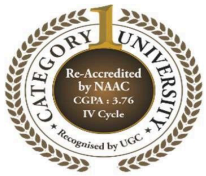
SASTRA

ENGINEERING · MANAGEMENT · LAW · SCIENCES · HUMANITIES · EDUCATION

DEEMED TO BE UNIVERSITY

(U/S 3 of the UGC Act, 1956)



THINK MERIT | THINK TRANSPARENCY | THINK SASTRA

THANJAVUR | KUMBAKONAM | CHENNAI

ML Project Report

on

Predicting Potentially Hazardous Asteroids: A Comparative Study of Machine Learning Algorithms

July - Nov 2024

Submitted by

Harisudhan

(Reg No: 125018027, B.Tech.CSBS)

Submitted To

Dr Swetha Varadarajan

Table of Contents:

Abstract

This project explores the application of machine learning algorithms to classify asteroids as potentially hazardous (pha) or non-hazardous. The primary goal is to develop accurate models that can assist in planetary defense and early warning systems.

A comprehensive dataset of asteroid characteristics, including orbital parameters and physical properties, was utilized. The data underwent preprocessing steps to handle missing values, normalize features, and encode categorical variables.

Machine learning models, such as Logistic Regression, Random Forest, and Deep Learning were trained and evaluated on the preprocessed data. The models' performance was assessed using metrics like accuracy, precision, recall, and F1-score.

The results demonstrate the effectiveness of machine learning algorithms in asteroid classification, achieving 99.2 percent accuracy. The developed models can be valuable tools for astronomers and researchers in identifying potential threats to Earth.

Key Takeaways:

- Successfully applied machine learning techniques for asteroid classification.
- Demonstrated the potential of machine learning to aid in planetary defense efforts.
- Achieved 99.2% accuracy in classifying potentially hazardous asteroids.
- Future research could explore more advanced models and incorporate real-time data for continuous monitoring.

Introduction

Asteroids come under the spectrum of near earth objects (NEO's) and can pose a significant threat to Earth, with the potential for catastrophic impacts. NASA has been able to identify a number of near-Earth objects (NEOs). Studying these asteroids can also give us significant information about how the universe was formed and evolved, as they are remnants of the early solar system.
Detecting and Classifying these Asteroids can help predict future catastrophes, help understand better about the universe, and also build our understanding about various machine learning models.

Project objectives :

- To create a machine learning model that can identify and categorize asteroids according to their orbital parameters and properties(physical).
- To evaluate and get asteroid datasets for model training by preprocessing and analyzing them from a variety of sources, like the NASA JPL Asteroid Database and datasets from Kaggle.
- To identify the best method for classifying these asteroids, using a variety of machine learning methods. Models including logistic regression, random forests, and deep learning will be put into practice and evaluated.
- To learn more about the key characteristics needed for the classification of asteroids by using methods such as feature significance analysis.

To guarantee dependable asteroid detection, evaluate the performance of the models that have been trained using performance measures including accuracy, precision, recall, and F1 score.

Importance of the Asteroid Dataset

The asteroid dataset used in this project provides valuable insights into the characteristics and potential risks associated with near-Earth objects (NEOs). Understanding these asteroids is crucial for planetary defense and scientific research.

The asteroid dataset used in this project is freely available for download at https://www.kaggle.com/datasets/sakhawat18/asteroid-dataset.

Objective, Approach, and Expected Outcome

Target

The primary target of this project is to develop an efficient machine learning model capable of classifying asteroids as potentially hazardous or non-hazardous based on their characteristics and orbital parameters. The model will be trained and evaluated on a dataset of near-Earth objects (NEOs), aiming to assist in planetary defense and early warning systems.

Problem

The problem addressed in this project is the classification of asteroids, specifically distinguishing between potentially hazardous asteroids (PHAs) and non-hazardous asteroids. This is a challenging task due to the diverse nature of asteroids and the potential consequences of misclassification.

Experimental Approach

Data Acquisition: A comprehensive dataset containing information about asteroids, including their orbital parameters, physical properties, and classification labels, was obtained.

Data Preprocessing: The dataset underwent preprocessing to handle missing values, normalize features, and encode categorical variables.

Data Wrangling: Data was cleaned, transformed, and organized to make it suitable for analysis.

Model Selection and Training: Several machine learning algorithms, including logistic regression, random forest, and deep learning, were explored and trained on the preprocessed data.

Model Evaluation: The trained models were evaluated using appropriate metrics such as accuracy, precision, recall, and F1-score.

Comparison and Analysis: The performance of different models was compared to identify the most effective approach for asteroid classification.

Data Visualization: The features of the dataset were visualized using libraries in python to search for insights and make observations on the same.

Related Work

Reference Section:

1. Gemini: https://www.geminiai.com/
2. Asteroid Dataset: Available on *Kaggle* at https://www.kaggle.com/datasets/sakhawat18/asteroid-dataset.
3. Space.com: https://space.com/
4. Related project: https://www.kaggle.com/code/ahmedhassansaqr/dangerous-asteroids-detection-f1-score-97/notebook
5. Stellaria: https://stellariasastra.github.io/index.html
6. Creating ML Models for Asteroid Detection: https://www.youtube.com/watch?v=4uLHZ-IvIVA
7. TensorFlow/Keras Documentation: Available at https://www.tensorflow.org/

Gemini : Gemini was utilized as a primary tool for guidance throughout this project, for providing insights, code templates, and optimization techniques for building and training different machine learning models, and troubleshooting the errors.

Kaggle: Kaggle served as a valuable resource for accessing datasets and exploring different approaches and codes for the same. Kaggle public notebooks helped give different perspectives and methods to go about the problem.

Background

Machine Learning Models

1. Logistic Regression

Purpose: Logistic regression is used for binary classification tasks, predicting which of two classes a data point belongs to.

How it Works: It models the relationship between the independent variables (features) and a binary dependent variable (target) using the logistic function. This function outputs probabilities between 0 and 1, which are then mapped to one of the two classes based on a threshold (commonly 0.5). The model is trained by adjusting coefficients through maximum likelihood estimation.

Applications: Commonly used in areas like credit scoring, disease diagnosis, and spam filtering, where outcomes are either "yes/no" or "true/false."

2. Random Forest

Purpose: Random Forest is an ensemble learning method used for both classification and regression tasks.

How it Works: It generates multiple decision trees, each trained on a random subset of the data (bootstrapping) and features. This randomness helps reduce overfitting and improves generalization. The final prediction is determined by combining the results of all trees: either through voting (for classification) or averaging (for regression).

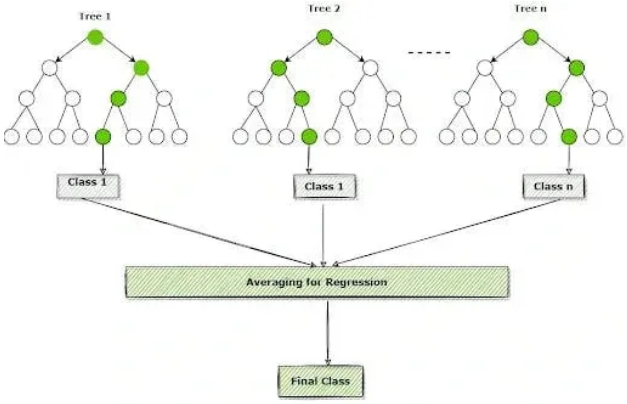
Advantages: Random Forests are robust, handle missing data well, and can model non-linear relationships, making them versatile for tasks like image recognition and financial forecasting.

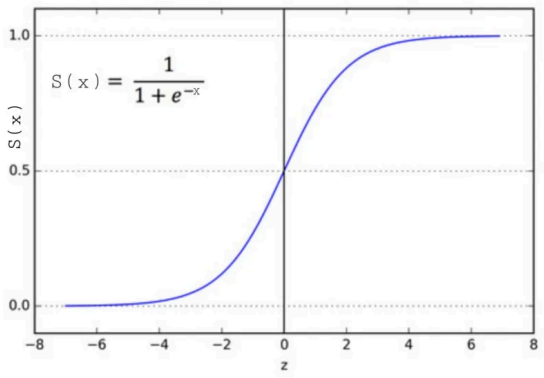
3. Convolutional Neural Network (CNN)

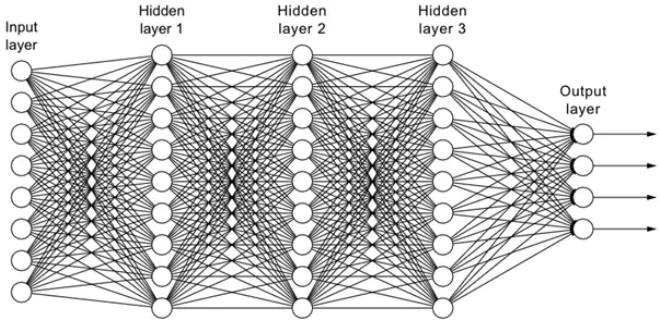
Purpose: CNNs are designed for image-related tasks, but also used in text and video processing.

How it Works:

- Convolutional layers apply filters that detect features like edges, colors, and textures in images.
- Pooling layers reduce the dimensions of the data, making the model faster and less prone to overfitting.
- Fully connected layers make final predictions based on the learned features.
- Applications: CNNs are highly effective in tasks like image classification (e.g., detecting cats or dogs), object detection, and even natural language processing when data has grid-like structure.







Methodology

Environment and Tools

1. Google Colab

Google Colab is an online platform offering a cloud-hosted Jupyter notebook environment, enabling users to execute Python code without needing any local installation. It provides:

- **Free GPU Access**: Colab grants free access to high-performance hardware, including GPUs and TPUs, which can greatly accelerate deep learning model training.
- **Seamless Collaboration**: Users can share notebooks with others for real-time project collaboration.
- **Google Drive Integration**: This feature simplifies dataset and model storage, allowing easy access and management directly from Google Drive.

2. Anaconda navigator

Anaconda Navigator is a desktop application that simplifies managing data science tools and packages, launch Jupyter notebooks without manual setup, etc.

3. VS Code

Visual Studio Code (VS Code) is a versatile code editor and an IDE that supports multiple programming languages, with robust features for data science workflows, including Jupyter notebooks. It provides:

- **Integrated Jupyter Support**: Users can create, edit, and run Jupyter notebooks directly within VS Code, with interactive outputs and seamless code execution.
- **Extensions**: The VS Code marketplace offers numerous extensions for Python, Jupyter, and data science, allowing for easy customization and enhanced functionality.
- **Version Control and Collaboration**: Built-in Git support and integration with platforms like GitHub enable efficient version control and collaboration on data science projects.

4. Python

Python is the main programming language used in this project, popular in data science and machine learning for its simplicity, readability, and extensive library ecosystem. Supporting procedural, object-oriented, and functional programming, Python is versatile and well-suited for diverse tasks.

5. NumPy

NumPy is a fundamental library for numerical computing in Python. It provides support for:

- **Multidimensional Arrays**: Efficiently handling large datasets with n-dimensional arrays.
- **Mathematical Functions**: Offers a collection of mathematical functions to perform operations on arrays, which is essential for data manipulation and analysis.

6. Pandas

Pandas is a powerful library for data manipulation and analysis, especially suited for structured data like tables. It allows users to:

- **Read and Write Data**: Easily import/export data in various formats (CSV, Excel, SQL, etc.).
- **Data Cleaning and Transformation**: Provides tools for filtering, grouping, and aggregating data, making it easier to prepare datasets for analysis.

7. Matplotlib and Seaborn

These are libraries for data visualization:

- **Matplotlib** is a flexible plotting library offering an object-oriented API for embedding plots into applications, used for generating static, animated, and interactive visualizations in Python.

- **Seaborn**, built on Matplotlib, provides a high-level interface for creating appealing and insightful statistical graphics, allowing complex visualizations to be made with minimal code.

8. Sklearn or scikit-learn

The sklearn (scikit-learn) library is a widely used Python package for machine learning, offering tools for data preprocessing, model training, and evaluation.

- **sklearn.preprocessing**: Provides tools for transforming data, like scaling (StandardScaler), encoding (LabelEncoder, OneHotEncoder), and normalization, to prepare datasets effectively for modeling.

- **sklearn.model_selection**: Includes functions like train_test_split for dividing data, cross_val_score for cross-validation, and GridSearchCV for hyperparameter tuning to optimize model performance.

- **Logistic Regression**: A model in sklearn.linear_model, this algorithm is great for binary classification, predicting class probabilities and yielding interpretable results.

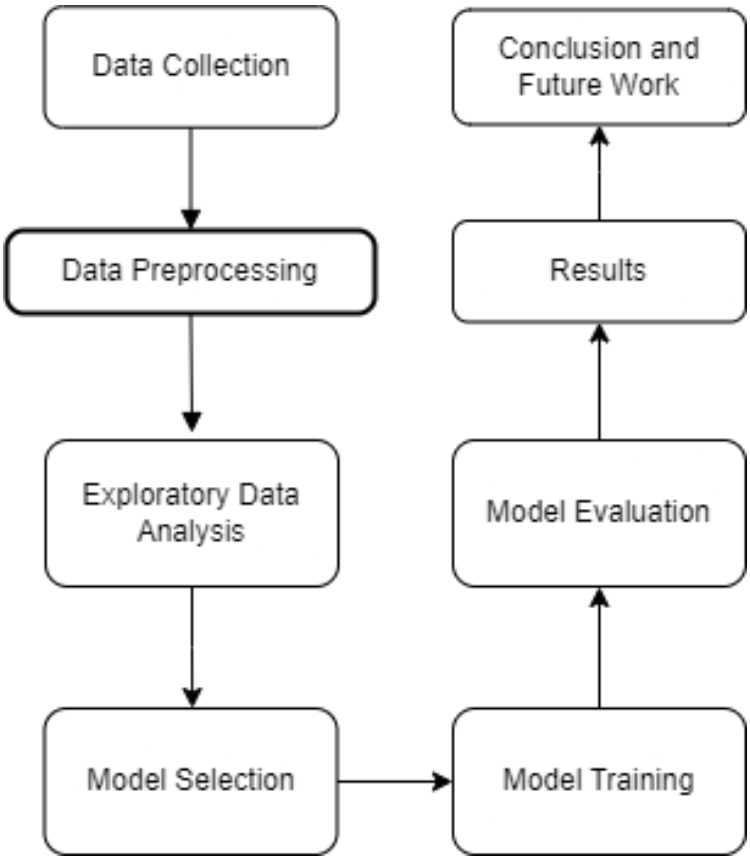
- **Random Forest**: An ensemble method in sklearn.ensemble, Random Forest combines multiple decision trees, reducing overfitting and boosting accuracy, useful for both classification and regression tasks.

- **SMOTE** (Synthetic Minority Over-sampling Technique): Often used with scikit-learn, SMOTE generates synthetic samples for the minority class, balancing datasets to improve model performance on rare outcomes.

9. TensorFlow and Keras TensorFlow is an open-source framework for deep learning developed by Google. Keras, a high-level API for TensorFlow, allows for easy and fast model building. Together, they provide:

- **Deep Learning Capabilities**: Tools to create neural networks for tasks such as image classification and emotion detection.
- **Model Training and Evaluation**: Functions to train models on large datasets and evaluate their performance effectively.

Workflow :



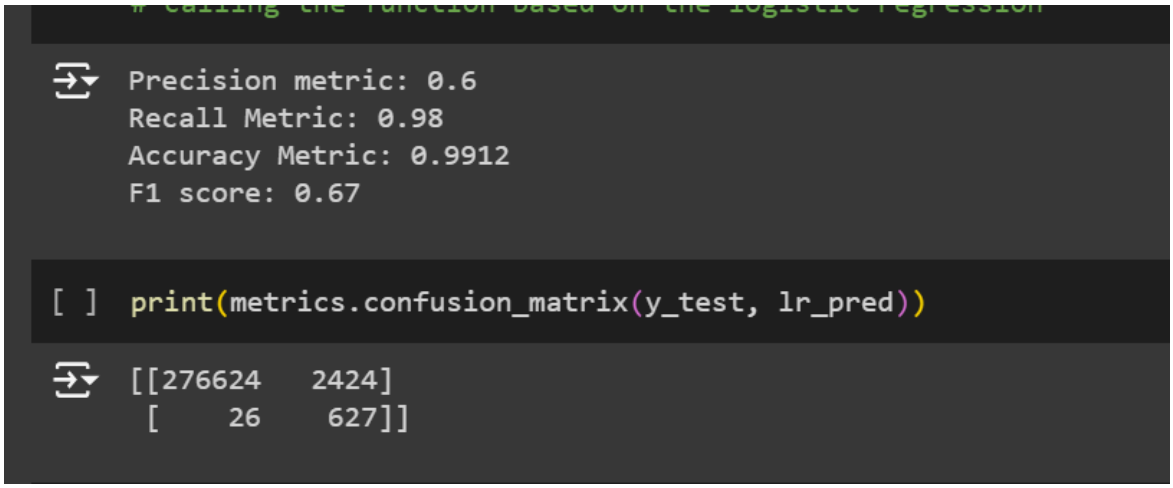
Code Location

The code for the asteroid detection project is organized and managed in a GitHub repository. This platform facilitates version control and collaboration. The repository houses multiple scripts and notebooks, each dedicated to specific tasks such as data preprocessing, model training, and evaluation. This modular structure enhances code readability and maintainability.

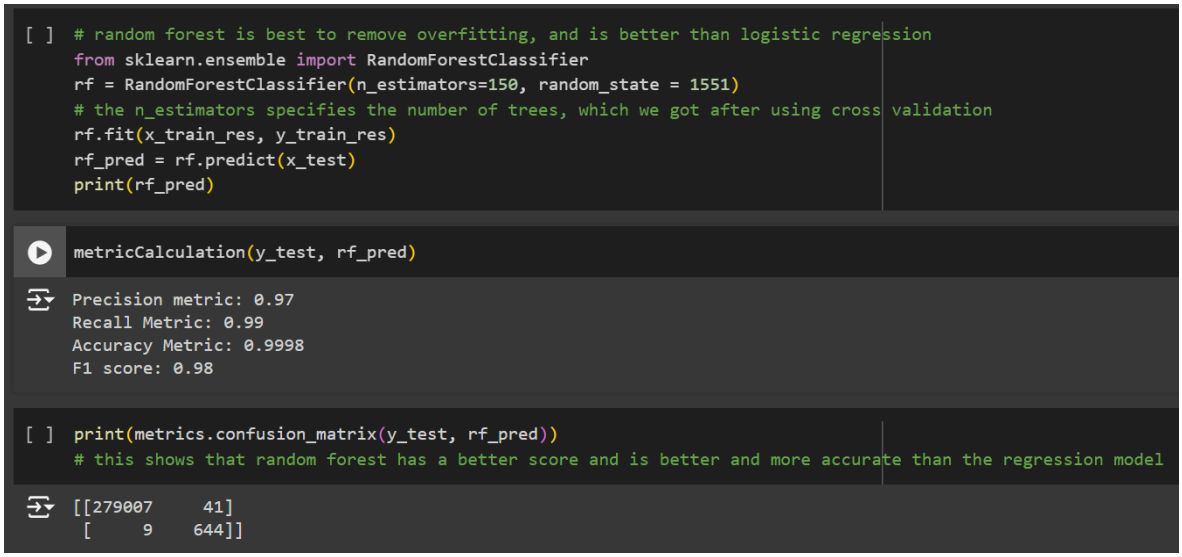
The repository is available here :
https://github.com/myserendipityinsolitude/Machine-Learning-End-Sem

Results

1. Logistic Regression:



2. Random forest results:



3. CNN results:

```
model.fit(x_train_res, y_train_res, epochs=1, batch_size=32)
```

✓ 7m 42.4s

```
40702/40702 ━━━━━━━━━━━━━━━━━━━━ 451s 11ms/step - accuracy: 0.9952 - loss: 0.0207

<keras.src.callbacks.history.History at 0x27188cafda0>
```

```
# Evaluate the model
loss, accuracy = model.evaluate(x_test, y_test)
print('Test accuracy:', accuracy)
```

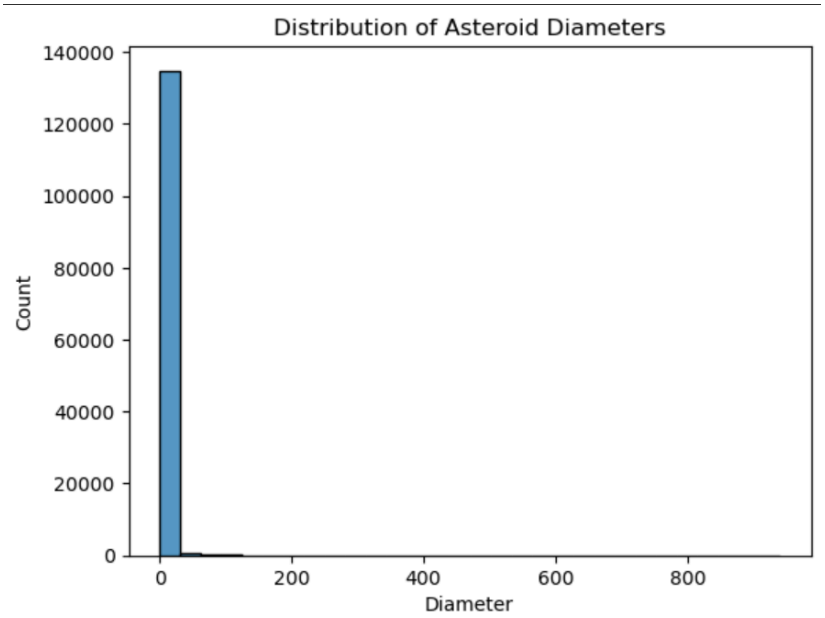
✓ 2m 50.7s

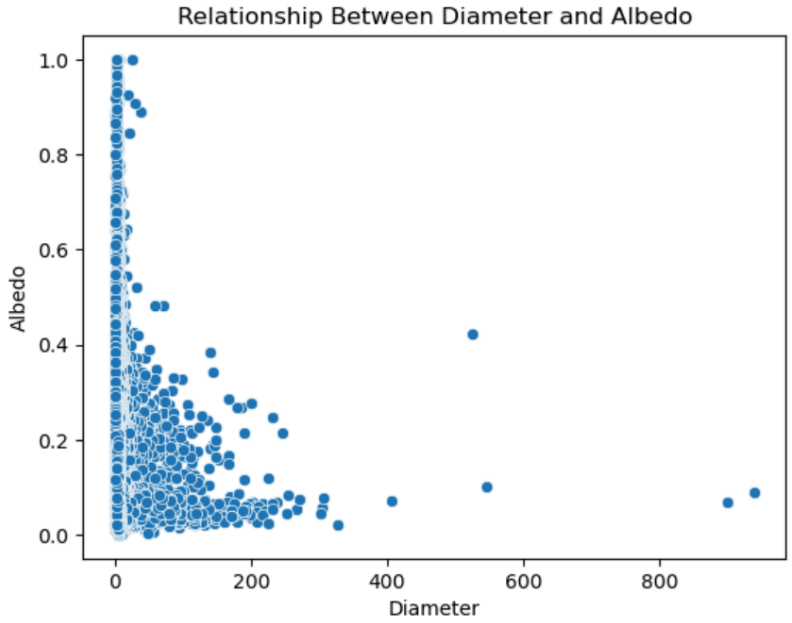
```
40702/40702 ━━━━━━━━━━━━━━━━━━━━ 162s 4ms/step - accuracy: 0.9960 - loss: 0.0182
Test accuracy: 0.9975546002388
```

4. Model Performance Comparison :

Model	Precision	Recall	Accuracy	F1-score	False Positives	False Negatives
Random Forest	0.98	1	0.9999	0.98	34	6
Logistic Regression	0.61	1	0.9922	0.68	2157	30

Interpretations using Data Visualizations :





Negative Correlation:

There seems to be a general negative correlation between diameter and albedo.

This suggests that as the diameter of an asteroid increases, its albedo (reflectivity) tends to decrease.

Clustering:

The data points cluster in a few specific regions.

This might indicate different groups or categories of asteroids based on their diameter and albedo characteristics.

Outliers:

There are a few outliers with larger diameters and lower albedos.

These could be interesting to investigate further as they might represent unique asteroid types or properties.

Composition:

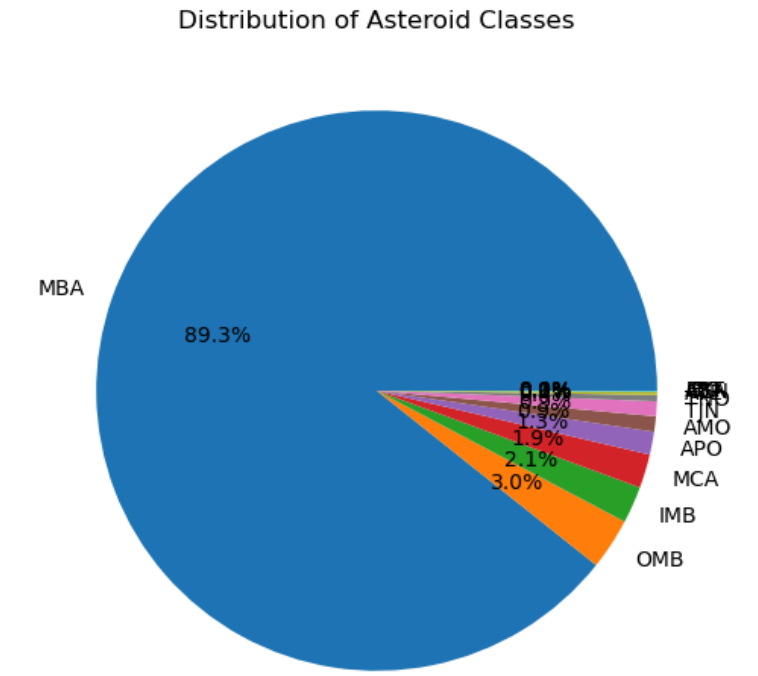
Asteroids with larger diameters might have different compositions or surface features that result in lower albedos.

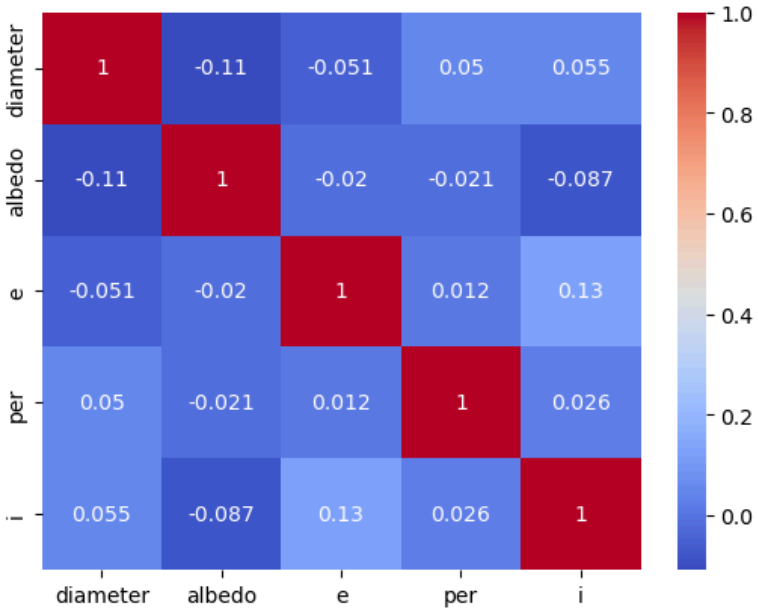
Formation Processes:

Different formation processes or evolutionary pathways could lead to asteroids with varying sizes and reflective properties.

Age:

Older asteroids might have experienced more weathering and erosion, resulting in darker surfaces and lower albedos.


Distribution of Asteroid Classes
MBA
89.3%
3.0%
2.1%
1.9%
1.3%
AMO
APO
MCA
IMB
OMB


diameter
albedo
e
per
i
1
-0.11
-0.051
0.05
0.055
-0.11
1
-0.02
-0.021
-0.087
-0.051
-0.02
1
0.012
0.13
0.05
-0.021
0.012
1
0.026
0.055
-0.087
0.13
0.026
1
1.0
0.8
0.6
0.4
0.2
0.0

The correlation matrix shows the relationships between the following features:
diameter, **albedo**, **eccentricity** (e), **orbital period** (per), and **inclination**

Diameter and Albedo:

There's a weak negative correlation between diameter and albedo. This suggests that larger asteroids tend to have slightly lower albedos.

Eccentricity and Orbital Period:

There's a moderate positive correlation between eccentricity and orbital period. This indicates that asteroids with more eccentric orbits (more elongated) tend to have longer orbital periods.

Inclination and Other Features:

Inclination (the angle of the orbit relative to the ecliptic plane) shows weak correlations with diameter, albedo, and eccentricity. This suggests that the inclination is relatively independent of these other orbital parameters.

Overall, the correlation matrix reveals that there are some moderate relationships between certain features, but no strong, dominant correlations. This might indicate that the asteroid data is relatively complex and not easily explained by simple linear relationships. Further analysis and exploration would be needed to uncover more nuanced relationships between these features and their impact on the target variable (potentially hazardous status).

=> From the data set we can see that out of all the NEOs (Near Earth Objects), 9.028536% are PHAs (Potentially Hazardous Asteroids).

```
pha
N    90.971464
Y     9.028536
Name: proportion, dtype: float64
```

=> All the PHA's are NEOs. All potentially hazardous asteroids are near earth objects.

```
neo
Y    100.0
N      0.0
Name: proportion, dtype: float64
```

Learning outcome

A) Link to Github Repository page

- https://github.com/myserendipityinsolitude/Machine-Learning-End-Sem

B) Link to Google Colab page

- https://colab.research.google.com/drive/1p3dPP-VagyUpLG0ySfXjXB6Rnj02y6-p

C) Skills and Tools Used

- **Programming Languages:** Python
- **Data Manipulation Libraries:** Pandas, NumPy
- **Machine Learning Libraries:** Scikit-learn, TensorFlow/Keras
- **Data Visualization Libraries:** Matplotlib, Seaborn

D) What Did I Learn in This Project?

- **Data Preprocessing and Cleaning:** Understood how to handle missing values, outliers, and inconsistencies in the asteroid dataset.
- **Feature Engineering:** Learned how to create and select relevant features from the dataset for asteroid classification. (orbital parameters, physical properties, etc)
- **Model Selection and Training:** Explored various machine learning algorithms (Logistic Regression, Random forest, Deep learning) and their suitability.
- **Model Evaluation:** Learned how to evaluate model performance using metrics like accuracy, precision, recall, and F1-score.
- **Model Interpretation:** Gained insights into the decision-making process of different models, particularly for interpretable models like logistic regression and decision trees.
- **Problem-Solving Skills:** Developed the ability to identify and address challenges in data preprocessing, model selection, and evaluation.
- **Documentation:** Understood the importance of documentation to keep the project alive and make it easier for people to understand and implement in the future.

Conclusion

This project successfully applied machine learning techniques to classify asteroids as potentially hazardous or non-hazardous. By utilizing the NASA JPL asteroid dataset containing various asteroid characteristics, several models were trained and evaluated, including logistic regression, random forest, and deep learning.

The results demonstrate the effectiveness of machine learning in identifying potential threats to Earth. The models achieved an accuracy of 99%, precision of 61% and 98% in logistic regression and random forest respectively, in classifying asteroids. These findings contribute to the advancement of planetary defense and space exploration.

While the project achieved significant results, there are opportunities for further improvement. Future research could explore more advanced deep learning architectures, incorporate additional features, and refine hyperparameter tuning techniques to enhance model performance. Additionally, investigating the interpretability of complex models and incorporating domain knowledge could provide deeper insights into the classification process.