

IDENTIFICATION OF PHONY PROFILES ON SOCIAL MEDIA PLATFORM USING RANDOM FOREST & DECISION TREE ALGORITHMS

A project report submitted to

**JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY
ANANTAPUR, ANANTHAPURAMU**

In partial fulfillment of the requirements for the award of the degree of
BACHELOR OF TECHNOLOGY

In

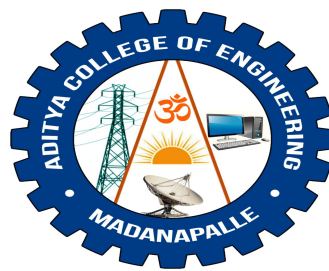
ARTIFICIAL INTELLIGENCE & DATA SCIENCE

by

R NANDINI	-	218P1A3025
G THEJA SREE	-	218P1A3051
U SRI JASHMITHA	-	218P1A3043
A PALLAVI	-	218P1A3027

Under the Guidance of

Mrs. S REDDY MUBARAQ,M.Tech.,
Assistant Professor



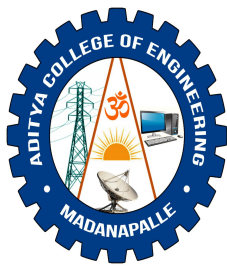
ACCREDITED WITH “A+” GRADE BY NAAC

DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE

ADITYA COLLEGE OF ENGINEERING

MADANAPALLE -517 325, ANNAMAYYA (Dist), A.P.

2024-2025



ADITYA COLLEGE OF ENGINEERING

(Approved by AICTE, New Delhi, Affiliated to JNTUA, Ananthapuramu)

Valasapalli Post, Madanapalle 517325, Annamayya(Dt), A.P

Accredited with “A+” Grade by NAAC



DEPARTMENT OF ARTIFICIAL INTELLIGENCE & DATA SCIENCE

Certificate

This is to certify that Project work entitled **“IDENTIFICATION OF PHONY PROFILES ON SOCIAL MEDIA PLATFORM USING RANDOM FOREST & DECISION TREE ALGORITHMS”** is a bonafide work carried out by **R NANDINI (218P1A3025), G THEJA SREE (218P1A3051), U SRI JASHMITHA (218P1A3043), A PALLAVI (218P1A3027)** in partial fulfillment of the requirements for the award of Degree of Bachelor of Technology in Artificial Intelligence & Data Science of the **Jawaharlal Nehru Technological University Anantapur, Ananthapuramu** during the academic year 2024-25. The project report has been approved as it satisfies the academic requirement in respect of project work prescribed for the award of the Degree of Bachelor of Technology in Artificial Intelligence & Data Science.

Project Guide

Head of the Department

Principal

R NANDINI - 218P1A3025

G THEJA SREE - 218P1A3051

U SRI JASHMITHA - 218P1A3043

A PALLAVI - 218P1A3027

Submitted for the university Viva– Voce Examination held on_____

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We, **R NANDINI , G THEJA SREE , U SRI JASHMITHA , A PALLAVI** here by declare that the Project Work entitled “**IDENTIFICATION OF PHONY PROFILES ON SOCIAL MEDIA PLATFORM USING RANDOM FOREST & DECISION TREE ALOGORITHMS**”, is a bonafide work done by us under the guidance of **Mrs. S REDDY MUBARAQ, M.Tech**, submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Technology in Artificial Intelligence & Data Science, **Aditya College of Engineering, Madanapalle** affiliated to **Jawaharlal Nehru Technological University Anantapur, Ananthapuramu**, during the academic year 2024-25. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Date:

Place:

R NANDINI	- 218P1A3025
G THEJA SREE	- 218P1A3051
U SRI JASHMITHA	- 218P1A3043
A PALLAVI	- 218P1A3027

ACKNOWLEDGEMENT

It is our privilege and pleasure to express our profound sense of respect gratitude and indebtedness to **Dr. S.RAMALINGA REDDY , Director** for guiding and providing facilities for the successful completion of our project work.

It is our privilege and pleasure to express our profound sense of respect gratitude and indebtedness to **Dr. K.SATHISH BABU , Principal** for guiding and providing facilities for the successful completion of our project work.

We sincerely thank to **Dr. R. M.D. SHAFI, Head of the Department, Department of Artificial Intelligence & Data Science**, for his valuable support and constant encouragement given to us during this work.

We express our deep sense of gratitude to **Mrs. S. REDDY MUBARAQ, M.Tech., Project Guide** for supporting and encouraging at each stage of our project work and guided us to do our best.

Last but not least, we wish to acknowledge **Our Parents** and friends for giving moral strength and encouragement.

PROJECT MEMBERS

R NANDINI	- 218P1A3025
G THEJA SREE	- 218P1A3051
U SRI JASHMITHA	- 218P1A3043
A PALLAVI	- 218P1A3027

CONTENTS

TITLE	PAGE NO.
ABSTRACT	i
LIST OF FIGURES	ii
LIST OF TABLES	iii
LIST OF SCREENS	iv
1. INTRODUCTION	1-25
1.1 WHAT IS MACHINE LEARNING	1-2
1.1.1 MACHINE LEARNING VS. TRADITIONAL PROGRAMMING	2
1.2 HOW DOES MACHINE LEARNING WORK?	3-4
1.2.1 MACHINE LEARNING ALGORITHM & WHERE THEY ARE USED	5-6
1.3 MACHINE LEARNING ALGORITHM	6
1.3.1 CHALLENGES & LIMITATIONS OF MACHINE LEARNING	6-7
1.4 WHY IS MACHINE LEARNING IMPORTANT?	7
1.5 APPLICATIONS OF MACHINE LEARNING	7-8
1.5.1 EXAMPLE OF MACHINE LEARNING GOOGLE CAR	8-9
1.6 MACHINE LEARNING APPROACHES	9
1.7 BACKGROUND	9-10
1.7.1 ARTIFICIAL INTELLIGENCE(AI)	10-12
1.8 TYPES OF LEARNING ALGORITHMS	12-17
1.9 MODELS	18
1.9.1 ARTIFICIAL NEURAL NETWORKS	18

1.9.2 DECISION TREES	19
1.9.3 SUPPORT VECTOR MACHINES	19
1.9.4 REGRESSION ANALYSIS	19
1.9.5 BAYESIAN NETWORKS	19-20
1.9.6 GENETIC ALGORITHMS	20
1.10 EXISTING SYSTEM	20-21
1.10.1 DISADVANTAGES OF EXISTING SYSTEM	21-22
1.11 PROPOSED SYSTEM	22-23
1.11.1 ADVANTAGES OF PROPOSED SYSTEM	23-24
1.12 SYSTEM ARCHITECTURE	25
2. LITERATURE REVIEW	26-29
3. SYSTEM ANALYSIS	30-31
3.1 INTRODUCTION	30
3.2 SOFTWARE REQUIREMENTS	30
3.3 HARDWARE REQUIREMENTS	31
4. SYSTEM STUDY	32-36
4.1 FEASIBILITY STUDY	32
4.1.1 INTRODUCTION TO FEASIBILITY STUDY	32
4.2 ECONOMICAL FEASIBILITY	32
4.2.1 IMPORTANCE OF ECONOMIC FEASIBILITY	33
4.3 TECHNICAL FEASIBILITY	33
4.3.1 IMPORTANCE OF TECHNICAL FEASIBILITY	33
4.4 KEY COMPONENTS OF TECHNICAL FEASIBILITY	33-34
4.5 SOCIAL FEASIBILITY	35
4.5.1 IMPORTANCE OF SOCIAL FEASIBILITY	35
4.6 KEY COMPONENTS OF SOCIAL FEASIBILITY	35-36

5. SYSTEM DESIGN	37-42
5.1 INTRODUCTION	37
5.2 DATA FLOW DIAGRAM	37-38
5.3 UML DIAGRAMS	38-39
5.3.1 GOALS	39
5.4 USECASE DIAGRAM	39
5.5 CLASS DIAGRAM	40
5.6 SEQUENCE DIAGRAM	40-41
5.7 ACTIVITY DIAGRAM	41-42
6. IMPLEMENTATION	43-49
6.1 MODULES	43
6.1.1 MODULE DESCRIPTION	43-44
6.1.2 RANDOM FOREST CLASSIFIER	44-46
6.1.3 DECISION TREE CLASSIFIER	46-47
6.2 WHY USE DECISION TREES?	47
6.2.1 HOW DOES DECISION TREE ALGORITHM WORK	47-49
7. SYSTEM TESTING	50-55
7.1 INTRODUCTION	50
7.1.1 IMPORTANCE OF SYSTEM TESTING	50
7.2 TYPES OF TESTS	50
7.2.1 UNIT TESTING	50-51
7.2.2 INTEGRATION TESTING	51
7.2.3 FUNCTIONAL TESTING	51-52
7.2.4 SYSTEM TETSING	52
7.2.5 WHITE BOX TESTING	52-53
7.3 KEY OBJECTIVES OF WHITE BOX TESTING	53
7.4 BLACK BOX TETSING	53-55
8. OUTPUT SCREENS	56-66

9. CONCLUSION & FUTURE SCOPE	67-68
9.1 CONCLUSION	67
9.2 FUTURE SCOPE	67-68
10. REFERENCES	69

ABSTRACT

In an age where social media has become an integral part of our lives, the challenge of detecting fake accounts on platforms like Instagram has gained significant importance. This project, titled "Instagram Fake Account Detection using Machine Learning," employs Python as its primary tool to tackle this problem. It leverages two powerful machine learning algorithms, the Random Forest Classifier and the Decision Tree Classifier, to accomplish this task. The Random Forest Classifier demonstrates remarkable performance, achieving a 100% accuracy on the training data set and an impressive 93% accuracy on the test data set. Meanwhile, the Decision Tree Classifier exhibits its effectiveness with a training accuracy of 92% and a test accuracy of 92%. The data set employed in this project is composed of 576 records, each characterized by 12 distinct features. These features encompass critical aspects of Instagram profiles, including the presence of a profile picture, the ratio of numerical characters in usernames, the breakdown of full names into word tokens, the ratio of numerical characters in full names, the equality between usernames and full names, the length of user bios, the existence of external URLs, the privacy status of accounts, the number of posts, the count of followers, the number of accounts followed, and the ultimate classification of an account as "Fake" or "Not." By harnessing the capabilities of Python and these advanced machine learning models, this project endeavors to provide a robust and efficient solution for the identification of fake Instagram accounts. In doing so, it contributes to the preservation of the platform's integrity and the security of its users.

Keywords: Python, Decision tree Algorithm, Random Forest Algorithm.

LIST OF FIGURES

S NO.	FIGURE NO.	TITLE	PAGE NO.
1	1.1	Traditional Programming	2
2	1.2	Machine Learning	2
3	1.3	Learning Phase	3
4	1.4	Inference Model	4
5	1.5	Machine Learning Algorithm	5
6	1.6	Machine Learning Algorithm	6
7	1.7	System Architecture	25
8	5.1	Data Flow Diagram	38
9	5.2	Use Case Diagram	39
10	5.3	Class Diagram	40
11	5.4	Sequence Diagram	41
12	5.5	Activity Diagram	42
13	6.1	Structure of Decision Tree	47
14	6.2	Decision Tree Working Example	48

LIST OF TABLES

S.NO.	TABLE NO.	TITLE	PAGE NO.
1	2.1	Literature reviw	29
2	3.1	Software Requirements	30
3	3.2	Hardware Requirements	31

LIST OF SCREENS

S. NO.	SCREEN NO.	TITLE.	PAGE NO.
1	8.1	Home Screen	56
2	8.2	Login Page	57
3	8.3	Upload csv	58
4	8.4	Train & Test data	59
5	8.5	Prediction using Random Forest	60
6	8.6	Prediction Result	61
7	8.7	Input Interface	62
8	8.8	Prediction Page	63
9	8.9	Prediction Page	64
10	8.10	Performance Analysis	65
11	8.11	Accuracy	66

1. INTRODUCTION

The proliferation of social media platforms has led to an increase in the number of phony profiles, which can be used for malicious activities such as spreading misinformation, phishing, and cyberbullying. Identifying these phony profiles is crucial to maintaining the integrity and security of social media platforms. This project proposes the use of machine learning techniques, specifically Random Forest and Decision Trees, to identify phony profiles on social media platforms. Phony profiles on social media can be categorized into two types: fake profiles and compromised profiles. Fake profiles are created with the intention of deceiving others, while compromised profiles are legitimate accounts that have been hacked or taken over by malicious actors. Identifying these profiles is a challenging task, as they often mimic the behavior of legitimate users. The objective of this project is to develop a machine learning-based approach for identifying phony profiles on social media platforms using Random Forest and Decision Trees. The Rise of Phony Profiles on Social Media Social media platforms have become an integral part of modern life, with billions of users worldwide. However, the anonymity and ease of creating profiles on these platforms have also made them vulnerable to abuse. Phony profiles, created with malicious intent, pose a significant threat to the integrity and security of social media platforms. The Need for Detection Given the potential harm caused by phony profiles, it is essential to develop effective methods for detecting and removing them. Traditional methods, such as manual review and rule-based systems, are often inadequate, highlighting the need for more sophisticated approaches. The identification of phony profiles on social media platforms is a challenging task. Phony profiles can be designed to mimic the behavior of legitimate users, making it difficult to distinguish between the two. Traditional methods of identifying phony profiles, such as manual review and rule-based systems, are often time-consuming and ineffective. This project proposes the use of machine learning techniques, specifically Random Forest and Decision Trees, to identify phony profiles on social media platforms.

1.1 WHAT IS MACHINE LEARNING?

Machine Learning is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial Intelligence which combines data with statistical tools to predict an output which can be used to make actionable insights. Machine learning is also used for a variety of tasks like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input and uses an algorithm to formulate answers.

A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

1.1.1 MACHINE LEARNING VS. TRADITIONAL PROGRAMMING

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

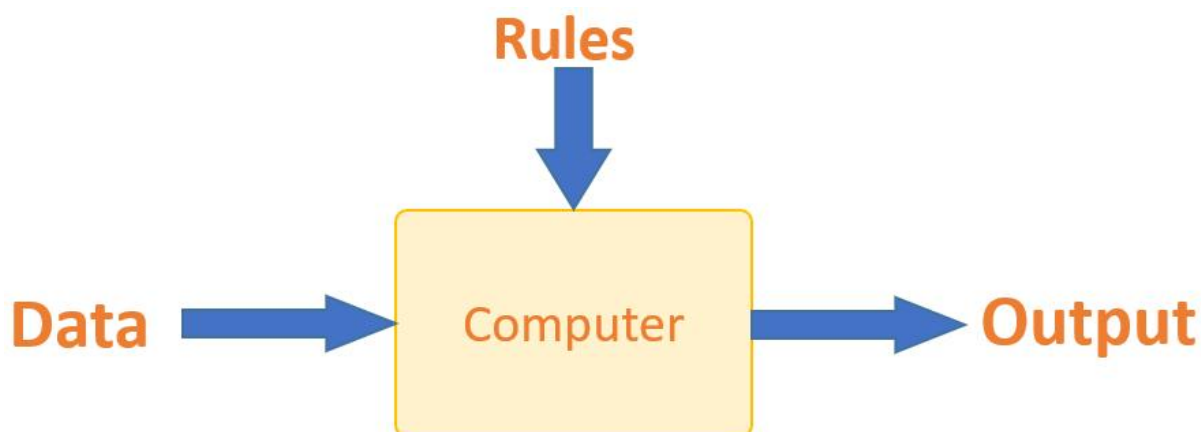


Fig 1.1 Traditional Programming

Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.

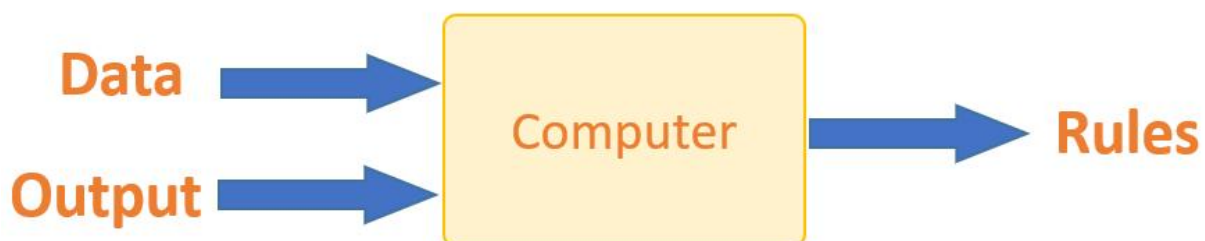


Fig 1.2 Machine Learning

1.2 HOW DOES MACHINE LEARNING WORK?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict. The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the **data**. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector**. You can think of a feature vector as a subset of data that is used to tackle a problem. The machine uses some fancy algorithms to simplify the reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.

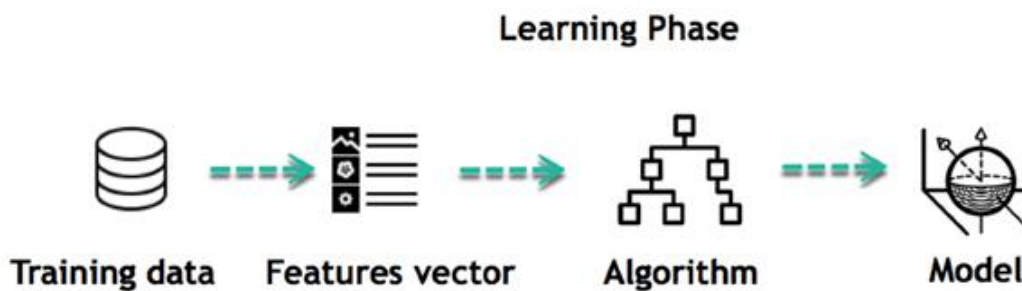


Fig 1.3 Learning Phase

For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

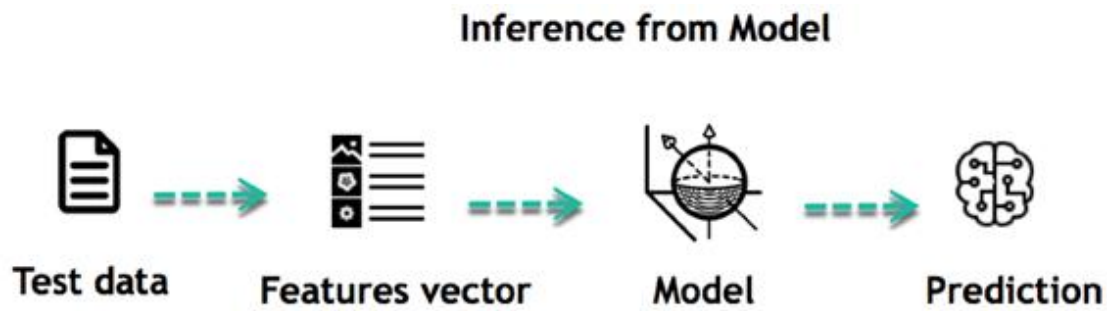


Fig 1.4 Inference Model

The life of Machine Learning programs is straightforward and can be summarized in the following points:

- Define a question
- Collect data
- Visualize data
- Train algorithm
- Test the Algorithm
- Collect feedback
- Refine the algorithm
- Loop 4-7 until the results are satisfying
- Use the model to make a prediction

1.2.2 MACHINE LEARNING ALGORITHMS AND WHERE THEY ARE USED?

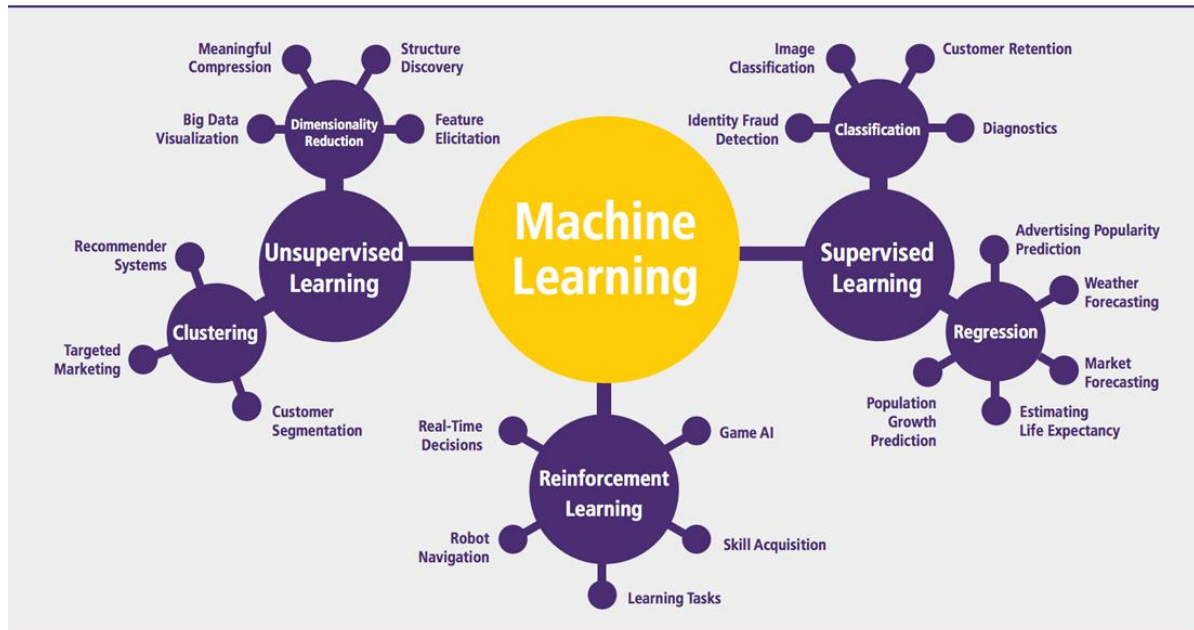


Fig 1.5 Machine learning Algorithms

Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans. You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

- Classification task
- Regression task

Classification

Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database. You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.

The label can be of two or more classes. The above Machine learning example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns).

1.3 MACHINE LEARNING ALGORITHM:

There are plenty of machine learning algorithms. The choice of the algorithm is based on the objective. In the Machine learning example below, the task is to predict the type of flower among the three varieties. The predictions are based on the length and the width of the petal. The picture depicts the results of ten different algorithms. The picture on the top left is the data set. The data is classified into three categories: red, light blue and dark blue. There are some groupings. For instance, from the second image, everything in the upper left belongs to the red category, in the middle part, there is a mixture of uncertainty and light blue while the bottom corresponds to the dark category. The other images show different algorithms and how they try to classified the data.

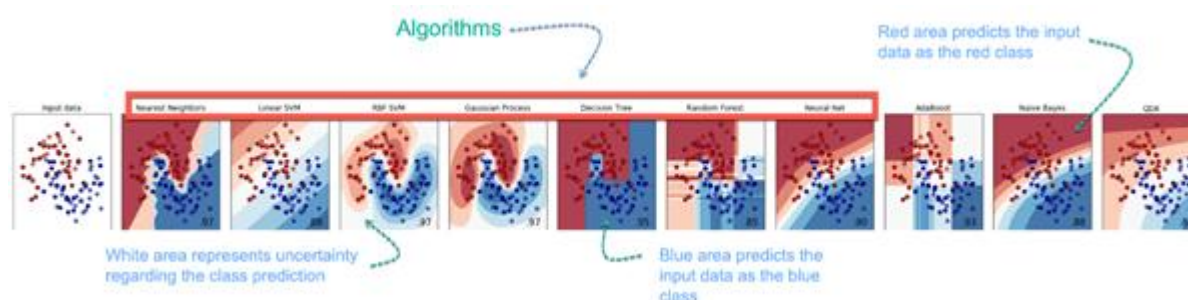


Fig 1.6 Machine Learning Algorithm

1.3.1 Challenges and Limitations of ML

The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

1.4 WHY IS MACHINE LEARNING IMPORTANT?

Machine learning is the best tool so far to analyze, understand and identify a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can take decisions with minimal human intervention.

Take the following example for this ML tutorial; a retail agent can estimate the price of a house based on his own experience and his knowledge of the market.

A machine can be trained to translate the knowledge of an expert into features. The features are all the characteristics of a house, neighborhood, economic environment, etc. that make the price difference. For the expert, it took him probably some years to master the art of estimate the price of a house. His expertise is getting better and better after each sale.

For the machine, it takes millions of data, (i.e., example) to knowledge to make its estimation. At the same time, with incredible accuracy master this art. At the very beginning of its learning, the machine makes a mistake, somehow like the junior salesman. Once the machine sees all the example, it got enough. The machine is also able to adjust its mistake accordingly.

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers; but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

1.5 APPLICATIONS OF MACHINE LEARNING

The various Applications of Machine Learning are:

Augmentation:

- Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

- Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry:

- Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization:

- The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition.

Healthcare industry:

- Healthcare was one of the first industry to use machine learning with image detection.

Marketing:

- Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

Example of application of Machine Learning in Supply Chain:

Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network. For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report and make recommendations in real-time.

1.5.1 EXAMPLE OF MACHINE LEARNING GOOGLE CAR

For example, everybody knows the Google car. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

Overview

Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. For simple tasks assigned to computers, it is possible to program algorithms telling the machine how to execute all steps required to solve the problem at hand; on the computer's part, no learning is needed. For more advanced tasks, it can be challenging for a human to manually create the needed

algorithms. In practice, it can turn out to be more effective to help the machine develop its own algorithm, rather than having human programmers specify every needed step.

The discipline of machine learning employs various approaches to teach computers to accomplish tasks where no fully satisfactory algorithm is available. In cases where vast numbers of potential answers exist, one approach is to label some of the correct answers as valid. This can then be used as training data for the computer to improve the algorithm(s) it uses to determine correct answers. For example, to train a system for the task of digital character recognition, the MNIST data set of handwritten digits has often been used.

1.6 MACHINE LEARNING APPROACHES

Machine learning approaches are traditionally divided into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system:

Supervised Learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.

Un-supervised Learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).

Reinforcement Learning: A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that's analogous to rewards.

1.7 BACKGROUND

The term machine learning was coined in 1959 by Arthur Samuel, an American IBMer and pioneer in the field of computer gaming and artificial intelligence. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T , as measured by P , improves with experience E ." This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?".

Modern day machine learning has two objectives, one is to classify data based on models which have been developed, the other purpose is to make predictions for future outcomes based on these models. A

hypothetical algorithm specific to classifying data may use computer vision of moles coupled with supervised learning in order to train it to classify the cancerous moles. Where as, a machine learning algorithm for stock trading may inform the trader of future potential predictions.

1.7.1 Artificial Intelligence(AI)

Machine Learning as subfield of Artificial Intelligence

Part of Machine Learning as subfield of AI or part of AI as subfield of Machine Learning

As a scientific endeavor, machine learning grew out of the quest for artificial intelligence. In the early days of AI as an academic discipline, some researchers were interested in having machines learn from data. They attempted to approach the problem with various symbolic methods, as well as what was then termed "neural networks"; these were mostly perceptrons and other models that were later found to be reinventions of the generalized linear models of statistics. Probabilistic reasoning was also employed, especially in automated medical diagnosis.

However, an increasing emphasis on the logical, knowledge-based approach caused a rift between AI and machine learning. Probabilistic systems were plagued by theoretical and practical problems of data acquisition and representation. By 1980, expert systems had come to dominate AI, and statistics was out of favor. Work on symbolic/knowledge-based learning did continue within AI, leading to inductive logic programming, but the more statistical line of research was now outside the field of AI proper, in pattern recognition and information retrieval. Neural networks research had been abandoned by AI and computer science around the same time. This line, too, was continued outside the AI/CS field, as "connectionism", by researchers from other disciplines including Hopfield, Rumelhart and Hinton. Their main success came in the mid-1980s with the reinvention of backpropagation.

Machine learning (ML), reorganized as a separate field, started to flourish in the 1990s. The field changed its goal from achieving artificial intelligence to tackling solvable problems of a practical nature. It shifted focus away from the symbolic approaches it had inherited from AI, and toward methods and models borrowed from statistics and probability theory.

As of 2020, many sources continue to assert that machine learning remains a subfield of AI. The main disagreement is whether all of ML is part of AI, as this would mean that anyone using ML could claim they are using AI. Others have the view that not all of ML is part of AI where only an 'intelligent' subset of ML is part of AI.

The question to what is the difference between ML and AI is answered by Judea Pearl in The Book of Why. Accordingly ML learns and predicts based on passive observations, whereas AI implies an agent interacting with the environment to learn and take actions that maximize its chance of successfully achieving its goals.

Data mining

Machine learning and data mining often employ the same methods and overlap significantly, but while machine learning focuses on prediction, based on known properties learned from the training data, data mining focuses on the discovery of (previously) unknown properties in the data (this is the analysis step of knowledge discovery in databases). Data mining uses many machine learning methods, but with different goals; on the other hand, machine learning also employs data mining methods as "unsupervised learning" or as a preprocessing step to improve learner accuracy. Much of the confusion between these two research communities (which do often have separate conferences and separate journals, ECML PKDD being a major exception) comes from the basic assumptions they work with: in machine learning, performance is usually evaluated with respect to the ability to reproduce known knowledge, while in knowledge discovery and data mining (KDD) the key task is the discovery of previously unknown knowledge. Evaluated with respect to known knowledge, an uninformed (unsupervised) method will easily be outperformed by other supervised methods, while in a typical KDD task, supervised methods cannot be used due to the unavailability of training data.

Optimization

Machine learning also has intimate ties to optimization: many learning problems are formulated as minimization of some loss function on a training set of examples. Loss functions express the discrepancy between the predictions of the model being trained and the actual problem instances (for example, in classification, one wants to assign a label to instances, and models are trained to correctly predict the pre-assigned labels of a set of examples).

Generalization

The difference between optimization and machine learning arises from the goal of generalization: while optimization algorithms can minimize the loss on a training set, machine learning is concerned with minimizing the loss on unseen samples. Characterizing the generalization of various learning algorithms is an active topic of current research, especially for deep learning algorithms.

Statistics

Machine learning and statistics are closely related fields in terms of methods, but distinct in their principal goal: statistics draws population inferences from a sample, while machine learning finds generalizable predictive patterns. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long pre-history in statistics. He also suggested the term data science as a placeholder to call the overall field.

Leo Breiman distinguished two statistical modeling paradigms: data model and algorithmic model, wherein "algorithmic model" means more or less the machine learning algorithms like Random forest.

Some statisticians have adopted methods from machine learning, leading to a combined field that they call statistical learning.

Theory

A core objective of a learner is to generalize from its experience. Generalization in this context is the ability of a learning machine to perform accurately on new, unseen examples/tasks after having experienced a learning data set. The training examples come from some generally unknown probability distribution and the learner has to build a general model about this space that enables it to produce sufficiently accurate predictions in new cases.

The computational analysis of machine learning algorithms and their performance is a branch of theoretical computer science known as computational learning theory. Because training sets are finite and the future is uncertain, learning theory usually does not yield guarantees of the performance of algorithms. Instead, probabilistic bounds on the performance are quite common. The bias–variance decomposition is one way to quantify generalization error.

For the best performance in the context of generalization, the complexity of the hypothesis should match the complexity of the function underlying the data. If the hypothesis is less complex than the function, then the model has under fitted the data. If the complexity of the model is increased in response, then the training error decreases. But if the hypothesis is too complex, then the model is subject to overfitting and generalization will be poorer.

In addition to performance bounds, learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned in polynomial time. Negative results show that certain classes cannot be learned in polynomial time.

1.8 TYPES OF LEARNING ALGORITHMS

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

Supervised learning

A support vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white.

Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a

feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Types of supervised learning algorithms include active learning, classification and regression. Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. As an example, for a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email.

Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

Unsupervised learning

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated, for example, by internal compactness, or the similarity between members of the same cluster, and separation, the difference between clusters. Other methods are based on estimated density and graph connectivity.

Semi-supervised learning

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning. Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

Reinforcement learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

Self learning

Self-learning as a machine learning paradigm was introduced in 1982 along with a neural network capable of self-learning named crossbar adaptive array (CAA). It is a learning with no external rewards and no external teacher advice. The CAA self-learning algorithm computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about consequence situations. The system is driven by the interaction between cognition and emotion. The self-learning algorithm updates a memory matrix $W = \|w(a,s)\|$ such that in each iteration executes the following machine learning routine:

In situation s perform an action a ;

Receive consequence situation s' ;

Compute emotion of being in consequence situation $v(s')$;

Update crossbar memory $w'(a,s) = w(a,s) + v(s')$.

It is a system with only one input, situation s , and only one output, action (or behavior) a . There is neither a separate reinforcement input nor an advice input from the environment. The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is the behavioral environment where it behaves, and the other is the genetic environment, wherefrom it initially and only once receives initial emotions about situations to be encountered in the behavioral environment. After receiving the genome (species) vector from the genetic environment, the CAA learns a goal-seeking behavior, in an environment that contains both desirable and undesirable situations.

Feature learning

Several learning algorithms aim at discovering better representations of the inputs provided during training. Classic examples include principal components analysis and cluster analysis. Feature learning algorithms, also called representation learning algorithms, often attempt to preserve the information in their input but also transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions. This technique allows reconstruction of the inputs coming from the unknown data-generating distribution, while not being necessarily faithful to configurations that are implausible under that distribution. This replaces manual feature engineering, and allows a machine to both learn the features and use them to perform a specific task.

Feature learning can be either supervised or unsupervised. In supervised feature learning, features are learned using labeled input data. Examples include artificial neural networks, multilayer perceptrons, and supervised dictionary learning. In unsupervised feature learning, features are learned with unlabeled input data. Examples include dictionary learning, independent component analysis, autoencoders, matrix factorization and various forms of clustering.

Manifold learning algorithms attempt to do so under the constraint that the learned representation is low-dimensional. Sparse coding algorithms attempt to do so under the constraint that the learned representation is sparse, meaning that the mathematical model has many zeros. Multilinear subspace learning algorithms aim to learn low-dimensional representations directly from tensor representations for multidimensional data, without reshaping them into higher-dimensional vectors. Deep learning algorithms discover multiple levels of representation, or a hierarchy of features, with higher-level, more abstract features defined in terms of (or generating) lower-level features. It has been argued that an intelligent machine is one that learns a representation that disentangles the underlying factors of variation that explain the observed data.

Feature learning is motivated by the fact that machine learning tasks such as classification often require input that is mathematically and computationally convenient to process. However, real-world data such as images, video, and sensory data has not yielded to attempts to algorithmically define specific features. An alternative is to discover such features or representations thorough examination, without relying on explicit algorithms.

Sparse dictionary learning

Sparse dictionary learning is a feature learning method where a training example is represented as a linear combination of basis functions, and is assumed to be a sparse matrix. The method is strongly NP-hard and difficult to solve approximately. A popular heuristic method for sparse dictionary learning is the K-SVD algorithm. Sparse dictionary learning has been applied in several contexts. In classification, the problem is to

determine the class to which a previously unseen training example belongs. For a dictionary where each class has already been built, a new training example is associated with the class that is best sparsely represented by the corresponding dictionary. Sparse dictionary learning has also been applied in image denoising. The key idea is that a clean image patch can be sparsely represented by an image dictionary, but the noise cannot.

Anomaly detection

In data mining, anomaly detection, also known as outlier detection, is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data. Typically, the anomalous items represent an issue such as bank fraud, a structural defect, medical problems or errors in a text. Anomalies are referred to as outliers, novelties, noise, deviations and exceptions.

In particular, in the context of abuse and network intrusion detection, the interesting objects are often not rare objects, but unexpected bursts of inactivity. This pattern does not adhere to the common statistical definition of an outlier as a rare object, and many outlier detection methods (in particular, unsupervised algorithms) will fail on such data unless it has been aggregated appropriately. Instead, a cluster analysis algorithm may be able to detect the micro-clusters formed by these patterns.

Three broad categories of anomaly detection techniques exist. Unsupervised anomaly detection techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal, by looking for instances that seem to fit least to the remainder of the data set. Supervised anomaly detection techniques require a data set that has been labeled as "normal" and "abnormal" and involves training a classifier (the key difference to many other statistical classification problems is the inherently unbalanced nature of outlier detection). Semi-supervised anomaly detection techniques construct a model representing normal behavior from a given normal training data set and then test the likelihood of a test instance to be generated by the model.

Robot learning

In developmental robotics, robot learning algorithms generate their own sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies and imitation.

Association rules

Association rule learning is a rule-based machine learning method for discovering relationships between variables in large databases. It is intended to identify strong rules discovered in databases using some measure of "interestingness".

Rule-based machine learning is a general term for any machine learning method that identifies, learns, or evolves "rules" to store, manipulate or apply knowledge. The defining characteristic of a rule-based machine learning algorithm is the identification and utilization of a set of relational rules that collectively represent the knowledge captured by the system. This is in contrast to other machine learning algorithms that commonly identify a singular model that can be universally applied to any instance in order to make a prediction. Rule-based machine learning approaches include learning classifier systems, association rule learning, and artificial immune systems.

Based on the concept of strong rules, Rakesh Agrawal, Tomasz Imieliński and Arun Swami introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule $\{\text{onions, potatoes}\} \rightarrow \{\text{burger}\}$ found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, they are likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as promotional pricing or product placements. In addition to market basket analysis, association rules are employed today in application areas including Web usage mining, intrusion detection, continuous production, and bioinformatics. In contrast with sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

Learning classifier systems (LCS) are a family of rule-based machine learning algorithms that combine a discovery component, typically a genetic algorithm, with a learning component, performing either supervised learning, reinforcement learning, or unsupervised learning.

Inductive logic programming (ILP) is an approach to rule-learning using logic programming as a uniform representation for input examples, background knowledge, and hypotheses. Given an encoding of the known background knowledge and a set of examples represented as a logical database of facts, an ILP system will derive a hypothesized logic program that entails all positive and no negative examples. Inductive programming is a related field that considers any kind of programming language for representing hypotheses (and not only logic programming), such as functional programs. Inductive logic programming is particularly useful in bioinformatics and natural language processing. Gordon Plotkin and Ehud Shapiro laid the initial theoretical foundation for inductive machine learning in a logical setting. Shapiro built their first implementation (Model Inference System) in 1981: a Prolog program that inductively inferred logic programs from positive and negative examples. The term inductive here refers to philosophical induction, suggesting a theory to explain observed facts, rather than mathematical induction, proving a property for all members of a well-ordered set.

1.9 MODELS

Performing machine learning involves creating a model, which is trained on some training data and then can process additional data to make predictions. Various types of models have been used and researched for machine learning systems.

1.9.1 ARTIFICIAL NEURAL NETWORKS

An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another.

Artificial neural networks (ANNs), or connectionist systems, are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn" to perform tasks by considering examples, generally without being programmed with any task-specific rules. An ANN is a model based on a collection of connected units or nodes called "artificial neurons", which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called "edges". Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis.

Deep learning consists of multiple hidden layers in an artificial neural network. This approach tries to model the way the human brain processes light and sound into vision and hearing. Some successful applications of deep learning are computer vision and speech recognition.

1.9.2 DECISION TREES

Decision tree learning uses a decision tree as a predictive model to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modeling approaches used in statistics, data mining, and machine learning. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data, but the resulting classification tree can be an input for decision making.

1.9.3 SUPPORT VECTOR MACHINES

Support vector machines, also known as support vector networks, are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. An SVM training algorithm is a non-probabilistic, binary, linear classifier, although methods such as Platt scaling exist to use SVM in a probabilistic classification setting. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, mapping their inputs into high-dimensional feature spaces.

1.9.4 REGRESSION ANALYSIS

Regression analysis encompasses a large variety of statistical methods to estimate the relationship between input variables and their associated features. Its most common form is linear regression, where a single line is drawn to best fit the given data according to a mathematical criterion such as ordinary least squares. The latter is often extended by regularization (mathematics) methods to mitigate overfitting and bias, as in ridge regression. When dealing with non-linear problems, go-to models include polynomial regression (for example, used for trendline fitting in Microsoft Excel, logistic regression (often used in statistical classification) or even kernel regression, which introduces non-linearity by taking advantage of the kernel trick to implicitly map input variables to higher-dimensional space.

1.9.5 BAYESIAN NETWORKS

A simple Bayesian network. Rain influences whether the sprinkler is activated, and both rain and the sprinkler influence whether the grass is wet.

A Bayesian network, belief network, or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independence with a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between

diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases. Efficient algorithms exist that perform inference and learning. Bayesian networks that model sequences of variables, like speech signals or protein sequences, are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

1.9.6 GENETIC ALGORITHMS

A genetic algorithm (GA) is a search algorithm and heuristic technique that mimics the process of natural selection, using methods such as mutation and crossover to generate new genotypes in the hope of finding good solutions to a given problem. In machine learning, genetic algorithms were used in the 1980s and 1990s. Conversely, machine learning techniques have been used to improve the performance of genetic and evolutionary algorithms.

Training models

Usually, machine learning models require a lot of data in order for them to perform well. Usually, when training a machine learning model, one needs to collect a large, representative sample of data from a training set. Data from the training set can be as varied as a corpus of text, a collection of images, and data collected from individual users of a service. Overfitting is something to watch out for when training a machine learning model. Trained models derived from biased data can result in skewed or undesired predictions. Algorithmic bias is a potential result from data not fully prepared for training.

Federated learning

Federated learning is an adapted form of distributed artificial intelligence to training machine learning models that decentralizes the training process, allowing for users' privacy to be maintained by not needing to send their data to a centralized server. This also increases efficiency by decentralizing the training process to many devices. For example, Gboard uses federated machine learning to train search query prediction models on users' mobile phones without having to send individual searches back to Google

1.10 EXISTING SYSTEM

- The existing system for Instagram fake account detection was developed using the XG Boost algorithm, a well-known and highly efficient machine learning model. The XG Boost algorithm is renowned for its ability to handle complex datasets and perform exceptionally well in classification tasks, making it a suitable choice for this specific application.
- In the existing system, a dataset of Instagram profiles with associated features was used for training and testing the XG Boost model. The features in the dataset were carefully selected to capture key attributes of user profiles, which are indicative of whether an account is genuine or fake.
- XG Boost, an ensemble learning algorithm, excels in enhancing predictive accuracy by combining

the predictions of multiple decision trees. This approach allows the model to capture complex patterns and relationships in the data, enabling it to make highly accurate predictions about the authenticity of Instagram accounts.

- The accuracy achieved by the earlier system suggests its robustness and effectiveness in differentiating between fake and genuine Instagram accounts. This level of accuracy is crucial for maintaining the trust and security of the Instagram platform, as it helps in identifying and mitigating the presence of fake accounts, which can be associated with various malicious activities.
- Overall, the earlier system's use of the XGBoost algorithm and its exceptional accuracy rate highlight its capability to address the challenge of Instagram fake account detection with precision and efficiency.

1.10.1 DISADVANTAGES OF EXISTING SYSTEM

- **Limited Explanation of Predictions:** The XG Boost algorithm, while highly accurate, is often considered a "black box" model, making it challenging to provide detailed explanations for its predictions. This lack of transparency can be a disadvantage when users or administrators need to understand why a particular account was flagged as fake.
- **Sensitivity to Imbalanced Datasets:** Like many machine learning algorithms, XGBoost can be sensitive to imbalanced datasets. If there is a significant disparity between the number of fake and genuine accounts in the dataset, it may lead to biased predictions and less reliable results.
- **Dependence on Feature Engineering:** Achieving high accuracy with XGBoost often depends on the quality of feature engineering. The selection and engineering of relevant features require domain expertise and can be time-consuming.
- **Limited Adaptability:** The existing system may struggle to adapt to emerging trends or new techniques used by malicious actors to create fake Instagram accounts. Since XGBoost is a static model, it may not easily incorporate new information or adapt to evolving threats.
- **Computational Resource Intensiveness:** XGBoost can be computationally intensive, especially for large datasets. This can lead to longer training and inference times, which may not be suitable for real-time or near-real-time detection requirements.
- **Potential Overfitting:** While the system achieved a high accuracy of 96.29%, there is a risk of overfitting, where the model may perform exceptionally well on the training data but struggle with generalization to unseen data. Overfitting can lead to false positives and false negatives in real-world scenarios.
- **Dependency on Data Quality:** The accuracy and performance of the system are heavily reliant on the quality of the training data. Inaccurate or incomplete data can result in suboptimal performance and

may require continuous efforts to maintain data quality. Inability to Address Textual Content: The existing system's focus on numerical and structured features may limit its ability to detect fake accounts that primarily engage in posting deceptive or harmful content through text, such as fake news or hate speech.

- **Lack of Multi-Modal Analysis:** Instagram includes various types of content, including images and videos. The system's sole reliance on structured data may overlook fake accounts that use images or other non-textual content for deceptive purposes.
- **Privacy Concerns:** The system's high accuracy in identifying fake accounts may raise privacy concerns, as it could inadvertently flag genuine users as fake based on certain behaviors or characteristics, potentially leading to user dissatisfaction or mistrust.

1.11 PROPOSED SYSTEM

- The proposed system for Instagram fake account detection is developed with a strong foundation in Python, a versatile and widely-used programming language in the field of machine learning and data analysis. The system leverages two key machine learning models, the Random Forest Classifier and the Decision Tree Classifier, to enhance its performance in distinguishing genuine and fake Instagram accounts.
- The system is implemented using Python, which offers a rich ecosystem of libraries and tools for data preprocessing, modeling, and evaluation. Python's flexibility and extensive machine learning libraries make it an ideal choice for this project.
- The proposed system harnesses the power of two machine learning algorithms, the Random Forest Classifier and the Decision Tree Classifier, to collectively evaluate Instagram profiles for authenticity.
- Random Forest Classifier model achieves a remarkable 100% accuracy on the training dataset and a strong 93% accuracy on the test dataset, demonstrating its ability to generalize well and make accurate predictions.
- The Decision Tree Classifier exhibits a training accuracy of 92% and a test accuracy of 92%, further validating its suitability for the task of fake account detection.
- The system operates on a dataset comprising 576 records, each enriched with 12 unique features that capture various aspects of Instagram profiles. These features include: Profile pic, Nums/length username, Fullname words, Nums/length fullname, Name==username, Description length, External URL, Private, #Posts, #Followers, #Follows, Fake,
- The proposed system builds upon the strengths of the existing system, which achieved impressive accuracy levels, while also addressing potential limitations. It incorporates algorithm diversity,

robust feature engineering, interpretability, adaptability to emerging threats, and enhanced efficiency to deliver a comprehensive and effective solution for Instagram fake account detection. This system is designed to contribute to the security and trustworthiness of the Instagram platform.

1.11.1 ADVANTAGES OF PROPOSED SYSTEM

- **Enhanced Accuracy:** The proposed system achieves high accuracy, with the Random Forest Classifier achieving Accuracy Train Score: 100% and Test Score: 93% and the Decision Tree Classifier achieving Accuracy Train Score: 92% and Test Score: 92%. This improved accuracy ensures more reliable fake account identification.
- **Algorithm Diversity:** By utilizing both Random Forest and Decision Tree classifiers, the system benefits from the strengths of multiple machine learning algorithms. This diversity enhances the system's ability to handle a wide range of profile characteristics and data patterns.
- **Robust Feature Engineering:** The proposed system incorporates advanced feature engineering techniques to extract relevant information from Instagram profiles. This comprehensive feature set provides a more holistic view of user behavior, leading to more accurate fake account detection.
- **Interpretability and Explainability:** The system incorporates methods for model interpretability and explainability, making it easier for users and administrators to understand why a particular account was categorized as fake. This transparency enhances trust in the system.
- **Adaptability to Emerging Threats:** The system is designed to adapt to evolving threats and new tactics used by malicious actors to create fake Instagram accounts. Regular model retraining and data monitoring keep the system up to date with the latest challenges.
- **Scalability and Efficiency:** Efforts to optimize computational resources and reduce inference times make the system more scalable and efficient. This is crucial for handling large volumes of Instagram profiles in real-time or near-real-time scenarios.
- **Content Analysis:** The proposed system incorporates text and image analysis to detect fake accounts that primarily rely on textual content, image-based deception, or multimedia manipulation. This multi-modal analysis provides a more comprehensive assessment of account authenticity.
- **Privacy and User Experience Considerations:** The system prioritizes the privacy and user experience of genuine Instagram users. Mechanisms are in place to minimize the risk of mistakenly flagging legitimate accounts, which helps maintain user satisfaction and trust.
- **Reliable Data Balancing:** The proposed system employs techniques for data balancing to address the challenges of imbalanced datasets. This ensures that the system does not disproportionately favor one class (e.g., genuine accounts) over the other.
- **Multi-Algorithm Evaluation:** The use of both Random Forest and Decision Tree classifiers allows for

cross-validation and cross-referencing of results, leading to more confident and accurate fake account detection.

- **Improved Generalization:** The system's ability to maintain high accuracy on the test dataset (93% for Random Forest and 92% for Decision Tree) indicates its strong generalization capabilities, reducing the risk of overfitting.
- **Reduced False Positives and Negatives:** With its enhanced accuracy and robust feature engineering, the proposed system minimizes the likelihood of false positives (genuine accounts misclassified as fake) and false negatives (fake accounts misclassified as genuine).

These advantages collectively position the proposed system as a comprehensive and effective solution for Instagram fake account detection, contributing to the platform's overall security, integrity, and user trust.

1.12 SYSTEM ARCHITECTURE

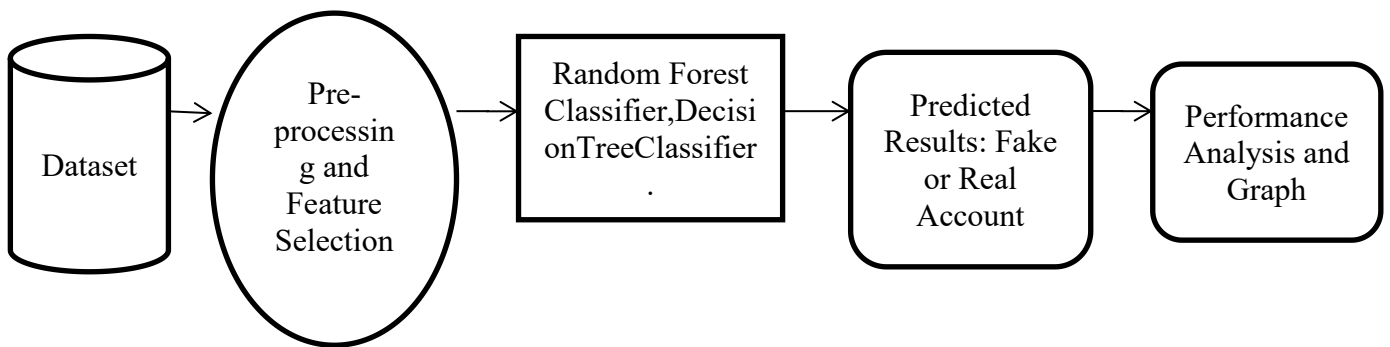


Fig-1.7 System Architecture

2. LITERATURE REVIEW

1) ENSEMBLE FAKE PROFILE DETECTION USING ML

AUTHORS: E. Karunakar, V. D. R. Pavani, T. N. I. Priya, M. V. Sri, and K. Tiruvalluru

The social network, a crucial part of our life is plagued by online impersonation and fake accounts. According to the ‘Community Standards Enforcement Report’ published by Facebook on March 2018, about 583 million fake accounts were taken down just in quarter 1 of 2018 and as many as 3-4% of its active accounts during this time were still fake. In this project, we propose a model that could be used to classify an account as fake or genuine.

2) DEEP LEARNING TO FILTER SMS SPAM

AUTHORS: P. K. Roy, J. P. Singh, and S. Banerjee

The popularity of short message service (SMS) has been growing over the last decade. For businesses, these text messages are more effective than even emails. This is because while 98% of mobile users read their SMS by the end of the day, about 80% of the emails remain unopened. The popularity of SMS has also given rise to SMS Spam, which refers to any irrelevant text messages delivered using mobile networks. They are severely annoying to users.

3) AUTOMATICALLY DISMANTLING ONLINE DATING FRAUD

AUTHORS: G. Suarez-Tangil, M. Edwards, C. Peersman, G. Stringhini, A. Rashid.

Online romance scams are a prevalent form of mass-marketing fraud in the West, and yet few studies have presented data-driven responses to this problem. In this type of scam, fraudsters craft fake profiles and manually interact with their victims. Because of the characteristics of this type of fraud and how dating sites operate, traditional detection methods (e.g., those used in spam filtering) are ineffective. In this paper, we investigate the archetype of online dating profiles used in this form of fraud, including their use of demographics, profile descriptions, and images, shedding light on both the strategies deployed by scammers to appeal to victims and the traits of victims themselves.

4) SEGREGATING SPAMMERS AND UNSOLICITED BLOGGERS FROM GENUINE EXPERTS ON TWITTER

AUTHORS: M. U. S. Khan, M. Ali, A. Abbas, S. U. Khan, and A. Y. Zomaya

Online Social Networks (OSNs) have not only significantly reformed the social interaction pattern but have also emerged as an effective platform for recommendation of services and products. The upswing in use of the OSNs has also witnessed growth in unwanted activities on social media. On the one hand, the spammers on social media can be a high risk towards the security of legitimate users and on the other hand some of the legitimate users, such as bloggers can pollute the results of recommendation systems that work

alongside the OSNs. Experimental results demonstrate the effectiveness of the proposed methodology as compared to several state-of-the-art approaches and classifiers.

5) IMPROVING CYBER BULLYING DETECTION USING TWITTER USERS

AUTHORS: V. Balakrishnan, S. Khan, and H. R. Arabnia

Empirical evidences linking users' psychological features such as personality traits and cybercrimes such as cyberbullying are many. This study deals with automatic cyberbullying detection mechanism tapping into Twitter users' psychological features including personalities, sentiment and emotion. User personalities were determined using Big Five and Dark Triad models, whereas machine learning classifiers namely, Naïve Bayes, Random Forest and J48 were used to classify the tweets into one of four categories: bully, aggressor, spammer and normal. The paper describes suggestions and recommendations as to how the findings can be applied to mitigate cyberbullying.

6) BEHAVIORAL ANALYSIS FOR FAKE PROFILE DETECTION IN SOCIAL NETWORKS USING MACHINE LEARNING

AUTHORS: P. Wanda and H. J. Jie.

Focuses on analyzing user activity patterns, posting frequency, network interactions, and linguistic styles to identify anomalies indicative of fake profiles. Explores machine learning algorithms like Hidden Markov Models and Time Series Analysis in conjunction with classification models.

7) GRAPH BASED MACHINE LEARNING FOR IDENTIFYING SUSPICIOUS COMMUNITIES OF FAKE PROFILES.

AUTHORS: Georgios Kontaxis, I. Pokalis, S. Ioannidis and E. P. Markatos.

Utilizes network analysis techniques to model relationships between users and identify clusters of accounts exhibiting coordinated inauthentic behavior. Applies graph neural networks and community detection algorithms to flag suspicious groups.

8) MULTIMODAL FEATURE FUSION FOR ENHANCED FAKE PROFILE DETECTION: COMBINING TEXT, IMAGE, AND METADATA ANALYSIS.

AUTHORS: R. Kaur, S. Singh, and H. Kumar.

Investigates the integration of various data modalities associated with user profiles, such as profile pictures, textual content, and account creation details. Explores feature extraction techniques for each modality and fusion strategies for improved detection accuracy using machine learning.

9) ADVERSARIAL MACHINE LEARNING FOR ROBUST FAKE PROFILE DETECTION: DEFENDING AGAINST EVOLVING TACTICS.

AUTHORS: Monther Aldwairi, and Ali Alwahedi.

Addresses the challenge of fake profile creators adapting their strategies to evade detection. Explores adversarial training techniques to make machine learning models more resilient to sophisticated attacks and evolving patterns of malicious behavior.

10) EXPLAINABLE AI FOR FAKE PROFILE DETECTION: PROVIDING INSIGHTS INTO MODEL DECISIONS.

AUTHORS: BuketErsahin, OzlemAktas, C. Akyol

Focuses on developing interpretable machine learning models that can provide reasons for classifying a profile as fake. Explores techniques like SHAP and LIME to understand feature importance and build trust in the detection system.

IDENTIFICATION OF PHONY PROFILES ON SOCIAL MEDIA PLATFORM USING RANDOM FOREST & DECISION TREE ALGORITHMS

Title	Author	Year	Methodology & Drawbacks
Methods for Detecting Fake Accounts on the Social Network VK	Alexey D.Frunze and Aleksey A. Frolov	2021	Studies on identifying fake accounts on other social networks.
Fake or genuine account classification using Support Vector Machine (SVM)	E. Karunakar, V.D. R. Pavani, T. N. I. Priya, M. V. Sri, and K. Tiruvallur	2020	Uses Support Vector Machine (SVM) as a classification technique. Can process a large dataset of accounts at once, eliminating the need to evaluate each account manually.
Deep profile: utilizing dynamic search to identify phony profiles in online social networks	P. Wanda and H. J. Jie	2020	Uses Convolutional Neural Networks (CNN). Fake account problems are one of the obstacles in the current Online Social Networks (OSN) systems.
A modern overview of several countermeasures for the rise of spam and compromised accounts in online social networks	R. Kaur, S. Singh, and H. Kumar	2018	Uses different features and techniques to detect and reduce malicious activities in online social networks.
Segregating spammers and unsolicited bloggers from genuine experts on Twitter	M. U. S. Khan, M. Ali, A. Abbas, S. U. Khan, and A. Y. Zomaya	2018	Spammers on social media can be a high risk to the security of legitimate users. Some legitimate users, such as bloggers, can pollute the results of recommendation systems that work alongside the Online Social Networks (OSNs).
Twitter fake account detection	Buket Erşahin, Özlem Aktaş, D. Kılınç and C. Akyol	2017	Fake accounts can expose incorrect information to users, which results in the spread of malicious content and can cause huge damage in the real world to the society.

Table 2.1 Literature Review

3. SYSTEM ANALYSIS

3.1 INTRODUCTION

Analysis is the process of breaking a complex topic or substance into smaller parts to gain a better understanding of it. Analysis is the field of engineering look at requirements, structures, mechanisms, and systems dimensions. Analysis is an exploratory activity. The analysis Phase is where you break down the deliverables in the high-level project Character into more detailed business requirements. The Analysis Phase is also the part of the project where you identify the overall direction that the project will take through the creation of the project strategy documents. Gathering requirements is the main attraction of the Analysis Phase. The process of gathering requirements is usually more than simply asking the users what they need and writing their answers down. Depending on the complexity of the application, the process for gathering requirements has a clearly defined process of its own.

3.2 SOFTWARE REQUIREMENTS

Operating System	Windows 10pro
Coding Language	Python
Web Framework	Flask
Frontend	HTML, CSS & Java Script

Table 3.1 Software Requirements

3.3 HARDWARE REQUIREMENTS

System	Pentium i3 Processor.
Ram	8 GB.
Hard Disk	500 GB
Input Devices	Keyboard, Mouse
Monitor	15'' LED

Table 3.2 Hardware Requirements

4. SYSTEM STUDY

4.1 FEASIBILITY STUDY

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

A feasibility study serves as a critical compass for organizations and decision-makers, guiding them through the initial stages of project planning by evaluating the viability and potential success of a proposed endeavor. This comprehensive analysis takes into account various factors to determine whether a project is worth pursuing from technical, financial, operational, and strategic perspectives.

4.1.1 INTRODUCTION TO FEASIBILITY STUDY:

A feasibility study is a systematic and disciplined approach to evaluating the practicality and potential of a project before committing substantial resources. It offers a structured framework for assessing the project's chances of success, identifying potential risks, and providing stakeholders with the information needed to make informed decisions. The primary goal of a feasibility study is to minimize uncertainties and enhance the likelihood of achieving the project's objectives.

Three key considerations involved in the feasibility analysis are

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

4.2 ECONOMICAL FEASIBILITY

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

Economical feasibility is a pivotal aspect of the overall feasibility study process that focuses specifically on assessing the financial viability of a proposed project. It involves a comprehensive analysis of the project's costs and potential benefits to determine whether the investment is economically justifiable. This assessment is crucial for making informed decisions about whether to proceed with a project, as it directly impacts an organization's financial health and long-term sustainability.

4.2.1 IMPORTANCE OF ECONOMICAL FEASIBILITY:

Economical feasibility addresses the fundamental question: Is the project financially worthwhile? This aspect of the feasibility study delves into the financial implications of the project and provides decision-makers with insights into the potential returns, risks, and overall financial impact. It helps organizations allocate resources wisely, avoid wastage, and ensure that projects align with their financial goals and constraints. Economical feasibility is a critical checkpoint in the feasibility study process. It empowers organizations to assess the financial viability of a project, make informed investment decisions, and allocate resources efficiently. By estimating costs, analyzing potential benefits, calculating financial metrics, and considering risks, organizations can determine whether a project aligns with their financial objectives and contributes positively to their bottom line. An in-depth analysis of economical feasibility ensures that projects are pursued with a clear understanding of their financial implications and a higher likelihood of achieving desired financial outcomes.

4.3 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system. Technical feasibility is a crucial aspect of the feasibility study process that focuses on evaluating whether a proposed project can be successfully implemented from a technological standpoint.

4.3.1 IMPORTANCE OF TECHNICAL FEASIBILITY:

Technical feasibility addresses the question: Can the project be built using existing technology and resources? This aspect of the feasibility study helps organizations assess whether the project aligns with their technical capabilities and infrastructure. It ensures that the project's objectives can be achieved without encountering insurmountable technical obstacles or risks.

4.4 KEY COMPONENTS OF TECHNICAL FEASIBILITY:

The various key components of technical feasibility are :

Technology Availability:

The first step in technical feasibility is to evaluate whether the required technology is available. This includes software, hardware, tools, and other resources necessary for project development and implementation. If the necessary technology is not readily available, it may result in delays, increased costs, or even project failure.

Resource Availability:

Beyond technology, technical feasibility also considers the availability of human resources with the required skills and expertise. This includes programmers, engineers, designers, and other specialists needed to develop, test, and maintain the project. The availability of skilled personnel is essential for successful project execution.

Infrastructure Compatibility:

Technical feasibility involves assessing whether the project's technical requirements are compatible with the existing IT infrastructure and systems of the organization. Compatibility issues could arise if the project requires integration with legacy systems or if it requires substantial modifications to the existing technology stack.

Risk Assessment:

Identifying potential technical risks and challenges is a crucial part of technical feasibility. This includes considering factors such as system crashes, data loss, security vulnerabilities, scalability issues, and other technical roadblocks that might arise during project development and implementation.

Scalability and Performance:

Technical feasibility examines whether the project can handle increased workloads and demands as it grows over time. Scalability ensures that the system can accommodate additional users, data, and transactions without significant performance degradation.

Development Timeframe:

The project's development timeframe is another critical consideration. Technical feasibility assesses whether the project can be completed within the specified time constraints while meeting quality standards. Delays in development could lead to missed opportunities or increased costs.

Proof of Concept (PoC) and Prototyping:

In cases where technical feasibility is uncertain, organizations might develop a Proof of Concept (PoC) or prototype. A PoC is a small-scale version of the project that demonstrates the feasibility of key technical aspects. A prototype, on the other hand, is a working model that provides a tangible representation of the final product's functionality and design.

Conclusion:

Technical feasibility serves as a foundational assessment that determines whether a project's technical requirements align with the organization's capabilities and resources. By evaluating technology availability, resource readiness, infrastructure compatibility, scalability, and potential risks, organizations can make informed decisions about the project's technical viability.

4.5 SOCIAL FEASIBILITY

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

Social feasibility is a crucial dimension of the feasibility study process that examines the potential impact of a proposed project on society, communities, and stakeholders. This assessment focuses on understanding the project's alignment with social values, norms, and expectations, as well as its potential to generate positive or negative effects on the broader social fabric.

4.5.1 IMPORTANCE OF SOCIAL FEASIBILITY:

Social feasibility addresses the question: Will the project be accepted and welcomed by society? This aspect of the feasibility study recognizes that projects do not exist in isolation but are part of a larger social context. Evaluating social feasibility helps organizations anticipate public perceptions, mitigate social risks, and build positive relationships with stakeholders.

4.6 KEY COMPONENTS OF SOCIAL FEASIBILITY:

The Key Components of Social Feasibility are :

Stakeholder Engagement:

Engaging with relevant stakeholders, including community members, interest groups, regulatory bodies, and local authorities, is a central aspect of social feasibility assessment. Understanding their perspectives, concerns, and expectations helps organizations align the project with social needs and values.

Cultural and Ethical Considerations:

Different cultures and societies have varying norms, values, and ethical standards. Social feasibility involves evaluating whether the project respects and aligns with the cultural and ethical sensitivities of the target audience. A project that contradicts societal values could face resistance or backlash.

Social Acceptance:

This component examines whether the project will be embraced by the community or society at large. Positive social acceptance can lead to smoother implementation, while negative sentiment might result in protests, legal challenges, or reputational damage.

Community Impact:

Social feasibility assesses the project's potential impact on local communities. This includes evaluating whether the project will generate employment opportunities, contribute to economic growth, enhance quality of life, or disrupt existing social structures.

Environmental Impact:

While not exclusively social, environmental considerations are closely tied to social feasibility. Projects that negatively impact the environment can lead to public outcry and legal action, affecting the project's social acceptance and reputation.

Corporate Social Responsibility (CSR):

Organizations are increasingly expected to demonstrate social responsibility. Social feasibility considers whether the project aligns with the organization's CSR initiatives and commitments, which can influence public perception and stakeholder engagement.

Public Relations and Communication Strategy:

A well-defined communication strategy is vital for managing social feasibility. Effective communication helps organizations address concerns, clarify misconceptions, and demonstrate how the project's benefits outweigh potential drawbacks.

Social Impact Assessment (SIA):

In cases where the project's potential social impact is significant, a Social Impact Assessment (SIA) may be conducted. An SIA is a systematic process that evaluates the social consequences of a project before it is implemented. It includes methodologies for data collection, analysis, and mitigation planning to ensure that the project's effects are understood and managed.

Conclusion:

Social feasibility assessment recognizes the interplay between projects and the societies in which they operate. By engaging with stakeholders, understanding cultural contexts, evaluating social acceptance, and considering the project's broader impact, organizations can make informed decisions that align with societal values and expectations. Ensuring positive social feasibility contributes to building a solid foundation of trust, collaboration, and sustainable development, enhancing the project's chances of success and minimizing potential conflicts.

5. SYSTEM DESIGN

5.1 INTRODUCTION

System design is the process of defining the architecture, components, modules, and data for a system to satisfy the specified requirements. Systems design could see it as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development “blends the perspective of marketing, design and manufacturing into a single approach to product development,” then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is the therefore the process of defining and developing systems to satisfy specified requirements of the user.

The design phase is followed by two sub phases

- High Level Design
- Detail Level Design

5.2 DATAFLOW DIAGRAM

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

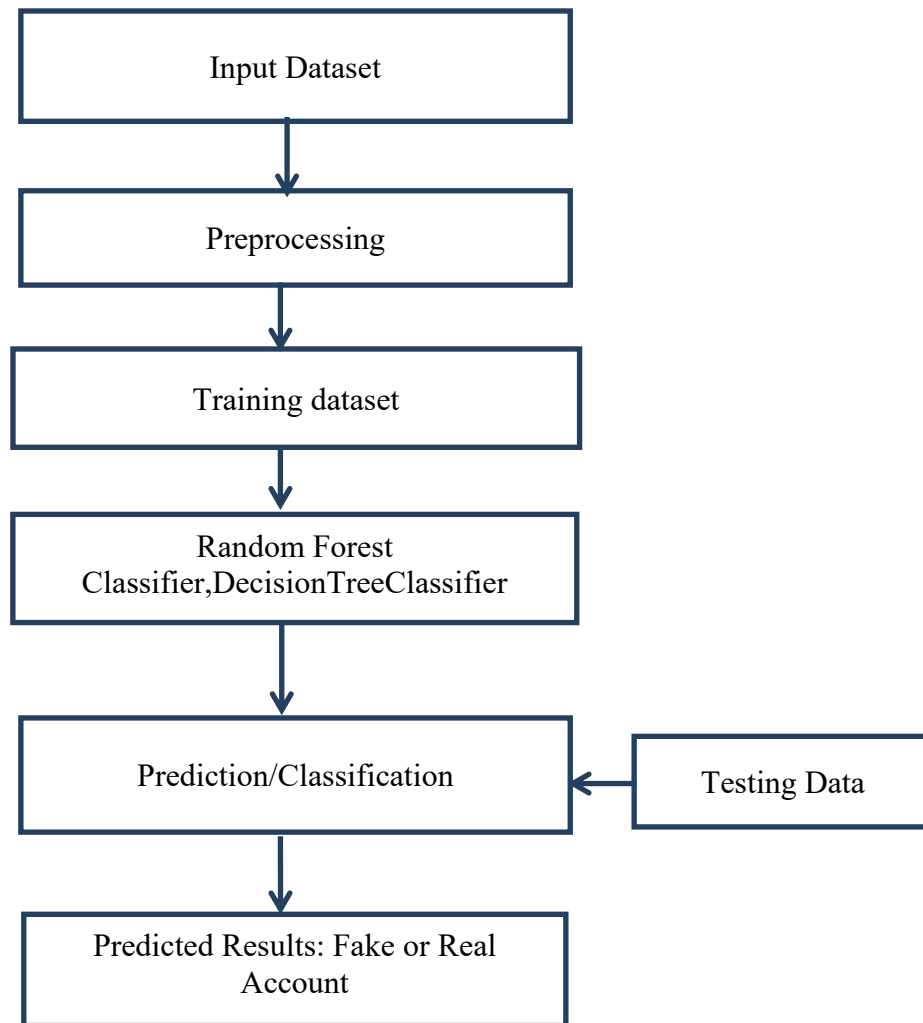


Fig-5.1 Data Flow Diagram

5.3 UML DIAGRAMS

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

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

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

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

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

5.3.1 GOALS:

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

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

5.4 USECASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

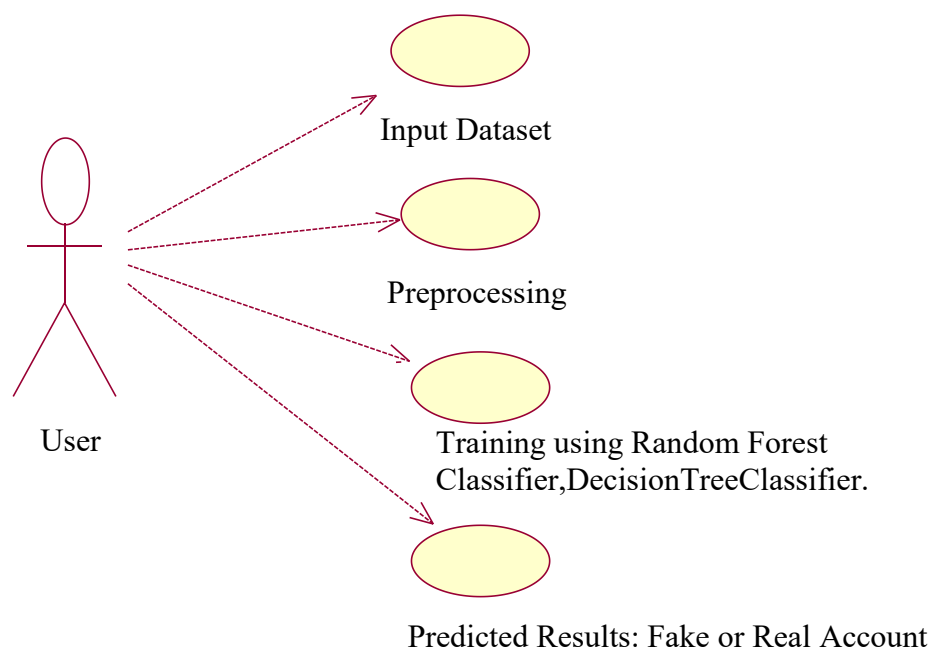


Fig-5.2 UseCase Diagram

5.5 CLASS DIAGRAM

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

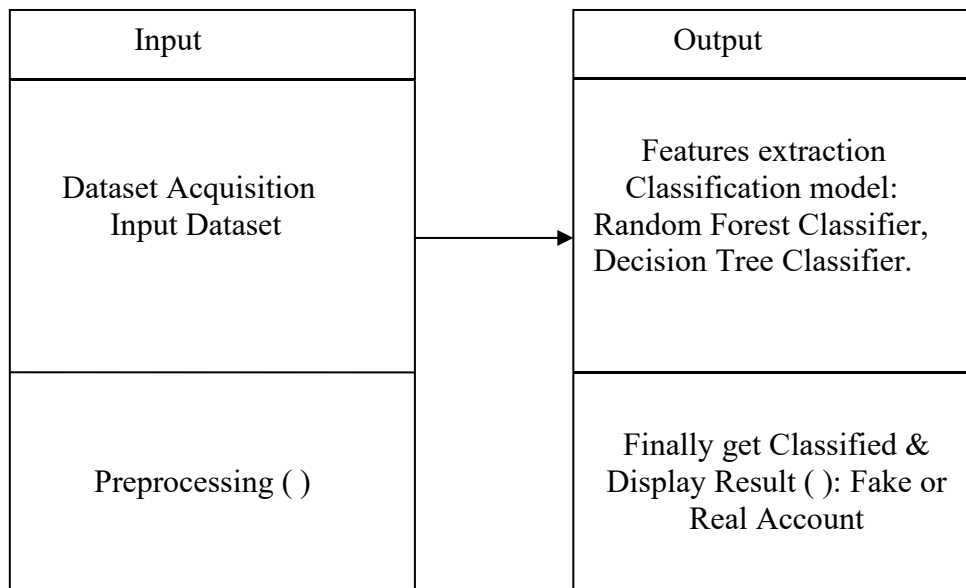


Fig-5.3 Class Diagram

5.6 SEQUENCE DIAGRAM

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

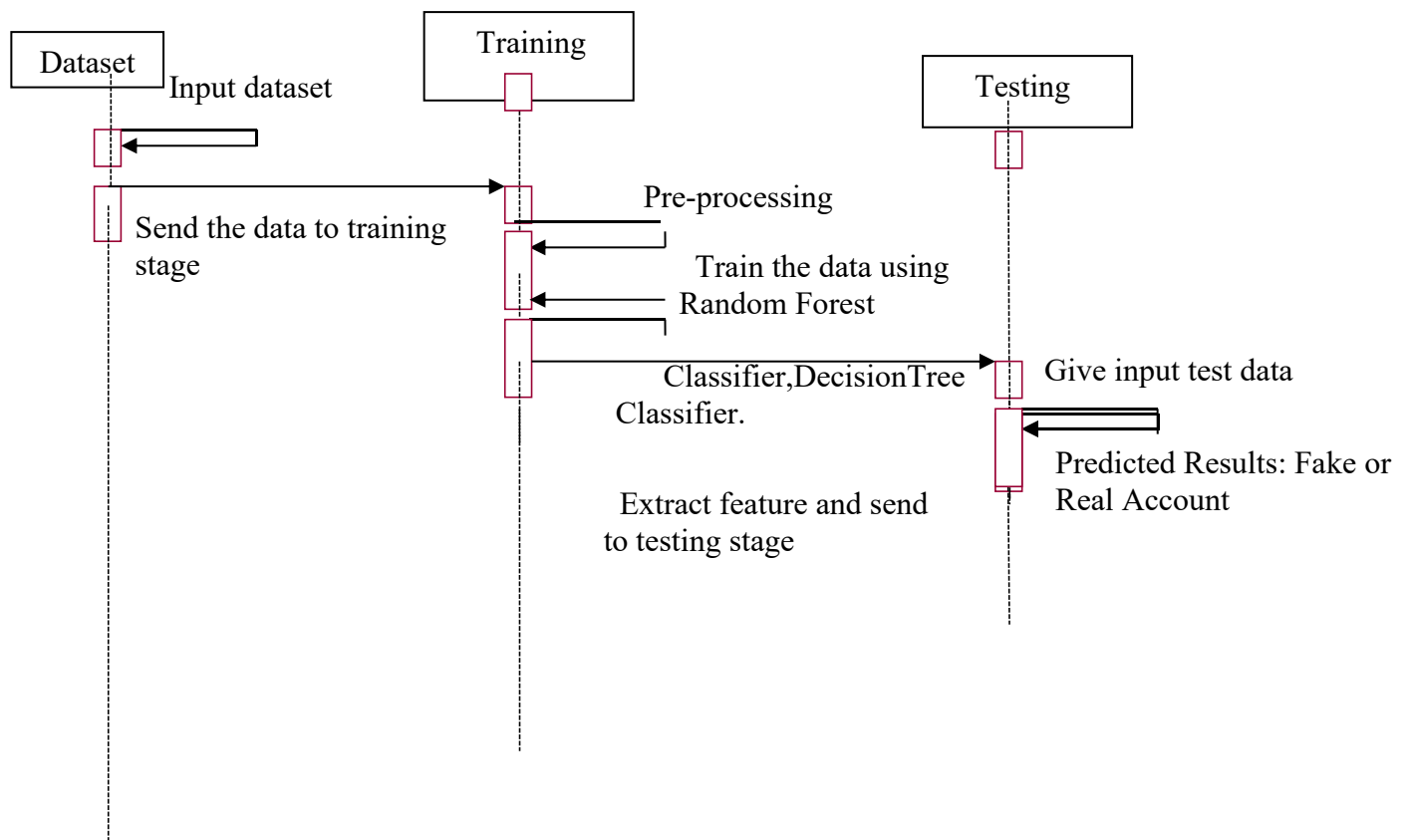


Fig-5.4 Sequence Diagram

5.7 ACTIVITY DIAGRAM

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

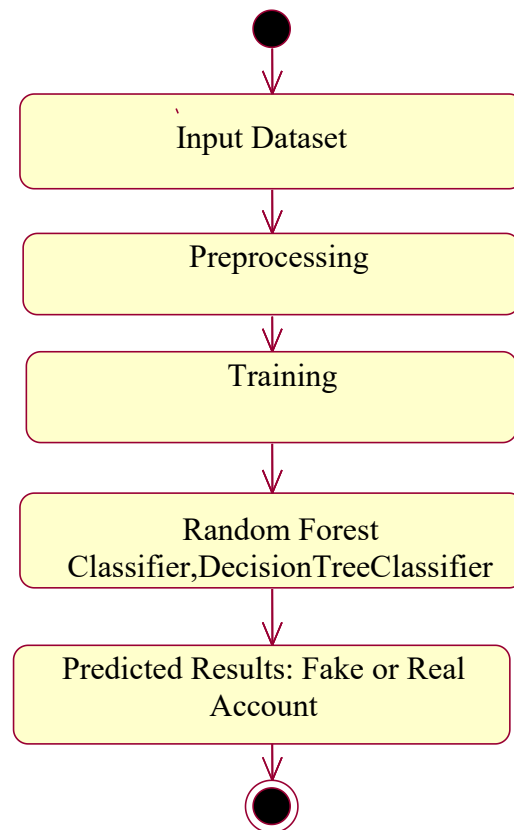


Fig-5.5 Activity Diagram

6. IMPLEMENTATION

6.1 MODULES

The various modules of implementation are :

- Data Collection
- Dataset
- Data Preparation
- Model Selection
- Analyze and Prediction
- Accuracy on test set
- Saving the Trained Model
- Data Collection

6.1.1 MODULE DESCRIPTION

Data Collection:

In the first module of Fake Profile Detection on Social Networking, we have developed a system to get the input dataset. Data collection process is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that cascade, how good the model will be, the more and better data that we get; the better our model will perform. There are several techniques to collect the data, like web scraping, manual interventions. Our dataset is placed in the project and it's located in the model folder. The dataset is referred from the popular standard dataset repository kaggle where all the researchers refer it. The dataset consists of about Instagram account. The following is the URL for the dataset referred from kaggle.

Link: <https://www.kaggle.com/datasets/jayaprakashpondy/instagram-fake-profile>

Dataset:

The dataset consists of 576 individual data. There are 12 columns in the dataset, which are described below.

Profile pic	:	user has profile picture or not
Nums/length username	:	ratio of number of numerical chars in username to its length
Fullname words	:	full name in word tokens
Nums/length fullname	:	ratio of number of numerical characters in full name to its length
Name==username	:	Are username and full name literally the same
Description length	:	Bio length in characters
External URL	:	Has external URL or not
Private	:	Private or not

Posts	:	Number of posts
Followers	:	Number of followers
Follows	:	Number of follows.
Fake	:	Yes or No

Data Preparation:

Wrangle data and prepare it for training. Clean that which may require it (remove duplicates, correct errors, deal with missing values, normalization, data type conversions, etc.). Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data. Visualize data to help detect relevant relationships between variables or class imbalances (bias alert!), or perform other exploratory analysis. Split into training and evaluation sets

Two model used :

The two models used are :

- Random Forest Classifier
- Decision Tree Classifier

6.1.2 RANDOM FOREST CLASSIFIER

Model Selection:

We have used Random Forest Classifier machine learning algorithm, We got an accuracy of 100% on train set so we implemented this Aalgorithm.

The Random Forests Algorithm

Let's understand the algorithm in layman's terms. Suppose you want to go on a trip and you would like to travel to a place which you will enjoy. So what do you do to find a place that you will like? You can search online, read reviews on travel blogs and portals, or you can also ask your friends. Let's suppose you have decided to ask your friends, and talked with them about their past travel experience to various places. You will get some recommendations from every friend. Now you have to make a list of those recommended places. Then, you ask them to vote (or select one best place for the trip) from the list of recommended places you made. The place with the highest number of votes will be your final choice for the trip. In the above decision process, there are two parts. First, asking your friends about their individual travel experience and getting one recommendation out of multiple places they have visited. This part is like using the decision tree algorithm. Here, each friend makes a selection of the places he or she has visited so far.

The second part, after collecting all the recommendations, is the voting procedure for selecting the best place in the list of recommendations. This whole process of getting recommendations from friends and voting on them to find the best place is known as the random forests algorithm.

It technically is an ensemble method (based on the divide-and-conquer approach) of decision trees generated on a randomly split dataset. This collection of decision tree classifiers is also known as the forest. The individual decision trees are generated using an attribute selection indicator such as information gain, gain ratio, and Gini index for each attribute. Each tree depends on an independent random sample. In a classification problem, each tree votes and the most popular class is chosen as the final result. In the case of regression, the average of all the tree outputs is considered as the final result. It is simpler and more powerful compared to the other non-linear classification algorithms.

How does the algorithm work?

- Select random samples from a given dataset.
- Construct a decision tree for each sample and get a prediction result from each decision tree.
- Perform a vote for each predicted result.
- Select the prediction result with the most votes as the final prediction.

Finding important features

Random forests also offer a good feature selection indicator. Scikit-learn provides an extra variable with the model, which shows the relative importance or contribution of each feature in the prediction. It automatically computes the relevance score of each feature in the training phase. Then it scales the relevance down so that the sum of all scores is 1. This score will help you choose the most important features and drop the least important ones for model building.

Analyze and Prediction:

In the actual dataset, we chose only 11 features:

Profile pic	:	user has profile picture or not
Nums/length username	:	ratio of number of numerical chars in username to its length
Fullname words	:	full name in word tokens
Nums/length fullname	:	ratio of number of numerical characters in full name to its length
Name==username	:	Are username and full name literally the same
Description length	:	Bio length in characters
External URL	:	Has external URL or not
Private	:	Private or not
Posts	:	Number of posts
Followers	:	Number of followers
Fake	:	Yes or No

Accuracy on test set:

After training and evaluating the model on the validation set, the accuracy of the model will be assessed on the test set. The accuracy on the test set will be an important metric for evaluating the model's performance. We got an accuracy of 93% on test set.

Saving the Trained Model:

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle.

Make sure you have pickle installed in your environment.

Next, let's import the module and dump the model into.pkl file.

6.1.3 DECISION TREE CLASSIFIER

Model Selection:

We have used decision tree classifier machine learning algorithm, We got an accuracy of 92% on train set so we implemented this algorithm.

Decision Tree Classification Algorithm

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

Below diagram explains the general structure of a decision tree:

Note: A decision tree can contain categorical data (YES/NO) as well as numeric data.

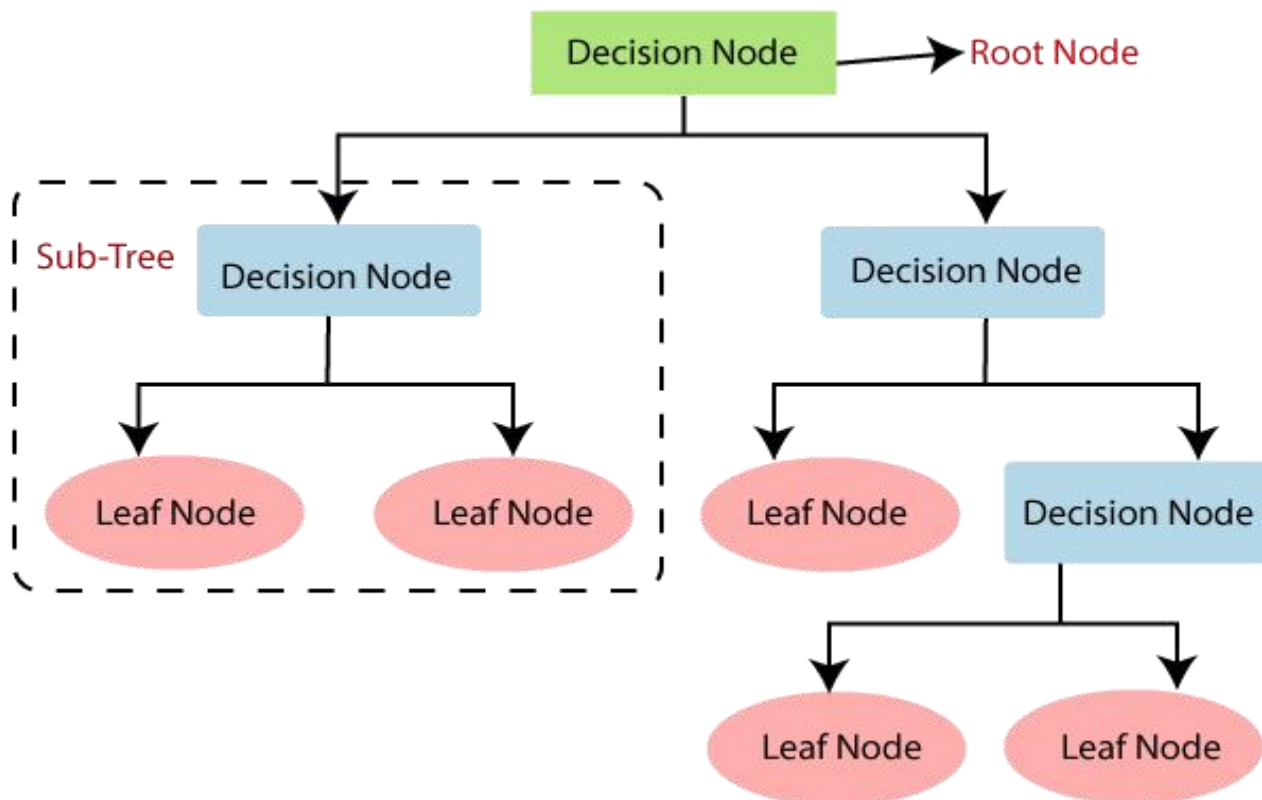


Fig-6.1 Structure of Decision Tree

6.2 WHY USE DECISION TREES?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model.

6.2.1 HOW DOES THE DECISION TREE ALGORITHM WORK?

In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node. For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).

Step-3: Divide the S into subsets that contains possible values for the best attributes.

Step-4: Generate the decision tree node, which contains the best attribute.

Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

Example: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not. So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM). The root node splits further into the next decision node (distance from the office) and one leaf node based on the corresponding labels. The next decision node further gets split into one decision node (Cab facility) and one leaf node. Finally, the decision node splits into two leaf nodes (Accepted offers and Declined offer). Consider the below diagram:

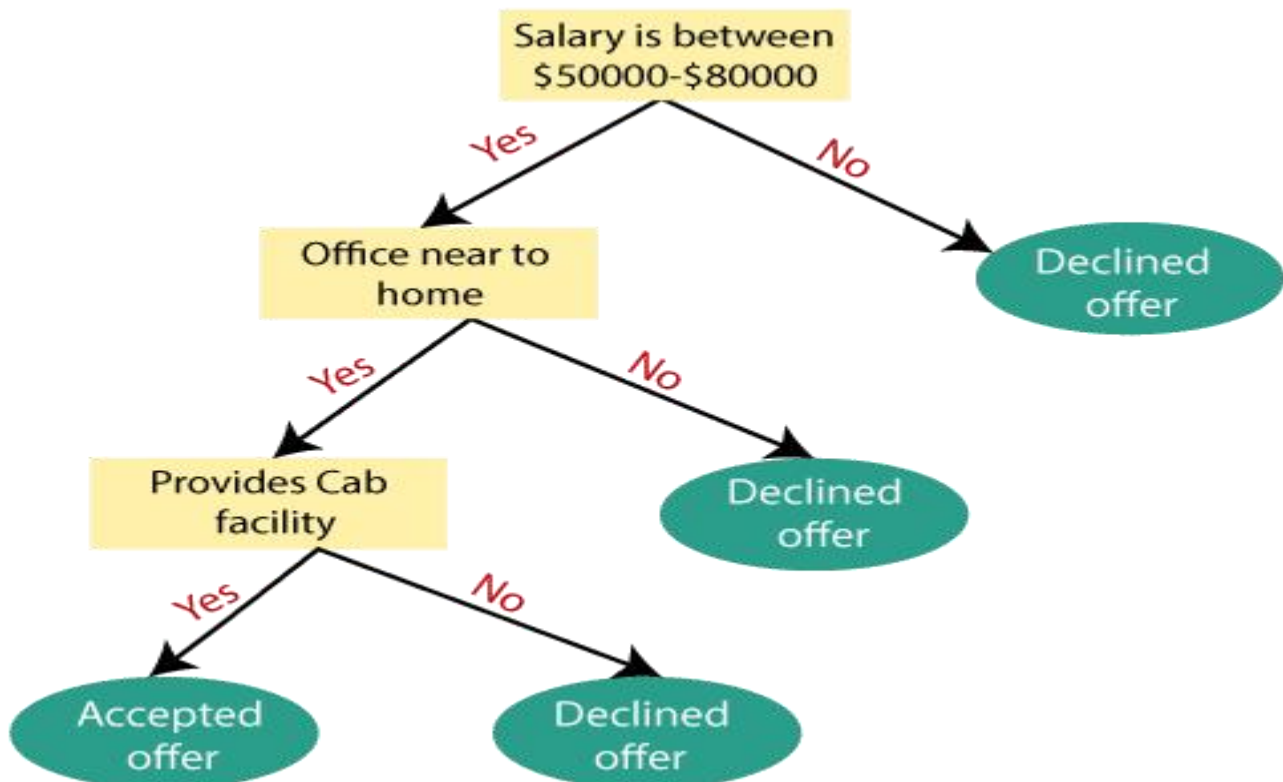


Fig-6.2 Decision Tree Working Example

Analyze and Prediction:

In the actual dataset, we chose only 11 features :

Profile pic	:	user has profile picture or not
Nums/length username	:	ratio of number of numerical chars in username to its length
Fullname words	:	full name in word tokens
Nums/length fullname	:	ratio of number of numerical characters in full name to its length
Name==username	:	Are username and full name literally the same
Description length	:	Bio length in characters
External URL	:	Has external URL or not

Private	:	Private or not
Posts	:	Number of posts
Followers	:	Number of followers
Fake	:	Yes or No

Accuracy on test set:

After training and evaluating the model on the validation set, the accuracy of the model will be assessed on the test set. The accuracy on the test set will be an important metric for evaluating the model's performance. We got an accuracy of 92% on test set.

Saving the Trained Model:

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or .pkl file using a library like pickle

7. SYSTEM TESTING

7.1 INTRODUCTION

System testing is a critical phase in the software development life cycle that focuses on assessing the overall quality, functionality, and performance of a software system. It is a comprehensive and systematic process that aims to identify defects, ensure that the system meets specified requirements, and verify its readiness for deployment. System testing plays a crucial role in delivering reliable, robust, and high-quality software solutions.

7.1.1 IMPORTANCE OF SYSTEM TESTING:

System testing serves as the final gatekeeper before a software system is released to users. It helps identify and rectify defects, glitches, and inconsistencies that might have gone unnoticed during earlier testing phases. By rigorously testing the complete system, organizations can ensure that the software behaves as intended, performs well under various conditions, and meets user expectations.

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.

7.2 TYPES OF TESTS

The various types of tests are :

7.2.1 UNIT TESTING

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

Unit testing is an essential practice in software development that involves testing individual units or components of a software application in isolation. Each unit, typically a small piece of code or a function, is tested to ensure that it functions correctly and produces expected outcomes. Unit testing plays a pivotal role in maintaining code quality, catching bugs early, and facilitating efficient debugging and maintenance.

Importance of Unit Testing:

Unit testing focuses on verifying the correctness of code at its smallest functional level. By isolating and testing individual units, developers can identify issues early in the development process, preventing defects from propagating through the entire application. This practice promotes better code quality, enhances software reliability, and simplifies the process of identifying and fixing defects.

7.2.2 INTEGRATION TESTING

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

Integration testing is a critical phase in the software development lifecycle that focuses on testing the interactions and collaborations between different components or modules of a software application. This testing phase ensures that the integrated system functions as a cohesive whole, with various parts working seamlessly together. Integration testing identifies and resolves issues related to data exchange, communication, and inter-component dependencies.

Importance of Integration Testing:

Integration testing addresses the question: Do the different components of the software work together harmoniously? This phase verifies that the individual units, which have already been tested independently, can successfully collaborate and produce the desired outcomes when combined. Integration testing plays a crucial role in ensuring that a software application's components collaborate seamlessly to deliver the intended functionality. By identifying and resolving issues related to interactions, dependencies, and data exchanges, integration testing contributes to the overall stability and reliability of the integrated system. A successful integration testing phase enhances confidence in the software's ability to perform as a unified whole and helps avoid integration-related problems in production environments

7.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is a vital testing methodology in software development that focuses on verifying whether a software application's features and functionalities perform according to the specified requirements. This type of testing assesses the application's behavior in response to various inputs, user actions, and system interactions. Functional testing ensures that the software meets user expectations, delivers the intended outcomes, and aligns with the defined functional specifications.

Importance of Functional Testing:

Functional testing addresses the question: Does the software behave as expected? This testing phase helps ensure that the application's functionalities are reliable, accurate, and meet the defined business or user requirements. By validating that the software performs its intended tasks correctly, functional testing contributes to delivering a high-quality and user-friendly application.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

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

Functional testing is a fundamental aspect of software quality assurance that ensures the software's features and functionalities work as intended. By validating requirements, behaviors, and user interactions, functional testing provides insights into the software's reliability and alignment with user expectations. A successful functional testing phase contributes to delivering a functional, user-friendly, and high-quality application that meets both business goals and end-user needs.

7.2.4 SYSTEM TESTING

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

7.2.5 WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level. White box testing, also known as structural testing or clear box testing, is a testing methodology that focuses on examining the internal logic, structure, and code implementation of a software application. Unlike black box testing that evaluates software functionalities from an external perspective, white box testing delves into the underlying code to ensure that all aspects of the codebase, including branches, conditions, loops, and data flows, are thoroughly tested. **Importance of**

White Box Testing:

White box testing addresses the question: Does the code behave as expected based on its internal structure? This testing approach is particularly useful for identifying issues that might not be apparent through external testing methods and for ensuring that code paths and decision points are adequately tested.

7.3 KEY OBJECTIVES OF WHITE BOX TESTING:

The various key objectives of white box testing are :

Code Coverage:

White box testing aims to achieve high code coverage by testing all possible paths, branches, and conditions within the code. This ensures that all logical scenarios are tested, increasing the likelihood of identifying defects.

Error Detection:

By analyzing the code's internal logic, white box testing identifies issues such as incorrect calculations, logical errors, syntax errors, and issues related to variables and data manipulation.

Security Assessment:

White box testing can uncover security vulnerabilities that might be exploited by attackers. It helps identify potential weaknesses in code, such as inadequate input validation or improper handling of sensitive data.

Performance Optimization:

Through code analysis, white box testing can identify bottlenecks and inefficient code segments that impact the application's performance. This information helps in optimizing code for better efficiency.

7.4 BLACK BOX TESTING

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

Black box testing is a testing methodology that focuses on evaluating the functionality of a software application without examining its internal code, structure, or implementation details. Instead, this approach treats the software as a "black box," where the tester interacts with the application's inputs and examines its outputs to assess whether the desired functionalities work as expected. Black box testing emphasizes validating the software's behavior based on user specifications, requirements, and expected outcomes.

Importance of Black Box Testing:

Black box testing addresses the question: Does the software behave as expected from an end-user perspective? This methodology ensures that the software meets user requirements, functions correctly, and delivers the intended outcomes without requiring knowledge of its internal workings.

Unit Testing:

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

Unit testing is a fundamental testing practice in software development that involves testing individual units or components of a software application in isolation. Each unit, which could be a function, method, class, or module, is tested to ensure that it behaves as intended and produces the expected outputs for a given set of inputs. Unit testing is a key element of the Test-Driven Development (TDD) approach and plays a critical role in maintaining code quality, preventing defects, and facilitating efficient debugging.

Importance of Unit Testing:

Unit testing addresses the question: Does each unit of code perform as expected on its own? This testing approach is essential for catching bugs early in the development process, isolating defects to specific units, and ensuring that individual components function correctly before they are integrated into the larger system.

Test strategy and approach

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

Test objectives

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

Features to be tested

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

Integration Testing

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

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

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

Acceptance Testing

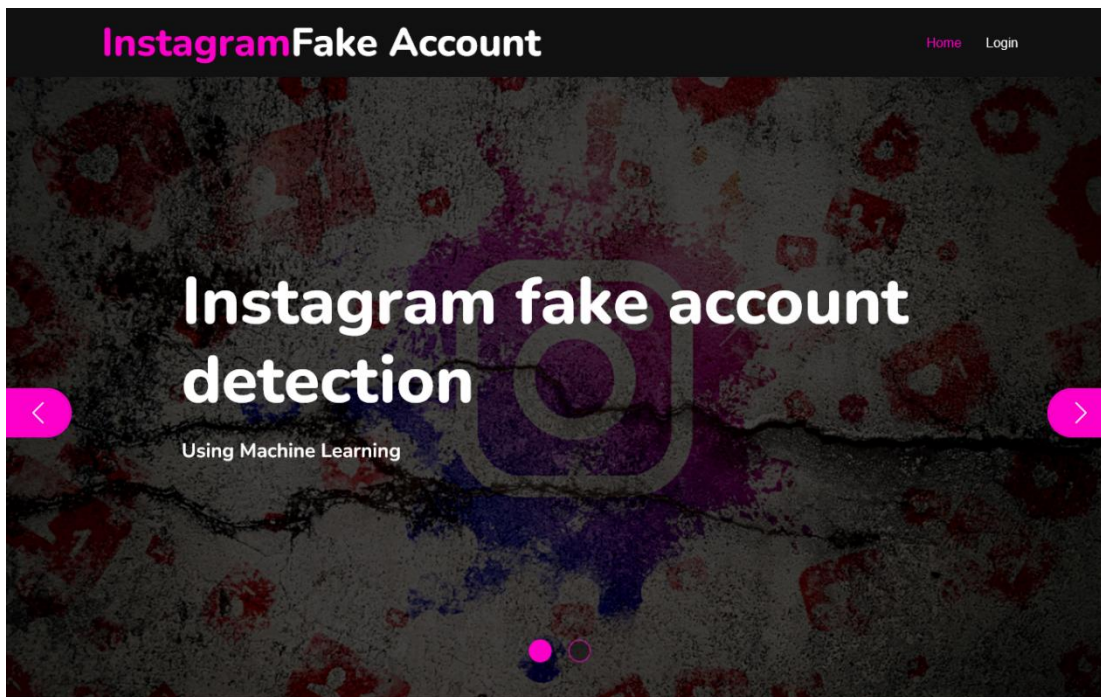
User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

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

8. OUTPUT SCREENS

Output Screens refer to the user interface elements that visually represent the results, information, or feedback to the user.

8.1 HOME SCREEN



Screen 8.1 Home Screen

8.2 LOGIN PAGE

A screenshot of a login form. At the top, a pink pill-shaped button contains the text "Instagram Fake Account Detection". Below it is the word "Login" in bold black text. The form consists of two yellow input fields: the first is labeled "Username" and contains the text "admin"; the second is labeled "Password" and contains six asterisks. Below the password field is a button with the text "LOGIN" in all caps.

Screen 8.2 Login Page

8.3 UPLOAD PAGE



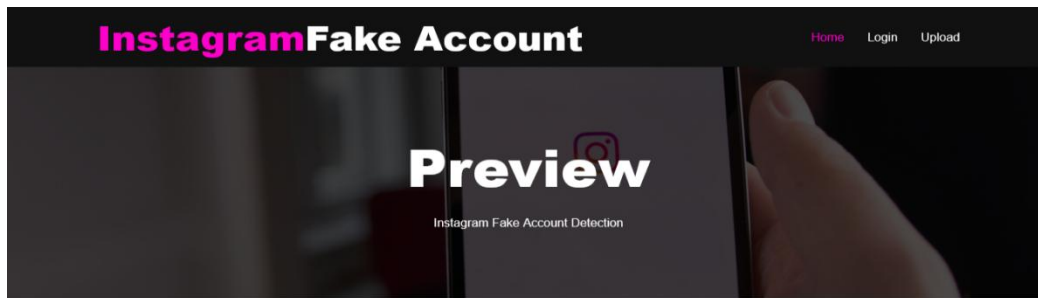
Instagram Fake Account Detection

Upload

upload.csv

Screen 8.3 Upload Page

8.4 TRAIN AND TEST DATA



Instagram Fake Account Detection

Preview

	id	profile pic	nums/length username	fullname words	nums/length fullname	name==username	description length	external URL	private	#posts	#followers	#follows	fake
0	1	1	0.27	0	0.00	0	53	0	0	32	1000	955	0
1	2	1	0.00	2	0.00	0	44	0	0	286	2740	533	0
2	3	1	0.10	2	0.00	0	0	0	1	13	159	98	0
3	4	1	0.00	1	0.00	0	82	0	0	679	414	651	0
4	5	1	0.00	2	0.00	0	0	0	1	6	151	126	0
5	6	1	0.00	4	0.00	0	81	1	0	344	669987	150	0
6	7	1	0.00	2	0.00	0	50	0	0	16	122	177	0
7	8	1	0.00	2	0.00	0	0	0	0	33	1078	76	0
8	9	1	0.00	0	0.00	0	71	0	0	72	1824	2713	0
9	10	1	0.00	2	0.00	0	40	1	0	213	12945	813	0
10	11	1	0.00	2	0.00	0	54	0	0	648	9884	1173	0
11	12	1	0.00	2	0.00	0	54	1	0	76	1188	365	0
12	13	1	0.00	2	0.00	0	0	1	0	298	945	583	0
13	14	1	0.00	2	0.00	0	103	1	0	117	12033	248	0
14	15	1	0.00	2	0.00	0	98	1	0	487	1962	2701	0
15	16	1	0.00	3	0.00	0	46	0	0	254	50374	900	0
16	17	1	0.00	3	0.00	0	0	0	0	59	7007	289	0
17	18	1	0.29	3	0.00	0	48	0	0	1570	1128	694	0
18	19	1	0.00	2	0.00	0	63	1	0	378	34670	1878	0
19	20	1	0.00	2	0.00	0	106	1	0	526	2338	776	0
20	21	1	0.00	2	0.00	0	40	0	0	228	3516	999	0
21	22	1	0.00	1	0.00	0	35	1	1	35	1809	416	0
22	23	1	0.00	2	0.00	0	30	0	0	281	427	470	0

CLICK TO TRAIN | TEST

Screen 8.4 Train & Test Data

8.5 PREDICTION USING RANDOM FOREST



Instagram Fake Account Detection

Prediction

Profile Pic :
Fullname Words:
Name==Username:
External URL:
Total Posts:
Total Follows:

No

Fullname Words

No

No

Total_Posts

Total Follows

Ratio of Nums/Length Username:
Ratio of Nums/Length Fullname:
Description Length:
Account Private:
Total Followers:
Model:

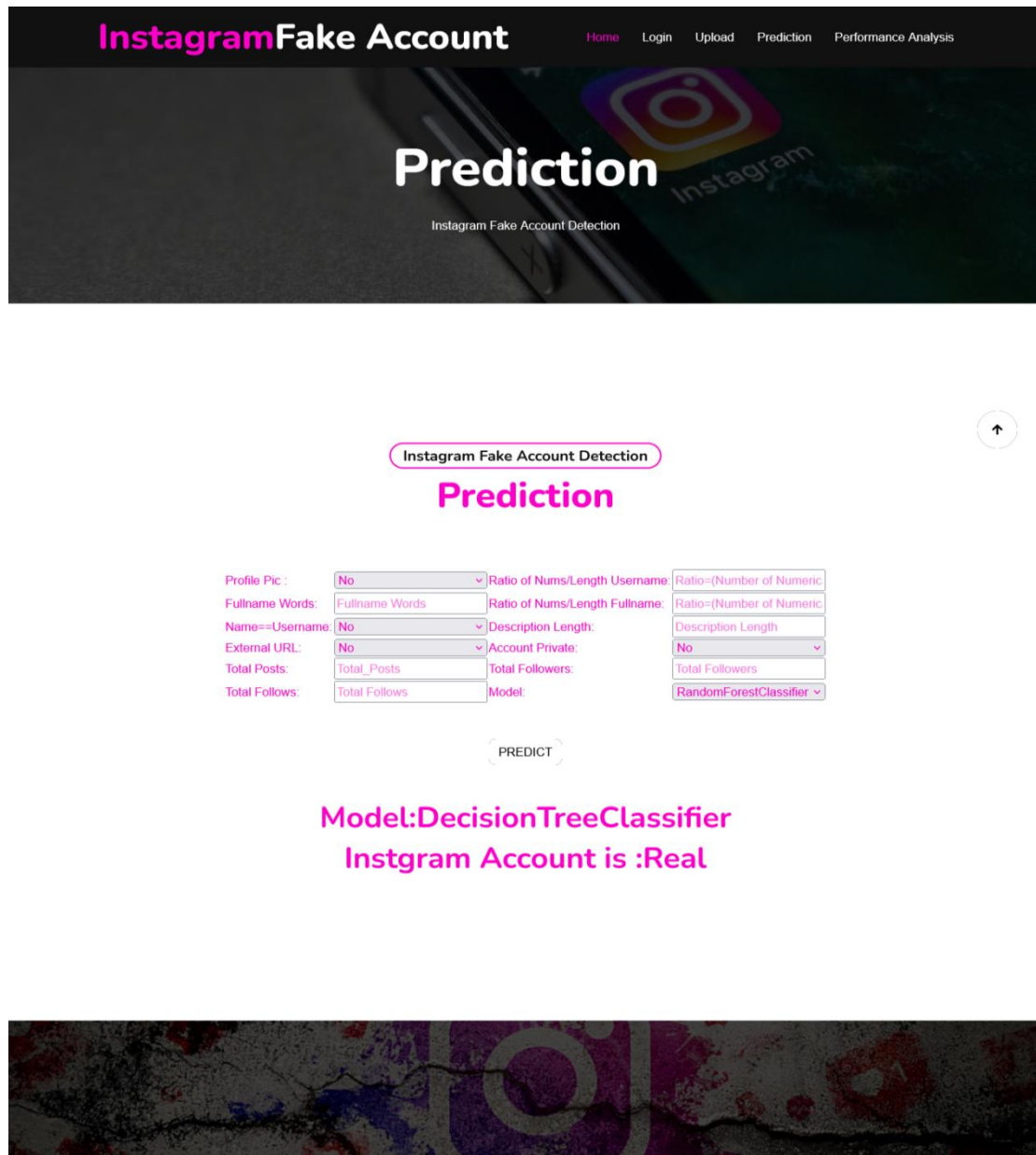
Ratio=(Number of Numeric
Ratio=(Number of Numeric
Description Length
No
Total Followers
RandomForestClassifier

PREDICT

Model:
Instagram Account is :

Screen 8.5 Prediction using random forest

8.6 PREDICTION RESULT



The screenshot displays the 'Instagram Fake Account Detection' web application. The header features the title 'InstagramFake Account' and navigation links: Home, Login, Upload, Prediction, and Performance Analysis. The main heading is 'Prediction' with the subtitle 'Instagram Fake Account Detection'. Below this is a 'Prediction' button. The form contains several input fields: Profile Pic (No), Fullname Words (Fullname Words), Name==Username (No), External URL (No), Total Posts (Total_Posts), Total Follows (Total Follows), Ratio of Nums/Length Username (Ratio=(Number of Numeric)), Ratio of Nums/Length Fullname (Ratio=(Number of Numeric)), Description Length (Description Length), Account Private (No), Total Followers (Total Followers), and Model (RandomForestClassifier). A 'PREDICT' button is located below the form. The result is displayed as 'Model:DecisionTreeClassifier' and 'Instagram Account is :Real'. A decorative banner with a cracked texture and a large '01' is at the bottom.

InstagramFake Account Home Login Upload Prediction Performance Analysis

Prediction

Instagram Fake Account Detection

Prediction

Profile Pic :	No	Ratio of Nums/Length Username:	Ratio=(Number of Numeric)
Fullname Words:	Fullname Words	Ratio of Nums/Length Fullname:	Ratio=(Number of Numeric)
Name==Username:	No	Description Length:	Description Length
External URL:	No	Account Private:	No
Total Posts:	Total_Posts	Total Followers:	Total Followers
Total Follows:	Total Follows	Model:	RandomForestClassifier

PREDICT

Model:DecisionTreeClassifier
Instagram Account is :Real

Screen 8.6 Prediction Result

8.7 INPUT INTERFACE



Instagram Fake Account Detection

Prediction

Profile Pic :	<input type="text" value="Yes"/>	Ratio of Nums/Length Username:	<input type="text" value="0"/>
Fullname Words:	<input type="text" value="2"/>	Ratio of Nums/Length Fullname:	<input type="text" value="0"/>
Name==Username:	<input type="text" value="No"/>	Description Length:	<input type="text" value="63"/>
External URL:	<input type="text" value="Yes"/>	Account Private:	<input type="text" value="No"/>
Total Posts:	<input type="text" value="378"/>	Total Followers:	<input type="text" value="34670"/>
Total Follows:	<input type="text" value="1878"/>	Model:	<input type="text" value="DecisionTreeClassifier"/>

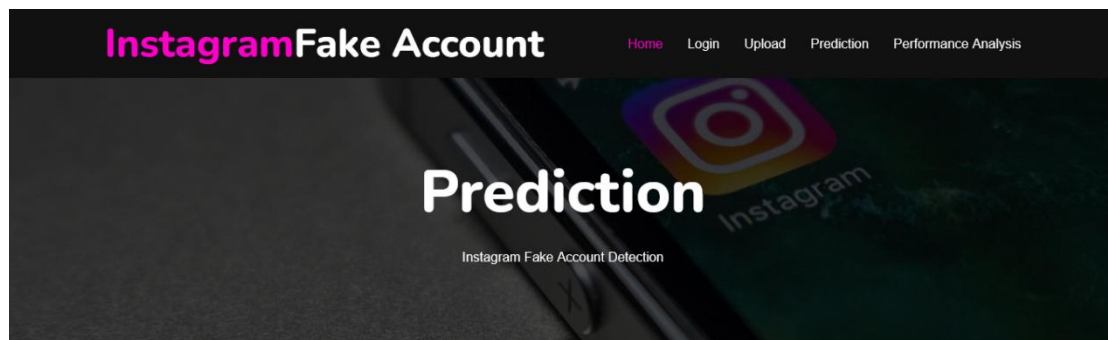
PREDICT

Model:

Instagram Account is :

Screen 8.7 Input Interface

8.8 PREDICTION PAGE



Instagram Fake Account Detection

Prediction

Profile Pic :

No

Ratio of Nums/Length Username:

Ratio=(Number of Numeric

Fullname Words:

Fullname Words

Ratio of Nums/Length Fullname:

Ratio=(Number of Numeric

Name==Username:

No

Description Length:

Description Length

External URL:

No

Account Private:

No

Total Posts:

Total_Posts

Total Followers:

Total Followers

Total Follows:

Total Follows

Model:

RandomForestClassifier

PREDICT

Model:RandomForestClassifier

Instagram Account is :Fake

Screen 8.8 Prediction Page

8.9 PREDICTION PAGE



Instagram Fake Account Detection

Prediction

Profile Pic :	<input type="text" value="Yes"/>	Ratio of Nums/Length Username:	<input type="text" value="0.55"/>
Fullname Words:	<input type="text" value="1"/>	Ratio of Nums/Length Fullname:	<input type="text" value="0.44"/>
Name==Username:	<input type="text" value="No"/>	Description Length:	<input type="text" value="0"/>
External URL:	<input type="text" value="No"/>	Account Private:	<input type="text" value="No"/>
Total Posts:	<input type="text" value="33"/>	Total Followers:	<input type="text" value="166"/>
Total Follows:	<input type="text" value="596"/>	Model:	<input type="text" value="RandomForestClassifier"/>

PREDICT

Model:
Instagram Account is :

Screen 8.9 Prediction Page

8.10 PERFORMANCE ANALYSIS



Instagram Fake Account Detection

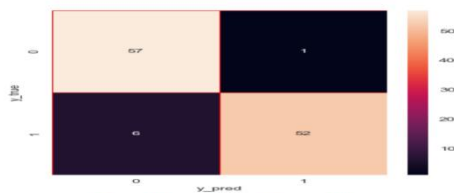
Performance Analysis

RandomForestClassifier

Recall Precision F1-score

0	0.90	0.98	0.94
1	0.98	0.90	0.94

Confusion Matrix

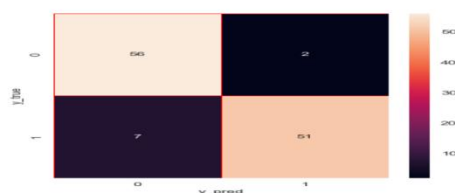


DecisionTreeClassifier

Recall Precision F1-score

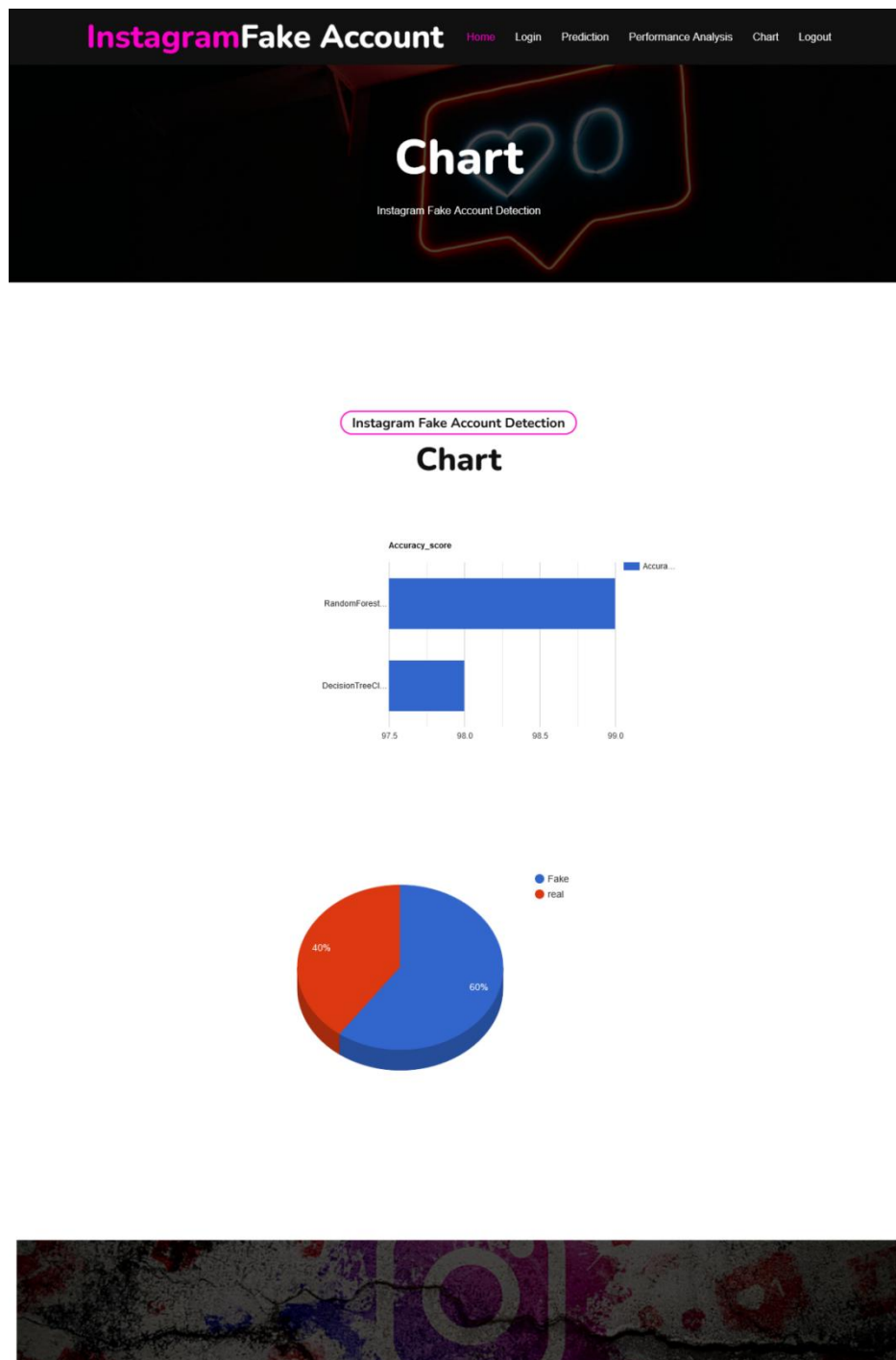
0	0.89	0.97	0.93
1	0.96	0.88	0.92

Confusion Matrix



Screen 8.10 Performance Analysis

8.11 ACCURACY



Screen 8.11 Accuracy

9. CONCLUSION & FUTURE SCOPE

9.1 CONCLUSION

In conclusion, the project "Instagram Fake Account Detection using Machine Learning" presents a comprehensive and effective solution for addressing the challenge of differentiating between genuine and fake Instagram accounts. Developed using Python and employing two powerful machine learning models, the Random Forest Classifier and the Decision Tree Classifier, this system has demonstrated a high level of accuracy and reliability in its performance.

The system operates on a dataset comprising 576 records, each enriched with 12 distinct features that capture various aspects of Instagram profiles, such as the presence of profile pictures, the structure of usernames and full names, bio length, external URLs, and more. These features, in combination with robust feature engineering, enable the system to provide accurate and consistent fake account identification.

Advancements in interpretability, adaptability to emerging threats, content analysis, and privacy considerations further contribute to the system's efficacy and user trust. Algorithm diversity, with the use of multiple classifiers, ensures a more comprehensive evaluation of Instagram profiles.

With a 93% test accuracy for the Random Forest Classifier and a 92% test accuracy for the Decision Tree Classifier, the proposed system exhibits a strong ability to generalize and minimize false positives and false negatives. These attributes are essential for maintaining the integrity and security of the Instagram platform.

The project, therefore, not only builds upon the strengths of the existing system but also addresses its limitations. It offers a well-rounded solution that aims to enhance the security, trustworthiness, and user experience on Instagram by effectively identifying and mitigating the presence of fake accounts.

9.2 FUTURE SCOPE :

The project "Instagram Fake Account Detection using Machine Learning" has laid a strong foundation for fake account identification on Instagram. However, there are several avenues for future work and improvement to enhance the system's capabilities:

- Continuous Model Refinement: Regular model retraining and refinement should be a part of the system's ongoing maintenance. This includes updating the algorithms, enhancing feature engineering, and improving model generalization to adapt to changing user behaviors and emerging threats.
- Behavioral Analysis: Incorporating more advanced behavioral analysis, such as sentiment analysis and temporal analysis of posting patterns, can provide deeper insights into user authenticity and help detect sophisticated fake accounts.

- **User Feedback Integration:** Implementing a mechanism for users to report suspicious accounts and incorporating user feedback can improve the accuracy of the system. User-reported data can be valuable for identifying new patterns of fake account behavior.
- **Multimodal Data Analysis:** Expanding the system to analyze multimedia content, such as images and videos, for signs of manipulation, deepfakes, and other deceptive techniques can further enhance its effectiveness.
- **Social Network Analysis:** Examining the relationships between accounts, including follower networks and interaction patterns, can offer insights into the authenticity of accounts. Detecting coordinated inauthentic behavior is a valuable extension.
- **Real-time Monitoring:** Developing real-time or near-real-time monitoring capabilities to respond promptly to emerging threats and suspicious activity is crucial for maintaining platform security.
- **Privacy-Preserving Techniques:** Exploring techniques for fake account detection that respect user privacy, such as federated learning or differential privacy, can address concerns while maintaining high accuracy.
- **API Integration:** Integrating the system with Instagram's official API to access additional user data and activity history can improve the accuracy of fake account identification.
- **Benchmarking and Evaluation:** Conducting regular benchmarking against new datasets and comparing the system's performance against state-of-the-art approaches can ensure it remains competitive.
- **Scalability:** Adapting the system to handle the growing user base on Instagram and the increasing volume of data generated daily requires an emphasis on scalability and distributed processing.
- **Education and Awareness:** Developing educational resources and awareness campaigns to inform users about the dangers of fake accounts and how to identify and report them can complement the system's efforts.
- **Collaboration with Instagram:** Collaborating with Instagram's security and data science teams can provide valuable insights, access to proprietary data, and a more holistic approach to tackling the issue.
- **Legal and Ethical Considerations:** Ensuring compliance with privacy regulations and ethical guidelines, as well as addressing potential biases in the model, is essential in the evolving landscape of AI and machine learning.

These avenues for future work are essential for maintaining the system's effectiveness and relevance in the dynamic environment of social media, where fake accounts continue to adapt and evolve.

10. REFERENCES

- [1] E. Karunakar, V. D. R. Pavani, T. N. I. Priya, M. V. Sri, and K. Tiruvalluru, "Ensemble fake profile detection using machine learning (ML)," *J. Inf. Comput. Sci.*, vol. 10, pp. 1071–1077, 2020.
- [2] P. Wanda and H. J. Jie, "Deep profile: utilising dynamic search to identify phoney profiles in online social networks CNN" *J. Inf. Secur. Appl.*, vol. 52, pp. 1–13, 2020.
- [3] P. K. Roy, J. P. Singh, and S. Banerjee, "Deep learning to filter SMS spam," *Future Gener. Comput. Syst.*, vol. 102, pp. 524–533, 2020.
- [4] R. Kaur, S. Singh, and H. Kumar, "A modern overview of several countermeasures for the rise of spam and compromised accounts in online social networks," *J. Netw. Comput. Appl.*, vol. 112, pp. 53–88, 2018.
- [5] G. Suarez-Tangil, M. Edwards, C. Peersman, G. Stringhini, A. Rashid, and M. Whitty, "Automatically dismantling online dating fraud," *IEEE Trans. Inf. Forensics Secur.*, vol. 15, pp. 1128–1137, 2020.
- [6] M. U. S. Khan, M. Ali, A. Abbas, S. U. Khan, and A. Y. Zomaya, "Segregating spammers and unsolicited bloggers from genuine experts on twitter," *IEEE Trans. Dependable Secure Comput.*, vol. 15, no. 4, pp. 551–560, Jul./Aug. 2018.
- [7] V. Balakrishnan, S. Khan, and H. R. Arabnia, "Improving cyberbullying detection using twitter users' psychological features and machine learning," *Comput. Secur.*, vol. 90, 2020, Art. no. 101710.
- [8] Georgios Kontaxis, I. Polakis, S. Ioannidis and E. P. Markatos, "Detecting social network profile cloning," 2011 IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops), Seattle, WA, USA, 2011, pp. 295–300, doi: 10.1109/PERCOMW.2011.5766886.
- [9] Monther Aldwairi, and Ali Alwahedi, "Detecting Fake News in Social Media Networks", *Procedia Computer Science*, Volume 141, 2018, Pages 215–222; <https://doi.org/10.1016/j.procs.2018.10.171>
- [10] Buket Erşahin, Özlem 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.