Exoplanet Detection & Characterization Challenge (Kepler DR25)

Background

The **Kepler Space Telescope** identified thousands of potential exoplanets by monitoring stellar brightness and detecting periodic transit signals. However, not all detected signals correspond to real planets — many arise from **astrophysical false positives**, instrumental effects, or stellar variability.

In this challenge, you are provided with a dataset derived from the **final Kepler Data Release (DR25)**, consisting of **transit observables** for Kepler Objects of Interest (KOIs) merged with **host-star parameters**. The dataset reflects the uncertainty, incompleteness, and biases inherent in real astronomical observations.

Project Goal

You will work on **two supervised learning tasks** using the same dataset and further integrate the model into a **modularized** web application

---

Objectives

- Accepts user inputs via a frontend interface
- Ensures robust input validation
- Communicates securely with a backend service
- Applies a consistent preprocessing pipeline
- Performs real-time predictions using pre-trained models
- Presents results with interpretable outputs and visual insights
- Optionally maintains a history of user inputs and predictions

---

Dataset Description

The dataset contains **one row per Kepler Object of Interest (KOI)** and includes:

Transit-related features

- Orbital period
- Transit depth and duration
- Impact parameter
- Signal-to-noise metrics
- Number of observed transits

Host-star properties

- Effective temperature
- Surface gravity
- Metallicity
- Stellar mass, radius, and density
- Associated measurement uncertainties

All features are provided **in raw form**, without normalization, imputation, or feature synthesis.

Required System Architecture

1. Frontend Layer (User Interface)

- Provide a clean and intuitive interface for users to enter input features required by the model

- Display brief descriptions or tooltips explaining each input feature and relevant ML terms
- Allow users to submit input data for prediction
- Display prediction results and visualizations clearly

2. Frontend Validation

- Perform client-side validation to ensure:
    - Required fields are not empty
    - Input values are within acceptable ranges
    - Data types (numeric, categorical, etc.) are correct
- Provide immediate feedback to users for invalid inputs before making API calls

3. API Communication with Backend

- Send validated input data from the frontend to the backend via RESTful API calls
- Ensure secure and structured data transfer (e.g., JSON format)
- Handle API responses, errors, and timeouts gracefully

4. Build ML Pipeline

- Reproducible preprocessing pipeline
- Feature selection informed by EDA
- Support for both classification and regression inference

5. Model Inference

- Load a pre-trained machine learning model in the backend
- Perform real-time inference on preprocessed input data
- Support extensibility for future model updates

6. Prediction Output and Visualization

- Return prediction results to the frontend in a structured format
- Display:
    - Final prediction values (e.g., class label, probability, regression output)
    - Confidence scores if applicable
- Generate simple visualizations (e.g., bar charts, probability distributions, trend graphs) to improve interpretability

7. Bonus Implementations

- Store user inputs, timestamps, and prediction outputs in a database
- Enable tracking of past predictions for auditing, debugging, or analytics
- Ensure data privacy and secure storage practices
- Build low latency pipeline and reliable predictions

Predictive Tasks

Task A — Classification

Goal:
Distinguish **CONFIRMED exoplanets** from **FALSE POSITIVE** transit signals.

- **Target column:** `koi_disposition`

Participants are expected to carefully inspect the dataset and frame the problem accordingly.

Task B — Regression

Goal:
Predict the **planetary radius** (in Earth radii) for exoplanets.

- **Target column:** `koi_prad`

Important Notes

- Some features may not be available at prediction time in real observational scenarios.
- Certain columns may be appropriate for one task but not the other.
- Participants are expected to perform **exploratory data analysis** and make **informed feature-selection decisions**.
- Care should be taken to **avoid target leakage**.
- You can choose any tech stack you want.
- **Suggestions for beginners** -
  You can use
  React.js for Frontend
  Firestore firebase or PostgreSql for Databases
  Node/ Django/ Flask for backend

  Whichever you find easier to work with
- A deployed web application would gain more points than a locally hosted application.

Evaluation

Classification

- Metrics: **F1-score**, **ROC–AUC**
- Evaluation focuses on the binary distinction defined in Task A.

Regression

- Metrics: **RMSE**, **MAE**

- Evaluation is performed on physically meaningful predictions.

System Development Performance

- Correctness and consistency of input validation and preprocessing
- Reliability and latency of real-time inference
- Robust API communication and error handling
- Clarity and interpretability of prediction outputs

Data Source

The dataset is derived from the **NASA Exoplanet Archive**, based on the **Kepler DR25 cumulative KOI catalog** and corresponding stellar parameters.

Participants are encouraged to cite the NASA Exoplanet Archive in any publications or reports resulting from this work.