Evaluation Report

Contents

[Introduction 2](#_Toc84909756)

[Research setup 2](#_Toc84909757)

[Dataset 2](#_Toc84909758)

[Experiments 3](#_Toc84909759)

[Analysis 1: Prediction model 4](#_Toc84909760)

[AUC Score 4](#_Toc84909761)

[F1-score 5](#_Toc84909762)

[Analysis 2: Feature reduction 6](#_Toc84909763)

[Correlation-based feature selection 6](#_Toc84909764)

[Recursive Feature Elimination 8](#_Toc84909765)

[Analysis 3: Feature selection method 10](#_Toc84909766)

[Correlation-based feature selection 10](#_Toc84909767)

[Recursive feature elimination 10](#_Toc84909768)

[Model Highlight 12](#_Toc84909769)

[Analysis 4: Performance comparison with other algorithms 13](#_Toc84909770)

[Graphical Comparison 14](#_Toc84909771)

[Conclusion 15](#_Toc84909772)

[References 16](#_Toc84909773)

# Introduction

This report covers our research analysis for our application’s performance based on available configurations within our application. Our objective is to find the combination which returns the best results, particularly when handling imbalanced datasets. Additionally, the results gained will also provide insights towards this field of research.

# Research setup

## Dataset

During our research, we analysed the datasets from NASA and promise repository. Through our analysis, we studied the properties of each dataset and tabulate the information as followed:

**Table 1: Properties of datasets from NASA (Shepherd et al., 2014) repository**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Procedural Metric** | **Object oriented Metric** | **Misc.**  **Metric** | **Metric count** | **Defective** | **Not defective** | **Module count** | **Degree of imbalance** |
| CM1.arff | 25 | 11 | 2 | 38 | 12.84% | 87.16% | 327 | Moderate |
| JM1.arff | 17 | 4 | 0 | 21 | 20.88% | 79.12% | 7720 | Mild |
| KC1.arff | 17 | 4 | 0 | 21 | 25.3% | 74.7% | 1162 | Mild |
| KC3.arff | 25 | 13 | 2 | 40 | 18.56% | 81.44% | 194 | Moderate |
| KC4.arff  (Raw) | 24 | 14 | 2 | 40 | 48.8% | 51.2% | 125 | Normal |
| MC1.arff | 24 | 13 | 2 | 39 | 1.84% | 98.16% | 1952 | High |
| MC2.arff | 24 | 13 | 2 | 39 | 35.48% | 64.52% | 124 | Mild |
| MW1.arff | 24 | 11 | 2 | 37 | 10% | 90% | 250 | Moderate |
| PC1.arff | 24 | 11 | 2 | 37 | 8.1% | 91.9% | 679 | Moderate |
| PC2.arff | 23 | 11 | 2 | 36 | 2.22% | 97.78% | 722 | High |
| PC3.arff | 24 | 11 | 2 | 37 | 12.35% | 87.65% | 1053 | Moderate |
| PC4.arff | 24 | 11 | 2 | 37 | 13.86% | 86.14% | 1270 | Moderate |
| PC5.arff | 23 | 13 | 2 | 38 | 27.04% | 72.96% | 1694 | Mild |

**Table 2: Properties of datasets from Promise (Menzies, 2004) repository**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Dataset** | **Procedural Metric** | **Object oriented Metric** | **Misc.**  **Metric** | **Metric count** | **Defective** | **Not defective** | **Module count** | **Degree of imbalance** |
| kc1.arff | 17 | 4 | 0 | 21 | 15.46% | 84.54% | 2109 | Moderate |
| cm1.arff | 17 | 4 | 0 | 21 | 9.84% | 90.16% | 498 | Moderate |
| kc2.arff | 17 | 4 | 0 | 21 | 20.5% | 79.5% | 522 | Mild |
| jm1.arff | 17 | 4 | 0 | 21 | 19.35% | 80.65% | 10885 | Moderate |
| pc1.arff | 17 | 4 | 0 | 21 | 19.35% | 80.65% | 1109 | Moderate |

These datasets will be used throughout this research. Furthermore, these datasets were also used for our second evaluation report which focuses on comparing our results with other research papers.

For the following experiments, our focus would mainly be on datasets which are highly imbalanced. As such, our experiments were conducted using datasets that were categorized to have a moderate or high degree of imbalance. The selected datasets are as followed:

* Moderate: CM1, KC3, MW1, PC1, PC3, PC4
* High: MC1, PC2

## Experiments

Our devised experiments were used to study the performance of our algorithm based on the following factors:

1. Degree of imbalance
2. The prediction models selected
3. The feature selection used
4. The number of features reduced

The application we devised will allow us to perform experiments for analysing the factors mentioned. In addition, our program allows us to test and evaluate the performance of our built models using 4 evaluation metrics.

From the results gained, we will be able to determine the influence of each factor towards the performance of our fault detection method. Furthermore, we will be able to identify the optimal values and selections which allows will the best results to be achieved.

# Analysis 1: Prediction model

There are several prediction models included within our program, each having different levels of proficiency towards handling imbalanced datasets. To compare the performance of each model, we used our program to obtain the performance result for all models in chart and tabular form. On a side note, no feature selection methods were used as this analysis focuses solely on the effectiveness of each model.

## AUC Score

**Table 3: AUC results for the model analysis**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model name** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** |
| CM1.arff | 0.559 | 0.669 | 0.705 | 0.481 | 0.697 | 0.798 | 0.723 | 0.745 |
| KC3.arff | 0.415 | 0.566 | 0.501 | 0.424 | 0.506 | 0.545 | 0.533 | 0.528 |
| MC1.arff | 0.602 | 0.93 | 0.909 | 0.61 | 0.82 | 0.968 | 0.933 | 0.926 |
| MW1.arff | 0.475 | 0.455 | 0.508 | 0.378 | 0.461 | 0.636 | 0.6 | 0.566 |
| PC1.arff | 0.611 | 0.818 | 0.75 | 0.714 | 0.796 | 0.864 | 0.769 | 0.841 |
| PC2.arff | 0.646 | 0.665 | 0.92 | 0.479 | 0.876 | 0.788 | 0.717 | 0.878 |
| PC3.arff | 0.612 | 0.739 | 0.831 | 0.563 | 0.766 | 0.845 | 0.788 | 0.836 |
| PC4.arff | 0.672 | 0.893 | 0.831 | 0.646 | 0.783 | 0.909 | 0.829 | 0.894 |
| **Average performance** | 0.574 | 0.716875 | 0.744375 | 0.536875 | 0.713125 | 0.794125 | 0.7365 | 0.77675 |

**Fig. 1 Bar chart displaying the AUC evaluation scores for the model analysis**

As shown, most models performed outstandingly well in this field. In literature, Naïve Bayes and Logistic Regression are known to be proficient for handling imbalanced dataset. So, the scores for these two predictions were expected to perform greater than other base predictors. If observe from Fig 1, these two base models shown to achieve scores which outperform others, these scores fall between acceptable and excellent range. If we look at the datasets with the highest degree of imbalanced, MC1 and PC2, both models show outstanding performance. Overall, the Logistic regression model shown to have results for handling imbalanced datasets.

For the ensemble predictors, these models overall perform greater than most base prediction models. Consequently, the voting and random forest models achieved the best performance among all models.

## F1-score

**Table 4: F1-score results for the model analysis**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model name** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** |
| CM1.arff | 0.236 | 0.312 | 0.272 | 0.13 | 0.321 | 0.365 | 0.324 | 0.337 |
| KC3.arff | 0.113 | 0.182 | 0.143 | 0.126 | 0.135 | 0.15 | 0.224 | 0.137 |
| MC1.arff | 0.021 | 0.303 | 0.086 | 0 | 0.136 | 0.433 | 0.306 | 0.277 |
| MW1.arff | 0.076 | 0.025 | 0.069 | 0 | 0.04 | 0.178 | 0.104 | 0.05 |
| PC1.arff | 0.111 | 0.257 | 0.253 | 0.201 | 0.176 | 0.259 | 0.22 | 0.25 |
| PC2.arff | 0.067 | 0.114 | 0.171 | 0.067 | 0.087 | 0.067 | 0.178 | 0.126 |
| PC3.arff | 0.219 | 0.282 | 0.296 | 0.168 | 0.247 | 0.312 | 0.294 | 0.283 |
| PC4.arff | 0.251 | 0.525 | 0.387 | 0.237 | 0.343 | 0.449 | 0.416 | 0.392 |
| **Average performance** | 0.13675 | 0.25 | 0.209625 | 0.116125 | 0.185625 | 0.276625 | 0.25825 | 0.2315 |

**Fig. 2 Bar chart displaying the F1-score evaluation scores for the model analysis**

For F1-score, none of models were able to achieve outstanding results, which is a common outcome for this field. An interesting observation was that every model shown to perform poorly for the MW1, which can be observed in 3rd row of Table 4.

An interesting observation was that the best scores were achieved mainly by the tree-based models. The decision tree achieved promising results and shows to have the best results when compared with the other base prediction models. The Rotation Forest had the best performance, achieving remarkable scores and shown good consistency.

One thing to highlight would be the performance of the Logistic regression and Voting models. While the scores achieved by these models were not greater than the tree-based models, they are above average and show consistency.

# Analysis 2: Feature reduction

For this analysis, we will introduce feature selection methods to our experiment. These methods will reduce the metric for the data to fit each model. The idea behind the feature selection methods is that not all metrics are good indicators for faultiness of a software. These algorithms are used to reduce the metrics so that only useful metrics remain which will improve the performance of our program. There are several discoveries that were identified through testing which relates to this topic.

## Correlation-based feature selection

The correlation-based feature selection is a supervised method which will rank attributes for a given dataset based on its subset’s correlation with the class label. For our algorithm, the user will be able to input the number of features to reduce to from a given dataset. Below are the average results of each model for CFS using the 8 datasets stated previously with varying feature reduction values:

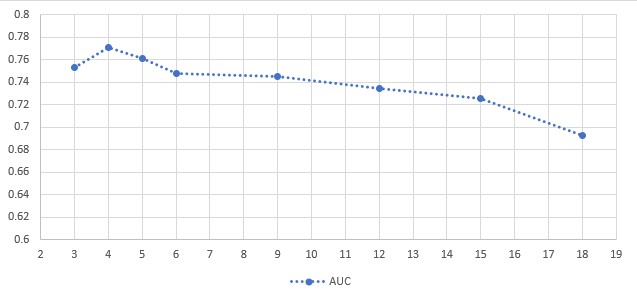
**Table 5: Average AUC results for CFS analysis**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature reduction** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** | **Average**  **model**  **score** |
| 3 | 0.647 | 0.711 | 0.77575 | 0.740875 | 0.8295 | 0.792875 | 0.72825 | 0.800625 | 0.753234 |
| 4 | 0.680625 | 0.73425 | 0.79925 | 0.751375 | 0.828125 | 0.80825 | 0.75125 | 0.813 | 0.770766 |
| 5 | 0.689125 | 0.732125 | 0.791375 | 0.719375 | 0.827875 | 0.804125 | 0.715625 | 0.809625 | 0.761156 |
| 6 | 0.61625 | 0.72375 | 0.780375 | 0.6825 | 0.814 | 0.794125 | 0.754375 | 0.8185 | 0.747984 |
| 9 | 0.63875 | 0.72325 | 0.777625 | 0.623625 | 0.807375 | 0.8135 | 0.765875 | 0.809375 | 0.744922 |
| 12 | 0.70425 | 0.70075 | 0.7595 | 0.591125 | 0.787125 | 0.7875 | 0.73825 | 0.807375 | 0.734484 |
| 15 | 0.685375 | 0.692 | 0.778 | 0.557 | 0.76575 | 0.788625 | 0.728 | 0.80975 | 0.725563 |
| 18 | 0.590875 | 0.663 | 0.7375 | 0.581375 | 0.717875 | 0.7725 | 0.7065 | 0.768625 | 0.692281 |
| **Max** | 0.70425 | 0.73425 | 0.79925 | 0.751375 | 0.8295 | 0.8135 | 0.765875 | 0.8185 |  |
| **Min** | 0.590875 | 0.663 | 0.7375 | 0.557 | 0.717875 | 0.7725 | 0.7065 | 0.768625 |

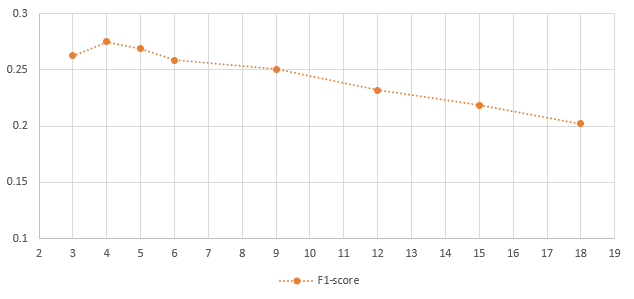
**Table 6: Average F1-score results for CFS analysis**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature reduction** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** | **Average**  **model**  **score** |
| 3 | 0.2075 | 0.272625 | 0.222375 | 0.26225 | 0.26175 | 0.28775 | 0.27775 | 0.308625 | 0.262578 |
| 4 | 0.195375 | 0.291375 | 0.25125 | 0.299875 | 0.23875 | 0.303375 | 0.307125 | 0.312625 | 0.274969 |
| 5 | 0.190875 | 0.287375 | 0.236 | 0.295875 | 0.242625 | 0.312 | 0.2805 | 0.304875 | 0.268766 |
| 6 | 0.1775 | 0.289625 | 0.224625 | 0.229625 | 0.240125 | 0.291 | 0.327625 | 0.29 | 0.258766 |
| 9 | 0.1845 | 0.27025 | 0.234 | 0.214875 | 0.231 | 0.312625 | 0.282875 | 0.2735 | 0.250453 |
| 12 | 0.184 | 0.239 | 0.21625 | 0.19 | 0.213375 | 0.2955 | 0.26725 | 0.248875 | 0.231781 |
| 15 | 0.16725 | 0.233 | 0.22625 | 0.150875 | 0.2165 | 0.273875 | 0.244625 | 0.23725 | 0.218703 |
| 18 | 0.143875 | 0.22575 | 0.196125 | 0.14375 | 0.1905 | 0.265875 | 0.231375 | 0.221 | 0.202281 |
| **Max** | 0.2075 | 0.291375 | 0.25125 | 0.299875 | 0.26175 | 0.312625 | 0.327625 | 0.312625 |  |
| **Min** | 0.143875 | 0.22575 | 0.196125 | 0.14375 | 0.1905 | 0.265875 | 0.231375 | 0.221 |

From the results gained, we can observe that the CFS method works particularly well when the feature reduction value is below 6. Logistic regression along with all ensemble models shown consistent results when the feature reduction value varies. The Decision tree, Multi-Layer Perceptron and Naïve Bayes base predictors has a more visible effect with greater decrease in performance from a reduction value of 6 onwards. The Complement Naïve Bayes shows no visible pattern with random changes between each interval.



**Fig. 3 Line chart displaying the Average model AUC for CFS analysis**



**Fig. 4 Line chart displaying the Average model F1-score for CFS analysis**

From the charts shown, we can view that the performance peak when the feature reduction value is between 3 to 6. Additionally, any value above 6 shown to gradually decrease the overall score for both AUC and F1-score.

## Recursive Feature Elimination

The recursive feature elimination function is a supervised method which uses a recursive approach to remove the least important feature until a set number of features remain. Similar to the CFS method, our program allows us to configure the number of features to reduce. Below are the average results of each model for RFE using the 8 datasets stated previously with varying feature reduction values:

**Table 7: Average AUC results for RFE analysis**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature reduction** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** | **Average**  **model**  **score** |
| 3 | 0.646125 | 0.653625 | 0.698 | 0.66175 | 0.68575 | 0.707625 | 0.676625 | 0.71525 | 0.680594 |
| 6 | 0.7145 | 0.669 | 0.785875 | 0.65825 | 0.73475 | 0.771 | 0.683375 | 0.77775 | 0.724313 |
| 9 | 0.778125 | 0.686875 | 0.777 | 0.735125 | 0.755 | 0.777625 | 0.710375 | 0.785125 | 0.750656 |
| 12 | 0.76925 | 0.731375 | 0.760375 | 0.71275 | 0.74675 | 0.797375 | 0.72175 | 0.783125 | 0.752844 |
| 15 | 0.7875 | 0.714875 | 0.786875 | 0.75475 | 0.7555 | 0.790625 | 0.7265 | 0.782125 | 0.762344 |
| 16 | 0.802 | 0.7205 | 0.77775 | 0.721 | 0.759125 | 0.8035 | 0.7315 | 0.78875 | 0.763016 |
| 17 | 0.78925 | 0.727625 | 0.76675 | 0.722 | 0.751125 | 0.797125 | 0.7275 | 0.77775 | 0.757391 |
| 18 | 0.78775 | 0.70025 | 0.806375 | 0.72525 | 0.746875 | 0.796875 | 0.714125 | 0.80075 | 0.759781 |
| **Max** | 0.802 | 0.731375 | 0.806375 | 0.75475 | 0.759125 | 0.8035 | 0.7315 | 0.80075 |  |
| **Min** | 0.646125 | 0.653625 | 0.698 | 0.65825 | 0.68575 | 0.707625 | 0.676625 | 0.71525 |

**Table 8: Average AUC results for CFS analysis**

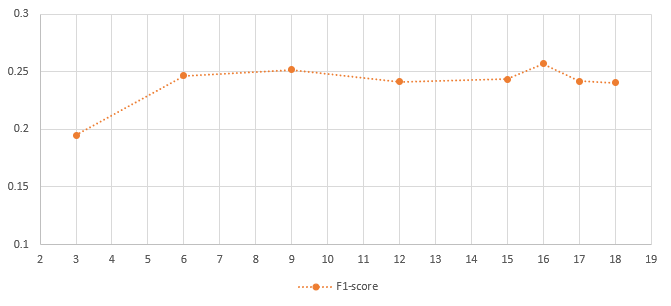
|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Feature reduction** | **Complement Naive Bayes** | **Decision Tree** | **Logistic regression** | **Multi-Layer Perceptron** | **Naive Bayes** | **Random Forest** | **Rotation Forest** | **Voting** | **Average**  **model**  **score** |
| 3 | 0.181375 | 0.230375 | 0.113625 | 0.206875 | 0.162125 | 0.250875 | 0.20975 | 0.202625 | 0.194703 |
| 6 | 0.191 | 0.243625 | 0.244625 | 0.25625 | 0.207375 | 0.281125 | 0.276625 | 0.27075 | 0.246422 |
| 9 | 0.217125 | 0.246375 | 0.236 | 0.29125 | 0.214375 | 0.256375 | 0.2705 | 0.281875 | 0.251734 |
| 12 | 0.1855 | 0.260375 | 0.261625 | 0.242875 | 0.201125 | 0.27175 | 0.260375 | 0.245375 | 0.241125 |
| 15 | 0.196375 | 0.2535 | 0.272125 | 0.251625 | 0.202875 | 0.242375 | 0.246625 | 0.281 | 0.243313 |
| 16 | 0.20575 | 0.259375 | 0.28775 | 0.27625 | 0.196625 | 0.275 | 0.2585 | 0.297 | 0.257031 |
| 17 | 0.194375 | 0.2695 | 0.2345 | 0.231875 | 0.194375 | 0.264875 | 0.27825 | 0.267 | 0.241844 |
| 18 | 0.198 | 0.2435 | 0.253375 | 0.252 | 0.198 | 0.255375 | 0.259 | 0.263 | 0.240281 |
| **Max** | 0.217125 | 0.2695 | 0.28775 | 0.29125 | 0.214375 | 0.281125 | 0.27825 | 0.297 |  |
| **Min** | 0.181375 | 0.230375 | 0.113625 | 0.206875 | 0.162125 | 0.242375 | 0.20975 | 0.202625 |

Unlike the CFS method, a greater feature reduction value seems to show better results for all models, as observed in the tables above. Unlike CFS, only ensemble methods show consistent results. While the changes in performance are visible for the base models, there is less value difference between each interval. Hence, the acceptable range for better performance is much wider as compared to CFS.

Chart, line chart

Description automatically generated

**Fig. 5 Line chart displaying the Average model AUC for RFE analysis**



**Fig. 6 Line chart displaying the Average model F1-score for RFE analysis**

From the charts shown, we can view that the performance peak when the feature reduction value is between 15 to 16. Unlike the CFS method, an increase in feature reduction shown greater improvements towards the performance for both evaluation scores.

# Analysis 3: Feature selection method

This analysis uses the results from analysis 2 to determine the best feature selection methods for each model and finding the performance increase when compared to the results from analysis 1.

**Table 9: Results for comparison between feature selection methods**

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Model Name | CFS | | | | | | RFE | | | | | |
| AUC | | | F1-score | | | AUC | | | F1-score | | |
| Max | Min | Average | Max | Min | Average | Max | Min | Average | Max | Min | Average |
| Complement Naive Bayes | 0.7043 | 0.5909 | 0.6565 | 0.2075 | 0.1439 | 0.1814 | 0.8020 | 0.6461 | 0.7593 | 0.2171 | 0.1814 | 0.1962 |
| Decision Tree | 0.7343 | 0.6630 | 0.7100 | 0.2914 | 0.2258 | 0.2636 | 0.7314 | 0.6536 | 0.7005 | 0.2695 | 0.2304 | 0.2508 |
| Logistic regression | 0.7993 | 0.7375 | 0.7749 | 0.2513 | 0.1961 | 0.2259 | 0.8064 | 0.6980 | 0.7699 | 0.2878 | 0.1136 | 0.2380 |
| Multi-Layer Perceptron | 0.7514 | 0.5570 | 0.6559 | 0.2999 | 0.1438 | 0.2234 | 0.7548 | 0.6583 | 0.7114 | 0.2913 | 0.2069 | 0.2511 |
| Naive Bayes | 0.8295 | 0.7179 | 0.7972 | 0.2618 | 0.1905 | 0.2293 | 0.7591 | 0.6858 | 0.7419 | 0.2144 | 0.1621 | 0.1971 |
| Random Forest | 0.8135 | 0.7725 | 0.7952 | 0.3126 | 0.2659 | 0.2928 | 0.8035 | 0.7076 | 0.7802 | 0.2811 | 0.2424 | 0.2622 |
| Rotation Forest | 0.7659 | 0.7065 | 0.7360 | 0.3276 | 0.2314 | 0.2774 | 0.7315 | 0.6766 | 0.7115 | 0.2783 | 0.2098 | 0.2575 |
| Voting | 0.8185 | 0.7686 | 0.8046 | 0.3126 | 0.2210 | 0.2746 | 0.8008 | 0.7153 | 0.7763 | 0.2970 | 0.2026 | 0.2636 |

The red encoded text indicates the best results between the two selection methods. We determine the best method for each model using the table above by the count of red encoded data. The best feature selection method based on our analysis are as followed:

# Correlation-based feature selection

* Decision Tree
* Naive Bayes
* Random Forest
* Rotation Forest
* Voting

# Recursive feature elimination

* Complement Naive Bayes
* Logistic regression
* Multi-Layer Perceptron

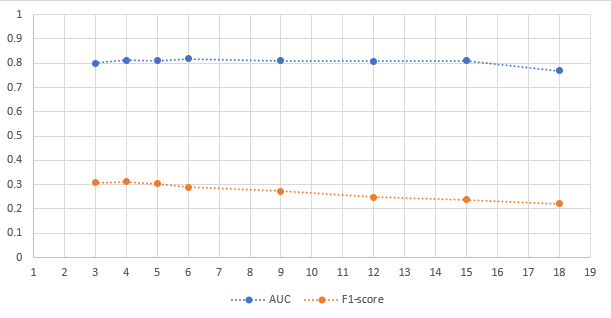
**Table 10: Performance increase from the base and max score for each model**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model Name | AUC | | | F1-score | | |
| Base | Max | % Increase | Base | Max | % Increase |
| Complement Naive Bayes | 0.574 | 0.802 | 39.72125 | 0.13675 | 0.217125 | 58.77514 |
| Decision Tree | 0.716875 | 0.73425 | 2.423714 | 0.25 | 0.291375 | 16.55 |
| Logistic regression | 0.744375 | 0.806375 | 8.329135 | 0.209625 | 0.28775 | 37.26893 |
| Multi-Layer Perceptron | 0.536875 | 0.75475 | 40.58207 | 0.116125 | 0.29125 | 150.8073 |
| Naive Bayes | 0.713125 | 0.8295 | 16.31902 | 0.185625 | 0.26175 | 41.0101 |
| Random Forest | 0.794125 | 0.8135 | 2.439792 | 0.276625 | 0.312625 | 13.01401 |
| Rotation Forest | 0.7365 | 0.765875 | 3.988459 | 0.25825 | 0.327625 | 26.8635 |
| Voting | 0.77675 | 0.8185 | 5.37496 | 0.2315 | 0.312625 | 35.0432 |

With the introduction of feature selection methods, the models shown to have great improvements towards the F1-score. There is also an increase in performance for the AUC of all models, with significant improvements for the Multi-Layer Perceptron and Complement Naïve Bayes models. As such, we can conclude that our proposed method has improved the overall performance of all models within the systems, with significant improvements particularly on models that by nature are not suited for handling imbalanced datasets.

# Model Highlight

From these experiments, we were able to study the nature of various models with various changes. As such, we were also able to identify the model which performs best which we would like to highlight in this section. This model being the Voting ensemble model.



**Fig. 7 Line chart displaying the Average score for Voting model**

From previous analysis, we were able to identify the best feature selection method for this model so this section will be focused only on this combination of model and feature selection. If we observe the chart in Figure 7, the voting ensemble model has shown the most consistent results. Additionally, the Voting model also achieved the second highest score on the average AUC and F1-score among all models. In essence, the Voting model outperforms other models in terms of consistency and achieving scores greater than majority of the models available. This led us to decide that the Voting method is the best model for our system.

# Analysis 4: Performance comparison with other algorithms

This analysis covers the performance of the best algorithm from our proposed system against other software fault prediction algorithms devised from other research papers.

**Table 10: AUC and F1-score results comparisons with other proposed algorithms**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Datasets used** | Results | | | | | |
| Our Algorithm | | VOT(ROF, RF, LB)  (Yucalar et al.) | | SDAEsSTE (Tong et al.) | |
| AUC | F1 | AUC | F1 | AUC | F1 |
| CM1 | 0.806 | **0.391** | N/A | N/A | **0.8373** | 0.2882 |
| JM1 | 0.654 | **0.325** | N/A | N/A | **0.7731** | 0.3174 |
| KC1 | 0.625 | 0.386 | N/A | N/A | **0.7426** | **0.3946** |
| MC1 | 0.958 | **0.457** | N/A | N/A | **0.9614** | 0.2217 |
| MC2 | **0.815** | **0.635** | N/A | N/A | 0.6846 | 0.5871 |
| MW1 | 0.73 | 0.08 | N/A | N/A | **0.8597** | **0.4073** |
| PC1 | 0.846 | 0.244 | N/A | N/A | **0.9062** | **0.3062** |
| PC2 | 0.913 | **0.25** | N/A | N/A | **0.9264** | 0.2154 |
| PC3 | **0.859** | 0.326 | N/A | N/A | 0.8381 | **0.3545** |
| PC4 | **0.889** | 0.527 | N/A | N/A | 0.8846 | **0.5544** |
| cm1 | 0.752 | 0.264 | **0.784** | N/A | N/A | N/A |
| jm1 | 0.633 | 0.285 | **0.753** | N/A | N/A | N/A |
| kc1 | 0.636 | 0.291 | **0.845** | N/A | N/A | N/A |
| pc1 | 0.865 | 0.354 | **0.876** | N/A | N/A | N/A |

After evaluating the performances of each of our algorithms, we then selected the best performing algorithm for comparison with other algorithms from other research papers, such as the SDAEsSTE approach devised by Tong et al. The algorithms are compared based on the AUC and F1-scores results which were present on the research paper themselves, and the data is tabulated based on the various datasets which were used on the algorithms. The comparisons can be found in the table at the results section. The table above shows our results when compared against other algorithms from other research papers. To better visualise the results of the analysis, graphs have also been created, and they can be viewed in the next page.

# Graphical Comparison

**Fig. 7 Bar chart displaying the differences in performance of our algorithm and other algorithms in different datasets for the AUC performance metric**

**Fig. 8 Bar chart displaying the differences in performance of our algorithm and other algorithms in different datasets**

From the results above, we can see that our algorithm outperforms Tong et al’s SDAEsSTE algorithm in certain datasets. On the other hand, our algorithm is not able to outperform Yucalar et al’s algorithm which is a combination of ensemble predictors.

# Conclusion

From this analysis, we were able to determine the optimal configurations for achieving the best results from every models. The experiments performed has revealed valuable information on nature of each model and its changes with the introduction of feature selection methods. The results also led us to identify the best configuration for our system, that is the Voting model with the CFS feature selection method. Though that is said, our proposed method lacks slightly in performance when compared with the two other algorithms mentioned in this report. Despite this being the case, we have come to the conclusion that although our best algorithm is not the best in terms of performance, it is still a decently viable method to use when a dataset is found to be imbalanced, as parts of our algorithm are built to handle those situations.

# References

Menzies, T. (2004). *Promise Software Engineering Repository.* Retrieved May 15, 2021 from

<http://promise.site.uottawa.ca/SERepository/datasets-page.html>

Shepherd et al. (2014). *NASADefectDataset.* Retrieved May 15, 2021 from

<https://github.com/klainfo/NASADefectDataset/tree/master/CleanedData/MDP/D>''