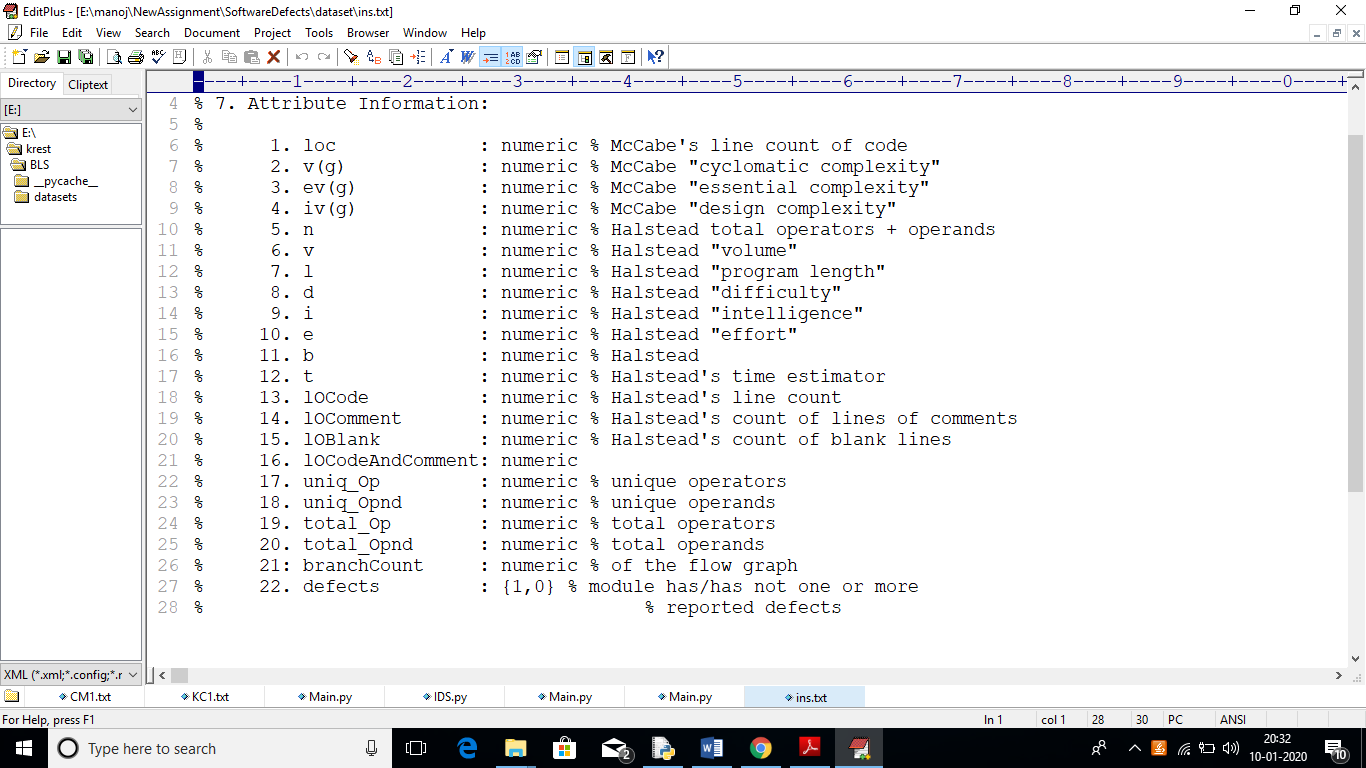
Software Defect Estimation Using Machine Learning Algorithms

In this paper author is evaluating performance of various machine learning algorithms such as SVM, Bagging, Naïve Bayes, Multinomial Naïve Bayes, RBF, Random Forest and Multilayer Perceptron Algorithms to detect bugs or defects from Software Components. Defects will occur in software components due to poor coding which may increase software development and maintenance cost and this problem leads to dis-satisfaction from customers. To detect defects from software components various techniques were developed but right now machine learning algorithms are gaining lots of popularity due to its better performance. So in this paper also author is using machine learning algorithms to detect defects from software modules. In this paper author is using dataset from NASA Software components and the name of those datasets are CM1 and KC1. I am also using same datasets to evaluate performance of above mention algorithms.

Dataset contains following columns showing in below screen



In dataset total 22 columns are there and last column refers to defects which has two values 0 and 1, if 0 means no defects and 1 means software contains defect. In above screen loc, v(g), ev(g) and others are the names of dataset columns. Beside all columns you can see column description also. This datasets I saved inside ‘dataset’ folder.

Using those datasets we will train machine learning algorithms and generate a model and whenever user gives new test software values then algorithm will apply train model on that new test values to predict whether given software values contains defect or not.

Algorithm details

**SVM Algorithm**: Machine learning involves predicting and classifying data and to do so we employ various machine learning algorithms according to the dataset. SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyper plane which separates the data into classes. In machine learning, the radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. In particular, it is commonly used in support vector machine classification. As a simple example, for a classification task with only two features (like the image above), you can think of a hyper plane as a line that linearly separates and classifies a set of data.

Intuitively, the further from the hyper plane our data points lie, the more confident we are that they have been correctly classified. We therefore want our data points to be as far away from the hyper plane as possible, while still being on the correct side of it.

So when new testing data is added, whatever side of the hyper plane it lands will decide the class that we assign to it.

**Random Forest Algorithm:** it’s an ensemble algorithm which means internally it will use multiple classifier algorithms to build accurate classifier model. Internally this algorithm will use decision tree algorithm to generate it train model for classification.

**Bagging**: This algorithms work similar to learning tree the only difference is voting concept where each class will get majority of votes based on values close to it and that class will form a branch. If new values arrived then that new value will applied on entire tree to get close matching class.

**Naive Bayes**: Naive Bayes which is one of the most commonly used algorithms for classifying problems is simple probabilistic classifier and is based on Bayes Theorem. It determines the probability of each features occurring in each class and returns the outcome with the highest probability.

**Multinomial Naive Bayes:** Multinomial Naive Bayes classifier is obtained by enlarging Naive Bayes classifier. Differently from the Naive Bayes classifier, a multinomial distribution is used for each features.

**Multilayer Perceptron**: Multilayer Perceptron which is one of the types of Neural Networks comprises of one input layer, one output layer and at least one or more hidden layers. This algorithm transfers the data from the input layer to the output layer, which is called feed forward. For training, the back propagation technique is used. One hidden layer with (attributes + classes) / 2 units are used for this experiment. Each dataset has 22 attributes and 2 classes which are false and true. We determined the learning rate as 0.3 and momentum as 0.2 for each dataset.

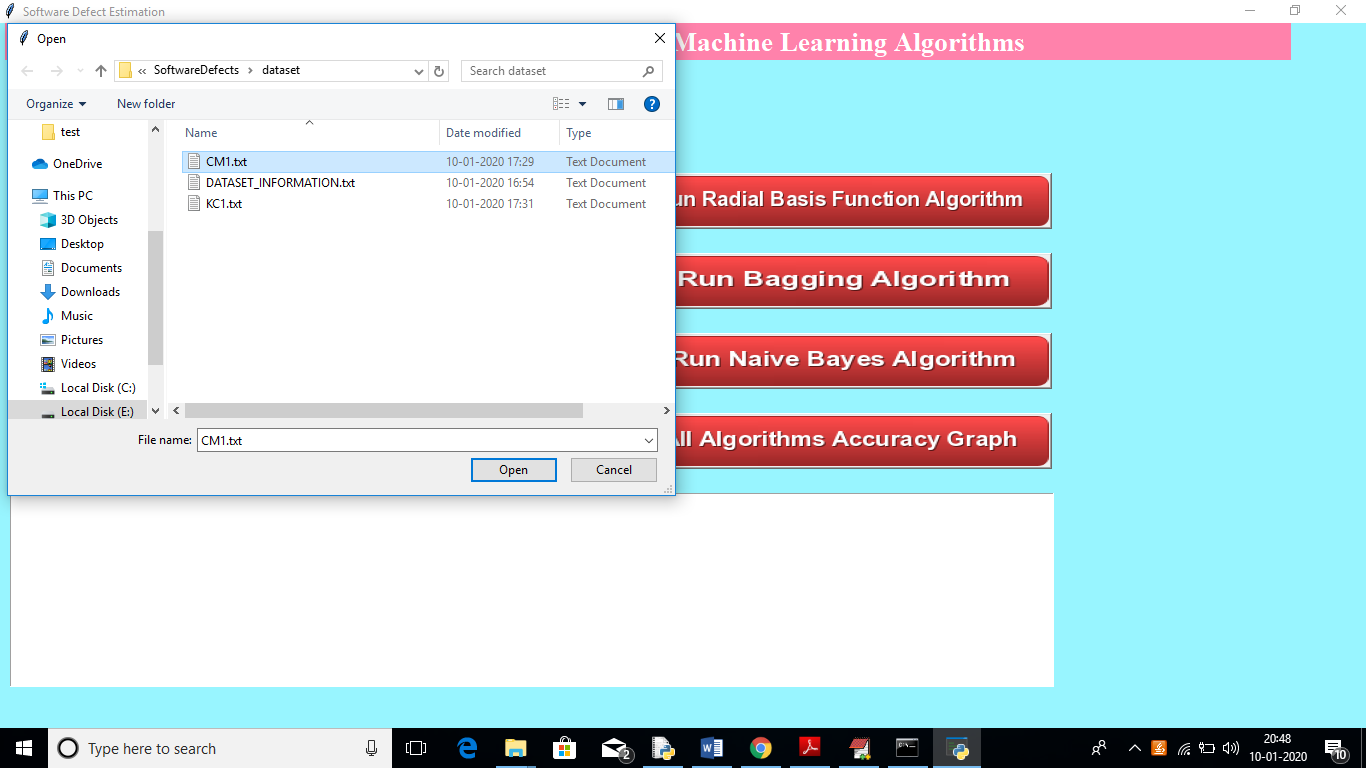
**Radial Basis Function**: Radial Basis Function Network includes an input vector for classification, a layer of RBF neurons, and an output layer which has a node for each class. Dot products method is used between inputs and weights and for activation sigmoidal activation functions are used in MLP while in RBFN between inputs and weights Euclidean distances method is used and as activation function, Gaussian activation functions are used.

Screen shots

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



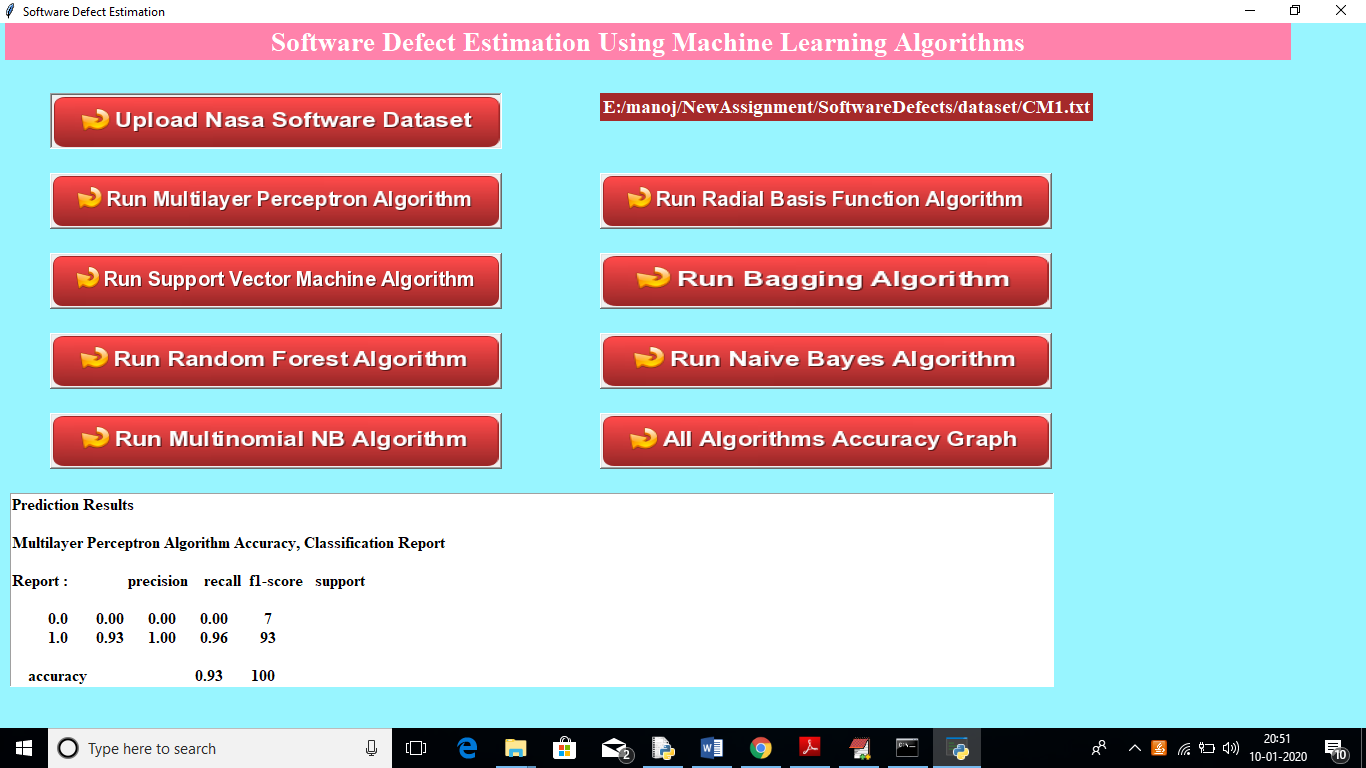
In above screen click on ‘Upload Nasa Software Dataset’ button to upload dataset



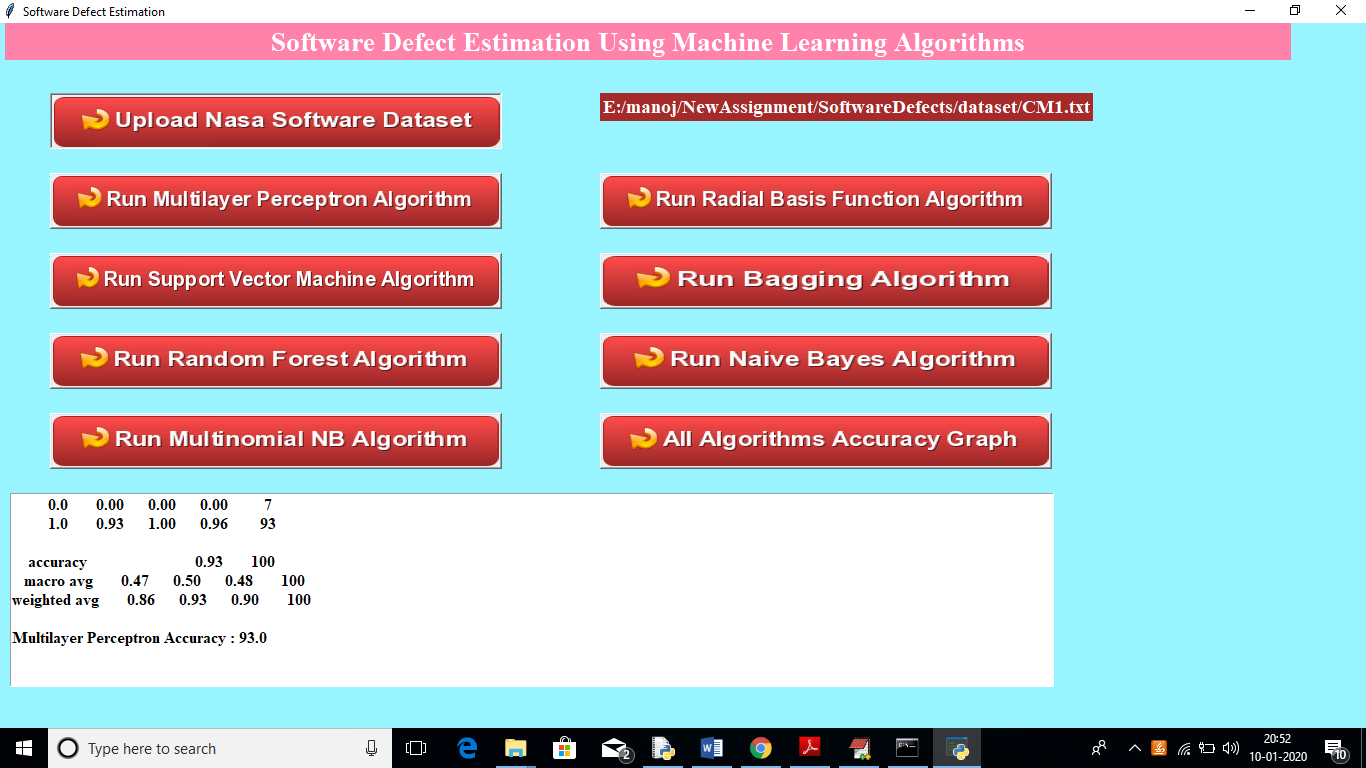
In above screen uploading ‘CM1.txt’ dataset and information of this dataset you can read from internet of ‘DATASET\_INFORMATION’ file from above screen. After uploading dataset will get below screen



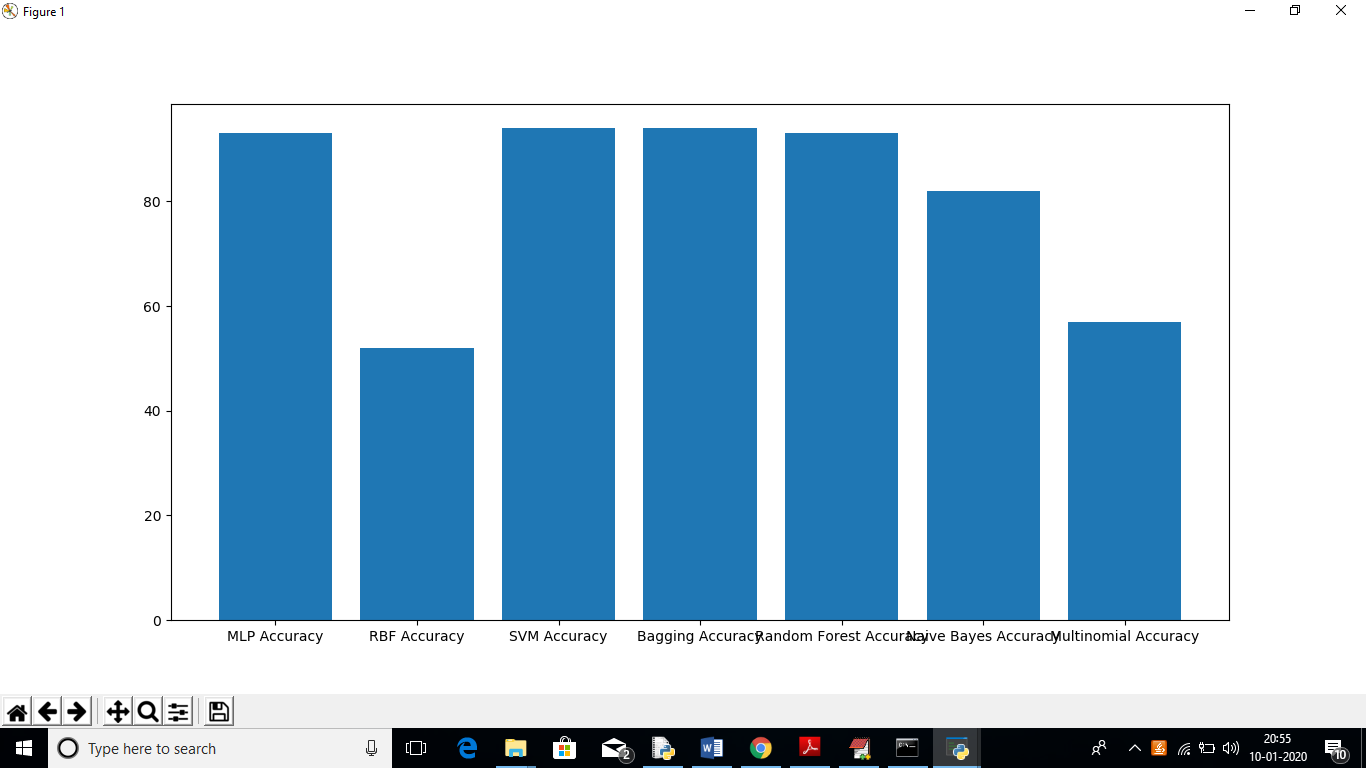
In above screen we can see total dataset size and training size records and testing size records application obtained from dataset to build train model. Now click on ‘Run Multilayer Perceptron Algorithm’ button to generate model and to get its accuracy



In above screen we can see multilayer perceptron fmeasure, recall and accuracy values and scroll down in text area to see all details.



In above screen we can see multilayer perceptron accuracy is 93%. Similarly you click on all other algorithms button to see their accuracies and then click on ‘All Algorithms Accuracy Graph’ button to see all algorithms accuracy in graph to understand which algorithm is giving high accuracy.



In above graph x-axis represents algorithm name and y-axis represents accuracy of those algorithms. In all algorithms we can see MLP, Bagging is giving better accuracy.