MedCrave
Step into the World of Research

International Robotics & Automation Journal

Research Article

Open Access

CrossMark

# Resampling imbalanced class and the effectiveness of feature selection methods for heart failure dataset

## Abstract

The real dataset has many shortcomings that pose challenges to machine learning. High dimensional and imbalanced class prevalence is two important challenges. Hence, the classification of data is negatively impacted by imbalanced data, and high dimensional could create suboptimal performance of the classifier. In this paper, we explore and analyse different feature selection methods for a clinical dataset that suffers from high dimensional and imbalance data. The aim of this paper is to investigate the effect of imbalanced data on selecting features by implementing the feature selection methods to select a subset of the original data and then resample the dataset. In addition, we resample the dataset to apply feature selection methods on a balanced class to compare the results with the original data. Random forest and J48 techniques were used to evaluate the efficacy of samples. The experiments confirm that resampling imbalanced class obtains a significant increase in classification performance, for both taxonomy methods Random forest and J48. Furthermore, the biggest measure affected by balanced data is specificity where it is sharply increased for all methods. What is more, the subsets selected from the balanced data just improve the performance for information gain, where it is played down for the performance of others.

**Keywords:** clinical data, imbalance class, feature selection, oversampling, under-sampling, resampling

Volume 4 Issue 1 - 2018

Mohammad Al Khaldy, Chandrasekhar Kambhampati

Department of Computer Science, University of Hull, United Kingdom

**Correspondence:** Mohammad Al Khaldy, Department of Computer Science, University of Hull, United Kingdom, Tel (0)1482 465744, Email m.a.al-khaldy@2014.hull.ac.uk

**Received:** October 27, 2017 | **Published:** February 01, 2018

**Abbreviations:** PCA, principal component analysis; LDA, linear discriminant analysis; SVM, support vector machine; IR, imbalance ratio; RF, random forest; TN, true negatives; FP, false positive; PPV, positive predicted values

## Introduction

Clinical datasets commonly have an imbalanced class distribution and high dimensional variables. Imbalanced class means that one class is represented by a large number (majority) of samples more than another (minority) one in binary classification.[1] For example, in our research dataset there are 1459 instances classified as "Alive" while 485 are classified as "Dead". Machine learning is generally predisposed by imbalanced data because most standard algorithms expect balanced class distributions, thereupon learning classification techniques achieve poorly for imbalanced data.[1,2] Many real world applications are critical for imbalanced data learning such as medical diagnosis, pattern recognition, and fraud detection.[3] Methods that can used to solve imbalanced data are categorized as the pre-processing approach and the algorithmic approach. The handling obtained by resampling the class distribution is by under-sampling the majority class, or over-sampling the minority class in the training set.[2,4] While boosting is an example of an algorithmic approach that recalculates weights with each iteration to place different weights on the training examples.[5] High dimensionality is one of the obstacles facing the mining of clinical data because high dimensionality causes high computational costs, difficulties interpreting data and may influence the classification performance. The dimensionality reduction categories have two types; feature extraction and feature selection. Feature extraction transforms the existing features into a lower dimensional space, for example, principal component analysis (PCA) and linear Discriminant analysis (LDA). Feature selection plays a crucial role in machine learning and pattern recognition.[6] It is generally the main data processing step prior to applying a learning algorithm.[7] Feature selection leads to reducing computation requirements, reducing the effect of the curse of dimensionality and developing the predictor performance.[8]

Feature selection is an important technique used in pattern analysis; it decreases data dimensionality by eliminating irrelevant and redundant variables. As Poolsawad et al.[1] stated, feature selection attempts to find the best subset of the input feature set, the selected subset must be the most relevant attributes, without transformation variables. The feature selection has three approaches

a. Wrapper methods

b. Filter methods and

c. Embedded methods.

In the filter methods, the evaluation is independent of the classification algorithm, and the objective function evaluates attributes' subsets by their information content i.e. interclass distance, statistical dependence or information measures. Conversely, the evaluation of wrapper methods uses criteria related to the classification algorithm, where the objective function is a pattern classifier that evaluates the subsets by the predictive accuracy. However, wrapper methods are slow in execution because they must train a classifier for each feature subset. Additionally, the wrappers lack generality since the optimal feature subset will be specific to the classifier under consideration. Embedded methods is a new procedure that tries to combine ranking and wrapper, these methods measure the usefulness of feature subsets by searching among the learning process where it uses cross-validation for assessment. This means that embedded is similar to wrapper with less computationally expense and is less prone to over fitting. In this research, we investigate feature selection issues in the imbalance situation. The paper is structured as follows: section 3.1 discusses the balanced class methods. Section 3.2 discusses the feature selection methods used in this paper. Information about the

dataset used, the classification algorithm, and performance measure are discussed in sections 4.1, 4.2, and 4.3 respectively. Section 5 presents and discusses the results obtained by the experiments and section 6 presents the conclusions.

## Materials and methods

### Class imbalance

When processing a large volume of data it is essential to come up with a random sample, smaller in size, which is called sampling.[9] Random sampling means each instance has an equal chance to be included in the dataset, this could involve with replacement, or without replacement. Sampling with replacement refers to selecting an instance more than once; this is used for the bootstrap algorithm. On the other hand, sampling without replacement for each instance selected simply rejects the second copy. Bootstrap is the mechanism that each time a sample was taken from a dataset to form a test or training set, it was drawn without replacement.[9]

The imbalance ratio (IR) is measured by dividing a number of samples of the minority class by a number of samples of a majority class, as:

$$IR = \frac{Number\ of\ negative\ class\ instances}{Number\ of\ positive\ class\ instances} \quad (1)$$

Several approaches can be used to solve imbalances class:

**Data level:** these methods create balanced data from the imbalanced training dataset.[10] Methods in this level are called resampling methods that can be divided into:

a. Resampling (external): Re-sampling is basically a method that can balance the imbalanced class; it provides a convenient and effective way to deal with imbalanced learning problems using standard classifiers because it alters the original training set rather than modifying the learning algorithm.[3] Therefore, the following methods can be used:

I. Oversampling: In the training set the technique increases the frequency of the majority class, see Figure 1. The drawback of this method is the result in over fitting of the data due to it making exact copies of the minority class.[11] Moreover, the size of the training set rises then increases the time to build a classifier.

II. Under-sampling: In the training set the technique decreases the frequency of the majority class. Under-sampling can remove a lot of informative examples which could be useful in the development of the classifiers.[12]

b. Active learning (internal): improves learner performance by selecting the more valuable sample to learn and leaving the less valuable.[13]

c. Weighting the data space: to avoid costly errors the training set distribution is modified using information concerning misclassification costs.[14]

**Algorithm level (Cost-sensitive learning):** by changing the classifier algorithm so it is more precise with the minority class.[10] Cost-sensitive learning attempts to learn more characteristics of minority samples, thus to minimize higher cost errors by considering higher costs for the misclassification of positive class examples with respect to the negative class.[3,14]

Methods used in this study to handle imbalance class are:

a. Resample: Produces a random subsample of a dataset using either sampling with replacement or without replacement. The original dataset must fit entirely in memory and the number of instances in the generated dataset may be specified. Using supervised learning, the dataset must have a nominal class attribute. The filter can be made to maintain the class distribution in the subsample, or to bias the class distribution toward a uniform distribution.

b. Spread Subsample: Produces a random subsample of a dataset. The original dataset must fit entirely in memory. This filter allows you to specify the maximum "spread" between the rarest and most common class. For example, you may specify that there must be, at most, a 2:1 difference in class frequencies.

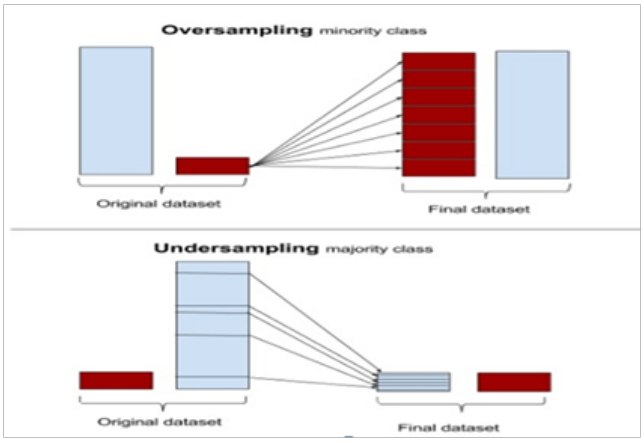


**Figure 1** Oversampling increases the minority class by copying instances for minority class. Under-sampling removes instances from the majority class.

### Feature selection

**Chi-Squared attribute evaluation:** the most simple filter approach is ranking of attributes according to value chi-square for two-way tables; the two-way table for this case is confusion matrix. The method tests the dependence of two variables, therefore does not examine the redundancy between the attributes due to the variables measured individually with respect to the class. It is used to test if the amount of a specific variable and the amount of a specific class are independent. For example, the dataset has features and a target variable, the method calculates chi-squared between every feature and the target and observes the existence of the relationship. Then, if the relation is independent we can exclude this variable. Chi-square is identified by:

$$\chi^2 = \Sigma \frac{(O-E)^2}{E} \quad (2)$$

Where o is the observed value and e is the expected value, high scores mean that features A and B are dependent. Chi-squared shares similarities with coefficient of determination, $R^2$, where $R^2$ is applicable only to numeric data. The ranking methods are very simple and the computational time is cheap and equal to $O(\log n)$ for $n$ features.

**Correlation coefficient evaluation:** one of the simple methods is Pearson's Correlation Coefficient which measures a linear correlation between two variables where the resulting value lies between [-1, 1], where -1 refers to a negative relation, 1 refers to positive relation and 0 is no relation between the two variables. The Pearson correlation coefficient is defined as:

$$R(i) = \frac{cov(f_i, Y)}{\sqrt{var(f_i)var(Y)}} \quad (3)$$

Where $cov$ designates the covariance and $var$ the variance, $f_i$ an input feature and $Y$ the output feature. Feature selection technique uses this algorithm to estimate the relationship between each variable and the concept variable, and chooses only the variables that have a high positive or negative relation

**ReliefF attribute evaluation**: a supervised feature weight estimation, that measures attribute quality to select a feature subset.[15] The idea of a Relief algorithm is it randomly selects features to estimate the feature's quality that has weights more than the threshold weight using the difference in features value with the nearest sample.[16] By this algorithm, it selects a random instance $R$, and then searches for $k$ nearest neighbor from the same class (called nearest hit) and each different class (called nearest miss). The formula for updating the weight is as follows:

$$W_x = W_x - \frac{diff(F,R,H)^2}{m} + \frac{diff(F,R,M)^2}{m} \quad (4)$$

Where $W_x$ the weight for is attribute $F$, $R$ is a random sample instance, $M$ is the nearest miss, $H$ is the nearest hit, $m$ is the number of random sample instances.

**Information Gain Attribute Evaluation:** it quotes the information between two features using entropy to measure the relevance between the attributes and the concept target, which describes the priority of an inconsistency.[17] It computes the mutual information principle with reference to the number of times: feature value follows a class, the feature value follows without the class, and the class follows without the feature value.[18] We can calculate the information achieved by learning a variable $f_i$ by the information entropy as:

$$IG(f_i) = H(T) - \Sigma \frac{T_i}{T} H(T_i) \quad (5)$$

Where $H(T)$ the entropy of the given dataset is, $H(T_i)$ is the entropy after the data split using $f_i$ and can calculate as:

$$H(T) = \sum_c P(c) \log\left(P(c)\right) \quad (6)$$

**Wrapper filter:** The wrapper uses the induction algorithm itself as part of the evaluation function to search for a good subset.[19] The feature selection algorithm then occurs as a wrapper around the induction algorithm used as a "Black Box". Black Box is used to prompt a classifier that will be convenient to classify future instances; therefore the algorithm leads to a search for a high-performance subset in terms of classification.[20] The evaluation involves the induction algorithm using cross validation to evaluate the precision of the learning scheme for a set of features approximating the accuracy using estimation accuracy techniques. The search space size for n features is $2^n$ with complexity $O(n)$ due to each state in the search space representing a subset. Wrapper is unfeasible for computationally intensive methods as it must train a classifier for each feature subset.[21]

The wrapper can rely on heuristic searching for all possible search subsets, an example of a heuristic search is hill climbing. The hill climbing algorithm shown in Figure 2 displays adding features one at a time until no further improvement can be achieved.

1: Let S ← initial state.
2: Expand S: apply all operators to S, given v's children.
3: Apply the evaluation function *Eval* to each child w of s.
4: Let v' = the child w with highest evaluation *Eval* (5)
5: If $(v') > f(S)$ then $S \leftarrow v'$; *goto* 2
6: Return S

**Figure 2** Hill-Climbing Search Algorithm.

**Support vector machine (SVM)**: It is an embedded feature selection which endeavors to select attributes and conclude model parameters instantaneously.[22] Embedded methods are different from the Filter and Wrapper methods due to the feature selection performed in the process of learning.[23] The embedded method attempts to find an optimal subset of variables adapting the structure of the classifier, embedded algorithms compromise the interaction with the classification model. The SVM algorithm uses margin hyper plane to ensure the two patterns are separated linearly; the hyper plane maximizes the sum of distances between the margin and hyper plane.[24] The model is trained with all features by setting the coefficients associated with the features to $0$ and attempting to remove these features while preserving model performance. If the classes are not linearly then use a variant of SVM.[22,25]

## Experiments setting

### Dataset

The dataset used is a real life heart failure dataset obtained from the *Hull-LifeLab* clinical database (University of Hull, UK). In this dataset, there are 60 features with 1944 patient records, after cleansing data and impute missing values the samples become 1750 instances. The variable 61 is a target output variable that classifies the instances to "Dead" or "Alive", see Table 1. The imbalance class plain by the percentage of 1 to 3 for Dead to Alive class, and by imbalance ratio which is 33%.

### Classification method

**C4.5/J48:** The j48 technique is an enhancement of the ID3 algorithm,[26] the improvement of ID3 are features like speed, size, memory and a rule set output.[27] The algorithm steps are:[28–30]

a. The tree represents a leaf for the instances existing in the same class, the leaf is returned by labeling with the same class.

b. Calculate the information for each variable, selected by a test on the attributes, and then calculate the achieved information that would be decided from a test on the variable.

c. Iteratively apply the present selection principle to find the highest information gain; this variable is selected for branching.

**Table 1** Target Classes Distribution on Hull-Life Lab

| | | |
|---|---|---|
| No. of features | 60 | |
| No. of samples | 1750 | |
| Target output | Mortality | |
| Class | Alive | Dead |
| Frequency | 1313 | 437 |
| Percentage | 3 | 1 |
| Imbalance ratio (IR) | 0.33 | |

**Random Forest (RF):** RF consists of lots of decision trees based on a random selection of data and attributes. The $X_n$ independent variables can be used for building a decision tree, the variables will be selected randomly into sets and these random decision trees create a forest.[31] The benefit of the large number of trees is that most of the trees can provide correct prediction of class. Another point is that all the trees do not make mistakes in the same place.[32] The final classifier gains accurate results since it is taken as a combination of more than one classifier.[33]

### Performance measures

After the classifiers were trained, the performance of the classifiers is compared using the confusion matrix to find accuracy, sensitivity, and specificity. The confusion matrix is a specific table, where each row represents the instances in a predicted class, with each column representing the instances in an actual class (or vice-versa), as in Table 2.

**Table 2** Confusion Matrix

| | **Predicted no** | **predicted yes** |
|---|---|---|
| Actual NO | True Negative (TN) | False Positive (FP) |
| Actual YES | False Negative (FN) | True Positive (TP) |

Where the outcomes are:

True Positive (TP): are the positive elements that classified correctly.

True Negative (TN): are the negative elements that classified correctly.

False Positive (FP): are the negative elements that classified as positive.

False Negative (FN): are the positive elements that classified as negative.

The performance metrics can be measured by different equations such as accuracy, sensitivity, and specificity. Accuracy is simply measured by the possibility that the algorithm can predict negative and positive instances correctly.[34,35] as:

$$Accuracy\left(ACC\right) = \frac{TP+TN}{TP+TN+FP+FN} \tag{7}$$

Sensitivity and specificity are the possibility that the algorithms can correctly predict positive and negative instances respectively, as:

$$Sensitivity\left(SEN\right) = \frac{TP}{TP+FN} \tag{8}$$

$$Specificity\left(SPEC\right) = \frac{TN}{TN+FP} \tag{9}$$

Precession (PPV) measures the proportions of positive and negative results,

$$Precision\left(PPV\right) = \frac{TP}{TP+FP} \tag{10}$$

## Results and discussion

Five feature selection methods were implemented in this study; they are Chi-squared, information gain, ReliefF, embedded-SVM, and wrapper, to select the best subset of the original dataset that consists of 60 variables. The subsets groups chosen are 11, 23, 34, 44 variables. Then, we find the performance of each one of these groups for each feature selection technique using random forest and J48 methods for classification. After that, we balance the imbalance classes using resample and spread subsample methods and employ random forest and J48 classification again.

Tables 3 illustrate the results of accuracy, specificity, sensitivity, and PPV for a different number of selected features by applying the RF technique. Table 4 shows the outcomes of accuracy, specificity, sensitivity, and PPV for a different number of selected features by applying J48 learning algorithms. From the tables, we notice that chi-squared and wrapper obtained high results when selecting 11 and 23 variables, the performance then decreases when more variables are added. In contrast, information gain starts with an accuracy of 78.87% with 11 features and then increases when more features are added until 86.05% with 44 variables is reached, which is almost equal to other methods. ReliefF and embedded methods starts with 87% accuracy which is a little bit less than the wrapper and chi-squared techniques, then decreases slowly until 85% with 44 variables is reached, which is less than other methods, including information gain. The interpretation for this is that information gain with more variables returns more information improving the gained information which increases the performance. On the other hand, wrapper returns the best results with fewer variables because this method works by repeatedly searching for the best subset that has the best performance with the least features. Further, adopting the embedded SVM technique to pick the features that obtain the best performance with the SVM diagonal, and the observations show that it gets good results with a lower number of attributes, but with an increased number of attributes quantitative the performance is greatly affected. For the research dataset, there are features that can improve the classification performance if they are selected in a subset selection. By measuring the rank correlation between variables rankings, we discern similar techniques that produce the best outcomes. For example, features selected in chi-squared and wrapper mainly appear in both subsets. Similarly, embedded-SVM and ReliefF have many of these features that appear in a selected minimum number of attributes.

As we can see, the accuracy of balanced data using the resample

method has significant enhancement since resampling increases the minority class by copying instances called oversampling, or decreases the majority class by deleting instances called undersampling. Meanwhile, using the spread subsample method does not show significant improvement and almost equal the results with the imbalanced class. We also use a method called Class Balancer that balance classes making the number of each class value equal to another by increasing minority by duplicating their instances and decreasing majority by deleting some of the instances. We did not display the Class Balancer method because it is not a suitable method to balance data and this reduction in classification performance is due to changing the integrity of the dataset, i.e. if the balancing method cannot enhance the classification results we do not need to use it.

**Table 3** Accuracy, Specificity, Sensitivity, and PPV Results in Implementing Random Forest Classification for Several Feature Selection Methods on Different Number of Subsets comparing with imbalanced class and balanced class used resample and spread subsample methods

| | | Accuracy | | | Specificity | | | Sensitivity | | | PPV | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. of Features | Feature selection method | Imbalanced data | Resample Data | Spread Sub Sample | Imbalanced data | Resample Data | Spread Sub Sample | Imbalanced data | Resample Data | Spread Sub Sample | Imbalanced data | Resample Data | Spread Sub Sample |
| 11 | Chi-squared | 88.00% | 97.02% | 88.91% | 62.20% | 90.62% | 62.47% | 96.80% | 99.16% | 97.72% | 88.50% | 96.95% | 88.67% |
| 11 | Information gain | 78.97% | 90.97% | 79.31% | 37.30% | 73.23% | 38.90% | 92.80% | 96.88% | 92.76% | 81.60% | 91.58% | 82.02% |
| 11 | Relief | 87.02% | 95.25% | 88.11% | 57.90% | 85.35% | 59.04% | 96.70% | 98.55% | 97.79% | 87.30% | 95.29% | 87.76% |
| 11 | Embedded-SVM | 87.25% | 94.91% | 86.74% | 60.60% | 83.52% | 58.58% | 96.10% | 98.71% | 96.12% | 88.00% | 94.74% | 87.64% |
| 11 | Wrapper | 88.22% | 95.42% | 87.88% | 59.70% | 83.75% | 57.89% | 97.70% | 99.31% | 97.87% | 87.90% | 94.84% | 87.47% |
| 23 | Chi-squared | 88.20% | 95.08% | 87.88% | 59.30% | 82.84% | 59.50% | 97.90% | 99.16% | 97.33% | 87.80% | 94.55% | 87.84% |
| 23 | Information gain | 83.37% | 92.91% | 83.54% | 48.70% | 76.66% | 49.20% | 94.90% | 98.32% | 94.97% | 84.80% | 92.68% | 84.89% |
| 23 | Relief | 86.74% | 94.11% | 86.74% | 55.80% | 80.09% | 56.06% | 97.00% | 98.78% | 96.95% | 86.80% | 93.71% | 86.89% |
| 23 | Embedded-SVM | 86.91% | 94.51% | 86.90% | 58.40% | 81.46% | 58.35% | 96.40% | 98.86% | 96.42% | 87.40% | 94.13% | 87.43% |
| 23 | Wrapper | 87.82% | 95.14% | 87.77% | 58.40% | 82.84% | 59.04% | 97.60% | 99.24% | 97.33% | 87.60% | 94.56% | 87.71% |
| 34 | Chi-squared | 86.28% | 94.57% | 86.74% | 55.10% | 81.24% | 54.69% | 96.60% | 99.01% | 97.41% | 86.60% | 94.07% | 86.59% |
| 34 | Information gain | 86.62% | 94.40% | 85.94% | 57.20% | 80.78% | 54.69% | 96.40% | 98.93% | 96.43% | 87.10% | 93.93% | 86.47% |
| 34 | Relief | 85.42% | 96.29% | 85.54% | 54.20% | 86.50% | 52.63% | 95.80% | 99.54% | 96.50% | 86.30% | 95.68% | 85.96% |
| 34 | Embedded-SVM | 86.28% | 94.05% | 86.34% | 55.60% | 79.18% | 55.38% | 96.50% | 99.01% | 96.65% | 86.70% | 93.46% | 86.68% |
| 34 | Wrapper | 87.20% | 94.97% | 87.25% | 57.40% | 81.92% | 57.21% | 97.10% | 99.31% | 97.26% | 87.30% | 94.29% | 87.23% |
| 44 | Chi-squared | 86.17% | 94.22% | 85.82% | 53.30% | 79.86% | 52.17% | 97.10% | 99.01% | 97.03% | 86.20% | 93.66% | 85.91% |
| 44 | Information gain | 86.05% | 94.00% | 85.82% | 54.00% | 78.49% | 53.09% | 96.70% | 99.16% | 96.73% | 86.30% | 93.27% | 86.10% |
| 44 | Relief | 85.08% | 94.11% | 85.02% | 49.00% | 78.72% | 48.74% | 97.10% | 99.24% | 97.11% | 85.10% | 93.34% | 85.06% |
| 44 | Embedded-SVM | 85.65% | 93.88% | 85.82% | 51.30% | 78.49% | 51.72% | 97.10% | 99.01% | 97.18% | 84.90% | 92.65% | 84.94% |
| 44 | Wrapper | 86.40% | 93.94% | 86.57% | 53.10% | 78.72% | 54.92% | 97.50% | 99.01% | 97.11% | 86.20% | 93.32% | 86.62% |

**Table 4** Accuracy, SPEC, SEN, and PPV Results in Implementing J48 Learning Algorithm for Several Feature Selection Methods on Different Number of Subsets, comparing with imbalanced class and balanced class used resample and spread subsample methods

| | | Accuracy | | | Specificity | | | Sensitivity | | | PPV | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| No. of Features | Feature selection method | Imbalanced class | Resample Data | Spread Sub Sample | Imbalanced data | Resample Data | Spread Sub Sample | Imbalanced Class | Resample Data | Spread Sub sample | Imbalanced class | Resample data | Spread Sub Sample |
| 11 | Chi-squared | 85.02% | 92.00% | 84% | 62.24% | 89.39% | 63.16% | 92.61% | 95.05% | 91.01% | 88.05% | 94.33% | 88.13% |
| 11 | Information gain | 77.25% | 86.17% | 76.74% | 37.76% | 66.36% | 33.41% | 90.40% | 92.76% | 91.17% | 81.36% | 89.23% | 80.44% |
| 11 | Relief | 85.71% | 90.85% | 83.94% | 57.89% | 79.86% | 58.35% | 94.97% | 94.59% | 92.46% | 87.14% | 93.38% | 86.96% |
| 11 | Embedded-SVM | 82.11% | 89.60% | 83.25% | 52.86% | 75.51% | 53.78% | 91.85% | 94.29% | 93.07% | 85.41% | 92.04% | 85.81% |
| 11 | Wrapper | 84.05% | 91.25% | 83.94% | 60.87% | 78.72% | 58.58% | 91.77% | 95.43% | 92.38% | 87.57% | 93.09% | 87.02% |
| 23 | Chi-squared | 82.40% | 91.08% | 83.25% | 60.64% | 74.85% | 62.24% | 89.64% | 94.59% | 90.25% | 87.25% | 93.59% | 87.78% |
| 23 | Information gain | 79.02% | 87.71% | 77.60% | 55.15% | 72.77% | 52.40% | 86.98% | 92.69% | 85.99% | 85.35% | 91.09% | 84.44% |
| 23 | Relief | 82.00% | 90.11% | 81.77% | 61.33% | 77.35% | 58.81% | 88.88% | 94.36% | 89.41% | 87.35% | 92.60% | 86.71% |
| 23 | Embedded-SVM | 82.71% | 91.25% | 82.85% | 60.41% | 81.01% | 62.70% | 89.41% | 94.67% | 89.57% | 87.15% | 93.74% | 87.83% |
| 23 | Wrapper | 82.34% | 90.68% | 82.85% | 61.56% | 79.63% | 62.01% | 89.26% | 94.36% | 89.79% | 87.46% | 93.30% | 87.66% |
| 34 | Chi-squared | 82.68% | 91.42% | 82.05 | 60.41% | 80.78% | 63.16% | 90.10% | 94.97% | 88.35% | 87.24% | 93.69% | 87.81% |
| 34 | Information gain | 82.91% | 90.22% | 81.65% | 61.56% | 81.24% | 58.58% | 90.02% | 93.22% | 89.34% | 87.56% | 93.72% | 86.63% |
| 34 | Relief | 81.82% | 89.88% | 80.68% | 61.10% | 78.49% | 59.95% | 88.73% | 93.68% | 87.59% | 87.27% | 92.90% | 86.79% |
| 34 | Embedded-SVM | 82.57% | 90.91% | 81.77% | 60.87% | 80.09% | 62.70% | 89.79% | 94.52% | 88.12% | 87.33% | 93.45% | 87.65% |
| 34 | Wrapper | 83.00% | 91.54% | 81.48% | 60.87% | 81.24% | 64.07% | 90.48% | 94.97% | 87.28% | 87.42% | 93.83% | 87.95% |
| 44 | Chi-squared | 81.48% | 90.85% | 82.28% | 60.18% | 80.32% | 61.56% | 88.58% | 94.36% | 89.19% | 86.99% | 93.51% | 87.45% |
| 44 | Information gain | 81.31% | 90.22% | 82.17% | 61.33% | 81.24% | 60.87% | 87.97% | 93.22% | 89.26% | 87.24% | 93.72% | 87.27% |
| 44 | Relief | 80.74% | 90.68% | 80.51% | 59.95% | 79.18% | 59.04% | 87.59% | 94.52% | 87.66% | 86.79% | 93.17% | 86.54% |
| 44 | Embedded-SVM | 82.57% | 91.48% | 81.94% | 61.78% | 81.01% | 62.70% | 89.49% | 94.97% | 88.35% | 87.56% | 93.76% | 87.68% |
| 44 | Wrapper | 81.25% | 90.85% | 82.05% | 60.41% | 78.95% | 62.01% | 88.18% | 94.82% | 88.73% | 87.00% | 93.12% | 87.53% |

The most interesting aspect to note is the specificity results, as we can see it doubled in some cases such as information gain for 11 variables, there was enhancement seen in all other cases for specificity. From equation 9, specificity, also called true negative rate that measures the number of negatives correctly identified. After resampling, the classification of the new dataset has dramatically raised the number of True negatives (TN) and in many cases this has doubled. TN means the true classification of "Dead" instances, where "Dead" value refers to the minority class meaning that resampling can enforce the minority classes by enhancing their samples. On the other hand, equation 9 for the specificity also depends on the false positive (FP) that measure the wrong classification of positive class. In this paper's dataset, positive referred to the "Alive" class which means the majority class. Therefore, resampling decreases the error of the positives classified. Moreover, the specificity of spread sub-sampling is equal to the specificity of the imbalanced class for all feature selection methods used in this research. The specificity and accuracy with 11 and 23 variables are differentiated depending on the feature selection methods in the two cases of balanced and imbalanced class. But by increasing variables to 34 and 44, the specificity and accuracy become equal for all feature selection methods which is around 79% and 94% for specificity and accuracy, respectively, using RF for classification of the balanced class. The same indication is seen in PPV for both classification methods used, where with 44 features it outperform around 93% for all feature selection methods used. From the sensitivity and PPV for balanced classes employed by RF and J48, respectively, it can be seen that the sensitivity becomes more than 99% with small variations either way to all feature selection methods with a different number of variables. The increase in percentage of sensitivity is because of the growing of quantity of the true positive (TP). TP refers to the majority values correctly classified; the resample method raises the number of these values which increases the value of sensitivity to 99%. While positive predicted values (PPV), are raised in the balanced class in the same way as accuracy with significant aggregate.

**Citation:** Khaldy MAI, Kambhampati C. Resampling imbalanced class and the effectiveness of feature selection methods for heart failure dataset. *Int Rob Auto J.* 2018;4(1):37–45. DOI: 10.15406/iratj.2018.04.00090

We resample the dataset to balance the class then implemented feature selection methods to compare the results with the outcome of the original data. We found that after resampling data the selected subsets were not improved but were cut back in all, except the information gain method which was highly upgraded, especially with fewer numbers of features. Referring to Table 5, chi-squared and wrapper decreased in performance in all subsets selected after balance class, while embedded-SVM and ReliefF dropped sharply in the subset selected with 11 variables. In contrast, information gain dramatically increased in accuracy from 78.97% to 86.90% using balanced class for a subset of 11 attributes. Then with 44 variables, all methods had a lesser performance for balanced data than the original dataset, including information gain. Table 6 shows the same comparison but using J48 classification, there is the same observation with 11 and 23 features. Nevertheless, for subsets with 34 and 44 variables, it changes and shows that the results before and after the resampling are almost the same. The explanation for these observations is that the true positive values (TP) of confusion matrix shows little variation between classification of the balanced and imbalanced class, where the changes come to false classified especially false positive (FP). In our dataset the positive class is the "Alive" value which is the majority, and the negative class is "Dead", which are the minority. Thereupon, resampling has grown the false positives and declined the true negatives, the reason being the majority class was 1313 for the original dataset and became 1329 for the balanced data which means 16 instances changed from positive to negative classes. Accordingly, the relation between attributes changes the combination of the data.

**Table 5** Classify balanced class using Random Forest after resampling data to compare with imbalanced class classification

| | | Accuracy | | Specificity | | Sensitivity | | PPV | |
|---|---|---|---|---|---|---|---|---|---|
| No. of Features | Feature selection method | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification |
| 11 | Chi-squared | 88.00% | 86.40% | 62.20% | 55.15% | 96.80% | 96.80% | 88.50% | 86.64% |
| 11 | Information gain | 78.97% | 86.90% | 37.30% | 56.75% | 92.80% | 96.95% | 81.60% | 87.07% |
| 11 | Relief | 87.02% | 80.97% | 57.90% | 39.36% | 96.70% | 94.82% | 87.30% | 82.45% |
| 11 | Embedded-SVM | 87.25% | 82.17% | 60.60% | 46.22% | 96.10% | 94.14% | 88.00% | 84.02% |
| 11 | Wrapper | 88.22% | 86.91% | 59.70% | 56.75% | 97.70% | 96.95% | 87.90% | 87.07% |
| 23 | Chi-squared | 88.20% | 84.80% | 59.30% | 49.43% | 97.90% | 96.57% | 87.80% | 85.16% |
| 23 | Information gain | 83.37% | 85.08% | 48.70% | 50.11% | 94.90% | 96.73% | 84.80% | 85.35% |
| 23 | Relief | 86.74% | 83.71% | 55.80% | 46.00% | 97.00% | 96.27% | 86.80% | 84.27% |
| 23 | Embedded-SVM | 86.91% | 83.20% | 58.40% | 47.83% | 96.40% | 94.97% | 87.40% | 84.54% |
| 23 | Wrapper | 87.82% | 86.17% | 58.40% | 52.86% | 97.60% | 97.26% | 87.60% | 86.11% |
| 34 | Chi-squared | 86.28% | 84.68% | 55.10% | 48.97% | 96.60% | 96.57% | 86.60% | 85.04% |
| 34 | Information gain | 86.62% | 84.40% | 57.20% | 48.51% | 96.40% | 96.34% | 87.10% | 84.90% |
| 34 | Relief | 85.42% | 83.08% | 54.20% | 45.31% | 95.80% | 95.66% | 86.30% | 84.01% |
| 34 | Embedded-SVM | 86.28% | 82.91% | 55.60% | 43.71% | 96.50% | 95.96% | 86.70% | 83.67% |
| 34 | Wrapper | 87.20% | 85.14% | 57.40% | 50.57% | 97.10% | 96.65% | 87.30% | 85.45% |
| 44 | Chi-squared | 86.17% | 83.31% | 53.30% | 44.85% | 97.10% | 96.12% | 86.20% | 83.97% |
| 44 | Information gain | 86.05% | 83.88% | 54.00% | 46.45% | 96.70% | 96.34% | 86.30% | 84.39% |
| 44 | Relief | 85.08% | 83.48% | 49.00% | 45.54% | 97.10% | 96.12% | 85.10% | 84.13% |
| 44 | Embedded-SVM | 85.65% | 82.74% | 51.30% | 44.85% | 97.10% | 95.35% | 84.90% | 83.86% |
| 44 | Wrapper | 86.40% | 83.02% | 53.10% | 42.56% | 97.50% | 96.50% | 97.11% | 83.47% |

**Citation:** Khaldy MAI, Kambhampati C. Resampling imbalanced class and the effectiveness of feature selection methods for heart failure dataset. *Int Rob Auto J.* 2018;4(1):37–45. DOI: 10.15406/iratj.2018.04.00090

**Table 6** Classify balanced class using J48 after resample data to compare with imbalanced class classification

| | | Accuracy | | Specificity | | Sensitivity | | PPV | |
|---|---|---|---|---|---|---|---|---|---|
| No. of Features | Feature selection method | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification | Imbalanced data | Resample before classification |
| 11 | Chi-squared | 85.02% | 85.02% | 62.24% | 62.24% | 92.61% | 92.61% | 88.05% | 88.05% |
| 11 | Information gain | 77.25% | 84.11% | 37.76% | 59.95% | 90.40% | 92.16% | 81.36% | 87.36% |
| 11 | Relief | 85.71% | 79.42% | 57.89% | 34.10% | 94.97% | 94.52% | 87.14% | 81.16% |
| 11 | Embedded-SVM | 82.11% | 79.60% | 52.86% | 47.14% | 91.85% | 90.40% | 85.41% | 83.71% |
| 11 | Wrapper | 84.05% | 84.11% | 60.87% | 59.95% | 91.77% | 92.16% | 87.57% | 87.36% |
| 23 | Chi-squared | 82.40% | 82.91% | 60.64% | 63.16% | 89.64% | 89.49% | 87.25% | 87.95% |
| 23 | Information gain | 79.02% | 83.65% | 55.15% | 63.39% | 86.98% | 90.40% | 85.35% | 88.12% |
| 23 | Relief | 82.00% | 81.48% | 61.33% | 56.75% | 88.88% | 89.72% | 87.35% | 86.17% |
| 23 | Embedded-SVM | 82.71% | 82.22% | 60.41% | 61.10% | 89.41% | 89.26% | 87.15% | 87.33% |
| 23 | Wrapper | 82.34% | 83.37% | 61.56% | 62.24% | 89.26% | 90.40% | 87.46% | 87.80% |
| 34 | Chi-squared | 82.68% | 82.45% | 60.41% | 61.78% | 90.10% | 89.34% | 87.24% | 87.54% |
| 34 | Information gain | 82.91% | 82.97% | 61.56% | 61.78% | 90.02% | 90.02% | 87.56% | 87.62% |
| 34 | Relief | 81.82% | 81.31% | 61.10% | 64.07% | 88.73% | 87.05% | 87.27% | 87.92% |
| 34 | Embedded-SVM | 82.57% | 82.05% | 60.87% | 60.41% | 89.79% | 89.26% | 87.33% | 87.14% |
| 34 | Wrapper | 83.00% | 82.91% | 60.87% | 62.24% | 90.48% | 89.79% | 87.42% | 87.72% |
| 44 | Chi-squared | 81.48% | 81.48% | 60.18% | 61.10% | 88.58% | 88.27% | 86.99% | 87.21% |
| 44 | Information gain | 81.31% | 81.54% | 61.33% | 61.33% | 87.97% | 88.27% | 87.24% | 87.27% |
| 44 | Relief | 80.74% | 80.91% | 59.95% | 61.56% | 87.59% | 87.36% | 86.79% | 87.22% |
| 44 | Embedded-SVM | 82.57% | 81.94% | 61.78% | 60.87% | 89.49% | 88.96% | 87.56% | 87.23% |
| 44 | Wrapper | 81.25% | 81.54% | 60.41% | 61.10% | 88.18% | 88.35% | 87.00% | 87.22% |

## Conclusion

In this paper, we demonstrated the effectiveness of imbalanced class for the high dimensional dataset, moreover, we discussed this issue concerning the implementation of five different feature selections. The feature selection techniques used were information gain, Chi-squared, ReliefF, embedded-SVM, and wrapper, while the manipulation of imbalanced class was employed by resampling and spread subsample, and the learning test used was the J48 and random forest algorithms. The implementation was a supervised random under-sampling for the majority class. Our findings from this paper are that handling a high dimensional imbalanced dataset is valuable for all feature selection methods that are filtering, wrapper, and embedded, depending on the number of features has been selected. This holds for clinical data classification fields, it seems that imbalanced classes provide samples that are unequal to make overviews about groups of the instances. The outcomes of the experiments show that resampling of the imbalanced class generates a good enhancement in performance results for all measurements such as accuracy, specificity, sensitivity, and PPV. In contrast, using spread sub-samples to balance the class distributions the results were no different when compared with classification for the imbalanced class. Specificity had a dramatic increase using resample methods because it raised the number of negative class (minority class) correctly classified, and reduced the number of positive class (majority class) incorrectly classified. All other performance measures performed better using resample to balance class, although they performed less well compared with specificity measure. Even though sensitivity is high for all feature selection methods it goes up to 99%, there are small variation to this for all feature selection methods in a different number of features. The biggest improvement in performance was for the information gain method, considering the resampling added more information to the features which increased the prediction results. The second analysis of this paper was the feature selection on a balanced dataset; the experiments show that the resampling data has highly improved only the information gain method. All other methods reduced their performance, or produced the same performance as the original data. This is because when information is added the interrelation between attributes and the target class is increased. However, other methods affected by dispersion of the new samples, for example, ReliefF which depends on nearest instances with class's values, new samples causes mistakes in this relation. For any future work, another technique of balancing class could be performed, and experiments repeated for other datasets.

**Citation:** Khaldy MAI, Kambhampati C. Resampling imbalanced class and the effectiveness of feature selection methods for heart failure dataset. *Int Rob Auto J.* 2018;4(1):37–45. DOI: 10.15406/iratj.2018.04.00090

## Acknowledgments

None.

## Conflicts of interest

None.

## References

1. Poolsawad N, Kambhampati C, Cleland J. Balancing class for performance of classification with a clinical dataset. *Proceedings of the World Congress on Engineering*. 2014;1:1–6.
2. He H, Garcia EA. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*. 2009;21(9):1263–1284.
3. Cao P, Liu X, Zhang J, et al. ℓ2, 1 norm regularized multi–kernel based joint nonlinear feature selection and over–sampling for imbalanced data classification. *Neurocomputing*. 2016;234:38–57.
4. Kirshners A, Parshutin S, Gorskis H. Entropy–based classifier enhancement to handle imbalanced class problem. *Procedia Computer Science*. 2017;104:586–591.
5. Mahdiyah, Irawan MI, Imah EM. Integrating data selection and extreme learning machine for imbalanced data. *Procedia Computer Science*. 2015;59:221–229.
6. Hu Q, Che X, Zhang L, et al. Feature evaluation and selection based on neighborhood soft margin. Neurocomputing. 2010;73(10–12):2114–2124.
7. Hall MA, Smith LA. Feature Selection for Machine Learning: Comparing a Correlation–Based Filter Approach to the Wrapper. *FLAIRS conference*. 1999. p. 235–239.
8. Chandrashekar G, Sahin F. A survey on feature selection methods. *Computers & Electrical Engineering*. 2014;40(1):16–28.
9. Witten IH, Frank E. *Data Mining: Practical machine learning tools and techniques*. 2011. p. 1–558.
10. Loyola–González O, Medina–Pérez MA, Martínez–Trinidad JF, et al. PBC4cip: A new contrast pattern–based classifier for class imbalance problems. *Knowledge–Based Systems*. 2017;115:100–109.
11. Al–Shahib A, Breitling R, Gilbert D. Feature selection and the class imbalance problem in predicting protein function from sequence. *Applied Bioinformatics*. 2005;4(3):195–203.
12. Batuwita, Palade V. Efficient resampling methods for training support vector machines with imbalanced datasets. *Neural Networks (IJCNN), The 2010 International Joint Conference*. 2010. p. 1–8.
13. Branco P, Torgo L, Ribeiro R. A survey of predictive modeling under imbalanced distributions. *arXiv preprint arXiv:1505.01658*. 2015.
14. López V, Fernandez A, Garcia S, et al. Classification with imbalanced datasets. *Information Sciences*. 2007;250(2013):113–141.
15. Demšar J. Algorithms for subsetting attribute values with Relief. *Machine Learning*. 2010;78(3):421–428.
16. Jia J, Yang N, Zhang C, et al. Object–oriented feature selection of high spatial resolution images using an improved Relief algorithm. *Mathematical and Computer Modelling*. 2013;58(3–4):619–626.
17. Novakovic J. Using information gain attribute evaluation to classify sonar targets. *17th Telecommunications forum TELFOR*. 2009. p. 24–26.
18. Furht B, Escalante A. Handbook of Data Intensive Computing. Springer Science. 2011.
19. Hirsh H, Cohen WW. The learnability of description logics with equality constraints. *Machine Learning*. 1994;17(2–3):169–199.
20. Kohavi R, John GH. Wrappers for feature subset selection. *Artificial Intelligence*. 1997;97(1–2):273–324.
21. Zhang X, Wu G, Dong Z, Crawford C. Embedded feature–selection support vector machine for driving pattern recognition. *Journal of the Franklin Institute*. 2015;352(2):669–685.
22. Huang SH. Supervised feature selection: A tutorial. *Artificial Intelligence Research*. 2015;4(2):22–37.
23. Hamed T, Dara R, Kremer SC. An accurate, fast embedded feature selection for SVMs. *Machine Learning and Applications 13th International Conference*. 2014. p. 135–140.
24. Wahed MA, Wahba K. Data mining based–assistant tools for physicians to diagnose diseases. Circuits and Systems. *IEEE 46th Midwest Symposium*. 2003. p. 388–391.
25. Ozcift A. SVM feature selection based rotation forest ensemble classifiers to improve computer–aided diagnosis of Parkinson disease. *Journal of Medical Systems*. 2012;36(4):2141–2147.
26. Agrawal GL, Gupta H. Optimization of C4. 5 Decision Tree Algorithms for Data Mining Application. International Journal of Emerging Technology and Advanced Engineering. 2013;3(3):341–345.
27. Sharma P, Singh D, Singh A. Classification algorithms on a large continuous random dataset using rapid miner tool. *Electronics and Communication Systems, 2nd International Conference*. 2015. p. 704–709.
28. Kaur G, Chhabra A. Improved J48 Classification Algorithm for the Prediction of Diabetes. *International Journal of Computer Applications*. 2014;98(22):13–22.
29. Almutairi A, Parish D. Using classification techniques for creation of predictive intrusion detection model. *Internet Technology and Secured Transactions, 9th International Conference*. 2014. p. 223–228.
30. Galathiya A, Ganatra A, Bhensdadia C. Classification with an improved Decision Tree Algorithm. *International Journal of Computer Applications*. 2012;46(23):1–6.
31. Jian X, Chen P, Bin L. Random forest for relational classification with
32. application to terrorist profiling. *Granular Computing, IEEE International Conference*. 2009. p. 630–633.
33. Svetnik V, Liaw A, Tong C, et al. Random forest: a classification and regression tool for compound classification and QSAR modeling. *Journal of chemical information and computer sciences*. 2003;43(6):1947–1958.
34. Cuzzocrea A, Francis SL, Gaber MM. An Information–Theoretic Approach for Setting the Optimal Number of Decision Trees in Random Forests. Systems, Man, and Cybernetics. 2013. p. 1013–1019.
35. Chauhan H, Kumar V, Pundir S, et al. A comparative study of classification techniques for intrusion detection. *Computational and Business Intelligence, International Symposium*. 2013. p. 40–43.
36. M Al–khaldy, Kambhampati C. Performance Analysis of Various Missing Value Imputation Methods on Heart Failure Dataset. SAI Intelligent Systems Conference. 2016. p. 415–425.