Expert Systems With Applications 188 (2022) 116015

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

When is resampling beneficial for feature selection with imbalanced wide data?

Ismael Ramos-Pérez *, Álvar Arnaiz-González, Juan J. Rodríguez, César García-Osorio

Universidad de Burgos, Department of Computer Engineering, Escuela Politécnica Superior, Avda. Cantabria s/n, Province of Burgos, 09006, Spain

ARTICLE INFO

Keywords:
Feature selection
Wide data
High dimensional data
Very low sample size
Unbalanced
Machine learning

ABSTRACT

This paper studies the effects that combinations of balancing and feature selection techniques have on wide data (many more attributes than instances) when different classifiers are used. For this, an extensive study is done using 14 datasets, 3 balancing strategies, and 7 feature selection algorithms. The evaluation is carried out using 5 classification algorithms, analyzing the results for different percentages of selected features, and establishing the statistical significance using Bayesian tests.

Some general conclusions of the study are that it is better to use RUS before the feature selection, while ROS and SMOTE offer better results when applied afterwards. Additionally, specific results are also obtained depending on the classifier used, for example, for Gaussian SVM the best performance is obtained when the feature selection is done with SVM-RFE before balancing the data with RUS.

1. Introduction

The term "wide data" has been used to refer to datasets characterized by a high number of features and a low number of instances (Kuncheva et al., 2020), which severely impairs the smooth performance of learning algorithms. We have not been able to find in the literature a proper definition of the term "wide data". Although strictly speaking a dataset could be considered wide when its number of features (#Features) is just greater than its number of examples (#Examples), #Features > #Examples, we believe that for a dataset to deserve to be called wide, this difference must be substantial, #Features ≫ #Examples, for example, of at least one order of magnitude. For reference, in the datasets used in the experimental part of this article, that difference is even greater, with a #Features/#Examples ratio that is greater than 20 for all datasets, and with at least 2000 features.

Several real-world datasets suffer from these problems, especially biological and genomics datasets, discussed in Liu et al. (2020), where data from electroencephalography analyses were used for epilepsy diagnosis, analysis in early detection of type 2 diabetes (Bernardini et al., 2020), and the prediction of mortality among patients admitted to an intensive care unit (Vidya et al., 2019). However, this type of data also arises in other areas such as fault detection in engineering: for example, the diagnosis of engine system faults from measurements taken from the bearing assembly (Hang et al., 2019), the detection of induction motor failures (Alshorman et al., 2020; Juez-Gil et al., 2020), and in solar radiation estimation (Karasu & Altan, 2019), among others. It also appears in computer security environments, for intrusion detection in network environments (Yang et al., 2019). The presence of a large number of features (high dimensionality) decreases the efficiency of learning algorithms and increases their execution time (Maldonado et al., 2014).

As it is well known, the aim of feature selection (FS) algorithms is to find the optimal combination of features that will help to create models that are simpler, faster, and easier to interpret. However, this task is not easy and is, in fact, an NP-hard problem (Guyon et al., 2006). In addition, this type of data is known as unbalanced data when the number of instances belonging to each class is very different between classes (Fernández et al., 2018). When dealing with unbalanced datasets, even if the classifier achieves high global accuracy, it is often the case that the identification of the instances belonging to the minority class is not highly precise. Classifiers work well with instances of the majority classes, though these instances are usually the least interesting, which means that the classifier is largely useless. Some of the most popular algorithms that try to solve this problem are based on oversampling techniques (Abdi & Hashemi, 2016) that increase the number of instances of minority classes and undersampling techniques (Ng et al., 2015) that decrease the number of instances of the majority classes. And there are more recent proposals that use ensembles to combat imbalance (Díez-Pastor et al., 2015), or deal with this problem in the context of big data (Juez-Gil et al., 2021).

* Corresponding author.
E-mail addresses: ismaelrp@ubu.es (I. Ramos-Pérez), alvarag@ubu.es (Á. Arnaiz-González), jjrodriguez@ubu.es (J.J. Rodríguez), cgosorio@ubu.es (C. García-Osorio).

https://doi.org/10.1016/j.eswa.2021.116015
Received 30 April 2021; Received in revised form 30 July 2021; Accepted 30 September 2021
Available online 15 October 2021

The objective of this study is to conduct a series of experiments to assess whether balancing techniques improve classification performance when they are used in conjunction with feature selection on wide data. For this purpose, we use the most common algorithms at each of these stages. Moreover, we are not only interested in finding the more suitable balancing algorithm, but we also seek to determine the most appropriate moment to use it: either before or after feature selection. There is too little literature comparing the performance of these two strategies (Pes, 2020; Zhang et al., 2017), the main novelties of the present paper are:

- The use of a wider variety of algorithms.
- The use of a larger number of datasets.
- The evaluation of the results for different percentages of selected features, avoiding the bias of using a fixed percentage, and hence providing a more complete overview of the problem.
- The performance comparison of different combinations classifier-balancing method.

The *R* code used for the feature selection and to create the figures can be found on *GitHub*.[1]

The rest of the paper will be organized as follows. In Section 2, the background on the different approaches of the feature selection methods will be given. In Section 3, the same will be done for the techniques dealing with unbalanced problems. Information will be provided in Section 4 on the experimental setup and the results will be presented and analyzed in Section 5. The main insights of the study are discussed in Section 6. Finally, the conclusions and future work will be presented in Section 7.

2. Feature selection

In data science, it is often very important to know which features of a dataset are the most relevant for training learning algorithms (Saeys et al., 2007). The use of certain features may not only make no contribution to the improvement of the learning algorithm, but their use might even worsen its performance. This reason explains why FS algorithms are used to find the subset of features that improves the performance of the models obtained using machine learning algorithms. Moreover, FS algorithms also prevent the learning algorithm from overfitting and speeds up its training. In addition, knowledge of which the selected features actually are can provide useful insight into the datasets.

This problem is even more relevant when dealing with wide data, where the number of features is extremely high. At the same time, this technique is widely used for big data (Peralta et al., 2015) where reducing both data size and execution times are paramount.

Some taxonomies (Saeys et al., 2007; Zhu et al., 2007) can be found in the literature on the different FS algorithms, the most widely used of which and, from our point of view also the most convenient, classify the features by their relationship with the learning algorithm. This taxonomy is shown in Fig. 1, in which we can find three main types: **filters**, **wrappers** and **embedded** methods (or nested subset methods):

- **Filters** (Bommert et al., 2020) are used to calculate the relevance of each feature, mainly based on its statistical properties, providing a numerical score for each feature that depends on its contribution to the performance of the algorithm, also called importance.
  Since the operation of these methods will not depend on the use of any particular classifier, it means that the feature sets can be used with any classifier. This approach will reduce overfitting, but it cannot guarantee the best performance, unlike other FS algorithms.
  It should be noted that since this type of algorithms cannot determine the optimal number of features, which would provide the best performance. That number is an additional parameter that must be set to select the subset of relevant features.
  An advantage of filter methods is that they are only executed once before the training, avoiding having to adjust a lot of hyperparameters, making the training faster and more scalable. This feature makes them specially suitable for big-data problems.
  Two kinds of filters may be identified: univariate and multivariate. While the univariate FS algorithms find dependencies between each feature and the output, the multivariate ones try to find dependencies between the features, unfortunately, this is at the cost of more processing time, which makes them less scalable.
- ***Wrapper*** methods (Kohavi & John, 1997a) search throughout the entire space of feature subsets looking for the subset that provides the best performance to the specific learning algorithm given as parameter. They usually offer better performance than the filter methods, however the risk of overfitting is higher. They are also slower and less scalable, because the learning algorithm has to be executed every time a subset of features needs to be evaluated.
- **Embedded** methods (Hamed et al., 2014; Xiao et al., 2008) take advantage of the inner properties of certain learning algorithms, in order to discover the most relevant features in the dataset, as is the case with Random Forest.
  Unlike wrapper methods, the classifier on which the embedded methods are based is not necessarily the same as the one used to classify. These methods have lower risks of overfitting and are faster than the wrapper-based methods, although they are still slower than the filter-based methods.

The strategies of the previous methods can be combined to obtain new algorithms, the two combination approaches are:

- **Hybrid algorithms** take advantage of filter and wrapper methods, sequentially combining their outputs. The output of the filter is given as the input to the wrapper, which reduces the wrapper computation time, by making an initial selection using the filter method and exploiting the efficiency that the wrapper can obtain. Although the sequence of filter-wrapper is generally used, different combinations can also be found (Sahu et al., 2018).
- **Ensemble algorithms** (Bolón-Canedo & Alonso-Betanzos, 2018) combine the output of several individual methods to improve the results that would have been obtained from using each of them in isolation.
  In the same way as ensemble classifiers, ensembles for feature selection can be classified into homogeneous (those that use the same FS method) and heterogeneous (that use different FS methods). The latter are the most widely used.

For this study, we focused on the most cited FS algorithms in the state of the art. To facilitate the comparison of methods, in the experiments we only included those that return a ranking, among which we can find filters (T-test, ANOVA, Information gain, Chi squared, and ReliefF) and embedded methods (Random Forest importance and SVM Recursive Feature Elimination):

- T-test (Peck & Devore, 2011) is a popular statistical test that may be used for an individual evaluation of feature relevance. It computes the ratio between the differences of two class means and the variability between them.
- ANOVA (Johnson & Synovec, 2002), an acronym that stands for "ANalysis Of VAriance", is a simple and well-known method that can be used for feature selection. It works in a similar way to the previous method, testing the differences of means between groups.

[1] https://github.com/Ismael-rp/feature_selection_wide_data.

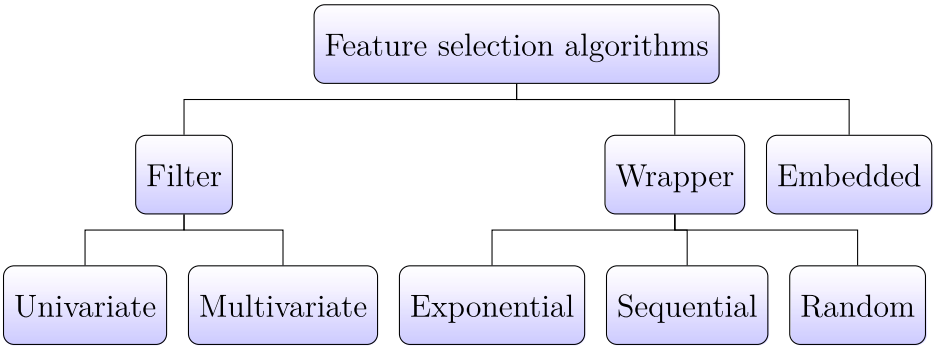


Fig. 1. Taxonomy of feature selection algorithms (Saeys et al., 2007).

- Chi squared (Chi-S) is a feature selection algorithm proposed by Liu and Setiono (1995) that uses the χ^2 statistic and works in two phases. The first phase uses the ChiMerge of Kerber (1992), automatically increasing the χ^2 threshold. The second phase attempts to merge the features, by using the values computed in the previous phase; if two features can be merged, it means that one of them is irrelevant and can be discarded.
- Information gain (Info-Gain) is a measure commonly used in the construction of decision trees for finding the most informative feature to use for splitting each node. The use of information gain for feature selection involves an evaluation of the information gain of each feature with respect to the target feature. More specifically, information gain measures the expected reduction in entropy (Mitchell, 1997).
- Random Forest importance (RF-Imp) determines the usefulness of each feature by measuring the performance difference of the out-of-bag data when noise is added, as proposed by Breiman (2001).
- ReliefF (Kononenko, 1994) is an extended version of the original Relief feature selector of Kira and Rendell (1992). Based on instance-based learning (kNN is used for searching similar instances), it estimates the importance of a feature in relation to other features, and it is non-parametric, that is, no assumption of any distribution is made (Urbanowicz et al., 2018).
- SVM Recursive Feature Elimination (SVM-RFE) (Guyon et al., 2002) is an iterative process that recursively removes features according to the feature weights in a support vector machine classifier (SVM). It is an example of backward feature elimination (Kohavi & John, 1997b) where the elimination is recursive and the classifier used is the popular SVM.

3. Unbalanced data

A dataset is said to be unbalanced when it has a very different number of instances for each class, which affects negatively the classifiers performance, is commonly measured using the imbalance ratio (IR) (Luque et al., 2019), it is even more common in wide data due to the low number of instances.

A straightforward solution for dealing with unbalanced data is to resample the original data set by adjusting the balance ratio as desired. On the one hand, undersampling methods eliminate instances corresponding to the majority class (Ng et al., 2015); on the other hand, oversampling methods create artificial instances for the minority class (Abdi & Hashemi, 2016), and hybrid methods combine both approaches.

The resampling methods used in this study are described below:

- *Random undersampling* (RUS) (Japkowicz, 2000) algorithm randomly selects instances from the majority class (*i.e.*, it removes instances of the majority class).
- *Random oversampling* (ROS) (Japkowicz, 2000) algorithm duplicates instances from the minority class.
- *Synthetic Minority Over-sampling Technique* (SMOTE) (Chawla et al., 2002) algorithm creates new instances, interpolating two instances of the original data set, the first one chosen randomly and the second one chosen randomly from among its k closest neighbors where k is a predefined parameter.

4. Experimental setup

Evaluating the performance of an FS algorithm is not as simple as with a common classifier, where the evaluation is simply based on the number and the type of correctly classified instances.

We aim to evaluate how good an FS algorithm is at selecting the features. These algorithms will attach greater weight to the more relevant features and less weight to the less relevant ones. Using an artificial dataset, we would be able to compare the real weight of each feature with the one provided by the FS. However, as with the case presented here where real datasets are used, it is common to evaluate the selected features according to the classifier performance (Bolón-Canedo & Alonso-Betanzos, 2018). In this section, the steps followed during the experiments are explained, in order to evaluate the performance of the combination of feature selection and balancing on imbalanced wide data.

4.1. Datasets

The 14 high dimensional unbalanced datasets used in this study are summarized in Table 1. For each dataset, the number of examples, the number of features, the relation between the number of examples and features, the class names, the percentage of examples for each class, the class imbalance ratio (IR), and the reference are shown. All the dataset features were numeric and the original multi-class labels were grouped into a new one, in order to obtain new two-class unbalanced datasets. The dataset name indicates which classes were used (where rem represents the combination of the remaining classes).

4.2. Cross validation

The experiments were performed using 5 × 2-fold cross validation. Having been randomly divided into 2 parts, both parts were used for training and testing and the same step was repeated 5 times. This kind of cross validation is very useful when processing datasets with classes that have a very low number of instances, since using 10-fold cross validation will leave a very low number (or even no one) of instances belonging to the minority class in the test (Dietterich, 1998).

Moreover, all the datasets were normalized for transforming all the features: the mean to 0 and the standard deviation to 1.

Table 1
Datasets used in the experimental study. Datasets 1–9 were previously used in Zhu et al. (2007). Datasets 10–14 were used in Li et al. (2018).

	Dataset	#Ex.	#Feat.	$\frac{\#Feat.}{\#Ex.}$	Class (min.; max.)	%min.; %max.	IR
1	Colon[a]	62	2000	32.26	(Normal; Tumor)	0.35; 0.65	1.82
2	MLL_ALL[a]	72	12582	174.75	(ALL; rem)	0.33; 0.67	2.00
3	MLL_AML[a]	72	12582	174.75	(AML; rem)	0.39; 0.61	1.56
4	MLL_MLL[a]	72	12582	174.75	(MLL; rem)	0.28; 0.72	2.60
5	SRBCT_1[a]	83	2308	27.81	(1; rem)	0.35; 0.65	1.86
6	SRBCT_4[a]	83	2308	27.81	(4; rem)	0.30; 0.70	2.32
7	Lung_1[a]	203	12600	62.07	(rem; 1)	0.32; 0.68	2.17
8	Lung_4[a]	203	12600	62.07	(rem; 4)	0.10; 0.90	8.67
9	Lung_5[a]	203	12600	62.07	(rem; 5)	0.10; 0.90	9.15
10	Leukemia_BM[b]	72	7130	99.03	(BM; rem)	0.29; 0.71	2.43
11	TOX_171_1[b]	171	5748	33.61	(1; rem)	0.26; 0.74	2.80
12	TOX_171_2[b]	171	5748	33.61	(2; rem)	0.26; 0.74	2.80
13	TOX_171_3[b]	171	5748	33.61	(3; rem)	0.23; 0.77	3.38
14	TOX_171_4[b]	171	5748	33.61	(4; rem)	0.25; 0.75	3.07

[a] https://jundongl.github.io/scikit-feature/datasets.html.
[b] http://csse.szu.edu.cn/staff/zhuzx/Datasets.html.

Table 2
Resampling algorithms used in this study with their parameters and the R packages used: class (https://cran.r-project.org/web/packages/class/index.html), e1071 (https://cran.r-project.org/web/packages/e1071/index.html), RWeka (https://cran.r-project.org/web/packages/RWeka/index.html), randomForest (https://cran.r-project.org/web/packages/randomForest/randomForest.pdf), naivebayes (https://cran.r-project.org/web/packages/naivebayes/index.html), sigFeature (https://www.bioconductor.org/packages/release/bioc/html/sigFeature.html), mlr3filters (https://cran.r-project.org/web/packages/mlr3filters/index.html), FSelector (https://cran.r-project.org/web/packages/FSelector/index.html), and unbalanced (https://cran.r-project.org/web/packages/unbalanced/index.html).

Algorithms	Parameters	Package
Classifier		
KNN	$K = 1$	class
SVM-G	$c = 1e+09$, $g = 1e-07$	e1071
C4.5	Default	RWeka
Random forest	Default	randomForest
Naive Bayes	Default	naivebayes
Feature selection		
T-test	–	sigFeature
ANOVA	–	mlr3filters
Chi-Squared	–	FSelector
Info Gain	–	FSelector
RF-Imp	–	FSelector
ReliefF	Neighbors = 1	FSelector
SVM-RFE	–	sigFeature
Balancing		
ROS	Ratio 1:1	Own impl.
RUS	Ratio 1:1	Own impl.
SMOTE	Ratio 1:1, k=5	Unbalanced

4.3. Feature selection and balancing

Since our goal is to assess how data balancing affects feature selection, we will combine the 7 FS algorithms with the 4 balancing strategies explained in Sections 2 and 3 (all the methods are listed in Table 2).

There are two possible ways to combine these algorithms: either to perform feature selection first and then to balance the dataset (FS+bal), or in reverse, to balance the dataset first and then to perform feature selection (bal+FS).

All of these combinations (7 FS algorithms, 4 balancing strategies, and 2 ways to combine them) added up to a total of 56 configurations that were all tested.

Since all the FS algorithms used were rankers, they only offered a list of features sorted by importance, without ever indicating the optimal number of features for any algorithm. In other studies (Pes, 2020), the authors selected a specific number of features what could induce a bias in the results. Here, we wished to conduct a broader and more in-depth study of the process, so as to offer a better overview of the behavior of the interactions between the two preprocessing steps, which is why up to 20 different scenarios were considered, each using a different percentage of selected features, with a separation of 5 points between them.

4.4. Classifiers

Some of the most popular classifiers were used for testing feature selection: k-nearest neighbors (KNN), SVM-Gaussian, C4.5 trees, Random Forest, and Naive Bayes (Bolón-Canedo & Alonso-Betanzos, 2018).

4.5. Parameters

All the algorithms together with the parameters used and the libraries that were applied are shown in Table 2. For the SVM-G classifier we performed a grid parameter search and we found that using $c = 1e+09$ and $g = 1e-07$, we obtained an optimum performance in all datasets. Regarding ReliefF, as it needs a long time to rank the features, we set the parameter n to 1 in order to reduce its execution time. For SMOTE, we tested a range of values from 1 to 20 for the parameter k, we observed that the performance is very similar for all of them and the recommended parameter 5 gives usually slightly better

Table 3
Confusion matrix.

	Actual positive	Actual negative
Predicted true	True positive (TP)	False positive (FP)
Predicted false	False negative (FN)	True negative (TN)

performance than the others. Finally, with the balancing algorithms, we left the balancing ratio to 1 for all the datasets (*i.e.*, the same number of instances for both the majority and the minority classes).

4.6. Metrics

Testing an algorithm with unbalanced data can be problematic. If, for example, the unbalance ratio were 1:10, with 100% majority-class accuracy and 0% minority-class accuracy, the accuracy rate can be set at 90%, however these results are of no use, as the trained classifier cannot distinguish between the two classes.

We selected some of the most accepted metrics in feature selection and data balancing, to evaluate the performance of each configuration: the *Area Under the ROC Curve* (AUC), *Geometric Mean* (G-Mean), and F_1-*Score*.

These metrics are based on the confusion matrix (see Table 3) where the following values can be found:

- *True Positive* (TP): positive instances correctly classified (minority class in our data).
- *True Negative* (TN): negative instances correctly classified.
- *False Positive* (FP): positive instances incorrectly classified.
- *False Negative* (FN): negative instances incorrectly classified.

Our objective was to maximize the balance between TP and TN values in the diagonal. Using these values, the three basic ratios can be calculated before computing our three main metrics:

- **Recall** is the probability of considering a positive instance as positive.

$$recall = \frac{TP}{TP + FN} \tag{1}$$

- **Specificity**, as opposed to recall, is the probability of classifying a negative instance as negative.

$$specificity = \frac{TN}{TN + FP} \tag{2}$$

- **Precision** is the probability of an instance classified as positive.

$$precision = \frac{TP}{TP + TN + FP + FN} \tag{3}$$

From these ratios, the measures used to evaluate the results of the experiments can be defined.

- ***Area Under the ROC Curve*** (AUC) the mean between recall and specificity. Note that, although it is often used to evaluate multiple possible classifiers, here we just use it with a single point, which is the mean between recall and specificity. This measure has been used in earlier studies (Galar et al., 2012).

$$\text{AUC} = \frac{recall + specificity}{2} \tag{4}$$

- **F_1-Score** is the harmonic mean between precision and recall.

$$F_1\text{-Score} = 2 \times \frac{precision \times recall}{precision + recall} \tag{5}$$

- **G-Mean**, widely used in unbalanced problems, is the geometric mean between sensitivity and specificity.

$$\text{G-Mean} = \sqrt{recall \times specificity} \tag{6}$$

5. Results

This section summarizes the results of the 6860 experiments that were performed: the performance of 5 classifiers on 14 wide datasets (Table 1) using all possible combinations of 7 strategies for FS algorithms and 7 balancing strategies.[2]

For the sake of readability and simplicity, only the results of the *AUC* metric are shown, since the conclusions drawn from the F_1-Score and G-Mean metrics were very similar. Nevertheless, the figures of all the metrics are compiled in the additional material.

Fig. 2 shows all combinations of FS algorithms and classifiers in a matrix of stacked graphs. In the stacked graphs, the ordinate axis is the percentage of victories, and the abscissa axis is the percentage of selected features (20 different percentages with separation steps of 5 percent; the minimum percentage of features used is 5 percent and the maximum is 100 percent). The different shades of blue correspond to different balancing strategies (as shown in the top legend). The rows of the matrix of graphs correspond to different classifiers (shown on the right side), the columns to the selectors (specified on the top of each column).

Moreover, Fig. 3 is included where the area plots are replaced by line plots: the y axis represents the average ranks obtained for each balancing strategy and the x axis represents the percentage of the features selected by the corresponding FS algorithm (sorted from best to worst).

In Figs. 2 and 3, it can be seen that the results depend more on the classifier than on the other parameters. Broadly speaking, we can see the following insights for each classifier:

- KNN: The best performance was achieved when resampling is performed before the FS algorithm, more specifically, SMOTE+FS and ROS+FS were on the top.
- SVM-G: Although it may not be so clear in the rankings, it can be seen in the area graphs that FS+RUS usually provided the best results. Specially when SVM-RFE was the feature selector in use.
- C4.5: While not using any balancing strategies showed better results in the area graphs and ROS+FS in the average ranks, both RUS balancing strategies were the lowest in the performance rankings.
- RF: The balancing configuration FS+ROS in the area plots, and FS+RUS in the average ranks, showed better performance than the others. Unlike C4.5, the rankings showed that no use of balancing was by far the worst combination, a behavior also supported by the results of a previous study (Pes, 2020).
- NBayes: The use of RUS appeared to provide the biggest area, especially FS+RUS which can also be seen in the average ranks. According to the rankings, SMOTE was the worst, regardless of when the feature selection was implemented, either before or after its application.

Considering all the percentages of features and the different selectors used, we summarize the most remarkable balancing strategies for each classifier in Table 4.

Additionally, it is interesting to note that the number of selected features appeared not to have much impact on the final rankings of the balancing strategies.

So far, the balancing strategies have been compared to show which one worked best for each classifier. In what follows, similar graphs will be used to compare and to determine which the best combinations of classifiers and FS algorithms.

Fig. 4 compares the FS algorithms which are differentiated by colors. The columns represent each different balancing strategy used,

[2] Each balancing strategy applied in two orders, before or after the feature selection, and the case of not balancing.

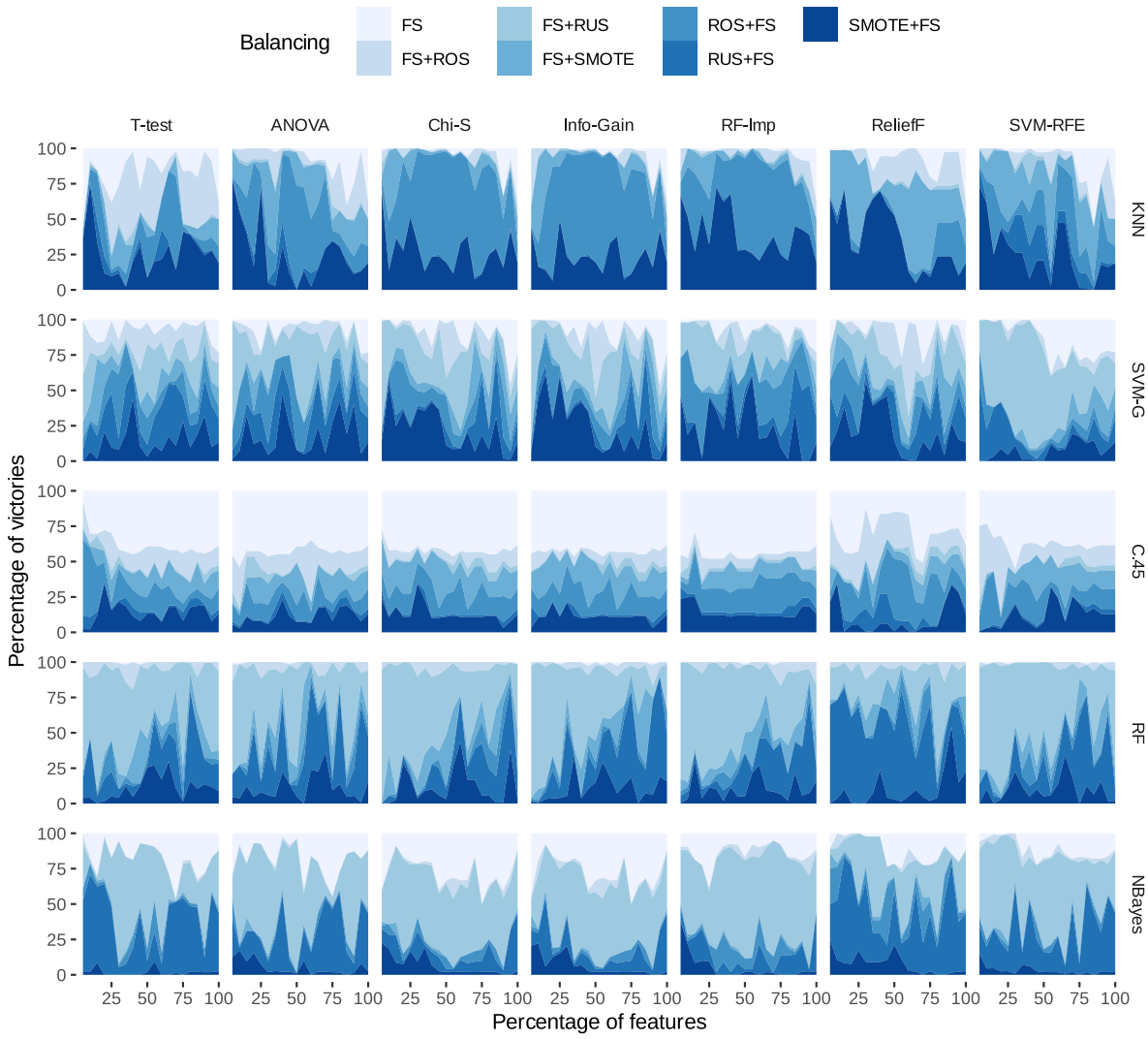


Fig. 2. Comparison of balancing methods: each row corresponds to a classifier, each column to an FS algorithm. At the intersection, a stacked graph shows the results of the different balancing methods for the corresponding combination of FS and classifier in shades of blue (the abscissa axis is the percentage of selected characteristics; the ordinate axis is the percentage of victories).

Table 4
The best balancing configurations (rows) for each classifier (columns).

	KNN	SVM-G	C4.5	RF	NBayes
FS			✓		
FS+ROS				✓	
FS+RUS		✓		✓	✓
FS+SMOTE					
ROS+FS	✓		✓		
RUS+FS					✓
SMOTE+FS	✓				

and the rows represent each classifier. It can be seen that among those that provide the best performance are SVM-RFE alongside most of the classifiers, specially with NBayes, KNN, and T-test. It also achieves good results with the KNN, making some appearances with the smallest number of features in SVM-G, C4.5, and RF.

Finally, looking at Fig. 5, where each graph shows the percentage of times that a method has been the best for different sizes of feature subsets, the classifier that shows the best overall performance was SVM-G, with KNN in second place. However, the behavior of NBayes was also noteworthy, which yielded good results for some combinations of selectors and balancing strategies, when the percentage of selected characteristics was low.

After the general overview provided by the previous figures, we used average rankings to compare the configurations of the best classifiers (KNN and SVM-G). In Table 5 can be seen the most promising combinations of classifiers, FS algorithms and balancing methods from the previous figures. It shows some interesting patterns. Combinations that use SVM-G as a classifier appear to perform better than those that use KNN. Regarding the FS algorithm, the dominance of the combinations that use SVM-RFE appears clear, as these combinations occupy the top positions. Finally, it appears that the balancing strategies have little influence on the positions that the combinations occupy; not only there is no clear pattern, but both the best and worst combinations use the same balancing strategy FS+RUS.

In an attempt to assess the effect of balancing the data and to complete the analyses performed so far, in what follows, we use the Bayesian test (Benavoli et al., 2014) to compare the FS configurations, which are the ones that do not balance the data, with the other configurations, which use different balancing strategies.

The Bayesian hypothesis testing is a relatively recent approach to the analysis of experimental results that tries to overcome the limits and problems which characterize null-hypothesis significance tests. It can be used to compare two different methods, obtaining the probabilities

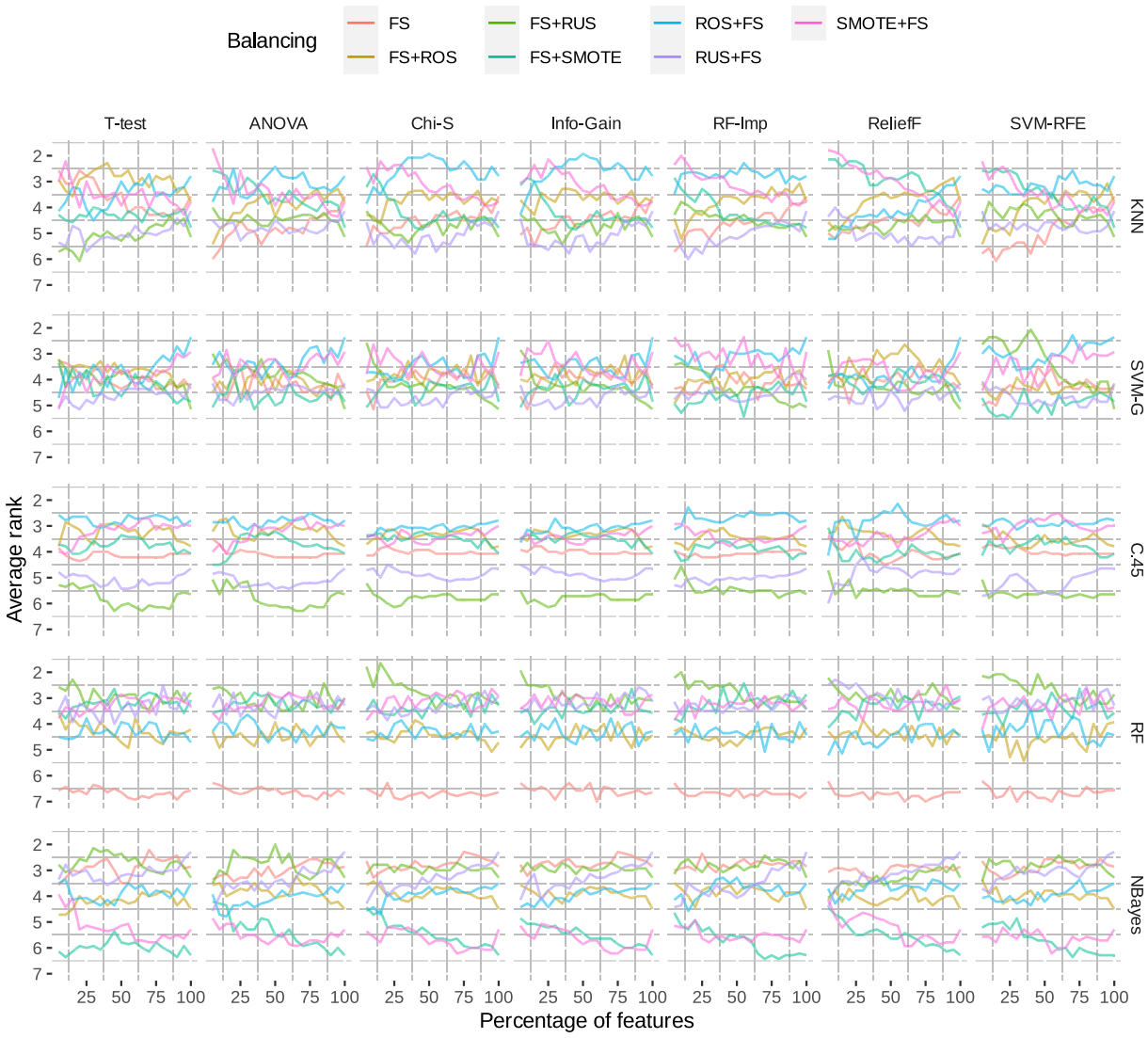


Fig. 3. Comparison of balancing methods by means of the average ranks, each row corresponds to a classifier, each column to an FS algorithm. At the intersection, a line graph shows the average rank for the different balancing methods for the corresponding combination of FS and classifier (the abscissa axis is the percentage of selected characteristics; the ordinate axis is the average rank).

that one is better than the other, or that both have a performance that is practically equivalent. In Bayesian tests, this "*practical equivalence*" is given by the value of a parameter called the Region Of Practical Equivalence (ROPE), which was set at 0.001 for this work. In addition, only the performance values for the best balancing strategies in previous experiments were used, applying only the best 10% of selected characteristics, to reduce the number of results and to facilitate the analysis.

A Bayesian test constructs a probability distribution whose parameters are obtained from the differences in the experimental results observed when comparing two methods, using certain weights that are assumed to follow a Dirichlet distribution (Ongaro & Migliorati, 2013). Given specific weights, the distribution can be used to calculate three probabilities: (*i*) the probability that the first method is better than the second; (*ii*) the probability that the second is better than the first; (*iii*) the probability that both methods are practically equivalent. Through a Monte Carlo process (Kroese et al., 2014), the weights can be sampled, obtaining different probability triplets. The three values of these triplets can be considered as barycentric coordinates that can be drawn as points in a simplex of coordinates $\{(1,0,0),(0,1,0),(0,0,1)\}$ (Benavoli et al., 2017).

As shown in Fig. 6, where the classifier KNN and the feature selector ReliefF (with the 10% of the best features) without balancing is compared to the strategies FS+ROS, FS+RUS, and FS+SMOTE. Each triangle represents one test where the left corner shows the probability of not using any balancing, the right corner represents one of the different resampling techniques (ROS, RUS, and SMOTE), and the top corner represents the probability of no significant differences between them.

Rather than using triangles, we substituted each triangle for colored tiles in the heatmaps of Fig. 7, due to the high number of tests that were performed, to facilitate the presentation of the results. Two probability numbers are displayed in each tile: the top one is the probability that the best option is to use the balancing strategy shown at the bottom of each column, the lower one is the probability that no balancing is the best option. The color of the tile is obtained from the difference between these two values, with a scale that goes from green, when the difference is more in favor of using the balancing strategy, to red, when the difference is more in favor of not balancing. The white color represents the cases in which there is no clear winner between the two approaches. Each row is for a different combination of classifier and FS algorithm (shown on the left).

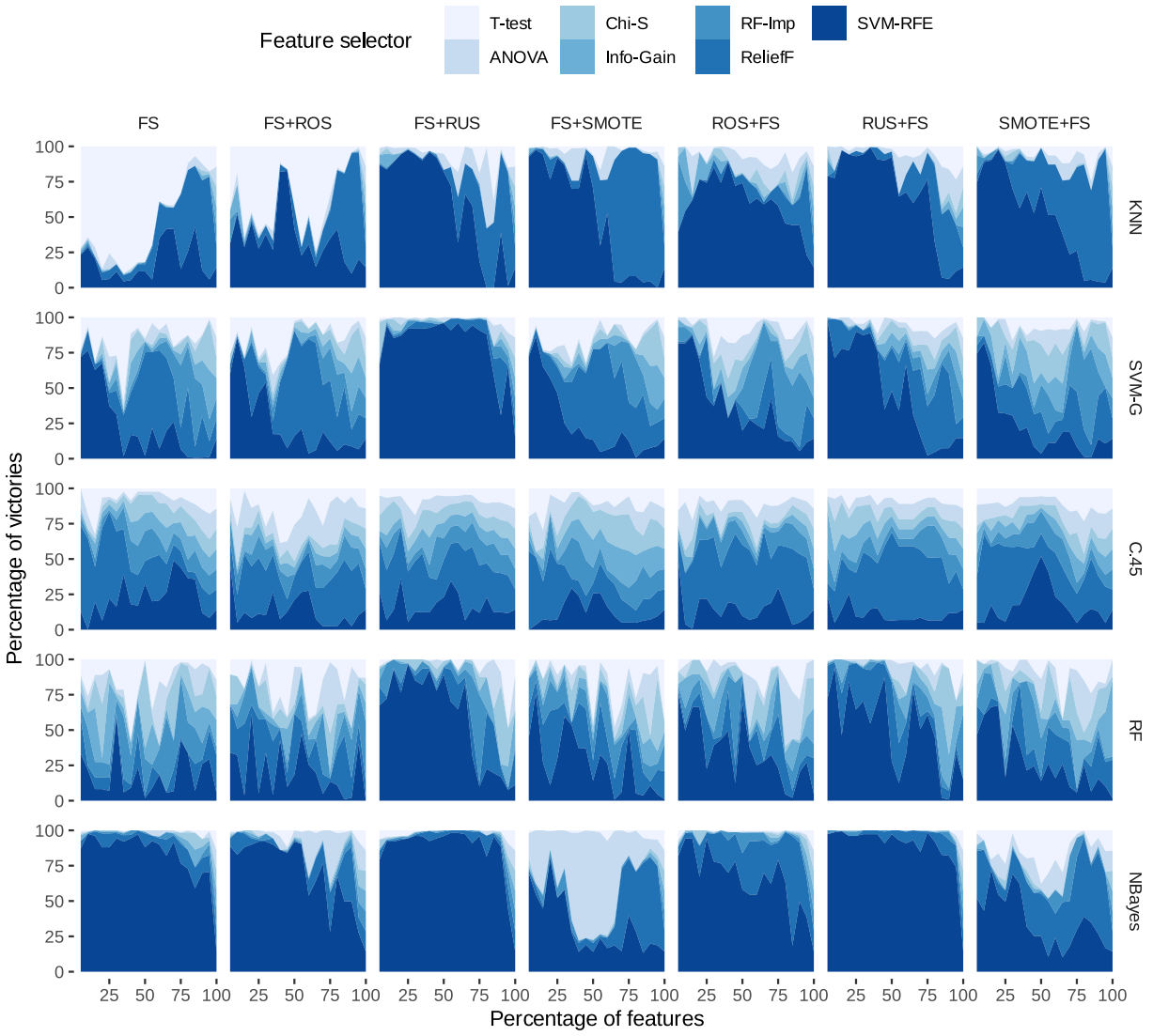


Fig. 4. Comparison of feature selector methods using the percentage of victories. The results are organized by balancing methods (columns) and classifiers (rows), while showing the percentage of features selected on the x axis.

There are two heatmaps on the figure, the left one shows the results when the AUC metric is used, and the right side when the F_1-Score is used (we also computed the results for the G-Mean, but they have been left as additional material, since they are similar to those obtained with AUC). Note that the first three tiles on the top left of the right matrix (F_1-Score) display the values corresponding to the triangles explained in Fig. 6.

Finally, we also applied Bayesian analysis to answer the question of whether it is better to balance before or after FS. The results of this comparison were very interesting (see Fig. 8). The figure is divided into three blocks according to the balancing method used (RUS, ROS, and SMOTE). The results for the different combinations of classifier and selector are grouped by rows (only those combinations that gave the best results in previous experiments have been considered). The results obtained for the different performance measures are grouped in each column. In each tile, the upper number is the probability that resampling before the FS stage is better, and the lower number the probability that resampling after the FS stage is better. The color of the tile is given by the difference between both values; if the difference is in favor of resampling before FS, the greener its color will be; if the difference is in favor of resampling after the FS stage, the color will be redder. When both strategies give similar results, the color will be close to white.

6. Discussion

As can be seen in Fig. 7, in most cases the balancing strategies (top probability) are the ones that offered the best results, especially for KNN+ANOVA and KNN+SVM-RFE. The exception is when considering the RUS+FS balancing strategy, which seems to be clearly counter-productive for most combinations of classifiers and FS algorithms. The results of the combination FS+RUS are equally discouraging.

Also of interest is the behavior of the KNN+T-test combination, which only seems to benefit from the balance obtained with the SMOTE+FS strategy, if considering the AUC, or with the SMOTE+FS and ROS+FS strategies, if considering the F_1-Score.

When analyzing the order in which the balancing and feature selection methods should be applied, Fig. 8 clearly shows that, in the case of RUS, it is better to apply balancing before feature selection. This result is confirmed by all the performance measures considered. The only exception to this general trend seems to be when using the KNN+T-test, where applying the balancing before the FS does not seem to offer many advantages, being in fact worse, if we look at the F_1 measurement. Interestingly, for KNN+T-test, this deviation from the general trend also appears for the other two balancing methods.

On the contrary, the results suggest that, in general, SMOTE and ROS perform better if applied after feature selection. This order of

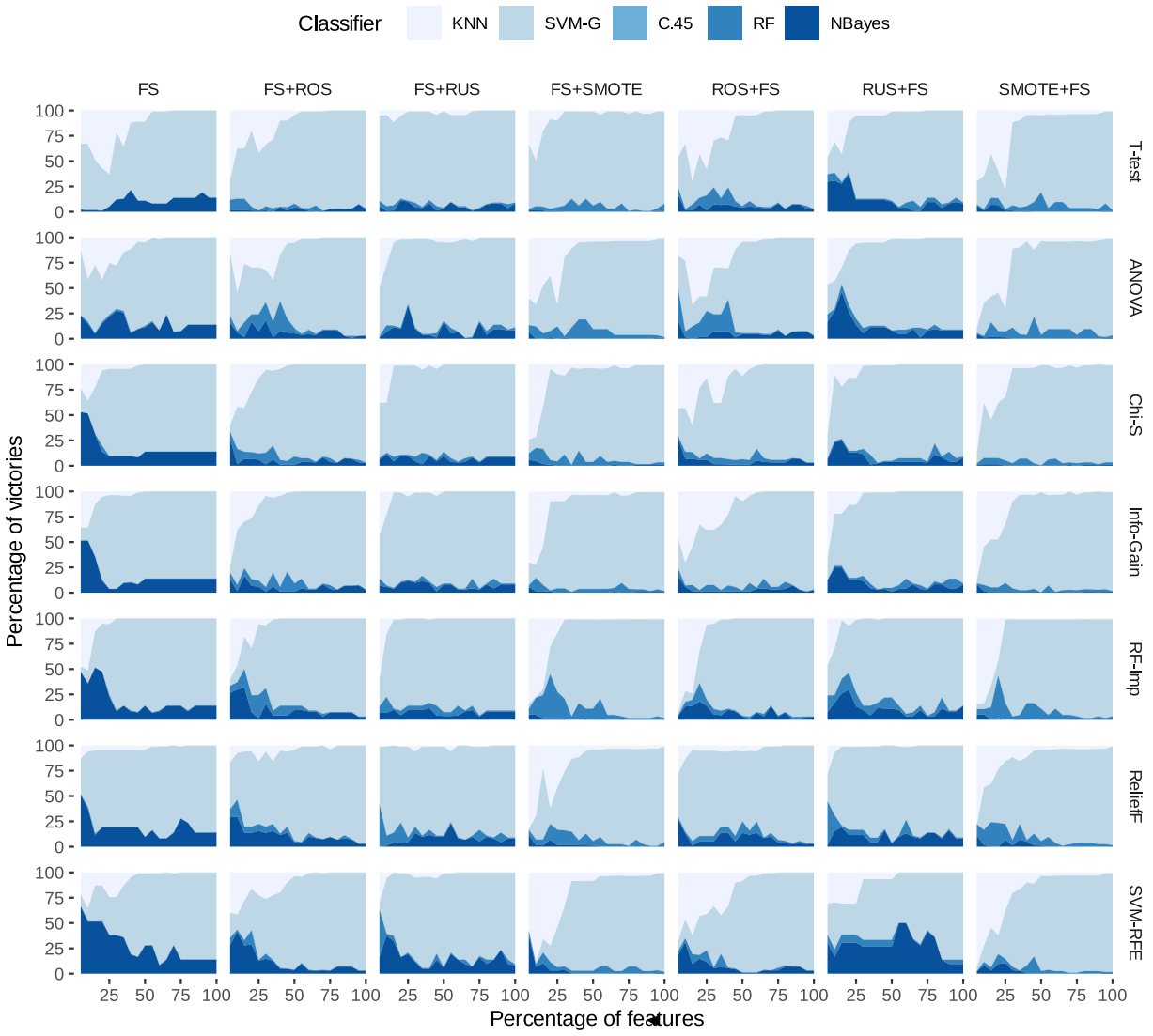
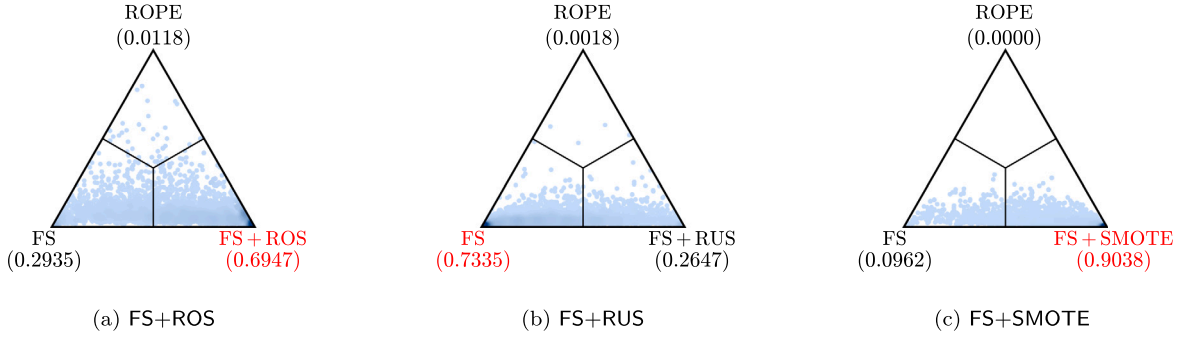


Fig. 5. Comparison of classifier performance using the percentage of victories. The results are divided by balancing methods (columns) and feature selector used (rows), while showing the percentage of features selected on the x axis.



(a) FS+ROS (b) FS+RUS (c) FS+SMOTE

Fig. 6. Example of three Bayesian tests for the classifier KNN and the feature selection ReliefF, where no balancing strategy is compared against using ROS, RUS, and SMOTE.

application is especially beneficial when the balancing method is ROS and the classifier is KNN. Although KNN+T-test is again an exception if we look at the values of the AUC and G-Mean measurements. The results observed with the combination of SMOTE with KNN+T-test are also very remarkable, suggesting that with this combination, balancing necessarily has to be done after feature selection to obtain the best results.

It is also interesting to note that the results with SMOTE appear to be halfway between using RUS and ROS.

These results extend the findings of previous studies (Pes, 2020; Zhang et al., 2017), which only gave as a general rule that to improve the final results balancing had to be done before the feature selection stage. According to our results, to choose the most appropriate order, one must also take into account the particular balancing method that

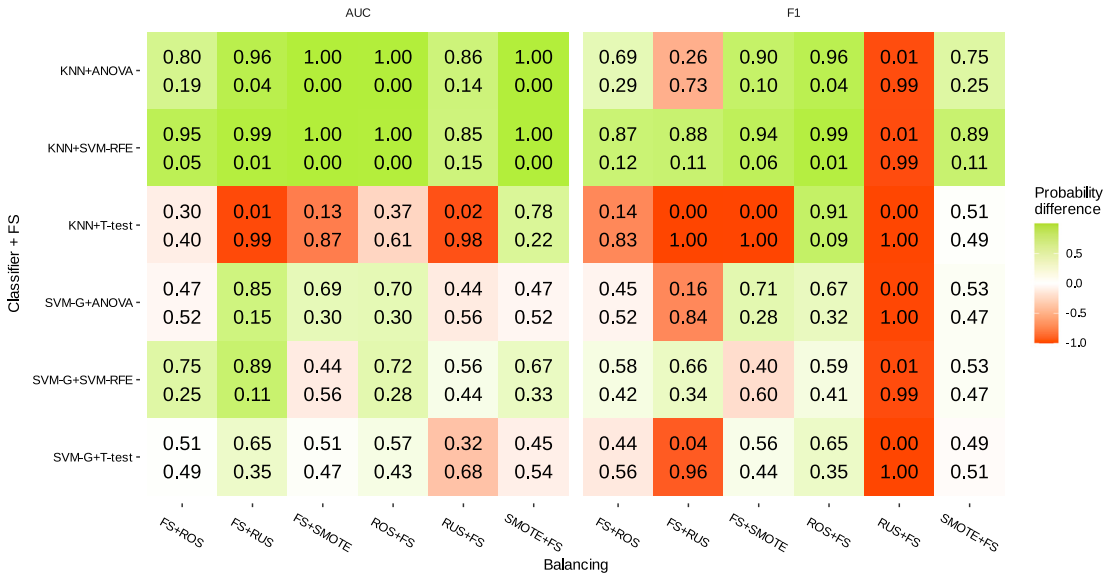


Fig. 7. Bayesian test results comparing each resampling strategy with not use any resampling.

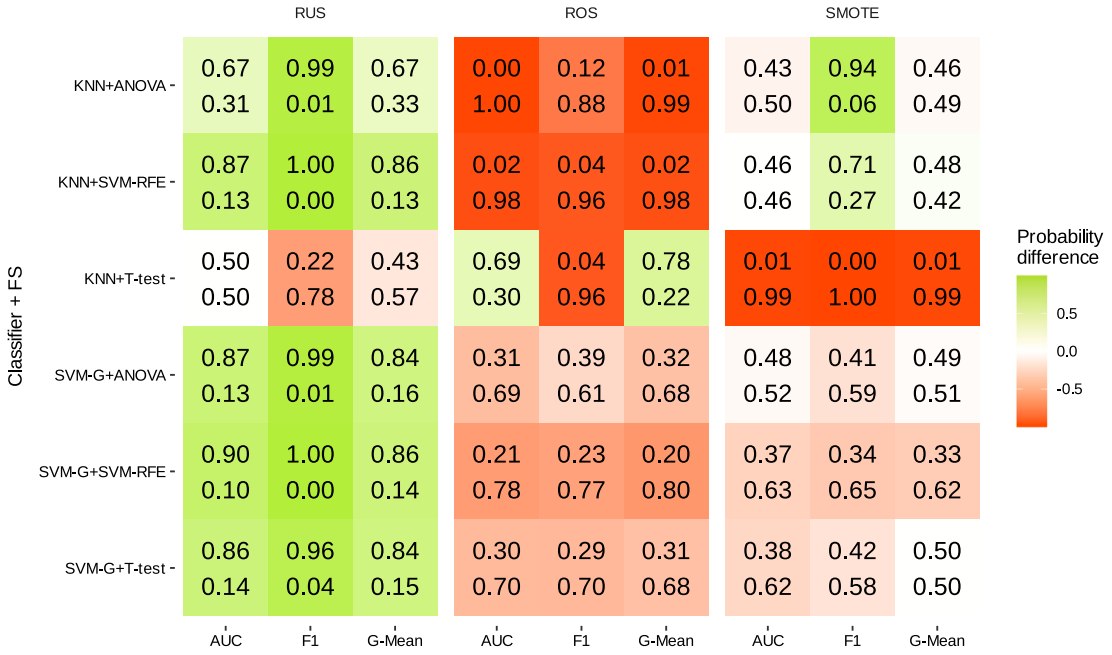


Fig. 8. Summary of the Bayesian tests to compare at which point it is better to apply the balancing techniques (top value corresponds to the application of balancing before FS, the lower value to the application afterwards).

will be used and, in some cases, even the classifier and the feature selection technique.

7. Conclusions

The objective of our study has been to test whether balancing improves the classification performance when used alongside a FS on unbalanced wide data.

The main novelty of this study is its much broader view than other previous studies, both in terms of the number of methods that were tested, and the number of datasets. Furthermore, when evaluating the methods, another notable novelty is that we have considered the results of various sizes of selected features, rather than restricting ourselves to a single size as in previous studies.

The conclusions we have reached, following thorough experimentation, have confirmed some of the results of previous studies. According to the Bayesian test using the 10% best features selected, we can state that using a strategy that includes balancing generally outperforms the use of no balancing.

However, not all balancing strategies work in the same way and their performance is highly dependent on the classifier and the FS that is used. In so far as one balancing can improve the classifier or can be counterproductive. The same happens at the time of resampling (before or after), since resampling before the FS stage is generally better with RUS while resampling after the FS stage generally works better with ROS and SMOTE.

We can conclude that the best classifiers (among those used in this study) for wide data were KNN and SVM-G, while ReliefF, T-test, and SVM-RFE were the best FS algorithms. Furthermore, the

Table 5
Average ranks of the most promising configurations previously identified.

Classifier	FS	Balancing	Avg. rank
SVM-G	SVM-RFE	FS+RUS	9.43
KNN	SVM-RFE	SMOTE+FS	12.93
KNN	SVM-RFE	FS+SMOTE	13.79
SVM-G	SVM-RFE	ROS+FS	14.00
SVM-G	SVM-RFE	SMOTE+FS	17.57
KNN	SVM-RFE	ROS+FS	20.21
SVM-G	SVM-RFE	FS+ROS	20.93
SVM-G	SVM-RFE	FS	22.00
SVM-G	SVM-RFE	FS+SMOTE	22.00
KNN	SVM-RFE	FS+RUS	22.86
SVM-G	SVM-RFE	RUS+FS	23.36
KNN	T-test	SMOTE+FS	25.14
KNN	ANOVA	SMOTE+FS	26.43
KNN	ANOVA	FS+SMOTE	27.64
SVM-G	ANOVA	ROS+FS	28.00
SVM-G	ANOVA	FS+RUS	28.57
SVM-G	T-test	FS	29.07
SVM-G	T-test	FS+RUS	29.14
SVM-G	T-test	ROS+FS	29.36
KNN	SVM-RFE	RUS+FS	29.64
KNN	T-test	FS	30.43
SVM-G	T-test	FS+ROS	30.64
KNN	T-test	FS+ROS	30.79
SVM-G	T-test	FS+SMOTE	31.00
KNN	SVM-RFE	FS+ROS	32.00
SVM-G	ANOVA	SMOTE+FS	32.21
KNN	ANOVA	ROS+FS	32.64
KNN	T-test	ROS+FS	32.93
SVM-G	T-test	SMOTE+FS	34.93
SVM-G	ANOVA	FS+ROS	35.07
SVM-G	ANOVA	FS+SMOTE	35.64
KNN	SVM-RFE	FS	35.93
KNN	T-test	FS+SMOTE	35.93
SVM-G	ANOVA	RUS+FS	35.93
SVM-G	ANOVA	FS	36.21
SVM-G	T-test	RUS+FS	36.64
KNN	ANOVA	FS+RUS	39.29
KNN	ANOVA	RUS+FS	42.57
KNN	ANOVA	FS+ROS	43.71
KNN	T-test	RUS+FS	45.00
KNN	ANOVA	FS	45.50
KNN	T-test	FS+RUS	46.21

best configuration was the SVM-RFE feature selector used before RUS for the SVM-G classifier.The percentage of chosen features among the best selected slightly affected the results, but not as much as using a balance method more suitable for the classifier. Finally, the best results are obtained using RUS as the balancing method, SVM-RFE as the feature selector (applied before RUS) and SVM-G as the classifier. So this is a good combination with which to initially process wide data. If it is necessary to use any of the other classifiers included in our study, Table 4 summarizes the best balancing and feature selector combinations for each of them.

Based on the results of this study, we plan to use more advanced algorithms for this type of problem in future works, such as ensembles and hybrid algorithms for feature selection and other balancing algorithms. Given that the final performance of the combination of selector and balancing method (and the moment at which they are applied) may also depend on the characteristics of the dataset to which it is applied, in the future we will consider using meta-learning to analyze whether relationships can be established between the characteristics of the data and the best combination.

CRediT authorship contribution statement

Ismael Ramos-Pérez: Investigation, Software, Formal analysis, Visualization, Writing - original draft. **Álvar Arnaiz-González:** Supervision, Writing - review & editing. **Juan J. Rodríguez:** Supervision, Writing - review & editing. **César García-Osorio:** Supervision, Writing - review & editing.

Acknowledgments

The project leading to these results has received funding from "la Caixa" Foundation, under agreement LCF/PR/PR18/51130007. This work was also supported by the Junta de Castilla *y* León under project BU055P20 (JCyL/FEDER, UE) and by the Ministry of Science and Innovation under project PID2020-119894GB-I00, co-financed through European Union FEDER funds.

Appendix A. Supplementary data

In the supplementary material[3] the results corresponding to the metrics F_1-*Score* and G-Mean can be found, these figures have not been included due to their similarity with the already showed AUC.

References

Abdi, L., & Hashemi, S. (2016). To combat multi-class imbalanced problems by means of over-sampling techniques. *IEEE Transactions on Knowledge and Data Engineering*, *28*(1), 238–251. http://dx.doi.org/10.1109/TKDE.2015.2458858.

Alshorman, O., Irfan, M., Saad, N., Zhen, D., Haider, N., Glowacz, A., & Alshorman, A. (2020). A review of artificial intelligence methods for condition monitoring and fault diagnosis of rolling element bearings for induction motor. In *Shock and vibration*. http://dx.doi.org/10.1155/2020/8843759.

Benavoli, A., Corani, G., Demšar, J., & Zaffalon, M. (2017). Time for a change: a tutorial for comparing multiple classifiers through Bayesian analysis. *Journal of Machine Learning Research*, *18*(1), 2653–2688.

Benavoli, A., Corani, G., Mangili, F., Zaffalon, M., & Ruggeri, F. (2014). A Bayesian Wilcoxon signed-rank test based on the Dirichlet process. In *International conference on machine learning* (pp. 1026–1034). PMLR.

Bernardini, M., Romeo, L., Misericordia, P., & Frontoni, E. (2020). Discovering the type 2 diabetes in electronic health records using the sparse balanced support vector machine. *IEEE Journal of Biomedical and Health Informatics*, *24*(1), 235–246. http://dx.doi.org/10.1109/JBHI.2019.2899218.

Bolón-Canedo, V., & Alonso-Betanzos, A. (2018). *Recent advances in ensembles for feature selection, Vol. 147*. Springer, http://dx.doi.org/10.1007/978-3-319-90080-3.

Bommert, A., Sun, X., Bischl, B., Rahnenführer, J., & Lang, M. (2020). Benchmark for filter methods for feature selection in high-dimensional classification data. *Computational Statistics & Data Analysis*, *143*, Article 106839. http://dx.doi.org/10.1016/j.csda.2019.106839, URL https://www.sciencedirect.com/science/article/pii/S016794731930194X.

Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. http://dx.doi.org/10.1023/A:1010933404324, URL https://link.springer.com/article/10.1023/A:1010933404324.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation*, http://dx.doi.org/10.1162/089976698300017197.

Díez-Pastor, J. F., Rodríguez, J. J., García-Osorio, C. I., & Kuncheva, L. I. (2015). Diversity techniques improve the performance of the best imbalance learning ensembles. *Information Sciences*, *325*, 98–117. http://dx.doi.org/10.1016/j.ins.2015.07.025, URL https://www.sciencedirect.com/science/article/pii/S0020025515005186.

Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from imbalanced data sets. In *Learning from imbalanced data sets*. http://dx.doi.org/10.1007/978-3-319-98074-4.

Galar, M., Fernández, A., Barrenechea, E., Bustince, H., & Herrera, F. (2012). A review on ensembles for the class imbalance problem: Bagging-, boosting-, and hybrid-based approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, *42*(4), 463–484. http://dx.doi.org/10.1109/TSMCC.2011.2161285.

Guyon, I., Gunn, S., Nikravesh, M., & Zadeh, L. A. (Eds.), (2006). *Feature extraction: foundations and applications, Vol. 207*. Springer.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, *46*(1–3), 389–422. http://dx.doi.org/10.1023/A:1012487302797, URL https://link.springer.com/article/10.1023/A:1012487302797.

Hamed, T., Dara, R., & Kremer, S. C. (2014). An accurate, fast embedded feature selection for SVMs. In *2014 13th International conference on machine learning and applications* (pp. 135–140). http://dx.doi.org/10.1109/ICMLA.2014.104.

Hang, Q., Yang, J., & Xing, L. (2019). Diagnosis of rolling bearing based on classification for high dimensional unbalanced data. *IEEE Access*, *7*, 79159–79172. http://dx.doi.org/10.1109/ACCESS.2019.2919406.

[3] https://doi.org/10.1016/j.eswa.2021.116015.

Japkowicz, N. (2000). The class imbalance problem: Significance and strategies. In: *Proceedings of the 2000 international conference on artificial intelligence (ICAI)* (pp. 111–117).

Johnson, K. J., & Synovec, R. E. (2002). Pattern recognition of jet fuels: comprehensive GC×GC with ANOVA-based feature selection and principal component analysis. In *Fourth international conference on environ metrics and chemometrics held in Las Vegas, NV, USA, 18-20 September 2000 Chemometrics and Intelligent Laboratory Systems*, *60*(1), 225–237. http://dx.doi.org/10.1016/S0169-7439(01)00198-8, URL https://www.sciencedirect.com/science/article/pii/S0169743901001988.

Juez-Gil, M., Arnaiz-González, A., Rodríguez, J. J., & García-Osorio, C. (2021). Experimental evaluation of ensemble classifiers for imbalance in big data. *Applied Soft Computing*, *108*, Article 107447. http://dx.doi.org/10.1016/j.asoc.2021.107447, URL https://www.sciencedirect.com/science/article/pii/S1568494621003707.

Juez-Gil, M., Saucedo-Dorantes, J. J., Arnaiz-González, A., López-Nozal, C., García-Osorio, C., & Lowe, D. (2020). Early and extremely early multi-label fault diagnosis in induction motors. *ISA Transactions*, *106*, 367–381.

Karasu, S. k., & Altan, A. (2019). Recognition model for solar radiation time series based on random forest with feature selection approach. In *2019 11th international conference on electrical and electronics engineering (ELECO)* (pp. 8–11). http://dx.doi.org/10.23919/ELECO47770.2019.8990664.

Kerber, R. (1992). Chimerge: Discretization of numeric attributes. In *Proceedings of the tenth national conference on artificial intelligence* AAAI'92, (pp. 123–128). AAAI Press.

Kira, K., & Rendell, L. A. (1992). A practical approach to feature selection. In D. Sleeman, & P. Edwards (Eds.), *Machine learning proceedings 1992* (pp. 249–256). San Francisco (CA): Morgan Kaufmann, http://dx.doi.org/10.1016/B978-1-55860-247-2.50037-1, URL https://www.sciencedirect.com/science/article/pii/B9781558602472500371.

Kohavi, R., & John, G. H. (1997a). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1), 273–324, Relevance. http://dx.doi.org/10.1016/S0004-3702(97)00043-X. URL https://www.sciencedirect.com/science/article/pii/S000437029700043X.

Kohavi, R., & John, G. H. (1997b). Wrappers for feature subset selection. *Artificial Intelligence*, *97*(1), 273–324. http://dx.doi.org/10.1016/S0004-3702(97)00043-X, URL https://www.sciencedirect.com/science/article/pii/S000437029700043X.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of RELIEF. In *LNCS*: *Vol. 784*, *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (pp. 171–182). Springer Verlag, http://dx.doi.org/10.1007/3-540-57868-4_57, URL https://link.springer.com/chapter/10.1007/3-540-57868-4_57.

Kroese, D. P., Brereton, T., Taimre, T., & Botev, Z. I. (2014). Why the Monte Carlo method is so important today. *Wiley Interdisciplinary Reviews: Computational Statistics*, *6*(6), 386–392.

Kuncheva, L. I., Matthews, C. E., Arnaiz-González, A., & Rodríguez, J. J. (2020). Feature selection from high-dimensional data with very low sample size: A cautionary tale. CoRR arXiv:2008.12025.

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., & Liu, H. (2018). Feature selection: A data perspective. *ACM Computing Surveys*, *50*(6), 1–45.

Liu, Y., Jiang, B., Feng, J., Hu, J., & Zhang, H. (2020). Classification of EEG signals for epileptic seizures using feature dimension reduction algorithm based on LPP. *Multimedia Tools and Applications*, http://dx.doi.org/10.1007/s11042-020-09135-7.

Liu, H., & Setiono, R. (1995). Chi2: feature selection and discretization of numeric attributes. In *Proceedings of the international conference on tools with artificial intelligence* (pp. 388–391). IEEE, http://dx.doi.org/10.1109/tai.1995.479783.

Luque, A., Carrasco, A., Martín, A., & de las Heras, A. (2019). The impact of class imbalance in classification performance metrics based on the binary confusion matrix. *Pattern Recognition*, *91*, 216–231. http://dx.doi.org/10.1016/J.PATCOG.2019.02.023.

Maldonado, S., Weber, R., & Famili, F. (2014). Feature selection for high-dimensional class-imbalanced data sets using support vector machines. *Information Sciences*, *286*, 228–246. http://dx.doi.org/10.1016/j.ins.2014.07.015, URL https://www.sciencedirect.com/science/article/pii/S0020025514007154.

Mitchell, T. (1997). *McGraw-Hill international editions*, *Machine earning*. McGraw-Hill.

Ng, W. W. Y., Hu, J., Yeung, D. S., Yin, S., & Roli, F. (2015). Diversified sensitivity-based undersampling for imbalance classification problems. *IEEE Transactions on Cybernetics*, *45*(11), 2402–2412. http://dx.doi.org/10.1109/TCYB.2014.2372060.

Ongaro, A., & Migliorati, S. (2013). A generalization of the Dirichlet distribution. *Journal of Multivariate Analysis*, *114*, 412–426. http://dx.doi.org/10.1016/j.jmva.2012.07.007, URL https://www.sciencedirect.com/science/article/pii/S0047259X12001753.

Peck, R., & Devore, J. L. (2011). *Statistics: The exploration & analysis of data*. Cengage Learning.

Peralta, D., Del Río, S., Ramírez-Gallego, S., Triguero, I., Benitez, J. M., & Herrera, F. (2015). Evolutionary feature selection for big data classification: A MapReduce approach. *Mathematical Problems in Engineering*, http://dx.doi.org/10.1155/2015/246139.

Pes, B. (2020). Learning from high-dimensional biomedical datasets: The issue of class imbalance. *IEEE Access*, *8*, 13527–13540. http://dx.doi.org/10.1109/ACCESS.2020.2966296.

Saeys, Y., Inza, I., & Larrañaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, *23*(19), 2507–2517. http://dx.doi.org/10.1093/bioinformatics/btm344, URL https://doi.org/10.1093/bioinformatics/btm344.

Sahu, B., Dehuri, S., & Jagadev, A. (2018). A study on the relevance of feature selection methods in microarray data. *The Open Bioinformatics Journal*, *11*(1), http://dx.doi.org/10.2174/1875036201811010117.

Urbanowicz, R. J., Meeker, M., La Cava, W., Olson, R. S., & Moore, J. H. (2018). Relief-based feature selection: Introduction and review. *Journal of Biomedical Informatics*, *85*, 189–203. http://dx.doi.org/10.1016/j.jbi.2018.07.014, URL https://www.sciencedirect.com/science/article/pii/S1532046418301400.

Vidya, A., Shanthi, D., Gokulakrishnan, P., & Manivannan, K. (2019). Lehality prediction of highly disproportionate data of ICU deceased using extreme learning machine. *International Journal of Innovative Technology and Exploring Engineering*, http://dx.doi.org/10.35940/ijitee.I1149.0789S219.

Xiao, Z., Dellandréa, E., Dou, W., & Chen, L. (2008). ESFS: A new embedded feature selection method based on SFS.

Yang, J., Li, T., Liang, G., He, W., & Zhao, Y. (2019). A simple recurrent unit model based intrusion detection system with DCGAN. *IEEE Access*, *7*, 83286–83296. http://dx.doi.org/10.1109/ACCESS.2019.2922692.

Zhang, C., Bi, J., & Soda, P. (2017). Feature selection and resampling in class imbalance learning: Which comes first? An empirical study in the biological domain. In *2017 IEEE international conference on bioinformatics and biomedicine (BIBM)* (pp. 933–938). http://dx.doi.org/10.1109/BIBM.2017.8217782.

Zhu, Z., Ong, Y.-S., & Dash, M. (2007). Markov blanket-embedded genetic algorithm for gene selection. *Pattern Recognition*, *40*(11), 3236–3248. http://dx.doi.org/10.1016/j.patcog.2007.02.007, URL https://www.sciencedirect.com/science/article/pii/S0031320307000945.