Group Mates **Milestone 1**

The group consist of Kenny Zhao, Alan Zhou, and Richard Nguyen. Everyone will take part in everything and evenly, both splitting the research component for our topic and developing our ML model.

Context

Identifying the Problem: As Earth's resources dwindle, we must look beyond our solar system for solutions. With over 5,600 confirmed exoplanets (NASA) and billions more uncharted, the challenge lies in identifying which might sustain life without direct exploration.

The Challenge: Exoplanets are too distant for direct exploration, with even the nearest being four light-years away. The complexity of life and planetary systems adds to the difficulty, as countless variables must be analyzed.

The Role of Machine Learning: Due to the inefficiency of humans, Machine Learning can efficiently classify exoplanets, analyzing vast datasets to identify those with potential for habitability. This method saves significant time and effort, enabling astronomers to focus on promising candidates.

Aspects to Solve: Our goal is to streamline the identification of potentially habitable exoplanets. This will guide future exploration, minimizing wasted resources and ensuring efficient prioritization of interstellar travel targets. Not only that, but learning about other planets can be beneficial for ours as well.

Relevance and Impact: This research addresses pressing issues like overpopulation and resource scarcity, while also satisfying humanity's curiosity about extraterrestrial life. Eventually we will need to reach for the stars and the pursuit of this knowledge drives technological innovation with benefits for life on Earth.

Datasetsw

NASA Exoplanet Archive (Orbital and Planetary Properties):
Source: NASA Exoplanet Archive
Description/Use: Contains key orbital and planetary properties such as orbital period, planetary radius, and eccentricity for confirmed exoplanets and candidates. Critical for understanding the planetary environment and its potential habitability.
Size: The cumulative KOI table contains 9,000+ entries (Kepler Objects of Interest), with features relevant to transit detection and validation.

Gaia DR3 Subset (Host Star Properties):
Source: Gaia DR3 Auxiliary Data (fetched dataset from query the database)
Description/Use: Provides stellar characteristics of exoplanet host stars, including effective temperature, luminosity, age, radius, and surface gravity. Essential for calculating habitability metrics such as equilibrium temperature and habitable zone constraints.
Data Query Details: Selected from the gaiadr3.gaia_source and gaiadr3.astrophysical_parameters tables using ADQL.
Size: The query returns 1,000 matched host stars with complete stellar data.

PHL Habitable Zone Catalog (Habitability Labels):
Source: PHL Habitable Zone Catalog via Kaggle.
Description/Use: Pre-determined habitability labels for exoplanets based on position within habitable zone and planetary characteristics. Provides ground truth habitability labels for training and evaluating machine learning models.
Size: Contains 4,000 entries with features such as habitable zone classification and Earth Similarity Index (ESI).

Kepler Light Curve Data (Transit-based Features):
Source: Kepler KOI Cumulative Table
Description/Use: Contains light curve transit-based features derived from Kepler observations (transit duration, depth, and signal-to-noise ratio (SNR)). Crucial for understanding dynamics of exoplanetary transits and detecting potential candidates. Transit-based features will help characterize potential exoplanets and correlate transit data with habitability labels.
Size: Covers 9,000+ KOIs with 50 relevant features.

All datasets will be divided using proper cross validation techniques such as K-fold to get both training and validation set. We want our model to be well versed in multiple sets to prevent overfitting in one set of data.

Proposed solution
Our model's objective is predictive. We will attempt to predict whether or not an exoplanet is habitable by grouping it into three categories: 0 (not habitable), 1 (habitable), 2 (unknown). Our ML model will be trained based on a combination of datasets incorporating a plethora of input features, to predict if these observable features can represent a habitable exoplanet.

Target Variable: Not Habitable (0), Habitable (1), Unknown (2) | (Unknown (2) may represent cases where features fall into ranges that overlap between habitable and non-habitable classes, or when there is insufficient data to make a clear decision)

Input Variables (Features): Planetary Attributes, Orbital and Stellar Attributes, Transit-Based Features, Habitability Levels

- Specific features of these attributes are previously mentioned in the Dataset section
- It is worth noting that with this many features, we expect some of them to not be critical to making our prediction. Thus, there is a high probability that not all these features will be used in our final ML model.

Our ML model will use neural networks. Neural networks are a versatile choice for structured, tabular data like exoplanet attributes. By using neural networks, we aim to capture non-linear relationships between planetary and stellar properties, adapt to the complexity of the problem by automatically learning feature interactions, and scale well to the large datasets and a high number of input features.

Data preprocessing: normalization (scale/standardize numerical features), categorical-encoding (use one-hot encoding for categorical features), data split around 70% train and 30% test.

Evaluation: K-fold cross validation and Metric calculation (Accuracy, Precision, Recall) As mentioned in the Dataset section of our report, we intend to use all datasets and split them appropriately to avoid overfitting and ensuring an accurate model in all ranges of data.

From the article, there are already existing projects that use machine learning to detect exoplanets. The one below uses CNN to see if an exoplanet exists given a star. We wanted to take inspiration from this and change it up a bit by identifying and classifying if exoplanets are habitable or not.
Source: Detection of Exoplanets using Machine Learning | International Journal of Research Publication and Reviews

We plan to use the **PyTorch** or **TensorFlow** libraries to design, train, and evaluate the neural network. **Numpy**, **Pandas**, and **Scikit-learn** will handle data preprocessing and feature engineering, while **Matplotlib** and **Seaborn** will be used for visualizations.

Group Mates **Milestone 2**

Kenny worked on implementing the model and writing the model specification/limitations portion of the report. Alan worked on preprocessing our data and worked on the preprocessing section of the report. Richard worked on evaluating/testing the model and worked on the evaluation/results portion of the report.

Context

Identical to that of milestone 1.

Datasets

After careful inspection of our datasets, we have done the slight modifications from milestone 1.

After careful inspection, we have reduced our total datasets to just 3. We noticed that the dataset Kepler Light Curve Data (Transit-based Features) and NASA Exoplanet Archive (Orbital and Planetary Properties) both share almost identical features. Thus we have decided to not use Kepler Light Curve Data (Transit-based Features) to remove redundancy.

Information for the rest of the datasets are identical to that of milestone 1, with the added information that the dataset NASA Exoplanet Archive (Orbital and Planetary Properties) contains 49 features, PHL Habitable Zone Catalog (Habitability Labels) contains 118 features and Gaia DR3 Subset (Host Star Properties) contains 8 features.

Preprocessing

Before beginning our preprocessing methods, we first remove redundant features from all datasets and convert them all to CSV's. This way we optimize our datasets better and not waste resources preprocessing useless features or having those features affect our model.

After thorough research of every feature from credible sources such as NASA, we dropped the necessary columns in each dataset respectively before merging them. The columns dropped are features that we think do not contribute much to the overall prediction whether an exoplanet is habitable or not or are features related to other features, which indicates redundancy.

With the datasets prepared, we combined all available data into a single comprehensive dataset by aligning records based on shared attributes, such as planet name. This process did not involve a full merge of the datasets, meaning some entries might include more features than others. However, this approach enables a greater variety of data and increases the flexibility of our model, which we account for during training. Currently, we have excluded the GAIA dataset due to the incompatibility of its features. For our preliminary model, we are focusing on testing and training with a joined dataset derived from NASA and PHL datasets.

For handling NA values, since we are doing just a preliminary analysis, we filled them in with placeholder values (0's for continuous variables and unknown for categorical)

Labels remain to be 0 (Not Habitable), 1 (Habitable), and 2 (Unknown). These labels are given to each exoplanet based on a combination of 7 features: koi_prad, koi_teq, koi_steff, koi_srad, koi_period, koi_impact, koi_slogg. If a specific combination of these 7 features satisfies, the exoplanet will be assigned label 1 (indicating that it is Habitable). For instance, koi_prad represents Planetary radius. We set the habitability range to be between 0.5 Earth radii, and 2 Earth radii, to exclude the small celestial bodies likes moons or asteroids, as well as gas giants. A similar logical approach applies for the remaining features.

With the preparations done, we scale our continuous variables using a standard scalar so that each feature keeps their respective range of values and for categorical variables, we use one hot encoding. We do not have any time series data and thus do not need to handle it.

Finally, we do a 20-80 randomized split of our data to reduce bias for testing and training respectively.

Model Specification
For our preliminary model, we are really interested in neural networks and thus implemented one as our starting benchmark. We wanted to keep it simple just to set a foundation for our model and thus kept it with 3 layers, ReLU activation function with Cross Entropy loss function and optimizer of stochastic gradient descent. We did not want to implement a regularization technique as we want our model to be as simple as possible and see how much error and improvement can be done for milestone 3.

For our training parameters, we did a batch size of 64, learning rate of 0.001 and epoch of 100. We believed these parameters are a good foundation to begin with.

Finally, for every 10 epochs, we printed out the training loss along with its respective validation loss. More detail is explained in our evaluation section.

Evaluation

Our evaluation strategy is staying the same as milestone 1, however for our preliminary model, we decided during training, we will do a held-out validation set to plot the losses (20% validation, 80% test). This way we can see our performance of our model live during training which provides us with crucial information in tuning our model.

For now, we will evaluate our model's performance using accuracy. This approach aligns with our primary objective: predicting the habitability of exoplanets. Accuracy serves as a fundamental metric to assess how effectively the model achieves this goal.

Preliminary Results:
From our initial testing, here are the results:
Number of features: 283
Training data shape: torch.Size([2590, 283])
Test data shape: torch.Size([810, 283])
Epoch [10/100], Training Loss: 0.7145, Validation Loss: 0.6672, Validation Accuracy: 1.0000
Epoch [20/100], Training Loss: 0.1782, Validation Loss: 0.1296, Validation Accuracy: 1.0000
Epoch [30/100], Training Loss: 0.0094, Validation Loss: 0.0032, Validation Accuracy: 1.0000
Epoch [40/100], Training Loss: 0.0048, Validation Loss: 0.0002, Validation Accuracy: 1.0000
Epoch [50/100], Training Loss: 0.0034, Validation Loss: 0.0001, Validation Accuracy: 1.0000
Epoch [60/100], Training Loss: 0.0037, Validation Loss: 0.0000, Validation Accuracy: 1.0000
Epoch [70/100], Training Loss: 0.0035, Validation Loss: 0.0000, Validation Accuracy: 1.0000
Epoch [80/100], Training Loss: 0.0037, Validation Loss: 0.0000, Validation Accuracy: 1.0000
Epoch [90/100], Training Loss: 0.0026, Validation Loss: 0.0000, Validation Accuracy: 1.0000
Epoch [100/100], Training Loss: 0.0019, Validation Loss: 0.0001, Validation Accuracy: 1.0000
Test Accuracy: 1.0000

Limitations
We can see here from initial testing, our data is a a bit misleading. Without a regularization technique implemented, we can see our model overfitting here as indicated by the validation loss being 0 post 50 epochs. Not only that, but a constant test accuracy of 1 is very concerning as our model should not be this accurate and means something could be wrong with our preprocessing or model itself. We plan to implement a regularization technique, change up the parameters a bit, possibly the architecture of the neural network and test our model against others such as SVM's for milestone 3.

Group Mates **Milestone 3**

For this portion, roles have stayed identical. Additional roles includes: Kenny worked evaluation of the model and Evaluation/Limitations/References portion of the report, Alan worked on improvements to our preprocessing strategy and worked on the Preprocessing Improvements section of the report, Richard worked on improving our original model and worked on Model Specifications Improvements portion of the report.

Preprocessing Improvements

After inspection, we saw that our labeling strategy was off and producing labels that had a lot of bias towards labeling planets Not Habitable. Hence, our model over fitted quickly and always hit 100% validation accuracy.

Thus we re-tuned our parameters for labeling based off our research done through various sources such as NASA. Instead of the previous set of features used, we modified our criteria to be based on the following 5 features: P_RADIUS_EST, P_TEMP_EQUIL, P_FLUX, P_PERIOD, and P_MASS_EST. For instance, P_TEMP_EQUIL represents equilibrium temperature. The habitability range is set to be between 100K and 450K, the points where water would freeze entirely, and boil off. In order to reduce the previous heavy bias towards the exoplanet labeling of 0 (Not Habitable), we label the planet 1 (Habitable) if at least 4 of the 5 features satisfy its own criteria. Next, if less than 3 of the 5 features satisfy, we label the planet 0 (Not Habitable). Finally, if exactly 3 of the 5 features satisfy, we label the exoplanet as 2 (Unknown).

Implementing this labeling techniques results in labelling distribution of approximately the same amount of Unknown and Not Habitable cases, and a small but noticeable amount of Habitable cases. This is a significant improvement compared to the previously used heavily biased technique.

Model Specifications Improvements

Our first improvement is to our optimizer. After through research and testing, we switched over to Adam optimizer compared to our original of Stochastic Gradient Descent. Adam optimizer in general is very good at handling high-dimensional and sparse datasets, which is perfect for us. Not only that, but its adaptive learning rates help distinguish between features that are less impactful compared to useful ones. With our dataset containing over 200 features, this improved our model accuracy by over 70%.

Next improvement is to our layers. Adding an additional 4th layer helped increase our model accuracy by about 10%. We did test for more additional layers, but the increase complexity of more hidden layers seemed to make our model over fit more than it should. A good balance came with 4th layers.

We kept our learning rate the same and increased our epochs to 200. The reason why is because after applying our regularization technique of early stopping, we notice our model plateaus at around 150 epochs. Furthermore, the regularization technique helps our model avoid over fitting, producing a more accurate result.
Lastly, we kept our loss function the same as Cross Entropy Loss. This loss function is usually best for Neural Networks and performed very well for us.

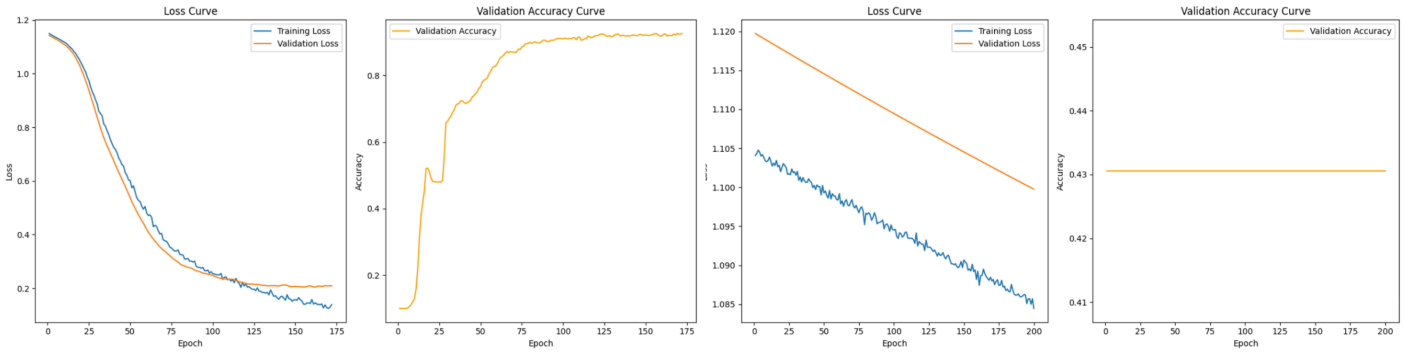

Figure 1: Adam Optimizer, accuracy 91% (Left) Vs. SGD Optimizer, accuracy 43% (Right)

Evaluation

For our evaluation, the biggest impact came from our optimizer choice which drastically changed our model as shown in Figure 1. Otherwise, all other improvements to our model showed minor adjustments and overall our model looked about the same as the graphs in Figure 1 with the Adam Optimizer. We can see the big point where our model fails comes from the optimizer choice.

We also decided to test our model against other models as well to show how well it holds up to models that are built for classification such as Logistic Regression. In Figure 2, we kept the same parameters for both models and we can see that our model outperforms the multi logistic regression using less epochs. As a matter of fact, during further testing, it took the MLR model around 5000 epochs just to reach the same performance as our NN model. This concludes that our model requires much less resources to achieve an accurate result and is much faster and more efficient.

Our final model showed a very good accuracy of around 90-91% with our newly selected parameters. Finally, as mentioned in the preprocessing section, a big part in our model showing good accuracy is due to our new labeling strategy.

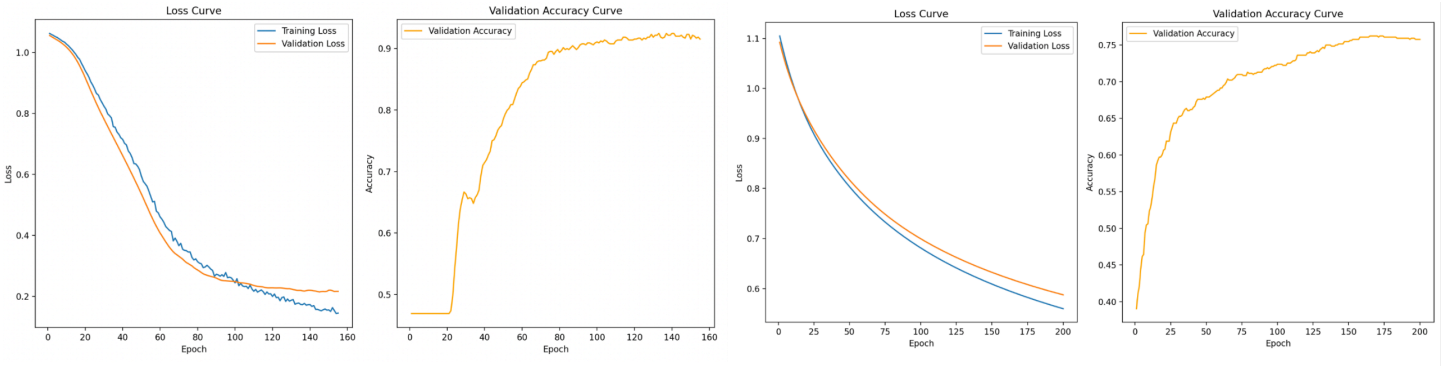

Figure 2: NN Model, accuracy 91% (Left) Vs. MLR Model, accuracy 75% (Right)

Limitations

The regularization technique was implemented from Milestone 2. With a finalized neural network architecture and a regularization technique implemented, the major limitation remains dependent on the initial labeling of the data. By observation, results (accuracy) vary greatly between our training from Milestone 2 and that from Milestone 3. This can be attributed to the different labeling technique along with regularization. It is worth noting that around 10 different labeling techniques were tested on, before we settled on the current outlined above in the preprocessing section.

References

Lecture 6 Classification II (+demo code), Lecture 7 Data Preprocessing (+OneHotEncoding demo code), Lecture 13 Neural Networks, Lecture 14 Backpropogation, Lecture 15 Evaluation, Lecture 16 Fairness, Lecture 17 Measuring Bias | Introduction to Statistical Learning Chapters: 2, 4, 6, 10 | NASA, Adam Optimizer Research, Preprocessing