# MTP-1

*by* Aditya --

**Submission date:** 18-Oct-2021 10:24AM (UTC+0530)
**Submission ID:** 1676810913
**File name:** 2017IMT_005_MTP1_1.pdf (319.5K)
**Word count:** 3148
**Character count:** 17368

# ABSTRACT

Accurately predicting bugs and defects in software units helps the developers and testers to find the defective unit and save their efforts in other software developing aspects. Previous studies used the concept of machine-learning to build models to detect defective units in software. I revisited the previous studies and pointed out the areas of potential improvements. The conventional classifiers give poor results as they perform well in only some part of data, so there is a need of an advance classifier which is not very complex in implementation and give a decent result. This study will provide a way to use a popular concept of Mixture of Experts in this domain and will provide comparison of various variations.

*Keywords:* Software defect prediction, Machine Learning, Mixture of experts.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

## ABBREVIATIONS

| | |
|---|---|
| ANN | Artificial Neural Network |
| AUC | Area under Curve |
| ME | Mixture of Experts |
| MELE | Mixture of explicitly localised experts |
| MILE | Mixture of implicitly localised experts |
| ML | Machine Learning |
| ROC | Receiving Operating Characteristics |
| SMOTE | Synthetic Minority Oversampling Technique |
| SVM | Support Vector Machine |

# CHAPTER 1

# Introduction

This chapter will provide the information related to the background of the problem and possible solution for the same. It also describe the motivation and objective of this work and lastly the overall research workflow is provided to the readers.

## 1.1 Context

Prediction of faulty code in software is an important aspect of the software development lifecycle. The quality assurance team of any organization is usually limited to software maintenance, so anticipating faulty units precisely permits coders and developers to focus on their activities in the product development cycle. In inadequate programming, a defective unit may result from different variables that are hard to recognize by human efforts, like in a code review. Hence it is very necessary to predict these bugs accurately because of which this area of defect prediction in software units is gaining lot of attention these days. Mixture of Experts or ME which is a very effective way of tackling such problems will be studied and implemented in this project.

### 1.1.1 Software Defect Prediction

The use of machine learning is very popular in the software engineering domain. It can be used here to predict bugs and defects in software units by implementing a software defect prediction model. The idea of software defect prediction is very straightforward (fig 1.1), there are many object-oriented as well as other parameters like Line of code, Depth of inheritance, Cyclomatic Complexity, etc which can be treated as features for detecting defects in software. A machine learning-based classifier can be trained on any publicly available dataset and can classify a piece of software as buggy or bug-free.

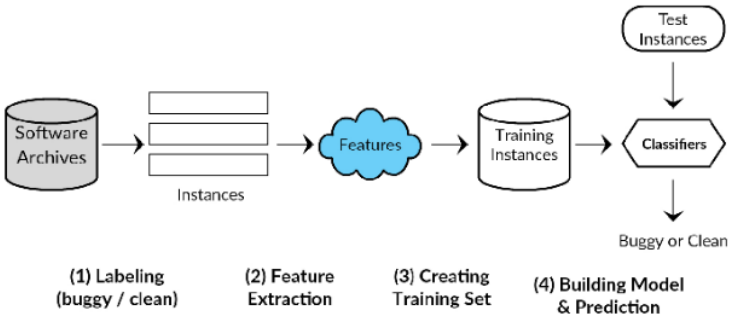



Figure 1.1: Software Defect Prediction

### 1.1.2 Mixture of Experts or ME

Since most of the units in any software are faultless, the problem at hand becomes a class imbalance problem. Normal classifiers such as support vector machine(SVM), neural networks, and fuzzy-system are very popular for performing classification tasks; however, both specific machine learning applications and scientific studies verify that these classification methods achieve significant results only for a very specific set of problem data and it is very difficult to get a method which achieves better results for the overall problem domain. As a consequence, there is a requirement for a method like ME, which combines various local classifiers and exploits their local behavior (fig 1.2), thus improving the reliability as well as the accuracy of the overall classification problem domain.

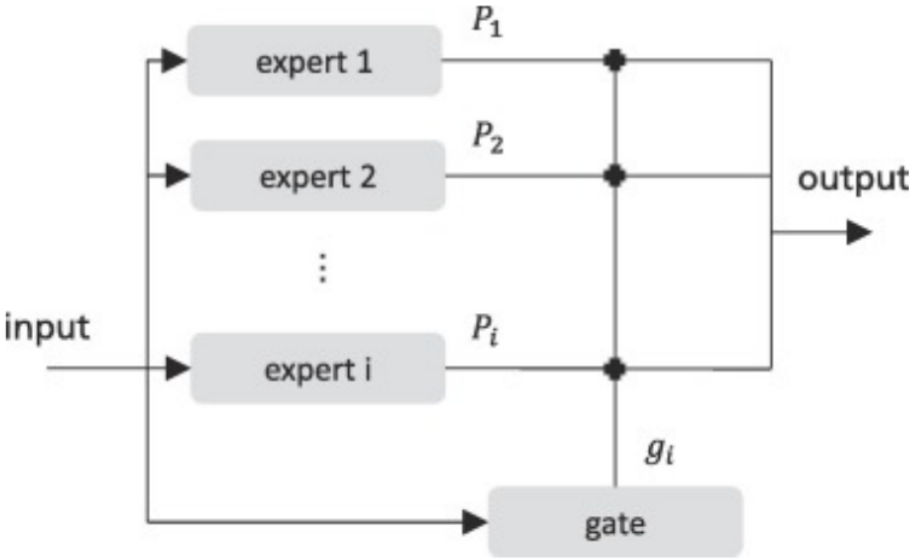



Figure 1.2: Mixture of Experts

## 1.2 Problem and Motivation

The Mixture of Experts or ME has a great potential to solve the problem in hand of defect prediction as it works on divide and conquer principle in which problem space is divided between various experts which are supervised by a gating logic, thus making the best use of training data. The existing papers and works mainly revolve around exploring the concepts of ME and have provided various advantages and disadvantages of various permutations on one another depending on the problem in hand but no evidence of applying ME on software defect domain is available. Also since the problem of software defect prediction is of a class imbalance nature so many works which used traditional classifiers or other famous Ensemble methods have a potential to give better results. The accuracy prediction of software defect prediction techniques is considerably low, ranging between 70% and 80%, with very high misclassification rate. An important issue related to software defect prediction is the lack of appropriate evaluation measures that can justify the capability of defect prediction models. Another major concern is about the unequal distribution of bugs in the software defect datasets that may lead to the biased learning. Not only this, the results of ME also depends on various factors like, determining the number of clusters in to which the global data is to be partitioned, the clustering method to be adopted, the nature of the gating function applied and the criteria to select the experts. Therefore motivation here is to study and apply the most appropriate and suitable concept of ME on software defect prediction data by training various local experts on different data clusters and then combining the results of various local experts, thus enhancing the performance of the classification.

## 1.3 Objectives

The project aims to fulfill the following objectives:

- Study and compare the performance of conventional ML models for software defect prediction
- Apply the concept of Mixture of Experts(ME) for predicting the software units as faulty or faultless
- Study and compare the performance of different permutations of ME on the dataset provided

## 1.4 Workflow

The entire workflow of the research is shown below (fig1.3). First of all problem understanding and research work related to the problem will be carried out. Secondly, all the limitations and areas of improvements in the existing papers will be properly studied. Third, some basic classifier will be applied to get an understanding about the nature of the data in hand. Fourth will be to select important features and to do more data preprocessing. After this Mixture of Experts concept will be implemented and its various variations and their effects on the datasets will be studied and lastly results will be tabulated and visualized.

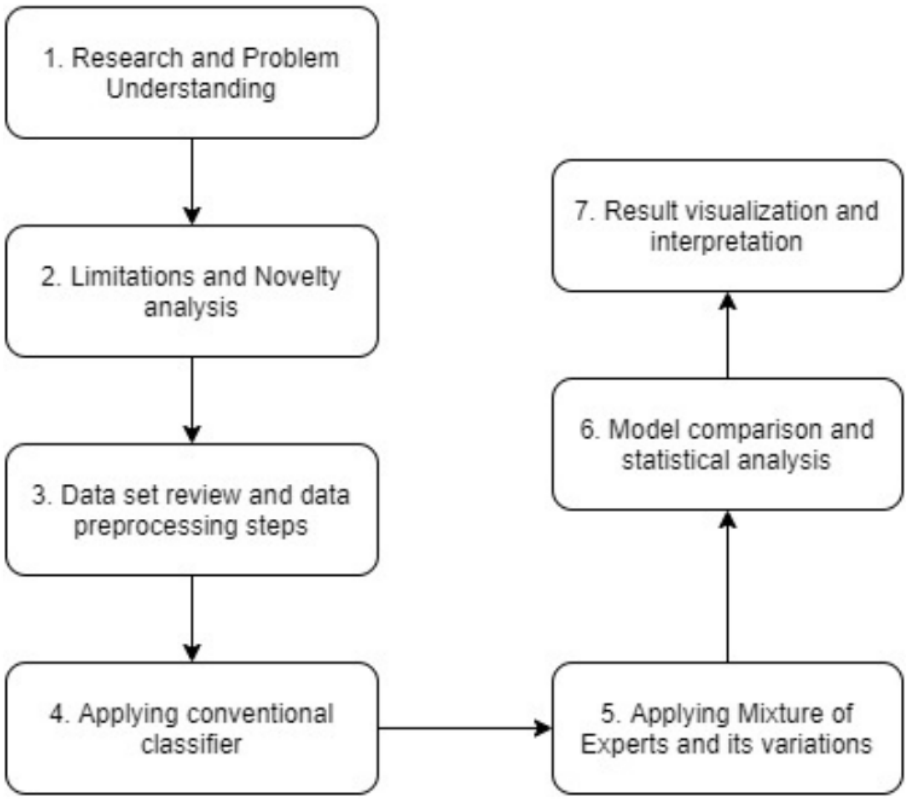



Figure 1.3: Research Workflow

# CHAPTER 2

# Literature review

## 2.1 Key related research

Following are some research works that give insights into the current state-of-the-art technology and its challenges:

- **Jaechang Nam [5]** This survey provided insights about the common defect prediction process and its evaluation. Secondly, various existing defect prediction models were compared based on various metrics and algorithms as well as identified major challenges of the software defect prediction that exist. However no argument related to poor performance of conventional classifiers was provided.

- **Libo, Lessmann and Baesens [3]** The research proposed a revised benchmarking configuration for evaluating software defect prediction performance. The configuration added many new dimensions, of class distribution sampling, evaluation metrics, and testing procedures in the existing system. They compared 17 different classifiers on basic of their AUC scores and founded that random forest model and neural network model achieved better results. However proper tuning and balancing of the data was missing in their study which provide the scope of better results.

- **Masoudnia and Ebrahimpour [4]** In this paper a detailed description of the Mixture of expert is provided and a comparision is made between various ME implementations (MILE versus MELE), describing the advantages and disadvantages of one another as well as the features of the ME was compared with other methods like boosting and negative correlation learning and results of the comparison was presented in a satisfactory manner. However nothing was mentioned about the problem domains which can be tackled by various permutations of ME.

- **Ferrari and Milioni [1]** This paper provided information to apply the concept of mixture-of-experts modeling technique to the solve a regression problem like the Boston housing data problem. Apart from this, it also provided information on some important decisions that one has to take related to aspects such as the clustering method and the gating functions when building a mixture-of-experts model. Depending on how these decisions various mixtures can be formed yielding different results. No discussion on how to use Mixture of Expert concept to solve classification problem like defect prediction was given.

- **Kamei and Shihab [2]** introduced a review that gives a short outline of software defect prediction and its parts. The review featured achievements made in software defect prediction and talked about latest things in this domain. Also, a few challenges for software defect prediction have been recognized and examined. Benefits and disadvantages of existing works in this domain have not been given.

## 2.2 Research gaps

Previous works in the field of software defect prediction revolve mainly around using conventional ML models. Very little research has been done to apply the concept of an advanced method such as ME in this domain. Major limitations of the previous works are highlighted below :

- The problem of Software defect prediction is a class imbalance problem and due to this most of the conventional classifiers give poor results

- Previous works on Mixture of Experts showed how effective it can be in tackling the class imbalance problem but no research was done in the software defect domain

## 2.3 Problem Formulation

First problem that is to be encountered in this project is about solving a class imbalance problem in general. Secondly, the different methods that are available to tackle such problems. Third, which conventional classifier is better for the selected dataset and how will the classification will be affected if the classifier properties are changed. Fourth challenge is to apply the concept of Mixture of experts and to decide the local experts, clustering algorithm and gating logic to be used. Lastly which variation of ME will be best suited for the selected dataset and how well it will perform.

# CHAPTER 3

# Methodology

## 3.1 Overall approach

The overall approach is very straight-forward (fig 3.1), first of all the fault prediction dataset will be preprocessed by unwanted information from the dataset will be removed by application suitable data preprocessing techniques. Then, the preprocessed data will be partitioned into train and test data, after that in order to obtain accurate results, data balancing has to be applied to the train data. Finally various local experts will be trained on various clusters obtained after applying a clustering algorithm and their results will be aggregated by the help of a gating logic. All the results and comparisons will be presented at last.

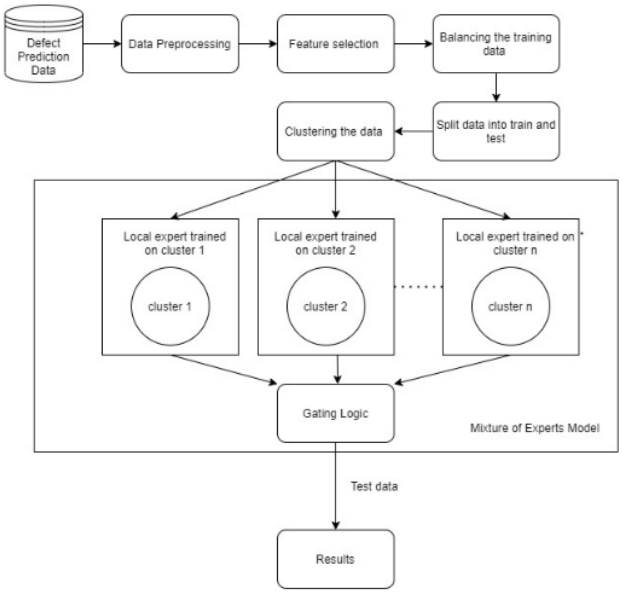



Figure 3.1: Overall Approach

## 3.2 Tools and Software used

Anaconda environment along with python was used, some important python libraries and packages used is listed below-

- **Pandas**: In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license.
- **NumPy**: NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays.
- **MatplotLib**: Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.It has a collection of functions that make matplotlib work like MATLAB.

## 3.3 Dataset Description

Dataset used will be a publicly available dataset provided by AEEEM-JIRA-PROMISE, which includes 22 software defect datasets in ARFF format. It is one of the biggest exploration information stores in computer programming. It is an assortment of various datasets, including the NASA dataset which was utilized by various investigations before. Other than numerous different classes like code investigation, testing, software maintenance, it also contains a category for defects.

## 3.4 Data Preprocessing

All the datasets were firstly analysed and preprocessed. They were cleaned and outliers were removed. Only the appropriate features were selected to avoid overfitting and train data was balanced by using Smote technique.

### 3.4.1 Data cleaning

All the missing values were checked and filled with the mean of that particular column and outliers were removed. All the byte objects fields were converted to string and later converted into binary classes of 0 and 1 by using **str.decode** function

### 3.4.2 Feature Selection using Backward Elimination

To avoid overfitting of data all the unwanted featured were removed and 10 most suitable features were selected by using Backward Elimination method, which includes beginning with all competitor features, testing the importance of every feature utilizing a picked model fit standard, erasing the variable whose loss gives the most genuinely inconsequential crumbling of the model fit, and rehashing this interaction until no further factors can be erased without a measurably unimportant loss of fit.

### 3.4.3 Data balancing using SMOTE

In order to obtain accurate model, training data was balanced by using Synthetic Minority Oversampling Technique or SMOTE technique, in which new samples were synthesised using examples of minority class.
SMOTE selects examples that are close in the feature space, drawing a line between the selected examples in the feature space and creating a new sample at any point along that line. Specifically, any random candidate from the minority class is chosen first. Then k of its nearest neighbors is calculated (typically k=5). Then from the selected neighbor, a random neighbour is selected and a synthetic example is obtained by selecting a point randomly between two points in feature space.

## 3.5 Mixture of expert

After balancing the training data, a suitable local experts were selected from Decision trees, SVM or Artificial neural network. Secondly a clustering algorithm was selected from hierarchical methods, quick partitions, mixture models, sum-of squares methods or Kmeans. Lastly a suitable gating logic was selected from Inverse Distance, Softmax, or Gaussian function. Based on these three selected parameters, many permutation of ME was implemented like MILE & MELE and they were compared based on suitable metrics and results were tabulated and visualized.

## 3.6 Efficacy Measurement

The important metrics for evaluating and comparing various classifiers will be -

- Confusion Matrix
- Accuracy, Recall, Precision
- F1 Score
- Receiving Operating Characteristics(ROC) and Area under the Curve(AUC)

# CHAPTER 4

# Progress made so far

The important features of software defect dataset was successfully preprocessed, balanced and important features selected by using backward elimination. Firstly a normal decision tree was applied on the training data and all the metrics were obtained after that data was implicitly partitioned and different local decision trees were trained of individual clusters and to gate the results of each local decision trees a voting classifier was used, which gave a significant better results then the first model (table 4.1).

Table 4.1: Result Comparison

| Model type | Accuracy score | F1 score |
|---|---|---|
| Conventional Decision tree | 0.75384 | 0.75238 |
| Mixture of Experts | 0.78461 | 0.78208 |

## 4.1 Results from Backward elimination

Out of 61 features in the selected dataset 10 important features were selected using backward elimination (fig 4.1).

```
Features: 10/10
```

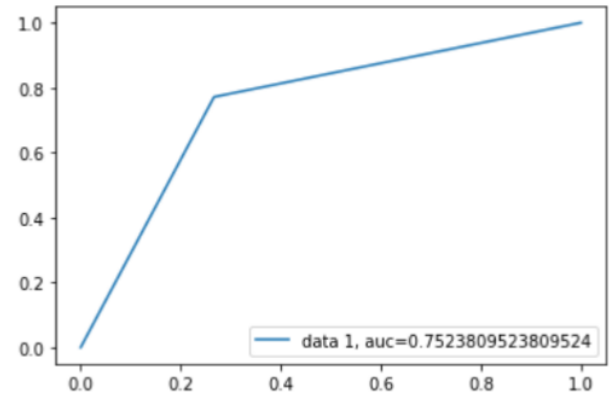
```
feat_names = list(sfs2.k_feature_names_)
print(feat_names)
```

```
['LDHH_lcom', 'WCHU_fanIn', 'CvsEntropy', 'WCHU_numberOfPrivateMethods', 'WCHU_numberOfPrivateAttributes', 'WCHU_noc', 'ck_oo_n
umberOfLinesOfCode', 'WCHU_numberOfAttributesInherited', 'ck_oo_cbo', 'CvsExpEntropy']
```

Figure 4.1: Results from feature selection

## 4.2 Results with conventional classifier

Th accuracy, f1 score and roc curve of a normal decision tree classifier is shown below (fig 4.2 and 4.3).




```
Accuracy of Entropy based Decision Tree: 0.7538461538461538
F1 score 0.7523809523809524
```

Figure 4.2: Results with conventional classifier

```
<matplotlib.axes._subplots.AxesSubplot at 0x2779571c760>
```

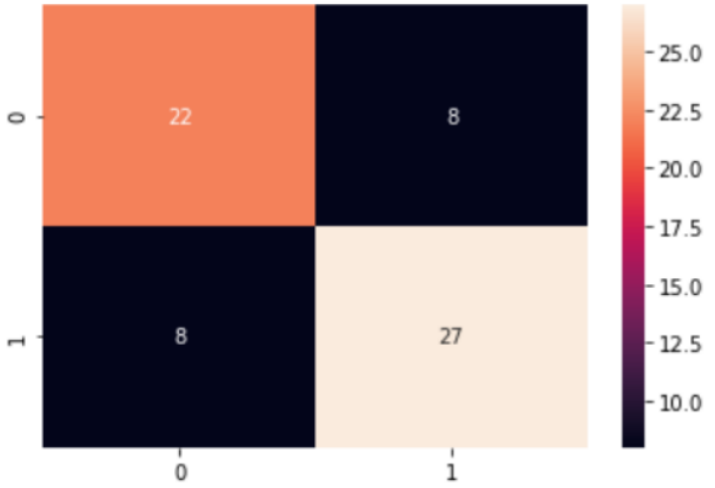



Figure 4.3: Confusion Matrix of conventional classifier

## 4.3 Results with Mixture of Experts

The results after gating the results of various local decision trees trained on individual clusters were significantly better in terms of accuracy and f1 score values as shown (fig 4.4 and 4.5).

```
print("Accuracy of Mixture of Experts:",metrics.accuracy_score(y1_test, predictions))
print("F1 score", f1_score(y1_test,predictions, average='macro'))
```

```
Accuracy of Mixture of Experts: 0.7846153846153846
F1 score 0.782088122605364
```

Figure 4.4: Results with Mixture of Experts

```
: <matplotlib.axes._subplots.AxesSubplot at 0x277957682b0>
```

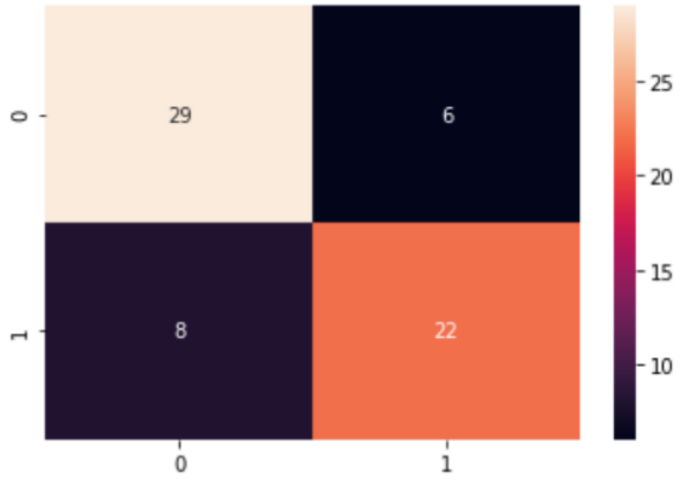



Figure 4.5: Confusion Matrix of ME

# CHAPTER 5

# Work to be done

So far data cleaning, data preprocessing and a basic implementation of Mixture of Expert is done while considering decision tree as local experts and training them on clusters obtained by implicit partitioning and then using a voting classifier as a gating logic. The things that I am going to implement next is given below-

- Compare other conventional classifier like SVM, ANN as a local expert with the decision tree and select the best local expert for each cluster.
- Implement an explicit clustering technique like K-means and compare the results with that of implicit Mixture of Experts model
- Find the best gating logic for combining the results of local experts.

# REFERENCES


[1] Ferrari, D. and Milioni, A.: 2011, Choices and pitfalls concerning mixture-of-experts modeling, *Pesquisa Operacional* **31**, 95–111.

[2] Kamei, Y. and Shihab, E.: 2016, Defect prediction: Accomplishments and future challenges, *2016 IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering (SANER)* **5**, 33–45.

[3] Li, L., Lessmann, S. and Baesens, B.: 2019, Evaluating software defect prediction performance: an updated benchmarking study, *CoRR* **abs/1901.01726**.
**URL:** *http://arxiv.org/abs/1901.01726*

[4] Masoudnia, S. and Ebrahimpour, R.: 2014, Mixture of experts: a literature survey, *Artif. Intell. Rev.* **42**(2), 275–293.
**URL:** *https://doi.org/10.1007/s10462-012-9338-y*

[5] Nam, J.: 2014, Survey on software defect prediction, *Department of Computer Science and Engineering HKUST* .

# MTP-1

ORIGINALITY REPORT

| 16% | 8% | 10% | 7% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | link.springer.com<br>Internet Source | 3% |
| 2 | Denise B Ferrari, Armando Z Milioni. "Choices and pitfalls concerning mixture-of-experts modeling", Pesquisa Operacional, 2011<br>Publication | 3% |
| 3 | Rahul Goel, Rajesh Sharma. "Studying leaders & their concerns using online social media during the times of crisis - A COVID case study", Social Network Analysis and Mining, 2021<br>Publication | 2% |
| 4 | Submitted to University of Strathclyde<br>Student Paper | 2% |
| 5 | Utkarsh Singh, Aditi Totla, Dr. Prakash Kumar. "Deep Learning Model to Predict Pneumonia Disease based on Observed Patterns in Lung X-rays", 2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA), 2020<br>Publication | 1% |

| | | |
|---|---|---|
| 6 | Submitted to Kent Institute of Business and Technology<br>Student Paper | 1 % |
| 7 | eprint.iitd.ac.in<br>Internet Source | 1 % |
| 8 | Submitted to ABV-Indian Institute of Information Technology and Management Gwalior<br>Student Paper | 1 % |
| 9 | www.fao.org<br>Internet Source | <1 % |
| 10 | www.semanticscholar.org<br>Internet Source | <1 % |
| 11 | raw.githubusercontent.com<br>Internet Source | <1 % |
| 12 | Beyza Eken, Ayse Tosun. "Investigating the performance of personalized models for software defect prediction", Journal of Systems and Software, 2021<br>Publication | <1 % |
| 13 | Sushant Kumar Pandey, Ravi Bhushan Mishra, Anil Kumar Tripathi. "Machine learning based methods for software fault prediction: A survey", Expert Systems with Applications, 2021<br>Publication | <1 % |

| | | |
|---|---|---|
| 14 | www.jisc.ac.uk<br>Internet Source | <1% |
| 15 | Steffen Herbold, Alexander Trautsch, Jens Grabowski. "A Comparative Study to Benchmark Cross-Project Defect Prediction Approaches", IEEE Transactions on Software Engineering, 2018<br>Publication | <1% |
| 16 | opastonline.com<br>Internet Source | <1% |
| 17 | Saeed Masoudnia, Reza Ebrahimpour. "Mixture of experts: a literature survey", Artificial Intelligence Review, 2012<br>Publication | <1% |
| 18 | "Software Defect Prediction using Support Vectorised Data and Intelligent Techniques", International Journal of Innovative Technology and Exploring Engineering, 2020<br>Publication | <1% |

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | Off |
| Exclude bibliography | On | | |