A

Project Report

on

SHARK TANK - STARTUP SUCCESS AND FAILURE PREDICTION USING MACHINE LEARNING ALGORITHMS

Submitted in partial fulfillment of the requirement

for the award of the Degree of

BACHELOR OF TECHNOLOGY

in

CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

Submitted by

C. Saraswathi Satya Swetha	20A31A4205
D. Sreeja	20A31A4208
K. Naga Sri Lakshmi	20A31A4214
S. Charishma	20A31A4206
U. Sai Nikitha Devi	20A31A4227

Under the esteemed guidance of

Mrs. P. SATYAVATHI M.Tech.,

Assistant Professor of CSE (AI & ML)

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

PRAGATI ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, KAKINADA, Accredited by NBA)

ADB Road, Surampalem, Near Peddapuram, Kakinada District, AP- 533437

2020-2024

A

Project Report

on

SHARK TANK - STARTUP SUCCESS AND FAILURE PREDICTION USING MACHINE LEARNING ALGORITHMS

Submitted in partial fulfillment of the requirement

for the award of the Degree of

BACHELOR OF TECHNOLOGY

in

CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

Submitted by

C. Saraswathi Satya Swetha 20A31A4205

D. Sreeja 20A31A4208

K. Naga Sri Lakshmi 20A31A4214

S. Charishma 20A31A4206

U. Sai Nikitha Devi 20A31A4227

Under the esteemed guidance of

Mrs. P. SATYAVATHI M.Tech.,

Assistant Professor of CSE (AI & ML)

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

PRAGATI ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, KAKINADA, Accredited by NBA)

ADB Road, Surampalem, Near Peddapuram, Kakinada District, AP- 533437

2020-2024

PRAGATI ENGINEERING COLLEGE

(AUTONOMOUS)

(Approved by AICTE, Permanently Affiliated to JNTUK, Kakinada, Accredited by NBA)

ADB Road, Surampalem, Near Peddapuram, Kakinada District, AP-533437

CERTIFICATE

DEPARTMENT OF CSE (ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING)

విద్యా పరా దేవతా

Learning is Supreme Deity

This is to certify that the project report entitled **"SHARK TANK - STARTUP SUCCESS AND FAILURE PREDICTION USING MACHINE LEARNING ALGORITHMS"** is being submitted by **C. Saraswathi Satya Swetha** (20A31A4205), **D. Sreeja** (20A31A4208), **K. Naga Sri Lakshmi** (20A31A4214), **S. Charishma** (20A31A4206), **U. Sai Nikitha Devi** (20A31A4227) in partial fulfillment for the award of the Degree of **Bachelor of Technology**, during the year **2020-2024** in CSE (Artificial Intelligence and Machine Learning) of Pragati Engineering College, for the record of a bonafide work carried out by them.

Project Supervisor:	Head of the Department:
Mrs. P. SATYAVATHI M.Tech.	**Dr. A. RADHA KRISHNA** Ph.D.
Assistant Professor	Professor & Head of Department
Department of CSE (AI & ML)	Department of CSE (AI & ML)
Pragati Engineering College	Pragati Engineering College

ACKNOWLEDGEMENT

We express our gratitude and special thanks to our Project Supervisor **Mrs. P. Satyavathi**, Assistant Professor in CSE (AI & ML) Department, who deserves a special note of thanks, for having extended her full co-operation and guidance throughout the project when needed, and to materialize the project successfully.

We express our gratitude and thanks to our college Chairman **Dr. P. Krishna Rao**, and Director of Management **Sri. M. V. Haranatha Babu** and Vice President **Sri. M. Satish** for providing us all the facilities that were required to complete the project successfully.

We even wish to express our thanks to our beloved Principal **Dr. K. Satyanarayana** and Vice Principal **Dr. G. Naresh**, for fostering an environment of learning and creativity in our project to be done in the college.

We express our sense of gratitude and thanks to our Head of the Department **Dr. A. Radha Krishna,** for having shown keen interest at every stage of development of our project and for guiding us in every aspect.

We wish to express our sincere heartful thanks to all our faculty members of the Department for their valuable suggestions when needed in the project. Even we thank all the teaching and non-teaching faculty of our college.

We are thankful for the resources provided by our organization that enabled the successful execution of this project.

Finally, we would like to thank our parents and family members who motivated us and boosted our morale whenever we are in need. Even sincere thanks to all our friends who shared their thoughts whenever needed in this project work.

C. Saraswathi Satya Swetha	20A31A4205
D. Sreeja	20A31A4208
K. Naga Sri Lakshmi	20A31A4214
S. Charishma	20A31A4206
U. Sai Nikitha Devi	20A31A4227

ABSTRACT

In the dynamic and competitive milieu of startup enterprises, the ability to accurately prognosticate success transcends mere desirability to assume a pivotal role in strategic decision-making. Statistical data from 2019 accentuates the formidable challenge, with an overwhelming 90% of startups encountering adversities, thereby emphasizing the imperative of employing rigorous methodologies for foretelling a startup's trajectory towards success. This study adopts a comprehensive approach, harnessing the formidable analytical capabilities of Machine Learning (ML) algorithms such as AdaBoost, Gradient Boosting, and Random Forest to conduct an exhaustive evaluation of diverse key performance indicators (KPIs) germane to startup enterprises. The parameters subjected to scrutiny extend beyond conventional metrics such as funding rounds and investor demographics to encompass more nuanced variables, including but not limited to, the composition of founding members, industry classification, and participant demographics.

Notably, AdaBoost emerges as the preeminent algorithm, boasting a commendable predictive accuracy rate of 78%, thereby attesting to its efficacy in discerning patterns indicative of startup success. Drawing upon a rich corpus of historical data pertaining to startup enterprises, the models developed in this study furnish nuanced insights into the probabilistic landscape of startup success. By elucidating the interplay between diverse factors and the likelihood of achieving desired outcomes, our findings furnish invaluable guidance to stakeholders and prospective investors navigating the labyrinthine pathways of the startup ecosystem. Consequently, the insights gleaned from this research serve to inform strategic decision-making processes, thereby fostering more informed and judicious allocation of resources within the volatile and uncertain milieu of startup enterprises.

Keywords: *Startup, Machine Learning, AdaBoost, Gradient Boost, Random Forest*

LIST OF FIGURES

TABLE OF CONTENTS

CHAPTER – 1

INTRODUCTION

In recent years, the global landscape has witnessed an unprecedented surge in the proliferation of startups, marking a significant departure from traditional economic models and signaling a paradigm shift towards entrepreneurship-driven growth. This surge has been particularly pronounced in the aftermath of the COVID-19 pandemic, which has not only accelerated the adoption of digital technologies but also spurred a remarkable surge in entrepreneurial activity worldwide. In the United States alone, the number of applications for new business formations reached a record high of 551,657 in July 2020, representing a staggering 95% increase compared to the same period in 2019.

The success of a startup is multifaceted, encompassing various dimensions that extend beyond mere profitability. While financial viability is undoubtedly a crucial aspect, the success of a startup is also contingent upon factors such as innovation, market traction, scalability, and the ability to adapt to changing market dynamics. Successful startups are characterized by their ability to identify unmet needs or pain points in the market and develop innovative solutions that resonate with customers. Moreover, scalability is essential for startups to capitalize on growth opportunities and achieve sustainable long-term success.

However, navigating the path to success is fraught with challenges, and not all startups succeed in realizing their aspirations. The failure rate of startups in 2019 was alarmingly high, standing at around 90%. Research indicates that the failure rate varies across different stages of a startup's lifecycle, with 21.5% failing in the first year, 30% in the second year, 50% in the fifth year, and 70% by their tenth year.

The determinants of startup success are manifold and can vary depending on factors such as industry, market conditions, and the competitive landscape. Common challenges faced by startups include limited access to capital, intense competition, regulatory hurdles, and the risk of market saturation. Predicting success enables entrepreneurs, investors, and stakeholders to make informed decisions regarding resource allocation, strategic planning, and risk management.

By leveraging predictive analytics and machine learning algorithms, stakeholders can gain valuable insights into the factors that drive startup success and identify potential pitfalls that may impede growth. Moreover, predictive models can help identify promising startups early on, thereby enabling investors to allocate capital more efficiently and mitigate investment risks. Successful startups not only generate employment opportunities but also stimulate innovation, foster competition, and contribute to overall economic growth.

1.1 Objective of the Project

The overarching objective of this project is to develop a sophisticated and dynamic system that employs state-of-the-art machine learning algorithms to predict the success or failure of startup enterprises in real-time. By harnessing the formidable analytical capabilities of algorithms such as AdaBoost, Gradient Boosting, and Random Forest, the project endeavors to conduct an exhaustive evaluation of a diverse range of key performance indicators (KPIs) pertinent to the startup ecosystem.

The primary aim is to move beyond conventional metrics, such as funding rounds and investor demographics, and delve into more nuanced variables that encompass the intricate fabric of startup enterprises. This includes but is not limited to, the composition of founding members, industry classification, geographical location, market trends, and participant demographics. By embracing a holistic approach to startup evaluation, the project seeks to capture the multifaceted nature of success and failure in the entrepreneurial landscape.

Moreover, the project aspires to provide stakeholders with actionable insights derived from the analysis of historical data and real-time inputs. By elucidating the interplay between various factors and the likelihood of achieving desired outcomes, the system aims to empower entrepreneurs, investors, and policymakers with the knowledge required to make informed decisions. This includes strategic planning, resource allocation, risk mitigation strategies, and the identification of growth opportunities within the volatile and competitive startup environment.

Ultimately, the project seeks to foster a more vibrant and sustainable startup ecosystem by equipping stakeholders with the tools and insights needed to navigate the complexities of entrepreneurship. By leveraging the power of machine learning and predictive analytics, the project aims to mitigate risks, maximize opportunities, and contribute to the overall growth and prosperity of the startup community.

CHAPTER - 2

LITERATURE SURVEY

The contemporary entrepreneurial landscape has been teeming with dynamic activity, particularly underscored by the burgeoning global startup ecosystem. This phenomenon has experienced an exponential surge, especially in the aftermath of the seismic disruptions wrought by the COVID-19 pandemic, which served as a catalyst for a pronounced proliferation of novel enterprises. Notably, amidst the socioeconomic turbulence precipitated by the pandemic, a substantial cohort of individuals, compelled by circumstances such as job displacement and economic uncertainty, opted to pivot towards entrepreneurial pursuits as a means of economic sustenance and self-actualization. Evidentially, official records and empirical data gleaned from reputable sources attest to an unprecedented proliferation of new businesses throughout the tumultuous year of 2020, eclipsing antecedent annual metrics and evincing an unmistakable entrepreneurial fervor that permeated various sectors and geographies.

In the Indian context, which serves as a focal point of burgeoning entrepreneurial endeavors, the trajectory of startup proliferation has been particularly noteworthy, underpinned by a confluence of factors including favorable regulatory frameworks, access to capital, burgeoning technological infrastructure, and an increasingly robust support ecosystem comprising incubators, accelerators, and venture capital firms. The ascendancy of India's startup landscape assumes paramount significance not only as a testament to the nation's burgeoning entrepreneurial spirit but also as a barometer of its economic dynamism and innovation prowess on the global stage. Understanding the nuanced contours of the Indian startup milieu is thus imperative, affording insights into the underlying drivers propelling founders, the multifarious challenges besieging them, and the requisite support mechanisms indispensable for their sustenance and growth trajectory.

An incisive examination of data gleaned from a myriad of sources, including qualitative interviews, quantitative surveys, and extant scholarly discourse, constitutes a foundational pillar in elucidating the intricacies characterizing the Indian entrepreneurial landscape. Such empirical analyses serve not only to discern prevailing trends and patterns but also to identify emergent phenomena and nascent opportunities that may have hitherto eluded scholarly scrutiny. Moreover, they provide a robust evidentiary basis for policymakers, practitioners, and stakeholders to formulate informed strategies aimed at fostering an enabling environment conducive to entrepreneurial dynamism and innovation.

Of particular intrigue within the realm of contemporary entrepreneurship is the burgeoning prevalence of startups helmed by individuals hailing from diverse sociocultural milieus. These nascent enterprises, often characterized by their agility, innovation, and propensity for disruption, epitomize the democratization of entrepreneurship, transcending traditional barriers and hierarchies to empower individuals from marginalized or underrepresented backgrounds. The motivational substrates underpinning the entrepreneurial aspirations of such individuals, as well as the attendant vicissitudes and challenges they confront along their entrepreneurial journey, constitute fertile terrain for scholarly inquiry and empirical investigation.

Not withstanding the palpable exuberance surrounding entrepreneurial ventures, the harsh veracity of startup failure looms large as a sobering reminder of the inherent risks and uncertainties endemic to entrepreneurial endeavors. Scrutiny into the determinants underpinning startup demise not only sheds light on the pitfalls and challenges confronting nascent ventures but also underscores the imperative for a nuanced comprehension of the elements engendering entrepreneurial success. Leveraging methodological tools such as Machine Learning algorithms, coupled with an exhaustive examination of diverse facets of startup gestation, engenders a repository of discernments pivotal for aspiring entrepreneurs, investors, policymakers, and other stakeholders alike. By distilling empirical findings into actionable insights, scholars and practitioners can inform evidence-based interventions aimed at mitigating risk, enhancing resilience, and catalyzing success within the entrepreneurial ecosystem. Ultimately, a nuanced understanding of the dynamics undergirding entrepreneurial success or failure furnishes a foundational substrate for the cultivation of a nurturing ecosystem conducive to entrepreneurial endeavor, thereby fostering innovation, economic growth, and societal prosperity.

CHAPTER - 3

SYSTEM ANALYSIS

3.1 Existing System

The prior research landscape in the field of startup success prediction underwent extensive examination by scholars such as Pan et al. and Arroyo et al., who scrutinized various aspects of startup outcomes using diverse methodological approaches. Pan et al. focused on forecasting specific events within startup trajectories, such as mergers, acquisitions, and initial public offerings, employing algorithms like Logistic Regression, Random Forests, and K Nearest Neighbors. Notably, K Nearest Neighbors emerged as the best performer, displaying superior F1 scores.

In contrast, Arroyo et al. expanded their investigation to include a broader range of startup outcomes, including subsequent funding rounds and closures. They employed algorithms such as Support Vector Machines, Decision Trees, Random Forests, Extremely Randomized Trees, and Gradient Tree Boosting. Remarkably, Gradient Tree Boosting achieved an accuracy level of approximately 82%.

However, the prevailing trend in startup success prediction leaned heavily on conventional statistical models and a limited set of predictors. This reliance is evident in the repeated use of methodologies like K-nearest neighbors (KNN) to forecast success rates among established firms. While effective for mature organizations, these conventional approaches often struggle with the complexities of the dynamic and heterogeneous early-stage startup ecosystems. The limitations of these methodologies include difficulties in addressing the inherent uncertainty and ambiguity characteristic of the initial stages of venture development. Consequently, such constraints lead to inaccurate predictions and suboptimal decision-making processes.

3.2 Proposed System

Our proposed system represents a significant advancement over existing methodologies, offering several advantages that enhance its effectiveness in predicting startup success. By incorporating AdaBoost, Gradient Boosting, and Random Forest algorithms, our system leverages the strengths of each approach. These ensemble methods are known for their ability to handle complex relationships within data and mitigate overfitting, resulting in more accurate predictions compared to single algorithms. AdaBoost, one of the key components of our system, has demonstrated the highest accuracy among the employed algorithms, achieving a notable 78%. This superior performance is indicative of its robustness in capturing the intricate patterns and dynamics inherent in startup data.

Unlike previous models that relied on a limited set of predictors, our approach considers a wide range of parameters such as investor count, founder demographics, funding details, industry type, and investor profiles. By integrating diverse factors that influence startup success, our model provides a holistic view of the startup ecosystem, leading to more reliable predictions. The output format of our system is designed to be clear and straightforward, providing stakeholders and investors with concise information about the likelihood of a startup's success or failure. This transparency enables informed decision-making and facilitates proactive measures to mitigate risks or capitalize on opportunities.

Leveraging extensive historical data, our models offer nuanced insights into the potential success of startups. By analyzing past trends and patterns, our system identifies key drivers of success and highlights areas of concern, enabling stakeholders to make strategic adjustments and optimize their investment strategies. The dynamic nature of the startup landscape demands adaptive prediction models capable of accommodating evolving trends and market conditions. Our system's flexibility makes it well-suited to navigate the complexities of the ever-changing startup ecosystem, ensuring relevance and reliability over time.

Modules:

1. Upload Dataset
2. Ada Boost Algorithm
3. Gradient Boost Algorithm
4. Random Forest Algorithm
5. Comparison Graph

1. Upload Dataset

In this module, we can select and upload the ‘dataset’ folder and then click on the ‘Select Folder’ button to load the dataset.

2. Ada Boost Algorithm

It iteratively corrects the errors of the weak classifiers by giving more weight to the misclassified data points.

3. Gradient Boost Algorithm

It sequentially trains multiple weak models to correct errors made by preceding models, optimizing a differentiable loss function.

4. Random Forest Algorithm

It builds multiple decision trees and merges their predictions to improve accuracy and reduce overfitting.

5. Comparison Graph

In the graph, the x-axis represents Name of the Algorithm and the y-axis represents Accuracy, Recall, Precision and F1 score of those algorithms.

3.3. Process Model Used With Justification

3.3.1 SDLC

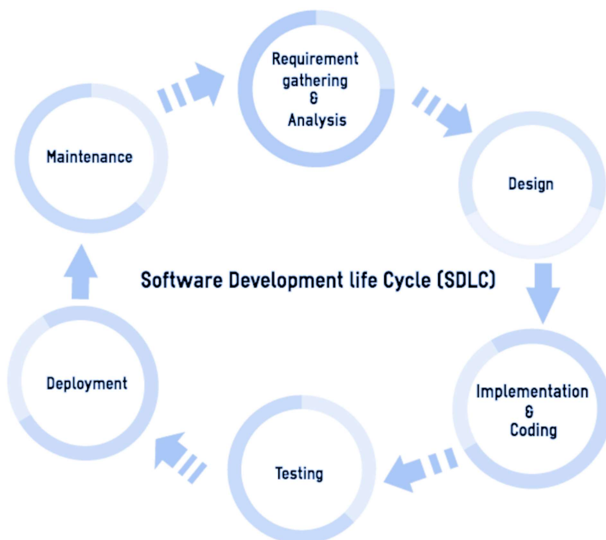


Figure 1: SDLC

- Requirement Gathering & Analysis
- Design
- Development (Coding)
- Testing
- Deployment
- Maintenance

3.3.2 Requirements Gathering & Analysis stage

The requirements gathering process takes as its input the goals identified in the high-level requirements section of the project plan. Each goal will be refined into a set of one or more requirements.

These requirements define the major functions of the intended application, define operational data areas and reference data areas, and define the initial data entities. Major functions include critical processes to be managed, as well as mission critical inputs, outputs and reports. A user class hierarchy is developed and associated with these major functions, data areas, and data entities. Each of these definitions is termed a Requirement. Requirements are identified by unique requirement identifiers and, at minimum, contain a requirement title and textual description.

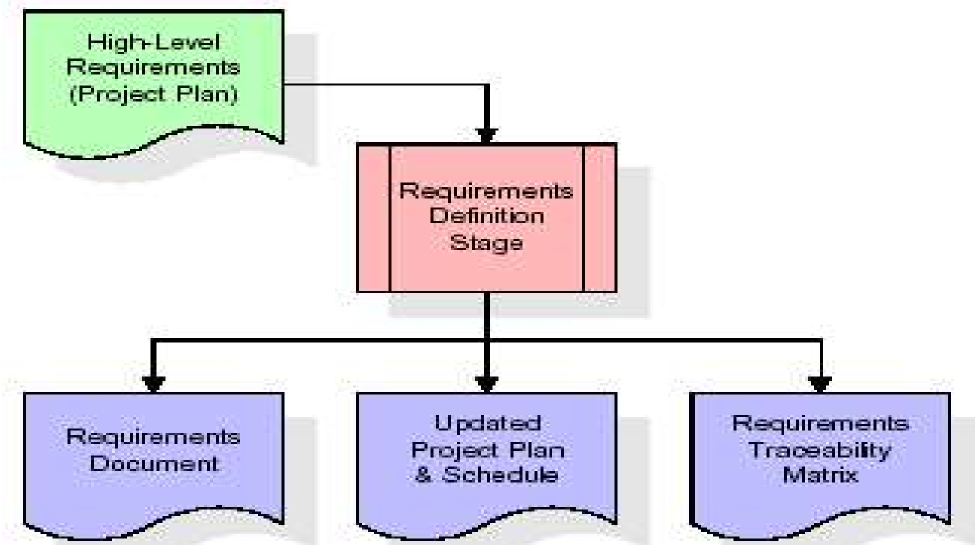


Figure 2: Requirements Gathering stage

These requirements are fully described in the primary deliverables for this stage: the Requirements Document and the Requirements Traceability Matrix (RTM). The requirements document contains complete descriptions of each requirement, including diagrams and references to external documents as necessary. Note that detailed listings of database tables and fields are *not* included in the requirements document.

The title of each requirement is also placed into the first version of the RTM, along with the title of each goal from the project plan. The purpose of the RTM is to show that the product components developed during each stage of the software development lifecycle are formally connected to the components developed in prior stages.

In the requirements stage, the RTM consists of a list of high-level requirements, or goals, by title, with a listing of associated requirements for each goal, listed by requirement title. In this hierarchical listing, the RTM shows that each requirement developed during this stage is formally linked to a specific product goal. In this format, each requirement can be traced to a specific product goal, hence the term requirements traceability.

The outputs of the requirements definition stage include the requirements document, the RTM, and an updated project plan.

- ♦ Feasibility study is all about identification of problems in a project.
- ♦ No. of staff required to handle a project is represented as Team Formation, in this case only modules with individual tasks will be assigned to employees who are working for that project.
- ♦ Project Specifications are all about representing various possible inputs submitting to the server and corresponding outputs along with reports maintained by the administrator.

Analysis

The planning stage establishes a bird's eye view of the intended software product, and uses this to establish the basic project structure, evaluate feasibility and risks associated with the project, and describe appropriate management and technical approaches.

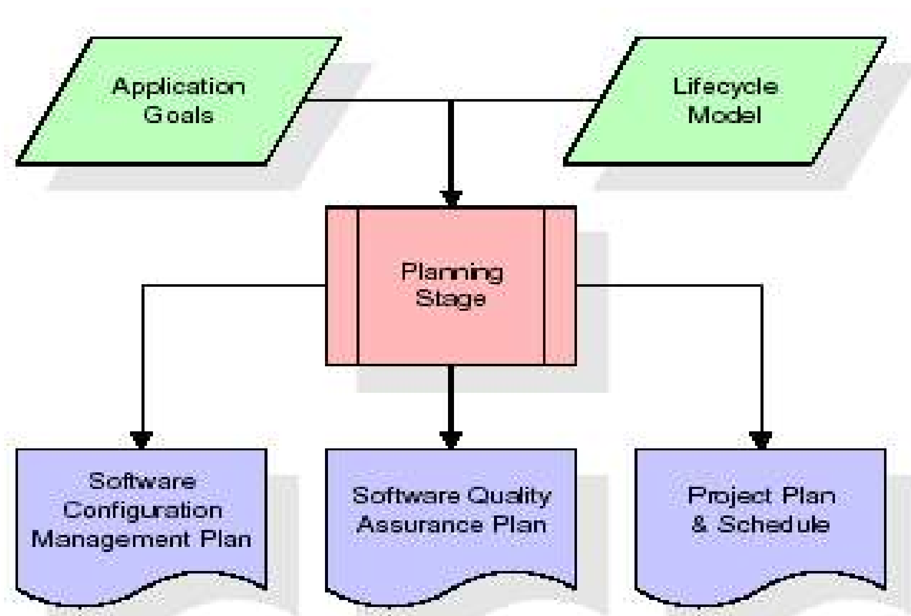


Figure 3: Analysis Stage

The most critical section of the project plan is a listing of high-level product requirements, also referred to as goals. All of the software product requirements to be developed during the requirements definition stage flow from one or more of these goals. The minimum information for each goal consists of a title and textual description, although additional information and references to external documents may be included.

3.3.3 Design Stage

The design stage takes as its initial input the requirements identified in the approved requirements document. For each requirement, a set of one or more design elements will be produced as a result of interviews, workshops, and/or prototype efforts. Design elements describe the desired software features in detail, and generally include functional hierarchy diagrams, screen layout diagrams, tables of business rules, business process diagrams, pseudo code, and a complete entity-relationship diagram with a full data dictionary. These design elements are intended to describe the software in sufficient detail that skilled programmers may develop the software with minimal additional input.

When the design document is finalized and accepted, the RTM updated to show that each design element is formally associated with a specific requirement. The outputs are the design stage document, an updated RTM, and updated project plan.

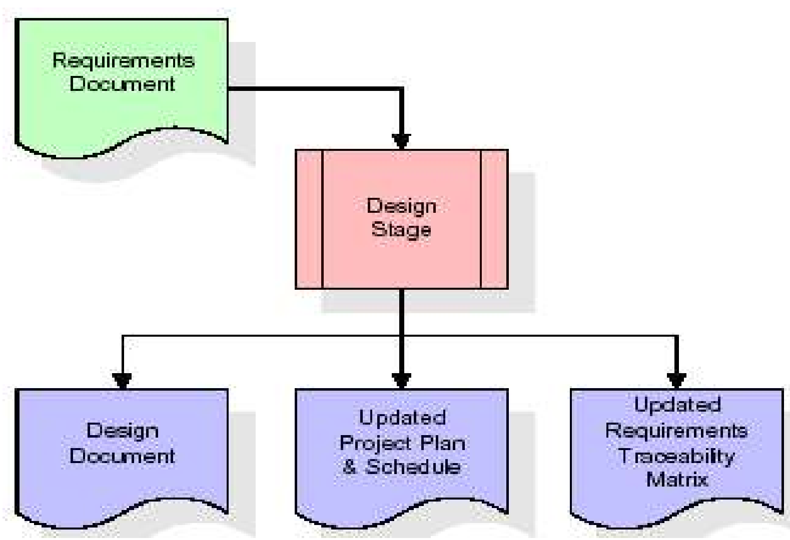


Figure 4: Designing State

3.3.4 Development (Coding) Stage

The development stage takes as its primary input the design elements described in the approved design document. For each design element, a set of one or more software artifacts will be produced. Software artifacts include but are not limited to menus, dialogs, and data management forms, data reporting formats, and specialized procedures and functions. Appropriate test cases will be developed for each set of functionally related software artifacts, and an online help system will be developed to guide users in their interactions with the software.

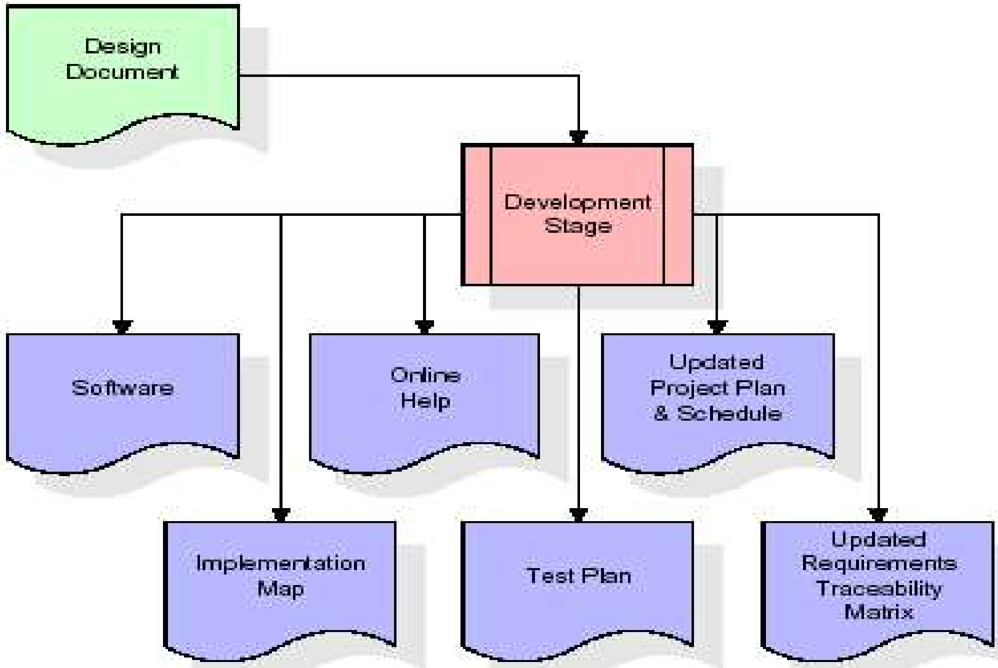


Figure 5: Development Stage

The RTM will be updated to show that each developed artifact is linked to a specific design element, and that each developed artifact has one or more corresponding test case items. At this point, the RTM is in its final configuration. The outputs of the development stage include a fully functional set of software that satisfies the requirements and design elements previously documented, an online help system that describes the operation of the software, an implementation map that identifies the primary code entry points for all major system functions, a test plan that describes the test cases to be used to validate the correctness and completeness of the software, an updated RTM, and an updated project plan.

3.3.5 Integration & Test Stage

During the integration and test stage, the software artifacts, online help, and test data are migrated from the development environment to a separate test environment. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite confirms a robust and complete migration capability. During this stage, reference data is finalized for production use and production users are identified and linked to their appropriate roles. The final reference data (or links to reference data source files) and production user list are compiled into the Production Initiation Plan.

The outputs of the integration and test stage include an integrated set of software, an online help system, an implementation map, a production initiation plan that describes reference data and production users, an acceptance plan which contains the final suite of test cases, and an updated project plan.

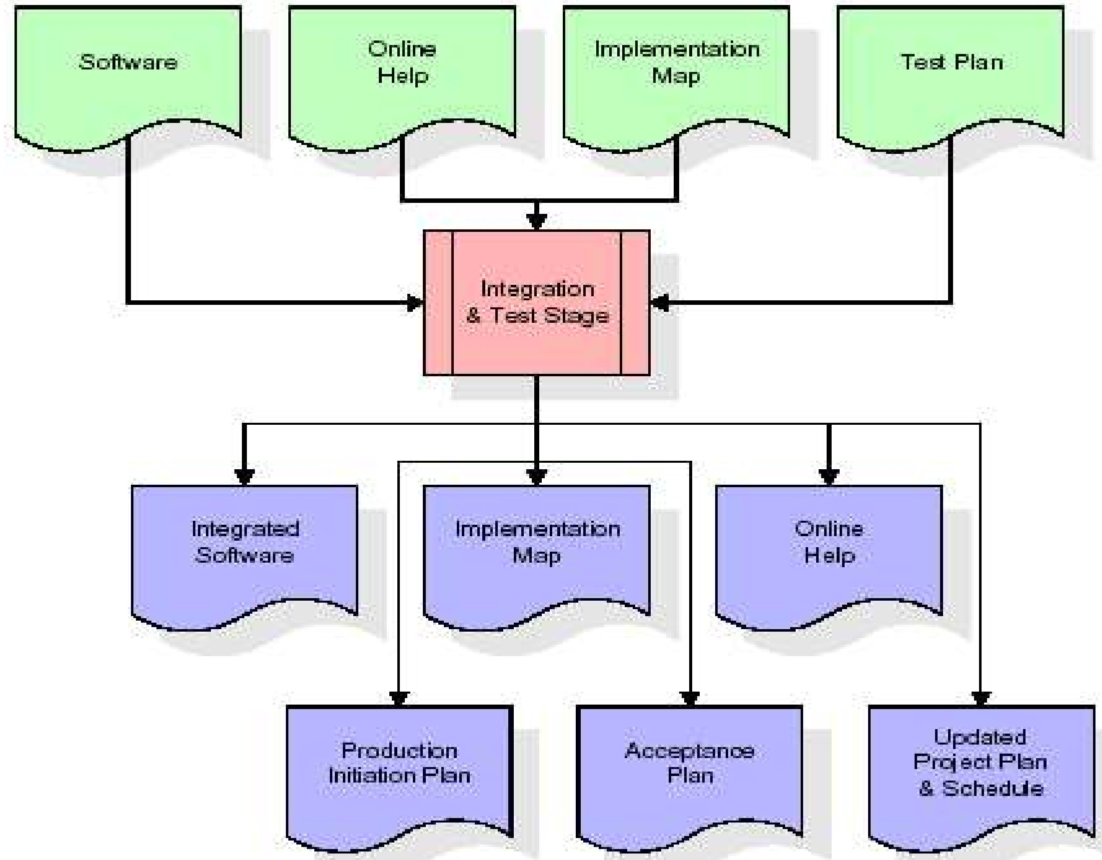


Figure 6: Integration & Testing Stage

Installation & Acceptance Test

During the installation and acceptance stage, the software artifacts, online help, and initial production data are loaded onto the production server. At this point, all test cases are run to verify the correctness and completeness of the software. Successful execution of the test suite is a prerequisite to acceptance of the software by the customer.

After customer personnel have verified that the initial production data load is correct and the test suite has been executed with satisfactory results, the customer formally accepts the delivery of the software.

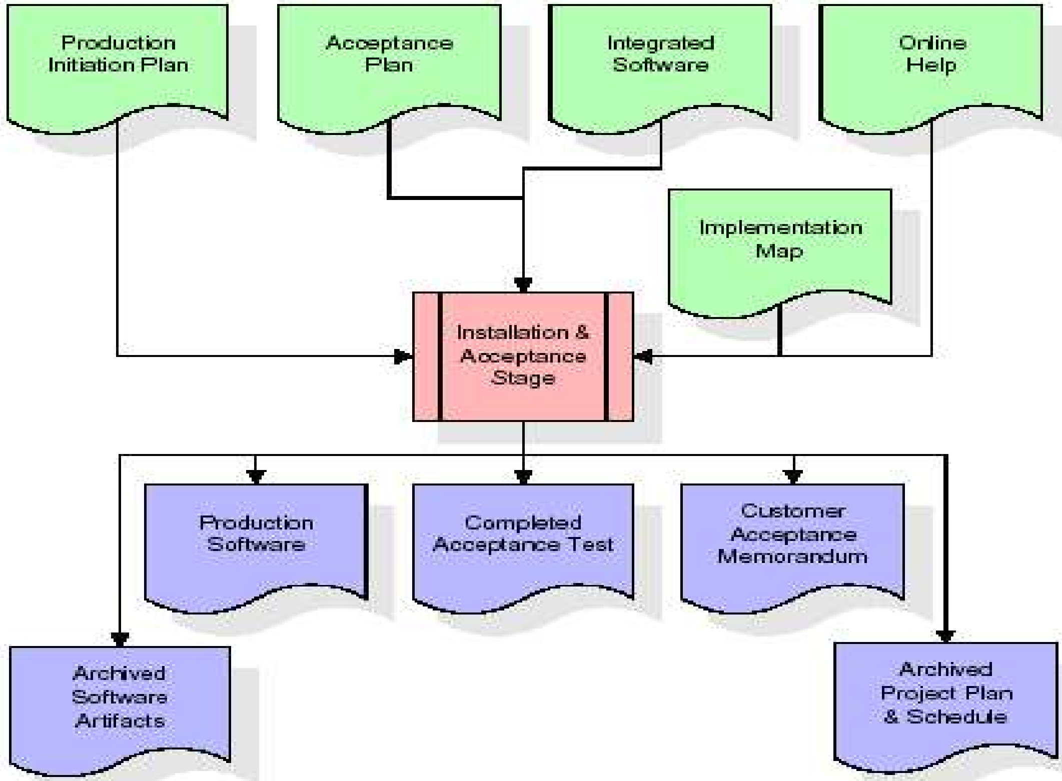


Figure 7: Installation & Acceptance Test

The primary outputs of the installation and acceptance stage include a production application, a completed acceptance test suite, and a memorandum of customer acceptance of the software. Finally, the PDR enters the last of the actual labor data into the project schedule and locks the project as a permanent project record. At this point the PDR "locks" the project by archiving all software items, the implementation map, the source code, and the documentation for future reference.

3.3.6 Deployment

This phase involves deploying the tested software in the live or production environment. Usually, it is preceded by User-acceptance testing (UAT). During UAT, a production-like system is provided to the customers who test it along with developers. Once the customer is satisfied, they give a sign-off, and developers deploy the system to production.

3.3.7 Maintenance

Outer rectangle represents maintenance of a project, Maintenance team will start with requirement study, understanding of documentation, and later employees will be assigned work and they will undergo training in that particular assigned category.

3.4. Software Requirement Specification

3.4.1. Overall Description

A Software Requirements Specification (SRS) – a requirements specification for a software system is a complete description of the behavior of a system to be developed. It includes a set of use cases that describe all the interactions the users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Nonfunctional requirements are requirements which impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System requirements specification: A structured collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Within the systems development lifecycle domain, the BA typically performs a liaison function between the business side of an enterprise and the information technology department or external service providers. Projects are subject to three sorts of requirements:

- Business requirements describe in business terms what must be delivered or accomplished to provide value.
- Product requirements describe properties of a system or product.
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify.

There are aspects in the feasibility study portion of the preliminary investigation:

- **Economic Feasibility:**

  A system can be developed technically and that will be used if installed must still be a good investment for the organization. In the economical feasibility, the development cost in creating the system is evaluated against the ultimate benefit derived from the new systems. Financial benefits must equal or exceed the costs. The system is economically feasible.

- **Operational Feasibility:**

  Proposed projects are beneficial only if they can be turned into an information system. That will meet the organization’s operating requirements. Operational feasibility aspects of the project are to be taken as an important part of the project implementation.

- **Technical Feasibility:**

  Earlier no system existed to cater to the needs of 'Secure Infrastructure Implementation System'. The current system developed is technically feasible. It is a web based user interface for audit workflow at NIC-CSD. Thus it provides easy access to the users.

3.4.2. External Interface Requirements

- **User Interface**

  The user interface of this system is a user-friendly Django Graphical User Interface.

- **Hardware Interfaces**

  The interaction between the user and the console is achieved through python capabilities.

- **Software Interfaces**

  The required software is Python

- **Operating Environment**

  Windows 10/11

HARDWARE REQUIREMENTS:

- Processor - Intel Core i5 or AMD Ryzen 5 processor
- Speed - 2.5 Ghz
- RAM - 8 GB
- Hard Disk - 256 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - Full HD (1920 x 1080) resolution

SOFTWARE REQUIREMENTS:

- Operating System - Windows 10 or macOS 10.14 (or higher)
- Programming Language - Python (Python 3.10)

CHAPTER - 4

SYSTEM DESIGN

In this chapter, we delve into the design architecture of our project, which aims to predict startup success and failure using machine learning algorithms. The system design is crucial as it lays the foundation for the implementation and functionality of our project. We employ various modeling techniques and notations to illustrate the architecture and components of our system.

The design phase is a critical stage in the development lifecycle, where we translate the project requirements into a structured blueprint. Through careful planning and consideration of design principles, we aim to create a robust, scalable, and user-friendly system that meets the needs of our stakeholders.

This chapter provides an overview of the system design, including data flow diagrams, UML diagrams, and deployment diagrams. Each diagram serves a specific purpose in modeling different aspects of the system architecture and functionality.

We begin by presenting how UML diagrams are utilized to describe the system structure and behavior, followed by an explanation of the Data Flow Diagram (DFD), which depicts the flow of information within the system. Additionally, we discuss the use of deployment diagrams to illustrate the physical hardware and software deployment architecture.

Through these diagrams and notations, we aim to provide a comprehensive understanding of the system design, laying the groundwork for the subsequent implementation and testing phases.

UML Diagrams:

The Unified Modeling Language (UML) is a standardized modeling language extensively employed in object-oriented software engineering. Governed by the Object Management Group, UML serves as a universal framework for expressing system designs and architectures. Its versatility lies in the variety of diagrams it offers, each tailored to capture specific aspects of a system.

For instance, Use Case Diagrams depict user interactions and system functionalities, while Class Diagrams illustrate class relationships and structures within the system. Sequence Diagrams showcase the sequence of operations or messages exchanged between objects, aiding in understanding system behavior over time. Lastly, Activity Diagrams provide insights into workflow processes and the sequential flow of activities within the system.

By leveraging these diverse UML diagrams, software engineers can effectively communicate and analyze different perspectives of the system throughout the software development lifecycle.

The Unified Modeling Language (UML) aims to become a common language for modeling object-oriented computer software, comprising two major components: a Meta-model and a notation. In its current form, UML serves as a standard language for specifying, visualizing, constructing, and documenting software system artifacts, alongside business modeling and other non-software systems. It embodies a collection of proven engineering practices, particularly effective in modeling large and complex systems, thus playing a crucial role in object-oriented software development processes. Utilizing predominantly graphical notations, UML facilitates the expression of software project designs.

Goals of UML:

- Provide users a ready-to-use, expressive visual modeling language to develop and exchange meaningful models.
- Offer extendibility and specialization mechanisms to augment core concepts.
- Ensure independence from particular programming languages and development processes.
- Establish a formal basis for understanding the modeling language.
- Stimulate the growth of the object-oriented tools market.
- Support higher-level development concepts such as collaborations, frameworks, patterns, and components.
- Integrate best practices into software modeling endeavors.

4.1 Class Diagram

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 serves as a blueprint for the system's object-oriented design, providing insights into the entities and their relationships within the system.

The class diagram is the main building block of object-oriented modeling, utilized for both general conceptual modeling of the system and detailed modeling translating the models into programming code. Additionally, class diagrams can be employed for data modeling, representing the classes that correspond to main objects, interactions in the application, and classes to be implemented.

In a class diagram, classes are represented with boxes, which are divided into three parts:

Class Name (Upper Part): The upper part of the box holds the name of the class, providing a clear identifier for the class entity.

Attributes (Middle Part): The middle part of the box contains the attributes of the class, representing the data or properties associated with the class. These attributes define the characteristics or state of the objects instantiated from the class.

Operations/Methods (Bottom Part): The bottom part of the box lists the methods or operations that the class can perform or undertake. These methods encapsulate the behavior or functionality of the class, defining the actions that objects of the class can execute.

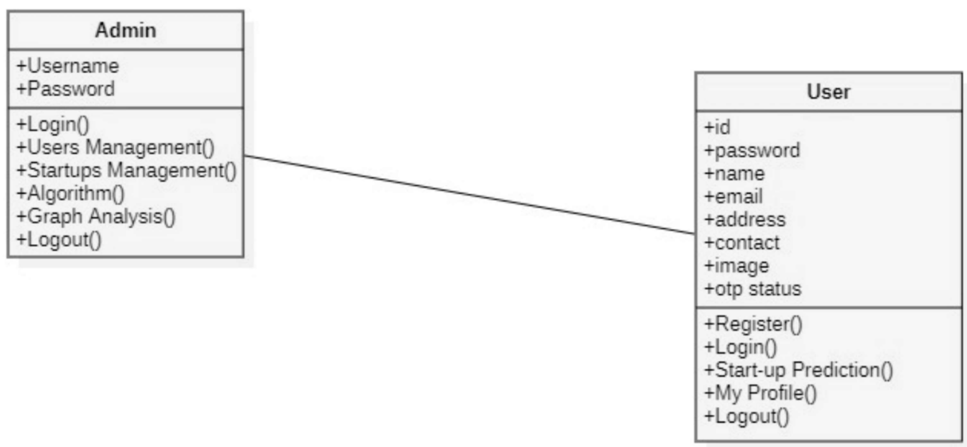


Figure 8: Class Diagram

4.2 Use Case Diagram

A use case diagram in the Unified Modeling Language (UML) is a behavioral diagram created from a Use-case analysis, aiming to provide a graphical overview of the functionality offered by a system. It illustrates the interactions between actors (users or external systems) and the system, depicting their goals, known as use cases, and any dependencies between them.

At its core, a use case diagram portrays a user's interaction with the system and outlines the specifications of each use case. It showcases different types of users and their respective interactions with the system, providing a clear understanding of the system's functionality from a user's perspective.

Use Case Diagrams help to identify system features, functionalities, and dependencies, facilitating effective communication between stakeholders and development teams.

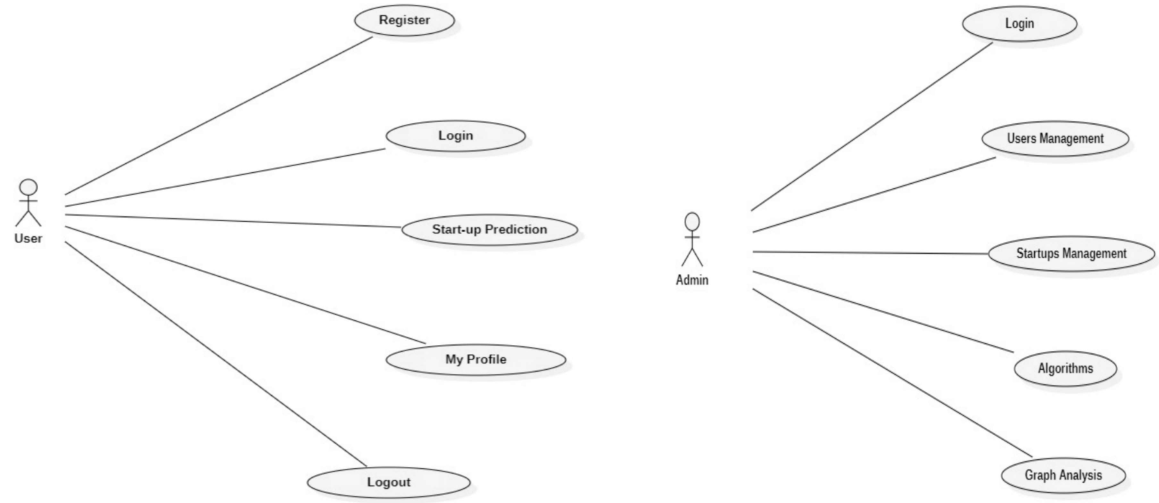
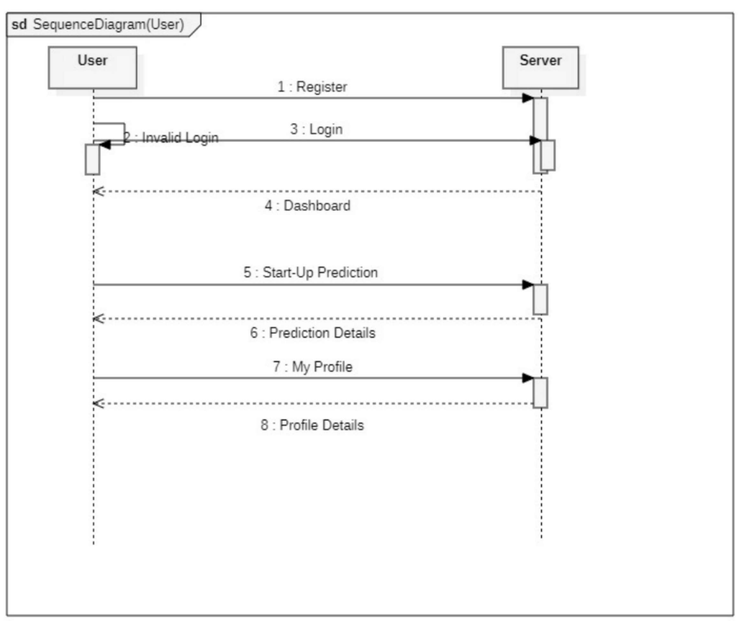


Figure 9: Use Case Diagram (User & Admin)

4.3 Sequence Diagram

A sequence diagram is an interaction diagram that portrays how processes within a system interact with each other in a chronological order. It serves as a visual representation of the object interactions arranged in a time sequence. This diagram depicts the objects and classes involved in a scenario, along with the sequence of messages exchanged between them to execute the functionality of the scenario. Sequence diagrams are crucial for understanding the dynamic behavior of a system during runtime. They illustrate the flow of messages and the order of operations between objects or components within the system.



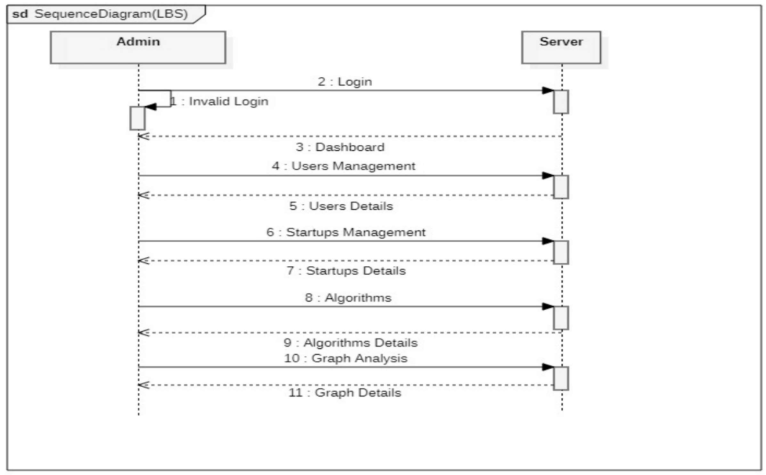


Figure 10: Sequence Diagram (User & Admin)

4.4 Component Diagram

A component diagram in the Unified Modeling Language (UML) illustrates how components are interconnected to form larger components or software systems. It provides a visual representation of the structure of complex systems, showcasing how different parts are organized and interact with each other. Components are connected using assembly connectors, which establish the relationship between a service consumer and a service provider. This diagram helps in understanding how various components collaborate to fulfill the functionalities of the system.

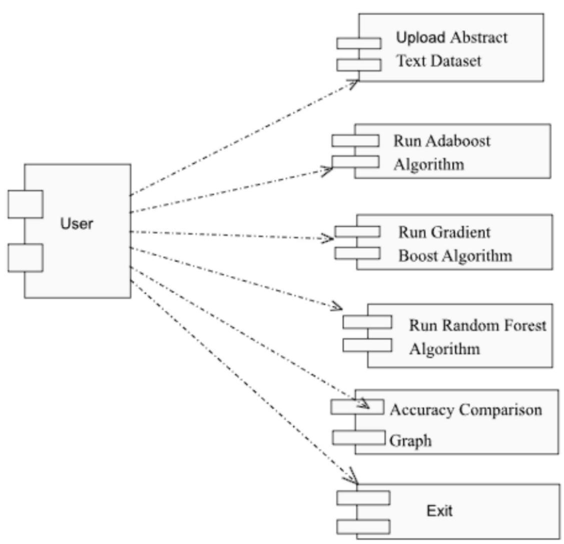


Figure 11: Component Diagram

4.5 Deployment Diagram

A deployment diagram in the Unified Modeling Language (UML) specifies the physical hardware on which the software system will execute and how the software is deployed on that hardware. It maps software components to the devices that will execute them, depicting the software architecture in relation to the physical system architecture. In distributed systems, it models the distribution of software across physical nodes. The primary purpose of a deployment diagram is to describe how software is deployed into the hardware system, illustrating the interaction between software and hardware to execute the complete functionality of the system. It facilitates understanding of software-to-hardware and hardware-to-software interactions.

Deployment diagrams consist of nodes representing hardware components and artifacts representing software components. Nodes may include web servers, application servers, or database servers, while artifacts could be web applications or databases. The diagram shows how artifacts are allocated to nodes and how they are connected, indicating the communication paths between nodes. In a deployment diagram, nodes are depicted as boxes, with artifacts allocated to each node represented as rectangles within the boxes. Nodes may have sub-nodes, shown as nested boxes. A single node can represent multiple physical nodes, such as a cluster of database servers.

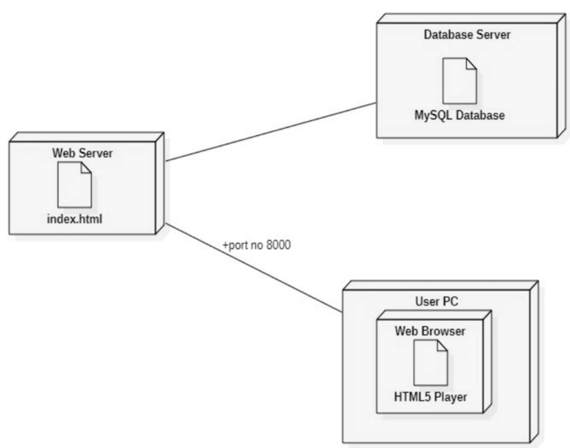


Figure 12: Deployment Diagram

4.6 Activity Diagram

Activity diagrams are essential in UML for depicting the dynamic aspects of a system. They serve as flowcharts to illustrate the flow from one activity to another within the system. An activity, in this context, refers to an operation of the system. Thus, the control flow is represented as transitions from one operation to another, which can be sequential, branched, or concurrent. These diagrams represent stepwise workflows of activities and actions within the system, enabling the depiction of choice, iteration, and concurrency. By visually mapping the flow of control and decision-making processes, activity diagrams aid in understanding the overall workflow and business logic of the system.

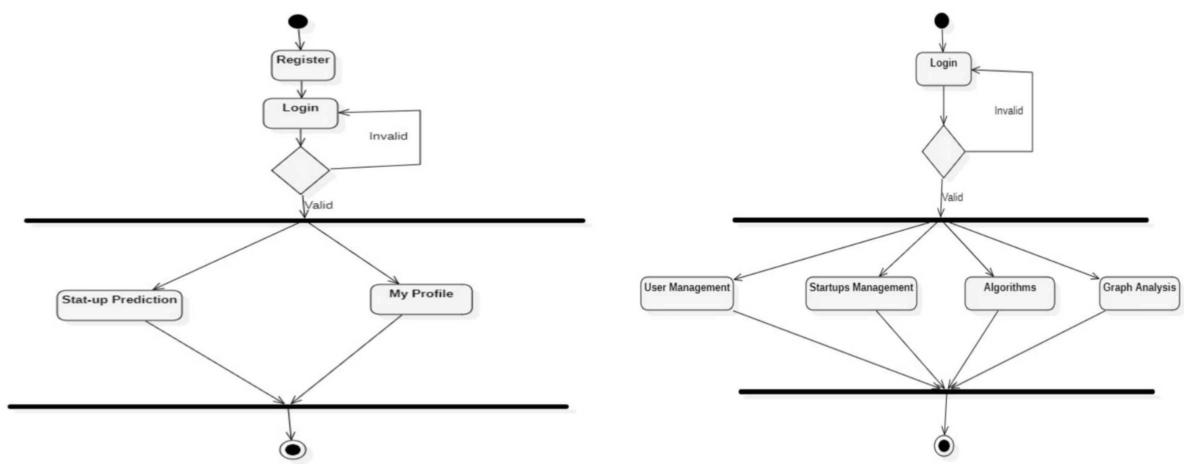


Figure 13: Activity Diagram (User & Admin)

4.7 Data Flow Diagram (DFD)

The Data Flow Diagram (DFD) is a visual depiction of how data moves through a system, illustrating input sources, processing steps, and output destinations. It provides a clear roadmap of the information flow within the system, offering insights into data handling processes and dependencies. Commonly referred to as a Process Model, the DFD showcases both business and technical processes, demonstrating the support of external data, the flow between processes, and the resulting outcomes. DFD serves as a simplifying tool, representing system components including processes, data, external entities, and information flows.

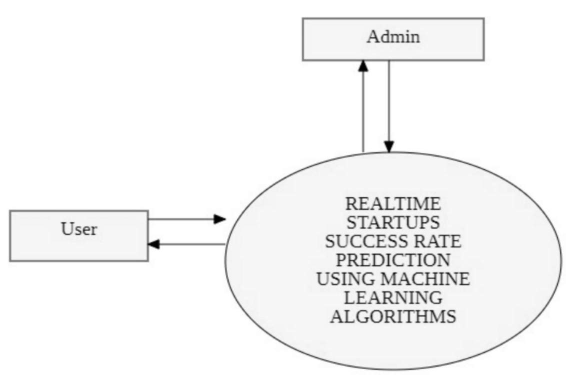


Figure 14: Data Flow Diagram

CHAPTER - 5

IMPLEMENTATION

5.1 Django

Django is a web application framework written in Python programming language. It is based on the MVT (Model View Template) design pattern. Django is very demanding due to its rapid development feature. It takes less time to build an application after collecting client requirements. By using Django, we can build web applications in very less time. Django is designed in such a manner that it handles much of the configuration automatically, so we can focus on application development only.

History:

Django was designed and developed by Lawrence journal world in 2003 and publicly released under BSD license in July 2005. Currently, DSF (Django Software Foundation) maintains its development and release cycle. Django was released on 21, July 2005. Its current stable version is 2.0.3 which was released on 6 March, 2018.

Components of Django:

MVT Architecture

Django's approach is sometimes called **Model-View-Template (MVT)** but it is really a 4-part pattern that also incorporates URL configuration. Something like **Model-View-Template-URL (MVTU)** would be a more accurate description:

- **Model:** Manages data and core business logic
- **View**: Describes *which* data is sent to the user but not its presentation
- **Template:** Presents the data as HTML with optional CSS, JavaScript, and static assets
- **URL Configuration:** Regular expression components configured to a View

ORM (Object - Relational Manager)

- Django includes its own ORM, which maps Python objects to database tables.
- You define models as Python classes, and Django handles the creation and manipulation of corresponding database tables.

Admin Interface

- Django provides a built-in admin interface for managing site content.
- You can register your models with the admin site to perform CRUD operations without writing custom views or forms.

Forms

- Django includes form handling functionality, simplifying the process of collecting and validating user input.
- You define forms as Python classes, which Django can render in templates and process submitted data.

Security Features

- Django includes several security features out of the box to help developers build secure web applications.
- These features include protection against common web vulnerabilities like SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), etc.
- Django also provides tools for user authentication, authorization, and session management.

URL Routing

- Django uses a URL routing mechanism to map incoming HTTP requests to the appropriate view functions.
- URL patterns are defined using regular expressions or path converters, allowing for flexible URL configurations.
- URL routing is typically done in the **urls.py** module of each Django app.

Middleware

- Middleware is a framework of hooks into Django's request/response processing.
- It allows you to modify request objects, response objects, or even short-circuit the processing of a request.
- Middleware components are defined in settings.py and can be used for tasks like authentication, logging, or modifying HTTP headers.

Django REST Framework (DRF)

- DRF is a powerful toolkit for building Web APIs in Django.
- It provides serializers for converting complex data types to and from native Python data types, views for handling HTTP requests, and authentication mechanisms.
- DRF simplifies the creation of RESTful APIs, making it easier to build web services that communicate with other systems or clients.

Features Of Django

Rapid Development

Django was created with the goal of creating a framework that would allow developers to build web applications in less time. The project implementation phase takes a long time, but Django makes it happen quickly.

Enhance Security

Django's security goes beyond its explicit security features: the extensive experience and expertise of the Django user base bolster security efforts. You run the risk of accidentally introducing a security vulnerability into your module if you build your entire web app from scratch. You can be more confident that Django packages will protect your data because they are widely used, open-source, and well-reviewed by web developers.

Versatile

Django is a versatile framework that can be used to create applications in a variety of domains. Companies are now using Django to create a variety of applications, such as content management systems, social networking sites, and scientific computing platforms, among others.

Open Source

Django is a web application framework that is free and open source. It is freely available to the public. The source code is available for download from the public repository. The total cost of developing an application is reduced by using open source.

Working of Django:

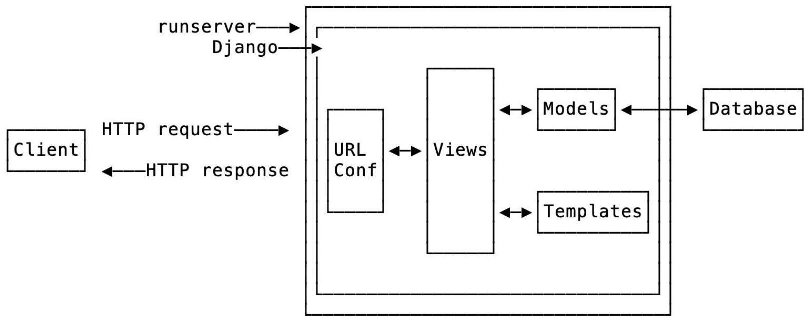


Figure 15: Django Working Diagram

Django Installation :

1. Open a new command line shell or use the built-in terminal in VS Code
2. **Navigate to Directory:**

   cd onedrive\desktop\code

3. **Create Project Directory:**

   mkdir helloworld

   cd helloworld

4. **Create Virtual Environment and Install Django:**
   - Create a virtual environment named ".venv":

     python -m venv .venv

   - Activate the virtual environment:

     .venv\Scripts\Activate.ps1

   - Install Django using pip:

     python -m pip install django~=4.2.0

5. **Create a Django project:**

    - Use the django-admin command to create a new project:

      django-admin startproject project_name

    - This will create a directory with the specified project name containing necessary files and directories.

6. **Create Django apps:**

    - Django projects are made up of multiple apps. Each app serves a specific purpose.
    - Create a new app using the manage.py script:

      python manage.py startapp app_name

7. **Define models:**

    - Models represent your data structure. Define models in the models.py file of your app.
    - Each model is a Python class that subclasses django.db.models.Model.

8. **Create database tables:**

    - Run migrations to create database tables based on your models:

      python manage.py makemigrations

      python manage.py migrate

9. **Create views:**

    - Views handle the logic of your application. They are Python functions that take a web request and return a web response.

10. **Define URLs:**

    - Map URLs to views in the urls.py file of your app.
    - You can have project-level URLs (in urls.py of your project) and app-level URLs (in urls.py of your app).

11. Create templates:

- Templates define the HTML structure of your web pages.
- Create HTML templates in the templates directory of your app.

12. Static files:

- Store CSS, JavaScript, images, etc., in the static directory of your app.
- You can use these static files in your templates.

13. Run the development server:

- Start the Django development server:

  python manage.py runserver

- Access your application in a web browser at http://localhost:8000/.

Uses of Django

- Usage of the Django framework is for complete and fast development like other programming languages like C, C#, Java, etc.
- Django can handle large amounts of data effectively, manage heavy continuous traffic, and facilitate dynamic data handling.

Advantages of Django

Spotify

Developers primarily used Django technology and Python-based machine learning algorithms to develop this application. We can create our playlists, and based on the song we like to hear, there would be many recommendations pouring in based on that.

YouTube

The trending video-sharing application has incorporated the Django framework. The application has transitioned from PHP to Python, specifically the Django framework, to improve scalability. Handling such data and delivering appropriate content to billions of millions of users, Django has been preferred for addressing this.

5.2 Source Code

asgi.py

```
import os
from django.core.asgi import get_asgi_application
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'ssp_project.settings')
application = get_asgi_application()
```

settings.py

```
from pathlib import Path
import os
BASE_DIR = Path(__file__).resolve().parent.parent
SECRET_KEY = 'django-insecure-kpbw=527t3biu15b4m=$%ed0-i!vt!v80ztao_2f2gkjb9%&@t'
DEBUG = True
ALLOWED_HOSTS = []
INSTALLED_APPS = [
    'django.contrib.admin',
    'django.contrib.auth',
    'django.contrib.contenttypes',
    'django.contrib.sessions',
    'django.contrib.messages',
    'django.contrib.staticfiles',
    'userapp',
    'mainapp',
    'adminapp',
]
```

```
MIDDLEWARE = [
    'django.middleware.security.SecurityMiddleware',
    'django.contrib.sessions.middleware.SessionMiddleware',
    'django.middleware.common.CommonMiddleware',
    'django.middleware.csrf.CsrfViewMiddleware',
    'django.contrib.auth.middleware.AuthenticationMiddleware',
    'django.contrib.messages.middleware.MessageMiddleware',
    'django.middleware.clickjacking.XFrameOptionsMiddleware',
]
ROOT_URLCONF = 'ssp_project.urls'
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR,'assets/templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
            ],
        },
    },
]
```

```
WSGI_APPLICATION = 'ssp_project.wsgi.application'
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'SSP' ,
        'USER': 'root',
        'PASSWORD': '',
        'HOST' : 'localhost',
        'PORT' : '3306'
    }
}
AUTH_PASSWORD_VALIDATORS = [
    {
        'NAME': 'django.contrib.auth.password_validation.UserAttributeSimilarityValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.MinimumLengthValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.CommonPasswordValidator',
    },
    {
        'NAME': 'django.contrib.auth.password_validation.NumericPasswordValidator',
    },
]
```

```
LANGUAGE_CODE = 'en-us'
TIME_ZONE = 'UTC'
USE_I18N = True
USE_TZ = True
STATIC_URL = 'static/'
STATICFILES_DIRS = [os.path.join(BASE_DIR,'assets/static')]
MEDIA_URL = '/media/'
MEDIA_ROOT = os.path.join(BASE_DIR,'media')
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'
```

urls.py

```
from django.contrib import admin
from userapp import views as userapp_views
from adminapp import views as adminapp_views
from mainapp import views as mainapp_views
from django.urls import path
from django.conf import settings
from django.conf.urls.static import static
```

```
urlpatterns = [
    path('admin/', admin.site.urls),
    #userapp
    path('dashboard',userapp_views.user_dashboard,name="dashboard"),
    path('predict',userapp_views.prediction,name="prediction"),
    path('user_profile',userapp_views.user_profile,name="user_profile"),
    path('user_login',userapp_views.user_login,name="user_login"),
    #adminapp
    path('admin_login',adminapp_views.admin_login,name="admin_login"),
    path('index',adminapp_views.index,name="index"),
    path('pending_users',adminapp_views.pending_users,name="pending_users"),
    path('all_users',adminapp_views.all_users,name="all_users"),
    path('upload_dataset',adminapp_views.upload_dataset,name="upload_dataset"),
    path('view_dataset',adminapp_views.view_dataset,name="view_dataset"),
    path('algorithm1',adminapp_views.gradient_boosting_classifier,name="algorithm1"),
    path('algorithm2',adminapp_views.ada_boost_classifier,name="algorithm2"),
    path('algorithm3',adminapp_views.random_forest_classifier,name="algorithm3"),
    path('gbc-runalgo/<int:id>/',adminapp_views.gbc_runalgo,name="gbc_runalgo"),
    path('ada-runalgo/<int:id>/',adminapp_views.ada_runalgo,name="ada_runalgo"),
    path('rfc-runalgo/<int:id>/',adminapp_views.rfc_runalgo,name="rfc_runalgo"),
    path('analasis',adminapp_views.graph_analasis,name="analasis"),
```

```
    #mainapp
    path('',mainapp_views.home,name="home"),
    path('prediction/<int:id>/',mainapp_views.prediction_results,name="prediction_results"),
    path('contact',mainapp_views.contact,name="contact"),
    path('user_register',mainapp_views.user_register,name="user_register"),
    path('user_logout',userapp_views.user_logout,name="user_logout"),
    #Button functions urls
    path('user_accept/<int:id>',adminapp_views.accept,name="accept"),
    path('user_reject/<int:id>',adminapp_views.reject,name="reject"),
    path('user_change_status<int:id>',adminapp_views.change_status,name="change_status"),
    path('remove_status/<int:id>',adminapp_views.remove,name="remove"),
    path('admin_logout',adminapp_views.admin_logout,name="admin_logout"),
]+static(settings.MEDIA_URL,document_root=settings.MEDIA_ROOT)
```

wsgi.py :

```
import os
from django.core.wsgi import get_wsgi_application
os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'ssp_project.settings')
application = get_wsgi_application()
```

Startup_funding.ipynb

```
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

from sklearn.ensemble  import GradientBoostingClassifier
from sklearn.metrics import accuracy_score,f1_score, recall_score, precision_score, auc, roc_auc_score, roc_curve
from sklearn.ensemble  import AdaBoostClassifier
from  xgboost import XGBClassifier
from sklearn.metrics import classification_report,confusion_matrix
from sklearn.model_selection import cross_val_score

from sklearn import metrics
from sklearn.naive_bayes import BernoulliNB,MultinomialNB
from sklearn.multiclass import OneVsRestClassifier


import time
```

```
models = [LogisticRegression(),AdaBoostClassifier(),XGBClassifier(),BernoulliNB(),
          KNeighborsClassifier(),GradientBoostingClassifier(),DecisionTreeClassifier(),RandomForestClassifier()]
Name = []
Accuracy = []
Precision = []
F1_Score = []
Recall = []
Time_Taken = []
```

```
for model in models:
    name = type(model).__name__
    Name.append(name)
    model = OneVsRestClassifier(model)
    begin = time.time()
    model.fit(x_train,y_train)
    prediction = model.predict(x_test)
    end = time.time()
    Accuracy.append(accuracy_score(prediction,y_test))
    Precision.append(precision_score(prediction,y_test,average = 'macro'))
    Recall.append(recall_score(prediction,y_test,average = 'macro'))
    F1_Score.append(f1_score(prediction,y_test,average = 'macro'))
    Time_Taken.append(end-begin)
    print(name + ' Successfully Trained')
```

```
model=RandomForestClassifier()
model = OneVsRestClassifier(model)
begin = time.time()
model.fit(x_train,y_train)
prediction = model.predict(x_test)
end = time.time()
print(accuracy_score(prediction,y_test))
print(precision_score(prediction,y_test,average = 'macro'))
print(recall_score(prediction,y_test,average = 'macro'))
print(f1_score(prediction,y_test,average = 'macro'))
```

CHAPTER - 6

TESTING

Implementation and Testing:

Implementation is one of the most important tasks in a project . It is the phase in which one has to be cautious because all the efforts undertaken during the project will be very interactive. Implementation is the most crucial stage in achieving a successful system and giving the users confidence that the new system is workable and effective. Each program is tested individually at the time of development using the sample data and has verified that these programs link together in the way specified in the program specification. The computer system and its environment are tested to the satisfaction of the user.

Implementation

The implementation phase is less creative than system design. It is primarily concerned with user training, and file conversion. The system may be requiring extensive user training. The initial parameters of the system should be modified as a result of programming. A simple operating procedure is provided so that the user can understand the different functions clearly and quickly. The different reports can be obtained either on the inkjet or dot matrix printer, which is available at the disposal of the user. The proposed system is very easy to implement. In general implementation is used to mean the process of converting a new or revised system design into an operational one.

Testing

Testing is the process where the test data is prepared and is used for testing the modules individually and later the validation given for the fields. Then the system testing takes place which makes sure that all components of the system property functions as a unit. The test data should be chosen such that it passed through all possible conditions. Actually testing is the state of implementation which aims at ensuring that the system works accurately and efficiently before the actual operation commences. The following is the description of the testing strategies, which were carried out during the testing period.

System Testing

Testing has become an integral part of any system or project especially in the field of information technology. The importance of testing is a method of justifying, if one is ready to move further, be it to check if one is capable to withstand the rigors of a particular situation cannot be underplayed and that is why testing before development is so critical. When the software is developed before it is given to the user, the software must be tested whether it is solving the purpose for which it is developed. This testing involves various types through which one can ensure the software is reliable. The program was tested logically and patterns of execution of the program for a set of data are repeated. Thus the code was exhaustively checked for all possible correct data and the outcomes were also checked.

Module Testing

To locate errors, each module is tested individually. This enables us to detect errors and correct them without affecting any other modules. Whenever the program is not satisfying the required function, it must be corrected to get the required result. Thus all the modules are individually tested from bottom up starting with the smallest and lowest modules and proceeding to the next level. Each module in the system is tested separately. For example the job classification module is tested separately. This module is tested with different jobs and its approximate execution time and the result of the test is compared with the results that are prepared manually. The comparison shows that the results proposed system works more efficiently than the existing system. Each module in the system is tested separately. In this system the resource classification and job scheduling modules are tested separately and their corresponding results are obtained which reduces the process waiting time.

Integration Testing

After the module testing, the integration testing is applied. When linking the modules there may be a chance for errors to occur, these errors are corrected by using this testing. In this system all modules are connected and tested. The testing results are very correct. Thus the mapping of jobs with resources is done correctly by the system.

Acceptance Testing

When that user finds no major problems with its accuracy, the system passes through a final acceptance test. This test confirms that the system needs the original goals, objectives and requirements established during analysis without actual execution which eliminates waste of time and money acceptance tests on the shoulders of users and management, it is finally acceptable and ready for the operation.

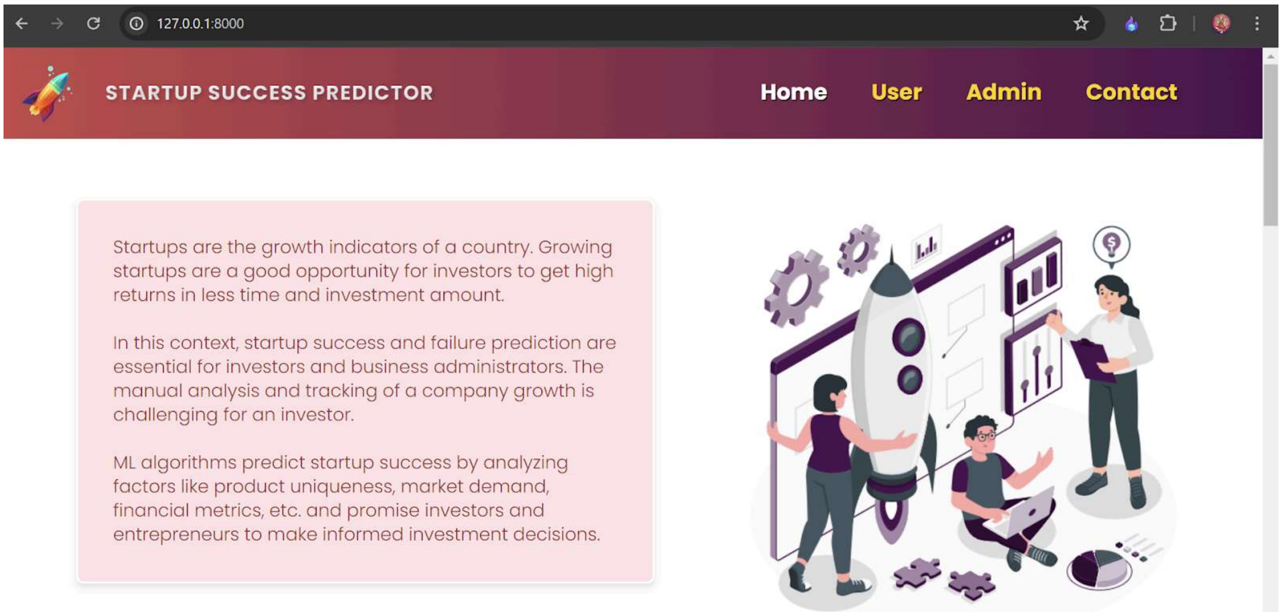
Test Case Id	Test Case Name	Test Case Description	Test Steps			Test Case Status	Test Priority
			Step	Expected	Actual		
01	Upload Dataset	Test whether the dataset loaded correctly or not	If the dataset may not load	We cannot do further process	The dataset loaded successfully	Pass	High
02	Adaboost Algorithm	Verify whether the accuracy displayed or not	Without loading the dataset	We cannot run Adaboost Algorithm	Accuracy displayed successfully	Pass	High
03	Gradient Boost Algorithm	Verify either the accuracy displayed or not	Without loading the dataset	We cannot run Gradient Boost Algorithm	Accuracy displayed successfully	Pass	High
04	Random Forest Algorithm	Verify either the accuracy displayed or not	Without loading the dataset	We cannot run Random Forest Algorithm	Accuracy displayed successfully	Pass	High
5	Comparison Graph	Verify either the Accuracy, Precision, Recall, F1 Score Comparison Graph is displayed or not	Without saving the values	The Comparison Graph is not displayed	The Comparison Graph is displayed successfully	Pass	High

Table 1: Algorithms

CHAPTER - 7

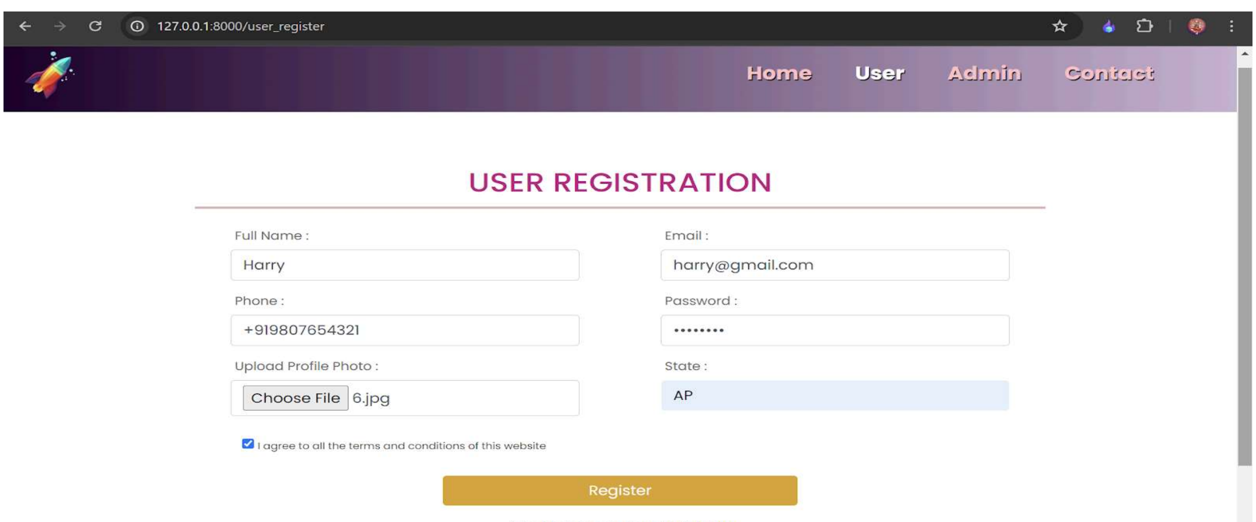
SCREENSHOTS

Step 1: In the project folder, open a new terminal and run the command "Python manage.py runserver" and click on the link which gets generated and the Home Page of the website will be displayed.



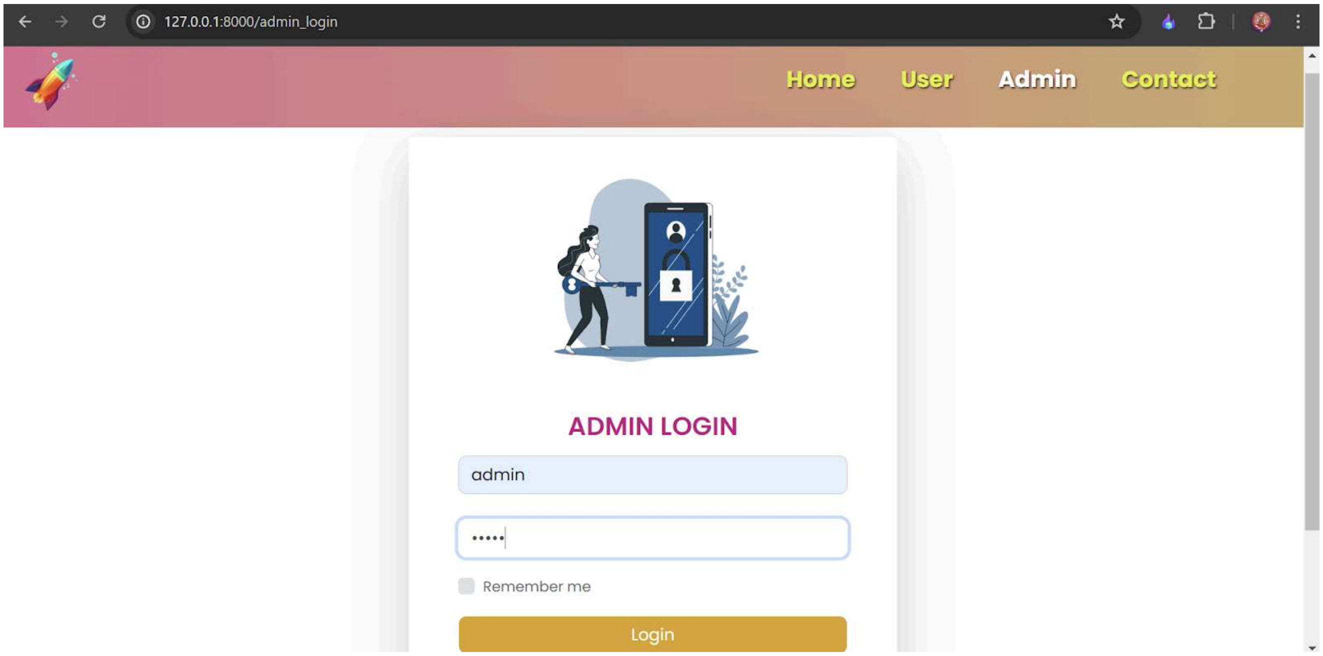
Screenshot 1: Home Page

Step 2: User can register his/her profile using their details.

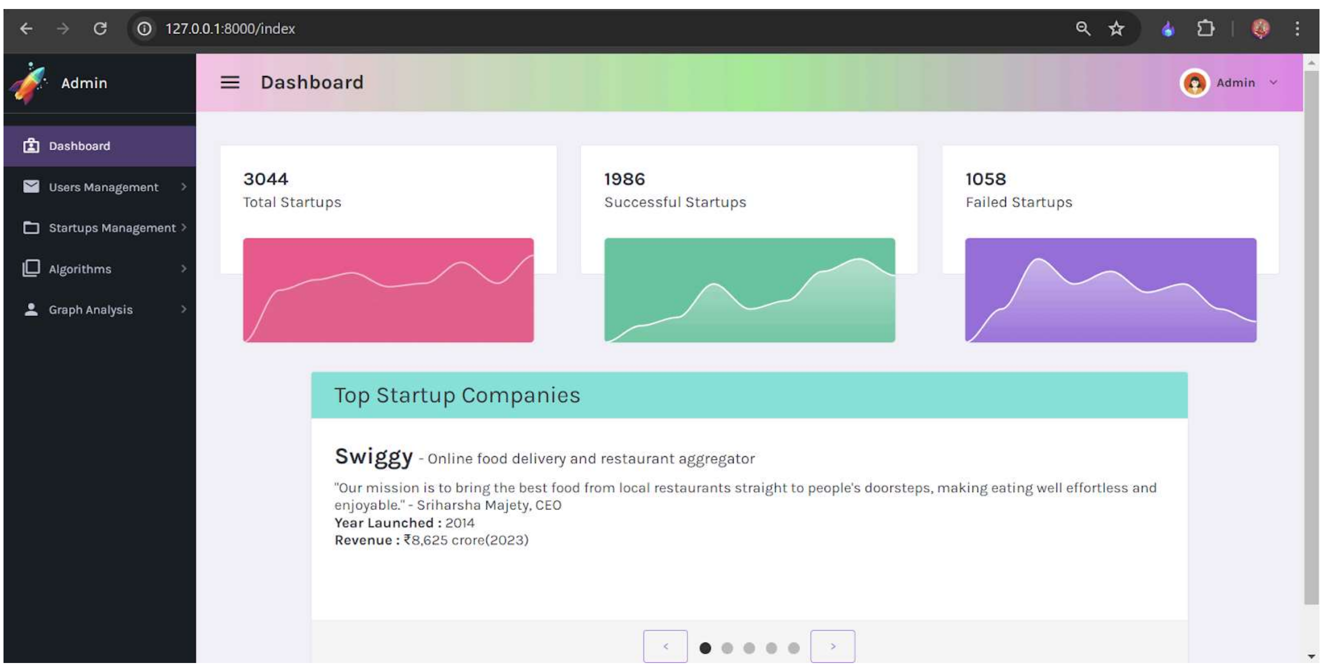


Screenshot 2: User Registration

Step 3: Admin can login using their credentials in order to accept the user.

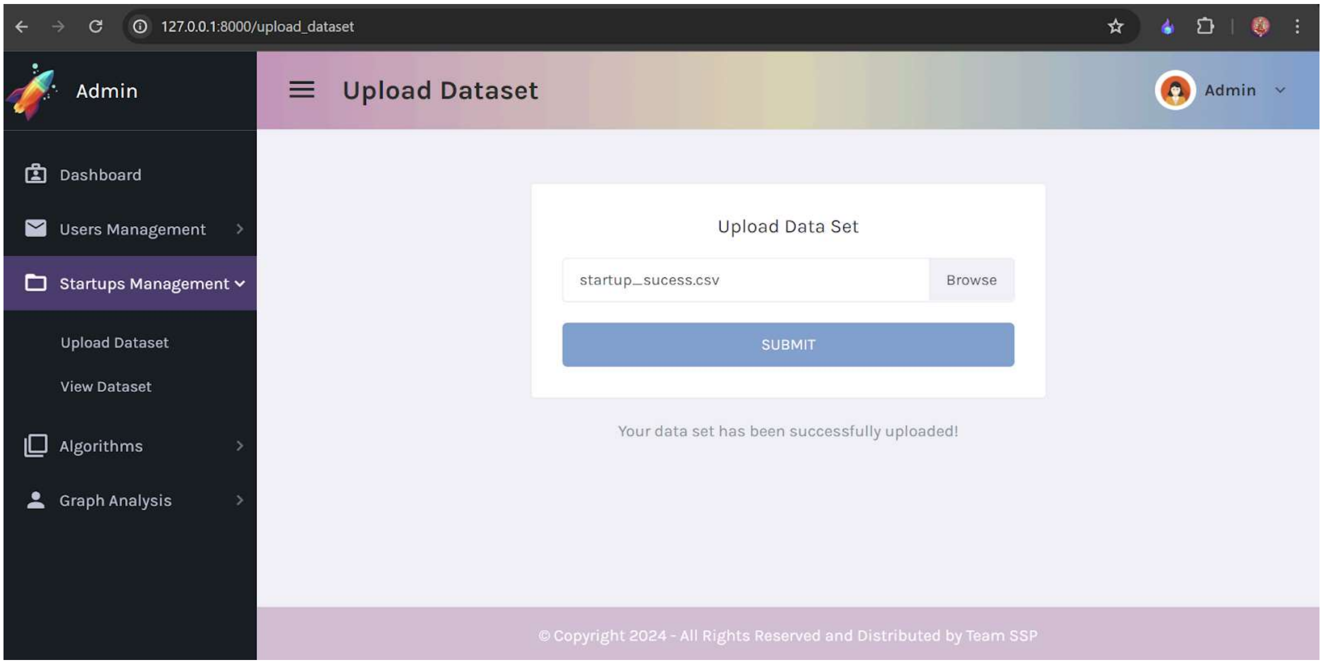


Screenshot 3: Admin Login

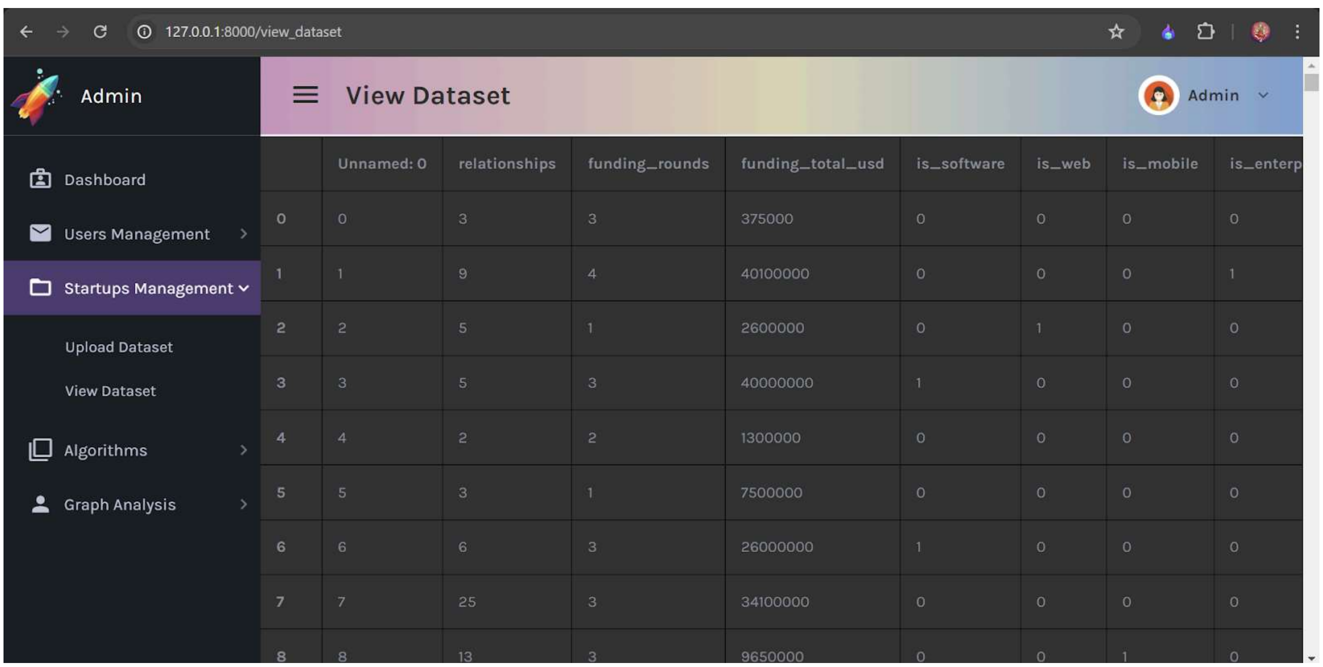


Screenshot 4: Admin Dashboard

Step 4: Dataset can be browsed and appropriate dataset needs to be selected and submitted.



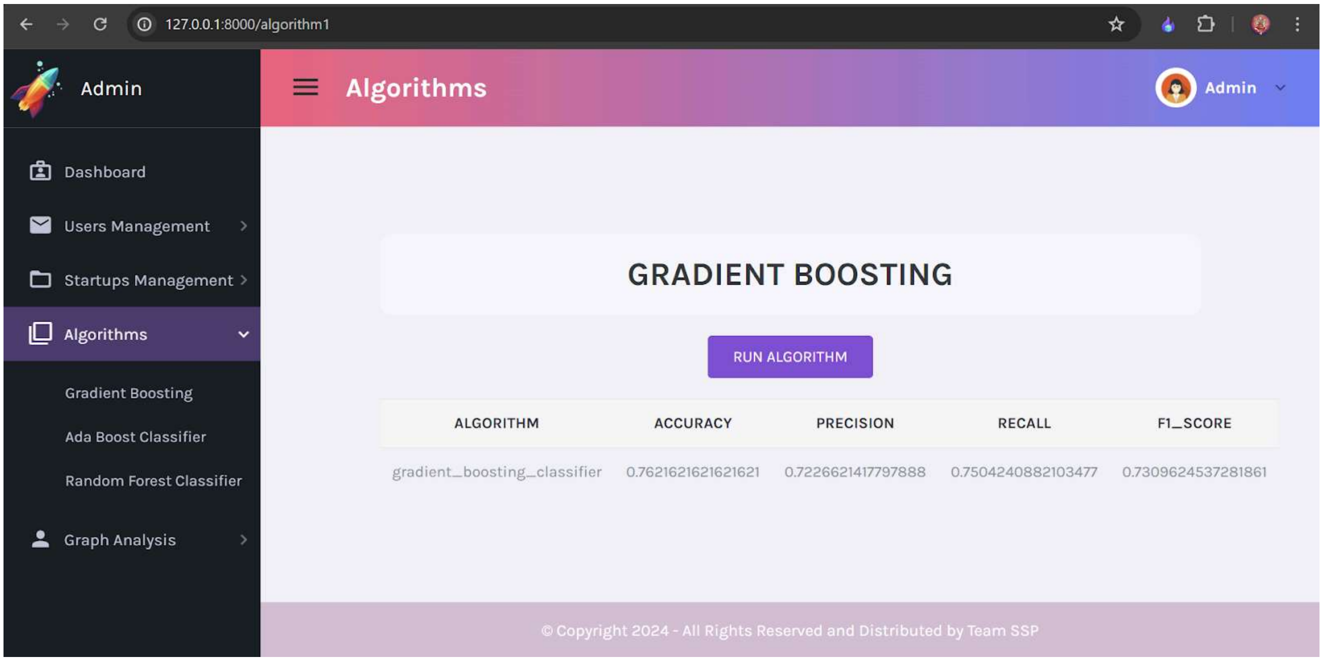
Screenshot 5: Upload Dataset



	Unnamed: 0	relationships	funding_rounds	funding_total_usd	is_software	is_web	is_mobile	is_enterp
0	0	3	3	375000	0	0	0	0
1	1	9	4	40100000	0	0	0	1
2	2	5	1	2600000	0	1	0	0
3	3	5	3	40000000	1	0	0	0
4	4	2	2	1300000	0	0	0	0
5	5	3	1	7500000	0	0	0	0
6	6	6	3	26000000	1	0	0	0
7	7	25	3	34100000	0	0	0	0
8	8	13	3	9650000	0	0	1	0

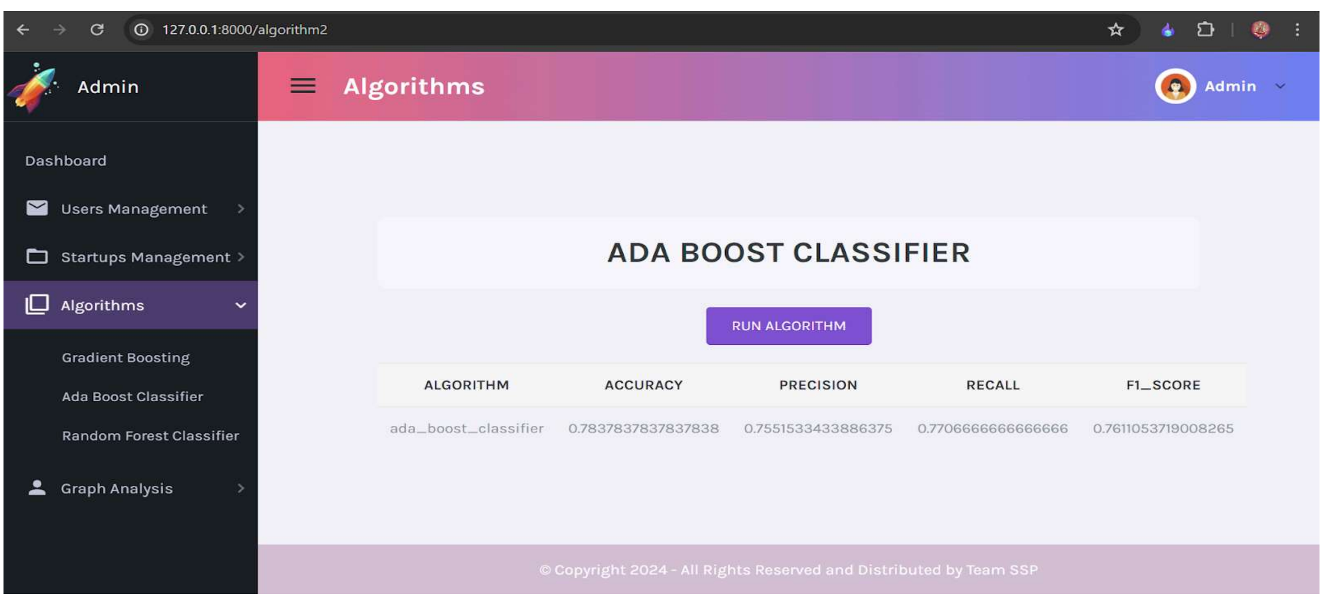
Screenshot 6: View Dataset

Step 5: Gradient Boost Algorithm can be run on the dataset and corresponding Accuracy, Precision, Recall and F1 Score can be generated like 0.7621, 0.7226, 0.7504 and 0.7309 in this case.



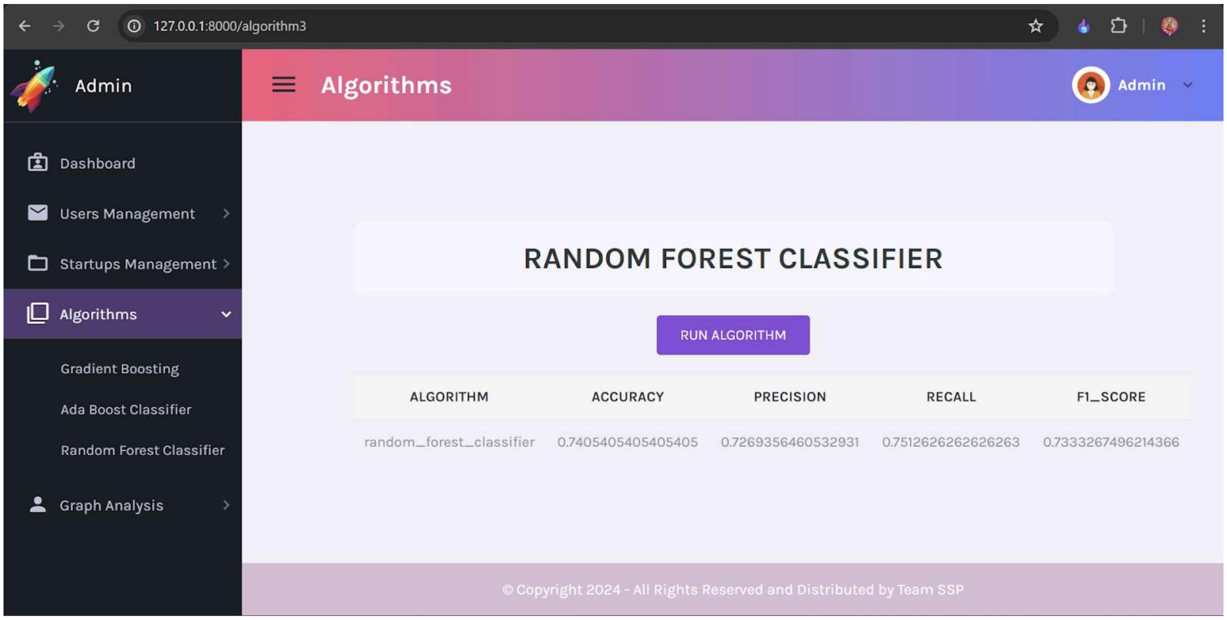
Screenshot 7: Gradient Boost

Step 6: Ada Boost Algorithm can be run on the dataset and corresponding Accuracy, Precision, Recall and F1 Score can be generated like 0.7837, 0.7551, 0.7706 and 0.7611 in this case.



Screenshot 8: Ada Boost

Step 7: Random Forest Classifier can be run on the dataset and corresponding Accuracy, Precision, Recall and F1 Score can be generated like 0.7405, 0.7269, 0.7512 and 0.7333 in this case.

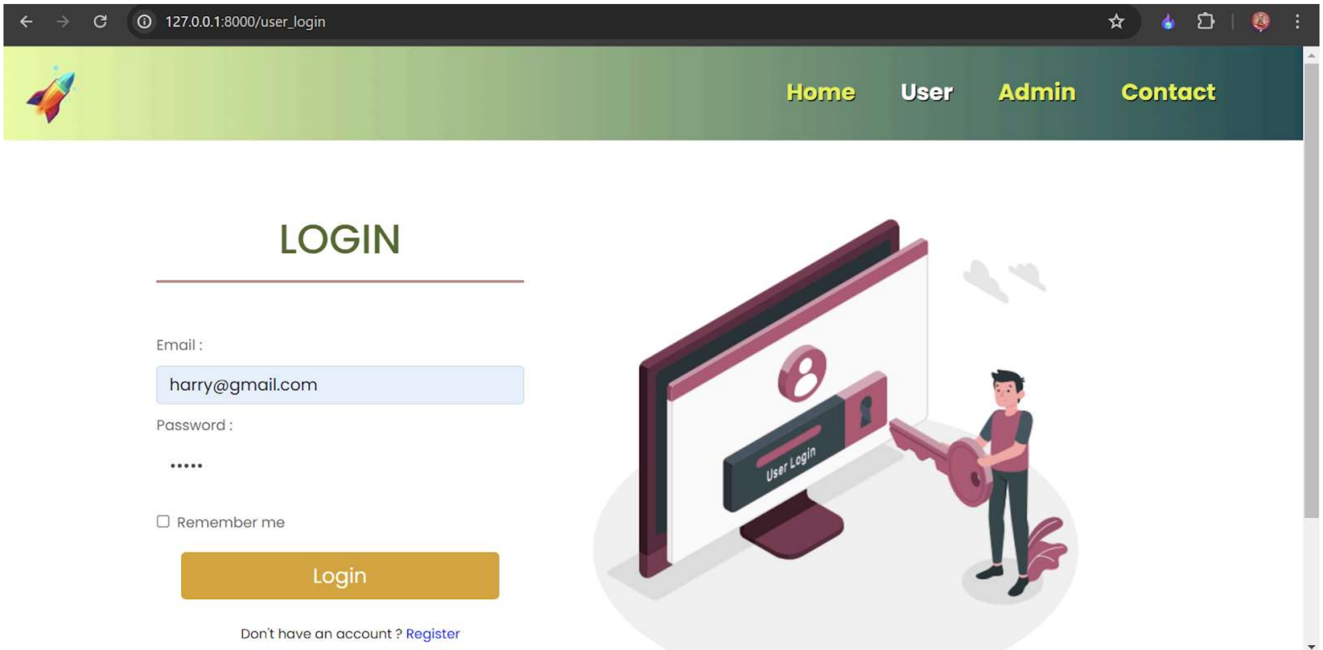


Screenshot 9: Random Forest

Step 8: The comparison between Gradient Boost, AdaBoost and Random Forest Classifier can be done by contrasting Accuracy [Blue colour], Precision [Green colour], Recall [Yellow colour] and F1 Score [Red colour] of each algorithm in the graph.

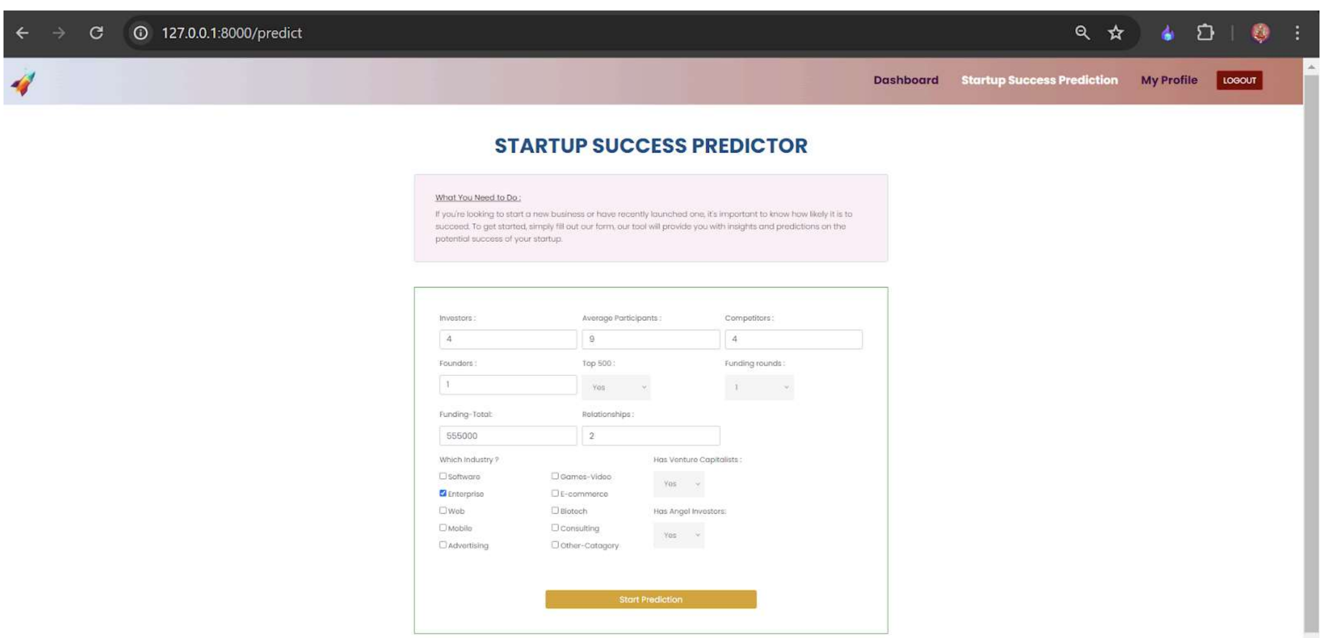
Screenshot 10: Comparison Graph

Step 9: Users can log in using their credentials.

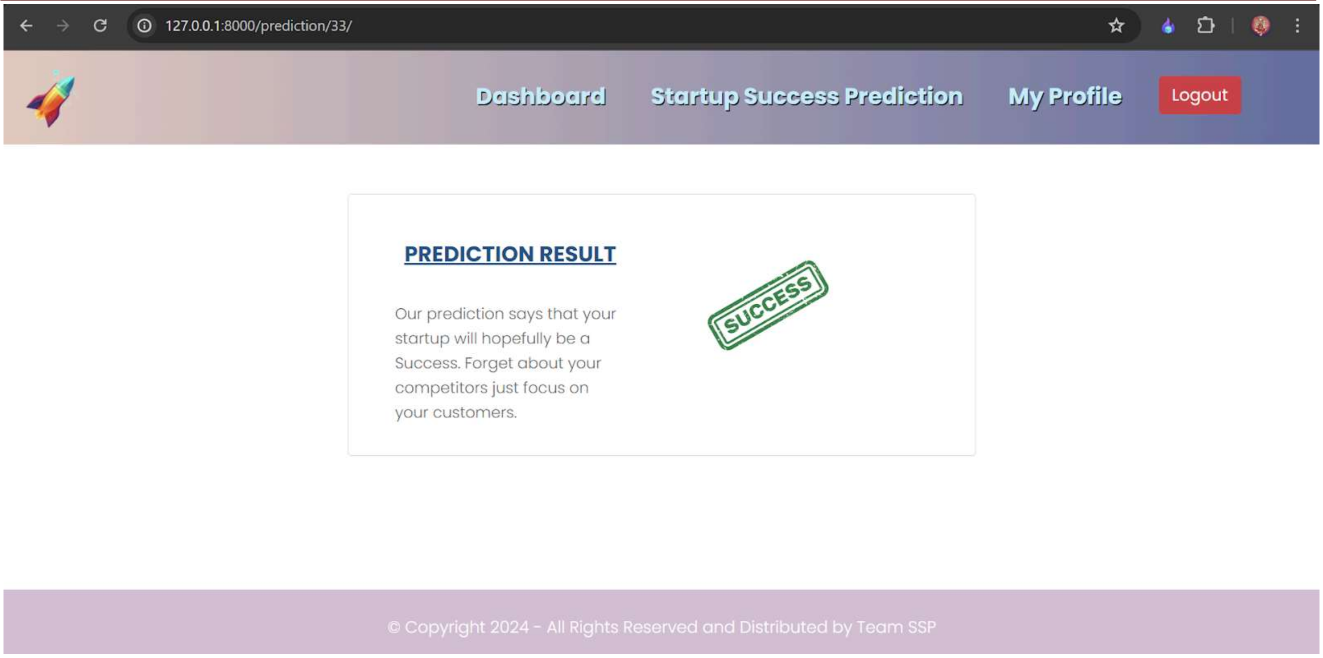


Screenshot 11: User Login

Step 10: User can start predicting the success/failure of his/her startup by plugging in all the details required.

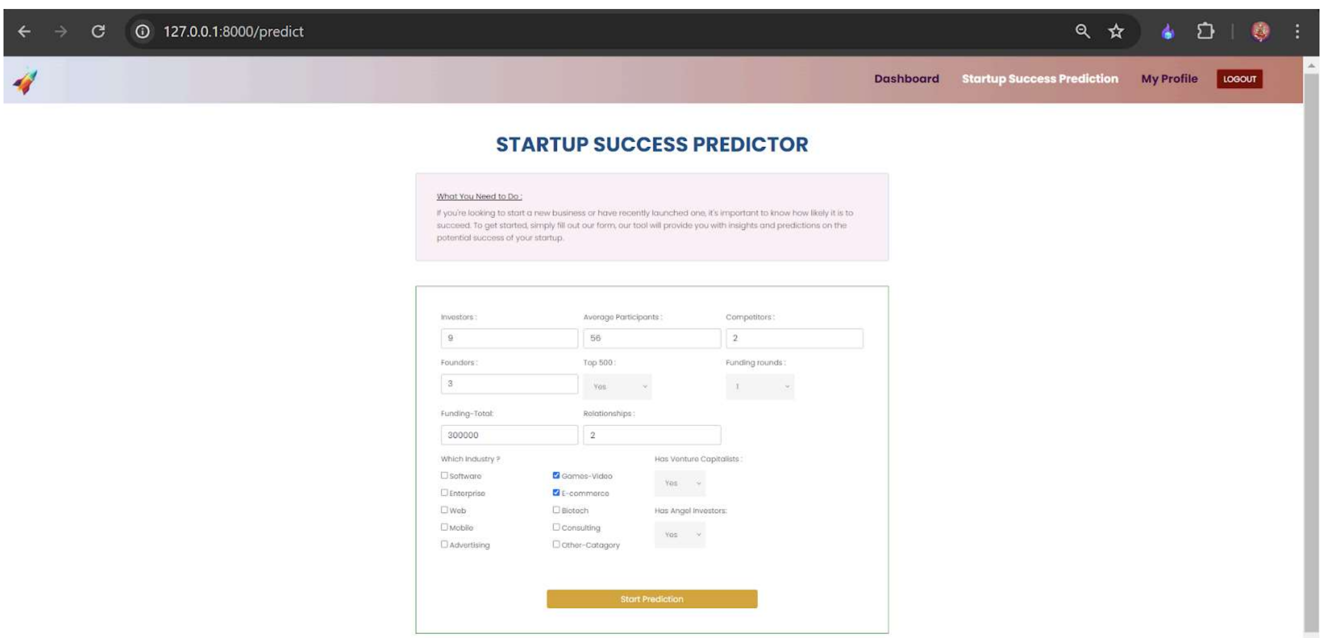


Screenshot 12: Prediction Input - 1

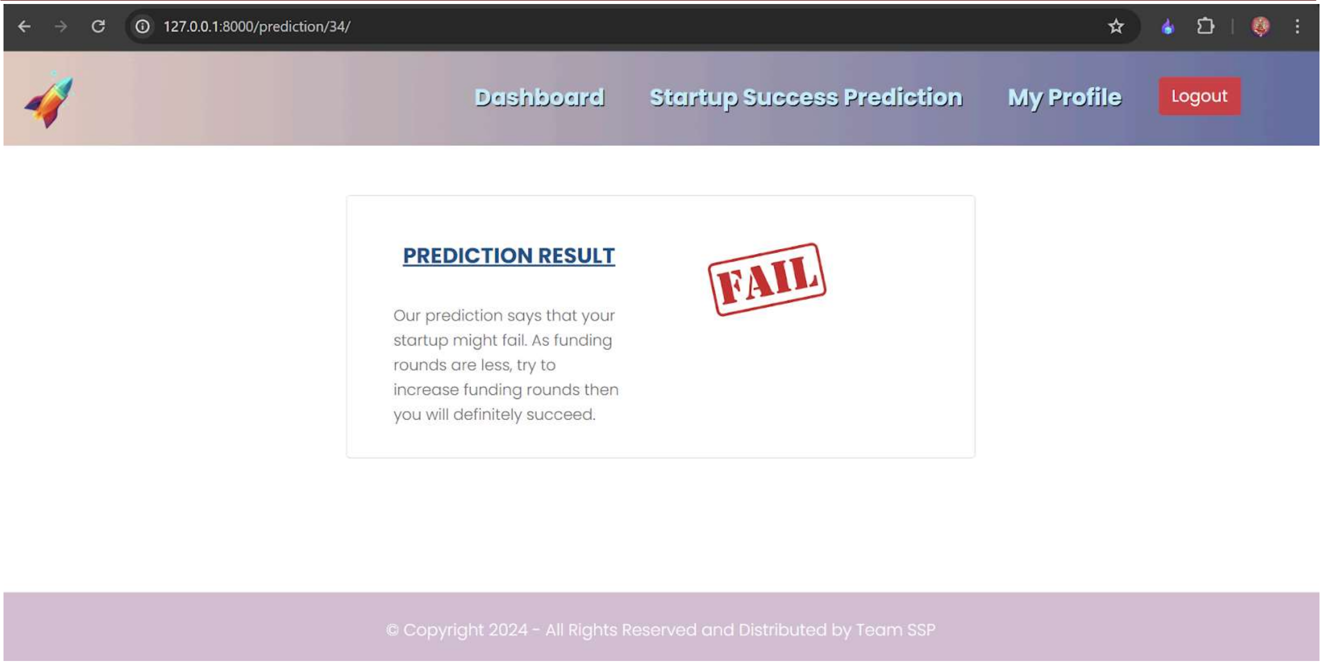


Screenshot 13: Prediction Output - 1

Step 12: User gives details of another startup.

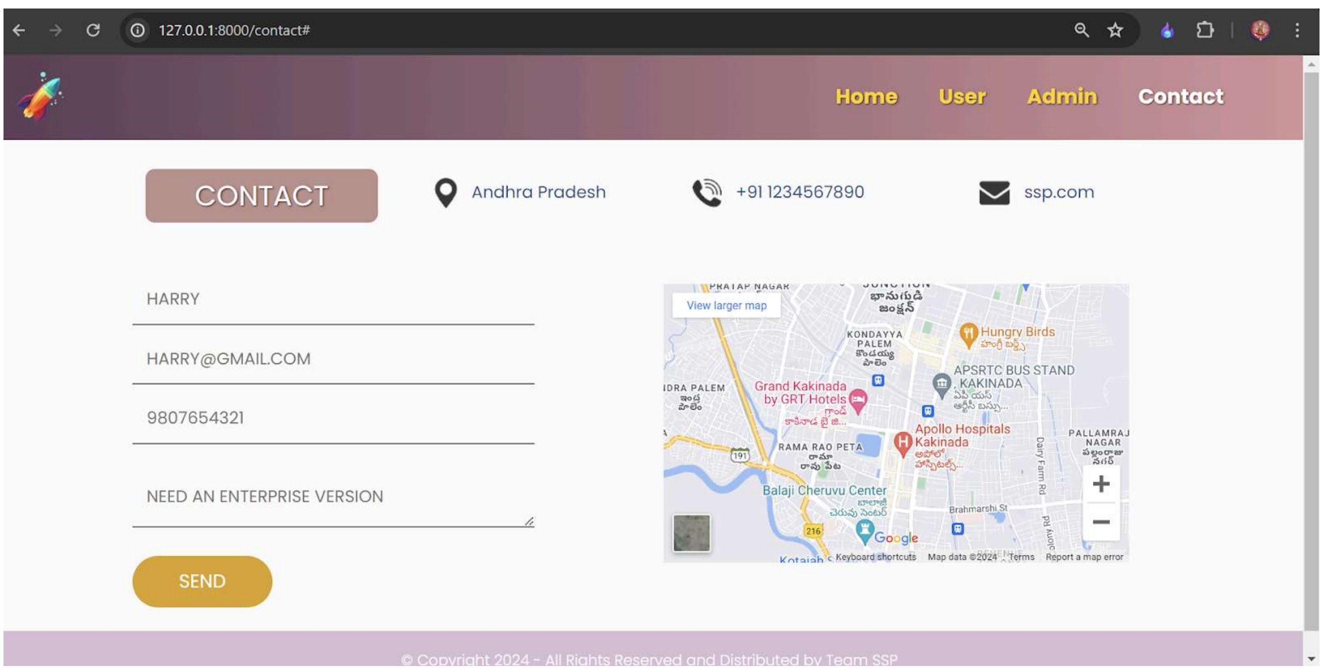


Screenshot 14: Prediction Input - 2



Screenshot 15: Prediction Output - 2

Step 13: User can also contact by providing their details and click on "SEND" .



Screenshot 16: Contact

CHAPTER - 8

CONCLUSION AND FUTURE ENHANCEMENTS

In conclusion, this study presents a significant contribution to the domain of startup success prediction by leveraging the power of machine learning algorithms. Our exploration, which employed a combination of AdaBoost, Gradient Boosting, and Random Forest algorithms, achieved a commendable accuracy of 78% and demonstrated the effectiveness of these models in identifying patterns that signal a startup's potential trajectory. Furthermore, by incorporating a broader range of success indicators compared to existing research, our approach offers a more comprehensive framework for startup viability assessment.

This study not only advances the field of predictive modelling within the startup ecosystem but also empowers decision-makers with valuable information. By enabling stakeholders to make more informed choices regarding resource allocation and investment opportunities, this study has the potential to fuel economic growth and foster a more vibrant environment for entrepreneurship.

Our study serves as a stepping stone for future advancements in startup success prediction as the field continues to evolve. By incorporating the proposed avenues for future exploration, such as real-time data integration, and Explainable AI, this study can be further extended to develop even more comprehensive and adaptable models. These enhanced models have the potential to provide stakeholders with unparalleled insights into the complex world of startups, ultimately contributing to a more successful and sustainable startup ecosystem.

Additionally, future enhancements could involve exploring ensemble learning techniques, which combine multiple models to improve predictive accuracy and robustness. Furthermore, incorporating alternative data sources such as social media sentiment analysis can provide valuable insights into consumer perceptions and market trends. Integrating advanced feature engineering methods represents another avenue for enhancing model performance and interpretability. These efforts will not only refine the accuracy of startup success prediction but also increase the transparency and trustworthiness of the models' outputs, empowering stakeholders to make more informed decisions. Overall, these endeavors will continue to refine and strengthen the predictive capabilities of startup success models, driving innovation and fostering entrepreneurial success in the years to come.

REFERENCES

[1] Y.B. Altun ”Pandemic Fuels Global Growth Of Entrepreneurship And Startup Frenzy”forbes.com.https://www.forbes.com/sites/forbestechcouncil/2021/04/09/pandemicfuels-global-growth-of-entrepreneurship-and startup frenzy/?sh=292d4e3f7308 (accessed: May 7,2022)

[2] S. Korreck. The Indian Startup Ecosystem: Drivers, Challenges and Pillars of Support. ORF Occasional Paper #210

[3] S. Torkington ”How the Great Resignation is driving a boom in startups from more diverse founders” weforum.org. https://www.weforum.org/agenda/2022/02/the-great-resignation-boomin-startups-from-more-diverse-founders/ (accessed: May 7,2022)

[4] T. Eschberger ”5 TOP reasons why startups fail” lead-innovation.com. https://www.lead-innovation.com/english-blog/reasons-startups-fail (accessed: May 7,2022)

[5] M. Van Gelderen, R. Thurik, and N. Bosma. Success and risk factors in the pre-startup phase. Small Business Economics, 24(4):365–380, 2005.

[6] T. M. Begley and W.-L. Tan. The socio-cultural environment for entrepreneurship: A comparison between east asian and anglo-saxon countries. Journal of international business studies, pages 537–553, 2001.

[7] R. Dickinson. Business failure rate. American Journal of Small Business, 6(2):17–25, 1981.

[8] W. B. Gartner. Who is an entrepreneur? is the wrong question. American journal of small business, 12(4):11–32, 1988

[9] N. K. Malhotra. Marketing Research: An Applied Orientation, 5/E. Pearson Education India, 2008.

[10] D. C. McClelland. Characteristics of successful entrepreneurs*. The journal of creative behavior, 21(3):219–233, 1987.

[11] Marco Felgueiras, Fernando Batista, Joao Paulo Carvalho, Creating Classification Models from Textual Descriptions of Companies Using Crunchbase”, IPMU 2020, CCIS 1237, pp. 695–707.

[12] Chenchen Pan, Yuan Gao, Yuzi Luo,” Machine Learning Prediction of Companies’ Business Success”, CS229: Machine Learning, Fall 2018, Stanford University, CA,2018.

Industrial Engineering Journal
ISSN: 0970-2555
Volume: 56, Issue 04, No. 4, April, 2024

SHARK TANK – REALTIME STARTUP SUCCESS AND FAILURE PREDICTION USING MACHINE LEARNING ALGORITHMS

Mrs.P.Satyavathi[1], Chaganti Saraswathi Satya Swetha[2], Devisetti Sreeja[3], Kotha Naga Sri Lakshmi[4], Charishma Sama[5], Uppuluri Sai Nikitha Devi[6]

[1] Assistant Professor, Department of AI & ML in Pragati Engineering College, Surampalem, Andhra Pradesh, India.
[2,3,4,5,6] UG Students Department of AI & ML in Pragati Engineering College, Surampalem, Andhra Pradesh, India.

Abstract: In the dynamic and competitive milieu of startup enterprises, the ability to accurately prognosticate success transcends mere desirability to assume a pivotal role in strategic decision-making. Statistical data from 2019 accentuates the formidable challenge, with an overwhelming 90% of startups encountering adversities, thereby emphasizing the imperative of employing rigorous methodologies for foretelling a startup's trajectory towards success. This study adopts a comprehensive approach, harnessing the formidable analytical capabilities of Machine Learning (ML) algorithms such as AdaBoost, Gradient Boosting, and Random Forest to conduct an exhaustive evaluation of diverse key performance indicators (KPIs) germane to startup enterprises. The parameters subjected to scrutiny extend beyond conventional metrics such as funding rounds and investor demographics to encompass more nuanced variables, including but not limited to, the composition of founding members, industry classification, and participant demographics. Notably, AdaBoost emerges as the preeminent algorithm, boasting a commendable predictive accuracy rate of 78%, thereby attesting to its efficacy in discerning patterns indicative of startup success. Drawing upon a rich corpus of historical data pertaining to startup enterprises, the models developed in this study furnish nuanced insights into the probabilistic landscape of startup success. By elucidating the interplay between diverse factors and the likelihood of achieving desired outcomes, our findings furnish invaluable guidance to stakeholders and prospective investors navigating the labyrinthine pathways of the startup ecosystem. Consequently, the insights gleaned from this research serve to inform strategic decision-making processes, thereby fostering more informed and judicious allocation of resources within the volatile and uncertain milieu of startup enterprises.

Keywords: Startup Success Prediction, Machine Learning, AdaBoost, Gradient Boosting, Random Forest, Predictive Modeling, Startup Metrics.

1. Introduction

In recent years, the global landscape has witnessed an unprecedented surge in the proliferation of startups, marking a significant departure from traditional economic models and signaling a paradigm shift towards entrepreneurship-driven growth. This surge has been particularly pronounced in the aftermath of the COVID-19 pandemic, which has not only accelerated the adoption of digital technologies but also spurred a remarkable surge in entrepreneurial activity worldwide. In the United States alone, the number of applications for new business formations reached a record high of 551,657 in July 2020, representing a staggering 95% increase compared to the same period in 2019.

The success of a startup is multifaceted, encompassing various dimensions that extend beyond mere profitability. While financial viability is undoubtedly a crucial aspect, the success of a startup is also contingent upon factors such as innovation, market traction, scalability, and the ability to adapt to changing market dynamics. Successful startups are characterized by their ability to identify unmet needs or pain points in the market and develop innovative solutions that resonate with customers. Moreover, scalability is essential for startups to capitalize on growth opportunities and achieve sustainable long-term success.

However, navigating the path to success is fraught with challenges, and not all startups succeed in realizing their aspirations. The failure rate of startups in 2019 was alarmingly high, standing at around 90%. Research indicates that the failure rate varies across different stages of a startup's lifecycle, with 21.5% failing in the first year, 30% in the second year, 50% in the fifth year, and 70% by their tenth year.

The determinants of startup success are manifold and can vary depending on factors such as industry, market conditions, and the competitive landscape. Common challenges faced by startups include limited access to capital, intense competition, regulatory hurdles, and the risk of market saturation. Predicting success enables entrepreneurs, investors, and stakeholders to make informed decisions regarding resource allocation, strategic planning, and risk management.

By leveraging predictive analytics and machine learning algorithms, stakeholders can gain valuable insights into the factors that drive startup success and identify potential pitfalls that may impede growth. Moreover, predictive models can help identify promising startups early on, thereby enabling investors to allocate capital more efficiently and mitigate investment risks. Successful startups not only generate employment opportunities but also stimulate innovation, foster competition, and contribute to overall economic growth.

Furthermore, successful startups have the potential to address pressing societal challenges, ranging from healthcare and education to environmental sustainability and social inequality. By understanding the determinants of startup success and leveraging predictive analytics, stakeholders can enhance their ability to identify promising ventures, allocate resources effectively, and contribute to economic development and societal welfare.

2. Literature review

The contemporary entrepreneurial landscape has been teeming with dynamic activity, particularly underscored by the burgeoning global startup ecosystem. This phenomenon has experienced an exponential surge, especially in the aftermath of the seismic disruptions wrought by the COVID-19 pandemic, which served as a catalyst for a pronounced proliferation of novel enterprises. Notably, amidst the socioeconomic turbulence precipitated by the pandemic, a substantial cohort of individuals, compelled by circumstances such as job displacement and economic uncertainty, opted to pivot towards entrepreneurial pursuits as a means of economic sustenance and self-actualization. Evidentially, official records and empirical data gleaned from reputable sources attest to an unprecedented proliferation of new businesses throughout the tumultuous year of 2020, eclipsing antecedent annual metrics and evincing an unmistakable entrepreneurial fervor that permeated various sectors and geographies.

In the Indian context, which serves as a focal point of burgeoning entrepreneurial endeavors, the trajectory of startup proliferation has been particularly noteworthy, underpinned by a confluence of factors including favorable regulatory frameworks, access to capital, burgeoning technological infrastructure, and an increasingly robust support ecosystem comprising incubators, accelerators, and venture capital firms. The ascendancy of India's startup landscape assumes paramount significance not only as a testament to the nation's burgeoning entrepreneurial spirit but also as a barometer of its economic dynamism and innovation prowess on the global stage. Understanding the nuanced contours of the Indian startup milieu is thus imperative, affording insights into the underlying drivers propelling founders, the multifarious challenges besieging them, and the requisite support mechanisms indispensable for their sustenance and growth trajectory.

An incisive examination of data gleaned from a myriad of sources, including qualitative interviews, quantitative surveys, and extant scholarly discourse, constitutes a foundational pillar in elucidating the intricacies characterizing the Indian entrepreneurial landscape. Such empirical analyses serve not only to discern prevailing trends and patterns but also to identify emergent phenomena and nascent opportunities that may have hitherto eluded scholarly scrutiny. Moreover, they provide a robust evidentiary basis for policymakers, practitioners, and stakeholders to formulate informed strategies aimed at fostering an enabling environment conducive to entrepreneurial dynamism and innovation.

Of particular intrigue within the realm of contemporary entrepreneurship is the burgeoning prevalence of startups helmed by individuals hailing from diverse sociocultural milieus. These nascent enterprises, often characterized by their agility, innovation, and propensity for disruption, epitomize the democratization of entrepreneurship, transcending traditional barriers and hierarchies to empower individuals from marginalized or underrepresented backgrounds. The motivational substrates underpinning the entrepreneurial aspirations of such individuals, as well as the attendant vicissitudes and challenges they confront along their entrepreneurial journey, constitute fertile terrain for scholarly inquiry and empirical investigation.

Notwithstanding the palpable exuberance surrounding entrepreneurial ventures, the harsh veracity of startup failure looms large as a sobering reminder of the inherent risks and uncertainties endemic to entrepreneurial endeavors. Scrutiny into the determinants underpinning startup demise not only sheds light on the pitfalls and challenges confronting nascent ventures but also underscores the imperative for a nuanced comprehension of the elements engendering entrepreneurial success. Leveraging methodological tools such as Machine Learning algorithms, coupled with an exhaustive examination of diverse facets of startup gestation, engenders a repository of discernments pivotal for aspiring entrepreneurs, investors, policymakers, and other stakeholders alike. By distilling empirical findings into actionable insights, scholars and practitioners can inform evidence-based interventions aimed at mitigating risk, enhancing resilience, and catalyzing success within the entrepreneurial

Industrial Engineering Journal
ISSN: 0970-2555
Volume: 56, Issue 04, No. 4, April, 2024

ecosystem. Ultimately, a nuanced understanding of the dynamics undergirding entrepreneurial success or failure furnishes a foundational substrate for the cultivation of a nurturing ecosystem conducive to entrepreneurial endeavor, thereby fostering innovation, economic growth, and societal prosperity.

3. Existing System

The prior research landscape in the field of startup success prediction underwent extensive examination by scholars such as Pan et al. and Arroyo et al., who scrutinized various aspects of startup outcomes using diverse methodological approaches. Pan et al. focused on forecasting specific events within startup trajectories, such as mergers, acquisitions, and initial public offerings, employing algorithms like Logistic Regression, Random Forests, and K Nearest Neighbors. Notably, K Nearest Neighbors emerged as the best performer, displaying superior F1 scores.

In contrast, Arroyo et al. expanded their investigation to include a broader range of startup outcomes, including subsequent funding rounds and closures. They employed algorithms such as Support Vector Machines, Decision Trees, Random Forests, Extremely Randomized Trees, and Gradient Tree Boosting. Remarkably, Gradient Tree Boosting achieved an accuracy level of approximately 82%.

However, the prevailing trend in startup success prediction leaned heavily on conventional statistical models and a limited set of predictors. This reliance is evident in the repeated use of methodologies like K-nearest neighbors (KNN) to forecast success rates among established firms. While effective for mature organizations, these conventional approaches often struggle with the complexities of the dynamic and heterogeneous early-stage startup ecosystems. The limitations of these methodologies include difficulties in addressing the inherent uncertainty and ambiguity characteristic of the initial stages of venture development. Consequently, such constraints lead to inaccurate predictions and suboptimal decision-making processes.

4. Proposed System

Our proposed system represents a significant advancement over existing methodologies, offering several advantages that enhance its effectiveness in predicting startup success. By incorporating AdaBoost, Gradient Boosting, and Random Forest algorithms, our system leverages the strengths of each approach. These ensemble methods are known for their ability to handle complex relationships within data and mitigate overfitting, resulting in more accurate predictions compared to single algorithms. AdaBoost, one of the key components of our system, has demonstrated the highest accuracy among the employed algorithms, achieving a notable 78%. This superior performance is indicative of its robustness in capturing the intricate patterns and dynamics inherent in startup data.

Unlike previous models that relied on a limited set of predictors, our approach considers a wide range of parameters such as investor count, founder demographics, funding details, industry type, and investor profiles. By integrating diverse factors that influence startup success, our model provides a holistic view of the startup ecosystem, leading to more reliable predictions. The output format of our system is designed to be clear and straightforward, providing stakeholders and investors with concise information about the likelihood of a startup's success or failure. This transparency enables informed decision-making and facilitates proactive measures to mitigate risks or capitalize on opportunities.

Leveraging extensive historical data, our models offer nuanced insights into the potential success of startups. By analyzing past trends and patterns, our system identifies key drivers of success and highlights areas of concern, enabling stakeholders to make strategic adjustments and optimize their investment strategies. The dynamic nature of the startup landscape demands adaptive prediction models capable of accommodating evolving trends and market conditions. Our system's flexibility makes it well-suited to navigate the complexities of the ever-changing startup ecosystem, ensuring relevance and reliability over time.

5. Software Environment

Software Environment	Technology
Operating System	Windows 10/11
Development Language	Python 3.10
IDE	Visual Studio Code
Front-End Technologies	HTML5, CSS3, JavaScript
Back-End Framework	Django
Database Language	SQL
Database Management System	MySQL
Local Development Server	XAMPP
Development Web Server	Django Development Server
Design & Modelling Tool	Rational Rose

Table1: Software Environment

Industrial Engineering Journal
ISSN: 0970-2555
Volume: 56, Issue 04, No. 4, April, 2024

6. System Architecture

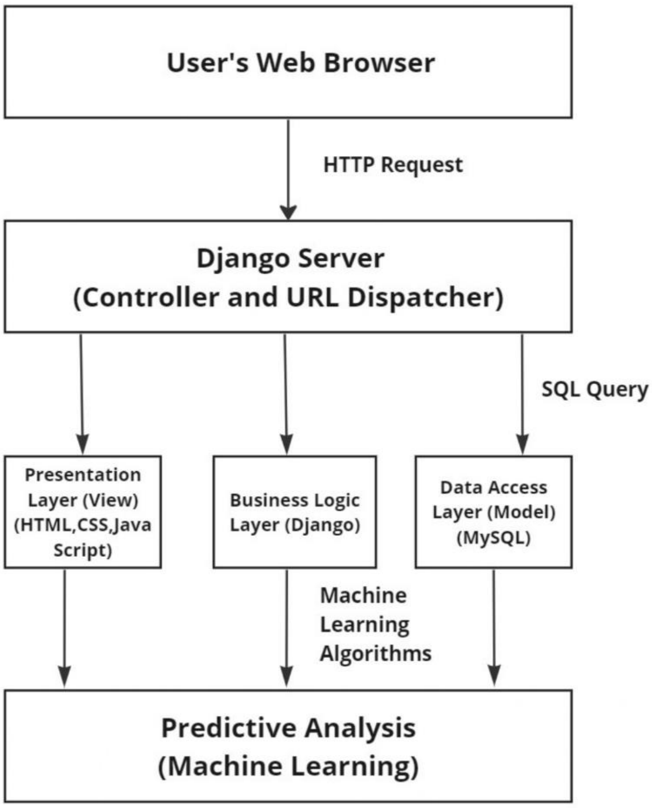
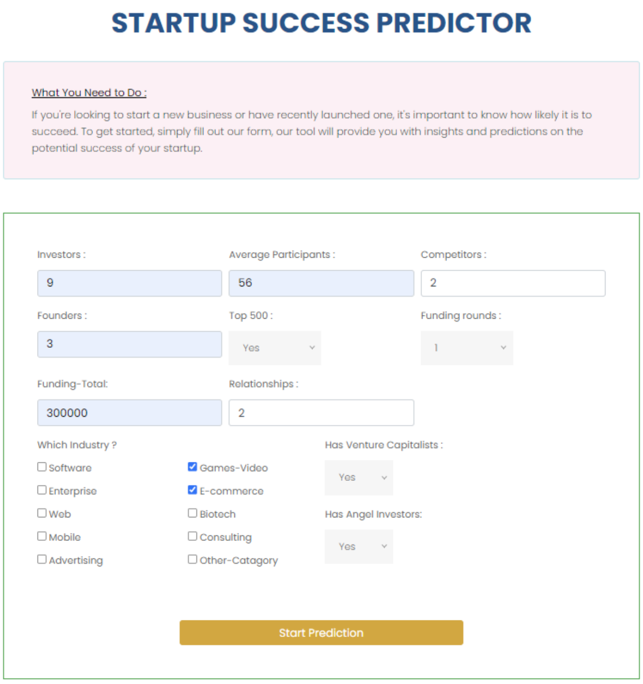


Figure 1: System architecture

7. Results



Industrial Engineering Journal
ISSN: 0970-2555
Volume: 56, Issue 04, No. 4, April, 2024

Figure 2: Predicting Startup Company – 1

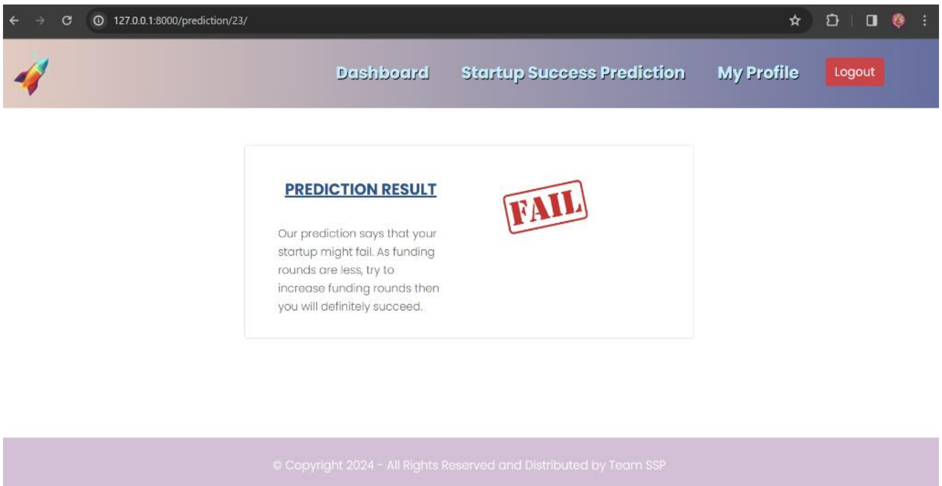


Figure 3: Startup Company – 1 Result Predicted as Fail

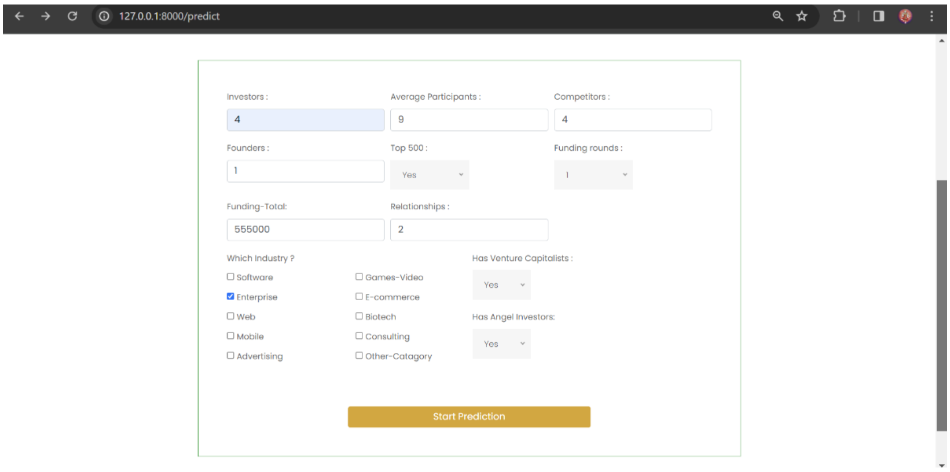


Figure 4: Predicting Startup Company – 2

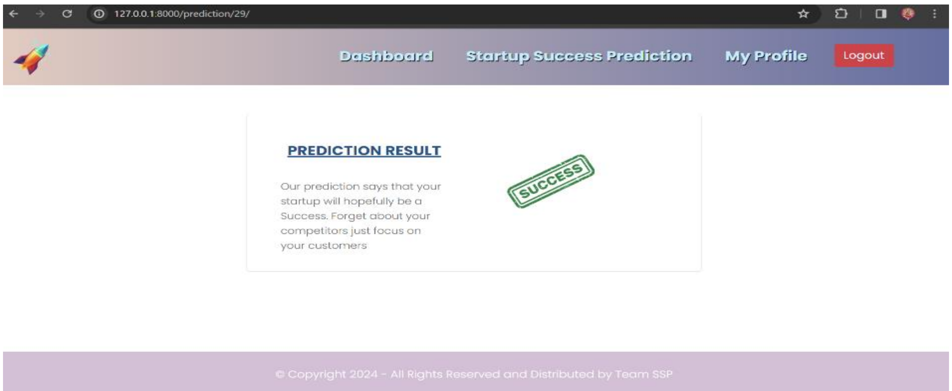


Figure 5: Startup Company – 2 Result Predicted as Success

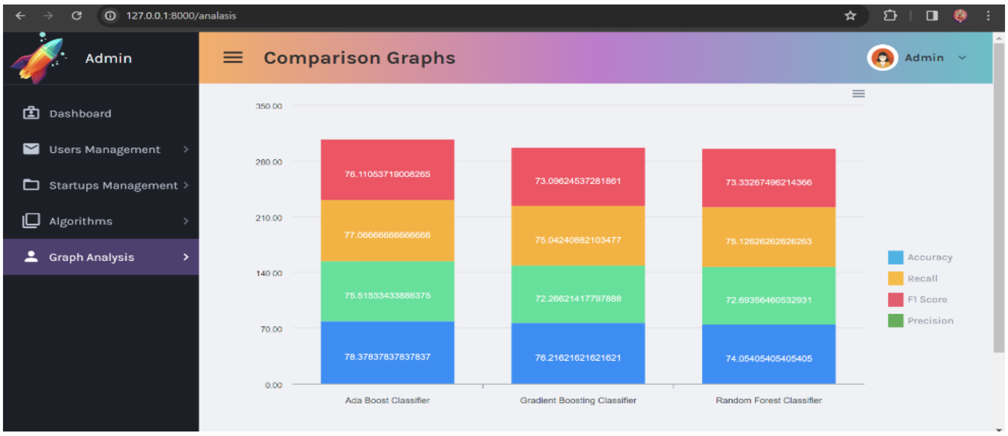


Figure 6: Comparison Graph

8. Future Directions

The pursuit of leveraging machine learning algorithms to forecast startup success is an ongoing endeavor marked by continuous refinement and innovation. While our current research has established a solid groundwork, the horizon is brimming with enticing opportunities, each poised to elevate our models to greater heights of potency and adaptability. Real-time Data Integration is a focal point, exploring the integration of dynamic data streams like market trends and user feedback to enhance model adaptability. Incorporating User Feedback from social media or customer reviews holds promise in capturing market dynamics and detecting early signs of traction or challenges.

Additionally, prioritizing Explainable AI (XAI) techniques ensures transparency in predictions, aiding stakeholders' understanding of underlying factors contributing to success probabilities. Investigating Cross-border Startup Analysis aims to extend model applicability across diverse regions, while Temporal Analysis offers insights into startup trajectory evolution over time. Integration of Unstructured Data sources like sentiment analysis and news articles enriches predictive models with deeper market insights. Ethical Considerations remain paramount, necessitating the development of frameworks to ensure responsible and equitable model application in the startup ecosystem.

9. Conclusion

In conclusion, this study presents a significant contribution to the domain of startup success prediction by leveraging the power of machine learning algorithms. Our exploration, which employed a combination of AdaBoost, Gradient Boosting, and Random Forest algorithms, achieved a commendable accuracy of 78% and demonstrated the effectiveness of these models in identifying patterns that signal a startup's potential trajectory. Furthermore, by incorporating a broader range of success indicators compared to existing research, our approach offers a more comprehensive framework for startup viability assessment.

This study not only advances the field of predictive modelling within the startup ecosystem but also empowers decision-makers with valuable information. By enabling stakeholders to make more informed choices regarding resource allocation and investment opportunities, this study has the potential to fuel economic growth and foster a more vibrant environment for entrepreneurship.

Our study serves as a stepping stone for future advancements in startup success prediction as the field continues to evolve. By incorporating the proposed avenues for future exploration, such as real-time data integration, and Explainable AI, this study can be further extended to develop even more comprehensive and adaptable models. These enhanced models have the potential to provide stakeholders with unparalleled insights into the complex world of startups, ultimately contributing to a more successful and sustainable startup ecosystem.

References

[1] Y.B. Altun ”Pandemic Fuels Global Growth Of Entrepreneurship And Startup Frenzy”forbes.com.https://www.forbes.com/sites/forbestechcouncil/2021/04/09/pandemicfuels-global-growth-of-entrepreneurship-and startup frenzy/?sh=292d4e3f7308 (accessed: May 7,2022)

[2] S. Korreck. The Indian Startup Ecosystem: Drivers, Challenges and Pillars of Support. ORF Occasional Paper #210

[3] S. Torkington ”How the Great Resignation is driving a boom in startups from more diverse founders” weforum.org. https://www.weforum.org/agenda/2022/02/the-great-resignation-boomin-startups-from-more-diverse-founders/ (accessed: May 7,2022)

[4] T. Eschberger ”5 TOP reasons why startups fail” lead-innovation.com. https://www.lead-innovation.com/english-blog/reasons-startups-fail (accessed: May 7,2022)

[5] M. Van Gelderen, R. Thurik, and N. Bosma. Success and risk factors in the pre-startup phase. Small Business Economics, 24(4):365–380, 2005.

[6] T. M. Begley and W.-L. Tan. The socio-cultural environment for entrepreneurship: A comparison between east asian and anglo-saxon countries. Journal of international business studies, pages 537–553, 2001.

[7] R. Dickinson. Business failure rate. American Journal of Small Business, 6(2):17–25, 1981.

[8] W. B. Gartner. Who is an entrepreneur? is the wrong question. American journal of small business, 12(4):11–32, 1988

[9] N. K. Malhotra. Marketing Research: An Applied Orientation, 5/E. Pearson Education India, 2008.

[10] D. C. McClelland. Characteristics of successful entrepreneurs*. The journal of creative behavior, 21(3):219–233, 1987.

[11] Marco Felgueiras, Fernando Batista, Joao Paulo Carvalho, Creating Classification Models from Textual Descriptions of Companies Using Crunchbase”, IPMU 2020, CCIS 1237, pp. 695–707.

[12] Chenchen Pan, Yuan Gao, Yuzi Luo,” Machine Learning Prediction of Companies' Business Success”, CS229: Machine Learning, Fall 2018, Stanford University, CA,2018.

[13] Javier Arroyo , Francesco Corea, Guillermo Jimenez-D ´ ´ıaz, Juan A. Recio-Garc´ıa,“ Assessment of Machine Learning Performance for Decision Support in Venture Capital Investments”, IEEE Access 7,2019, pp. 124233-124243

[14] Amar Krishna, Ankit Agrawal, Alok Choudhary, “Predicting the Outcome of Startups: Less Failure, More Success”, IEEE, 2016.