# Identification of Exoplanet Orbitals


| | |
|---|---|
| Name: | **Akanksha Singh** |
| Registration No./Roll No.: | 2311001 |
| Institute/University Name: | IISER Bhopal |
| Program/Stream: | DSE |
| Problem Release date: | August 17, 2023 |
| Date of Submission: | November 19,2023 |


## 1 Introduction

The objective of the project was to prepare a regression model that can predict the semi-major axis of exoplanets by relying on the designed feature extraction framework, providing crucial estimates about those orbital parameters that offer insights into the planet's distance from its host star, using the data provided by the NASA Exoplanet Archive. Defining an exoplanet as a planet outside our solar system, traditional methods for discovering exoplanets involve observing changes in a star's brightness or the wobble in its movement caused by the gravitational pull of orbiting planets. However, machine learning offers a promising avenue to sift through large datasets, extract patterns, and make predictions based on the gathered information. This paper contains various methods and techniques to extract the relevant parameters from the Kepler Dataset and use them to classify a candidate as an exoplanet. It provides methods for data classification, data cleaning, and various models for exoplanet candidate detection. The description of the dataset is given below:

• exoplanet trn data.csv: This file contains training feature vectors. It has 17969 rows (data points) and 288 columns (features).

• exoplanet trn data targets.csv: This file contains training target values (semi major axis) . It has 17969 rows (data points) and 2 columns (index and semi major axis).

• exoplanet tst data.csv: This file contains test feature vectors. It has 1997 rows (data points) and 288 columns (features).

• exoplanet data desc.txt: This is a text file that contains the description of all 288 features available in our dataset.

## 2 Methods

The github page of the project can be found here. The project's proceeding can be divided into two sections, Data Processing and Model Execution. The various methods used in each of these sections are extensively discussed below:

### 2.1 Preprocessing

There were four data files as discussed in the section above. Three of them were imported to form three databases (using the Pandas module for Python), called train data, which contained the training data, train labels, which contained the labels for each of the instances in the training dataset, and test data which contained no labels and was a relatively shorted data as compared to the training data. The models were supposed to predict the labels for the test data. The training data contains, 17969 instances, whereas the test data has 1997 instances with a total of 288 features each. A thorough look at the training dataset revealed that the dataset contained a lot of features that had no values for most of the instances. According to this article, any feature which contained more than 90training. Hence we dropped such features, to improve the working of our models.

### 2.2 Missing Data Feature Removal

1. Calculate the total number of null values for each feature in the dataset.
2. Arrange the features in descending order of number of null values present in them.
3. Select the top features that have more null values than the decided threshold.
4. Remove these features from the dataset and create a new dataset named tr05.

Since some of the instances in few 'numerical' features contained non-numeric values, it lead to their classification as a Categorical Column, which is actually incorrect. This is resolved in the following section.

### 2.3 Wrongly Classified Column Correction

1. Create a list of 'Object Type' features in the dataset.
2. Find the total number of such columns. (Found to be 37)
3. Use domain expertise to manually identify such column and make a list of wrong classified columns.
4. Convert the datatype of the above identified columns as floats. (The datatype to which they actually should be belonging)

### 2.4 Filling the Missing Data

As we can observe that there are a lot of 'NaN' values in the columns, we will replace these values with the median of the data as it tends to be a more appropriate choice here, because the range of data is huge, and the extremities are separated by a large gap.
1. Calculate the Median of each feature vector.
2. Find the null values in each of the feature vector and replace it with the respective median.
3. Check if all the feature columns are free of Nan values.
4. Remove such instances when features contain non-numerical values for numerical features.

### 2.5 Outlier Removal and Data Cleaning

Several columns in the dataset tend to contain contrasting data values, ranging from micro-range to mega-range. This disparity within the dataset exceeds the acceptable range of standard variance. Consequently, we removed outliers in an attempt to enhance correlation.

We eliminated all instances that lie beyond the following threshold:

$$\textbf{Outlier value} \geq \mu \pm (10 \times \sigma),$$

where $\mu$ is the mean of the data, and $\sigma$ is the standard deviation of the dataset.

Some feature vectors became null after this operation; hence, we removed them as well.

Where μ is the mean of the data, and $\sigma$ is the variance of the dataset. Some feature vectors become null after this operation hence we remove them as well.

### 2.6 Correlation Calculation

The Correlation was calculated for each of the feature in the training set and a C-map was produced to visualise the features having maximum correlation with the target variable. The Planet Orbital Period, and its errors were found to have maximum correlation with the target variable.

### 2.7 Feature clustering

In this step, K-means clustering was applied to the top 20 highly correlated features to the target variable. It reinforced the initial feature selection based on correlation. Along with the Planet Orbital Period, Number of Radial Velocity Time Series, and Magnitude Upper Unc have shown the highest correlation with the target variable $_{\text{semi\_major\_axis}}$, as per figures 1 and 2.

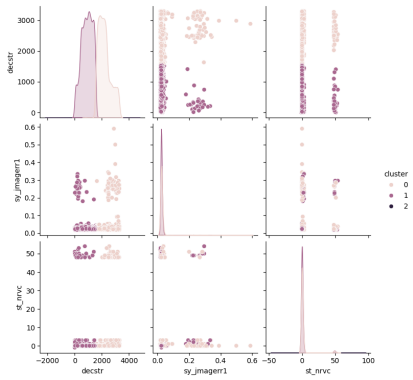


Figure 1: Scatter Plot with Cluster Assignment

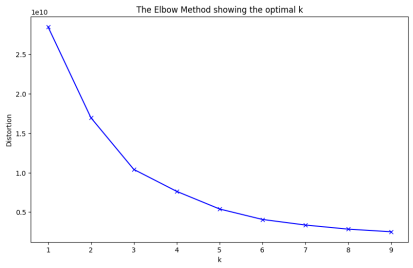


Figure 2: The Elbow method showing optimal k

### 2.8 Feature Engineering

Applying Kepler's third law, we applied certain mathematical transformations on the orbital period to form two new feature vectors which were included in the training dataset in order to increase training quality. The correlation with these new features was also found to be substantially high.

### 2.9 Modelling

The following models were performed on the training set and the the corresponding RMS values were calculated for each of the training models. The models run on the dataset were, 1. Linear Regression 2. Decision Tree 3. Support Vector Regression 4. Random Forest Regression The random forest regression was found to provide the minimum error hence it was utilised to perform the final prediction on the test dataset. The final values of the target variable as predicted by the Random Forest were saved into a text file named, predicted semi major axis.txt.

## 3 Experimental Analysis

**Mean Square Error (MSE)** is the average of the square of the errors. The lesser the error, the better the model.

The *Mean Square Error (MSE)* for different models after hyperparameter tuning are as follows:

- Linear Regression: 0.061709498950965926
- Decision Tree Regression: 0.027570866998770534
- Support Vector Regression: 0.010523071118444638
- Random Forest Regression: 0.007954253841898715
- Ridge Regression: 0.0626013702565612
- Lasso Regression: 0.08235888994081808
- Adaboost Regression: 0.07281883489378284

The $\mathbf{R^2}$ score for different regression models are as follows:

- Linear Regression: 0.9096385144532235
- Decision Tree Regression: 0.9572435527737777
- Support Vector Regression: 0.9845910215616465
- Random Forest Regression: 0.9883525517823237
- Lasso Regression: 0.8794015221392542
- Ridge Regression: 0.908332543452646
- Adaboost Regression: 0.8933710659032232

The lowest value of mean-squared error is found to be for Random Forest Regression. Hence, this is our best model. Also, we can see that Random Forest Regression is giving the highest $\mathbf{R^2}$ score as well, which supports our decision of considering it the best model.

# 4 Discussions

The proposed method is capable of handling all types of missing data, outliers and wrongly classified categorical data. The research data is not always clean for use. This method is self-sufficient to clean any new research data and can make predictions of the semi-major axis for the exoplanets. The method is not dependent on any prior domain knowledge, it learns by itself the important features that can be used for regression problems. There is no bias to any single feature in our model. The feature selection is done on the basis of the correlation matrix. This way, even a person who is not from the Astrophysics domain can use this method and get insights into the Exoplanet data.

The main limitation of our method is that the time complexity is high. The parameter tuning of different models on the large Exoplanet dataset takes a lot of computational power. Also, the parameters chosen for the hyperparameter tuning are hand-picked. This approach can be automized. We have done feature selection on the basis of only correlation matrix. We planned to try on with some other metrics like p-value for feature selection. We also planned on using ensemble techniques where we can use various regression models and combine them using boosting techniques to come up with a better regression model than the individual regression models. Finally, if we get our hands on the eccentricity of the exoplanets and habitable zone of star, we can use these semi-major axis values to detect whether an exoplanet is habitable or not. This will open a completely new horizon for this project, where it can be used for finding our new home!