NASA Space Apps Challenge 2025: A World Away - Hunting for Exoplanets with AI
5-Day Preparation Plan

Team WebbScope

September 30, 2025

Prepared for the NASA Space Apps Challenge, October 4-5, 2025
Challenge: A World Away: Hunting for Exoplanets with AI
Team Name: WebbScope

Contents

1 Introduction

The "A World Away: Hunting for Exoplanets with AI" challenge, part of the NASA Space Apps Challenge 2025, tasks participants with developing an AI-driven tool to detect exoplanets in astronomical datasets, such as light curves from the Kepler or TESS missions. The goal is to automate the identification of planetary transits (periodic dips in starlight) or radial velocity signals amidst noisy data, contributing to discoveries for missions like the Habitable Worlds Observatory. This advanced challenge emphasizes machine learning (ML), NASA data integration (at least 70% of the solution), innovation, technical execution, impact, and storytelling.

This document outlines a detailed 5-day preparation plan (September 30, 2025, to October 4, 2025) for a small team (2-4 members) with basic Python/ML skills to build a working prototype for the 48-hour hackathon. The plan assumes 4-6 hours of daily work, using free tools (Python, Google Colab, GitHub) to create an ML model with a web interface. The final deliverable is a functional pipeline (data preprocessing, AI model, Streamlit app) achieving at least 85% F1 score on test data, documented for submission.

2 Challenge Overview

The challenge requires processing vast telescope datasets to identify exoplanet signatures, addressing the "needle in a haystack" problem of distinguishing true signals from noise (e.g., stellar variability, eclipsing binaries). Key components include:

- **Objective**: Build an AI tool to detect exoplanets with high precision, reducing false positives.
- **Data Sources**: NASA Exoplanet Archive, MAST Portal (Kepler/TESS light curves), Kaggle datasets.
- **Judging Criteria**:
  - NASA Data Use (70%+ of solution).
  - Innovation (novel ML approaches).
  - Technical Execution (scalable, robust code).
  - Impact (e.g., faster candidate identification).
  - Storytelling (clear demo and explanation).
- **Impact**: Accelerate exoplanet discovery for future space missions.

The prototype will include a convolutional neural network (CNN) or random forest classifier, a Streamlit web app for user input, and a GitHub repository with code, documentation, and a demo video.

3 5-Day Preparation Plan

This section details daily tasks, tools, and deliverables to prepare for the hackathon, assuming a team with basic Python/ML skills and access to a laptop (8GB+ RAM) and Google Colab for GPU training.

3.1 Day 1: Research, Data Acquisition, and Setup (September 30, 2025)

Goal: Build a foundational understanding and acquire datasets. **Time**: 4 hours.

3.1.1 Morning: Background Research (1-2 hours)

- Study the challenge page (https://www.spaceappschallenge.org/2025/challenges/a-world-away-hunting-for-exoplanets-with-ai) for specific requirements (e.g., focus on transit method).
- Learn the transit method: Watch NASA's 10-minute video on exoplanet detection (https://www.youtube.com/watch?v=exoplanet-transit). Key terms: Light curves, period folding, box-least squares (BLS).
- Review ML applications: Skim papers like "Automated Exoplanet Detection with Deep Learning" on arXiv (https://arxiv.org). Note challenges: Imbalanced data, overfitting.
- **Output**: 1-page Google Doc with 3-5 takeaways (e.g., "Target: Detect transits with <1% depth").

3.1.2 Afternoon: Data Exploration (2-3 hours)

- Access datasets:
  - Kepler Q1-17 DR25 (https://lightkurve.org/tutorials): 10,000 labeled light curves.
  - TESS Sector 1-50 from MAST (https://mast.stsci.edu): Download 100-500 curves.
  - Kaggle dataset: "Kepler Exoplanet Search Results" (https://kaggle.com/datasets/keplergo/lightcurves-tess).
- Setup environment: Install `lightkurve`, `astropy`, `scikit-learn`, `tensorflow`, `streamlit` via `pip`. Use Colab for GPU.
- Explore data: Plot 5-10 light curves using:

```
from lightkurve import search_lightcurve
lc = search_lightcurve("Kepler-10", mission="Kepler").download()
lc.plot()
```

- Compute stats (depth, period) with `search`$_{p}eriodogram()$.`Output` $: Folderwith200 - 500preprocessedcurves(CSV/JSON : time, flux, label).InitializeGitHubrepo :$ `spaceapps-exoplane`

3.1.3 Evening Review

Identify data gaps (e.g., need more negative samples). Plan questions for Space Apps Discord.

3.2 Day 2: Brainstorming and Model Design (October 1, 2025)

Goal: Design the ML pipeline and sketch a baseline model. **Time**: 4 hours.

3.2.1 Morning: Solution Scope (1-2 hours)

- Define MVP: Input = light curve file; Output = Planet detection (Yes/No, confidence, period/depth).
- Team roles: One on data prep, one on ML, one on UI (if applicable).
- Metrics: Aim for 80% F1 score to minimize false positives.
- Innovation: Compare CNN on folded curves vs. traditional BLS.

3.2.2 Afternoon: Model Selection and Pseudocode (2-3 hours)

- Choose framework: `scikit-learn` for random forest; `tensorflow` for CNN.
- Pipeline:
  - Preprocess: Normalize flux, sigma-clip outliers, fold by period.
  - Features: BLS peaks, Fourier transforms, or raw time-series.
  - Model: Random forest baseline; optional CNN (1D, input shape: (1000, 1)).
- Pseudocode:

```
import numpy as np
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
model = RandomForestClassifier(n_estimators=100)
model.fit(X_train, y_train)
preds = model.predict(X_test)
from sklearn.metrics import f1_score
print(f1_score(y_test, preds))
```

- Augment data: Use `batman` package for synthetic transits.
- **Output**: Notebook with baseline model (70% accuracy). Commit to GitHub.

3.2.3 Evening Review

Evaluate model choice. Prep forum questions (e.g., ”Handling TESS noise?”).

3.3 Day 3: Prototype Development (October 2, 2025)

Goal: Build a working end-to-end pipeline. **Time**: 4-5 hours.

3.3.1 Morning: Data Pipeline (1-2 hours)

- Automate loading: Script for 500+ curves (50% planets).

- Feature engineering: Use `lightkurve.flatten()` and `to`$_{p}eriodogram()$.

3.3.2 Afternoon: Model and UI (2-3 hours)

- Train model: Implement CNN (if feasible):

  ```
  import tensorflow as tf
  model = tf.keras.Sequential([
      tf.keras.layers.Conv1D(32, 3, activation='relu', input_shape=(1000, 1)),
      tf.keras.layers.GlobalMaxPooling1D(),
      tf.keras.layers.Dense(1, activation='sigmoid')
  ])
  model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy']
  model.fit(X_train, y_train, epochs=10)
  ```

- Build UI: Streamlit app for file upload and visualization:

  ```
  import streamlit as st
  uploaded_file = st.file_uploader("Upload light curve CSV")
  if uploaded_file:
      # Load, predict, plot
      st.plotly_chart(fig)
  ```

- **Output**: Local demo video (1-min). Commit to GitHub.

3.3.3 Evening Review

Test on 10 new curves. Note bugs (e.g., memory issues—downsample).

3.4 Day 4: Iteration, Testing, and Optimization (October 3, 2025)

Goal: Refine prototype for robustness. **Time**: 4 hours.

3.4.1 Morning: Validation and Debugging (1-2 hours)

- Split data: 70/15/15 train/val/test. Use cross-validation.
- Test edge cases: Noisy/short curves, multi-planet systems.

3.4.2 Afternoon: Enhancements (2-3 hours)

- Optimize: Use Colab GPU; prune model for <1 min/inference.
- Storytelling: Write README.md (problem, innovation, impact).
- Ethics: Discuss biases (e.g., over-detection in bright stars).
- **Output**: Improved model (>85% F1); deploy to Streamlit Sharing.

3.4.3 Evening Review

Mock 5-min pitch: Problem, Solution, Demo, Impact.

3.5 Day 5: Polish and Submission Prep (October 4, 2025)

Goal: Finalize deliverables. **Time**: 3 hours.

3.5.1 Morning: Documentation (1-2 hours)

- Polish repo: Clear structure (`data/`, `src/`, `docs/`), MIT license.
- Record 2-3 min demo video (Loom: upload → predict → viz).
- Write 300-word abstract tying to NASA goals.

3.5.2 Afternoon: Final Tests (1-2 hours)

- Run pipeline on unseen TESS data. Backup to drive.
- Rehearse hackathon pivot (e.g., add API).
- **Output**: Submission-ready repo, video, abstract.

3.5.3 Evening Review

Join Space Apps orientation. Relax for hackathon.

4 Tools and Resources

- **Environment**: Python 3.10+, Jupyter/Colab, GitHub.
- **Libraries**: `lightkurve`, `astropy`, `scikit-learn`, `tensorflow`, `streamlit`, `batman`.
- **Data**: NASA Exoplanet Archive (https://exoplanetarchive.ipac.caltech.edu), MAST Portal (https://mast.stsci.edu), Kaggle (https://kaggle.com).
- **Community**: Space Apps Discord, Lightkurve tutorials (https://lightkurve.org).

5 Potential Pitfalls and Mitigations

- **Data Overload**: Start with 100 samples, scale to 500.
- **Compute Limits**: Use pre-trained models from Hugging Face if training fails.
- **Team Dynamics**: Daily Zoom standups.
- **Hackathon Strategy**: Use 48 hours for polish (e.g., ensemble models) and networking.

6 Conclusion

This 5-day plan equips Team WebbScope to deliver a competitive prototype for the "A World Away" challenge, balancing technical rigor with hackathon constraints. By leveraging NASA data, a robust ML pipeline, and a user-friendly interface, the team aims to achieve >85% F1 score and a compelling demo, positioning for top judging. Updates and refinements will occur during the hackathon, with a focus on clear storytelling and real-world impact.