CSE4003

—

Cyber Security

Digital Assignment -2

Yerramalli Sai Sreekar / 20BCE1296

C1 / Slot

Dr. Subbulakshmi T/ Professor

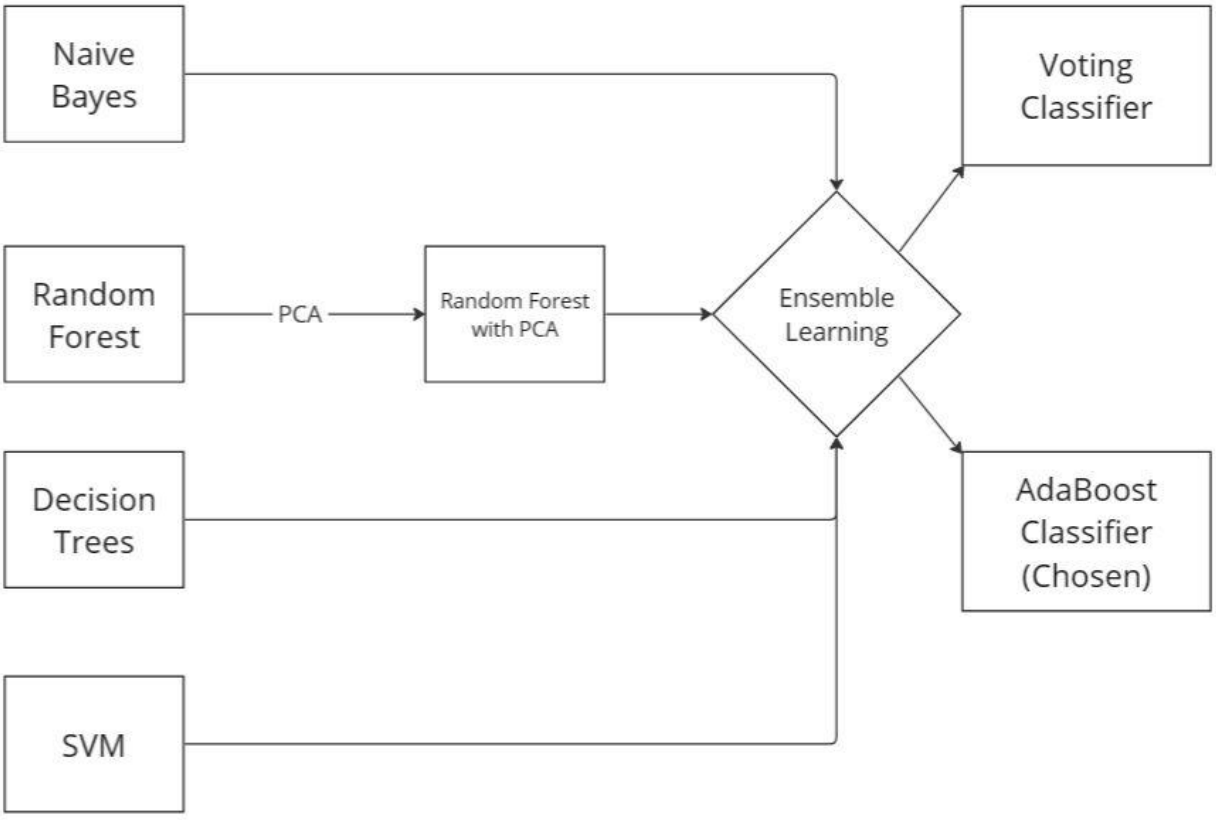
Proposed Architecture & Module Description:

The Architecture will be are following is as follows:

- Data Preprocessing: This module involves collecting, cleaning, and processing the data before it can be used for machine learning. It may include steps such as removing duplicates, filling missing values, and transforming the data into a suitable format for machine learning algorithms.
- Feature Importance: This module is responsible for identifying the most relevant features from the preprocessed data that contribute the most to the classification task. The module may use techniques such as correlation analysis or feature selection algorithms to determine the importance of each feature.
- Model Training: This module involves training different machine learning models on the preprocessed data and the selected features. The module may use techniques such as cross-validation and hyperparameter tuning to optimize the performance of the models. Examples of machine learning models that can be used for malware detection include random forest, support vector machines, and neural networks.

The Model we planned is built as follows:

- Naive Bayes, Random Forest,Decision Trees & Support Vector Machines are the Base Models.
- We then Applied PCA to have better specificity
- The New Improved Models with PCA are now used as Base model for ensemble Learning
- Two models were Created, One with Hard voting and another with a boosting algorithm using Adaboost



- Model Testing: This module evaluates the performance of the trained models on a separate set of test data to measure their accuracy, precision, recall, and F1 score. The module may also generate a confusion matrix and ROC curve to visualize the model's performance.
- Frontend: This module is responsible for providing an interactive user interface for the malware detection system. It may include features such as file uploading, processing, and displaying the classification results. The frontend can be implemented using web frameworks such as Flask, Django, or ReactJS.