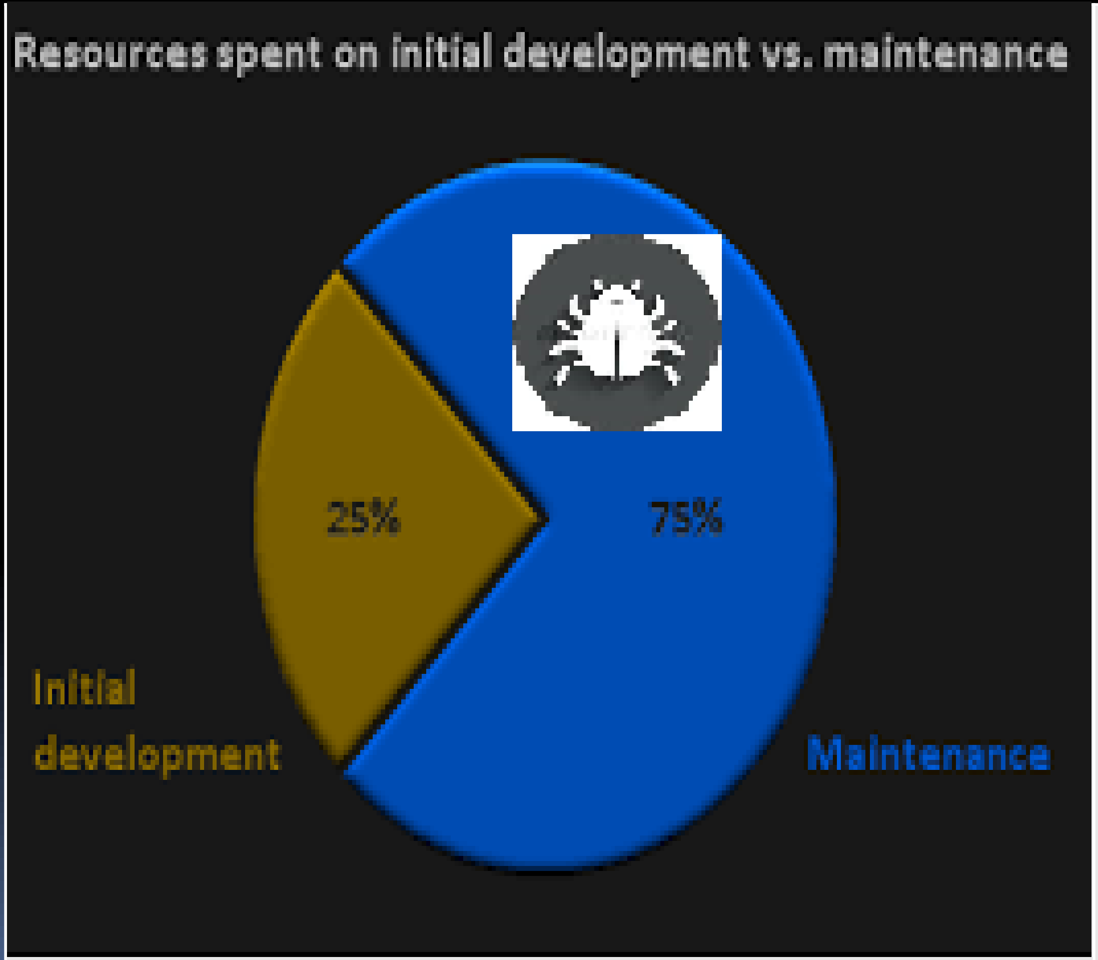
BENCHMARKING MACHINE LEARNING TECHNIQUES FOR SOFTWARE DEFECT DETECTION

Presented By:
Sheenam Sharma
Masters of Computer Science

OUTLINE

- Introduction
- Related Work
- Dataset
- Supervised techniques for machine learning
    - Classification
        - Various classification algorithms
- Unsupervised techniques for machine learning
    - Clustering
- Performance indicators
- Results
- Conclusion
- Discussion

INTRODUCTION



❖ Software maintenance is the most challenging task.

❖Good to fix a bug before the deliverable.

❖Machine learning can help in analyzing the data and retrieval of useful information which can help developers in detecting defects.

❖Machine learning techniques used for bug detection:

  - ❖Classification
  - ❖Clustering

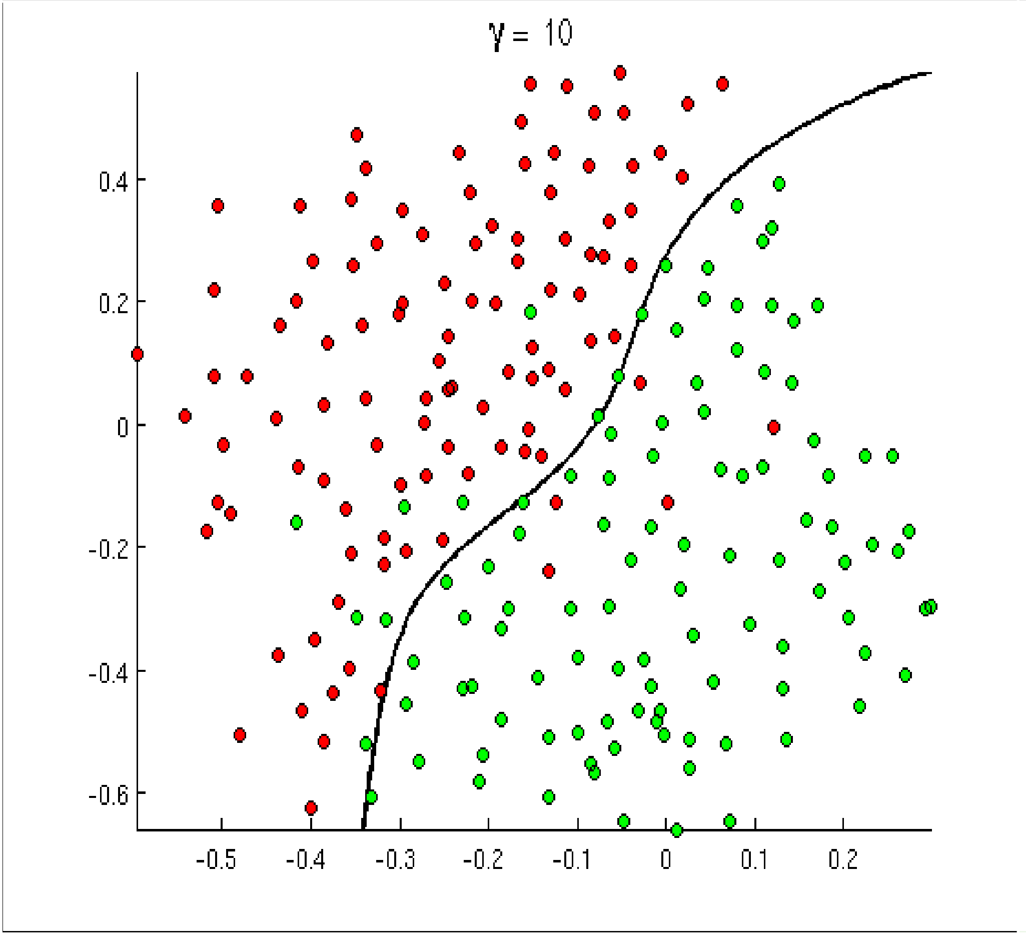
RELATED WORK

- Classification algorithms have been benchmarked using AUC curves. AUC was selected as the authors believed that it is the most informative source for benchmarking.
- Some authors focused only on the decision tree classification algorithms present in Weka.

DATA SET

- The datasets from PROMISE data repository were used in the experiments.
- The datasets were collected from real software projects by NASA and have many software modules.
- Public domain datasets in the experiments were used as the research was a benchmarking procedure of defect prediction.
- Weka tool was used for performing experiments.

CLASSIFICATION



❖Classification is a data mining and machine learning approach.

❖In software bug prediction it involves categorization of software modules into defective or non-defective that is denoted by a set of software complexity metrics by utilizing a classification model.

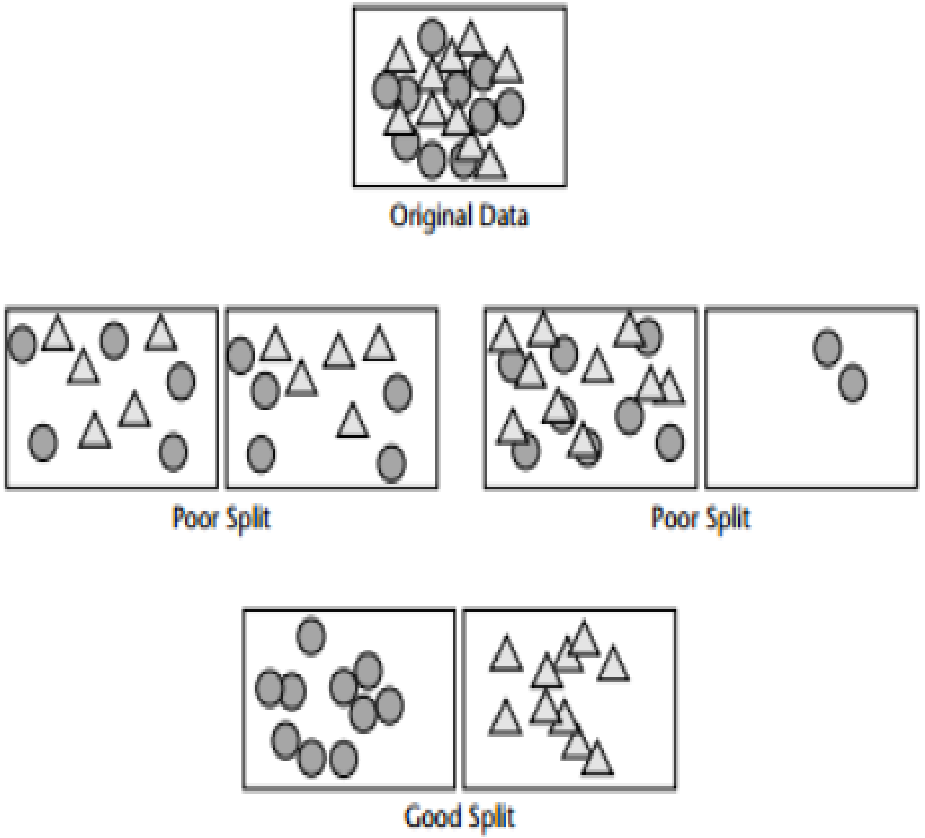
CLASSIFICATION ALGORITHMS

- Decision Tree
- Bagging
  - Random Forest
- Boosting
- SMO or SVM
- ZeroR

DECISION TREE

❖DT builds a tree-like structure by recursively partitioning the input attribute space into a set of non-overlapping spaces. A DT can also be seen as a set of decision rules from the root to the leaves. It is often used due to its natural interpretation.

❖A DT takes one attribute, and divides the instances forming a tree.



BAGGING

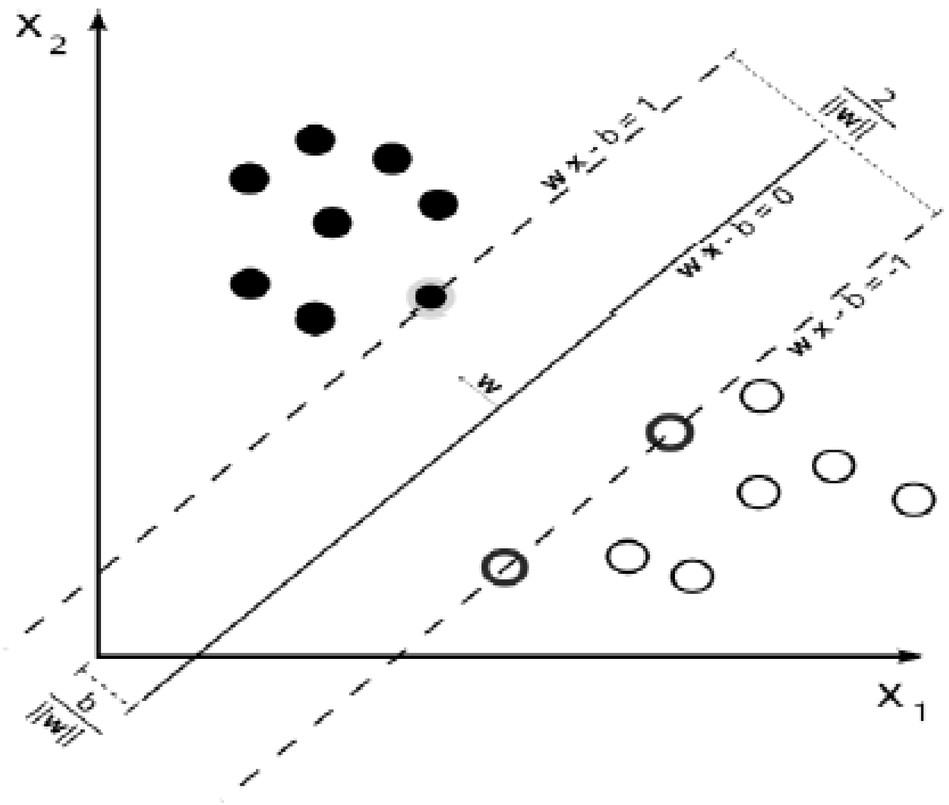
- Bagging uses multiple weak learners to form a strong learner that has better predictive performance than individual weak learners.
- A huge data set is divided into small chunks and base classifier is selected which operates separately but concurrently on these chunks. In the last, results from these classifiers are voted so as to get the best outcome.
- Random Forest is an example of Bagging.

BOOSTING

- Just like Bagging, Boosting also uses multiple weak learners to form a strong learner that has better predictive performance than individual weak learners.
- Unlike Bagging, it divides the problem into small chunks and one chunk is selected on which base classifier operates. From the results, instances which were wrongly classified are selected and classifier is again trained on it so as to increase the accuracy.

SVM

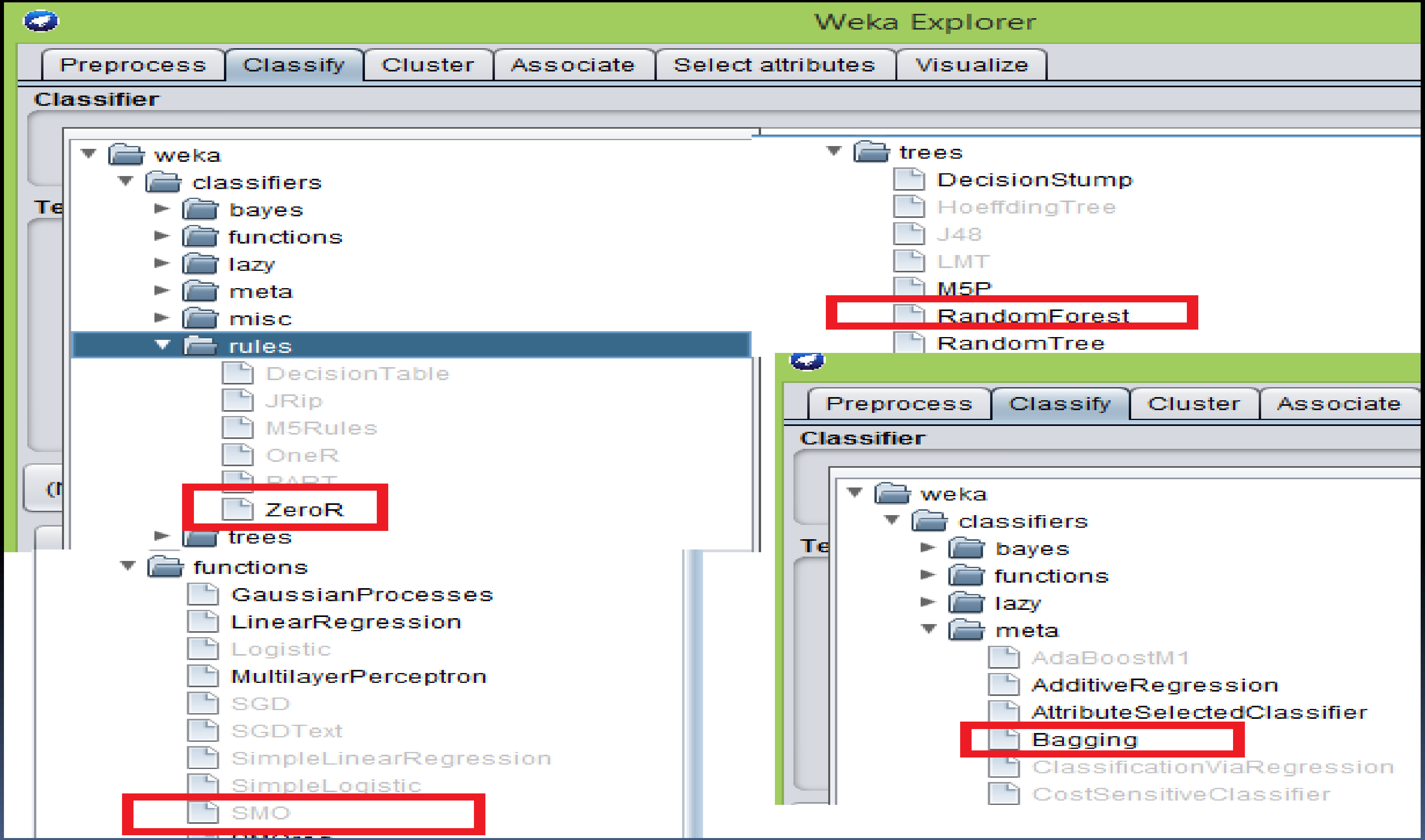
- Suppose we have a set of training examples {x1, x2, …., xn}, where xi⊆Rd along with their class labels {y1, y2, …., yn} and yi ∈ {-1, 1}. The task of SVM is to find a hyper plane to separate the data with the maximum margin between the hyper plane and the nearest data points of each class.



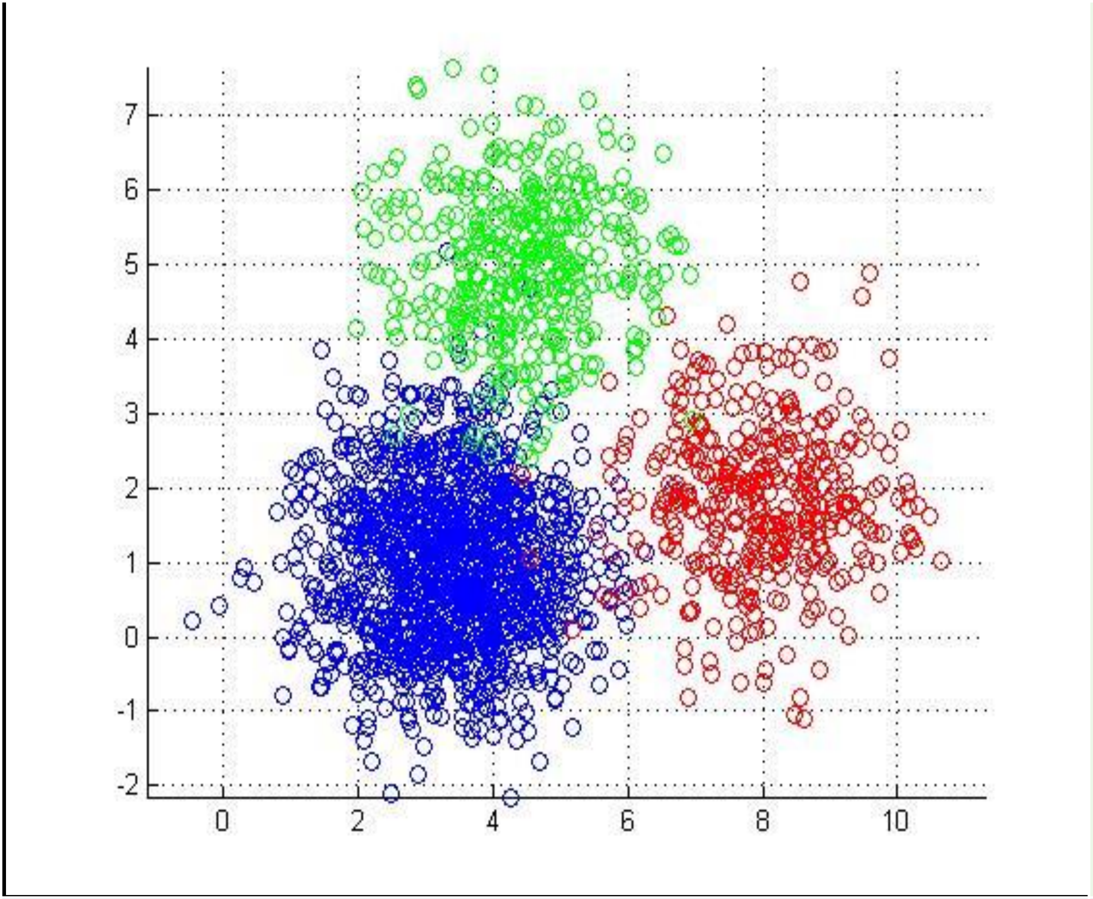
ZEROR

- This classification algorithm is used mostly as a baseline algorithm for experiments.
- The results of these classifier are based on majority.
- Suppose class A has 51 instances and class B has 48 instances, this classifier will classify all the instances in class A.

CLASSIFICATION ALGORITHMS IN WEKA



CLUSTERING

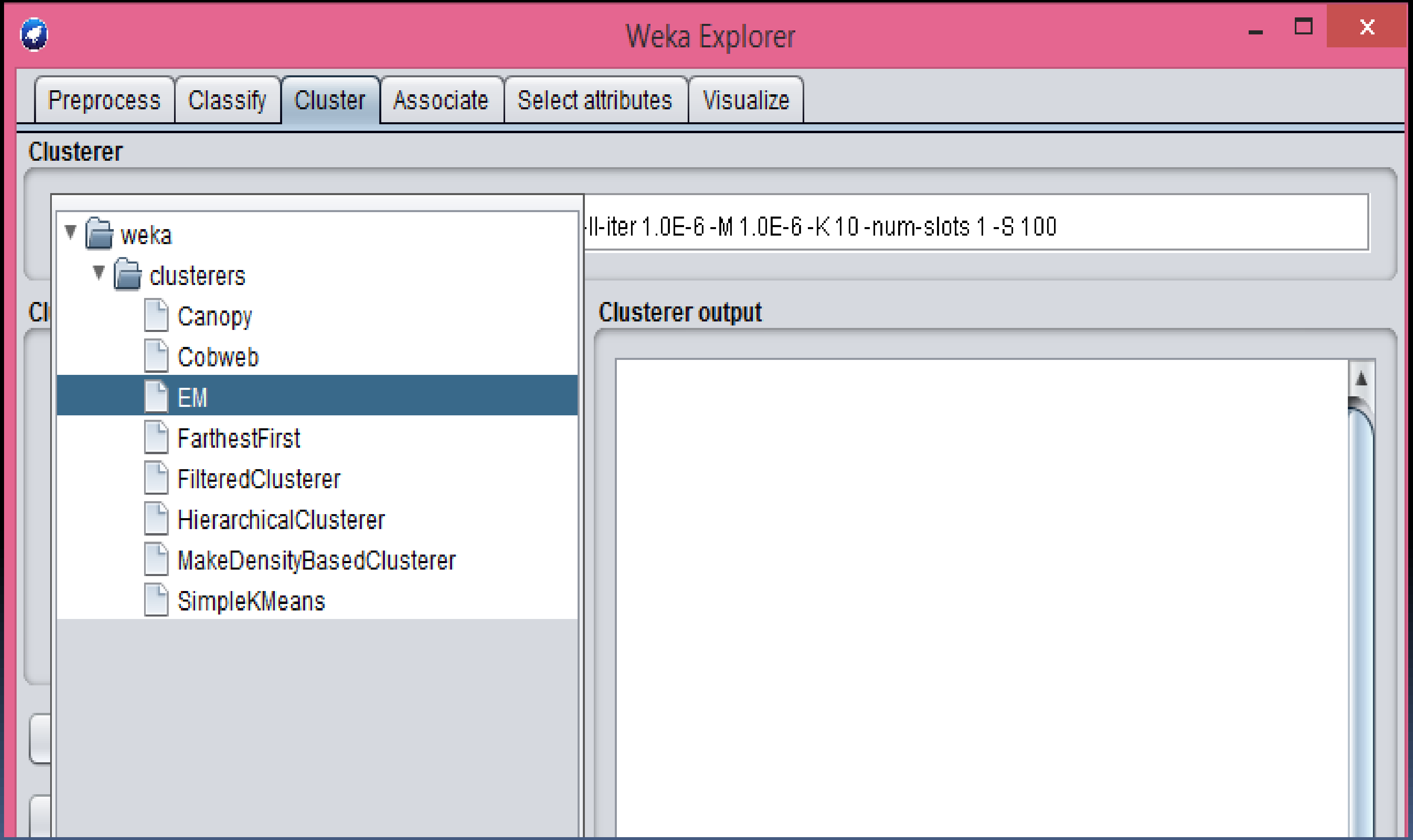


❖Clustering is method that moves data points among a set of clusters until similar item clusters are formed or a desired set is acquired.

❖Clustering methods make assumptions about the data set.

❖ If that assumption holds, then it results into a good cluster.

CLUSTERING ALGORITHMS IN WEKA



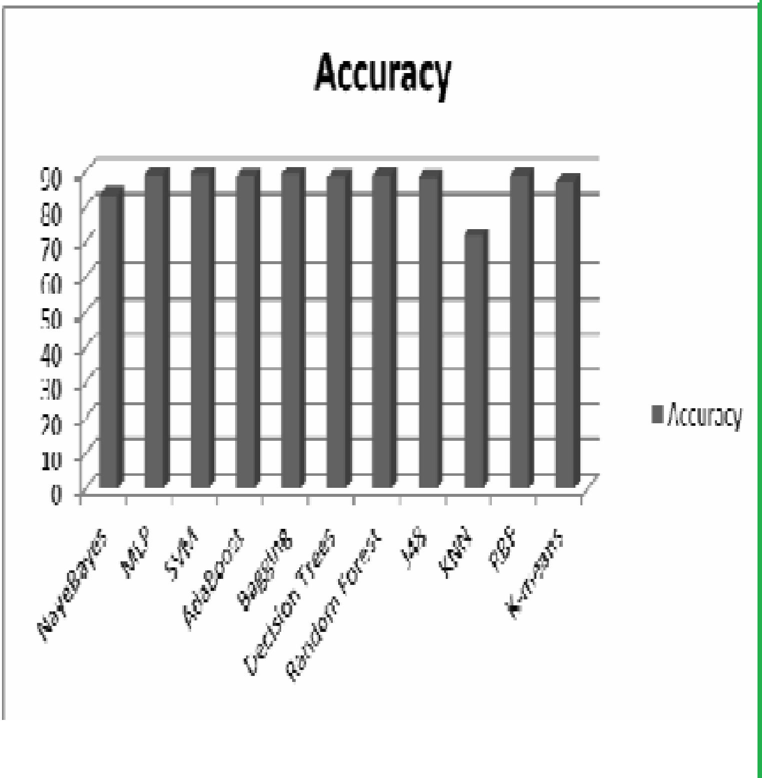
-ll-iter 1.0E-6 -M 1.0E-6 -K 10 -num-slots 1 -S 100

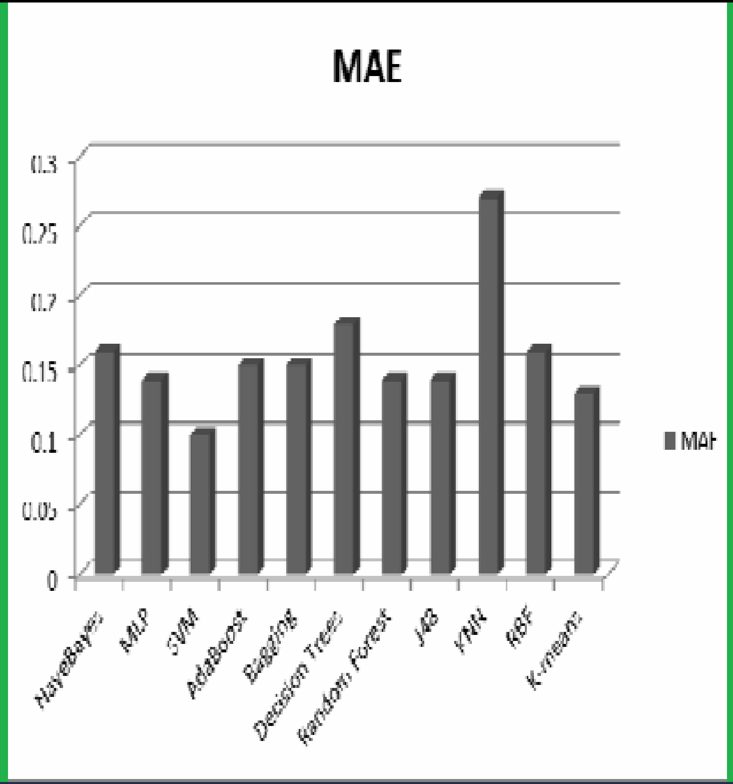
- weka
  - clusterers
    - Canopy
    - Cobweb
    - EM
    - FarthestFirst
    - FilteredClusterer
    - HierarchicalClusterer
    - MakeDensityBasedClusterer
    - SimpleKMeans

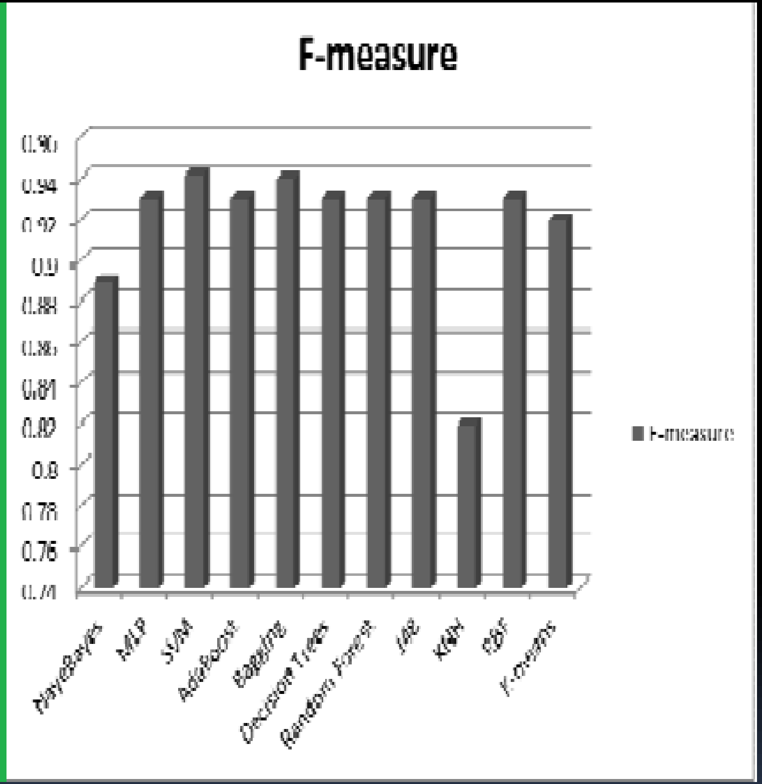
PERFORMANCE INDICATORS

- Accuracy = (correctly classified software bugs/ Total software bugs) * 100
- Precision= TP / (TP+FP)
- Recall= TP / (TP + FN)
- F-measure= (2 * precision * recall) / (Precision + recall)
- Mean Absolute Error (MAE)

EXPERIMENTAL RESULTS


Accuracy


MAE


F-measure

CONCLUSION

From the experiments it was concluded that SVM and Bagging had maximum accuracy and f-measure. Moreover, MAE for both these algorithms was low. Hence, the authors concluded that these two techniques performed well on the bug's dataset.

REFERENCES

- Luiz Fernando Capretz and Faheem Ahmed Saiqa Aleem, "BENCHMARKING MACHINE LEARNING TECHNIQUES FOR SOFTWARE DEFECT DETECTION," International Journal of Software Engineering & Applications (IJSEA), vol. 6, no. 3, pp. 11-23, May 2015.
- J.Van Ryzin, Classification and Clustering. New York: Harcourt Brace Jovanovich.
- Sahil Saroop, "Exploring Mediatoil Imagery: A Content-Based Approach," University of Ottawa, Ottawa, Thesis 2016.
- David M. W POWERS, "EVALUATION: FROM PRECISION, RECALL AND F-MEASURE TO ROC, INFORMEDNESS, MARKEDNESS & CORRELATION," Journal of Machine Learning Technologies, vol. 2, no. 1, pp. 37-63, Feb. 2011.

DISCUSSION

- ❖In which other fields of software engineering can you think where machine learning techniques can be applied?
- ❖Can you think of areas other than software engineering where you can use machine learning?
- ❖Can the results concluded by the author be generalized on all the dataset's related to bugs?

THANK
Y☺U!