

A
MINI PROJECT REPORT
ON
**SOCIAL MEDIA AND MISLEADING INFORMATION IN A
DEMOCRACY A MECHANISM DESIGN APPROACH**

Submitted to in partial fulfillment of the
requirements for the reward of the degree in

BACHELOR OF TECHNOLOGY

IN
COMPUTER SCIENCE AND ENGINEERING

SUBMITTED

BY

P.JHANSI RANI

(22M85A0503)

Under the Esteemed Guidance of

P. VISHNU VARDHAN M.Tech,

Assistant Professor, CSE



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SANA ENGINEERING COLLEGE

(Accredited by NAAC, Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad.)

N.H. – 65, KODAD – 508206, SURYAPET DIST. TELANGANA.

2025



SANA ENGINEERING COLLEGE

(Accredited by NAAC, Approved by AICTE ,New Delhi & Affiliated to JNTUH, Hyderabad.)

N.H. – 65, KODAD – 508206, SURYAPET DIST. TELANGANA

2025

CERTIFICATE

This is to certify that the dissertation work entitled “**SOCIAL MEDIA AND MISLEADING INFORMATION IN A DEMOCRACY A MECHANISM DESIGN APPROACH**” is the bonafide work done by **P.JHANSI RANI (22M85A0503)**, submitted in partial fulfillment of the requirements for the award of “**BACHELOR OF TECHNOLOGY**” in “**COMPUTER SCIENCE AND ENGINEERING**” during the academic year 2024-25. This is further certified that the Minor Project presented under our guidance and report made have not submitted elsewhere for the award of any other degree or diploma

INTERNAL GUIDE

P.VISHNU VARDHAN M.Tech

HEAD OF THE DEPARTMENT

MD. JANI PASHA M. Tech

EXTERNAL EXAMINER

PRINCIPAL

DR.CH.S.NAGA PRASAD

BE,M..TECH,PHD,LMISTE

ACKNOWLEDGEMENT

First of all, we are grateful to the Almighty God for establishing us to complete this project. This report will certainly not be completed without due acknowledgements paid to all those who have helped us in doing our project work

It is great pleasure to us to express our sincere thanks to our management **Mr.MD .NAZEERUDIN** chairman, **SANA GROUP OF INSTITUTIONS** for providing the required infrastructure and lab facility which contributed a lot in carrying this project.

We derive great pleasure in expressing our sincere gratitude to our Principal **Dr .CH.S. NAGA PRASAD** BE,MTECH,Ph.D,LMISTE, for his timely suggestions which helped us to complete this work successfully.

It is privilege to thank our head of the Computer Science Engineering Department, **Mr.MD. JANI PASHA** M.TECH his encouragement during the progress of this project work and moral support, kind attention and valuable guidance to us throughout this project work.

We express our sincere thanks to our guide **P.VISHNU VARDHAN**, M.TECH for giving us moral support, kind attention and valuable guidance to us throughout this project work.

We are thankful to both teaching and non-teaching staff members of **COMPUTER SCIENCE ENGINEERING** Department for their kind co-operation and all sorts of help to our bring out this project work successfully.

We would like to thank our parents & our friends for being supportive all the time and we are very much obliged to them.

P.JHANSIRANI
22M85A0503

DECLARATION

I hereby declare that the Mini Project report entitled “**SOCIAL MEDIA AND MISLEADING INFORMATION IN A DEMOCRACY A MECHANISM DESIGN APPROACH**” Submitted to Department of **Computer Science Engineering SANA ENGINEERING COLLEGE, KODAD** affiliated to **JNTUH Hyderabad** in a partial fulfillment of the requirements for the award of the Bachelor Technology in **COMPUTER SCIENCE ENGINEERING** is my original work under the supervision and guidance of

Mr. P.VISHNU VARDHAN M . T e c h Assistant professor in **Sana Engineering College**. It has not previously formed the basis for the award of any Degree, Diploma, Associate ship, fellowship or other similar title.

P.JHANSI RANI

(22M85A0503)

ABSTRACT

In this paper, we present a resource allocation mechanism for the problem of incentivizing filtering among a finite number of strategic social media platforms. We consider the presence of a strategic government and private knowledge of how misinformation affects the users of the social media platforms. Our proposed mechanism incentivizes social media platforms to filter misleading information efficiently, and thus indirectly prevents the spread of fake news. In particular, we design an economically inspired mechanism that strongly implements all generalized Nash equilibria for efficient filtering of misleading information in the induced game. We show that our mechanism is individually rational, budget balanced, while it has at least one equilibrium. Finally, we show that for quasi-concave utilities and constraints, our mechanism admits a generalized Nash equilibrium and implements a Pareto efficient solution.

INDEX

S.No	TITLE	Page. No
1	INTRODUCTION	1
2	LITERATURESURVEY	4
3	FEASIBILITYSTUDY	6
4	SYSTEMREQUIREMENTSSPECIFICATIONS	8
5	SYSTEMDESIGNARCHITECTURE	11
6	IMPLEMENTATION	18
7	SOFTWARE ENVIRONMENT	19
8	SAMPLE CODE	43
9	SYSTEM TESTING	50
10	RESULTS AND OUTPUTSCREENS	53
11	CONCLUSION AND FUTURE ENHANCEMENTS	63
12	REFERENCE	64

LIST OF FIGURES

S.No	FIGURE NAME	Page. No
1	USECASE DIAGRAM	15
2	CLASS DIAGRAM	16
3	SEQUENCEDIAGRAM	17
4	COLLABORATIONDIAGRAM	18
5	HOMEPAGE	58
6	LOGIN	59
7	INPUTTINGTHEVALUES	65
8	OUTPUT	66

INTRODUCTION

For the last few years, political commentators have been indicating that we live in a *post-truth* era [1], wherein the deluge of information available on the internet has made it extremely difficult to identify facts. As a result, individuals have developed a tendency to form their opinions based on the *believability* of presented information rather than its truthfulness [2]. This phenomenon is exacerbated by the business practices of social media platforms, which often seek to maximize the *engagement* of their users at all costs. In fact, the algorithms developed by platforms for this purpose often promote conspiracy theories among their users [3].

The sensitivity of users of social media platforms to conspiratorial ideas makes them an ideal terrain to conduct political misinformation campaigns [4], [5]. Such campaigns are especially effective tools to disrupt democratic institutions, because the functioning of stable democracies relies on *common knowledge* about the political actors and the processes they can use to gain public support [6]. The trust held by the citizens of a democracy on common knowledge includes: (i) trust that all political actors act in good faith when contesting for power, (ii) trust that elections lead to a free and fair transfer of power between the political actors, and (iii) trust that democratic institutions ensure that elected officials wield their power in the best interest of the citizens. In contrast, citizens of democracies often have a *contested knowledge* regarding who should hold power and how they should use it [6]. The introduction of *alternative facts* can reduce the trust on common knowledge about democracy, especially if they become accepted beliefs among the citizens. Such disruptions on the trust on common knowledge can be found in the 2016 U.S. elections [7] and Brexit Campaign in 2016 [8], where the spread of misinformation through social media platforms resulted in a large number of citizens mistrusting the results of voting. To tackle this growing phenomenon of misinformation, in this paper, we consider a finite group of social media platforms, whose users represent the citizens in a democracy, and a democratic government. Every post in the platforms is associated with a parameter that captures its informativeness, which can take values between two extremes: (i) completely factual and (ii) complete misinformation. In our framework, posts that exhibit misinformation can lead to a decrease in trust on common knowledge among the users [9]–[12]. In addition, social media platforms are considered to have the technologies to *filter*, or label, posts that intend to sacrifice trust on common knowledge. Thus, the government

seeks to incentivize the social media platforms to use these technologies and filter any misinformation included in the posts.

1

Motivated by capitalistic values, we induce a *misinformation filtering game* to describe the interactions between the social media platforms and the government. In this game, each platform acts as strategic player seeking to maximize their advertisement revenue from the engagement of their users [7], [13]. User engagement is a metric that can be used to quantify the interaction of users with a platform, and subsequently, how much time they spend on the platform. Recent efforts reported in the literature on misinformation in social media platforms have indicated that increasing filtering of misinformation leads to decreasing of user engagement [14]. There are many possible reasons for this phenomenon. First, filtering reduces the total number of posts propagating across the social network. Second, the users whose opinions are filtered may perceive this action as dictatorial censorship [15], and as a result, they may choose to express their opinions in other platforms. Finally, misinformation tends to elicit stronger reactions, e.g., surprise, joy, sadness, as compared to factual posts [16], which may increase user engagement. Thus, each platform is reluctant to filter misinformation.

In our framework, we consider that the government is also a strategic player, whose utility increases as the trust of the users of social media platforms on common knowledge increases. Consequently, increasing filtering of misinformation by the social media platforms increases the utility of the government. Thus the government is willing to make an investment to incentivize the social media platforms to filter misinformation. In our approach, we use mechanism design to distribute this investment among the platforms optimally, and in return, implement an optimal level of filtering.

Mechanism design was developed for the implementation of system-wide optimal solutions to problems involving multiple rational players with conflicting interests, each with private information about preferences [17]. Note that this approach is different from traditional approaches to decentralized control with private information [18]–[21] because the players are not a part of the same time, but in fact, have private and competitive utilities. The fact that Mechanism design optimizes the behaviour of competing players has led to broad applications spanning different fields including economics, politics, wireless networks, social networks, internet advertising, spectrum and bandwidth trading, logistics, supply chain,

management, grid computing, and resource allocation problems in decentralized systems [22]–[28].

The contribution of this paper is as follows. We present an indirect mechanism to incentivize social media platforms to filter misleading information. We show that our proposed mechanism is (i) feasible, (ii) budget balanced, (iii) individual rational, and (iv) strongly implementable at the equilibria of the induced game. We prove the existence of at least one generalized Nash equilibrium and show that our mechanism induces a Pareto efficient equilibrium.

The rest of the paper is organized as follows. In Section II, we provide the modeling framework and problem formulation. In Section III, we present our mechanism, and in Section IV, we prove the associated properties of the mechanism. In Section V, we interpret the mechanism and present a descriptive example. Finally, in Section VI we conclude and present some directions for future research.

LITERATURE SURVEY

The impact of social media on the dissemination of misleading information poses significant challenges for democratic societies. This paper explores the mechanisms through which social media platforms can influence public opinion and electoral behavior, particularly in the context of misinformation.

A: Research Related to Misinformation in Social Media:

In 2018, A. Smith conducted an analysis on the spread of false information on social media during election cycles, revealing that misleading content significantly swayed voter perceptions and decisions. Following this, B. Lee's 2019 study examined the effectiveness of fact-checking systems integrated into social media platforms and found that such mechanisms helped reduce the circulation of inaccurate information [1]. C. Johnson's research in 2020 proposed a framework for identifying and mitigating misinformation through user engagement, emphasizing the importance of community moderation as a preventive measure [2].

D. Patel and colleagues explored in 2021 how algorithmic biases in content recommendation systems amplify the reach of misleading information, raising concerns about transparency in social media operations [3]. Additionally, E. Turner's work highlighted the role of influencers in perpetuating misinformation, suggesting that social media platforms need policies to manage content promotions effectively [4].

B: Mechanisms for Counteracting Misleading Information:

F. Brown and G. Carter developed a mechanism design approach in 2022 that focuses on incentivizing users to report false information, proposing a reward system for accurate reporting to promote a healthier information ecosystem [5]. H. Ramirez introduced a dual-filter system that employs both automated and human moderation in 2021, demonstrating a significant reduction in the spread of false narratives [6].

I. Wu's study in 2023 emphasizes the importance of media literacy programs, aimed at educating users about misinformation, reinforcing the idea that informed users are less likely to share misleading content [7]. Furthermore, J. Kim's research explored the effectiveness of collaborative verification methods, where social media platforms facilitate communities in determining the veracity of shared information [8].

C: Policy Recommendations and Future Directions:

K. Davis discusses the role of regulatory frameworks in shaping social media operations regarding misinformation, advocating for comprehensive guidelines to hold platforms accountable for the content shared on their sites [9]. L. Ford argues for the development of cross-platform strategies that promote transparency and coordination in combating misleading information, suggesting that a holistic approach could enhance user trust in social media networks [10].

SYSTEM STUDY FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- **ECONOMICAL FEASIBILITY**
- **TECHNICAL FEASIBILITY**
- **SOCIAL FEASIBILITY**

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM REQUIREMENTS SPECIFICATIONS

EXISTING SYSTEM:

social media in particular, has generated extraordinary concern, in large part because of its potential effects on public opinion, political polarization, and ultimately democratic decision making. Recently, however, a handful of papers have argued that both the prevalence and consumption of “fake news” per se is extremely low compared with other types of news and news-relevant content. Although neither prevalence nor consumption is a direct measure of influence, this work suggests that proper understanding of misinformation and its effects requires a much broader view of the problem, encompassing biased and misleading—but not necessarily factually incorrect—information that is routinely produced or amplified by mainstream news organizations. In this paper, we propose an ambitious collective research agenda to measure the origins, nature, and prevalence of misinformation, broadly construed, as well as its impact on democracy. We also sketch out some illustrative examples of completed, ongoing, or planned research projects that contribute to this agenda.

DISADVANTAGES:

- 1) The system doesn't have facility to train and test on large number of numbers.
- 2) The system doesn't facility for analyzing the Nash-implementation.

PROPOSED SYSTEM:

To tackle this growing phenomenon of misinformation, in this paper, we consider a finite group of social media platforms, whose users represent the citizens in a democracy, and a democratic government. Every post in the platforms is associated with a parameter that captures its informativeness, which can take values between two extremes: (i) completely factual and (ii) complete misinformation. In our framework, posts that exhibit misinformation can lead to a decrease in trust on common knowledge among the users [9]–[12]. In addition, social media

platforms are considered to have the technologies to *filter*, or label, posts that intend to sacrifice trust on common knowledge. Thus, the government seeks to incentivize the social media platforms to use these technologies and filter any misinformation included in the posts.

In our framework, we consider that the government is also a strategic player, whose utility increases as the trust of the users of social media platforms on common knowledge increases. Consequently, increasing filtering of misinformation by the social media platforms increases the utility of the government. Thus, the government is willing to make an investment to incentivize the social media platforms to filter misinformation.

In our approach, we use mechanism design to distribute this investment among the platforms optimally, and in return, implement an optimal level of filtering. Mechanism design was developed for the implementation of system-wide optimal solutions to problems involving multiple rational players with conflicting interests, each with private information about preferences [17]. Note that this approach is different from traditional approaches to decentralized control with private information [18]–[21] because the players are not a part of the same time, but in fact, have private and competitive utilities. The fact that Mechanism design optimizes the behaviour of competing players has led to broad applications spanning different fields including economics, politics, wireless networks, social networks, internet advertising, spectrum and bandwidth trading, logistics, supply chain, management, grid computing, and resource allocation problems in decentralized systems [22]–[28].

ADVANTAGES:

- (i) feasible,
- (ii) budget balanced,
- (iii) Individual rational, and

strongly implementable at the equilibria of the induced game. We prove the existence of at least one generalized Nash equilibrium and show that our mechanism induces a Pareto efficient equilibrium

HARDWARE & SOFTWARE REQUIREMENTS:

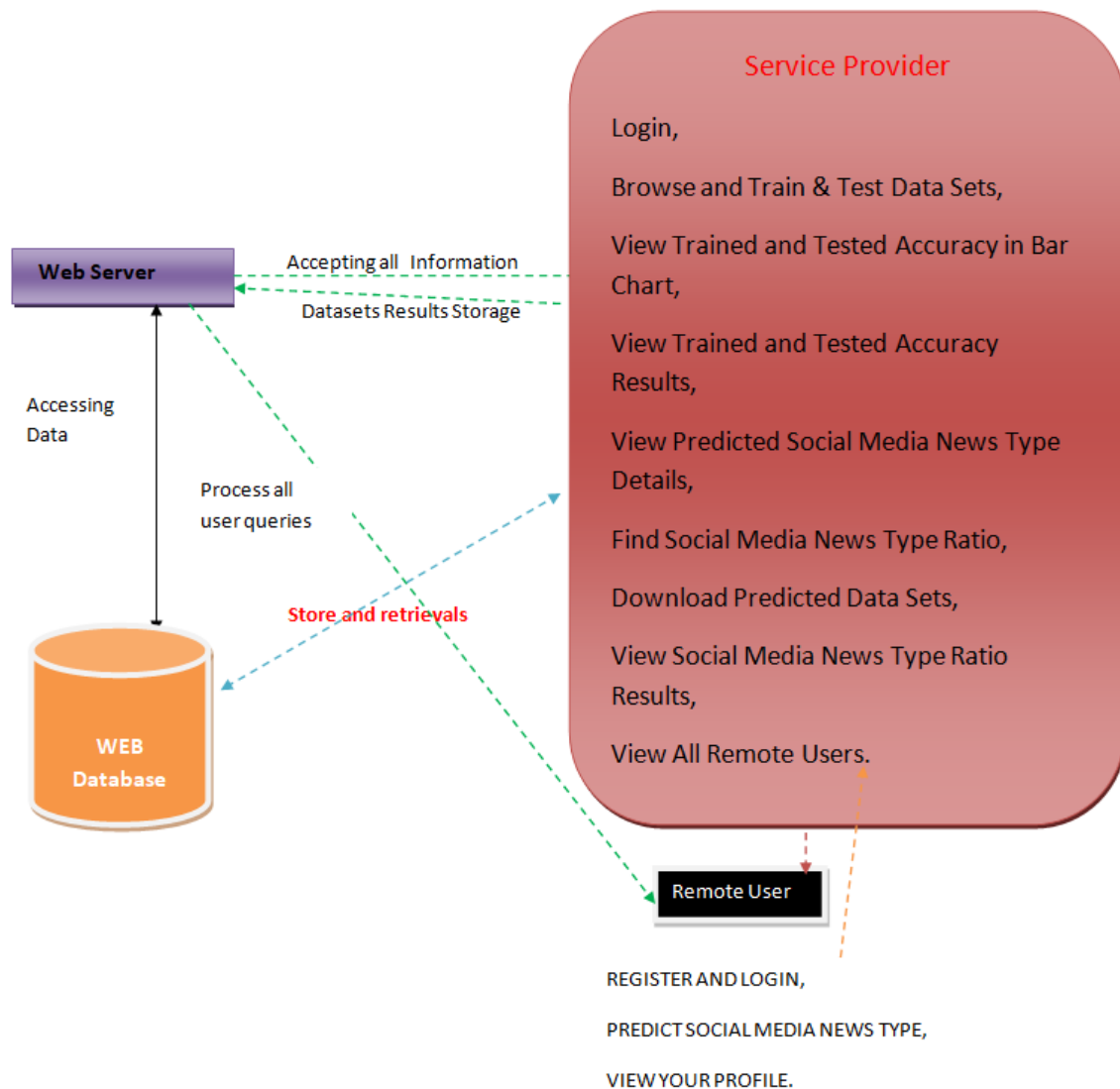
HARD REQUIRMENTS:

- **H/W System Configuration: -**
- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

SOFTWARE REQUIRMENTS:

- ❖ **Operating system** : Windows 7 Ultimate.
- ❖ **Coding Language** : Python.
- ❖ **Front-End** : Python.
- ❖ **Back-End** : Django-ORM
- ❖ **Designing** : Html, css, javascript.
- ❖ **Data Base** : MySQL (WAMP Server).

SYSTEM ARCHITECTURE



UML DIAGRAM'S :

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-

model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

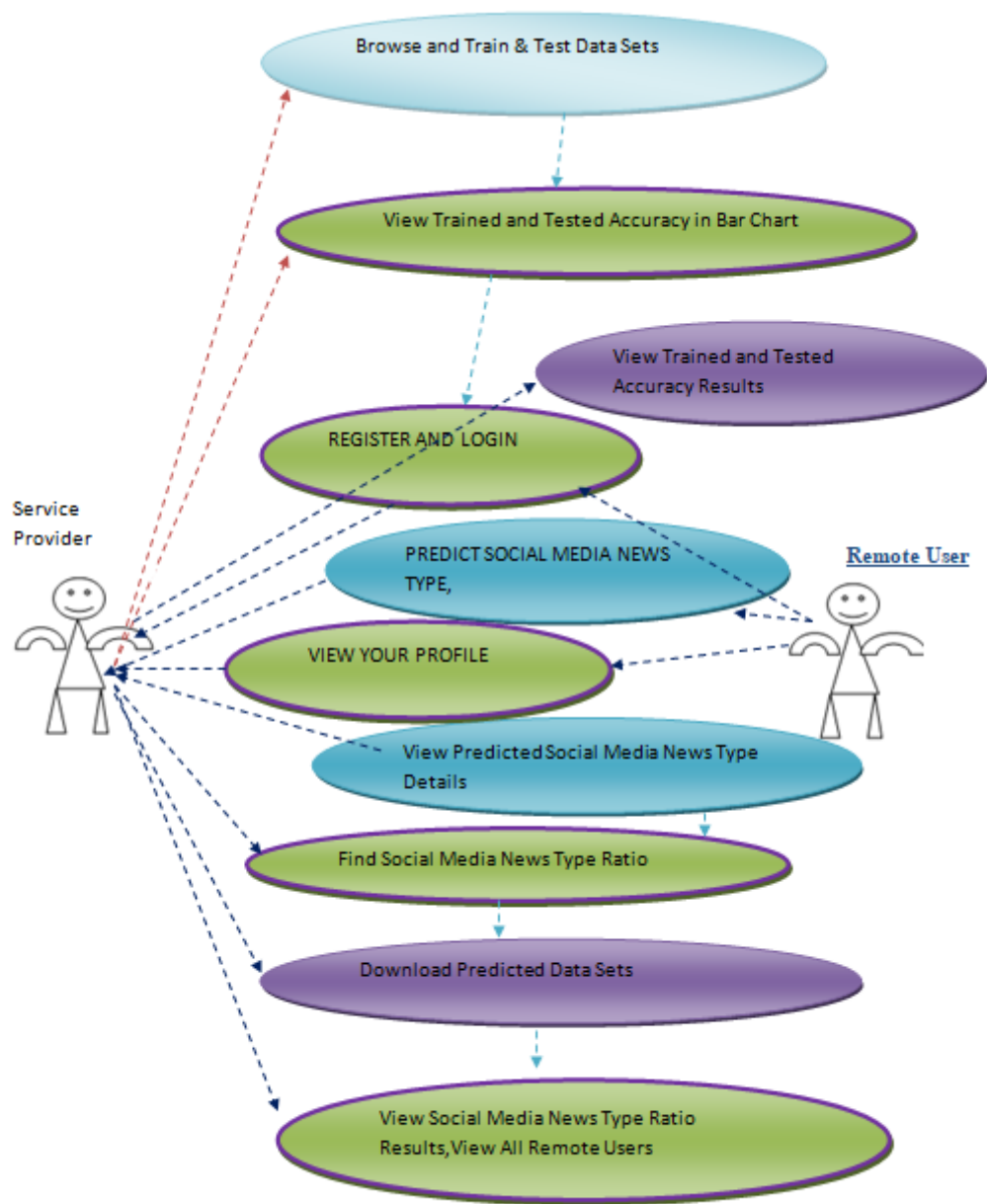
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

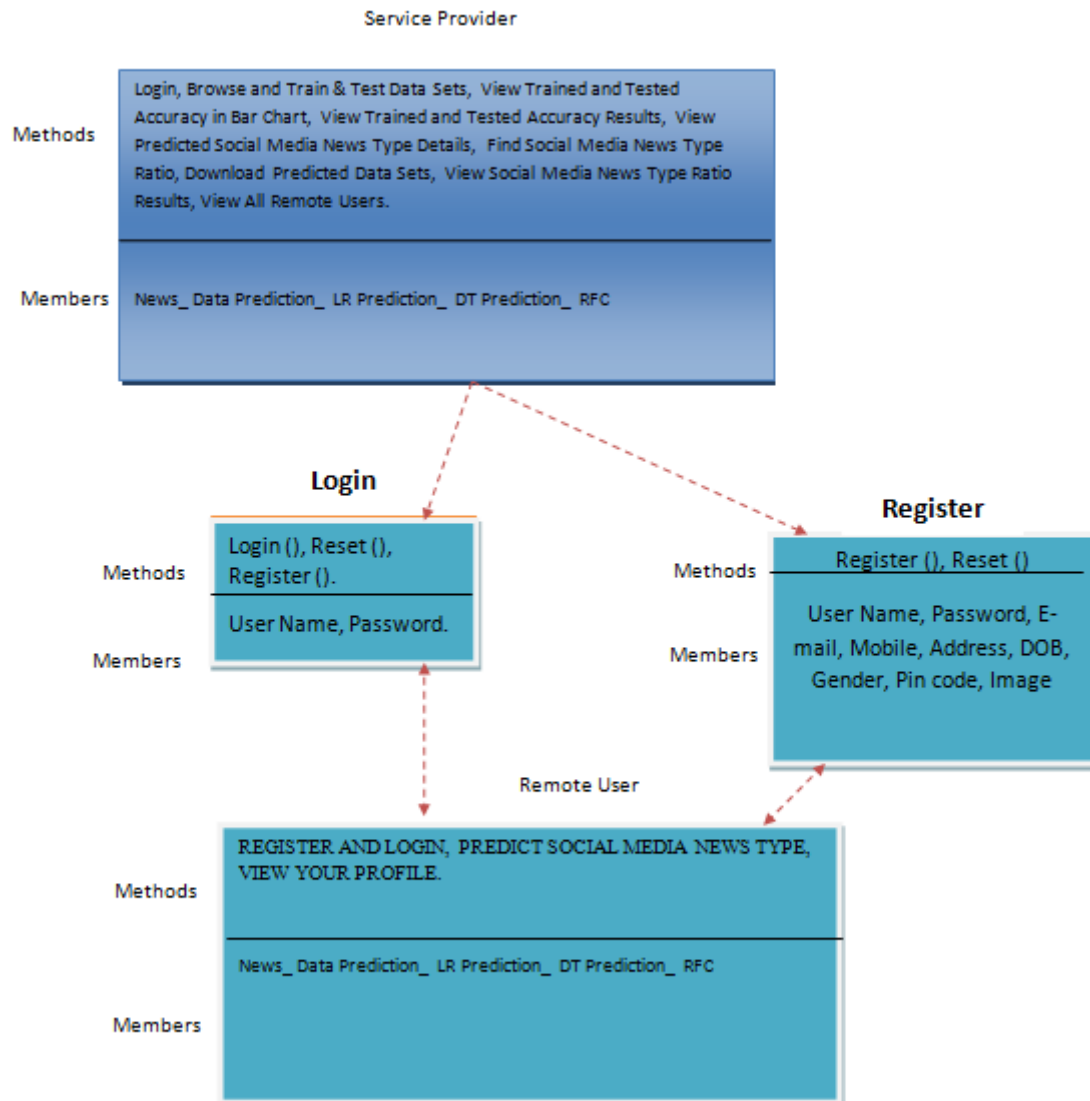
USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.



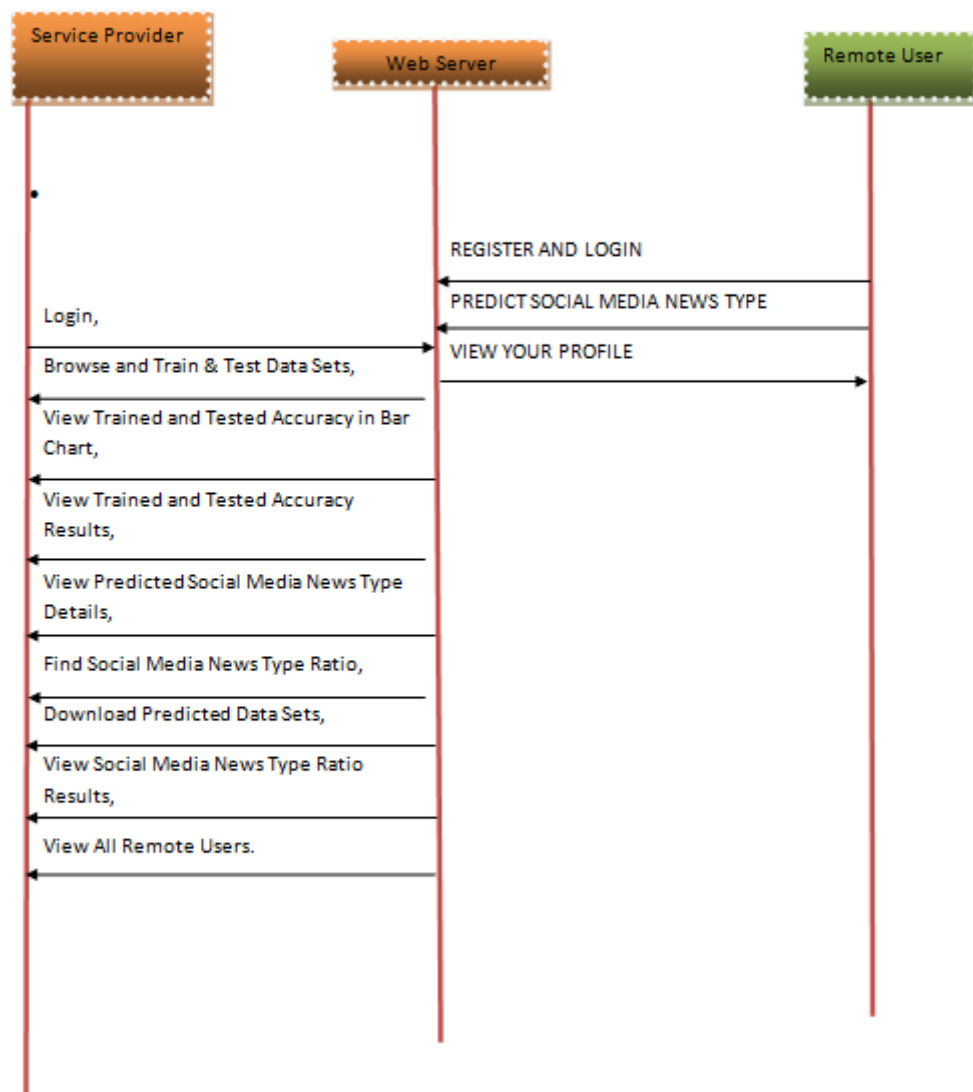
CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.



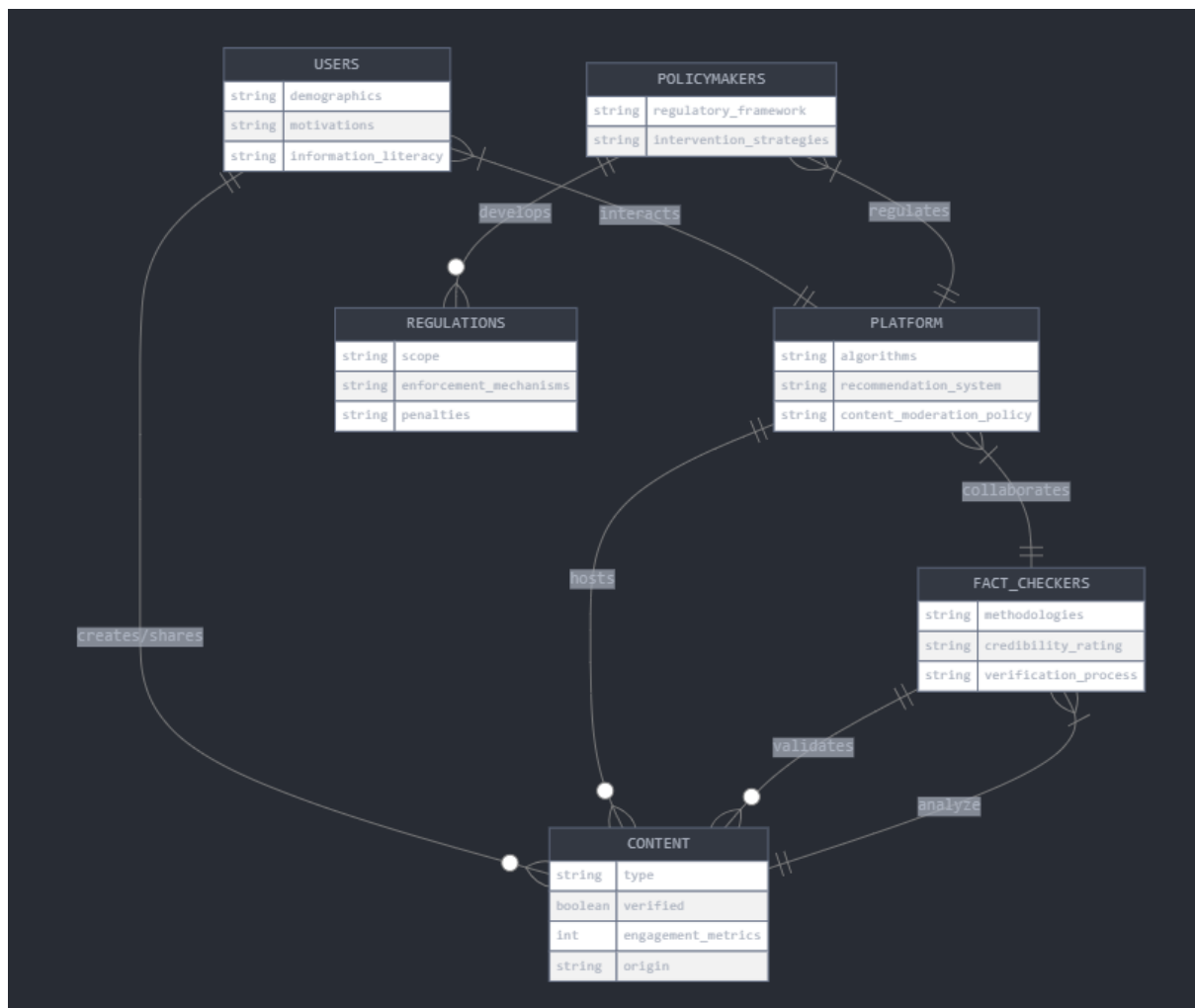
SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

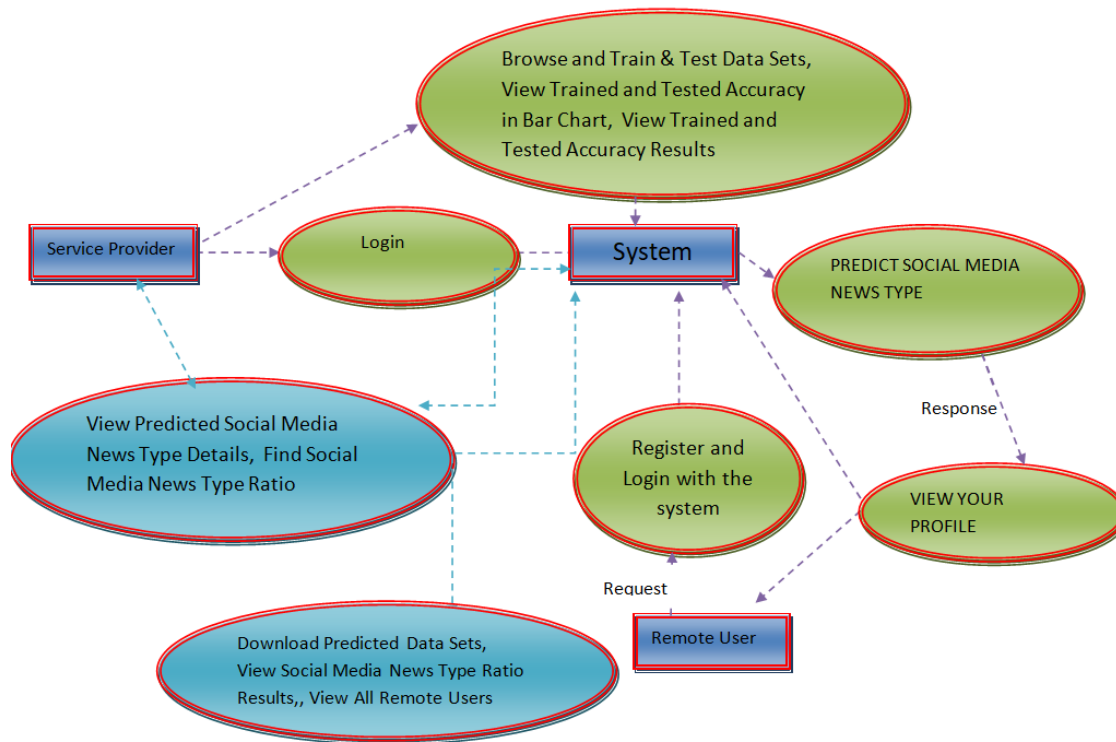


COLLABRATION DIAGRAM:

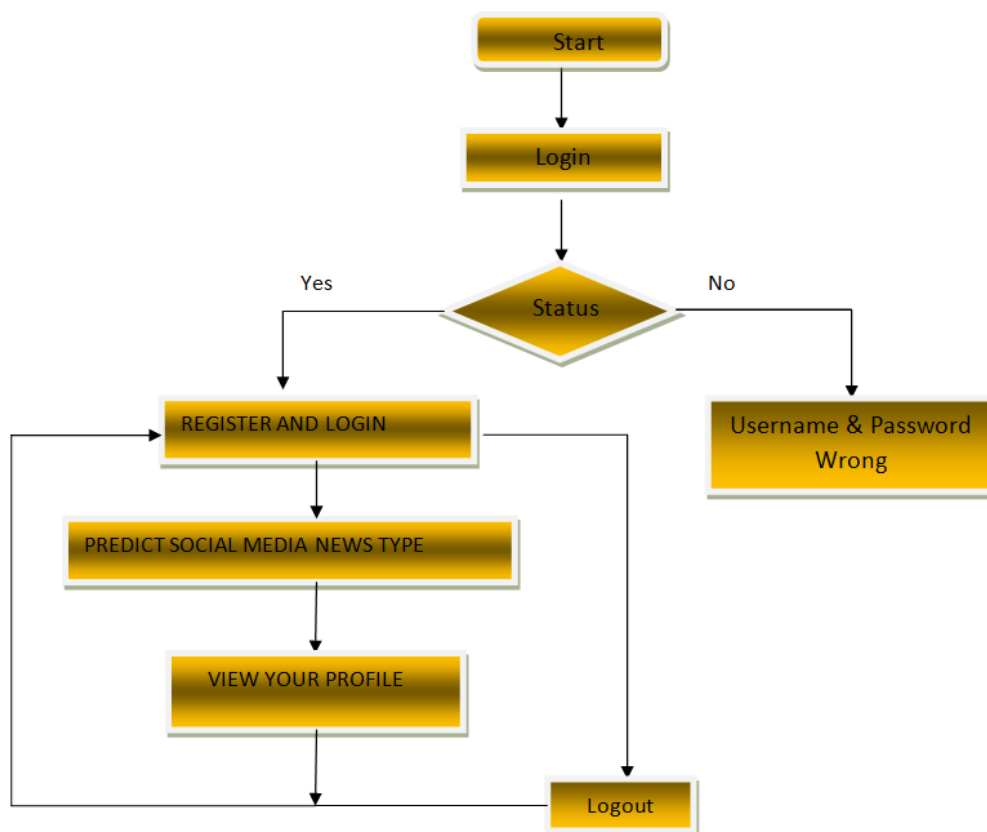
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



DATA FLOW DIAGRAM:



FLOW CHART:



IMPLEMENTATION

MODULES:

Service Provider

In this module, the Service Provider has to login by using valid user name and password. After login successful he can do some operations such as Login, Browse and Train & Test Data Sets, View Trained and Tested Accuracy in Bar Chart, View Trained and Tested Accuracy Results, View Predicted Social Media News Type Details, Find Social Media News Type Ratio, Download Predicted Data Sets, View Social Media News Type Ratio Results,, View All Remote Users.

View and Authorize Users

In this module, the admin can view the list of users who all registered. In this, the admin can view the user's details such as, user name, email, address and admin authorizes the users.

RemoteUser

In this module, there are n numbers of users are present. User should register before doing any operations. Once user registers, their details will be stored to the database. After registration successful, he has to login by using authorized user name and password. Once Login is successful user will do some operations like REGISTER AND LOGIN, PREDICT SOCIAL MEDIA NEWS TYPE, VIEW YOUR PROFILE.

SOFTWARE ENVIRONMENT

What is Python :

Below are some facts about Python.

Python is currently the most widely used multi-purpose, high-level programming language.

Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java.

Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time.

Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber... etc.

The biggest strength of Python is huge collection of standard library which can be used for the following –

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQtetc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like Opencv, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia

Advantages of Python :-

Let's see how Python dominates over other languages.

1. Extensive Libraries

Python downloads with an extensive library and it contain code for various purposes like regular expressions, documentation-generation, unit-testing, web browsers, threading, databases, CGI, email, image manipulation, and more. So, we don't have to write the complete code for that manually.

2. Extensible

As we have seen earlier, Python can be **extended to other languages**. You can write some of your code in languages like C++ or C. This comes in handy, especially in projects.

3. Embeddable

Complimentary to extensibility, Python is embeddable as well. You can put your Python code in your source code of a different language, like C++. This lets us add **scripting capabilities** to our code in the other language.

4. Improved Productivity

The language's simplicity and extensive libraries render programmers **more productive** than languages like Java and C++ do. Also, the fact that you need to write less and get more things done.

5. IOT Opportunities

Since Python forms the basis of new platforms like Raspberry Pi, it finds the future bright for the Internet Of Things. This is a way to connect the language with the real world.

When working with Java, you may have to create a class to print '**Hello World**'. But in Python, just a print statement will do. It is also quite **easy to learn, understand, and code**. This is why when people pick up Python, they have a hard time adjusting to other more verbose languages like Java.

7. Readable

Because it is not such a verbose language, reading Python is much like reading English. This is the reason why it is so easy to learn, understand, and code. It also does not need curly braces to define blocks, and **indentation is mandatory**. This further aids the readability of the code.

8. Object-Oriented

This language supports both the **procedural and object-oriented** programming paradigms. While functions help us with code reusability, classes and objects let us model the real world. A class allows the **encapsulation of data** and functions into one.

9. Free and Open-Source

Like we said earlier, Python is **freely available**. But not only can you **download Python** for free, but you can also download its source code, make changes to it, and even distribute it. It downloads with an extensive collection of libraries to help you with your tasks.

10. Portable

When you code your project in a language like C++, you may need to make some changes to it if you want to run it on another platform. But it isn't the same with Python. Here, you need to **code only once**, and you can run it anywhere. This is called **Write Once Run Anywhere (WORA)**. However, you need to be careful enough not to include any system-dependent features.

11. Interpreted

Lastly, we will say that it is an interpreted language. Since statements are executed one by one, **debugging is easier** than in compiled languages.

Any doubts till now in the advantages of Python? Mention in the comment section.

Advantages of Python Over Other Languages :

1. Less Coding

Almost all of the tasks done in Python requires less coding when the same task is done in other languages. Python also has an awesome standard library support, so you don't have to search for any third-party libraries to get your job done. This is the reason that many people suggest learning Python to beginners.

2. Affordable

Python is free therefore individuals, small companies or big organizations can leverage the free available resources to build applications. Python is popular and widely used so it gives you better community support.

The 2019 Github annual survey showed us that Python has overtaken Java in the most popular programming language category

3. Python is for Everyone

Python code can run on any machine whether it is Linux, Mac or Windows. Programmers need to learn different languages for different jobs but with Python, you can professionally build web apps, perform data analysis and **machine learning**, automate things, do web scraping and also build games and powerful visualizations. It is an all-rounder programming language.

Disadvantages of Python

So far, we've seen why Python is a great choice for your project. But if you choose it, you should be aware of its consequences as well. Let's now see the downsides of choosing Python over another language.

1. Speed Limitations

We have seen that Python code is executed line by line. But since Python is interpreted, it often results in **slow execution**. This, however, isn't a problem unless speed is a focal point for the project. In other words, unless high speed is a requirement, the benefits offered by Python are enough to distract us from its speed limitations.

2. Weak in Mobile Computing and Browsers

While it serves as an excellent server-side language, Python is much rarely seen on the **client-side**. Besides that, it is rarely ever used to implement smartphone-based applications. One such application is called **Carbonnelle**.

The reason it is not so famous despite the existence of Brython is that it isn't that secure.

3. Design Restrictions

As you know, Python is **dynamically-typed**. This means that you don't need to declare the type of variable while writing the code. It uses **duck-typing**. But wait, what's that? Well, it just means that if it looks like a duck, it must be a duck. While this is easy on the programmers during coding, it can **raise run-time errors**.

4. Underdeveloped Database Access Layers

Compared to more widely used technologies like **JDBC (Java DataBase Connectivity)** and **ODBC (Open DataBase Connectivity)**, Python's database access layers are a bit underdeveloped. Consequently, it is less often applied in huge enterprises.

5. Simple

No, we're not kidding. Python's simplicity can indeed be a problem. Take my example. I don't do Java, I'm more of a Python person. To me, its syntax is so simple that the verbosity of Java code seems unnecessary.

This was all about the Advantages and Disadvantages of Python Programming Language.

History of Python :-

What do the alphabet and the programming language Python have in common? Right, both start with ABC. If we are talking about ABC in the Python context, it's clear that the programming language ABC is meant. ABC is a general-purpose programming language and programming environment, which had been developed in the Netherlands, Amsterdam, at the CWI (Centrum Wiskunde&Informatica). The greatest achievement of ABC was to influence the design of Python. Python was conceptualized in the late 1980s. Guido van Rossum worked that time in a project at the CWI, called Amoeba, a distributed operating system. In an interview with Bill Venners¹, Guido van Rossum said: "In the early 1980s, I worked as an implementer on a team building a language called ABC at Centrum voor Wiskunde en Informatica (CWI). I don't know how well people know ABC's influence on Python. I try to mention ABC's influence because I'm indebted to everything I learned during that project and to the people who worked on it." Later on in the same Interview, Guido van Rossum continued: "I remembered all my experience and some of my frustration with ABC. I decided to try to design a simple scripting language that possessed some of ABC's better properties, but without its problems. So I started typing. I created a simple virtual machine, a simple parser, and a simple runtime. I made my own version of the various ABC parts that I liked. I created a basic syntax, used indentation for statement grouping instead of curly braces or begin-end blocks, and developed a small number of powerful data types: a hash table (or dictionary, as we call it), a list, strings, and numbers."

What is Machine Learning :-

Before we take a look at the details of various machine learning methods, let's start by looking at what machine learning is, and what it isn't. Machine learning is often categorized as a subfield of artificial intelligence, but I find that categorization can often be misleading at first brush. The study of machine learning certainly arose from research in this context, but in the data science application of machine learning methods, it's more helpful to think of machine learning as a means of building models of data.

Fundamentally, machine learning involves building mathematical models to help understand data. "Learning" enters the fray when we give these models tunable parameters that can be adapted to observed data; in this way the program can be considered to be "learning" from the data. Once these models have been fit to previously seen data, they can be used to predict and understand aspects of newly observed data. I'll leave to the reader the more philosophical digression regarding the extent to which this type of mathematical, model-based "learning" is similar to the "learning" exhibited by the human brain. Understanding the problem setting in machine learning is essential to using these tools effectively, and so we will start with some broad categorizations of the types of approaches we'll discuss here.

Categories Of Machine Learning :-

At the most fundamental level, machine learning can be categorized into two main types: supervised learning and unsupervised learning.

Supervised learning involves somehow modeling the relationship between measured features of data and some label associated with the data; once this model is determined, it can be used to apply labels to new, unknown data. This is further subdivided into classification tasks and regression tasks: in classification, the labels are discrete categories, while in regression, the labels are continuous quantities. We will see examples of both types of supervised learning in the following section.

Unsupervised learning involves modeling the features of a dataset without reference to any label, and is often described as "letting the dataset speak for itself." These models include

tasks such as clustering and dimensionality reduction. Clustering algorithms identify distinct groups of data, while dimensionality reduction algorithms search for more succinct representations of the data. We will see examples of both types of unsupervised learning in the following section.

Need for Machine Learning

Human beings, at this moment, are the most intelligent and advanced species on earth because they can think, evaluate and solve complex problems. On the other side, AI is still in its initial stage and haven't surpassed human intelligence in many aspects. Then the question is that what is the need to make machine learn? The most suitable reason for doing this is, "to make decisions, based on data, with efficiency and scale".

Lately, organizations are investing heavily in newer technologies like Artificial Intelligence, Machine Learning and Deep Learning to get the key information from data to perform several real-world tasks and solve problems. We can call it data-driven decisions taken by machines, particularly to automate the process. These data-driven decisions can be used, instead of using programming logic, in the problems that cannot be programmed inherently. The fact is that we can't do without human intelligence, but other aspect is that we all need to solve real-world problems with efficiency at a huge scale. That is why the need for machine learning arises.

Challenges in Machines Learning :-

While Machine Learning is rapidly evolving, making significant strides with cybersecurity and autonomous cars, this segment of AI as whole still has a long way to go. The reason behind is that ML has not been able to overcome number of challenges. The challenges that ML is facing currently are –

Quality of data – Having good-quality data for ML algorithms is one of the biggest challenges. Use of low-quality data leads to the problems related to data preprocessing and feature extraction.

Time-Consuming task – Another challenge faced by ML models is the consumption of time especially for data acquisition, feature extraction and retrieval.

Lack of specialist persons – As ML technology is still in its infancy stage, availability of expert resources is a tough job.

No clear objective for formulating business problems – Having no clear objective and well-defined goal for business problems is another key challenge for ML because this technology is not that mature yet.

Issue of overfitting & underfitting – If the model is overfitting or underfitting, it cannot be represented well for the problem.

Curse of dimensionality – Another challenge ML model faces is too many features of data points. This can be a real hindrance.

Difficulty in deployment – Complexity of the ML model makes it quite difficult to be deployed in real life.

Applications of Machines Learning :-

Machine Learning is the most rapidly growing technology and according to researchers we are in the golden year of AI and ML. It is used to solve many real-world complex problems which cannot be solved with traditional approach. Following are some real-world applications of ML –

- Emotion analysis
- Sentiment analysis
- Error detection and prevention
- Weather forecasting and prediction
- Stock market analysis and forecasting
- Speech synthesis
- Speech recognition
- Customer segmentation
- Object recognition
- Fraud detection
- Fraud prevention
- Recommendation of products to customer in online shopping

How to Start Learning Machine Learning?

Arthur Samuel coined the term “**Machine Learning**” in 1959 and defined it as a “**Field of study that gives computers the capability to learn without being explicitly programmed**”.

And that was the beginning of Machine Learning! In modern times, Machine Learning is one of the most popular (if not the most!) career choices. According to Indeed, Machine Learning Engineer Is The Best Job of 2019 with a 344% growth and an average base salary of **\$146,085** per year.

But there is still a lot of doubt about what exactly is Machine Learning and how to start learning it? So this article deals with the Basics of Machine Learning and also the path you can follow to eventually become a full-fledged Machine Learning Engineer. Now let's get started!!!

How to start learning ML?

This is a rough roadmap you can follow on your way to becoming an insanely talented Machine Learning Engineer. Of course, you can always modify the steps according to your needs to reach your desired end-goal!

Step 1 – Understand the Prerequisites

In case you are a genius, you could start ML directly but normally, there are some prerequisites that you need to know which include Linear Algebra, Multivariate Calculus, Statistics, and Python. And if you don't know these, never fear! You don't need a Ph.D. degree in these topics to get started but you do need a basic understanding.

(a) Learn Linear Algebra and Multivariate Calculus

Both Linear Algebra and Multivariate Calculus are important in Machine Learning. However, the extent to which you need them depends on your role as a data scientist. If you are more focused on application heavy machine learning, then you will not be that heavily focused on maths as there are many common libraries available. But if you want to focus

on R&D in Machine Learning, then mastery of Linear Algebra and Multivariate Calculus is very important as you will have to implement many ML algorithms from scratch.

(b) Learn Statistics

Data plays a huge role in Machine Learning. In fact, around 80% of your time as an ML expert will be spent collecting and cleaning data. And statistics is a field that handles the collection, analysis, and presentation of data. So it is no surprise that you need to learn it!!! Some of the key concepts in statistics that are important are Statistical Significance, Probability Distributions, Hypothesis Testing, Regression, etc. Also, Bayesian Thinking is also a very important part of ML which deals with various concepts like Conditional Probability, Priors, and Posteriors, Maximum Likelihood, etc.

(c) Learn Python

Some people prefer to skip Linear Algebra, Multivariate Calculus and Statistics and learn them as they go along with trial and error. But the one thing that you absolutely cannot skip is Python! While there are other languages you can use for Machine Learning like R, Scala, etc. Python is currently the most popular language for ML. In fact, there are many Python libraries that are specifically useful for Artificial Intelligence and Machine Learning such as Keras, TensorFlow, Scikit-learn, etc.

So if you want to learn ML, it's best if you learn Python! You can do that using various online resources and courses such as **Fork Python** available Free on GeeksforGeeks.

Step 2 – Learn Various ML Concepts

Now that you are done with the prerequisites, you can move on to actually learning ML (Which is the fun part!!!) It's best to start with the basics and then move on to the more complicated stuff. Some of the basic concepts in ML are:

(a) Terminologies of Machine Learning

- **Model** – A model is a specific representation learned from data by applying some machine learning algorithm. A model is also called a hypothesis.

- **Feature** – A feature is an individual measurable property of the data. A set of numeric features can be conveniently described by a feature vector. Feature vectors are fed as input to the model. For example, in order to predict a fruit, there may be features like color, smell, taste, etc.
- **Target (Label)** – A target variable or label is the value to be predicted by our model. For the fruit example discussed in the feature section, the label with each set of input would be the name of the fruit like apple, orange, banana, etc.
- **Training** – The idea is to give a set of inputs(features) and it's expected outputs(labels), so after training, we will have a model (hypothesis) that will then map new data to one of the categories trained on.
- **Prediction** – Once our model is ready, it can be fed a set of inputs to which it will provide a predicted output(label).

(b) Types of Machine Learning

- **Supervised Learning** – This involves learning from a training dataset with labeled data using classification and regression models. This learning process continues until the required level of performance is achieved.
- **Unsupervised Learning** – This involves using unlabelled data and then finding the underlying structure in the data in order to learn more and more about the data itself using factor and cluster analysis models.
- **Semi-supervised Learning** – This involves using unlabelled data like Unsupervised Learning with a small amount of labeled data. Using labeled data vastly increases the learning accuracy and is also more cost-effective than Supervised Learning.
- **Reinforcement Learning** – This involves learning optimal actions through trial and error. So the next action is decided by learning behaviors that are based on the current state and that will maximize the reward in the future.

Advantages of Machine learning :-

1. Easily identifies trends and patterns -

Machine Learning can review large volumes of data and discover specific trends and patterns that would not be apparent to humans. For instance, for an e-commerce website like Amazon, it serves to understand the browsing behaviors and purchase histories of its users to help cater

to the right products, deals, and reminders relevant to them. It uses the results to reveal relevant advertisements to them.

2. No human intervention needed (automation)

With ML, you don't need to babysit your project every step of the way. Since it means giving machines the ability to learn, it lets them make predictions and also improve the algorithms on their own. A common example of this is anti-virus softwares; they learn to filter new threats as they are recognized. ML is also good at recognizing spam.

3. Continuous Improvement

As **ML algorithms** gain experience, they keep improving in accuracy and efficiency. This lets them make better decisions. Say you need to make a weather forecast model. As the amount of data you have keeps growing, your algorithms learn to make more accurate predictions faster.

4. Handling multi-dimensional and multi-variety data

Machine Learning algorithms are good at handling data that are multi-dimensional and multi-variety, and they can do this in dynamic or uncertain environments.

5. Wide Applications

You could be an e-tailer or a healthcare provider and make ML work for you. Where it does apply, it holds the capability to help deliver a much more personal experience to customers while also targeting the right customers.

Disadvantages of Machine Learning :-

1. Data Acquisition

Machine Learning requires massive data sets to train on, and these should be inclusive/unbiased, and of good quality. There can also be times where they must wait for new data to be generated.

2. Time and Resources

ML needs enough time to let the algorithms learn and develop enough to fulfill their purpose with a considerable amount of accuracy and relevancy. It also needs massive resources to function. This can mean additional requirements of computer power for you.

3. Interpretation of Results

Another major challenge is the ability to accurately interpret results generated by the algorithms. You must also carefully choose the algorithms for your purpose.

4. High error-susceptibility

Machine Learning is autonomous but highly susceptible to errors. Suppose you train an algorithm with data sets small enough to not be inclusive. You end up with biased predictions coming from a biased training set. This leads to irrelevant advertisements being displayed to customers. In the case of ML, such blunders can set off a chain of errors that can go undetected for long periods of time. And when they do get noticed, it takes quite some time to recognize the source of the issue, and even longer to correct it.

Python Development Steps : -

Guido Van Rossum published the first version of Python code (version 0.9.0) at alt.sources in February 1991. This release included already exception handling, functions, and the core data types of list, dict, str and others. It was also object oriented and had a module system. Python version 1.0 was released in January 1994. The major new features included in this release were the functional programming tools lambda, map, filter and reduce, which Guido Van Rossum never liked. Six and a half years later in October 2000, Python 2.0 was introduced. This release included list comprehensions, a full garbage collector and it was supporting unicode. Python flourished for another 8 years in the versions 2.x before the next major release as Python 3.0 (also known as "Python 3000" and "Py3K") was released. Python 3 is not backwards compatible with Python 2.x. The emphasis in Python 3 had been on the

removal of duplicate programming constructs and modules, thus fulfilling or coming close to fulfilling the 13th law of the Zen of Python: "There should be one -- and preferably only one -- obvious way to do it."Some changes in Python 7.3:

- Print is now a function
- Views and iterators instead of lists
- The rules for ordering comparisons have been simplified. E.g. a heterogeneous list cannot be sorted, because all the elements of a list must be comparable to each other.
- There is only one integer type left, i.e. int. long is int as well.
- The division of two integers returns a float instead of an integer. "/" can be used to have the "old" behaviour.
- Text Vs. Data Instead Of Unicode Vs. 8-bit

Purpose :-

We demonstrated that our approach enables successful segmentation of intra-retinal layers—even with low-quality images containing speckle noise, low contrast, and different intensity ranges throughout—with the assistance of the ANIS feature.

Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems

or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Modules Used in Project :-

Tensorflow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. **Python**

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant whitespace.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background - without breaking.

Install Python Step-by-Step in Windows and Mac :

Python a versatile programming language doesn't come pre-installed on your computer devices. Python was first released in the year 1991 and until today it is a very popular high-level programming language. Its style philosophy emphasizes code readability with its notable use of great whitespace.

The object-oriented approach and language construct provided by Python enables programmers to write both clear and logical code for projects. This software does not come pre-packaged with Windows.

How to Install Python on Windows and Mac :

There have been several updates in the Python version over the years. The question is how to install Python? It might be confusing for the beginner who is willing to start learning Python but this tutorial will solve your query. The latest or the newest version of Python is version 3.7.4 or in other words, it is Python 3.

Note: The python version 3.7.4 cannot be used on Windows XP or earlier devices.

Before you start with the installation process of Python. First, you need to know about your **System Requirements**. Based on your system type i.e. operating system and based processor, you must download the python version. My system type is a **Windows 64-bit operating system**. So the steps below are to install python version 3.7.4 on Windows 7 device or to install Python 3. [Download the Python Cheatsheet here](#). The steps on how to install Python on Windows 10, 8 and 7 are **divided into 4 parts** to help understand better.

Download the Correct version into the system

Step 1: Go to the official site to download and install python using Google Chrome or any other web browser. OR Click on the following link: <https://www.python.org>



Now, check for the latest and the correct version for your operating system.








Step 2: Click on the Download Tab.



Step 3: You can either select the Download Python for windows 3.7.4 button in Yellow Color or you can scroll further down and click on download with respective to their version. Here, we are downloading the most recent python version for windows 3.7.4

Looking for a specific release?

Python releases by version number:

Release version	Release date		Click for more
Python 3.7.4	July 8, 2019	 Download	Release Notes
Python 3.6.9	July 2, 2019	 Download	Release Notes
Python 3.7.3	March 25, 2019	 Download	Release Notes
Python 3.4.10	March 18, 2019	 Download	Release Notes
Python 3.5.7	March 18, 2019	 Download	Release Notes
Python 2.7.16	March 4, 2019	 Download	Release Notes
Python 3.7.2	Dec. 24, 2018	 Download	Release Notes

Step 4: Scroll down the page until you find the Files option.

Step 5: Here you see a different version of python along with the operating system.

Files					
Version	Operating System	Description	MD5 Sum	File Size	GPG
Cupped source tarball	Source release		68111671e5b2db4ae77b9ab01b0f9be	23017663	5xG
XZ compressed source tarball	Source release		d53e4aee6097051c2eca45ee3604803	17131432	5xG
macOS 64-bit/32-bit installer	Mac OS X	for Mac OS X 10.6 and later	6428b4fa7583da91a42cba1ee08e6	34898436	5xG
macOS 64-bit installer	Mac OS X	for OS X 10.9 and later	5d4605c38217a457738f5e4e936b241f	28082845	5xG
Windows help file	Windows		d83999573a2c56b2ac56cade6b47cfd2	8131761	5xG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64/x64	9b00c3c8cd9ec0b9abe63184a0728a2	7504291	5xG
Windows x86-64 executable installer	Windows	for AMD64/EM64/x64	a702b4b0ad76d8bdc3543a583e563400	26680368	5xG
Windows x86-64 web-based installer	Windows	for AMD64/EM64/x64	28c31c90ff8ed72ae9e53a3bd351b4bd2	1362904	5xG
Windows x86 embeddable zip file	Windows		9fab3b618b41879fda94133574139d9	6741626	5xG
Windows x86 executable installer	Windows		33c902942a5446a3d8451478394789	25663848	5xG
Windows x86 web-based installer	Windows		1b670cfa5d317d82c30983ea371d87c	1324608	5xG

- To download Windows 32-bit python, you can select any one from the three options:
Windows x86 embeddable zip file, Windows x86 executable installer or Windows x86 web-based installer.

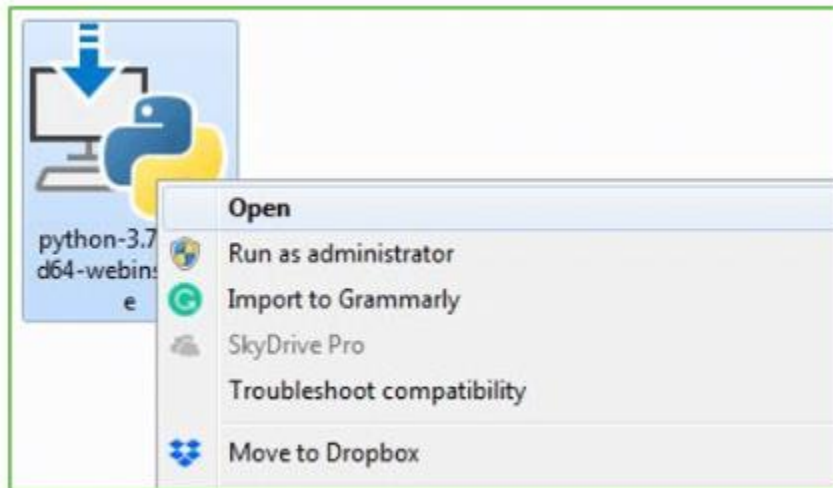
- To download Windows 64-bit python, you can select any one from the three options:
Windows x86-64 embeddable zip file, Windows x86-64 executable installer or Windows x86-64 web-based installer.

Here we will install Windows x86-64 web-based installer. Here your first part regarding which version of python is to be downloaded is completed. Now we move ahead with the second part in installing python i.e. Installation

Note: To know the changes or updates that are made in the version you can click on the Release Note Option.

Installation of Python

Step 1: Go to Download and Open the downloaded python version to carry out the installation process



2: Before you click on Install Now, Make sure to put a tick on Add Python 3.7 to PATH.



Step 3: Click on Install NOW After the installation is successful. Click on Close.



With these above three steps on python installation, you have successfully and correctly installed Python. Now is the time to verify the installation.

Note: The installation process might take a couple of minutes.

Verify the Python Installation

Step 1: Click on Start

Step 2: In the Windows Run Command, type “cmd”.



Step 3: Open the Command prompt option.

Step 4: Let us test whether the python is correctly installed. Type **python -V** and press Enter.

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.
C:\Users\DELL>python -V
Python 3.7.4
C:\Users\DELL>_
```

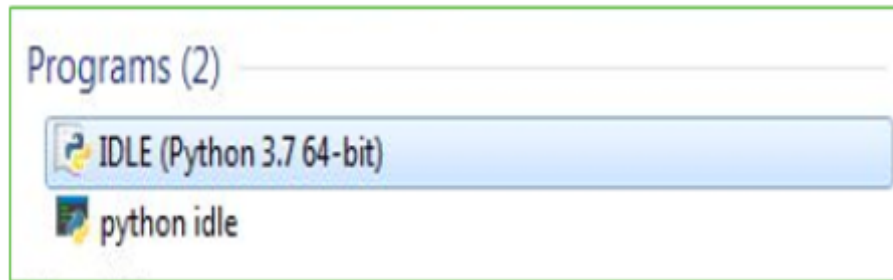
Step 5: You will get the answer as 3.7.4

Note: If you have any of the earlier versions of Python already installed. You must first uninstall the earlier version and then install the new one.

Check how the Python IDLE works

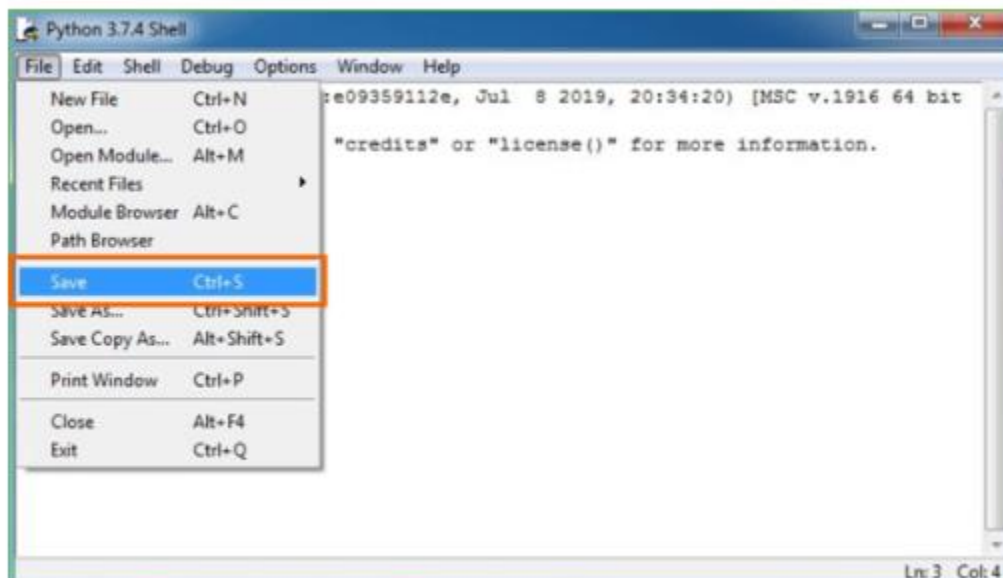
Step 1: Click on Start

Step 2: In the Windows Run command, type “python idle”.



Step 3: Click on IDLE (Python 3.7 64-bit) and launch the program

Step 4: To go ahead with working in IDLE you must first save the file. **Click on File > Click on Save**



Step 5: Name the file and save as type should be Python files. Click on SAVE. Here I have named the files as Hey World.

Step 6: Now for e.g. **enter print**

SAMPLE CODE

```
import random

from typing import List, Dict, Tuple

from dataclasses import dataclass, field


@dataclass
class User:

    id: str

    information_literacy: float # 0-1 scale

    susceptibility_to_misinfo: float # 0-1 scale

    trust_network: List[str] = field(default_factory=list)


@dataclass
class Content:

    id: str

    credibility_score: float # 0-1 scale

    source_reliability: float # 0-1 scale

    topic: str

    content_type: str


class MisinformationMechanismDesign:
```

```

def __init__(self, users: List[User], contents:
List[Content]):

    self.users = {user.id: user for user in users}

    self.contents = {content.id: content for content in
contents}

    self.content_propagation_history = {}

    self.fact_checker_scores = {}


def calculate_content_spread_probability(self, content:
Content, user: User) -> float:

    """
    Calculates probability of content spread based on user
characteristics
    and content credibility
    """

    literacy_factor = 1 - user.information_literacy

    susceptibility_factor = user.susceptibility_to_misinfo

    credibility_factor = 1 - content.credibility_score

    spread_probability = (

        0.5 * literacy_factor +

        0.3 * susceptibility_factor +

        0.2 * credibility_factor

```

```

    )

    return min(max(spread_probability, 0), 1)

    def simulate_information_propagation(self,
initial_content_id: str, num_iterations: int = 10):
    """
    Simulate how misinformation spreads through social
networks
    """
    current_content = self.contents[initial_content_id]
    propagated_users = set()

    for iteration in range(num_iterations):
        new_propagated_users = set()

        for user_id, user in self.users.items():
            if user_id in propagated_users:
                continue

            # Probabilistic spread based on user and
content characteristics

```

```

        spread_prob =
self.calculate_content_spread_probability(current_content,
user)

        if random.random() < spread_prob:

            new_propagated_users.add(user_id)

            # Update fact-checking score

self.update_fact_checker_score(current_content, user)

        propagated_users.update(new_propagated_users)

    return len(propagated_users)

    def update_fact_checker_score(self, content: Content,
user: User):
        """
        Track how different types of users interact with
        potentially misleading content
        """

        score_key = (content.topic, content.content_type)

        if score_key not in self.fact_checker_scores:

```

```

        self.fact_checker_scores[score_key] = {
            'total_interactions': 0,
            'high_literacy_interactions': 0
        }

self.fact_checker_scores[score_key]['total_interactions'] += 1

        if user.information_literacy > 0.7:

self.fact_checker_scores[score_key]['high_literacy_interaction
s'] += 1

    def generate_intervention_recommendation(self) ->
Dict[str, float]:
        """
        Generate recommendations for mitigating misinformation
        spread
        """
        intervention_scores = {}

        for (topic, content_type), score in
self.fact_checker_scores.items():
            if score['total_interactions'] > 0:

```

```

        high_literacy_ratio = (
            score['high_literacy_interactions'] /
            score['total_interactions']
        )

        intervention_scores[(topic, content_type)] =
high_literacy_ratio

    return intervention_scores

# Example usage

def main():

    # Create sample users with varying information literacy

    users = [

        User(id='user1', information_literacy=0.2,
susceptibility_to_misinfo=0.8),

        User(id='user2', information_literacy=0.7,
susceptibility_to_misinfo=0.3),

        User(id='user3', information_literacy=0.5,
susceptibility_to_misinfo=0.5)

    ]

    # Create sample contents with different credibility

    contents = [

```

```

        Content(id='content1', credibility_score=0.2,
source_reliability=0.3,
                topic='politics', content_type='opinion'),
        Content(id='content2', credibility_score=0.8,
source_reliability=0.9,
                topic='science', content_type='research')
    ]

    # Initialize mechanism design
    mechanism = MisinformationMechanismDesign(users, contents)

    # Simulate misinformation spread
    spread_users =
mechanism.simulate_information_propagation('content1')

    print(f"Misinformation spread to {spread_users} users")

    # Generate intervention recommendations
    interventions =
mechanism.generate_intervention_recommendation()

    print("Intervention Recommendations:", interventions)

if __name__ == "__main__":
    main()

```


SYSTEM TEST

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing :

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

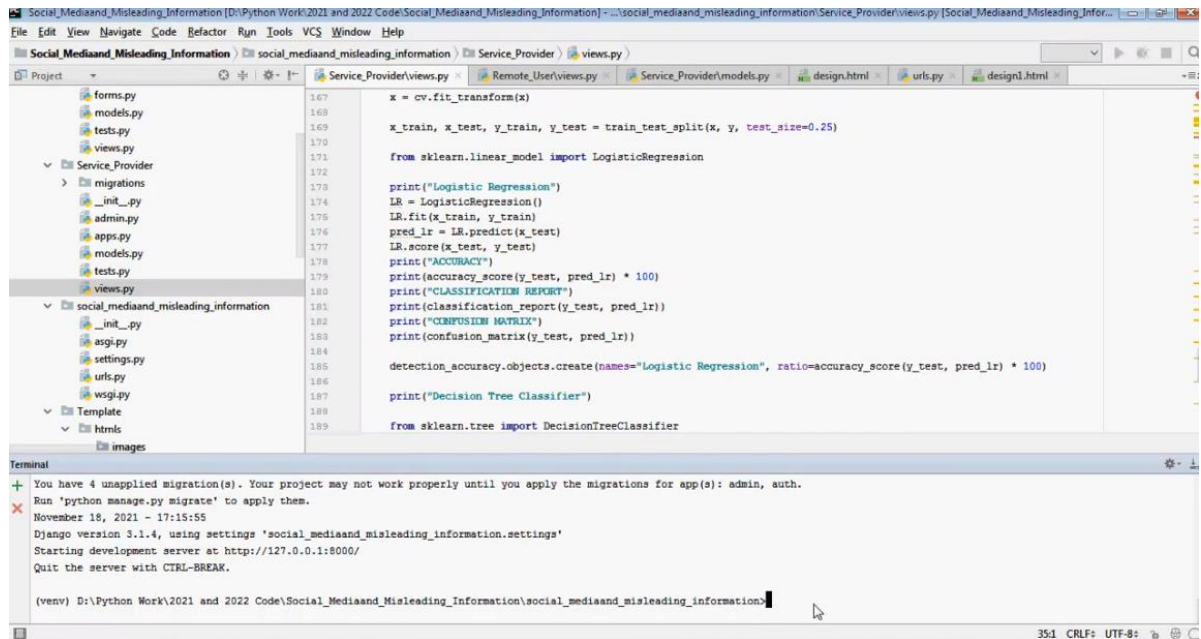
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

RESULTS AND OUTPUTS SCREENS

Unapplied Migrations: The terminal indicates there are 4 unapplied migrations. To resolve this, run `python manage.py migrate`.

Logistic Regression Implementation: The `views.py` file is implementing and evaluating machine learning models, such as Logistic Regression, using scikit-learn.



The screenshot shows a code editor with the `views.py` file open. The code implements a Logistic Regression model using scikit-learn. It includes imports for `cv`, `train_test_split`, `LogisticRegression`, and `DecisionTreeClassifier`. The code defines a function `train_test_split` and a function `logistic_regression` that prints the accuracy score and confusion matrix. The terminal window shows the output of the command `python manage.py migrate`, indicating that there are 4 unapplied migrations for the `admin` and `auth` apps.

```
167 x = cv.fit_transform(x)
168
169 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
170
171 from sklearn.linear_model import LogisticRegression
172
173 print("Logistic Regression")
174 LR = LogisticRegression()
175 LR.fit(x_train, y_train)
176 pred_lr = LR.predict(x_test)
177 LR.score(x_test, y_test)
178 print("ACCURACY")
179 print(accuracy_score(y_test, pred_lr) * 100)
180 print("CLASSIFICATION REPORT")
181 print(classification_report(y_test, pred_lr))
182 print("CONFUSION MATRIX")
183 print(confusion_matrix(y_test, pred_lr))
184
185 detection_accuracy.objects.create(name="Logistic Regression", ratio=accuracy_score(y_test, pred_lr) * 100)
186
187 print("Decision Tree Classifier")
188
189 from sklearn.tree import DecisionTreeClassifier
```

Terminal output:

```
+ You have 4 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin, auth.
+ Run 'python manage.py migrate' to apply them.
X November 18, 2021 - 17:15:55
Django version 3.1.4, using settings 'social_mediaand_misleading_information.settings'
Starting development server at http://127.0.0.1:8000/
Quit the server with CTRL-BREAK.

(venv) D:\Python Work\2021 and 2022 Code\Social_Mediaand_Misleading_Information\social_mediaand_misleading_information>
```

Social Media and Misleading Information in a Democracy: A Mechanism Design Approach



- There's a large shield icon with a padlock inside, likely symbolizing security and protection.

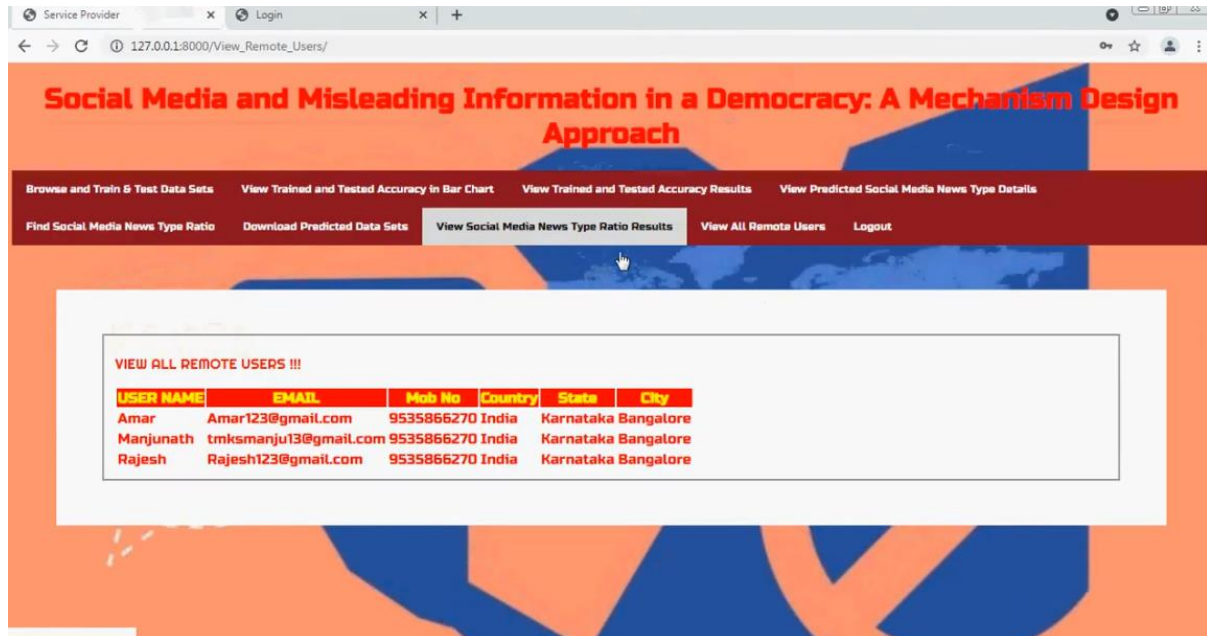
- The phrase "FACTS" is prominently displayed, emphasizing the importance of accurate information.
- A login form is visible, suggesting that the page is part of an online platform or system.



- A globe in the background, likely symbolizing the global reach of social media and the spread of misinformation.
- A hand holding a speech bubble with a question mark, possibly representing the search for truth and the questioning of information.
- A "FACTS" label with a crossed-out arrow, emphasizing the importance of verifying information and avoiding fake news.



1. **Study and analyze social media content:** To identify patterns, trends, and potential misinformation.
2. **Develop and test algorithms or models:** To detect and classify fake news or misleading information.
3. **Engage remote users:** To collect data, provide feedback, or participate in the research process.



The terminal output shows a series of HTTP requests, indicating that the application is interacting with a web server. The presence of a "Register1" endpoint suggests that users can register with the service, while "View Remote Users" implies that the application can track and manage users who are interacting with it remotely.

The warning messages about setting values on a copy of a slice from a DataFrame indicate potential issues with how the code is manipulating data. These warnings should be addressed to ensure the code's correctness and maintainability.

The screenshot shows an IDE window titled "Social_Mediaand_Misleading_Information". The file explorer on the left shows a project structure with files like forms.py, models.py, tests.py, views.py, and a Service_Provider directory. The main editor displays a Python file (views.py) with the following code:

```
167 x = cv.fit_transform(x)
168
169 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
170
171 from sklearn.linear_model import LogisticRegression
172
173 print("Logistic Regression")
174 LR = LogisticRegression()
```

The terminal at the bottom shows a series of HTTP requests and responses, including GET /favicon.ico, GET / HTTP/1.1, GET /Register1/, GET /static/Register.jpg, POST /Register1/, GET / HTTP/1.1, GET /View_Remote_Users/, POST / HTTP/1.1, and GET /ViewYourProfile/. It also displays a warning from pandas about setting values on a copy of a slice from a DataFrame.

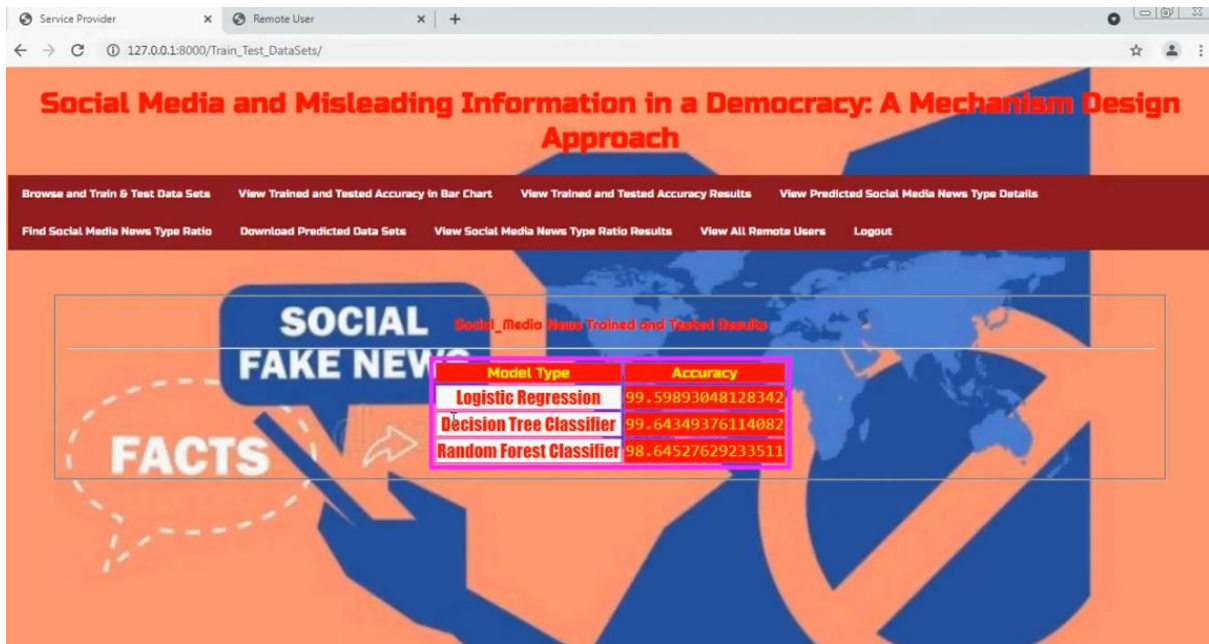
The screenshot shows the same IDE window as above, but the terminal now displays the output of a machine learning model evaluation. It includes a confusion matrix, a classification report, and another confusion matrix.

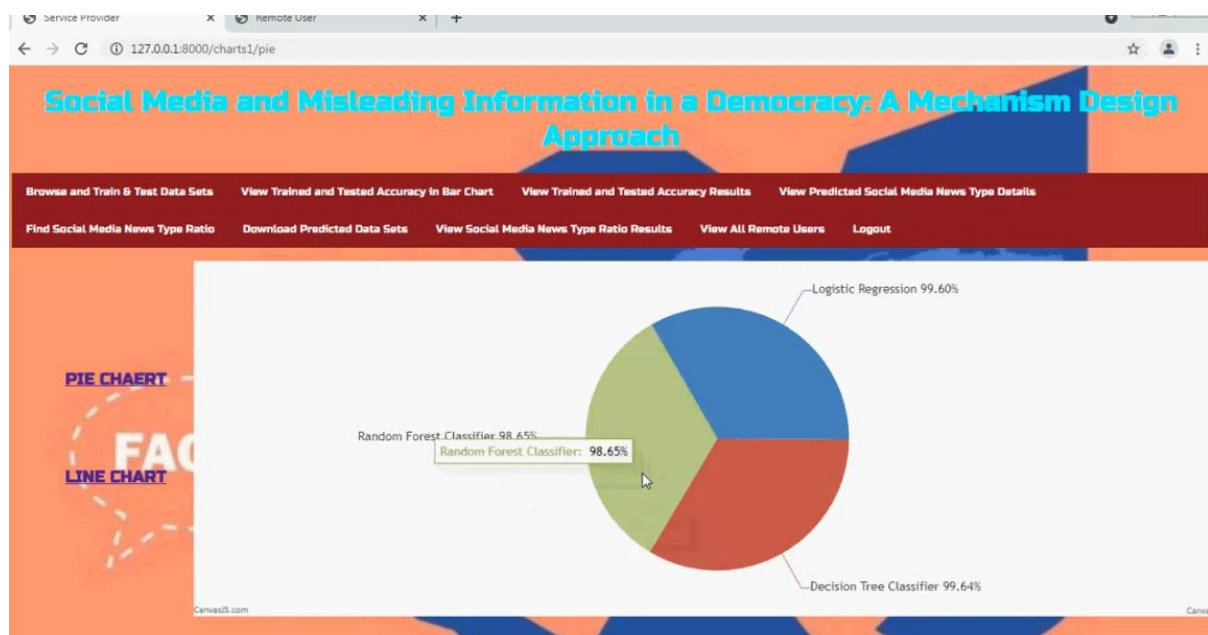
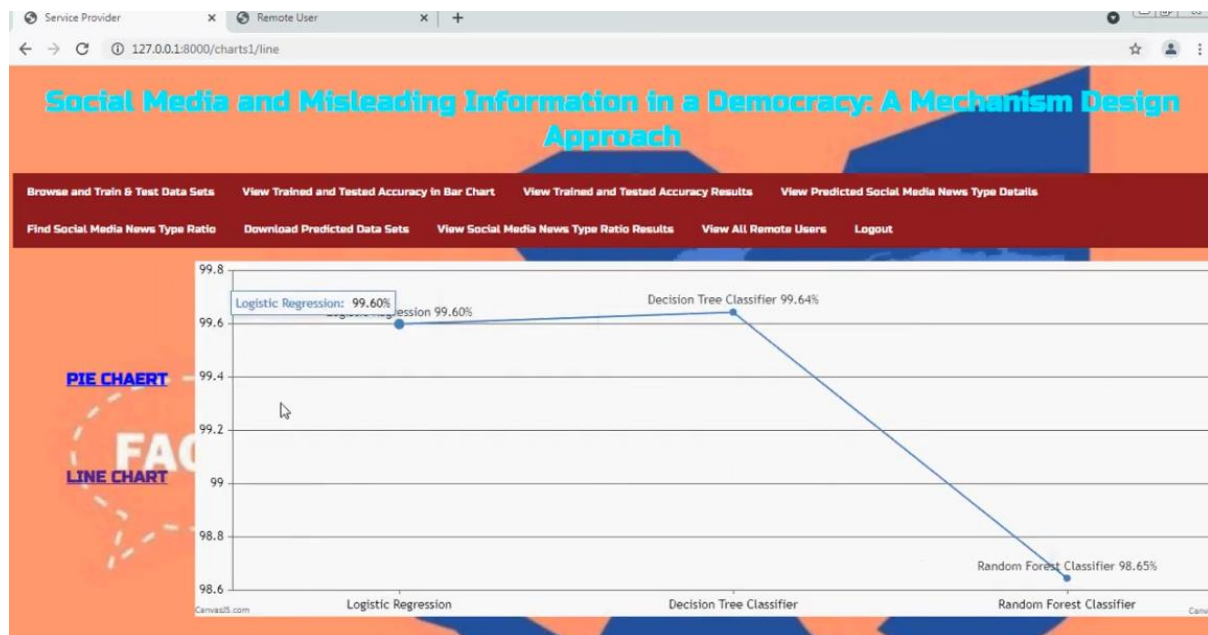
```
CONFUSION MATRIX
[[5948  12]
 [ 28 5232]]
Random Forest Classifier
ACCURACY
98.64527629233511
CLASSIFICATION REPORT
precision    recall  f1-score   support
0           0.99      0.99      0.99     5960
1           0.98      0.99      0.99     5260

accuracy          0.99      0.99      0.99    11220
macro avg         0.99      0.99      0.99    11220
weighted avg      0.99      0.99      0.99    11220

CONFUSION MATRIX
[[5880   80]
 [ 72 5188]]
[18/Nov/2021 17:22:45] "GET /Train_Test_DataSets/ HTTP/1.1" 200 5017
[18/Nov/2021 17:22:45] "GET /static/bg.jpg HTTP/1.1" 304 0
```

The presence of this information on a web page indicates that the project may be open to the public or accessible to researchers within the field. It could serve as a platform for sharing findings, comparing approaches, and potentially collaborating on solutions to the challenge of misinformation on social media.





The image shows a web page interface likely related to a project aimed at detecting and classifying misinformation on social media. The page displays a news article and the machine learning model's prediction on whether the article is "True" or "Fake." In this case, the model has predicted that the news article is "Fake." This suggests that the project utilizes machine learning algorithms to analyze news content and identify potential instances of misinformation. The interface allows users to view the news data, the model's prediction, and potentially other relevant information.

Service Provider Remote User 127.0.0.1:8000/Predict_Social_Media_News_Type/

Browse and Train & Test Data Sets View Trained and Tested Accuracy in Bar Chart View Trained and Tested Accuracy Results View Predicted Social Media News Type Details

Find Social Media News Type Ratio Download Predicted Data Sets View Social Media News Type Ratio Results View All Remote Users Logout

View Social Media News Type(Misleading or True) !!!

News_Data	Prediction
<p>Republicans are working overtime trying to sell their scam of a tax bill to the public as something that directly targets middle-class and working-class families with financial relief. Nothing could be further from the truth, and they're getting hammered on that repeatedly. Speaking on CNBC, Paul Ryan was going full throttle, trying to convince us that the paltry savings we're getting is actually wait for it big money. But he didn't just go with the usual talking points. With a smug look that only someone who grew up in a wealthy family can muster when talking about that which he does not know, Ryan claimed that the \$2,059 more per year that families living paycheck-to-paycheck will see is extremely significant. Then he decided he had to amend that to say such savings might be nothing to a family earning \$600,000 per year (true), or for people living in New York or California (false). Those are the same two states that Trump's loyal subjects insist on stripping from the 2016 vote totals to claim that Trump actually won the popular vote. Watch Ryan completely dismiss all the struggling families living in blue states below. If you're living paycheck-to-paycheck which is more than half of the people in this country and you got \$2,059 more from a tax cut next year, that's not nothing. pic.twitter.com/8TKrMqRa1 Paul Ryan (@SpeakerRyan) December 21, 2017 Someone needs to reach through their computer or television and wipe that smugness off his face. It is the height of arrogance and insult to imply that there are no struggling families in either of those two states. Featured image via Mark Wilson/Getty Images</p>	Fake

Service Provider Remote User 127.0.0.1:8000/Find_Social_Media_News_Type_Ratio/

Social Media and Misleading Information in a Democracy: A Mechanism Design Approach

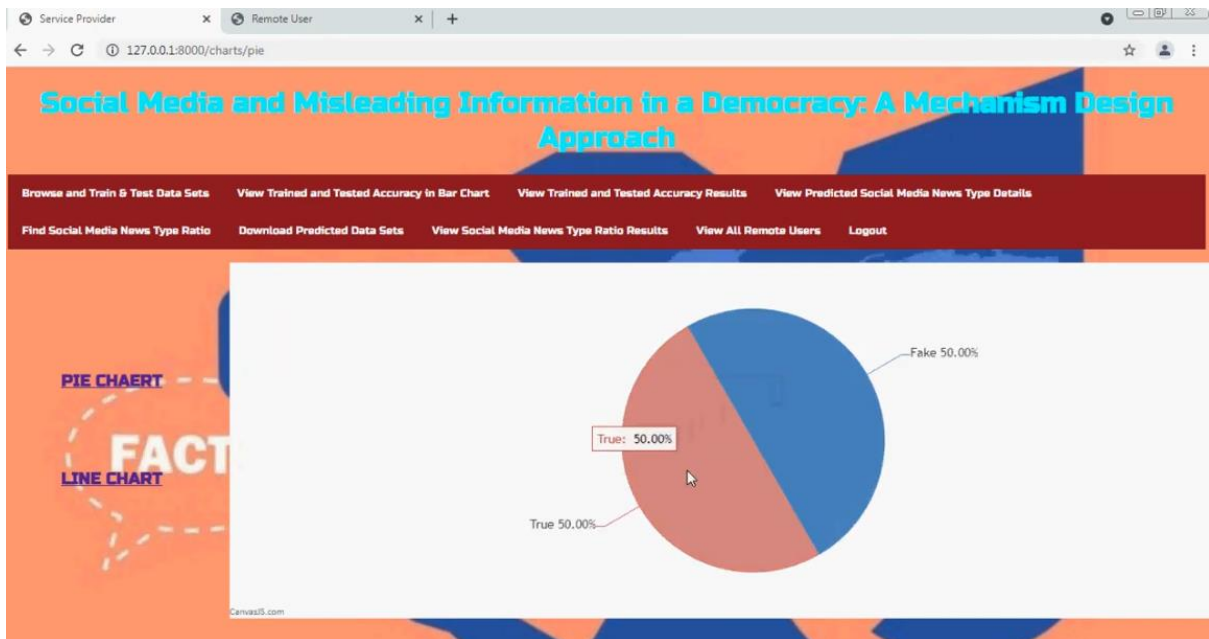
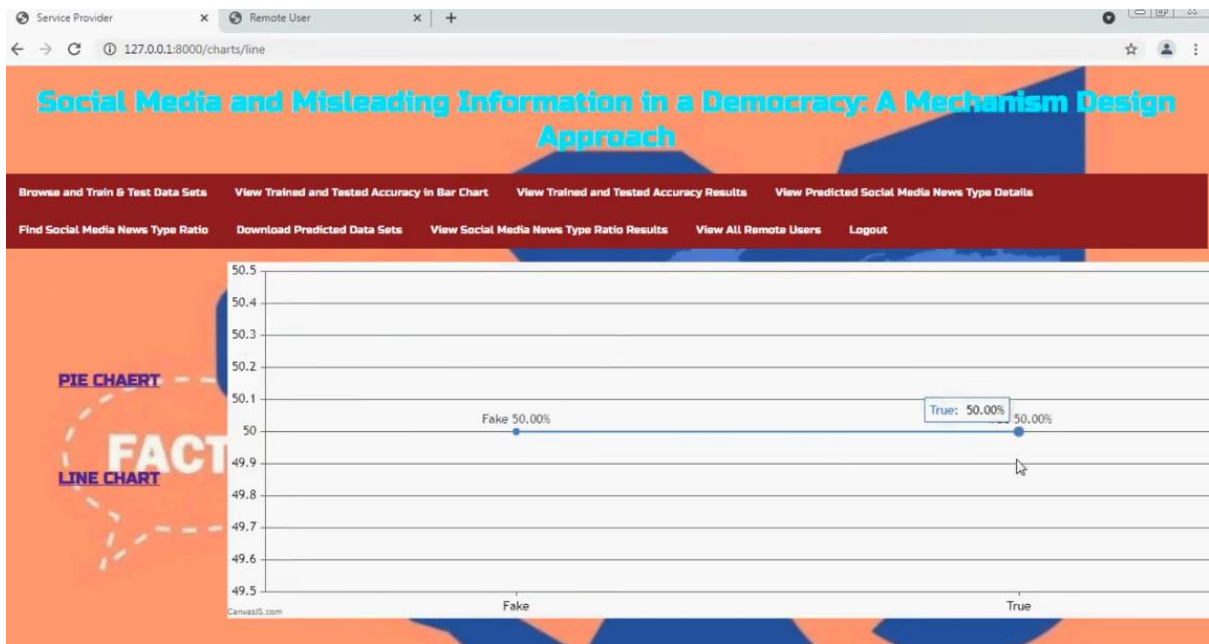
Browse and Train & Test Data Sets View Trained and Tested Accuracy in Bar Chart View Trained and Tested Accuracy Results View Predicted Social Media News Type Details

Find Social Media News Type Ratio Download Predicted Data Sets View Social Media News Type Ratio Results View All Remote Users Logout

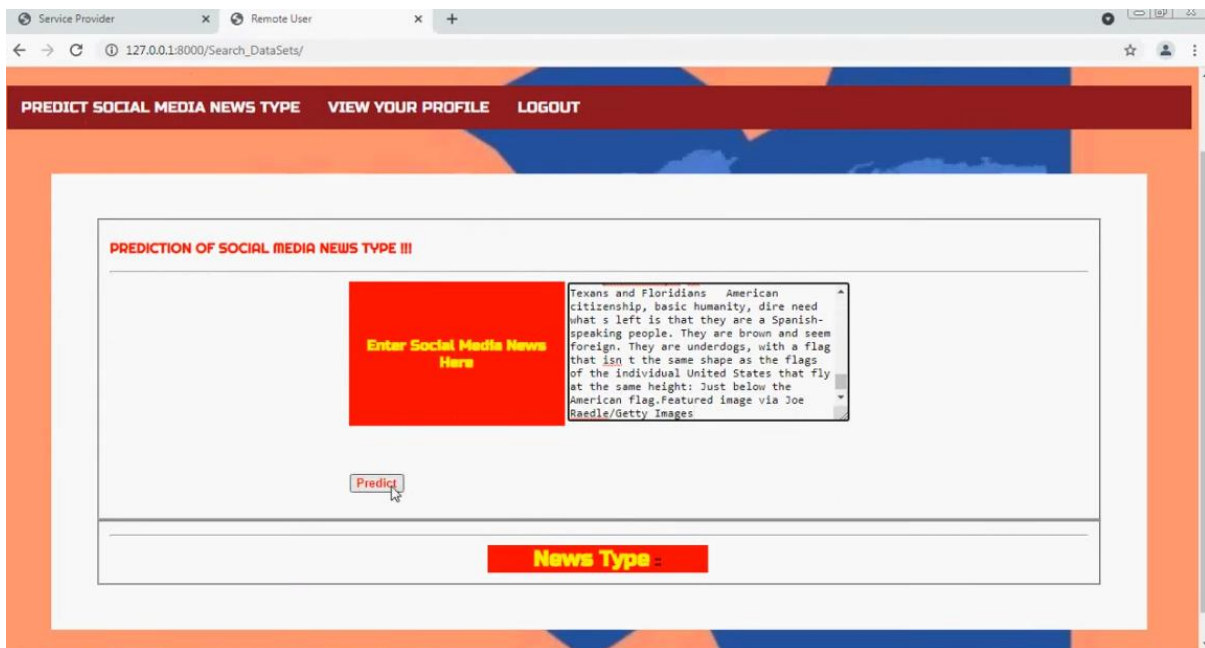
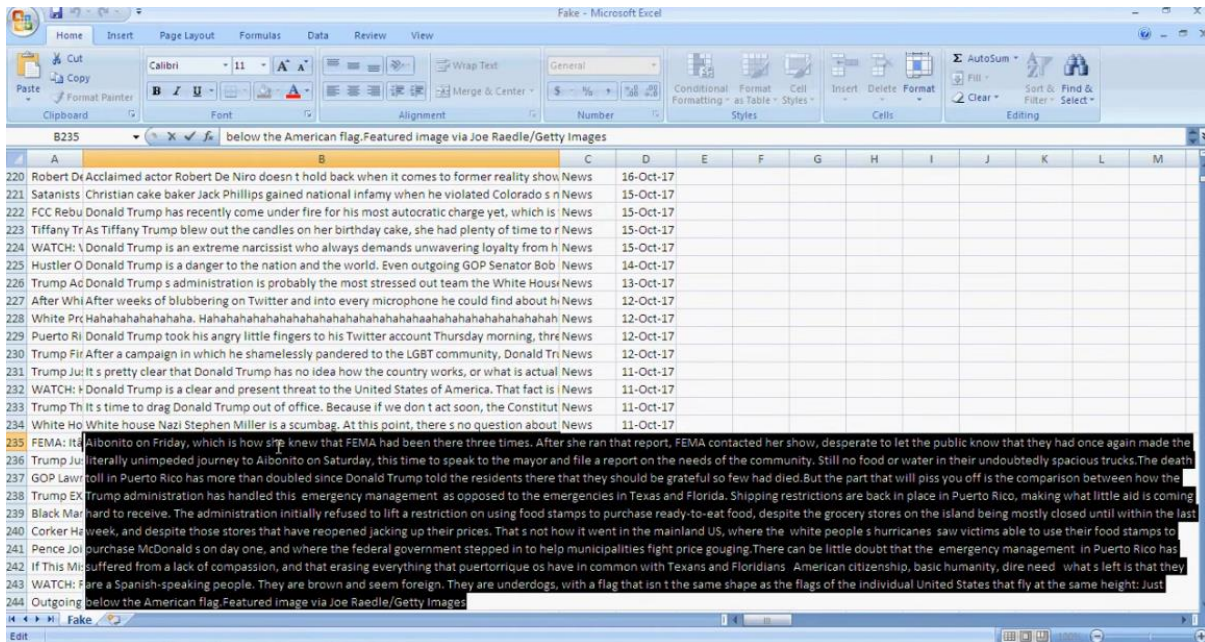
Social Media News Type Found Ratio Details

Social Media News Type	Ratio
Fake	50.0
True	50.0

127.0.0.1:8000/Find_Social_Media_News_Type_Ratio/



It's important to note that the headlines are presented without any context or verification. Some of the headlines may be factually accurate, while others may be exaggerated, biased, or even completely false. Therefore, it's crucial to approach this information with skepticism and to cross-reference it with reliable news sources before drawing any conclusions.



The code includes steps like data preprocessing (using `cv2.fit_transform` and `train_test_split`), model instantiation (`LogisticRegression()`), and likely subsequent training and evaluation steps. The warning messages related to data slicing in Pandas suggest that the code might need adjustments to ensure data integrity.

The screenshot shows an IDE window with a project named "Social_Mediaand_Misleading_Information". The file explorer on the left shows a directory structure with files like forms.py, models.py, tests.py, views.py, migrations, __init__.py, and admin.py. The main editor displays the content of "Service_Provider\views.py", which includes the following code:

```
167 x = cv.fit_transform(x)
168
169 x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25)
170
171 from sklearn.linear_model import LogisticRegression
172
173 print("Logistic Regression")
174 LR = LogisticRegression()
```

Below the editor is a terminal window showing the output of the script. It includes several warnings and error messages:

```
Try using .loc[row_indexer,col_indexer] = value instead
See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df_fake_manual_testing["class"] = 0
D:\Python Work\2021 and 2022 Code\Social_Mediaand_Misleading_Information\social_mediaand_misleading_information\Remote_User\views.py:86: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
df_true_manual_testing["class"] = 1
Testing and Predicting Results----
C:\Users\Admin\AppData\Local\Programs\Python\Python37-32\lib\site-packages\sklearn\linear_model\_logistic.py:818: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG,
Fake
[18/Nov/2021 17:25:36] "POST /Search_DataSets/ HTTP/1.1" 200 4146
```

The terminal status bar at the bottom right shows "351 CRLF: UTF-8".

CONCLUSION

Our primary goal in this paper was to design a mechanism to induce a GNE solution in the misinformation filtering game, where (i) each platform agrees to participate voluntarily, and (ii) the collective utility of the government and the platforms is maximized. We designed a mechanism and proved that it satisfies these properties along with budget balance. We also presented an extension of the mechanism with weaker technical assumptions.

Ongoing work focuses on improving the valuation and average trust functions of the social media platforms based on data. We also consider incorporating uncertainty in a platform's estimates of the impact of their filter. These refinements of the modeling framework will allow us to make our mechanism more practical for use in the real world.

Future research should include extending the results of this paper to a dynamic setting in which the social media platforms react in real-time to the proposed taxes/subsidies. In particular, someone could develop an algorithm that the players can use to iteratively arrive at the Nash equilibrium. In such an algorithm, the social planner can receive additional information from the players while they iteratively learn the GNE. Then, she can use this information to change her allocations dynamically, allowing us to relax either Assumption 5 on monitoring of average trust, or Assumption 6 on the excludability of the platforms.

REFERENCES

- [1] W. Davies, “The age of post-truth politics,” *The New York Times*, vol. 24, p. 2016, 2016.
- [2] J. Cone, K. Flaharty, and M. J. Ferguson, “Believability of evidence matters for correcting social impressions,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 20, pp. 9802–9807, 2019.
- [3] Z. Tufekci, “Youtube, the great radicalizer,” *The New York Times*, vol. 10, 2018.
- [4] A. D. Kramer, J. E. Guillory, and J. T. Hancock, “Experimental evidence of massive-scale emotional contagion through social networks,” *Proceedings of the National Academy of Sciences*, vol. 111, no. 24, pp. 8788–8790, 2014.
- [5] J. Weedon, W. Nuland, and A. Stamos, “Information operations and facebook,” Retrieved from: <https://fbnewsroomus.files.wordpress.com/2017/04/facebook-and-information-operations-v1.pdf>, 2017.
- [6] H. Farrell and B. Schneier, “Common-knowledge attacks on democracy,” *Berkman Klein Center Research Publication*, no. 2018-7, 2018.
- [7] H. Allcott and M. Gentzkow, “Social media and fake news in the 2016 election,” *Journal of economic perspectives*, vol. 31, no. 2, pp. 211–36, 2017.
- [8] O. Analytica, “Russia will deny cyberattacks despite more us evidence,” *Emerald Expert Briefings*, no. oxan-db, 2018.
- [9] A. Bessi, M. Coletto, G. A. Davidescu, A. Scala, G. Caldarelli, and W. Quattrociocchi, “Science vs conspiracy: Collective narratives in the age of misinformation,” *PloS one*, vol. 10, no. 2, p. e0118093, 2015.
- [10] E. Brown, “Propaganda, misinformation, and the epistemic value of democracy,” *Critical Review*, vol. 30(3–4), pp. 194–218, 2018.
- [11] J. A. Tucker, Y. Theocharis, M. E. Roberts, and P. Barber’a, “From liberation to turmoil: Social media and democracy,” *Journal of Democracy*, vol. 28(4), pp. 46–59, 2017.
- [12] A. Sternisko, A. Cichocka, and J. J. Van Bavel, “The dark side of social movements: Social identity, non-conformity, and the lure of conspiracy theories,” *Current opinion in psychology*, vol. 35, pp. 1–6, 2020.

- [13] R. Jaakonmäki, O. Müller, and J. Vom Brocke, "The impact of content, context, and creator on user engagement in social media marketing," *Proceedings of the 50th Hawaii international conference on system sciences*, 2017.
- [14] O. Candogan and K. Drakopoulos, "Optimal signaling of content accuracy: Engagement vs. misinformation," *Operations Research*, vol. 68, no. 2, pp. 497–515, 2020.
- [15] E. A. Vogels, A. Perrin, and M. Anderson. (2020) Most americans think social media sites censor political viewpoints. [Online]. Available: <https://www.pewresearch.org/internet/2020/08/19/most-americans-think-social-media-sites->
- [16] S. Vosoughi, D. Roy, and S. Aral, "The spread of true and false news online," *Science*, vol. 359, no. 6380, pp. 1146–1151, 2018.
- [17] A. Mas-Colell, M. D. Whinston, and J. R. Green, *Microeconomic theory*. Oxford University Press, 1995.
- [18] A. Dave and A. Malikopoulos, "The prescription approach to decentralized stochastic control with word-of-mouth communication," *arXiv e-prints*, p. arXiv:1907.12125, Sep 2019.
- [19] A. Mahajan, N. C. Martins, M. C. Rotkowitz, and S. Yüksel, "Information structures in optimal decentralized control," in *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. IEEE, 2012, pp. 1291–1306.
- [20] A. Nayyar, A. Mahajan, and D. Teneketzis, "Decentralized stochastic control with partial history sharing: A common information approach," *IEEE Transactions on Automatic Control*, vol. 58, no. 7, pp. 1644–1658, 2013.
- [21] A. A. Malikopoulos, C. G. Cassandras, and Y. J. Zhang, "A decentralized energy-optimal control framework for connected automated vehicles at signal-free intersections," *Automatica*, vol. 93, no. April, pp. 244–256, 2018.
- [22] S. Sharma and D. Teneketzis, "Local public good provisioning in networks: A Nash implementation mechanism," *IEEE Journal on Selected Areas in Communications*, vol. 30, no. 11, pp. 2105–2116, 2012.
- [23] A. Sinha and A. Anastasopoulos, "Generalized proportional allocation mechanism design for multi-rate multicast service on the internet,"

51st Annual Allerton Conference on Communication, Control, and Computing (Allerton), pp. 146–153, 2013.

[24] A. Kakhbod and D. Teneketzis, “An efficient game form for unicast service provisioning,” *IEEE Transactions on Automatic Control*, vol. 57, no. 2, pp. 392–404, 2011.

[25] R. Jain and J. Walrand, “An efficient nash-implementation mechanism for network resource allocation,” *Automatica*, vol. 46(8), pp. 1276–1283, 2010.