University of San Diego
SHILEY-MARCOS SCHOOL OF ENGINEERING

ADS508 Data Science With Cloud Computing

Authors: Anahit Shekikyan, Taylor Kirk, Alli McKernan

Company Name: Space Waste

Company Industry: Aerospace and Environmental Technology

Company Size: Startup

Abstract

Space debris is becoming an increasingly urgent issue, threatening both current and future space missions. This hypothetical startup company is dedicated to leveraging data science to analyze, track, and predict the impact of space debris while identifying high-risk areas for cleanup. With the support of organizations such as NASA, this initiative contributes to space sustainability through data-driven solutions that will help keep Earth's orbit safe for exploration and innovation.

Problem Statement

The growing amount of inactive satellites and debris in Earth's orbit poses a significant challenge to space operations. The concept known as "Kessler Syndrome" describes the risk of a chain reaction where collisions between defunct satellites create more debris, leading to an uncontrollable cascade of destruction. This issue not only endangers operational spacecraft and the International Space Station (ISS) but could also disrupt critical communication and navigation systems on Earth.

Unaddressed, the issue may render certain orbits unusable and escalate the cost and risk of future missions. This project proposes a data-driven approach to assess debris accumulation, forecast potential collisions, and identify high-risk orbital zones. Utilizing machine learning and analytics, the project supports NASA and other agencies in risk mitigation and in promoting the long-term sustainability of Earth’s orbital environment.

Goals

- Analyze historical trends from 1960 to the present using NORAD and ESA data to establish the rate and scale of debris accumulation, helping stakeholders understand the urgency of mitigation efforts.
- Reduce projected debris growth in LEO by 15% over the next 10 years through predictive modeling, enabling organizations to allocate resources for collision avoidance and debris mitigation.
- Identify the top 5 highest-density debris fields with a noise rate of less than 50% using clustering techniques, providing insights to satellite operators and policymakers on areas with the highest collision probability to minimize operational disruptions and financial losses.
- Achieve an R^2score of at least 0.70 when predicting the collision likelihood of lost objects, based on Object Class, Shape, and Cross-Sectional Area, improving risk assessment and enhancing safety measures for active satellites and space missions.
- Provide NASA with data-driven recommendations on which objects to prioritize for removal, optimizing debris mitigation strategies to reduce high-risk object density by 20% while minimizing costs. This target is based on expected improvements in predictive modeling and clustering techniques compared to current debris mitigation efforts, which have achieved approximately a 10% reduction in high-risk object density.

Non-Goals

- Physically removing space debris (focusing on data analysis and predictive modeling).
- Addressing issues related to terrestrial waste management.
- Establishing legal policies regarding space debris cleanup.
- Providing real-time satellite tracking.
- Managing commercial space traffic systems.

Data Sources

1. **Satellites and Debris in Earth's Orbit:** Data from space-track.org API and the European Space Agency's API (DISCOSweb).
2. **Objects in low earth orbit (LEO):** Data on active and inactive objects in low Earth orbit (LEO), sourced from Space-Track.org.
3. **Collection of lost objects:** Objects for which no GP data has been recorded for 30 or more days CelesTrak

4. **Collision data from Celestrek:** Data on reported collision windows between objects and prediction scores

All data will be stored securely in an AWS S3 bucket, with access managed through IAM roles. AWS Glue and SageMaker will be used for data processing.

Data Exploration

S3 Bucket and Paths

- **AWS S3 Bucket Name:** sagemaker-us-east-1-266917422334

- **Data Paths**
  - Raw ESA data: "s3://sagemaker-us-east-1-266917422334/data/raw_data/full_esa_folder/"
  - Satellite catalog: "s3://sagemaker-us-east-1-266917422334/data/raw_data/satcat_folder/"
  - LEO objects: "s3://sagemaker-us-east-1-266917422334/data/raw_data/leo_objects_folder/"
  - Lost objects: "s3://sagemaker-us-east-1-266917422334/data/raw_data/lost_objects_folder/"
  - Collision data: "s3://sagemaker-us-east-1-266917422334/data/raw_data/simplified_collision_folder/"

- **Data Ingestion Workflow**
  - **Manual Upload:** Initial datasets are manually uploaded via AWS S3 Console.
  - **Automated API Ingestion (AWS):** AWS Glue extracts data from APIs (e.g., Space-Track.org, CelesTrak APIs daily, ensuring up-to-date information).
- **Versioning and Security**
  - S3 enables versioning to maintain historical datasets.
  - IAM policies restrict access to authorized users.
  - Data is encrypted using ASW KMS (Key Management Service)

GitHub: https://github.com/tkbarb10/ADS508-GroupProject

Exploratory Data Analysis

To support the stated project goals, a total of five datasets were utilized, comprising 252,035 records and 83 features. Much of the data included overlapping, missing, or irrelevant information; therefore, part of the preparation process involved identifying features to eliminate and applying appropriate methods for handling missing values.

Key fields identified as essential to the analysis included Object Names, NORAD Category IDs, Object Class, Launch Dates, Decay Dates, physical size-related features, location-related variables, and recorded collision probabilities. Overlapping or inconsistently labeled features (e.g., Object_Type in one dataset and objectClass in another) were standardized to prevent duplication and ensure compatibility for dataset joins.

Bias Exploration

Two project objectives required predictive modeling: estimating the growth of objects in low Earth orbit (LEO) and predicting the collision probabilities of lost objects. For modeling LEO object growth, the target variable was Object Class. Two key sources of bias needed to be addressed in this context: class imbalance and survivorship bias. The current LEO dataset includes four object classes, with two classes representing approximately 94% of the data. Additionally, the dataset only reflects objects still in orbit, contributing to survivorship bias.

When developing a predictive model for collision probabilities of lost objects, additional forms of bias—specifically variable bias, data leakage, and selection bias—were also considered. The lost objects and collision datasets were distinct; the collision dataset contained information not present in the lost objects dataset. Furthermore, the lost objects dataset included satellites for which no positional data had been recorded for 30 days or more, while the collision dataset encompassed all documented collision events, creating the potential for partial overlap. Lastly, because the training data may not accurately reflect the characteristics of truly lost objects, selection bias was a concern during model development.

Data Preparation

Data Cleansing Techniques

Handling Missing Values: The dataset may contain missing or null values, which can adversely affect analysis and modeling. Implementing data imputation techniques, such as filling missing values with

the mean, median, or mode, or using more sophisticated methods like k-nearest neighbors imputation, can help maintain data integrity.

Removing Duplicates: Duplicate records can skew analysis results. Identifying and removing such duplicates ensures that each observation is unique, leading to more accurate insights.

Addressing Outliers: Outliers can distort statistical analyses and model performance. Detecting and handling these anomalies, either by transformation or removal, can lead to more robust models.

Feature Selection

Five separate datasets were sourced and the feature selection process varied on each of them to prepare them for modeling, joining, and visualization.

Full European Space Agency (ESA) catalogue:

It was decided to remove the following features: ['vimpelId', 'satno', 'predDecayDate', 'active', 'mission', 'shape', 'mass', 'width', 'height', 'depth', 'xSectMax', 'xSectMin']. VimpelId and satno were object identification information from other sources and therefore redundant. Shape, mass, width, height, depth, predDecayDate, and mission do not align with the objectives of the project and were therefore also removed. The active feature contained too many missing values, and lastly, the xSectAvg feature is available in the data which better suits the purposes of the project than xSectMax and xSectMin.

Low Earth Orbit (LEO) object data:

The following features were selected for removal: ['DECAY', 'COMMENT', 'COMMENTCODE', 'FILE', 'LAUNCH_YEAR', 'LAUNCH_NUM', 'RCSVALUE', 'OBJECT_NUMBER', 'OBJECT_ID, 'OBJECT_NAME, 'CURRENT']. DECAY, COMMENT, and RCSVALUE were empty features and provided no useful information. COMMENTCODE, LAUNCH_NUM, FILE, and LAUNCH_PIECE do not align with the objectives of the project. OBJECT_NUMBER, OBJECT_ID, and OBJECT_NAME provide the same information as other available features and were therefore removed. CURRENT is all the same value and therefore provides no useful information for modeling.

Full Satellite Catalogue (SatCat) data:

The following were selected for removal: ['DATA_STATUS_CODE', 'RCS', 'OPS_STATUS_CODE']. The code features are unrelated to the objectives of this project. RCS stands for Radar Cross Section. There were two features available in the data that provided this information. RCS is the numerical estimate while the other feature represents the numerical estimates organized into ordinal categories. For the purposes of

the project, it was decided to use the categorical feature. Lastly, the ORBIT_CENTER feature contains various labels indicating the orbital centers of the objects. Since the project focuses only on objects orbiting Earth, the dataset was filtered to include only those with an ORBIT_CENTER value of Earth, after which the feature was removed.

Collision and Lost Object data:

These datasets were available and sourced almost entirely in the format required for modeling, so no feature selection was required.

Feature Creation

Extracting Active Status: The object names in the collision dataset include indicators like '[+]' and '[-]' denoting their active status. By parsing these indicators, a new feature, 'Activity_Status', can be created to represent the active state of each object. This enhances the dataset by explicitly capturing this important attribute.

Combining Related Features: Certain fields, such as 'TCA_RANGE' and 'TCA_RELATIVE_SPEED', could be combined to create a new feature that encapsulates the relative motion dynamics between objects. This synthesized feature might offer better insights into collision probabilities.

Temporal Features: The 'TCA' (Time of Closest Approach) field can be decomposed into components like year, month, day, or even specific time intervals (e.g., morning, afternoon) to identify temporal patterns in collision occurrences.

Feature Transformation

Similar data transformation steps were applied across all datasets to prepare them for visualization and modeling. Date fields such as Decay Date and Launch Date were originally imported as object data types, so they were converted to proper datetime formats. Many numerical features contained missing values, which were imputed using the median to reduce the impact of outliers. These features were then standardized to support modeling.

Some identifier features, like NORAD_CAT_ID, were numeric but represented categorical information. Since they were incorrectly read as integers, they were converted to strings for accurate

representation.

To ensure consistency across datasets, some column values were standardized. For example, in the Lost Object data, the Object Type feature used abbreviations like "PAY" for Payload and "R/B" for Rocket Body, whereas other datasets used full labels such as "PAYLOAD" and "ROCKET BODY." These discrepancies were corrected for consistency.

In the Satellite Catalogue data, location-related features such as Apogee, Inclination, and Perigee retained their last recorded values even for objects that had already decayed and were no longer in orbit. Since these values could interfere with modeling objectives, the data was first filtered by the Orbit Type feature to identify objects that had impacted or landed. For these objects, the location values were set to zero. Then, the dataset was filtered to include only active, in-orbit objects, and any remaining missing location values were imputed using the median. Finally, all numerical features were standardized.

Data Balancing and Splitting Strategy

To ensure fair and effective model training, each dataset was carefully examined for class imbalance. In real-world datasets, especially those involving satellite object classifications such as debris, payloads, and rocket bodies—certain classes often dominate. To prevent the models from being biased toward majority classes, appropriate balancing and splitting strategies were implemented.

To assess the impact of balancing, visualizations were created before and after applying SMOTE (Synthetic Minority Over-sampling Technique) using bar plots. These visualizations allowed for a clear comparison of class distribution. This process not only validated the effectiveness of the balancing strategy but also confirmed that each dataset was adequately prepared for model training.

Balancing the Dataset

The class imbalance was especially noticeable in datasets like loe_objects_df, where the "DEBRIS" class had significantly more instances compared to underrepresented classes like "PAY+", "R/B", and "Unknown". Such imbalances can bias the model to favor the dominant class, reducing its ability to detect critical but rare events.

To address this SMOTE (Synthetic Minority Over-sampling Technique) was applied, which generates synthetic samples for the minority class based on their nearest neighbors. This helps the model learn balanced decision boundaries and improves generalization during training.

Why SMOTE?

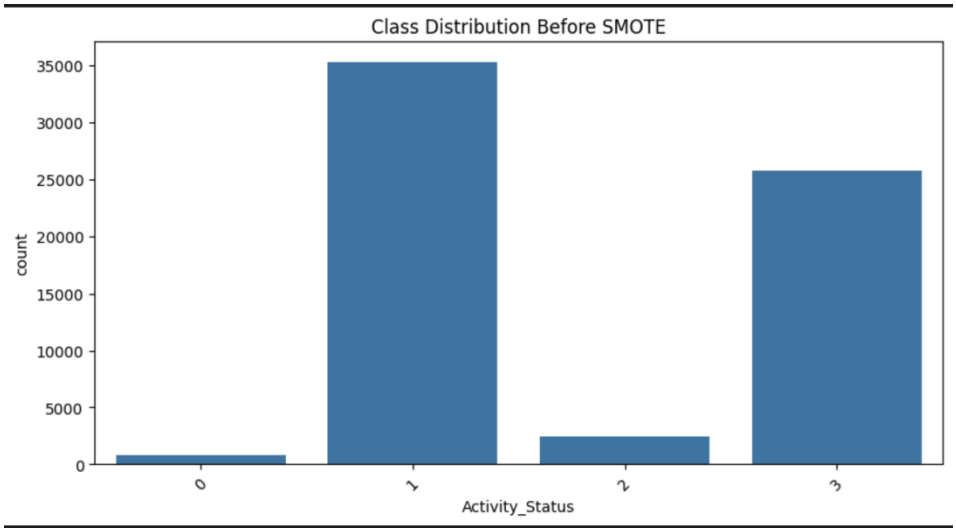
SMOTE was selected over random oversampling or undersampling for several reasons:

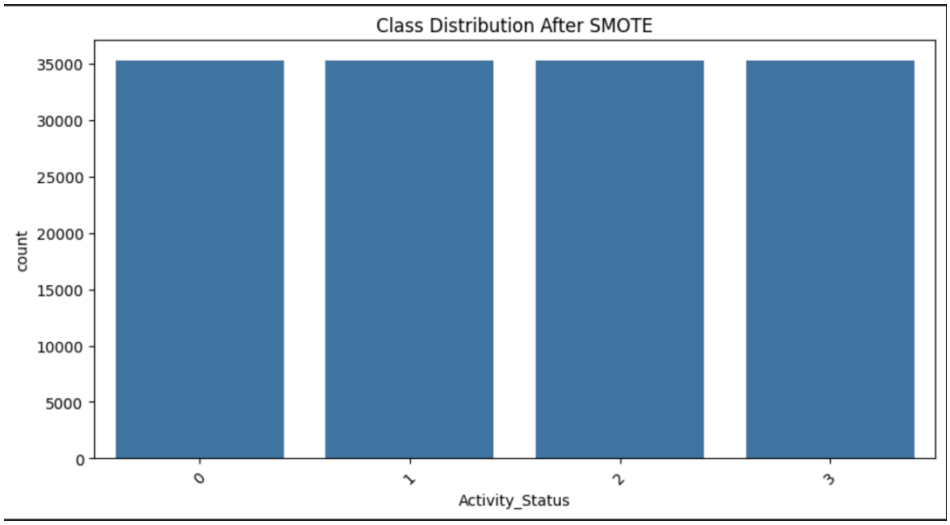
- It avoids loss of data that occurs with undersampling.
- It doesn't simply duplicate records but instead creates new, realistic samples.
- It is well-suited to structured tabular data, which is the nature of our satellite datasets.

Balancing was applied only to the training set, to avoid data leakage and ensure fair model evaluation on untouched test data.

Collision Data (cleaned_collision_data.csv)

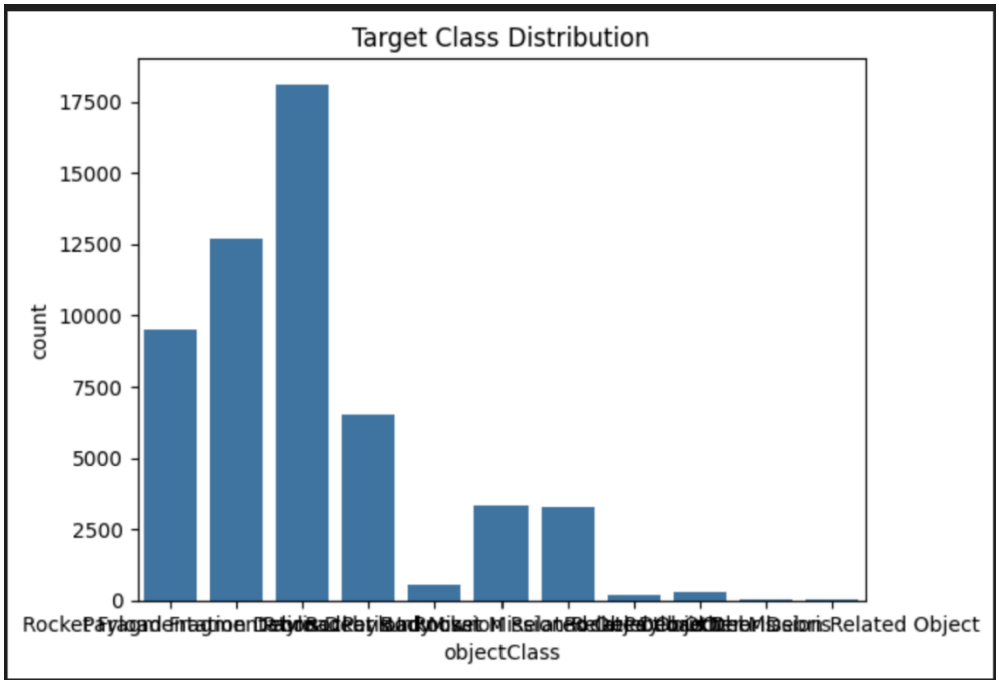
This dataset had fewer instances of collisions than non-collisions. The target variable Activity_Status included class labels ranging from 0–3, with a noticeable imbalance. SMOTE was applied to the training set to improve the detection of active vs. inactive collision states and prevent the model from defaulting to majority predictions.

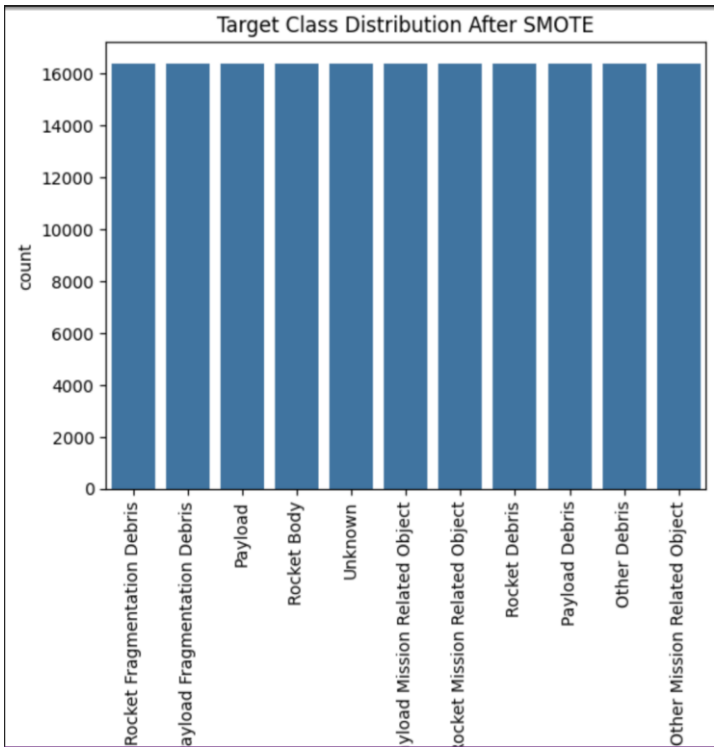




ESA Satellite Data (cleaned_esa.csv)

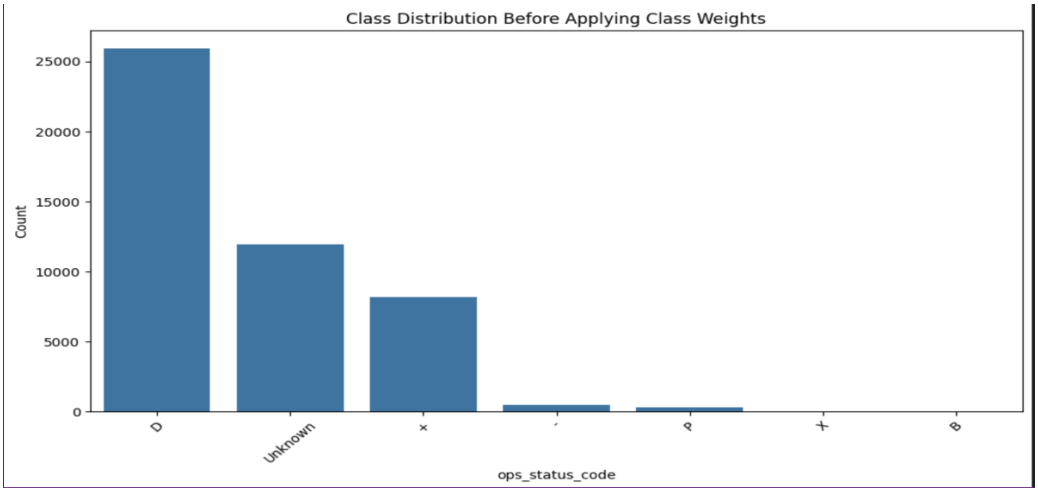
Here, the target variable was objectClass, with 11 categories. Some classes, like “Other Debris,” had as few as 52 records. To address this, SMOTE with k_neighbors=1 was applied, allowing effective balancing even with small minority class sizes. This helped the model better identify operational vs. non-operational status, essential for failure prediction.

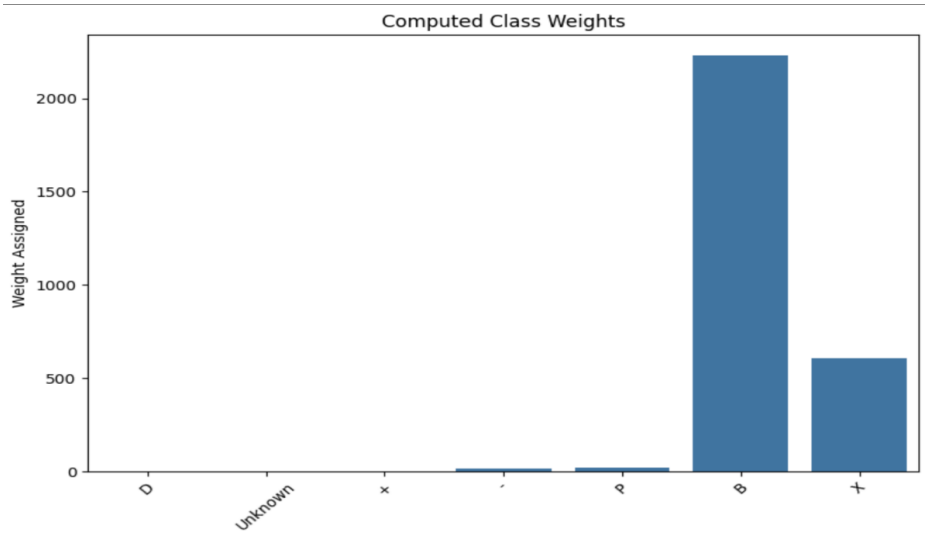




Full SATCAT Data (full_satcat_cleaned.csv)

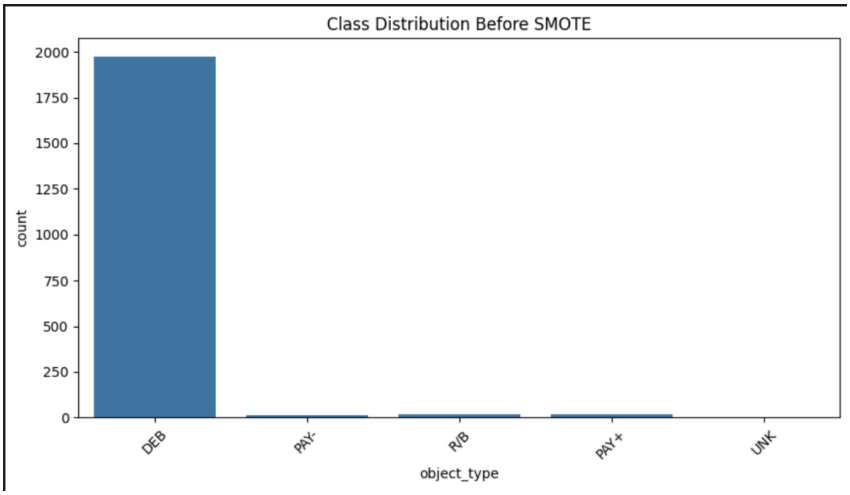
In this case, the target variable was ops_status_code. Rather than using SMOTE, class weighting was used during model training to give more importance to rare categories without modifying the dataset itself.

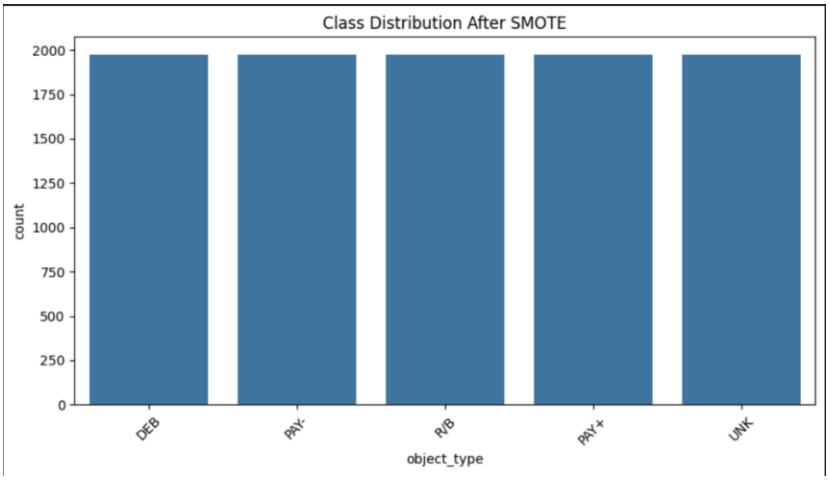




Lost Objects Data (lost_objects_cleaned.csv)

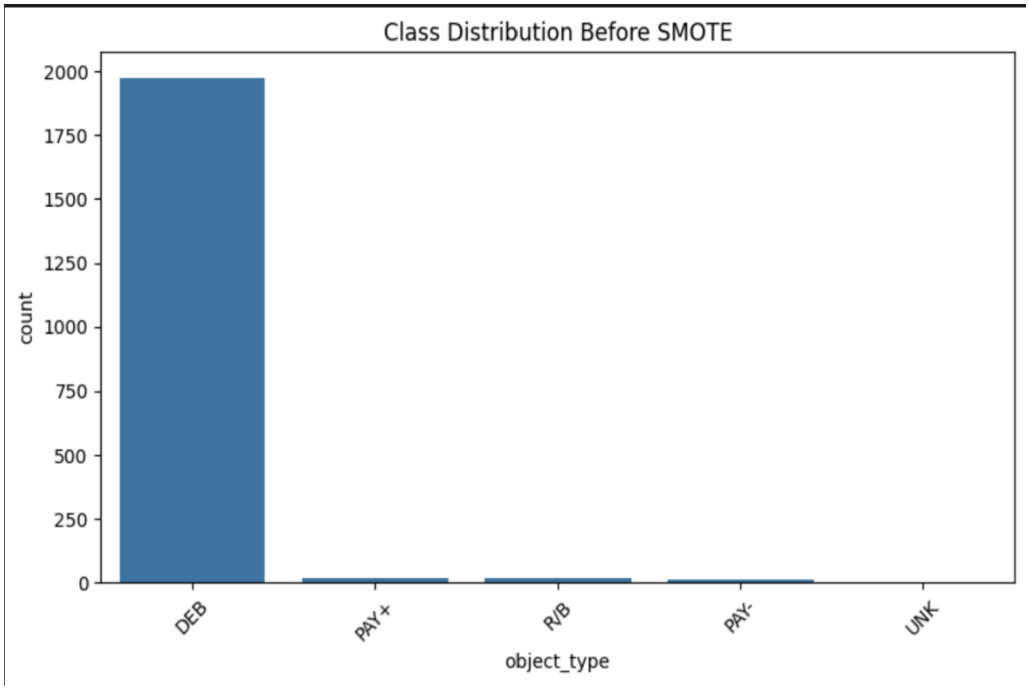
This dataset records satellites that lost contact. The target variable was object_type, and a significant imbalance existed. SMOTE was used for categories with extremely low counts (less than 6), and k_neighbors were reduced to generate viable synthetic records. One-hot encoding was also applied to prepare categorical features before training.

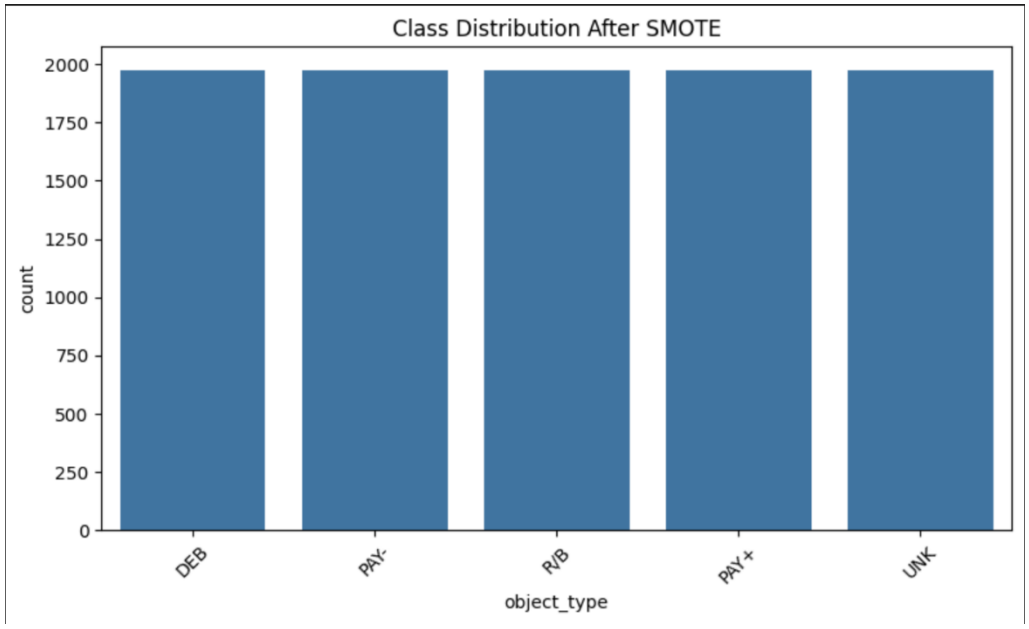




LEO Objects Data (leo_objects_cleaned.csv)

The imbalance was present in mission-critical labels associated with low-Earth orbit satellites. Using SMOTE helped the model learn from rare but important cases like object failure or anomalies, increasing model sensitivity in real-world conditions.





All balanced datasets were stored in the shared Amazon S3 bucket: s3://sagemaker-us-east-1-738272695228, which is now being used for training models in AWS SageMaker Studio.

Data Splitting Strategy

For all five datasets, an 80/20 train-test split was applied, with stratification based on the target variable. Stratified splitting ensures that the class distribution in the training and testing sets reflects the original data, which is especially important in cases of class imbalance.

Why 80/20?

The 80/20 ratio is a widely accepted standard in the data science community. It provides a good balance by:

- Giving the model access to a large portion of the data (80%) for training and learning complex patterns.
- Preserving a significant portion (20%) for testing, ensuring a reliable and unbiased evaluation of model performance.

Stratification was particularly crucial for datasets such as loe_objects_df, collision_df, and esa_df, where class imbalance was present. Without stratification, the test set didn't include any examples of rare classes, leading to misleading performance metrics.

Handling Data Leakage

SMOTE was applied only to the training set after splitting to prevent data leakage. This is the best practice because applying oversampling before splitting could allow synthetic data to leak into the test set, causing overfitting and inflated performance metrics. By restricting SMOTE to the training portion, the model evaluation remains realistic and representative of real-world deployment.

Data Training

Training Method

For this phase, a combination of Sagemaker built-in-algorithm's, bring your own algorithm and AutoML available in Amazon SageMaker Studio was utilized. For predicting debris growth in LEO, Sagemakers XGBoost was utilized because of its seamless integration with the Sagemaker ecosystem and a

better performance with larger datasets. For the clustering of high-density object fields, we chose Python's hdbscan method as density clustering is not natively supported by Sagemaker. For the predictive modeling component (e.g., estimating collision probability), we applied SageMaker AutoML to determine the most suitable regression model with minimal manual tuning.

Model Training (Debris Growth)

The LEO is preprocessed and split 80/20 to prepare for training, then uploaded to S3. The XGBoost Estimator object was then created, and hyperparameters set before fitting the modeling to the train and test data.

Parameters Passed

The following were chosen as a result of model tuning.

- num_round=100
- max_depth=5
- eta=0.2
- gamma=4
- min_child_weight=6
- subsample=0.8

Instance Size/Count

The instance ml.m5.large was used for the XGBoost algorithm. this is a general purpose instance count and it was found that a larger instance was unnecessary.

Model Evaluation

The model was found to have an accuracy of 93.70%, exceeding the desired benchmark. Model was deployed via AWS Sagemaker to test incoming predictions then the endpoint was deleted to save on resources until needed.

Model Training (Debris Clustering)

Python's built in hdbscan module was used as Sagemaker does not natively support density clustering, and HDBSCAN is more robust to outliers and discovering irregular clusters than DBSCAN is.

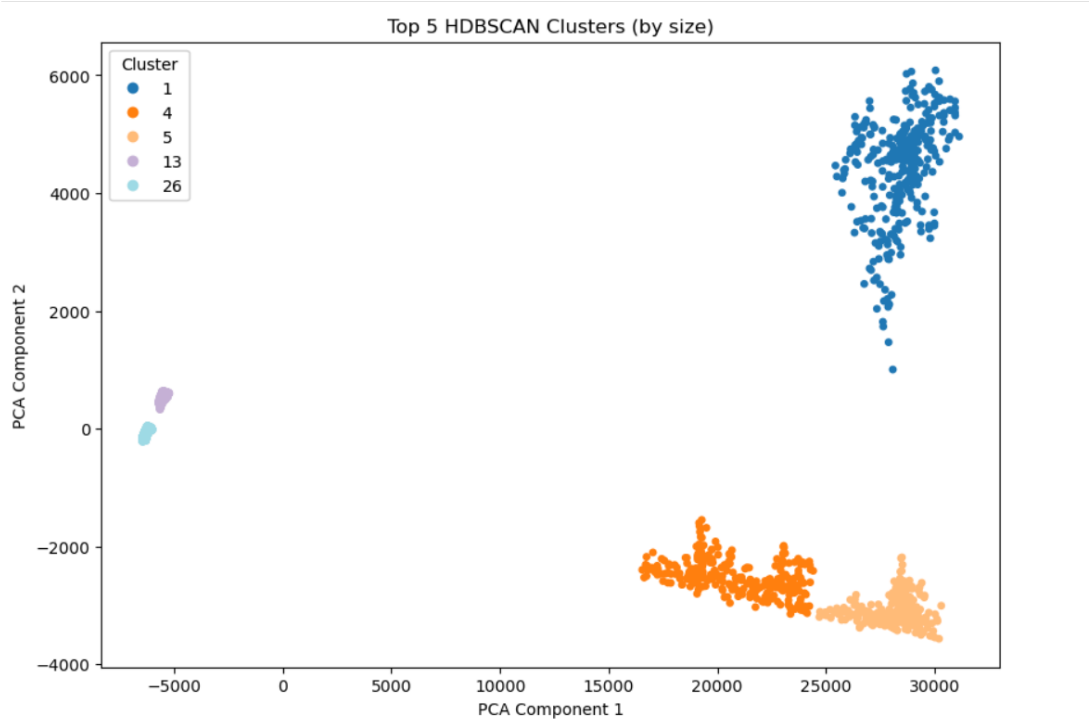
HDBSCAN Parameters Passed

Parameters determined using a grid search to find the parameters that resulted in the lowest noise rate (percent of objects not aggregated into a cluster).

- Min_cluster_size: 50
- Min_samples: 5
- Epsilon: 0

Model evaluation

Noise Rate was used to evaluate the validity and usefulness of the clusters. As this type of clustering is an unsupervised training task, the traditional metrics of accuracy or R^2 values are not useful here. A high noise rate would indicate most of the objects were unable to be separated into meaningful clusters. The final model resulted in 33 clusters and a noise rate of 41.74%, which was below the desired rate of 50%. The top 5 clusters were then identified by count, and a PCA decomposition was used on the data to visualize the clusters.



Model Training (collision probability)

For making predictions on the collision probability with the lost object data, we are using AutoML.

This particular goal is a little more lofty than the others, and there is uncertainty as to what the best model would be to make these predictions, if it even is possible. So AutoML will be used to quickly test several models to determine any that could be viable.

Algorithm

AutoML selects the best algorithm among the three candidates automatically. This saves time and allows for unbiased model selection.

Parameters Passed

- For AutoML:
    - ContentType: parquet (the dataset was saved as Parquet for better data type preservation)
    - ProblemType: Regression
    - AutoMLObjective: RMSE

Instance Size/Count

The instance ml.m5.xlarge was used for AutoML training jobs. This instance type includes 4 vCPUs and 16 GB RAM, providing an optimal balance of compute power and cost-efficiency. It was appropriate given our dataset of ~250,000 records and moderate feature dimensionality. The choice ensures quick training times without incurring excessive AWS costs. AutoML also allows dynamic scaling if needed based on training resource demand.

Model Evaluation

Models are evaluated based on RMSE (Root Mean Square Error), which is robust to outliers and provides an interpretable measure in the same unit as the target feature. This helps assess how well the model predicts numeric collision risk values. R^2 is also used to determine how much of the variance is described by the modeling. After running 3 AutoML jobs, the job with the best RMSE was .00024, however the R^2 was –0.0026 indicating that the model had no predictive power and that the available features were inadequate to predict collision probability.

Measuring Impact

List at least two specific metrics you expect to change with this project. These should tie back to the goals above.

Metric	Target Value	Justification
Debris Growth Forecast Accuracy	Acc > 0.85	Ensures predictive models reliably estimate debris accumulation.
Collision Risk Model Performance	R^2 ≥ 0.7	Validates the effectiveness of predicting satellite collision probabilities.
Clustering Accuracy	Noise Rate < 50%	Ensures meaningful clusters exist
Reduction in High-Risk Objects	20%	Demonstrates the effectiveness of the removal prioritization strategy.

Security Checklist, Privacy, and Other Risks

Will this store or process PHI or PII data?	**No health data or person-identifying data will be stored or processed as the information presented here is data on objects and is a matter of public record.**
Will user behavior be tracked and stored?	No, individual data is not applicable here.
Will this store or process credit card data?	No payment information was used for this project.
What data bias should be considered?	Class imbalance, survivorship bias, selection bias, and data leakage.
IAM Role Access	Only authorized AWS users can access the S3

	bucket. SageMaker instances assume an IAM role with read-only access. No public access is enabled for the dataset.
Data Protection Measures	Encryption in transit (uses SSL/TLS) Encryption at rest (uses S3 SSE-KMS) Monitoring (CloudTrail track access and modifications).
S3 buckets	Data is stored in AWS S3 with controlled IAM role access.
Cybersecurity	Implementing safeguards to protect against unauthorized data access.
Bias Awareness	Implementing safeguards to protect against unauthorized data access.
Are there any ethical concerns with the data or business problems that should be addressed?	The datasets used in this project are sourced from Kaggle, which enforces ethical data-sharing guidelines. These datasets comply with open data policies, ensuring responsible use. Additionally, we collaborate with space agencies to enhance transparency and accountability. Predictions of collisions will be uncertain given selection bias and missing information and given the gravity of the situation, that uncertainty should be considered when using these predictions to make decisions

	As this project explores ways to clean up and mitigate the accumulation of debris in space, there are political ramifications including but not limited to blaming countries and organizations There is a general responsibility of all organizations and entities involved in space to minimize the impact of space-related activity given the high potential for damage and cost both in orbit and on the ground.

Future Enhancements

1. **Image-Based Object Detection for Real-Time Tracking:** Developing AI-based tools for continuous debris monitoring. Most of the data used in this project was ground-based radar data. That's the most plentiful and easiest to work with given the scope of the objectives. However, there is plenty of image data out there, and for the lost objects in particular, utilizing images to locate lost objects would have improved the ability to capitalize on making accurate predictions for collision risk
2. **Automated Collision Avoidance:**
   Automated Collision Avoidance is a future enhancement designed to autonomously detect and respond to potential physical collisions in space, such as those between satellites, spacecraft, or orbital debris. This system would utilize advanced algorithms, real-time telemetry, and predictive modeling to continuously monitor the trajectory of objects in orbit. Upon identifying a potential collision risk, the system would autonomously calculate and execute avoidance maneuvers, reducing the need for manual intervention and response delays. This enhancement would be highly impactful because space traffic is rapidly increasing, and with it, the risk of catastrophic collisions. Currently, collision avoidance often relies on human analysis and manual decision-making, which can be time-consuming and prone to error, especially in time-sensitive scenarios. Automating this

process would not only improve mission safety and asset protection but also significantly reduce operational overhead. By enabling spacecraft to respond instantly and intelligently to threats, the system enhances reliability, ensures mission continuity, and contributes to the sustainability of space operations.

3. **AI-Driven Cleanup Planning:** Using reinforcement learning to optimize debris removal strategies. One of the most impactful future enhancements would be the integration of reinforcement learning (RL) into our debris removal strategy. By simulating thousands of orbital scenarios, an RL agent could be trained to learn the optimal sequence of debris collection actions to minimize collision risk and cost. Unlike traditional heuristic approaches, reinforcement learning dynamically adapts to evolving orbital conditions, learning from feedback loops and unexpected environmental changes, such as new debris fields or satellite movements. This adaptability enables continuous refinement and improvement of cleanup strategies over time. This enhancement is particularly vital for our mission because it directly supports NASA's objective of reducing high-risk object density by 20%. It shifts debris mitigation from static recommendations to adaptive, intelligent decision-making systems that evolve in response to real-world conditions. When coupled with the insights from clustering and predictive collision models, AI-driven planning has the potential to become the foundation of proactive orbital debris management. Additionally, implementing this capability through AWS SageMaker RL ensures both scalability and seamless integration within the existing cloud-based pipeline.

References

1. AWS. (n.d.). Amazon SageMaker Documentation. Retrieved April 13, 2025, from https://docs.aws.amazon.com/sagemaker/
2. Barth, A., & Fregly, C. (2021). Data science on AWS: Implementing end-to-end, continuous AI and machine learning pipelines. O'Reilly Media.

3. Barth, A., & Zolkowski, J. (2023). Generative AI on AWS: Building next-generation applications with foundation models. O'Reilly Media.
4. CelesTrak. (n.d.). Satellite tracking and collision data. Retrieved April 13, 2025, from https://www.celestrak.com
5. European Space Agency. (n.d.). Space Debris and DISCOS Database. Retrieved April 13, 2025, from https://www.esa.int/Safety_Security/Space_Debris
6. Kaggle. (n.d.). Public Datasets on Satellite and Debris Tracking. Retrieved April 13, 2025, from https://www.kaggle.com
7. NASA Orbital Debris Program Office. (n.d.). Orbital Debris Research and Reports. Retrieved April 13, 2025, from https://orbitaldebris.jsc.nasa.gov
8. Shmueli, G., Bruce, P. C., Gedeck, P., Patel, N. R., & Lichtendahl, K. C. (2023). Data mining for business analytics: Concepts, techniques, and applications in Python (4th ed.). Wiley.
9. Loth, A. (2019). Visual analytics with Tableau. Wiley.
10. Dykes, B. (2020). Effective data storytelling: How to drive change with data, narrative, and visuals. Wiley.