Name : Swaraj Mohite
Course : Diploma in Computer Engineering (Last year)
Data Science Experience : Yes
Course work :
1. Learning through Youtube,
2. Completed Data Mining Subject last year.
3. Know Data Process(cleaning and all) and some ML algorithms.
4. Worked on beginner level projects.

Milestone 1 - Completed

MileStone 2 -

1. Performed PCA and t-SNE for dimensionality reduction means visualized.
2. Handled class imbalance using SMOTE, Borderline-SMOTE, SMOTE-Tomek, ADASYN, and Random Undersampling.
3. Trained multiple ML models: Logistic Regression, KNN, Naive Bayes, SVM, Random Forest, XGBoost.
4. Applied class-weighted and imbalance aware modeling strategies.
5. Top Models -
   a) XGBoost : Best macro F1 + minority recall

XGBoost - scale_pos_weight

[[1198 1 0]

[0 6 0]

[1 0 9]]

precision recall f1-score support

0 1.00 1.00 1.00 1199

1 0.86 1.00 0.92 6

2 1.00 0.90 0.95 10

accuracy 1.00 1215

macro avg 0.95 0.97 0.96 1215

weighted avg 1.00 1.00 1.00 1215

b) Random Forest：Balanced & stable performance

Random Forest

[[1195 1 3]

[0 5 1]

[6 1 3]]

precision recall f1-score support

0 1.00 1.00 1.00 1199

1 0.71 0.83 0.77 6

2 0.43 0.30 0.35 10

accuracy 0.99 1215

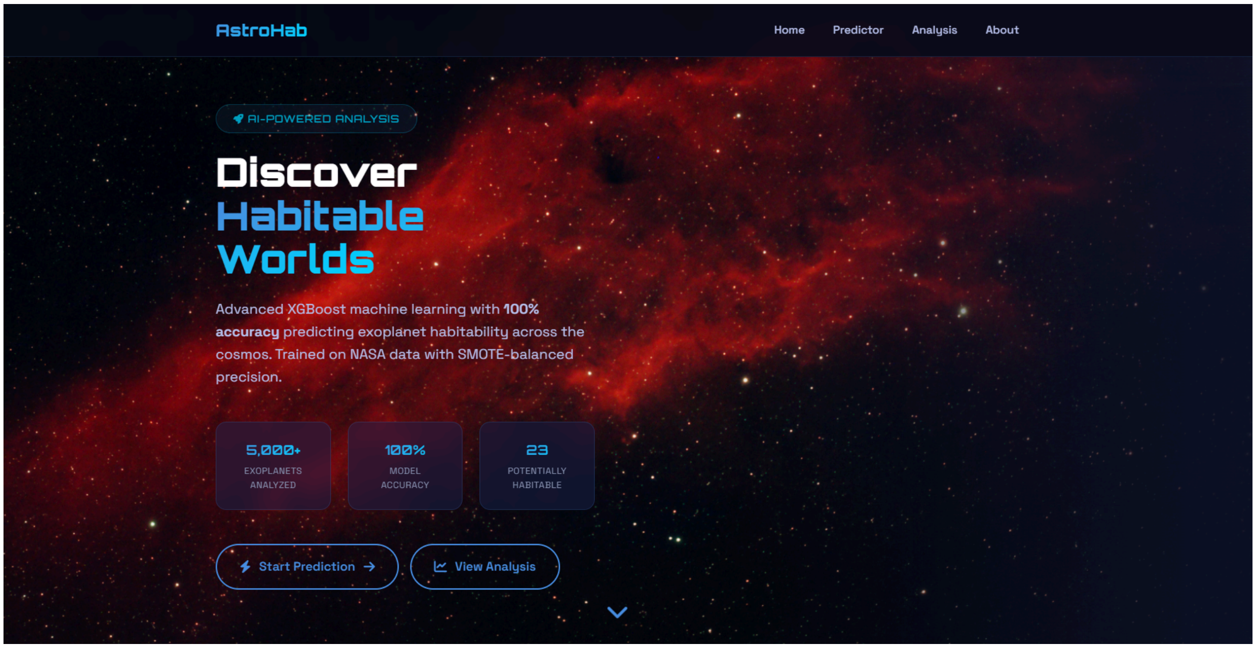
macro avg 0.71 0.71 0.71 1215

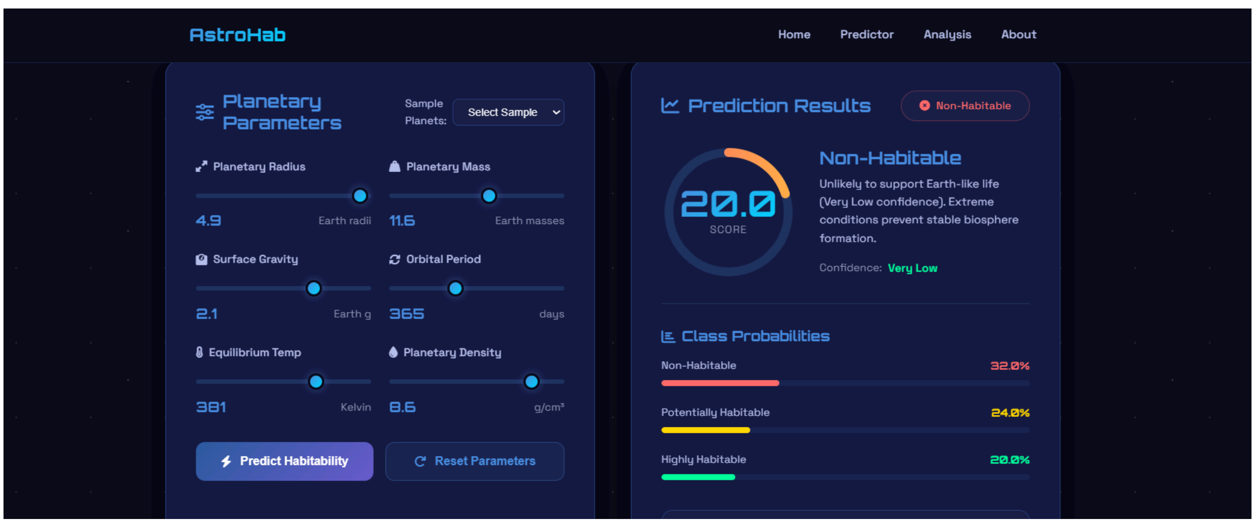
weighted avg 0.99 0.99 0.99 1215

C) RF_SMOTETomek

Also, another is Balanced Random Forest

```
[[1143   14   42]
 [   0    6    0]
 [   0    1    9]]
```


AstroHab
Home
Predictor
Analysis
About
AI-POWERED ANALYSIS
Discover
Habitable
Worlds
Advanced XGBoost machine learning with 100% accuracy predicting exoplanet habitability across the cosmos. Trained on NASA data with SMOTE-balanced precision.
5,000+
EXOPLANETS ANALYZED
100%
MODEL ACCURACY
23
POTENTIALLY HABITABLE
Start Prediction
View Analysis


AstroHab
Home
Predictor
Analysis
About
Planetary Parameters
Sample Planets:
Select Sample
Planetary Radius
4.9
Earth radii
Planetary Mass
11.6
Earth masses
Surface Gravity
2.1
Earth g
Orbital Period
365
days
Equilibrium Temp
381
Kelvin
Planetary Density
8.6
g/cm³
Predict Habitability
Reset Parameters
Prediction Results
Non-Habitable
20.0
SCORE
Non-Habitable
Unlikely to support Earth-like life (Very Low confidence). Extreme conditions prevent stable biosphere formation.
Confidence: Very Low
Class Probabilities
Non-Habitable
32.0%
Potentially Habitable
24.0%
Highly Habitable
20.0%


PLANETARY COMPARISON
Comparison with Earth
Earth
1.0
RADIUS
1.0
MASS
1.0
GRAVITY
VS
Super-Earth
4.9
RADIUS
11.6
MASS
2.1
GRAVITY

Module 1: Data Collection and Management

Week 1 Day 1 : 01/12/2025

1. **DataSet Link 1 - (Kaggle)**
   https://www.kaggle.com/datasets/gauravkumar2525/kepler-exoplanet-dataset

   - Observations :
     a) Have total 12 columns and 9564 rows,
     b) Few features are missing like distance and mass,
     c) Unique and no duplicate values.
     d) There are no null values but need to validate data.
     e) Have both planetary and host start properties but many features are irrelevant and not useful
     f) dtypes: float64(9), int64(2), object(1)
   - Useful or not : Useful if added more features or integrate with other. And if do Feature Engineering.

   - *Question* - Sir, Should we combine multiple datasets to get more data and features, or stick to one dataset and enhance it ? Which is better ?

2. **Dataset Link 2 - (NASA Exoplanet)**
   https://exoplanetarchive.ipac.caltech.edu/cgi-bin/TblView/nph-tblView?app=ExoTbls&config=PS

   - Observations :
     a) 39119 rows and 289 columns
     b) Latest and trusted dataset (officially from NASA)
     c) Have all features but too much missing values . Need to clean and process the data.
     d) Many features are irrelevant here also. Proper feature selection important.
     e) dtypes: float64(235), int64(26), object(28)
   - Useful if cleaned and processed properly.

   - *Question* - Sir, is it better to use a big dataset even if it has many missing/wrong values, or a smaller dataset that is fully cleaned and accurate?

3. **Dataset link 3 - (Kaggle)**

https://www.kaggle.com/datasets/chandrimad31/phl-exoplanet-catalog

- Observations :
  a) 112 columns with 4048 unique record
  b) Have all features
  c) Have some null values and inconsistent and noisy data , which needs to clean and preprocess.
  d) Have 'habitable' column but unbalanced.
  e) dtypes: float64(94), int64(4), object(14)
- Useful but required too much data cleaning and data preprocessing.

Comments -

- Some Column names and Unit names / values are different in different datasets .
- 3rd and 2nd Dataset is good.
- Need of proper feature engineering and transformation.

Week 1 Day 2 : No Class

Week 1 Day 3 : 03/12/2025

- Features required specified in document : Planet radius, mass, density, surface temperature, orbital period, distance from star, Host star type, luminosity, temperature, metallicity
- Additional features may useful : Planet gravity, orbit shape, Earth-likeness score
- Planet gravity helps determine if a planet can hold an atmosphere and support life. Orbit shape affects temperature stability and climate over time. Earth-likeness score measures how similar a planet is to Earth, indicating potential habitability.

Question - Sir, since the some datasets don't have a 'habitable' label, then ? should we create it ourselves during feature engineering ?

Is this correct ?

Input features like planet radius, mass, orbital period, distance from star, star temperature, luminosity, metallicity, and insolation flux etc.
And the output target is habitability class (Habitable or Not), Habitability Score..

- Features description :

    - [ ] Planet radius: Bigger or smaller size affects how much air the planet can hold.
    - [ ] Planet mass: The weight of the planet helps keep an atmosphere for life.
    - [ ] Density: Tells if the planet is rocky like Earth or made of gas like Jupiter.
    - [ ] Surface temperature: Shows if the planet is warm enough for water to be liquid.
    - [ ] Orbital period: How long the planet takes to go around its star, affecting seasons/temperature.
    - [ ] Distance from star: Must be just right to stay not too hot or cold for water.
    - [ ] Host star type: Different stars give different amounts of light and heat.
    - [ ] Luminosity: How bright the star is, affecting warmth on the planet.
    - [ ] Star temperature: Hotter stars have larger zones where life might exist.
    - [ ] Metallicity: More metals in the star means better chance of planet formation and life.

This dataset is best till now -
https://www.kaggle.com/datasets/chandrimad31/phl-exoplanet-catalog
Having habitable feature (P_HABITABLE column),
but

Having distrinution like:

0 : 3993
1 : 21
2 : 34

Means, **highly imbalanced**. If we train using same dataset then there are chances of biased towards 0 (No habitability).

Week 1 Day 4 : 04/12/2025

Things we can do -

- For imbalanced dataset :

    - Resampling : adding or removing where the data is imbalanced

        - Oversampling : Duplicating values which are less (in our case 1s and 2s). Comment - Can cause Overfitting because of Duplicates.
        - Undersampling : Removing values which are more to balance (0s).

Comment - **Can lose valuable information.**

Comment : Just duplicating values or reducing values, helps temporarily. But may still cause baise.

- SMOTE (Synthetic Minority Oversampling) : using *ibmlearn* library,
  - It creates new samples using nearest neighbour.
  - **It may create unrealistic samples.**

- **Using Random Forest or XGboost. (Class Weighting) .**
  - It internally handles imbalanced dataset.
  - By adjusting class importance (need to research more).
  - Means by giving more importance to 1s and 2s in our habitable column.

- Create y column based on features we have like temperature, radius, gravity (which affects habitability).
  - I think it is not ideal method.
  - Can lead to wrong assumption. (many problems)

Since, **Habitable planets are actually very few in nature**, Can we go with Class weighting or SMOTE technique?

I will do more research on it. Working and all.
In Class weighting XGBoost

Week 1 Day 5 : 05/12/2025

- Descriptive Strategies : to summarize, understand, and describe data.
- Discussion about descriptive strategies and concepts like percentage, percentile, distribution etc.
- Revised topics
- Task is given to apply descriptive strategies to the dataset (which lastly discussed).

Required libraries : numpy, pandas, and for visualization (matplotlib nd seaborn).

Analysis:

- Dataset have total 7 columns which are of no use (all having null values, means total 4048 null values which is equal to length)..

Module 2: Data Cleaning and Feature Engineering

Week 2 Day 1 : 08/12/2025

Analysis :

1. Paper 1 - https://ijrpr.com/uploads/V3ISSUE2/ijrpr2746-detection-of-exoplanets-using-machine-learning.pdf

    - Observations :
        - Used SMOTE oversampling, which heavily improves model accuracy.
        - Raw light curve (used to detect exoplanet) are too large, so need to transform.
            - PCA (Principal Component analysis)
            - FFT
        - SVM and CNN works well, with proper feature eng.

2. Paper 2 - https://ijrpr.com/uploads/V3ISSUE2/ijrpr2746-detection-of-exoplanets-using-machine-learning.pdf

    - Observations :
        - Used TSFresh for feature engineering.
        - LightGBM model for that extracted features.
        - Ways used to handle imbalanced data :
            - Threshold tuning (lowers threshold) [it increases recall]

3. Article 3 - https://www.kdnuggets.com/2020/01/exoplanet-hunting-machine-learning.html

- Observations :
    - Data Preprocessing by : Normalization, Gaussian Smoothing (to reduce noise), feature scaling (StandardScaler), and PCA
    - Handling Imbalanced : Using SMOTE , by balancing dataset so both have 5050 samples
    - Models used : SVM, Random Forest, ANN

4. Kaggle Notebook - https://www.kaggle.com/code/nickoreese/exoplanet-habitability-prediction

    - Observations :
        - Used PHL dataset.
        - For Imbalanced used : SMOTE and ENN (removes noisy points).
        - Features Selection : Random Forest, AdaBoost, ExtraTrees
        - Models : Decision Tree(fastest), KNN, GB(High accuracy)

- **Overall Observations :**
    - Most of them started with cleaning light curve (main focus is light curve) and then feature engineering part. (whoever used kepler dataset).
    - For Imbalanced they used : SMOTE (mostly), threshold tuning etc.
    - As Dimensionality is huge, they reduced using PCA, TSFresh (open source) or feature selection.
    - Model used : Classical ML Models (SVM, LightGBM, Random Forest, GB) and DL Models (CNN, ANN etc.).

Comments :
- SVM works well on PCA compressed data.
- KNN works with high accuracy with PHL dataset.
- Using SMOTEENN for imbalanced dataset (combining both oversampling and undersampling using SMOTE and Nearest Neighbour).

Week 2 Day 2 : 09/12/2025

Clean data : remove missing columns, impute numeric values, fix outliers.
Transform features → normalization, encoding, smoothing light curves.
Reduce dimensions → PCA or TSFresh to extract meaningful patterns.
Fix imbalance → SMOTE or threshold tuning.

Train models → SVM, LightGBM, RF, ANN.
Evaluate properly → PR curves, recall, cross-validation, visual checks.

Week 2 Day 3 : 10/12/2025

- Session was about Reading and Git/Github Collaboration.
- Forked, Cloned repo and made first commit.
- Working on Dataset (PHL).
- Exploring the Dataset as told in session.
- Loaded the **PHL Exoplanet Catalog (2019)** dataset and examined:
    - Shape, column names, data types, and memory usage
    - Descriptive statistics for numerical and categorical features

Week 2 Day 4 : 11/12/2025

- Cleaned dataset and handled missing values
- Calculated **missing value counts and percentages per feature**
- Dropped features with **>75% missing values** to reduce noise and instability
- Visualized missingness using **heatmaps**
- Identified numerical and categorical columns separately.

    - Reduced dataset dimensionality by removing unreliable columns
    - Obtained a cleaner base dataframe (new_df) for further processing

Week 2 Day 5 : 12/12/2025

- Completed EDA and some preprocessing work on dataset.
- Identified outliers using **IQR (Interquartile Range)** for numerical features
- Quantified outlier counts per feature
- Checked **skewness** of numerical variables
- Imputed missing values:
    - Numerical → **Median** (robust to outliers)
    - Categorical → **Mode**
- Cleaned categorical values (trimmed spaces, standardized casing)
- Applied **one-hot encoding** to categorical features
- Separated target variable (y) from feature matrix (X)
- Standardized all features using **StandardScaler**

Module 3: Machine Learning Dataset Preparation

Week 3 Day 1 : 15/12/2025

- Applied **PCA (2D)** on scaled features for visualizing class separation.
- Applied **t-SNE (2D)** for a nonlinear projection of the feature space.
- Visualized the 2D projections using scatterplots colored by P_HABITABLE.

    - ❖ PCA shows variance explained mainly by first 2 components; some overlap between classes.
    - ❖ t-SNE provides a clearer separation of minority and majority classes.
    - ❖ These visualizations help understand class distribution and potential model performance.

Week 3 Day 2 : 16/12/2025

- Split dataset into **training (70%)** and **testing (30%)** with stratification.
- Checked class distributions in train and test sets.
- Applied **resampling techniques** on training data to handle imbalance:
    - **SMOTE**
    - **Borderline-SMOTE**
    - **SMOTE + Tomek Links**
    - **ADASYN**
    - **Random Undersampling**
- Saved all resampled training datasets

- ❖ Original training data is imbalanced; minority class underrepresented.
- ❖ SMOTE and ADASYN oversample minority classes; Random Undersampling reduces majority class.
  Borderline-SMOTE focuses on samples near decision boundaries for better model learning.
- ❖ SMOTE + Tomek Links combines oversampling and cleaning to remove noisy points.

Week 3 Day 3 : 17/12/2025

Tasks Done:

- Trained **SVM** with class_weight='balanced' on original training data.
- Trained **XGBoost** with scale_pos_weight to adjust for class imbalance.
- Predicted test set and evaluated using **confusion matrix** and **classification report**.

Observations:

- Class weighting improves minority class prediction without modifying data.
- XGBoost performed better for the majority class due to gradient boosting handling imbalance.
- Both models provide a baseline before using ensemble resampling techniques.

Week 3 Day 4 : 18/12/2025

Tasks Done:

- Trained **Balanced Random Forest** classifier on original training data.
- Predicted test set and evaluated performance.
- Saved predictions along with actual values
- Also saved training dataset info post-sampling: train_data_post_sampling.csv.

Observations:

- BRF inherently balances classes during training.
- Improved performance on minority classes without manually resampling.
- Predicted labels are ready for further analysis or comparison with other models.

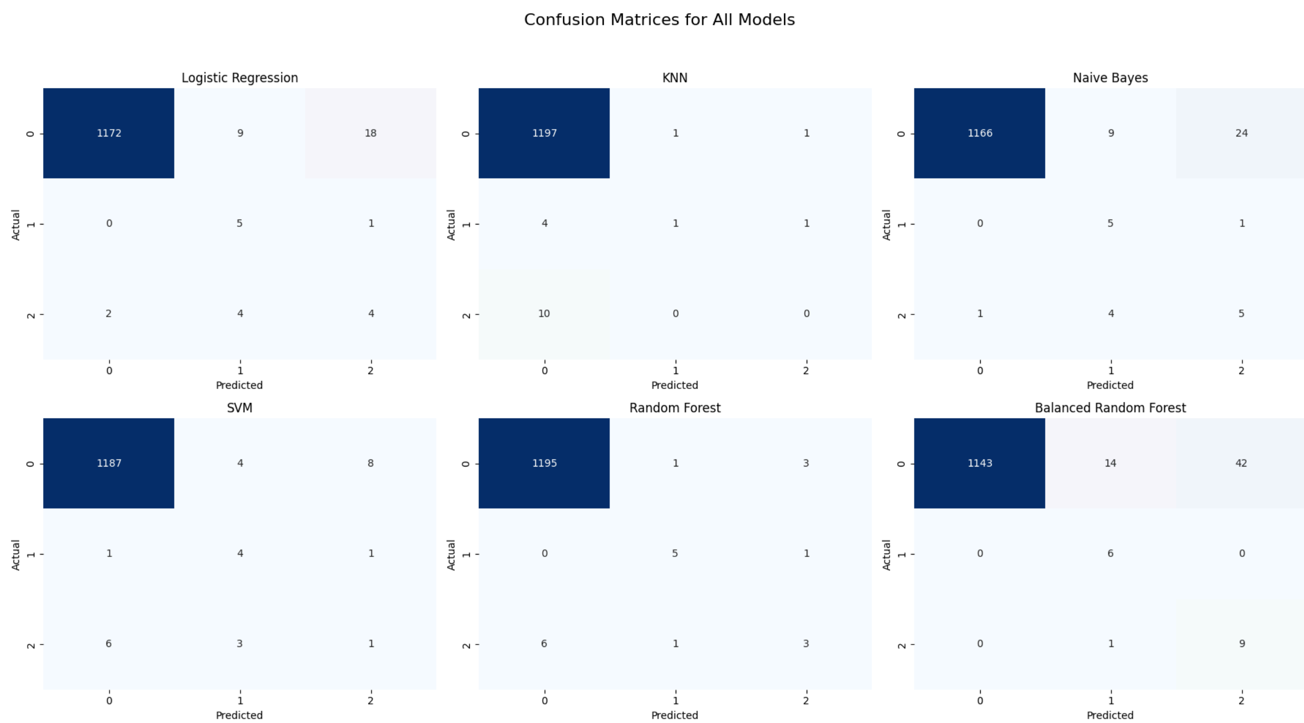
Week 3 Day 5 : 19/12/2025

- Compared performance of all sampling techniques (SMOTE, Borderline-SMOTE, SMOTE + Tomek, ADASYN, Random Undersampling) using training and test sets.
- Pushed Notebook to git.

Week 3 Checkpoints :

- Cleaned data (missing values, outliers, skewness)
- Feature Eng (categorical cleaning, one-hot encoding)
- Normalized numeric features.
- Defined target variable and performed train–test split before sampling
- Handled class imbalance (SMOTE, ADASYN, SMOTE-Tomek, undersampling) and saved it
- Applied PCA and t-SNE for feature space visualization
- Trained initial models (SVM, XGBoost, Balanced Random Forest)

I think 80 - 85 % done.



Module 4: AI Model for Habitability Prediction

Week 4 Day 1 : 22/12/2025

- Reading and Coding
- Studied how different models react to:
    - Class imbalance
    - Oversampling side effects
    - High-dimensional feature space
- Also reviewed evaluation metrics beyond accuracy, especially **macro F1-score, minority class recall, and ROC-AUC**, which are more reliable for habitability prediction where positive samples are extremely rare.

Week 4 Day 2 : 23/12/2025

- Tried to find more good model
- Observed that after aggressive oversampling (SMOTE, ADASYN), the dataset size increased significantly. This raised concerns about **model overfitting.**
- Preferred **class-weighted learning** over excessive data duplication

Week 4 Day 3 : 24/12/2025

- Tried to build ML Pipeline.
- Studied and experimented with building an end-to-end **Machine Learning Pipeline**
- Understood the importance of pipelines for:
    - Preventing data leakage
    - Ensuring consistent preprocessing across training and testing
    - Making experiments reproducible and maintainable

Week 4 Day 4 : 25/12/2025

- Holiday

Week 4 Day 5 : 26/12/2025

- Did remaining part.
- Completed the remaining model training and testing tasks.
- Identified inconsistencies in some results due to preprocessing order and sampling effects. To address this, started working on a **new Jupyter Notebook** dedicated

Module 5: Flask Backend API

Week 5 Day 1 : 29/12/2025

- Set up the Flask application for backend development.
- Learned basic concepts of Flask such as routing, request handling, and application structure.
- Designed a clean folder structure for the backend.
- Created initial REST API endpoints and tested them using sample requests.

Observation: Flask is lightweight and easy to use for building backend APIs quickly.

Week 5 Day 2 : 30/12/2025

- Integrated the trained XGBoost machine learning model with the Flask backend.
- Loaded the serialized model file for prediction purposes.
- Added error handling to manage invalid inputs and system errors.
- Tested the prediction API to ensure correct output generation.

Week 5 Day 3 : 31/12/2025

- Holiday

Week 5 Day 4 :01/01/2026

- Holiday

Week 5 Day 5 :02/01/2026

- Completed backend development usign flask.
- Tested all API endpoints for correct responses and stability.
- Made minor improvements to enhance performance and response consistency.

Key Concepts Learned During This Week

- **REST API:** Enables structured communication between client and server using stateless HTTP requests, commonly returning JSON responses.

- **Synchronous vs Asynchronous Requests:**

    - Synchronous requests block execution until a response is received.

- Asynchronous requests allow concurrent task execution, improving system efficiency.

- **Multithreading vs Concurrency:**
    - Multithreading allows parallel execution based on CPU cores.
    - Concurrency enables handling many tasks efficiently through scheduling.

- **Flask vs FastAPI:**
    - Flask is simple and synchronous by default.
    - FastAPI supports asynchronous execution, automatic data validation, and higher throughput.

- **Throughput:** Measures how many requests a system can handle per unit time.

- **Inference Speed:** Measures how quickly a trained model generates predictions.

- **SHAP Values:** Provide explainability by quantifying feature contributions to model predictions.

Module 6: Frontend UI Development

Week 6 Day 1 :05/01/2026

- Designed the basic HTML page structure for the frontend.
- Created user input forms to collect required data for prediction.
- Added essential form fields with labels and placeholders for better clarity.
- Implemented basic CSS styling to improve layout and readability.

Week 6 Day 2 :06/01/2026

- Improved the overall user interface using enhanced HTML and CSS.
- Added multiple sections to organize the frontend content properly.

- Improved alignment, spacing, and visual consistency of the webpage.
- Good UI design improves usability and enhances user engagement.

Week 6 Day 3 :07/01/2026

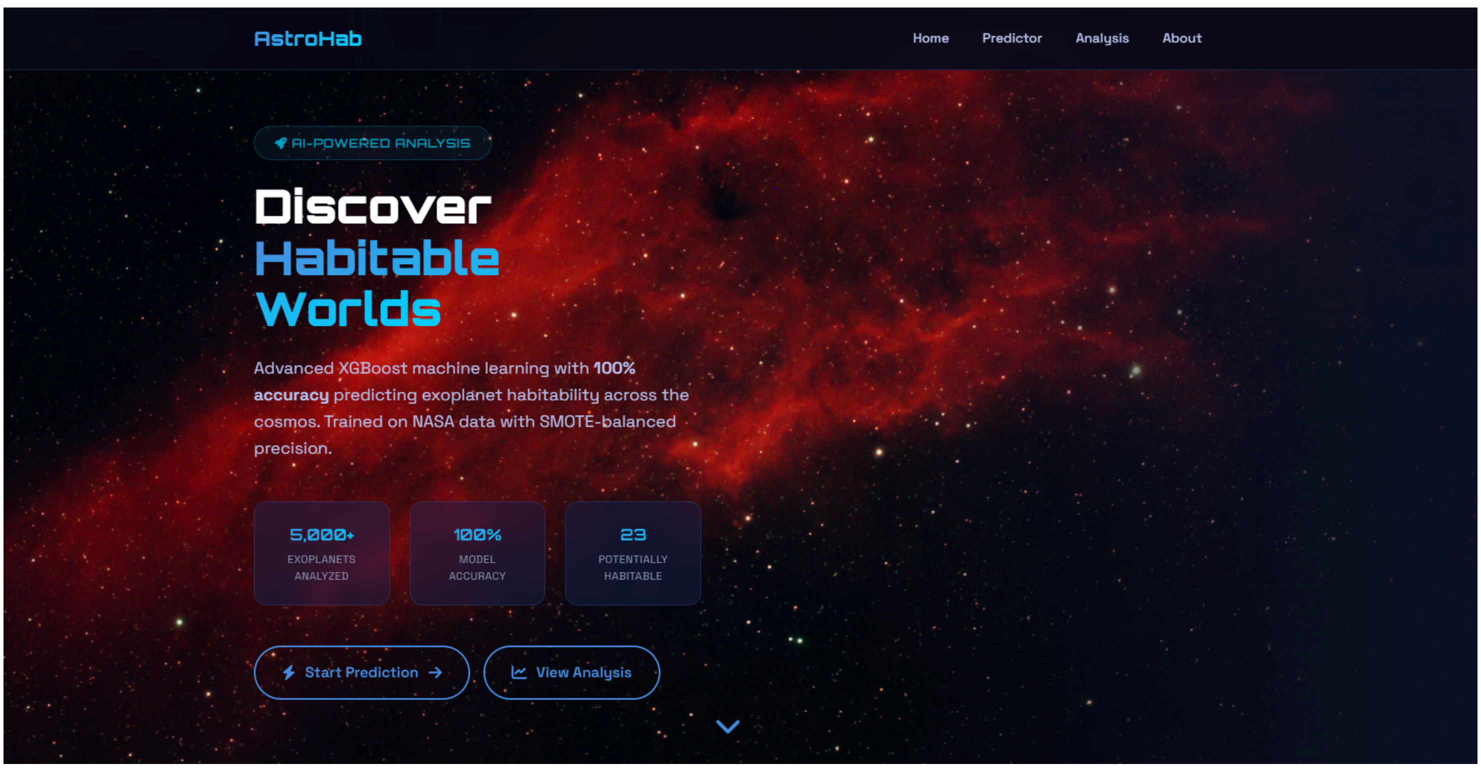
- Leave

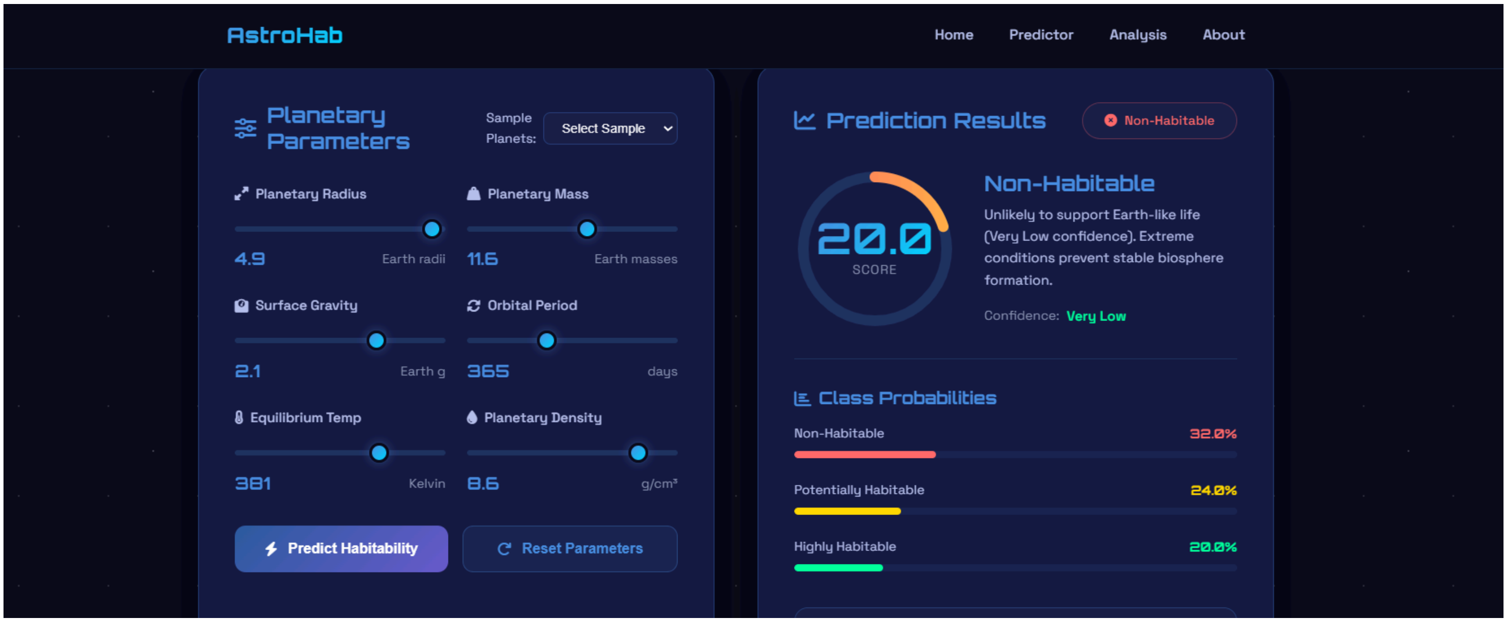
Week 6 Day 4 :08/01/2026

- Implemented JavaScript logic for frontend functionality.
- Connected frontend form inputs with backend APIs using scripts.
- Handled form submission and response display dynamically.
- Completed frontend development and verified end-to-end workflow.
- JavaScript plays a key role in enabling interaction between frontend and backend.

Week 6 Day 5 :09/01/2026

- Leave





Module 7: Visualization & Dashboard

Week 7 Day 1 :12/01/2026

- Continued working on data visualization scripts.
- Reviewed existing plots and improved visualization logic for better clarity.
- Focused on understanding how different plots represent model performance.
- Visualizations help in quickly analyzing model behavior and performance differences.

Week 7 Day 2 :13/01/2026

- Generated multiple plots using **Matplotlib** and **Seaborn** libraries.
- Created confusion matrices for different machine learning models:
    - Balanced Random Forest confusion matrix
    - SVM confusion matrix
    - XGBoost confusion matrix
    - XGBoost with SMOTE confusion matrix

- Generated dimensionality reduction visualizations:
    - PCA 2D plot with SMOTE
    - t-SNE 2D plot with SMOTE

Week 7 Day 3 :14/01/2026

- Attempted to display generated plots on the frontend of the website.
- Explored methods to integrate visual outputs with the web interface.

Week 7 Day 4 :15/01/2026

- Learned about Power BI. and Visualization in it.
- Power BI provides powerful tools for interactive and professional visual reporting.

Week 7 Day 5 :16/01/2026

- Leave

Module 8: Deployment & Documentation

Week 8 Day 1 :19/01/2026

- Learned about deployment on **Render**.
- Attempted deployment with project; initial attempt unsuccessful.
- Created **PPT presentation**.

Week 8 Day 2 :20/01/2026

- Re-attempted deployment; analyzed mistakes from previous try.
- Tried deploying on a different platform.
- Prepared **project report** and updated **README** file.

Week 8 Day 3 :21/01/2026

- Successfully deployed **backend on Render** and **frontend on Netlify**.
- Final live project URL: https://exoplanet-swaraj.netlify.app/

Key Concepts learned

- Fundamentals of web application deployment
- Deploying backend services and frontend applications separately
- Understanding cloud platforms such as Render and Netlify
- Identifying and fixing common deployment issues

Important Links :

1. Report : Project Report
2. PPT : Project PPT
3. Project GitHub : Swaraj Github
4. Deployed URL : https://exoplanet-swaraj.netlify.app/