International Journal of Data Science and Analytics manuscript No.
(will be inserted by the editor)

Resampling Strategies for Imbalanced Time Series Forecasting

Nuno Moniz · Paula Branco · Luís Torgo

Received: date / Accepted: date

Abstract Time series forecasting is a challenging task, where the non-stationary characteristics of data portrays a hard setting for predictive tasks. A common issue is the imbalanced distribution of the target variable, where some values are very important to the user but severely underrepresented. Standard prediction tools focus on the average behaviour of the data. However, the objective is the opposite in many forecasting tasks involving time series: predicting rare values. A common solution to forecasting tasks with imbalanced data is the use of resampling strategies, which operate on the learning data by changing its distribution in favor of a given bias. The objective of this paper is to provide solutions capable of significantly improving the predictive accuracy on rare cases in forecasting tasks using imbalanced time series data. We extend the application of resampling strategies to the time series context and introduce the concept of temporal and relevance bias in the case selection process of such strategies, presenting new proposals. We evaluate the results of standard forecasting tools and the use of resampling strategies, with and without bias over 24 time series data sets from 6 different sources. Results show a significant increase in predictive accuracy on rare cases associated with using resampling strategies, and the use of biased strategies further increases accuracy over non-biased strategies.

Keywords Time Series · Data Imbalance · Resampling Strategies · Temporal Bias

This paper is an extension version of the DSAA'2016 paper "Resampling Strategies for Imbalanced Time Series" [33]

Nuno Moniz
LIAAD - INESC Tec
DCC - Faculdade de Ciências da Universidade do Porto
E-mail: nmmoniz@inescporto.pt

Paula Branco
LIAAD - INESC Tec
DCC - Faculdade de Ciências da Universidade do Porto
E-mail: paobranco@gmail.com

Luís Torgo
LIAAD - INESC Tec
DCC - Faculdade de Ciências da Universidade do Porto
E-mail: ltorgo@dcc.fc.up.pt

1 Introduction

Mining time series data is one of the most challenging problems in data mining [52]. Time series forecasting holds a key importance in many application domains, where time series data is highly imbalanced. This occurs when certain ranges of values are over-represented in comparison to others and the user is particularly interested in the predictive performance on values that are the least represented. Such examples may be found in financial data analysis, intrusion detection in network forensics, oil spill detection and prognosis of machine failures. In these scenarios of imbalanced data sets, standard learning algorithms bias the models toward the more frequent situations, away from the user preference biases, proving to be an ineffective approach and a major source of performance degradation [10].

A common solution for the general problem of mining imbalanced data sets is to resort to resampling strategies. These strategies change the distribution of learning data in order to balance the number of rare and normal cases, attempting to reduce the skewness of the data. Resampling strategies commonly achieve their goal by under or oversampling the data. In the former, some of the cases considered as normal (*i.e.* the majority of cases) are removed from the learning data; in

the latter, cases considered to be rare (*i.e.* the minority) are generated and added to the data. For example, in fraud detection problems, fraud cases are infrequent, and detecting them is the prime objective. Also, in intrusion detection problems, most of the behaviour in networks is normal, and cases of intrusion, which one aims to detect, are scarce. This task of predicting rare occurrences has proven to be a difficult task to solve, but due to its importance in so many domains, it is a fundamental problem within predictive analytics [16].

Resampling strategies are a popular method for dealing with imbalanced domains. This is a simple, intuitive and efficient method for dealing with imbalanced domains. Moreover, it allows the use of any out-of-the-box learner, enabling a diversity of choices at the learning step. An alternative could be to develop special-purpose learning methods, or to act at the post-processing level. Generally, special-purpose learning methods have the advantage of exhibiting a good performance on the problem they were though for. However, they require a thorough knowledge of the learning algorithm manipulated and their application to other problems typically fails. Regarding post-processing methods, they have not been much explored, and usually involve the output of conditional probabilities.

Most existing work using resampling strategies for predictive tasks with an imbalanced target variable distribution involves classification problems ([38, 48, 6, 26]). Recently, efforts have been made to adapt existing strategies to numeric targets, *i.e.* regression problems ([46, 45]). To the best of our knowledge, no previous work addresses this question using resampling strategies in the context of time series forecasting. Although time series forecasting involves numeric predictions, there is a crucial difference compared to regression tasks: the time dependency among the observed values. The main motivation of the current work is our claim that this order dependency should be taken into account when changing the distribution of the training set, *i.e.* when applying resampling. Our work is driven by the hypothesis that by biasing the sampling procedure with information on this order dependency, we are able to improve predictive performance.

In this paper, we study the use of resampling strategies in imbalanced time series. Our endeavour is based on three strategies: *i)* the first is based on undersampling (random undersampling [24]); *ii)* the second is based on oversampling (random oversampling [19]); and *iii)* the third combines undersampling and oversampling (random undersampling with **S**ynthetic **M**inority **O**ver-sampling **TE**chnique [9]). These strategies were initially proposed for classification problems, and were then extended for regression tasks [46, 45, 4]. We will refer to the extension of the SMOTE resampling strategy as SmoteR.

Time series often exhibit systematic changes in the distribution of observed values. These non-stationarities are often known as *concept drift* [51]. This concept describes the changes in the conditional distribution of the target variable in relation to the input features (*i.e.* predictors), whilst the distribution of the latter stays unchanged. This raises the question of how to devise learning approaches capable of coping with this issue. We introduce the concept of temporal bias in resampling strategies associated with forecasting tasks using imbalanced time series. Our motivation is the idea that in an imbalanced time series, where concept drift occurs, it is possible to improve forecasting accuracy by introducing a temporal bias in the case selection process of resampling strategies. This bias favours cases that are within the temporal vicinity of apparent regime changes. In this paper we propose two alternatives for the resampling strategies used in our work: undersampling, oversampling and SmoteR with *1)* temporal bias, and *2)* with temporal and relevance bias.

An extensive experimental evaluation was carried out to evaluate our proposals comprising 24 time series data sets from 6 different sources. The objective is to verify if resampling strategies are capable of improving the predictive accuracy in comparison to standard forecasting tools, including those designed specifically for time series (*e.g.* ARIMA models [8]).

The contributions of this paper are:

- The extension of resampling strategies for time series forecasting tasks;
- The proposal of novel resampling strategies that introduce the concept of temporal and relevance bias;
- An extensive evaluation including standard regression tools, time series specific models and the use of resampling strategies.

The remainder of this paper is structured as follows. In Section 2 the problem tackled in our work is introduced and the hypotheses in which our proposals are based are presented. Resampling strategies are described in Section 3 along with the adaptation of previous proposals, and new proposals. The data used to evaluate the proposals is introduced in Section 4, as well as the regression tools used and the evaluation methods. The evaluation process is described and results presented in Section 5, followed by a discussion in Section 6. Finally, previous work is discussed in Section 7 and conclusions are presented in Section 8.

2 Problem Definition

The main objective of our proposals is to provide solutions that significantly improve the predictive accuracy on relevant (rare) cases in forecasting tasks involving imbalanced time series.

The task of time series forecasting assumes the availability of a time-ordered set of observations of a given continuous variable $y_1, y_2, \ldots, y_t \in Y$, where y_t is the value measured at time t. The objective of this predictive task is to forecast future values of variable Y. The overall assumption is that an unknown function correlates the past and future values of Y, *i.e.* $Y_{t+h} = f(\langle Y_{t-k}, \ldots, Y_{t-1}, Y_t \rangle)$. The goal of the learning process is to provide an approximation of this unknown function. This is carried out using a data set with historic examples of the function mapping (*i.e.* training set).

Time series forecasting models usually assume the existence of a degree of correlation between successive values of the series. A form of modeling this correlation consists of using the previous values of the series as predictors of the future value(s), in a procedure known as time delay embedding [39]. This process allows the use of standard regression tools on time series forecasting tasks. However, specific time series modelling tools also exist, such as the ARIMA models [8].

In this work we focus on imbalanced time series, where certain ranges of values of the target variable Y are more important to the end-user, but severely underrepresented in the training data. As training data we assume a set of cases built using a time delay embedding strategy, *i.e.* where the target variable is the value of Y in the next time step (y_{t+1}) and the predictors are the k recent values of the time series, *i.e.* $y_t, y_{t-1}, \cdots, y_{t-k}$.

To formalise our prediction task, namely in terms of criteria for evaluating the results of modeling approaches, we need to specify what we mean by "more important" values of the target variable. We resort to the work of Ribeiro [36], that proposes the use of a relevance function to map the domain of continuous variables into a $[0, 1]$ scale of relevance, *i.e.* $\phi(Y) : \mathcal{Y} \rightarrow [0, 1]$. Normally, this function is given by the users, attributing levels of importance to ranges of the target variable specific to their interest, taking into consideration the domain of the data. In our work, due to the lack of expert knowledge concerning the domains, we employ an automatic approach to define the relevance function using box plot statistics, detailed in Ribeiro [36], which automatically assigns more relevance/importance to the rare extreme low and high values of the target variable. This automatic approach uses a piecewise cubic Hermite interpolation polynomials [12] (*pchip*) algorithm to interpolate a set of points describing the distribution of the target variable. These points are given by box plot statistics. The outlier values according to box plot statistics (either extreme high or low) are given a maximum relevance of 1 and the median value of the distribution is given a relevance of 0. The relevance of the remaining values is then interpolated using the *pchip* algorithm.

Based on the concept of relevance, Ribeiro [36] has also proposed an evaluation framework that allows us to assert the quality of numeric predictions considering the user bias. We use this evaluation framework to ascertain the predictive accuracy when using imbalanced time series data, by combining standard learning algorithms and resampling strategies.

The hypotheses tested in our experimental evaluation are:

Hypothesis 1 *The use of resampling strategies significantly improves the predictive accuracy of forecasting models on imbalanced time series in comparison to the standard use of out of the box regression tools.*

Hypothesis 2 *The use of bias in case selection of resampling strategies significantly improves the predictive accuracy of forecasting models on imbalanced time series in comparison to non-biased strategies.*

Hypothesis 3 *The use of resampling strategies significantly improves the predictive accuracy of forecasting models on imbalanced time series in comparison to the use of time series specific models.*

From a practical point of view, only time series forecasting tasks with rare important cases may benefit from the proposed approach. Our target applications are forecasting tasks where the user has a preference bias towards the rare values which also motivates the use of specific performance assessment measures that are able to capture what is important to the user. Also the hypotheses tested are only meaningful in the context of time series with imbalanced distributions where the user is more interested in obtaining more accurate predictions on the least represented cases. This means that our proposed approach is not suitable for forecasting tasks whose goal is accurate predictions across the entire domain irrespectively of the errors location.

3 Resampling Strategies

Resampling strategies are pre-processing approaches that change the original data distribution in order to meet some user-given criteria. Among the advantages of preprocessing strategies is the ability of using any standard learning tool. However, to match a change in the data

distribution with the user preferences is not a trivial task. The proposed resampling strategies aim at preprocessing the data for obtaining an increased predictive performance in cases that are scarce and simultaneously important to the user. As mentioned before, this importance is described by a relevance function $\phi(Y)$. Being domain-dependent information, it is the user responsibility to specify the relevance function. Nonetheless, when lacking expert knowledge, it is possible to automatically generate the relevance function. Being a continuous function on the scale $[0, 1]$, we require the user to specify a relevance threshold, t_R, that establishes the minimum relevance score for a certain value of the target variable to be considered relevant. This threshold is only required because the proposed resampling strategies need to be able to decide which values are the most relevant when the distribution changes.

Figure 2 shows an example of an automatically generated relevance function, with a 0.9 relevance threshold, defined for the Temperature time series (Figure 1) obtained from the Bike Sharing data source [14] using observations between 22 March and 1 May 2011. In this example, we assign more importance to the highest and lowest values of Y.

Our resampling strategies proposals for imbalanced time series data are based on the concept of relevance bins. These are successive observations of the time series where the observed value is either relevant or irrelevant, for the user. Algorithm 1 describes how these bins are created from the original time series. The algorithm uses time stamp information and the relevance of the values from the original time series, to cluster the observations into bins that have the following properties:

1. Each bin contains observations whose target variable value has a relevance score that is either all above or all below the relevance threshold t_R; and
2. Observations in a given bin are always consecutive cases in terms of the time stamp.

Figure 3 shows the bins obtained in the Temperature time series displayed in Figure 1. The six dashed rectangles represent the bins containing consecutive observations with relevant value of the target variable, while the non-dashed regions correspond to consecutive observations with common values with a lower relevance to the user, based on the automatically generated relevance function (Figure 2). This means that, for the example under consideration, we have 13 bins: 6 bins with relevant values, and 7 bins with common values (non-dashed areas).

Our first proposals are an adaption to the time series context of the random undersampling, random oversampling and SmoteR strategies proposed by Torgo et al. [46] and Branco [4] for tackling imbalanced regression tasks. The main change applied in both algorithms is the way the sampling is carried out. Instead of pure random selection as in the original algorithms, here we carry out sampling within each individual bin.

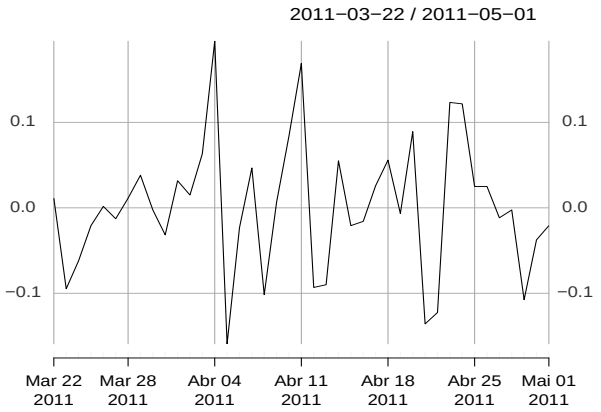

Fig. 1 Sample of Temperature time series from the Bike Sharing data source [14].

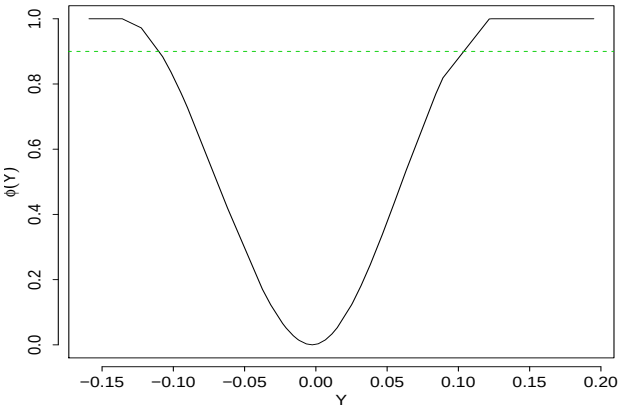

Fig. 2 Relevance function $\phi(Y)$ with a relevance threshold of 0.9 (dashed line) for the time series showed in Figure 1

The random undersampling (**U_B**) strategy is described in Algorithm 2. This approach has the default behaviour of balancing the number of normal and rare values by randomly removing examples from the bins with normal cases, *i.e.*, bins with low relevance examples. In this case, the number of examples removed is automatically calculated to ensure that: 1) each undersampled bin gets the same number of normal cases; and 2) the total number of normal and rare cases are balanced. The algorithm also allows the specification of a particular undersampling percentage through the parameter u. When the user sets this percentage, the number of cases removed is calculated for each bin with normal values. The percentage $u < 1$ defines the number of examples that are maintained in each bin.

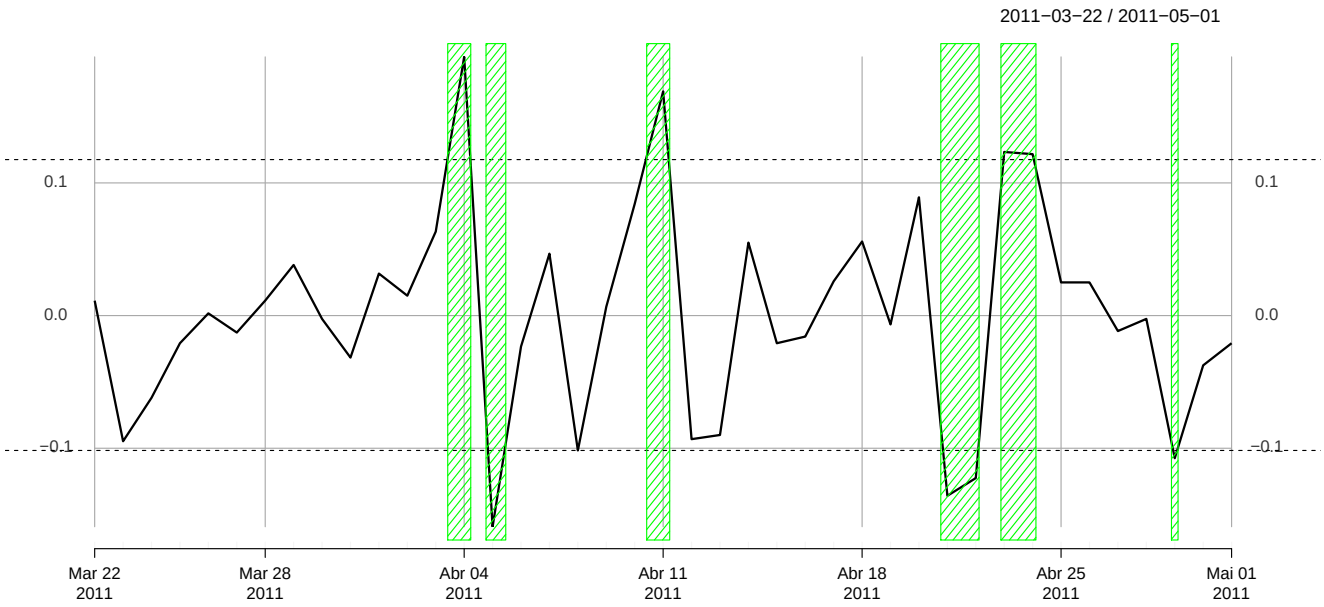

Fig. 3 Bins generated for time series of Figure 1 with relevance function ($\phi()$) provided in Figure 2 using a relevance threshold of 0.9 (dashed ranges represent bins with important cases).

Algorithm 1 Algorithm for the construction of Bins.

function BINSCONSTRUCTOR($\mathcal{D}, y, \phi(y), t_R$)
  // $\mathcal{D}$ - A data set
  // y - The target variable
  // $\phi(y)$ - User specified relevance function
  // t_R - The threshold for relevance on y values

  $p \leftarrow 1$
  for $i \leftarrow 1$ **to** NROW($\mathcal{D}$) **do** // Collect examples into the bins based on $\phi()$
    $Bins_p \leftarrow Bins_p \bigcup \{\langle \mathbf{x_i}, y_i \rangle \in \mathcal{D}\}$
    if $\phi(y_i) \leq t_R < \phi(y_{i+1}) \vee \phi(y_i) > t_R \geq \phi(y_{i+1})$ **then**
      $p \leftarrow p + 1$
    end if
  end for
  return Bins
end function

Our second proposal is the random oversampling (**O_B**) approach that is described in Algorithm 3. In this strategy, the default behaviour is to balance the number of normal and rare cases with the introduction of replicas of the most relevant and rare cases in the bins containing examples with high relevance. The number of copies included is automatically determined to ensure: 1) balance between rare and normal cases and 2) the same frequency in the oversampled bins. An optional parameter o allows the user to select a specific percentage of oversampling to apply in each bin with relevant values.

Algorithm 2 The Random Undersampling algorithm (**U_B**).

function RANDUNDER($\mathcal{D}, y, \phi(y), t_R, u$)
  // $\mathcal{D}$ - A data set
  // y - The target variable
  // $\phi(y)$ - User specified relevance function
  // t_R - The threshold for relevance on y values
  // u - (optional parameter) Percentage of undersampling

  $Bins \leftarrow$ BINSCONSTRUCTOR($\mathcal{D}, y, t_R$)
  $BinsU \leftarrow \{Bins_i : \forall (x,y) \in Bins_i \phi(y) < t_R\}$ // Bins where undersampling will be applied
  $newData \leftarrow Bins \backslash BinsU$
  for each $B \in BinsU$ **do**
    if u **then**
      $TgtNr \leftarrow |B| \times u$
    else
      $tgtNr \leftarrow \frac{\text{Nr. examples in } Bins \backslash BinsU}{\text{Nr of } BinsU}$
    end if
    $selNormCases \leftarrow$ SAMPLE($tgtNr, B$) // randomly select a number of normal cases from bin B
    $newData \leftarrow c(newData, selNormCases)$ // add the normal cases to the new data set
  end for
  return $newData$
end function

The third strategy (**SM_B**) is an adaptation of the SmoteR algorithm to the time series context. The SmoteR algorithm combines random undersampling with oversampling through the generation of synthetic cases. The default behaviour of this strategy is to automatically balance the number of examples in the bins. The ran-

Algorithm 3 The Random Oversampling algorithm (**O_B**).

function RandOver($\mathcal{D}, y, \phi(y), t_R, o$)
  // $\mathcal{D}$ - A data set
  // y - The target variable
  // $\phi(y)$ - User specified relevance function
  // t_R - The threshold for relevance on y values
  // o - (optional parameter) Percentage of oversampling

  $Bins \leftarrow$ BinsConstructor($\mathcal{D}, y, t_R$)
  $BinsO \leftarrow \{Bins_i : \forall (x,y) \in Bins_i \phi(y) \geq t_R\}$ // Bins where oversampling will be applied
  $newData \leftarrow Bins$
  for each $B \in BinsO$ **do**
    if o **then**
      $tgtNr \leftarrow |B| \times o$
    else
      $tgtNr \leftarrow \frac{\text{Nr. examples in } Bins \setminus BinsO}{\text{Nr of } BinsO}$ // Target nr of elements in each $BinsO$
    end if
    $selRareCases \leftarrow$ sample($tgtNr, B$) // randomly select a number of rare cases from bin B
    $newData \leftarrow c(newData, selRareCases)$ // add the rare cases replicas to the new data set
  end for
  return $newData$
end function

Algorithm 4 Generating synthetic cases.

function genSynthCases($\mathcal{D}, ng, k$)
  // $\mathcal{D}$ - A data set
  // ng - Number of synthetic cases to generate for each existing case
  // k - The number of neighbours used in case generation

  $newCases \leftarrow \{\}$
  for all $case \in \mathcal{D}$ **do**
    if $|\mathcal{D} \setminus \{case\}| < k$ **then** // Less examples than number of neighbours required
      $nns \leftarrow$ kNN($|\mathcal{D} \setminus \{case\}|, case, \mathcal{D} \setminus \{case\}$)
    else
      $nns \leftarrow$ kNN($k, case, \mathcal{D} \setminus \{case\}$) // k-Nearest Neighbours of $case$
    end if
    for $i \leftarrow 1$ **to** ng **do**
      $x \leftarrow$ randomly choose one of the nns
      for all $a \in$ attributes **do** // Generate attribute values
        $diff \leftarrow case[a] - x[a]$
        $new[a] \leftarrow case[a] + \text{random}(0,1) \times diff$
      end for
      $d_1 \leftarrow$ dist($new, case$) // Decide the target value
      $d_2 \leftarrow$ dist(new, x)
      $new[Target] \leftarrow \frac{d_2 \times case[Target] + d_1 \times x[Target]}{d_1 + d_2}$
      $newCases \leftarrow newCases \bigcup \{new\}$
    end for
  end for
  return $newCases$
end function

dom undersampling part is carried out through the process described in Algorithm 2. The oversampling strategy generates new synthetic cases by interpolating a seed example with one of its k-nearest neighbours from the respective bin of rare examples. The main difference between **SM_B** and the original SmoteR algorithm is on the process used to select the cases for both under- and over-sampling. **SM_B** works with time series data and thus it must take the time ordering of the cases into account, which we have done by defining the relevance bins that are formed by subsets of cases that are adjacent in terms of time.

Algorithm 4 shows the process for generating synthetic examples and Algorithm 5 describes the **SM_B** algorithm. This algorithm by default balances the cases in the bins. Alternatively, the user may set the percentages of under/oversampling to be applied in the bins using parameters u and o. These are optional parameters that allow the user to completely control the percentages applied.

3.1 Resampling with Temporal Bias

Concept drift is one of the main challenges in time series forecasting. This is particularly true for our target applications where the preference bias of the user concerns rare values of the series. In effect, this rarity makes it even more important to understand and anticipate when these shifts of regime occur.

A first step in the identification of these different regimes according to user preferences is implemented by the previously described creation of relevance bins using Algorithm 1 (c.f. Figure 3). Still, within each bin the cases are not equally relevant. We claim that the most recent cases within each bin may potentially contain important information for understanding these changes in regime. In this context, we propose three new algorithms (Undersampling, Oversampling and SmoteR with Temporal Bias) that favour the selection of training cases that are in the vicinity of transitions between bins. This resembles the adaptive learning notion of gradual forgetting, where the older cases have a higher likelihood of being excluded from the learning data. However, this concept is applied to the full extent of the data and in our proposal of temporal bias it is applied in each bin of normal cases.

The Undersampling with Temporal Bias (**U_T**) proposal is based on Algorithm 2. The main difference is the process of selecting examples to undersample within each bin of normal cases. Instead of randomly selecting cases, we use a biased undersampling procedure. In **U_T**, for each bin where undersampling is applied,

Algorithm 5 The main SmoteR algorithm (**SM_B**).

1: **function** SMOTER($\mathcal{D}, y, \phi(y), t_R, k, o, u$)
2: // $\mathcal{D}$ - A data set
3: // y - The target variable
4: // $\phi(y)$ - User specified relevance function
5: // t_R - The threshold for relevance on y values
6: // k - The number of neighbours used in case generation
7: // o, u - (optional parameters) Percentages of over and undersampling
8:
9: $Bins \leftarrow$ BINSCONSTRUCTOR($\mathcal{D}, y, t_R$)
10: $BinsU \leftarrow \{Bins_i : \forall(x,y) \in Bins_i \phi(y) \leq t_R\}$ // Bins where undersampling will be applied
11: $BinsO \leftarrow \{Bins_i : \forall(x,y) \in Bins_i \phi(y) \geq t_R\}$ // Bins where oversampling will be applied
12: $newData \leftarrow \{\}$
13: **for each** $B \in BinsU$ **do** // Apply undersampling
14: **if** u **then**
15: $TgtNr \leftarrow |B| \times u$
16: **else**
17: $tgtNr \leftarrow \frac{\text{Nr of examples in } \mathcal{D}}{\text{Nr of } Bins}$
18: **end if**
19: $selNormCases \leftarrow$ SAMPLE($tgtNr, B$)
20: $newData \leftarrow newData \bigcup selNormCases$
21: **end for**
22: **for each** $B \in BinsO$ **do** // Generate synthetic examples
23: **if** o **then**
24: $TgtNr \leftarrow |B| \times o$
25: **else**
26: $tgtNr \leftarrow \frac{\text{Nr of examples in } \mathcal{D}}{\text{Nr of } Bins}$
27: **end if**
28: $synthCases \leftarrow$ GENSYNTHCASES($B, tgtNr - |B|, k$)
29: $newData \leftarrow newData \bigcup synthCases \bigcup B$
30: **end for**
31: **return** $newData$
32: **end function**

Algorithm 6 The Undersampling with Temporal Bias algorithm (**U_T**).

1: **function** UNDERT($\mathcal{D}, y, \phi(y), t_R, u$)
2: // $\mathcal{D}$ - A data set
3: // y - The target variable
4: // $\phi(y)$ - User specified relevance function
5: // t_R - The threshold for relevance on y values
6: // u - (optional parameter) Percentage of undersampling
7: ...
11: $OrdB \leftarrow$ order B by increasing time
12: $prefs \leftarrow c(\frac{1}{|OrdB|}, \frac{2}{|OrdB|}, \cdots, 1)$ // Define higher preferences for most recent cases
13: $selNormCases \leftarrow$ SAMPLE($tgtNr, OrdB, prefs$) // sample normal cases from bin B based on $prefs$
...
14: **end function**

Algorithm 7 The oversampling with Temporal Bias algorithm (**O_T**).

1: **function** OVERT($\mathcal{D}, y, \phi(y), t_R, o$)
2: // $\mathcal{D}$ - A data set
3: // y - The target variable
4: // $\phi(y)$ - User specified relevance function
5: // t_R - The threshold for relevance on y values
6: // o - (optional parameter) Percentage of oversampling
7: ...
11: $OrdB \leftarrow$ order B by increasing time
12: $prefs \leftarrow c(\frac{1}{|OrdB|}, \frac{2}{|OrdB|}, \cdots, 1)$ // Define higher preferences for most recent rare cases
13: $selRareCases \leftarrow$ SAMPLE($tgtNr, OrdB, prefs$) // sample rare cases from bin B based on $prefs$
...
14: **end function**

the older the example is, the lower the probability of being selected for the new training set. This provides a modified distribution which is balanced in terms of normal and rare cases with a probabilistic preference towards the most recent cases, *i.e.* those in the vicinity of bin transitions. The integration of the temporal bias is performed as follows:

- order the cases in each bin B of normal cases by increasing time in a new bin $OrdB$;
- assign the preference of $i \times \frac{1}{|OrdB|}$ for selecting ex_i in $OrdB$, where $i \in (1, \ldots, |OrdB|)$;
- select a sample from $OrdB$ based on the former preferences.

This corresponds to substituting line 17 in Algorithm 2 by the lines 11, 12 and 13 previously presented in Algorithm 6.

Our second proposed strategy, oversampling with temporal bias (**O_T**), is based in Algorithm 3. This strategy performs oversampling giving an higher preference to the most recent examples. This way, the strategy incorporates a bias towards the newer cases in the replicas selected for inclusion. The integration of the temporal bias is achieved as follows:

- order the cases in each bin B of rare cases by increasing time in a new bin $OrdB$;
- assign the preference of $i \times \frac{1}{|OrdB|}$ for selecting ex_i in $OrdB$, where $i \in (1, \ldots, |OrdB|)$;
- select a sample from $OrdB$ based on the former preferences.

This corresponds to replacing line 17 in Algorithm 3 by the lines 11, 12 and 13 presented in Algorithm 7.

Our third proposed strategy is SmoteR with Temporal Bias (**SM_T**). This approach combines undersampling with temporal bias in the bins containing normal cases, with an oversampling mechanism that also integrates a temporal component. The undersampling with temporal bias strategy is the same as described in Algorithm 6. Regarding the oversampling strategy, we included in the SmoteR generation of synthetic examples a preference for the most recent examples. This means that when generating a new synthetic case, after evaluating the k-nearest neighbours of the seed example, the neighbour selected for the interpolation process is the most recent case. This includes, in the synthetic cases

generation, a time bias towards the most recent examples instead of randomly selecting cases. Algorithm 8 shows the lines that were changed in Algorithm 5. To include the temporal bias we have replaced line 19 in Algorithm 5 referring to the undersampling step, by lines 12, 13 and 14 in Algorithm 8. Also, concerning the oversampling step, we replaced line 28 in Algorithm 5 by line 15 in Algorithm 8.

Regarding the function for generating synthetic examples, Algorithm 9 describes what was necessary to change in Algorithm 4 for including the temporal bias. In this case, only line 13 of Algorithm 4 was changed, in order to consider the time factor, so that the nearest neighbour is not randomly selected.

Algorithm 8 The SmoteR with temporal bias algorithm (**SM_T**).

1: **function** SmoteRT($\mathcal{D}, y, \phi(y), t_R, k, o, u$)
2: // $\mathcal{D}$ - A data set
3: // y - The target variable
4: // $\phi(y)$ - User specified relevance function
5: // t_R - The threshold for relevance on y values
6: // k - The number of neighbours used in case generation
7: // o, u - (optional parameters) Percentages of over and undersampling
8: $\cdots$
12: $OrdB \leftarrow$ order B by increasing time
13: $prefs \leftarrow c(\frac{1}{|OrdB|}, \frac{2}{|OrdB|}, \cdots, 1)$
14: $selNormCases \leftarrow$ SAMPLE$(tgtNr, OrdB, prefs)$
$\cdots$
15: $synthCases \leftarrow$ genSynthCasesT$(B, tgtNr, k)$
$\cdots$
16: **end function**

Algorithm 9 Generating synthetic cases with temporal bias.

1: **function** genSynthCasesT($\mathcal{D}, ng, k$)
2: // $\mathcal{D}$ - A data set
3: // ng - Number of synthetic cases to generate for each existing case
4: // k - The number of neighbours used in case generation
$\cdots$
13: $x \leftarrow$ choose the nns most recent in time
$\cdots$
14: **end function**

3.2 Resampling with Temporal and Relevance Bias

This section describes our final proposals of resampling strategies for imbalanced time series forecasting. The idea of the three algorithms described in this section is to also include the relevance scores in the sampling bias. The motivation is that while we assume that the most recent cases within each bin are important as they precede regime changes, we consider that older cases that are highly relevant should not be completely disregarded given the user preferences. To combine the temporal and relevance bias we propose three new algorithms: undersampling (Algorithm 10), oversampling (Algorithm 11) and SmoteR with temporal and relevance bias (Algorithm 12).

The integration of temporal and relevance bias in undersampling (**U_TPhi**) is performed as follows:

- order examples in each bin B of normal cases by increasing time in a new bin $OrdB$;
- for each example ex_i in $OrdB$ use $\frac{i}{|OrdB|} \times \phi(ex_i[y])$ as the preference of selecting example ex_i;
- sample a number of examples from $OrdB$ assuming the previously determined preferences.

This process corresponds to replacing the line 17 in Algorithm 2 by the lines 11, 12 and 13 in Algorithm 10.

Algorithm 10 The Undersampling with Temporal and Relevance Bias algorithm (**U_TPhi**).

1: **function** UnderTPhi($\mathcal{D}, y, \phi(y), t_R, u$)
2: // $\mathcal{D}$ - A data set
3: // y - The target variable
4: // $\phi(y)$ - User specified relevance function
5: // t_R - The threshold for relevance on y values
6: // u - (optional parameter) Percentage of undersampling
7: $\cdots$
11: $OrdB \leftarrow$ order B by increasing time
12: $prefs \leftarrow c(\frac{1}{|OrdB|} \times \phi(y_1), \frac{2}{|OrdB|} \times \phi(y_2), \cdots, \phi(y_{|OrdB|}))$ // Preferences based on time and relevance
13: $selNormCases \leftarrow$ SAMPLE$(tgtNr, OrdB, prefs)$ // sample normal cases from bin B based on $prefs$
$\cdots$
14: **end function**

In order to incorporate a temporal and relevance bias in the oversampling algorithm (**O_TPhi**) the following steps were necessary:

- order examples in each bin B of rare cases by increasing time in a new bin $OrdB$;
- for each example ex_i in $OrdB$ use $\frac{i}{|OrdB|} \times \phi(ex_i[y])$ as the preference of selecting example ex_i;
- sample a number of examples from $OrdB$ assuming the above preferences.

This corresponds to replacing line 17 in Algorithm 3 by lines 11, 12 and 13 in Algorithm 11. These changes allow to bias oversampling procedures towards recent cases of high relevance.

The same integration of time and relevance bias is also done in the SmoteR algorithm. In this case, we altered both the undersampling and oversampling steps

Algorithm 11 The oversampling with Temporal and Relevance Bias algorithm (**O_TPhi**).

```
1: function OverTPhi(D, y, φ(y), t_R, o)
2:     // D - A data set
3:     // y - The target variable
4:     // φ(y) - User specified relevance function
5:     // t_R - The threshold for relevance on y values
6:     // o - (optional parameter) Percentage of oversampling
7: ...
11:    OrdB ← order B by increasing time
12:    prefs ← c(1/|OrdB| × φ(y_1), 2/|OrdB| × φ(y_2), ..., φ(y_|OrdB|)) // Preferences based on time and relevance
13:    selRareCases ← sample(tgtNr, OrdB, prefs) // sample rare cases from bin B based on prefs
       ...
14: end function
```

Algorithm 12 The SmoteR with temporal and relevance bias algorithm (**SM_TPhi**).

```
1: function SmoteRTPhi(D, y, φ(y), t_R, k, o, u)
2:     // D - A data set
3:     // y - The target variable
4:     // φ(y) - User specified relevance function
5:     // t_R - The threshold for relevance on y values
6:     // k - The number of neighbours used in case generation
7:     // o, u - (optional parameters) Percentages of over and undersampling
8: ...
12:    OrdB ← order B by increasing time
13:    prefs ← c(1/|OrdB| × φ(y_1), 2/|OrdB| × φ(y_2), ..., φ(y_|OrdB|))
14:    selNormCases ← sample(tgtNr, OrdB, prefs)
       ...
15:    synthCases ← genSynthCasesTPhi(B, tgtNr, k, φ(y))
       ...
16: end function
```

Algorithm 13 Generating synthetic cases with temporal and relevance bias.

```
1: function genSynthCasesTPhi(D, ng, k, φ(y))
2:     // D - A data set
3:     // ng - Number of synthetic cases to generate for each existing case
4:     // k - The number of neighbours used in case generation
5:     // φ(y) - User specified relevance function
       ...
13:    y.rel ← φ(nns[Target])  // relevance value of nns
14:    y.time ← time position of nns sorted by ascending order normalized to [0, 1]
15:    x ← argmax_{neig∈nns} y.rel(neig) × y.time(neig)
       ...
16: end function
```

of SmoteR algorithm. Algorithm 12 shows what was changed in Algorithm 5 to accomplish this. Lines 19 and 28 of Algorithm 5 were replaced by lines 12, 13 and 14, and by line 15 in Algorithm 12, respectively. These changes correspond to biasing the undersampling process to consider time and relevance of the examples in each bin, as previously described: the most recent examples with higher relevance are preferred to others for staying in the changed data set. Regarding the oversampling strategy, the generation of synthetic examples also assumes this tendency, *i.e.*, the new examples are built using the function GenSynthCasesTPhi(), by prioritising the selection of highly relevant and recent examples. Algorithm 13 shows the changes made in Algorithm 4 (line 13 in Algorithm 4 was replaced by lines 13, 14 and 15). The bias towards more recent and high relevance examples is achieved in the selection of a nearest neighbour for the interpolation, as follows:

- calculate the relevance of the k-nearest neighbours;
- calculate the time position of k-nearest neighbours by ascending order and normalized to $[0, 1]$;
- select the nearest neighbour with the highest value of the product of relevance by time position.

These changes bias the undersampling and the generation of new cases of **SmoteR** algorithm towards the most recent and relevant cases.

In summary, for each of the three resampling strategies considered (random undersampling, random oversampling and SmoteR), we have proposed three new variants that try to incorporate some form of sampling bias that we hypothesize as being advantageous in terms of forecasting accuracy on imbalanced time series tasks where the user favours the performance on rare values of the series. The first variants (**U_B**, **O_B** and **SM_B**) carry out sampling within relevance bins that are obtained with the goal of including successive cases with similar relevance according to the user preference. The second variants (**U_T**, **O_T** and **SM_T**) add to the first variant a preference toward the most recent cases within each bin as these are the cases that precede regime transitions. Finally, the third variants (**U_TPhi**, **O_TPhi** and **SM_TPhi**) add a third preference to the sampling procedures, to also include the relevance scores of the cases and avoid discarding cases that may not be the most recent, but are the most relevant for the user.

4 Materials and Methods

4.1 Data

The experiments described in this paper use data from 6 different sources, totaling 24 time series from diverse real-world domains. For the purposes of evaluation we assumed that each time series is independent from others of the same source (*i.e.* we did not use the temperature time series data in the Bike Sharing source to predict the count of bike rentals). All proposed re-

sampling strategies, in combination with each of the regression tools, are tested on these 24 time series which are detailed in Table 1. All of the time series were pre-processed to overcome some well-known issues with this type of data, as is non-available (NA) observations. To resolve issues of this type, we resorted to the imputation of values using the **R** function **knnImputation** of the package **DMwR** [42]. For each of these time series we applied the previously described approach of the time delay coordinate embedding. It requires an essential parameter: how many values to include as recent values, *i.e.* the size of the embed, k. This is not a trivial task as it requires to try different values of embed size in order to decide on an acceptable value. In our experiments we have used $k = 10$. Experiments with a few other values have not shown significant differences in results. The outcome of the application of this embedding approach produces the data sets used as learning data.

For each of these data sets we need to decide which are the relevant ranges of the time series values. To this purpose, we use a relevance function. As previously mentioned, due to the lack of expert knowledge concerning the used domains, we resort to an automatic approach to define the relevance function, detailed in Ribeiro [36]. This approach uses box plot statistics to derive a relevance function that assigns higher relevance scores to values that are unusually high or low, *i.e.* extreme and rare values. We use this process to obtain the relevance functions for all our time series. An example of the application of this approach, where only high extreme values exist (from a data set on water consumption in the area of Rotunda AEP in the city of Porto), is depicted in Figure 4, while in Figure 2 a case with both type of extremes is shown. Having defined the relevance functions we still need to set a threshold on the relevance scores above which a value is considered important, *i.e.*, the relevance threshold t_R. The definition of this parameter is domain dependent. Still, we have used a relevance threshold t_R of 0.9, which generally leads to a small percentage of the values to be considered important. In Table 1 we added an indication concerning the proportion of rare cases (both very high and low values) for each used data set.

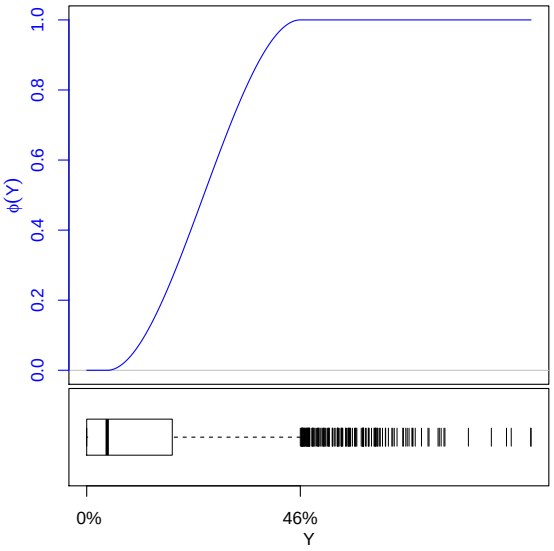

Fig. 4 Relevance function $\phi()$ with high extreme values and box plot of Y distribution.

4.2 Regression Algorithms

To test our hypotheses we selected a diverse set of standard regression tools. Our goal is to verify that our conclusions are not biased by a particular tool.

Table 2 shows the regression methods used in our experiments. To ensure that our work is easily replicable we used the implementations of these tools available in the free and open source **R** environment. Concerning the parameter settings for each of these regression methods, we carried out a preliminary test to search for the optimal parametrization (*i.e.* the setting that obtains the best possible results within a certain set of values of the parameters). The search for optimal parameters was carried out for each combination regression method - dataset, and the results are detailed in Annex A. In addition to these standard regression tools, we also include two time series specific forecasting approaches: *i)* the ARIMA model [8] and *ii)* a bagging approach proposed by Oliveira and Torgo [34]. Regarding the first, ARIMA models require a significant tuning effort in terms of parameters. To tackle this issue, we used the **auto.arima** function available in the R package **forecast** [17], which implements an automatic search method for the optimal parameter settings. The second describes a bagging approach for time series forecasting tasks using bagged regression trees, proposed by Oliveira and Torgo [34]. The authors discuss the difficulties in optimizing the size of the embed (w.r.t. time delay embedding [39]) and propose the use of ensembles with models using different values for embed size. The authors report best results using ensembles where a third of the models use the maximum embed k_{max}, another third uses an embed of $k_{max}/2$ and the last third uses $k_{max}/4$. Additionally, all models within the ensemble use the mean and variance of the respective embed as extra features. This approach will be henceforth referred to as **BDES**.

Table 1 Description of the data sets used.

ID	Time Series	Data Source	Granularity	Characteristics	% Rare
DS1	Temperature	Bike Sharing [14]	Daily	From 01/01/2011 to 31/12/2012 (731 values)	9.9%
DS2	Humidity				9.3%
DS3	Windspeed				7.8%
DS4	Count of Bike Rentals				13.3%
DS5	Temperature		Hourly	From 01/01/2011 to 31/12/2012 (7379 values)	3.5%
DS6	Humidity				4.8%
DS7	Windspeed				12.5%
DS8	Count of Bike Rentals				17.6%
DS9	Flow of Vatnsdalsa River	Icelandic River [41]	Daily	From 01/01/1972 to 31/12/1974 (1095 values)	21.1%
DS10	Minimum Temperature	Porto weather[1]	Daily	From 01/01/2010 to 28/12/2013 (1457 values)	4.8%
DS11	Maximum Temperature				13.3%
DS12	Maximum Steady Wind				11%
DS13	Maximum Wind Gust				11.1%
DS14	SP	Istanbul Stock Exchange [1]	Daily	From 05/01/2009 to 22/02/2011 (536 values)	16.3%
DS15	DAX				11.4%
DS16	FTSE				9.7%
DS17	NIKKEI				11.6%
DS18	BOVESPA				10.1%
DS19	EU				8.2%
DS20	Emerging Markets				6.8%
DS21	Total Demand	Australian electricity load [23]	Half-Hourly	From 01/01/1999 to 01/09/2012 (239602 values)	1.8%
DS22	Recommended Retail Price				10.2%
DS23	Pedrouços	Water Consumption of Oporto[2]	Half-Hourly	From 06/02/2013 to 11/01/2016 (51208 values)	0.08%
DS24	Rotunda AEP				3.4%

[1] Source: Freemeteo `http://freemeteo.com.pt/`
[2] Source: Águas do Douro e Paiva `http://addp.pt/`

Table 2 Regression algorithms and respective R packages

ID	Method	R package
LM	Multiple linear regression	stats [35]
SVM	Support vector machines	e1071 [31]
MARS	Multivariate adaptive regression splines	earth [32]
RF	Random forests	randomForest [27]
RPART	Regression Trees	rpart [40]

4.3 Evaluation Metrics

When the interest of the user is predictive performance at a small proportion of cases (*i.e.* rare cases), the use of standard performance metrics will lead to biased conclusions [36]. In effect, standard metrics focus on the "average" behaviour of the prediction models and for the tasks addressed in this paper, the user goal is a small proportion of cases. Although most of the previous studies on this type of issues are focused on classification tasks, Torgo and Ribeiro [44, 36] have shown that the same problems arise on numeric prediction tasks when using standard metrics, such as Mean Squared Error.

In this context, we will base our evaluation on the utility-based regression framework proposed in the work by Torgo and Ribeiro [44, 36] which also assumes the existence of a relevance function ϕ, as the one previously described. Using this approach and the user-provided relevance threshold, the authors defined a series of metrics that focus the evaluation of models on the cases that the user is interested. In our experiments we used the value 0.9 as relevance threshold.

In our evaluation process we mainly rely on the utility-based regression metric F1-Score, denoted as $F1_\phi$. It integrates the precision and recall measures proposed by the mentioned framework of Ribeiro [36] and extended by Branco et al. [3]. In this context, precision, recall and F1-Score are defined as:

$$prec_\phi = \frac{\sum_{\phi(\hat{y}_i)>t_R}(1+u(\hat{y}_i,y_i))}{\sum_{\phi(\hat{y}_i)>t_R}(1+\phi(\hat{y}_i))} \quad (1)$$

$$rec_\phi = \frac{\sum_{\phi(y_i)>t_R}(1+u(\hat{y}_i,y_i))}{\sum_{\phi(y_i)>t_R}(1+\phi(y_i))} \quad (2)$$

$$F1_\phi = 2 \times \frac{prec_\phi \times rec_\phi}{prec_\phi + rec_\phi} \quad (3)$$

where $\phi(y_i)$ is the relevance associated with the true value y_i, $\phi(\hat{y}_i)$ is the relevance of the predicted value $\hat{y}_i$, t_R is the user-defined threshold signalling the cases

that are relevant for the user, and $u(\hat{y}_i, y_i)$ is the utility of making the prediction $\hat{y}_i$ for the true value y_i, normalized to $[-1, 1]$.

Utility is commonly referred to as being a function combining positive benefits and negative benefits (costs). In this paper, we use the approach for utility surfaces by Ribeiro [36]. Differently from classification tasks, utility is interpreted as a continuous version of the benefit matrix proposed by Elkan [13]. Coarsely, utility U is defined as the difference between benefits B and costs C, $U = B - C$. To calculate utility two factors are taken into consideration: *i)* if the true and predicted values and their respective relevance belong to similar relevance bins (*e.g.* both values are high extremes and highly relevant); and *ii)* that the prediction is reasonably accurate, given a factor of maximum admissible loss, defined by the author. Figures 5 and 6 illustrate the utility surfaces given by the approach of Ribeiro [36] for the relevance functions presented in Figures 2 and 4, where the former has both high and low extreme values, and the latter only has high extreme values.

In Figure 5, we observe that, for the accurate predictions (on the diagonal), the utility values range between 0 and 1. The higher utility values are given to both extremes (low and high) of the target variable. Outside the diagonal, we have an error that must also be taken into account. Predictions reasonably close to the true values have a positive utility. However, as the predicted and true values increase its distance, also the utility becomes negative, tending to -1. Figure 6 shows a similar setting with only one type of extremes: extreme high values.

5 Experimental Evaluation

This section presents the results of our experimental evaluation on three sets of experiments concerning forecasting tasks with imbalanced time series data sets. Each of these experiments was designed with the objective of testing the hypothesis set forth in Section 2. In the first set we evaluate the predictive accuracy of standard regression tools in combination with the proposed resampling strategies. In the second set of experiments, the evaluation is focused on the task of inferring the possibility of the biased resampling strategies over-performing the non-biased strategies. Finally, in the third set, we evaluate the hypothesis of enabling a better predictive performance of models using standard regression tools with resampling strategies over time series specific forecasting approaches such as ARIMA and BDES models. These models and all of the proposed resampling strategies combined with each of the standard regression tools were tested on 24 real-world time series data sets, obtained from six different data sources described in Table 1. In every application of the proposed resampling strategies, an inference method is applied in order to set the parameters concerning the amount of undersampling and oversampling. The objective of this method is to balance the number of normal and relevant cases in order to have an equal number of both in the training data.

The evaluation process is based on the evaluation metric $F1_\phi$, as described by the referred utility-based regression framework (see Section 4.3). Concerning the testing of our hypothesis, we resort to paired comparisons using Wilcoxon signed rank tests in order to infer the statistical significance (with $p-\text{value} < 0.05$) of the paired differences in the outcome of the approaches.

Concerning evaluation algorithms, caution is required in the decision on how to obtain reliable estimates of the evaluation metrics. Since time series data are temporally ordered, we must ensure that the original order of the cases is maintained as to guarantee that prediction models are trained with past data and tested with future data, thus avoiding over-fitting and over-estimated scores. As such, we rely on Monte Carlo estimates as the chosen experimental methodology for our evaluation. This methodology selects a set of random points in the data. For each of these points a past window is selected as training data (Tr) and a subsequent window as test data (Ts). This methodology guarantees that each method used in our forecasting task is evaluated using the same training and test sets, thus ensuring a fair pairwise comparison of the estimates obtained. In our evaluation 50 repetitions of the Monte Carlo estimation process are carried out for each data set with 50% of the cases used as training set and the subsequent 25% used as test set. Exceptionally, due to their size, in the case of the data sets *DS21* and *DS22* we used 10% of the cases as training set and the following 5% as test set, and 20% of the cases as training set and the following 10% as test set for data sets *DS23* and *DS24*. This process is carried out using the infrastructure provided by the R package **performanceEstimation** [43].

In order to clarify the nomenclature associated with the standard regression tools used in this evaluation process, the experiments include results given by multiple linear regression (**LM**), support vector machine (**SVM**), multivariate adaptive regression splines (**MARS**), random forest (**RF**) and regression trees (**RPART**) models. As for the resampling strategies, we use random undersampling (**U_B**), random oversampling (**O_B**), SmoteR (**SM_B**), undersampling (**U_T**), oversampling (**O_T**) and SmoteR (**SM_T**) with temporal bias, and undersampling (**U_TPhi**), oversampling (**O_TPhi**) and

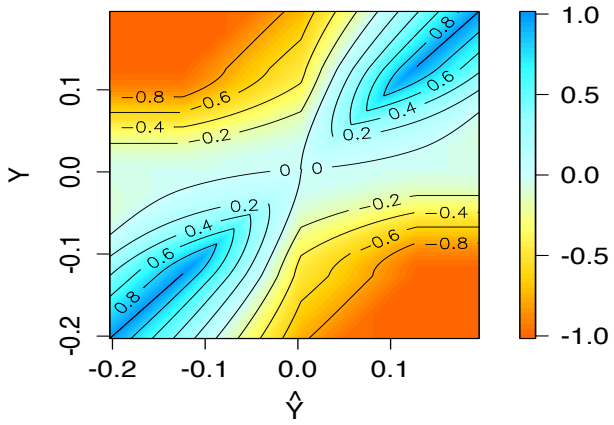

Fig. 5 Utility surface for the relevance function depicted in Figure 2.

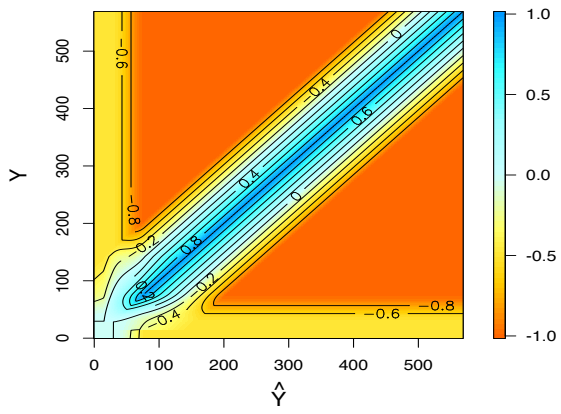

Fig. 6 Utility surface for the relevance function depicted in Figure 4.

SmoteR (**SM_TPhi**) with temporal and relevance bias. The overall results given by the $F1_\phi$ evaluation metric proposed by Ribeiro [36], obtained with Monte Carlo estimates, concerning all 24 time series data sets are presented in Figure 7.

From the obtained results, we observe that the application of resampling strategies shows great potential in terms of boosting the performance of forecasting tasks using imbalanced time series data. This is observed within each of the standard regression tools used (vertical analysis), but also regarding the data sets used (horizontal analysis), where it is clear that the approaches employing resampling strategies obtain the best results overall, according to the averaged $F1_\phi$ evaluation metric. We should note that the results obtained by the baseline **SVM** models with the optimal parameter search method employed, are very competitive and provide a better result than the resampled approaches in several occasions. We should also note that although an optimal parameter search method was employed for the baseline regression algorithms, and such parameters were used in the resampled alternatives, a similar approach was not employed concerning the optimal parameters for under and oversampling percentages. This is intended, as our objective is to assert the impact of these resampling strategies in a default setting, *i.e.* balancing the number of normal and rare cases.

5.1 Hypothesis 1

The first hypothesis brought forth in our work proposes that the use of resampling strategies significantly improves the predictive accuracy of imbalanced time series forecasting tasks in comparison to the use of standard regression tools. Although results presented in Figure 7 point to the empirical confirmation of this hypothesis, it still remains unclear the degree of statistical significance concerning the difference in evaluation between the use or non-use of resampling strategies combined with standard regression tools.

Table 3 presents the paired comparisons of the application of random undersampling (**U_B**), random oversampling (**O_B**) and SmoteR (**SM_B**), and the standard regression tools with the application of the optimal parameter search method and without any applied resampling strategy. The information in the columns represents the number of wins and losses for each approach against the baseline. In this case, the baseline represents the optimized models from the regression tools, without the application of resampling strategies.

We can observe that the use of resampling strategies adds a significant boost in terms of forecasting relevant cases in imbalanced time series data, when compared to its non-use, in all standard regression tools employed in the experiment, except for the **SVM** models. Although not in a considerable magnitude, these models collected more significant wins. Nonetheless, these experiments still provide sufficient overall empirical evidence to confirm our first hypothesis.

Given the results on $F1_\phi$ measure, a natural question arises: Are these results a reflection of a good performance in only one of the two metrics from which $F1_\phi$ depends? To assess this, we observed the results of both rec_ϕ and $prec_\phi$ on all alternative approaches tested.These figures are available at `http://tinyurl.com/z4xlup5`. Generally, the results obtained with re-

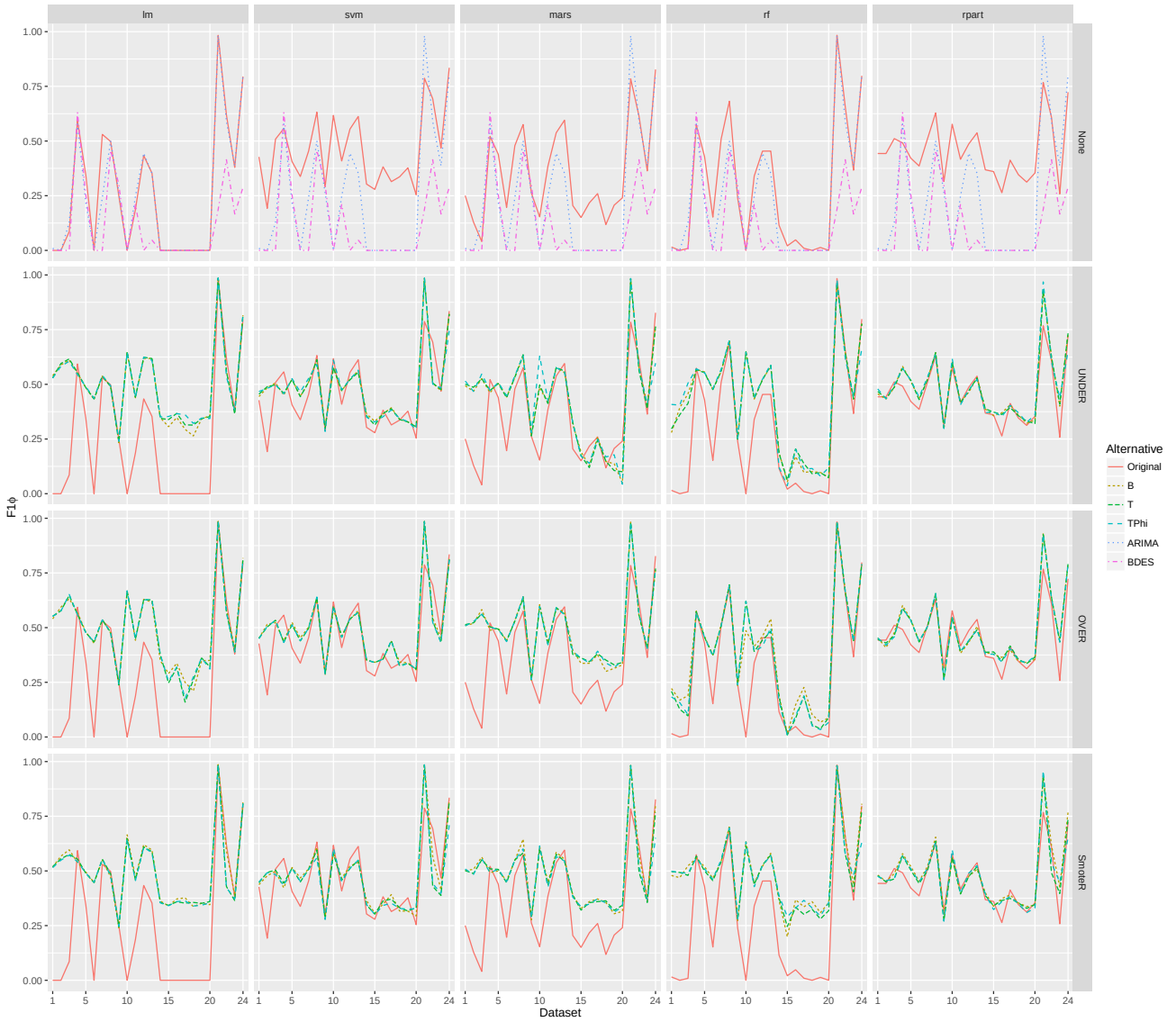

Fig. 7 Evaluation of regression algorithms and resampling strategies, with the mean utility-based regression metric $F1_\phi$.

Table 3 Paired comparisons results of each Regression Algorithm Baseline with the application of Resampling Strategies, in the format Number of Wins (Statistically Significant Wins) / Number of Losses (Statistically Significant Losses).

	LM	**SVM**	**MARS**	**RF**	**RPART**
U_B	19 (18) / 5 (4)	8 (6) / 16 (8)	15 (12) / 9 (7)	18 (17) / 6 (2)	12 (8) / 12 (3)
O_B	18 (17) / 6 (3)	7 (6) / 17 (10)	17 (17) / 7 (4)	20 (15) / 4 (1)	11 (9) / 13 (8)
SM_B	19 (18) / 5 (3)	7 (6) / 17 (10)	18 (17) / 6 (4)	20 (20) / 4 (1)	10 (10) / 14 (7)

sampling strategies for $prec_\phi$ measure present higher gains than those obtained with rec_ϕ. Still, we do not observe a performance decrease with rec_ϕ metric in the time series data used. This means that higher $F1_\phi$ results are obtained mostly due to higher $prec_\phi$ values.

5.2 Hypothesis 2

The second hypothesis states that the use of a temporal and/or relevance bias in resampling strategies significantly improves the predictive accuracy of time series forecasting tasks in comparison to the baseline versions of each respective strategy. In order to empirically prove this hypothesis, results in Table 4 present the paired comparisons of the application of the resampling strategies **U_T**, **U_TPhi**, **O_T**, **O_TPhi**, **SM_T** and **SM_TPhi**, against the respective resampling strategies **U_B**, **O_B** and **SM_B**, for each standard regression tool. For this experiment set, the baseline is defined as being the application of random undersampling, ran-

dom oversampling and SmoteR in their initial adaptation to imbalanced time series.

Results show an overall advantage of the use of temporal and/or relevance bias in the case selection process of the resampling strategies used in our experiments for random undersampling and random oversampling. In the case of SmoteR, results show that the use of temporal and/or relevance bias did not improve results, given the experimental design used. In the case of random undersampling results show that the use of temporal bias does not provide any clear advantage to the baseline version of the resampling strategy. However, when applying both temporal and relevance bias, results show significant ability for improvement. As to random oversampling, both proposals (temporal and temporal and relevance bias) show that in many cases it is possible to obtain a significant advantage result-wise, but there is no clear advantage for either one. As such, the application of temporal or temporal and relevance bias does provide empirical evidence that confirm our second hypothesis, in the case of under and oversampling.

5.3 Hypothesis 3

The third hypothesis proposed in our work is that the use of resampling strategies significantly improves the predictive accuracy of time series forecasting tasks in comparison to the use of ARIMA and BDES models. These models are approaches design specifically for time series forecasting. In this context, we want to check if our proposals based on resampling are able to significantly improve the predictive performance of these models. We remind that in this evaluation we employed a version of ARIMA models that automatically searches for the optimal number of past values to build the embed, while the standard regression tools are used with an optimal parameter setting for their baseline regression algorithm, and enhanced through the proposed resampling strategies. The results from the paired comparisons of all the approaches employing resampling strategies and the ARIMA and BDES models (considered the baseline) are presented in Table 5.

Results show that independently of the regression tool used, the application of resampling strategies provides a highly significant improvement over the results obtained by the ARIMA and BDES models. This goes to show the validity of our third and final hypothesis.

6 Discussion

The results presented in the experimental evaluation although proving to some extent the hypothesis set forth in our work, they may not provide the strongest evidence given the experimental settings. The main reason for this is related to the optimal parameter search method applied to the regression algorithms.

This method derives multiple models using diverse parameter settings in order to find the best option for each pair of regression algorithm and dataset. These optimal parameter settings are also used in the models where resampling strategies are applied. This option was intended, to ensure any observed differences are being caused only by the usage of the resampling strategies. Nonetheless, there is no underlying evidence or intuition that the best parameter settings for the baseline regression algorithms should be the best setting for the models when resampling strategies are applied.

This raises a problem as to uncovering the real potential of the application of resampling strategies when optimized by a similar optimal parameter search method, by testing additional parameters concerning such strategies (*i.e.* the percentage of cases to remove and/or add). However, this may come at a great computational cost. For example, when using the search method as described in Annex A with an additional five possible values for under sampling percentage and four values for over sampling, the amount of models produced for deciding the optimal parameter settings could amount to about 600 for a single pair of regression algorithm - data set.

Despite these issues, it is important to assert the performance of models when applying the proposed resampling strategies with optimized parameters. Therefore, we proceeded with a smaller experimental setting, where all components of each approach are optimized. This small subset includes data sets 4, 10 and 12 and the regression algorithm **SVM**. This decision is based on the analysis of previous results, where **SVM** models provided better evaluation results than the models where resampling strategies were applied, in several occasions. As such, we focus on this regression model, and on three data sets where the results of the baseline regression algorithm models provided better results than any other resampled alternative. The optimal parametrization efforts and results are described in Annex B and the results of repeating the same experimental evaluation described in the previous section considering only the **SVM** models and the three mentioned datasets are presented in Table 6.

Results show that by optimizing the parameters of both the regression algorithms and the resampling strategies, the results obtained by the latter significantly improve the results over the baseline models of the former. Additionally, it further shows the potential positive impact in terms of evaluation, when using the temporal or temporal and relevance bias.

Table 4 Paired comparisons results of each Regression algorithm with Baseline Resampling Strategies and the application of Biased Resampling Strategies, in the format Number of Wins (Statistically Significant Wins) / Number of Losses (Statistically Significant Losses).

	LM.U_B	**SVM.U_B**	**MARS.U_B**	**RF.U_B**	**RPART.U_B**
U_T	14 (2) / 10 (0)	10 (0) / 14 (0)	11 (0) / 13 (2)	12 (1) / 12 (2)	14 (4) / 10 (0)
U_TPhi	15 (10) / 9 (3)	11 (5) / 13 (4)	17 (6) / 7 (1)	16 (6) / 8 (3)	16 (7) / 8 (5)

	LM.O_B	**SVM.O_B**	**MARS.O_B**	**RF.O_B**	**RPART.O_B**
O_T	14 (8) / 10 (9)	12 (5) / 12 (6)	11 (3) / 13 (4)	8 (4) / 16 (4)	12 (3) / 12 (2)
O_TPhi	14 (9) / 10 (7)	12 (4) / 12 (7)	11 (3) / 13 (2)	8 (2) / 16 (5)	14 (3) / 10 (2)

	LM.SM_B	**SVM.SM_B**	**MARS.SM_B**	**RF.SM_B**	**RPART.SM_B**
SM_T	6 (5) / 18 (13)	10 (5) / 14 (10)	9 (6) / 15 (10)	9 (3) / 15 (10)	8 (1) / 15 (11)
SM_TPhi	6 (4) / 18 (11)	9 (6) / 15 (12)	12 (4) / 12 (6)	12 (5) / 12 (10)	6 (4) / 17 (9)

Table 5 Paired comparisons results of ARIMA and BDES models and the application of Resampling Strategies in each Regression algorithm, in the format Number of Wins (Statistically Significant Wins) / Number of Losses (Statistically Significant Losses).

Algorithm	**Strategy**	**ARIMA**	**BDES**
LM	U_B	18 (18) / 6 (3)	22 (22) / 2 (2)
	U_T	18 (18) / 6 (3)	22 (22) / 2 (2)
	U_TPhi	18 (18) / 6 (5)	22 (22) / 2 (2)
	O_B	21 (18) / 3 (2)	22 (22) / 2 (2)
	O_T	18 (18) / 6 (3)	22 (22) / 2 (2)
	O_TPhi	18 (18) / 6 (3)	22 (22) / 2 (2)
	SM_B	20 (18) / 4 (3)	22 (22) / 2 (2)
	SM_T	18 (17) / 6 (5)	22 (20) / 2 (2)
	SM_TPhi	18 (18) / 6 (5)	22 (20) / 2 (2)
SVM	U_B	21 (21) / 3 (3)	22 (22) / 2 (1)
	U_T	21 (21) / 3 (3)	22 (22) / 2 (1)
	U_TPhi	20 (20) / 4 (4)	22 (22) / 2 (2)
	O_B	21 (21) / 3 (1)	22 (22) / 2 (2)
	O_T	21 (21) / 3 (3)	22 (22) / 2 (2)
	O_TPhi	21 (21) / 3 (3)	22 (22) / 2 (2)
	SM_B	19 (19) / 5 (1)	22 (22) / 2 (2)
	SM_T	20 (20) / 4 (3)	20 (20) / 4 (2)
	SM_TPhi	19 (19) / 5 (4)	22 (20) / 2 (2)
MARS	U_B	23 (18) / 1 (1)	21 (20) / 3 (3)
	U_T	20 (18) / 4 (2)	21 (19) / 3 (2)
	U_TPhi	22 (19) / 2 (2)	21 (21) / 3 (3)
	O_B	19 (18) / 5 (1)	22 (22) / 2 (2)
	O_T	18 (18) / 6 (2)	22 (22) / 2 (2)
	O_TPhi	18 (18) / 6 (2)	22 (22) / 2 (2)
	SM_B	19 (19) / 5 (1)	22 (22) / 2 (2)
	SM_T	19 (19) / 5 (4)	22 (22) / 2 (2)
	SM_TPhi	19 (19) / 5 (4)	22 (22) / 2 (2)
RF	U_B	19 (18) / 5 (1)	19 (18) / 5 (2)
	U_T	21 (18) / 3 (2)	19 (18) / 5 (2)
	U_TPhi	21 (17) / 3 (2)	18 (18) / 6 (2)
	O_B	20 (17) / 4 (2)	18 (16) / 6 (2)
	O_T	19 (17) / 5 (2)	15 (15) / 9 (3)
	O_TPhi	19 (16) / 5 (2)	15 (15) / 9 (3)
	SM_B	22 (22) / 2 (1)	22 (22) / 2 (2)
	SM_T	20 (20) / 4 (2)	22 (22) / 2 (2)
	SM_TPhi	20 (20) / 4 (2)	22 (22) / 2 (2)
RPART	U_B	22 (20) / 2 (2)	22 (22) / 2 (1)
	U_T	22 (20) / 2 (2)	22 (22) / 2 (1)
	U_TPhi	20 (18) / 4 (1)	23 (22) / 1 (1)
	O_B	20 (20) / 4 (1)	22 (22) / 2 (2)
	O_T	20 (20) / 4 (1)	22 (22) / 2 (2)
	O_TPhi	21 (20) / 3 (1)	22 (22) / 2 (2)
	SM_B	22 (18) / 2 (1)	22 (22) / 2 (1)
	SM_T	17 (17) / 7 (4)	22 (22) / 2 (2)
	SM_TPhi	19 (18) / 5 (3)	22 (22) / 2 (2)

Table 6 Evaluation of **SVM** models and resampling strategies, with parameter optimization for three datasets, using the mean utility-based regression metric $F1_\phi$.

	DS4	**DS10**	**DS12**
svm	0.584	0.638	0.554
U_B	**0.668**	**0.652**	0.610
U_T	0.659	0.643	0.614
U_TPhi	0.651	0.647	**0.630**
O_B	**0.653**	0.651	0.611
O_T	0.650	**0.652**	**0.615**
O_TPhi	0.651	**0.652**	0.611
SM_B	**0.662**	0.675	0.609
SM_T	0.656	0.698	0.600
SM_TPhi	0.649	**0.721**	**0.620**

The relations between data characteristics and the performance of methods for addressing imbalanced domains has been explored in other studies [30]. To assess if some time series characteristics are related with our results we observed the $F1_\phi$, rec_ϕ and $prec_\phi$ metrics on the data sets sorted according to the following criteria:

- by ascending order of imbalance (i.e., increasing percentage of rare cases);
- by increasing number of total values in the data series; and
- by increasing number of rare cases, i.e., ascending total number of rare cases in the time series.

Figure 8 shows the results of $F1_\phi$ on the data sets sorted by ascending number of rare cases. The remaining results are available in `http://tinyurl.com/z4xlup5`. We observe that the characteristic that has most impact in our results is the total number of rare cases. In fact, time series with a low percentage of rare cases having a large number of values are not as problematic as time series with fewer values and a higher percentage of rare cases. This is related with the small sample problem and is in accordance with other works (e.g., [20, 21]) where it is observed that when the data set is large enough the learners can more easily detect rare cases.

Notwithstanding the predictive evaluation results presented, the impact of our proposed resampling strategies in terms of computation requirements has not been addressed so far. Considering that changing the data set

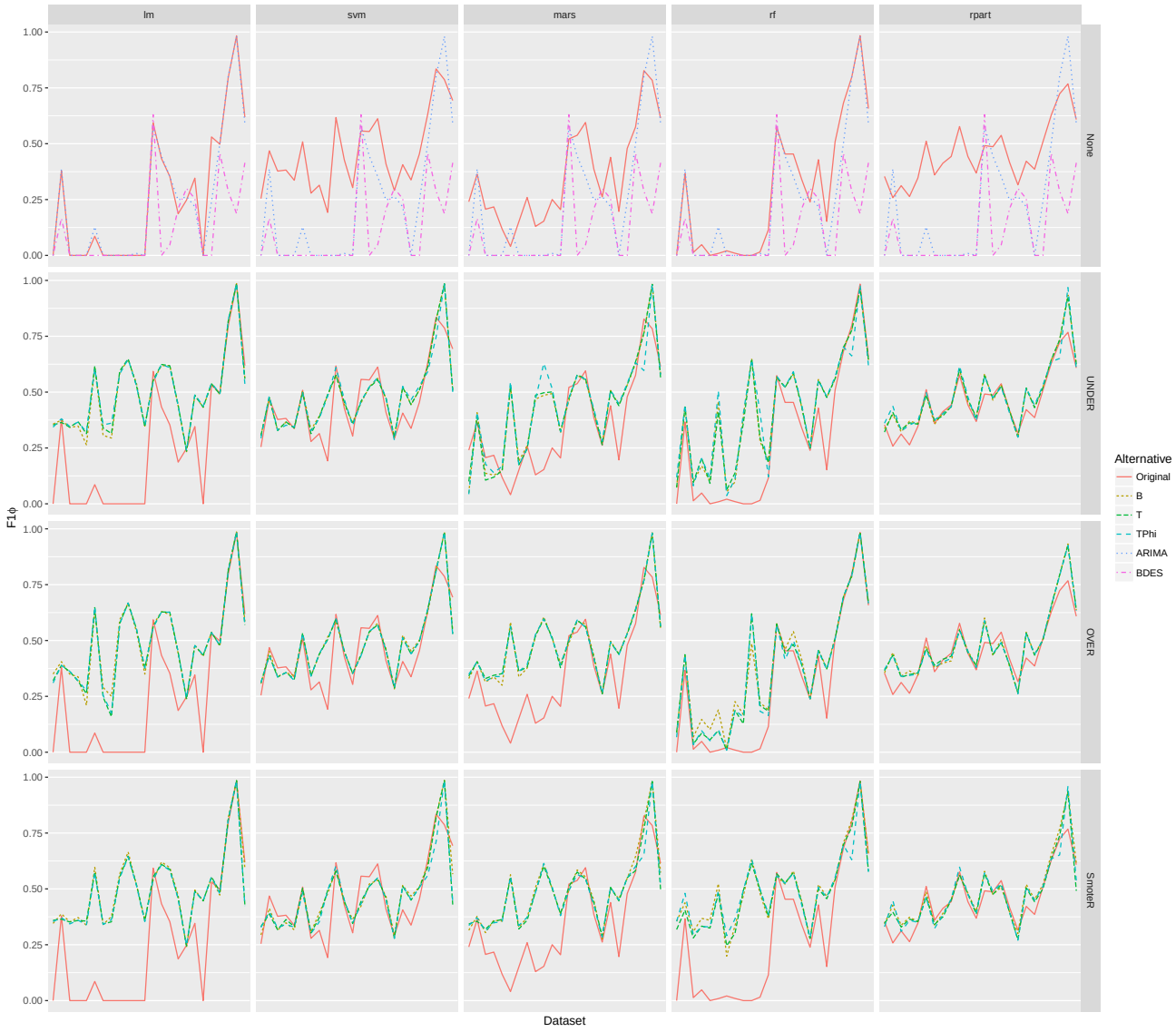

Fig. 8 Evaluation of regression algorithms and resampling strategies, with the mean utility-based regression metric $F1_\phi$ with data sets sorted by increasing number of rare cases.

may have a computational impact in building the models and forecasting future values, this issue should be studied and discussed. As such, in Figure 9 we present a comparative evaluation of the average computational time necessary to build models using each of the regression algorithms with application of resampling strategies, for all datasets, in the same experimental setting defined for the experimental evaluation described in Section 5. The results report to the proportion of computational time required to train each model using resampling strategies in comparison to the non-resampled versions (*i.e.* baseline regression algorithms). The environment for these tests was an 8-core AMD Opteron 6300 processor with 2.5 GHz and 32 GBytes of main memory, running Ubuntu 14.04 with kernel 3.16.0-30-generic.

By analysing the results shown by the computational time comparative evaluation we are able to reach strong conclusions. First, that the use of resampling strategies have a different impact concerning computational time: *i)* under sampling considerably improves the computational time required to train the models; *ii)* over sampling requires a much longer computational time to train the models; and *iii)* the SmoteR resampling strategy shows a similar computational time to train the models in comparison to the baseline regression algorithms. Results also show that these conclusions are applicable across all of the regression algorithms used in the evaluation. Secondly, results show

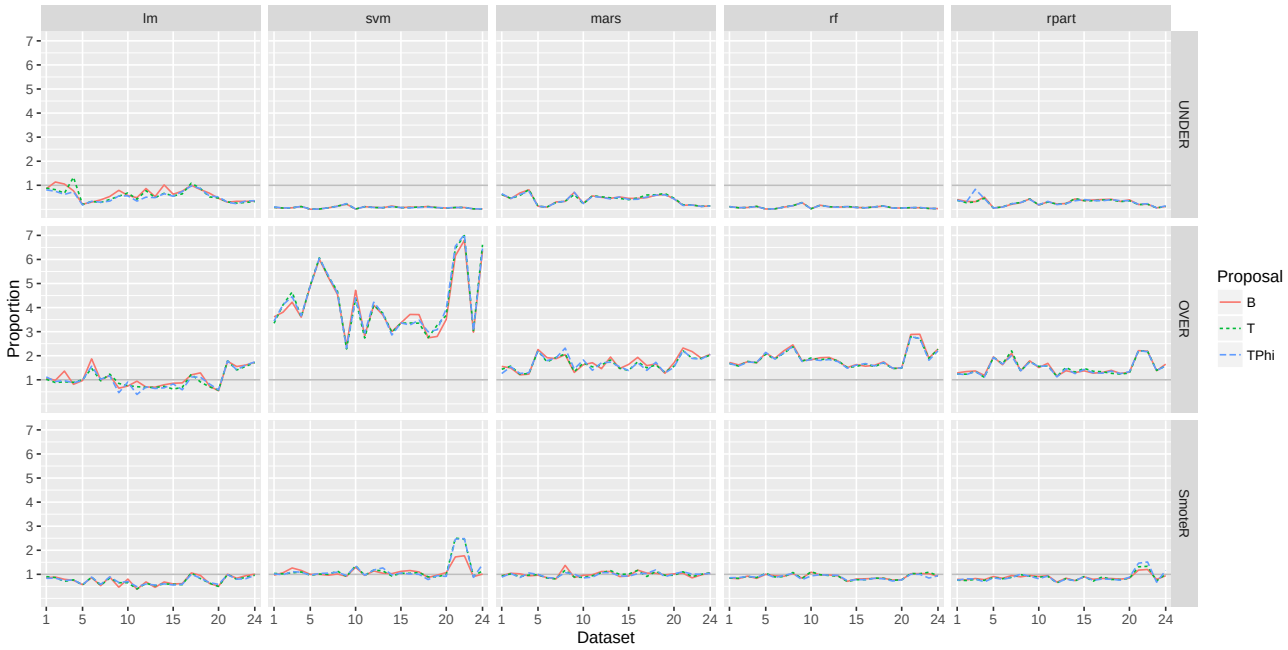

Fig. 9 Evaluation of computational time required to build models where resampling strategies are applied in comparison to the computational time of baseline regression algorithms.

that the use of temporal or temporal and relevance bias does not show a significant advantage or disadvantage in comparison to the computational time required to train the models by the baseline version of the resampling strategies.

7 Related Work

Typically the problem of imbalanced domains is tackled either by pre-processing methods, special-purpose learning methods or post-processing methods [5]. In the specific context of forecasting tasks with imbalanced time series data, we did not find any previous work that proposes the use of resampling strategies. However, we found different approaches related to the scope of our endeavour, in the problems of rare event forecasting and anomaly detection, which we describe below. Most of this work is focused in specific problems for which special-purpose learners are developed. These proposals tend to be very effective in the context for which they were developed for. However, these methods performance is severely affected when their use is extrapolated to other problems. This means that they can not be used as general methods for imbalanced time series, as opposed to resampling strategies.

A genetic-base machine learning system, *timeweaver*, was proposed by Weiss and Hirsh [50], designed to address rare event prediction problems with categorical features, by identifying predictive temporal and sequential patterns. The genetic algorithm used is responsible for updating a set of prediction patterns, where each individual should perform well at classifying a subset of the target events and which collectively should cover most of those events.

Vilalta and Ma [47] proposed an algorithm to address prediction of rare events in imbalanced time-series. The authors proposed to resolve the class-imbalance by transforming the event prediction problem into a search for all frequent event sets (patterns) preceding target events, focused solely on the minority class. These patterns are then combined into a rule-based model for prediction. Both the work of Weiss and Hirsh [50] and of Vilalta and Ma [47] assume that events are characterized by categorical features and display uneven inter-arrival times. However, this is not assumed in classical time-series analysis.

A new algorithm, ContrastMiner, is proposed by Wei Wei et. al. [49] for detection of sophisticated online banking fraud. This algorithm distinguishes between fraudulent and legitimate behaviours through contrast patterns. Then, a pattern selection and risk scoring are performed by combining different models predictions.

Temporal sequence associations are used by Chen et al. [11] for predicting rare events. The authors propose a heuristic for searching interesting patterns associated with rare events in large temporal event sequences. The authors combine association and sequential pattern discovery with a epidemiology-based measure of risk in order to assess the relevance of the discovered patterns.

Another interesting direction was pursued by Cao et. al. [7] with the development of new algorithms for discovering rare impact-targeted activities.

In anomaly detection [15] problems, applications for several domains have been proposed using diverse techniques. In the Medical and Public Health Domain, Lin et al. [28] use nearest neighbor based techniques to detect these rare cases. These same techniques are used by Basu and Mackenshimer [2] and parametric statistical modelling is used by Keogh et al. [22] in the domain of mechanical units fault detection. Finally, Scott [37] and Ihler et al. [18] propose Poisson-based analysis techniques for the respective domains of intrusion detection in telephone networks and Web Click data.

Concerning our proposal of temporal and temporal and relevance bias in imbalanced time series forecasting tasks, it is somewhat related to the seminal work of Japkowicz [19] in classification tasks. The author proposes the concept of focused resampling, for both under and oversampling. The former reduces the number of cases further away from the boundaries between the positive class (*i.e.* rare cases) and the negative class. The latter increases the number of cases closest to this boundary. Several other proposals of informed resampling have been presented since then (e.g. [25, 29]).

8 Conclusions

In this work we study the application of resampling strategies with imbalanced time series data. Our overall objective is to enhance the predictive accuracy on rare and relevant cases as this is the goal in several application domains. This fact increases the interest in finding ways to significantly improve the predictive accuracy of prediction models in these tasks.

In this context, we have proposed the extension of existing resampling methods to time series forecasting tasks. Resampling methods can be used to change the distribution of the available learning sets with the goal of biasing learning algorithms to the cases that are more relevant to the users. Our proposals build upon prior work on resampling methods for numeric prediction tasks. Besides the extension of existing resampling strategies, we propose new resampling strategies with the goal of adapting them to the specific characteristics of time series data. Specifically, we have proposed sampling strategies that introduce a temporal bias that we claim to be useful when facing non-stationary time series that are frequently subjected to concept drift. We also propose a relevance bias that makes more relevant cases have a higher preference of being selected for the final training sets.

An extensive set of experiments was carried out to ascertain the advantages of applying resampling strategies to such problems. Results from the experimental evaluation show a significant improvement in the predictive accuracy of the models, focusing on rare and relevant cases of imbalanced time series data. This is confirmed by all tested evaluation metrics. Results show that: *1)* the application of resampling strategies in combination with standard regression tools can significantly improve the ability to predict rare and relevant cases in comparison to not applying these strategies; *2)* the use of a temporal and/or relevance bias can improve the results in relation to the non-biased resampling approaches; and *3)* the combination of resampling approaches with standard regression tools provides a significant advantage in comparison to models (ARIMA and BDES) specifically developped for time series forecasting. Additionally, by studying the computational time associated to learning prediction models with and without resampling strategies, we observe that undersampling allows for a significant reduction of this required computation time, that oversampling greatly increases the required time and that SmoteR presents a similar computational time in relation to the baseline regression tools.

Concerning future work, we plan to further evaluate these proposals concerning the effect of additional parameters values such as the relevance threshold or the k number of nearest neighbours in SmoteR, and study ways of automatically adapting these parameters to the distribution. We also plan to generalize the concept of bias in resampling strategies as to study the possibility of its use not only in time series problems, but also in classification and regression tasks using various types of dependency-oriented data, such as discrete sequences, spatial and spatiotemporal data.

For the sake of reproducible science, all code and data necessary to replicate the results shown in this paper are available in the Web page `http://tinyurl.com/zr9s6tz`. All code is written in the free and open source R software environment.

Acknowledgements This work is financed by the ERDF – European Regional Development Fund through the COMPETE 2020 Programme within project POCI-01-0145-FEDER-006961, and by National Funds through the FCT – Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) as part of project UID/EEA/50014/2013. The work of N. Moniz is supported by a PhD scholarship of FCT (SFRH/BD/90180/2012). The work of P. Branco is supported by a PhD scholarship of FCT (PD/BD/105788/2014). The authors would like to thank the anonymous reviewers of the DSAA'16 conference and the anonymous reviewers of this extended version for their remarks. The authors would also like to thank Rita Ribeiro for her comments and inputs.

Conflict of Interest

On behalf of all authors, the corresponding author states that there is no conflict of interest.

Appendices

A Annex 1

The following list describes the parameters tested in each of the regression algorithms used in the experimental evaluation (Section 5).

- **svm**: $cost \in \{10, 150, 300\}$, $gamma \in \{0.01, 0.001\}$;
- **mars**: $nk \in \{10, 17\}$, $degree \in \{1, 2\}$, $thresh \in \{0.01, 0.001\}$;
- **rf**: $mtry \in \{5, 7\}$, $ntree \in \{500, 750, 1500\}$;
- **rpart**: $minsplit \in \{10, 20, 30\}$, $cp \in \{0.1, 0.01, 0.001\}$

To find the optimal combination of parameters for each of the standard regression algorithms an experimental evaluation was carried out. We applied the same experimental methodology as described in Section 5. The combination of parameters for each regression algorithm, in each data set used, is detailed in Table 7.

Table 7 Optimal parametrization for each standard regression algorithm in each data set used for the experimental evaluation.

	SVM		MARS			RF		RPART	
	cost	gamma	nk	degree	thr	mtry	ntree	minsplit	cp
DS1	300	0.01	17	1	0.001	5	1500	10	0.01
DS2	300	0.01	17	2	0.001	7	750	10	0.001
DS3	300	0.01	17	1	0.001	7	500	10	0.001
DS4	150	0.01	10	1	0.001	7	750	10	0.1
DS5	300	0.001	10	2	0.001	7	750	20	0.001
DS6	300	0.01	17	2	0.001	5	500	10	0.001
DS7	300	0.01	10	1	0.001	7	750	30	0.001
DS8	300	0.01	17	2	0.001	7	750	30	0.001
DS9	10	0.01	10	2	0.001	5	750	30	0.001
DS10	300	0.01	17	2	0.001	7	500	10	0.001
DS11	10	0.01	17	1	0.001	7	500	20	0.001
DS12	300	0.01	17	1	0.001	7	750	10	0.001
DS13	150	0.01	17	2	0.001	7	750	10	0.001
DS14	150	0.01	17	2	0.001	7	1500	10	0.001
DS15	300	0.01	17	2	0.001	5	1500	10	0.001
DS16	300	0.01	17	2	0.001	7	750	10	0.001
DS17	300	0.01	17	2	0.001	7	500	10	0.001
DS18	300	0.01	17	2	0.001	5	500	10	0.001
DS19	150	0.01	17	1	0.01	5	500	10	0.001
DS20	300	0.01	17	2	0.001	7	500	10	0.001
DS21	150	0.001	17	2	0.001	7	500	10	0.001
DS22	150	0.001	10	2	0.001	7	500	10	0.001
DS23	10	0.001	10	1	0.001	5	500	10	0.001
DS24	150	0.01	17	1	0.001	7	750	10	0.001

B Annex 2

To optimize **SVM** models and the resampling strategies applied the following parameters were tested: $cost \in \{10, 150, 300\}$, $gamma \in \{0.01, 0.001\}$, $over \in \{2, 3, 5, 10\}$, $under \in \{0.1, 0.2, 0.4, 0.6, 0.8\}$.

To find the optimal combination of parameters for each of the standard regression algorithms an experimental evaluation was carried out. We applied the same experimental methodology as described in Section 5, using 10 repetitions of the Monte Carlo simulations. The combination of parameters in the three data sets used (4, 10 and 12), is detailed in Table 8.

Table 8 Optimal parametrization for **SVM** regression algorithm with the application of resampling strategies in three data sets. Parameters optimized include cost (c), gamma (g), percentage of undersampling (u) and oversampling (o).

		UNDER			OVER			SmoteR			
		c	g	u	c	g	o	c	g	u	o
DS4	**B**	10	0.01	0.4	10	0.001	5	150	0.001	0.8	2
	T	10	0.01	0.4	150	0.001	2	150	0.001	0.6	2
	TPhi	10	0.01	0.8	150	0.01	2	10	0.001	0.8	2
DS10	**B**	10	0.001	0.1	10	0.001	2	10	0.001	0.8	10
	T	150	0.001	0.1	150	0.001	2	10	0.001	0.6	5
	TPhi	300	0.001	0.1	150	0.001	2	300	0.001	0.6	3
DS12	**B**	150	0.001	0.2	10	0.001	10	10	0.001	0.2	3
	T	300	0.001	0.2	150	0.001	3	10	0.001	0.8	5
	TPhi	150	0.001	0.2	150	0.001	3	150	0.001	0.4	2

References

1. Oguz Akbilgic, Hamparsum Bozdogan, and M. Erdal Balaban. A novel hybrid RBF neural networks model as a forecaster. *Statistics and Computing*, 24(3):365–375, 2014.
2. Sabyasachi Basu and Martin Meckesheimer. Automatic outlier detection for time series: An application to sensor data. *Knowl. Inf. Syst.*, 11(2): 137–154, February 2007. ISSN 0219-1377.
3. Paula Branco. Re-sampling approaches for regression tasks under imbalanced domains. Master's thesis, Universidade do Porto, 2014.
4. Paula Branco, Rita P. Ribeiro, and Luis Torgo. UBL: an R package for utility-based learning. *CoRR*, abs/1604.08079, 2016.
5. Paula Branco, Luís Torgo, and Rita P. Ribeiro. A survey of predictive modeling on imbalanced domains. *ACM Comput. Surv.*, 49(2):31:1–31:50, 2016.
6. Chumphol Bunkhumpornpat, Krung Sinapiromsaran, and Chidchanok Lursinsap. Dbsmote: Density-based synthetic minority over-sampling technique. *Applied Intelligence*, 36(3):664–684, 2012.
7. Longbing Cao, Yanchang Zhao, and Chengqi Zhang. Mining impact-targeted activity patterns in imbalanced data. *IEEE Transactions on knowledge and data engineering*, 20(8):1053–1066, 2008.
8. Chris Chatfield. *The analysis of time series: an introduction*. CRC Press, 6th edition, 2004.

9. N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. Smote: Synthetic minority oversampling technique. *JAIR*, 16:321–357, 2002.
10. Nitesh V. Chawla, Nathalie Japkowicz, and Aleksander Kotcz. Editorial: Special issue on learning from imbalanced data sets. *SIGKDD Explor. Newsl.*, 6(1):1–6, June 2004. ISSN 1931-0145.
11. Jie Chen, Hongxing He, Graham J. Williams, and Huidong Jin. Temporal sequence associations for rare events. In *Proc. of the 8th PAKDD*, pages 235–239. Springer, 2004.
12. Randall L. Dougherty, Alan Edelman, and James M. Hyman. Nonnegativity-, monotonicity-, or convexity-preserving cubic and quintic Hermite interpolation. *Mathematics of Computation*, 52(186):471–494, 1989. ISSN 00255718. doi: 10.2307/2008477.
13. Charles Elkan. The foundations of cost-sensitive learning. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'01, pages 973–978, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1-55860-812-5, 978-1-558-60812-2.
14. Hadi Fanaee-T and João Gama. Event labeling combining ensemble detectors and background knowledge. *Prog. in Art. Int.*, pages 1–15, 2013. ISSN 2192-6352.
15. Tom Fawcett and Foster Provost. Activity monitoring: Noticing interesting changes in behavior. In *Proc. of the 5th ACM SIGKDD*, pages 53–62, 1999.
16. T. Ryan Hoens, Qi Qian, Nitesh V. Chawla, and Zhi-Hua Zhou. Building decision trees for the multi-class imbalance problem. In *Proc. of the 16th PAKDD*, pages 122–134. Springer Berlin Heidelberg, 2012.
17. Rob Hyndman and Yeasmin Khandakar. Automatic time series forecasting: The forecast package for r. *Jour. of Stat. Soft.*, 27(1):1–22, 2008. ISSN 1548-7660.
18. Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proc. of the 12th ACM SIGKDD*, pages 207–216, New York, NY, USA, 2006.
19. Nathalie Japkowicz. The class imbalance problem: Significance and strategies. In *Proc. of the 2000 International Conference on Artificial Intelligence (ICAI)*, pages 111–117, 2000.
20. Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.
21. Taeho Jo and Nathalie Japkowicz. Class imbalances versus small disjuncts. *ACM Sigkdd Explorations Newsletter*, 6(1):40–49, 2004.
22. Eamonn Keogh, Stefano Lonardi, and Bill 'Yuan-chi' Chiu. Finding surprising patterns in a time series database in linear time and space. In *Proc. of the 8th ACM SIGKDD*, pages 550–556, New York, NY, USA, 2002.
23. I. Koprinska, M. Rana, and V.G. Agelidis. Yearly and seasonal models for electricity load forecasting. In *Proc. of 2011 IJCNN*, pages 1474–1481, July 2011.
24. M. Kubat and S. Matwin. Addressing the curse of imbalanced training sets: One-sided selection. In *Proc. of the 14th ICML*, pages 179–186, Nashville, TN, USA, 1997. Morgan Kaufmann.
25. Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Conference on Artificial Intelligence in Medicine in Europe*, pages 63–66. Springer, 2001.
26. Kewen Li, Wenrong Zhang, Qinghua Lu, and Xianghua Fang. An improved smote imbalanced data classification method based on support degree. In *Proc. of 2014 International Conference IIKI*, pages 34–38. IEEE, 2014.
27. A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.
28. Jessica Lin, Eamonn J. Keogh, Ada Wai-Chee Fu, and Helga Van Herle. Approximations to magic: Finding unusual medical time series. In *CBMS*, pages 329–334. IEEE Computer Society, 2005. ISBN 0-7695-2355-2.
29. Xu-Ying Liu, Jianxin Wu, and Zhi-Hua Zhou. Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(2):539–550, 2009.
30. Victoria López, Alberto Fernández, Salvador García, Vasile Palade, and Francisco Herrera. An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. *Information Sciences*, 250: 113–141, 2013.
31. D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel, and F. Leisch. *e1071: Misc Functions of the Department of Statistics (e1071), TU Wien*, 2012. R package version 1.6-1.
32. S. Milborrow. *earth: Multivariate Adaptive Regression Spline Models*, 2013.
33. Nuno Moniz, Paula Branco, and Luís Torgo. Resampling strategies for imbalanced time series. In *Proc. 3rd IEEE Int. Conf. on Data Science and Advanced Analytics (DSAA)*, Montreal, Canada, 2016.
34. Mariana Oliveira and Luis Torgo. Ensembles for time series forecasting. In *Proc. of the 6th Asian*

Conference on Machine Learning (ACML), Nha Trang City, Vietnam, 2014.
35. R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2014.
36. R. Ribeiro. *Utility-based Regression*. PhD thesis, Dep. Computer Science, Faculty of Sciences - University of Porto, 2011.
37. Steven L. Scott. Detecting network intrusion using a markov modulated nonhomogeneous poisson process. *Subm. to Jour. ASA*, 2000.
38. Chris Seiffert, Taghi M Khoshgoftaar, Jason Van Hulse, and Amri Napolitano. Rusboost: A hybrid approach to alleviating class imbalance. *IEEE Trans. SMC*, 40(1):185–197, 2010.
39. Floris Takens. *Detecting strange attractors in turbulence*, pages 366–381. Springer Berlin Heidelberg, Berlin, Heidelberg, 1981. ISBN 978-3-540-38945-3. doi: 10.1007/BFb0091924. URL `http://dx.doi.org/10.1007/BFb0091924`.
40. Terry Therneau, Beth Atkinson, and Brian Ripley. *rpart: Recursive Partitioning and Regression Trees*, 2015. URL `https://CRAN.R-project.org/package=rpart`. R package version 4.1-10.
41. H. Tong, B. Thanoon, and G. Gudmundsson. Threshold time series modeling of two icelandic riverflow systems1. *JAWRA*, 21(4):651–662, 1985. ISSN 1752-1688.
42. L. Torgo. *Data Mining with R, learning with case studies*. Chapman and Hall/CRC, 2010.
43. L. Torgo. An infra-structure for performance estimation and experimental comparison of predictive models in R. *CoRR*, abs/1412.0436, 2014.
44. L. Torgo and R. Ribeiro. Utility-based regression. In Springer, editor, *Proc. of 11th PKDD*, pages 597–604, 2007.
45. L. Torgo, P. Branco, R. P. Ribeiro, and B. Pfahringer. Resampling strategies for regression. *Exp. Sys.*, 32(3):465–476, 2015.
46. Luís Torgo, Rita P Ribeiro, Bernhard Pfahringer, and Paula Branco. Smote for regression. In *Prog. in Art. Int.*, pages 378–389. Springer, 2013.
47. R. Vilalta and Sheng Ma. Predicting rare events in temporal domains. In *Proc. of the 2002 IEEE ICDM*, pages 474–481, 2002.
48. Byron C Wallace, Kevin Small, Carla E Brodley, and Thomas A Trikalinos. Class imbalance, redux. In *Proc. of 11th ICDM*, pages 754–763. IEEE, 2011.
49. Wei Wei, Jinjiu Li, Longbing Cao, Yuming Ou, and Jiahang Chen. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web*, 16(4):449–475, 2013.
50. Gary M. Weiss and Haym Hirsh. Learning to predict rare events in event sequences. In *Proc. of the 4th KDD*, pages 359–363. AAAI Press, 1998.
51. Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Mach. Lear.*, 23(1):69–101, 1996. ISSN 0885-6125.
52. Qiang Yang and Xindong Wu. 10 challenging problems in data mining research. *Int. Jour. of Inf. Tech. & Dec. Mak.*, 5(4):597–604, 2006.