Marcos Hernandez
CSC 350 Data Mining
Professor Gao

Software Quality Classification Project Report

The aim of this project is to build and investigate the performance of different two-class classification models (the dependent variable being either fault-prone or not fault-prone): Naïve Bayes (NB), Multilayer Perceptron (MLP), k-Nearest Neighbors (kNN, or IBK), Support Vector Machine (SMV, or SMO), Logistic Regression (LR), C4.5 (J48), and Random Forest(RF100). Using datasets one having more features than the other.

The Area Under ROC (AUC) is used as the metric to measure the prediction accuracy for the different classifier models. As described in the provided case study, AUC is "a single-value measurement originated from the field of signal detention", and it ranges from 0 to 1. A value of 1 is ideal. As in the study provided, the results in this report are derived from models built using the WEKA tool. WEKA has default settings for its classification models, but these can be changed in order to increase the algorithms prediction performance. However, all but three of the classifiers—MLP, kNN, and SVM, use the WEKA provided default settings. These three classifiers were configured with custom parameters to optimize their performance:

MLP(settings: c=5.0 and validationSetSize= 10),
kNN (distanceWeighting= Weight by 1/distance),
kNN=5, crossValidation = true), and SMV (c= 5.0, and buildLogisticModels= true)

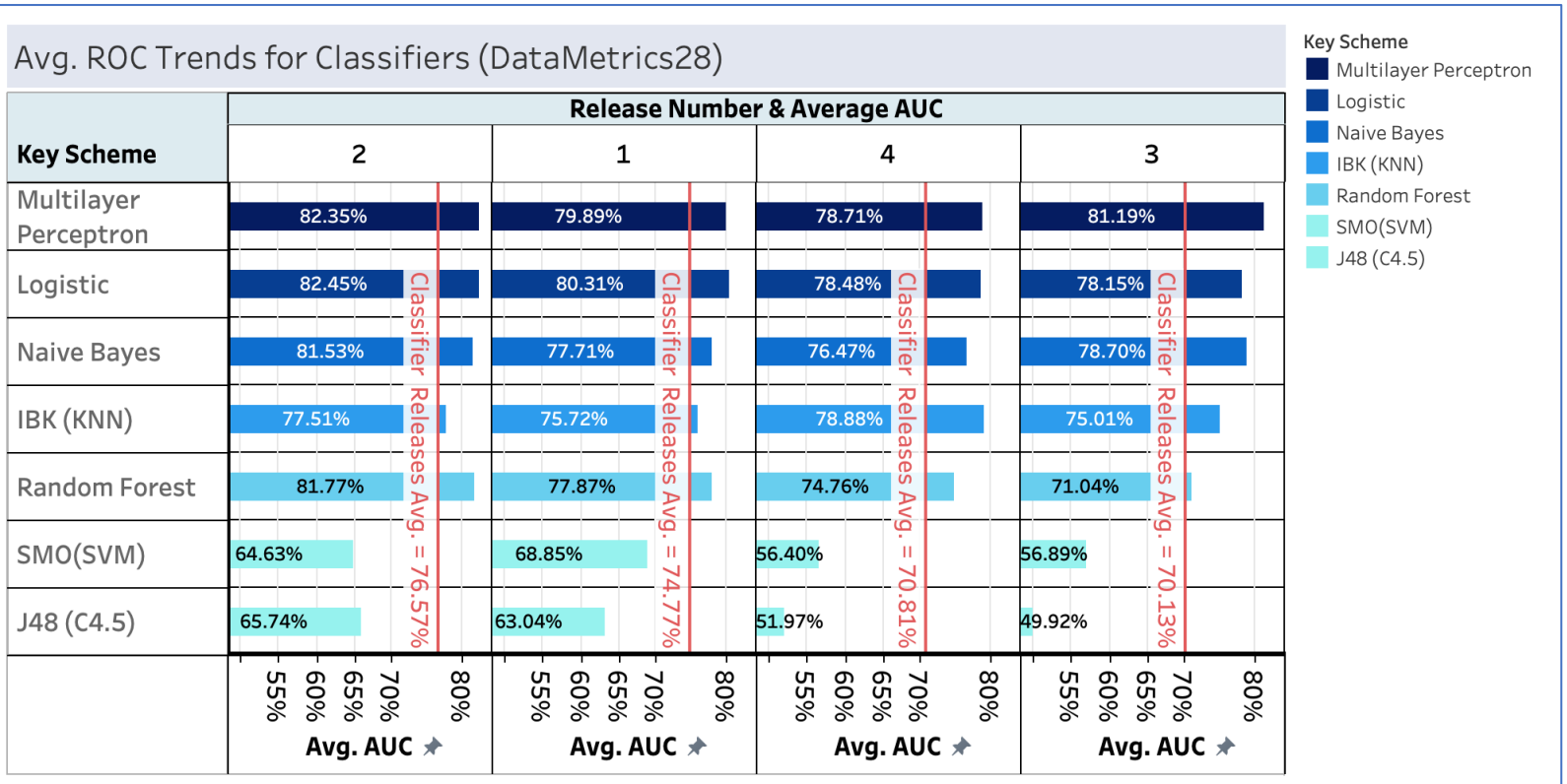
To build and validate the models, the 10-runs of 5-fold classification method is used. This process was repeated twice using different sets of data (Data28metrics, and Data42metrics). The datasets originate from a Large Legacy Telecommunications Software system, and they contain software metrics and fault data, which was collected over 4 historical releases for both datasets, meaning each of the datasets is composed of four historical system releases. Both datasets contain 24 product metrics, and 4 execution metrics. In addition to these features, the Data42metrics also contains an additional 14 process metrics. The most notable difference between the two datasets is that in comparison to the Data28metrics, the Data42metrics contains and additional 14 extra features:

1	PRS_PRS	8	DIFFFT
2	VO_PRS	9	SGRTHTOT
3	INT	10	SMODTOT
4	VO	11	UNQUPDID
5	EXT	12	VLOUPDSM
6	FTUPD	13	LOUPDSM
7	UPD	14	UPD_CAR

The goal of the investigation is to see if we can reduce the number of features in a dataset to reduce the computational power and time needed to train and run the classifier models, while maintaining or maybe even increasing the prediction accuracy. The classifier models will predict the two class dependent variable. Which in this case consists of the mentioned two class feature, fault-prone or not-fault-prone.

Marcos Hernandez
CSC 350 Data Mining
Professor Gao

When comparing the classification results for all of the classifier models on both datasets, using the AUC as a metric for comparison, we can clearly see some trends As shown in the visualizations below.



Tabular Average Area under ROC (DataMetrics28)

Key Scheme	Release Number & Average AUC 1	2	3	4
Multilayer Perceptron	79.89%	82.35%	81.19%	78.71%
Logistic	80.31%	82.45%	78.15%	78.48%
Naive Bayes	77.71%	81.53%	78.70%	76.47%
IBK (KNN)	75.72%	77.51%	75.01%	78.88%
Random Forest	77.87%	81.77%	71.04%	74.76%
SMO(SVM)	68.85%	64.63%	56.89%	56.40%
J48 (C4.5)	63.04%	65.74%	49.92%	51.97%

Avg. Area under ROC
49.92% 82.45%

(A bar graph, and tabular representation of the average AUC for the Data28metrics for the 5-fold 10-run cross validation method for the classifier models.)

In both dashboards above and below, the names of classifiers models are under the Key Scheme label. Along side the different releases(1-4) for the different datasets are also labeled. Their respective AUC is also depicted. The bar graph visualization has both, the key scheme and the releases sorted. The bar graph was first sorted from the left to right (for Data28metric above and Data42metric below) by release number from greatest average AUC of all classifying

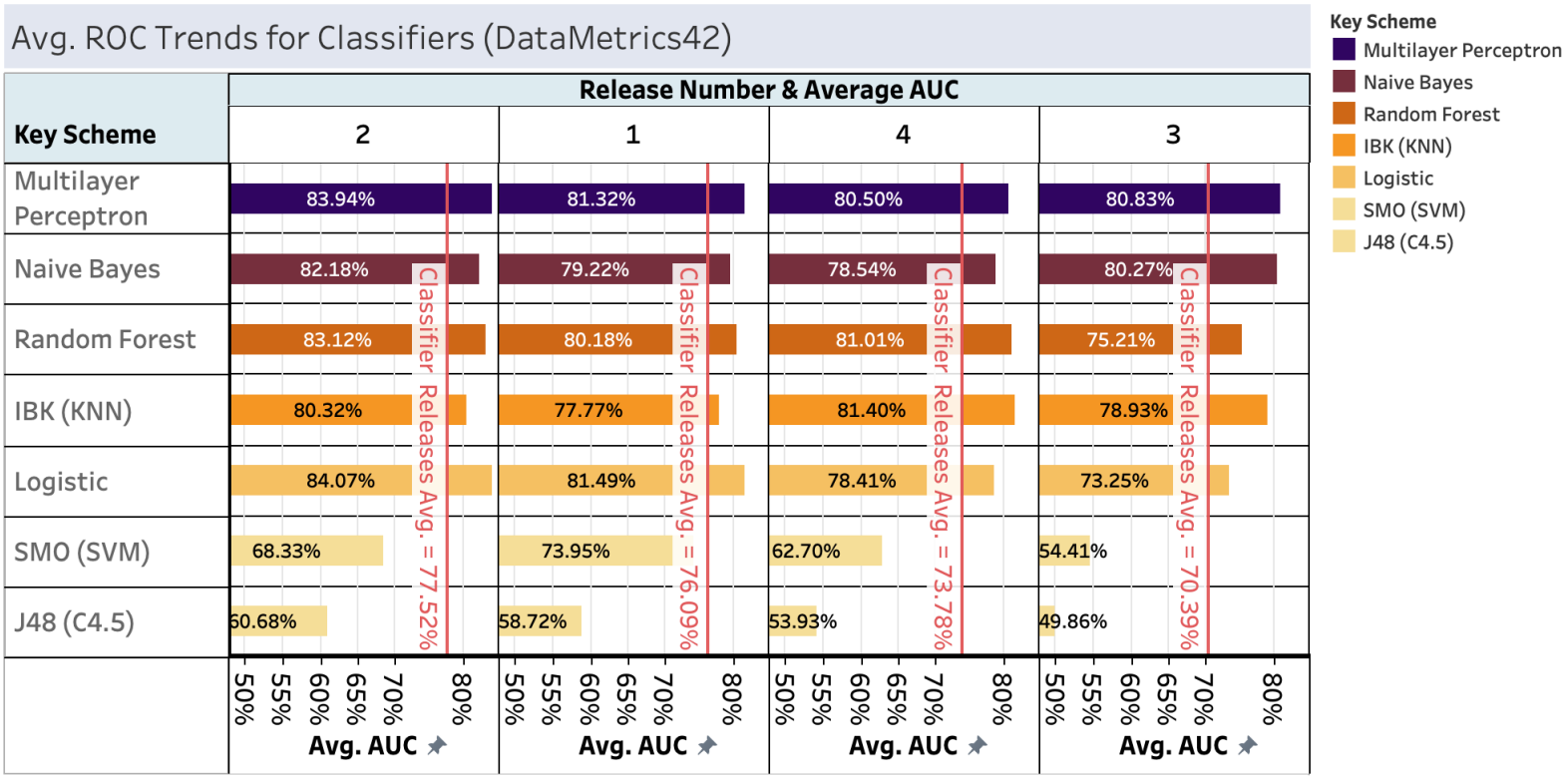
models for the different dataset releases(1-4). Meaning that as depicted in the graph above, the release number 2 yielded the greatest average AUC, with an average AUC of 76.57% for all of the different classifier models combined. Followed by release number 1 with a 74.77%, then 4 with a 70.81%, and lastly by release number 3 which yielded an average AUC of 70.13% for all of the classifier models using the release number 3 dataset form the Data28metric. Then bar graphs also sort the classifier models from the greatest average AUC at the top, to the classifier model with the lowest average AUC at the bottom, which are Multilayer Perceptron and C4.5(J48) respectively for the dashboard above. This effectively sorts the values as greatest average AUC at the top left, to the lowest average AUC at the bottom right. The tabular graph, only sorts the classifier models with greatest average AUC at the top and the model with the lowest at the bottom. The results by release number are not ordered in any particular order other than the one specified in the project assignment handout.

The dashboard below is in the same format but this one using the results of the Data42metric instead. The order of the release numbers in both bar graphs for both datasets is the same as shown in both dashboards. This means that there is consistency across both Data42metrics and Data28metrics for which release has the greatest AUC, release 2. As well as which release has the lowest average AUC, 1. Their order is the same for both datasets. Another interesting trend that can be seen when comparing the dashboards is that in both, the Multilayer Perceptron model had the greatest average AUC, while the C4.5 model performed the lowest in both data metrics.

When comparing the performance of the classifier models across both datametrics using both dashboards, we can see a lot of similarity. In both, the Multilayer Perceptron performed best while C4.5 performed the worst, and the kNN classifier model remained in the same position for both performing right at the center when benchmarked against the others. A major difference when comparing the model performance across both metrics was that in the dashboard above for the Data28metrics, the Logistic Regression (logistic) performed second best when compared to the other models. But, as depicted in the bar graph below, in the Data42metrics results dashboard, the Logistic Regression was overtaken by both the Naïve Bayes, and Random Forest models. As well as the kNN model which simply stayed in the middle placement, but the Logistic model still performed under it.

As mentioned at the beginning the goal was to reduce the number of features in the dataset and maintain or maybe even increase the performance of the models. In this case this was true for some of the classifiers, but not all. Overall the average of the AUC for all of the models and releases was extremely similar from one data metric to the other.

Marcos Hernandez
CSC 350 Data Mining
Professor Gao



Tabular Average Area under ROC (DataMetrics42)

Key Scheme	Release Number & Average AUC 1	2	3	4
Multilayer Perceptron	81.32%	83.94%	80.83%	80.50%
Naive Bayes	79.22%	82.18%	80.27%	78.54%
Random Forest	80.18%	83.12%	75.21%	81.01%
IBK (KNN)	77.77%	80.32%	78.93%	81.40%
Logistic	81.49%	84.07%	73.25%	78.41%
SMO (SVM)	73.95%	68.33%	54.41%	62.70%
J48 (C4.5)	58.72%	60.68%	49.86%	53.93%

Avg. Area under ROC
49.86% 84.07%

(A bar graph, and tabular representation of the average AUC for the Data28metrics for the 5-fold 10-run cross validation method for the classifier models.)

In conclusion, after comparing both results I would say that this was a success, as we were able to decrease the number of features from the Data42metrics by 14 on the Data28metrics and yet have extremely similar results. The benefit of this increases exponentially as the datasets get larger and larger because lowering the number of features used train prediction models also lowers both time and computational costs. I averaged all of the results across the board for each of the two data metrics, and while the Data42metrics(total AUC average: 74.446642%) outperformed the Data28metrics(total AUC average: 73.0679392%) results, I would still argue that the investigation was a success because it lagged behind by only 1.3787028%.

Marcos Hernandez
CSC 350 Data Mining
Professor Gao

References

Gao et al. “Choosing software metrics for defect prediction: an investigation on feature selection techniques”, Software: Practice and Experience. Special Issue: Practical Aspects of Search-Based Software Engineering, vol. 41, no 5, 2011, pp. 579-606. Wiley. DOI: 10.1002/spe.1043