**Project 3: Classifying malware data and finding sustainable ML approach**

**Due Date:** Nov. 20, 2024 (Wed), by 11:59 p.m.   
**Submission Format:** Individual

For this project, you will implement two (or more) machine learning classifiers using scikit-learn. These classifiers will be applied to a Android malware dataset in the folder input\_data.

**Dataset**:

The data consists of malware and goodware of Android apps for 8 years. Android malware dataset has a property of data drift. Data drift occurs when the data used to train a machine-learning model is no longer representative of the new data the model encounters.

Data drift refers to the change in the statistical properties of the data over time, which can cause a machine learning model's predictions to become less accurate. It typically happens because:

1. **Feature Drift**: The distribution of individual features changes. For example, if you trained a model to predict sales based on seasonality, a sudden change in consumer behavior might impact your model's performance.
2. **Concept Drift**: The relationship between the input and output changes. For instance, a model predicting product demand might experience concept drift if the factors influencing demand evolve (like new trends or policies affecting consumer behavior).

Detecting and addressing data drift is crucial to maintaining model accuracy over time. Common ways to manage data drift include monitoring model performance, retraining models on recent data, and using adaptive algorithms that can adjust to changes in data distribution.

The first column of each file should be discarded.

Other than 1st column is the information about API call information of Android apps.

Each file has 1500 rows \* 1849 columns.

Target class: binary (malware, goodware)

Labeling for target class: each file name includes the target class of each app in the file.

**Tasks:**

**Step 1: Prepare the Data**

1. **Load the Data:** Load the files from the input\_data folder. Each file represents an app with its respective features and target label.
2. **Discard the First Column:** As specified, drop the first column from each file.
3. **Extract Labels:** Extract the label (malware/goodware) from the filename and add it as a target variable.
4. **Combine Files:** Concatenate all files into a single dataset, with each row representing an app and each column (excluding the target) representing API call information.

**Step 2: Handle Data Drift**

Since this dataset spans 8 years, it likely has **data drift** (feature drift or concept drift). Here are some suggestions:

* **Split by Year:** Train and test on different years to observe how the models handle data drift.
* **Monitoring Drift:** Consider using statistical methods (e.g., Kolmogorov-Smirnov test) to measure the feature distribution changes over the years.

**Step 3: Preprocess Data for Model Training**

1. **Normalize/Scale Features:** Scale the features to standardize the data, which is beneficial for many machine learning models.
2. **Split Data:** Split the data into a training and testing set. Optionally, use a time-based split (e.g., first few years for training, later years for testing) to simulate real-world drift.

**Step 4: Apply Machine Learning Classifiers**

1. **Select Classifiers:** Implement at least two classifiers (e.g., Logistic Regression, Random Forest, or Gradient Boosting) from scikit-learn.
2. **Train Models:** Train the classifiers on the training dataset.

**Step 5: Evaluate Classifier Performance**

1. **Accuracy Comparison:** Use accuracy as your primary metric to evaluate each classifier on the training data.
2. **Additional Metrics:** Consider using F1-score, precision, and recall, as malware classification is often an imbalanced problem.

**Step 6: Test Data Prediction**

1. **Test Each Classifier:** Predict on the test data with each trained classifier.
2. **Analyze Performance on Drifted Data:** If possible, compare the performance of classifiers on subsets of data from different years.

**Step 7: Sustainable ML Approach**

Since you are looking for a sustainable ML approach, you might consider:

* **Periodic Retraining:** Retrain models periodically as new data becomes available.
* **Adaptive Algorithms:** Explore incremental training for classifiers to adapt to changes in data distribution over time.

**Submission Guidelines**

* Submit your code in either a .py file or a Jupyter Notebook.
* Make sure your code is well-documented, with comments explaining each major step.