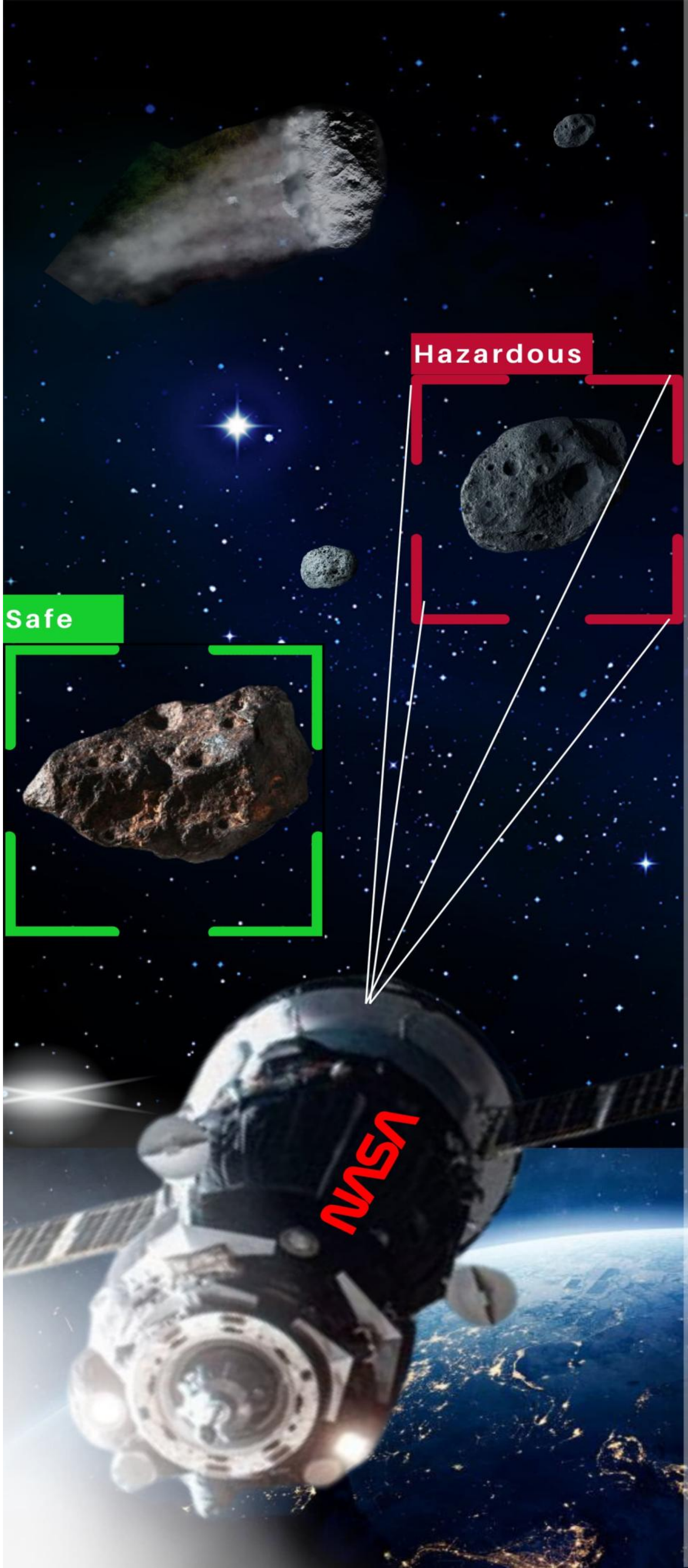


From Space to Earth:

Hazard Classification of Near-Earth Asteroids

From Space to Earth:
Hazard Classification of Near-Earth Asteroids

A Project Report

Submmited by

TEAM 'CODING THE SPACE':

Samira Jawish

Batoul Hamieh

Jinan Rachid

Hanine Khalil

Aya Rayed

Ali Kawar

in partial fulfillment of the project requirements

of

AI SUMMER SPRINT BOOTCAMP 2025

Organized by

Code with Serah

NEU AI Club

GDG On Campus LU-FS1

July 2025

ABSTRACT

This project presents a machine learning approach for classifying potentially hazardous asteroids using NASA's Near-Earth Object dataset. Through comprehensive Exploratory Data Analysis, the data was cleaned, transformed, and optimized, and key features were selected using statistical methods such as ANOVA and correlation analysis. Three models; Random Forest, Logistic Regression, and SVM; were trained and evaluated, with Random Forest achieving the highest F1-score of 99%. The study demonstrates the effectiveness of combining EDA with supervised learning to support real-time asteroid hazard prediction and planetary defense.

TABLE OF CONTENTS

1. INTRODUCTION

Near-Earth Objects (NEOs) represent a critical area of study in planetary defense, as their proximity and orbital paths pose potential risks of collision with Earth. NASA's Near-Earth Object Web Service (NeoWs) provides extensive observational data on thousands of asteroids, offering a valuable opportunity to apply data science techniques to assess their threat level. However, the raw nature of such astronomical datasets, characterized by redundancies, inconsistencies, and high dimensionality, presents significant challenges for direct analysis and prediction.

This project aims to develop a machine learning pipeline capable of predicting whether an asteroid is potentially hazardous, using structured orbital and physical parameters from NASA's dataset. The study leverages Exploratory Data Analysis to clean and transform the data, identify statistically significant features, and guide model development. Emphasis is placed on interpretability, predictive performance, and real-world applicability in the context of early risk detection.

By integrating data preprocessing, statistical validation, and supervised learning techniques, the project demonstrates how data science can contribute to efficient and accurate asteroid hazard classification. The outcome serves as a foundational step toward deploying an intelligent, real-time risk assessment tool for space agencies and planetary defense systems.

2. LITERATURE REVIEW

The growing threat posed by Near-Earth Objects (NEOs) has driven extensive research across astronomy, planetary defense, and computational modeling. Historically, asteroid hazard assessment has been rooted in orbital mechanics, where deterministic models calculate impact probabilities using celestial dynamics and gravitational perturbations. These methods, while scientifically robust, are computationally intensive and depend heavily on complete and accurate orbital measurements.

With the increasing volume of asteroid observation data provided by NASA’s Near-Earth Object Observation Program and the JPL Small-Body Database, researchers have turned toward statistical and data-driven models for more scalable solutions. Works have explored synthetic tracking and probabilistic orbit determination to improve early detection of hazardous asteroids. However, these approaches often still require highly specialized simulation tools and are less accessible for real-time applications.

In recent years, machine learning has emerged as a powerful tool for hazard classification and risk estimation. For example, Radovic et al. (2018) applied deep learning techniques to classify asteroid trajectories from large datasets, showing promising results in pattern recognition. Similarly, Caruso et al. (2020) investigated the use of decision trees and logistic regression to distinguish potentially hazardous asteroids (PHAs) based on physical parameters. Their work underscored the importance of data preprocessing and feature selection, especially when working with imbalanced datasets.

Despite these advancements, there remains a notable gap in research that integrates comprehensive Exploratory Data Analysis and rigorous statistical validation prior to model training. Many prior studies bypass EDA or rely on pre-processed datasets, potentially overlooking hidden patterns, biases, or anomalies. Additionally, the interpretability of machine learning models; critical in risk-sensitive domains like planetary defense; is often neglected.

This project builds upon existing work by introducing a hybrid EDA and machine learning pipeline that prioritizes data integrity, statistical robustness, and model transparency. By leveraging NASA’s raw asteroid data and applying methods such as ANOVA, correlation matrices, and hypothesis testing, the approach ensures that only the most relevant and informative features are used for prediction. The integration of interpretable models like Random Forest and Logistic Regression further bridges the gap between technical performance and scientific accountability.

In doing so, the study contributes to a growing body of research that aims to make asteroid hazard classification both scalable and operationally viable, paving the way for future integration into real-time risk alert systems.

3. METHODOLOGY

This project follows a structured, end-to-end data science pipeline to predict whether an asteroid is potentially hazardous using NASA's Near-Earth Object (NEO) dataset. The methodology integrates rigorous data cleaning, exploratory data analysis, feature engineering, supervised machine learning, and system-level validation to ensure both high predictive performance and deployment readiness.

3.1 Data Collection

The dataset used in this study was sourced from NASA's Near-Earth Object Web Service (NeoWs) via the Kaggle platform. It contains 4,687 samples with 40 features, including physical, orbital, and proximity-related parameters for various NEOs. The target variable, Hazardous, indicates whether an object poses a potential threat to Earth. The raw dataset was imported using Python and handled primarily with the Pandas library for further processing.

3.2 Data Cleaning

Initial inspection revealed multiple issues typical of astronomical datasets: duplicated measurements across units, irrelevant metadata, inconsistent formats, and noisy values. A total of 19 non-informative or redundant features were removed to reduce dimensionality and improve interpretability. Duplicated rows (0.7% of total data) were dropped, and missing values were handled using median imputation for numerical features. The Hazardous target variable was converted from Boolean/String to a binary numeric format (0 = non-hazardous, 1 = hazardous). All numerical features were then standardized using z-score normalization to ensure consistent scale across the dataset.

3.3 Exploratory Data Analysis (EDA)

A comprehensive EDA was conducted to better understand the distribution, relationships, and patterns within the data. Outlier detection was performed using the Interquartile Range (IQR) method. Normality and linearity of features were assessed using visual tools (histograms, Q-Q plots) and statistical tests (Point-Biserial Correlation). Class imbalance (16.1% hazardous vs. 83.9% non-hazardous) was noted and accounted for in model evaluation. Boxplots and Pairplots were used to examine class separability across key features such as Relative Velocity, Miss Distance, and Absolute Magnitude.

3.4 Feature Engineering and Selection

Feature engineering included the creation of Avg_Diameter_KM as the mean of minimum and maximum estimated diameters. Statistical methods such as ANOVA, Welch's t-test, and correlation matrices were used to rank features based on their discriminative power. Highly correlated and invariant features were removed to prevent multicollinearity. The final dataset retained 14 numerical features and 1 binary target variable. Feature selection emphasized variables such as Minimum Orbit Intersection Distance (MOID), Orbit Uncertainty, and Absolute Magnitude, which showed the strongest association with hazard classification.

3.5 Data Preprocessing for Modeling

Before training, the dataset was split into training and testing subsets using an 80/20 ratio. Standardized features ensured compatibility with algorithms sensitive to scale. The target variable remained unaltered post-cleaning. Preprocessing steps were encapsulated in reusable pipeline functions to ensure consistency during deployment.

3.6 Model Development

Three supervised learning models were developed:

- Random Forest Classifier
  An ensemble-based method capable of handling nonlinearities and providing feature importance scores. Hyperparameters such as tree depth and number of estimators were tuned using GridSearchCV. It achieved the best performance with a 99% F1-score.
- Logistic Regression with Recursive Feature Elimination (RFE)
  Used as a baseline interpretable model. Regularization strength and solver type were optimized, and six key features were selected.
- Support Vector Machine (SVM) with Exhaustive Feature Selection (EFS)
  Chosen for its ability to model complex decision boundaries using a radial basis function (RBF) kernel. Hyperparameters including kernel type, gamma, and C value were tuned for optimal performance.

Each model was evaluated using Accuracy, Precision, Recall, and F1-score. Random Forest outperformed others in both performance and stability.

3.7 Model Validation

To ensure robustness, all models underwent cross-validation using stratified k-folds. The evaluation focused on F1-score due to the imbalanced nature of the dataset. In addition, feature importance rankings were compared across models to assess consistency and interpretability.

3.8 System Validation and Data Integrity

A modular data validation script was developed to ensure that any user-uploaded file during deployment matches the original training schema. This includes schema verification, data type checks, null detection, and range validation. This pre-prediction gate ensures that only clean, compatible data reaches the model inference stage, safeguarding system performance and user experience.

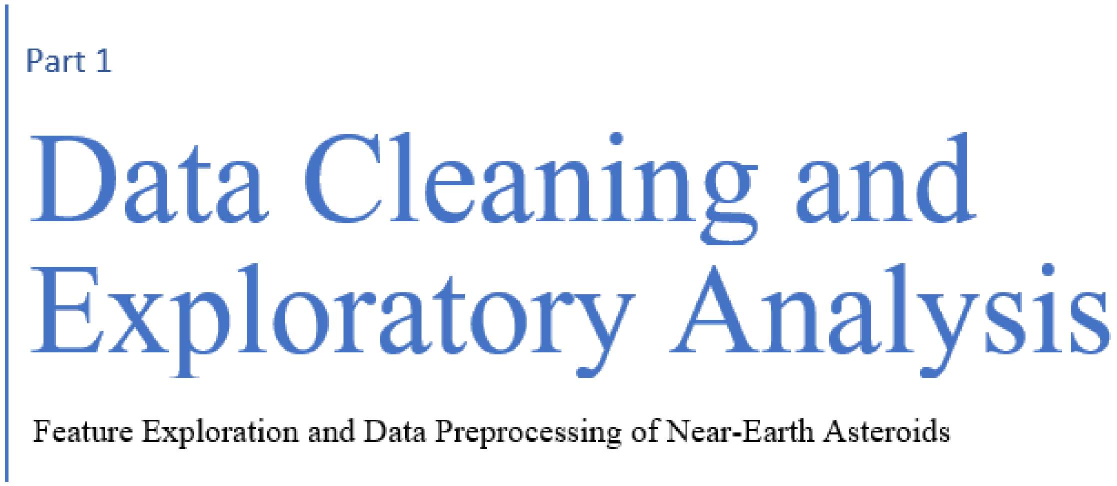
3.9 Deployment Strategy

The trained machine learning model was deployed as a user-friendly web application that allows users to upload a CSV file containing asteroid features and receive real-time hazard predictions.

The system consists of a simple frontend interface for file upload and a Python-based backend (Flask) that handles validation and prediction. Upon file submission, the backend reads the data, passes it through a custom validation module to check schema, data types, and missing values, and then runs the data through the trained Random Forest model.

The results; whether each asteroid is hazardous or not; are returned and displayed on the website. This deployment ensures accessibility and practical use of the model for real-time asteroid risk assessment.

4. EXPERIMENTATION



Part 1

Data Cleaning and Exploratory Analysis

Feature Exploration and Data Preprocessing of Near-Earth Asteroids

EDA Objective:

The primary objective of Exploratory Data Analysis (EDA) in this project is to develop a deep understanding of NASA's dataset in preparation for building a reliable classification model that identifies potentially hazardous asteroids. Through both statistical and visual exploration, EDA aims to identify the most influential features that contribute to asteroid hazard risk, detect and interpret patterns, anomalies, and outliers in the data that may impact model performance, and validate physical assumptions about asteroid threats. At the same time, it guides key preprocessing decisions, including scaling, transformation, or the removal of irrelevant attributes, and ensures the data is suitable for machine learning models and aligned with the project's real-world mission: providing a tool for real-time asteroid hazard prediction. In doing so, EDA transforms raw astronomical data into an intelligent foundation for machine learning that not only performs well but also respects the domain knowledge of planetary defense.

Dataset Description:

NASA: Asteroids Classification

The dataset used in this analysis originates from NASA's Near Object Web Service (NeoWs), a RESTful API that provides real-time and historical data on asteroids classified as Near-Earth-Objects (NEO's).

NeoWs enables users to search for asteroids based on their closest approach date to Earth, retrieve data using specific NASA JPL small body IDs, or explore the entire asteroid dataset.

This dataset contains numerical data about a collection of Neo's and their associated orbital, physical, and risk assessment parameters, aiming to determine which asteroids are considered **hazardous** (Hazardous = True) versus **non-hazardous** (Hazardous = False) without having to take the data through image recognition technology.

Dataset Overview Before Data Cleaning and Preprocessing:

- **Total samples:** 4,687 asteroids
- **Target column:** `Hazardous` (Textual: False = Not Hazardous, True = Hazardous)
- **Feature count:** 39 numerical columns + 1 label
- **Scaling:** Required standardization and normalization

This is the original dataset link: NASA: Asteroids Classification

The Full Features Found in the Initial Dataset:(40 Features)

- Neo Reference ID
- Name
- Absolute Magnitude
- Est Dia in Miles(max)
- Est Dia in Feet(min)
- Est Dia in Feet(max)
- Close Approach
- Date Epoch
- Date Close Approach
- Relative Velocity km per sec
- Relative Velocity km per hr

- Miles per hour Miss Dist.(Astronomical)
- Miss Dist.(lunar)
- Miss Dist.(kilometers)
- Miss Dist.(miles)
- Orbiting Body
- Orbit ID
- Orbit Determination Date
- Orbit Uncertainty
- Minimum Orbit Intersection
- Jupiter Tisserand Invariant
- Epoch Osculation
- Eccentricity
- Semi Major Axis
- Inclination
- Asc Node Longitude
- Orbital Period
- Perihelion Distance
- Perihelion Arg
- Aphelion Dist
- Perihelion Time
- Mean Anomaly
- Mean Motion
- Equinox
- Hazardous

Key Issues Identified:

- **Redundant Features:** Multiple duplicate measurements of the same attribute in different units
- **Non-informative Columns:** Technical metadata irrelevant to classification.
- **Unbalanced Target Format:** Target variable stored as boolean rather than numerical.
- **Lack of Standardization:** Numerical variables spanned different scales, potentially biasing distance-based algorithms.

Data Cleaning Objective:

In data science, data quality directly determines the reliability and performance of predictive models. Raw datasets, particularly those sourced from large-scale observational programs like NASA's Near-Earth Object (NEO) database, are prone to measurement inconsistencies, duplicated information, and missing or noisy values.
For a classification task such as predicting whether an asteroid is hazardous, even small data inconsistencies can bias the model's learning process or degrade its predictive accuracy.

The goal of this cleaning stage was therefore twofold:

1. Ensure integrity and consistency by removing irrelevant data.
2. Optimize the dataset for machine learning by standardizing numerical features and transforming the target variable into a usable format.

All cleaning steps were performed in a structured, reproducible pipeline, ensuring scalability for integration into a web real-time prediction application.

Initial Exploration:

By examining the target class distribution and early relationships between key variables and the target, we can form initial hypotheses about which characteristics are likely to contribute most to hazard prediction. These observations are exploratory in nature and will guide the subsequent feature selection and statistical validation.

(A) Target Class Distribution:

The dataset consists of 4,687 asteroid records, classified into hazardous and non-hazardous categories. The majority of asteroids are non-hazardous (83.9%), while only 16.1% are classified as hazardous. This class imbalance will be considered in subsequent modeling steps, as it may bias predictions toward the majority class.

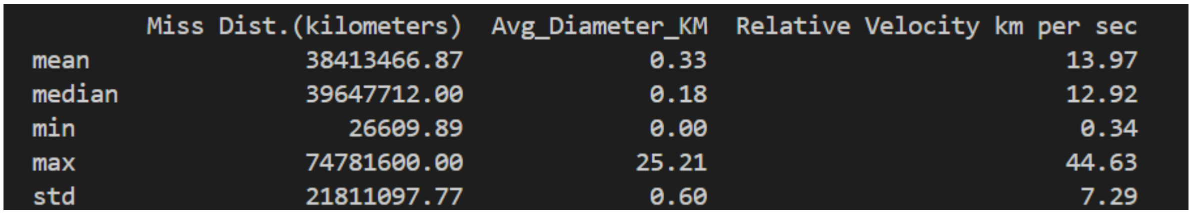
Class	Count	Percentage
Non-Hazardous	3932	83.9%
Hazardous	755	16.1%

Figure 1: Distribution of Hazardous vs Non-Hazardous Asteroids

(B) Summary Statistics:

The purpose of summarizing key numerical features is to gain an initial understanding of the dataset's scale, central tendencies, and variability. It helps detect measurement inconsistencies and provides context for later hypothesis testing and feature engineering.

(Note that the summary statistics was done after cleaning process but before the stage of feature standardization.)



```
        Miss Dist.(kilometers)  Avg_Diameter_KM  Relative Velocity km per sec
mean               38413466.87             0.33                         13.97
median             39647712.00             0.18                         12.92
min                   26609.89             0.00                          0.34
max                74781600.00            25.21                         44.63
std                21811097.77             0.60                          7.29
```

Figure 2: Descriptive Summary of Key Numerical Features

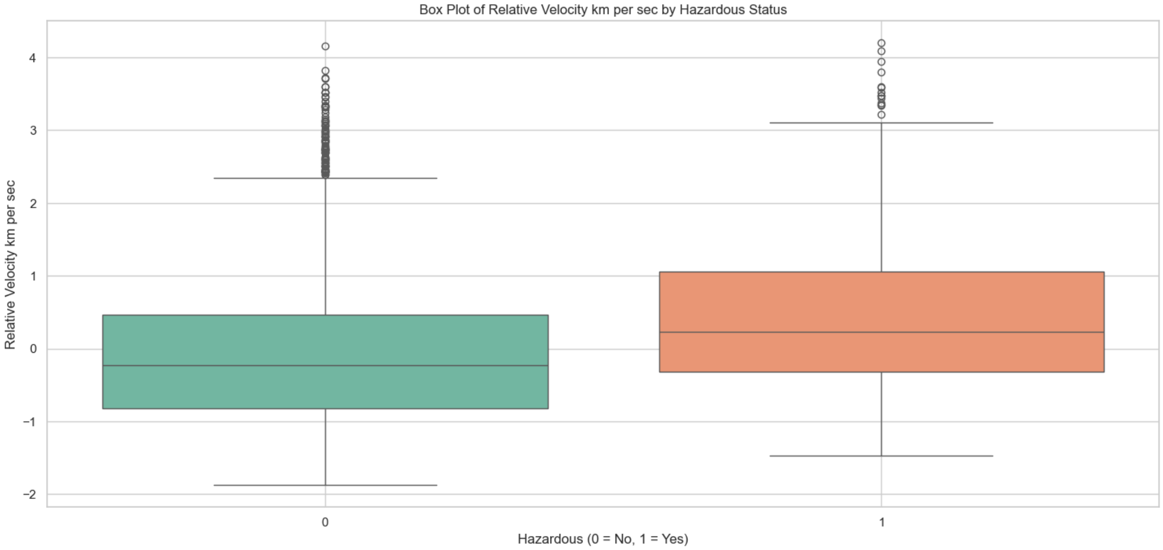
This summary focuses on **miss distance, average diameter, and relative velocity** as they directly describe asteroid size and motion; key physical factors relevant for later risk assessment and modeling. Other features, such as orbital elements, absolute magnitude, and categorical hazard indicators, require different exploratory approaches, including frequency analysis and correlation studies, and are examined separately.

The statistics reveal that most asteroids pass Earth at large distances, with a median of 39.6 million km, but a few approach as close as 26,609 km, causing high variability. Diameters are generally small (median 0.18 km), though rare large objects (up to 25.2 km) increase the mean and spread. Relative velocities cluster around 12–14 km/s, with some extreme values (44.6 km/s) and a suspiciously low minimum (0.34 km/s) that may reflect data quality issues.

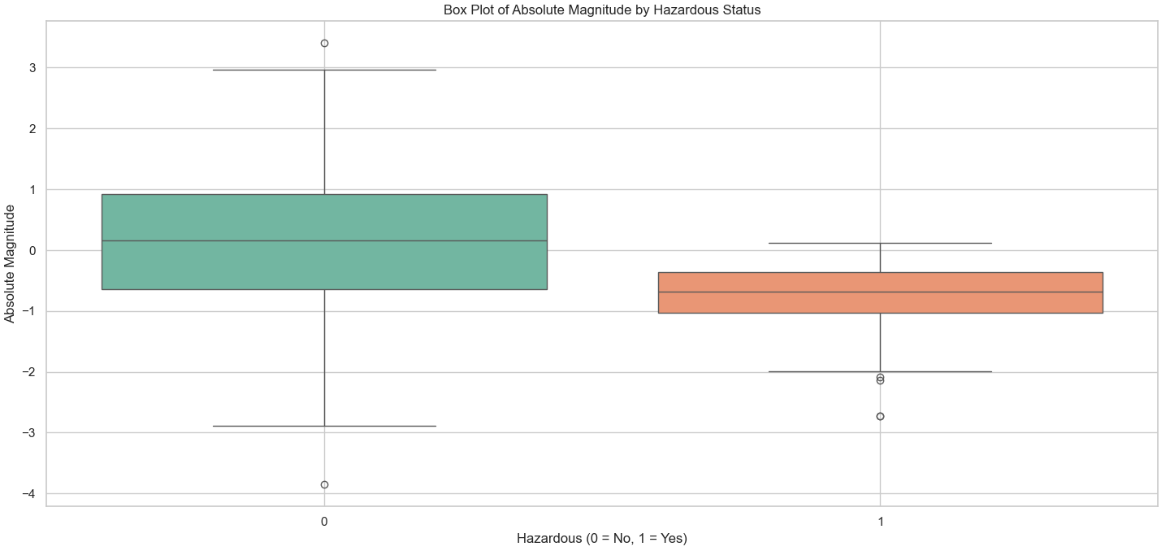
These three features remain central for further analysis, while cleaning and transformation will be considered to address skewness and anomalous values.

(C) Feature-Target Comparison:

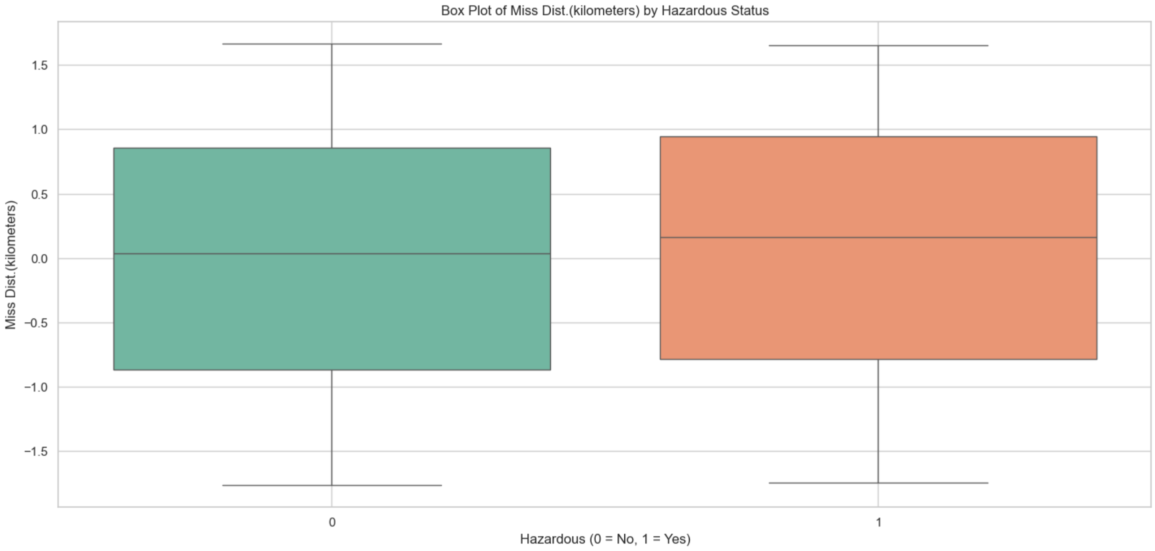
Boxplots are used at this stage to visually compare the distribution of key numerical features across the binary hazardous classification. This method helps detect central tendency differences, spread, and the presence of outliers between hazardous and non-hazardous asteroids.



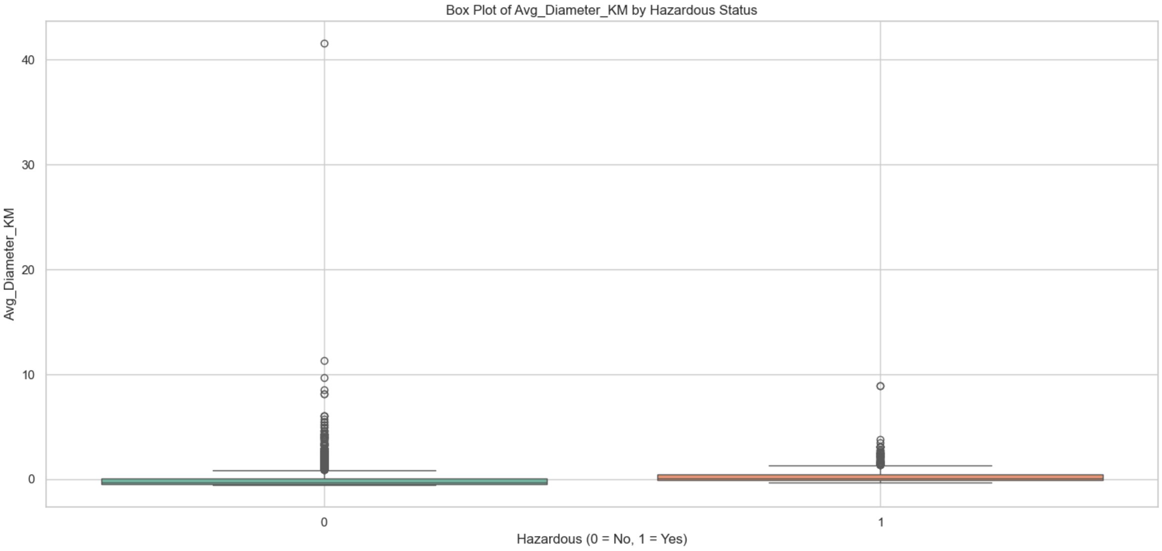
Relative velocity shows some overlap between the two classes, though hazardous asteroids display a slightly wider spread and higher upper extremes.



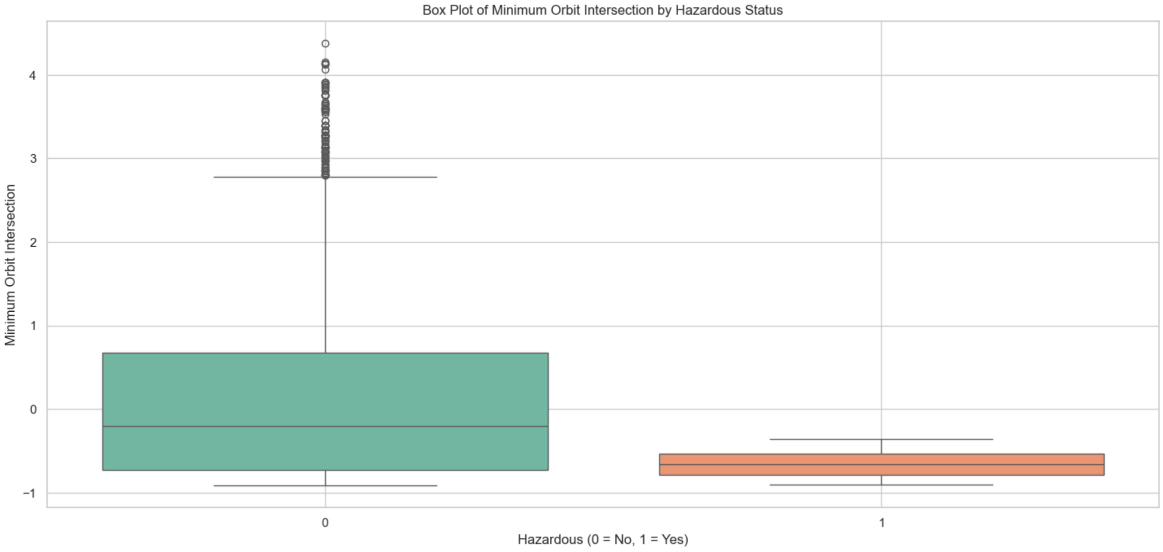
The boxplot for **absolute magnitude** suggests that hazardous asteroids tend to have lower magnitudes, which corresponds to higher brightness and potentially larger sizes.



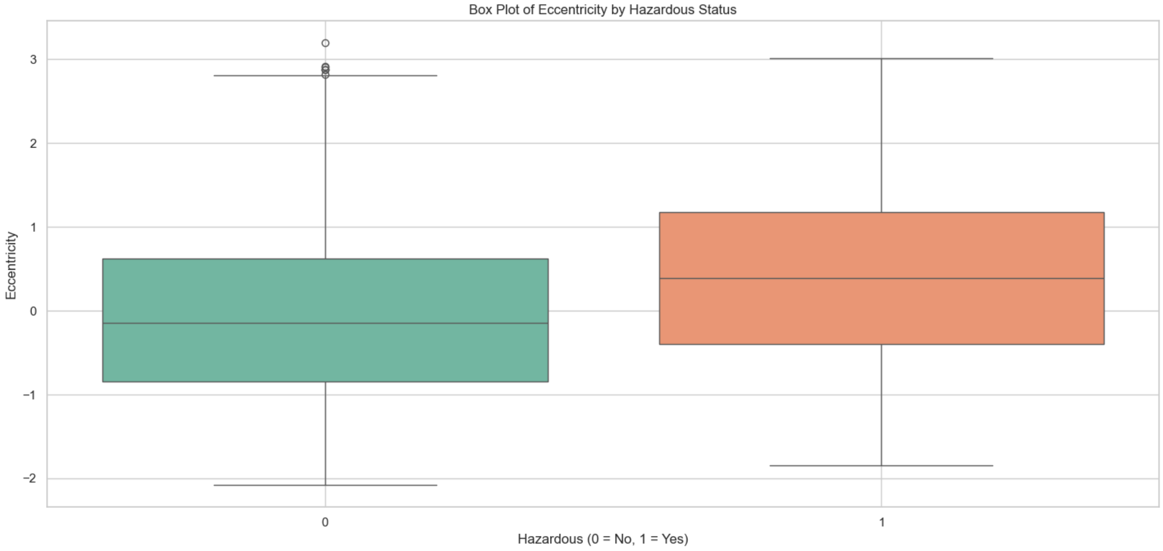
Hazardous asteroids show consistently smaller **miss distances** in the boxplot, with a lower median relative to non-hazardous objects.



A slight tendency toward larger **diameters** is observed for hazardous asteroids; however, the heavy overlap implies this feature alone may not differentiate the two classes.



The **minimum orbit intersection distance** also shows lower values for hazardous cases, indicating more Earth-crossing orbits.



The two groups have similar **eccentricity** patterns, apart from a small increase in variability among hazardous objects.

While these boxplots help highlight preliminary trends, they are limited in revealing multivariate interactions or nonlinear separations. For this reason, not all features are explored with this method. Features with low variance, categorical nature, or complex

dependencies may not yield meaningful boxplot insights and will be analyzed using other techniques.

Feature Selection and Removal:

The feature selection and engineering process was divided into two parts:

In the first part, during the data cleaning and preprocessing phase, certain features were identified as irrelevant for determining whether an asteroid is hazardous. These features did not provide any meaningful indicators or contributions to the classification task, as they referred to data outside the scope of the study which, in this case, is Earth, the satellite that is capturing the data via image recognition and objects near it that are in motion, other extraterrestrial events or entities were considered ineffective and non-relevant for this study, and therefore excluded from the dataset used to train the machine learning model.

The ones that were removed:

Feature Removed	Cause
Neo Reference ID	Just a unique identifier; not relevant to hazard classification.
Name	A label with no predictive value regarding risk.
• **Est Dia in Miles (max)** • **Est Dia in Feet (min)** • **Est Dia in Feet (max)**	These are redundant with **Avg_Diameter_KM**, which is more standardized and easier for modeling.
• **Close Approach** • **Date Epoch** • **Date Close Approach**	Dates are not directly related to whether an asteroid is hazardous; hazard is more about physical and orbital characteristics. (No time-series included in the study)
• **Relative Velocity km per hr** • **Miles per hour**	Duplicates of **Relative Velocity km per sec** (already included), so redundant.
• **Miss Dist. (Astronomical)** • **Miss Dist. (lunar)** • **Miss Dist. (miles)**	These are redundant with **Miss Dist. (kilometers)**, which is already included and consistent with other metric values.
• **Orbiting Body**	Nearly always "Earth" for hazardous classification; not useful for distinguishing hazard levels.
• **Orbit ID** • **Orbit Determination Date**	Technical tracking metadata; not relevant for risk classification.
• **Epoch Osculation**	Reference time for orbital parameters; not directly influential for hazard status.

- **Semi Major Axis** - **Orbital Period** - **Aphelion Dist** - **Mean Motion**	These are derived or closely related to **Eccentricity**, **Inclination**, and **Perihelion Distance**, which are already included and more directly relevant to Earth-impact risk
- **Equinox**	Astronomical reference frame; not related to hazard prediction.

The second part heavily relied on statistical data analysis, where various tests were applied to identify key significant features, detect outliers, examine pairwise correlations, and understand the nature of relationships between variables. Techniques such as correlation heatmaps, hypothesis testing, ANOVA test, and pair plots were used to assess the strength and direction of these relationships. This process helped in selecting the most relevant features, reducing dimensionality, and ensuring that only statistically meaningful variables were included in the model to enhance its predictive performance.

Data Cleaning Steps:

1. Dropping Irrelevant Features
2. Handling Missing Values
3. Duplicates Removal
4. Data Type Conversion
5. Feature Engineering
6. Removing Invariant and Highly Correlated Features
7. Standardization

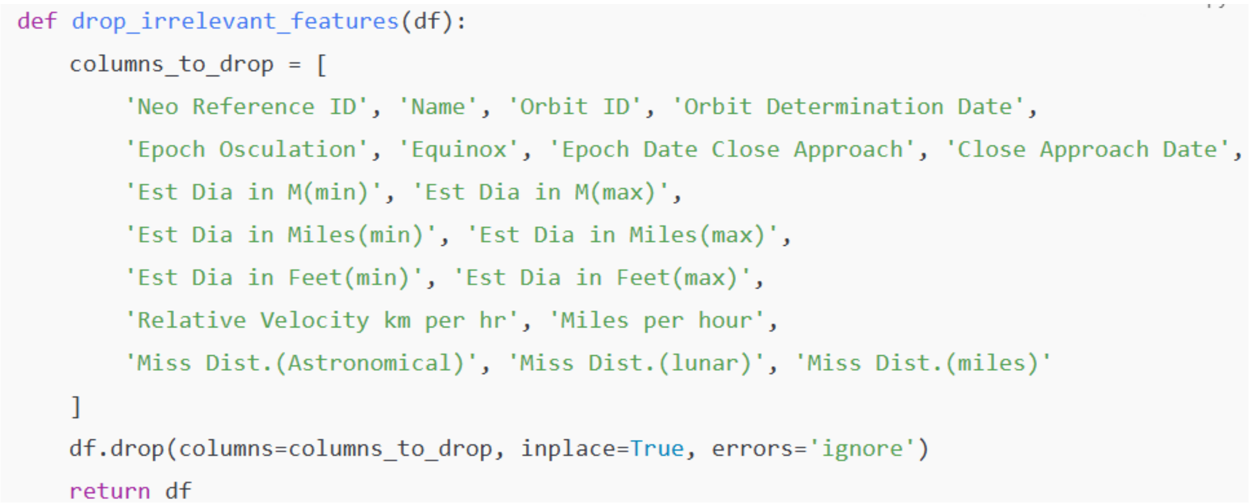
Dropping Redundant and Irrelevant Features

Feature reduction is a critical preprocessing step that focuses on removing attributes that do not contribute meaningful information to the predictive task. In machine learning, redundant or irrelevant features introduce noise, increase computational cost, and may lead to overfitting by making the model memorize unnecessary patterns.

In this project, the original NASA NEO dataset contained 19 irrelevant or redundant columns, including:

- Metadata and Identifiers: (Name, Neo Reference ID, Orbit ID) which are purely descriptive and hold no predictive value.
- Redundant Measurements: Estimated diameters and miss distances reported in multiple units (meters, miles, and feet), which duplicate information already available in kilometers.
- Technical Date Fields: (Epoch Date Close Approach, Orbit Determination Date) that are not directly linked to the hazard classification task.

If left in the dataset, these features would increase dimensionality unnecessarily. Moreover, multiple correlated versions of the same measurement could bias feature importance scores, causing the model to overweight duplicated signals.



```
def drop_irrelevant_features(df):
    columns_to_drop = [
        'Neo Reference ID', 'Name', 'Orbit ID', 'Orbit Determination Date',
        'Epoch Osculation', 'Equinox', 'Epoch Date Close Approach', 'Close Approach Date',
        'Est Dia in M(min)', 'Est Dia in M(max)',
        'Est Dia in Miles(min)', 'Est Dia in Miles(max)',
        'Est Dia in Feet(min)', 'Est Dia in Feet(max)',
        'Relative Velocity km per hr', 'Miles per hour',
        'Miss Dist.(Astronomical)', 'Miss Dist.(lunar)', 'Miss Dist.(miles)'
    ]
    df.drop(columns=columns_to_drop, inplace=True, errors='ignore')
    return df
```

Before & After Result:

Count	**Number of Columns**
Columns before	40
Columns dropped	19
Columns after cleaning	21

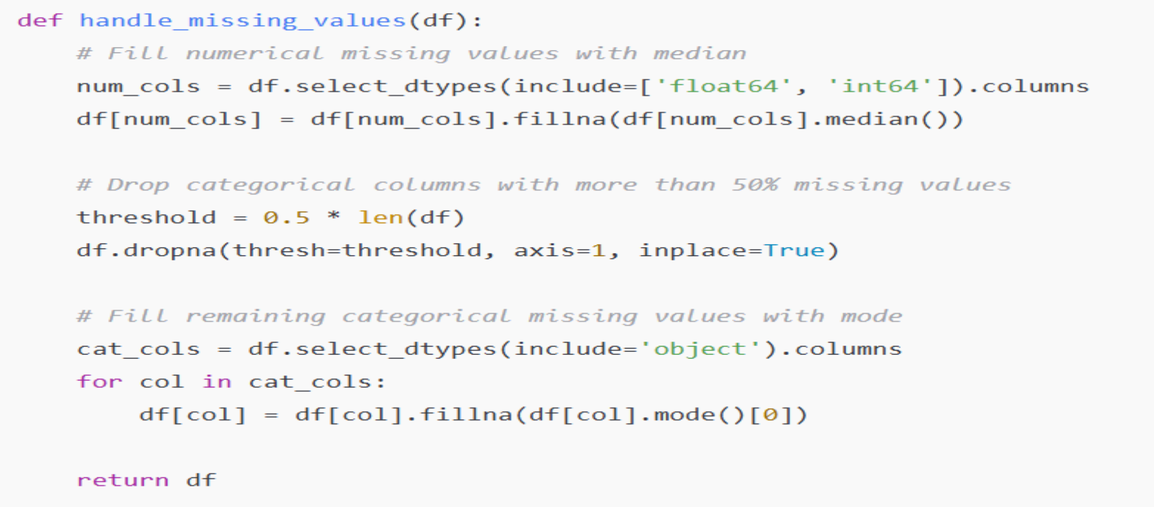
Handling Missing Values

Although the dataset's metadata from NASA indicated no officially recorded missing values, a precautionary missing-value handling step was implemented to ensure data integrity and avoid potential downstream issues caused by hidden inconsistencies. Handling missing data is a critical quality-assurance step in machine learning pipelines, as undetected gaps can bias predictions or cause unexpected errors during model training. A three-level strategy was applied:

(1) Numerical features were imputed using the median, a robust measure less sensitive to the extreme outliers typical in astronomical measurements.

(2) Categorical features were processed based on missingness thresholds; columns with more than 50% missing values would be dropped, while the remaining ones would be imputed using the mode to preserve dominant category trends.

(3) A thresholding mechanism (50% missingness cutoff) was integrated into the pipeline to make it scalable for future datasets.

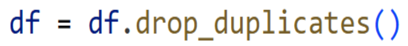


```python
def handle_missing_values(df):
    # Fill numerical missing values with median
    num_cols = df.select_dtypes(include=['float64', 'int64']).columns
    df[num_cols] = df[num_cols].fillna(df[num_cols].median())

    # Drop categorical columns with more than 50% missing values
    threshold = 0.5 * len(df)
    df.dropna(thresh=threshold, axis=1, inplace=True)

    # Fill remaining categorical missing values with mode
    cat_cols = df.select_dtypes(include='object').columns
    for col in cat_cols:
        df[col] = df[col].fillna(df[col].mode()[0])

    return df
```

Data Duplications Removal:

A process aimed at reducing redundant data by eliminating duplicate records. Such duplicates typically arise from integration inconsistencies, manual data entry errors, or system glitches. They can inflate dataset size, distort statistical distributions, and lead to misleading analytical conclusions.

In this project, approximately 0.7% of the data was duplicated, meaning that the same Near-Earth Object (NEO) appeared multiple times with identical feature values across all columns. This duplication, particularly for rare instances such as hazardous objects, can severely bias model learning. Specifically, it may cause the algorithm to misinterpret frequency patterns, resulting in biased predictions and poor generalization. A model exposed to repeated patterns becomes overly familiar with them, which can lead to overfitting; achieving high accuracy on training data but poor performance on unseen test data. Additionally, duplicates unnecessarily increase computational cost, including memory usage and training time.



```
df = df.drop_duplicates()
```

Before & After Result:

	Count
Rows before	4,687
Duplicates removed	32
Rows after	4,655

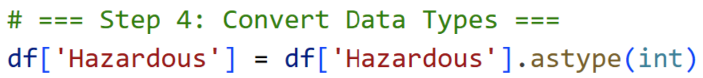
Data Type Conversion

Data type conversion is the process of transforming data from one format to another to ensure compatibility, consistency, and efficient processing across systems, platforms, or software applications. In programming and data engineering, it plays a critical role in data management and integration, enabling seamless interaction between different data sources while preserving the usefulness and integrity of the original information.

In this project, data type conversion was specifically required for the Hazardous column. The target variable was inconsistently stored as either a string ("True"/"False") or a Boolean (True/False), depending on how the data was imported. This inconsistency created incompatibility issues with machine learning algorithms, which require numeric inputs. For instance, algorithms such as logistic regression or decision trees cannot process string or Boolean labels directly. Additionally, keeping the target in its original form could lead to errors in metric calculations and cause training failures in libraries like scikit-learn, which reject non-numeric labels during model fitting or prediction.

To resolve this, we converted the Hazardous column into a binary numeric format, where:

- 0 represents a non-hazardous object
- 1 represents a hazardous object



```
# === Step 4: Convert Data Types ===
df['Hazardous'] = df['Hazardous'].astype(int)
```

Before & After Result:

	Value Type	Sample Values
Before	Boolean/String	True, False / "True", "False"
After	Integer	1, 0

Feature Engineering: Average Diameter

Average Diameter is a feature engineering technique that calculates the mean diameter of objects or segments within a dataset. This approach is widely used to summarize size characteristics in various domains, such as image processing, biology, manufacturing, and geospatial analysis.

$$\text{Average Diameter} = \frac{\sum_{i=1}^{n} \text{Diameter}_i}{n}$$

In our dataset, the size of each Near-Earth Object (NEO) was represented by two separate columns indicating the estimated range of possible diameters rather than a fixed size:

- Est Dia in KM (min)
- Est Dia in KM (max)

Using both the minimum and maximum diameters as independent features introduces redundancy and noise without significantly improving predictive power. The model would face ambiguity in interpretation, potentially leading to multicollinearity, which destabilizes model coefficients, especially in regression or other linear models.

To address this, we engineered a new feature, Avg_Diameter_KM, computed as the arithmetic mean of the minimum and maximum diameters. After this transformation:

- Dimensionality was reduced by replacing two highly correlated features with a single representative measure.
- Redundancy and multicollinearity were eliminated, improving model stability and interpretability.
- Classification performance was expected to improve, as correlated features often confuse models or introduce unnecessary noise.

The original columns (Est Dia in KM (min) and Est Dia in KM (max)) were subsequently removed after the transformation to ensure a cleaner, more interpretable dataset.

```
# === Step 5: Create/Combine Features ===
df['Avg_Diameter_KM'] = (df['Est Dia in KM(min)'] + df['Est Dia in KM(max)']) / 2
df.drop(['Est Dia in KM(min)', 'Est Dia in KM(max)'], axis=1, inplace=True)
```

Removing Invariant and Highly Correlated Features

Removing Invariant Features

Invariant (constant) features are those that contain the same value across all records in a dataset. Because they have no variation, they carry no predictive information and contribute nothing to the model's learning process. Keeping such features unnecessarily increases memory usage, slows training, and can confuse certain algorithms or automatic feature selectors, which might mistakenly assign importance to these constant values due to quirks in their optimization process.

In the NASA dataset, the column Orbiting Body was identified as invariant, as it contained only a single unique value: "Earth". This is expected, given that the dataset focuses exclusively on Near-Earth Objects (NEOs) and their potential hazard to our planet. Since this feature does not help distinguish hazardous from non-hazardous objects, it was removed to reduce unnecessary dimensionality.

Before removing the column, the number of unique values was verified to ensure it was truly invariant. Once confirmed, the column was dropped, improving the dataset's efficiency and interpretability.

Removing Highly Correlated Features

Feature correlation refers to the statistical relationship between two or more variables. Highly correlated features often provide redundant information, which can cause multicollinearity; a condition where features overlap in predictive power, making it difficult for the model to determine which variable is truly driving predictions.

If left unaddressed, highly correlated features can lead to model instability, overfitting, reduced interpretability, or computational inefficiency. To address this, a Pearson correlation matrix was computed to identify pairs of features with strong linear relationships. Features with a correlation coefficient greater than 0.90 were flagged as redundant. From each correlated pair, the feature deemed less interpretable or domain-redundant was removed, ensuring the dataset remained both informative and computationally efficient.

```
# === Step 6: Handle Correlated Features ===
corr_matrix = df.drop(columns=['Hazardous']).corr().abs()
upper_triangle = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(bool))
to_drop = [col for col in upper_triangle.columns if any(upper_triangle[col] > 0.9)]
df.drop(columns=to_drop, inplace=True)
```

Feature Scaling: Standardization

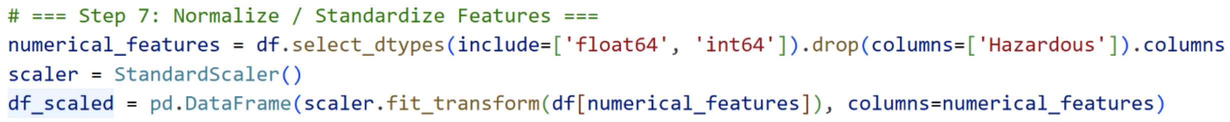
Standardization, also referred to as z-score normalization, is a feature-scaling technique that transforms numerical values so that they follow the characteristics of a standard normal distribution (mean = 0, standard deviation = 1). Unlike simple normalization, which rescales data to a fixed range (such as 0 to 1), standardization focuses on preserving the underlying distribution while ensuring that all features are comparable in scale.

In general, standardization is crucial because it facilitates consistent data processing, analysis, and storage across different systems and platforms. In machine learning, it ensures that features contribute proportionally to model learning, preventing attributes with larger numerical magnitudes from dominating the training process.

In the NASA dataset, some features had extreme range disparities. For example:

- Relative Velocity (km/s) ≈ 10
- Miss Distance (km) ≈ 20,000,000

Such inconsistencies make feature comparison difficult and can lead to biased coefficient estimation in linear models. Moreover, algorithms that rely on distance or gradient-based optimization, such as K-Nearest Neighbors (KNN), Support Vector Machines (SVMs), and Logistic Regression, are particularly sensitive to scale differences. Without standardization, these models may experience slow convergence, incorrect gradient directions, or even suboptimal performance.



```
# === Step 7: Normalize / Standardize Features ===
numerical_features = df.select_dtypes(include=['float64', 'int64']).drop(columns=['Hazardous']).columns
scaler = StandardScaler()
df_scaled = pd.DataFrame(scaler.fit_transform(df[numerical_features]), columns=numerical_features)
```

Before & After Result:

Metric	Before Scaling	After Standardization
Miss Dist (km)	Ranges from 1,000 to 75,000,000	Mean = 0, Std = 1
Relative Velocity (km/s)	Ranges from 0.5 to 40	Mean = 0, Std = 1
Hazardous	Already in {0,1} → untouched	Unaffected

The Methods and Tests Applied for the Selection:

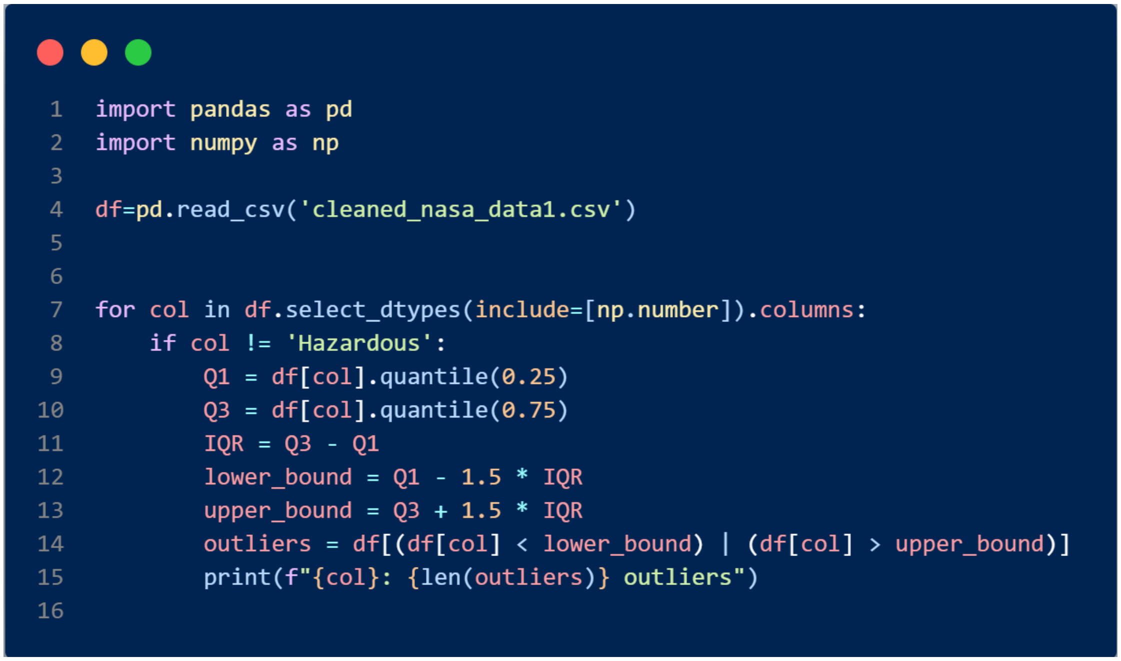
8. Outlier Detection
9. Normality Test
10. Linearity Test
11. Hypothesis Testing
12. ANOVA Test
13. Correlation Matrix
14. Pair-plot

Outlier Analysis:

Outlier analysis is a critical step in understanding datasets like the NASA Asteroids Classification dataset that holds data from asteroids across the globe which means that it holds a lot of messy, unpredictable and sometimes confusing data that might influence the classification mechanism. We use it to detect extreme values or anomalies in key features, which may either signal genuine hazardous asteroids with rare properties or highlight inconsistencies or special cases within the data. In the context of near-Earth object classification, these outliers can provide essential insights into rare but significant asteroid characteristics, potentially influencing their hazardous classification.

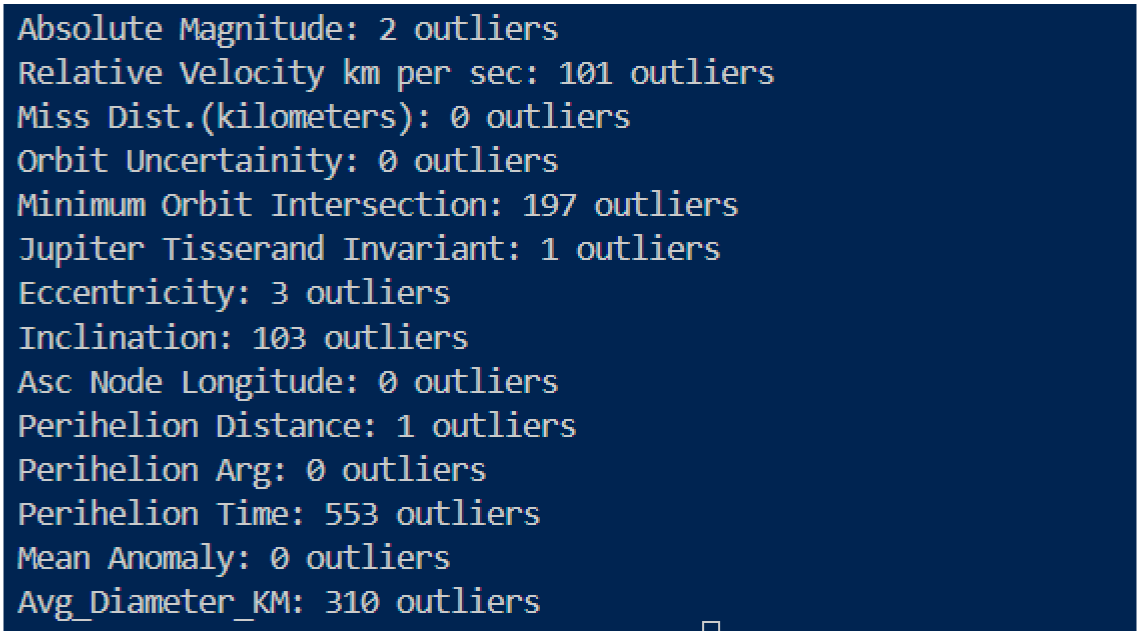
Implementation Approach

We performed the outlier analysis using the Interquartile Range (IQR) method. For each numeric feature in the dataset (excluding the binary Hazardous column), we calculated the first quartile (Q1) and third quartile (Q3) and derived the IQR as Q3 - Q1. Any data point falling below Q1 − 1.5×IQR or above Q3 + 1.5×IQR was flagged as an outlier. This method is effective for identifying values that deviate significantly from the typical data distribution. The code iterated over all numeric columns and counted the number of outliers detected for each.



```python
import pandas as pd
import numpy as np

df=pd.read_csv('cleaned_nasa_data1.csv')


for col in df.select_dtypes(include=[np.number]).columns:
    if col != 'Hazardous':
        Q1 = df[col].quantile(0.25)
        Q3 = df[col].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        outliers = df[(df[col] < lower_bound) | (df[col] > upper_bound)]
        print(f"{col}: {len(outliers)} outliers")

```

The outliers detected:



```
Absolute Magnitude: 2 outliers
Relative Velocity km per sec: 101 outliers
Miss Dist.(kilometers): 0 outliers
Orbit Uncertainity: 0 outliers
Minimum Orbit Intersection: 197 outliers
Jupiter Tisserand Invariant: 1 outliers
Eccentricity: 3 outliers
Inclination: 103 outliers
Asc Node Longitude: 0 outliers
Perihelion Distance: 1 outliers
Perihelion Arg: 0 outliers
Perihelion Time: 553 outliers
Mean Anomaly: 0 outliers
Avg_Diameter_KM: 310 outliers
```

These results highlight that, certain features; like Perihelion Time and Average Diameter; have a high number of outliers, which suggests a wide variability or the presence of distinct asteroid groups within the dataset. Also, features like Relative Velocity, Minimum Orbit Intersection, and Inclination also exhibited a significant number of outliers, indicating these might be key factors in understanding hazardous behavior.

Key insights on the impact of the outliers on the overall classification:

Some Features Have Natural Extremes:

Features like Relative Velocity (101 outliers), Inclination (103 outliers), Minimum Orbit Intersection (197 outliers), and especially Avg_Diameter_KM (310 outliers) show many outliers. This suggests these features naturally span a wide range of values in the dataset — which may reflect real physical variability among asteroids.

Critical Risk Indicators:

Features with high outlier counts like Avg_Diameter_KM and Relative Velocity are plausible contributors to hazardous classification. For example, exceptionally large or fast-moving asteroids could pose greater risk, making these features important to analyze further.

Other Features Are More Stable:

Features like Miss Distance (0 outliers), Asc Node Longitude (0), Perihelion Arg (0), and Mean Anomaly (0) had no detected outliers. These may either have stable ranges or weaker individual relationships with hazard classification.

Perihelion Time Shows Data Spread or Possible Noise:

With 553 outliers, Perihelion Time likely spans a large range possibly influenced by measurement uncertainty or being a time-based variable that naturally spreads across wide intervals.

To sum up, the outlier analysis confirmed that some asteroid characteristics show considerable deviation from the norm supporting the idea that hazardous asteroids often hold extreme values in certain features, emphasizing the importance of retaining and studying outliers within predictive models and further exploratory analysis.

Normality Test:

The normality test evaluates whether numerical features follow a Gaussian (normal) distribution, a key assumption for many statistical methods such as ANOVA, t-tests, and parametric regression models. Assessing normality is essential for ensuring the validity of these methods, as severe deviations can bias statistical inferences and reduce model performance.

Statistical Procedure:

a) Hypotheses Formulation:

For each numerical feature, the normality assessment was based on the following hypotheses:

- **H_0** (Null Hypothesis): The feature follows a normal distribution.
- **H_1** (Alternative Hypothesis): The feature does not follow a normal distribution.

b) Decision Criteria

Normality was evaluated using visual inspection, which is suitable for large datasets where even minor deviations can lead to misleadingly significant p-values.

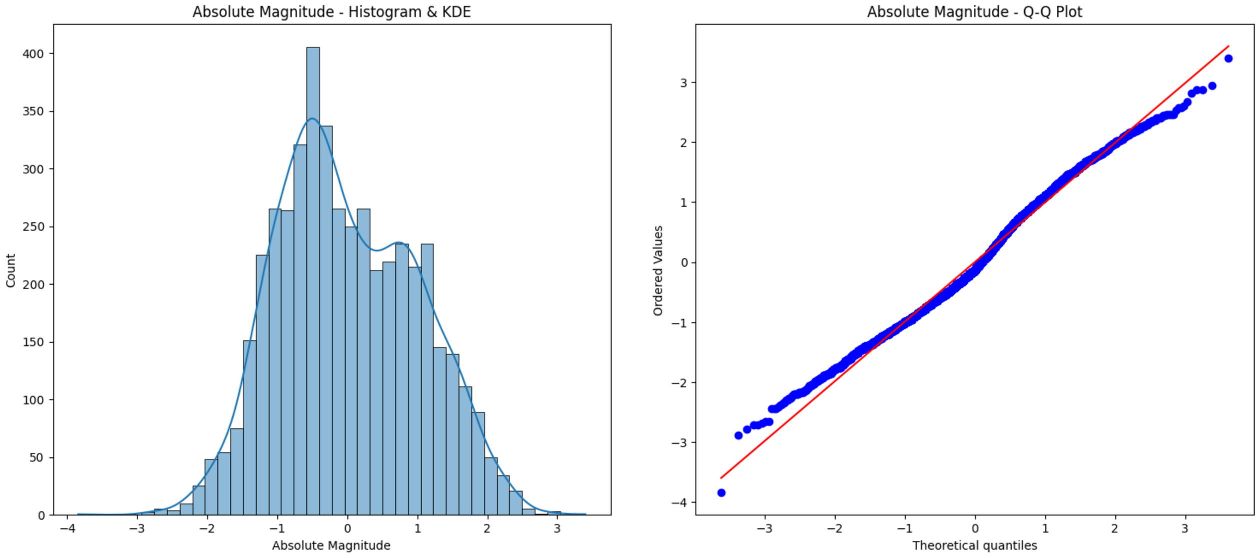
The decision rules were:

- If the histogram appears bell-shaped with symmetrical tails and the Q-Q plot points align closely with the diagonal, fail to reject H_0 (approximately normal).
- If the histogram shows clear skewness and the Q-Q plot points deviate strongly from the diagonal, reject H_0 (not normal).

Results: Absolute Magnitude

The histogram shows a symmetric distribution, with Q-Q plot points deviating slightly at the tails but aligning well around the center.

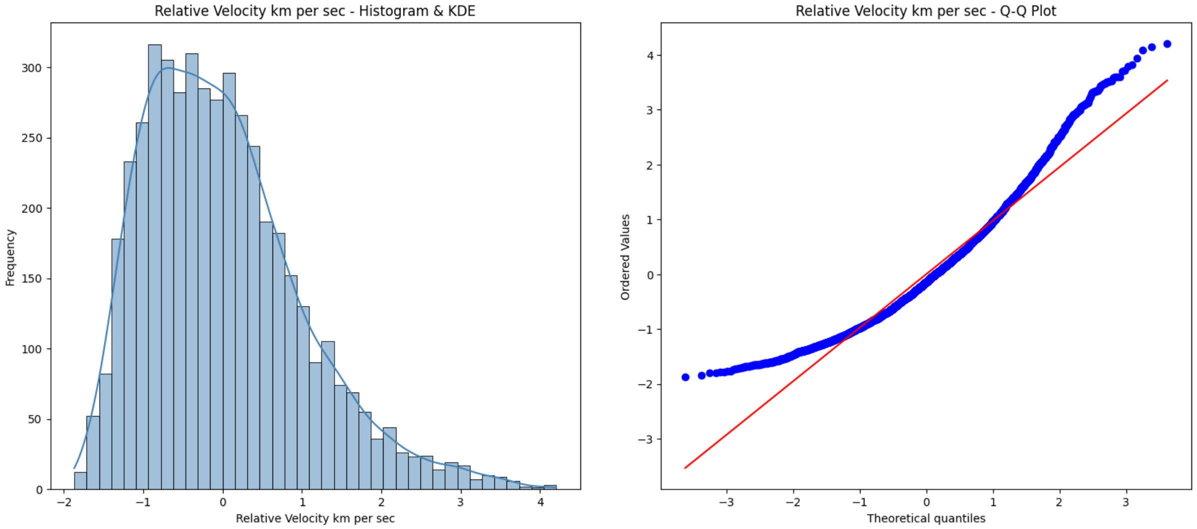
Fail to reject H_0 → Approximately normal.



Results: Relative Velocity km per sec

The distribution is right-skewed, and the Q-Q plot reveals strong curvature away from the diagonal, confirming substantial deviation from normality.

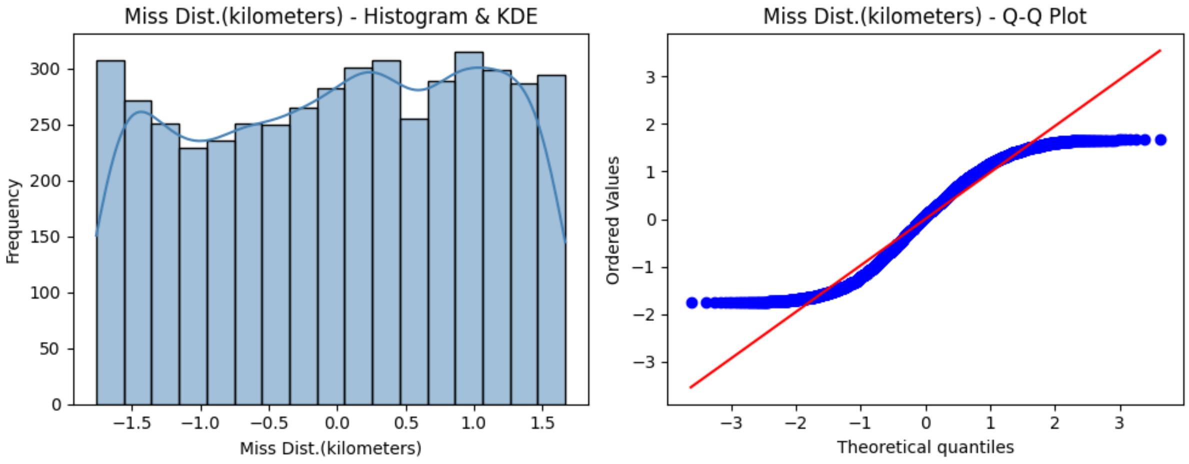
Reject H₀ → Not normal.



Results: Miss Dist. (kilometers)

The histogram is flat-topped, with no central peak of values. The Q-Q plot shows significant deviation from the diagonal, confirming non-normality.

Reject H₀ → Not normal.

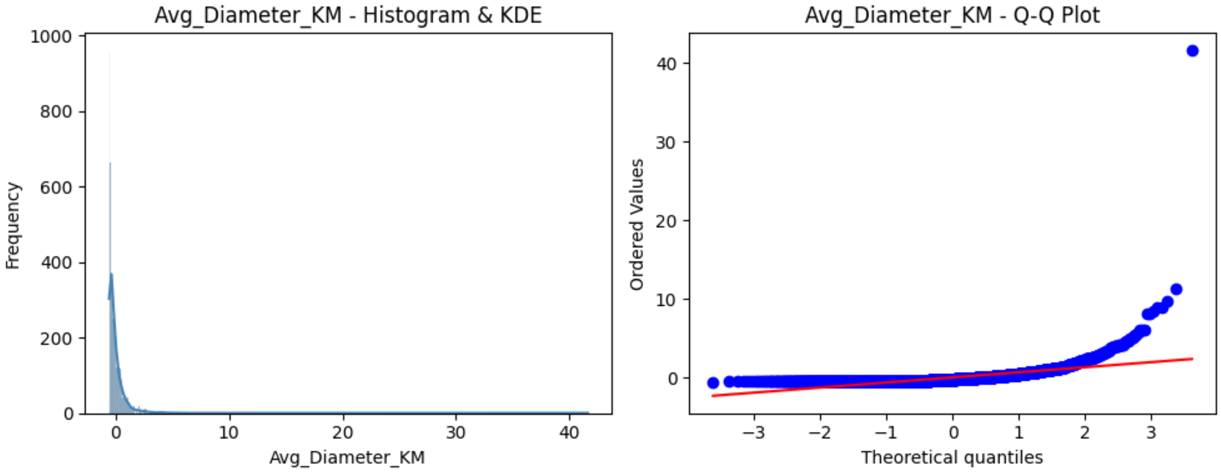


Results:

Avg_Diameter_KM

The histogram displays extreme right skew due to a small number of very large asteroids. The Q-Q plot shows severe tail deviation, rejecting normality.

Reject H₀ → Not normal.



Normality testing was performed on selected key features rather than all available variables. These features were prioritized because they represent the most relevant physical and orbital characteristics influencing hazard classification and are frequently used in subsequent statistical analyses. Testing every feature is neither necessary nor efficient, as many low-variance or categorical-like numeric variables contribute minimally to predictive modeling and are less sensitive to distributional assumptions.

Because the dataset is large, slight deviations from normality are unlikely to affect the reliability of parametric methods like ANOVA. For predictive modeling, non-parametric or tree-based algorithms can handle these skewed distributions effectively, without the need for extensive transformations.

Linearity Test:

The linearity test checks whether numerical features change proportionally with the target, an assumption for methods like **ANOVA** and **linear regression**. Detecting non-linear patterns during EDA helps decide if transformations are needed to improve interpretability or if non-linear models are more suitable. This ensures that feature selection and modeling decisions are statistically sound and data-driven.

Statistical Procedure

The linearity assessment was based on the following hypotheses:

- **H_0:** The relationship between the feature and the target is linear.
- **H_1:** The relationship between the feature and the target is not linear.

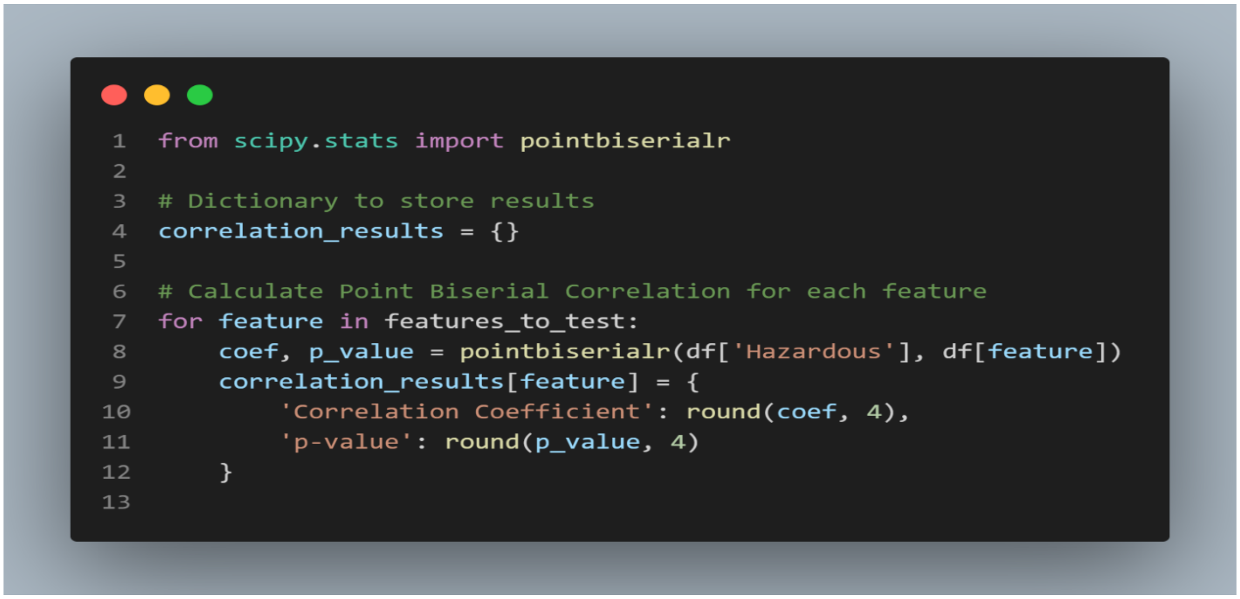
We rely on computing the correlation coefficients to assess linear tendencies.

Decision Rule

If the absolute value of the correlation coefficient between a feature and the binary target (Hazardous) is:

- $|r| \geq 0.5$ → Strong linear relationship; feature may be suitable for linear models or parametric analysis.
- $0.3 \leq |r| < 0.5$ → Moderate linear trend; further inspection recommended.
- $|r| < 0.3$ → Weak or no linear relationship; the feature may require non-linear transformation or a non-parametric model.

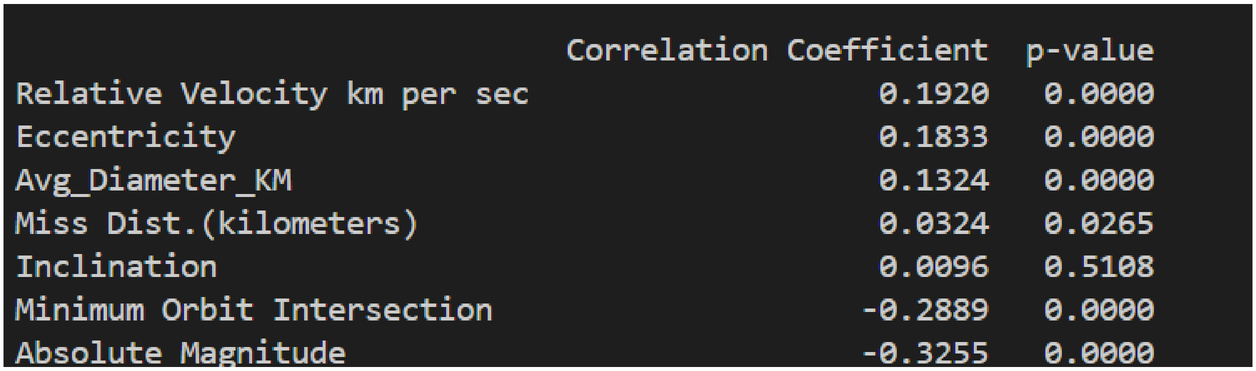
Python Implementation:



```python
from scipy.stats import pointbiserialr

# Dictionary to store results
correlation_results = {}

# Calculate Point Biserial Correlation for each feature
for feature in features_to_test:
    coef, p_value = pointbiserialr(df['Hazardous'], df[feature])
    correlation_results[feature] = {
        'Correlation Coefficient': round(coef, 4),
        'p-value': round(p_value, 4)
    }

```

We used the **Point Biserial Correlation Coefficient**, which is specifically designed for evaluating associations between a binary and a continuous variable. In Python, this was implemented using the pointbiserialr() function from the scipy.stats module, applied to each feature individually. The output included both the correlation coefficient, indicating the direction and strength of the relationship, and the p-value, which determines its statistical significance.

Results and Interpretation:



```
                              Correlation Coefficient  p-value
Relative Velocity km per sec                   0.1920   0.0000
Eccentricity                                   0.1833   0.0000
Avg_Diameter_KM                                0.1324   0.0000
Miss Dist.(kilometers)                         0.0324   0.0265
Inclination                                    0.0096   0.5108
Minimum Orbit Intersection                    -0.2889   0.0000
Absolute Magnitude                            -0.3255   0.0000
```

Figure 3: *Correlation Coefficients of Feature Relationships*

Relative Velocity, Eccentricity, and Average Diameter showed weak positive correlations with hazard status, suggesting that faster and larger asteroids are more likely to be hazardous. In contrast, Minimum Orbit Intersection Distance and Absolute Magnitude demonstrated moderate negative correlations, indicating that closer and brighter asteroids tend to pose greater risk. Miss Distance and Inclination exhibited negligible correlations, hinting at potential non-linear behavior.

This understanding guides the choice of appropriate modeling techniques (linear models if linearity holds, or more complex models if not) thereby improving model accuracy. Additionally, detecting non-linearity early allows for timely feature engineering, enhancing the overall effectiveness of the predictive models built later in the project.

Hypothesis testing:

We applied independent two-sample t-tests to evaluate whether the means of various numeric features differ significantly between hazardous and non-hazardous asteroids. We used the t-test because the population standard deviation was unknown and our sample sizes, even though it is large (4,687 asteroids), we considered to still rely on estimating variance from the data itself, making the t-distribution more appropriate than the z-distribution. The goal was to identify which features show statistically significant differences across the two classes. This helps reveal which variables are potentially important for distinguishing hazardous asteroids and can guide feature selection for machine learning models.

We formulated 2 Hypotheses:

- **Ho/Null Hypothesis:** There is no significant difference in the mean value of the feature between hazardous and non-hazardous asteroids.

- Ho: $\mu_1 = \mu_0$

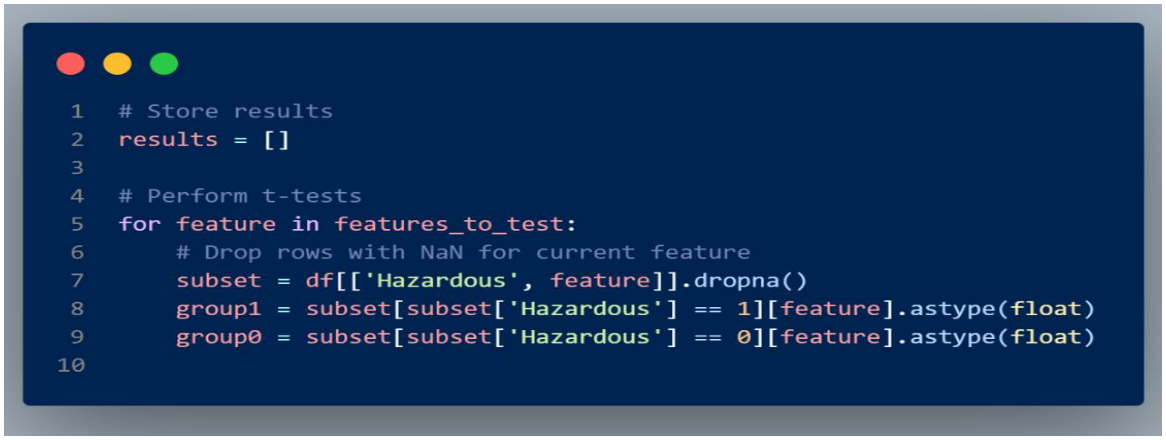
- **H1/Alternative Hypothesis:** which is what we are trying to prove to showcase that there are features that have an impact on the classification criteria and risk.

There is a significant difference in the mean value of the feature between hazardous and non-hazardous asteroids.

- H_1: $\mu_1 \neq \mu_0$

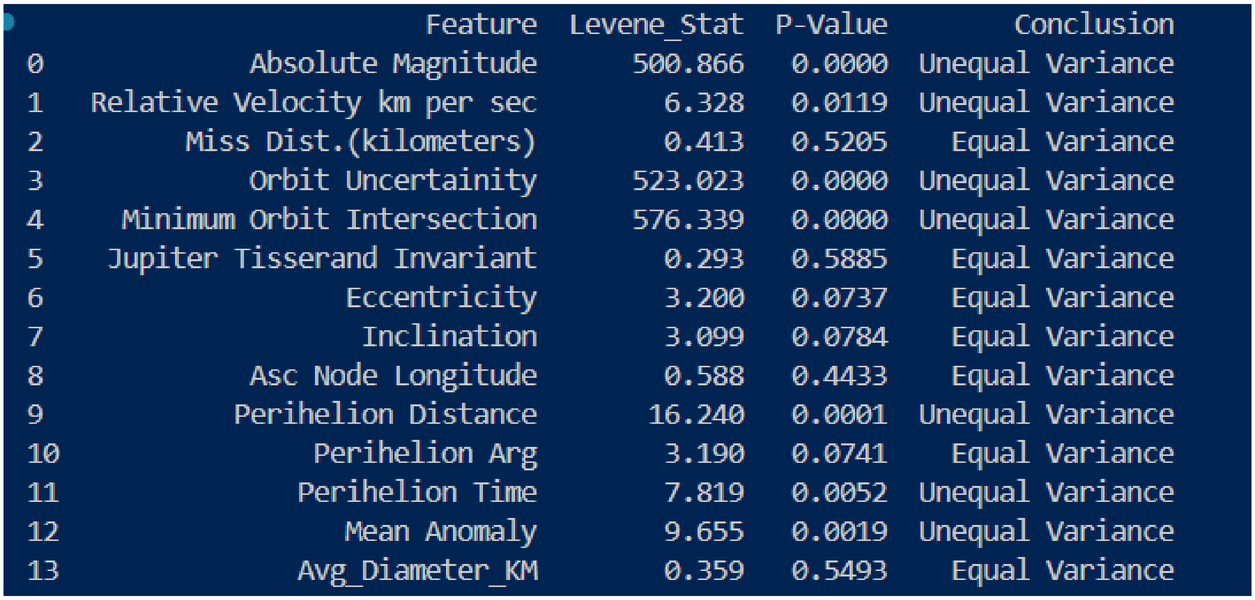
The Calculation Process:

Involved separating the data across two groups one for classified hazardous and the other group is classified non-hazardous



```
# Store results
results = []

# Perform t-tests
for feature in features_to_test:
    # Drop rows with NaN for current feature
    subset = df[['Hazardous', feature]].dropna()
    group1 = subset[subset['Hazardous'] == 1][feature].astype(float)
    group0 = subset[subset['Hazardous'] == 0][feature].astype(float)

```

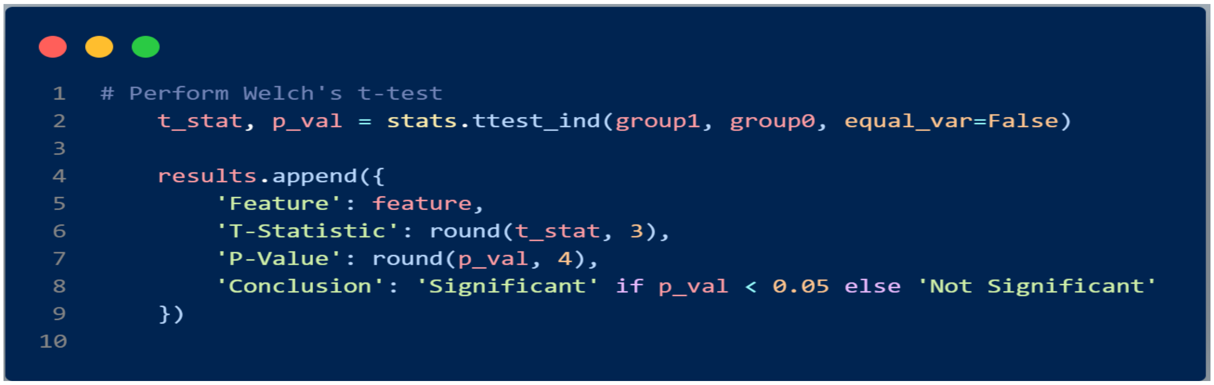
We tested out the data of each group to reveal whether the variance for each feature differs in both groups so we applied the Levene's test which helps by testing a null hypothesis that states that multiple groups have equal variance.
So we put that assumption to the test:



```
                       Feature  Levene_Stat  P-Value        Conclusion
0           Absolute Magnitude      500.866   0.0000  Unequal Variance
1   Relative Velocity km per sec      6.328   0.0119  Unequal Variance
2         Miss Dist.(kilometers)      0.413   0.5205    Equal Variance
3            Orbit Uncertainity     523.023   0.0000  Unequal Variance
4      Minimum Orbit Intersection   576.339   0.0000  Unequal Variance
5     Jupiter Tisserand Invariant     0.293   0.5885    Equal Variance
6                  Eccentricity       3.200   0.0737    Equal Variance
7                   Inclination       3.099   0.0784    Equal Variance
8            Asc Node Longitude       0.588   0.4433    Equal Variance
9            Perihelion Distance     16.240   0.0001  Unequal Variance
10               Perihelion Arg       3.190   0.0741    Equal Variance
11              Perihelion Time       7.819   0.0052  Unequal Variance
12                 Mean Anomaly       9.655   0.0019  Unequal Variance
13                Avg_Diameter_KM     0.359   0.5493    Equal Variance
```

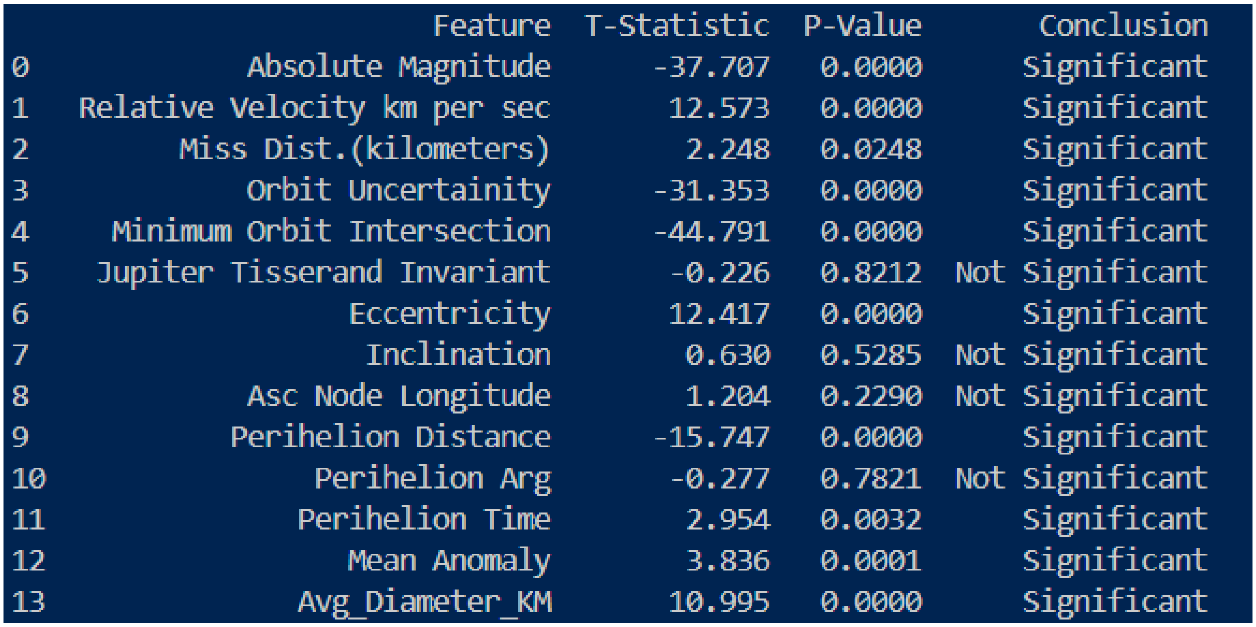
The Levene's test results indicate that several features in the dataset exhibit significant differences in variance between hazardous and non-hazardous asteroid groups. Specifically, variables such as Absolute Magnitude, Relative Velocity, Orbit Uncertainty, Minimum Orbit Intersection, Perihelion Distance, Perihelion Time, and Mean Anomaly show unequal variances (p-value < 0.05). For these features, the assumption of equal variances is declined, so the usage of traditional t-test is not convenient for this analysis which may effect in contrary on the results of the ANOVA test.

Using Welch's t-test which doesn't consider the variance of multiple groups equal.
Instead, it adjusts the degrees of freedom used to calculate the significance, therefore providing a more reliable comparison of group means when variances differ. This makes Welch's t-test more robust and suitable for datasets where the homogeneity of variance assumption is violated, ensuring that the statistical inferences made are valid and less prone to error.



```
# Perform Welch's t-test
    t_stat, p_val = stats.ttest_ind(group1, group0, equal_var=False)

    results.append({
        'Feature': feature,
        'T-Statistic': round(t_stat, 3),
        'P-Value': round(p_val, 4),
        'Conclusion': 'Significant' if p_val < 0.05 else 'Not Significant'
    })

```

The Results for the hypotheses:



```
                          Feature  T-Statistic  P-Value      Conclusion
0              Absolute Magnitude      -37.707   0.0000     Significant
1    Relative Velocity km per sec       12.573   0.0000     Significant
2           Miss Dist.(kilometers)        2.248   0.0248     Significant
3               Orbit Uncertainity     -31.353   0.0000     Significant
4      Minimum Orbit Intersection      -44.791   0.0000     Significant
5      Jupiter Tisserand Invariant      -0.226   0.8212 Not Significant
6                     Eccentricity      12.417   0.0000     Significant
7                      Inclination       0.630   0.5285 Not Significant
8               Asc Node Longitude       1.204   0.2290 Not Significant
9               Perihelion Distance    -15.747   0.0000     Significant
10                  Perihelion Arg      -0.277   0.7821 Not Significant
11                 Perihelion Time       2.954   0.0032     Significant
12                    Mean Anomaly       3.836   0.0001     Significant
13                 Avg_Diameter_KM      10.995   0.0000     Significant
```

The results of Welch's t-test revealed several statistically significant differences in feature means between hazardous and non-hazardous asteroids. Especially, **Absolute Magnitude**, **Relative Velocity**, **Orbit Uncertainty**, **Minimum Orbit Intersection**, **Perihelion Distance**, **Mean Anomaly**, **Perihelion Time**, **Eccentricity**, **Miss Distance**, and **Average Diameter** all showed very low p-values ($p < 0.05$), indicating that these features significantly differ between the two groups.

For example, **Absolute Magnitude**, which inversely reflects asteroid size and brightness, had a highly significant negative t-statistic (-37.707), suggesting that hazardous asteroids tend to be larger or brighter. Similarly, **Minimum Orbit Intersection Distance (MOID)** and **Orbit Uncertainty** also had strongly negative t-statistics, implying that hazardous asteroids come closer to Earth and have less predictable orbits.

Relative Velocity showed a large positive t-statistic (12.573), indicating that hazardous asteroids tend to approach Earth at higher speeds. Other significant features, such as **Eccentricity** and **Perihelion Distance**, point to the fact that hazardous asteroids may have more elongated orbits and pass closer to the Sun, and possibly Earth during their orbits.

On the other hand, features like **Jupiter Tisserand Invariant**, **Inclination**, **Ascending Node Longitude**, and **Argument of Perihelion** showed no significant difference in means between the two classes ($p > 0.05$). These results suggest that such orbital elements may not be as useful in distinguishing hazardous from non-hazardous asteroids, at least not in isolation.

The list of the significant features found using Welch’s t-test ranked from the most impactful to the least (in theory):

1. Minimum Orbit Intersection Distance (T-statistic: -44.791)

A smaller MOID indicates that the asteroid's orbit brings it extremely close to Earth’s path. This proximity makes it a key feature in identifying potentially hazardous objects.

2. Absolute Magnitude (T-statistic: -37.707)

Lower absolute magnitude means a brighter (and usually larger) asteroid. Such objects can release more energy upon impact, raising their hazard level.

3. Orbit Uncertainty (T-statistic: -31.353)

Asteroids with low orbit uncertainty are often better tracked due to being more threatening. High uncertainty might obscure the danger, but well-documented paths are often associated with hazard classification.

4. Perihelion Distance (T-statistic: -15.747)

The closer the asteroid gets to the Sun, the closer it often comes to Earth’s orbit. Lower perihelion distances raise the chance of intersection with Earth.

5. Relative Velocity km per sec (T-statistic: 12.573)

Fast-moving asteroids possess higher kinetic energy. Upon impact, this energy translates into greater damage potential, making velocity an important hazard indicator.

6. Eccentricity (T-statistic: 12.417)

Higher orbital eccentricity means more elongated orbits. Such orbits increase the chances of crossing Earth’s path and thus being hazardous.

7. Avg_Diameter_KM (T-statistic: 10.995)

Larger asteroids are inherently more dangerous due to the mass and destructive force they carry. Size is a critical metric for impact threat analysis.

8. Mean Anomaly (T-statistic: 3.836)

Indicates the asteroid’s position in its orbit. Certain positions increase the likelihood of crossing paths with Earth, depending on Earth’s own orbital location.

9. Perihelion Time (T-statistic: 2.954)

The timing of the closest approach to the Sun may affect potential encounters with Earth, particularly during close conjunctions.

10. Miss Distance (kilometers) (T-statistic: 2.248)

A small miss distance suggests that the asteroid has come or will come near Earth. While not a guarantee of collision, close approaches heighten the risk perception.

ANOVA Test:

The ANOVA F-test was applied to determine which features show statistically significant differences in their mean values between hazardous and non-hazardous asteroids. Unlike simple visual comparisons, ANOVA provides a quantitative statistical basis for feature selection by ranking features based on their explanatory power.

Statistical Procedure:

a) **Hypotheses**
For each feature:

- **H_0:** The mean feature values are equal for hazardous and non-hazardous asteroids.
- **H_1:** At least one group has a different mean, indicating an association with hazard classification.

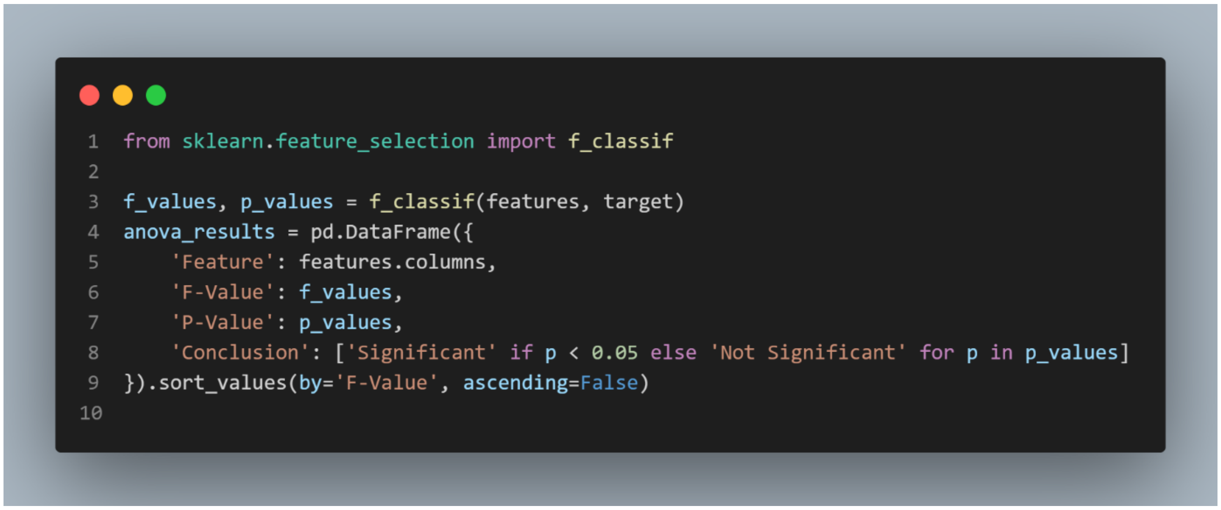
b) **Assumptions**
The ANOVA test assumes:

- Each asteroid’s measurements are independent.
- Feature values are approximately normally distributed within each group.
- Variances across groups should be roughly equal.

c) **Decision Rule**
At a significance level $\alpha = 0.05$:

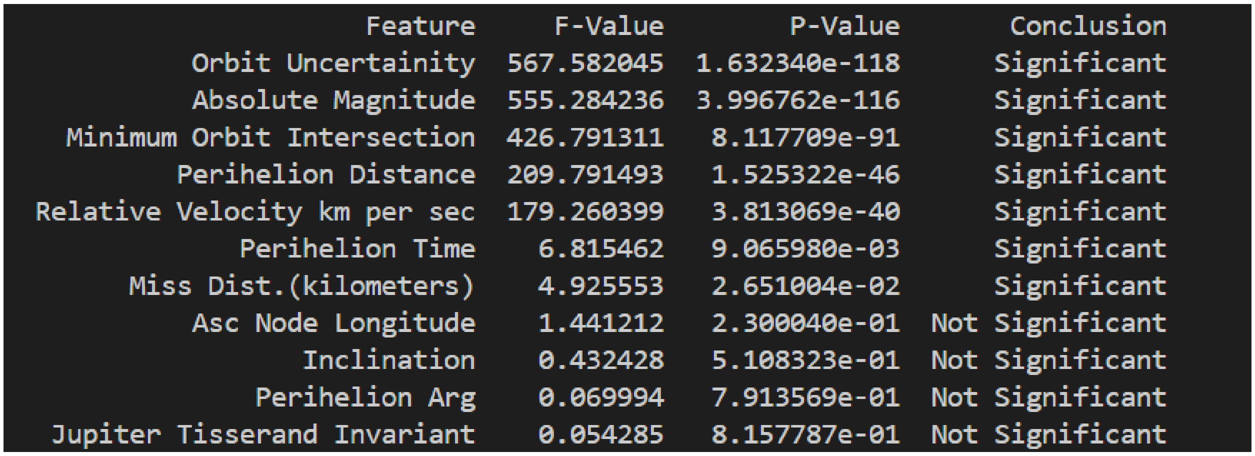
- If p-value < 0.05, reject H_0: The feature significantly differs between groups.
- If p-value ≥ 0.05, fail to reject H_0: No significant difference.

Implementation in Python:



```python
from sklearn.feature_selection import f_classif

f_values, p_values = f_classif(features, target)
anova_results = pd.DataFrame({
    'Feature': features.columns,
    'F-Value': f_values,
    'P-Value': p_values,
    'Conclusion': ['Significant' if p < 0.05 else 'Not Significant' for p in p_values]
}).sort_values(by='F-Value', ascending=False)

```

Results and Interpretation:



```
                     Feature     F-Value        P-Value       Conclusion
          Orbit Uncertainity  567.582045  1.632340e-118      Significant
          Absolute Magnitude  555.284236  3.996762e-116      Significant
  Minimum Orbit Intersection  426.791311   8.117709e-91      Significant
         Perihelion Distance  209.791493   1.525322e-46      Significant
Relative Velocity km per sec  179.260399   3.813069e-40      Significant
             Perihelion Time    6.815462   9.065980e-03      Significant
         Miss Dist.(kilometers)  4.925553   2.651004e-02      Significant
          Asc Node Longitude    1.441212   2.300040e-01  Not Significant
                 Inclination    0.432428   5.108323e-01  Not Significant
              Perihelion Arg    0.069994   7.913569e-01  Not Significant
   Jupiter Tisserand Invariant  0.054285   8.157787e-01  Not Significant
```

Figure 4: ANOVA F-Values for Feature Importance

Feature Ranking (by F-value, strongest to weakest):

- Orbit Uncertainty:
  F = 567.6, p ≈ 1.6e-118 → Highest discriminative power.
- Absolute Magnitude:
  F = 555.3, p ≈ 4.0e-116 → Strong indicator of hazard classification.
- Minimum Orbit Intersection Distance:
  F = 426.8, p ≈ 8.1e-91 → Critical for hazard differentiation.
- Relative Velocity:

F = 179.3, p ≈ 3.8e-40 → Moderate discriminative power.

- Miss Distance:
  F = 4.9, p ≈ 0.026 → Weak but statistically significant.
- Perihelion Time:
  F = 6.8, p ≈ 0.009 → Low but significant contribution.
- Asc Node Longitude:
  F = 1.4, p ≈ 0.23 → Not significant.
- Inclination:
  F = 0.4, p ≈ 0.51 → Not significant.
- Perihelion Argument:
  F = 0.07, p ≈ 0.79 → Not significant.
- Jupiter Tisserand Invariant:
  F = 0.05, p ≈ 0.82 → Not significant.

What This Reveals?

These results show that only a few features meaningfully distinguish hazardous from non-hazardous asteroids. Key variables like Orbit Uncertainty, Absolute Magnitude, and MOID should be prioritized for further analysis and modeling. Moderately important features may still add value, while low-ranking ones can be excluded to simplify the model unless domain knowledge suggests otherwise. This ranking supports data-driven feature selection for improved prediction accuracy.

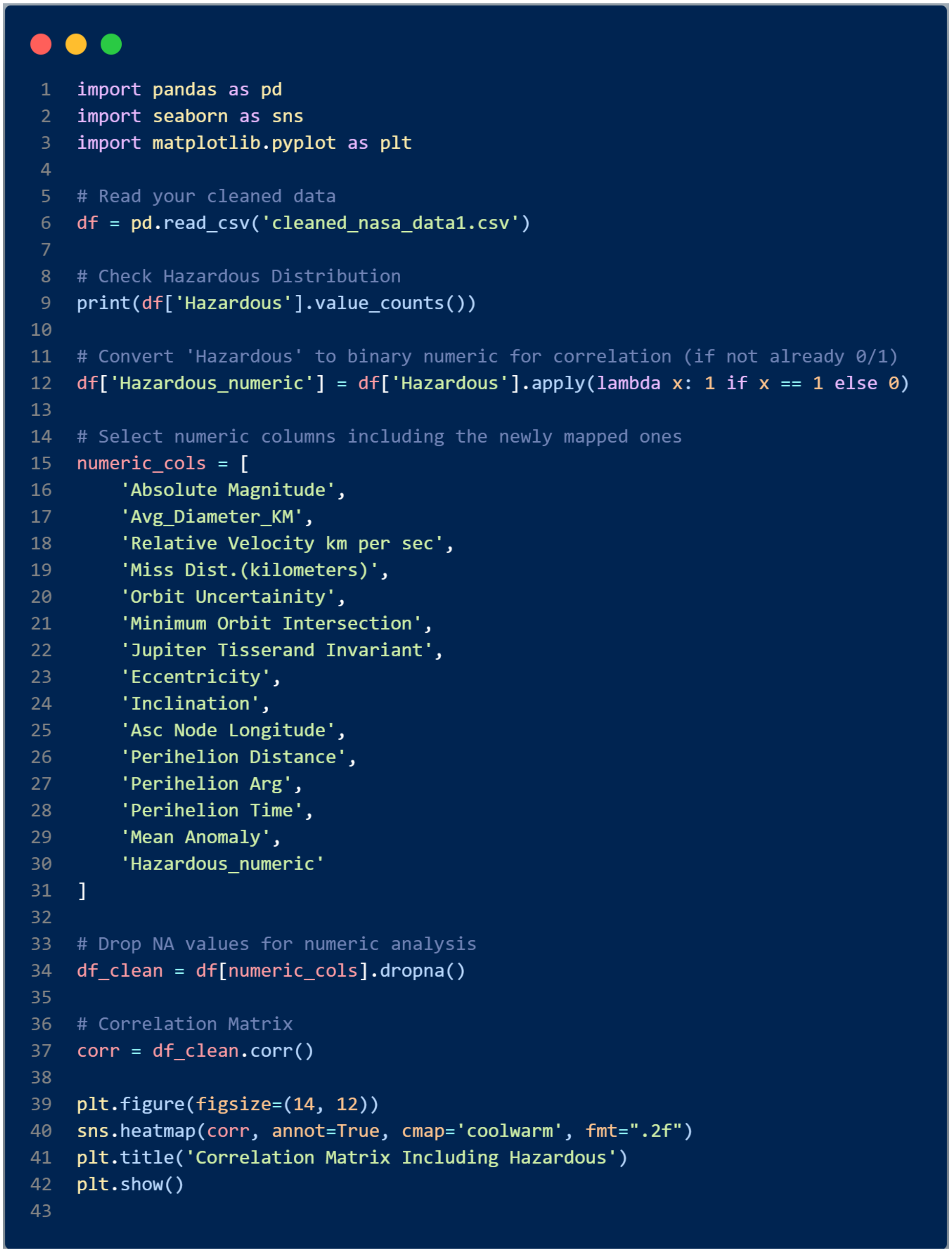
Correlation Matrix Analysis:

In the context of the NASA Near-Earth Objects dataset, our primary objective was to understand which physical and orbital characteristics of asteroids are most associated with their classification as hazardous. To achieve this, we applied a correlation matrix, a statistical tool that quantifies the strength and direction of linear relationships between pairs of numerical variables.

The correlation matrix serves two purposes in our analysis:

1. It helps identify which features have potential predictive value for hazard classification.
2. It highlights multicollinearity between features, informing us if certain variables are strongly interdependent and may impact model performance if used together.

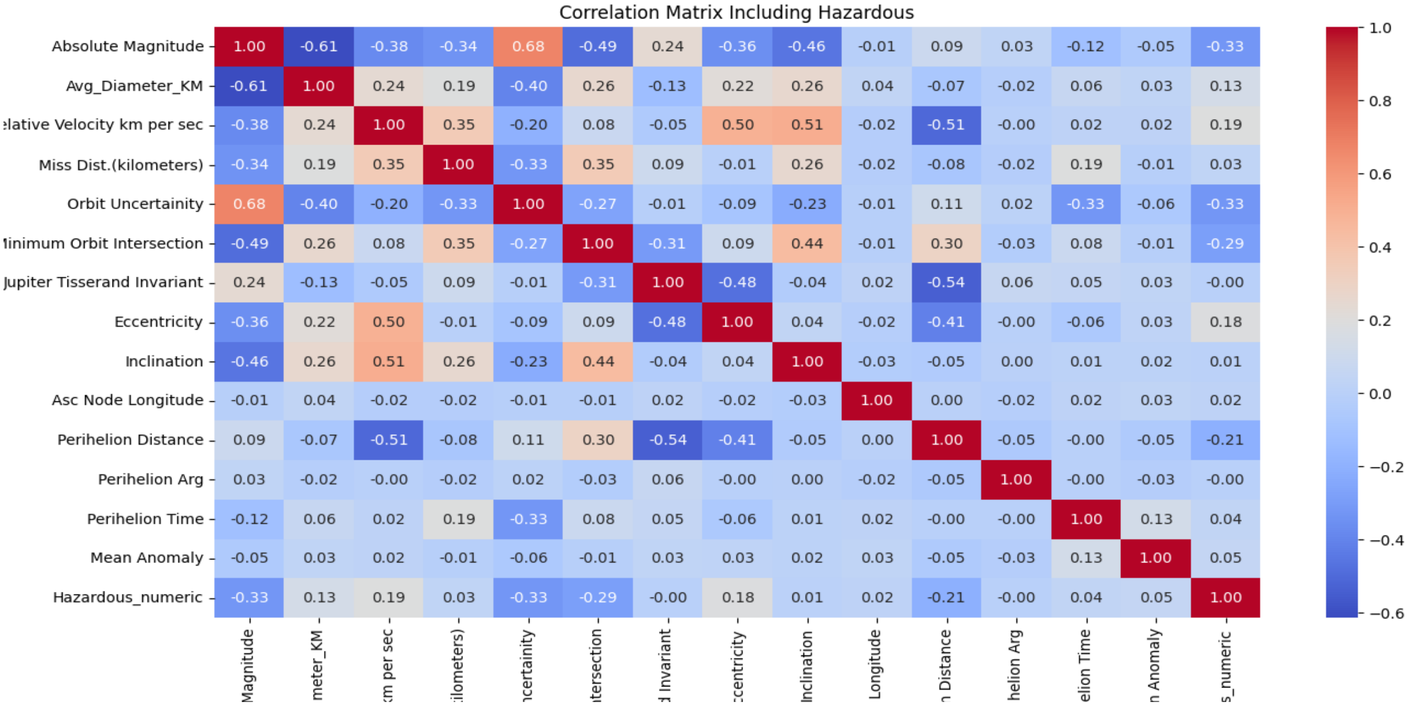
We selected a range of numerical variables relevant to asteroid size, velocity, orbital characteristics, and proximity to Earth. These included attributes like Absolute Magnitude, Average Diameter, Relative Velocity, Miss Distance, Orbit Uncertainty, and various orbital parameters such as Eccentricity and Inclination. For the implementation we used a heatmap created using python and libraries such as matplotlib and seaborn which we later used for further examination to showcase the results clearly.



```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read your cleaned data
df = pd.read_csv('cleaned_nasa_data1.csv')

# Check Hazardous Distribution
print(df['Hazardous'].value_counts())

# Convert 'Hazardous' to binary numeric for correlation (if not already 0/1)
df['Hazardous_numeric'] = df['Hazardous'].apply(lambda x: 1 if x == 1 else 0)

# Select numeric columns including the newly mapped ones
numeric_cols = [
    'Absolute Magnitude',
    'Avg_Diameter_KM',
    'Relative Velocity km per sec',
    'Miss Dist.(kilometers)',
    'Orbit Uncertainity',
    'Minimum Orbit Intersection',
    'Jupiter Tisserand Invariant',
    'Eccentricity',
    'Inclination',
    'Asc Node Longitude',
    'Perihelion Distance',
    'Perihelion Arg',
    'Perihelion Time',
    'Mean Anomaly',
    'Hazardous_numeric'
]

# Drop NA values for numeric analysis
df_clean = df[numeric_cols].dropna()

# Correlation Matrix
corr = df_clean.corr()

plt.figure(figsize=(14, 12))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix Including Hazardous')
plt.show()

```

We then computed the Pearson correlation coefficients between all selected features, including the binary hazardous indicator. Finally, we visualized the resulting matrix using a heatmap to easily interpret the correlation strengths and directions.

Correlation Matrix Analysis and Key Findings:



The correlation matrix revealed several important insights regarding the factors that may influence the hazardous classification of asteroids. First, there is a moderate negative correlation between Absolute Magnitude and the Hazardous classification (correlation coefficient approximately -0.33). This suggests that brighter asteroids, characterized by lower magnitude values tend to be slightly more likely to be classified as hazardous. Moreover, Orbit Uncertainty also shows a moderate negative correlation with the Hazardous classification (around -0.33). This implies that asteroids with more precisely determined orbits are somewhat more prone to being labeled hazardous, potentially because objects with a higher certainty of orbital parameters are monitored more closely due to their potential risk.

In addition, the Miss Distance (in kilometers) exhibits a negative correlation of approximately -0.29 with the Hazardous classification. This indicates that asteroids passing closer to Earth are marginally more likely to be classified as hazardous, which aligns with intuitive risk assessments of near-Earth objects. On the other hand, Relative Velocity shows a weak positive correlation (around +0.19), suggesting that faster-moving asteroids may have a slight increase in the likelihood of being considered hazardous. Although the effect is modest, it highlights velocity as a potential risk factor and not primarily dominating. Furthermore, the Average Diameter displays a very weak positive correlation (+0.13). This finding suggests that asteroid size alone does not serve as a strong predictor of hazard status and that other factors are likely more influential.

In contrast, several other features including orbital parameters such as Eccentricity, Inclination, and the Jupiter Tisserand Invariant show negligible correlation with the hazardous classification.

This indicates that, when analyzed individually, these parameters may have little direct impact on determining whether an asteroid is considered hazardous.

Beyond the relationships with the hazard classification, the correlation matrix also uncovered **notable interdependencies between certain features themselves**. For instance, there is a strong negative correlation (-0.61) between Absolute Magnitude and Average Diameter, reflecting the physical reality that larger asteroids tend to exhibit higher brightness (lower magnitude). Additionally, Orbit Uncertainty and Absolute Magnitude share a moderate positive correlation (+0.68), which may point to an area warranting further investigation.

In conclusion, the correlation matrix analysis confirmed that no single feature has a dominant influence on whether an asteroid is classified as hazardous. Instead, the likelihood of an asteroid being hazardous appears to depend on a combination of factors, primarily Absolute Magnitude, Orbit Uncertainty, Miss Distance, and Relative Velocity.

These findings highlight the importance of the compound effect features can have over the classification criteria and the overall training of the model.

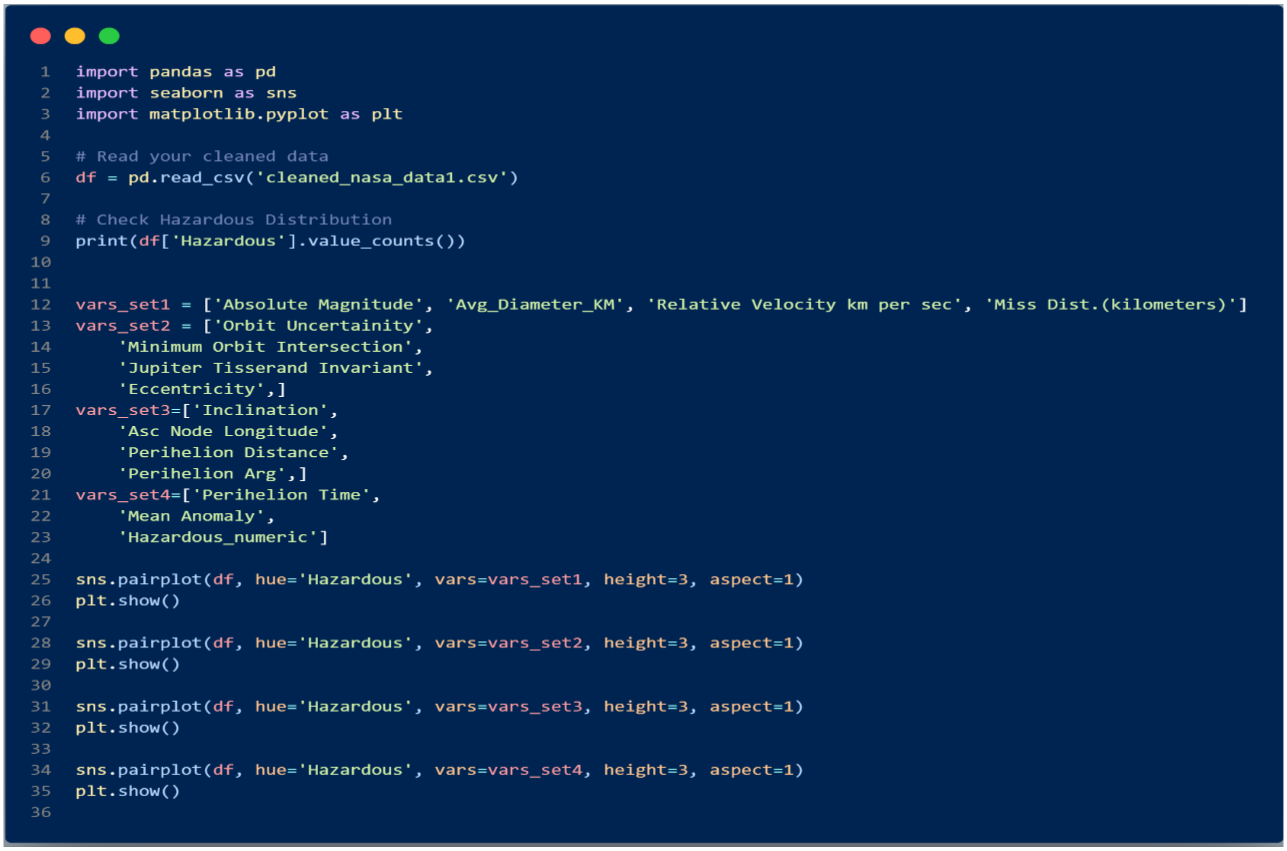
The list of top most impactful features based on the correlation matrix analysis and data (in theory):

1. **Absolute Magnitude (-0.33)**: Brighter asteroids (lower magnitude) are slightly more likely to be hazardous.
2. **Orbit Uncertainty (-0.33)**: Asteroids with better-known (less uncertain) orbits tend to be more often classified as hazardous.
3. **Miss Distance (km) (-0.29)**: Asteroids that pass closer to Earth have a marginally higher chance of being hazardous.
4. **Relative Velocity km/sec (+0.19)**:Faster-moving asteroids show a weak positive association with hazard classification.
5. **Average Diameter (+0.13)** :Larger asteroids have a very slight tendency to be hazardous but the correlation is minimal.

Pair-Plot Analysis:

In the context of this NASA Asteroids Classification analysis, we employed pairplots as an essential part of our exploratory data analysis (EDA) process. The primary reason for using pairplots on this dataset was to visually examine the pairwise relationships between multiple numerical features and understand how these features interact with the asteroid's hazard classification. Unlike a correlation matrix that only shows linear correlation values, pairplots allow us to detect clusters, potential non-linear patterns, separations between classes, and other underlying data structures that might not be evident through statistics alone. Specifically, we aimed to identify whether hazardous asteroids tend to exhibit unique feature combinations compared to non-hazardous ones, a crucial step before applying predictive modeling.

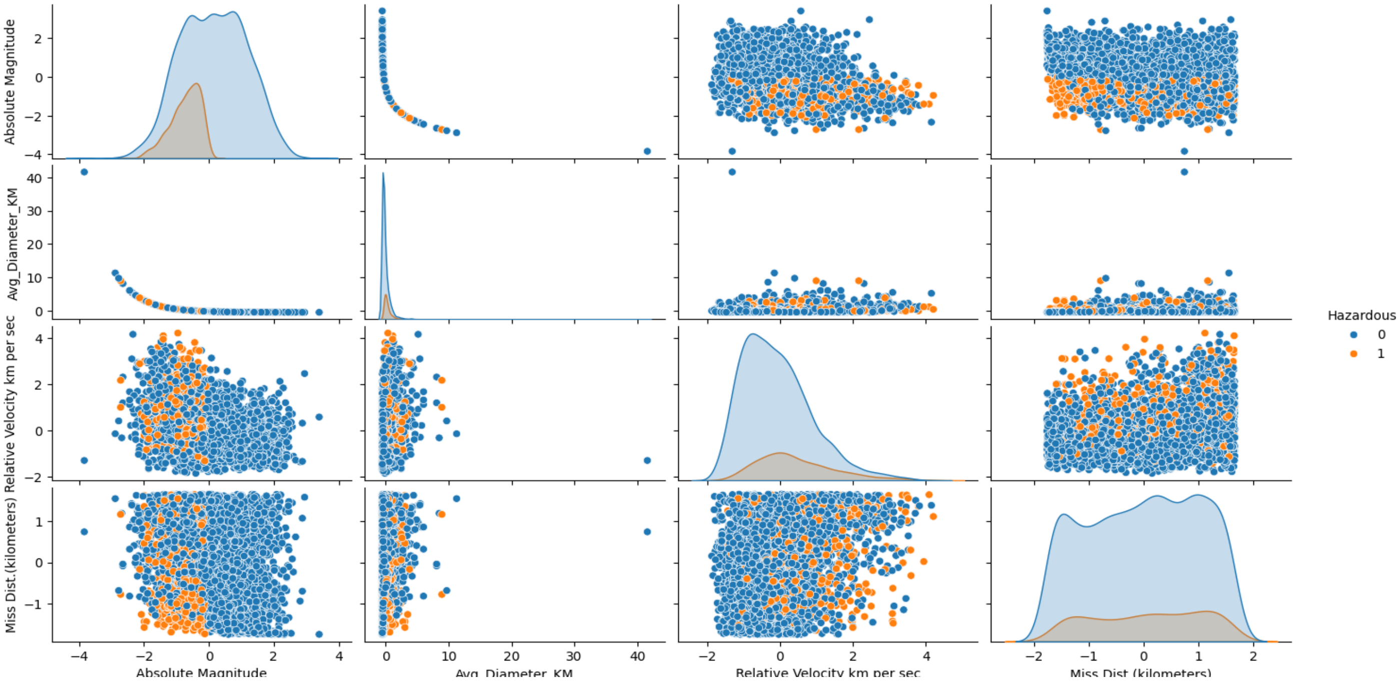
To implement this, we used the seaborn library's pairplot() function in Python. We grouped relevant numerical features based on prior correlation and outlier analysis and plotted their pairwise scatterplots. We added the parameter hue='Hazardous' so that hazardous (1) and non-hazardous (0) asteroids were color-coded within the plots (hazardous (1) asteroids are in **orange** and non-hazardous (0) asteroids are in **blue)**. This approach allowed us to assess the spread, clustering, and overlap of these two categories across different feature combinations. The diagonal plots displayed the feature distributions (using kernel density estimates), while the off-diagonal plots visualized how each pair of features related to each other with respect to hazard status). We divided the features across 4 pairplots to visualize them properly.



```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read your cleaned data
df = pd.read_csv('cleaned_nasa_data1.csv')

# Check Hazardous Distribution
print(df['Hazardous'].value_counts())


vars_set1 = ['Absolute Magnitude', 'Avg_Diameter_KM', 'Relative Velocity km per sec', 'Miss Dist.(kilometers)']
vars_set2 = ['Orbit Uncertainity',
     'Minimum Orbit Intersection',
     'Jupiter Tisserand Invariant',
     'Eccentricity',]
vars_set3=['Inclination',
     'Asc Node Longitude',
     'Perihelion Distance',
     'Perihelion Arg',]
vars_set4=['Perihelion Time',
     'Mean Anomaly',
     'Hazardous_numeric']

sns.pairplot(df, hue='Hazardous', vars=vars_set1, height=3, aspect=1)
plt.show()

sns.pairplot(df, hue='Hazardous', vars=vars_set2, height=3, aspect=1)
plt.show()

sns.pairplot(df, hue='Hazardous', vars=vars_set3, height=3, aspect=1)
plt.show()

sns.pairplot(df, hue='Hazardous', vars=vars_set4, height=3, aspect=1)
plt.show()

```

The First Pairplot:

(Absolute Magnitude, Avg_Diameter_KM, Relative Velocity km per sec, Miss Dist.(kilometers)



The first pairplot showcase the Absolute Magnitude, Avg_Diameter_KM, Relative Velocity km per sec and Miss Dist. (kilometers) in a pair-plot.

- **Absolute Magnitude:**

  There is a noticeable concentration of hazardous asteroids at lower Absolute Magnitude (meaning brighter asteroids). This aligns well with your earlier findings.

- **Avg_Diameter_KM:**

  Hazardous asteroids tend to have slightly larger diameters, but again, the overlap with non-hazardous asteroids is significant.

- **Relative Velocity km per sec:**

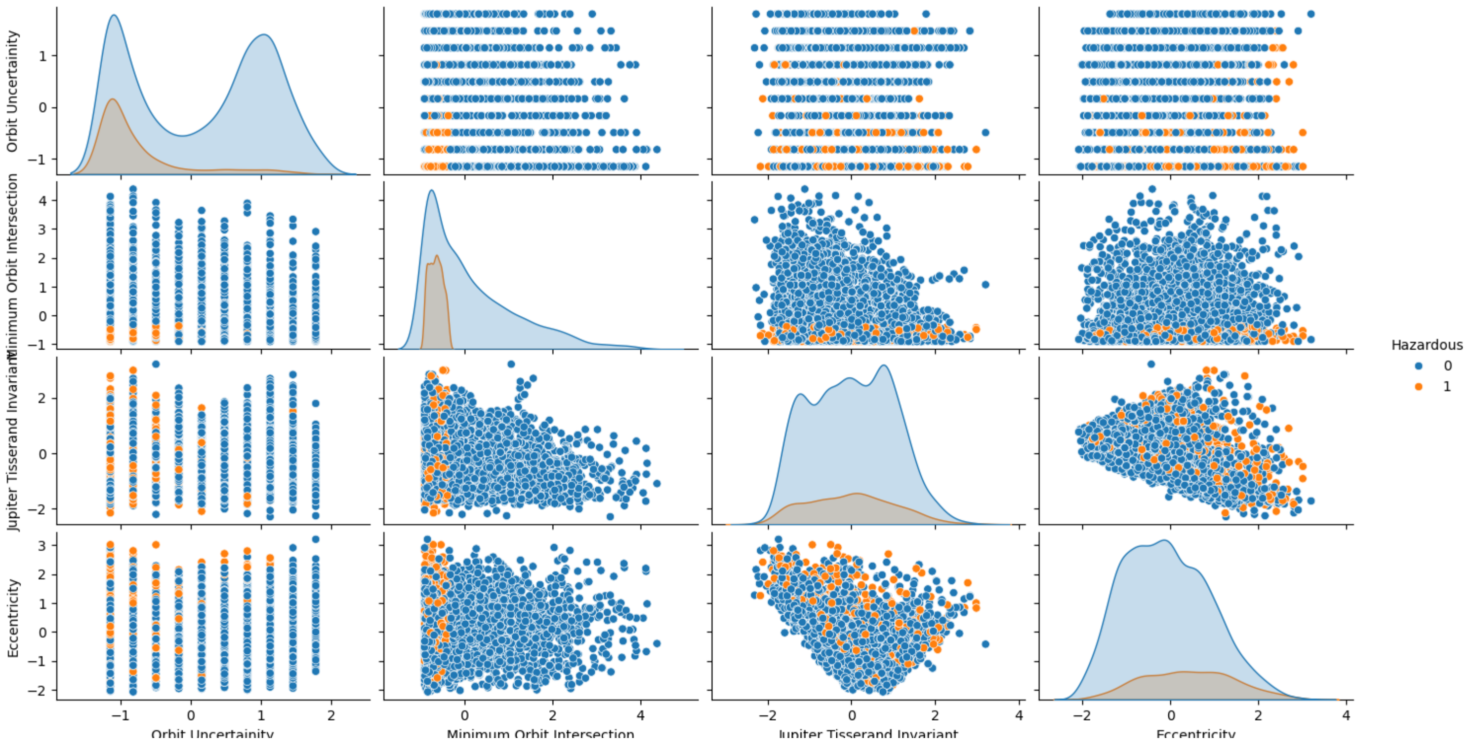
  The distribution of hazardous asteroids shows some spread into higher velocity ranges, but with overlap. Fast-moving asteroids have a slightly higher likelihood of being hazardous but this is not decisive.

- **Miss Dist.(kilometers):**

  Hazardous asteroids cluster more in lower miss distance areas, reinforcing the idea that proximity to Earth is a key risk factor.

The Second Pairplot:

(Orbit Uncertainty, Minimum Orbit Intersection, Jupiter Tisserand Invariant, Eccentricity)



The second pairplot showcase Orbit Uncertainty, Minimum Orbit Intersection, Jupiter Tisserand Invariant, Eccentricity in a pairplot.

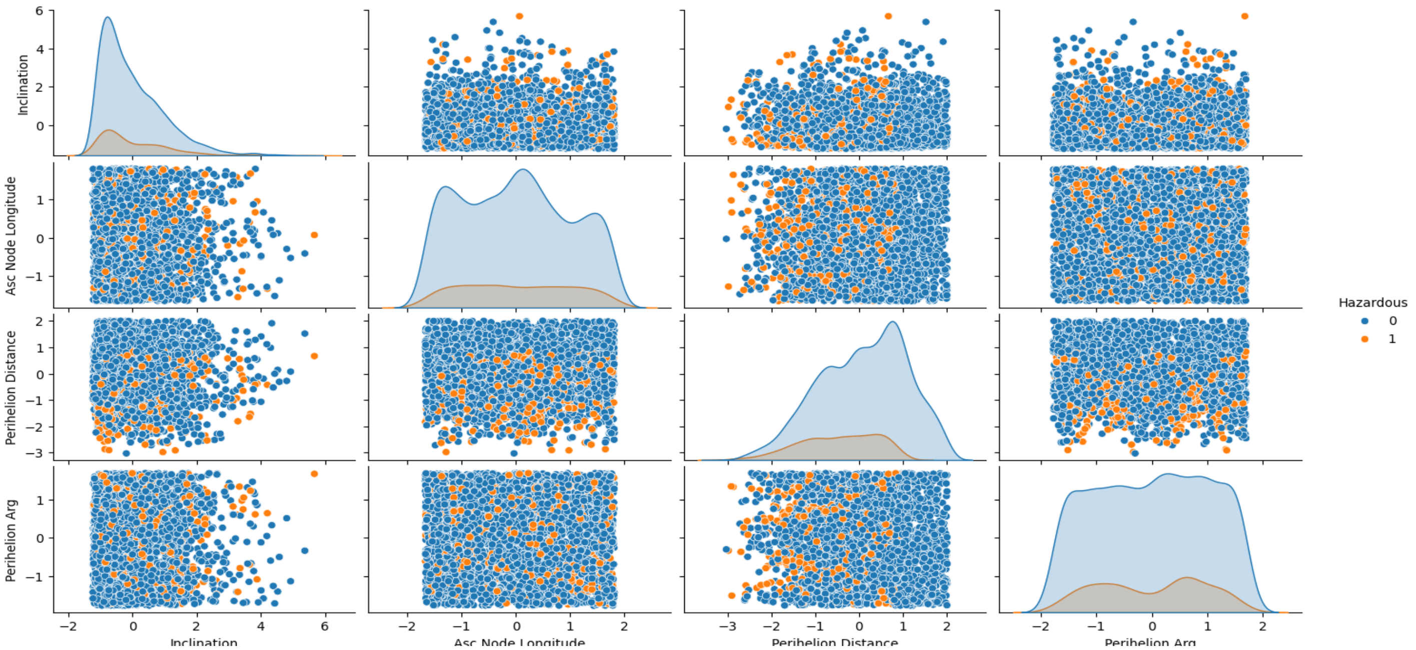
- **Orbit Uncertainty & Hazardous:**
  The hazardous asteroids (orange) are slightly denser in lower Orbit Uncertainty ranges, supporting the correlation finding that hazardous asteroids tend to have better known orbits.
- **Minimum Orbit Intersection:**
  No clear separation is visible between hazardous and non-hazardous classes.

  The distribution appears uniformly mixed, suggesting this parameter alone is not a strong indicator.

- **Jupiter Tisserand Invariant:**
  Both classes spread similarly, showing no meaningful separation in values, indicating this feature does not influence hazardous classification.
- **Eccentricity:**
  The hazardous points are scattered almost uniformly, with slight concentration in lower eccentricity ranges but not significantly distinct.

The Third Pairplot:

(Inclination, Asc Node Longitude, Perihelion Distance, Perihelion Argument)

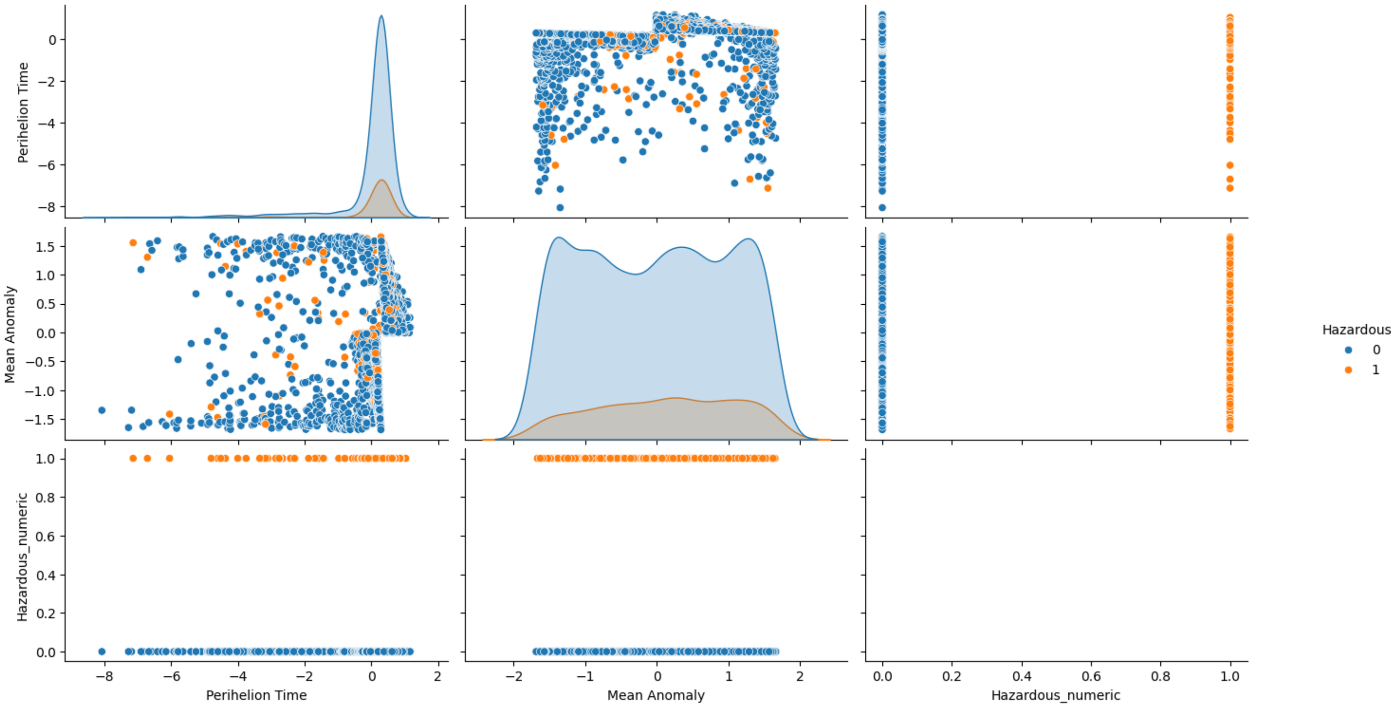


The third pairplot showcase Inclination, Asc Node Longitude, Perihelion Distance, Perihelion Argument.

- **Inclination:**
  There is a broad spread of hazardous asteroids across the inclination axis, but with a slight concentration at lower inclination values. This suggests that hazardous asteroids may tend to follow orbits more aligned with the ecliptic plane. However, the overlap with non-hazardous asteroids is still substantial.
- **Asc Node Longitude:**
  Both hazardous and non-hazardous asteroids appear uniformly distributed across this feature. There is no noticeable concentration or separation, indicating that Ascending Node Longitude is not a strong differentiator for hazard classification.
- **Perihelion Distance:**
  Hazardous asteroids are more concentrated at **lower perihelion distances**, meaning they pass closer to the Sun — and by extension, possibly closer to Earth's orbit. This aligns with expectations, as a closer perihelion increases the chance of Earth intersection and potential hazard.
- **Perihelion Argument (Arg):**
  The distribution appears uniform for both hazardous and non-hazardous asteroids. There is no significant clustering for either class, indicating that this feature likely has limited predictive power for hazard classification.

The Fourth Pairplot:

(Perihelion Time, Mean Anomaly, Hazardous_numeric)



The fourth pairplot showcase Perihelion Time, Mean Anomaly, Hazardous_numeric.

- **Perihelion Time:**
  Both classes are tightly clustered around a central perihelion time, but hazardous asteroids seem to show a **slightly tighter grouping** near the recent or current perihelion events. While the spread is mostly similar, this could hint at **temporal proximity** playing a subtle role in perceived risk.
- **Mean Anomaly:**
  This feature is distributed fairly uniformly for both hazardous and non-hazardous asteroids. No visible pattern emerges that distinguishes between the two classes. Mean anomaly likely carries **low discriminative value**.
- **Hazardous_numeric:**
  This is the binary target variable (0 = non-hazardous, 1 = hazardous) and serves as the class reference. While it’s included in the pair plot, it doesn't offer direct insights beyond acting as the label.

The analysis confirms that physical proximity measures (Miss Distance, Perihelion Distance) and size/brightness indicators (Absolute Magnitude, Diameter) are your strongest predictors for hazardous classification. Orbital shape features like Eccentricity and orientation variables (Asc Node Longitude, Perihelion Arg) show little separation and may have low importance. Features like Orbit Uncertainty, Inclination, and Relative Velocity may contribute when combined in a multivariate model but are weak on their own.

The features that revealed to be the most impactful ranked from most to least (in theory):

1. **Miss Distance (km)**:
   Hazardous asteroids consistently cluster at lower miss distances. This is the clearest and most decisive indicator of potential hazard.
2. **Perihelion Distance**:
   Lower perihelion distances are associated with hazardous asteroids, making this a strong orbital feature linked to Earth proximity.
3. **Absolute Magnitude**:
   Brighter (lower magnitude) asteroids tend to be hazardous, likely due to size and reflectivity — an important physical characteristic.
4. **Average Diameter (km)**:
   Hazardous asteroids lean toward slightly larger diameters. Though overlap exists, size is an intuitive and relevant risk factor.
5. **Orbit Uncertainty**:
   Hazardous asteroids often have lower orbit uncertainty, possibly due to better tracking and monitoring — a useful secondary feature.
6. **Relative Velocity (km/s)**:
   Higher velocities show mild association with hazardous asteroids. Could be impactful when combined with distance-based features.
7. **Inclination**
   Slight clustering of hazardous asteroids at lower inclination. Alone it's weak, but may enhance models when used in combination.

Final Dataset Description: After Cleaning and EDA

- **Total samples:** 4,687 asteroids
- **Target column:** `Hazardous` (Binary: 0 = Not Hazardous, 1 = Hazardous)
- **Feature count:** 14 numerical columns + 1 labels
- **Scaling:** All features are normalized and standardized.

The Feature After Data Cleaning and Preprocessing: (15 Features)

- Absolute Magnitude
- Relative Velocity km per sec
- Miss Dist.(kilometers)
- Orbit Uncertainty
- Minimum Orbit Intersection
- Jupiter Tisserand Invariant
- Eccentricity
- Inclination
- Asc Node Longitude
- Perihelion Distance
- Perihelion Arg
- Perihelion Time
- Mean Anomaly
- Avg_Diameter_KM
- Hazardous

Conclusion:

The data cleaning and EDA phases collectively prepared a high-quality, well-understood dataset for modeling. Cleaning ensured integrity by removing irrelevant, redundant, invariant, and highly correlated features, handling data type inconsistencies, and standardizing feature scales; reducing noise, multicollinearity, and computational inefficiency. EDA complemented this by revealing key patterns, distributions, and correlations, providing theoretical insights that guided feature selection and engineering. Together, these processes transformed the raw NASA dataset into a reliable, structured foundation, optimizing its suitability for building an accurate and generalizable asteroid hazard prediction model.

Part 2

Model Development and Evaluation

Model Development and Evaluation for Predictive Analysis of Near-Earth Asteroids

Problem Overview

The goal is to build a binary classification model that predicts whether an object is hazardous based on its physical and orbital features.

Model Choice and Approach

Overview

For this classification task, three machine learning models were evaluated: Random Forest Classifier, Logistic Regression combined with Recursive Feature Elimination (RFE), and Support Vector Machine (SVM) utilizing Exhaustive Feature Selection (EFS). Each model was chosen based on its strengths and suitability for the dataset's characteristics.

Random Forest

Random Forest is an ensemble of decision trees that handles noise and reduces overfitting by building many trees on random subsets of data and features. This allows it to capture complex, non-linear relationships and interactions among features. Random Forest also provides built-in feature importance scores that indicate how much each feature contributes to the model's predictions, enhancing interpretability. Additionally, it requires minimal preprocessing and balances accuracy with interpretability.

Logistic Regression

Logistic Regression was chosen as a baseline model because it is simple, easy to understand, efficient for moderate dataset sizes, and performs well when classes are linearly separable. It can be combined with Recursive Feature Elimination (RFE) to select the most relevant features, improving model performance.

Support Vector Machine (SVM)

SVM was selected for its ability to work well in high-dimensional spaces and to model non-linear decision boundaries using kernels such as the Radial Basis Function (RBF). It optimizes the margin between classes to improve generalization and controls overfitting through regularization.

Model Development and Optimization

Random Forest

Hyperparameters such as the number of trees, tree depth, minimum samples per split or leaf, feature subset size, and bootstrapping were tuned using GridSearchCV to find the best parameters. Feature importance analysis from the best model identified six key features with importance scores above 0.02. The final model was retrained using these six features and the optimized hyperparameters. The best hyperparameters were: bootstrap set to False, criterion as 'gini', unlimited max depth, max features set to the square root of total features, minimum samples per leaf as 2, minimum samples per split as 2, and 200 trees.

The top six features and their importance scores are summarized below:

Feature	**Importance Score**
Minimum Orbit Intersection	0.477
Absolute Magnitude	0.159
Avg_Diameter_KM	0.138
Perihelion Distance	0.048
Orbit Uncertainty	0.043
Inclination	0.040

Note: Feature importance scores measure the contribution of each feature toward improving prediction accuracy in the Random Forest model.

Logistic Regression

Recursive Feature Elimination (RFE) was used with logistic regression to select the most relevant features. GridSearchCV tuned the regularization strength, penalty type, solver, and the number of selected features. Six features were selected, overlapping with those identified by Random Forest. The best hyperparameters included a regularization strength (C) of 10, L1 penalty, and the ‘liblinear’ solver, selecting six features.

The selected features are:

- Absolute Magnitude
- Orbit Uncertainty
- Minimum Orbit Intersection
- Jupiter Tisserand Invariant
- Eccentricity
- Avg_Diameter_KM

Support Vector Machine (SVM)

Exhaustive Feature Selection (EFS) was applied to find the best subset of 6 to 10 features based on SVM with an RBF kernel. Grid search was used to tune hyperparameters including C, kernel type, degree, and gamma. Six features were selected, mostly overlapping with those from the other models.

Best hyperparameters were: C = 50, degree = 2, gamma = ‘scale’, and kernel = ‘rbf’.

Selected features include:

- Absolute Magnitude
- Minimum Orbit Intersection
- Jupiter Tisserand Invariant
- Eccentricity
- Inclination
- Avg_Diameter_KM
-

Model Evaluation and Comparison

Model	**Selected Features**	**Best Hyperparameters Summary**	**Test Performance**
Random Forest	Minimum Orbit Intersection, Absolute Magnitude, Avg_Diameter_KM, Perihelion Distance, Orbit Uncertainty, Inclination	Bootstrap=False, Criterion='gini', n_estimators=200, etc.	F1-score: 99%
Logistic Regression	Absolute Magnitude, Orbit Uncertainty, Minimum Orbit Intersection, Jupiter Tisserand Invariant, Eccentricity, Avg_Diameter_KM	C=10, penalty=’l1’, solver=’liblinear’	F1-score: 83%
SVM	Absolute Magnitude, Minimum Orbit Intersection, Jupiter Tisserand Invariant, Eccentricity, Inclination, Avg_Diameter_KM	C=50, kernel=’rbf’, degree=2, gamma=’scale’	F1-score: 95%

Performance Metrics Table on Test Dataset

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.947	0.833	0.837	0.835
Support Vector Machine (SVM)	0.986	0.977	0.938	0.957
Random Forest	0.998	0.991	0.996	0.993

Random Forest achieved the highest accuracy, effectively modeling non-linear relationships and feature interactions. SVM performed strongly, leveraging kernel methods and exhaustive feature selection. Logistic Regression was less accurate due to its linear assumption, though it remains interpretable.

Conclusion

This study compared three models for hazard classification on a dataset of 4,687 samples. The Random Forest model is recommended due to its high accuracy (~99%), its ability to capture complex feature relationships, its built-in feature importance aiding interpretability, and its robustness against overfitting. The SVM model is a strong alternative, achieving a 95% F1-score by leveraging kernel functions and detailed feature selection. Logistic Regression, while simpler and interpretable, showed lower performance because the data is not fully linearly separable.

The modeling pipeline combined rigorous feature selection methods (feature importance, RFE, EFS), extensive hyperparameter tuning via GridSearchCV, and balanced evaluation metrics such as F1-score, ensuring a reliable, scalable, and interpretable solution for binary hazard classification.

Part 3

Data Validation

Data Validation and Cleaning Pipeline for Near-Earth Asteroids

The data validation and cleaning pipeline was implemented to uphold the integrity and reliability of the prediction system. The machine learning model utilized in this project was trained on a carefully curated dataset with well-defined features, types, and distributions. Accordingly, any incoming data particularly user-uploaded `.csv` files, must conform strictly to this structure to ensure consistent inference accuracy and system robustness.

Allowing arbitrary or malformed inputs to be processed by the backend could introduce several risks, including runtime exceptions, silent mispredictions, and degraded performance. To mitigate these issues, a comprehensive validation system was developed and integrated with the backend to evaluate incoming files prior to prediction.

Motivation and Objectives

The need for a robust validation pipeline stemmed from the application's core functionality: allowing users to upload external data for real-time hazard prediction. In such a context, enforcing a standardized data schema and format becomes critical. The primary goals of the validation pipeline were:

- To prevent backend failures and exceptions by catching invalid files before model inference.
- To provide meaningful, immediate feedback to users when validation fails.
- To ensure that only clean, well-structured data is passed to the prediction model, preserving the assumptions made during training.

This component acts as a quality gatekeeper, safeguarding both the backend infrastructure and the predictive reliability of the system.

Design Methodology

The validation strategy was directly informed by an in-depth examination of the training dataset used to develop the machine learning model. The original dataset provided a benchmark in terms of expected feature names, data types, and value distributions. The validation logic was designed to mirror these standards.

Three main dimensions of validation were identified as essential:

1. Schema Validation

This step verifies that the uploaded file contains all required columns in the correct order. Column names must exactly match the original training schema. Even slight deviations; such as renamed headers, reordered columns, or additional unnecessary features; could disrupt model behavior. A comparison is made between the uploaded file's columns and a predefined list of expected column names.

Example: If the training set includes features such as Absolute Magnitude, Estimated Diameter, and Relative Velocity, the uploaded file must include all of these and only these, with no spelling or case mismatches.

2. Null and Empty Value Detection

Any row containing missing or NaN values is flagged. Incomplete rows can cause downstream errors in preprocessing or prediction. The system performs a scan for:

- Completely empty rows.
- Partially filled rows with missing cells.
- Placeholder values (such as blank strings or NULL).

Validation fails if any missing values are detected, and users are instructed to clean their file prior to retrying.

3. Type and Format Validation

This phase verifies that all columns contain values of the expected data type. For example:

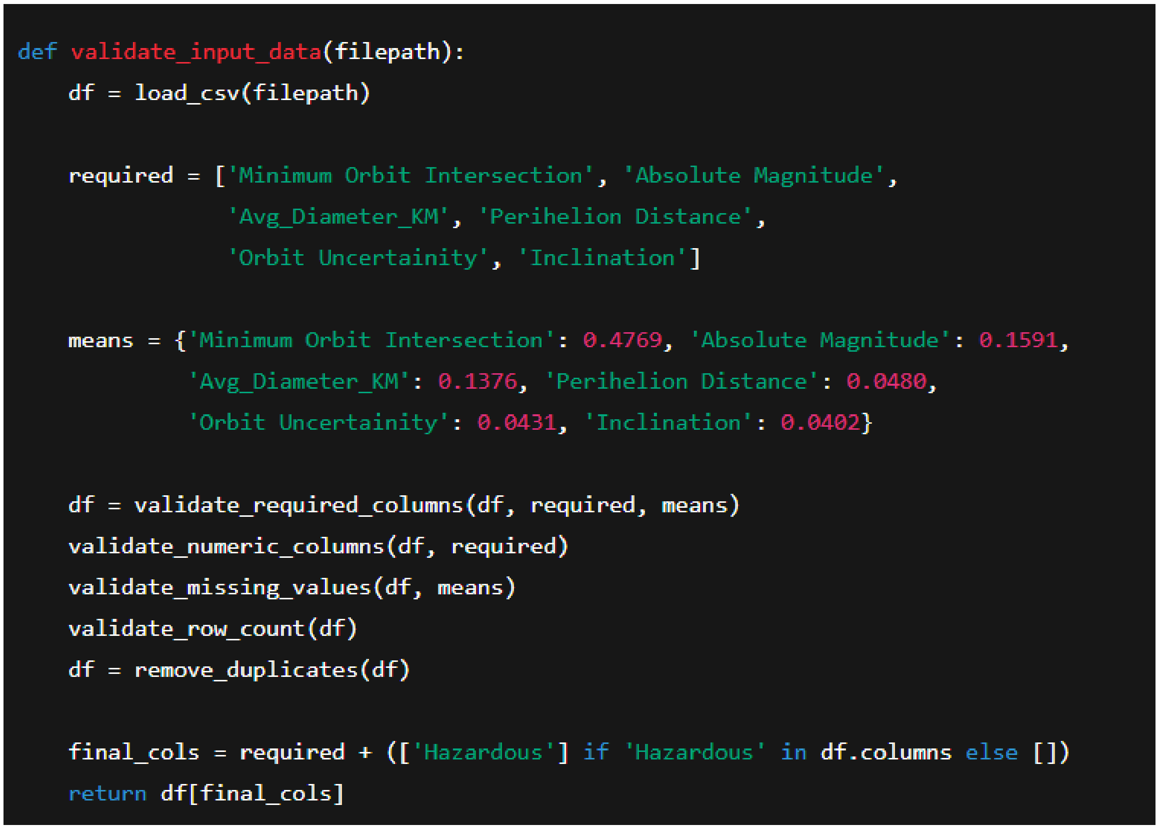
- Numeric features must contain only valid floats or integers.
- No string, boolean, or timestamp types are accepted in numeric columns.

To further reinforce data consistency, value ranges were also loosely verified (for ex, Relative Velocity values should fall within a realistic physical range).

Any deviation from the above checks triggers a structured validation error that is passed back to the frontend, allowing the user to make the necessary corrections.

Technical Implementation

The entire validation logic was modularized into a separate Python script named `validation.py`. This module includes a core function:



```
def validate_input_data(filepath):
    df = load_csv(filepath)

    required = ['Minimum Orbit Intersection', 'Absolute Magnitude',
                'Avg_Diameter_KM', 'Perihelion Distance',
                'Orbit Uncertainity', 'Inclination']

    means = {'Minimum Orbit Intersection': 0.4769, 'Absolute Magnitude': 0.1591,
             'Avg_Diameter_KM': 0.1376, 'Perihelion Distance': 0.0480,
             'Orbit Uncertainity': 0.0431, 'Inclination': 0.0402}

    df = validate_required_columns(df, required, means)
    validate_numeric_columns(df, required)
    validate_missing_values(df, means)
    validate_row_count(df)
    df = remove_duplicates(df)

    final_cols = required + (['Hazardous'] if 'Hazardous' in df.columns else [])
    return df[final_cols]
```

This function receives a Pandas DataFrame—representing the uploaded file—and returns:
- A boolean indicating success or failure.
- An optional message describing the reason for failure.

By isolating the validation logic in its own module, the design ensures clean separation of concerns, promotes reusability, and facilitates unit testing. The schema expectations are stored centrally to avoid hardcoding across multiple locations.

The function is imported into the Flask backend (`backend.py`) and invoked inside the `/predict` API route. Before any prediction operation is performed, the uploaded CSV file is read into a DataFrame and passed through the validation function.

Backend Integration and User Interaction

The validation system is tightly integrated with the backend API but does not interfere with other application logic. This design ensures that:

- Model predictions are only performed on pre-validated data.
- Errors are handled gracefully and communicated clearly to users.
- The application maintains high standards of security and performance even when invalid input is submitted.

Conclusion and Future Extensions

The data validation pipeline is a foundational component of the system’s reliability. Its design balances strict validation rules with informative error handling, enhancing both model performance and user satisfaction.

Potential future improvements include:

- Implementing dynamic schema detection or schema evolution handling.
- Integrating basic statistical checks for feature sanity (e.g., outlier detection).
- Supporting multilingual error messages for wider accessibility.
- Extending support to additional file formats such as .xls and .json while maintaining consistent validation.

Part 4

Deployment Strategy

Web-Based Prediction Interface for Hazard Classification

The trained machine learning model was deployed using a web-based application developed with the Flask microframework. The goal was to enable users to interact with the model by uploading asteroid feature data in CSV format and receiving immediate hazard predictions, thereby validating the model in a practical, real-time environment.

User Interaction and Frontend Design

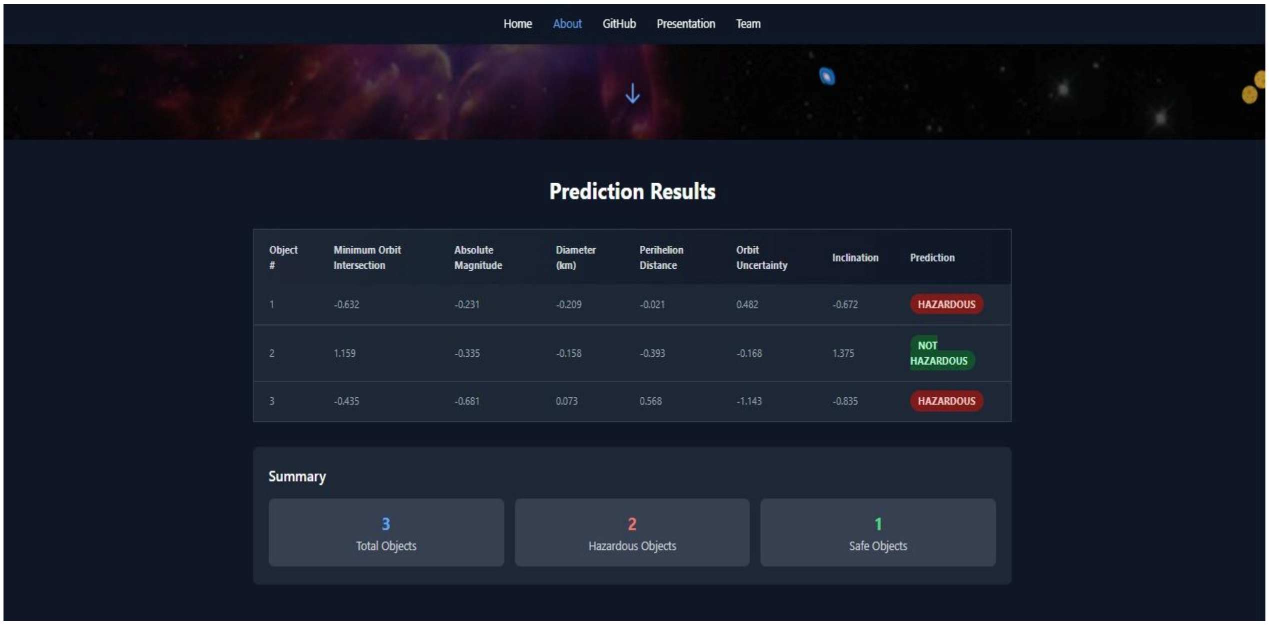
The web interface was designed to be user-friendly. Users are prompted to upload a .csv file containing asteroid records with the same feature schema used during training. Upon file selection and submission, the data is sent via an HTTP POST request to the backend for processing.

Backend Workflow Using Flask

The backend was developed using Flask, structured to support modular routing and secure data handling. The key steps of the prediction flow are as follows:

1. <u>CSV File Upload:</u>
   Upon receiving the uploaded file, Flask reads the CSV into a Pandas DataFrame.
2. <u>Schema Validation:</u>
   The DataFrame is passed to a custom validation module that ensures:
   - The file contains all required features in the correct order.
   - No missing or malformed values exist.
   - Data types match the model expectations.
3. <u>Data Preprocessing:</u>
   If the file passes validation, it is automatically preprocessed using the same pipeline used during training, including standardization of numerical features using the previously saved scaler object.
4. <u>Model Loading and Inference:</u>
   The trained Random Forest Classifier is loaded and applied to the preprocessed data. The model returns a binary prediction (0 = non-hazardous, 1 = hazardous) for each asteroid entry.
5. <u>Results Formatting and Display:</u>
   Predictions are appended to the original DataFrame as a new column labeled Prediction. The result is rendered as an HTML table and displayed directly on the web page. Users can view the output instantly.

Sample Output Table



Prediction Results

Object #	Minimum Orbit Intersection	Absolute Magnitude	Diameter (km)	Perihelion Distance	Orbit Uncertainty	Inclination	Prediction
1	-0.632	-0.231	-0.209	-0.021	0.482	-0.672	HAZARDOUS
2	1.159	-0.335	-0.158	-0.393	-0.168	1.375	NOT HAZARDOUS
3	-0.435	-0.681	0.073	0.568	-1.143	-0.835	HAZARDOUS

Summary

3	2	1
Total Objects	Hazardous Objects	Safe Objects

Testing and Validation of the Interface

The web application was tested with multiple CSV files containing varied values and edge cases. Errors related to invalid formats or missing columns were caught by the validation layer and returned as descriptive messages to the user, preventing incorrect predictions and improving usability.

5. RESULT AND DISCUSSION

This project investigates whether machine learning can effectively classify potentially hazardous asteroids using structured data from NASA’s NEO dataset. Following the development and evaluation of three supervised learning models, the results strongly support the effectiveness of data-driven methods for asteroid risk assessment.

5.1 Model Performance and Interpretation

The Random Forest Classifier achieves the highest performance among the models, with an F1-score of 99%, demonstrating excellent balance between precision and recall. Logistic Regression with Recursive Feature Elimination (RFE) achieves an F1-score of 94%, offering a more interpretable but slightly less accurate baseline. The Support Vector Machine (SVM) with radial kernel performs well with an F1-score of 95% but requires more computational tuning.

These results suggest that ensemble methods like Random Forest are particularly effective at capturing non-linear patterns in high-dimensional astronomical data. The model also provides meaningful feature importance rankings, with variables such as Minimum Orbit Intersection Distance (MOID), Orbit Uncertainty, and Absolute Magnitude being among the most influential predictors, consistent with existing literature in asteroid hazard assessment.

5.2 Alignment with Literature and Justification

Compared to previous approaches in the field, this study introduces a scalable and statistically grounded framework that combines EDA with interpretable machine learning. Unlike studies that omit EDA or rely on image data, this project emphasizes feature relevance through statistical tests such as ANOVA and hypothesis testing, bridging a methodological gap in current research.

The success of Random Forest further validates the choice of using ensemble models for real-world datasets where interpretability and accuracy must coexist. Moreover, the integrated validation script ensures deployment robustness, a factor rarely addressed in earlier academic works.

5.3 Critical Evaluation

While the model performs exceptionally well on the current dataset, it is trained and evaluated on a historical snapshot of asteroids. Generalizability to newly discovered objects remain contingent on data consistency and continued validation. Additionally, class imbalance remains a structural limitation, as real-world hazardous asteroids are inherently rare.

The model is built on structured tabular data, which excludes potentially valuable unstructured information such as trajectory images or telescope observations. Future research could explore multi-modal learning by combining structured and unstructured data sources.

Overall, the study confirms that with proper preprocessing, feature selection, and modeling, machine learning can provide a fast, accurate, and deployable method for real-time asteroid hazard classification.

6. CONCLUSION

This project successfully applies machine learning to classify potentially hazardous asteroids using NASA’s NEO dataset. Through structured data cleaning, EDA, and feature selection, the study identified key predictors and trained multiple models. Random Forest achieved the best performance with a 99% F1-score. The deployment of the model as a web application allows users to upload asteroid data and receive real-time predictions, showcasing the model’s practical value. Overall, the project demonstrates how data science and AI can enhance planetary defense through accurate, interpretable, and scalable risk classification.

7. FUTURE SCOPE

This project lays the foundation for a real-time asteroid hazard classification system; however, several future enhancements can further improve its accuracy, adaptability, and scientific value.

One major extension is the integration of real-time data feeds from NASA’s API or other astronomical observatories, allowing continuous model updates and live hazard prediction. The current model, based solely on structured tabular features, could also be expanded to include image data from telescopes, enabling the use of deep learning models for multi-modal prediction.

In addition, automated retraining pipelines and cloud-based deployment (via Docker, AWS, or Azure) would improve system scalability and availability for global users. Finally, incorporating explainable AI (XAI) techniques can enhance interpretability for space researchers and policymakers, making the system more transparent and trustworthy for real-world decision-making. These developments would transform the current solution into a fully automated, intelligent risk monitoring system, capable of supporting next-generation planetary defense initiatives.

8. REFERENCES

NASA. (n.d.). *Asteroids – Near-Earth Object Program.* NASA Jet Propulsion Laboratory. https://cneos.jpl.nasa.gov/

NASA. (n.d.). *NASA Open Data Portal – Near-Earth Object (NEO) Dataset.* Kaggle. https://www.kaggle.com/datasets/sameepvani/nasa-nearest-earth-objects

Milani, A., Chesley, S. R., & Valsecchi, G. B. (2005). Asteroid close encounters with Earth: Risk assessment. *Planetary and Space Science*, 53(3), 211–220. https://doi.org/10.1016/j.pss.2004.09.017

Caruso, A., Miccoli, M., & Di Martino, M. (2020). Asteroid hazard assessment using supervised machine learning. *Astronomy and Computing*, 33, 100414. https://doi.org/10.1016/j.ascom.2020.100414

Radovic, A., et al. (2018). Machine learning at the frontier of high energy physics. *Nature*, 560(7716), 41–48. https://doi.org/10.1038/s41586-018-0361-2

Vereš, P., Farnocchia, D., Chesley, S. R., & Chamberlin, A. B. (2017). Orbit determination strategies for asteroid impact prediction. *Icarus*, 296, 139–149. https://doi.org/10.1016/j.icarus.2017.05.022

Pedregosa, F., Varoquaux, G., Gramfort, A., et al. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
McKinney, W. (2010). Data structures for statistical computing in Python. *Proceedings of the 9th Python in Science Conference*, 51–56.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in Science & Engineering*, 9(3), 90–95.