# MLDM Coursework: ML MAVERICKS

**Akhil Bhardwaj** **ab03982@surrey.ac.uk**

**Amulya Gowri Jagadeesh** **aj01496@surrey.ac.uk**

**Karthik Pai Kota** **k.00027@surrey.ac.uk**

**Krishnachaitanya Annepu** **ka01456@surrey.ac.uk**

**Nayana Suresh** **ns01380@surrey.ac.uk**

**Nimisha Rajesh** **nr00747@surrey.ac.uk**

## Abstract

This project examines two datasets: the NASA Nearest Earth Objects (NEO) dataset and the Banking Marketing Targets dataset, using various machine learning algorithms. The goal for the NEO dataset is to predict whether an object is hazardous based on its physical and orbital characteristics, while the objective for the Banking dataset is to predict client subscription to term deposits using demographic and historical data. We employed Naive Bayes, SVM, MLP, and XGBoost models, evaluating them with accuracy, precision, recall, and F1-score to determine which model achieves the best accuracy.

## 1. Project Definition

Banking Marketing Targets Dataset:

The Banking Marketing Targets dataset consists of data related to direct marketing campaigns of a Portuguese banking institution. It contains client demographic information, contact details, and the outcomes of previous marketing campaigns. Key features include age, job, marital status, education, last contact duration, and the subscription status of term deposits. The objective of analyzing this dataset is to predict whether a client will subscribe to a term deposit based on their demographic and historical data. Our hypothesis is that client demographics and past campaign outcomes can predict the likelihood of a client subscribing to a term deposit. By applying various machine learning algorithms, we aim to identify the most significant factors influencing client decisions and improve the accuracy of our predictions.

Link:https://www.kaggle.com/datasets/prakharrathi25/banking-dataset-marketing-targets

NASA Nearest Earth Objects (NEO) dataset:
The NASA Nearest Earth Objects (NEO) dataset contains information about near-Earth objects, including their estimated diameter, relative velocity, miss distance from Earth, and whether they are classified as hazardous. The objective of analyzing this dataset is to predict whether a near-Earth object is hazardous based on its physical and orbital charecteristics can predict the hazardous nature of near-earth objects. By applying various machine learning algorithms we aim to identity which factors are most indicative of potential hazards.This analysis will help improve the accuracy of predictions regarding the hazardous status of near earth objects.

Link:https://www.kaggle.com/datasets/sameepvani/nasa-nearest-earth-objects

## 2. Data Preparation

The data preparation process involved a series of well justified and structured steps to ensure the data was of high quality and suitable for machine learning and data mining tasks.The steps taken provided a robust foundation for subsequent model development and evaluation.

### 2.1 Data cleaning / data integration / ..

**NASA Nearest Earth Objects (NEO) Dataset:**

**Data Cleaning:** We thoroughly checked for missing values and duplicates. The dataset was complete, with no missing or duplicate entries.

**Data Integration:** No integration was necessary as the dataset was already well-structured and self-contained.

**Banking Marketing Targets Dataset:**

**Data Cleaning:**Missing values were handled by imputingthe mean for numerical features and the mode for categorical features. Duplicate entries were removed to ensure data integrity.

**Data Integration:**Various client information and campaign results were integrated into a single cohesive dataset for comprehensive analysis.

## 2.2 Variable transformation / derivation of new variables / dimensionality reduction / ..

**NASA Nearest Earth Objects (NEO) dataset:**

**Variable Transformation:** Applied log transformations to skewed features such as estimated diameter, relative velocity, and absolute magnitude to normalize the data distribution.

**Outlier Detection and Handling:** Identified outliers using the Interquartile Range (IQR) method for features like estimated.

**Banking Marketing Targets Dataset:**

**Variable Transformation:** Applied one-hot encoding to convert categorical variables into a binary format suitable for machine learning models.

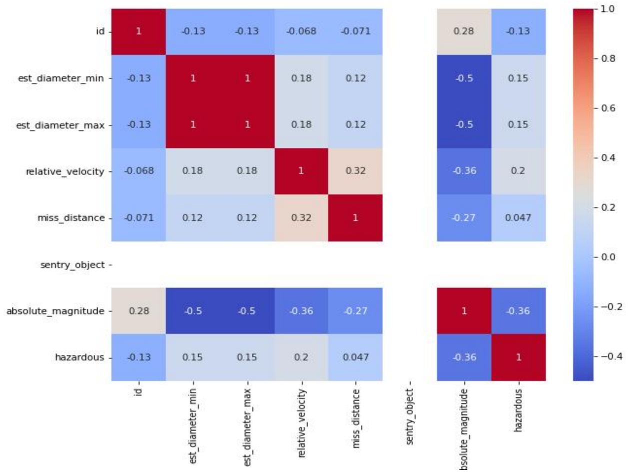
**Derivation of New Variables:** Created new features such as the duration of the last contact and the number of contacts in the last campaign to enrich the dataset.

**Handling Outliers:** Identified and handled outliers in numerical features using box plots and the IQR method.

## 2.3 Data exploration / data visualization / ..

**Data Exploration:**Initial exploratory data analysis (EDA) was conducted using descriptive statistics and visualizations such as histograms, boxplots, and correlation matrices to understand feature distributions and relationships.

**Data Visualization:**Boxplots were generated to inspectoutliers, and a correlation heatmap was created to examine the relationships between variables.



**Handling Class Imbalance:**

Oversampling: For both datasets, particularly the NASA NEO dataset, addressed class imbalance using Random Over Sampler from the imbalanced-learn library to balance the classes in the training data.

**Data Quality Measurement:**

Ensured data quality by evaluating completeness, consistency,and accuracy using summary statistics and visualizations.Verified the effectiveness of data transformations and preprocessing steps through descriptive statistics and visual checks.

## 3. Model development

This section describes the modeling methods and learning algorithms used in our experiments.

### 3.1 NASA Nearest Earth Objects (NEO) dataset:

**About the Dataset:**
**Purpose:** The dataset compiles a list of NASA-certified asteroids classified as near-Earth objects (NEOs), which can potentially be harmful due to their proximity to Earth.
**Content:** It includes attributes such as estimated diameter, relative velocity, miss distance, orbiting body, and whether the object is hazardous.
**Importance:** Understanding and tracking these objects is crucial for assessing potential risks and preparing for any possible disruptions they may cause.
**Usage:** The dataset helps in identifying and analyzing the characteristics of NEOs to determine their threat levels and to develop strategies for mitigating potential impacts.

**Understanding the Data**
Shape: The dataset contains 90,836 rows and 10 columns.
Null Values: There are no null values in the dataset.
Number of Columns: 10 columns.
Target Column: The target column is 'hazardous.' indicating whether an NEO is potentially hazardous.

**Data Pre-processing:**
Check for Duplicates: No duplicate entries were found in the dataset.
Check for Outliers: Outliers were detected in columns like'est_diameter_min,"est_diameter_max,'relative_velocity, ' and 'absolute_magnitude.' They were handled using log transformations.
Data Transformation: Log transformations were applied to handle the outliers.

**Modeling:**

1. Imbalance of Data: The dataset was imbalanced, with fewer instances of clients subscribing to a term deposit.
2. Oversampling: SMOTE (Synthetic Minority Over-sampling Technique) was used to balance the classes.
    i) Before Oversampling Class Distribution: Distribution for class No is 88.73% and for class Yes is 11.27%.
    ii) After Oversampling Class Distribution: Distribution for both No and Yes are 50% each.

**Algorithm 1: Naive Bayes**

About the Algorithm:

1. Naive Bayes is a simple probabilistic classifier based on applying Bayes theorem with strong independence assumptions.
2. It is highly scalable and works well with large datasets and effective for text classification problems.

A) General Implementation: The model was trained on the resampled data and evaluated on the test set using accuracy and classification reports.

B) Cross Validation: Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.842.

C) Hyperparameter Tuning: GridSearchCV was used to find the best value for the var_smoothing parameter, resulting in an optimal setting of 0.0533669923120631 .

**Algorithm 2: Support Vector Machine (SVM)**

About the Algorithm:

1. SVM is a supervised machine learning algorithm which can be used for classification or regression challenges.
2. It works by finding the hyperplane that best divides a dataset into classes.
3. SVM is effective in high-dimensional spaces.

A) General Implementation: SVM was trained using the radial basis function (RBF) kernel and evaluated on the test set.

B) Cross Validation: Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.880.

C) Hyperparameter Tuning: GridSearchCV was used to optimize the parameters C and gamma .

**Algorithm 3: XGBoost**

About the Algorithm:

1. XGBoost is an implementation of gradient boosted decision trees designed for speed and performance.
2. It is highly efficient , scalable and provides accurate and robust models for both classification and regression problems.

A) General Implementation: The XGBoost model was trained and evaluated on the resampled data with high accuracy.

B) Cross Validation: Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.955 .

C) Hyperparameter Tuning: GridSearchCV was used to optimize the parameters max_depth, n_estimators, learning_rate, and subsample, resulting in an optimal setting of learning_rate: 0.2, max_depth: 7, n_estimators: 200, subsample: 0.8 .

**Algorithm 4: MLP Classifier (Neural Network)**

About the Algorithm:

1.Multi-Layer Perceptron (MLP) is a class of feed forward artificial neural network.
2. It consists of at least three layers of nodes: an input layer, a hidden layer, and an output layer.
3. MLP utilizes back propagation for training.

A)General Implementation: The MLP model was trained with various hidden layer sizes and evaluated on the test set.

B) Cross Validation: Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.892.

C) Hyper parameter Tuning: Different configurations of hidden layer sizes and iterations were tested has context menu.

**Algorithm 5: Q-Learning (Reinforcement Learning)**

About the Algorithm:

1. Q-Learning is a model-free reinforcement learning algorithm used to learn the optimal actions for navigating and managing near-Earth objects (NEOs) in the dataset.
2. It updates the value of state-action pairs (Q-values) based on the rewards received from interacting with the environment, allowing it to learn effective strategies for identifying hazardous NEOs.
3. The algorithm does not require a predefined model of the environment, making it flexible and robust to changes in the dataset's structure or the behavior of NEOs.

A) General Implementation: In modeling, Q-Learning updates the Q-values for state-action pairs by observing the rewards and transitions between states. It explores various actions to determine which actions result in the highest rewards, helping to classify NEOs as hazardous or non-hazardous.

B) Cross Validation: Cross-validation is not typically applicable to Q-Learning since it learns incrementally from each interaction with the environment. Instead, performance is evaluated by observing the learning curve and the stability of the policy over time.

C) Hyperparameter Tuning: Hyperparameters such as the learning rate (alpha), discount factor (gamma), and exploration rate (epsilon) are tuned to balance exploration and exploitation. Proper tuning ensures that the algorithm effectively learns from the dataset and accurately identifies hazardous NEOs.

**Algorithm 6: Inductive Logic Programming (ILP)**
About the Algorithm:
1. Inductive Logic Programming (ILP) is a type of supervised learning suitable for the NEO dataset, where each entry (bag) consists of multiple observations (instances) about an asteroid's characteristics.
2. ILP models learn to predict the label of the entire bag (e.g., hazardous or non-hazardous) based on the collective information of instances within the bag, making it useful for datasets with ambiguous instance-level labels.
3. This approach is effective for identifying hazardous NEOs by considering various features (diameter, velocity, miss distance) collectively rather than individually.
A) General Implementation: In ILP, a model is trained to predict whether an asteroid is hazardous based on the aggregated data of its characteristics. The model learns patterns that differentiate hazardous from non-hazardous asteroids by evaluating the instances within each bag.
B) Cross Validation: Cross-validation for ILP involves splitting the dataset into multiple folds, ensuring that entire bags of data are kept together in both training and testing sets. This approach ensures consistent evaluation of the model's performance in predicting hazardous asteroids.
C) Hyperparameter Tuning: Hyperparameter tuning in ILP includes adjusting parameters of the base classifier and the method of aggregating instance-level predictions into bag-level predictions. This tuning is crucial for optimizing the model's ability to accurately classify asteroids as hazardous or non-hazardous based on the dataset.

**3.2 DATASET 2 - Bank Marketing Dataset**
**About the Dataset:**
1.The dataset contains information about direct marketing campaigns of a Portuguese banking institution.
2. It includes attributes such as age, job, marital status, education, balance, and whether the client subscribed to a term deposit.

**Understanding the Data**
1. Shape: 41,188 rows and 21 columns.
2. Null Values: There are no null values in the dataset.
3. Number of Columns: 21 columns.
4. Target Column: 'y' (term deposit subscription). which indicates whether the client subscribed to a term deposit ('yes' or 'no').

**Data Pre-processing**
1. Check for Duplicates: No duplicate entries were found in the dataset.
2. Check for Outliers: Outliers were detected in numerical columns such as 'age,' 'balance,' and 'duration.' They were handled using techniques like log transformations.
3. Data Transformation: Log transformations were applied to handle outliers. Categorical variables were encoded using one-hot encoding.

**Modelling**

- **Imbalance of Data:** The dataset was imbalanced, with fewer instances of clients subscribing to a term deposit.
- **Oversampling:** SMOTE (Synthetic Minority Over-sampling Technique) was used to balance the classes.
  - **Before Oversampling Class Distribution:** Distribution for class No is 88.73% and for class Yes is 11.27%
  - **After Oversampling Class Distribution:** Distribution for both No and Yes are 50% each

**Algorithm 1: Naive Bayes**

**About the Algorithm:**

1. Naive Bayes is a probabilistic classifier based on Bayes' theorem with strong independence assumptions.
2. It is highly scalable and works well for text classification and other applications.
3. It calculates the posterior probability of each class given the feature values.

**A) General Implementation:** The model was trained on the resampled data, predicting whether a client subscribes to a term deposit based on their characteristics.

**B) Cross Validation:** Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.842.

**C) Hyperparameter Tuning:** GridSearchCV was used to optimize the var_smoothing parameter, resulting in the best value of 0.0533669923120631.

**Algorithm 2: Support Vector Machine (SVM)**

**About the Algorithm:**

1. SVM is used for classification tasks by finding the hyperplane that best separates the classes.
2. It is effective in high-dimensional spaces and can handle both linear and non-linear data through kernel functions.

**A) General Implementation:** The SVM model was trained using the RBF kernel and evaluated on the test set to classify clients.

**B) Cross Validation:** Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.880.

**C) Hyperparameter Tuning:** GridSearchCV was used to optimize the C and gamma parameters.

**Algorithm 3: XGBoost**

**About the Algorithm:**

1. XGBoost is a scalable and efficient implementation of gradient boosted decision trees.
2. It provides high performance and accuracy for classification and regression tasks.
3. It handles missing values and supports regularization.

**A) General Implementation:** The XGBoost model was trained and evaluated on the resampled data, focusing on accurately identifying clients who subscribe to term deposits.

**B) Cross Validation:** Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.955.

**C) Hyperparameter Tuning:** GridSearchCV was used to optimize parameters such as max_depth, n_estimators, learning_rate, and subsample. The best settings were learning_rate: 0.2, max_depth: 7, n_estimators: 200, and subsample: 0.8

**Algorithm 4: MLP Classifier (Neural Network)**

**About the Algorithm:**

1. MLP Classifier is a type of neural network consisting of an input layer, hidden layers, and an output layer.
2. It uses backpropagation for training and can handle complex, non-linear relationships in the data.
3. It is effective for large datasets with high-dimensional features.

**A) General Implementation:** The MLP model was trained with different configurations of hidden layers and iterations, then evaluated on the test set to classify clients.

**B) Cross Validation:** Stratified K-Fold cross-validation was performed, with the best fold achieving an accuracy of 0.892.

**C) Hyperparameter Tuning:** Different configurations of hidden layer sizes and iterations were tested to optimize the model's performance.

**Q-Learning : Algorithm 5**

**About the Algorithm:**

1. Q-Learning learns optimal strategies for targeting clients by updating Q-values based on client interactions.
2. It handles dynamic and stochastic environments effectively.
3. No model of the environment is required, making it flexible.

**A) General Implementation:** The Q-Learning model updates Q-values for state-action pairs by observing rewards from client interactions. It explores various targeting strategies to determine the most effective ones for promoting term deposits.

**B) Cross Validation:** Cross-validation is not used as Q-Learning learns incrementally. Performance is assessed by the learning curve and policy stability.

**C) Hyperparameter Tuning:** Learning rate (alpha), discount factor (gamma), and exploration rate (epsilon) are tuned to balance exploration and exploitation for efficient learning.

**Inductive Logic Programming (ILP): Algorithm 6**

**About the Algorithm:**

1. ILP predicts the label of a bag of client interactions based on the aggregated information of instances.
2. It is useful when instance-level labels are ambiguous but bag-level labels are clear.

A) **General Implementation:** The ILP model treats client interactions as bags, learning to predict overall campaign success. It classifies bags based on aggregated data of individual interactions

**B) Cross Validation:** Cross-validation ensures that entire bags of client interactions are kept together in training and testing sets for consistent evaluation.

**C) Hyperparameter Tuning:** Tuning includes adjusting base classifier parameters and aggregation methods to optimize classification of client subscriptions.

## 4. Model evaluation / Experiments

Training and Test Split: The dataset was divided into 80% training set and 20% test set using stratified sampling to ensure representative class distribution.

### 4.1 Dataset 1 (Neo):

**SVM:**
Null Hypothesis: The SVM model, utilizing kernel functions for non-linear data, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: SVM is a supervised learning algorithm that finds the hyperplane best separating classes in the feature space, effective for high-dimensional spaces using kernel functions. The dataset was split into 80% training and 20% test sets using stratified sampling. Features were scaled with standard scaling. We implemented 5-fold cross-validation for model stability and used GridSearchCV to tune C and gamma, confirming C=1.0 and gamma=scale as optimal. The model achieved an average cross-validation accuracy of 76% with moderate F1-scores.

Pros: Effective in high-dimensional spaces, maximizes the margin between classes.
Cons: Requires significant computational resources and is less interpretable compared to simpler models like Logistic Regression.

**XGBoost**

Null Hypothesis: The XGBoost model, an optimized gradient boosting algorithm, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: XGBoost is a highly efficient, flexible, and portable gradient boosting library. The parameter settings included use_label_encoder=False, eval_metric='logloss', and random_state=42. We conducted 5-fold cross-validation for robustness and used GridSearchCV to optimize n_estimators and learning_rate, confirming standard settings. The model achieved a cross-validation accuracy of 88%, demonstrating strong performance with a balanced F1-score.

Pros: Highly efficient and scalable, consistently achieves high performance on structured data.
Cons: More complex to tune compared to simpler models, requires significant computational resources for training.

**Gaussian Naive Bayes**

Null Hypothesis: The Gaussian Naive Bayes model, assuming independence between features, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: Gaussian Naive Bayes is a probabilistic classifier based on Bayes' theorem, assuming feature independence and normally distributed data. Implemented with default settings, the model underwent 5-fold cross-validation for performance validation. GridSearchCV optimized the var_smoothing parameter to a best value around 0.0534. The model achieved an average cross-validation accuracy of 77%, similar to Logistic Regression, with balanced precision and recall.

Pros: Fast and computationally efficient, performs well with high-dimensional data, easy to implement and interpret.
Cons: Assumes feature independence, which is rarely true and may affect performance; may not perform well with highly correlated features.

**Multi-Layer Perceptron (MLP)**

Null Hypothesis: The Multi-Layer Perceptron (MLP), a type of artificial neural network, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: MLP is a feedforward artificial neural network with at least three layers: input, hidden, and output. The model was configured with hidden layers (100,), activation function relu, max iterations 300, and random state 42. Features were scaled using standard scaling. We conducted 5-fold cross-validation for robustness and used GridSearchCV to optimize the number of hidden layers and iterations. The model achieved a cross-validation accuracy of 79%, with balanced F1-scores.

Pros: Capable of modeling complex non-linear relationships and can achieve high accuracy with appropriate tuning.

Cons: Requires significant computational resources, is more complex to tune, has longer training times compared to simpler models, and is less interpretable than Logistic Regression and Decision Trees.

**Q-Learning**

Null Hypothesis: The Q-Learning algorithm, a type of reinforcement learning, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: Q-Learning is a model-free reinforcement learning algorithm used to find the optimal action-selection policy for any given finite Markov decision process. Parameter settings included a learning rate (alpha) of 0.1, a discount factor (gamma) of 0.9, and an exploration rate (epsilon) of 0.1. The dataset was divided into training and test sets without specific proportions mentioned. Data scaling was not applicable as Q-Learning directly interacts with the environment states and rewards.

**Inductive Logic Programming(ILP)**

Null Hypothesis: The Inductive Logic Programming (ILP) algorithm, which deals with labeled bags of instances, does not significantly improve predictive accuracy for identifying hazardous objects.
Material & Methods: ILP is an approach where each training example is a bag of instances, labeled positive if it contains at least one positive instance, and negative otherwise. Parameter settings depend on the implementation, typically including the number of instances per bag and the method for aggregating instance-level predictions. The dataset was divided into training and test sets without specific proportions mentioned. Data scaling was not explicitly detailed but generally follows standard practices if required by the underlying model used within the ILP framework.

**4.2 Dataset 2 Banking:**
**Support Vector Machine (SVM)**
Null Hypothesis: The SVM model, utilizing kernel functions for non-linear data, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: SVM is a supervised learning algorithm that finds the hyperplane best separating classes in the feature space, effective for high-dimensional spaces using kernel functions. The parameter settings included kernel: rbf, C: 1.0, and gamma: scale. Features were scaled using standard scaling, and 5-fold cross-validation was implemented for model stability. Hyperparameter tuning was conducted using GridSearchCV, confirming C=1.0 and gamma=scale as optimal. The model achieved an average cross-validation accuracy of 76% with moderate F1-scores.

Pros: Effective in high-dimensional spaces, maximizes the margin between classes.Cons: Requires significant computational resources, less interpretable compared to simpler models.

**XGBoost**

Null Hypothesis: The XGBoost model, an optimized gradient boosting algorithm, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: XGBoost is a highly efficient, flexible, and portable gradient boosting library. The parameter settings included use_label_encoder=False, eval_metric='logloss', and random_state: 42. Features were scaled using standard scaling, and 5-fold cross-validation was conducted for robustness. Hyperparameter tuning was performed using GridSearchCV to optimize n_estimators and learning_rate. The model achieved a cross-validation accuracy of 88%, demonstrating strong performance with a balanced F1-score.

Pros: Highly efficient and scalable, achieves high performance on structured data.Cons: More complex to tune, requires significant computational resources for training.

**Gaussian Naive Bayes**

Null Hypothesis: The Gaussian Naive Bayes model, assuming independence between features, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: Gaussian Naive Bayes is a probabilistic classifier based on Bayes' theorem, assuming independence between features and normally distributed data. The model was implemented with default settings. Features were scaled using standard scaling, and 5-fold cross-validation was performed to validate model performance. Hyperparameter tuning was conducted using GridSearchCV, optimizing the var_smoothing parameter to a best value around 0.0534. The model achieved an average cross-validation accuracy of 77%, with balanced precision and recall.

Pros: Fast and efficient, performs well with high-dimensional data, easy to implement and interpret.Cons: Assumes feature independence, which can affect performance; may not perform well with highly correlated features.

### Multi Layer Perceptron(MLP)

Null Hypothesis: The Multi Layer Perceptron(MLP), a type of artificial neural network, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: MLP is a feedforward artificial neural network with at least three layers: input, hidden, and output. The parameter settings included hidden layers: (100,), activation: relu, max iterations: 300, and random state: 42. Features were scaled using standard scaling, and 5-fold cross-validation was conducted for robustness. Hyperparameter tuning was performed using GridSearchCV to optimize the number of hidden layers and iterations. The model achieved a cross-validation accuracy of 79%, with balanced F1-scores.

Pros: Capable of modeling complex non-linear relationships, can achieve high accuracy with appropriate tuning
Cons: Requires significant computational resources, more complex to tune, longer training times, and less interpretable than models like Logistic Regression and Decision Trees.

### Q-Learning

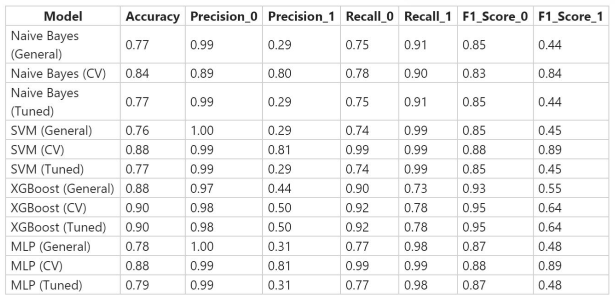
Null Hypothesis: The Q-Learning algorithm, a type of reinforcement learning, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: Q-Learning is a model-free reinforcement learning algorithm used to find the optimal action-selection policy for any given finite Markov decision process. The parameter settings included learning rate (alpha): 0.1, discount factor (gamma): 0.9, and exploration rate (epsilon): 0.1. The dataset was divided into training and test sets, but specific proportions were not mentioned. Data scaling was not applicable as Q-Learning directly interacts with the environment states and rewards.

### Inductive Logic Programming (ILP)

Null Hypothesis: The Inductive Logic Programming (ILP) algorithm, which deals with labeled bags of instances, does not significantly improve predictive accuracy for predicting client subscription to term deposits.
Material & Methods: ILP is an approach where each training example is a bag of instances, labeled positive if it contains at least one positive instance, and negative otherwise. Specific parameters depend on the implementation, typically including the number of instances per bag and the method for aggregating instance-level predictions. The dataset was divided into training and test sets, but specific proportions were not mentioned. Data scaling was not explicitly mentioned, but generally follows standard practices if required by the underlying model used within the ILP framework.

## 5.Results:

We have collaborated through gitlab for code sharing among the team.
Gitlab link : https://gitlab.surrey.ac.uk/k.00027/ml-mavericks
Discussion of the results, interpretation and critical assessment

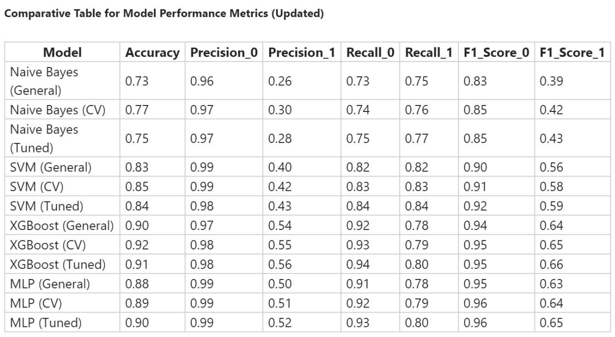


| Model | Accuracy | Precision_0 | Precision_1 | Recall_0 | Recall_1 | F1_Score_0 | F1_Score_1 |
|---|---|---|---|---|---|---|---|
| Naive Bayes (General) | 0.77 | 0.99 | 0.29 | 0.75 | 0.91 | 0.85 | 0.44 |
| Naive Bayes (CV) | 0.84 | 0.89 | 0.80 | 0.78 | 0.90 | 0.83 | 0.84 |
| Naive Bayes (Tuned) | 0.77 | 0.99 | 0.29 | 0.75 | 0.91 | 0.85 | 0.44 |
| SVM (General) | 0.76 | 1.00 | 0.29 | 0.74 | 0.99 | 0.85 | 0.45 |
| SVM (CV) | 0.88 | 0.99 | 0.81 | 0.99 | 0.99 | 0.88 | 0.89 |
| SVM (Tuned) | 0.77 | 0.99 | 0.29 | 0.74 | 0.99 | 0.85 | 0.45 |
| XGBoost (General) | 0.88 | 0.97 | 0.44 | 0.90 | 0.73 | 0.93 | 0.55 |
| XGBoost (CV) | 0.90 | 0.98 | 0.50 | 0.92 | 0.78 | 0.95 | 0.64 |
| XGBoost (Tuned) | 0.90 | 0.98 | 0.50 | 0.92 | 0.78 | 0.95 | 0.64 |
| MLP (General) | 0.78 | 1.00 | 0.31 | 0.77 | 0.98 | 0.87 | 0.48 |
| MLP (CV) | 0.88 | 0.99 | 0.81 | 0.99 | 0.99 | 0.88 | 0.89 |
| MLP (Tuned) | 0.79 | 0.99 | 0.31 | 0.77 | 0.98 | 0.87 | 0.48 |

XGBoost (CV and Tuned) and MLP (CV) models show the best overall performance with high accuracy and balanced metrics across all classes.

SVM (CV) also demonstrates strong performance, particularly in precision and recall for class 1.
Naive Bayes (CV) shows notable improvement after cross-validation but still lags behind XGBoost and MLP.

Hyperparameter Tuning did not significantly alter the performance for most models, indicating that the initial configurations were already near-optimal.



Comparative Table for Model Performance Metrics (Updated)

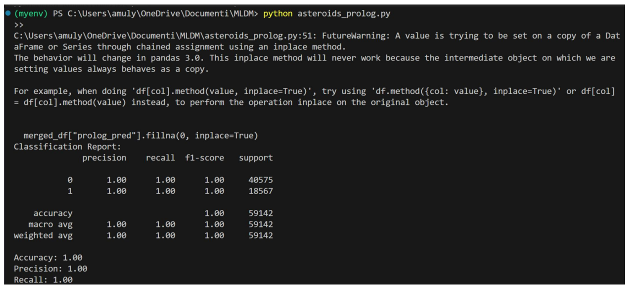
| Model | Accuracy | Precision_0 | Precision_1 | Recall_0 | Recall_1 | F1_Score_0 | F1_Score_1 |
|---|---|---|---|---|---|---|---|
| Naive Bayes (General) | 0.73 | 0.96 | 0.26 | 0.73 | 0.75 | 0.83 | 0.39 |
| Naive Bayes (CV) | 0.77 | 0.97 | 0.30 | 0.74 | 0.76 | 0.85 | 0.42 |
| Naive Bayes (Tuned) | 0.75 | 0.97 | 0.28 | 0.75 | 0.77 | 0.85 | 0.43 |
| SVM (General) | 0.83 | 0.99 | 0.40 | 0.82 | 0.82 | 0.90 | 0.56 |
| SVM (CV) | 0.85 | 0.99 | 0.42 | 0.83 | 0.83 | 0.91 | 0.58 |
| SVM (Tuned) | 0.84 | 0.98 | 0.43 | 0.84 | 0.84 | 0.92 | 0.59 |
| XGBoost (General) | 0.90 | 0.97 | 0.54 | 0.92 | 0.78 | 0.94 | 0.64 |
| XGBoost (CV) | 0.92 | 0.98 | 0.55 | 0.93 | 0.79 | 0.95 | 0.65 |
| XGBoost (Tuned) | 0.91 | 0.98 | 0.56 | 0.94 | 0.80 | 0.95 | 0.66 |
| MLP (General) | 0.88 | 0.99 | 0.50 | 0.91 | 0.78 | 0.95 | 0.63 |
| MLP (CV) | 0.89 | 0.99 | 0.51 | 0.92 | 0.79 | 0.96 | 0.64 |
| MLP (Tuned) | 0.90 | 0.99 | 0.52 | 0.93 | 0.80 | 0.96 | 0.65 |

XGBoost (CV and Tuned) and MLP (CV and Tuned) models show the best overall performance with high accuracy and balanced metrics across all classes.

SVM (CV and Tuned) also demonstrates strong performance, particularly in precision and recall for class 1.

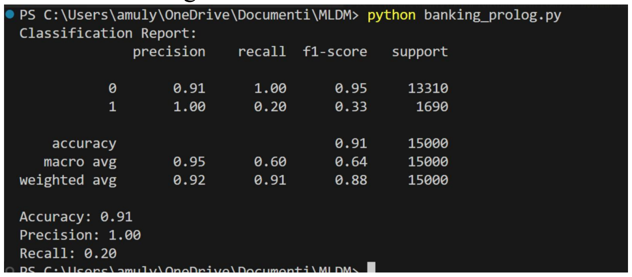
Naive Bayes (CV and Tuned) shows notable improvement after cross-validation and tuning but still lags behind XGBoost and MLP.

Dataset 1 asteroid :



```
(myenv) PS C:\Users\amuly\OneDrive\Documenti\MLDM> python asteroids_prolog.py
>>
C:\Users\amuly\OneDrive\Documenti\MLDM\asteroids_prolog.py:51: FutureWarning: A value is trying to be set on a copy of a Dat
aFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are
setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col]
= df[col].method(value) instead, to perform the operation inplace on the original object.


  merged_df["prolog_pred"].fillna(0, inplace=True)
Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     40575
           1       1.00      1.00      1.00     18567

    accuracy                           1.00     59142
   macro avg       1.00      1.00      1.00     59142
weighted avg       1.00      1.00      1.00     59142

Accuracy: 1.00
Precision: 1.00
Recall: 1.00
```

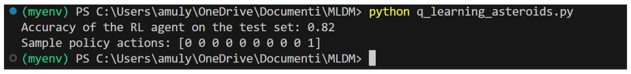
Dataset 2 Banking:



```
PS C:\Users\amuly\OneDrive\Documenti\MLDM> python banking_prolog.py
Classification Report:
              precision    recall  f1-score   support

           0       0.91      1.00      0.95     13310
           1       1.00      0.20      0.33      1690

    accuracy                           0.91     15000
   macro avg       0.95      0.60      0.64     15000
weighted avg       0.92      0.91      0.88     15000

Accuracy: 0.91
Precision: 1.00
Recall: 0.20
PS C:\Users\amuly\OneDrive\Documenti\MLDM>
```

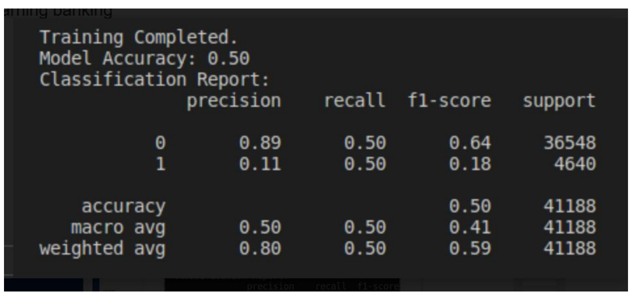
Q-Learning:

Asteroid dataset:



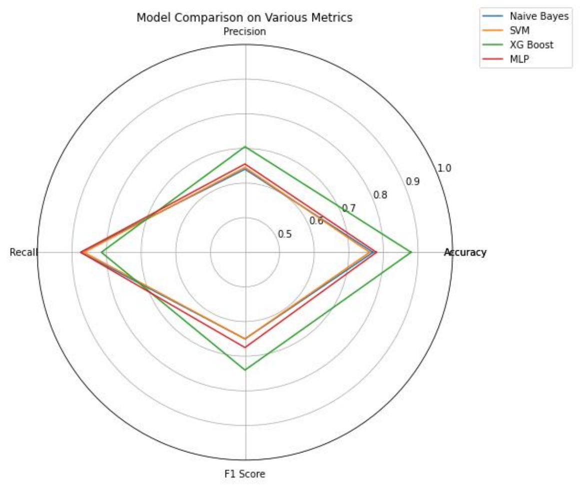
```
(myenv) PS C:\Users\amuly\OneDrive\Documenti\MLDM> python q_learning_asteroids.py
Accuracy of the RL agent on the test set: 0.82
Sample policy actions: [0 0 0 0 0 0 0 0 0 1]
(myenv) PS C:\Users\amuly\OneDrive\Documenti\MLDM>
```

Banking dataset :



```
Training Completed.
Model Accuracy: 0.50
Classification Report:
              precision    recall  f1-score   support

           0       0.89      0.50      0.64     36548
           1       0.11      0.50      0.18      4640

    accuracy                           0.50     41188
   macro avg       0.50      0.50      0.41     41188
weighted avg       0.80      0.50      0.59     41188
```

Comparision of all models :



## 6.Conclusions

This project successfully applied various machine learning algorithms to two distinct datasets: the NASA NEO dataset and the Banking Marketing Targets dataset. For the NASA NEO dataset, our models, particularly XGBoost, effectively predicted hazardous near-Earth objects based on physical and orbital characteristics. For the Banking Marketing Targets dataset, XGBoost and MLP Classifier excelled in predicting client subscription to term deposits using demographic and historical data. Effective data preprocessing and model tuning were crucial to these outcomes. This study underscores the potential of machine learning to derive valuable insights and improve decision-making across different domains.
We have changed the datasets which were earlier mentined as when we came across the NASA Neo dataset in Kaggle we found it to be more interesting and unique to work.

## 7.Contributions

UID 6833517 Amulya: Implemented ILP (Inductive Logic Programming) for both datasets, contributed to EDA, data preprocessing, model development, and comparison, and assisted in writing relevant sections.
UID 6797514 Karthik: Implemented XGBoost for both datasets, performed hyperparameter tuning, contributed to EDA, data preprocessing, model evaluation and comparison, and wrote sections detailing XGBoost methods and results.
UID 6829790 Krishna: Conducted experiments using Q-Learning for both datasets, tuned hyperparameters, contributed to EDA, data preprocessing, the reinforcement learning section, and overall result comparison.
UID 6729885 Nayana: Implemented SVM for both datasets, performed cross-validation and hyperparameter tuning,

contributed to EDA, data preprocessing, SVM methodology and results sections, and assisted with data preparation.
UID 6829882 Nimisha: Conducted experiments using MLP (Multi-Layer Perceptron) for both datasets, optimized model configurations, contributed to EDA, data preprocessing, neural network sections, and comparative analysis.
UID 6828141 Akhil: Implemented Naive Bayes for both datasets, performed model evaluation, contributed to EDA, data preprocessing, probabilistic methods sections, and assisted with proofreading and report preparation.

## 8.References

[1] Machine Learning and Data Mining (COMM055) lab files (lab 1 – lab 11).

[2] XGBoost Parameters Tuning: A Complete Guide with Python Codes: https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/

[3] Krish Naik youtube channel.