transform categorical data into numeric data, the values of identified categorical data columns are label encoded [**Fig. 6-6**].

```
[106] data.label.unique()

      array([0, 1])
```

```
data.dtypes

object_id              int64
name                   int64
id                     int64
type                   int64
url                    int64
comments_count         int64
shares_count           int64
reactions_count        int64
like_count             int64
love_count             int64
wow_count              int64
haha_count             int64
sad_count              int64
angry_count            int64
about                  int64
category               int64
fan_count              int64
talking_about_count    int64
label                  int64
dtype: object
```

Fig. 6-7: After label encoding data type.

Following label encoding, the categorical data is converted to numeric data. Check the whole datasets again to ensure that all of the values are of the numeric type [**Fig. 6-7**].

```
[108] # Load libraries
      import pandas as pd
      from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
      from sklearn.model_selection import train_test_split # Import train_test_split function
      from sklearn import metrics #Import scikit-learn metrics module for accuracy calculation

[109] ###check column name
      data.columns

      Index(['object_id', 'name', 'id', 'type', 'url', 'comments_count',
             'shares_count', 'reactions_count', 'like_count', 'love_count',
             'wow_count', 'haha_count', 'sad_count', 'angry_count', 'about',
             'category', 'fan_count', 'talking_about_count', 'label'],
            dtype='object')
```

Data split Feature and Target

```
#split dataset in features and target variable
feature_cols = ['object_id', 'name', 'id', 'type', 'url', 'comments_count',
       'shares_count', 'reactions_count', 'like_count', 'love_count',
       'wow_count', 'haha_count', 'sad_count', 'angry_count', 'about',
       'category', 'fan_count', 'talking_about_count']
X = data[feature_cols] # Features
y = data.label # Target variable
```

Fig. 6-8: Separated into feature and target (class label) data x, y.

Before splitting the whole datasets into train and test, the whole datasets are separated into features (x) and class label data (y) [**Fig. 6-8**].

Dataset split into train and test set

```
[200] # Split the dataset into training and test sets.
      X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=7) # 70% training and 30% test
```

Fig. 6-9: Split the whole Datasets into train and test sets**.**

The dataset has been separated into training (70 percent) and test (30 percent) data [**Fig. 6-9**]. Finally, the dataset is ready to be utilized to train a model.

6.4 Training Model

A dataset is a set of information. A dataset generally consists of two parts:

The variables in the data are known as "**features"** (also known as "inputs" or "characteristics"). It can be more than one. The phrase "feature names" refers to a list of all the feature names.

The **output** variable is known as the response (sometimes known as the target, label, or output). Features are independent factors that influence the label, which is the dependent variable.

Simple and efficient data mining and data analysis tools are important elements of scikit-learn. Learning (determining) suitable values for all the weights and the bias from labeled

samples is all that training a model entail. The assessment and building of algorithms that can learn from and make predictions on unseen data is a typical job in machine learning.

A training data set is a group of instances used to fit the parameters (e.g., weights) of a classifier during the learning process. A test data set is a dataset that is unrelated to the training data set but has the same probability distribution.

6.4.1 Decision Tree

A decision tree is fashioned like a tree, with a root at the top and multiple leaves connected by a branch at the bottom. Each leaf represents a class label, and to forecast a class, a route is followed from a leaf node to the root via a branch. It is a basic but efficient approach to categorizing a dataset.

Fig. 6-10: Decision Tree classifier.

To build a decision tree classifier in Scikit-learn, some essential libraries are required to import all of them. After that, a DT model is created to perform an evaluation of how correctly the classifier predicts the class label. This accuracy is computed by comparing the actual test set values with predicted values [**Fig. 6-10**].

To determine the optimal attribute for splitting the records, DT uses some attribute selection algorithms. The chosen attribute acts as a decision node and divides the dataset into several smaller subgroups. Recursively repeat this process for each child node until all conditions are met, such as there are no more instances, attributes, or tuples with the same attribute values [**Fig. 6-10**].

Fig. 6-11: Visualizing Decision Tree.

The root node is the top initial node in a decision tree [Fig. 6-12]. It learns to divide based on the value of the property. The visualization takes the shape of a flowchart diagram that can be read by humans, and internal decision-making logic is incorporated [Fig. 6-11]. It has a quicker learning time and can handle high-dimensional data with excellent accuracy.

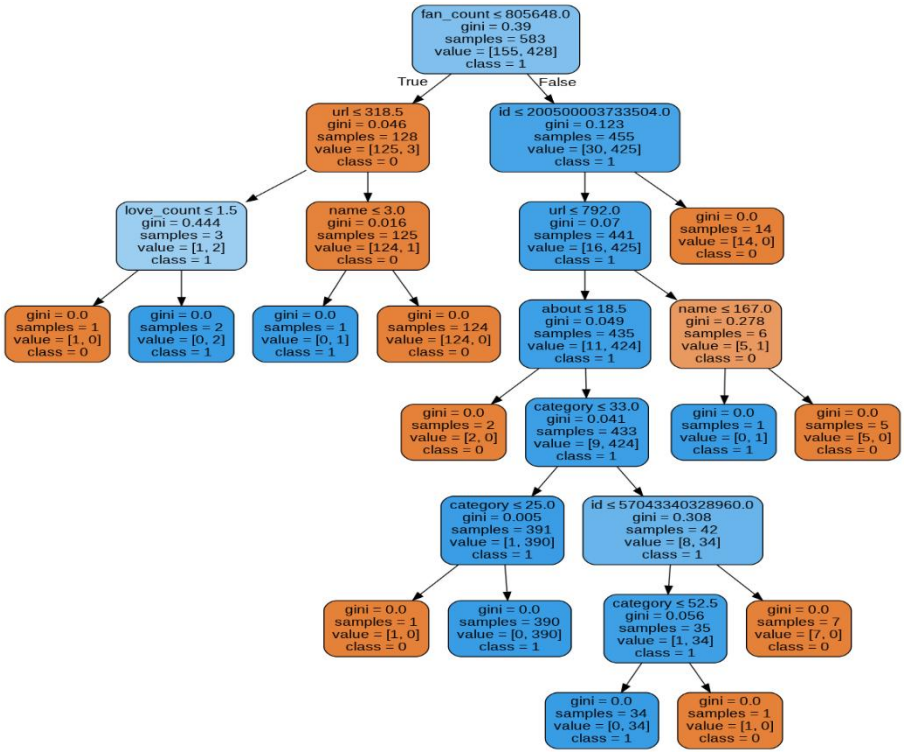


Fig. 6-12: Visualizing image of Decision Tree classifier.

Optimizing Decision Tree Performance

```
# DT object
clf = DecisionTreeClassifier(criterion="entropy", max_depth=3)

# Train DT
clf = clf.fit(X_train,y_train)

#Predict the label for test dataset
y_pred = clf.predict(X_test)

# Model Accuracy
print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.952191235059761
```

Fig. 6-13: Optimizing Decision Tree performance.

The default measure for the criterion is "gini" for the Gini index, and we may alternatively enable "entropy" for information gain in various attribute selections. The maximum depth (max depth) of DT is the default value of none. However, it may be changed to a higher integer number. Overfitting may occur when the value is very high, whereas underfitting occurs when the value is too low [**Fig. 6-13**].

Visualizing Optimized Decision Trees

```
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus
dot_data = StringIO()
export_graphviz(clf, out_file=dot_data,
                filled=True, rounded=True,
                special_characters=True, feature_names = feature_cols,class_names=['0','1'])
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
graph.write_png('diabetes.png')
Image(graph.create_png())
```

Fig. 6-14: Visualizing Optimized Decision Trees.

Compared to the previous decision tree model plot, this trimmed model and model plot are less difficult, explainable, and easier to comprehend **[Fig. 6-15]**.

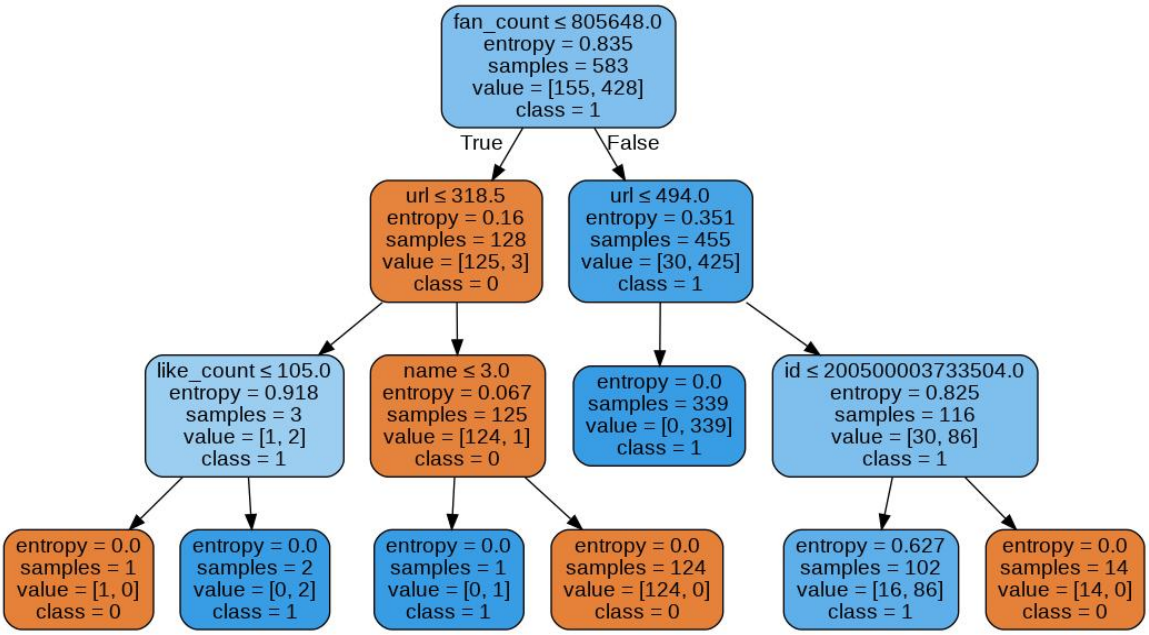


Fig. 6-15: Optimized Visualizing image of Decision Tree classifier.

6.4.2 Gaussian Naïve Bayes Classifier

It is a supervised learning algorithm and quite effective at accurately guessing the class label. It operates on the premise that each variable is independent of the others, making it suited for huge datasets. Nave Bayes may deliver fantastic results for a dataset under the right circumstances and is very scalable. It predicts an unknown class using Bayes' theory of probability.

2.Applying Gaussian Classifier

```
#Creating Gaussian Classifier
model = GaussianNB()

#Train the model by using the training sets
model.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = model.predict(X_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.852589641434263
```

Fig. 6-16: Training Gaussian Naïve Bayes Classifier.

In the following stages, the Naive Bayes classifier indicates the probability of an occurrence: Determine the prior probability for a given set of class labels and determine the likelihood probability for each characteristic in each class. Enter this number into the Bayes formula and calculate the posterior probability. If the input belongs to the higher probability class, determine which class has the higher likelihood. Some fundamental libraries from Scikit-learn are imported for doing the task [**Fig. 6-16**].

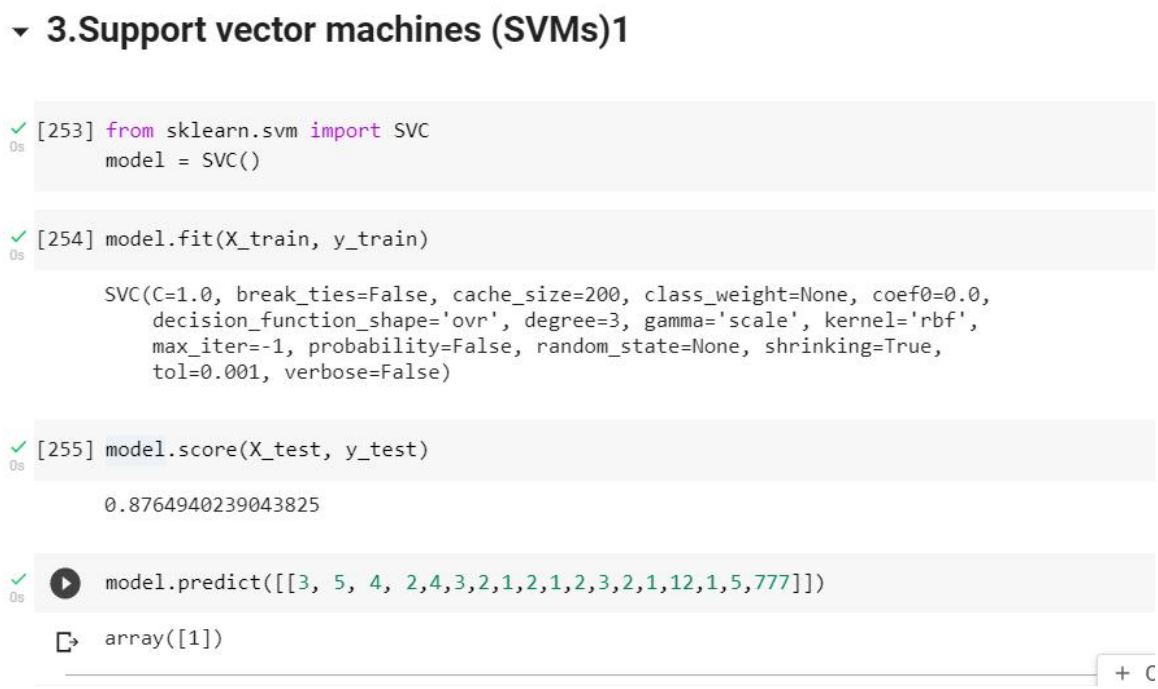


Fig. 6-17: Training Support Vector Machine.

6.4.3 Support Vector Machine (SVM)

The support vector machine may be used for regression as well as classification. SVM classification works with a linear separable dataset by separating it into two distinct sections using a hyperplane. The higher the marginal distance, the better the SVM classifier produces. In the case of a non-linear separable dataset, however, numerous hyperplanes are formed to gain more precision.

3.Support vector machines (SVMs)1

```
[253] from sklearn.svm import SVC
      model = SVC()

[254] model.fit(X_train, y_train)

      SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
          decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
          max_iter=-1, probability=False, random_state=None, shrinking=True,
          tol=0.001, verbose=False)

[255] model.score(X_test, y_test)

      0.8764940239043825

      model.predict([[3, 5, 4, 2,4,3,2,1,2,1,2,3,2,1,12,1,5,777]])

      array([1])
```

Fig. 6-18: Training Support Vector Machine.

The SVM module has been loaded from sklearn.svm and the model function svm() are invoked. The model is then trained to learn from training data such that it can predict the class label from test or unobserved data [Figure 6-18].

```
# Tuning parameters
# 1. Regularization (C)

model_C = SVC(C=1)
model_C.fit(X_train, y_train)
model_C.score(X_test, y_test)
```

```
0.8764940239043825
```

```
model_C = SVC(C=10)
model_C.fit(X_train, y_train)
model_C.score(X_test, y_test)
```

```
0.8924302788844621
```

Fig. 6-19: Tuning parameters of SVM.

Change the default parameter value of Support Vector Machines (SVM) to evaluate prediction accuracy. Everything comes down to prediction in order to discover a generalized answer [Fig. 6-19].

The outputs of these three classifiers are then compared to decide which the best classifier is.

6.5 Validation

This phase plays the most crucial role in objectively determining a model's superiority and reliability. We applied two statistical validation approaches to assess a model's performance. Confusion Matrix and K-Fold Cross Validation were applied to empirically compare these three models and select the best one.

6.5.1 K-Fold Cross Validation

The entire Dataset is evenly divided into "K" folds, and each component of the Dataset is later used as both a training and a test set. The results found on each stage were compared to evaluate a model. Due to the low bias and variance, it is generally accepted that the value of K should be ten even if the computing resources allow the number of fold or "K" to be greater than that [Morgan Book]. Thus, the value of K was taken as 10.

4.Here **10-fold Cross validation**..

K(10) fold cross validation applied

```
import numpy as np
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
########
folds=KFold(10)
```

```
from sklearn.tree import DecisionTreeClassifier

model1=DecisionTreeClassifier()
result1=cross_val_score(model1,X,y,cv=folds)
print(result1)
print(np.mean(result1))
```

```
[0.97619048 1.         0.8452381  0.96428571 0.97590361 0.96385542
 0.96385542 0.98795181 0.97590361 0.6626506 ]
0.9315834767641995
```

Fig. 6-20: Cross validation for DT classifier.

```
# #DecisionTree
scr=[]
for i in result1:
  scr.append(i)

fold_10 = {'Fold': [1,2,3,4,5,6,7,8,9,10],
           'cross_val_score': scr
          }
df = pd.DataFrame(fold_10, columns = ['Fold', 'cross_val_score'])
print (df)
```

	Fold	cross_val_score
0	1	0.976190
1	2	1.000000
2	3	0.845238
3	4	0.964286
4	5	0.975904
5	6	0.963855
6	7	0.963855
7	8	0.987952
8	9	0.975904
9	10	0.662651

Fig. 6-21: Visualizing Cross validation DT in a table form.

```
##GaussianNB
from sklearn.naive_bayes import GaussianNB

model2 = GaussianNB()
result2=cross_val_score(model2,X,y,cv=folds)
scr=[]
for i in result2:
  scr.append(i)

fold_10 = {'Fold': [1,2,3,4,5,6,7,8,9,10],
            'cross_val_score': scr
          }
df = pd.DataFrame(fold_10, columns = ['Fold', 'cross_val_score'])
print (df)
```

```
   Fold  cross_val_score
0     1         0.619048
1     2         1.000000
2     3         0.750000
3     4         0.857143
4     5         0.927711
5     6         0.698795
6     7         0.722892
7     8         0.975904
8     9         0.987952
9    10         0.975904
```

Fig. 6-22: Cross validation Gaussian Naïve Bayes in a table form.

```
from sklearn.svm import SVC

model3=SVC()
result3=cross_val_score(model3,X,y,cv=folds)
print(result3)

print(np.mean(result3))
```

```
[0.61904762 1.         0.75       0.85714286 0.92771084 0.69879518
 0.4939759  0.97590361 0.98795181 0.97590361]
0.8286431440045898
```

Fig. 6-23: Cross validation SVM.

```
[268] ##SVC
scr=[]
for i in result3:
  scr.append(i)

fold_10 = {'Fold': [1,2,3,4,5,6,7,8,9,10],
           'cross_val_score': scr
          }
df = pd.DataFrame(fold_10, columns = ['Fold', 'cross_val_score'])
print (df)
```

	Fold	cross_val_score
0	1	0.619048
1	2	1.000000
2	3	0.750000
3	4	0.857143
4	5	0.927711
5	6	0.698795
6	7	0.493976
7	8	0.975904
8	9	0.987952
9	10	0.975904

Fig. 6-24: Cross validation SVM in a table form.

6.5.2 Confusion Matrix

For a two-class problem, the table can be described as,

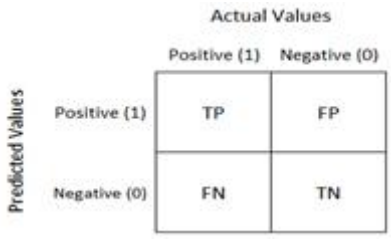


Fig. 6-25: Example of confusion matrix.

The table consists of the values predicted by the classifier and the actual value of the Dataset. By comparing these attributes (TP, FP, FN, TN), Accuracy, Precision, Recall, and F1-Score are calculated to assess a model.

Confusion_matrix_GNB

```
[401] from sklearn.metrics import confusion_matrix
      from sklearn.metrics import classification_report

      confusion_matrix(y_test, y_pred)
```

```
array([[ 28,  37],
       [  0, 186]])
```

```
model.classes_
```

```
array([0, 1])
```

```
[403] confusion_matrix(y_test, y_pred,labels=(0,1))
```

```
array([[ 28,  37],
       [  0, 186]])
```

```
[404] confusion_matrix(y_test, y_pred, labels=(0,1)).ravel()
```

```
array([ 28,  37,   0, 186])
```

Fig. 6-26: Confusion matrix in Gaussian Naïve Bayes.

```
[406] tn, fp, fn, tp

      (28, 37, 0, 186)

[407] accuracy=(tp+tn)/(tp+tn+fp+fn)

[408] accuracy

      0.852589641434263

[409] precision = tp/(tp+fp)
      precision

      0.8340807174887892

[410] print(classification_report(y_test,y_pred))

                   precision    recall  f1-score   support

               0       1.00      0.43      0.60        65
               1       0.83      1.00      0.91       186

        accuracy                           0.85       251
       macro avg       0.92      0.72      0.76       251
    weighted avg       0.88      0.85      0.83       251
```

Fig. 6-27: Confusion Matrix report of GNB.

```
[411] from sklearn.metrics import precision_recall_curve
      precision,recall,thresold=precision_recall_curve(y_test,y_pred)

[412] precision

      array([0.83408072, 1.        ])

[413] recall

      array([1., 0.])

[414] thresold

      array([1])
```

Fig. 6-28: Individual report value of GNB.

Confusion_matrix:Decision tree

```
[468] from sklearn.metrics import confusion_matrix
      from sklearn.metrics import classification_report

      confusion_matrix(y_test, y_pred)

      array([[ 57,   8],
             [  0, 186]])

[469] clf.classes_

      array([0, 1])

[470] confusion_matrix(y_test, y_pred,labels=(0,1))

      array([[ 57,   8],
             [  0, 186]])

[471]
      confusion_matrix(y_test, y_pred, labels=(0,1)).ravel()

      array([ 57,   8,   0, 186])
```

Fig. 6-29: Confusion matrix of Decision Tree.

```
[472] confusion_matrix(y_test, y_pred, labels=(0,1)).ravel()
      tn, fp, fn, tp  = confusion_matrix(y_test, y_pred, labels=(0,1)).ravel()
      precision = tp/(tp+fp)
      precision

      0.9587628865979382

      tn, fp, fn, tp

      (57, 8, 0, 186)

[474] accuracy=(tp+tn)/(tp+tn+fp+fn)
      accuracy

      0.9681274900398407

[475] print(classification_report(y_test,y_pred))

                    precision    recall  f1-score   support

                 0       1.00      0.88      0.93        65
                 1       0.96      1.00      0.98       186

          accuracy                           0.97       251
         macro avg       0.98      0.94      0.96       251
      weighted avg       0.97      0.97      0.97       251
```

Fig. 6-30: Confusion Matrix Report of DT.

```
[476] from sklearn.metrics import precision_recall_curve
      precision,recall,thresold=precision_recall_curve(y_test,y_pred)

[477] precision

      array([0.95876289, 1.        ])

[478] recall

      array([1., 0.])
```

Fig. 6-31: Individual report value of DT.

Confusin Matrix of SVMs

```
[525] from sklearn import metrics
      #accuracy
      print("acuracy:", metrics.accuracy_score(y_test,y_pred=y_pred))
      #precision score
      print("precision:", metrics.precision_score(y_test,y_pred=y_pred))
      #recall score
      print("recall" , metrics.recall_score(y_test,y_pred=y_pred))
```

```
acuracy: 0.9840637450199203
precision: 0.9789473684210527
recall 1.0
```

```
[526] print(classification_report(y_test,y_pred))
```

```
                   precision    recall  f1-score   support

                0       1.00      0.94      0.97        65
                1       0.98      1.00      0.99       186

         accuracy                           0.98       251
        macro avg       0.99      0.97      0.98       251
     weighted avg       0.98      0.98      0.98       251
```

Fig. 6-32: Confusion Matrix, CM report of SVM.

Chapter 7: Results

We have applied three classifiers to the Dataset to distinguish fake news from true news. A binary classifier was required by our problem statement. The dataset was divided into a training and test set, and the accuracy was determined using a classifier.

The results of these three classifiers are summarized below.

7.1 Gaussian Naïve Bayes

This classifier produced the lowest score among the three at 85.259%.

```
#Creating Gaussian Classifier
model = GaussianNB()

#Train the model by using the training sets
model.fit(X_train, y_train)

#Predict the response for test dataset
y_pred = model.predict(X_test)

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

```
Accuracy: 0.852589641434263
```

Fig. 7-1: Gaussian Naïve Bayes Accuracy.

7.2 Support Vector Machine

SVM performed a little bit better than Gaussian Naïve Bayes at 87.649%.

```
model.fit(X_train, y_train)

SVC(C=1.0, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma='scale', kernel='rbf',
    max_iter=-1, probability=False, random_state=None, shrinking=True,
    tol=0.001, verbose=False)

[54] model.score(X_test, y_test)

0.8764940239043825
```

Fig. 7-2: Support Vector Machine Accuracy.

7.3 Decision Tree

It achieved the most satisfactory outcome out of three at 96.414%.

```
[20] # DT classifer object
     clf = DecisionTreeClassifier()

     # Train the DT Classifer
     clf = clf.fit(X_train,y_train)

     #Predict for test dataset
     y_pred = clf.predict(X_test)

     # Model Accuracy
     print("Accuracy:",metrics.accuracy_score(y_test, y_pred))

Accuracy: 0.9641434262948207
```

Fig. 7-3: Decision Tree Accuracy.

Following the tuning of these three classifiers, the findings consistently showed that the Decision Tree produced the best results of the three. Confusion Matrix and K-Fold Cross Validation were later used to further investigate the results' credibility. The results from the first stage of the experiment were supported by both validation procedures.

Chapter 8: Conclusion

According to the conclusions of this study, the experiment on the Dataset, which was gathered from Facebook, achieved an accuracy of over 96 percent.

Our methodology is simple to use and can effectively and accurately discriminate between fake and true news. Our model was trained using only the data discovered in a Facebook post, and comments were collected but never used. The model's accuracy and dependability can be increased even more by extracting knowledge from each post's comments.

This model can detect fake news found only on Facebook. As a result, future work can be done on different social media sites. To ensure more accuracy, the Dataset can be updated from time to time.

References

[1] Shearer, E. and Gottfried, J., 2021. *News Use Across Social Media Platforms 2017*. [online] Pew Research Center's Journalism Project. Available at: <https://www.pewresearch.org/journalism/2017/09/07/news-use-across-social-media-platforms-2017/> [Accessed 15 November 2021].

[2] Aldwairi, M. and Alwahedi, A., 2018. Detecting fake news in social media networks. *Procedia Computer Science*, *141*, pp.215-222.

[3] Rubin, V., Chen, Y. and Conroy, N., 2015. Deception detection for news: Three types of fakes. *Proceedings of the Association for Information Science and Technology*, 52(1), pp.1-4.

[4] Grinberg, N., Joseph, K., Friedland, L., Swire-Thompson, B. and Lazer, D., 2019. Fake news on Twitter during the 2016 U.S. presidential election. *Science*, 363(6425), pp.374-378.

[5] Ahmad, I., Yousaf, M., Yousaf, S. and Ahmad, M., 2020. Fake News Detection Using Machine Learning Ensemble Methods. *Complexity*, 2020, pp.1-11.

[6] Silverman, C., 2021. *This Analysis Shows How Viral Fake Election News Stories Outperformed Real News On Facebook*. [online] Buzzfeednews.com. Available at: <https://www.buzzfeednews.com/article/craigsilverman/viral-fake-election-news-outperformed-real-news-on-facebook> [Accessed 15 November 2021].

[7] Fong, B., 2021. Analysing the behavioural finance impact of 'fake news' phenomena on financial markets: a representative agent model and empirical validation. *Financial Innovation*, 7(1).

[8] Grilli, R., Ramsay, C. and Minozzi, S., 2002. Mass media interventions: effects on health services utilisation. *Cochrane Database of Systematic Reviews*, 1, p.28.

[9] Matthews, A., Herrett, E., Gasparrini, A., Van Staa, T., Goldacre, B., Smeeth, L. and Bhaskaran, K., 2016. Impact of statin related media coverage on use of statins: interrupted time series analysis with UK primary care data. *BMJ*, 353, p.i3283.

[10] Curet, M., 2021. *PolitiFact - COVID-19 tests are not part of a conspiracy to microchip people*. [online] @politifact. Available at: <https://www.politifact.com/factchecks/2021/oct/21/facebook-posts/covid-19-tests-are-not-part-conspiracy-microchip-p/> [Accessed 15 November 2021].

[11] Travers, M., 2021. *Facebook Spreads Fake News Faster Than Any Other Social Website, According To New Research*. [online] Forbes. Available at: <https://www.forbes.com/sites/traversmark/2020/03/21/facebook-spreads-fake-news-faster-than-any-other-social-website-according-to-new-research/?sh=12bc863d6e1a&fbclid=IwAR2Bf_ubp42cFtdjtX1eOSaYSJdc8BJlReFcgwZok_qwWge805bWMIesRzk> [Accessed 15 November 2021].

[12] Bourgonje, P., Schneider, J. and Rehm, G., 2017. From clickbait to fake news detection: an approach based on detecting the stance of headlines to articles. In: *EMNLP workshop: natural language processing meets journalism*. pp.84-89.

[13] Stahl, K., 2018. Fake news detection in social media. *California State University Stanislaus*, *6*, pp.4-15.

[14] Pérez-Rosas, V., Kleinberg, B., Lefevre, A. and Mihalcea, R., 2017. Automatic detection of fake news. *arXiv preprint arXiv:1708.07104*.

[15] Liu, Y. and Wu, Y.F.B., 2018, April. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-second AAAI conference on artificial intelligence*.

[16] Castillo, C., Mendoza, M. and Poblete, B., 2011. Information credibility on twitter. *Proceedings of the 20th international conference on World wide web - WWW '11*, pp.675-684.

[17] Xinyi Zhou, Atishay Jain, Vir V. Phoha, and Reza Zafarani. 2020. Fake News Early Detection: A Theory-driven Model. Digital Threats: Research and Practice 1, 2, Article 12 (July 2020), 25 pages. DOI:https://doi.org/10.1145/3377478

[18] Monti, F., Frasca, F., Eynard, D., Mannion, D. and Bronstein, M.M., 2019. Fake news detection on social media using geometric deep learning. *arXiv preprint arXiv:1902.06673*, pp.1-15

[19] Shu, K., Sliva, A., Wang, S., Tang, J. and Liu, H., 2017. Fake News Detection on Social Media. *ACM SIGKDD Explorations Newsletter*, 19(1), pp.1-15.

[20] Ajao, O., Bhowmik, D. and Zargari, S., 2019, May. Sentiment aware fake news detection on online social networks. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 2507-2511). IEEE.

[21] Guo, C., Cao, J., Zhang, X., Shu, K. and Yu, M., 2019. Exploiting emotions for fake news detection on social media. *arXiv preprint arXiv:1903.01728*.

[22] Kula, S., Choraś, M., Kozik, R., Ksieniewicz, P. and Woźniak, M., 2020, June. Sentiment analysis for fake news detection by means of neural networks. In *International Conference on Computational Science* (pp. 653-666). Springer, Cham.

[23] Statista. 2021. *Most used social media 2021 | Statista*. [online] Available at: <https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/> [Accessed 15 November 2021].

[24] Tacchini, E., Ballarin, G., Della Vedova, M.L., Moret, S. and de Alfaro, L., 2017. Some like it hoax: Automated fake news detection in social networks. *arXiv preprint arXiv:1704.07506*.

[25] Nast, C., 2021. *Facebook Reactions, the Totally Redesigned Like Button, Is Here*. [online] Wired. Available at: <https://www.wired.com/2016/02/facebook-reactions-totally-redesigned-like-button/> [Accessed 15 November 2021].

[26] Leadstories.com. 2021. *Lead Stories*. [online] Available at: <https://leadstories.com/> [Accessed 15 November 2021].

[27] Politifact.com. 2021. *PolitiFact*. [online] Available at: <https://www.politifact.com/> [Accessed 15 November 2021].

[28] Republic World. 2021. *Fact Check*. [online] Available at: <https://www.republicworld.com/fact-check> [Accessed 15 November 2021].

[29] TheQuint. 2021. *Latest News, Breaking News LIVE, Top News Headlines, Viral Videos News Updates - The Quint*. [online] Available at: <https://www.thequint.com/> [Accessed 15 November 2021].

[30] GitHub. 2021. *GitHub - strohne/Facepager: Facepager was made for fetching public available data from YouTube, Twitter and other websites on the basis of APIs and webscraping.*. [online] Available at: <https://github.com/strohne/Facepager> [Accessed 15 November 2021].