INDIAN INSTITUTE OF INFORMATION TECHNOLOGY, ALLAHABAD

VI Semester B.Tech in Information Technology

Data Mining Warehouse

Submitted by :

Group No : 12

Members : -

Abhishek Kumar Gupta (IIT2018187)

Puja Kumari (IIT2018191)

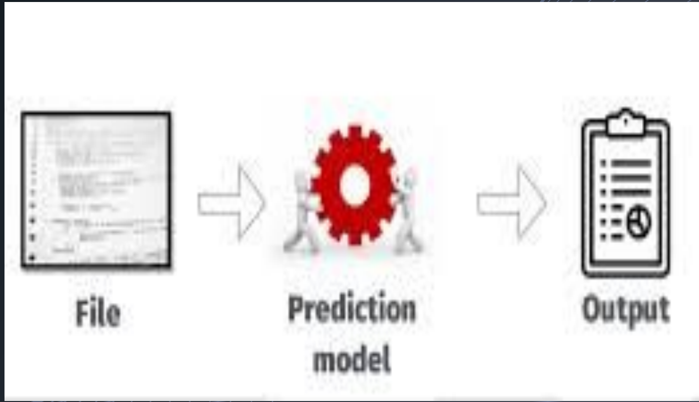
Prabha Kumari (IIT2018195)

Problem Identification & Definition

Software Defect Prediction Analysis using Machine Learning Algorithms

What is SDP?

- SDP is one of the activities of the testing phase of SDLC.
- Software Defect Prediction is an important aspect in order to ensure software quality.
- It Describe the relationship between various software metrics and software defect.



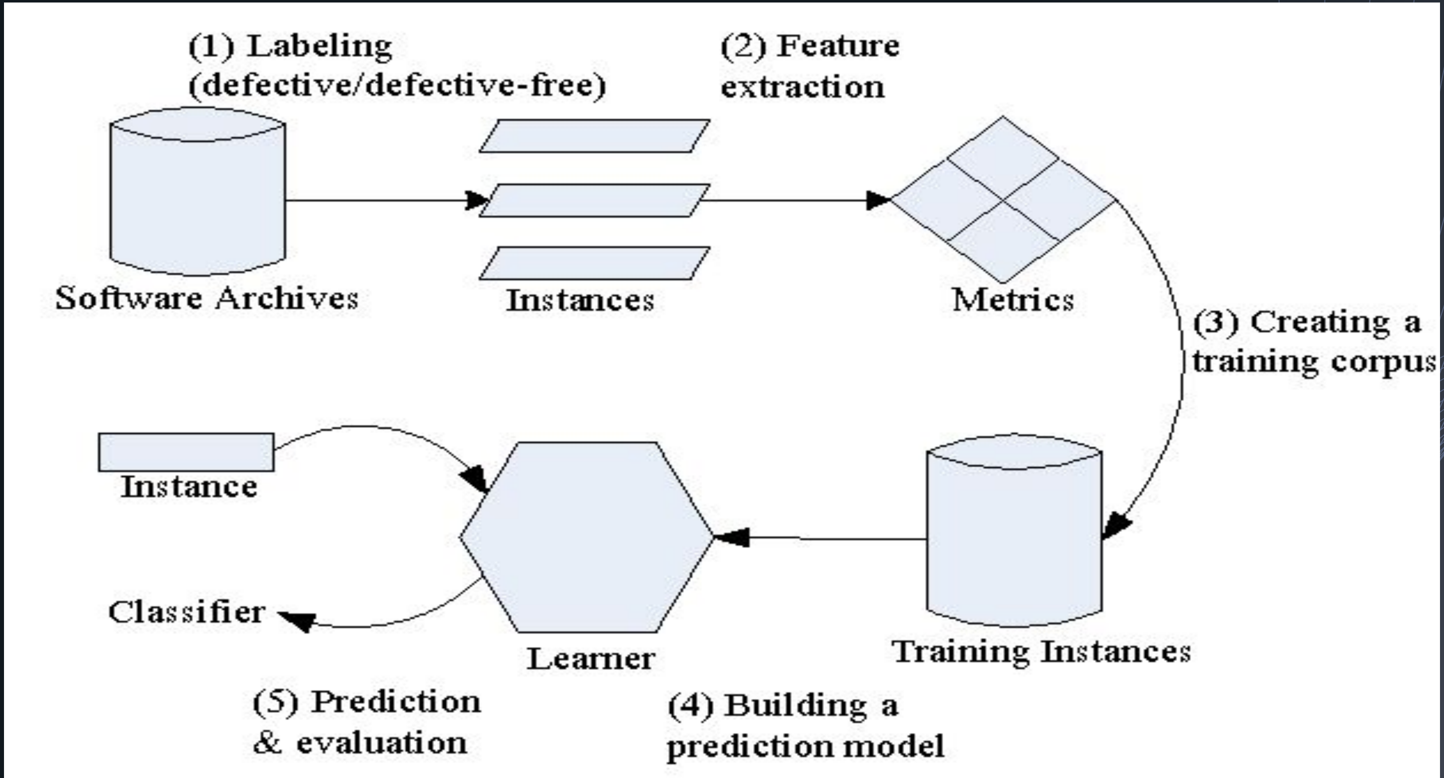
Abstract

We propose to extract a set of expressive features from an initial set of basic change measures using Artificial Neural Network (ANN), and then train a classifier based on the extracted features using Decision tree and compare it to three other methods wherein features are extracted from a set of initial change measures using dimensionality reduction techniques that include Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA) and Kernel PCA.

We use five open source datasets from NASA Promise Data Repository to perform this comparative study.

For evaluation, three widely used metrics: Accuracy, F1 scores and Areas under Receiver Operating Characteristic curve are used.

Defect Prediction Process



Introduction

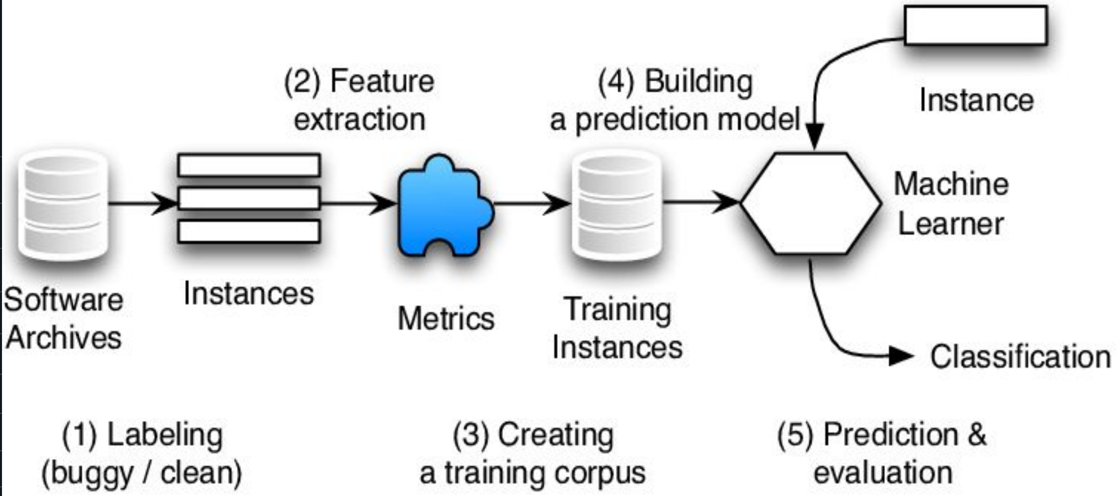
- In order to build high quality softwares, defect prediction has become an important aspect as a lot of time and effort is put in software testing and its debugging otherwise.
- Defect prediction techniques are proposed to help prioritize software testing and debugging; they can recommend software components that are likely to be defective to developers.
- A lot of parameters are considered while predicting whether a software is buggy or not which include
  - number of lines in the code,
  - its complexity,
  - the number of operators and operands used in the code and other factors.

We have considered a set of 22 initial features to predict whether the module is buggy or not .

Background

- Here, we will introduce the background of defect prediction techniques
- Defect Prediction
- The process of predicting code areas that contain defects is called Software defect prediction.
- It help developers allocate their testing efforts by first checking buggy code. It ensures the reliability of large -scale software.

Fig : Representing file-level defect prediction process.



File Level Defect Prediction Process

Algorithm of file level defect prediction process is following.

- Collect source code files (instances) from archives and level hem as buggy or clean.
  - File containing at least one post-release bug is labeled as buggy .
  - Otherwise the file is labeled as clean.
- Extract features (code metrics and CK features) from each file.
  - The instances with the corresponding features and labels to train classifiers using algorithms like ANN,LDA,PCA,KPCA.
- New instances are fed into trained classifier to predict whether the files are buggy or clean.

Dataset

The five datasets used for this project were taken from NASA Promise Dataset Repository http://promise.site.uottawa.ca/SERepository/datasets-page.html namely pc1, cm1, jm1, kc1, kc2 each having no missing values and 22 attributes that come from McCabe and Halstead features

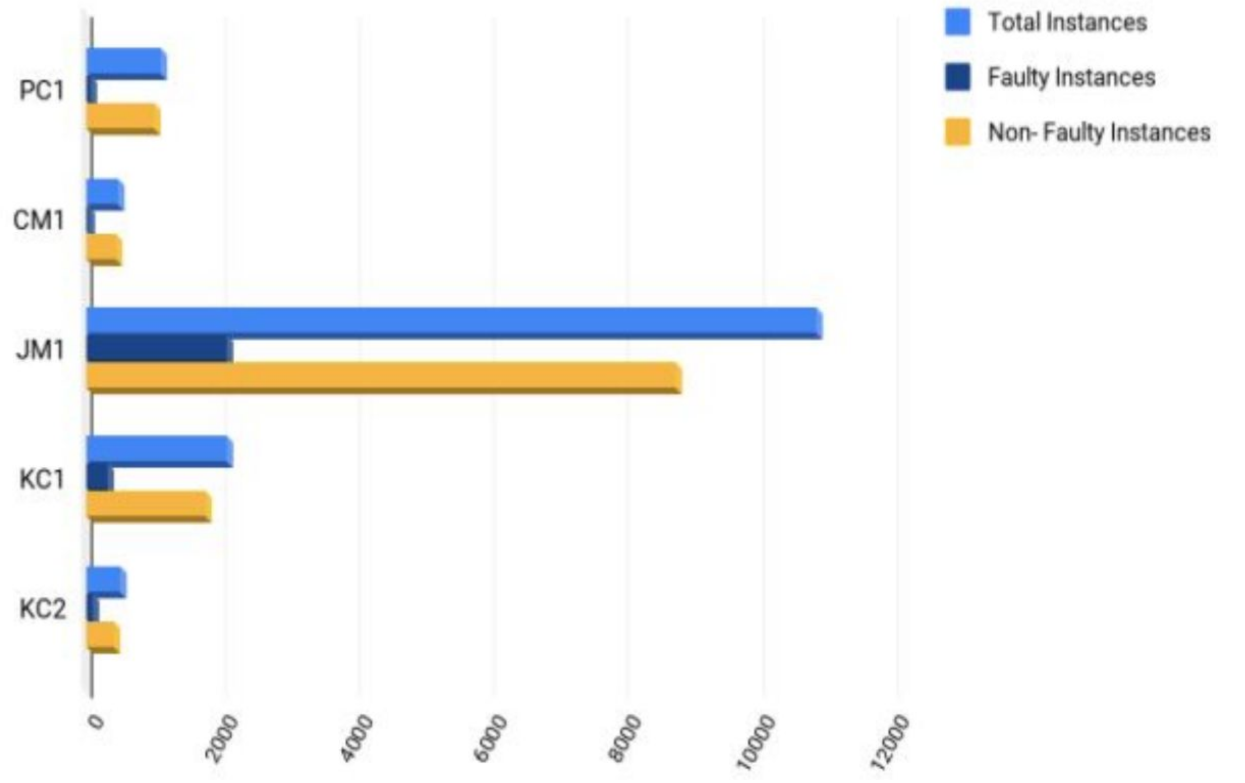


Figure 1: Dataset characteristics

Dataset	Language used	Total Instances	Defective Instances	Non-Defective Instances
PC1	C	1,109	77	1,032
CM1	C	498	49	449
JM1	C	10,885	2,106	8,779
KC1	C++	2,109	326	1,783
KC2	C++	522	105	415

Table 1: Characteristics of Datasets

Since the data had more instances of non buggy modules
So under sampling was done to prevent the model from being biased towards non buggy instances.

Literature Survey...

S.No.	Paper Title	Name of the Conference/journal (Year)	Methodology	Results	Paper Link
1.	Deep learning for just-in-time defect prediction	in QRS’15: Proc. of the International Conference on Software Quality, Reliability and Security, 2015	used learning algorithms to predict defects at change level. They made use of a deep learning algorithm to predict the same. They first created a Deep Belief Network to extract a set of expressive features from the initial set of linear features and then used Logistic Regression as a classifier to predict buggy and non-buggy changes.	Compare the result using accuracy f1 score and area under Roc	https://ieeexplore.ieee.org/document/7272910
2.	A large-scale empirical study of just-in-time quality assurance	IEEE Transactions on Software Engineering (Volume: 39, Issue: 6, June 2013)	In this paper we consider defect prediction models that focus on identifying defect-prone (“risky”) software changes instead of files or packages.To build a change risk model, we use a wide range of factors based on the characteristics of a software change, such as the number of added lines, and developer experience	Predictor achieves an average precision of 37 percent and recall of 67 percent for open source projects, which translates to an average improvement of 90 percent over the random predictor.	https://ieeexplore.ieee.org/document/6341763

3.	Software defect prediction analysis using machine learning algorithms	2017 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence	In this paper we have analyzed the most popular and widely used Machine Learning algorithms - ANN (Artificial Neural Network), PSO(Particle Swarm Optimization), DT (Decision Trees), NB(Naive Bayes) and LC (Linear classifier)	The results demonstrated the dominance of Linear Classifier over other algorithms in terms of defect prediction accuracy.	https://www.researchgate.net/publication/5924092_Learning_multiple_layers_of_representation
4.	Software Defect Prediction via Convolutional Neural Network	2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)	In this paper, we propose a framework called Defect Prediction via Convolutional Neural Network (DP-CNN), which leverages deep learning for effective feature generation	The experimental results show that in average, DP-CNN improves the state-of-the-art method by 12%	https://ieeexplore.ieee.org/document/8009936
5.	Personalized defect prediction	2013 28th IEEE/ACM International Conference on Automated Software Engineering (ASE)	This paper proposes personalized defect prediction-building a separate prediction model for each developer to predict software defects. As a proof of concept, we apply our personalized defect prediction to classify defects at the file change level	In this experiment result improves the F1-score by 0.01-0.06 compared to the traditional change classification	https://ieeexplore.ieee.org/document/6693087
6.	Automatically Learning Semantic Features for Defect Prediction	2016 IEEE/ACM 38th International Conference on Software Engineering (ICSE)	In this paper, we leverage Deep Belief Network (DBN) to automatically learn semantic features from token vectors extracted from programs' Abstract Syntax Trees (ASTs).	In this paper semantic features improve WPDP on average by 14.7% in precision, 11.5% in recall, and 14.2% in F1	https://ieeexplore.ieee.org/abstract/document/7886912
7.	Revisiting common bug prediction findings using effort-aware models	2010 IEEE International Conference on Software Maintenance	In this paper, we revisit two common findings in the bug prediction literature: 1) Process metrics (e.g., change history) outperform product metrics (e.g., LOC), 2) Package-level predictions outperform file-level predictions.	we find that if we test 20% of all modules based on the predicted fault density, we would detect 74% of faults using file-level models and 62% of faults using package-level models.	https://ieeexplore.ieee.org/document/5609530

8.	Recent advances in deep learning for speech research at Microsoft	In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013, pp. 8604–8608	This paper provides selected samples of our experiments on applying deep learning methods to advancing speech technology and related applications, including feature extraction, acoustic modeling, language modeling, speech understanding, and dialogue state estimation.	we presented experimental evidence that spectrogram features of speech are superior to MFCC with DNN, in contrast to the earlier long-standing practice with GMM-HMMs	https://www.researchgate.net/publication/261153438_Recent_advances_in_deep_learning_for_speech_research_at_Microsoft
9.	How, and why, process metrics are better	2013 35th International Conference on Software Engineering (ICSE)	In this paper we analyze the applicability and efficacy of process and code metrics from several different perspectives.	Our results suggest that code metrics, despite widespread use in the defect prediction literature, are generally less useful than process metrics for prediction. Second, we find that code metrics have high stasis; they don't change very much from release to release	https://ieeexplore.ieee.org/document/6606589
10.	Random Forests and Decision Trees	In International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012	We compared the classification results obtained from methods i.e. Random Forest and Decision Tree (J48). The classification parameters consist of correctly classified instances, incorrectly classified instances, F-Measure, Precision, Accuracy and Recall.	It can be concluded that the Random Forest achieves increased classification performance and yields results that are accurate and precise in the cases of large number of instances	https://www.researchgate.net/publication/259235118_Random_Forests_and_Decision_Trees
11.	Software defect prediction using neural networks	Proceedings of 3rd International Conference on Reliability, Infocom Technologies and Optimization	we have developed a model based on text mining techniques that will be used to assign the severity level to each defect report based on the classification of existing reports done using the machine learning method namely, Radial Basis Function of neural network	The values of sensitivity for medium, low and very low severity defects are in the range of 56% to 75%, in contrast to its values for high severity defects which is in the range of 70% to 100% for most of the runs	https://ieeexplore.ieee.org/document/7014673

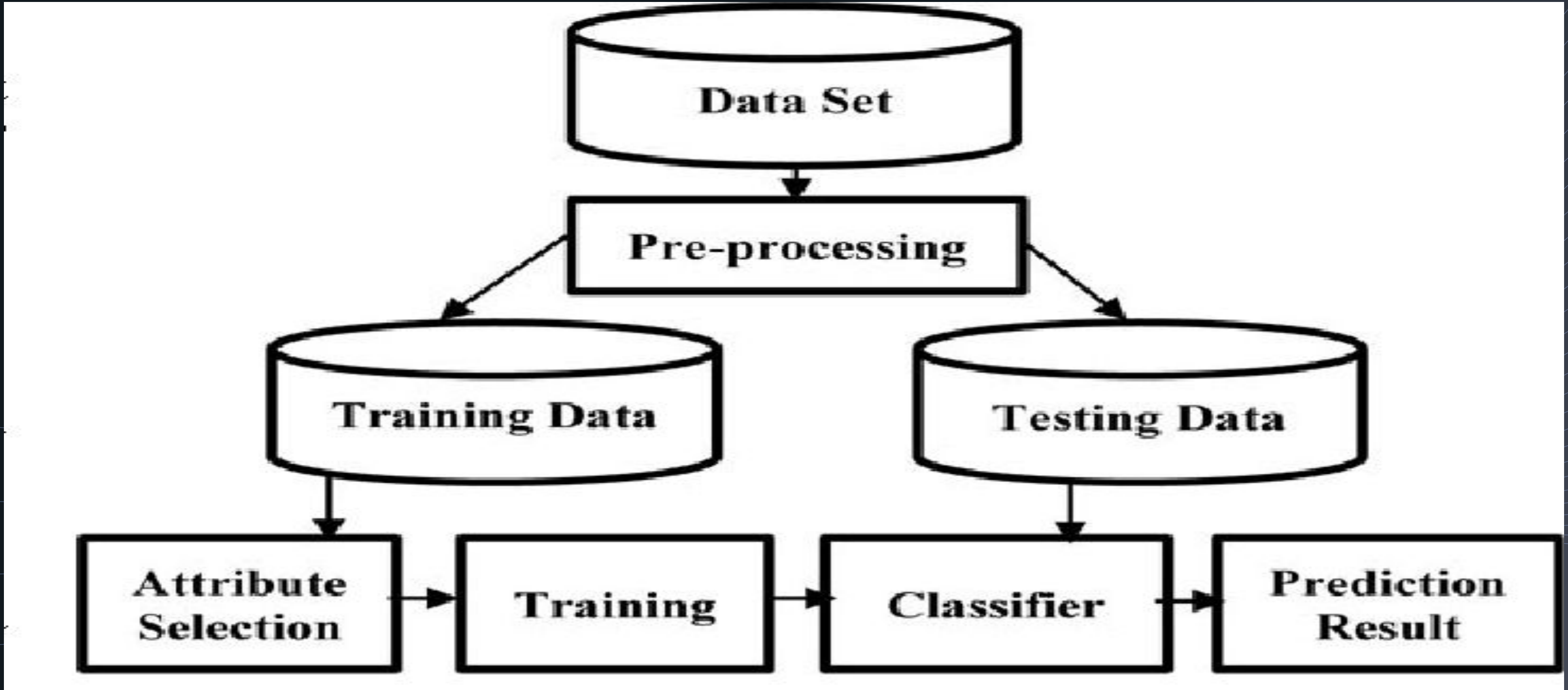
12.	A survey of dimensionality reduction techniques	in International Journal of Emerging Trends & Technology in Computer Science, Volume 3, Issue 6, November-December 2014.	A different family of algorithms poses the dimensionality reduction problem as one of projecting the original data onto a subspace with some interesting properties	It can be concluded that the Random Forest achieves increased classification performance and yields results that are accurate and precise in the cases of large number of instance	https://www.researchgate.net/publication/260755521_A_survey_of_dimensionality_reduction_techniques
13.	Dimensionality reduction and generalization	in the 24th International Conference on Machine Learning, Corvallis, OR, 2007.	Using probabilistic estimates for integral operators we can prove error estimates for KPCR and propose a parameter choice procedure allowing to prove consistency of the algorithm.	We show that perform- ing KPCA and then ordinary least squares on the projected data, a procedure known as kernel principal component regression (KPCR), is equivalent to spectral cut-o regularization.	https://www.researchgate.net/publication/221345784_Dimensionality_reduction_and_generalization
14.	A Survey Of Dimensionality Reduction And Classification Methods	In International Journal of Computer Science & Engineering Survey, Vol.3, No.3, June 2012	A different family of algorithms poses the dimensionality reduction problem as one of projecting the original data onto a subspace with some interesting properties	We have analyzed the number of citations that the most relevant papers in each section have received in the last decade (2003‑2012). In Table I we show the number of citations summarized by large areas as well as their share (%) for the different years	https://www.researchgate.net/publication/276197488_A_Survey_Of_Dimensionality_Reduction_And_Classification_Methods
15.	Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models	Rpi, Troy, Ny, Usa, 2011. Copyright 2011	We show some experiment results to compare the performance of kernel PCA and traditional PCA for pattern classification. We also implement the kernel PCA-based ASMs, and use it to construct human face models.	We found that Gaussian kernel PCA-based ASMs are promising in providing more deformation patterns than traditional ASMs.	https://www.researchgate.net/publication/229158312_Kernel_Principal_Component_Analysis_and_its_Applications_in_FaceRecognition_and_Active_Shape_Models
16.	The Meaning and Use of the Area Under a Receiver Operating Characteristic (ROC) Curve	Radiology, 143, 1982, pp. 29-36	A large number of theoretically bas.d measures has been proposed to reduce an entire ROC curve to a singlc quantitative inclex of diagnostic accuracy; all of these measures have been rooted in the assumption that the	To amplify the three-way equivaIence between the area under an ROC curv,	https://www.researchgate.net/publication/16134792_The_Meaning_and_Use_of_the_Area_Under_a_Receiver_Operating_Characteristic_ROC_Curve

17.	Learning multiple Layers of representation	In Department of Computer Science, University of Toronto,, Toronto,Trends in Cognitive Sciences 11(10), November 2007	we would like our approximate inference method to be as accurate as possible, and we might prefer a model that is slightly less likely to generate the data if it enables more accurate inference of hidden representations.	This experiment result is much more sensible first to learn a generative model that infers the hidden variables from the sensory data and then to learn the simpler mapping from the hidden variables to the labels.	https://www.researchgate.net/publication/260755521_A_survey_of_dimensionality_reduction_techniques
18.	Software defect prediction using software metrics - A survey	2013 International Conference on Information Communication and Embedded Systems (ICICES)	The objective of this research is to build an efficient Fuzzy inference system that can learn and predict bugs in software products. Here, we have applied SVM, a supervised training algorithm for classification of data into two sets, buggy and non-buggy.	On iris data set our model gives 100% recall (Probability of detection also denoted as PD) and 0% false alarm rate (PF) with 100% accuracy whereas on Pima Indians Diabetes data set it gives 68.6% and 29.5% PD and PF respectively which is better than previous known result (60% and 19%[23]) except in PF which should be low.	https://ieeexplore.ieee.org/document/6508369
19.	Software defect prediction using supervised learning algorithm and unsupervised learning algorithm	Confluence 2013: The Next Generation Information Technology Summit (4th) International Conference)	To predict software defect we analyzed classification and clustering techniques. The performance of three data mining classifier algorithms named J48, Random Forest, are evaluated based on various criteria like ROC, Precision, MAE, RAE etc.	It can be observed that out of three algorithms, Random Forest exhibits highest values of Accuracy, Recall, ROC and F-Measure in maximum number of datasets. Random Forest also gives minimum amount of Root Mean Square Error in all the cases. So we can say that there are minimum numbers of defects in Random Forest.	https://ieeexplore.ieee.org/document/6832328
20.	Improved Approach for Software Defect Prediction using Artificial Neural Networks	2016 5th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)	In this paper, neural system methodology is utilized to find whether quantitative and subjective variables can be utilized to decide the level or measure of number of faulty software module.	After this paper implementation we found that the ANN based approach is giving better results than fuzzy logic based approach	https://www.researchgate.net/publication/229158312_Kernel_Principal_Component_Analysis_and_its_Applications_in_FaceRecognition_and_Active_Shape_Models

Methodology

Decision Tree classifier is used to make the model learn from the test set and then the model is tested on the training set and the performance measures are calculated. However, having so many attributes and instances can lead the model to overfit. Hence, we first reduced the dimensionality of the data to a set of 6 cumulated features using 4 different techniques and then trained the model using Decision Tree classifier. A detailed comparison was then made based on the performance metrics that include Accuracy, F1-Scores and Area Under the Receiver Operating Characteristics (ROC). Following are the algorithms used for Dimensionality Reduction, along with a brief description.

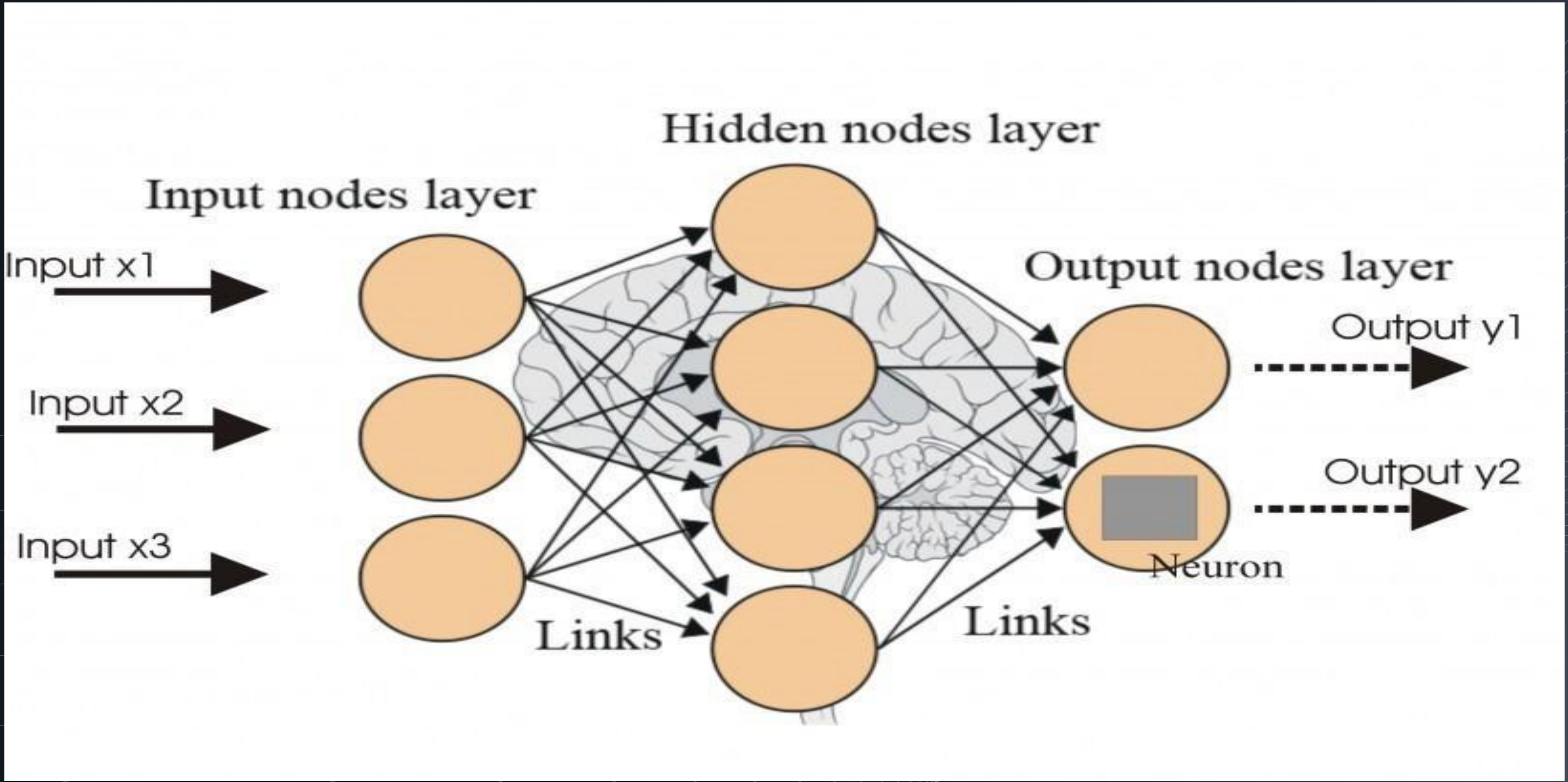
Overview of SDP Process



A. ANN

- This algorithm is somewhat based on the human brain or the human nervous system and uses a set of hidden layers with varied number of nodes called neurons.
- Each neuron takes inputs from either a few or all of the previous layer neurons and processes the input using initialised weights and an activation function.
- It then sends the output to many neurons of the next layer. Based on the output and the cost function, the weights are updated over a number of epochs until the parameters best fit the model.
- To train our model, we use 3 hidden layers with different number of neurons. The cumulated features extracted from the network are then used to train the model using Decision Tree Classifier

Artificial Neural Network

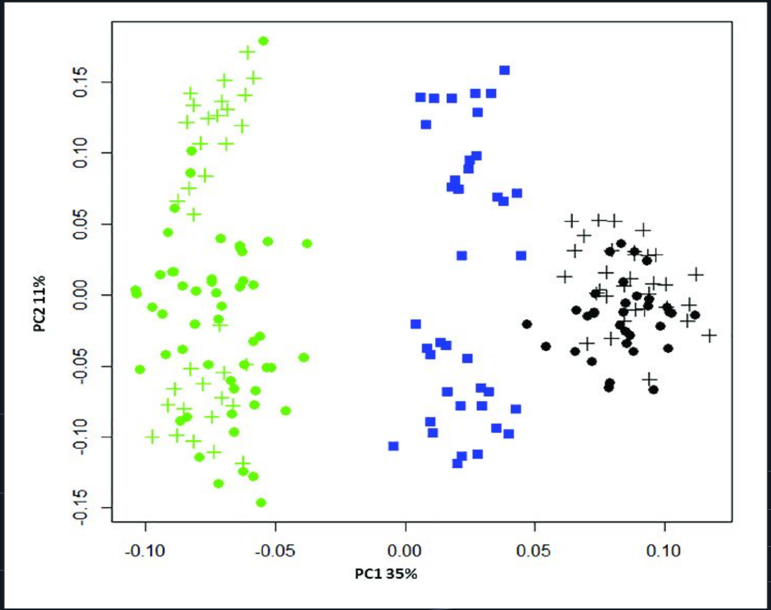


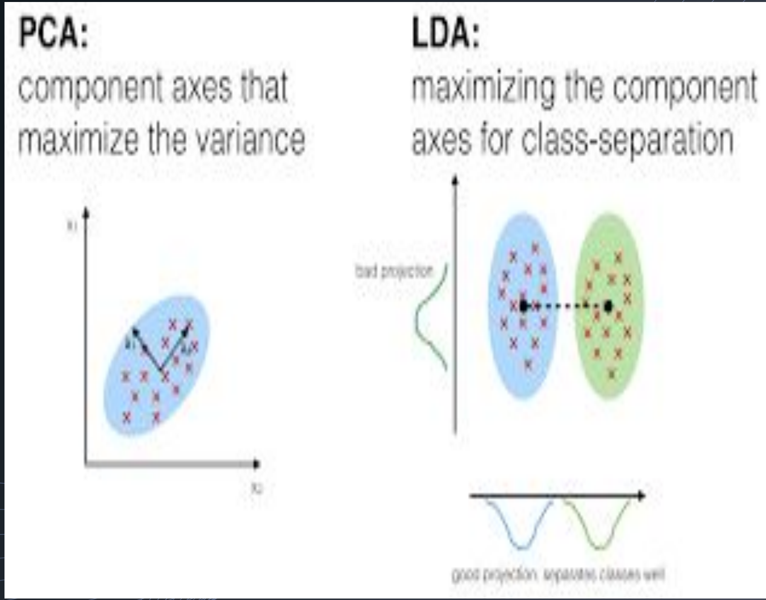
B. Principal Component Analysis (PCA)

- The PCA is a statistical data analysis method that transforms the initial set of variables into an assorted set of linear combinations, known as the principal components (PC), with specific properties with respect to variances.
- This condenses the dimensionality of the system while maintaining information on the variable connections .
- The PCA algorithm is applied such that it extracts 6 new independent features that explain most the variance of the dataset, regardless of the dependent variable.
- Since the final class of each instance is not considered while turning the data into a low dimensional one, hence it is an Unsupervised Model.

PCA

LDA


PC2 11%
0.15
0.10
0.05
0.00
-0.05
-0.10
-0.15
-0.10
-0.05
0.00
0.05
0.10
PC1 35%


PCA:
component axes that
maximize the variance
LDA:
maximizing the component
axes for class-separation
bad projection

C. Linear Discriminant Analysis (LDA)

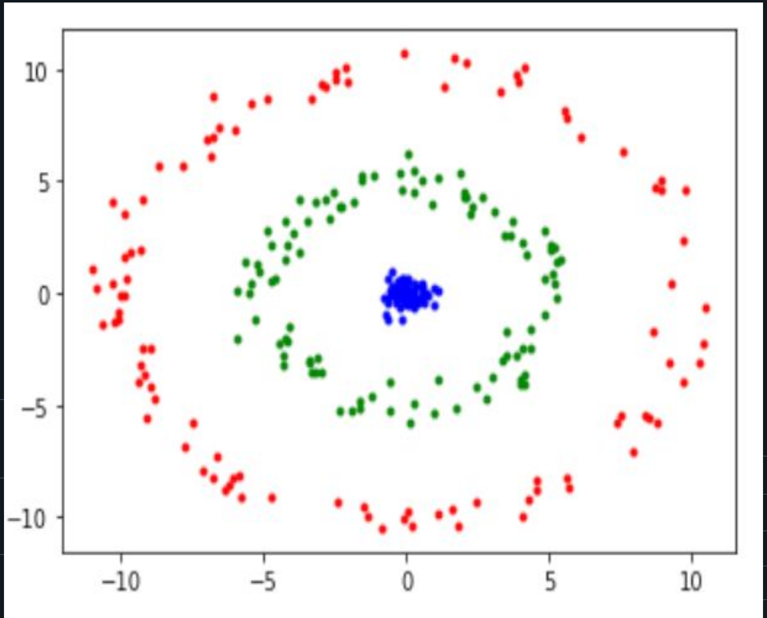
- In a high dimensional data, it is difficult to find similarities between different data points and hence the model is difficult to analyse.
- The LDA algorithm maps down the high dimensional data to a low dimensional space which is then fed to the classifier to train the model. LDA aims to maximize the between-class distance and minimize the within-class distance in the dimensionality reduced space .
- The LDA algorithm is applied such that it extracts 6 new independent features that separate most the classes of the dataset, that is the buggy and non buggy instances.
- Since the extracted features are obtained taking into consideration the dependent variable, hence it is a Supervised Model.

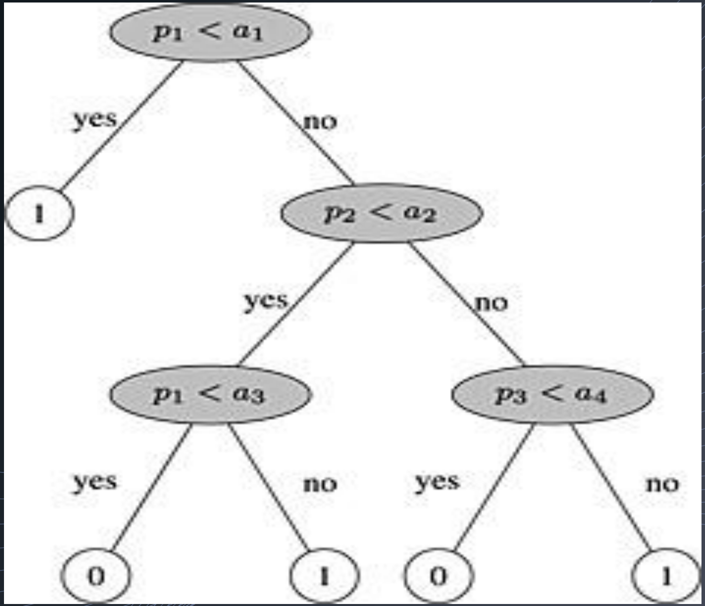
D. Kernel Principal Component Analysis (KPCA)

- Standard PCA only allows linear dimensionality reduction. However, if the data has more complicated structures which cannot be well represented in a linear subspace, standard PCA will not be very helpful. Kernel PCA thus extends conventional principal component analysis (PCA) to a high dimensional feature space using the kernel algorithm.
- An intuitive understanding of the Gaussian kernel PCA is that it makes use of the distances between different training data points, which is like k-nearest neighbor or clustering methods [19]. Gaussian kernel PCA reveals more complex hidden structures of the data than standard PCA.
- Gaussian RBF, Polynomial, Hyperbolic Tangent are some popular Kernel functions. We leveraged the Gaussian RBF kernel function to reduce the dimensionality of our data set.

KPCA

Decision Tree





Decision Tree Algorithm

- The Decision Tree algorithm, that is a supervised learning algorithm and works on principles of entropy and information gain, has been used as a classifier.
- Entropy of a dataset measures the impurity of the dataset i.e., how disordered the data set is.
- The most critical aspect of Decision Tree algorithm is the attribute selection method employed at each node of the tree, since there are some attributes that split the data more purely than other attributes. The algorithm works on principle of greediness i.e., it looks for the solution that appears to be best at the moment without looking at the picture at large. The Decision Tree algorithm uses the Information Gain, which calculates the reduction in entropy or gain in information, to split the data set using a particular attribute. The algorithm is advantageous as it requires less data cleaning and is not influenced by outliers and missing values to a fair extent.

PERFORMANCE MEASURES

	Predicted buggy	Predicted clean
True Buggy	TP	FN
True Clean	FP	TN

Table 2: Confusion Matrix

A. Accuracy

This refers to the ratio of correctly predicted instances of the test set to the total number of instances of the test set.

Accuracy = (TP + TN) / (TP + FN + FP + TN)

B. F1 scores

- At times, accuracy paradox can lead to misinterpretation of the results, hence we take another performance metrics called F1 score into consideration.
- F1 score is the harmonic mean of Precision and Recall, which are also calculated from the confusion matrix.

- Precision is the ratio of actual correctly predicted positive (buggy) instances to the total number of predicted positive instances (**Precision** = TP /(TP + FP).

Recall is also known as Sensitivity. Recall is the ratio of actual correctly predicted positive (buggy) instances to the total number of actual positive instances (**Recall** = TP /(TP + FN).)

Taking the harmonic mean,

we get **F1 score** = (2*Recall*Precision)/(Recall + Precision)

C. Area Under the ROC

- The performance of the predicted models was evaluated by plotting the Receiver Operating Characteristics (ROC) curve and evaluating the area under the curve.
- ROC curve, which is defined as a plot of sensitivity on the y-coordinate versus its 1-specificity (it is defined as the ratio of predicted non faulty classes to the number of classes actually non faulty) on the x coordinate, is an effective method of evaluating the quality or performance of predicted models

VALIDATION METHOD

- We have used hold out cross validation method to validate the data set.
- SInce all the data sets used had quite a large number of instances, the training set and test set were divided in the ratio 3:1.
- The training set was used to train the classifier and then the model was validated on the test set.

RESULTS

We got the confusion matrix by applying various techniques .

Accuracy and F1 score obtained are given below in the table :

Table 1 : Accuracy of each algorithm

DATA SET	ANN	PCA	LDA	KPCA
PC1	0.90	0.920	0.910	0.92
CM1	0.86	0.864	0.864	0.872
KC1	0.78	0.818	0.812	0.81
KC2	0.84	0.778	0.793	0.81
JM1	0.78	0.754	0.748	0.75

Table 2 : F1 score of each algorithm

DATA SET	ANN	PCA	LDA	KPCA
PC1	0.95	0.95	0.95	0.95
CM1	0.92	0.92	0.92	0.93
KC1	0.86	0.89	0.88	0.88
KC2	0.90	0.86	0.87	0.89
JM1	0.86	0.85	0.84	0.84

The performance of the predicted models is evaluated by plotting the ROC (Receiver Operating Characteristic) and evaluating the area under the curve .

Table 3 : ROC Curve Area

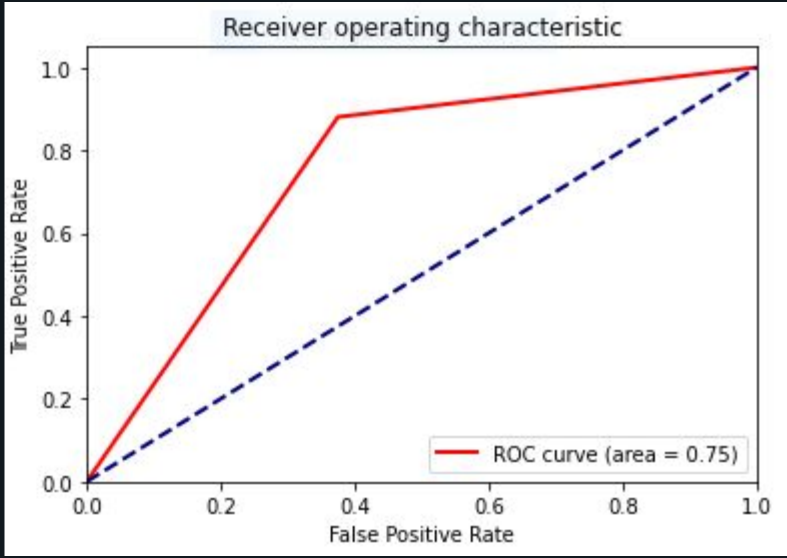
DATASET	ANN	PCA	LDA	KPCA
PC1	0.50	0.70	0.62	0.70
CM1	0.70	0.69	0.75	0.58
KC1	0.62	0.63	0.63	0.62
KC2	0.73	0.61	0.65	0.62
JM1	0.65	0.59	0.57	0.59

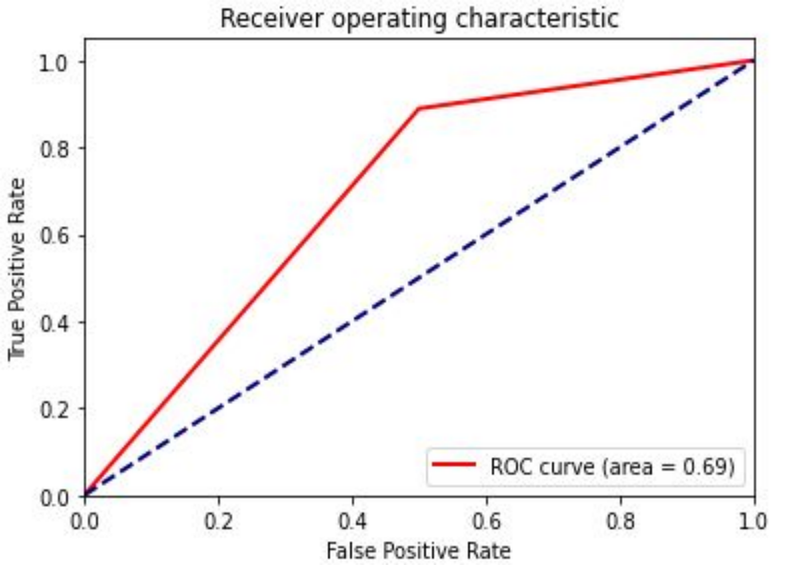
ROC for different Dataset of LDA and PCA Algorithm :

LDA

PCA

Dataset 1 : cm1

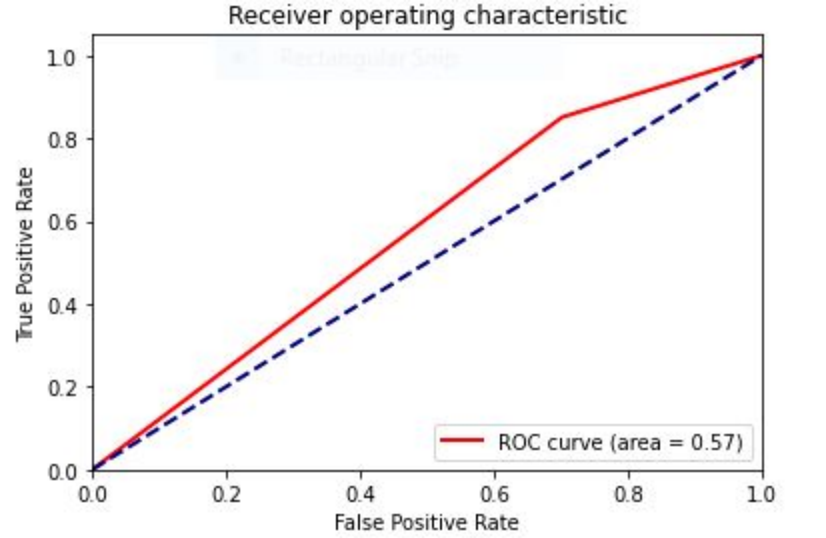


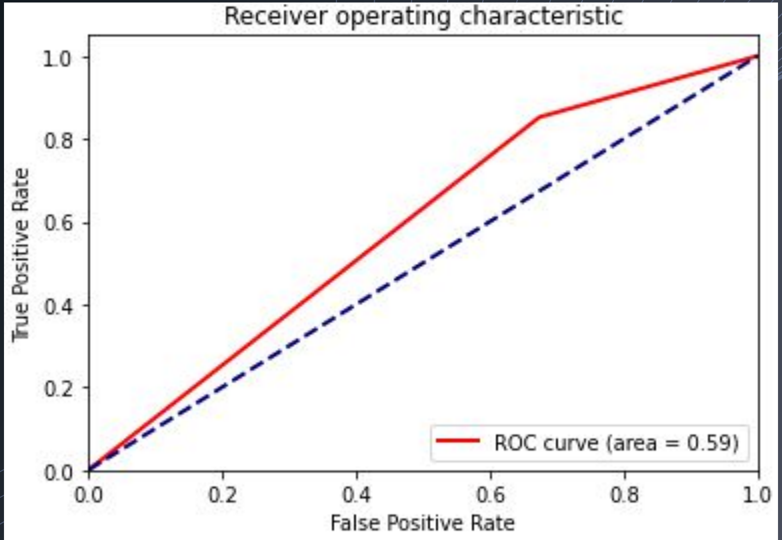


LDA

PCA

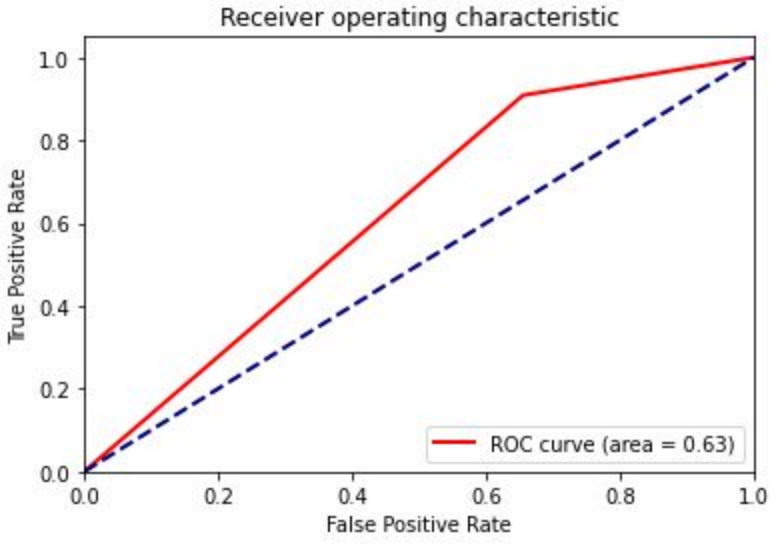
Dataset 2 : jm1



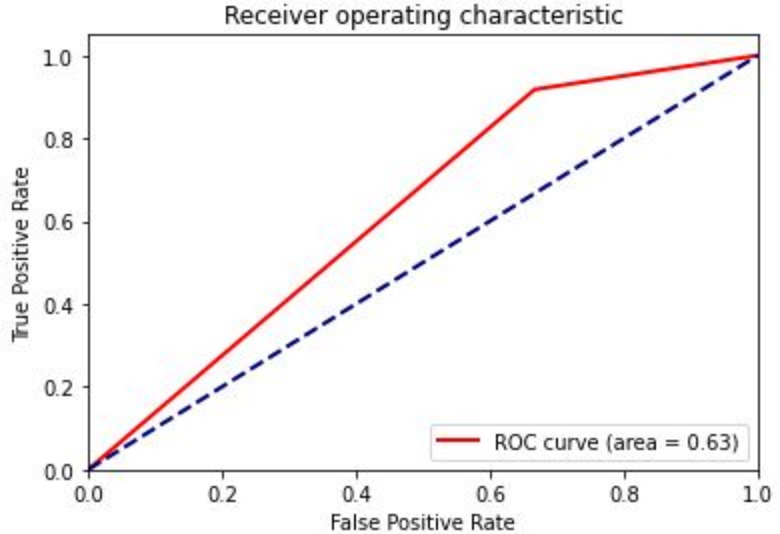


LDA

Dataset 3 : KC1



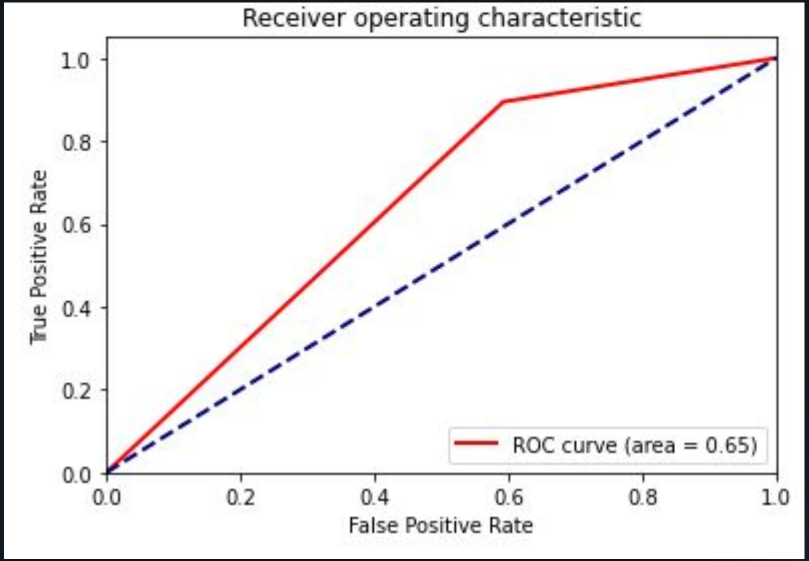
PCA

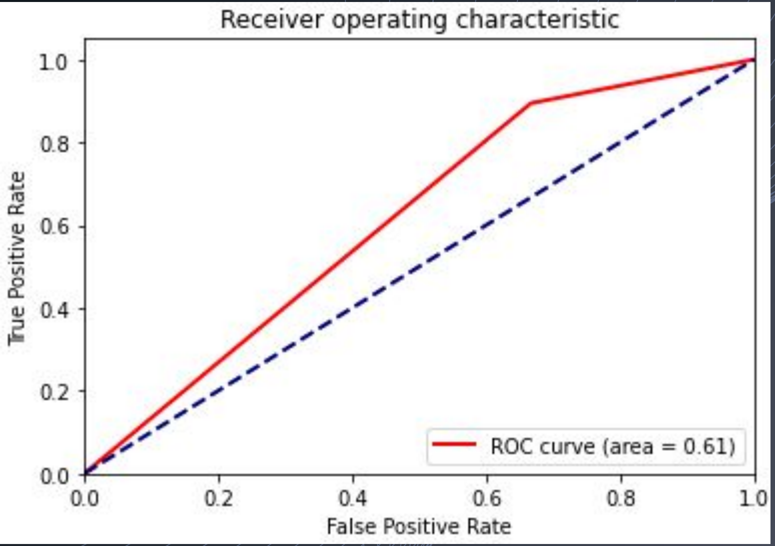


LDA

PCA

Dataset 4 : kc2

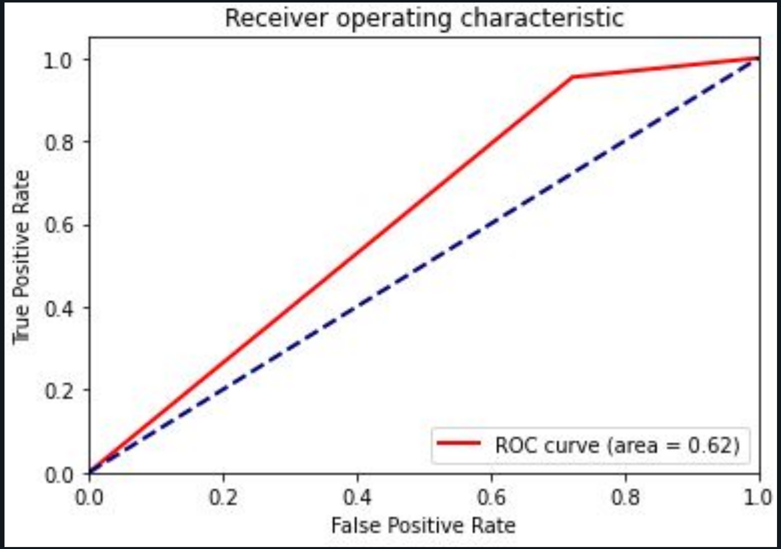


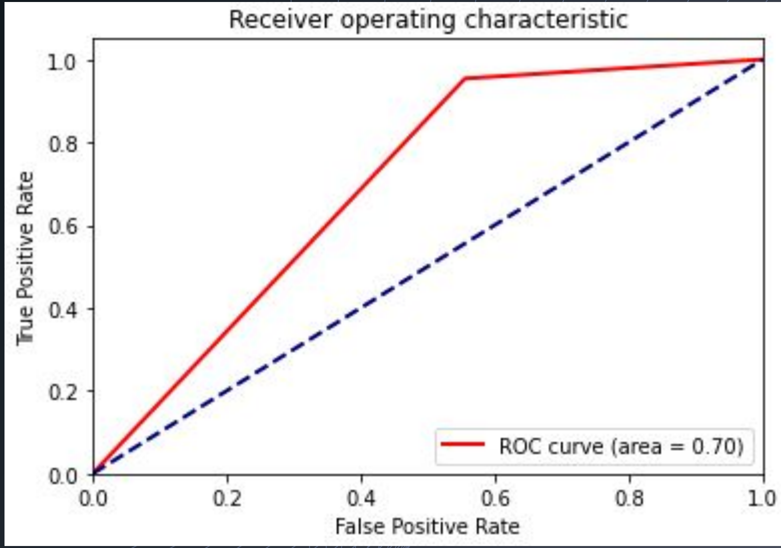


LDA

PCA

Dataset 5 : PC1



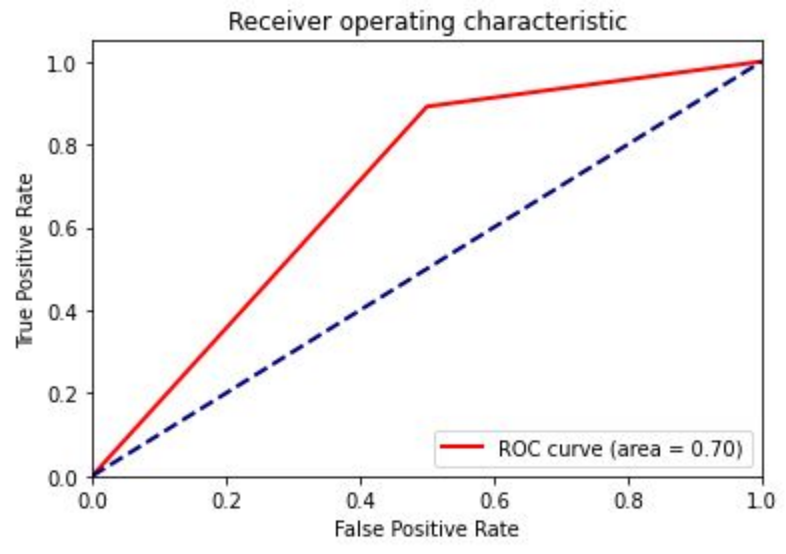


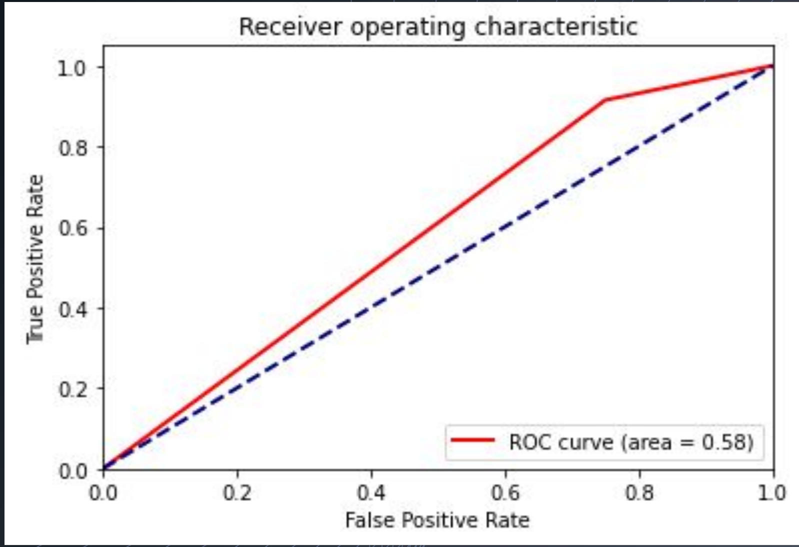
ROC for different Datasets of ANN and KPCA Algorithm :

ANN

KPCA

Dataset 1 : cm1

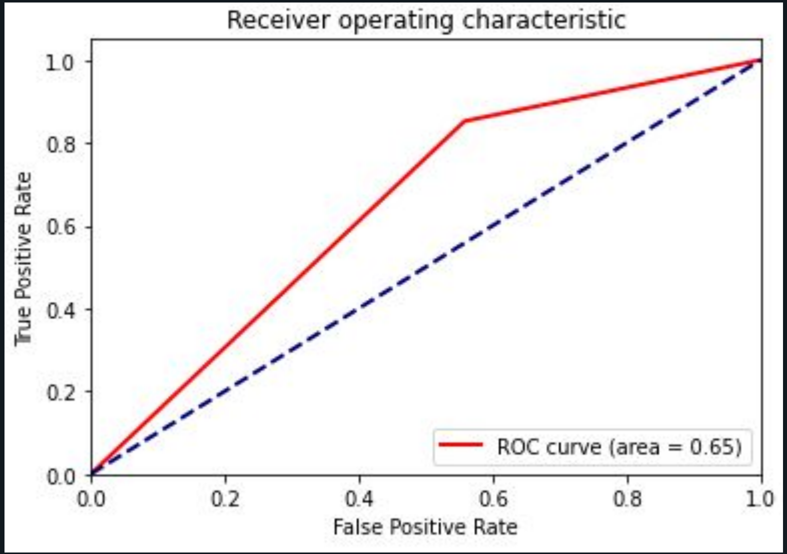


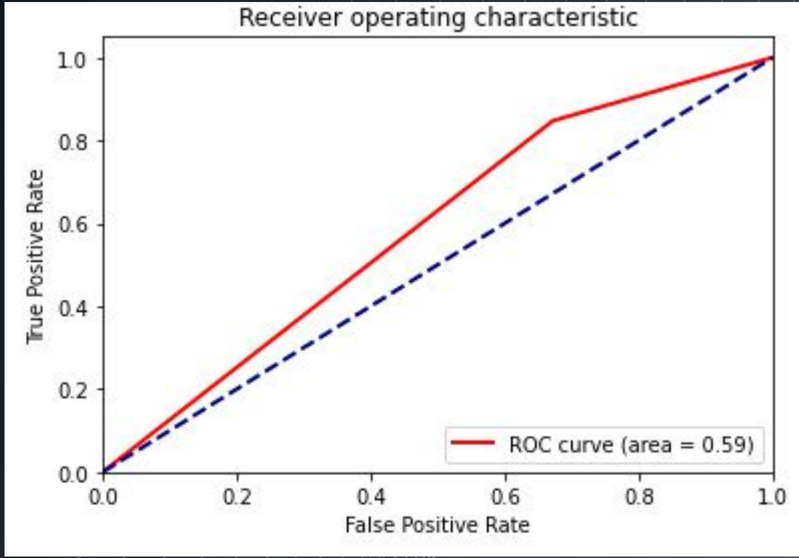


ANN

KPCA

Dataset 2 : jm1

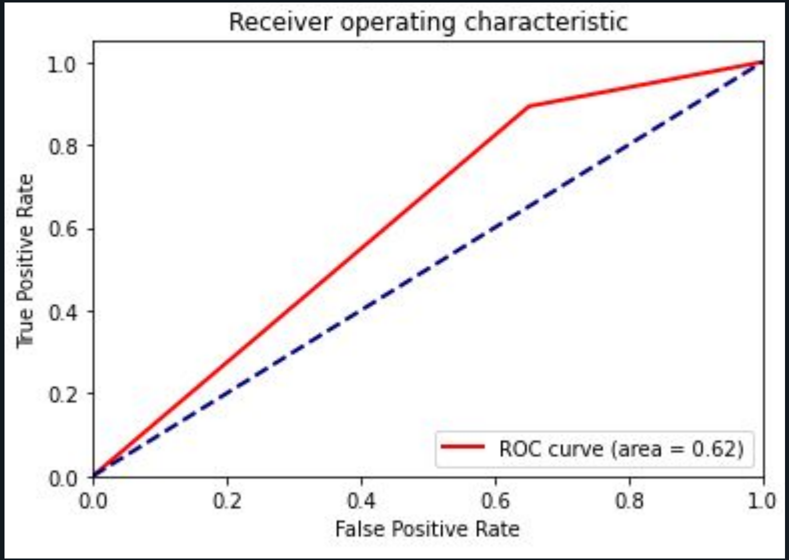


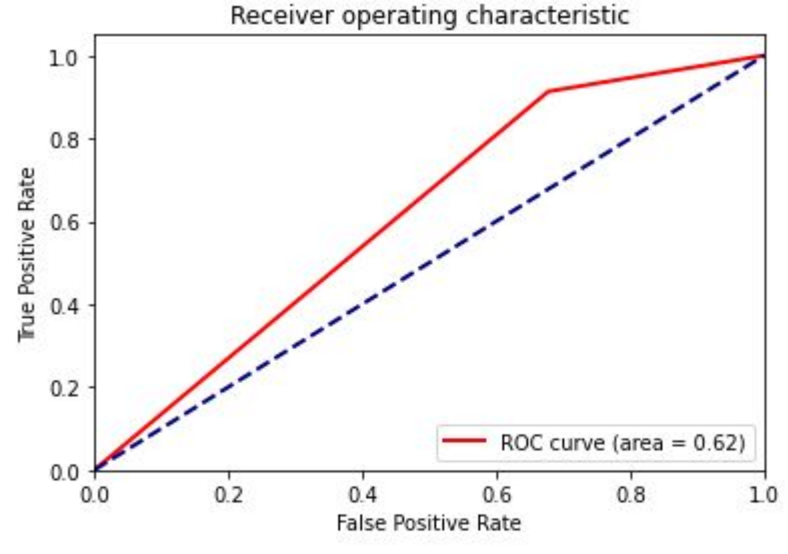


ANN

KPCA

Dataset 3 : kc1

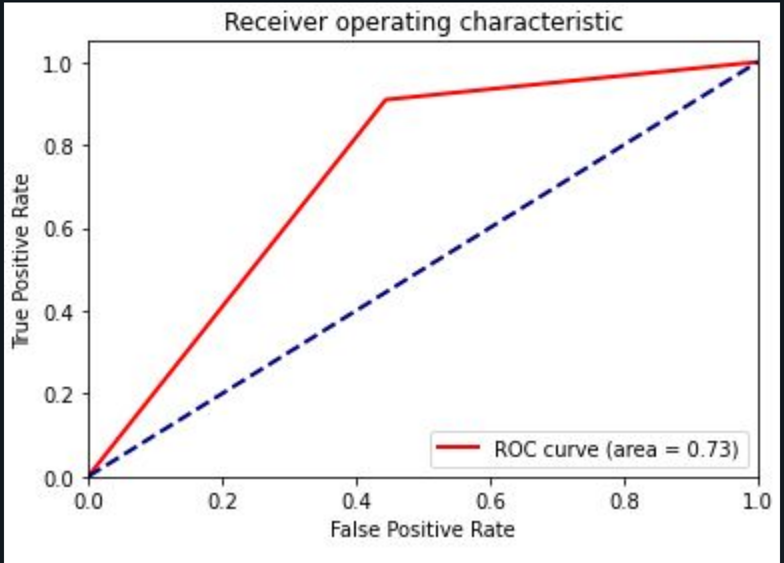


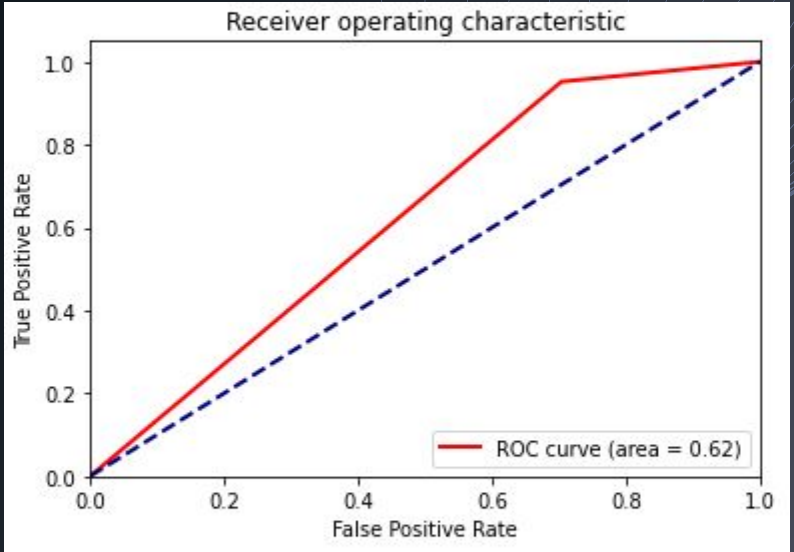


ANN

KPCA

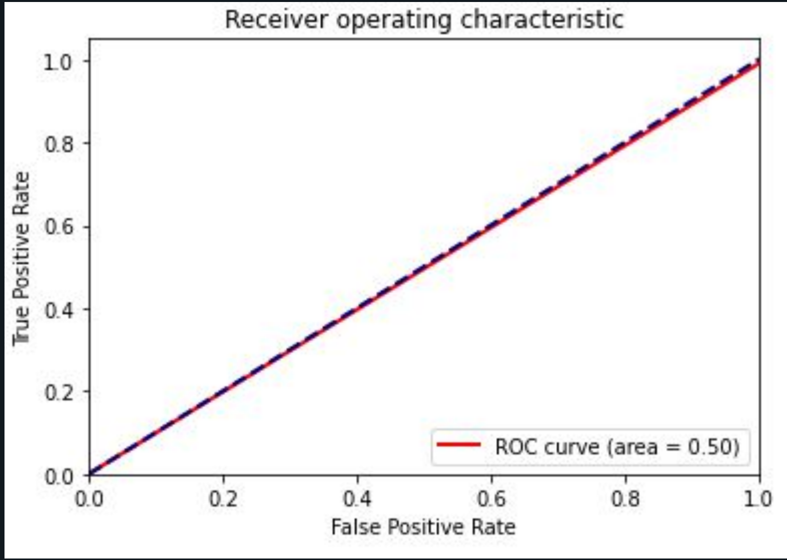
Dataset 4 : kc2



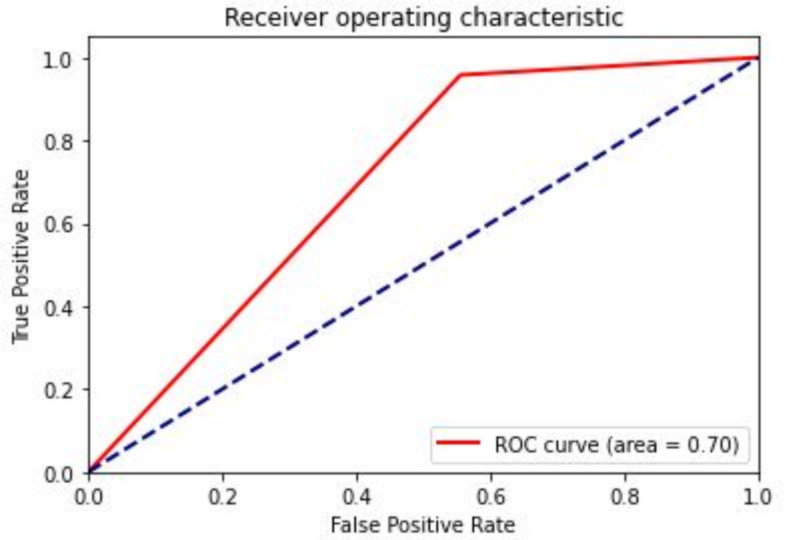


ANN

Dataset 5 : pc1



KPCA



THREATS OF VALIDITY

- Construct Validity: We estimate the performance of the model using the hold-out cross-validation method. Training and test sets are constructed randomly so they may overfit the data.
- Internal validity: The data set used contains information regarding features determined by McCabe and Halstead feature extractors, which are known to have certain limitations.
- External Validity: We used only 5 open-source data sets taken NASA Promise Repository and so our results may not generalize to all software. Replication of this a comparative study taking into account other datasets may produce more generalized results.
- Conclusion Validity:The datasets being used have a class imbalance problem. So, we used AUC to evaluate the performance of our model but it can still be partial for non-buggy instances

CONCLUSION

- In this paper, we proposed various size reduction strategies and compared the results obtained on the basis of the accuracy of the forecast, the F1 points and the area below the curve.
- The best model depends on the data set and the performance metrics.
- Artificial Neural Networks (ANN) and KPCA performed well in comparison to other models in terms of accuracy .

FUTURE SCOPE

- In future,we would like to improve the neural network model .
- We can improve it by changing its various parameters like number of hidden layers , neurons in each layer , optimizers and the cost function.
- We will also try other models like Naive Bayes , Kernel Support Vector Machine, Random Forest and compare its results with these models.

Appendix I:

S.No.	Metric	Description
1.	Process Metrics	It is used for improve software development and maintenance.
2.	Line of code(LOC)	It is a software metric used to measure the size of a computer program by counting the number of lines in the text of the program's source code.
3.	Number of changes	Number of builds in which a specific component has changed
4.	Number of Instances	Number of features which are extracted from the software archive
5.	Number of principal component	Number of those variable which are constructed as linear combinations or explain maximal amount of variance that is to saythe lines that capture most information of the data.
6.	Number of bugs	Number of defects or we can say defect prone-modules in software system

Appendix II:

S.No.	ALGORITHM	Description
1. .	ANN (Artificial Neural Network)	It is a computational and mathematical model that is inspired by the biological nervous system. It uses the processing of the brain as a basis to develop algorithms that can be used to model complex patterns and prediction problems.
2.	PCA (Principal Component Analysis)	PCA is a mathematical data analysis method that converts the first set of variables into a set of specific combinations, known as key components (PCs)with specific characteristic regarding variability. This maintains the size of the system while storing information on a flexible connection
3.	LDA (Linear Discriminant Analysis)	The LDA algorithm is used in such a way that it produces 6 new independent features that greatly differentiate the data classes, which are buggy and non-buggy.
4.	KPCA (Kernel Principal Component Analysis)	It is a non-linear dimensionality reduction technique and also extension of PCA Algorithm - which is a linear dimensionality reduction technique -using kernel methods.

REFERENCES

1. X. Yang, D. Lo, X. Xia, Y. Zhang, and J. Sun, "Deep learning for just-in-time defect prediction," in QRS'15: Proc. of the International Conference on Software Quality, Reliability and Security, 2015. [Deep learning for just-in-time defect prediction]

2. Y. Kamei, E. Shihab, B. Adams, A. E. Hassan, A. Mockus, A. Sinha, and N. Ubayashi, "A large-scale empirical study of just-in-time quality assurance," TSE, vol. 39, no. 6, pp. 757–773, 2013. [A large-scale empirical study of just-in-time quality assurance]

3. T. Jiang, L. Tan, and S. Kim, "Personalized defect prediction," in ASE, 2013, pp. 279–28. 4. M. Tan, L. Tan, S. Dara, and C. Mayeux, "Online defect prediction for imbalanced data," in ICSE'15, pages 99–108. [Personalized defect prediction]

4. Praman Deep Singh, Anuradha Chug, " Software defect prediction analysis using machine learning algorithms" in 2017 7th International Conference on Cloud Computing, Data Science & Engineering . Confluence. [Software defect prediction analysis using machine learning algorithms]

5. S. Wang, T. Liu, and L. Tan, "Automatically learning semantic features for defect prediction," in ICSE'16: Proc. of the International Conference on Software Engineering, 2016. Automatically Learning Semantic Features for Defect Prediction

6. Y. Kamei, S. Matsumoto, A. Monden, K.-i. Matsumoto, B. Adams, and A. E. Hassan, "Revisiting common bug prediction findings using effort-aware models," in ICSM, 2010, pp. 1–10. Revisiting common bug prediction findings using effort-aware models

7. Jian Li, Pinjia He, Jieming Zhu, and Michael R. Lyu, "Software Defect Prediction via Convolutional Neural Network". Software Defect Prediction via Convolutional Neural Network

8. P. D. Singh and A. Chugh, "Software Defect Prediction Analysis Using Machine Learning Algorithms," in International Conference on Cloud Computing, Data Science & Engineering – Confluence, 2017. Software Defect Prediction Analysis Using Machine Learning Algorithms

9. G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science, vol. 313, no. 5786, pp. 504–507, 2006. Reducing the dimensionality of data with neural networks

10. G. E. Hinton, “Learning multiple layers of representation,” Trends in cognitive sciences, vol. 11, no. 10, pp. 428–434, 2007. Learning multiple layers of representation

11. L. Deng, J. Li, J.-T. Huang, K. Yao, D. Yu, F. Seide, M. Seltzer, G. Zweig, X. He, J. Williams et al., “Recent advances in deep learning for speech research at microsoft,” in Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013, pp. 8604–8608. Recent advances in deep learning for speech research at microsoft,

12. F. Rahman and P. Devanbu, “How, and why, process metrics are better,” in ICSE, 2013, pp. 432–441. How, and why, process metrics are better

13. J. Ali, R. Khan, N. Ahmad, I. Maqsood, “Random Forests and Decision Trees,” in International Journal of Computer Science Issues, Vol. 9, Issue 5, No 3, September 2012 . Random Forests and Decision Trees,

14. R. Jindal, R. Malhotra, A. Jain, “Software Defect Prediction Using Neural Networks,” IEEE Conference 2014. Software Defect Prediction Using Neural Networks

15. V. A. Kumar, N. Elavarasan, “ A Survey on Dimensionality Reduction Technique,” in International Journal of Emerging Trends & Technology in Computer Science, Volume 3, Issue 6, November-December 2014. A Survey on Dimensionality Reduction Technique

16. S. Mosci, L. Rosasco, A. Verri, “Dimensionality Reduction and Generalization,” in the 24th International Conference on Machine Learning, Corvallis, OR, 2007. Dimensionality Reduction and Generalization

17. N. Varghese, V. Verghese, Gayathri. P and Dr. N. Jaisankar, “A Survey Of Dimensionality Reduction And Classification Methods,” in International Journal of Computer Science & Engineering Survey, Vol.3, No.3, June 2012. A Survey Of Dimensionality Reduction And Classification Methods

18. Q. Wang, “Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models,” Rpi, Troy, Ny, Usa, 2011. Copyright 2011. Kernel Principal Component Analysis and its Applications in Face Recognition and Active Shape Models

19. J. Hanley, BJ. McNeil, “The meaning and use of the area under a Receiver Operating Characteristic ROC curve”, Radiology, 143, 1982, pp. 29-36. The meaning and use of the area under a Receiver Operating Characteristic ROC curve


Thank
You