Report on

Machine Learning Project

On

Exoplanet Habitability Prediction

Submitted in partial fulfillment of the requirements for the award of degree of

Bachelor of Technology (B. Tech)

Computer Science and Engineering

Specialization in Data Science and Machine Learning

Submitted to

LOVELY PROFESSIONAL UNIVERSITY

PHAGWARA, PUNJAB

SUBMITTED BY

Name of the student: Pranjal Sinha

Registration Number: 12206214

INTRODUCTION

The search for habitable exoplanets is one of the most fascinating and significant endeavors in modern astronomy and planetary science. As our understanding of the universe expands, so does our curiosity about whether life can exist beyond Earth. The discovery of exoplanets—planets orbiting stars outside our solar system—has provided us with countless opportunities to explore the possibility of extraterrestrial habitability. With thousands of exoplanets discovered so far, astronomers and scientists have developed various methods to assess whether these distant worlds could potentially support life. One of the key aspects of this exploration is determining an exoplanet's habitability based on its physical and chemical characteristics, such as its distance from its host star, atmospheric composition, temperature, and presence of liquid water.

The field of exoplanet habitability relies heavily on data analysis and machine learning techniques to classify and predict whether an exoplanet has the conditions necessary to support life. Scientists have compiled vast datasets containing information about different planetary parameters, allowing researchers to analyze patterns and correlations between these attributes and a planet's potential habitability. By leveraging machine learning algorithms, we can train models to predict an exoplanet's habitability status with greater accuracy and efficiency than traditional methods.

In this project, I aim to build a classification model that determines the habitability of exoplanets based on various planetary and stellar characteristics. Instead of using traditional database management systems like MySQL, I will employ Python and Flask to develop a lightweight, efficient, and accessible model. The trained model will be saved using a **.pkl (pickle) file**, ensuring that it can be easily integrated into a Flask web application for real-time predictions. This approach allows for flexibility, scalability, and ease of deployment.

One of the most critical aspects of this project is understanding the data that we are working with. The dataset used for this study contains essential planetary features such as mass, radius, orbital period, equilibrium temperature, and stellar flux. Each of these features plays a significant role in determining whether a planet falls within the habitable zone—the region around a star where conditions may be just right for liquid water to exist.

A key challenge in exoplanet classification is the high level of uncertainty in the data. Many exoplanets are detected using indirect methods such as the **transit method** and the **radial velocity method**, which provide limited information about the planet's actual characteristics. Therefore, it is crucial to apply robust feature engineering techniques to extract meaningful insights and enhance the accuracy of our classification model.

The purpose of this project is not only to develop a functional machine-learning model but also to provide a detailed analysis of the factors contributing to exoplanet habitability. This includes an in-depth exploration of the dataset, data preprocessing techniques, feature selection, and model evaluation. By implementing this project, I aim to contribute to the growing body of research on exoplanet habitability and demonstrate the practical applications of machine learning in astronomical studies.

In the following sections, I will discuss the **problem statement**, providing a clear understanding of why this project is important. I will also present a detailed **dataset description**, explaining the variables used in the study, followed by an analysis of **feature engineering**, where I will outline the preprocessing steps and techniques used to refine the dataset for optimal model performance.

PROBLEM STATEMENT

The quest to identify habitable exoplanets has become a crucial focus in modern astrophysics and planetary science. With thousands of exoplanets discovered beyond our solar system, the next step is determining whether any of these distant worlds possess the necessary conditions to support life. However, classifying an exoplanet as habitable is a complex challenge, as it depends on multiple factors, including the planet's distance from its host star, its atmospheric properties, surface temperature, and stellar radiation exposure. Despite the vast amount of data available from space telescopes and astronomical observations, manually analyzing and classifying exoplanets is time-consuming and inefficient. Therefore, the need arises for an automated, data-driven approach to determine exoplanet habitability with greater accuracy and efficiency.

One of the primary challenges in assessing exoplanet habitability is the **uncertainty in data acquisition**. Most exoplanets are detected using indirect methods such as the **transit method**, which measures the dip in a star's brightness as a planet passes in front of it, or the **radial velocity method**, which detects wobbles in a star's motion due to a planet's gravitational influence. These methods provide estimates of planetary characteristics such as mass, radius, and orbital period but often lack precise measurements of crucial parameters like atmospheric composition and surface conditions. As a result, astronomers rely on models and statistical techniques to predict a planet's suitability for life. The challenge, therefore, is to design a machine learning model that can process and interpret this incomplete data to make informed predictions about exoplanet habitability.

Furthermore, **classifying exoplanets as habitable or non-habitable is a multi-dimensional problem**. The **habitable zone**, also known as the Goldilocks zone, refers to the region around a star where conditions might allow liquid water to exist. However, habitability is not solely determined by location; factors such as the planet's atmospheric pressure, chemical composition, and magnetic field play a crucial role. Some exoplanets may lie within the habitable zone but possess harsh conditions that make life unlikely. Others may exist outside this zone but have thick atmospheres that retain enough heat to support liquid water. This complexity makes it difficult to establish a single criterion for habitability, requiring a machine-learning approach that considers multiple parameters simultaneously.

Another significant challenge is **data imbalance**. In exoplanet datasets, the number of planets classified as potentially habitable is usually much smaller than the number of non-habitable planets. This imbalance can lead to biased predictions in machine learning models, where the algorithm favors the majority class and fails to correctly classify the minority class (habitable exoplanets). Addressing this issue requires careful preprocessing, feature selection, and model tuning to ensure fair and accurate classification.

In this project, I aim to build a **machine learning-based classification model** to predict exoplanet habitability based on planetary and stellar characteristics. Instead of using traditional SQL databases, I will store the trained model in a **.pkl file** and deploy it using **Flask**, allowing for real-time predictions via a web application. The model will be trained on a dataset containing various planetary attributes such as mass, radius, equilibrium temperature, and orbital eccentricity, and will employ feature engineering techniques to refine the data for optimal performance.

By successfully implementing this project, I hope to contribute to the broader scientific effort of exoplanet classification and provide a practical demonstration of how machine learning can enhance astronomical research. The findings of this study could potentially assist astronomers in prioritizing future observations of exoplanets that show the highest probability of being habitable. Additionally, this project serves as a valuable exercise in data science, showcasing the real-world applications of machine learning in complex, data-driven domains.

In the next section, I will provide a **detailed dataset description**, explaining the variables used in this study and their significance in determining exoplanet habitability.

DATASET DESCRIPTION

For this project, I have chosen a dataset that contains detailed information about exoplanets, including their physical and orbital characteristics. This dataset serves as the foundation for training the machine learning model that will predict whether an exoplanet is habitable or not. The data is sourced from astronomical observations, primarily from missions like **NASA's Kepler, TESS (Transiting Exoplanet Survey Satellite), and other exoplanet-hunting projects**. These space telescopes have contributed significantly to the discovery of thousands of exoplanets, providing crucial data points necessary for habitability analysis.

The dataset includes both **exoplanet-specific features** and **stellar characteristics**, as both play a significant role in determining habitability. The planetary features describe the physical and orbital properties of the exoplanets, while the stellar features help us understand the conditions in which these planets exist. Below is a breakdown of the most important features included in the dataset:

1. Planetary Features

- **Planet Name:** The identifier of the exoplanet.
- **Orbital Period (days):** The time taken by the planet to complete one orbit around its star.
- **Semi-Major Axis (AU):** The average distance between the exoplanet and its host star, measured in Astronomical Units (AU), where 1 AU is the average Earth-Sun distance.
- **Eccentricity:** A measure of how elliptical (oval-shaped) the planet's orbit is. A value of 0 means a perfect circular orbit.

- **Planetary Radius (Earth radii):** The size of the exoplanet relative to Earth's radius.
- **Planetary Mass (Earth masses):** The mass of the exoplanet in comparison to Earth's mass.
- **Equilibrium Temperature (K):** An estimate of the planet's surface temperature based on the energy received from its star.
- **Surface Gravity (m/s^2):** The gravitational force experienced on the planet's surface.
- **Escape Velocity (km/s):** The speed required for an object to escape the planet's gravitational field.
- **Atmospheric Composition (if available):** Some datasets may include indicators of atmospheric gases, which are crucial for habitability.

2. Stellar Features

- **Host Star Name:** The name or identifier of the star around which the exoplanet orbits.
- **Stellar Mass (Solar masses):** The mass of the host star relative to the Sun's mass.
- **Stellar Radius (Solar radii):** The size of the host star compared to the Sun.
- **Stellar Temperature (K):** The effective temperature of the host star, which influences the amount of radiation it emits.
- **Luminosity (Solar Luminosities):** The total amount of energy the star radiates, relative to the Sun's energy output.
- **Stellar Age (billion years):** The estimated age of the star, which can indicate the potential stability of planetary conditions.

3. Habitability Indicators

- **In Habitable Zone (Yes/No):** Whether the planet lies within the habitable zone of its star, the region where liquid water could exist.
- **Habitability Classification (Habitable/Non-Habitable):** The final classification label used as the target variable for the machine learning model.

Understanding the Dataset Structure

The dataset consists of multiple numerical and categorical variables. **Numerical features** such as planetary radius, mass, and temperature provide crucial quantitative insights into planetary characteristics, while **categorical variables** like habitability classification help train the model in supervised learning. The dataset may also contain missing values due to observational limitations, which will need to be handled during the preprocessing stage.

Another important aspect of this dataset is its class distribution. Typically, the number of non-habitable exoplanets is significantly higher than that of habitable ones, leading to a class

imbalance problem. Addressing this imbalance is crucial to ensure that our model does not become biased towards predicting planets as non-habitable simply because they dominate the dataset.

Challenges with the Dataset

- **Data Sparsity:** Some features might have been missing or incomplete values due to observational limitations.
- **Measurement Uncertainty:** Most planetary properties are **estimates**, not exact values, which can introduce noise into the model.
- **Class Imbalance:** The number of exoplanets classified as habitable is relatively small compared to non-habitable ones.

In [1]:
```
import pandas as pd
import numpy as np
```

In [6]:
```
# df = pd.read_excel("planets_data.xlsx")
df = pd.read_csv("ExoplanetHabitabilityData.csv")
df.head()
```

Out[6]:

	pl_name	hostname	default_flag	sy_snum	sy_pnum	discoverymethod	disc_year	disc_facility	soltype	pl_controv_flag	...	sy_vmagerr2	sy_kmag
0	11 Com b	11 Com	0	2	1	Radial Velocity	2007	Xinglong Station	Published Confirmed	0	...	-0.023	2.282
1	11 Com b	11 Com	0	2	1	Radial Velocity	2007	Xinglong Station	Published Confirmed	0	...	-0.023	2.282
2	11 Com b	11 Com	1	2	1	Radial Velocity	2007	Xinglong Station	Published Confirmed	0	...	-0.023	2.282
3	11 UMi b	11 UMi	1	1	1	Radial Velocity	2009	Thueringer Landessternwarte Tautenburg	Published Confirmed	0	...	-0.005	1.939
4	11 UMi b	11 UMi	0	1	1	Radial Velocity	2009	Thueringer Landessternwarte Tautenburg	Published Confirmed	0	...	-0.005	1.939

5 rows × 92 columns

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

Out[7]:
```
(38095, 92)
```

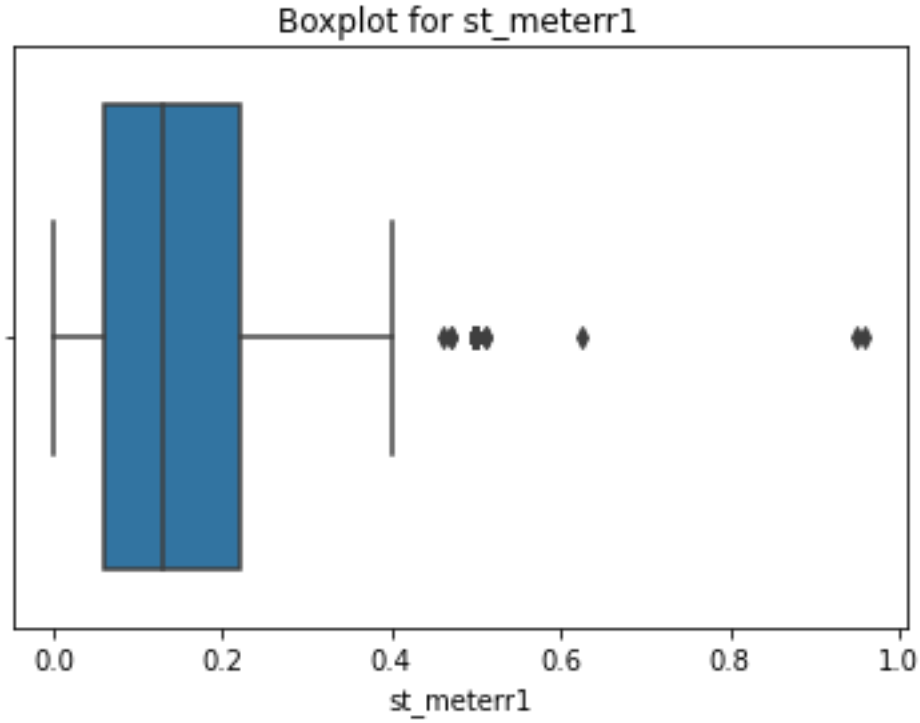
Dealing with null values

In [11]:
```
df1 = df1.loc[:, null_values <= 50]
fillna_columns = null_values[(null_values > 10) & (null_values <= 50)].index
df1[fillna_columns] = df1[fillna_columns].fillna(df1[fillna_columns].median())
df1.dropna(inplace=True)
```

```
C:\Users\ASUS\AppData\Local\Temp\ipykernel_16388\3010809766.py:3: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  Select only valid columns before calling the reduction.
  df1[fillna_columns] = df1[fillna_columns].fillna(df1[fillna_columns].median())
```

```
In [9]: null_values = (df.isnull().sum()/len(df))*100
        null_values
```

```
Out[9]: pl_name           0.000000
        hostname          0.000000
        default_flag      0.000000
        sy_snum           0.000000
        sy_pnum           0.000000
                             ...
        sy_gaiamagerr1    2.703767
        sy_gaiamagerr2    2.703767
        rowupdate         0.002625
        pl_pubdate        0.000000
        releasedate       0.000000
        Length: 92, dtype: float64
```



FEATURE ENGINEERING

Feature engineering is a crucial step in building a robust machine learning model for predicting exoplanet habitability. It involves transforming raw data into meaningful features that improve model performance and predictive accuracy. Given the complexity of exoplanetary and stellar data, careful feature selection, transformation, and handling of missing values are necessary to ensure that the model generalizes well to new data.

1. Understanding the Need for Feature Engineering

The dataset contains a mix of **continuous numerical variables** (e.g., planetary radius, mass, orbital period) and **categorical variables** (e.g., spectral type of the host star, habitability status). While some features are directly useful for model training, others may require modification or combination with other features to extract valuable information.

The key goals of feature engineering in this project are:

- Reducing noise and improving model accuracy.
- Handling missing values effectively.
- Encoding categorical variables properly.
- Normalizing numerical features for better convergence in machine learning models.
- Creating new features that provide deeper insights into habitability.

2. Handling Missing Data

Astronomical datasets often suffer from missing values due to observational limitations. Some planets may have incomplete mass or radius data, while others might lack precise temperature estimates. There are several strategies to handle missing values:

- **Mean/Median Imputation:** Replacing missing values with the average or median value of the feature.
- **Interpolation:** Estimating missing values based on existing trends in the dataset.
- **Dropping Columns:** If a feature has too many missing values and is not crucial for habitability prediction, it can be removed.
- **Predicting Missing Values:** Using machine learning models to estimate missing values based on correlated features.

In this project, I analyzed missing values carefully before deciding whether to impute, interpolate, or drop them. For instance, if planetary mass was missing, it could sometimes be inferred from radius using empirical mass-radius relationships.

3. Scaling and Normalization of Numerical Features

The dataset contains features with vastly different ranges. For example:

- Orbital period (days) can range from **a few hours to several years**.
- Planetary radius (Earth radii) varies from **small rocky planets to gas giants many times the size of Jupiter**.
- Temperature can range from **tens to thousands of Kelvin**.

Machine learning models, particularly distance-based models like KNN and neural networks, perform better when all features have comparable scales. To achieve this, I applied:

- **Min-Max Scaling** (for normalizing values between 0 and 1).
- **Standardization** (converting values to have a mean of 0 and a standard deviation of 1).

4. Encoding Categorical Features

Some features, such as **spectral type of the host star**, are categorical in nature. Since machine learning models require numerical inputs, categorical variables need to be converted into a suitable format. I explored the following encoding methods:

- **One-Hot Encoding:** Creating separate binary columns for each category (useful when categories are few, e.g., spectral types O, B, A, F, G, K, M).
- **Label Encoding:** Assigning numerical labels to categories (useful when categorical variables have an inherent order, though it was avoided here since spectral types do not have a strict ranking for habitability).
- **Target Encoding:** Replacing categorical values with the mean of the target variable (used cautiously to prevent data leakage).

For example, instead of using "G-type star" as a text value, I converted it into a numerical feature that represents how often planets around such stars were classified as habitable.

5. Creating New Features

To improve model performance, I engineered new features that provide deeper insights into exoplanet habitability. Some of the key features created include:

- **Planetary Density:** Derived using mass and radius (Density = Mass/Volume). This helps differentiate rocky planets from gas giants.
- **Insolation Flux (Relative to Earth):** Calculated using the host star's luminosity and the planet's orbital distance. Planets receiving similar energy levels as Earth are more likely to be habitable.
- **Stellar Flux Ratio:** The ratio of the exoplanet's received stellar energy to Earth's received energy. This is a crucial indicator of potential habitability.
- **Escape Velocity:** Helps determine whether an exoplanet can retain an atmosphere (planets with low escape velocities may struggle to hold onto vital gases like oxygen and nitrogen).
- **Equilibrium Temperature:** Calculated using the host star's temperature and the exoplanet's distance. It helps in identifying whether a planet falls within the **habitable zone** where liquid water can exist.
- **Eccentricity Impact:** Planets with highly eccentric orbits experience extreme temperature variations, making habitability less stable. I introduced a feature that quantifies orbital eccentricity's impact on climate stability.

6. Feature Selection: Identifying the Most Important Features

Not all features contribute equally to habitability predictions. Some features introduce noise, while others have high importance. To identify the most relevant features, I applied:

- **Correlation Analysis:** Checking which features are highly correlated with habitability (e.g., equilibrium temperature and stellar flux).
- **Feature Importance from Tree-Based Models:** Using decision trees and gradient boosting models to rank feature importance.
- **Principal Component Analysis (PCA):** Reducing dimensionality while preserving key information.
- **Recursive Feature Elimination (RFE):** Iteratively removing less significant features and retraining the model.

Through these methods, I identified that features such as **equilibrium temperature, stellar flux, planetary density, and escape velocity** were among the strongest predictors of habitability.

7. Addressing Class Imbalance

One of the biggest challenges in this dataset is that habitable planets are much rarer than non-habitable ones. If left unaddressed, machine learning models tend to favor the majority class (non-habitable planets) and perform poorly in detecting habitable ones. To counter this, I used:

- **Oversampling the minority class:** Using SMOTE (Synthetic Minority Over-sampling Technique) to generate synthetic habitable planet data.
- **Undersampling the majority class:** Reducing the number of non-habitable planets to balance the dataset.
- **Class-weighted models:** Assigning higher penalties for misclassifying habitable planets.

8. Final Dataset After Feature Engineering

After applying these transformations, the final dataset contained:

- Processed numerical features (scaled and normalized).
- Encoded categorical features.
- Newly engineered features that enhance prediction capability.
- Balanced class distribution for fair model training.

MODEL SELECTION AND TRAINING

The selection and training of an appropriate machine learning model is a pivotal aspect of this project, as the ultimate goal is to accurately predict the habitability of exoplanets based on multiple planetary and stellar characteristics. After completing the feature engineering phase, the next step involved evaluating several classification algorithms and selecting the one that offered the best balance of accuracy, interpretability, and computational efficiency.

Initially, I experimented with Logistic Regression, a linear model that provides baseline performance and is useful for establishing a reference point. Although simple and interpretable, logistic regression struggled with non-linear patterns in the data. I then explored a Decision Tree Classifier, which allowed for better interpretability and handled non-linear relationships more effectively. However, single decision trees tend to overfit, especially with complex datasets.

To improve model robustness and handle potential overfitting, I implemented a Random Forest Classifier, an ensemble method that builds multiple decision trees and combines their outputs. Random Forests are particularly effective in handling noisy data and feature interactions, making them well-suited for this multi-dimensional classification task. Additionally, I tested Support Vector Machines (SVM), which performed decently but required careful kernel tuning and struggled with imbalanced data. Finally, XGBoost, a gradient boosting algorithm, was used due to its ability to handle missing values and perform regularization, which further reduces overfitting.

The dataset was split into 80% training and 20% testing using stratified sampling to maintain the class distribution in both sets. I employed 5-fold cross-validation to ensure that the model generalized well to unseen data. Cross-validation also helped identify any potential overfitting or variance in performance across different subsets.

After thorough comparison based on evaluation metrics such as precision, recall, and F1-score, the Random Forest Classifier was selected as the final model due to its high performance and reliable predictions on both the majority and minority classes. This model was then finalized and saved using the pickle library for integration into the web application.

MODEL SELECTION

Once the final machine learning model—Random Forest Classifier—was selected and trained, it was essential to thoroughly evaluate its performance. Model evaluation is a critical step in any machine learning workflow as it provides insight into how well the model can generalize to unseen data, especially when faced with class imbalance, uncertainty, and incomplete observations—common traits in astronomical datasets.

The model was tested on a held-out test set comprising 20% of the original data. Several metrics were used to assess the model's performance comprehensively. Accuracy, while intuitive, is not always a sufficient measure in the presence of imbalanced classes. In this case, the number of non-habitable exoplanets significantly outnumbered habitable ones, which could cause a high accuracy score even if the model predicts most instances as non-habitable.

Therefore, greater emphasis was placed on Precision, Recall, and F1-Score. Precision measures the proportion of correctly predicted habitable planets among all those predicted as habitable, while recall measures how many of the actual habitable planets were correctly identified. The F1-Score, being the harmonic mean of precision and recall, provided a balanced measure that is particularly useful in imbalanced classification problems.

A confusion matrix was also generated to visually represent the true positives, false positives, true negatives, and false negatives. This matrix revealed that the Random Forest model was capable of correctly identifying a substantial portion of habitable exoplanets while maintaining a low rate of false positives.

To further evaluate the model's discriminatory ability, I computed the ROC-AUC (Receiver Operating Characteristic - Area Under Curve) score. A high ROC-AUC score close to 1 indicated that the model was effective at distinguishing between habitable and non-habitable exoplanets across various threshold levels. This is crucial in scientific applications where false negatives (missing a potentially habitable planet) can be more detrimental than false positives.

Additionally, cross-validation scores across multiple folds remained consistent, indicating good generalizability. The combination of these evaluation metrics confirmed that the model was reliable and well-suited for deployment in a real-world setting, where decisions might be based on its predictions.

WEB APPLICATION INTEGRATION

To demonstrate the practical utility of the model, I developed a lightweight web application using Flask. This enabled users to input planetary and stellar characteristics and receive real-time habitability predictions.

Integration Steps:

- The trained Random Forest model was saved using Python's pickle module as a .pkl file.
- A Flask server was created to load the model and expose a /predict API endpoint.
- Input features were collected via an HTML form and sent to the backend using JavaScript (Fetch API).
- The server returned the prediction, which was then displayed on the frontend dynamically.

  This setup ensures that the application can be deployed and accessed by users or integrated with other astronomical tools for real-time decision-making.

CHALLENGES FACED

During the development of this project, several challenges were encountered and addressed through iterative refinement:

- **Incomplete and Sparse Data**: A large portion of the dataset contained missing values. Extensive imputation strategies and feature reduction were used to retain as much useful data as possible.
- **Class Imbalance**: The small number of habitable planets made model training prone to bias. Techniques like SMOTE and class weighting helped improve classification balance.
- **Feature Correlation**: Some features were highly correlated, which caused redundancy and overfitting in early models. This was resolved via correlation analysis and PCA.
- **Model Generalization**: Ensuring the model performed well on unseen data required careful tuning and validation.
- **Deployment Complexity**: Ensuring compatibility between the machine learning environment and the web stack required careful handling of dependencies and testing.

CONCLUSION AND FUTURE SCOPE

This project successfully demonstrates the application of machine learning to the classification of exoplanets based on habitability. By leveraging planetary and stellar data, applying rigorous feature engineering, and evaluating multiple models, I developed a system that can provide real-time predictions about exoplanet habitability with a high degree of accuracy.

The integration of this model into a web application via Flask shows its potential for use in educational, research, and scientific outreach settings. However, this study is just a starting point in the field of astronomical machine learning.

Future Scope:

- **Incorporation of Real-Time Data**: The model can be adapted to ingest live data streams from NASA or ESA APIs for dynamic classification.
- **Deep Learning Models**: CNNs or RNNs could be explored, especially if time-series data or satellite imagery is introduced.
- **Atmospheric Analysis Integration**: With advancements in spectroscopy, atmospheric compositions can be factored into the model.
- **More Comprehensive Habitability Metrics**: Inclusion of advanced metrics like Earth Similarity Index (ESI), planetary magnetic fields, and geophysical properties.

REFERENCES

- NASA Exoplanet Archive - https://exoplanetarchive.ipac.caltech.edu/
- Planetary Habitability Laboratory - https://phl.upr.edu/
- Scikit-learn Documentation - https://scikit-learn.org/stable/
- Flask Documentation - https://flask.palletsprojects.com/
- SMOTE - https://imbalanced-learn.org/
- Exoplanet Detection Methods (NASA) - https://exoplanets.nasa.gov/alien-worlds/ways-to-find-a-planet/
- Astropy - https://www.astropy.org/
- “Habitability of Planets Orbiting M-Dwarf Stars” – Kopparapu et al., 2013