Nature-Based Prediction Model of Bug Reports Based on Ensemble Machine Learning Model

In software developments life cycle two stages are very much important such as Maintenance and upgrade. While upgrading coders often make mistakes which will raise errors while execution at customer side. This errors will be reported to maintenance team by customers who will assign to developers for rectification. Manual inspection will be time consuming so author of this paper trying to automate bug reports using machine learning algorithms.

All existing bug prediction models are based on individual machine learning algorithms whose detection accuracy is not accurate. So author of this paper employing ensemble algorithm by combining various algorithms using Voting Classifier and then this algorithm will vote out each algorithm and then select algorithm with highest accuracy. Voting classifier combining various classifier such as Logistic Regression, Multinomial Naïve Bayes, SVM and Random Forest. Each algorithms run individually and with Voting classifier and Voting Classifier giving high accuracy.

To train all algorithms author has used ‘Eclipse and Mozilla’ bug reports dataset with different bug types but this dataset is not available on internet so we use ‘Eclipse Mozilla bug’ dataset from below KAGGLE URL

<https://www.kaggle.com/datasets/saurabhshahane/long-lived-bug-prediction/data?select=mozilla_bug_report_data.csv>

From above dataset we have selected 6 different bug types such as 'Client', 'General', 'Hyades', 'Releng', 'Xtext', 'cdt-core'.

Extension Concept

In propose paper author has used all traditional machine learning algorithms so as extension we have experimented with XGBOOST algorithm which is consider as one of the advance ML algorithm and known for better prediction accuracy.

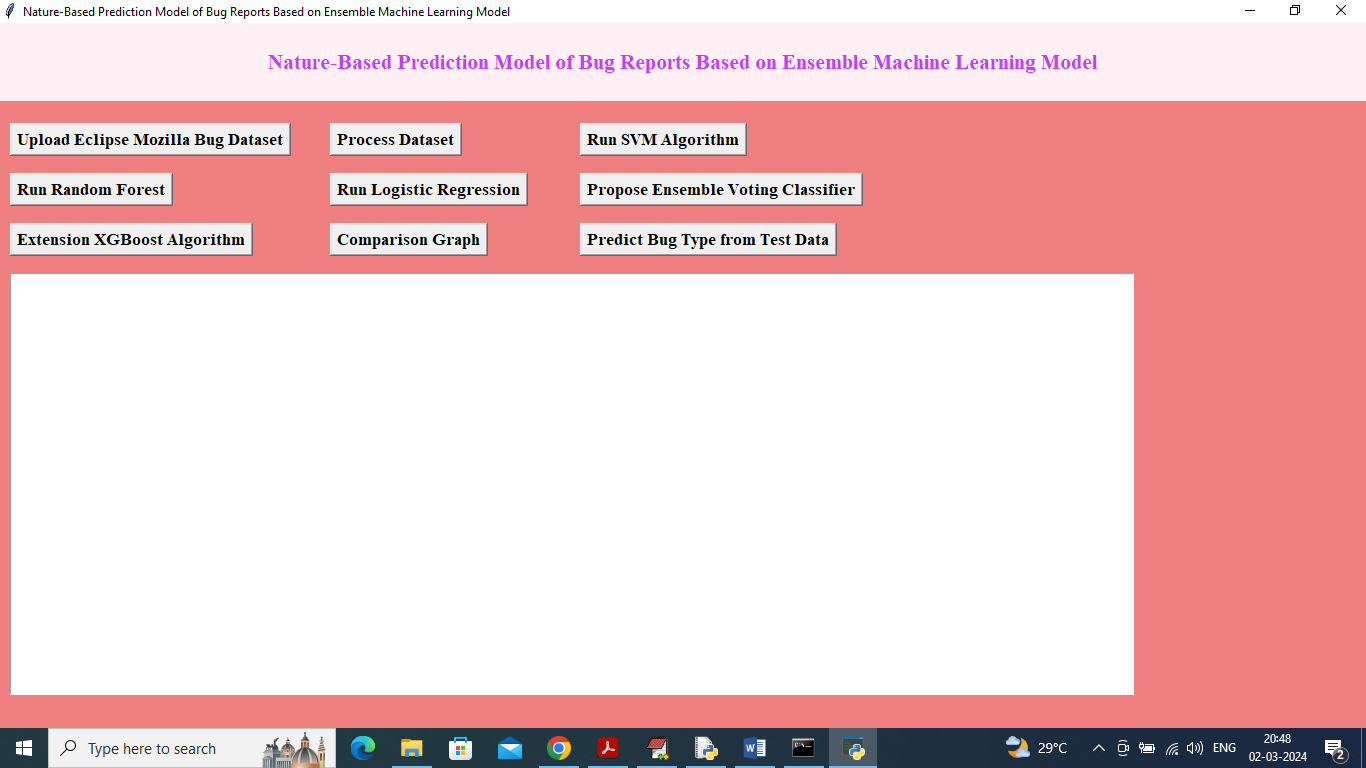
Note: paper and real implementation algorithm will vary little as we don’t have exact dataset.

To implement this project we have designed following modules

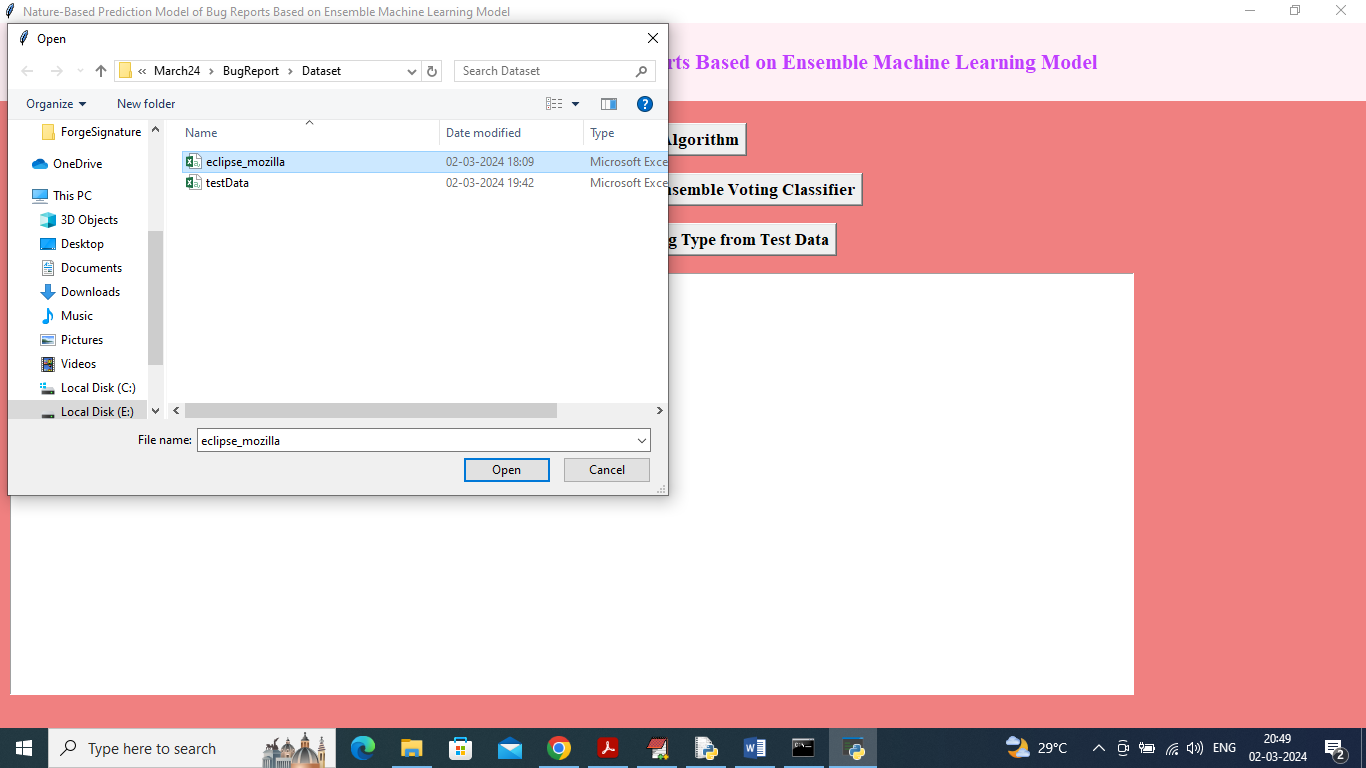
1. Upload Eclipse Mozilla Bug Dataset: using this module can upload dataset to application and then remove special symbols, stop words, apply lemmatization, stemming to clean all text data
2. Process Dataset: will convert entire text data into augmented numeric TF-IDF vector which will replace each words with its average frequency. TF-IDF will get normalize and shuffle. Processed data will be split into train and test where application will be using 80% dataset for training and 20% for testing
3. Run SVM Algorithm: 80% training data will be input to SVM algorithm to train a model and this model will be applied on 20% test data to calculate accuracy and other metrics
4. Run Random Forest Algorithm: 80% training data will be input to Random Forest algorithm to train a model and this model will be applied on 20% test data to calculate accuracy and other metrics
5. Run Logistic Regression Algorithm: 80% training data will be input to LR algorithm to train a model and this model will be applied on 20% test data to calculate accuracy and other metrics
6. Propose Ensemble Voting Classifier: 80% training data will be input to Voting Classifier algorithm to train a model and this model will be applied on 20% test data to calculate accuracy and other metrics
7. Extension XGBoost Algorithm: 80% training data will be input to XGBoost algorithm to train a model and this model will be applied on 20% test data to calculate accuracy and other metrics
8. Comparison Graph: will plot comparison graph between all algorithms
9. Predict Bug Type from Test Data: using this module will upload test data and then system will predict bug type

SCREEN SHOTS

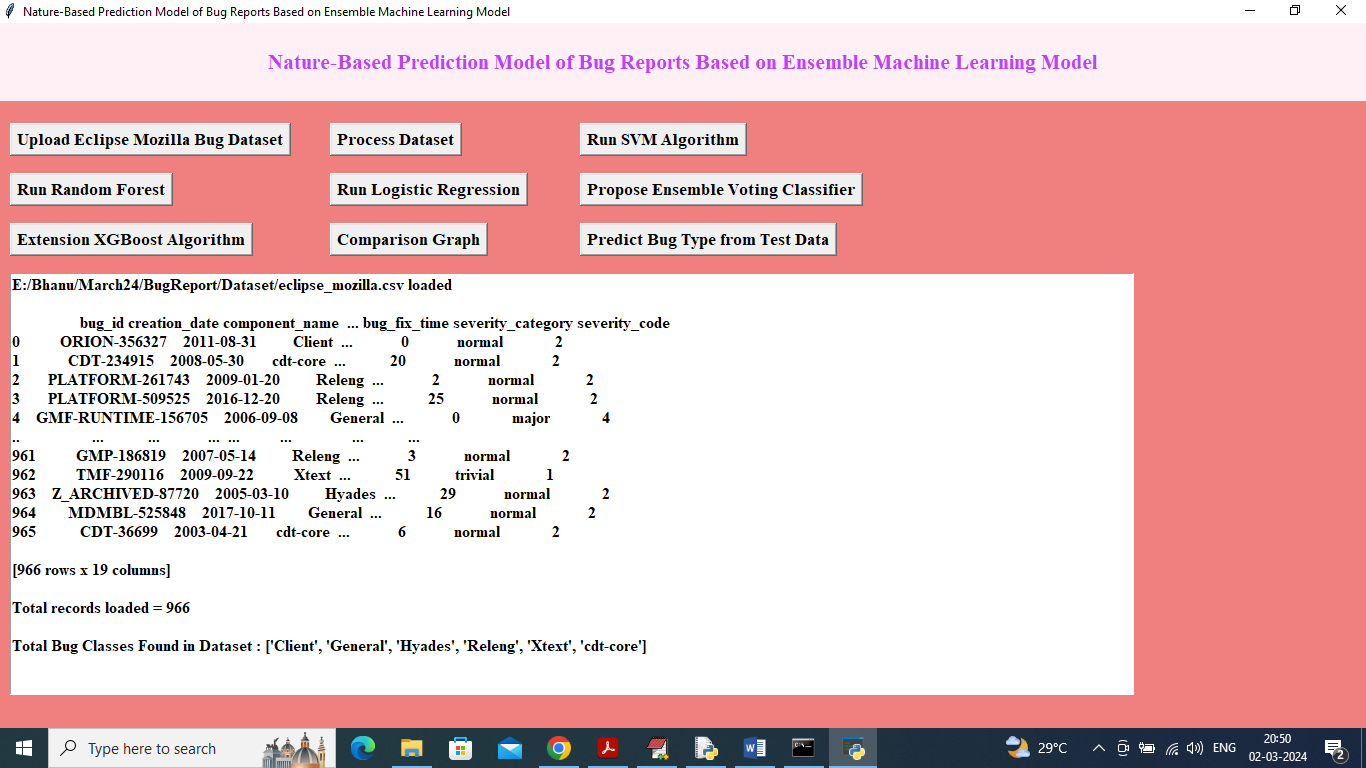
To run project double click on run.bat file to get below screen



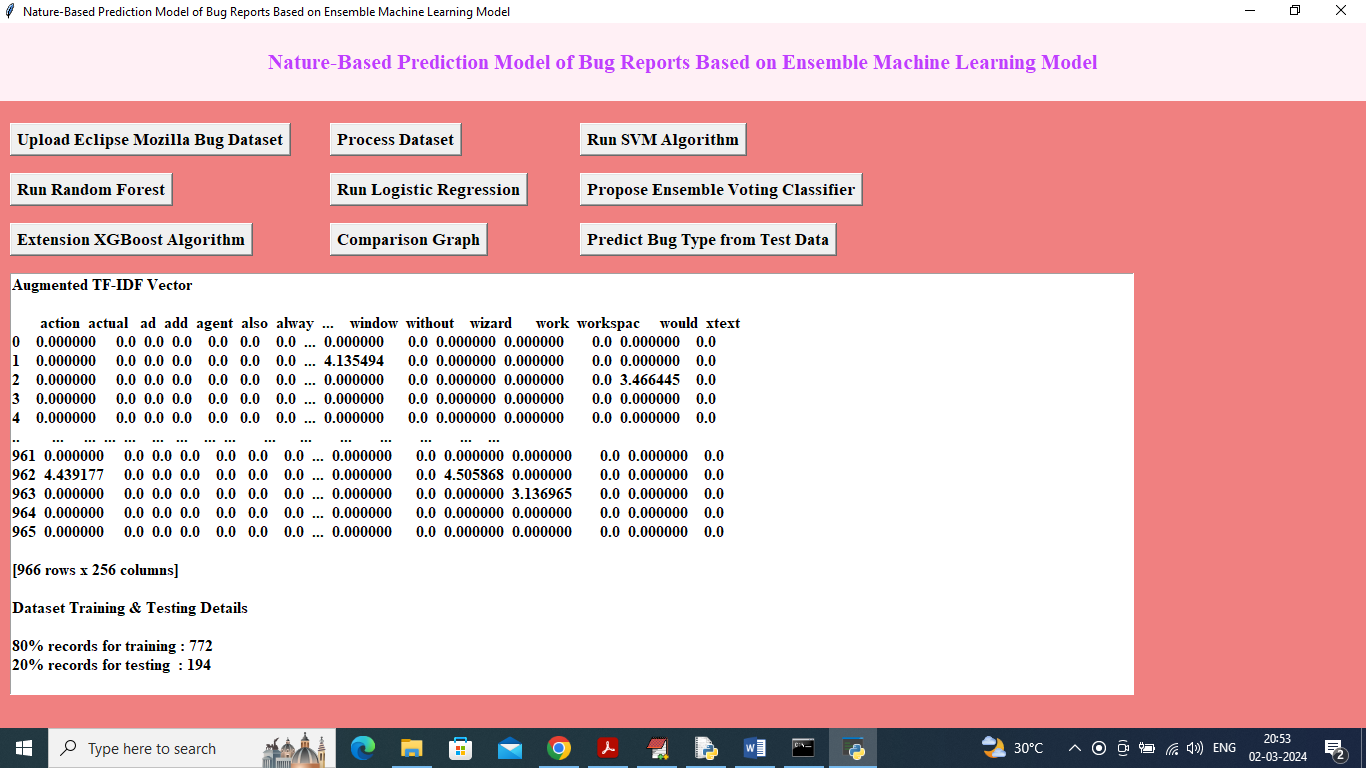
In above screen click on ‘Upload Eclipse Mozilla Bug Dataset’ button to upload dataset and get below page



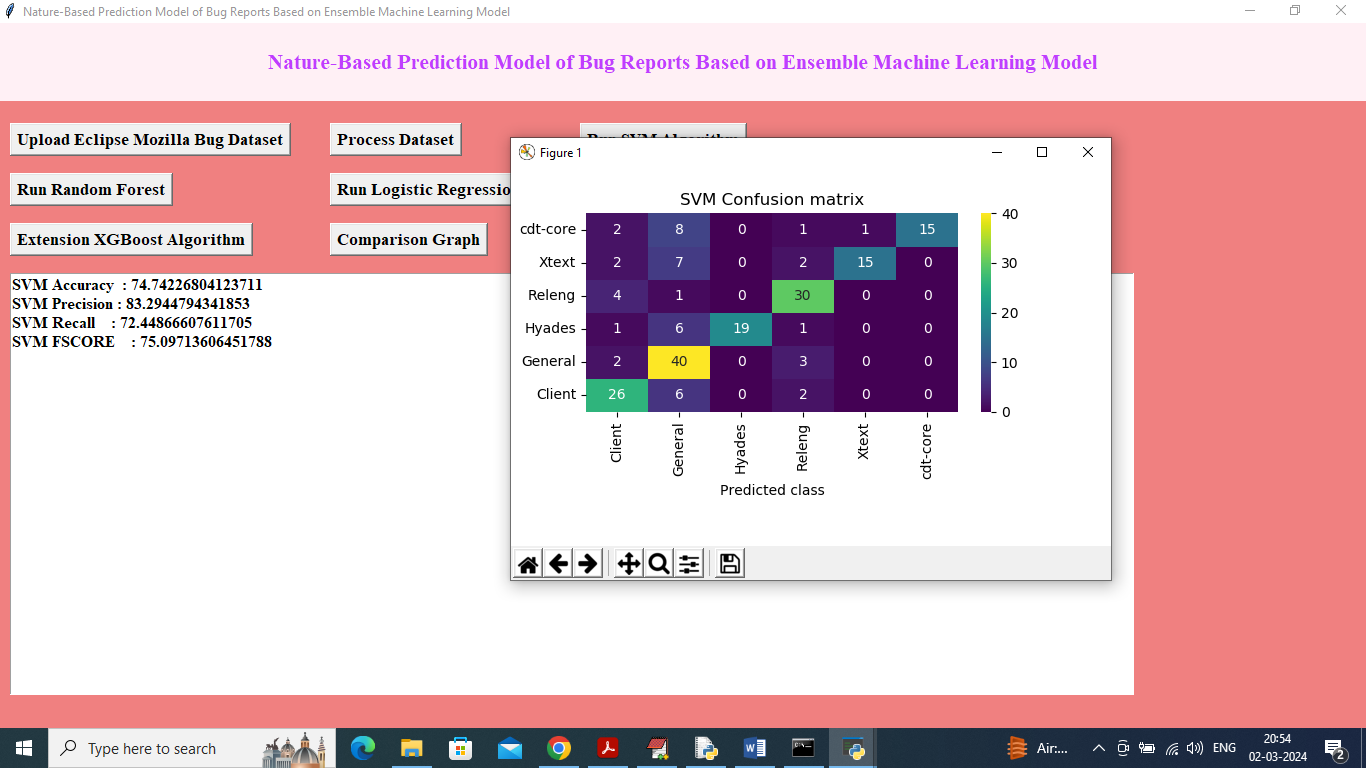
In above screen selecting and uploading dataset and then click on ‘Open’ button to select dataset and get below page



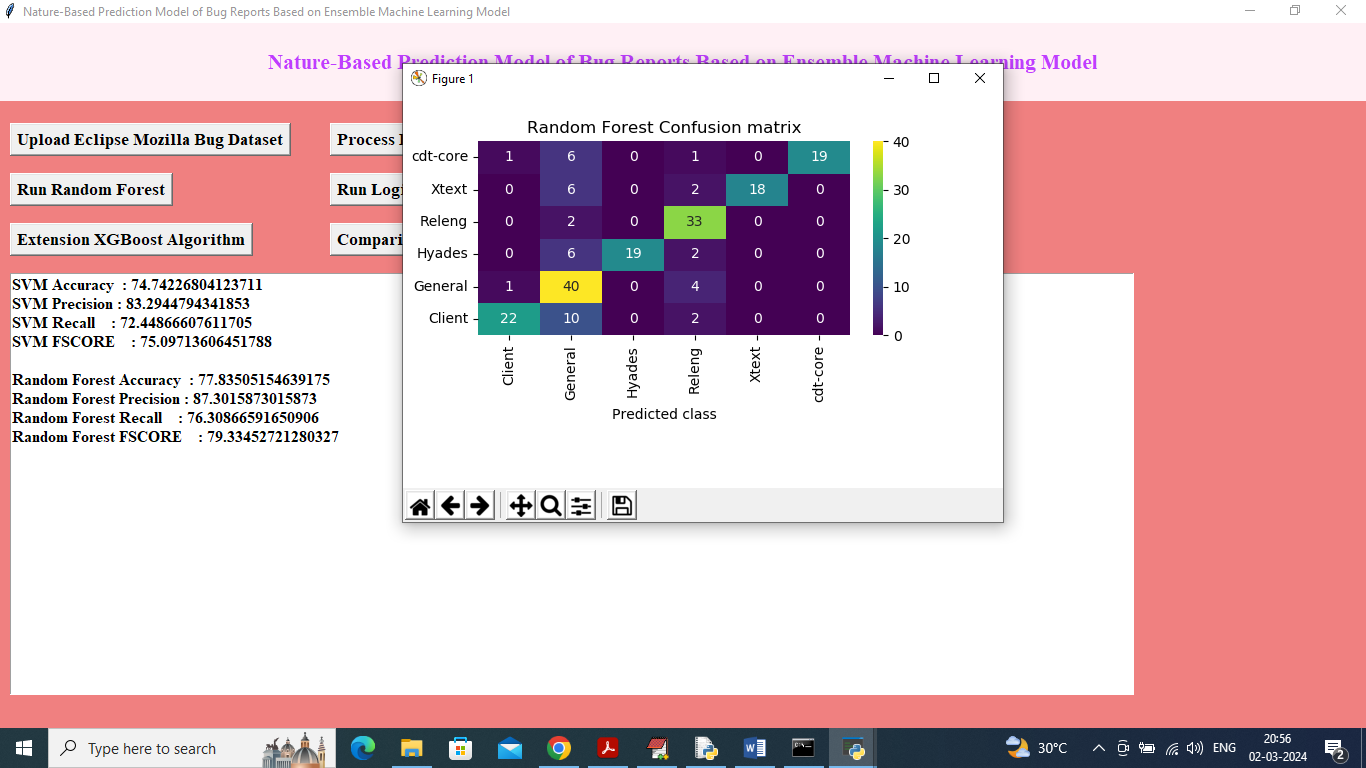
In above screen dataset loaded and can see total number of records loaded along with different class labels of type bug and now click on ‘Process Dataset’ button to clean dataset and then convert to augmented Vector



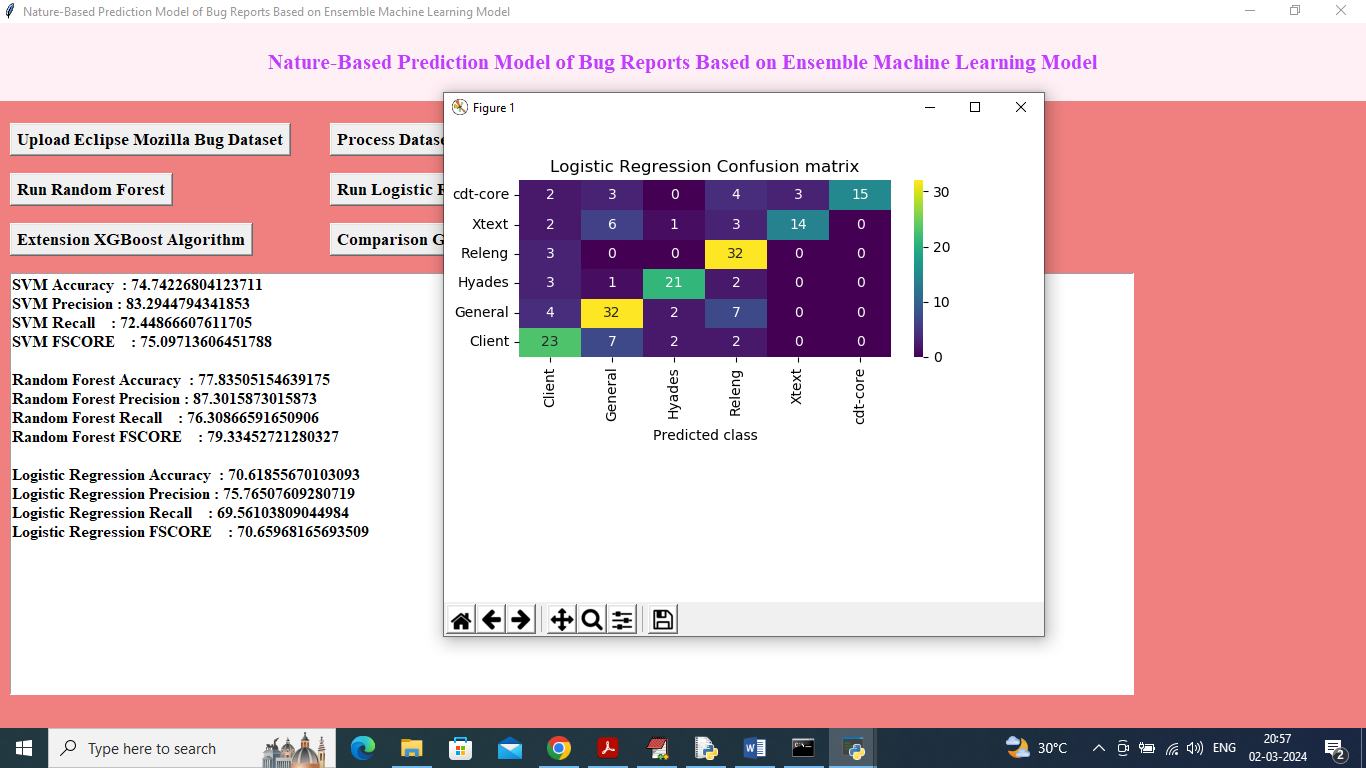
In above screen can see TFIDF vector generated where words can be seen on top and its average frequency in columns in next rows and can see train and test size. Now click on ‘Run SVM Algorithm’ button to train SVM and get below page



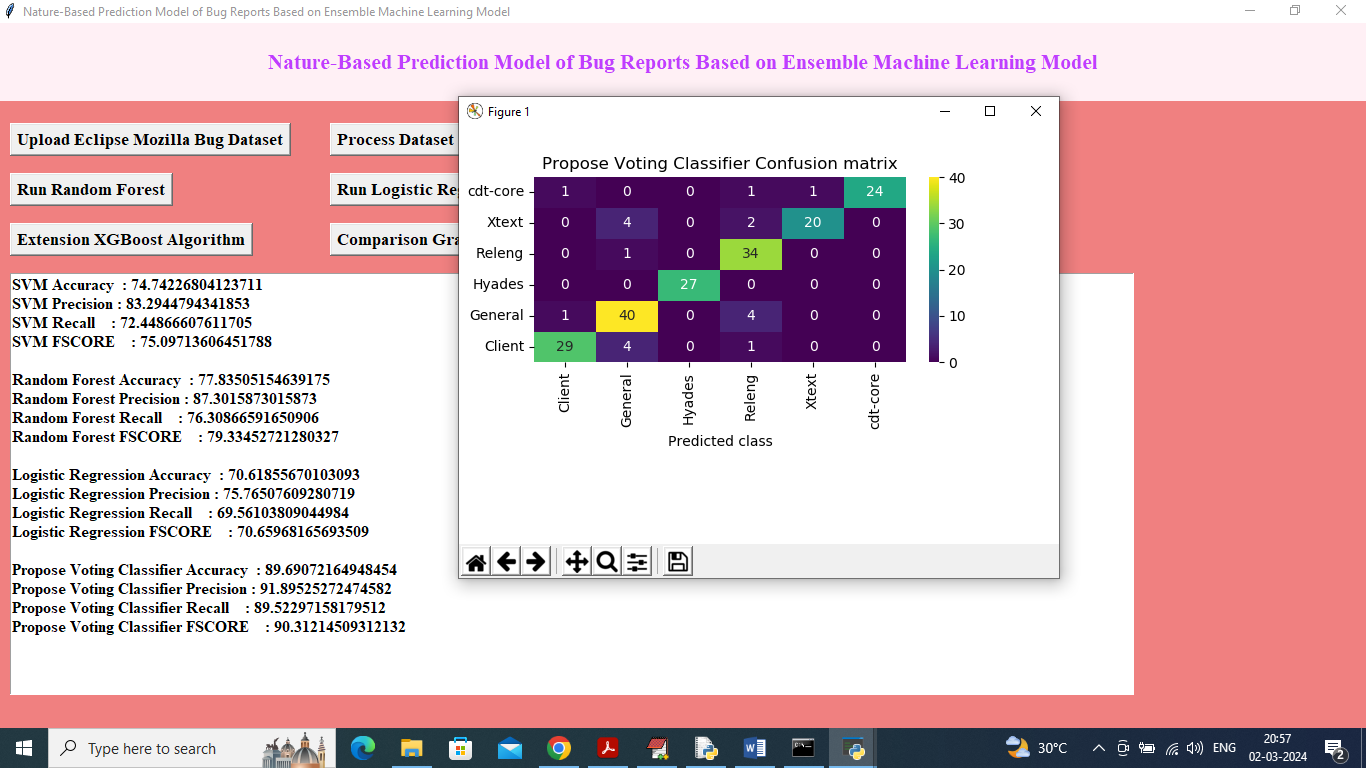
In above screen SVM got 74% accuracy and can see other metrics like precision, recall and FSCORE. In confusion matrix graph x-axis represents predicted BUG TYPE and y-axis represents True Bug Type and then all different colour boxes in diagnol represents correct prediction count and remaining blue boxes represents incorrect prediction count which are very few. Now click on ‘Run Random Forest’ button to get below output



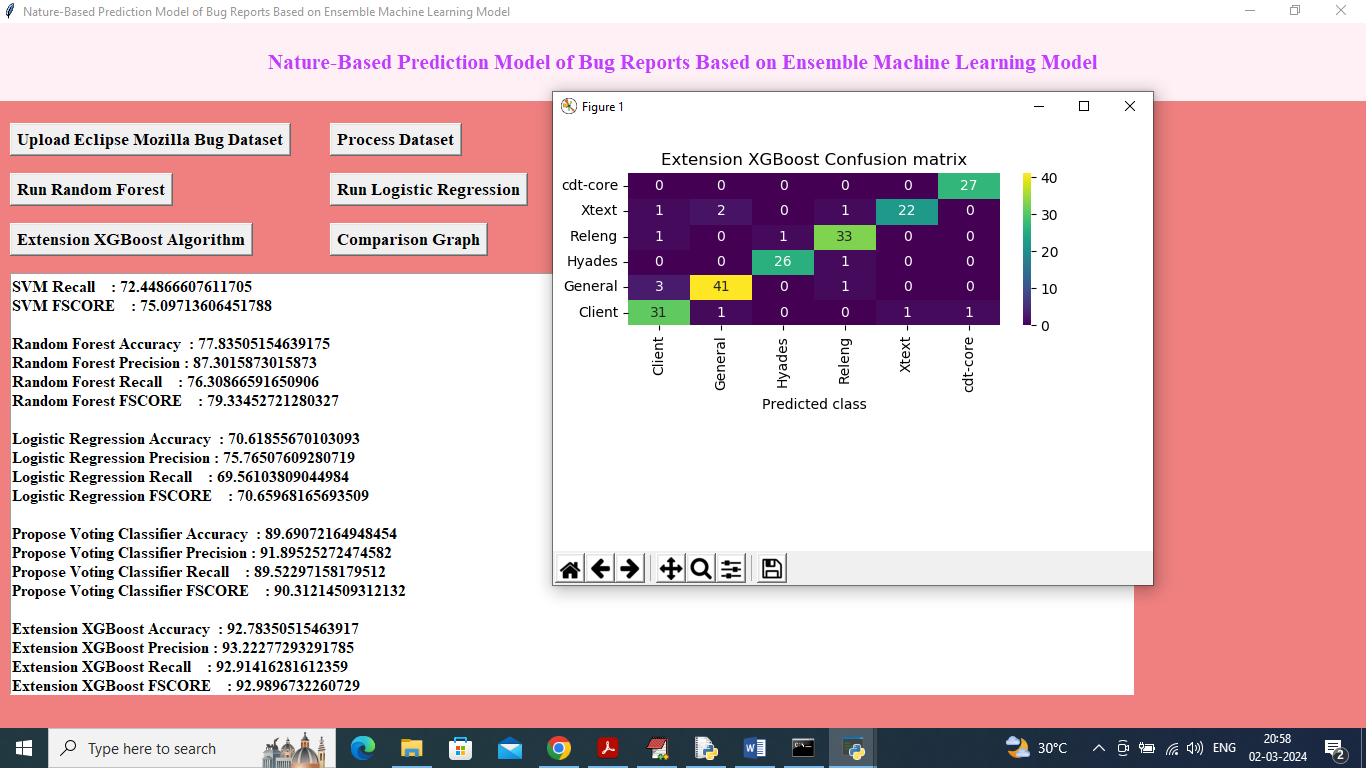
In above screen Random Forest got 77% accuracy



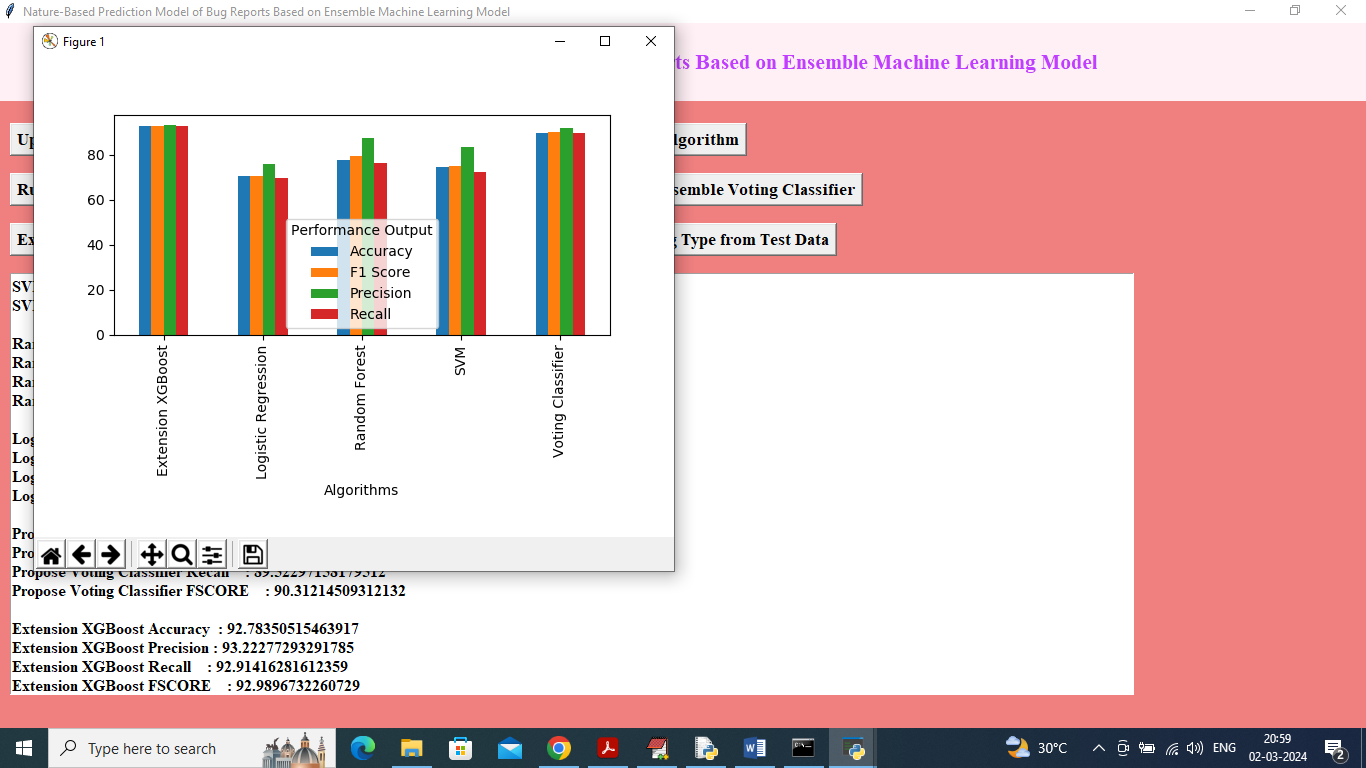
In above screen Logistic Regression got 70% accuracy



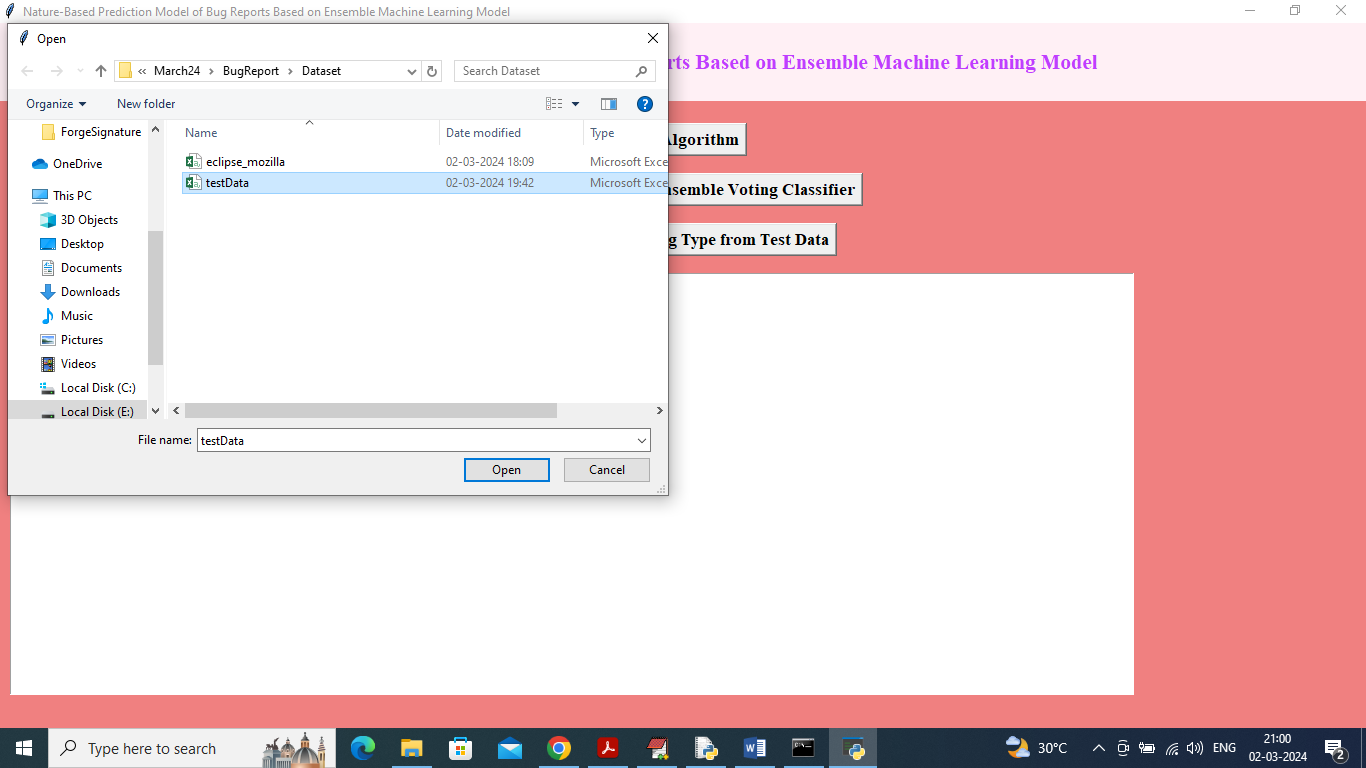
In above screen propose Voting Classifier got high accuracy as 89% and this accuracy may vary between 85 to 95% for different run as test data is dynamic split. Now click on Extension XGBOOST button



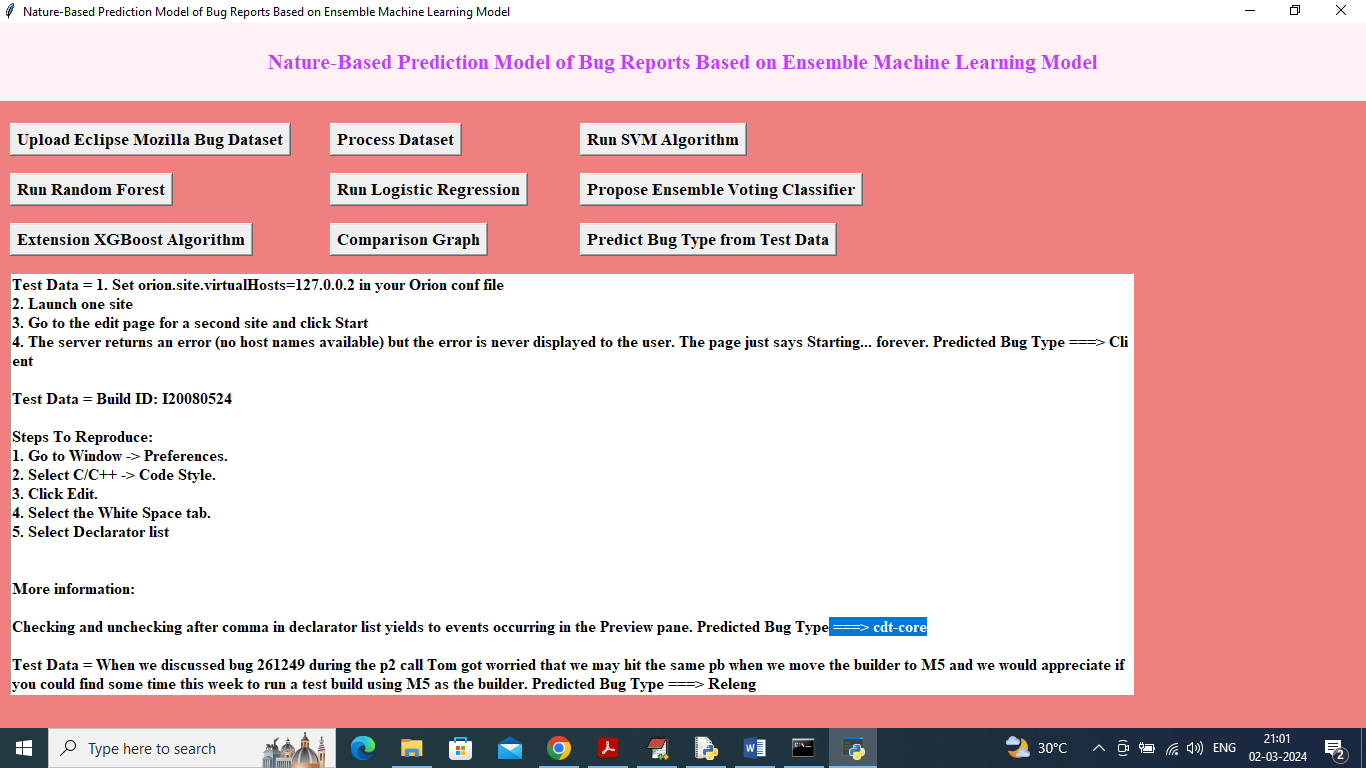
In above screen extension XGBOOST got 92% accuracy and now click on ‘Comparison Graph’ button to get below graph



In above graph x-axis represents algorithm names and y-axis represents accuracy and other metrics in different colour bars and in all algorithms Extension got high accuracy and now click on ‘Predict Bug Type from Test Data’ button to upload test data and get below output



In above screen selecting and uploading test data file and then click on ‘Open’ button to get below output



In above screen can see all text data from BUG and the after arrow symbol =🡺 can see predicted Bug Type and below is another sample

