A

Mini Project Report on

Machine Learning for Classification of Stars, Galaxies Through Exploring the SDSS Space Observation Dataset

Submitted for partial fulfilment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING (AI&ML)

By

KUPPAM AVINASH	**20K81A6622**
KUNAPAREDDY SHANTHI	**20K81A6621**
CHINTHA NEHA	**20K81A6610**
TEEDA YOGESHAJAY	**20K81A6651**

Under the Guidance of

Dr. K. SRINIVAS

ASSOCIATE PROFESSOR

DEPARTMENT OF

COMPUTER SCIENCE AND ENGINEERING (AI&ML)

St. MARTIN'S ENGINEERING COLLEGE

UGC Autonomous

NBA & NAAC A+ Accredited

Dhulapally, Secunderabad - 500 100

www.smec.ac.in

OCTOBER 2023

Certificate

This is to certify that the project entitled **“Machine Learning for Classification of Stars, Galaxies Through Exploring the SDSS Space Observation Dataset”** is being submitted By **KUPPAM AVINASH (20K81A6622)**, **KUNAPAREDDY SHANTHI (20K81A6621), CHINTHA NEHA (20K81A6610), TEEDA YOGESHAJAY (20K81A6651)** in fulfilment of the requirement for the award of degree of **BACHELOR OF TECHNOLOGY in DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)** is recorded of bonafide work carried out by them. The result embodied in this report have been verified and found satisfactory.

Guide	**Head of the Department**
Dr. K. SRINIVAS	**Dr. M. SREEDHAR REDDY**
Associate Professor	**Professor & Head**
Department of	**Department of**
CSE(AI&ML)	**CSE(AI&ML)**

Internal Examiner **External Examiner**

Date:

Place:

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING (AI&ML)

DECLARATION

We, the students of '**Bachelor of Technology in Department of Computer Science and Engineering (AI&ML)**', session: 2020 - 2024**, St. Martin's Engineering College, Dhulapally, Kompally, Secunderabad,** hereby declare that the work presented in this Project Work entitled "**Machine Learning for Classification of Stars, Galaxies Through Exploring the SDSS Space Observation Dataset"** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. This result embodied in this project report has not been submitted in any university for award of any degree.

KUPPAM AVINASH	**20K81A6622**
KUNAPAREDDY SHANTHI	**20K81A6621**
CHINTHA NEHA	**20K81A6610**
TEEDA YOGESHAJAY	**20K81A6651**

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whoseencouragement and guidance have crowded our efforts with success.

First and foremost, we would like to express our deep sense of gratitude and indebtednessto our College Management for their kind support and permission to use the facilities available inthe Institute.

We especially would like to express our deep sense of gratitude and indebtedness to **Dr. P. SANTOSH KUMAR PATRA**, Group Director, St. Martin’s Engineering College Dhulapally, for permitting us to undertake this project.

We wish to record our profound gratitude to **Dr. M. SREENIVAS RAO**, Principal, St. Martin’s Engineering College, for his motivation and encouragement.

We are also thankful to **Dr. M. SREEDHAR REDDY**, Head of the Department, Computer Science and Engineering (AI&ML), St. Martin’s Engineering College, Dhulapally, Secunderabad, for his support and guidance throughout our project as well as Project Coordinator **Dr. K. SRINIVAS**, Professor, Computer Science and Engineering (AI&ML) department for his valuable support.

We would like to express our sincere gratitude and indebtedness to our project supervisor **Dr. K. SRINIVAS**, Professor, Computer Science and Engineering (AI&ML), St. Martins Engineering College, Dhulapally, for his support and guidance throughout our project.

Finally, we express thanks to all those who have helped us successfully completing this project. Furthermore, we would like to thank our family and friends for their moral support and encouragement. We express thanks to all those who have helped us in successfully completing the project.

KUPPAM AVINASH	**20K81A6622**
KUNAPAREDDY SHANTHI	**20K81A6621**
CHINTHA NEHA	**20K81A6610**
TEEDA YOGESHAJAY	**20K81A6651**

CONTENTS

LIST OF FIGURES

ABSTRACT

The Sloan Digital Sky Survey (SDSS) stands as one of the most remarkable and ambitious astronomical surveys ever undertaken. It has bestowed upon us an immense wealth of data on stars and galaxies, offering an invaluable resource for unravelling the mysteries of the universe. Analyzing and categorizing these celestial objects plays a crucial role in various astrophysical studies, from understanding galaxy formation and evolution to delving into cosmology. Yet, the task of manually analyzing and classifying stars and galaxies proves to be arduous and time-consuming. The traditional methods of scrutinizing vast astronomical datasets entail human experts visually inspecting images and spectra to categorize objects based on their visual characteristics. However, with the sheer scale of the data, this manual approach becomes impractical and prone to human errors. In recent times, Machine Learning (ML) has emerged as a powerful solution for automating the classification process. ML algorithms can learn from the labelled data provided by astronomers, allowing them to recognize patterns and characteristics that differentiate different types of celestial objects. By training on a large dataset with labelled examples, ML models can generalize their knowledge and accurately classify new, unseen objects based on the features they have learned. Thus, this research work endeavours to tackle the challenge of efficiently classifying the vast number of stars and galaxies within the SDSS dataset. The proposed system delves into the application of ML techniques to automate the classification process, leveraging the rich features and information contained in the dataset. Employing state-of-the-art algorithms, this proposed system showcases how ML can achieve precise and dependable classification results, surpassing traditional manual approaches. The significance of this endeavour lies in its ability to streamline the analysis of astronomical data, providing astronomers with a valuable and time-saving tool for large-scale sky surveys. Ultimately, this advancement in classification technology will enhance our understanding of celestial objects and contribute to the broader exploration of the universe.

CHAPTER 1
INTRODUCTION

1.1 Introduction

The universe is an intricate tapestry of celestial objects, each holding unique secrets and insights into the cosmos. Amidst this vast expanse, the Sloan Digital Sky Survey (SDSS) stands as a monumental achievement in modern astronomy. Over the years, it has meticulously catalogued millions of stars, galaxies, and other celestial entities, offering a treasure trove of data waiting to be explored. However, as the universe unfolds in seemingly infinite diversity, so does the complexity of classifying these objects.

Machine Learning, a branch of artificial intelligence, has emerged as a powerful tool to unlock the secrets hidden within the astronomical data captured by the SDSS. This technology enables us to sift through the immense volumes of data and classify celestial objects with a precision and efficiency that was once unimaginable.

In this exploration, we delve into the exciting realm of "Machine Learning for Classification of Stars, Galaxies Through Exploring the SDSS Space Observation Dataset." We embark on a journey where algorithms and data intersect with the cosmos, striving to answer fundamental questions about the universe.

This endeavor encompasses not only the application of state-of-the-art machine learning techniques but also the fusion of astronomy and computer science. By harnessing the collective power of data and artificial intelligence, we aim to refine our understanding of the universe, discover new celestial phenomena, and push the boundaries of human knowledge.

In the following sections, we will outline the steps involved in this endeavour, from data preprocessing and feature engineering to model selection and deployment. We will also discuss the significance of interpretability in this context and emphasize the importance of collaboration between astronomers and data scientists. Together, we embark on a cosmic odyssey, using machine learning as our guide to unravel the mysteries of the universe, one data point at time.

1.2 Motivation

The study of the universe has been a perennial quest for humanity, a pursuit driven by an insatiable curiosity to comprehend the cosmos that surrounds us. As we peer into the night sky, astronomers and scientists have tirelessly sought to understand the celestial objects that dot the firmament — from the nearest stars in our own galaxy to the farthest galaxies that stretch the limits of our observational capabilities. This quest for cosmic knowledge has given birth to incredible advancements in technology and observation, one of which is the Sloan Digital Sky Survey (SDSS).

The SDSS, initiated in the early 2000s, has been a revolutionary leap forward in the field of astronomy. It has provided a comprehensive and meticulously curated database of celestial objects, covering billions of stars, galaxies, quasars, and more. Yet, this trove of astronomical data comes with its own set of challenges. With the sheer volume of data generated by the SDSS and the complexity of the universe it represents, manually classifying and cataloguing each object is an arduous, if not impossible, task. Herein lies the motivation for the application of machine learning. Machine learning algorithms can tirelessly process and classify astronomical data on an unprecedented scale. They can discern subtle patterns, distinguish between different types of celestial objects, and identify rare and unusual phenomena. This capability is not just a matter of convenience; it is a necessity for the modern astronomer and astrophysicist.

The motivation for employing machine learning for the classification of stars, galaxies, and other cosmic entities through the SDSS dataset is multi-faceted:

The universe is a vast laboratory for scientific exploration. By automating the classification of celestial objects, we enable astronomers to focus on more profound questions about the nature of the cosmos, such as the origins of galaxies, the evolution of stars, and the behaviour of exotic objects like quasars. The scale of the SDSS dataset is beyond human capacity to process efficiently. Machine learning provides a means to handle this data deluge, enabling astronomers to make discoveries more swiftly and effectively. The universe is replete with rare and transient objects. Machine learning models can identify these objects in real-time, allowing astronomers to capture fleeting phenomena like supernovae, gamma-ray bursts, and fast radio bursts. The fusion of astronomy and machine learning fosters interdisciplinary collaboration. Astronomers and data scientists working together can leverage each other's expertise to push the boundaries of astronomical research. Machine learning applications in astronomy offer engaging educational tools that can spark interest in science and technology among students and the general public.

In summary, the motivation behind using machine learning for classifying stars, galaxies, and celestial objects through the SDSS dataset is not only driven by the desire to expand our understanding of the universe but also by the necessity to efficiently manage and analyse the colossal volumes of data produced by modern observational surveys. It's a symbiotic relationship between data science and astronomy that promises to unlock the secrets of the cosmos and inspire generations to come.

1.3 Problem Statement

The Sloan Digital Sky Survey (SDSS) has provided the astronomical community with a vast and invaluable dataset containing information on millions of celestial objects, including stars, galaxies, quasars, and more. However, the manual classification of these objects is a monumental task due to the dataset's size and complexity. To address this challenge and facilitate scientific discovery, there is a pressing need to develop machine learning models capable of accurately and efficiently classifying celestial objects within the SDSS dataset.

Data Volume: The SDSS dataset encompasses an enormous volume of astronomical data, including images, spectra, and metadata, which demands scalable and efficient machine-learning solutions.

Data Heterogeneity: The dataset contains diverse types of astronomical objects, each requiring distinct classification approaches. Stars, galaxies, quasars, and other celestial entities exhibit unique spectral and morphological characteristics.

Data Quality: Astronomical observations are subject to noise, artefacts, and calibration errors, making data preprocessing and cleaning essential for reliable classification.

Rare Object Identification: Rare and transient astronomical objects must be identified and classified accurately, as they hold significant scientific value.

1.4 Applications

Applications of Machine Learning for the Classification of Stars, Galaxies:

- **Astronomical Object Classification:** Automated classification of stars, galaxies, quasars, and other celestial objects within the SDSS dataset, which assists astronomers in cataloguing and studying these objects on a large scale.

- **Transient Object Detection:** Early identification and classification of transient astronomical events such as supernovae, variable stars, and gamma-ray bursts, enabling rapid follow-up observations and contributing to our understanding of astrophysical phenomena.

- **Discovery of Rare Objects:** Identification of rare and exotic celestial objects, such as high-redshift quasars, brown dwarfs, and gravitational lensing events, which are valuable for scientific research and discovery.

- **Galaxy Morphology Analysis:** Classification of galaxies based on their morphological features (e.g., spiral, elliptical) to study galaxy evolution and formation.

- **Stellar Population Studies:** Distinguishing different types of stars, including main sequence, giant, and variable stars, for investigating stellar populations and their characteristics.

- **Redshift Estimation:** Predicting the redshift of galaxies and quasars, which provides critical information about their distance and cosmic evolution.

- **Cosmic Web Mapping:** Mapping the large-scale structure of the universe, including clusters and filaments of galaxies, by classifying galaxy positions and types.

- **Exoplanet Detection:** Identifying potential exoplanetary systems by analyzing the light curves of stars in the SDSS dataset for periodic dimming events caused by transiting exoplanets.

- **Spectral Analysis:** Analysing the spectra of celestial objects to determine their chemical composition, temperature, and other physical properties.

- **Scientific Exploration:** Facilitating scientific research by automating the classification of objects in large observational datasets, freeing up astronomers' time for more in-depth analysis and hypothesis testing.

- **Education and Outreach:** Developing educational tools and interactive applications that engage students and the public in astronomy and data science, promoting interest in STEM (Science, Technology, Engineering, and Mathematics) fields.

- **Data Quality Assurance:** Identifying and flagging potential issues with data quality, such as artifacts or calibration errors, for further investigation and refinement of observational techniques.

CHAPTER 2
LITERATURE SURVEY

Universal Domain Adaptation for Galaxy Morphology Classification

Ćiprijanović, et al. [11] a universal domain adaptation method, DeepAstroUDA, as an approach to overcome this challenge. This algorithm performs semi-supervised domain adaptation (DA) and can be applied to datasets with different data distributions and class overlaps. Non-overlapping classes can be present in any of the two datasets (the labeled source domain, or the unlabeled target domain), and the method can even be used in the presence of unknown classes. We apply our method to three examples of galaxy morphology classification tasks of different complexities (three-class and ten-class problems), with anomaly detection.

Photometric Redshift Estimation and Classification for Astronomical Source

Cunha, et al. [12] proposed a new machine-learning approach to the classic problem of astronomical source classification, which combines the outputs from the XGBoost, LightGBM, and CatBoost learning algorithms to create stronger classifiers. A novel step in our pipeline is that prior to performing the classification, SHEEP first estimates photometric redshifts, which are then placed into the data set as an additional feature for classification model training; this results in significant improvements in the subsequent classification performance.

Identifying Symbiotic Stars Candidates using Machine Learning

Jia, et al. [13] proposed a significant discrepancy between the observed population of symbiotic stars and the number predicted by theoretical models. To bridge this gap, this study utilized machine learning techniques to efficiently identify new symbiotic stars candidates. Three algorithms (XGBoost, LightGBM, and Decision Tree) were applied to a dataset of 198 confirmed symbiotic stars and the resulting model was then used to analyse data from the LAMOST survey, leading to the identification of 11,709 potential symbiotic stars candidates.

Effects of Perturbations in Imaging Data and Domain Adaptation

Ciprijanovic, et al. [14] proposed the effects of perturbations in imaging data. In particular, we examine the consequences of using neural networks when training on baseline data and testing on perturbed data. We consider perturbations associated with two primary sources: (a) increased observational noise as represented by higher levels of Poisson noise and (b) data processing noise incurred by steps such as image compression or telescope errors as represented by one-pixel adversarial attacks. We also test the efficacy of domain adaptation techniques in mitigating perturbation-driven errors.

Quenching of Star Formation in Galaxies Throughout Cosmic History

Bluck, et al. [15] proposed an analysis of the quenching of star formation in galaxies, bulges, and disks throughout the bulk of cosmic history, from z = 2 − 0. We utilise observations from the Sloan Digital Sky Survey and the Mapping Nearby Galaxies at Apache Point Observatory survey at low redshifts. We complement these data with observations from the Cosmic Assembly Near-Infrared Deep Extragalactic Legacy Survey at high redshifts. Additionally, we compare the observations to detailed predictions from the LGalaxies semi-analytic model.

Classification of Stellar Features in Galaxy and Star using Machine Learning

Mehta, et al. [16] proposed techniques for classifying stellar features into a Galaxy and a Star using various Machine Learning Algorithms on a dataset consisting of 100,000 observations. Techniques like K-Nearest Neighbours (KNN), Support Vector Classifier (SVC), Random Forest (RF), Logistic Regression (LR), Decision Tree (DT), and Naïve Bayes (NB) are used to train the model. Ultimately, a relative examination is done based on investigational outcomes from distinct models.

Deep Convolutional Neural Network for Galaxy Morphology Classification

Mahalakshmi, et al. [17] proposed A deep convolutional neural network architecture for galaxy morphology classification and feature prediction. The galaxy can be divided into ten morphological classes based on its characteristics. The proposed architecture consists of two models: one for class prediction and the other for galaxy features prediction. This paper attempts to classify astronomical objects into Galaxies, Stars and Quasars using data from SDSS, and LAMOST Surveys in addition to spectroscopic data analysis.

Training Set Composition Bias in Machine Learning and Rare Object Identification

Lake, et al. [18] proposed an exploration of how training set composition bias in machine learning affects identifying rare objects. It is also a frequent practice to train on restricted data where the balance of source types is closer to equal for the same reason. Here we show that these practices can bias the model toward over-assigning sources to the rare class. We also explore how to detect when training data bias has had a statistically significant impact on the trained model's predictions, and how to reduce the bias's impact.

Detecting Gravitational Lenses using Machine Learning

Wilde, et al. [19] proposed Detecting gravitational lenses using machine learning: exploring interpretability and sensitivity to rare lensing configurations. We design, build, and train several convolutional neural networks (CNNs) to identify strong gravitational lenses using VIS, Y, J, and H bands of simulated data, with F1 scores between 0.83 and 0.91 on 100 000 test set images.

We demonstrate for the first time that such CNNs do not select against compound lenses, obtaining recall scores as high as 76 per cent for compound arcs and 52 per cent for double rings.

Random Forest Algorithm for the Classification of Spectral Data

Solorio-Ramírez, et al. [20] proposed a Random Forest Algorithm for the Classification of Spectral Data of Astronomical Objects. This work aims to illustrate the versatility of machine learning algorithms, such as decision trees, to facilitate the identification and classification of celestial bodies by manipulating hyperparameters and studying the attributes of celestial body datasets. By applying a random forest algorithm to a well-known dataset that includes three types of celestial bodies, its effectiveness was compared against some supervised classifiers of the most important approaches (Bayes, nearest neighbours, support vector machines, and neural networks).

Robustness of Deep Learning Models for Galaxy Morphology Classification

Ćiprijanović, et al. [21] proposed examining the robustness of deep learning models for galaxy morphology classification. They examine the consequences of using neural networks when training on baseline data and testing on perturbed data. We consider perturbations associated with two primary sources: (a) increased observational noise as represented by higher levels of Poisson noise and (b) data processing noise incurred by steps such as image compression or telescope errors as represented by one-pixel adversarial attacks.

Identifying Merging Galaxies using H-α and Hi Observations

Lee, et al. [22] proposed Identifying Merging Galaxies in MaNGA using H-α and Hi Observations. we investigate how massive central galaxies cease their star formation by comparing theoretical predictions from cosmological simulations: EAGLE, Illustris, and IllustrisTNG with observations of the local Universe from the Sloan Digital Sky Survey (SDSS). Our machine learning (ML) classification reveals supermassive black hole mass (MBH) as the most predictive parameter in determining whether a galaxy is star forming or

quenched at redshift $z = 0$ in all three simulations.

Classification of Cosmic Structures for Galaxies with Deep Learning

Inoue, et al. [23] proposed Classification of cosmic structures for galaxies with deep learning: connecting cosmological simulations with observations. They demonstrate that deep learning can bridge the gap between the simulations and observations. Our models are based on 3D convolutional neural networks and trained with data of distribution of galaxies in a simulation to deduce the structure classes from the galaxies rather than DM. Our model can predict the class labels as accurate as a previous study using DM distribution for the training and prediction.

ULISSE: Identifying Lookalike Objects in Extragalactic Astronomy

oorenbos, et al. [24] proposed ULISSE, a new deep learning tool that, starting from a single prototype object, is capable of identifying objects that share common morphological and photometric properties, and hence of creating a list of candidate lookalikes. In this work, we focus on applying our method to the detection of active galactic nuclei (AGN) candidates in a Sloan Digital Sky Survey galaxy sample, because the identification and classification of AGN in the optical band still remains a challenging task in extragalactic astronomy.

Classifying MaNGA Velocity Dispersion Profiles by Machine Learning

Duann, et al. [25] proposed Classifying MaNGA Velocity Dispersion Profiles by Machine Learning. They presented a machine learning (ML) approach for classifying kinematic profiles of elliptical galaxies in the Mapping Nearby Galaxies at Apache Point Observatory (MaNGA) survey. Previous studies employing ML to classify spectral data of galaxies have provided valuable insights into morphological galaxy classification. This study aims to enhance the understanding of galaxy kinematics by leveraging ML.

CHAPTER 3

SYSTEM ANALYSIS AND DESIGN

3.1 Existing System

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:

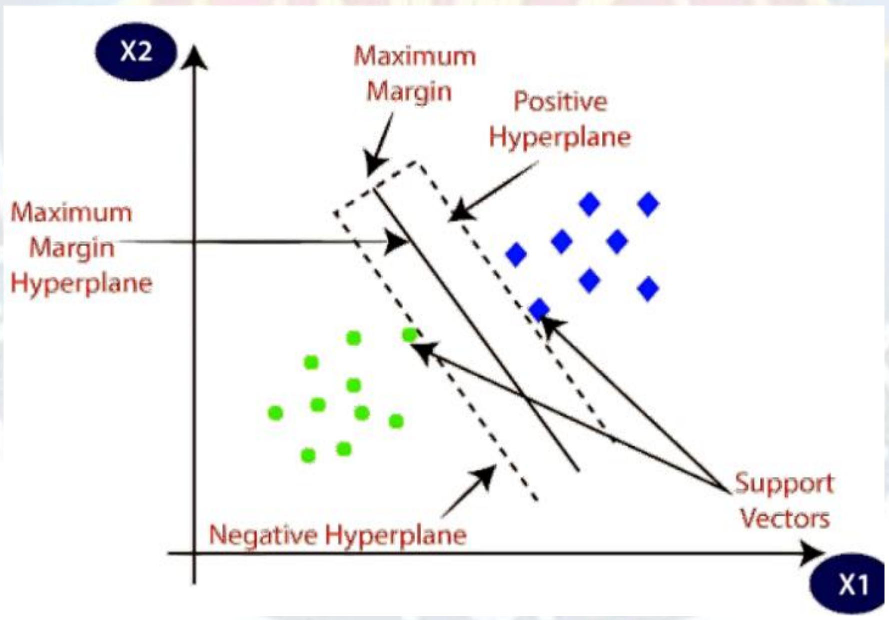


Figure 3.1. SVM Model.

Disadvantages of Existing System

1. Unsuitable to Large Datasets
2. Large training time
3. More features, more complexities
4. Bad performance on high noise
5. Does not determine Local optima

3.2 Proposed System

Classifying stars and galaxies based on images is a common task in astronomy and astrophysics. Here's a high-level overview of the process, including image preprocessing, Random Forest (RF) model training, and prediction. Figure 3.2 shows the proposed system model. This process represents a simplified overview of classifying stars and galaxies using image preprocessing, RF model training, and prediction. The effectiveness of the approach depends on the quality of the data, the choice of features, and the performance of the machine learning model.

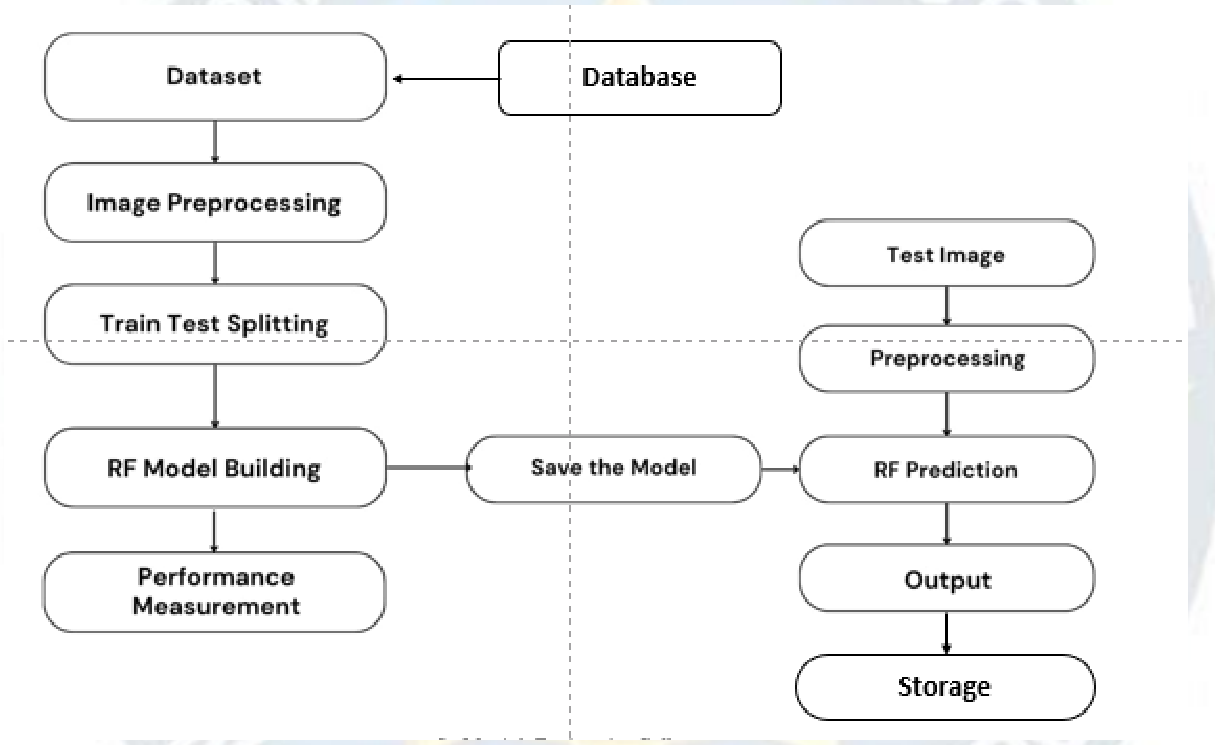


Figure 3.2. Proposed System Model.

Advantages of Proposed System:

1. It can be used in classification and regression problems.
2. It solves the problem of overfitting as output is based on majority voting or averaging.
3. It performs well even if the data contains null/missing values.
4. Each decision tree created is independent of the other thus it shows the property of parallelization.
5. It is highly stable as the average answers given by a large number of trees are taken.
6. It maintains diversity as all the attributes are not considered while making each decision tree though it is not true in all cases.
7. It is immune to the curse of dimensionality. Since each tree does not consider all the attributes, feature space is reduced.

3.3 DESIGN

3.3.1 Class Diagram

The **class diagram** is the main building block of object-oriented modelling. It is used both for general conceptual modelling of the systematic of the application, and for detailed modelling translating the models into programming code. Class diagrams can also be used for data modelling. The classes in a class diagram represent both the main objects, interactions in the application and the classes to be programmed. In the diagram, classes are represented with boxes which contain three parts:

- The upper part holds the name of the class
- The middle part contains the attributes of the class
- The bottom part gives the methods or operations the class can take or undertake

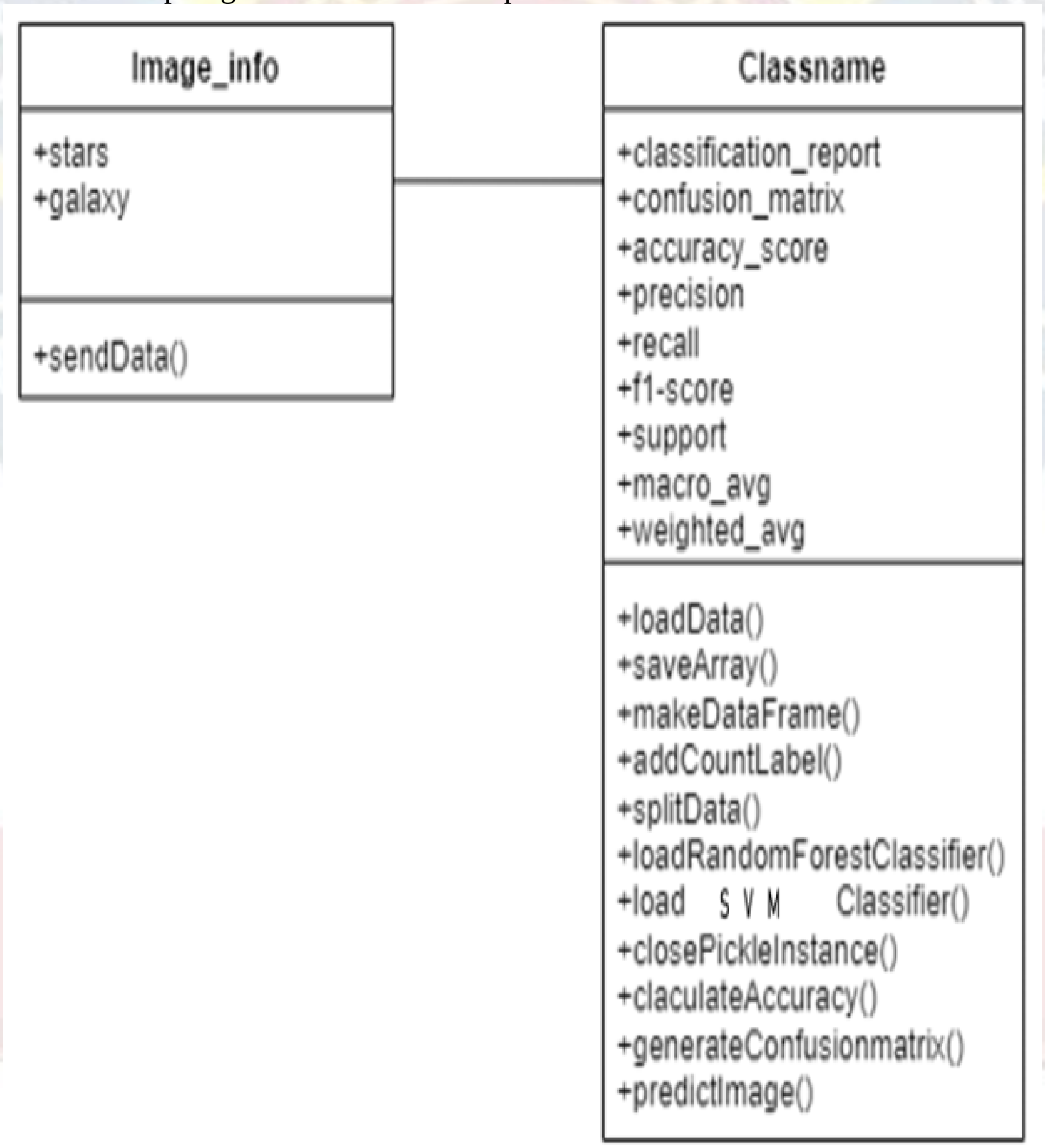


Fig 3.3.1 Class Diagram

3.3.2 Use case Diagram

A **use case diagram** at its simplest is a representation of a user's interaction with the system and depicting the specifications of a use case. A use case diagram can portray the different types of users of a system and the various ways that they interact with the system. This type of diagram is typically used in conjunction with the textual use case and will often be accompanied by other types of diagrams as well.

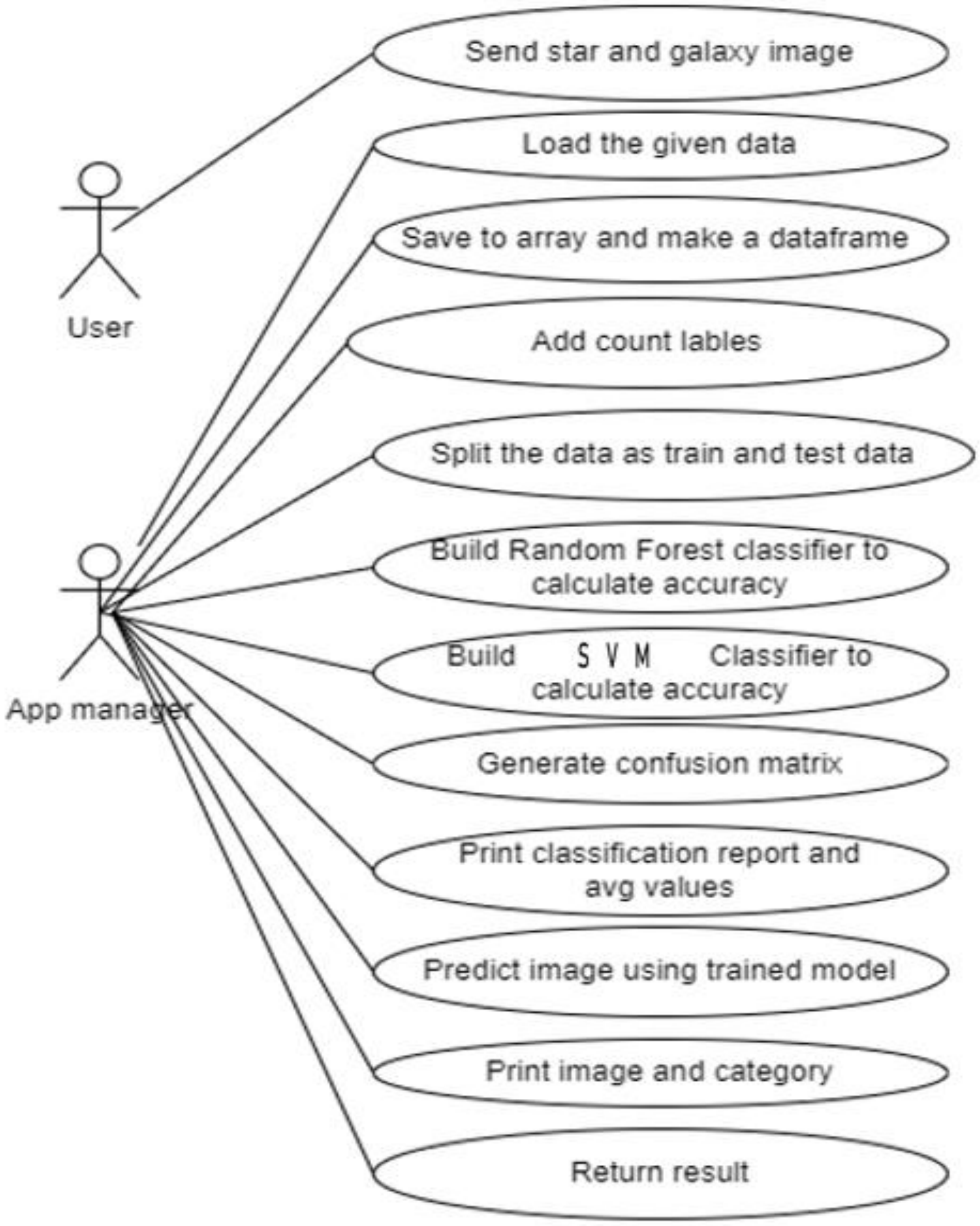

User
App manager
Send star and galaxy image
Load the given data
Save to array and make a dataframe
Add count lables
Split the data as train and test data
Build Random Forest classifier to calculate accuracy
Build SVM Classifier to calculate accuracy
Generate confusion matrix
Print classification report and avg values
Predict image using trained model
Print image and category
Return result

Fig 3.3.2 Use Case Diagram

3.3.3 Sequence diagram

A **sequence diagram** is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the Logical View of the system under development. Sequence diagrams are sometimes called **event diagrams**, **event scenarios**, and timing diagrams.

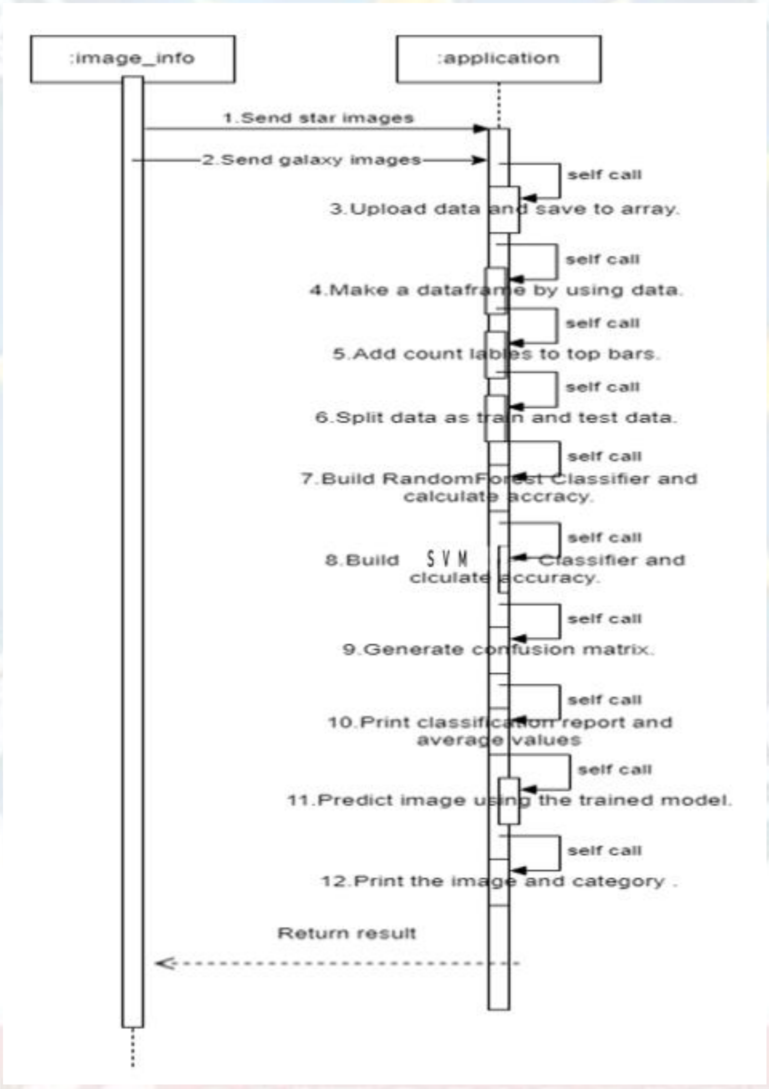
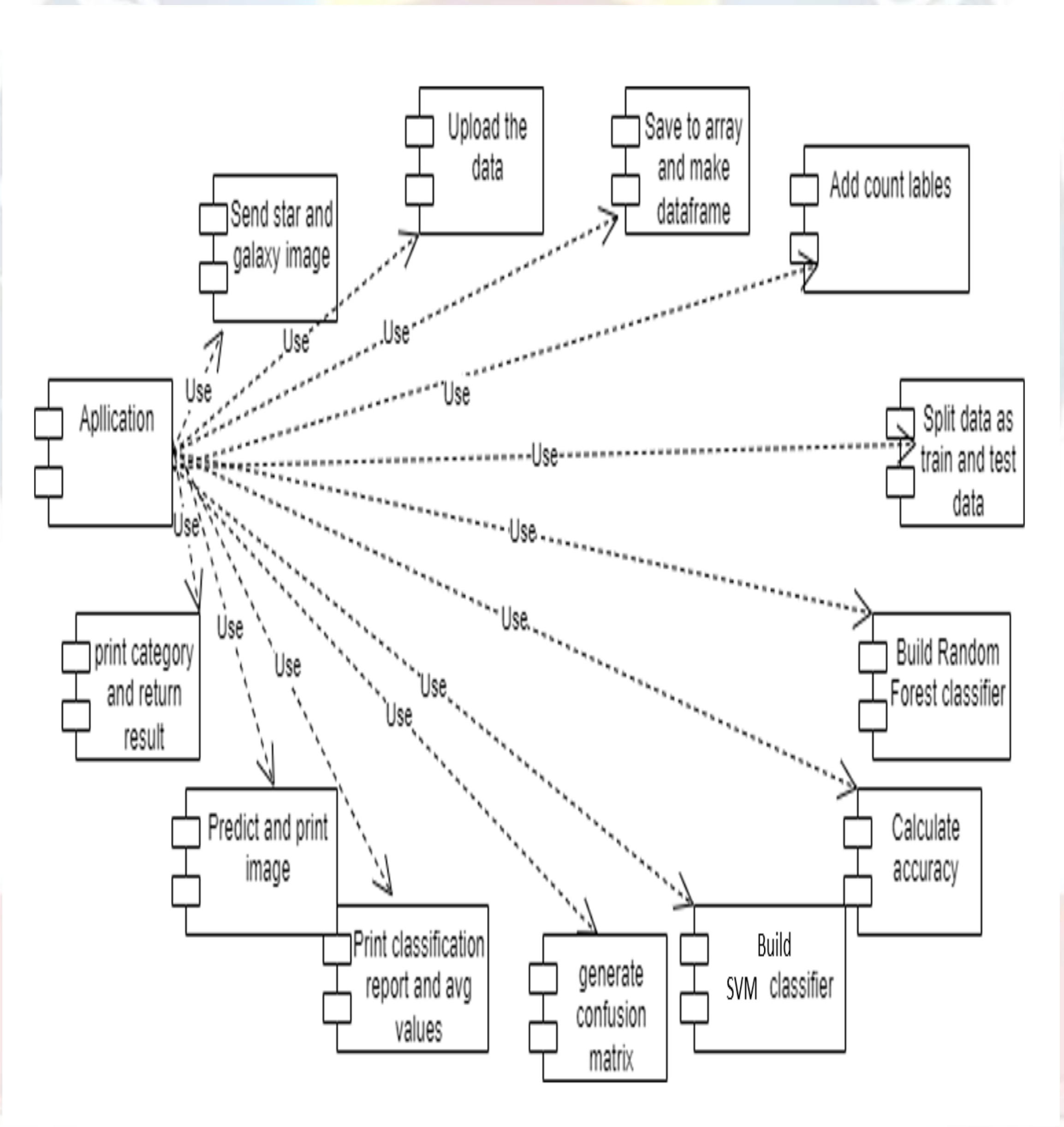

:image_info
:application
1.Send star images
2.Send galaxy images
self call
3.Upload data and save to array.
self call
4.Make a dataframe by using data.
self call
5.Add count lables to top bars.
self call
6.Split data as train and test data.
self call
7.Build RandomForest Classifier and calculate accracy.
self call
8.Build SVM Classifier and clculate accuracy.
self call
9.Generate confusion matrix.
self call
10.Print classification report and average values
self call
11.Predict image using the trained model.
self call
12.Print the image and category .
Return result

Fig 3.3.3 Sequence Diagram

3.3.4 Component Diagram

In the Unified Modelling Language, a **component diagram** depicts how components are wired together to form larger components and or software systems. They are used to illustrate the structure of arbitrarily complex systems.

Components are wired together by using an assembly connector to connect the required interface of one component with the provided interface of another component. This illustrates the service consumer - service provider relationship between the two components.



Upload the data
Save to array and make dataframe
Send star and galaxy image
Add count lables
Aplication
Split data as train and test data
Use
print category and return result
Build Random Forest classifier
Predict and print image
Calculate accuracy
Print classification report and avg values
generate confusion matrix
Build SVM classifier

Fig 3.3.4 Component Diagram

3.3.5 Deployment Diagram

A **deployment diagram** in the Unified Modelling Language models the *physical* deployment of artifacts on nodes. To describe a web site, for example, a deployment diagram would show what hardware components ("nodes") exist (e.g., a web server, an application server, and a database server), what software components ("artifacts") run on each node (e.g., web application, database), and how the different pieces are connected (e.g. JDBC, REST, RMI).

The nodes appear as boxes, and the artifacts allocated to each node appear as rectangles within the boxes. Nodes may have sub nodes, which appear as nested boxes. A single node in a deployment diagram may conceptually represent multiple physical nodes, such as a cluster of database servers.

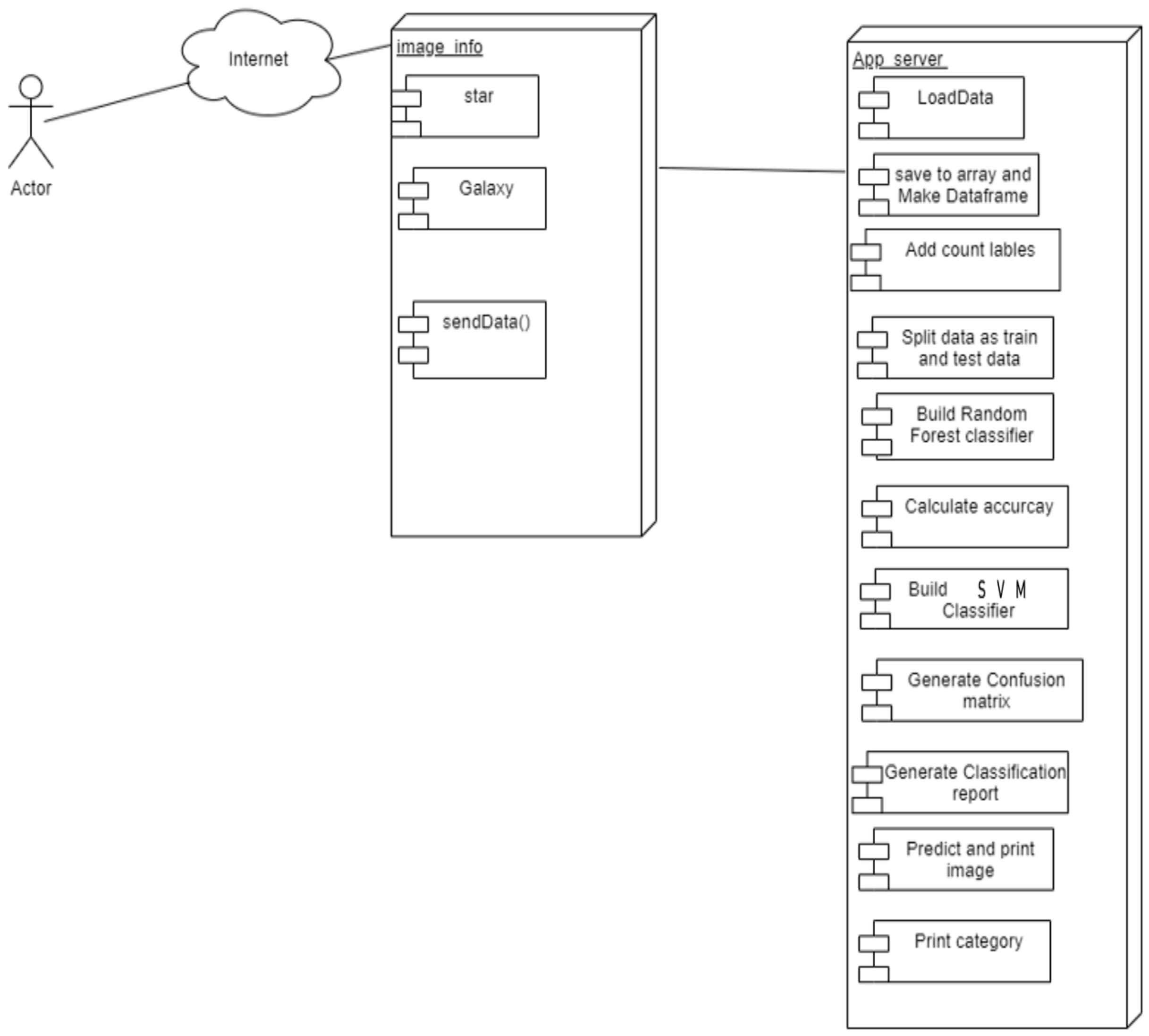
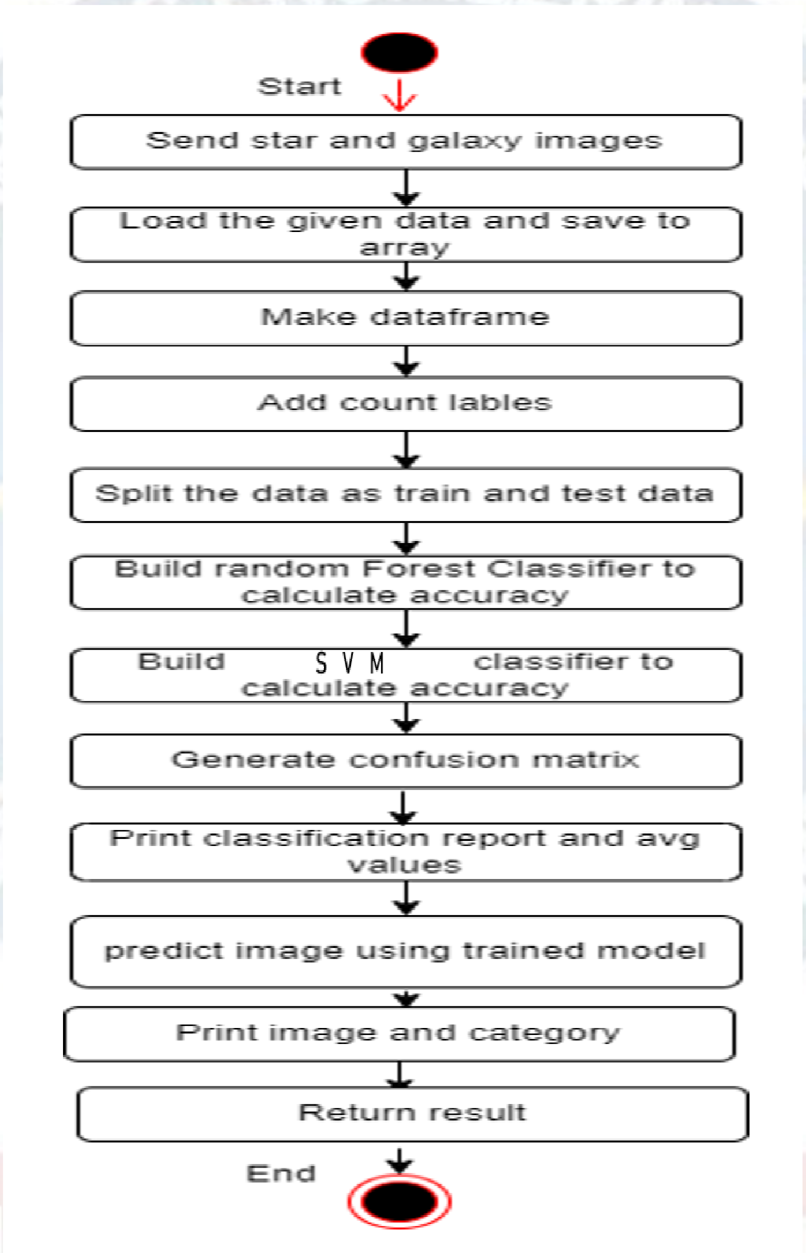


Fig 3.3.5 Deployment Diagram

3.3.6 Activity Diagram

Activity diagram is another important diagram in UML to describe dynamic aspects of the system. It is basically a flow chart to represent the flow form one activity to another activity. The activity can be described as an operation of the system. So, the control flow is drawn from one operation to another. This flow can be sequential, branched or concurrent.



Start
Send star and galaxy images
Load the given data and save to array
Make dataframe
Add count lables
Split the data as train and test data
Build random Forest Classifier to calculate accuracy
Build SVM classifier to calculate accuracy
Generate confusion matrix
Print classification report and avg values
predict image using trained model
Print image and category
Return result
End

Fig 3.3.6 Activity Diagram

3.3.7 Data Flow Diagram

Data flow diagrams illustrate how data is processed by a system in terms of inputs and outputs. Data flow diagrams can be used to provide a clear representation of any business function. This analysis can be carried out in precisely the level of detail required. The technique exploits a method called top-down expansion to conduct the analysis in a targeted way.

As the name suggests, Data Flow Diagram (DFD) is an illustration that explicates the passage of information in a process. A DFD can be easily drawn using simple symbols. Additionally, complicated processes can be easily automated by creating DFDs using easy- to-use, free downloadable diagramming tools. A DFD is a model for constructing and analysing information processes. DFD illustrates the flow of information in a process depending upon the inputs and outputs. A DFD can also be referred to as a Process Model. A DFD demonstrates business or technical process with the support of the outside data saved, plus the data flowing from the process to another and the end results.

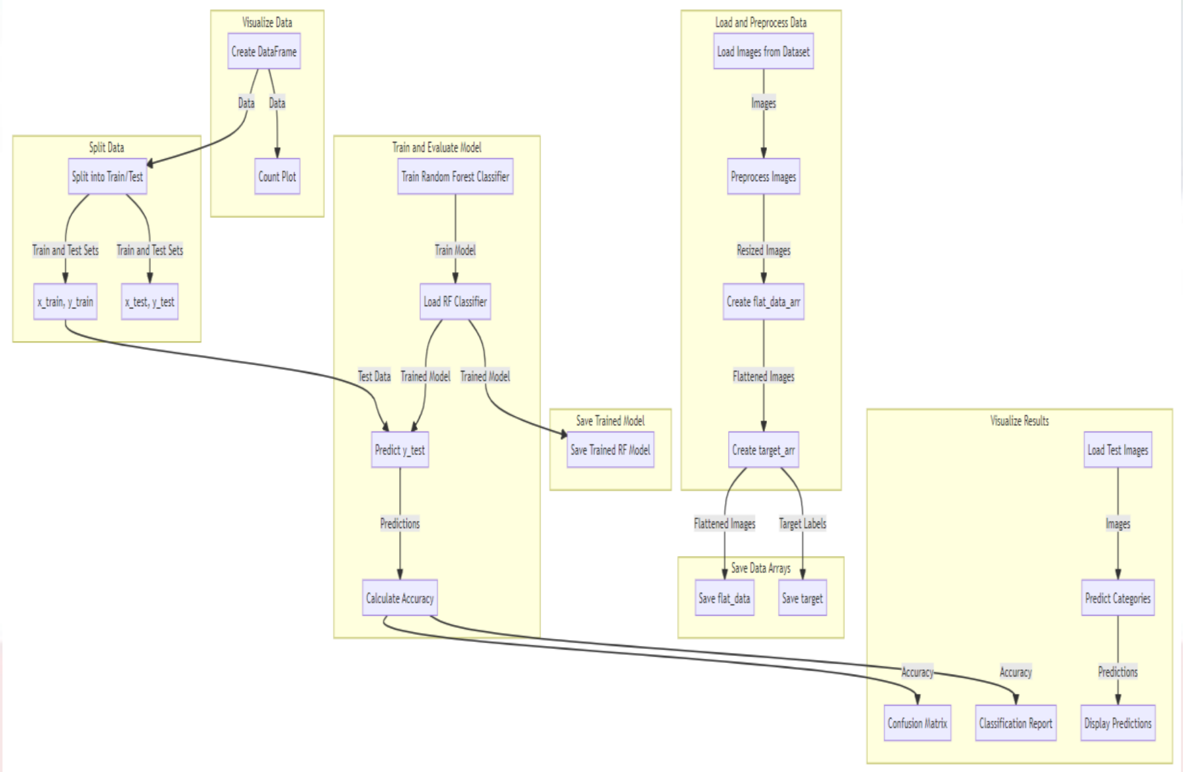


Fig 3.3.7 Dataflow Diagram

CHAPTER 4
SOFTWARE REQUIREMENT SPECIFICATION

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. Non-functional 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 analysing 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 (which could be one of several ways to accomplish a set of business requirements.)
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify. Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. 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 economic 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. It does not require any addition hardware or software. Since the interface for this system is developed using the existing resources and technologies available at NIC, there is nominal expenditure and economic feasibility for certain.

- **OPERATIONAL FEASIBILITY**

Proposed projects are beneficial only if they can be turned out into 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. This system is targeted to be in accordance with the above-mentioned issues. Beforehand, the management issues and user requirements have been taken into consideration. So there is no question of resistance from the users that can undermine the possible application benefits. The well-planned design would ensure the optimal utilization of the computer resources and would help in the improvement of performance status.

- **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 an easy access to the users. The database's purpose is to create, establish and maintain a workflow among various entities in order to facilitate all concerned users in their various capacities or roles. Permission to the users would be granted based on the roles specified. Therefore, it provides the technical guarantee of accuracy, reliability, and security.

4.1.1 External Interface Requirements

User Interface

The user interface of this system is a user-friendly python 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

HARDWARE REQUIREMENTS:

• Processor	-	Intel Core i3
• Speed	-	1.5 Ghz
• RAM	-	4 GB
• Hard Disk	-	250 GB

SOFTWARE REQUIREMENTS:

• Operating System	-	Windows
• Programming Language	-	Python IDLE 3.7 version (or) Anaconda 3.7

4.2 Random Forest Algorithm

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The below diagram explains the working of the Random Forest algorithm:

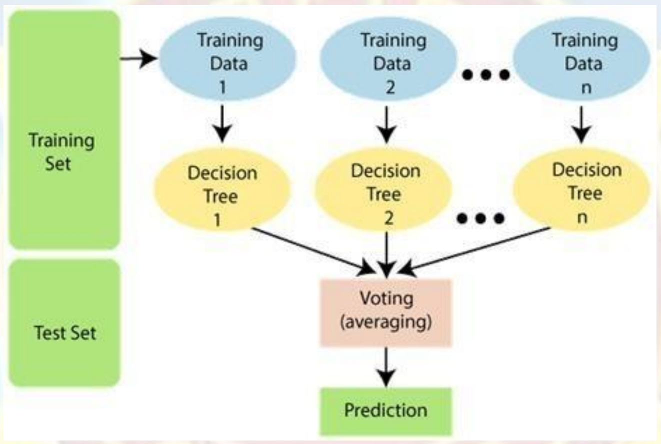


Fig 4.1 Random Forest Algorithm

Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random Forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

4.3 Types of Ensembles

Before understanding the working of the random forest, we must look into the ensemble technique. Ensemble simply means combining multiple models. Thus, a collection of models is used to make predictions rather than an individual model. Ensemble uses two types of methods:

Bagging– It creates a different training subset from sample training data with replacement & the final output is based on majority voting. For example, Random Forest. Bagging, also known as Bootstrap Aggregation is the ensemble technique used by random forest. Bagging chooses a random sample from the data set. Hence each model is generated from the samples (Bootstrap Samples) provided by the Original Data with replacement known as row sampling. This step of row sampling with replacement is called bootstrap. Now each model is trained independently which generates results. The final output is based on majority voting after combining the results of all models. This step which involves combining all the results and generating output based on majority voting is known as aggregation.

The below image is showing the logistic function:

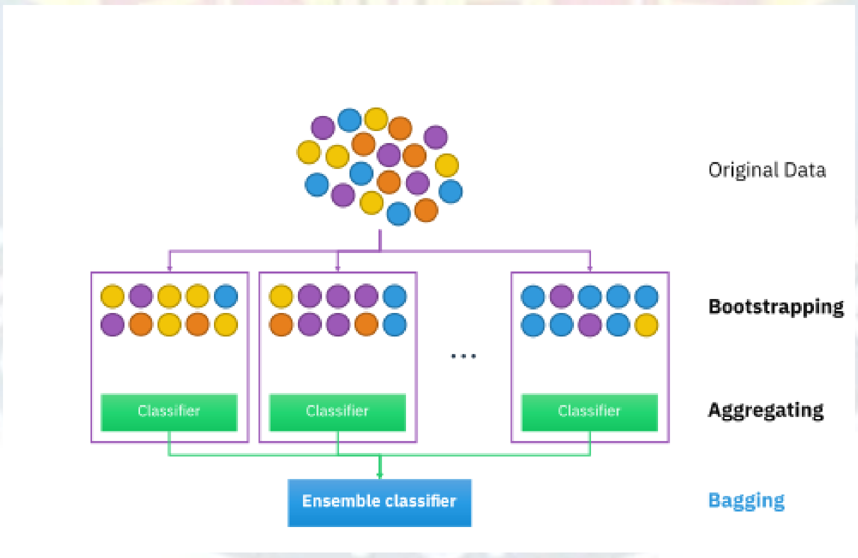


Fig. 4.2: RF Classifier analysis.

Boosting– It combines weak learners into strong learners by creating sequential models such that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST.

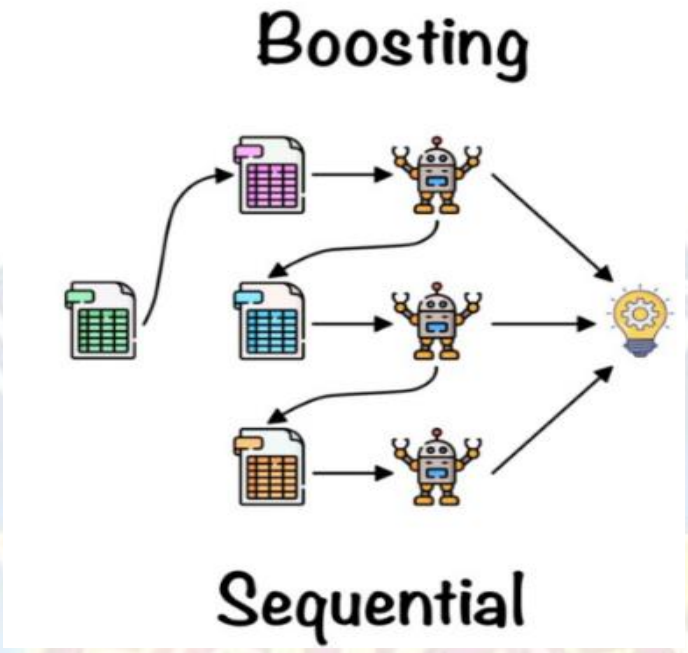


Fig. 4.3: Boosting RF Classifier.

4.4 Modules

1. Upload dataset
2. Image Pre-processing
3. Generate Train & Test Model
4. Perform prediction
5. Stars and Galaxies Classification
6. Classification report, confusion matrix, and accuracy

4.4.1 Upload dataset

Select and upload specified data set. Then the dataset may load into our project.

4.4.2 Image Pre-processing

Raw astronomical images are captured using telescopes and detectors. These images may be in various formats (e.g., FITS - Flexible Image Transport System) and often contain metadata such as exposure time and filter information. These images are later aligned in a standard frame size.

4.4.3 Generate Train & Test Model

Using this module, we will split dataset into train and test records. Application will used 80% data for training and 20% data for testing.

4.4.4 Perform prediction

In graph x-axis represents algorithm names and y-axis represents accuracy of those algorithms. From the graph we can conclude NN algorithm got highest accuracy.

4.4.5 Stars and Galaxies Classification

Once the RF model is trained and validated, it can be used for making predictions on new, unlabeled astronomical images.

4.4.6 Classification report, confusion matrix, and accuracy

We present a comprehensive evaluation of the machine learning model's performance, including a classification report detailing precision, recall, F1-score, and support for each class, a confusion matrix for visualizing the model's predictions, and the overall accuracy to gauge the model's overall correctness in classifying astronomical objects.

CHAPTER 5
IMPLEMENTATION

5.1 PYTHON

Python is a general-purpose language. It has a wide range of applications from Web development (like: Django and Bottle), scientific and mathematical computing (Orange, SymPy, NumPy) to desktop graphical user Interfaces (Pygame, Panda3D). The syntax of the language is clean, and the length of the code is relatively short. It's fun to work in Python because it allows you to think about the problem rather than focusing on the syntax.

5.1.1 History of Python

Python is an old language created by Guido Van Rossum. The design began in the late 1980s and was first released in February 1991.

5.1.2 Why Was Python Created?

In the late 1980s, Guido Van Rossum was working on the Amoeba distributed operating system group. He wanted to use an interpreted language like ABC (ABC has simple easy-to understand syntax) that could access the Amoeba system calls. So, he decided to create a language that was extensible. This led to the design of a new language which was later named Python

5.1.3 Why the Name Python?

No. It wasn't named after a dangerous snake. Rossum was fan of a comedy series from the late seventies. The name "Python" was adopted from the same series "Monty Python's Flying Circus".

5.1.4 Features of Python

A Simple Language Which Is Easier to Learn

Python has a very simple and elegant syntax. It's much easier to read and write Python programs compared to other languages like C++, Java, C#. Python makes programming fun and allows you to focus on the solution rather than syntax. If you are a newbie, it's a great choice to start your journey with Python.

Free And Open Source

You can freely use and distribute Python, even for commercial use. Not only can you use and distribute software’s written in it, but you can also even make changes to Python's source code. Python has a large community constantly improving it in each iteration.

Portability

You can move Python programs from one platform to another and run it without any changes. It runs seamlessly on almost all platforms including Windows, Mac OS X and Linux.

Extensible and Embeddable

Suppose an application requires high performance. You can easily combine pieces of C/C++ or other languages with Python code. This will give your application high performance as well as scripting capabilities which other languages may not provide out of the box.

A High-Level, Interpreted Language

Unlike C/C++, you don't have to worry about daunting tasks like memory management, garbage collection and so on. Likewise, when you run Python code, it automatically converts your code to the language your computer understands. You don't need to worry about any lower-level operations.

Large Standard Libraries to Solve Common Tasks

Python has several standard libraries which makes the life of a programmer much easier since you don't have to write all the code yourself. For example: Need to connect a MySQL database on a Web server? You can use the MySQL dB library using import MySQL db. Standard libraries in Python are well tested and used by hundreds of people. So, you can be sure that it won't break your application.

Object-Oriented

Everything in Python is an object. Object oriented programming (OOP) helps you solve a complex problem intuitively. With OOP, you can divide these complex problems into smaller sets by creating objects.

Expressiveness of the Language

Python allows you to write programs having greater functionality with fewer lines of code. Here's a link to the source code of the Tic-tac-toe game with a graphical interface and a smart computer opponent in less than 500 lines of code. This is just an example. You will be amazed how much you can do with Python once you learn the basics.

Great Community and Support

Python has a large supporting community. There are numerous active forums online which can be handy if you are stuck.

5.2 SOURCE CODE

```
import pandas as pd

import os

from skimage.transform import resize

from skimage.io import imread

import numpy as np

import matplotlib.pyplot as plt

from sklearn import svm

from sklearn.model_selection import

GridSearchCV

from sklearn.model_selection import

train_test_split

from sklearn.metrics import

accuracy_score

from sklearn.metrics import

classification_report,confusion_matrix

from skimage import io, transform

from sklearn import preprocessing

import numpy as np

import pickle

import seaborn as sns


Categories=['star','galaxy']

flat_data_arr=[] #input array

target_arr=[] #output array

datadir=r"Dataset"

flat_data_file = os.path.join(datadir,

'flat_data.npy')

target_file = os.path.join(datadir,

'target.npy')


if os.path.exists(flat_data_file) and

os.path.exists(target_file):
    # Load the existing arrays
    flat_data = np.load(flat_data_file)
    target = np.load(target_file)
else:
    #path which contains all the categories
of images
    for i in Categories:
        print(f'loading... category : {i}')
        path=os.path.join(datadir,i)
        for img in os.listdir(path):

img_array=imread(os.path.join(path,img))
img_resized=resize(img_array,(150,150,))
flat_data_arr.append(img_resized.flatten())
target_arr.append(Categories.index(i))
            print(f'loaded category:{i}
successfully')
            flat_data=np.array(flat_data_arr)
            target=np.array(target_arr)
    # Save the arrays
    np.save(os.path.join(datadir,
'flat_data.npy'), flat_data)
    np.save(os.path.join(datadir,
'target.npy'), target)

#dataframe
df=pd.DataFrame(flat_data)
df['Target']=target
df.shape
plt.figure(figsize=(8, 6))
ax = sns.countplot(data=df, x='Target')
plt.xlabel('Target', fontsize=12)
plt.ylabel('Count', fontsize=12)
plt.title('Count Plot for Target',
fontsize=14)

# Add count labels on top of the bars
for p in ax.patches:
    ax.annotate(f"{p.get_height()}",
                (p.get_x() + p.get_width() / 2.,
p.get_height()),
                ha='center', va='center',
fontsize=11, color='black', xytext=(0, 5),
                textcoords='offset points')

plt.show()
#input data
x=df.iloc[:,:-1]
#output data
y=df.iloc[:,-1]
y

# Splitting the data into training and
esting sets
x_train,x_test,y_train,y_test=train_test_sp
lit(x,y,test_size=0.20,random_state=77)
from sklearn.svm import SVC
filename='Svm_Classifier.pkl'
if os.path.exists('Svm_Classifier.pkl'):
    # Load the trained model from the
Pickle file
    with open (filename, 'rb') as
Svm_Model_pkl:
        svc = pickle.load(Svm_Model_pkl)
        y_pred1=svc.predict(x_test)
```

```
Acc=accuracy_score(y_test,y_pred1)*100
        print("Accuracy",Acc)
else:
    svc = SVC(kernel='linear')  # You can
also try other kernels like 'rbf' or 'poly'
    # Train the SVM classifier on the
training data
    svc.fit(x_train, y_train)
    y_pred1=svc.predict(x_test)

Acc=accuracy_score(y_test,y_pred1)*100
    print("Accuracy",Acc)
    # Dump the trained svm classifier with
Pickle
    filename = 'Svm_Classifier.pkl'
    # Open the file to save as pkl file
    Svm_Model_pkl = open(filename, 'wb')
    pickle.dump(svc, Svm_Model_pkl)
    # Close the pickle instances
    Svm_Model_pkl.close()

Acc=accuracy_score(y_test,y_pred1)*100
print("Accuracy",Acc)

cm=confusion_matrix(y_test,y_pred1)
cm

report=classification_report(y_test,y_pre)
print('ClassificationReport:\n\n',report)

class_labels=['star','galaxy']
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d",
cmap="Blues", xticklabels=class_labels,
yticklabels=class_labels)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Support vector Classifier
Confusion Matrix")
plt.show()

from sklearn.ensemble import
RandomForestClassifier

rf=RandomForestClassifier()

if os.path.exists('RF_Classifier.pkl'):
    # Load the trained model from the
Pickle file
    with open(filename, 'rb') as
RF_Model_pkl:
        rf = pickle.load(RF_Model_pkl)
        y_pred=rf.predict(x_test)

Acc=accuracy_score(y_test,y_pred)*100
        print("Accuracy",Acc)
else:
    rf.fit(x_train,y_train)
    y_pred=rf.predict(x_test)

Acc=accuracy_score(y_test,y_pred)*100
    print("Accuracy",Acc)
    # Dump the trained Naive Bayes
classifier with Pickle
    filename = 'RF_Classifier.pkl'
    # Open the file to save as pkl file
    RF_Model_pkl = open(filename, 'wb')
    pickle.dump(rf, RF_Model_pkl)
    # Close the pickle instances
    RF_Model_pkl.close()
Acc=accuracy_score(y_test,y_pred)*100
print("Accuracy",Acc)
cm=confusion_matrix(y_test,y_pred)
cm
class_labels=['star','galaxy']
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt="d",
cmap="Blues", xticklabels=class_labels,
yticklabels=class_labels)
plt.xlabel("Predicted Label")
plt.ylabel("True Label")
plt.title("Confusion Matrix")
plt.show()

report=classification_report(y_test,y_pre)
print(report)
path = r"test images"
Categories = {0:'star',1:'galaxy'} # Define
your categories with corresponding labels
for filename in os.listdir(path):
    img_path = os.path.join(path, filename)
# Construct the complete image path
    img = imread(img_path)

    plt.imshow(img)
    plt.show()

    img_resize = resize (img, (150, 150, 3))
    l = [img_resize.flatten()]
    # Make predictions using your pre-
trained model
    prediction = rf.predict(l)[0]
    predicted_category =
Categories[prediction]
    print ("The predicted image is:",
predicted_category)
```

CHAPTER 6
EXPERIMENTAL RESULTS

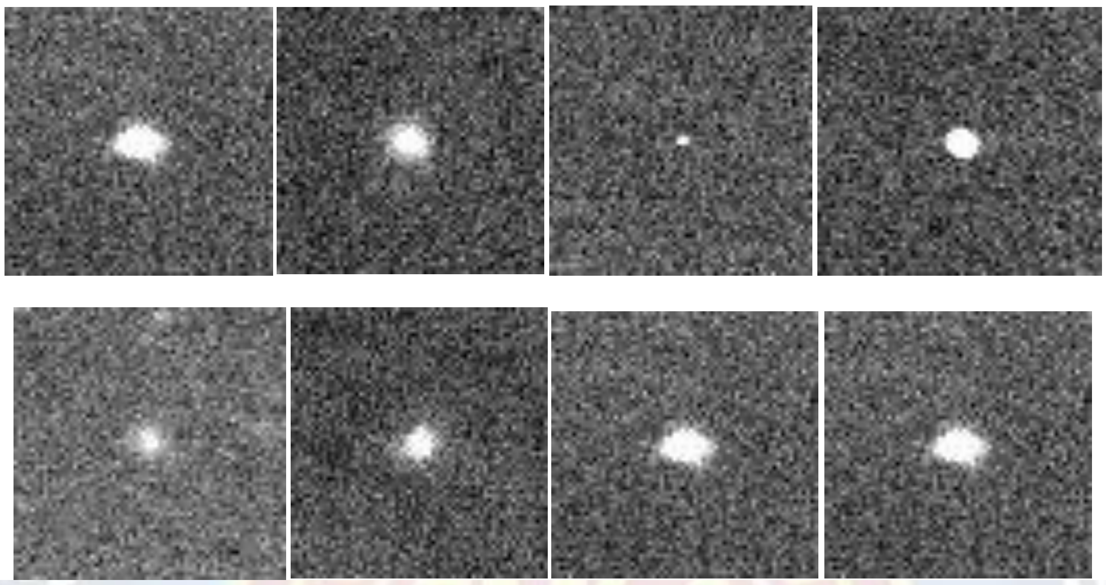


Fig 6.1 Sample images from dataset with star class.

Sample images from the dataset with the "star" class typically consist of astronomical images capturing various types of stars in the night sky. These images can include bright points of light representing stars of different sizes and colors.

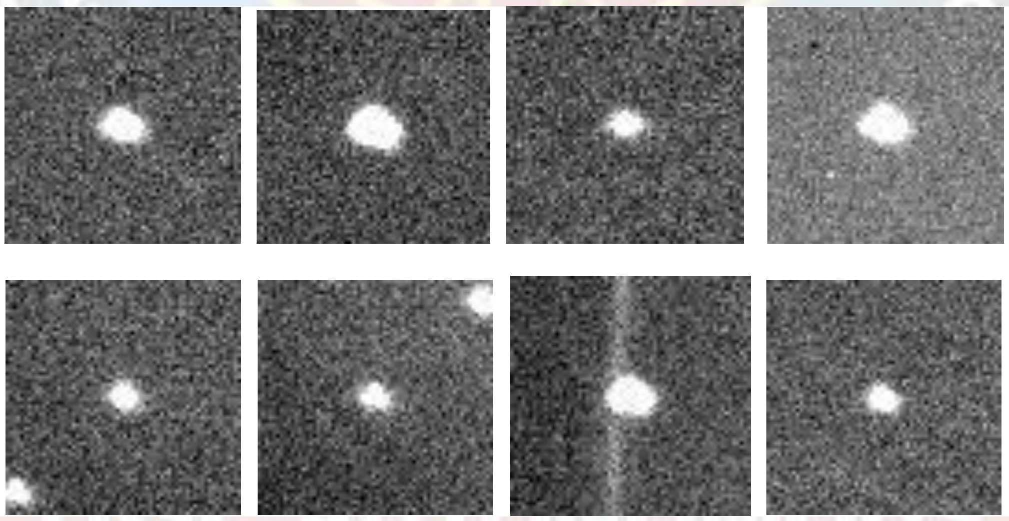
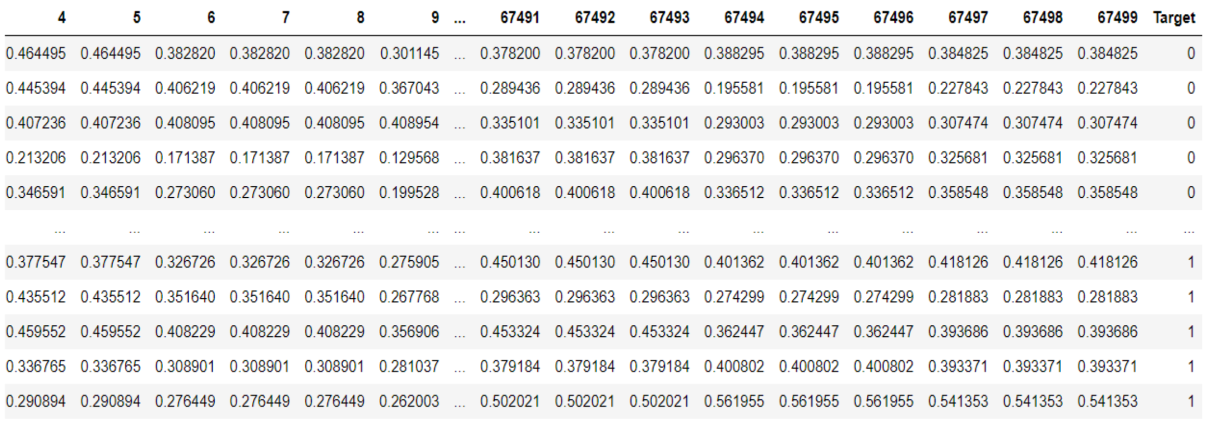


Fig 6.2 Sample images from dataset with galaxy class.

Sample images from the dataset with the "galaxy" class typically showcase various galaxies observed in the cosmos. These images may reveal spiral galaxies with distinctive arms, elliptical galaxies with smooth and rounded appearances, irregular galaxies with chaotic shapes, and even distant galaxies whose light has traveled millions of years to reach us.



4	5	6	7	8	9	...	67491	67492	67493	67494	67495	67496	67497	67498	67499	Target
0.464495	0.464495	0.382820	0.382820	0.382820	0.301145	...	0.378200	0.378200	0.378200	0.388295	0.388295	0.388295	0.384825	0.384825	0.384825	0
0.445394	0.445394	0.406219	0.406219	0.406219	0.367043	...	0.289436	0.289436	0.289436	0.195581	0.195581	0.195581	0.227843	0.227843	0.227843	0
0.407236	0.407236	0.408095	0.408095	0.408095	0.408954	...	0.335101	0.335101	0.335101	0.293003	0.293003	0.293003	0.307474	0.307474	0.307474	0
0.213206	0.213206	0.171387	0.171387	0.171387	0.129568	...	0.381637	0.381637	0.381637	0.296370	0.296370	0.296370	0.325681	0.325681	0.325681	0
0.346591	0.346591	0.273060	0.273060	0.273060	0.199528	...	0.400618	0.400618	0.400618	0.336512	0.336512	0.336512	0.358548	0.358548	0.358548	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
0.377547	0.377547	0.326726	0.326726	0.326726	0.275905	...	0.450130	0.450130	0.450130	0.401362	0.401362	0.401362	0.418126	0.418126	0.418126	1
0.435512	0.435512	0.351640	0.351640	0.351640	0.267768	...	0.296363	0.296363	0.296363	0.274299	0.274299	0.274299	0.281883	0.281883	0.281883	1
0.459552	0.459552	0.408229	0.408229	0.408229	0.356906	...	0.453324	0.453324	0.453324	0.362447	0.362447	0.362447	0.393686	0.393686	0.393686	1
0.336765	0.336765	0.308901	0.308901	0.308901	0.281037	...	0.379184	0.379184	0.379184	0.400802	0.400802	0.400802	0.393371	0.393371	0.393371	1
0.290894	0.290894	0.276449	0.276449	0.276449	0.262003	...	0.502021	0.502021	0.502021	0.561955	0.561955	0.561955	0.541353	0.541353	0.541353	1

Fig 6.3 Sample dataset after preprocessing.

A sample dataset after preprocessing is a cleaner, more structured, and well-organized collection of data that has undergone several critical transformations.

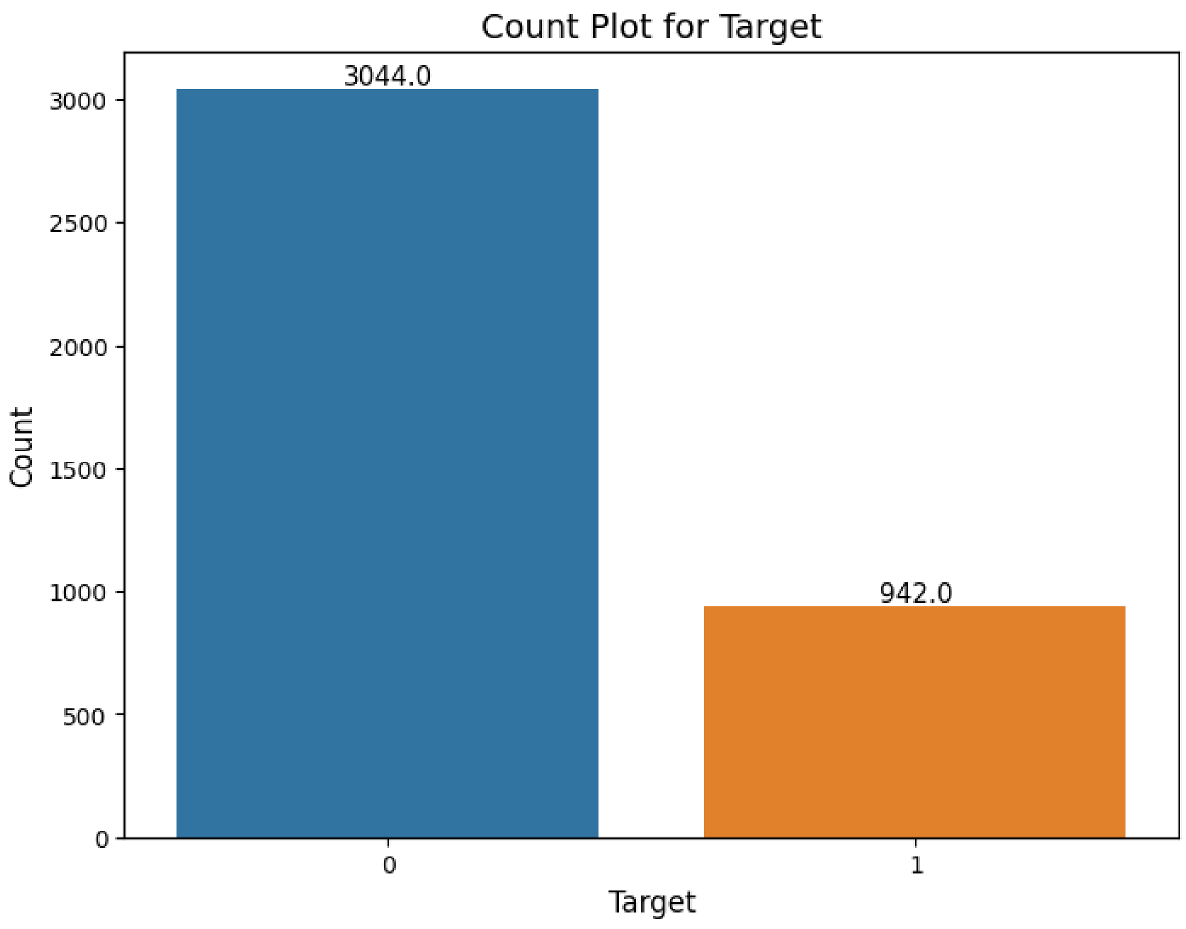
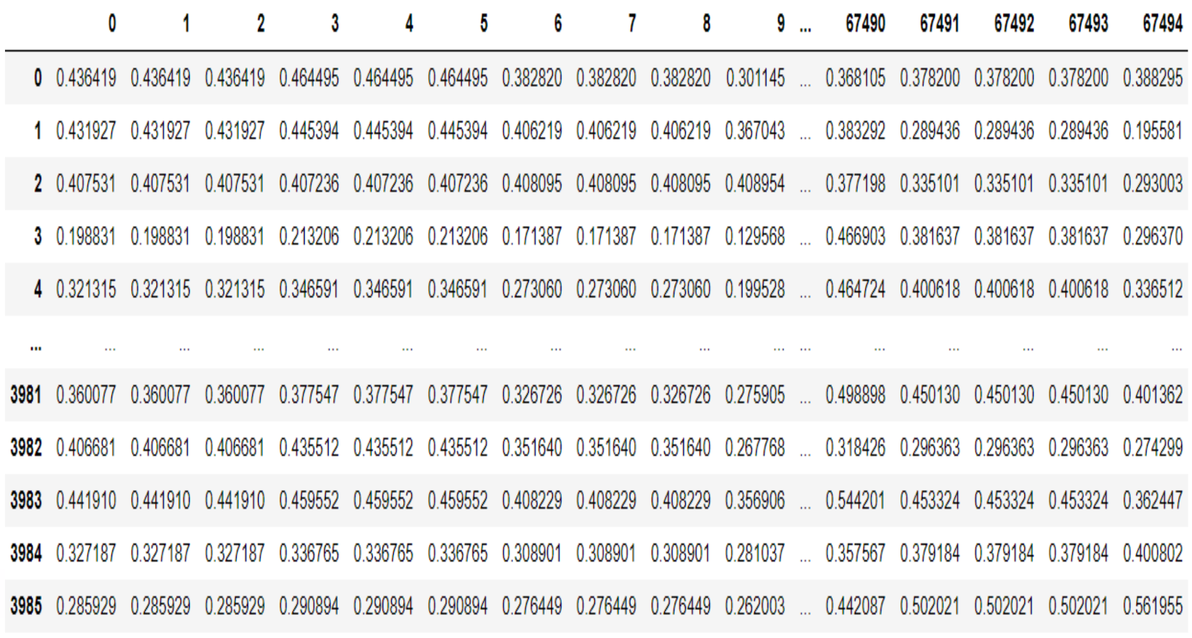


Fig 6.4 Count plot for Target Column.

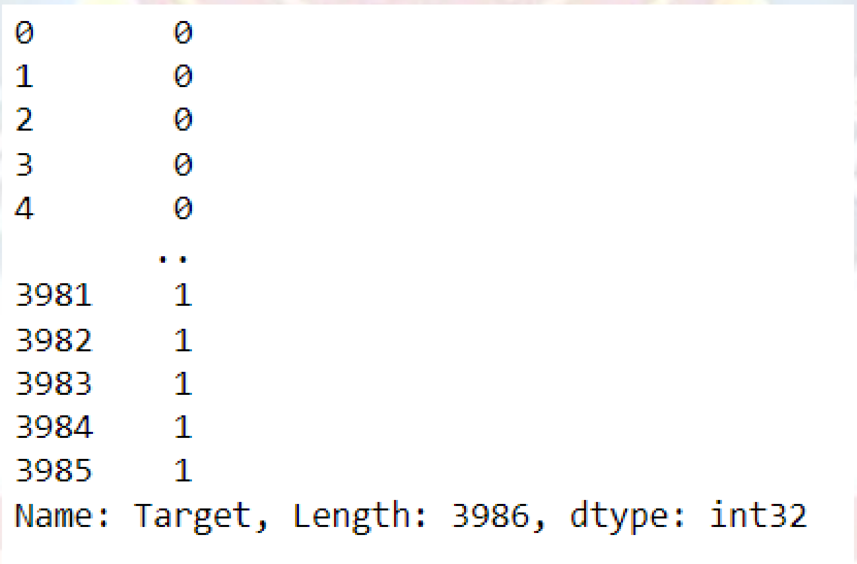
A count plot for the target column is a visual representation that displays the frequency of each unique category or class within the target variable. It's a useful visualization for understanding the distribution of classes in a classification problem. Here 0=Stars and 1=Galaxies.



	0	1	2	3	4	5	6	7	8	9	...	67490	67491	67492	67493	67494
0	0.436419	0.436419	0.436419	0.464495	0.464495	0.464495	0.382820	0.382820	0.382820	0.301145	...	0.368105	0.378200	0.378200	0.378200	0.388295
1	0.431927	0.431927	0.431927	0.445394	0.445394	0.445394	0.406219	0.406219	0.406219	0.367043	...	0.383292	0.289436	0.289436	0.289436	0.195581
2	0.407531	0.407531	0.407531	0.407236	0.407236	0.407236	0.408095	0.408095	0.408095	0.408954	...	0.377198	0.335101	0.335101	0.335101	0.293003
3	0.198831	0.198831	0.198831	0.213206	0.213206	0.213206	0.171387	0.171387	0.171387	0.129568	...	0.466903	0.381637	0.381637	0.381637	0.296370
4	0.321315	0.321315	0.321315	0.346591	0.346591	0.346591	0.273060	0.273060	0.273060	0.199528	...	0.464724	0.400618	0.400618	0.400618	0.336512
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
3981	0.360077	0.360077	0.360077	0.377547	0.377547	0.377547	0.326726	0.326726	0.326726	0.275905	...	0.498898	0.450130	0.450130	0.450130	0.401362
3982	0.406681	0.406681	0.406681	0.435512	0.435512	0.435512	0.351640	0.351640	0.351640	0.267768	...	0.318426	0.296363	0.296363	0.296363	0.274299
3983	0.441910	0.441910	0.441910	0.459552	0.459552	0.459552	0.408229	0.408229	0.408229	0.356906	...	0.544201	0.453324	0.453324	0.453324	0.362447
3984	0.327187	0.327187	0.327187	0.336765	0.336765	0.336765	0.308901	0.308901	0.308901	0.281037	...	0.357567	0.379184	0.379184	0.379184	0.400802
3985	0.285929	0.285929	0.285929	0.290894	0.290894	0.290894	0.276449	0.276449	0.276449	0.262003	...	0.442087	0.502021	0.502021	0.502021	0.561955

Fig 6.5 Data frame of input images after preprocessing.

A data frame of input images after preprocessing typically represents a structured and standardized collection of images in a tabular format. Each row of the data frame corresponds to an image, and columns may include features extracted from the images, metadata, or labels if applicable.



```
0        0
1        0
2        0
3        0
4        0
        ..
3981     1
3982     1
3983     1
3984     1
3985     1
Name: Target, Length: 3986, dtype: int32
```

Fig 6.6 Target Column (Star = 0, and Galaxy = 1).

In a dataset with a binary target column where "Star" is assigned the value 0 and "Galaxy" is assigned the value 1, this column serves as the ground truth or label for each data point. The target column's purpose is to indicate the class or category to which each observation belongs. In this context, a value of 0 signifies that the data point represents a star, while a value of 1 indicates that it represents a galaxy.

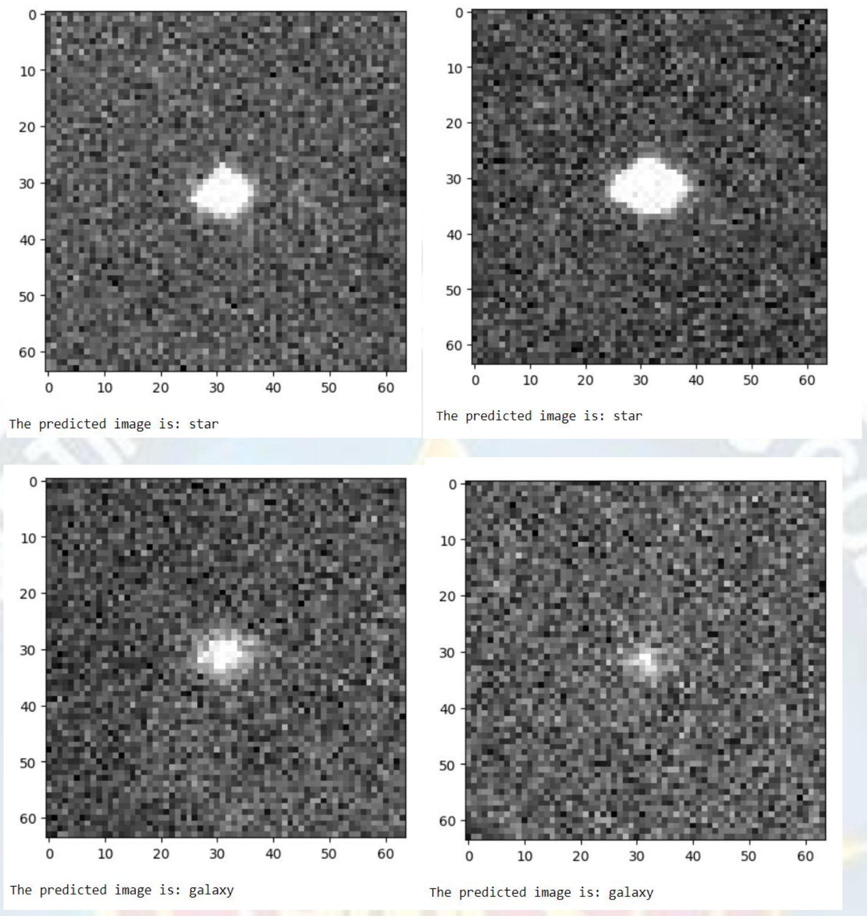

```
The predicted image is: star
The predicted image is: star
The predicted image is: galaxy
The predicted image is: galaxy
```

Fig 6.7 Sample prediction on test data using proposed ML model.

A sample prediction on test data using the proposed machine learning model involves applying the trained model to unseen data to make predictions or classifications. For instance, in a binary classification scenario with "Star" and "Galaxy" classes, the model predicts whether a given test image represents a star or a galaxy.

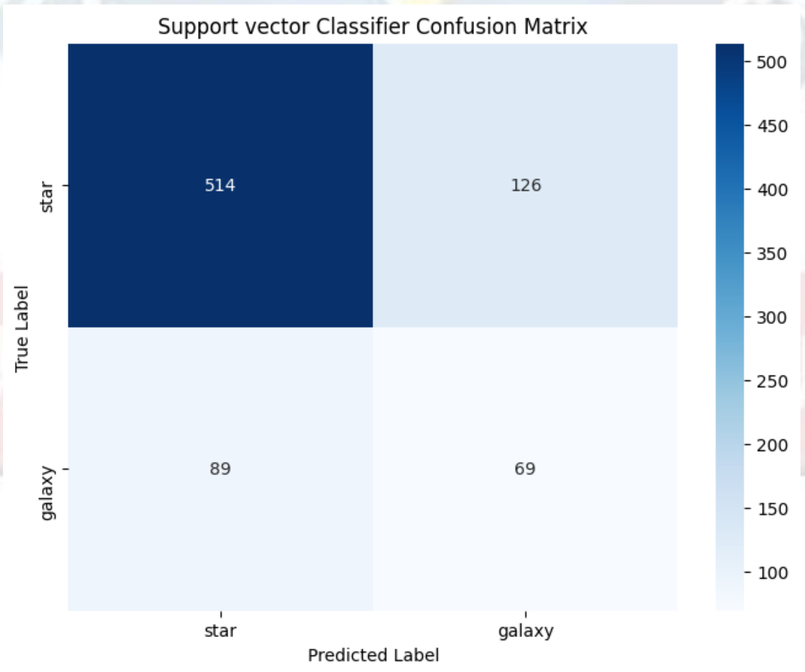
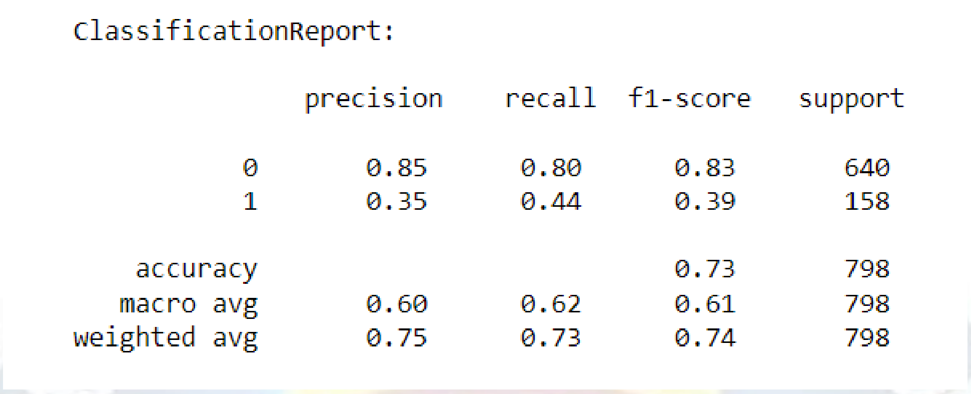


Fig 6.8 Confusion matrix for Support vector classifier model.

A confusion matrix for a Support Vector Classifier (SVC) model is a tabular representation that assesses the model's performance in a binary classification task.



```
ClassificationReport:

              precision    recall  f1-score   support

           0       0.85      0.80      0.83       640
           1       0.35      0.44      0.39       158

    accuracy                           0.73       798
   macro avg       0.60      0.62      0.61       798
weighted avg       0.75      0.73      0.74       798
```

Fig 6.9 Classification report of Support vector classifier model.

A classification report for a Support Vector Classifier (SVC) model is a concise summary of its performance metrics for each class in a binary or multiclass classification task.

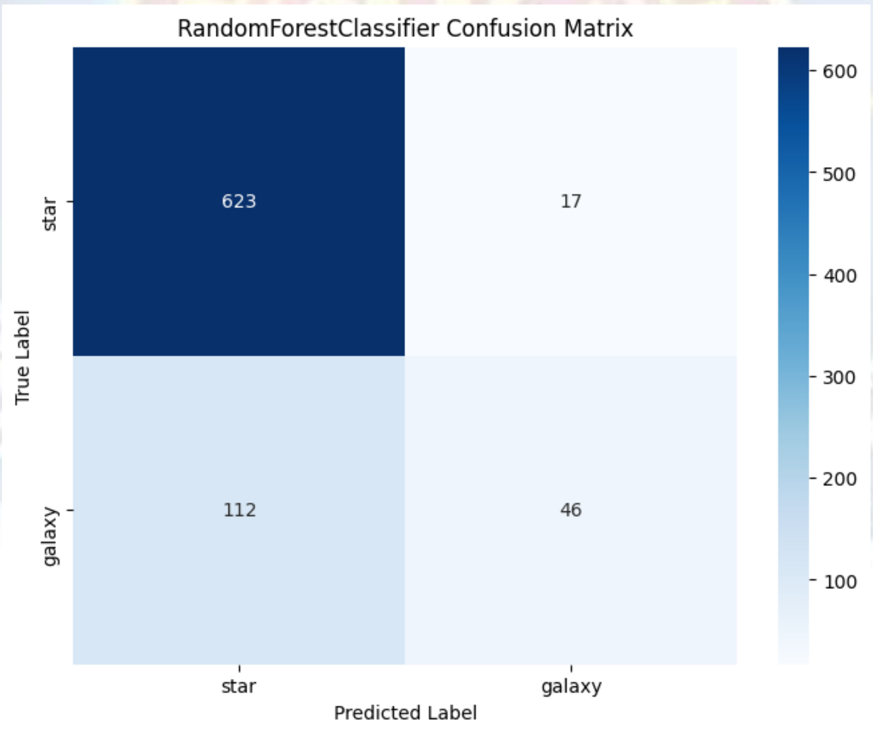
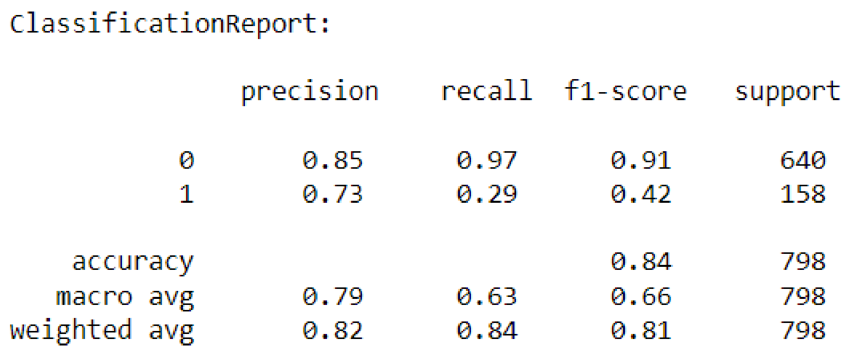


Fig 6.10 Confusion matrix of proposed Random Forest Classifier model.

The confusion matrix for a proposed Random Forest Classifier model is a visual representation that quantifies the model's performance in a classification task.



```
ClassificationReport:

              precision    recall  f1-score   support

           0       0.85      0.97      0.91       640
           1       0.73      0.29      0.42       158

    accuracy                           0.84       798
   macro avg       0.79      0.63      0.66       798
weighted avg       0.82      0.84      0.81       798
```

Fig 6.11 Classification report of proposed model.

A classification report for a proposed model provides a comprehensive evaluation of its performance across multiple metrics in a classification task.

CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENT

7.1 Conclusion

In conclusion, the classification of stars and galaxies from astronomical images involves a multifaceted process, beginning with meticulous image preprocessing steps, which are crucial to ensure data quality and feature extraction. Subsequently, Random Forest (RF) machine learning models are trained using labelled datasets and extracted features to learn the relationships between image characteristics and object classes. These models can effectively classify celestial objects within the images. During prediction, the RF model computes probability scores or class labels for new images, with post-processing steps applied as needed, and the results are analyzed to advance our understanding of the universe, catalogue celestial objects, or identify rare phenomena. The success of this process hinges on the accuracy of preprocessing steps, the informativeness of extracted features, and the robustness of the machine learning model, collectively contributing to valuable insights in the field of astronomy and astrophysics.

7.2 Future Enhancement

The future scope in the field of classifying stars and galaxies from astronomical images is promising and involves several exciting developments:

- **Advanced Machine Learning Models:** Future research will likely focus on exploring more advanced machine learning models, including deep learning architectures such as Convolutional Neural Networks (CNNs). These models can automatically learn hierarchical features from images and may outperform traditional machine learning algorithms like Random Forests.

- **Multi-wavelength Data:** Combining data from multiple wavelengths (e.g., optical, infrared, radio) will provide a more comprehensive view of celestial objects. Future classification systems will need to integrate and analyse multi-wavelength data, leading to more accurate and informative classifications.

- **Big Data Challenges:** As astronomical surveys generate vast amounts of data, future classification systems will need to address big data challenges, including data storage, processing, and analysis. Scalable and efficient algorithms will be essential.

- **Automated Pipeline Development:** The development of automated data preprocessing and analysis pipelines, including data reduction, calibration, feature extraction, and model training, will become increasingly important to handle the volume of data generated by next-generation telescopes and surveys.

- **Real-time Classification:** There is a growing demand for real-time classification systems that can process and classify astronomical events and objects as they are observed. Such systems are critical for rapidly responding to transient events like supernovae or gravitational wave detections.
- **Citizen Science Involvement:** Citizen science initiatives, where amateur astronomers and volunteers contribute to data labelling and classification, will likely play a significant role. Future systems may leverage crowdsourcing to handle large datasets and complex classifications.
- **Exoplanet Characterization:** The study of exoplanets is an evolving field. Future classification tasks may extend beyond stars and galaxies to include the characterization of exoplanets, their atmospheres, and habitability potential.
- **AI Explainability:** As machine learning models become more complex, understanding their decision-making processes (AI explainability) will be crucial, especially in scientific contexts. Researchers will work on developing methods to interpret model predictions.
- **Interdisciplinary Collaboration:** Collaboration between astronomers, computer scientists, and data scientists will be essential to advance classification methods and address complex astrophysical questions. Interdisciplinary teams can bring diverse expertise to the field.
- **Applications Beyond Classification:** Image analysis techniques developed for star and galaxy classification may find applications in other fields, such as medical imaging, remote sensing, and object detection, contributing to a broader range of scientific and industrial applications.
- **Quantum Computing:** The advent of quantum computing may offer new opportunities for optimizing complex classification algorithms and handling large-scale astronomical datasets more efficiently.

REFERENCES

[1] Merz, G. M., et al. "Detection, Instance Segmentation, and Classification for Astronomical Surveys with Deep Learning (DeepDISC): Detectron2 Implementation and Demonstration with Hyper Suprime-Cam Data." arXiv preprint arXiv:2307.05826 (2023).

[2] Smirnov, Anton A., et al. "Prospects for future studies using deep imaging: analysis of individual Galactic cirrus filaments." Monthly Notices of the Royal Astronomical Society 519.3 (2023): 4735-4752.

[3] Ćiprijanović, A., et al. "DeepAstroUDA: semi-supervised universal domain adaptation for cross-survey galaxy morphology classification and anomaly detection." Machine Learning: Science and Technology 4.2 (2023): 025013.

[4] Solorio-Ramírez, José-Luis, et al. "Random Forest Algorithm for the Classification of Spectral Data of Astronomical Objects." Algorithms 16.6 (2023): 293.

[5] Bruun, S. H., A. Agnello, and J. Hjorth. "VarIabiLity seLection of AstrophysIcal sources iN PTF (VILLAIN) I. Structure function fits to 71 million objects." arXiv preprint arXiv:2304.09903 (2023).

[6] Kumar, Rahul, Md Kamruzzaman Sarker, and Sheikh Rabiul Islam. "Vision Transformers for Galaxy Morphology Classification: Fine-Tuning Pre-trained Networks vs. Training from Scratch." International Conference on Deep Learning Theory and Applications. Cham: Springer Nature Switzerland, 2023.

[7] de los Rios, Martín, et al. "Determining the dark matter distribution in simulated galaxies with deep learning." Monthly Notices of the Royal Astronomical Society (2023): stad2614.

[8] Jia, Yongle, et al. "Identifying symbiotic stars with machine learning." Research in Astronomy and Astrophysics (2023).

[9] Duann, Yi, Yong Tian, and Chung-Ming Ko. "Classifying MaNGA Velocity Dispersion Profiles by Machine Learning." RAS Techniques and Instruments (2023): rzad044.

[10] Solorio-Ramírez, José-Luis, et al. "Random Forest Algorithm for the Classification of Spectral Data of Astronomical Objects." Algorithms 16.6 (2023): 293.

[11] Mehta, Tanvi, Nishi Bhuta, and Swati Shinde. "Experimental Analysis of Stellar Classification by using Different Machine Learning Algorithms." 2022 International Conference on Industry 4.0 Technology (I4Tech). IEEE, 2022.

[12] Mahalakshmi, G. S., et al. "Classification and Feature Prediction of Star, Galaxies, Quasars, and Galaxy Morphologies Using Machine Learning." (2022).

[13] Ćiprijanović, Aleksandra, et al. "DeepAdversaries: examining the robustness of deep learning models for galaxy morphology classification." Machine Learning: Science and Technology 3.3 (2022): 035007.

[14] Lake, Sean E., and C-W. Tsai. "An exploration of how training set composition bias in machine learning affects identifying rare objects." Astronomy and Computing 40 (2022): 100617.

[15] Guo, Xiaoyu, et al. "Unsupervised clustering and analysis of WISE spiral galaxies." Monthly Notices of the Royal Astronomical Society 517.2 (2022): 1837-1848.

[16] Biswas, Paul. An Assessment of Machine Learning Algorithms as Applied to Astronomical Data Sets. Diss. Texas A&M University-Commerce, 2022.

[17] Cunha, P. A. C., and A. Humphrey. "Photometric redshift-aided classification using ensemble learning." Astronomy & Astrophysics 666 (2022): A87.

[18] Inoue, Shigeki, et al. "Classification of cosmic structures for galaxies with deep learning: connecting cosmological simulations with observations." Monthly notices of the royal astronomical society 515.3 (2022): 4065-4081.

[19] Doorenbos, Lars, et al. "ulisse: A tool for one-shot sky exploration and its application for detection of active galactic nuclei." Astronomy & Astrophysics 666 (2022): A171.

[20] Lee, Danial Ahmad Ariffin, et al. "Identifying Merging Galaxies in MaNGA using H-α and Hi Observations." Sains Malaysiana 51.7 (2022): 2187-2196.

[21] Ramachandra, Nesar, et al. "Machine learning synthetic spectra for probabilistic redshift estimation: SYTH-Z." Monthly Notices of the Royal Astronomical Society 515.2 (2022): 1927-1941.

[22] Ciprijanovic, Aleksandra, et al. "DeepAdversaries: examining the robustness of deep learning models for galaxy morphology classification." Machine Learning: Science and Technology 3.035007 (2022): 2632-2153.

[23] Bluck, Asa FL, et al. "The quenching of galaxies, bulges, and disks since cosmic noon-A machine learning approach for identifying causality in astronomical data." Astronomy & Astrophysics 659 (2022): A160.

[24] Kembhavi, Ajit, and Rohan Pattnaik. "Machine learning in astronomy." Journal of Astrophysics and Astronomy 43.2 (2022): 76.

[25] Wilde, Joshua, et al. "Detecting gravitational lenses using machine learning: exploring interpretability and sensitivity to rare lensing configurations." Monthly Notices of the Royal Astronomial Society 512.3 (2022): ss3464-34.