



The Effects of Class Imbalance and Training Data Size on Classifier Learning: An Empirical Study

Wanwan Zheng¹ · Mingzhe Jin¹

Received: 4 December 2019 / Accepted: 7 February 2020 / Published online: 25 February 2020
© Springer Nature Singapore Pte Ltd 2020

Abstract

This study discusses the effects of class imbalance and training data size on the predictive performance of classifiers. An empirical study was performed on ten classifiers arising from seven categories, which are frequently employed and have been identified to be efficient. In addition, comprehensive hyperparameter tuning was done for every data to maximize the performance of each classifier. The results indicated that (1) naïve Bayes, logistic regression and logit leaf model are less susceptible to class imbalance while they have relatively poor predictive performance; (2) ensemble classifiers AdaBoost, XGBoost and parRF have a quite poorer stability in terms of class imbalance while they achieved superior predictive accuracies; (3) for all of the classifiers employed in this study, their accuracies decreased as soon as the class imbalance skew reached a certain point 0.10; note that although using datasets with balanced class distribution would be an ideal condition to maximize the performance of classifiers, if the skew is larger than 0.10, a comprehensive hyperparameter tuning may be able to eliminate the effect of class imbalance; (4) no one classifier shows to be robust to the change of training data size; (5) CART is the last choice among the ten classifiers.

Keywords Class imbalance · Training data size · Hyperparameter tuning · Classifier performance

Introduction

Choosing appropriate classifiers for a given dataset is very important in practice. However, consistent with the no free lunch (NFL) theorem¹ [26], no single classifier seems to be capable of ensuring optimal results for all datasets. Data characteristics are known to have an intrinsic relationship with classifier performance [2, 8, 16, 18]. This study focuses on two kinds of data characteristics: class imbalance and training data size.

Class imbalance is one of the most serious influential factors for the predictive performance of classifiers. The imbalanced data are characterized as having many more instances of certain classes than others. In this case, classifiers tend to make biased learning model that has a poorer predictive accuracy over the minority classes compared to the majority classes. This is because most standard classifier learning

algorithms, such as decision tree, backpropagation neural network and support vector machines, are designed based on assumptions that the class distribution is relatively balanced and the misclassification costs are equal, classification rules that predict the minority classes tend to be rare, undiscovered or ignored [19]. Consequently, test samples belonging to the minority classes are misclassified more often than those belong to the majority classes. It is said that classification of data with imbalanced class distribution has encountered a significant drawback of the performance attainable by most standard classifier learning algorithms.

However, many real-world data mining applications in many domains involve learning from imbalanced datasets and the correct classification of instances in the minority classes often has a greater value than the contrary cases. For example, in a customer churn prediction problem where the churn cases are usually quite rare as compared with normal ones, the goal is to estimate a future churn probability for every customer based on the historical knowledge. Therefore, a favorable classification model is one that provides a

✉ Wanwan Zheng
teiwawan@gmail.com

¹ Graduate School of Culture and Information Science, Doshisha University, 1-3 Tatara Miyakodani, Kyotanabe, Kyoto, Japan

¹ NFL theorem: If algorithm A outperforms algorithm B on some cost functions, then loosely speaking there must exist exactly as many other functions where B outperforms A.

higher identification rate on the churn category. Class imbalance problem is also emphasized in fields of fraud detection, medical diagnostics, spam filtering and network intrusion detection, and so on. Improving the predictive accuracy over the minority class is critical and especially required.

Brown and Mues [3] performed a study of various classifiers based on five real-world credit scoring datasets to deal with the imbalanced credit scoring problem, involving logistic regression (LR), neural networks (NNet), least square support vector machines (LS-SVMs), C4.5 decision trees (C4.5), K-nearest neighbors (KNN), random forests (RF), and gradient boosting (GB). The empirical study created percentage splits as 25%, 20%, 15%, 10%, 5%, 2.5%, and 1% bad observations to identify whether classifiers are adversely affected in prediction. The results indicated that the RF and GB perform very well and can cope with the class imbalance comparatively well. Caigny et al. [4] proposed logit leaf model (LLM) to better classify imbalanced data. LLM was benchmarked against decision trees, LR, RF and logistic model tree (LMT) with regard to the predictive performance using fourteen churn datasets. According to the results, LLM scores significantly better than LR and decision trees and performs at least as well as more advanced ensemble RF and LMT.

In classification studies, the more powerful machine learning algorithms should be able to learn complex non-linear relationships between input and output features. By definition, they are robust to noise, show high variance, and meaning predictions vary based on the specific data used to train them. This added flexibility and power comes at the cost of requiring more training data, often a lot more data. One representative example is deep neural networks, which have regained immense popularity in recent years. The growth of deep neural networks from the 8 layers AlexNet [12], to the 19 layers VGG [20], to the 22 layers GoogleNet [21], followed by the 152 layers ResNet [11] shows a clear generalization of the idea that deeper networks perform better, and the evolution trend is in the direction of even deeper or more complicated structures. However, this trend is ubiquitously built up on the assumption of a sufficiently high sample size [5]. According to Nguyen et al. [14], the quality and size of training data are crucial for supervised learning tasks. Zhu et al. [28] stated that given the size of existing datasets, the current state of the art would need significant additional data to continue producing consistent improvements in performance. Halevy and Norvig [10] reported that even very complex problems in artificial intelligence may be solved by simple statistical models trained on massive datasets.

Therefore, in studies of machine learning, collecting more data should be the most common and easiest way to improve performance of classifiers to a desired level. However, in numerous real-world applications, the number

of samples in a dataset can be relatively limited, such as in studies of rare diseases, extraordinary athletes or medical images, which often have limited sample size restricted by the availability of the respondents as well as the robotics application due to the cost of the data collection process.

To gain data size, Georgakis et al. [9] proposed an automated approach synthetic set generation to augment some hand-labeled training data with synthetic examples carefully composed onto scenes yields object detectors with comparable performance to use much more hand-labeled data. In addition, Zhu et al. [29] proposed a novel self-taught dimensionality reduction (STDR) approach, which is able to transfer external knowledge (or information) from freely available external (or auxiliary) data to the high-dimensional and small-sized target data. Because classifiers require large training datasets to make a complete statistical description of each class, the size of training data has received considerable attention and numerous researches have reported that classification accuracy tends to be positively related to training data size [6, 13, 15, 22].

About the comparison of classifiers, one of the most famous researches is Fernández-Delgado et al. [7]. A total of 179 classifiers arising from 17 categories were evaluated using 121 datasets. They stated that RF, SVM with Gaussian and polynomial kernels, extreme learning machine with Gaussian kernel, C5.0 and avNNet (a commit of multi-layer perceptions implemented in R with caret package) are clearly better than the remaining ones. In response to this paper, Wainberg et al. [25] argued that the results are biased by the lack of a held-out test set and the exclusion of trials with error.

Referring to the methodology, some classifiers find it harder to achieve high predictive accuracy if datasets have large class imbalance. On the other hand, some classifiers have more difficulties in handling low-sample size datasets, such as random forests and bagging which conduct random sampling when building the learning models. Therefore, this study compared a total of ten classifiers, attempted to clarify the effects of class imbalance and training data size on classifier learning and answer the following two questions.

1. How do class imbalance and training data size affect the predictive performance of classifiers?
2. Does the class ratio have a threshold after which the accuracy of classifiers begins to decrease?

The rest of this article is organized as follows: “[Datasets, classifiers, and performance metric](#)” describes the collection of datasets, classifiers and the performance metric employed in this study; “[Experimental analysis and discussion](#)” describes the experiments and discussion, and “[Conclusions and future work](#)” compiles the conclusions and future work.

Datasets, Classifiers, and Performance Metric

Datasets

Table 1 gives information about these datasets. Only binary class problems were considered in this study. For each dataset, five aspects are listed: the number of variables (#Var), the number of instances (#Ins), the instances of minority class (#Min), the instances of majority class (#Maj) and the skew (the #Min-to-#Maj ratio). As we can see, nine datasets have been chosen to be representatives of different kinds of applications, including climate data, physical data, image data, text data as well as artificial data.

Classifiers

A total of ten classifiers arising from seven categories of applications implemented in R were selected. They are decision tree (CART, C5.0), ensemble (parRF, AdaBoost, XGBoost), neural networks (avNNet), Bayesian (NB), regression (LR), hybrid (LLM) and kernel (svmPoly). All of them were frequently employed in previous research and have been identified to be efficient.

When it comes to machine learning, as mentioned above, there is no free lunch. To obtain a more accurate result, data scientists test all possible algorithms for data at hand to identify the champion algorithm. Besides, picking the right algorithm is not enough. The right configuration of the algorithm also must be chosen for a dataset by tuning the hyperparameters. This is where the magic happens.

Table 1 Collection of nine datasets from the OpenML (<https://www.openml.org/>)

| Datasets | #Var | #Ins | #Min | #Maj | skew |
|---------------|------|------|------|------|------|
| Ozone | 72 | 2534 | 160 | 2374 | 0.07 |
| Kc1 | 21 | 2109 | 326 | 1783 | 0.18 |
| Scene | 294 | 2407 | 431 | 1976 | 0.22 |
| Gesture (RS) | 32 | 4037 | 1087 | 2950 | 0.37 |
| Cpu_act | 21 | 8192 | 2477 | 5715 | 0.43 |
| Waveform-5000 | 40 | 5000 | 1692 | 3308 | 0.51 |
| Spambase | 57 | 4601 | 1813 | 2788 | 0.65 |
| Gesture (DP) | 32 | 4838 | 2097 | 2741 | 0.77 |
| Madelone | 500 | 2600 | 1300 | 1300 | 1.00 |

It shows the number of variables (#Var), instances (#Ins), instances of minority class (#Min) and instances of majority class (#Maj) for each dataset

skew = #Min/#Maj

Therefore, the hyperparameter tuning was done for every dataset to maximize the performance of each classifier in this study. The ten classifiers and the information of hyperparameter tuning are described as follows.

- **CART**, *classification and regression tree*, uses the function `rpart` in the `caret` package. The tunable hyperparameters are the `cp` (complexity parameter), values {0.001, 0.01, 0.1, 1}, the `minisplit` (the minimum number of observations that must exist in a node in order for a split to be attempted) parameter with value 10. Moreover, the “best” tree model is chosen using the Min. + 1 SE rule.
- **C5.0**, *C5.0 decision trees and rule-based models*, uses the function `C5.0` in the `caret` package. The tunable hyperparameters are the `trials` (#boosting iterations), values {1, 10, 20}; `models` (model type), values {“tree”, “rules”}; and `winnow` (feature selection), values {TRUE, FALSE}.
- **parRF**, *parallel random forest*, uses the function `parRF` in the `caret` package. The tunable hyperparameter is the `mtry` (#randomly selected predictors), values {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}.
- **AdaBoost**, *adaptive boosting*, uses the function `ada` in the `ada` package.
- **XGBoost**, *extreme gradient boosting*, uses the function `xgbTree` in the `caret` package. The tunable hyperparameters are the `eta` (shrinkage), values {0.1, 0.3, 0.5}; `rounds` (the number of boosting iterations), values {100, 300, 500}; `max_depth` (max tree depth), values {2, 6, 10}; `min_child_weight` (minimum sum of instance weight), values {1, 3, 5}; `colsample_bytree` (subsample ratio of columns), values {0.1, 0.3, 0.5}; `gamma` (minimum loss reduction), values 0; `subsample` (subsample percentage), values 1.
- **avNNet**, *model averaged neural network*, uses the function `avNNet` in the `caret` package. The tunable hyperparameters are the `size` (#hidden units), values {2, 4, 6, 8, 10}; `decay` (weight decay), values {0.00001, 0.0001, 0.001, 0.01}; `bag` (bagging), values {TRUE, FALSE}.
- **NB**, *naïve Bayes*, uses the function `naiveBayes` in the `e1071` package.
- **LR**, *logistic regression*, uses the function `glm` with parameter family was set as “binomial”.
- **LLM**, *logit leaf model*, uses the function `llm` in the `LLM` package. The tunable hyperparameters are the `threshold_pruning` (confidence threshold for pruning), values {0.25, 0.50, 0.75, 1} and the `nbr_obs_leaf` (the minimum number of observations in a leaf node), values {10, 20, 30, 40, 50, 60, 70, 80, 90, 100}.
- **svmPoly**, *support vector machines with polynomial kernel*, uses the function `svmPoly` in the `caret` package. The tunable hyperparameters are the degree (polynomial

degree), values {1, 2, 3}; scale, values {0.001, 0.01, 0.1}; and C (cost), values {0.25, 0.5, 1}.

Performance Metric

Numerous performance metrics have been proposed to evaluate how much a developed model is capable of distinguishing between classes. Those that are widely used include accuracy, precision, recall, F-measure, kappa statistics and AUC-ROC (area under the receiver operating characteristics curve), etc. Different performance metrics are used in different fields. For example, in the studies of rare diseases or customer churn prediction, the accuracy is criticized as an impractical technique because of the imbalance of class distribution [27]. Suppose we have a dataset with 1000 samples, the majority class and the minority class containing 990 samples and 10 samples, respectively. Now if a classifier classifies them all as the majority class, the accuracy will be 99%, even though the classifier missed all minority samples due to the highly imbalanced class distribution [1].

Considering the limitation of metrics like accuracy, macro-averaged F-measure (macro-F) was chosen to measure the predictive performance of classifiers. Given the skewed distribution of the classes, macro-F provides the class average. Indeed, it is widely employed in previous studies and provides a richer measure of classification performance [17, 23, 24]. Consider $L = \{I_i : i = 1, \dots, m\}$ is the set of all labels, macro-F is mathematically defined as (1) referring to Table 2:

$$P = \frac{tp}{tp + fp} \Rightarrow P_{\text{macro}} = \frac{1}{m} \sum_{i=1}^m P_i$$

$$R = \frac{tp}{tp + fn} \Rightarrow R_{\text{macro}} = \frac{1}{m} \sum_{i=1}^m R_i$$

$$F_{\text{macro}}(\beta) = \frac{(1 + \beta^2)P_{\text{macro}}R_{\text{macro}}}{\beta^2 P_{\text{macro}} + R_{\text{macro}}}, \quad (1)$$

Table 2 Cross-table of true condition and predicted condition

| | True condition | |
|---------------------|---------------------|---------------------|
| | Positive | Negative |
| Predicted condition | | |
| Positive | True positive (tp) | False positive (fp) |
| Negative | False negative (fn) | True negative (tn) |

most commonly, β is taken to be 1.

As described above, this study used nine benchmark datasets extracted from different domains to draw a relatively universal conclusion. Additionally, ten representative classifiers arising from seven categories were selected and the hyperparameter tuning was done for every data to maximize the performance of each classifier. Furthermore, to make the result more reliable, macro-F was chosen to measure the predictive performance of classifiers.

Experimental Analysis and Discussion

As explained in “[Datasets, classifiers, and performance metric](#)”, we empirically evaluated the performance of different classifiers (CART, C5.0, parRF, AdaBoost, XGBoost, avN-Net, NB, LR, LLM, svmPoly) with comprehensive hyperparameter tuning. A detailed description of this study is here presented; in particular, the experimental results are illustrated in “[The effect of class imbalance on the performance of classifiers](#)” and “[The effect of training data size on the performance of classifiers](#)”.

The Effect of Class Imbalance on the Performance of Classifiers

In this study, the degrees of class imbalance were set as 1.00, 0.85, 0.70, 0.55, 0.40, 0.25 and 0.10. Because the smallest number of instances in minority class is 160 (from data ozone), the total number of instances were kept 300 to avoid the effect of data size. For each dataset, the numbers of instances in minority class and majority class are 150 and 150 (skew: 1.00), 138 and 162 (skew: 0.85), 124 and 176 (skew: 0.70), 106 and 194 (skew: 0.55), 86 and 214 (skew: 0.40), 60 and 240 (skew: 0.25), 27 and 273 (skew: 0.10), respectively.

The calculation process is shown in Fig. 1, in which M is a certain classifier, class X and class Y are the minority class and majority class, respectively. S is used to make datasets with different skews. For example, in the second round, S is 2, #Min[2] is 138, and #Maj[2] is 162. 138 instances of class X and 162 instances of class Y are collected randomly to work as training data and the remaining data work as test data. After building the learning model, the predictive performance of classifier M on given test data is computed, which is judged by macro-F. Because of the uncertainty of random sampling, the comparison of different classifiers only using one time random sampling is clearly not enough, ten times random sampling were conducted and the averaged results were used for evaluation, which is shown as average (M, 2).

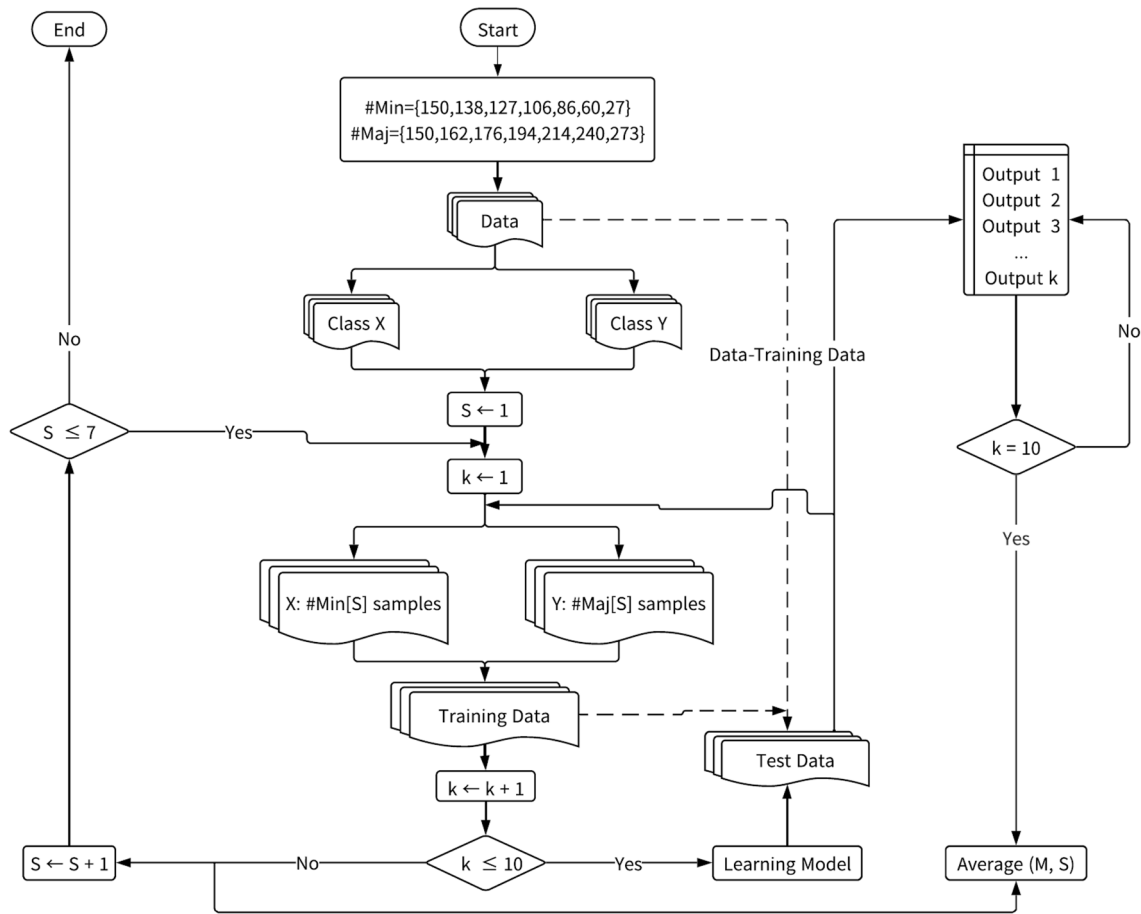


Fig. 1 The calculation process. M is classifier, class X is minority class, class Y is majority class, S is used to make datasets with different skews, and k is the times of random sampling

Predictive Performance and Discussion

First, the result of each dataset with different skews is shown (see Fig. 2), in which the color is proportional to the value of macro-F. The lighter the color, the higher the value. Because different datasets got different degrees of accuracy (e.g., in the case of data cpu, macro-F is around 0.85, while macro-F is about 0.68 for data ozone), the max–min normalization was performed for the result of each dataset to avoid the effect of data characteristics.

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}(M - m) + m, \quad (2)$$

where x' is the standard data, x , x_{\max} , x_{\min} are the original data, the maximum, the minimum, respectively. Normalization keeps x' between M and m . In this study, M is set to 2, and m is 1.

According to Fig. 2, classifiers do show different predictive performances for different datasets. However, something in common can be found: ten classifiers are

separated into two groups, CART, LLM, NB, and LR are one group, whose accuracies are lower; svmPoly, C5.0, avNNet, parRF, AdaBoost and XGBoost are one group, whose accuracies are higher. parRF, AdaBoost and XGBoost get around or greater than 0.80 for all datasets even if the skew decreases to 0.55. In the case of the skew being 0.25, parRF, AdaBoost and XGBoost achieve accuracies as much as 0.98, although it is limited to some certain datasets. Furthermore, the darkest area appears when skew is 0.10, because all classifiers get the lowest accuracies.

Table 3 shows the highest and lowest macro-F in different skews. The accuracy is improved prominently when skew changes from 0.10 to 0.25. Additionally, as we can see, even parRF, which was reported to be weak to handle imbalanced data, is capable of achieving substantial accuracy through hyperparameter tuning. Almost all of the lowest macro-F come from CART and LR. On the other hand, svmPoly, avNNet, parRF and XGBoost are showing superiority.

Although the data characteristics enable different classifiers to perform very well or very badly, as described above,

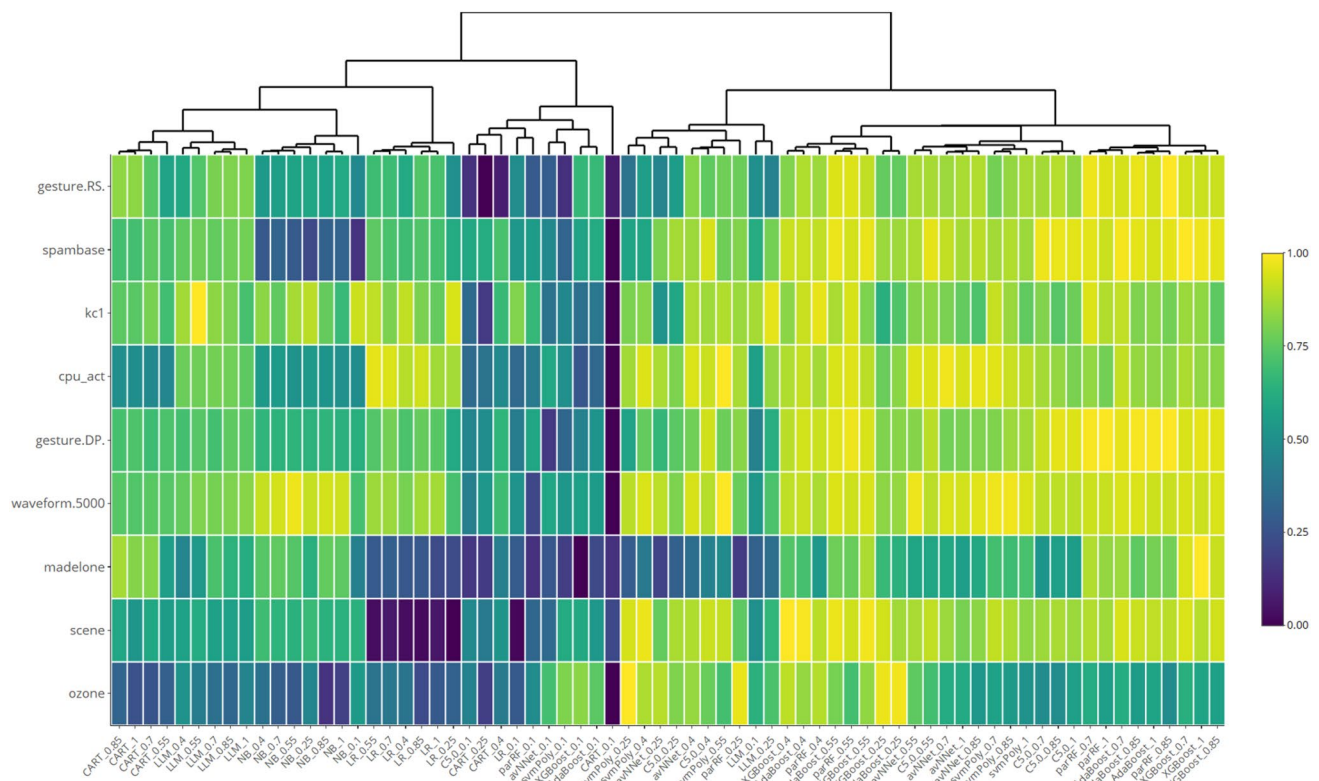


Fig. 2 Heatmap of classifiers with different skews. Euclidean distance is used as distance method and the hierarchical method is Ward.D2

no outlying dataset exists among the nine datasets. Therefore, the average macro-F across nine datasets was taken and is shown in Fig. 3. Ensemble classifiers AdaBoost, XGBoost and parRF are superior. The avNNet, C5.0 and svmPoly are placed on the second-class level of performance. In addition, NB, LR and LLM are less susceptible to class imbalance compared with the other applied classifiers. All of classifiers obtained the lowest macro-F when the skew is 0.10. However, on condition that the skew is larger than 0.5, the effect of class imbalance seems to be eliminated through hyperparameter tuning.

Furthermore, Tukey–Kramer test was performed to find whether there are significant differences between classifiers in the averaged macro-F and the coefficient of variation (CV). Standard deviation represents the degree of variation from the average value. It cannot be compared in simple terms when the average value and the unit of data are different. When comparing the variation of data with different average values and units, CV should be used. The formula is shown below:

$$CV = \frac{\sigma}{\bar{x}} \quad (3)$$

The averaged macro-F was used to evaluate the predictive performance of classifiers and the CV was used to

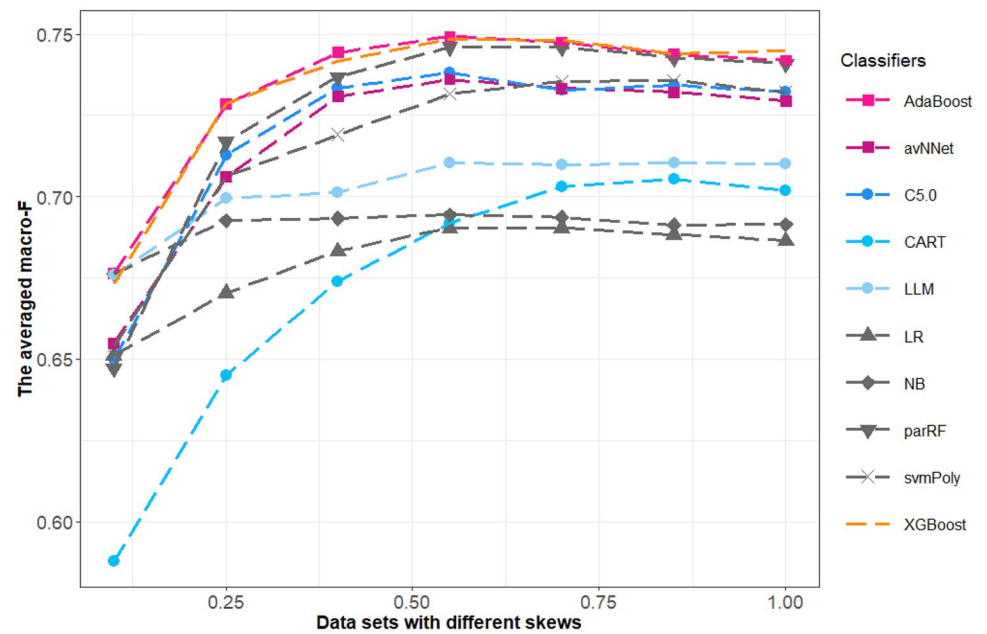
evaluate the stability of classifier in terms of data characteristics (i.e., robustness with respect to changes in the input data).

The results indicated that the predictive performance of CART and LR was significantly inferior to that of the other classifiers at 0.05 level. Because in the case of averaged macro-F, significant differences were detected between CART and AdaBoost ($p = 0.01$), CART and XGBoost ($p = 0.01$), CART and parRF ($p = 0.03$), LR and AdaBoost ($p = 0.02$), LR and XGBoost ($p = 0.02$). Furthermore, according to the results of Tukey–Kramer test on CV, no significant difference exists between CART and the other classifiers. On the other hand, the CV of LR was around 0.20, which was significantly larger than those of AdaBoost ($p = 1e-07$), avNNet ($p = 2.19e-03$), C5.0 ($p = 5.88e-03$), XGBoost ($p = 1e-07$), parRF ($p = 3.60e-06$), CART ($p = 7.33e-05$), LLM ($p = 1.95e-05$), NB ($p = 0$) and svmPoly ($p = 4.77e-03$) at 0.05 level. As we can see, LR exhibits a problematic behavior in terms of predictive accuracy and is more highly dependent on the data characteristics, although the predictive performance of all classifiers is affected to some extent.

Table 3 The highest and lowest macro-F in different skews

| | Skew | Cpu | Gesture (DP) | Gesture (RS) | Kc1 | Madelon | Scene | Spambase | Waveform | Ozone | Mean \pm SD |
|---------------------|------|------------------------|---------------------|---------------------|---------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------|
| The highest macro-F | 1.00 | 0.90 (avNNet) | 0.79 (parRF) | 0.67 (parRF) | 0.65 (parRF) | 0.60 (XGBoost) | 0.74 (Ada-Boost) | 0.92 (XGBoost) | 0.84 (svmPoly) | 0.65 (avNNet) | 0.75 \pm 0.11 |
| | 0.85 | 0.90 (avNNet) | 0.79 (parRF) | 0.68 (parRF) | 0.65 (svmPoly) | 0.59 (XGBoost) | 0.75 (XGBoost) | 0.91 (C5.0) | 0.85 (svmPoly) | 0.64 (XGBoost) | 0.75 \pm 0.11 |
| | 0.70 | 0.90 (avNNet) | 0.79 (parRF) | 0.68 (parRF) | 0.65 (parRF) | 0.59 (XGBoost) | 0.76 (XGBoost) | 0.92 (XGBoost) | 0.85 (svmPoly) | 0.65 (XGBoost) | 0.75 \pm 0.11 |
| | 0.55 | 0.90 (svm-Poly) | 0.78 (parRF) | 0.67 (parRF) | 0.66 (LLM) | 0.58 (XGBoost) | 0.76 (XGBoost) | 0.91 (Ada-Boost) | 0.86 (svmPoly) | 0.67 (XGBoost) | 0.76 \pm 0.11 |
| | 0.40 | 0.90 (svm-Poly) | 0.78 (Ada-Boost) | 0.65 (Ada-Boost) | 0.66 (parRF) | 0.56 (NB) | 0.77 (XGBoost) | 0.91 (C5.0) | 0.84 (svmPoly) | 0.67 (XGBoost) | 0.75 \pm 0.11 |
| The lowest macro-F | 0.25 | 0.89 (avNNet) | 0.76 (parRF) | 0.63 (parRF) | 0.66 (LLM) | 0.55 (XGBoost) | 0.75 (svmPoly) | 0.89 (C5.0) | 0.83 (NB) | 0.68 (svmPoly) | 0.74 \pm 0.11 |
| | 0.10 | 0.86 (avNNet) | 0.73 (NB) | 0.60 (Ada-Boost) | 0.65 (NB) | 0.52 (NB) | 0.68 (NB) | 0.83 (LLM) | 0.76 (NB) | 0.67 (XGBoost) | 0.70 \pm 0.10 |
| | 1.00 | 0.85 (CART) | 0.73 (NB) | 0.59 (NB) | 0.63 (NB) | 0.50 (LR) | 0.53 (LR) | 0.75 (NB) | 0.77 (CART) | 0.61 (NB) | 0.66 \pm 0.11 |
| | 0.85 | 0.85 (CART) | 0.73 (NB) | 0.58 (NB) | 0.63 (avNNet) | 0.50 (LR) | 0.52 (LR) | 0.75 (NB) | 0.77 (CART) | 0.6 (NB) | 0.66 \pm 0.11 |
| | 0.70 | 0.85 (CART) | 0.73 (NB) | 0.58 (NB) | 0.64 (NB) | 0.51 (LR) | 0.53 (LR) | 0.75 (NB) | 0.77 (CART) | 0.61 (CART) | 0.66 \pm 0.11 |
| | 0.55 | 0.85 (CART) | 0.73 (NB) | 0.58 (CART) | 0.63 (CART) | 0.50 (LR) | 0.52 (LR) | 0.74 (NB) | 0.77 (CART) | 0.61 (NB) | 0.66 \pm 0.11 |
| | 0.40 | 0.85 (CART) | 0.73 (CART) | 0.46 (CART) | 0.63 (CART) | 0.50 (LR) | 0.52 (LR) | 0.74 (NB) | 0.76 (CART) | 0.62 (NB) | 0.64 \pm 0.13 |
| | 0.25 | 0.84 (CART) | 0.70 (CART) | 0.44 (CART) | 0.59 (CART) | 0.49 (CART) | 0.51 (LR) | 0.72 (NB) | 0.71 (CART) | 0.61 (CART) | 0.62 \pm 0.12 |
| | 0.10 | 0.80 (CART) | 0.62 (CART) | 0.45 (CART) | 0.57 (CART) | 0.47 (XGBoost) | 0.51 (LR) | 0.67 (CART) | 0.53 (CART) | 0.59 (CART) | 0.58 \pm 0.10 |

The greatest and least macro-F of each data are emphasized in bold

Fig. 3 Averaged macro-F on test data of nine datasets**Table 4** Average classifier ranks across nine datasets at different skews

| | Skew | | | | | | | Mean \pm SD | CV |
|----------|-------------|-------------|-------------|-------------|-------------|-------------|-------------|-----------------|------|
| | 1.00 | 0.85 | 0.70 | 0.55 | 0.40 | 0.25 | 0.10 | | |
| CART | 8.22 | 7.89 | 8.44 | 9.22 | 9.00 | 9.44 | 9.67 | 8.84 \pm 0.62 | 0.07 |
| C5.0 | 5.11 | 5.44 | 5.56 | 5.22 | 4.33 | 5.00 | 6.56 | 5.32 \pm 0.62 | 0.12 |
| parRF | <u>3.33</u> | <u>3.56</u> | <u>3.33</u> | <u>3.00</u> | <u>3.89</u> | <u>4.33</u> | 6.22 | 3.95 \pm 1.01 | 0.26 |
| svmPoly | 4.78 | 3.89 | 4.22 | 4.67 | 5.33 | 5.22 | 5.44 | 4.79 \pm 0.54 | 0.11 |
| AdaBoost | <u>3.00</u> | 2.78 | 2.56 | 2.89 | 2.67 | 3.44 | <u>4.00</u> | 3.05 \pm 0.47 | 0.15 |
| XGBoost | 2.56 | <u>3.56</u> | 2.56 | <u>3.56</u> | <u>3.67</u> | <u>3.78</u> | <u>5.00</u> | 3.52 \pm 0.77 | 0.22 |
| NB | 8.22 | 7.89 | 8.33 | 7.44 | 7.33 | 5.67 | 3.22 | 6.87 \pm 1.70 | 0.25 |
| LR | 7.67 | 7.33 | 7.67 | 7.22 | 7.33 | 6.89 | 6.22 | 7.19 \pm 0.47 | 0.06 |
| LLM | 7.78 | 7.44 | 7.33 | 6.89 | 7.00 | 6.00 | 3.44 | 6.56 \pm 1.37 | 0.21 |
| avNNet | 4.33 | 5.22 | 5.00 | 4.89 | 4.44 | 5.22 | 5.22 | 4.90 \pm 0.35 | 0.07 |

Highest and second-highest macro-F are emphasized in bold and italic, respectively. Values with underlines indicate third-highest macro-F

CV coefficient of variation

Ranks of Classifiers and Discussion

The ranks of classifiers at different skews were computed to evaluate the stability of classifier in terms of class imbalance (i.e., robustness with respect to changes in the class skews). In this study, the overall rank of each classifier was simply obtained by averaging its rank across all datasets. The lower the average rank, the more effective the classifier (see Table 4).

Referring to Table 4 and all the results discussed above, for imbalanced data with skew larger than 0.25, AdaBoost, XGBoost and parRF are recommended. If the skew is about 0.10, AdaBoost and XGBoost are still good

choices. However, the rank of parRF varied a lot when the skew became 0.10, which shows that RF is not preferred if the skew is lower than 0.10. As for svmPoly, another major data analysis tool that has been successfully introduced in various prediction models has been indicated to be less susceptible to class imbalance than parRF (the CV is 0.11). On the other hand, although the CV of CART is comparatively low (0.07), it is because CART keeps ranking at the bottom. Therefore, CART should be the last choice among the ten classifiers. NB, LR and LLM keep going up in the ranking with the skew decrease, which illustrates that NB, LR and LLM are more robust to the changes in class imbalance degree. On the other hand, AdaBoost, XGBoost and parRF exhibited a quite

Table 5 The setting of data size

| Ozone (skew: 0.07) | | | Kc1 (skew: 0.18) | | | Scene (skew: 0.22) | | | |
|---------------------------|------|-----------|---------------------------|------|-----------|-----------------------|------|-----------|------|
| #Min | #Maj | Data size | #Min | #Maj | Data size | #Min | #Maj | Data size | |
| Level 1 | 160 | 2374 | 2534 | 326 | 1783 | 2109 | 431 | 1976 | 2407 |
| Level 2 | 135 | 2003 | 2138 | 301 | 1646 | 1947 | 406 | 1861 | 2267 |
| Level 3 | 110 | 2375 | 2485 | 276 | 1510 | 1786 | 381 | 1747 | 2128 |
| Level 4 | 85 | 1261 | 1346 | 251 | 1373 | 1624 | 356 | 1632 | 1988 |
| Level 5 | 60 | 890 | 950 | 226 | 1236 | 1462 | 331 | 1518 | 1849 |
| Level 6 | 35 | 519 | 554 | 201 | 1099 | 1300 | 306 | 1403 | 1709 |
| Gesture (RS) (skew: 0.37) | | | Cpu (skew: 0.43) | | | Waveform (skew: 0.51) | | | |
| #Min | #Maj | Data size | #Min | #Maj | Data size | #Min | #Maj | Data size | |
| Level 1 | 1087 | 2950 | 4037 | 2477 | 5715 | 8192 | 1692 | 3308 | 5000 |
| Level 2 | 1062 | 2882 | 3944 | 2452 | 5657 | 8109 | 1667 | 3259 | 4926 |
| Level 3 | 1037 | 2814 | 3851 | 2427 | 5600 | 8027 | 1642 | 3210 | 4852 |
| Level 4 | 1012 | 2746 | 3758 | 2402 | 5542 | 7944 | 1617 | 3161 | 4778 |
| Level 5 | 987 | 2679 | 3666 | 2377 | 5484 | 7861 | 1592 | 3112 | 4704 |
| Level 6 | 962 | 2611 | 3573 | 2352 | 5427 | 7779 | 1567 | 3064 | 4631 |
| Spambase (skew: 0.65) | | | Gesture (DP) (skew: 0.77) | | | Madelone (skew: 1.00) | | | |
| #Min | #Maj | Data size | #Min | #Maj | Data size | #Min | #Maj | Data size | |
| Level 1 | 1813 | 2788 | 4601 | 2097 | 2741 | 4838 | 1300 | 1300 | 2600 |
| Level 2 | 1788 | 2750 | 4538 | 2072 | 2708 | 4780 | 1275 | 1275 | 2550 |
| Level 3 | 1763 | 2711 | 4474 | 2047 | 2676 | 4723 | 1250 | 1250 | 2500 |
| Level 4 | 1738 | 2673 | 4411 | 2022 | 2643 | 4665 | 1225 | 1225 | 2450 |
| Level 5 | 1713 | 2634 | 4347 | 1997 | 2610 | 4607 | 1200 | 1200 | 2400 |
| Level 6 | 1688 | 2596 | 4284 | 1972 | 2578 | 4550 | 1175 | 1175 | 2350 |

#Min shows the number of instances in minority class and #Maj shows the number of instances in majority class

poor stability in terms of class imbalance although they achieved superior predictive accuracies, ensuring a satisfactory trade-off between class imbalance stability and predictive accuracy.

The Effect of Training Data Size on the Performance of Classifiers

To avoid the effect of class imbalance, the skew of each data is kept as original, for example, the skew of data ozone is 0.07 and that of data kc1 is 0.18. For starters, all the instances of minority and majority classes were used. Then, the number of instances of minority class decreased 25 each time and the number of majority class was adjusted simultaneously. The setting of data size is shown in Table 5. The level number, from 1 to 6, is given to each data set to make the later explanation of results easier. Furthermore, tenfold cross-validation was performed and the averaged macro-F was used to make the evaluation.

Predictive Performance and Discussion

First, the results of each dataset with different sizes are shown (see Fig. 4). Also, the max–min normalization was performed for the result of each dataset to avoid the effect of data characteristics. According to Fig. 4, the classifiers with data size level are separated into three groups: svmPoly, avNNet, LLM, LR, and NB are one group, whose predictive performance is poor and does not change a lot before the data size level becomes 1; the higher accuracies are achieved by parRF, C5.0, AdaBoost, XGBoost; for almost all data sets, classifiers achieve the highest accuracy when the level is 1, i.e., the data size is biggest.

Table 6 shows the highest and lowest macro-F in different data size levels. The accuracy is improved prominently when data size is the biggest. Almost all of the lowest macro-F come from NB, CART and LR. On the other hand, XGBoost shows superiority in achieving higher classification accuracy and being less affected by data size.

Furthermore, the averaged macro-F across nine datasets was taken and is shown in Fig. 5. The performance of all classifiers degrades in a very dramatic way when the data size level changes from 1 to 2. No one classifier is shown to be robust to the change of data size, because all of them

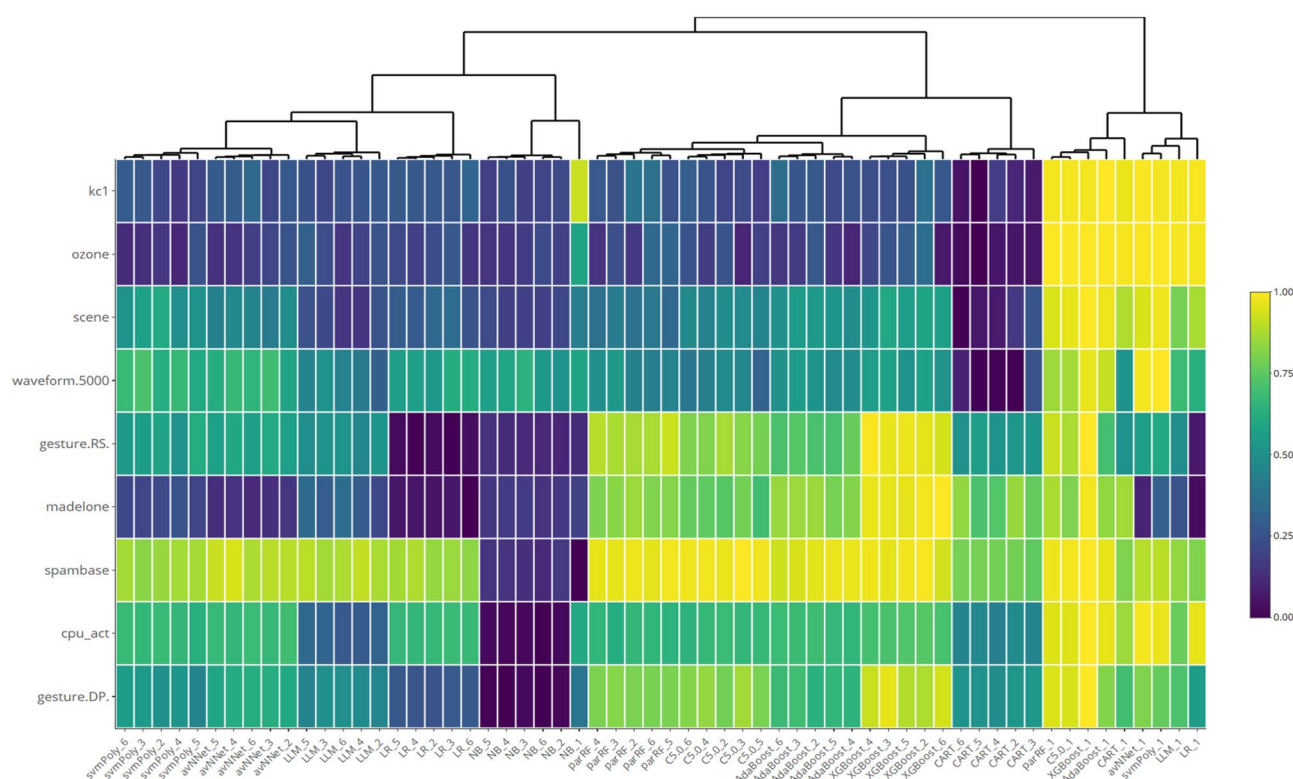


Fig. 4 Heatmap of classifiers with different data sizes. The number showed after classifier is the number of levels. Euclidean distance is used as distance method and the hierarchical method is Ward.D2

have similar trends. However, in the case of predictive accuracy, XGBoost, parRF, AdaBoost and C5.0 shows the superior performance. The avNNet and svmPoly are placed on the second-class level of performance. In addition, NB, LR, CART and LLM are on the third-class level. Compared to the result described above, C5.0 is highly rated. Referring to this, it is considered that C5.0 is less robust to the imbalanced data than XGBoost, parRF and AdaBoost, thus when the skew became lower, the performance of C5.0 is degraded.

Ranks of Classifiers and Discussion

Table 7 shows the average classifier ranks across nine datasets at different levels of data size. The CV of CART is 0.03, which is the lowest. However, stable but not accurate solutions would not be meaningful. On the contrary, XGBoost gets clearly lower average rank, but relatively higher CV (0.27). As we can see, the average rank of XGBoost is significantly lower in the case of the data size level is 1 or

2, which illustrates that XGBoost has a relatively poorer robustness to data size, because it takes significant advantage of data growth to tune a more accurate model. AdaBoost has a behavior quite similar to XGBoost, but turns out to be more stable. The parRF, another superior classifier we mentioned above, turns out to be good at classification and more robust than most of the classifiers applied here. However, considering the methodology of parRF, a lot of training data are intrinsically required. Such a result was obtained because XGBoost and AdaBoost seem to be able to better learn with large-scale data and the superior algorithm enables parRF to perform better than other classifiers even not though there are enough training data at hand. CART, NB, LR and LLM are still not recommended.

Conclusions and Future Work

Previous work [3, 4] found that RF performs well in handling datasets with large class imbalance. However, the earlier studies used a large amount of training data, which

Table 6 The highest and lowest macro-F in different level of data size

| Level | Skew | Cpu | Gesture (DP) | Gesture (RS) | Kc1 | Madelon | Scene | Spambase | Waveform | Mean \pm SD |
|---------------------|-----------------------|-----------------------|-----------------------|------------------------|------------------|-----------------------|-----------------------|-----------------------|-----------------------|-----------------|
| The highest macro-F | | | | | | | | | | |
| 1 | 0.96 (XGBoost) | 0.89 (XGBoost) | 0.80 (XGBoost) | 0.93 (AdaBoost) | 0.84 (XGBoost) | 0.94 (XGBoost) | 0.96 (XGBoost) | 0.93 (svmPoly) | 0.97 (XGBoost) | 0.91 \pm 0.06 |
| 2 | 0.93 (XGBoost) | 0.87 (XGBoost) | 0.79 (XGBoost) | 0.71 (parRF) | 0.84 (XGBoost) | 0.82 (svmPoly) | 0.95 (XGBoost) | 0.89 (svmPoly) | 0.71 (XGBoost) | 0.83 \pm 0.08 |
| 3 | 0.93 (XGBoost) | 0.89 (XGBoost) | 0.79 (XGBoost) | 0.67 (svmPoly) | 0.84 (XGBoost) | 0.82 (XGBoost) | 0.95 (C5.0) | 0.90 (svmPoly) | 0.68 (LR) | 0.83 \pm 0.10 |
| 4 | 0.93 (XGBoost) | 0.88 (XGBoost) | 0.79 (XGBoost) | 0.68 (avNNet) | 0.84 (XGBoost) | 0.80 (XGBoost) | 0.95 (C5.0) | 0.89 (avNNet) | 0.66 (XGBoost) | 0.82 \pm 0.10 |
| 5 | 0.93 (XGBoost) | 0.87 (XGBoost) | 0.79 (XGBoost) | 0.67 (avNNet) | 0.83 (XGBoost) | 0.81 (XGBoost) | 0.95 (parRF) | 0.89 (avNNet) | 0.70 (parRF) | 0.83 \pm 0.09 |
| 6 | 0.93 (NB) | 0.88 (NB) | 0.78 (LR) | 0.70 (CART) | 0.85 (LR) | 0.81 (CART) | 0.95 (NB) | 0.89 (CART) | 0.70 (CART) | 0.83 \pm 0.09 |
| The lowest macro-F | | | | | | | | | | |
| 1 | 0.92 (NB) | 0.79 (NB) | 0.57 (LR) | 0.90 (NB) | 0.55 (LR) | 0.76 (NB) | 0.76 (NB) | 0.88 (CART) | 0.80 (NB) | 0.77 \pm 0.13 |
| 2 | 0.86 (XGBoost) | 0.73 (XGBoost) | 0.56 (XGBoost) | 0.61 (parRF) | 0.55 (XGBoost) | 0.68 (svmPoly) | 0.79 (XGBoost) | 0.82 (svmPoly) | 0.59 (XGBoost) | 0.69 \pm 0.11 |
| 3 | 0.86 (NB) | 0.73 (NB) | 0.56 (LR) | 0.60 (CART) | 0.55 (LR) | 0.69 (NB) | 0.79 (NB) | 0.85 (CART) | 0.59 (CART) | 0.69 \pm 0.12 |
| 4 | 0.86 (NB) | 0.73 (NB) | 0.56 (LR) | 0.63 (svmPoly) | 0.55 (LR) | 0.65 (CART) | 0.79 (NB) | 0.82 (CART) | 0.59 (CART) | 0.69 \pm 0.11 |
| 5 | 0.86 (NB) | 0.73 (NB) | 0.56 (LR) | 0.57 (CART) | 0.56 (LR) | 0.65 (CART) | 0.79 (NB) | 0.82 (CART) | 0.57 (CART) | 0.68 \pm 0.12 |
| 6 | 0.86 (NB) | 0.73 (NB) | 0.57 (LR) | 0.59 (CART) | 0.54 (LR) | 0.63 (CART) | 0.78 (NB) | 0.83 (CART) | 0.58 (CART) | 0.68 \pm 0.12 |

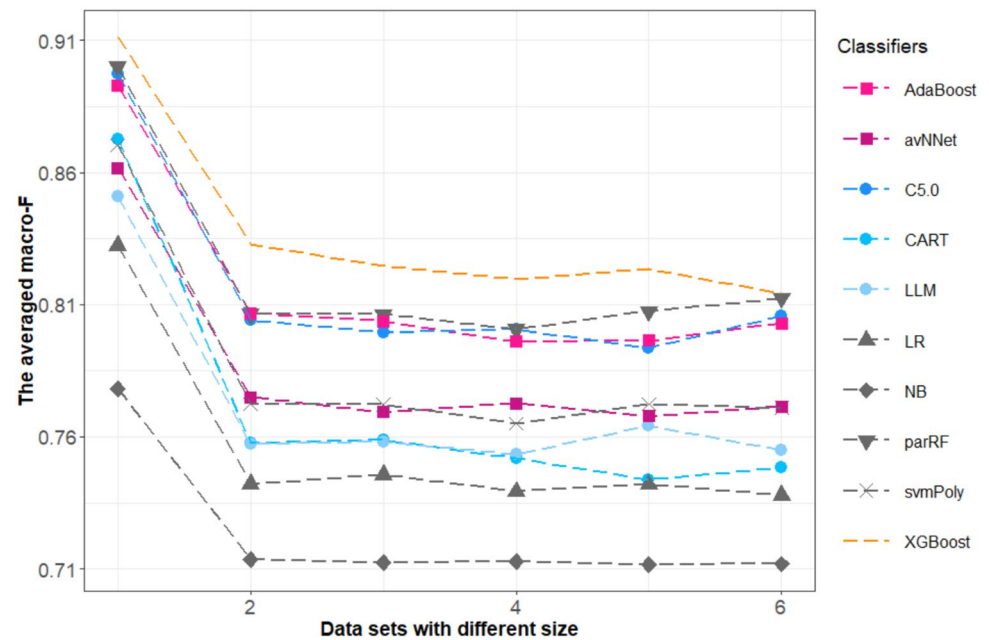
Greatest and least macro-F are emphasized in bold for each data

improves the performance of RF to a certain extent. In this comparative study, we clarified the effects of class imbalance and training data size on the predictive performance of classifiers. Ten frequently employed classifiers were selected and thorough hyperparameter tuning was performed. The predictive accuracy (macro-averaged F-measure) and the rank of classifiers were studied, and the results are summarized as follows.

About the effects of class imbalance on the performance of classifiers: (1) NB, LR and LLM are less susceptible to class imbalance while they have relatively poor predictive performance; (2) ensemble classifiers AdaBoost, XGBoost and parRF have a quite poor stability in terms of class imbalance while they achieved superior predictive accuracies; (3) avNNet, C5.0 and svmPoly are placed on the second-class level of performance; (4) CART is the last choice among the ten classifiers; (5) for all of the classifiers employed in this study, their accuracies decreased as soon as the class imbalance reached a certain point 0.10. Note that although using datasets with balanced class distribution would be an ideal condition to maximize the performance of classifiers. In the case that the skew is larger than 0.10, a comprehensive hyperparameter tuning may be able to eliminate the effect of imbalance.

About the effects of training data size on the performance of classifier: (1) no one classifier has been shown to be robust to the change of data size; (2) in the case of predictive accuracy, XGBoost, parRF, AdaBoost and C5.0 show the superior performance, especially with more training data; (3) the avNNet and svmPoly are on the second-class level of performance; (4) NB, LR, CART and LLM are on the third-class level.

Numerous researches have reported the challenge of class imbalance and small data size for classification algorithm. Generally, highly skewed training data results in biased learning model that has a poor predictive accuracy over the minority classes and underfitting often happens with too little training data, which means that the model was unable to capture the underlying pattern. However, according to the results of this study, ensemble classifiers, such as XGBoost, AdaBoost, parRF, are the superior algorithms that enable to achieve favorable accuracy from time to time even if training data are skewed or not enough. Ensemble classifiers combine several base learners into one predictive model to decrease variance and bias simultaneously, which has been improved to produce more accurate solutions than a single model would in a number of previous researches and machine learning competitions.

Fig. 5 Averaged macro-F on test data of nine datasets**Table 7** Average classifier ranks across nine datasets at different levels of data size

| | Level of data size | | | | | | Mean \pm SD | CV |
|----------|--------------------|-------------|-------------|-------------|-------------|-------------|-----------------|------|
| | 1 | 2 | 3 | 4 | 5 | 6 | | |
| CART | 7.67 | 8.22 | 8.22 | 8.22 | 8.22 | 8.33 | 8.15 ± 0.22 | 0.03 |
| C5.0 | 4.22 | 4.56 | 4.89 | <u>3.78</u> | 5.00 | 4.22 | 4.45 ± 0.42 | 0.09 |
| parRF | 4.11 | <u>4.33</u> | 4.00 | 4.56 | <u>3.44</u> | <u>3.67</u> | 4.02 ± 0.38 | 0.09 |
| svmPoly | <u>4.00</u> | 5.44 | 4.89 | 6.11 | 5.44 | 5.22 | 5.18 ± 0.64 | 0.12 |
| AdaBoost | 3.78 | <u>3.33</u> | <u>3.89</u> | 4.89 | <u>4.56</u> | 3.56 | 4.00 ± 0.55 | 0.14 |
| XGBoost | 1.56 | 1.44 | 2.22 | 2.33 | 2.22 | 3.22 | 2.17 ± 0.58 | 0.27 |
| NB | 9.56 | 8.44 | 8.33 | 7.78 | 8.33 | 7.89 | 8.39 ± 0.58 | 0.07 |
| LR | 7.67 | 7.33 | 5.89 | 6.11 | 6.56 | 6.89 | 6.74 ± 0.63 | 0.09 |
| LLM | 7.56 | 7.22 | 7.22 | 6.78 | 6.44 | 7.44 | 7.11 ± 0.39 | 0.05 |
| avNNet | 4.89 | 4.67 | 5.44 | <u>4.44</u> | 4.78 | 4.56 | 4.80 ± 0.32 | 0.07 |

Highest and second-highest macro-F are emphasized in bold and italic, respectively. Values with underlines indicate third-highest macro-F

CV coefficient of variation

As a limitation of this study, the largest number of variables and instances are 500 and 5000, which could be further increased. In future work, we will use bigger and more datasets to verify and optimize our findings here.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Ali S, Smith KA. On learning algorithm selection for classification. Appl Soft Comput. 2006;6(2):119–38.
2. Błaszczyński J, Stefanowski J. Local data characteristics in learning classifiers from imbalanced data. In: Gawęda A, Kacprzyk J, Rutkowski L, Yen G, editors. Advances in data analysis with computational intelligence methods: studies in computational intelligence, vol. 738. Cham: Springer; 2017. p. 51–85.
3. Brown I, Mues C. An experimental comparison of classification algorithms for imbalanced credit scoring data sets. Expert Syst Appl. 2012;39(3):3446–533.
4. Caigny AD, Coussement K, De Bock KW. A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees. Eur J Oper Res. 2018;269(2):760–72.

5. D'souza RN, Huang PY, Yeh FC. Small data challenge: structural analysis and optimization of convolutional neural networks with a small sample size. *bioRxiv*. 2018. <https://doi.org/10.1101/402610>.
6. Foody GM, Mathur A. A relative evaluation of multiclass image classification by support vector machine. *IEEE Trans Geosci Remote Sens*. 2004;42(6):1335–433.
7. Fernández-Delgado M, Cernadas E, Barro S. Do we need hundreds of classifiers to solve real world classification problems? *J Mach Learn Res*. 2014;15:3133–81.
8. García V, Marqués AI, Sánchez JS. Exploring the synergetic effects of sample types on the performance of ensembles for credit risk and corporate bankruptcy prediction. *Inform Fus*. 2019;47:88–101.
9. Georgakis G, Mousavian A, Berg AC, Kosecka J. Synthesizing training data for object detection in indoor scenes. 2017; arXiv:1702.07836. <https://arxiv.org/pdf/1702.07836.pdf>. Accessed 8 Sept 2017.
10. Halevy A, Norvig P, Pereira F. The unreasonable effectiveness of data. *IEEE Intell Syst*. 2009;24(2):1541–672.
11. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016; pp. 770–778.
12. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*. 2012; pp. 1097–1105.
13. Mathur A, Foody GM. Crop classification by a support vector machine with intelligently selected training data for an operational application. *Int J Remote Sens*. 2008;29(8):2227–40.
14. Nguyen T, özaslan T, Miller ID, Keller J, Loianno G, Taylor CJ, Lee DD, Kumar V, Harwood JH, Wozencraft J. U-Net for MAV-based penstock inspection: an investigation of focal loss in multi-class segmentation for corrosion identification. 2018; arXiv:1809.06576. <https://arxiv.org/pdf/1809.06576.pdf>. Accessed 11 Nov 2018.
15. Pal M, Mather PM. An assessment of the effectiveness of decision tree methods for land cover classification. *Remote Sens Environ*. 2003;86(4):554–65.
16. Rothe S, Kudsus B, Söfker D. Does classifier fusion improve the overall performance? Numerical analysis of data and fusion method characteristics influencing classifier fusion performance. *Entropy*. 2019;21(9):866. <https://doi.org/10.3390/e21090866>.
17. Rizwan M, Nadeem A, Sindhu M. Analyses of classifier's performance measures used in software fault prediction studies. *Digit Object Identif*. 2019;7:82764–75.
18. Sun MX, Liu KH, Wu QQ, Hong QQ, Wang BZ, Zhang HY. A novel ECOC algorithm for multiclass microarray data classification based on data complexity analysis. *Pattern Recogn*. 2019;90:346–62.
19. Sun YM, Wong AKC, Kamel MS. Classification of imbalanced data: a review. *Int J Pattern Recogn Artif Intell*. 2009;24(4):687–719.
20. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. 2014; arXiv:1409.1556. <https://arxiv.org/pdf/1409.1556.pdf>. Accessed 10 Apr 2015.
21. Szegedy C, Liu W, Jia YQ, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A. Going deeper with convolutions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015; pp. 1–9.
22. Sánchez JS, Molineda RA, Sotoca KM. An analysis of how training data complexity affects the nearest neighbor classifiers. *Pattern Anal Appl*. 2007;10:189–201.
23. Sokolova M, Lapalme G. A systematic analysis of performance measures for classification tasks. *Inf Process Manag*. 2009;45:427–37.
24. Santiso S, Pérez A, Casillas A. Smoothing dense spaces for improved relation extraction between drugs and adverse reactions. *Inform J Med Inform*. 2009;128:39–45.
25. Wainberg M, Alipanahi B, Frey BJ. Are random forests truly the best classifiers? *J Mach Learn Res*. 2016;17:1–5.
26. Wolpert DH, Macready WG. No free lunch theorem for search. Technical Report SFI-TR-05-010, Santa Fe Institute, Santa Fe, NM; 1995.
27. Weiss GM, Provost F. The effect of class distribution on classifier learning. Technical Report ML-TR-43, Department of Computer Science, Rutgers University; 2001. <https://pdfs.semanticscholar.org/45ca/1d5528a4e5beb5616c1ec822901be2de1d59.pdf>. Accessed 2 Aug 2001.
28. Zhu X, Vondrick C, Fowlkes C, Ramanan D. Do we need more training data? *Int J Comput Vision*. 2016;19(1):76–92.
29. Zhu XF, Huang Z, Yang Y, Shen H, Xu CH, Luo JB. Self-taught dimensionality reduction on the high-dimensional small-sized data. *Pattern Recogn*. 2013;46(1):215–29.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.