Predicting Hazardous Near-Earth Objects With Random-Forest and Class Imbalance Techniques

Nataly Jimenez Cruz
njimene1@stevens.edu
973-914-4282
Department of Computer Science
Stevens Institute of Technology

Advised by Professor Kevin Lu
klu2@stevens.edu
Department of Electrical and Computer Engineering
Stevens Institute of Technology

***Abstract*—This research utilizes a curated dataset of twenty years' worth of Near-Earth Object (NEO) close approach data and machine learning methods to predict Potentially Hazardous Asteroids (PHAs). This study aims to improve impact risk assessments and early warning systems, enhancing threat mitigation from small solar system bodies through comprehensive data and advanced models.**

***Index Terms*—Machine learning, Near-Earth Object (NEO), Potentially Hazardous Asteroid (PHA).**

I. Introduction and Literature Search

Near-Earth Objects (NEOs) are comets and asteroids with orbits that bring them close to Earth's vicinity [1]. Composed of debris from the formation of the solar system 4.6 billion years, these leftover bits can pose potential threats to Earth and are known as Potentially Hazardous Asteroids (PHAs) [1]. According to the Center for Near-Earth Objects (CNEOs), all asteroids with an Earth Minimum Orbit Intersection Distance (MOID) of 0.05 au or less and an absolute magnitude (H) of 22.0 or less are considered PHAs [1]. Alternatively, asteroids that are smaller than about 140 meters in diameter (i.e., H = 22.0 with assumed albedo of 14%) are not considered PHAs [1]. The types of global disasters a collision from a NEO can range vastly. From producing tidal waves inundating low lying coastal areas to destruction of plant life from acid rain, blocking full sunlight to firestorms [1]. Understanding and predicting their potential impact hazard has become crucial to avoid these past occurrences from reoccurring. Machine learning has been proven to be a powerful tool in predictive analysis, allowing researchers to assess NEO hazards effectively. For instance, numerous studies have achieved superior prediction results utilizing Random Forest models; therefore, this algorithm has been a central focus in this research. This research aims to use a curated dataset using two NASA APIs for data extraction to develop a machine learning model classifying hazardous NEOs. Throughout this study, data preprocessing, feature engineering, hyperparameter tuning and class imbalance techniques have been conducted to enhance model performance.

II. Problem Description

The primary objective of this research is to develop a machine learning model that accurately classifies NEOs as hazardous or non-hazardous. Accurate predictions allow for effective responses to potential threats and impact mitigation strategies. A significant challenge was dealing with a highly imbalanced dataset, where the 'Non-Hazardous' class vastly outnumbered the 'Hazardous' class. Below is a pie chart produced with Plotly.js showcasing the major class imbalance between non-hazardous and hazardous NEOs in the dataset (see Fig. 1). This necessitated the use of specialized techniques and metrics to handle imbalanced data. The class imbalance is mainly because of the rarity of a hazardous NEO. No object has ever been assigned an integer higher than 2 on the Torino scale. The Torino scale is a method for categorizing the impact hazard associated with NEOs, ranging from 0 (no risk) to 10 (certain collision with global catastrophic effects) [3]. Other technical challenges are discussed in their respective sections of the report.

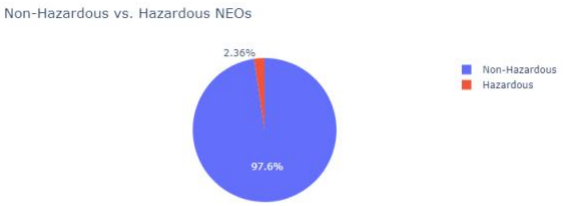

Fig. 1. NEO hazard classification distribution.

III. Research Conducted

A. Data Collection

The dataset was curated by first extracting all Earth close-approach data of NEOs from the Small-Body Database (SBDB)

Close-Approach Data API for dates between January 1, 2004 and June 1, 2024. An HTTP GET request was sent to the specific API URL to retrieve data. The response from the API was in JSON format, so the 'json.loads()' function was used to parse the JSON string and convert it into a Python dictionary. This dictionary was then used to create a data frame, with 11 fields as columns and NEO data as rows.

- **des** - primary designation of the asteroid or comet
- **orbit_id** - orbit ID used for the close-approach computation
- **jd** - time of close-approach (JD Ephemeris Time, TDB)
- **cd** - time of close-approach (formatted calendar date/time, TDB)
- **dist** - nominal approach distance (au)
- **dist_min** - minimum (3-sigma) approach distance (au)
- **dist_max** - maximum (3-sigma) approach distance (au)
- **v_rel** - velocity relative to the approach body at close approach (km/s)
- **v_inf** - velocity relative to a massless body (km/s)
- **t_sigma_f** - 3-sigma uncertainty in the time of close-approach (formatted in days, hours, and minutes; days are not included if zero)
- **h** - absolute magnitude H (mag)
- **fullname** - formatted full-name/designation of the asteroid or comet (optional - only output if requested with the fullname query parameter)

The last step was to write a python script that fetched orbital data of each object using one of NASA's Open APIs, Asteroids – Near Earth Object Web Service (NeoWs). This enhanced the original dataset with additional features for accurate hazardous prediction and visualizing orbits of celestial bodies. The orbital elements of each object could be viewed through Postman using the API URL and a NASA API Key. It is recommended to use the portion of an object's 'fullname' found within parenthesis to extract data using NeoWs. One of the biggest challenges with using this API or any one of NASA's Open APIs was the limit of API requests available (1,000 requests per hour). Because a request was made for each object, this totaled around 15 hours for complete data extraction. Time and progress libraries were used for handling sleep intervals and displaying progress bars.

The following orbital features were extracted from the NeoWs:

- **estimated_diameter** – estimated diameter of the NEO. A dictionary containing minimum and maximum diameters in kilometers
- **data_arc_in_day –** number of days spanned by the data-arc
- **observations_used –** number of recorded observations of this orbit
- **orbit_uncertainity –** MPC "U" parameter: orbit uncertainty estimates 0-9 with 0 being good and 9 being highly uncertain
- **minimum_orbit_intersection** – Earth Minimum Orbit Intersection Distance (MOID) in AU
- **jupiter_tisserand_invariant** – Jupiter Tisserand Invariant
- **epoch_osculation** – when these orbital elements were determined, in seconds from the epoch
- **eccentricity** – eccentricity of the orbit
- **semi_major_axis** – semi major axis of the orbit in au
- **inclination** – inclination of the NEO's orbit in degrees
- **ascending_node_longitude** – longitude of the ascending node in degrees
- **orbital_period** – orbital period in days
- **perihelion_distance** – perihelion distance in au
- **perihelion_argument** – argument of perihelion in degrees
- **aphelion_distance** – aphelion distance in au
- **perihelion_time** – time of perihelion passage in Barycentric Dynamical Time (TDB)
- **mean_anomaly** – mean anomaly in degrees
- **mean_motion** – mean motion in degrees per day
- **is_potentially_hazardous_asteroid** – returns True or False

B. Data Preprocessing and Feature Engineering

The result from the data extraction phase was a dataset containing 15,355 rows of NEOs, 31 columns of features and 1 label, 'is_potentially_hazardous_asteroid'. Rows with one or more NaN (not a number) values were dropped from the dataset. Additionally, a new column was created, combining the average between 'max_diameter_km' and 'min_diameter_km' to create 'avg_diameter_km'. The max and min diameter columns were dropped. Next, columns that were irrelevant for prediction were also dropped from the data frame. A label encoder function was used to convert the label column to numerical format i.e. 0 or 1. Heatmaps were constructed to visualize which features were most correlated to the label. The strongest correlations to the label were: avg_diamaeter (0.74), observations_used (0.47), data_arc_days (0.36), absolute_magnitude (-0.43), and orbit_uncertainity (-0.32) (see Fig. 2 and 3). Finally, density plots were created for each feature displaying the skewness of data (see Fig. 4). Outliers were not removed for this type of classification problem due to the importance in preserving each NEO and its metadata for accurate hazardous predictions.

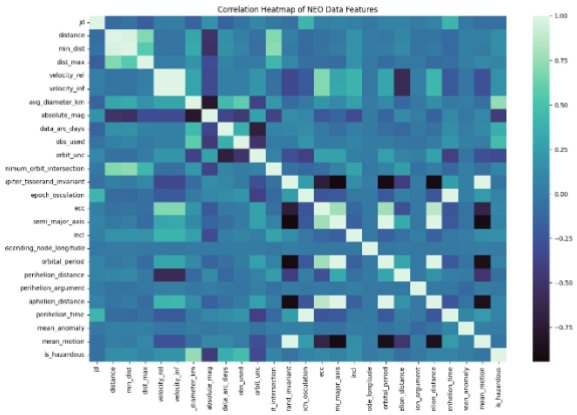

Fig. 2. Correlation heatmap of all NEO data features.

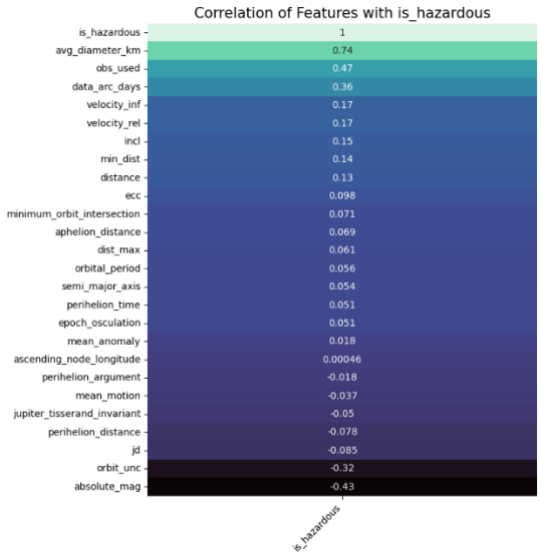

Fig. 3. Correlation of features with label only.

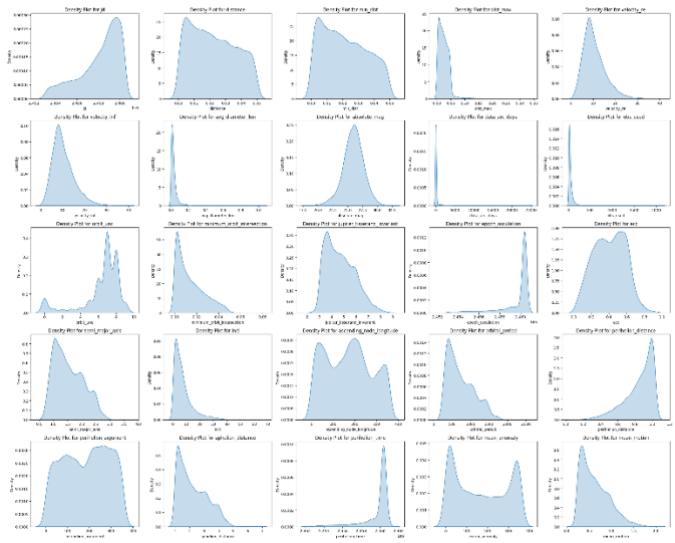

Fig. 4. Density plots of features.

C. *Model Development*

Five models were trained and tested using the Random Forest algorithm with four models using distinct class imbalance techniques and one model with no class imbalance technique, serving as a baseline model. One other technique was discussed but not further evaluated for training or testing. Due to the significant class imbalance and importance of identifying hazardous objects, F2 (F-beta) scores were the most important measures used to evaluate the models. False negatives were more important to classify, as it was crucial to identify all hazardous cases, even at the cost of misclassifying some non-hazardous cases [2]. A Random Forest classifier was initialized with 100 trees and a fixed random state. The make_pipeline() method was used to streamline multiple estimators and transformers. All models used a Stratified K-fold object as a cross-validation parameter to ensure each fold had the same proportion of class labels, maintaining the original class distribution. GridSearchCV was performed with a range of hyperparameters to find the best combination that maximizes the recall score. Once identified, the models were evaluated on the test set. Various performance metrics were calculated including recall, precision, F1 score, F2 score and accuracy. Below are the following discussed class imbalance techniques:

- **Technique 1:** No Under/Oversampling
  - This model was trained and tested without applying any under/over sampling techniques to address class imbalance. This approach served as baseline to evaluate the model's performance on imbalanced data.

- **Technique 2:** Random Oversampling
  - The Random Oversampling method was applied to resample all classes but the majority class. The dataset contained an equal distribution of 11, 545 non-hazardous and hazardous instances, with each class representing 50% of the total dataset.

- **Technique 3:** Random Undersampling
  - The Random Undersampling method was applied to resample all classes but the minority class. The dataset contained an equal distribution of 290 non-hazardous and hazardous instances, with each class representing 50% of the total dataset. This technique was not further evaluated on the training or testing sets due to its low sample size.

- **Technique 4:** SMOTE
  - Synthetic Minority Oversampling Technique (SMOTE) was applied to the minority class to generate synthetic instances.

- **Technique 5:** SMOTE-Tomek Links
    - Combining SMOTE with Tomek Links generates synthetic data for the minority class and removes sample of data from the majority class that is closest or neighbors with the minority class.

- **Technique 6:** Class Weights
    - This Random Forest model is initialized with the 'class_weight' set to 'balance.' This adjusts the weight of each class inversely proportional to their frequencies in the training data, providing more importance to the minority class.

IV. Results, Conclusion, and Future Work

A. Results

The Random Forest with Class Weights achieved the highest F2 score, balancing high recall and good precision (see TABLE I). Precision-recall curves (Fig. 5) and learning curves (Figs. 6-10) were plotted to assess model performance and overfitting. Notably, the model without class imbalance techniques showed significant overfitting (Fig. 6), while the Random Forest with Class Weights showed the least (Fig. 10).

TABLE I
RESULTS OF MODELS DESCENDING BY F2 SCORE

	Random Forest with	Recall	Precision	F1 Score	F2 Score	Accuracy
1	Class weights	1.0	0.922	0.960	0.983	0.999
2	Random oversampling	1.0	0.880	0.937	0.974	0.997
3	SMOTE oversampling	1.0	0.868	0.930	0.970	0.997
4	SMOTE-Tomek Links	1.0	0.855	0.921	0.967	0.997
5	No under/oversampling	0.950	0.966	0.957	0.952	0.998

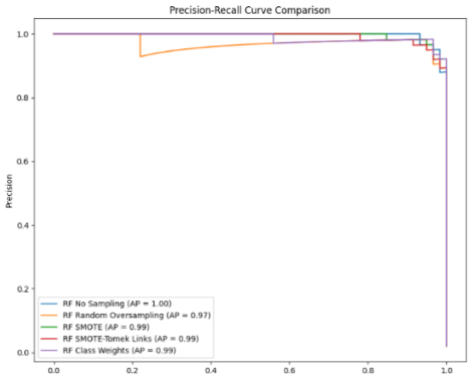

Fig. 5. Precision-recall curves.

Fig. 6. Learning curves for Random Forest with No Under/Oversampling.

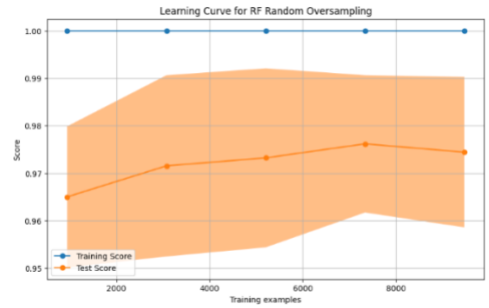

Fig. 7. Learning curves for Random Forest with Random oversampling.

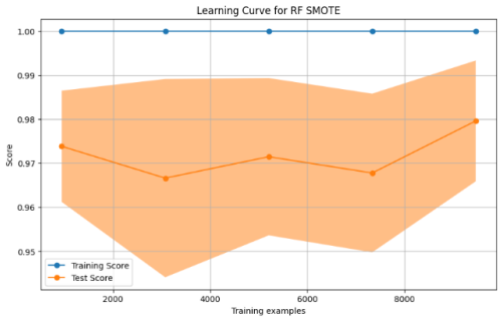

Fig. 8. Learning curves for Random Forest with SMOTE.

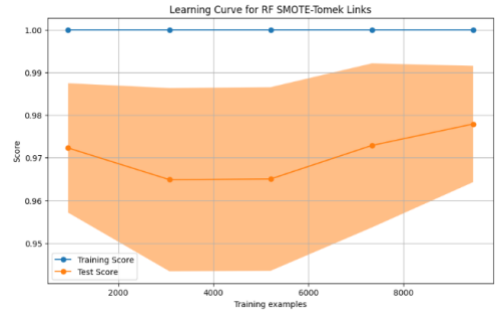

Fig. 9. Learning curves for Random Forest with SMOTE-Tomek Links.

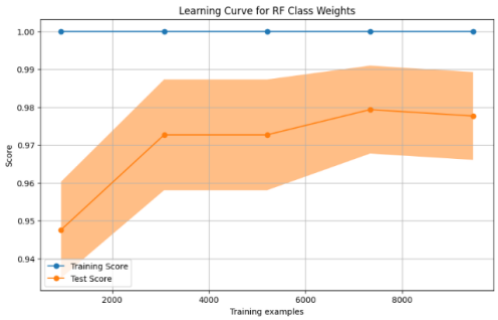

Fig. 10. Learning curves for Random Forest with Class Weights.

B. Conclusion and Future Work

Discrepancies in NASA's NEO classification highlight the need for additional features and possibly computer vision techniques. For example, 3D computer vision can visualize orbit trajectories, aiding in predicting hazardous NEOs (see Fig. 11).

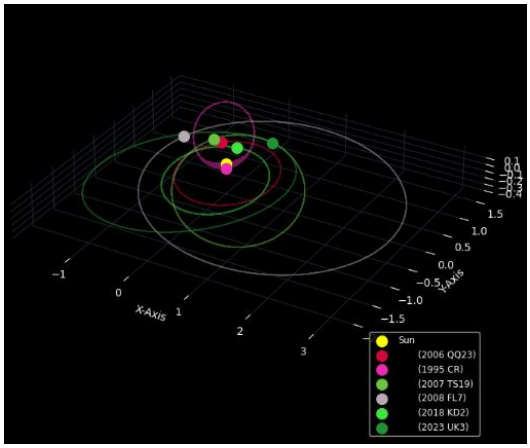

Fig. 11. 3D Visualization of hazardous and non-hazardous NEO orbital paths.

REFERENCES

[1] "NEO Basics," Center for Near Earth Object Studies (CNEOS). https://cneos.jpl.nasa.gov/about/basics.html

[2] J. Brownlee, "Tour of Evaluation Metrics for Imbalanced Classification," Machine Learning Mastery, Jan. 07, 2020. https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/

[3] "Torino Impact Hazard Scale," Center for Near Earth Object Studies (CNEOS). https://cneos.jpl.nasa.gov/sentry/torino_scale.html

[4] S. M. Malakouti, M. B. Menhaj, and A. A. Suratgar, "Machine learning techniques for classifying dangerous asteroids," MethodsX, vol. 11, p. 102337, Dec. 2023, doi: https://doi.org/10.1016/j.mex.2023.102337.

[5] D. Khajuria, A. Sharma, N. Sharma, and M. Mangla, "Classification and Comparative Analysis of Earth's Nearest Objects using Machine Learning Models," in 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom), pp. 16–23.