DETECTING FAKE ACCOUNTS ON SOCIAL MEDIA INSTAGRAM

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering

By

BHAVYA K (Reg.No - 39110454)
NIKHITHA K (Reg.No – 39110443)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY (DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE

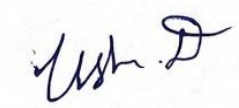
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600119

APRIL - 2023

SATHYABAMA
INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC | 12B Status by UGC | Approved by AICTE
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **Bhavya k (Reg.No - 39110454) and Nikhitha k (Reg.No - 39110443)** who carried out the Project Phase-2 entitled **"DETECTING FAKE ACCOUNTS ON SOCIAL MEDIA - INSTAGRAM"** under my supervision from January 2023 to April 2023.



Internal Guide

Dr. D. USHA NANDINI M.E., Ph.D.

Head of the Department

Dr. L. LAKSHMANAN, M.E., Ph.D.

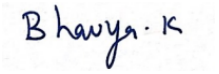
Submitted for Viva voce Examination held on 20.04.2023

Internal Examiner

External Examiner

DECLARATION

I, **Bhavya k(Reg.No- 39110454),** hereby declare that the Project Phase-2 Report entitled "**DETECTING FAKE ACCOUNTS ON SOCIAL MEDIA - INSTAGRAM**" done by me under the guidance of **Dr. D. Usha Nandini M.E., Ph.D.,** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in **Computer Science and Engineering**.



DATE: 20.04.2023
PLACE: Chennai

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr.T.Sasikala M.E., Ph.D., Dean**, School of Computing, **Dr.L.Lakshmanan M.E., Ph.D.,** Head of the Department of Computer Science and Engineering for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr.D.Usha Nandini M.E., Ph.D.,** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my phase-2 project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

.

ABSTRACT

As the growth of online social networks increases everyone is associated and linked with social media. A massive amount of personal data is being attacked and stolen by cyber attackers. These fraudulent profiles also spread negative publicity, false news, and various malicious programs one of the best ways is to detect these fake accounts, not only the best way but also it is the first step to stopping the negative news and false rumours. There are many efficient ways and techniques to detect these fake accounts. However, these techniques are based on or depend on the account features. In this paper, main aim is to examine three machine learning algorithms that are random forest, logistic regression, and decision tree. Now, progressing to compare the level of efficiency in the three mentioned methodologies and got the highest accuracy for random forest classifier.

The Proliferation of fake accounts on social media platforms like instagram has become significant challenge for both users and platform administrators. Fake accounts are created for a variety of purposes, including spreading spam, disseminating false information, and engaging in fraudulent activity. These accounts can also negatively impact the user experience by generating fake engagement, flooding comment sections with spam, and diluting the quality of content on the platform. Therefore, it become crucial to develop effective methods for detecting and removing fake accounts from instagram.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

In recent times, some of the most well-known social network sites like Instagram, Facebook and Twitter became integral parts of daily life. Social network sites are used by people for e-commerce, entertainment, information and idea sharing keeping in contact with long-lost friends and finding new acquaintances. Users of OSN can post pictures and videos, leave comments and also provide likes to pictures that have been shared. Fraudulent accounts are one of the downsides of Online Social Networks. We can check an account is fake or not by following parameters, the number of followers, and posts, or the characteristics of the posted information such as the number of likes, views, comments, and shares. By knowing this, many fake accounts are formed and started purchasing comments and increasing the number of followers to become popular. These are particularly created to commit fraudulent acts such as disseminating misleading information and spreading malware. While some are created to gain followers to become popular, spam comments and likes. In social media, one individual creates many profiles with distinct identities, email addresses, and phone numbers. The majority of them are likely fake accounts. There are two sorts of fake accounts: duplicate accounts and fraudulent accounts. Users create multiple accounts as secondary accounts to promote their e-businesses. People and influencers establish duplicate accounts to enhance their businesses and spread useful information. This type of account is not harmful and does not violate social media terms and conditions. Users create fake accounts mainly to spread negativity, false news, hate speech, and impersonate others; these accounts might be considered dangerous and called as false accounts.

This work primarily concentrated on detecting if an Instagram account is phony or real. This detection is based on some parameters such as username, private or public account, and number of followers, posts, follows and profile pictures. We can detect a user's activity using this attributes. The user’s behaviour on Instagram is determined by a collection of 12 features. Three machine learning

techniques are used to distinguish between genuine and fraudulent accounts.

For the period between 2014 and 2018 around 2.53 million U.S. dollars have been spent on sponsoring political ads on Facebook by non-profits. The open nature of OSNs and the massive amount of personal data for its subscribers have made them vulnerable to Sybil attacks. In 2012, Facebook noticed an abuse on their platform including publishing false news, hate speech, sensational and polarizing, and some others. However, online Social Networks (OSNs) have also attracted the interest of researchers for mining and analyzing their massive amount of data, exploring and studying users' behaviors as well as detecting their abnormal activities. In researchers have made a study to predict, analyses and explain customer's loyalty towards a social media-based online brand community, by identifying the most effective cognitive features that predict their customer's attitude. Facebook community continues to grow with more than 2.2 billion monthly active users and 1.4 billion Daily active users, with an increase of 11% on a year- over-year basis. In the second quarter of 2018 alone, Facebook reported that its total revenue was $13.2 billion with $13.0 billion from ads only. Similarly, in second quarter of 2018 Twitter has reported reaching about one billion of Twitter subscribers, with 335 million monthly active users. In 2017 twitter reported a steady revenue growth of 2.44 billion U.S. dollars, with 108 million U.S. dollars lower profit compared to the previous year. In 2015 Facebook estimated that nearly 14 million of its monthly active users are in fact undesirable, representing malicious fake accounts that have been created in violation of the website's terms of service. Facebook, for the first time, shared a report in the first quarter of 2018 that shows their internal guidelines used to enforce community standards covering their efforts between October 2017 to March 2018, this report illustrates the amount of undesirable content that has been removed by Facebook, and it covers six categories: graphic violence, adult nudity and sexual activity, terrorist propaganda, hate speech, spam, and fake accounts.

In recent years many celebrities and businesses have created their accounts on

Instagram, they use Instagram to grow their business and fans. Furthermore, many of them and other famous users use it as a platform for advertising. When someone is boosting the number of followers over a hundred thousand or millions,

it is no surprise to use that person's account as a lucrative earner. Instagram has widely used for sharing photos and videos and is profitable for celebrities, businesses, and people with a considerable number of followers. In the meantime, this high profit made this platform prone to be the potential place to be used for malicious activities.

Such versatility and spread for the proliferation of abnormal accounts, which behave in unusual ways. Most academic researchers have mostly focused on spammers and accounts, which put their efforts into spreading advertising, spam, malware and other suspicious activities. These malicious accounts are usually using automatic programs to improve their performance, hide their real identity, and look like real users. In past years, media have reported that accounts of celebrities, politicians, and some popular business has indicated suspicious inflation of followers. Fake Instagram accounts specially used to increase the number of followers of a target account

CHAPTER 2

LITERATURE SURVEY

2.1 INFERENCES FROM LITERATURE SURVEY

Detection of fake accounts is one of the major issues that should be solved as soon as possible. Though various methods have been already making an impact, these all methods are not completely accurate. Inspired by various research papers on detection, in this whole literature review section, we are going to discuss various applied methodologies and approaches.

Ersahin et.al [1], provided a categorization technique to find Twitter's Phony accounts. And pre-processed our dataset by applying the Entropy Minimization Discretization (EMD) method under supervision to numerical characteristics, and then they examined the output of the Naive Bayes algorithm. By merely pre-processing their dataset using the n discretization strategy on chosen characteristics, they were able to improve the accuracy with Nave Bayes from85.55% to.90.41% for a method for identifying bogus accounts on Twitter. Only by applying the numerical information from provide a safe platform that can forecast and spot fake news in social media networks.

Aditi Gupta et.al [2], used the Facebook Graph API, and collected Facebook user feeds. The intricate privacy controls on Facebook made it extremely difficult to gather data. On a minimal dataset that included their node, their acquaintances in their social neighbourhood, as well as a collection of manually recognized spam accounts, they performed the most widely used supervised machine learning classification approaches. They assessed these classifiers' performance to identify the top classifiers that produced high detection rates and their capacity to identify Phony Facebook accounts.

Yeh-Cheng Chen et.al [3], introduced a novel, efficient technique for detecting

bogus profiles using machine learning. Their method based on account-by-account behavioural analysis evaluate whether an account is genuinely fake or not accurately, as opposed to utilizing graph-based or manually predicted that just

cover basic facts. On real-world data, their detection models work admirably, and their findings indicate that the models are not over fitting.

Estee van der walt et.al, [4] employed artificial intelligence to identify Phony identities. The author of this research clarified how to spot false profiles made by both bots and humans. In this study, the fraudulent accounts made on social networks by people and bots were compared. Finally, it can be said that while the features employed to detect the bots were successful, they weren't entirely effective.

Lu Zhang et.al [5], suggested the PSGD, a partially supervised model, to identify scammer groups. As a scammer group detector, a classifier is studied using PU-Learning via PSGD. The suggested PSGD is effective and surpasses cutting-edge spammer detection techniques, according to tests on a real-world data set from Amazon.cn.Bucket online media, they believe it to be a significant and extremely promising outcome.

Sarah Khaled et.al [6], investigated effective methods for spotting bots and Phony accounts on the Twitter network. An innovative technique called SVM-NN is recommended in this to offer an effective detection for such profiles. Even though the recommended approach uses fewer features, it can still be categorized with around ninety- eight percent accuracy. In comparison to the other two classifiers, the newly suggested method performs better in terms of accuracy across all feature sets. The accuracy of the correlation feature set's records is astounding.

Zulfikar Alom et.al [7], investigated the nature of spam users on the Twitter platform to develop an existing spam-detecting mechanism. In this paper, it is designed a new mechanism and a more robust set of features to detect spammers on Twitter. In this Random Forest classifier is used which gives a better result compared to other methodologies. From this, they can plan to build a more effective model which classifies various types of spammers within different social networks.

Naman Singh et.al [8], proposed that on online networking platforms, more bogus accounts are being created for nefarious purposes. They create a forged human

profile in order to successfully detect, identify, and remove fake profiles. For bots, ML models employed a variety of factors to determine how many people an account has on each platform's friend list as followers. It is not possible to contrast between Phony profiles made by humans and cyborgs. By using a data set with Phony profiles and classifying them as fake or real, they can contrast fake and real accounts.

Shivangi Singhal et.al [9], formally introduced unveil Spot Fake a multi-modal platform for identifying bogus news. Without considering any other subtasks, their suggested approach detects bogus news. It makes use of article textual and graphic components. Text characteristics were learned using language models (such as BERT), and image features were learned using VGG-19 pre-trained on Image Net dataset. Twitter and Weibo, two publicly accessible databases, are used in all of the studies.

Sowmya P et.al [10], one serious issue involves generating duplicate profile users using the data of existing users. A collection of principles that may differentiate real or fake are used to detect Phony profiles. A detection method that can find fraudulent or clone profiles on online networking platforms like Twitter has been presented.

Zeinab Shahbazi et.al [11], proposed an integrated system for multiple block chains and NLP functions to leverage ML models to find false news and better anticipate bogus user accounts and posts. For this workflow, Reinforcement learning methodology is applied. The operation of the decentralized block chain architecture, which offers the framework of digital content authorization proof, improved the security of this platform. The idea behind this approach is to provide a safe platform that can forecast and spot fake news on social media networks.

Latha P et.al [12], stated different categorization techniques have been used in the past to identify Phony online networking media accounts. However, they must improve their ability to spot Phony accounts on these websites. To improve the

accuracy rate of identifying bogus accounts, they use ML technologies and NLP in our study. The Random Forest tree classifying algorithm was chosen.

Michael Jonathan Ekosputra et.al [13], study's findings make it evident that ml may be used to detect fake profiles; the models that have been examined in this article include Logistic Regression, Bernoulli Naive Bayes, Random Forest, svm, and ANN. Any changes or additions to features will have an impact on each model's accuracy. According to the research, adding parameters to a model will almost certainly make it more accurate than a model with no parameters or a default form. Based on the results of the research's initial trial, the Random Forest algorithm achieves the best outcome with a startling accuracy of 0.92.

Karishma Anklesaria et.al [14], used different classifiers to classify the profile as fake or real using database of user accounts. This is accomplished by using a methodical approach that includes cleaning, pre-processing, feature selection, and model training. After that, all the employed algorithms—Random Forest, AdaBoost, MLP, SGD, and Artificial neural network—are compared on the basis of different evaluation parameters. According to the study, Random Forest classifier performs better.

Md Mahadi Hassan Sohan et.al [15], showed the importance of comments and how they affect almost every aspect of social media data. People's perspectives are significantly influenced by reviews. Hence, identifying fake reviews is an exciting study area. A method for spotting fake reviews using machine learning was disclosed in the study. It detects both review characteristics and reviewer behaviour. The food review dataset was used to evaluate the proposed method. The technique is semi-supervised, with several classifiers being used. In the fake review detection procedure, the findings reveal that the Random Forest classifier beats another classifier. Moreover, the findings imply that including in viewers' behavioural traits raises the F-score by 55.5% and the overall accuracy to 97.7%.

2.2 OPEN PROBLEMS IN EXISTING SYSTEM

The factors used by the existing systems to detect the fake accounts are very

less. The prediction becomes accurate when the number of parameters used are more efficient. In previously used algorithms, if some of the inputs are not appropriate, the algorithm could not produce accurate results.

- Low Accuracy.
- High complexity.
- Highly inefficient.
- Requires skilled persons.

CHAPTER 3
REQUIREMENT ANALYSIS

3.1 FEASIBILITY STUDIES/RISK ANALYSIS OF THE PROJECT

The feasibility of the project is analysed 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.

3.1.1 Feasibility Studies

Four key considerations involved in the feasibility analysis are

- TECHNICAL FEASIBILITY
- FINANCIAL FEASIBILITY
- LEGAL FEASIBILITY
- OPERATIONAL FEASIBILITY

Technical Feasibility: The technical feasibility of the project involves assessing the availability of suitable data, machine learning algorithms, and hardware resources. It is important to ensure that the data used in the project is reliable and representative of the population of interest. The machine learning algorithms used should be appropriate for the task of fake account detection and should be able to handle large amounts of data. The hardware resources should be sufficient to handle the computational requirements of the machine learning algorithms.

Financial Feasibility: The financial feasibility of the project involves evaluating the cost of acquiring data, hardware resources, and expertise required for the project. The cost of acquiring data can be significant, particularly if it is collected using methods such as surveys or experiments. The cost of hardware resources such as servers and computing power should also be considered. The cost of expertise required for the project, such as data scientists and machine learning

experts, should also be evaluated.

Legal feasibility: The legal feasibility of the project involves investigating the legal implications of the project. The project should be compliant with relevant laws and regulations, particularly those related to privacy and data protection. The project should also adhere to ethical guidelines, particularly those related to the collection and processing of personal data.

Operational Feasibility: The operational feasibility of the project involves determining the availability of necessary resources, such as human expertise and technological infrastructure. The project should have access to experts in data collection, data analysis, and machine learning. The project should also have access to suitable technological infrastructure, such as servers and computing power, to handle the computational requirements of the machine learning algorithms.

3.1.2 Risk Analysis

Data quality risks: The quality of the data used to train the machine learning model can affect the accuracy of the model. The data used to train the model should be representative of the population of interest and should be free from bias. The data should also be of high quality, with minimal missing values and errors.

Model accuracy risks: The accuracy of the model depends on the selection of the right machine learning algorithms, feature engineering, and parameter tuning. The machine learning algorithms used should be appropriate for the task of fake account detection, and should be able to handle large amounts of data. Feature engineering should be carefully performed to identify the most relevant features for the task. Parameter tuning should be performed to optimize the performance of the model.

Privacy and ethical risks: The collection and processing of personal data raises ethical and privacy concerns. The project should adhere to ethical guidelines,

particularly those related to the collection and processing of personal data. The project should also ensure that appropriate privacy policies are in place to protect the privacy of individuals whose data is collected.

3.2 SOFTWARE REQUIREMENTS SPECIFICATION DOCUMENT

3.2.1 Hardware Requirements

Processor	-	I3/Intel Processor
RAM	-	4GB(min)
Hard Disk	-	160GB
Keyboard	-	Standard Windows
Keyboard Mouse	-	Two or Three Button Mouse
Monitor	-	SVGA

3.2.2 Software Requirements

Operating System	: Windows 10
Server-side Script	: Python
IDE	: PyCharm

3.3 SYSTEM USECASE

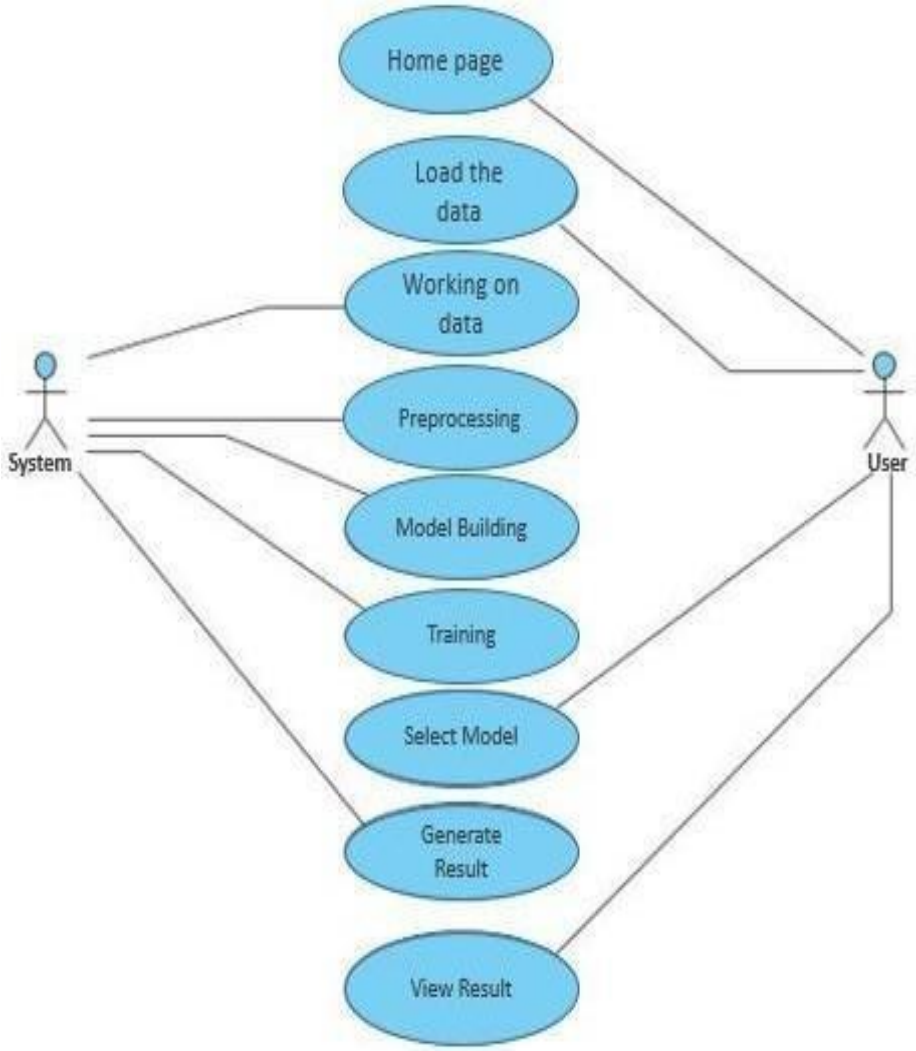


Fig. 3.1 Usecase Diagram

CHAPTER 4

DESCRIPTION OF PROPOSED SYSTEM

4.1 SELECTED METHODOLOGY OR PROCESS MODEL

An effective method to detect fake accounts on online social networks is proposed. This work may be regarded as a valuable system since it aids in lowering the restrictions brought on by conventional and other existing methodologies. The goal of this project is to provide an efficient and dependable system for precisely detecting fake. A potent algorithm in a Python-based framework to develop this system is employed. The process is depicted in the picture below.

The steps for executing the proposed work:

1. The required packages should be installed.
2. Establishing the issue association.
3. Create a framework-based UI.
4. Upload CSV file
5. View data
6. Select model
7. Output to predict.

This proposed work is completely based on Machine Learning approach which detects whether the profile is fake or real one. To do this, three well-known methodologies were compared. This proposed work has two phases, the first phase is frontend – which creates a platform another is backend – a model is created and the main part of the project exists.

The main part of the proposed work is the backend. Here, using flask framework an application is created that detects whether an account is fake or not. In this aspect, the data is collected and preprocessed. Then the data splitting takes

place. Now the data gets trained and need to build the model. After the model building with the help of the model data gets tested. And finally the prediction is done.

Firstly, download the dataset from Kaggle which is raw data. The size of the

dataset is 697 with 12 attributes such as profile picture, length of the username, full name words, length of the full name, the description provided by the user known as bio, any external URLs provided, private or public accounts, number of posts posted, number of followers and number of accounts the user follows.

In the application user first uploads a CSV file of the dataset then the system takes that file and undergoes preprocessing, which means Importing libraries such as pandas, and NumPy. Import dataset, cleaning of null values, categorical values will be turned into numerical values and removing unwanted columns. The null values are replaced by using imputation technique. Then splits the dataset into training and testing set. The target variable or Dependent variable undergoes training and independent variables will go to testing. Next is model building. Model buildings mean choosing a suitable algorithm and train it as per the requirements. Different models like decision tree, Random Forest, and Logistic Regression are considered, and train the models by importing those modules from Sklearn. Then, train the models using the training dataset with fit( ) function. After the model building, apply that model to make predictions and predict the response for test dataset. Then evaluate the model's performance by importing different metrics from sklearn metrics. Random forest got the highest accuracy with 90%. So, this is how the system will do preprocessing and User can view the dataset.

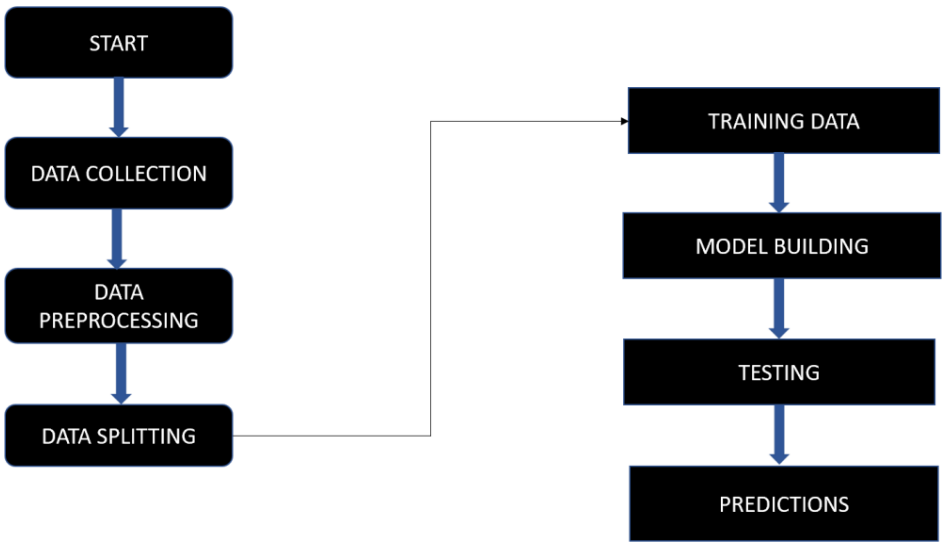


Fig. 4.1 Flow chart

4.2 ARCHITECTURE / OVERALL DESIGN OF PROPOSED SYSTEM

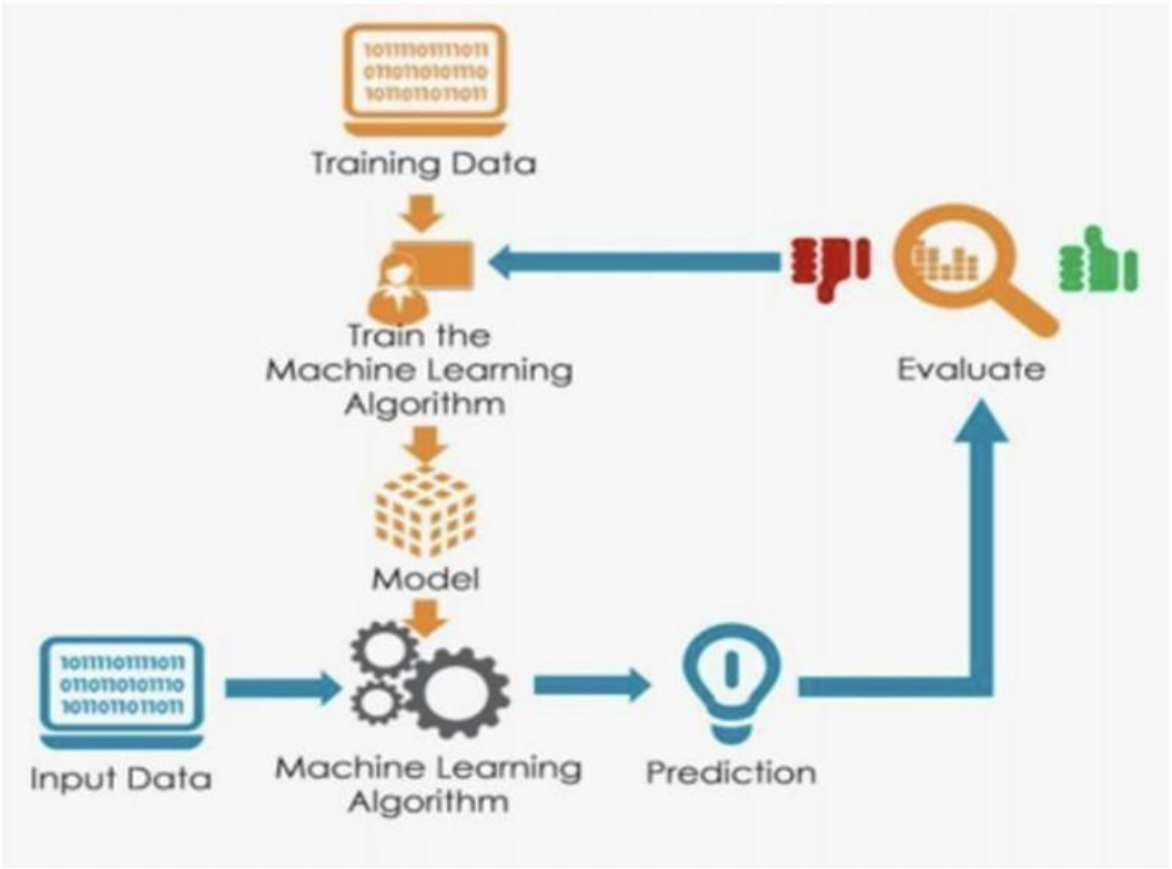


Fig. 4.2 System Architecture

4.3 DESCRIPTION OF SOFTWARE FOR IMPLEMENTATION AND TESTING PLAN OF THE PROPOSED MODEL/SYSTEM

PyCharm is an extremely popular Python IDE. An Integrated Development Environment or IDE features a code editor and a compiler for writing and compiling programs in one or many programming languages. The main reason PyCharm for the creation of this IDE was for Python programming, and to operate across multiple platforms like Windows, Linux, and macOS. The IDE comprises code analysis tools, debugger, testing tools, and also version control options. It also assists developers in building Python plugins with the help of various APIs available. The IDE allows us to work with several databases directly without

getting it integrated with other tools. Although it is specially designed for Python,

HTML, CSS, and JavaScript files can also be created with this IDE. It also comes with a beautiful user interface that can be customized according to the needs using plugins.

4.3.1. System

1. Store Dataset:
    - The System stores the dataset given by the user, where the dataset is downloaded from the kaggle by user.
2. Model Training:
    - The system takes the data from the user and fed that data to the selected model.

4.3.2 User

1. Load Dataset
    - The user can load the dataset he/she want to work on.
2. View Dataset
    - The size of the dataset is 697 with 12 attributes such as profile picture, length of the username, full name words, length of the full name, the description provided by the user known as bio, any external URLs provided, private or public accounts, number of posts posted, number of followers and number of accounts the user follows. The user can view the dataset
3. Select model:
    - User can apply the model among random forest, decision tree, logistic regression to the dataset for accuracy. where random forest got highest accuracy of 91.8%.
4. Prediction
    - Passing parameters to predict the output

4.4 PROJECT MANAGEMENT PLAN

Instagram is the easiest platform to spread rumors, to stole one's information etc., these all kind of malicious are carried using fake accounts. So should be aware of the account he/she interacting with. With the help of this project anyone can

detect fake accounts on instagram. In order to make the detection easily available we

also implemented an application and also carefully considered all the existing systems

In the project, the whole process is done in the pycharm platform. Compared three well known models and flask framework to create an application. Collected the dataset from kaggle which contains different users instagram information.

TABLE OF PROJECT MANAGEMENT PLAN

MODULES	**TIME TAKEN**
Introduction	September
Literature survey	October
Software design	November - December
Software implementation	January - Febrauary
Testing	March

Table no. 4.1 project plan

CHAPTER 5

IMPLEMENTATION DETAILS

5.1 DEVELOPMENT AND DEPLOYMENT SETUP

5.1.1 Pycharm

PyCharm is a popular integrated development environment (IDE) for Python programming language. Developed by JetBrains, PyCharm is designed to provide developers with a wide range of tools and features to help them write, debug, and maintain Python code more efficiently and effectively. In this article, we will take a more in-depth look at the key features and functionalities of PyCharm.

User Interface

PyCharm has a clean and intuitive user interface that is designed to be user-friendly and customizable. The IDE supports a range of themes and color schemes, which can be customized to suit individual preferences. PyCharm also supports multiple windows and tabbed interfaces, which allows developers to work with multiple files and projects simultaneously.

Code Completion and Suggestion

One of the most powerful features of PyCharm is its intelligent code completion and suggestion engine. The IDE can suggest method names, class names, and variable names based on the context of the code. PyCharm can also provide suggestions for import statements and function arguments. This feature can help developers write code faster and with fewer errors.

Debugging Tools

PyCharm comes with a powerful debugger that allows developers to step through their code, set breakpoints, inspect variables in real-time. The debugger also supports remote debugging, which can be useful for debugging code running on a remote server. PyCharm also provides support for Django, Flask, and Pyramid frameworks, which allows developers to debug web applications directly from the IDE.

Code Analysis and Error Highlighting

PyCharm has a built-in code analysis engine that can identify potential errors in the code and highlight them for the developer. The IDE can flag issues like

undefined variables, unused imports, and syntax errors, which can help catch bugs early and improve code quality. PyCharm also provides support for code inspections, which can help identify issues like code smells, design problems, and potential performance issues.

Version Control Integration

PyCharm has built-in integration with popular version control systems like Git, SVN, and Mercurial. This allows developers to manage their code changes, track changes over time, and collaborate with others on their projects. PyCharm provides support for features like code merging, conflict resolution, and commit history visualization.

Refactoring Tools

PyCharm provides a range of tools to help developers refactor their code. These tools include renaming variables, extracting code into functions, and moving code between files. PyCharm also provides support for safe delete, which allows developers to delete code and its references without introducing errors in the codebase.

Code Templates and Generation

PyCharm has a library of code templates that can be used to quickly generate common code patterns, such as loops, if statements, and function definitions. The IDE can also generate boilerplate code for popular frameworks like Flask and Django. PyCharm also provides support for code generation from UML diagrams and database schemas.

Testing Tools

PyCharm has built-in support for unit testing, including test runners, code coverage analysis, and debugging tools for tests. The IDE can also generate test stubs and test code automatically, based on the code being tested. PyCharm also provides support for running tests remotely, which can be useful for testing code running on a remote server.

Integration with Other Tools

PyCharm can be integrated with other tools in the development workflow, such as code linters, task managers, and build tools. PyCharm provides support for popular Python linters like Flake8, Pylint, and Pyflakes. The IDE can also be

integrated with build tools like setuptools, distutils, and pip.

In addition to these features, PyCharm also provides support for a range of other functionalities, such as code documentation, code formatting, code search, and code snippets. PyCharm is available in both a free, open-source community edition and a paid professional edition with additional features and support. It runs on Windows, macOS, and Linux, and it can be used for a wide range of Python projects, from small scripts to large web applications.

Steps Involved for installation:

You will have to follow the steps given below to install PyCharm on your system. These steps show the installation procedure starting from downloading the PyCharm package from its official website to creating a new project.

Step 1

Download the required package or executable from the official website of PyCharm https://www.jetbrains.com/pycharm/download/#section=windowsHere you will observe two versions of package for Windows as shown in the screenshot given below –

Fig. 5.1 Pycharm Installation

Note that the professional package involves all the advanced features and comes with free trial for few days and the user has to buy a licensed key for activation beyond the trial period. Community package is for free and can be downloaded and installed as and when required. It includes all the basic features needed for installation.

Step 2

Download the community package (executable file) onto your system and mention a destination folder as shown below –

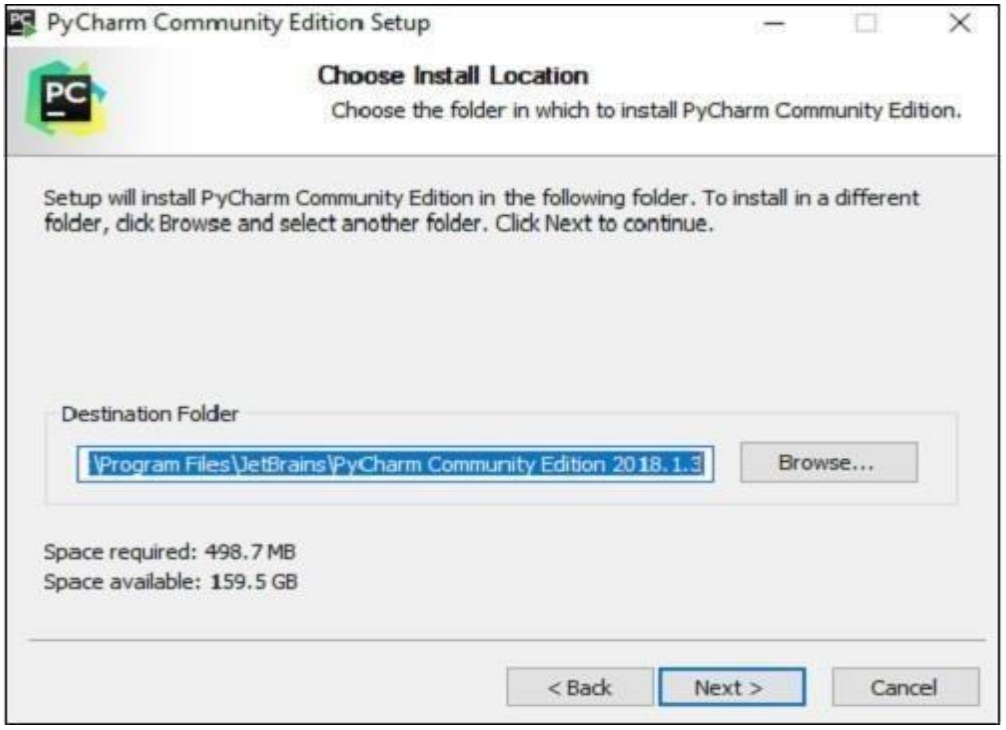


Fig.5.2 PyCharm Setup

PyCharm is the most popular IDE used for Python scripting language. This chapter will give you an introduction to PyCharm and explains its features.

PyCharm offers some of the best features to its users and developers in the following aspects –

- Code completion and inspection
- Advanced debugging
- Support for web programming and frameworks such as Django and Flask

5.1.2 Python

Python is a high-level, interpreted programming language that was first released in 1991 by Guido van Rossum. Since then, it has become one of the most popular programming languages in the world, with a wide range of applications, for python is a high-level, interpreted programming language that was first released in 1991 by Guido van Rossum. Since then, it has become one of the most popular programming languages in the world, with a wide range of applications, from web development and data science to machine learning and artificial intelligence.

Installing Python

1. To download and install Python visit the official website of

Fig.5.3 Python Installation

Python https://www.python.org/downloads/ and choose your version.

2. Once the download is complete, run the exe for install Python. Now click on Install Now.

3. You can see Python installing at this point.

4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Here are some key features of Python:

Readable and easy to learn: Python has a simple syntax and is easy to learn for beginners. Its code is also highly readable and expressive, making it easy to understand and maintain.

Interpreted language: Python is an interpreted language, meaning that code is executed directly by the interpreter, without the need for compilation.

Cross-platform: Python code can run on a wide range of platforms, including Windows, Mac, Linux, and mobile devices.

Dynamic typing: Python is a dynamically typed language, meaning that variable types are inferred at runtime rather than being explicitly declared.

Strong standard library: Python has a large standard library that provides a wide range of functionality out of the box, including modules for string processing, regular expressions, network programming, and more.

Object-oriented: Python is an object-oriented language, meaning that everything in Python is an object, including functions and modules.

High-level: Python is a high-level language, meaning that it provides abstractions that make it easier to express complex concepts without worrying about low-level details.

Versatile: Python can be used for a wide range of applications, including web development, scientific computing, data analysis, artificial intelligence, and more.

Large community: Python has a large and active community of developers, with a wealth of resources and libraries available.

Open source: Python is an open-source language, meaning that its source code is freely available and can be modified and distributed by anyone.

5.1.3 Flask Framework

Flask is a web application framework for Python, designed to make building web applications easy and fast. It is a lightweight framework that is easy to learn and use, making it a popular choice for small to medium-sized web applications.

Flask is built on top of the Werkzeug WSGI toolkit and the Jinja2 template engine,

which provides a powerful set of tools for handling requests and responses, as well as rendering HTML templates.

Installation:

Install libraries-flask in pycharm

- Windows users can install flask via pip command:

pip install flask

One of the main features of Flask is its simplicity. Flask's API is very simple and easy to use, with minimal boilerplate code required to get started. Flask is also highly modular, allowing developers to easily add or remove components as needed.

Flask provides a set of built-in tools for handling routing, input validation, form processing, and more. It also supports a wide range of plugins and extensions, making it easy to add additional functionality to your web application.

Flask uses a concept called "views" to handle requests from the user. A view is a Python function that is decorated with the @app.route decorator, which specifies the URL route that the function will handle. Views can return a variety of responses, including HTML templates, JSON data, and more.

Flask also provides built-in support for handling authentication and security, with support for OAuth, token-based authentication, and more. It also supports a variety of databases, including SQLite, MySQL, and PostgreSQL, making it easy to integrate your web application with a database.

Another feature of Flask is its support for testing. Flask provides a built-in testing framework that allows developers to write automated tests for their web application, ensuring that it is functioning correctly and reliably.

Overall, Flask is a powerful and flexible web application framework that is well-suited for small to medium-sized web applications. Its simplicity, modularity, and flexibility make it easy to learn and use, while its built-in tools and support for plugins andextensions make it easy to add additional functionality as needed. If you're looking to build a web application with Python, Flask is definitely worth considering.

5.1.4 Libraries Used

1. **Matplotlib**

Matplotlib is an open-source data visualization library for the Python programming language. It provides a wide range of tools for creating high-quality charts, graphs, and other visualizations.

The core component of Matplotlib is the pyplot module, which provides a convenient interface for creating plots and charts. Some of the most commonly used types of plots that can be created using Matplotlib include line plots, scatter plots, bar charts, histograms, and heatmaps. Matplotlib provides a lot of flexibility in terms of customization options for plots, such as changing the colors, line styles, and marker styles of data points, adding annotations, adjusting the size and position of the plot, and much more.

To use the Matplotlib library in your Python code, you can import it using the following command:

import matplotlib.pyplot as plt

Matplotlib provides a lot of customization options to make your plots look great. You can change the colors, line styles, and marker styles of data points, add annotations, adjust the size and position of the plot, and much more.

These are just a few examples of what you can do with Matplotlib. The library provides a wide range of tools for creating all kinds of visualizations, from simple line plots to complex 3D plots.

For 2D displays of arrays, Matplotlib is a fantastic Python visualization library. A multi-platform data visualization package called Matplotlib was created to deal with the larger SciPy stack and is based on NumPy arrays. One of visualization's biggest advantages is that it gives us visual access to vast volumes of data in forms that are simple to understand. There are numerous plots in Matplotlib, including line, bar, scatter, histogram, etc. Install the matplotlib package by running the command python -mpip install -U matplotlib, and then import matplotlib. pyplot as plt.

2. **Scikit-learn**

Scikit-learn (also known as sklearn) is an open-source machine learning library for the Python programming language. It provides a wide range of tools for building

and applying machine learning models, including supervised and unsupervised learning, dimensionality reduction, feature selection, and model selection and evaluation.

Scikit-learn includes implementations of many popular machine learning algorithms, such as linear regression, logistic regression, decision trees, random forests, support vector machines (SVMs), k-nearest neighbors (KNN), and clustering algorithms (k-means, hierarchical clustering, etc.). It also includes tools for preprocessing data (e.g., scaling, normalization, encoding categorical variables, etc.) and for evaluating the performance of machine learning models (e.g., cross-validation, metrics for classification and regression, etc.).

Scikit-learn is designed to be easy to use and provides a consistent interface for all the algorithms it implements. It is built on top of other popular scientific computing libraries in Python, such as NumPy, SciPy, and matplotlib, and is widely used in both academia and industry for a variety of machine learning tasks.

To use the scikit-learn library in your Python code, you can import it using the following command:

import sklearn

This will allow you to use all the functions and tools provided by scikit-learn.

Scikit-learn is an open-source data analysis library and the Python ecosystem's gold standard for Machine Learning (ML). The following are key concepts and features: Algorithmic decision-making techniques, such as:

Classification is the process of identifying and categorizing data based on patterns. Regression is the process of predicting or projecting data values using the average mean of existing and planned data.

Clustering is the automatic classification of similar data into datasets.

Installation:

Install libraries – scikit-learn in pycharm

- Users can install scikit-learn via pip command:

pip install scikit-learn

3. Pandas

Pandas is an open-source library designed primarily for working quickly and

logically with relational or labelled data. It offers a range of data structures and procedures for working with time - series data and quantitative information. The

NumPy library serves as the foundation for this library. Pandas is quick and offers its users exceptional performance & productivity. Checking to see if pandas is installed in the Python folder is the first step in using it. If not, use the pip command to install it on our machine. Enter the command cmd0 in the search window, and then use the cd command to find the location where the python-pip file is installed. Locate it and enter the following command: pip install panda and then import panda as pd. In our project, Panda's library helps us work with the csv format database we have acquired.

Pandas is an open-source data manipulation and analysis library for the Python programming language. It provides data structures for efficiently storing and manipulating large datasets, as well as tools for data cleaning, merging, reshaping, and aggregation.

The two primary data structures provided by Pandas are Series and DataFrame. A Series is a one-dimensional labeled array that can hold any data type (integers, floating-point numbers, strings, Python objects, etc.). A DataFrame is a two-dimensional labeled data structure with columns of potentially different types. It is similar to a spreadsheet or a SQL table.

Pandas provides a wide range of functions for working with data, including filtering, sorting, grouping, joining, and reshaping. It also includes powerful tools for data visualization, time series analysis, and data input/output.

To use the Pandas library in your Python code, you can import it using the following command:

import pandas as pd

This will allow you to use all the functions and data structures provided by Pandas.

Installation:

Install libraries-pandas in pycharm

- Users can install pandas via pip command:

pip install pandas

5. 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.

NumPy (short for Numerical Python) is an open-source Python library that is used for scientific computing and numerical analysis. It provides powerful tools for working with arrays and matrices of numerical data, as well as a wide range of mathematical functions for performing complex calculations.

To use the NumPy library in your Python code, you can import it using the following command:

import numpy as np

This will allow you to use all the functions and data structures provided by NumPy.It is open-source software. 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 anefficient 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.

NumPy's main object is the homogeneous multidimensional array.

- It is a table of elements (usually numbers), all of the same type, indexed by a tuple of positive integers.
- In NumPy dimensions are called *axes*. The number of axes is *rank*.
- NumPy's array class is called **ndarray**. It is also known by the alias **array**. **Installation:**

Install libraries numpy in pycharm

- Windows users can install NumPy via pip command:

pip install numpy

4. Seaborn

Seaborn is a Python library for data visualization that is built on top of Matplotlib. It provides a high-level interface for creating beautiful and informative statistical graphics. Seaborn is particularly useful for exploring and visualizing complex datasets with multiple variables.

Seaborn provides a wide range of built-in functions for creating different types of visualizations, including scatter plots, line plots, bar plots, heatmaps, and more. It also includes a number of advanced features, such as support for categorical data, automatic estimation and plotting of regression models, and support for multiple plot grids.

One of the key features of Seaborn is its support for creating aesthetically pleasing visualizations. It provides a number of built-in color palettes that are designed to be easy on the eyes, as well as a range of customizable themes that can be used to adjust the overall look and feel of the visualizations.

Seaborn also provides a number of tools for visualizing relationships between variables in datasets, including correlation matrices and pair plots. These tools make it easy to identify patterns and relationships in data, and can help to guide data analysis and modeling efforts.

Here are some of the specific plot types and functions that Seaborn provides:

Line plots: Seaborn provides several functions for creating line plots, including lineplot, relplot, and tsplot. These functions can be used to visualize trends in your data over time or across different variables.

Scatter plots: Seaborn's scatterplot function can be used to create scatter plots with one or more variables. You can customize the size and color of the points based on additional variables to create multi-dimensional scatter plots.

Bar plots: Seaborn's barplot function can be used to create bar plots with one or more variables. You can customize the colors and orientation of the bars to create different types of bar plots.

Heatmaps: Seaborn's heatmap function can be used to create heatmaps to visualize relationships between two variables. You can customize the color map

and annotations to create informative heatmaps.

Pair plots: Seaborn's pairplot function can be used to create scatter plots between all pairs of variables in your data set. This can be a useful way to explore the relationships between multiple variables.

Joint plot: A joint plot is used to visualize the relationship between two continuous variables and the distribution of each variable.

Histogram: A histogram is used to visualize the distribution of a single continuous variable.

Box plot: A box plot is used to visualize the distribution of a single continuous variable and to identify outliers.

Violin plot: A violin plot is similar to a box plot, but it provides a more detailed view of the distribution of the data.

Regression plots: Seaborn's regplot and lmplot functions can be used to create regression plots to visualize the relationship between two variables.

5.2 ALGORITHMS USED

5.2.1 Logistic Regression

Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable (target) is categorical.

For example,

To predict whether an email is spam (1) or (0)

Whether the tumor is malignant (1) or not (0)

Consider a scenario where, need to classify whether an email is spam or not. Use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done. Say if the actual class is malignant, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not malignant which can lead to serious consequence in real time.

From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. Their value strictly ranges from 0 to 1.

Uses Of Logistic Regression

Logistic regression has become particularly popular in online advertising, enabling

marketers to predict the likelihood of specific website users who will click on particular advertisements as a yes or no percentage.

- Logistic regression can also be used in:
- Healthcare to identify risk factors for diseases and plan preventive measures.
- Weather forecasting apps to predict snowfall and weather conditions.
- Voting apps to determine if voters will vote for a particular candidate.

Insurance to predict the chances that a policy holder will die before the term of the policy expires based on certain criteria, such as gender, age and physical examination.

Banking to predict the chances that a loan applicant will default on a loan or not, based on annual income, past defaults and past debts.

5.2.2 DECISION TREE CLASSIFIER

A tree has many analogies in real life, and turns out that it has influenced a wide area of machine learning, covering both classification and regression. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. As the name goes, it uses a tree-like model of decisions. Though a commonly used tool in data mining for deriving a strategy to reach a particular goal.

A decision tree is drawn upside down with its root at the top. In the image on the left, the bold text in black represents a condition/internal node, based on which the tree splits into branches/ edges. The end of the branch that doesn't split anymore is the decision/leaf, in this case, whether the passenger died or survived, represented as red and green text respectively.

Although, a real dataset will have a lot more features and this will just be a branch in a much bigger tree, but you can't ignore the simplicity of this algorithm. The feature importance is clear and relations can be viewed easily. This methodology is more commonly known as learning decision tree from data and above tree is called Classification tree as the target is to classify passenger as survived or died. Regression trees are represented in the same manner, just they predict

continuous values like price of a house. In general, Decision Tree algorithms are referred to as CART or Classification and Regression Trees.

So, what is actually going on in the background? Growing a tree involves deciding on which features to choose and what conditions to use for splitting, along with knowing when to stop. As a tree generally grows arbitrarily, you will need to trim it down for it to look beautiful. Let's start with a common technique used for splitting

5.2.3 RANDOM FOREST CLASSIFIER

A random forest is a machine learning technique that's used to solve regression and classification problems. It utilizes ensemble learning, which is a technique that combines many classifiers to provide solutions to complex problems.

A random forest algorithm consists of many decision trees. The 'forest' generated by the random forest algorithm is trained through bagging or bootstrap aggregating. Bagging is an ensemble meta-algorithm that improves the accuracy of machine learning algorithms.

The (random forest) algorithm establishes the outcome based on the predictions of the decision trees. It predicts by taking the average or mean of the output from various trees. Increasing the number of trees increases the precision of the outcome.

A random forest eradicates the limitations of a decision tree algorithm. It reduces the over fitting of datasets and increases precision. It generates predictions without requiring many configurations in packages (like Scikit-learn).

Features of a Random Forest Algorithm:

- It's more accurate than the decision tree algorithm.
- It provides an effective way of handling missing data.
- It can produce a reasonable prediction without hyper-parameter tuning.
- It solves the issue of over fitting in decision trees.
- In every random forest tree, a subset of features is selected randomly at the node's splitting point.

Decision trees are the building blocks of a random forest algorithm. A decision tree is a decision support technique that forms a tree-like structure. An overview

of decision trees will help us understand how random forest algorithms work.

A decision tree consists of three components: decision nodes, leaf nodes, and a

root node. A decision tree algorithm divides a training dataset into branches, which further segregate into other branches. This sequence continues until a leaf node is attained. The leaf node cannot be segregated further.

The nodes in the decision tree represent attributes that are used for predicting the outcome. Decision nodes provide a link to the leaves. The following diagram shows the three types of nodes in a decision tree.

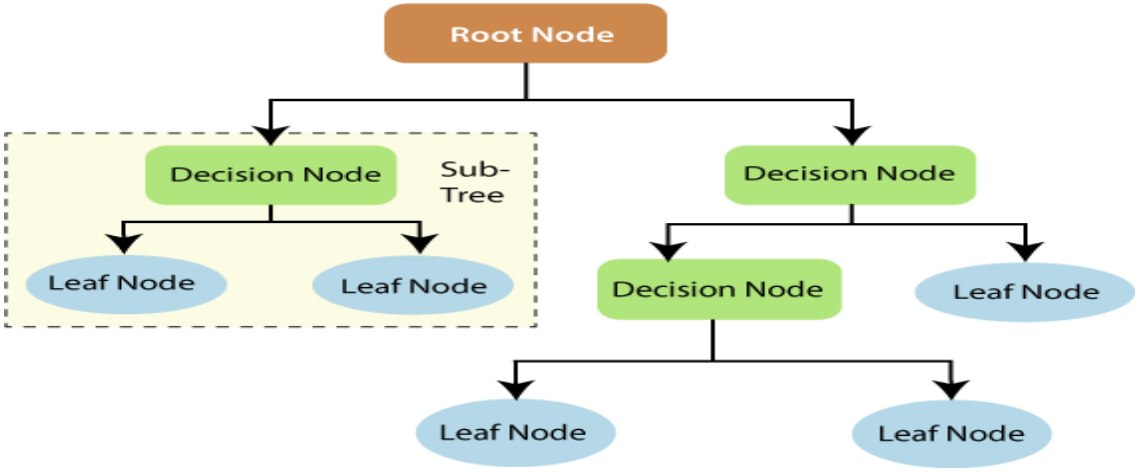


Fig. 5.4 ***Types of nodes in decision tree***

The information theory can provide more information on how decision trees work. Entropy and information gain are the building blocks of decision trees. An overview of these fundamental concepts will improve our understanding of how decision trees are built.

Entropy is a metric for calculating uncertainty. Information gain is a measure of how uncertainty in the target variable is reduced, given a set of independent variables.

The information gain concept involves using independent variables (features) to gain information about a target variable (class). The entropy of the target variable (Y) and the conditional entropy of Y (given X) are used to estimate the information gain. In this case, the conditional entropy is subtracted from the entropy of Y.

Information gain is used in the training of decision trees. It helps in reducing uncertainty in these trees. A high information gain means that a high degree of uncertainty (information entropy) has been removed. Entropy and information gain are important in splitting branches, which is an important activity in the

construction of decision trees.

Let's take a simple example of how a decision tree works. To predict if a customer will purchase a mobile phone or not. The features of the phone form the basis of his decision. This analysis can be presented in a decision tree diagram.

The root node and decision nodes of the decision represent the features of the phone mentioned above. The leaf node represents the final output, either buying or notbuying. The main features that determine the choice include the price, internal storage, and Random Access Memory (RAM). The decision tree will appear as follows.

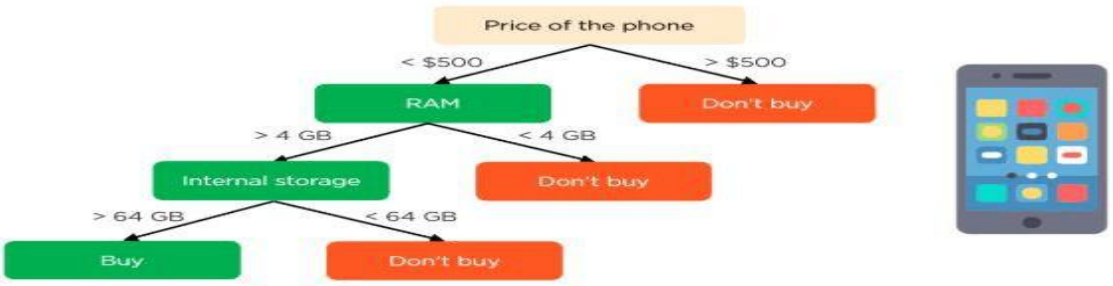


Fig. 5.5 ***Example of decision tree***

Applying decision trees in random forest

The main difference between the decision tree algorithm and the random forest algorithm is that establishing root nodes and segregating nodes is done randomly in the latter. The random forest employs the bagging method to generate the required prediction.

Bagging involves using different samples of data (training data) rather than just one sample. A training dataset comprises observations and features that are used for making predictions. The decision trees produce different outputs, depending on the training data fed to the random forest algorithm. These outputs will be ranked, and the highest will be selected as the final output.

Our first example can still be used to explain how random forests work. Instead of having a single decision tree, the random forest will have many decision trees. Let's assume, only four decision trees. In this case, the training data comprising the phone's observations and features will be divided into four root nodes.

The root nodes could represent four features that could influence the customer's

choice (price, internal storage, camera, and RAM). The random forest will split the nodes by selecting features randomly. The final prediction will be selected based

on the outcome of the four trees.
The outcome chosen by most decision trees will be the final choice. If three trees predict buying, and one tree predicts not buying, then the final prediction will be buying. In this case, it's predicted that the customer will buy the phone.

5.3 TESTING

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.

5.3.1 Unit Testing

Unit testing is a crucial part of the software development process, including machine learning models. Unit testing the detection of fake accounts on Instagram using machine learning:

Define the Input and Output: The first step is to define the input and output of the function or method that is testing. For example, the input could be a set of features extracted from an Instagram account, and the output could be a binary classification (real or fake).

Create Test Cases: Once you have defined the input and output, you can create test cases to verify that the function or method works correctly. Test cases should cover both positive and negative scenarios, including cases where the input is valid and invalid.

Set Up Test Environment: You will need to set up a test environment to run the

test cases. This includes creating a testing database or file with sample data, setting up any necessary dependencies, and configuring any environment

variables needed to run the tests.

Run Test Cases: Then run the test cases to check if the function or method is working as expected. The tests should validate that the model accurately detects real and fake accounts based on the provided features.

Evaluate Results: After running the tests, evaluate the results to determine if the model is correctly identifying real and fake accounts. If any test cases fail, will need to debug and fix the code to address the issue.

Repeat the Process: Unit testing is an iterative process, and need to repeat the above steps as you make changes to the model or function.

.

CHAPTER 6

RESULTS AND DISCUSSION

User can select any model among three models. Then the system will generate accuracy for a particular selected model. Users can also view that generated accuracy, this process will be done on the model page. The selection of the model is done with the highest accuracy given algorithm. Logistic regression gave 89.9% accuracy, Decision tree got 89.5% accuracy and Random forest gave the highest accuracy of 91.8%.

Home Page

In the application ,the home page shows a welcome page of showing information of detecting fake accounts.

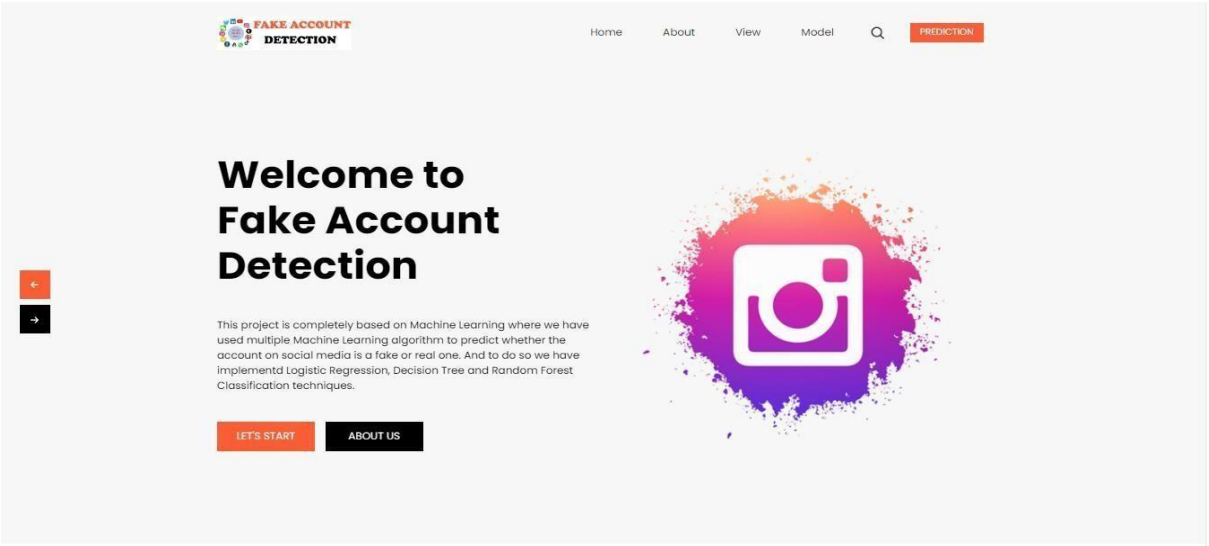


Fig.6.1 *Home*

About Page

In the application,the about page provides the information of fake account detection using machine learning, importance of detecting fake accounts and necessity of detecting fake accounts on social media.

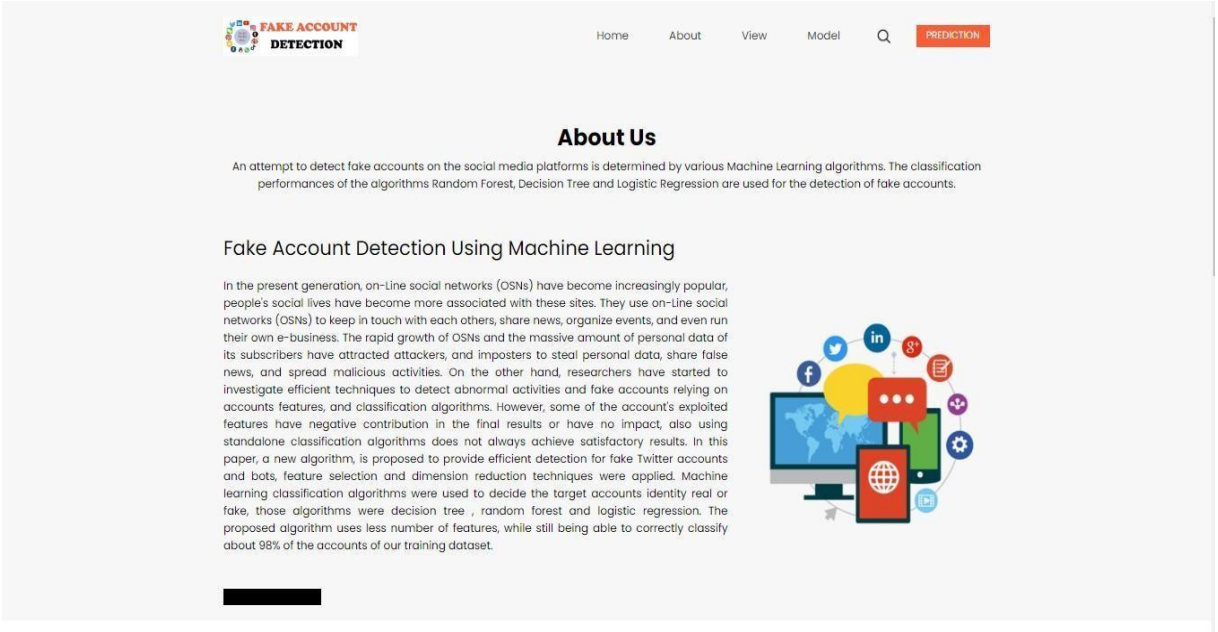
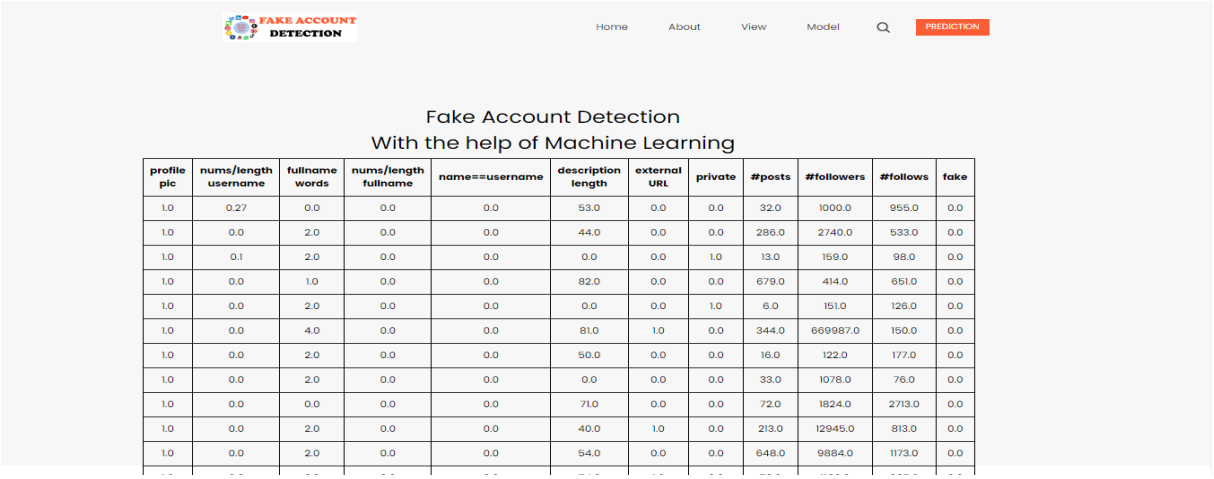


Fig. 6.2 *About*

View Page

In the application, view page shows the information about the dataset with 12 attributes which is downloaded from kaggle website



profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
1.0	0.27	0.0	0.0	0.0	53.0	0.0	0.0	32.0	1000.0	955.0	0.0
1.0	0.0	2.0	0.0	0.0	44.0	0.0	0.0	286.0	2740.0	533.0	0.0
1.0	0.1	2.0	0.0	0.0	0.0	0.0	1.0	13.0	159.0	98.0	0.0
1.0	0.0	1.0	0.0	0.0	82.0	0.0	0.0	679.0	414.0	651.0	0.0
1.0	0.0	2.0	0.0	0.0	0.0	0.0	1.0	6.0	151.0	126.0	0.0
1.0	0.0	4.0	0.0	0.0	81.0	1.0	0.0	344.0	669987.0	150.0	0.0
1.0	0.0	2.0	0.0	0.0	50.0	0.0	0.0	16.0	122.0	177.0	0.0
1.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	33.0	1078.0	76.0	0.0
1.0	0.0	0.0	0.0	0.0	71.0	0.0	0.0	72.0	1824.0	2713.0	0.0
1.0	0.0	2.0	0.0	0.0	40.0	1.0	0.0	213.0	12945.0	813.0	0.0
1.0	0.0	2.0	0.0	0.0	54.0	0.0	0.0	648.0	9884.0	1173.0	0.0

Fig. 6.3 *View*

Model Selection Page

In the application, there is an option to choose an algorithm within that need to selct the algorithm which gives highest accuracy which is random forest with 91.8%.

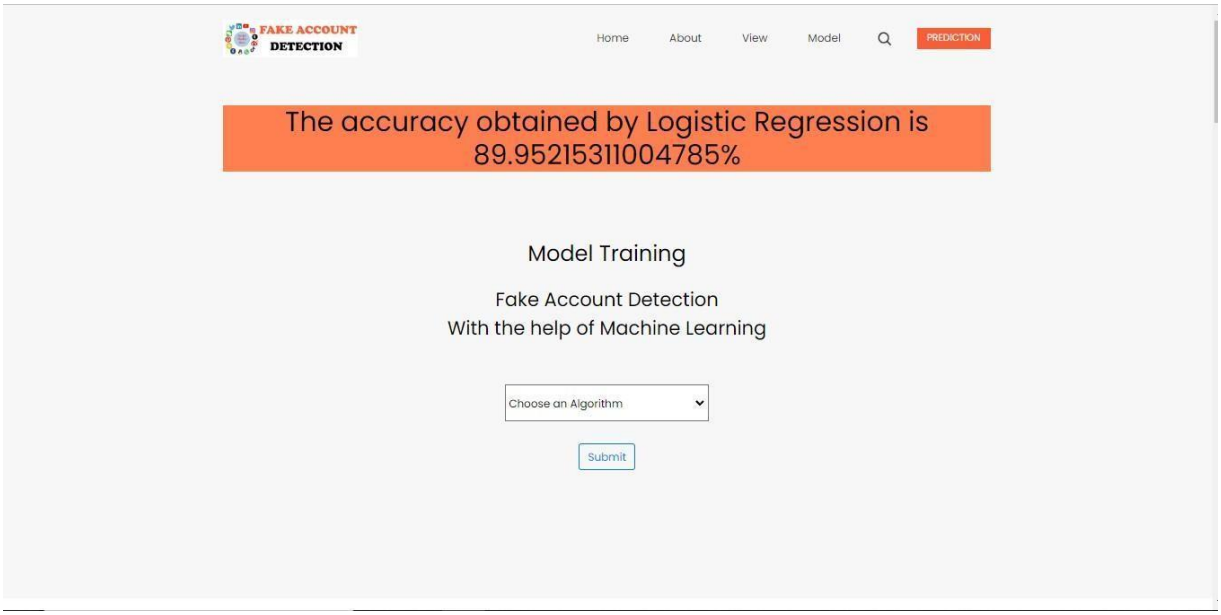


Fig : 6.4 Model Selection

Prediction

In the application, after choosing highest accuracy given algorithm, can able to predict an account is fake or not with the user input

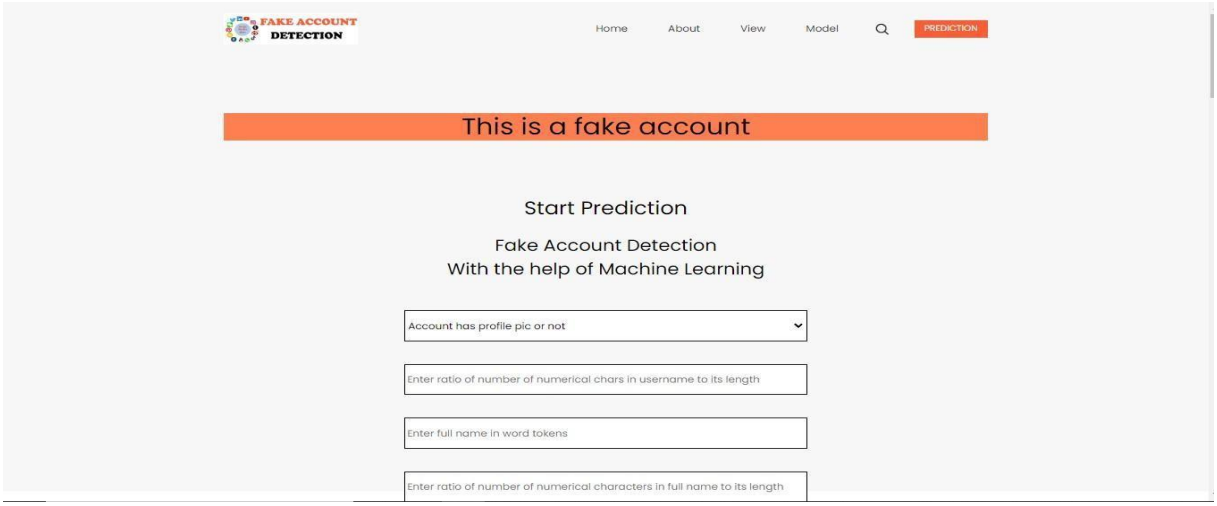


Fig. 6.5 Prediction

Feature Analysis

Feature analysis is a critical step in machine learning, which involves identifying and selecting the most relevant features in a dataset. Feature analysis helps to

improve model performance, reduce complexity, and optimize computational efficiency.

Accuracy

Accuracy = (Number of Correct Predictions) / (Total Number of Predictions)

To illustrate this formula, consider a binary classification problem where there are test set of 100 instances, of which 60 are labeled as class A and 40 are labeled as class B. We train a binary classification model on the training set, and when we evaluate the model on the test set.

Table of Accuracy

Sl No.	Model	Accuracy
1	Logistic Regression	89.9%
2	Random Forest	91.8%
3	Decision Tree	89.4%

Table no. 6.1 Accuracy

Accuracy for logistic regression is 88.9%.

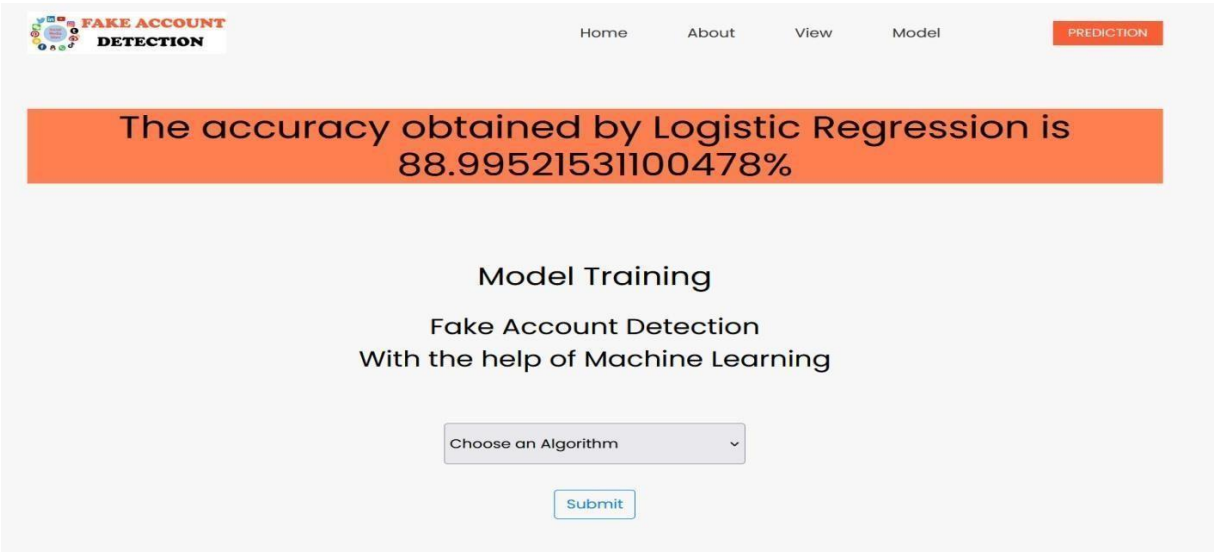


Fig. 6.6 Logistic regression accuracy

Accuracy for random forest is 91.8%

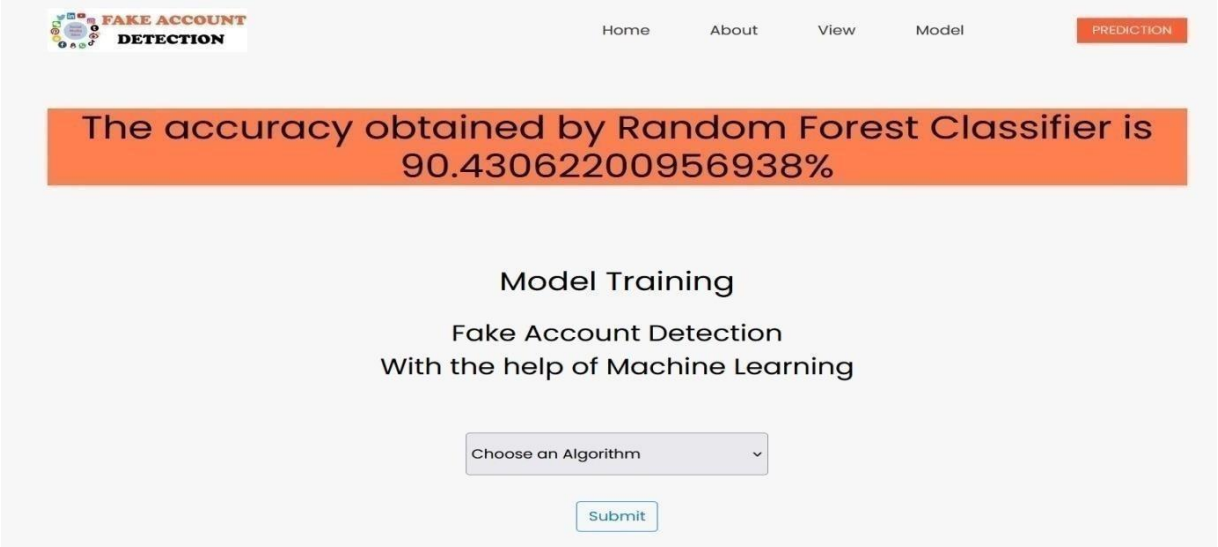


Fig. 6.7 Random Forest accuracy

Accuracy for decision tree is 89.4%.

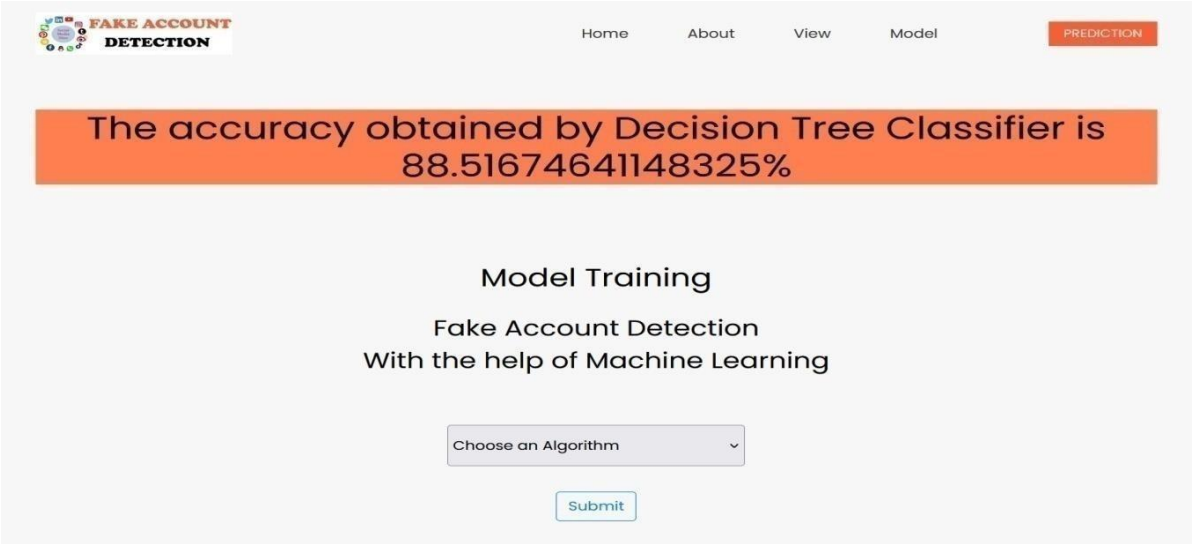


Fig. 6.8 Decision tree accuracy

After comparing all the three machine learning model logistic regression, random forest, and decision tree, Random forest got the highest accuracy with 91.8%.

CHAPTER 7

CONCLUSION

7.1 CONCLUSION

A machine-learning approach for detecting fake accounts on social media as previously stated, three popular algorithms were compared. After the comparison, random forest algorithm got the best accuracy. Hence, Random Forest classifier algorithm can be considered for the rest of the project. Simply, our project has two phases- frontend and backend. In the backend, did the data pre- processing, model selection, prediction and taking best accuracy given algorithm. Later, in the front end included this code in a framework and with the user input can detect the fake account or not. Hence, concluded that detecting fraudulent profiles depends on the features used in training the data set and classifiers used.

7.2 FUTURE WORK

Future work in developing of detecting fake accounts on Instagram. Here are some examples:

Larger And More Diverse Dataset: In order to increase the accuracy and generalizability of the fake account detection model, it may be useful to test it on a larger and more diverse dataset of Instagram users. This could involve collecting data from users in different regions, with different types of accounts and activity patterns.

Incorporating Additional Features: As new data becomes available, there may be additional features that could be incorporated into the fake account detection model to improve its accuracy. For example, Instagram's API could provide additional data on user engagement and activity patterns, which could be used to distinguish between real and fake accounts.

Implementing real-time detection: A potential area for future work is to implement

real-time detection of fake accounts. This could involve creating a system that monitors user behavior in real-time and provides alerts if suspicious activity is detected. This would be particularly useful for identifying and removing bot

accounts that are actively spamming or engaging in other malicious activities.

Investigating the use of Deep Learning: Deep learning techniques, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), have been shown to be effective in a variety of machine learning applications. Investigating the use of these techniques for fake account detection on Instagram could be a fruitful area for future research.

Collaborating with Instagram and other social media platforms: Collaborating with Instagram and other social media platforms could be a valuable way to improve the effectiveness of fake account detection methods. This could involve sharing data and insights, or working together to develop new tools and techniques for identifying and removing fake accounts from the platform.

7.3 RESEARCH ISSUES

Data collection: Collecting and annotating a large dataset of Instagram users is a challenging task, particularly when it comes to identifying fake accounts. Research in this area could focus on developing more efficient and accurate methods for data collection and annotation, as well as exploring the best practices for balancing the need for data quality with user privacy concerns.

Feature selection and extraction: Identifying the most relevant features for fake account detection is a crucial step in developing an effective machine learning model. Research in this area could focus on identifying new features that can help distinguish between real and fake accounts, as well as exploring the best methods for feature extraction and selection.

Model selection and evaluation: Choosing the most appropriate machine learning algorithm for fake account detection is an important decision, and there are many different options to choose from. Research in this area could explore the

effectiveness of different machine learning algorithms for fake account detection, as well as the best practices for model evaluation and selection.

Addressing imbalanced data: Imbalanced data is a common problem in fake account detection, as there are typically far more real accounts than fake accounts in any given dataset. Research in this area could focus on developing effective methods for addressing imbalanced data, such as oversampling, under sampling, or using cost-sensitive learning algorithms.

Ethics and privacy: Developing a fake account detection model raises important ethical and privacy concerns, particularly when it comes to user data collection and sharing. Research in this area could explore the best practices for balancing the need for data privacy with the need for data quality, as well as the ethical implications of using machine learning to detect and remove fake accounts on social media platforms.

7.3 IMPLEMENTATION ISSUES

Data collection: Collecting a large and diverse dataset of Instagram users is a crucial step in developing an effective fake account detection model. However, collecting data can be time-consuming and resource-intensive, particularly when it comes to identifying and annotating fake accounts. Implementing an efficient and effective data collection process will be important for the success of the project.

Feature engineering: Identifying the most relevant features for fake account detection is another important step in developing an effective machine learning model. However, feature engineering can be a complex process that requires careful consideration of both the features themselves and the algorithms used to extract them. Implementing an effective feature engineering process will be key to developing an accurate and efficient model.

Model selection and tuning: Choosing the most appropriate machine learning

algorithm and tuning its parameters is another important step in developing an effective fake account detection model. Different algorithms and parameter settings can have a significant impact on the performance of the model, and the process of selecting and tuning the model can be time-consuming and require a

deep understanding of machine learning techniques.

Deployment and scalability: Once a fake account detection model has been developed, deploying it to a production environment can be a complex process. The model may need to be integrated with existing systems and software, and it may need to be tested and optimized for scalability and performance. Implementing an effective deployment and scalability plan will be important for ensuring that the model can be used effectively in a real-world setting.

User privacy and ethical considerations: Developing a fake account detection model raises important ethical and privacy concerns, particularly when it comes to user data collection and sharing. Implementing appropriate privacy policies and procedures, and ensuring that the project adheres to ethical guidelines and regulations, will be important for ensuring that the project is both effective and responsible.

REFERENCES

[1] B. Erşahin, Ö. Aktaş, D. Kılınç, and C. Akyol, "Twitter fake account detection," *2017 International Conference on Computer Science and Engineering (UBMK)*, Antalya, Turkey, 2017, pp. 388-392, doi: 10.1109/UBMK.2017.8093420.

[2] A. Gupta and R. Kaushal, "Towards detecting fake user accounts in Facebook," *2017 ISEA Asia Security and Privacy (ISEASP)*, Surat, India, 2017, pp. 1-6, doi: 10.1109/ISEASP.2017.7976996.

[3] Y. -C. Chen and S. F. Wu, "Fake Buster: A Robust Fake Account Detection by Activity Analysis," *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, Taipei, Taiwan, 2018, pp. 108-110, doi: 10.1109/PAAP.2018.00026.

[4] E. Van Der Walt and J. Eloff, "Using Machine Learning to Detect Fake Identities: Bots vs Humans," in *IEEE Access*, vol. 6, pp. 6540-6549, 2018, doi: 10.1109/ACCESS.2018.2796018.

[5] L. Zhang, Z. Wu and J. Cao, "Detecting Spammer Groups from Product Reviews: A Partially Supervised Learning Model," in *IEEE Access*, vol. 6, pp. 2559-2568, 2018, doi: 10.1109/ACCESS.2017.2784370.

[6] S. Khaled, N. El-Tazi and H. M. O. Mokhtar, "Detecting Fake Accounts on Social Media," *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 3672-3681, doi: 10.1109/BigData.2018.8621913.

[7] Z. Alom, B. Carminati and E. Ferrari, "Detecting Spam Accounts on Twitter," *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, Barcelona, Spain, 2018, pp. 1191-1198, doi: 10.1109/ASONAM.2018.8508495.

[8] N. Singh, T. Sharma, A. Thakral and T. Choudhury, "Detection of Fake Profile in Online Social Networks Using Machine Learning," *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, Paris, France, 2018, pp. 231-234, doi: 10.1109/ICACCE.2018.8441713.

[9] S. Singhal, R. R. Shah, T. Chakraborty, P. Kumaraguru and S. Satoh, "Spot Fake: A Multi-modal Framework for Fake News Detection," *2019 IEEE Fifth*

International Conference on Multimedia Big Data (BigMM), Singapore, 2019, pp. 39-47, doi: 10.1109/BigMM.2019.00-44.

[10] P. Sowmya and M. Chatterjee, "Detection of Fake and Clone accounts in Twitter using Classification and Distance Measure Algorithms," *2020 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, India, 2020, pp. 0067-0070, doi: 10.1109/ICCSP48568.2020.9182353.

[11] Z. Shahbazi and Y. -C. Byun, "Fake Media Detection Based on Natural Language Processing and Blockchain Approaches," in *IEEE Access*, vol. 9, pp. 128442-128453, 2021, doi: 10.1109/ACCESS.2021.3112607.

[12] L. P, S. V, V. Sasikala, J. Arunarasi, A. R. Rajini and N. Nithiya, "Fake Profile Identification in Social Network using Machine Learning and NLP," *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, Chennai, India, 2022, pp. 1-4, doi: 10.1109/IC3IOT53935.2022.9767958.

M. J. Ekosputra, A. Susanto, F. Haryanto and D. Suhartono, "Supervised Machine Learning Algorithms to Detect Instagram Fake Accounts," *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Yogyakarta, Indonesia, 2021, pp. 396-400, doi: 10.1109/ISRITI54043.2021.9702833.

[13] K. Anklesaria, Z. Desai, V. Kulkarni and H. Balasubramaniam, "A Survey on Machine Learning Algorithms for Detecting Fake Instagram Accounts," *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2021, pp. 141-144, doi: 10.1109/ICAC3N53548.2021.9725724.

[14] M. M. Hassan Sohan, M. M. Khan, I. Nanda and R. Dey, "Fake Product Review Detection Using Machine Learning," *2022 IEEE World AI IoT Congress (AIIoT)*, Seattle, WA, USA, 2022, pp. 527-532, doi: 10.1109/AIIoT54504.2022.9817271.

[15] Praveena, M.A., Christy, A., Helen, L.S., Krishna, R.S. and Nandini, D.U., 2021, March. Examining and Predicting Helpfulness of reviews based

on Naive Bayes. In Journal of Physics: Conference Series (Vol. 1770, No. 1, p. 012021). IOP Publishing.

APPENDIX

A. SOURCE CODE :

Home.html

```
<!DOCTYPEhtml>
<html>
<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />

<title>Fake Account</title>
<link rel="icon" href="static/images/1.png" type="image/icon type">

<!-- slider stylesheet -->
<link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.1.3/assets/ow
l.carousel.min.css" />

<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="static/css/bootstrap.css" />

<!-- fonts style -->
<link
href="https://fonts.googleapis.com/css?family=Poppins:400,700&display=s
wa p" rel="stylesheet" />
<!-- Custom styles for this template -->
<link href="static/css/style.css" rel="stylesheet" />
<!-- responsive style -->
<link href="static/css/responsive.css" rel="stylesheet" />
</head>

<body>
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container pt-3">
<a class="navbar-brand" href="{{url_for('index')}}">
<img src="static/images/1.png" alt="" style="width: 200px;" /><span>
</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
```

Index.html

```
<!DOCTYPE html>
<html>
<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />

<title>Fake Account</title>
<link rel="icon" href="static/images/1.png" type="image/icon type">
<!-- slider stylesheet -->
<link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.1.3/assets/ow
l.carousel.min.css" />

<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="static/css/bootstrap.css" />

<!-- fonts style -->
<link
href="https://fonts.googleapis.com/css?family=Poppins:400,700&display=s
wa p" rel="stylesheet" />
<!-- Custom styles for this template -->
<link href="static/css/style.css" rel="stylesheet" />
<!-- responsive style -->
<link href="static/css/responsive.css" rel="stylesheet" />
</head>

<body>
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container pt-3">
<a class="navbar-brand" href="{{url_for('index')}}">
<img src="static/images/1.png" alt="" style="width: 200px;" /><span>
</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>

<div class="collapse navbar-collapse" id="navbarSupportedContent">
<div class="d-flex ml-auto flex-column flex-lg-row align-items-center">
<ul class="navbar-nav ">
<li class="nav-item active">
<a class="nav-link" href="{{url_for('index')}}">Home <span class="sr-
only">(current)</span></a>
</li>
```

View.html

```
<!DOCTYPE html>
<html>

<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />

<title>Fake Account</title>
<link rel="icon" href="static/images/1.png" type="image/icon type">

<!-- slider stylesheet -->
<link rel="stylesheet" type="text/css"

href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.1.3/assets/ow
l.carousel.min.css" />

<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="static/css/bootstrap.css" />

<!-- fonts style -->
<link
href="https://fonts.googleapis.com/css?family=Poppins:400,700&display=s
wa p" rel="stylesheet" />
<!-- Custom styles for this template -->
<link href="static/css/style.css" rel="stylesheet" />
<!-- responsive style -->
<link href="static/css/responsive.css" rel="stylesheet" />
</head>

<body>
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container pt-3">
<a class="navbar-brand" href="{{url_for('index')}}">
<img src="static/images/1.png" alt="" style="width: 200px;" /><span>

</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
            aria-controls="navbarSupportedContent" aria-expanded="false"
aria-label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
```

Model.html

```
<!DOCTYPE html>
<html>
<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />
<title>Fake Account</title>
<link rel="icon" href="static/images/1.png" type="image/icon type">
<!-- slider stylesheet -->
<link rel="stylesheet"
type="text/css"href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/
2.1.3/assets/owl.carousel.min.css" />
<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="static/css/bootstrap.css" />
<!-- fonts style -->
<link
href="https://fonts.googleapis.com/css?family=Poppins:400,700&display=s
wa p" rel="stylesheet" />
<!-- Custom styles for this template -->
<link href="static/css/style.css" rel="stylesheet" />
<!-- responsive style -->
<link href="static/css/responsive.css" rel="stylesheet" />
</head>
<body>
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container pt-3">
<a class="navbar-brand" href="{{url_for('index')}}">
<img src="static/images/1.png" alt="" style="width: 200px;" /><span>
</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<div class="d-f avbar-nav ">
<li class="nav-item active">
<a class="nav-link" href="{{url_for('index')}}">Home <span class="sr-
only">(current)</span></a>
</li>
<li class="nav-item">
<a class="nav-link" href="{{url_for('about')}}">About</a>
</li>
<li class="nav-item">
<a class="nav-link" href="{{url_for('view')}}">View </a>
</li
```

Prediction.html

```
<!DOCTYPE html>
<html>

<head>
<!-- Basic -->
<meta charset="utf-8" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<!-- Mobile Metas -->
<meta name="viewport" content="width=device-width, initial-scale=1,
shrink-to-fit=no" />
<!-- Site Metas -->
<meta name="keywords" content="" />
<meta name="description" content="" />
<meta name="author" content="" />

<title>Fake Account</title>
<link rel="icon" href="static/images/1.png" type="image/icon type">
<!-- slider stylesheet -->
<link rel="stylesheet"
type="text/css"href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/
2.1.3/assets/owl.carousel.min.css" />
<!-- bootstrap core css -->
<link rel="stylesheet" type="text/css" href="static/css/bootstrap.css" />
<!-- fonts style -->
<link
href="https://fonts.googleapis.com/css?family=Poppins:400,700&display=s
wa p" rel="stylesheet" />
<!-- Custom styles for this template -->
<link href="static/css/style.css" rel="stylesheet" />
<!-- responsive style -->
<link href="static/css/responsive.css" rel="stylesheet" />
</head>

<body>
<div class="hero_area">
<!-- header section strats -->
<header class="header_section">
<div class="container">
<nav class="navbar navbar-expand-lg custom_nav-container pt-3">
<a class="navbar-brand" href="{{url_for('index')}}">
<img src="static/images/1.png" alt="" style="width: 200px;" /><span>
</span>
</a>
<button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent"
aria-controls="navbarSupportedContent" aria-expanded="false" aria-
label="Toggle navigation">
<span class="navbar-toggler-icon"></span>
</button>
<div class="collapse navbar-collapse" id="navbarSupportedContent">
<div class="d-flex ml-auto flex-column flex-lg-row align-items-center">
<ul class="navbar-nav ">
<li class="nav-item active">
<a class="nav-link" href="{{url_for('index')}}">Home <span class="sr-
```

App.py

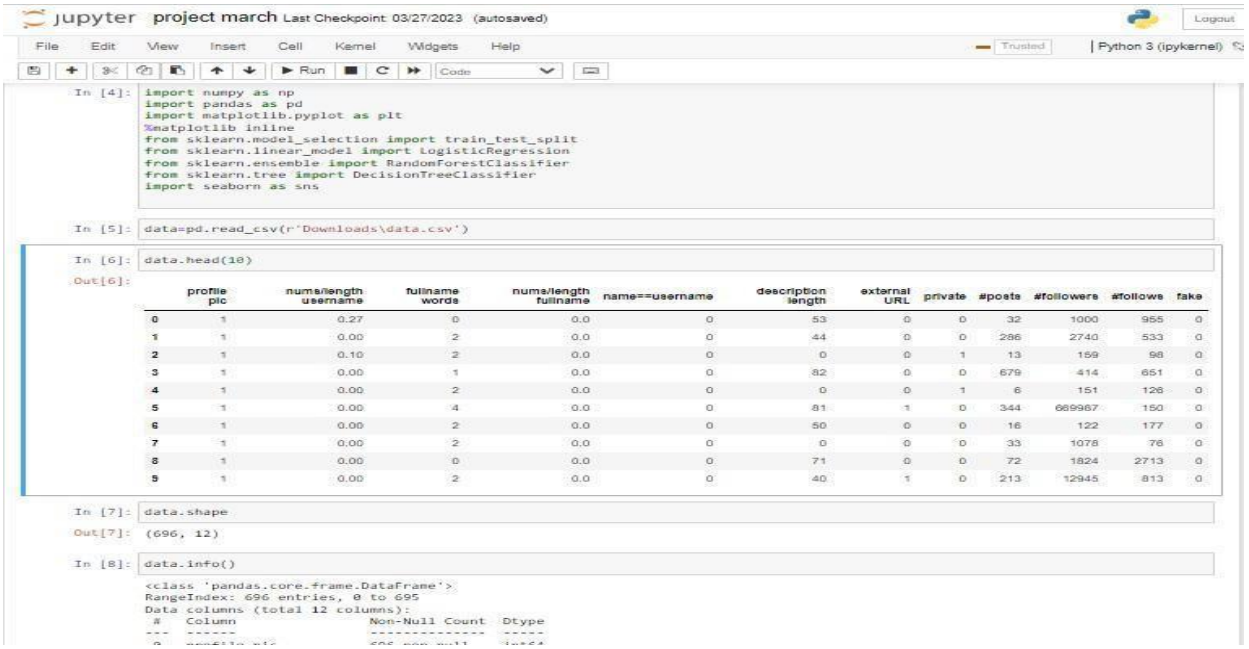
```
import pandas as pd
from flask import *
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
from sklearn.tree import DecisionTreeClassifier from
sklearn.ensemble import RandomForestClassifier from
sklearn.linear_model import LogisticRegression app =
Flask(__name__)
@app.route('/')
def index():
return render_template('index.html')
@app.route('/about')
def about():
return render_template('about.html')

# @app.route('/load',methods=["GET","POST"])
# def load():
#     global df, dataset
#     if request.method == "POST":
#         data = request.files['data']
#         df = pd.read_csv(data)
#         dataset = df.head(100)
#         msg = 'Data Loaded Successfully'
#         return render_template('load.html', msg=msg)
#     return render_template('load.html')
@app.route('/view')
def view():
global df, dataset
    df = pd.read_csv('data.csv')
    dataset = df.head(100)
return render_template('view.html', columns=dataset.columns.values,
rows=dataset.values.tolist())
@app.route('/model',methods=['POST','GET'])
def model():
if request.method=="POST":
        data = pd.read_csv('data.csv')
        data.head()
        x=data.iloc[:,:-1]
        y=data.iloc[:,-1]

        x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.3,stratify=y,random_state=42)print('cccc
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccc')
        s=int(request.form['algo'])
if s==0:
return render_template('model.html',msg='Please Choose an Algorithm to
Train')
elif
s==1:print('aaaaaaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb')
from sklearn.linear_model import   LogisticRegression
            lr = LogisticRegression()
            lr=lr.fit(x_train,y_train)
            y_pred = lr.predict(x_test)
            acc_lr = accuracy_score(y_test,y_pred)*100
print('aaaaaaaaaaaaaaaaaaaaaaaa')
            msg = 'The accuracy obtained by Logistic Regression is ' +
str(acc_lr) + str('%')
```

B. SCREEN SHOTS

Ipnyb.py



jupyter project march Last Checkpoint: 03/27/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Trusted | Python 3 (ipykernel)

Logout

Run Code

In [4]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.tree import DecisionTreeClassifier
import seaborn as sns
```

In [5]:
```
data=pd.read_csv(r'Downloads\data.csv')
```

In [6]:
```
data.head(10)
```

Out[6]:

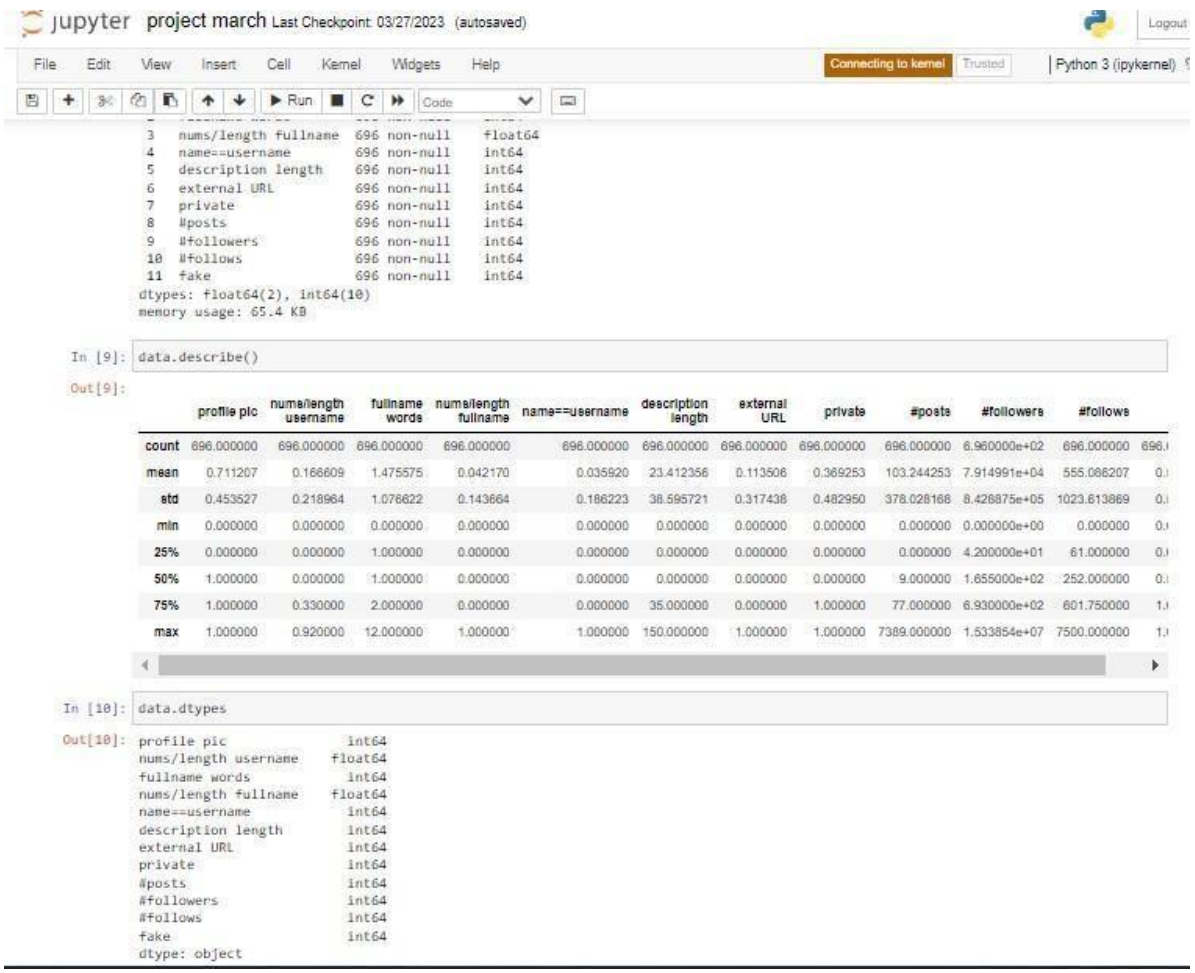
	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
0	1	0.27	0	0.0	0	53	0	0	32	1000	955	0
1	1	0.00	2	0.0	0	44	0	0	286	2740	533	0
2	1	0.10	2	0.0	0	0	0	1	13	159	98	0
3	1	0.00	1	0.0	0	82	0	0	679	414	651	0
4	1	0.00	2	0.0	0	0	0	1	6	151	126	0
5	1	0.00	4	0.0	0	81	1	0	344	669967	150	0
6	1	0.00	2	0.0	0	50	0	0	16	122	177	0
7	1	0.00	2	0.0	0	0	0	0	33	1078	76	0
8	1	0.00	0	0.0	0	71	0	0	72	1824	2713	0
9	1	0.00	2	0.0	0	40	1	0	213	12945	813	0

In [7]:
```
data.shape
```

Out[7]:
```
(696, 12)
```

In [8]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 696 entries, 0 to 695
Data columns (total 12 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
```


jupyter project march Last Checkpoint: 03/27/2023 (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Connecting to kernel
Trusted
Python 3 (ipykernel)
Run
Code
3 nums/length fullname 696 non-null float64
4 name==username 696 non-null int64
5 description length 696 non-null int64
6 external URL 696 non-null int64
7 private 696 non-null int64
8 #posts 696 non-null int64
9 #followers 696 non-null int64
10 #follows 696 non-null int64
11 fake 696 non-null int64
dtypes: float64(2), int64(10)
memory usage: 65.4 KB
In [9]:
data.describe()
Out[9]:
profile pic nums/length username fullname words nums/length fullname name==username description length external URL private #posts #followers #follows
count 696.000000 696.000000 696.000000 696.000000 696.000000 696.000000 696.000000 696.000000 696.000000 6.960000e+02 696.000000
mean 0.711207 0.166609 1.475575 0.042170 0.035920 23.412356 0.113506 0.369253 103.244253 7.914991e+04 555.086207
std 0.453527 0.218964 1.076622 0.143664 0.186223 38.595721 0.317438 0.482950 378.028168 8.428875e+05 1023.613869
min 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000e+00 0.000000
25% 0.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 4.200000e+01 61.000000
50% 1.000000 0.000000 1.000000 0.000000 0.000000 0.000000 0.000000 0.000000 9.000000 1.655000e+02 252.000000
75% 1.000000 0.330000 2.000000 0.000000 0.000000 35.000000 0.000000 1.000000 77.000000 6.930000e+02 601.750000
max 1.000000 0.920000 12.000000 1.000000 1.000000 150.000000 1.000000 1.000000 7389.000000 1.533854e+07 7500.000000
In [10]:
data.dtypes
Out[10]:
profile pic int64
nums/length username float64
fullname words int64
nums/length fullname float64
name==username int64
description length int64
external URL int64
private int64
#posts int64
#followers int64
#follows int64
fake int64
dtype: object

jupyter project march Last Checkpoint: 03/27/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Connecting to kernel — Trusted — Python 3 (ipykernel)

Run Code

In [11]:
```
data['fake'].value_counts()
```

Out[11]:
```
0    348
1    348
Name: fake, dtype: int64
```

In [12]:
```
data.isnull().sum()
```

Out[12]:
```
profile pic            0
nums/length username   0
fullname words         0
nums/length fullname   0
name==username         0
description length     0
external URL           0
private                0
#posts                 0
#followers             0
#follows               0
fake                   0
dtype: int64
```

In [13]:
```
sns.pairplot(data=data, hue="fake")
```

Out[13]:
```
<seaborn.axisgrid.PairGrid at 0x19e827d7cd0>
```

In [14]:
```
##Split dataset into independent snd dependent features
```

jupyter project march Last Checkpoint: 03/27/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Connecting to kernel — Trusted — Python 3 (ipykernel)

Run Code

In [14]:
```
##Split dataset into independent snd dependent features

x=data.iloc[:,:-1]
y=data.iloc[:,-1]
```

In [15]:
```
x
```

Out[15]:

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows
0	1	0.27	0	0.0	0	53	0	0	32	1000	955
1	1	0.00	2	0.0	0	44	0	0	286	2740	533
2	1	0.10	2	0.0	0	0	0	1	13	159	98
3	1	0.00	1	0.0	0	82	0	0	679	414	651
4	1	0.00	2	0.0	0	0	0	1	6	151	126
...	...	...	...	...	...	...	...	...	...	...	...
691	1	0.29	1	0.0	0	0	0	0	13	114	811
692	1	0.40	1	0.0	0	0	0	0	4	150	164
693	1	0.00	2	0.0	0	0	0	0	3	833	3572
694	0	0.17	1	0.0	0	0	0	0	1	219	1695
695	1	0.44	1	0.0	0	0	0	0	3	39	68

696 rows × 11 columns

In [16]:
```
y
```

Out[16]:
```
0      0
1      0
2      0
3      0
4      0
      ..
691    1
692    1
693    1
694    1
695    1
Name: fake, Length: 696, dtype: int64
```

jupyter project march Last Checkpoint: 03/27/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Connecting to kernel — Trusted — Python 3 (ipykernel)

Splitting

In [17]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.3,stratify=y,random_state=42)
```

In [18]:
```
x_train
```

Out[18]:

	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows
116	1	0.00	1	0.00	0	0	0	0	114	490	1093
28	1	0.00	2	0.00	0	126	1	0	230	2264	130
137	1	0.00	2	0.00	0	0	1	1	1020	464	1039
557	1	0.43	1	0.43	0	0	0	0	0	72	434
26	1	0.00	6	0.00	0	0	0	1	17	536	665
...	...	...	...	...	...	...	...	...	...	...	...
277	1	0.30	2	0.00	0	26	0	1	241	1456	1200
513	0	0.14	1	0.00	0	0	0	0	0	49	2
127	1	0.00	1	0.00	0	0	0	1	571	765	424
352	0	0.00	1	0.00	0	0	0	0	0	51	41
465	0	0.58	1	0.00	0	0	0	0	0	49	22

487 rows × 11 columns

In [19]:
```
y_train
```

Out[19]:
```
116    0
28     0
137    0
557    1
26     0
      ..
277    0
513    1
127    0
352    1
465    1
Name: fake, Length: 487, dtype: int64
```

In [20]:
```
from sklearn.metrics import accuracy_score,classification_report
lr = LogisticRegression()
lr=lr.fit(x_train,y_train)
y_pred = lr.predict(x_test)
acc = accuracy_score(y_test,y_pred)
print('The accuracy obtained by the logistic regression is :')
print(acc)
report = classification_report(y_test,y_pred)
print('The classification report for logistic regression is as follow:')
print(report)
```

```
The accuracy obtained by the logistic regression is :
0.8995215311004785
The classification report for logistic regression is as follow:
              precision    recall  f1-score   support

           0       0.91      0.89      0.90       105
           1       0.89      0.91      0.90       104

    accuracy                           0.90       209
   macro avg       0.90      0.90      0.90       209
weighted avg       0.90      0.90      0.90       209
```

jupyter project march Last Checkpoint: 03/27/2023 (autosaved)

File Edit View Insert Cell Kernel Widgets Help — Connecting to kernel — Trusted — Python 3 (ipykernel)

```
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
  n_iter_i = _check_optimize_result(
```

In [21]:
```
from sklearn.metrics import accuracy_score,classification_report
dt = DecisionTreeClassifier()
dt=dt.fit(x_train,y_train)
y_pred = dt.predict(x_test)
acc = accuracy_score(y_test,y_pred)
print('The accuracy obtained by the Decision Tree Classifier is :')
print(acc)
report = classification_report(y_test,y_pred)
print('The classification report for Decision Tree Classifier is as follow:')
print(report)
```

```
The accuracy obtained by the Decision Tree Classifier is :
0.8947368421052632
The classification report for Decision Tree Classifier is as follow:
              precision    recall  f1-score   support

           0       0.89      0.90      0.90       105
           1       0.90      0.88      0.89       104

    accuracy                           0.89       209
   macro avg       0.89      0.89      0.89       209
weighted avg       0.89      0.89      0.89       209
```

In [27]:
```
from sklearn.metrics import accuracy_score,classification_report
rf = RandomForestClassifier()
rf=rf.fit(x_train,y_train)
y_pred = rf.predict(x_test)
acc = accuracy_score(y_test,y_pred)
print("The accuracy obtained by the Random Forest Classifier is:")
print(acc)
report = classification_report(y_test,y_pred)
print('The classification report for Random Forest Classifier is as follow:')
print(report)
```

```
The accuracy obtained by the Random Forest Classifier is:
0.9138755980861244
The classification report for Random Forest Classifier is as follow:
              precision    recall  f1-score   support

           0       0.91      0.91      0.91       105
           1       0.91      0.91      0.91       104

    accuracy                           0.91       209
   macro avg       0.91      0.91      0.91       209
weighted avg       0.91      0.91      0.91       209
```

C. RESEARCH PAPER :

FAKE ACCOUNT DETECTION ON SOCIAL MEDIA USING RANDOM FORESTCLASSIFIER

Kancharla Venkata Nikhitha[1]
Student
Department of Computer Science
and Engineering,
Sathyabama Institute of Science and
Technology,
Chennai, India.
Venkatanikhitha9963@gmail.com

Karnati Bhavya[2]
Student
Department of Computer Scienceand Engineering,
Sathyabama Institute of Science and Technology,
Chennai, India.
bavya.karnati@gmail.com

Dr. D. Usha Nandini[3]
Associate Professor
Department of Computer Scienceand Engineering,
Sathyabama Institute of Science and Technology,
Chennai, India.
usha.sathyabama.ac.in@gmail.com

ABSTRACT–As the growth of online social networks increases everyone is associated and linked with social media. A massive amount of personal data is being attacked and stolen by cyber attackers. These fraudulent profiles also spread negative publicity, false news, and various malicious programs one of the best ways is to detect these fake accounts, not only the best way but also it is the first step to stopping the negative news and false rumours. There are many efficient ways and techniques to detect these fake accounts. However, these techniques are based on or depend on the account features. In this paper, main aim is to examine three machine learning algorithms that are random forest, logistic regression, and decision tree. Now, progressing to compare the level of efficiency in the three mentioned methodologies and got the highest accuracy for random forest classifier.

Keywords: *Social media, Regression, Fake accounts, Identify theft*

I. INTRODUCTION

In recent times, some of the most well-known social network sites like Instagram, Facebook and Twitter became integral parts of daily life. Social network sites are used by people for e- commerce, entertainment, information and idea sharing keeping in contact with long-lost friends and finding new acquaintances. Users of OSN can post pictures and videos, leave comments and also provide likes to pictures that have been shared.

Fraudulent accounts are one of the downsides of Online Social Networks. We can check an account is fake or not by following parameters, the number of followers, and posts, or the characteristics of the posted information such as the number of likes, views, comments, and shares. By knowing this, many fake accounts are formed and started purchasing comments and increasing the number of followers to become popular. These are particularly created to commit fraudulent acts such as disseminating misleading information and spreading malware. While some are created to gain followers to become popular, spam comments and likes. In social media, one individual creates many profiles with distinct identities, email addresses, and phone numbers. The majority of them are likely fake accounts. There are two sorts of fake accounts: duplicate accounts and fraudulent accounts. Users create multiple accounts as secondary accounts to promote their e-businesses. People and influencers establish duplicate accounts to enhance their businesses and spread useful information. This type of account is not harmful and does not violate social media terms and conditions. Users create fake accounts mainly to spread negativity, false news, hate speech, and impersonate others; these accounts might be considered dangerous and called as false accounts.

This work primarily concentrated on detecting if an Instagram account is phony or real. This detection is based on some parameters such as username, private or

public account, and number of followers, posts, follows and profile pictures. We can detect a user's activity using this attributes.The user’s behaviour on Instagram is determined by a collection of 12 features. Three

machine learning techniques are used to distinguish between genuine and fraudulent accounts.

II. LITERATURE REVIEW

Detection of fake accounts is one of the major issues that should be solved as soon as possible. Though various methods have been already making an impact, these all methods are not completely accurate. Inspired by various research papers on detection, in this whole literature review section, we are going to discuss various applied methodologies and approaches.

Ersahin et.al [1], provided a categorization technique to find Twitter's Phony accounts. And pre-processed our dataset by applying the Entropy Minimization Discretization (EMD) method under supervision to numerical characteristics, and then they examined the output of the Naive Bayes algorithm. By merely pre-processing their dataset using the n discretization strategy on chosen characteristics, they were able to improve the accuracy with Nave Bayes from85.55% to.90.41% for a method for identifying bogus accounts on Twitter. Only by applying the numerical information from provide a safe platform that can forecast and spot fake news in social media networks.

Aditi Gupta et.al [2], used the Facebook Graph API, and collected Facebook user feeds. The intricate privacy controls on Facebook made it extremely difficult to gather data. On a minimal dataset that included their node, their acquaintances in their social neighbourhood, as well as a collection of manually recognized spam accounts, they performed the most widely used supervised machine learning classification approaches. They assessed these classifiers' performance to identify the top classifiers that produced high detection rates and their capacity to identify Phony Facebook accounts. Additionally, they looked into the feature set to identify the characteristics that Facebook fake profiles are most successfully detected by.

Yeh-Cheng Chen et.al [3], introduced a novel, efficient technique for detecting bogus profiles using machine learning. Their method based on account-by-account behavioural analysis evaluate whether an account is genuinely fake or not accurately, as opposed to utilizing graph-based or manually predicted that just cover basic facts. On real-world data, their detection models work admirably, and their findings indicate that the models are not over fitting.

Estee van der walt et.al, [4] employed artificial intelligence to identify Phony identities. The author of this research clarified how to spot false profiles made by both bots and humans. In this study, the fraudulent accounts made on social networks by people and bots were compared. Finally, it can be said that while the features employed to detect the bots were successful, they weren't entirely effective.

Lu Zhang et.al [5], suggested the PSGD, a partially supervised model, to identify scammer groups. As a scammer group detector, a classifier is studied using PU-Learning via PSGD. The suggested PSGD is effective and surpasses cutting-edge spammer detection techniques, according to tests on a real-world data set from Amazon.cn.Bucket online media, they believe it to be a significant and extremely promising outcome.

Sarah Khaled et.al [6], investigated effective methods for spotting bots and Phony accounts on the Twitter network. An innovative technique called SVM-NN is recommended in this to offer an effective detection for such profiles. Even though the recommended approach uses fewer features, it can still be categorized with around ninety- eight percent accuracy. In comparison to the other two classifiers, the newly suggested method performs better in terms of accuracy across all feature sets. The accuracy of the correlation feature set's records is astounding. This is because it chooses the best set of traits rather than combining all features in a linear combination.

Zulfikar Alom et.al [7], investigated the nature of spam users on the Twitter platform to develop an existing spam-detecting mechanism. In this paper, it is designed a new mechanism and a more robust set of features to detect spammers on Twitter. In this Random Forest classifier is used which gives a better result compared to other methodologies. From this, they can plan to build a more effective model which classifies various types of spammers within different social networks.

Naman Singh et.al [8], proposed that on online networking platforms, more bogus accounts are being created for nefarious purposes. They create a forged human profile in order to successfully detect, identify, and remove fake profiles. For bots, ML models employed a variety of factors to determine how many people an account has on each platform's friend list

as followers. It is not possible to contrast between Phony profiles made by humans and cyborgs. By using a data set with Phony profiles and classifying them as fake or real, they can contrast fake and real accounts.

Shivangi Singhal et.al [9], formally introduced

unveil Spot Fake a multi-modal platform for identifying bogus news. Without considering any other subtasks, their suggested approach detects bogus news. It makes use of article textual and graphic components. Text characteristics were learned using language models (such as BERT), and image features were learned using VGG-19 pre-trained on Image Net dataset. Twitter and Weibo, two publicly accessible databases, are used in all of the studies.

Sowmya P et.al [10], one serious issue involves generating duplicate profile users using the data of existing users. A collection of principles that may differentiate real or fake are used to detect Phony profiles. A detection method that can find fraudulent or clone profiles on online networking platforms like Twitter has been presented.

Zeinab Shahbazi et.al [11], proposed an integrated system for multiple block chains and NLP functions to leverage ML models to find false news and better anticipate bogus user accounts and posts. For this workflow, Reinforcement learning methodology is applied. The operation of the decentralized block chain architecture, which offers the framework of digital content authorization proof, improved the security of this platform. The idea behind this approach is to provide a safe platform that can forecast and spot fake news on social media networks.

Latha P et.al [12], stated different categorization techniques have been used in the past to identify Phony online networking media accounts. However, they must improve their ability to spot Phony accounts on these websites. To improve the accuracy rate of identifying bogus accounts, they use ML technologies and NLP in our study. The Random Forest tree classifying algorithm was chosen.

Michael Jonathan Ekosputra et.al [13], study's findings make it evident that ml may be used to detect fake profiles; the models that have been examined in this article include Logistic Regression, Bernoulli Naive Bayes, Random Forest, svm, and ANN. Any changes or additions to features will have an impact on each model's accuracy. According to the research, adding parameters to a model will almost certainly make it more accurate than a model with no parameters or a default form. Based on the results of the research's initial trial, the Random Forest algorithm achieves the best outcome with a startling accuracy of 0.92.

Karishma Anklesaria et.al [14], used different classifiers to classify the profile as fake or real using database of user accounts. This is accomplished by using a methodical approach that includes cleaning, pre-processing, feature selection, and model training. After that, all the employed algorithms—Random Forest, AdaBoost, MLP, SGD, and Artificial neural network—are compared on the basis of different evaluation parameters. According to the study, Random Forest classifier performs better.

Md Mahadi Hassan Sohan et.al [15], showed the importance of comments and how they affect almost every aspect of social media data. People's perspectives are significantly influenced by reviews. Hence, identifying fake reviews is an exciting study area. A method for spotting fake reviews using machine learning was disclosed in the study. It detects both review characteristics and reviewer behaviour. The food review dataset was used to evaluate the proposed method. The technique is semi-supervised, with several classifiers being used. In the fake review detection procedure, the findings reveal that the Random Forest classifier beats another classifier. Moreover, the findings imply that including in viewers' behavioural traits raises the F-score by 55.5% and the overall accuracy to 97.7%.

III. PROPOSED SYSTEM:

In this article, an effective method to detect fake accounts on online social networks is proposed. This work may be regarded as a valuable system since it aids in lowering the restrictions brought on by conventional and other existing methodologies. The goal of this project is to provide an efficient and dependable system for precisely detecting fake. A potent algorithm in a Python-based framework to develop this system is employed. The process is depicted in the picture below.
The steps for executing the proposed work:

8. The required packages should be installed.
9. Establishing the issue association.
10. Create a framework-based UI.
11. Upload CSV file
12. View data
13. Select model
14. Output to predict.

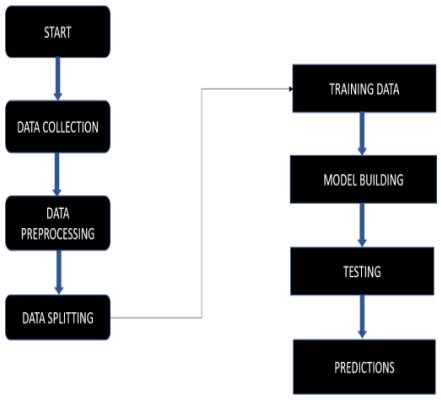


Figure 1: Flow Chart

This proposed work is completely based on Machine Learning approach which detects whether the profile isfake or real one. To do this, three well-known methodologies were compared. This proposed work has two phases, the first phase is frontend – which creates a platform another is backend – a model is created and the main part of the project exists.

The main part of the proposed work is the backend. Here, using flask framework an application is created that detects whether an account is fake or not. In this aspect, the data is collected and preprocessed. Then the data splitting takes place. Now the data gets trained and need to build the model. After the model building with the help of the model data gets tested. And finally the prediction is done.

Firstly, download the dataset from Kaggle which is raw data. The size of the dataset is 697 with 12 attributes such as profile picture, length of the username, full name words, length of the full name, the description provided by the user known as bio, any external URLs provided, private or public accounts, number of posts posted, number of followers and number of accounts the user follows.

profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
1.0	0.27	0.0	0.0	0.0	53.0	0.0	0.0	32.0	1000.0	955.0	0.0
1.0	0.0	2.0	0.0	0.0	44.0	0.0	0.0	286.0	2740.0	533.0	0.0
1.0	0.1	2.0	0.0	0.0	0.0	0.0	1.0	13.0	159.0	98.0	0.0
1.0	0.0	1.0	0.0	0.0	82.0	0.0	0.0	679.0	414.0	651.0	0.0
1.0	0.0	2.0	0.0	0.0	0.0	0.0	1.0	6.0	151.0	126.0	0.0
1.0	0.0	4.0	0.0	0.0	81.0	1.0	0.0	344.0	669987.0	150.0	0.0
1.0	0.0	2.0	0.0	0.0	50.0	0.0	0.0	16.0	122.0	177.0	0.0
1.0	0.0	2.0	0.0	0.0	0.0	0.0	0.0	33.0	1078.0	76.0	0.0
1.0	0.0	0.0	0.0	0.0	71.0	0.0	0.0	72.0	1824.0	2713.0	0.0
1.0	0.0	2.0	0.0	0.0	40.0	1.0	0.0	213.0	12945.0	813.0	0.0
1.0	0.0	2.0	0.0	0.0	54.0	0.0	0.0	648.0	9884.0	1173.0	0.0

Figure 2: Dataset

In the application user first uploads a CSV file of the dataset then the system takes that file and undergoes preprocessing, which means Importing libraries such as pandas, and NumPy. Import dataset, cleaning of null values, categorical values will be turned into numerical values and removing unwanted columns. The null values are replaced by using imputation technique. Then splits the dataset into training and testing set. The target variable or Dependent variable undergoes training and independent variables will go to testing. Next is model building. Model buildings mean choosing a suitable algorithm and train it as per the requirements. Different models like decision tree, Random Forest, and Logistic Regression are considered, and train the models by importing those modules from Sklearn. Then, train the models using the training dataset with fit( ) function. After the model building, apply that model to make predictions and predict the response for test dataset. Then evaluate the model's performance by importing different metrics from sklearn metrics. Random forest got the highest accuracy with 90%. So, this is how the system will do preprocessing and User can view the dataset.

ALGORITHMS USED:

Logistic regression:

Logistic Regression is commonly used machine learning algorithm for binary classification problem, that have two class values and include predictions like "yes" or "no" and "a" or "b". It is used when the dependent variable (target) is categorical.

For example, to predict whether an email is spam (1) or (0) whether the account is fake (1) or not (0).

Decision tree :

Decision tree algorithms are referred as CART or Classification And Regression Techniques. It is utilized in both regression and classification. A decision tree is drawn upside-down with its root at the top. As the name implies, it uses a tree-like model of decision.

Random forest :

A random forest classifier is a machine learning technique which is used for solving regression and classification problems. It makes use of ensemble learning, a technique that combines multiple classifiers to solve complicated problem. Random forest classifier provides an effective way of handling missing data and it is more accurate than the decision tree algorithm and also solves issue of overfitting in decision trees.

IV EXPERIMENTAL RESULTS:

The user can select any model among three models. Then the system will generate accuracy for a particular selected model. Users can also view that generated accuracy, this process will be done on the

model page. The selection of the model is done with the highest accuracy given algorithm. Logistic regression gave 89.9% accuracy, Decision tree got 89.5% accuracy and

Random forest gave the highest accuracy of 90%.

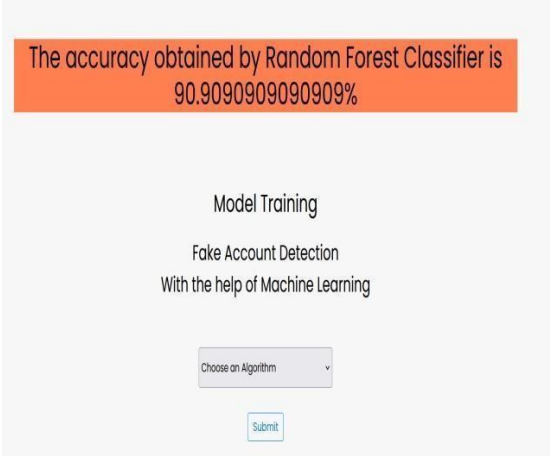


Figure 3: Model Accuracy

The system will generate a graph that compared the accuracy of three models and the user can view that graph. So, select the model and can start the prediction process with the user input we predict whether an account is fraudulent or not.

Figure 4: Prediction

V CONCLUSION:

In this paper, we proposed a machine-learning approach for detecting fake accounts on social media as previously stated, three popular algorithms were compared. After the comparison, random forest algorithm got the best accuracy. Hence, Random Forest classifier algorithm can be considered for the rest of the project.Simply, our project has two phases-frontend and backend. In the backend, did the data pre-processing, model selection, prediction and taking best accuracy given algorithm. Later, in the front end included this code in a framework and with the user input can detect the fake account or not. Hence, concluded that detecting fraudulent profiles depends on the features used in training the data set and classifiers used.

VI REFERENCE:

*[16]*B. Erşahin, Ö. Aktaş, D. Kılınç, and C. Akyol, "Twitter fake account detection," *2017 International Conference on Computer Science and Engineering (UBMK)*, Antalya, Turkey, 2017, pp. 388-392, doi: 10.1109/UBMK.2017.8093420.

[17] A. Gupta and R. Kaushal, "Towards detecting fake user accounts in Facebook," *2017 ISEA Asia Security and Privacy (ISEASP)*, Surat, India, 2017, pp. 1-6, doi: 10.1109/ISEASP.2017.7976996.

[18] Y. -C. Chen and S. F. Wu, "Fake Buster: A Robust Fake Account Detection by Activity Analysis," *2018 9th International Symposium on Parallel Architectures, Algorithms and Programming (PAAP)*, Taipei, Taiwan, 2018, pp. 108-110, doi: 10.1109/PAAP.2018.00026.

[19] E. Van Der Walt and J. Eloff, "Using Machine Learning to Detect Fake Identities: Bots vs Humans," in *IEEE Access*, vol. 6, pp. 6540-6549, 2018, doi: 10.1109/ACCESS.2018.2796018.

[20] L. Zhang, Z. Wu and J. Cao, "Detecting Spammer Groups from Product Reviews: A Partially Supervised Learning Model," in *IEEE Access*, vol. 6, pp. 2559-2568, 2018, doi: 10.1109/ACCESS.2017.2784370.

[21] S. Khaled, N. El-Tazi and H. M. O. Mokhtar, "Detecting Fake Accounts on Social Media," *2018 IEEE International Conference on Big Data (Big Data)*, Seattle, WA, USA, 2018, pp. 3672-3681, doi: 10.1109/BigData.2018.8621913.

[22] Z. Alom, B. Carminati and E. Ferrari, "Detecting Spam Accounts on Twitter," *2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, Barcelona, Spain, 2018, pp. 1191-1198, doi: 10.1109/ASONAM.2018.8508495.

[23] N. Singh, T. Sharma, A. Thakral and

T. Choudhury, "Detection of Fake Profile in Online Social Networks Using Machine Learning," *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*, Paris, France,

2018, pp. 231-234, doi: 10.1109/ICACCE.2018.8441713.

[24] S. Singhal, R. R. Shah, T. Chakraborty, P. Kumaraguru and S. Satoh, "Spot Fake: A Multi-modal Framework for Fake News Detection," *2019 IEEE Fifth International Conference on Multimedia Big Data (BigMM)*, Singapore, 2019, pp. 39-47, doi: 10.1109/BigMM.2019.00-44.

[25] P. Sowmya and M. Chatterjee, "Detection of Fake and Clone accounts in Twitter using Classification and Distance Measure Algorithms," *2020 International Conference on Communication and Signal Processing (ICCSP)*, Chennai, India, 2020, pp. 0067-0070, doi: 10.1109/ICCSP48568.2020.9182353.

[26] Z. Shahbazi and Y. -C. Byun, "Fake Media Detection Based on Natural Language Processing and Blockchain Approaches," in *IEEE Access*, vol. 9, pp. 128442-128453, 2021, doi: 10.1109/ACCESS.2021.3112607.

[27] L. P, S. V, V. Sasikala, J. Arunarasi, A. R. Rajini and N. Nithiya, "Fake Profile Identification in Social Network using Machine Learning and NLP," *2022 International Conference on Communication, Computing and Internet of Things (IC3IoT)*, Chennai, India, 2022, pp. 1-4, doi: 10.1109/IC3IOT53935.2022.9767958.

[28] M. J. Ekosputra, A. Susanto, F. Haryanto and D. Suhartono, "Supervised Machine Learning Algorithms to Detect Instagram Fake Accounts," *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Yogyakarta, Indonesia, 2021, pp. 396-400, doi: 10.1109/ISRITI54043.2021.9702833.

[29] K. Anklesaria, Z. Desai, V. Kulkarni and H. Balasubramaniam, "A Survey on Machine Learning Algorithms for Detecting Fake Instagram Accounts," *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, Greater Noida, India, 2021, pp. 141-144, doi: 10.1109/ICAC3N53548.2021.9725724.

[30] M. M. Hassan Sohan, M. M. Khan, I. Nanda and R. Dey, "Fake Product Review Detection Using Machine Learning," *2022 IEEE World AI IoT Congress (AIIoT)*, Seattle, WA, USA, 2022, pp. 527-532, doi: 10.1109/AIIoT54504.2022.9817271.

[31] Praveena, M.A., Christy, A., Helen, L.S., Krishna, R.S. and Nandini, D.U., 2021, March. Examining and Predicting Helpfulness of reviews based on Naive Bayes. In Journal of Physics: Conference Series (Vol. 1770, No. 1, p. 012021). IOP Publishing.