

# Instance Selection and Outlier Generation to Improve the Cascade Classifier Precision

Judith Neugebauer<sup>(✉)</sup>, Oliver Kramer, and Michael Sonnenschein

Department of Computing Science, Carl von Ossietzky University Oldenburg,  
Oldenburg, Germany

{judith.neugebauer,oliver.kramer,michael.sonnenschein}@uni-oldenburg.de

**Abstract.** Classification of high-dimensional time series with imbalanced classes is a challenging task. For such classification tasks, the cascade classifier has been proposed. The cascade classifier tackles high-dimensionality and imbalance by splitting the classification task into several low-dimensional classification tasks and aggregating the intermediate results. Therefore the high-dimensional data set is projected onto low-dimensional subsets. But these subsets can employ unfavorable and not representative data distributions, that hamper classification again. Data preprocessing can overcome these problems. Small improvements in the low-dimensional data subsets of the cascade classifier lead to an improvement of the aggregated overall results. We present two data preprocessing methods, instance selection and outlier generation. Both methods are based on point distances in low-dimensional space. The instance selection method selects representative feasible examples and the outlier generation method generates artificial infeasible examples near the class boundary. In an experimental study, we analyse the precision improvement of the cascade classifier due to the presented data preprocessing methods for power production time series of a micro Combined Heat and Power plant and an artificial and complex data set. The precision increase is due to an increased selectivity of the learned decision boundaries. This paper is an extended version of [19], where we have proposed the two data preprocessing methods. In this paper we extend the analysis of both algorithms by a parameter sensitivity analysis of the distance parameters from the preprocessing methods. Both distance parameters depend on each other and have to be chosen carefully. We study the influence of these distance parameters on the classification precision of the cascade model and derive parameter fitting rules for the  $\mu$ CHP data set. The experiments yield a region of optimal parameter value combinations leading to a high classification precision.

**Keywords:** Time series classification · High-dimensional classification · Imbalanced learning · Data preprocessing

## 1 Introduction

Classification of high-dimensional data sets with imbalanced or even severely imbalanced classes is influenced by the curse of dimensionality. This is also true

for time series classification tasks, where the ordering of the features (time steps) is important, [1]. Such tasks can be e.g., energy time series, where neighboring time steps are correlated. For these high dimensional time series classification tasks with imbalanced classes we have proposed the cascade classification model [18]. This model employs a cascade of classifiers based on features of overlapping time series steps. Therefore the high-dimensional feasible time series are projected on all neighboring pairs of time steps. In the low-dimensional space of the data subsets, the curse of dimensionality is no longer a problem.

Classification precision depends strongly on the distribution of the underlying data set, [16]. Therefore, an improvement of the data distribution could improve classification precision. Time series classification tasks with a cascade classifier have mainly two reason for unfavorable data distributions. Beside the original often not homogeneous distribution of the time series in feature space, the projection of feasible time series leads to an inhomogeneous distribution in low-dimensional space. A selection of more homogeneously distributed feasible examples (instances) would lead to an improvement in classification precision for a constant number of training examples or decrease the number of training examples, that are necessary to achieve a certain classification precision. In [19] we have proposed a resampling algorithm for feasible low-dimensional examples. The algorithm is based on distances between nearest neighbors. If the distance is greater than a certain threshold, the respective example is part of the new more homogeneous data set.

Additionally, infeasible examples can further improve the classification precision by increasing the selectivity of the decision boundaries, [27]. If there are enough infeasible examples, binary classification can be applied and yield better results than one-class classification, see [3]. But even if there are infeasible examples available in high-dimensional space, they can not be used for training of the low-dimensional classifiers. Energy time series e.g., are only feasible, if all time steps are feasible. Due to this property infeasible power production time series projected to low-dimensional space can be located in the region of feasible ones. Since projection of high-dimensional infeasible examples does not work, we have proposed a sampling procedure for artificial infeasible examples for the low-dimensional data subsets in [19]. Sampling of artificial infeasible examples is based on minimal distances to the nearest feasible neighbor. The infeasible examples are generated near the class boundary to improve the selectivity of the classifiers.

This paper is an extended version of [19]. The experiments in the original paper revealed that both distance parameters in the preprocessing methods have to be chosen carefully. Therefore we analyze additionally the combined effect of the distance parameters on the cascade classifier precision in this paper. We conduct the sensitivity analysis exemplarily for the combined heat and power plant power output data set and derive parameter fitting rules for the preprocessing methods.

This paper is structured as follows. In Sect. 2, we provide an overview on related work, instance selection, generation of artificial infeasible examples

(outliers) and sensitivity analysis. In Sect. 3 we describe the cascade classification approach and in Sect. 4 we introduce our data preprocessing methods to improve the cascade classifier. In Sect. 5, we compare the classification precision of the cascade approach with and without data preprocessing in an experimental study. This study is conducted on simulated micro combined heat and power plant ( $\mu$ CHP) data and an artificial complex data set. A sensitivity analysis of the distance parameters from the data preprocessing methods is presented in Sect. 6. In Sect. 7, we summarize and draw conclusions.

## 2 Related Work

In classification tasks, a lot of problems can arise due to not optimally distributed data, like not representative data samples or inhomogeneously distributed samples.

For the cascade classifier, [18], the projection of the feasible examples from high to low-dimensional space leads to additional inhomogeneity in the distribution of feasible examples. Unfavorable data distributions hamper classification, [16]. But data preprocessing methods that select representative examples from the data set and maintain the integrity of the original data set while reducing the data set can help to overcome the classification problems. Depending on the data distribution and the application several instance selection (also called record reduction/numerosity reduction/prototype selection) approaches have been developed. Beside data compression and classification precision improvement instance selection also works as noise filter and prototype selector, [4, 24, 25]. In the last years, several instance selection approaches have been proposed and an overview can be found e.g., in [9, 13, 17]. Based on these algorithms advanced instance selection algorithms e.g. based on ensembles, [4], genetic algorithms, [24] or instance selection for time series classification with hubs, [23] were developed. But all these instance selection approaches have more or less high computational complexity, because they are developed for  $d$ -dimensional data sets, while the cascade classifier has several similar structured data subsets in low-dimensional space. Therefore, we propose a simple and fast instance selection method for low-dimensional space.

As far as infeasible examples (outliers, counter examples) can improve (one-class) classification, [27], algorithms to sample infeasible examples have been proposed. One such algorithm generates counter examples around the feasible class based on points near the class boundary, [2]. Another algorithm presented in [22] can sample outliers from a hyperbox or a hypersphere, that cover the target object (feasible class). The artificial infeasible examples of these algorithms comprise either high computational complexity or contain some feasible examples. But the cascade classifier requires a fast and simple sampling approach for all low-dimensional data subsets, where the generated infeasible examples are located in the region of the infeasible class. Thus we propose an artificial outlier generation method for the data subsets of the cascade classifier.

Instance selection and outlier generation are applied to increase classification precision of the cascade classification model. The magnitude of precision increase

depends on the one hand on the application and on the other hand on the parametrization of the data preprocessing methods.

This influence of the preprocessing method parametrization on the classification precision can be analyzed with a sensitivity analysis. Sensitivity analysis (SA), also known as elastic theory, response surface methodology or design of experiment, examines the response of model output parameters to input parameter variations. For the sensitivity analysis of mathematical and statistical models several methods have been proposed see e.g., [5, 8, 10, 15, 26]. Sensitivity analysis is e.g., applied to data mining models in [7, 8] to analyze the black box behaviour of data mining models and to increase their interpretability. Several sensitivity analysis methods have been proposed in literature for local, global and screen methods, see [12]. Local methods are used to study the influence of one parameter on the output, while all other parameters are kept constant. Global methods are used to evaluate the influence of one parameter by varying all other parameters as well. Screen methods are used for complex tasks, where global methods are computationally too expensive.

Concerning the influence of data preprocessing parameters on the cascade model precision, only some parameters are of interest and therefore local methods are appropriate. The simplest approach is the one-at-a-time method (OAT), [10] where one parameter is varied within a given parameter range, while all other parameters are kept constant. The influence of the input parameters on the model output can be determined qualitatively e.g., with scatter plots or quantitatively e.g., with correlation coefficients or regression analysis, see [8, 10].

### 3 Cascade of Overlapping Feature Classifiers

In this section, we introduce the cascade approach for time series classification [18]. As the classification of the high-dimensional time series is difficult, a step-wise classifier has been proposed. The cascade classification model is developed for high-dimensional binary time series classification tasks with (severely) imbalanced classes. The small interesting class is surrounded by the other class. Both classes fill together a hypervolume, e.g. a hypercube. Furthermore the cascade classifier requires data sets with clearly separable classes, where the small interesting class has a strong correlation between neighboring features (time steps). The low-dimensional data subsets of the small class should preferably employ only one concept (cluster) and a shape, that can be easily learned.

The model consists of a cascade of classifiers, each based on two neighboring time series steps (features) with a feature overlap between the classifiers. The cascade approach works as follows. Let  $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$  be a training set of  $N$  time series  $\mathbf{x}_i = (x_i^1, x_i^2, \dots, x_i^d)^T \in \mathbb{R}^d$  of  $d$  time steps and  $y_i \in \{+1, -1\}$  the information about their feasibility. For each 2-dimensional training set

$$((x_1^j, x_1^{j+1}), y_1), \dots, ((x_N^j, x_N^{j+1}), y_N) \quad (1)$$

a classifier is trained. All  $d - 1$  classification tasks can be solved with arbitrary baseline classifiers, depending on the given data. Single classifiers employ similarly structured data spaces and thus less effort is needed for parameter tuning. Most of the times only feasible low-dimensional examples are available and in this

case baseline classifiers from one-class classification are suitable. The predictions  $f_1, \dots, f_{d-1}$  of all  $d - 1$  classifiers are aggregated to a final result

$$F(\mathbf{x}) = \begin{cases} +1 & \text{if } f_i \neq -1 \ \forall i = 1, \dots, d - 1 \\ -1 & \text{else} \end{cases} \quad (2)$$

for a time series  $\mathbf{x}$ . A new time series  $\mathbf{x}$  is feasible, only if all classifiers in the cascade predict each time step as feasible. The cascade classification approach can be modified and extended, e.g., concerning the length of the time series intervals, respectively the dimensionality of the low-dimensional data subsets.

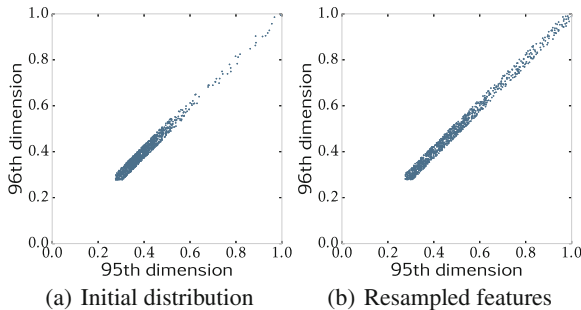
## 4 Data Preprocessing Methods

In this section the two data preprocessing methods for the cascade classification model are presented. Both methods operate on the low dimensional training subsets. The low-dimensional subsets fulfill the cascade model requirements. Both classes are clearly separable. The low-dimensional subsets incorporate similar structures in feature space and employ values in the same ranges for all time steps (features). For convenience all features are scaled, preferably to values between 0 and 1. Scaling of the features allows the use of the same parametrization for the data preprocessing methods for all low-dimensional subsets of the cascade classifier.

In the following we present two data preprocessing methods, an instance selection algorithm and an outlier generation algorithm for 2-dimensional training subsets. But just like the dimensionality of the low-dimensional subsets of the cascade approach could be changed, the proposed data preprocessing methods could be also applied to data subsets of other dimensionality.

### 4.1 Selection of Feasible Examples

Selection of feasible examples is an instance selection method for the low-dimensional feasible training subsets of the cascade classifier. The goal is to achieve more representative training examples by homogenizing the point density of the training subsets, see Fig. 1.



**Fig. 1.** 1000 examples of the 95th and 96th dimensions of the feasible class of the  $\mu$ CHP data set (initial and resampled).

The example figures for selection of feasible examples show an increase in the point density in the upper right corner and a decrease in the point density in the lower left corner, see Fig. 1(b) in comparison to the original distribution shown in Fig. 1(a). Homogenization is achieved by selecting feasible examples for the training subsets based on the distance to the nearest feasible neighbors. Therefore a large set of feasible examples is needed, from which representative examples can be chosen. We assume that the inhomogeneous distribution of the training examples and their rarity in some regions is due to relative rarity. Relative rarity means examples are observed (sampled) less frequently than others, see e.g., [11]. But the rare examples constitute a certain percentage of a data set and an increase of the number of examples in the data set increases the absolute number of rare examples. If the rarity would be an absolute rarity, the absolute number of rare examples could not be increased with an increase of examples in the data set, see e.g., [11]. For this reason selection of feasible examples can only increase homogeneity of training examples, if rarity is relative. Based on the data properties resulting from the cascade classifier and the above described requirements, selection of  $sn$  feasible examples works as follows for each low-dimensional training subset, see Algorithm 1.

---

**Algorithm 1.** Selection of feasible examples.

---

**Require:** 2-dimensional data set  $\mathbf{X}$  with  $n$  feasible examples

- 1: choose  $t$  start examples  $S$  from  $\mathbf{X}$
- 2: **repeat**
- 3:   choose  $t$  new examples  $E$  from  $\mathbf{X}$
- 4:   calculate euclidean distance  $\delta$  of the examples in  $E$  to their nearest neighbors in  $S$
- 5:   **if**  $\delta \geq \epsilon$  **then**
- 6:     append respective examples to  $S$
- 7:   **end if**
- 8: **until** all  $n$  examples are processed
- 9: shuffle  $S$

---

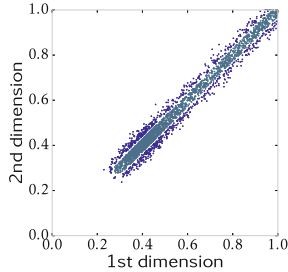
$t$  feasible examples are chosen from the given data set  $X$ . These  $t$  examples are the first examples of the homogenized training set  $S$ . Then  $t$  new examples (set  $E$ ) are taken and the distance between them and their nearest neighbors in  $S$  is computed. An example from  $E$  is added to the set  $S$  if the distance  $\delta$  to its nearest neighbor in  $S$  is larger than or equals a certain distance  $\epsilon$ . This procedure is repeated until all examples in  $X$  are processed. The algorithm parameters  $t$  and  $\epsilon$  depend on each other and the data set. The number of examples used for each comparison iteration is  $t \geq 1$ . The upper bound for the value of  $t$  depends on the number of selected feasible examples  $sn$  in  $S$  and should be about ( $t < sn/3$ ). The smaller  $\epsilon$  the larger can be  $t$ . An appropriate value for  $\epsilon$  has to be chosen in pre-tests in such a way, that the examples in set  $S$  are more or less homogeneously distributed for all low-dimensional training sets of the cascade classifier. Furthermore the number of selected feasible examples  $sn$  in  $S$  should be not much larger than the desired number of training and probably also

validation examples. These conditions guarantee a good data representation of the feasible class, because nearly all  $sn$  examples are used to train (and validate) the classifier. If the data sets are scaled to values between 0 and 1,  $\epsilon$  values from the interval  $[0.0005, 0.005]$  can be tried as initial values.

Training examples for the cascade classifier are taken from the respective set  $S$  for each training subset and validation examples can be also taken from the homogenized set  $S$  or from the remaining feasible examples, that do not belong to  $S$ .

## 4.2 Sampling of Infeasible Examples Near the Class Boundaries

Sampling of infeasible examples near the class boundary is an outlier generation algorithm for low-dimensional space. The aim of this algorithm is the generation of low-dimensional infeasible examples near the true class boundary as additional training and or validation examples, see Fig. 2.



**Fig. 2.** Resampled examples of the 1st and 2nd dimension of the feasible class of the  $\mu$ CHP data set with artificial infeasible examples. The feasible class shown as gray points is surrounded by artificial infeasible examples (blue points). (Color figure online)

Low-dimensional infeasible examples are generated at a certain distance to their nearest feasible neighbor. Due to this distance dependence to the feasible class, examples of the feasible class are required. These examples have to represent the feasible class as good as possible and furthermore they have to be distributed more or less homogeneously. Sampling of infeasible examples strongly relies on the cascade classification model requirement of clearly separable classes. Additionally the class boundaries should be clear lines in low-dimensional space. With consideration of these requirements, generation of artificial infeasible examples near the class boundaries can be applied as a second data preprocessing method after selection of feasible examples. The algorithm works as described in Algorithm 2 for all low-dimensional training subsets of the cascade classifier.

---

**Algorithm 2.** Sampling of infeasible examples.

---

**Require:** 2-dimensional data set  $\mathbf{X}$  with  $n$  feasible examples, where the distance between infeasibles and their feasible nearest neighbors  $\delta_b \gg \epsilon_b$  in about 95% of all cases

- 1:  $Y = \mathbf{X} + \mathcal{N}(\mu, \sigma) \cdot \alpha$
- 2: calculate euclidean distance  $\delta_b$  of all examples in  $Y$  to their nearest feasible neighbors in  $X$
- 3: **if**  $\delta_b \geq \epsilon_b$  **then**
- 4:   examples are infeasible examples ( $\Gamma$ )
- 5: **end if**
- 6: **repeat**
- 7:    $Y = \Gamma + \mathcal{N}(\mu, \sigma) \cdot \alpha$
- 8:   calculate euclidean distance  $\delta_b$  of all examples in  $Y$  to their nearest neighbors in  $X$
- 9:   **if**  $\delta_b \geq \epsilon_b$  **then**
- 10:     append example to  $\Gamma$
- 11:   **end if**
- 12: **until** number of examples in  $\Gamma$  is sufficient
- 13: shuffle  $\Gamma$

---

The low-dimensional feasible examples  $X$  are perturbed with gaussian noise  $\mathcal{N}(\mu, \sigma) \cdot \alpha$  and yield a new data set  $Y$ . Then the distance between the examples in  $Y$  and their nearest feasible neighbors in  $X$  is computed. A value in  $Y$  belongs to the set of artificial infeasible examples  $\Gamma$  if the distance to the nearest feasible neighbor  $\delta_b$  is larger than or equals a certain value  $\epsilon_b$ . To receive enough infeasible examples around the feasible class, the above described procedure is repeated with a perturbation of all examples in  $\Gamma$  instead of the examples in  $X$  until the set  $\Gamma$  contains a sufficient number of examples. The algorithm employs the parameter for the minimal distance between infeasible examples and their nearest feasible neighbors  $\epsilon_b$  and the gaussian distribution  $\mathcal{N}(\mu, \sigma) \cdot \alpha$ . The parameters depend on the distribution of feasible examples, mainly the distance between feasible nearest neighbors  $\epsilon$ . Therefore  $\epsilon_b$  has to be chosen in such a way, that  $\epsilon_b \gg \epsilon$  form the instance selection algorithm. The parameter  $\epsilon_b$  has to be chosen carefully, see Sect. 6.2. The  $\epsilon_b$  value should be at least so high, that at least 95% off all generated artificial infeasible examples lie outside the region of the feasible class. As far as the true class boundary is not known, the percentage of real feasible examples among the artificial infeasible ones has to be approximated. If the distance  $\delta_b$  between generated infeasible examples and their nearest feasible neighbors is  $\delta_b \gg \epsilon_b$  for at least 95% of all generated infeasible examples, then most of the generated infeasible examples are actually infeasible ones. The approximation relies on the requirement, that the examples of the feasible class are representatively and homogeneously distributed.

The closer the infeasible examples are located to the class boundary, the greater is the improvement of classification specificity. But the closer the infeasible examples are located to the class boundary, the higher is the probability, that these artificial infeasible examples could be located in the region of



the feasible class. False artificial infeasible examples can hamper classification improvement. Therefore a careful parametrization of the algorithm is necessary. All in all the minimum distance  $\epsilon_b$  between infeasible examples and their nearest feasible neighbors should be as small as possible and as large as necessary.

Noise for the generation of potentially infeasible examples should scatter in all directions without a drift. Therefore the gaussian distribution is chosen with a mean value of  $\mu = 0$ . The larger  $\epsilon_b$  the larger may be the standard deviation  $\sigma$ . A good initial choice is  $\sigma = 0.01$ . The range in which perturbed values can be found can be stretched with the factor  $\alpha$ . The default value is  $\alpha = 1$ .

## 5 Experimental Study

In this section, the effect of the proposed data preprocessing methods on the precision of the cascade classification approach is evaluated on two data sets. The first data set is an energy time series data set of micro combined heat and power plant ( $\mu$ CHP) power production time series. The second data set is an artificial complex data set where the small interesting class has a *Hyperbanana* shape. *Banana* and *Hyperbanana* data sets are often used to test new classifiers, because they are considered as difficult classification tasks. Therefore we take the test with the *Hyperbanana* data set as a representative result.

The experimental study is done with cascade classifiers on each data set. Altogether three classification experiments are conducted on both data sets. The first experiment is done without preprocessing (no prepro.), the second with selected feasible examples (fs) and the third with selected feasible and artificial infeasible examples (fs + infs). For all experiments a one-class baseline classifier is used. The third experiment is also done with binary baseline classifiers.

The experimental study is divided into a description of the data sets, the experimental setup and the results.

### 5.1 Data Sets

The experiments are conducted with simulated  $\mu$ CHP power output time series and an artificial *Hyperbanana* data set. Both data sets have 96 dimensions (time steps, resp. features).

**$\mu$ CHP.** A  $\mu$ CHP is a small decentralized power and heat generation unit. The  $\mu$ CHP power production time series are simulated with a  $\mu$ CHP simulation model<sup>1</sup>. The  $\mu$ CHP simulation model includes a  $\mu$ CHP model, a thermal buffer and the thermal demand of a detached house. A  $\mu$ CHP can be operated in different modes, where its technical constraints, the constraints of the thermal buffer and the conditions of the thermal demand of the building are complied. Power output time series can be either feasible or infeasible depending on these constraints. The  $\mu$ CHP simulation model calculates the power production time

<sup>1</sup> Data are available for download on our department website <http://www.uni-oldenburg.de/informatik/ui/forschung/themen/cascade/>.

series for feasible operation modes, but also infeasible power output time series can be generated, where at least one constraint is violated. Due to the different constraints the class of feasible power production time series consists of several clusters. For convenience only such feasible power output time series are chosen, where the power production is greater than 0 at each time step. Infeasible power output time series are sampled from the whole volume of the infeasible class. In data space the class of infeasible power output time series occupies a much larger volume than the class of feasible ones, [6]. The classes are severely imbalanced, but the experiments are conducted with equal numbers of examples from both classes.

The feasible and infeasible  $\mu$ CHP power output time series are scaled according to the maximal power production to values between 0 and 1.

**Hyperbanana.** As far as there is no 96-dimensional *Hyperbanana* data set, we have generated a data set from the extended d-dimensional Rosenbrock function, [21].

$$f(x) = \sum_{i=1}^{d-1} [100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2] \quad (3)$$

The small and interesting class, or here also called feasible class is sampled from the Rosenbrock valley with  $f(x) < 100$  and the infeasible class with  $f(x) \geq 100$  is sampled only near the class boundary to test the sensitivity of the decision boundaries of the classifiers.

Sampling of the banana shaped valley is done by disturbing the minimum of the extended 96-dimensional Rosenbrock function with gaussian distributed values  $(\mathcal{N}(0, 1) \cdot \beta)$  with  $\beta \in \{40, 50, 60, 70\}$ . The minima of the Rosenbrock function are presented in [21] for different dimensionalities, but the minimum for 96 dimensions is missing. Therefore we approximated the minimum with regard to the other minima with  $-0.99$  for the first dimension and  $0.99$  for all other dimensions. The procedure of disturbing and selecting values from the Rosenbrock valley is repeated with the sampled values until enough data points are found. As far as it is difficult to sample the banana “arms” all at the same time, we sampled them separately by generating points that are <or> than a certain value and continued sampling by repeating disturbance and selection with these values. Values from all these repetitions were aggregated to one data set and shuffled. Finally all dimensions (features)  $x_i$  of the data set are scaled to values between 0 and 1 by  $x_i = [x_i + (\min(x_i) + \text{offset})] / [\max(x_i) + \text{offset} - \min(x_i) + \text{offset}]$  with  $\text{offset} = 0.2$ .

The samples generated by this procedure are not homogeneously distributed in the Rosenbrock valley and they do not represent all *Hyperbanana* “arms” equally.

The 96-dimensional infeasible examples near the class boundary are sampled in the same way as the feasible ones but starting with the feasible *Hyperbanana* samples and selecting samples in the range  $100 \leq f(x) \leq 500$ .

## 5.2 Experimental Setting

The experimental setting is divided into two parts: data preprocessing and classification. All calculations are done in Python. The first part, data preprocessing (selection of feasible examples and generation of infeasible examples) is done according to Sects. 4.1 and 4.2.

Selection of feasible examples is parametrized differently for both data sets as a result of pre-studies. The pre-studies were conducted with different minimal distances  $\epsilon$  and  $\epsilon_b$  and evaluated according to the number of resulting feasible examples  $sn$  and their distribution in the 2-dimensional data subset. For the  $\mu$ CHP data set instance selection is parametrized as follows, the minimal distance between feasible examples is set to  $\epsilon = 0.001$  and the number of new examples used for each iteration  $t$  is set to  $t = 1000$ . Generation of artificial infeasible examples is parameterized with  $n = 15000$  initially feasible examples disturbance  $= \mathcal{N}(0, 0.01) \cdot \alpha$  with  $\alpha = 1$  and minimal distance between infeasible examples and their nearest feasible neighbors  $\epsilon_b = 0.025$ . For the *Hyperbanana* data set the instance selection parameters are set to  $\epsilon = 0.002$  and  $t = 1000$  and parameters for generating artificial infeasible examples are set to  $n = 20000$ , disturbance  $= \mathcal{N}(0, 0.02) \cdot \alpha$  with  $\alpha = 1$  and  $\epsilon_b = 0.025$ .

The second part of the experimental study, the three classification experiments, are done with the cascade classifier, see Sect. 3, with different baseline classifiers from SCIKIT-LEARN, [20], a One-Class SVM (OCSVM) and two binary classifiers, k-nearest neighbors (kNN) and Support Vector Machines (SVMs). The OCSVM baseline classifier is used for all three experiments. The two binary classifiers kNN and binary SVM are used for the third experiment with both preprocessing methods (fs + infs).

All experiments are conducted identically on both data sets except for the parametrization. For all experiments the number of feasible training examples  $N$  is varied in the range of  $N = \{1000, 2000, \dots, 5000\}$  for the  $\mu$ CHP data set and  $N = \{1000, 2000, \dots, 10000\}$  for the *Hyperbanana* data set. For binary classification  $N$  infeasible examples are added to the  $N$  feasible training examples.

Parameter optimization is done with grid-search on separate validation sets with the same number of feasible examples  $N$  as the training sets and also  $N$  artificial infeasible examples for the third experiment. For the first experiment (no prepro.) and the second experiment (fs) the parameters are optimized according to true positive rates (TP rate or only TP), (TP rate = (true positives)/(number of feasible examples)).

For the third experiment, where the validation is done with  $N$  additional infeasible examples, parameters are optimized according to accuracy (acc = (true positives + true negatives)/(number of positive examples + number of negative examples)). The OCSVM parameters are optimized in the ranges  $\nu \in \{0.0001, 0.0005, 0.001, 0.002, \dots, 0.009, 0.01\}$ ,  $\gamma \in \{50, 60, \dots, 200\}$ , the SVM parameters in  $C \in \{1, 10, 50, 100, 500, 1000, 2000\}$ ,  $\gamma \in \{1, 5, 10, 15, 20\}$  and the kNN parameter in  $k \in \{1, 2, \dots, 26\}$ .

Evaluation of the trained classifiers is done on a separate independent data set with 10000 feasible and 10000 real infeasible 96-dimensional examples according

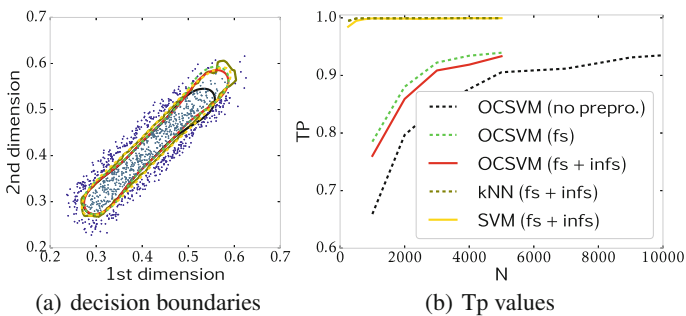
to TP and TN values for varying numbers of training examples  $N$ . The classification results could be evaluated with more advanced measures, see e.g. [11, 14]. For better comparability of the results on both data sets and the option to distinguish effects on the classification of feasible and infeasible examples we use the simple TP and TN values. TN values on both data sets are difficult to compare, because the infeasible  $\mu$ CHP power output time series are distributed in the whole region of infeasible examples, while the infeasible *Hyperbanana* examples are distributed only near the class boundary. As far as most classification errors occur near the class boundary, the TN values of the *Hyperbanana* set are expected to be lower than the TN values on the  $\mu$ CHP data set.

### 5.3 Results

The proposed data preprocessing methods, selection of feasible examples and generation of artificial infeasible examples show an increase in classification precision of the cascade classifier in the experiments.

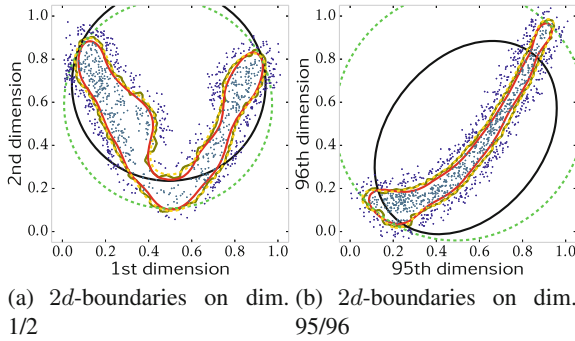
On both data sets ( $\mu$ CHP and *Hyperbanana*) data preprocessing leads to more precise decision boundaries than without data preprocessing, see Figs. 3(a) and 4. This can be also seen in the TP and TN values of the classification results, see Figs. 3(b) and 5.

For the  $\mu$ CHP data set, all three experiments lead to TN values of 1, therefore only the TP values are plotted in Fig. 3(b). But high TN values for the  $\mu$ CHP data set do not necessarily mean, that further infeasible time series are classified correctly. The applied infeasible test examples are taken from the whole volume of the large infeasible class and therefore most of the examples are not located



**Fig. 3.** Decision boundaries and TP values. The left figure shows the decision boundaries on the 1st and 2nd dimension of the  $\mu$ CHP data set trained with  $N = 1000$  feasible (+1000 infeasible) training examples, no prepro. (dashed black), fs (dashed green), OCSVM (fs + infs) (red), kNN (fs + infs) (olive) and SVM (fs + infs) (yellow). The gray points indicate 500 of the selected feasible training examples and the blue points 500 of the artificial infeasible examples. The right figure shows the corresponding TP values on the high-dimensional data set. (Color figure online)

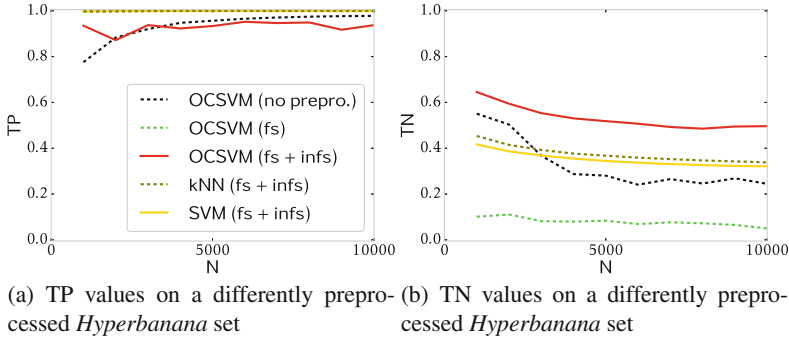
near the class boundary. The first experiment without data preprocessing (no prepro.) yields the lowest TP values of all experiments for all numbers of training values  $N$  and the second experiment with selection of feasible examples (fs) leads already to higher TP values. The third experiment with selection of feasible examples and artificial infeasible examples (fs + infs) leads to different results with the OCSVM baseline classifier and the binary SVM and kNN baseline classifiers. While the OCSVM (fs + infs) achieves slightly lower TP values than OCSVM (fs) in the second experiment, the binary baseline classifiers SVM (fs + infs) and kNN (fs + infs) achieve TP values near 1.



**Fig. 4.** Decision boundaries on the *Hyperbanana* data set trained with  $N = 1000$  feasible (+1000 infeasible) training examples, no prepro. (dashed black), fs (dashed green), OCSVM (fs + infs) (red), kNN (fs + infs) (olive) and SVM (fs + infs) (yellow). The gray points indicate 500 of the selected feasible training examples and the blue points 500 of the artificial infeasible examples. (Color figure online)

For the *Hyperbanana* data set with a more complex data structure, data preprocessing influences the TP values, see Fig. 5(a) and the TN values, Fig. 5(b) of the classification results. In the first experiment (no prepro.) and second experiment (fs) the classification achieves relatively high TP values and at the same time the lowest TN values of all experiments due to an overestimation of the feasible class, see Fig. 4. The third experiment (fs + infs) revealed an opposed behavior of the OCSVM baseline classifier and the SVM and kNN baseline classifiers. The OCSVM (fs + infs) achieves lower TP values than the OCSVM in the previous experiments but also the highest TN values of all experiments. SVM and kNN baseline classifiers with (fs + infs) achieve the highest TP values of all experiments and at the same time lower TN values than the OCSVM (fs + infs).

In summary, data preprocessing increases the classification precision of the cascade classifier on both data sets. While the selection of feasible examples increases the classification precision, artificial infeasible examples can lead to an even greater increase depending on the data set and the baseline classifier.



**Fig. 5.** TP and TN values on the *Hyperbanana* data set for different preprocessing steps and different baseline classifiers. The legend in Fig. 5(a) is also valid for Fig. 5(b). The green line of OCSVM (fs) in Fig. 5(a) is covered by the olive and the yellow lines. (Color figure online)

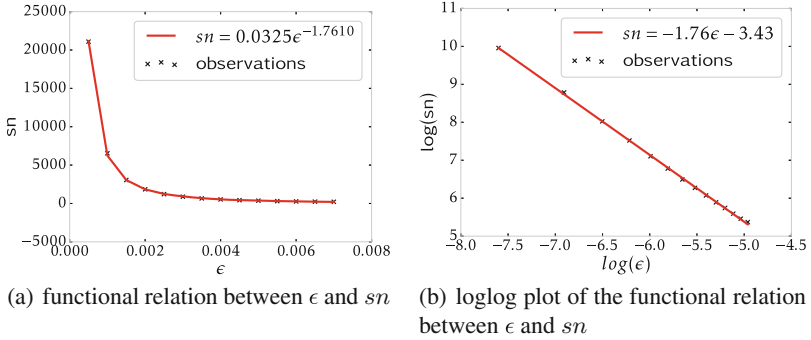
## 6 Parameter Sensitivity Study and Parameter Fitting Rules

In this section the influence of the data preprocessing distance parameters on the cascade classifier precision is analyzed and parameter fitting rules are derived. Therefore selection of feasible instances is applied first with different values of the minimal distance  $\epsilon$  between feasible nearest neighbors. Then outlier generation is applied to the different data sets of selected feasible examples. Outlier generation is conducted with different minimal distances  $\epsilon_b$  between infeasibles and their nearest feasible neighbors. Next a cascade classifier is built on each preprocessed data set and the classification precision is tested on a test set. The cascade classifier precision is measured as true positive rates (TP) and true negative rates (TN). Based on high TP and TN values a region with optimal distance parameter value combinations is identified.

### 6.1 Experimental Setup

The classification precision experiments are conducted with various combinations of both data preprocessing distance parameter values. The experiments are performed on the  $\mu$ CHP data set from Sect. 5.1, consisting of 239,131 feasible and 1,000,000 infeasible examples.

First of all both data preprocessing methods are applied and after that the preprocessed data sets are classified with the cascade classification model. Data preprocessing starts with the selection of feasible examples on all 2-dimensional data subsets of the  $\mu$ CHP data set. The parameter  $t = 100$  remains constant and the value of the distance parameter  $\epsilon$  is increased. The  $\epsilon$  values are chosen with respect to the range of the power production values  $[0, 1]$  at each time step and the number of required training and validation examples. The number of selected feasible examples  $sn$  decreases for increasing values of  $\epsilon$ , see Fig. 6.



**Fig. 6.** Functional relationship between the minimal distance between nearest feasible neighbors  $\epsilon$  and the number of selected examples  $sn$  of the subset with the fewest selected examples. The selected feasible examples have to be divided into training and validation sets or can be used only as training data while validation examples are taken from the remaining feasible examples, that were not selected.

But the number of selected feasible examples  $sn$ , resulting from one  $\epsilon$  value, differ among the low-dimensional data subsets of the high-dimensional data set. Therefore the smallest number  $sn$  of all low-dimensional data sets is used for all low-dimensional data subsets resulting from the same  $\epsilon$  value. Overall the adapted number of selected examples  $sn$  decreases for increasing values of  $\epsilon$  with a power function  $sn = 0.0325\epsilon^{-1.7610}$ , see Fig. 6. In the previous experiments with data preprocessing for the  $\mu$ CHP data set in Sect. 5, training sets with more than  $N = 250$  feasible training examples turned out to be reasonable for the kNN baseline classifier, see Fig. 3(b). With respect to the minimum number of feasible training examples  $N$  and the constant parametrization of  $t = 100$ , we have chosen the number of feasible training examples as  $N \in [100, 10000]$  examples. In the experiments we employ  $N = sn/2$  as training examples and the remaining  $sn/2$  examples as validation values, therefore  $sn$  has to be twice as large as the number of training examples  $sn = 2N$ . These numbers of feasible training examples and the respective  $sn$  values correspond to  $\epsilon \in \{0.001, 0.0015, 0.002, \dots, 0.0055\}$ .

Then artificial infeasible outliers are generated for each of the new data sets  $S$  consisting of  $sn$  selected feasible examples. Outlier generation is parametrized as follows. Noise is taken from  $\mathcal{N}(\mu, \sigma) \cdot \alpha$  with  $\mu = 0$ ,  $\sigma = 0.01$  and  $\alpha = 1$  and  $\epsilon_b$  is increased for all data sets generated with the different  $\epsilon$  values. The  $\epsilon_b$  values are chosen from  $\{0.001, 0.002, \dots, 0.05\}$  with  $\epsilon_b \geq \epsilon$ . Depending on  $\epsilon_b$  the number of algorithm iterations is adapted until at least the same number of outlier examples are generated as the respective number of selected feasible examples  $sn$ .

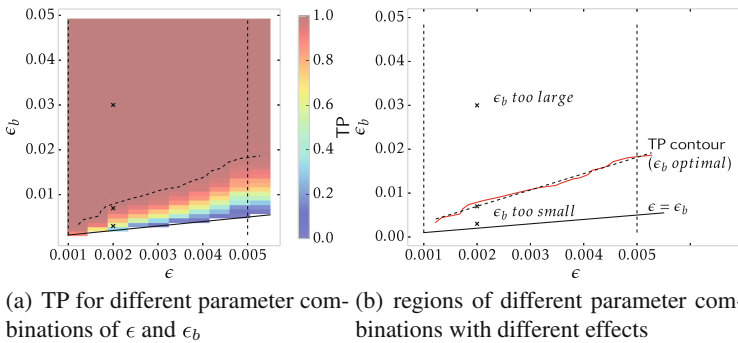
Next the cascade classifiers are built on all preprocessed data sets with the k nearest neighbor baseline classifier from *scikit learn*, [20]. The training sets contain ( $N = n/2$ ) half of the number of selected feasible examples. Additionally

the training sets contain the same number of artificial infeasible examples as feasible training examples. The number of nearest neighbors  $k$  for each classifier is taken from  $k \in \{1, 2, \dots, 26\}$  and optimized with a validation set. The validation set contains the same number of feasible and infeasible examples as the training set. Feasible and infeasible examples are taken from the remaining selected feasible ones and the remaining artificial infeasibles.

The classifiers are tested on a set of 10,000 feasible and 10,000 infeasible high-dimensional examples without data preprocessing. TP and TN values of the cascade classifier are stored for all preprocessed data sets. The cascade classifier precision is evaluated graphically on the achieved TP and TN values according to the underlying data preprocessing distance parameter value combinations.

### 6.2 Results

The cascade classifier precision yielded different TP values for the differently preprocessed data sets shown in Fig. 7(a) and  $TN = 1$ . The high TN values are due to the location of the infeasible high-dimensional test examples far away from the class boundary, see Sect. 5.1.

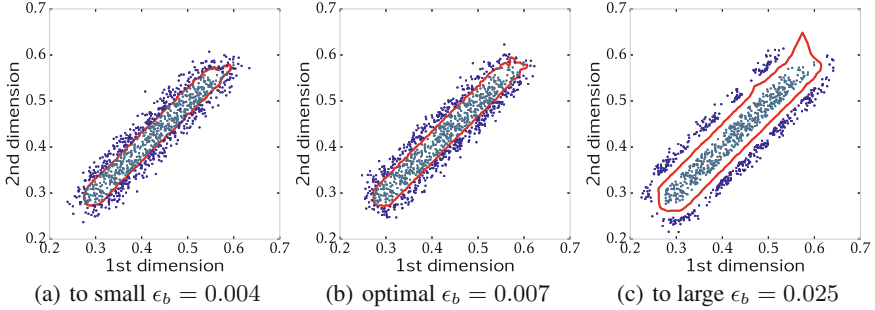


**Fig. 7.** The left figure shows the TP values for the differently preprocessed  $\mu$ CHP data set with the corresponding distance parameter values. The two vertical dashed black lines mark the range of reasonable  $\epsilon$  values and the solid black line at the bottom indicates the lowest bound for  $\epsilon_b$ :  $\epsilon = \epsilon_b$ . The black line above is the contour with  $TP = 0.99$ . The figure on the left shows the same boundary lines. Furthermore the resulting regions are indicated. The three points in both figures mark the parameter combinations used for Fig. 8.

Based on the TP values different regions of classification precision are identified and separated by the restrictions resulting from the distance parameters, see Fig. 7(b). The regions of different distance parameter combinations are bounded for this  $\mu$ CHP data set in the  $\epsilon$  range and the  $\epsilon_b$  range. The  $\epsilon$  range is bounded by the number of selected feasible examples, see Sect. 6.1. After excluding insensible  $\epsilon$  values ( $\epsilon < 0.001$  and  $\epsilon > 0.0055$ ), the  $\epsilon_b$  values can be divided into three



groups, too small  $\epsilon_b$  values, optimal values and too large values. For each of these groups the distribution of the preprocessed data sets and the learned decision boundaries are shown exemplarily in Fig. 8.



**Fig. 8.** Decision boundaries learned for the first and second dimension on the  $\mu$ CHP data set for  $\epsilon_b$  values from the different regions and a fixed value of  $\epsilon = 0.002$ . These parameter value combinations are marked in Fig. 7. (Color figure online)

The region of too small  $\epsilon_b$  values is bounded by  $\epsilon_b = \epsilon$  at the lower bound and the  $TP = 0.99$  contour at the upper bound. The  $TP$  contour can be approximated by a linear function with linear regression:  $\epsilon_b = 3.7251\epsilon - 0.0005$ . In the region of too small  $\epsilon_b$  values more or less artificial infeasible examples are generated in the region of the feasible class. This can be seen in Fig. 8(a) showing some artificial infeasible examples (blue points) between the gray points in the region of feasible. This mixture of feasible and artificial infeasible examples leads to non smooth class boundaries and relatively low  $TP$  values.

The region of optimal  $\epsilon_b$  values is only described by the  $TP = 0.99$  contour for the  $\mu$ CHP data set, because  $TN = 1$  for all parameter value combinations. If there would be infeasible test examples near the true class boundary, the  $TN$  values should decrease for increasing  $\epsilon_b$  values. From these decreasing  $TN$  values a respective  $TN$  contour could be computed. This  $TN$  contour could be used in combination with the  $TP$  contour to determine the optimal value region. The corresponding artificial infeasible examples are distributed around the feasible ones with a small gap in between the classes and hardly any overlap, see Fig. 8(b). Due to the gap feasible and infeasible examples are clearly separable and  $TP$  and  $TN$  values are high in this region.

The region of too large  $\epsilon_b$  values has a lower bound resulting from the  $TP$  contour for the  $\mu$ CHP data set. But the lower bound could be also determined by a  $TN$  contour, if there would be infeasible test examples near the true class boundary. As far as the  $\mu$ CHP data set contains hardly any infeasible examples near the class boundary, the lower bound cannot be determined by a  $TN$  contour. But wrong classification results are due to incorrect decision boundaries. If the  $\epsilon_b$  value is too large, feasible and infeasible examples are clearly separable and there is a large gap between the examples of both classes, see Fig. 8(c). The

learned decision boundary is located in the middle of this gap. The larger the gap, the further is the decision boundary away from the feasible examples. This phenomenon is an overestimation of the feasible class and infeasible test examples near the true class boundary would be classified as feasible.

Even though the parameter  $\epsilon_b$  has to be chosen carefully, the parameter sensitivity analysis yielded a region of optimal data preprocessing distance parameter value combinations, where the cascade classifier performs best. Deviations from the optimal parameter value combinations lead either to an over- or underestimation of the feasible class with decreasing (TN) or TP values.

## 7 Conclusions

In this paper, we presented two data preprocessing methods to improve the precision of the cascade classification model (selection of feasible examples and generation of artificial infeasible examples). Both methods operate on the low-dimensional data sets. Selection of feasible examples leads to more representative training data and artificial infeasible examples lead to more precise decision boundaries, due to the availability of infeasible examples near the class boundary. Depending on the baseline classifier, the application of both data preprocessing methods yields for the  $\mu$ CHP power output time series data set and an artificial and complex *Hyperbanana* data set the best classification precision. The application of only selection of feasible examples and no data preprocessing yielded always worse results, that can be observed as lower TP values on the  $\mu$ CHP data set and especially very low TN values on the *Hyperbanana* data set. Additionally the parameter sensitivity of the distance parameters of both data preprocessing methods was analyzed with respect to the cascade classifier precision on the  $\mu$ CHP data set. The analysis yielded a region of optimal parameter value combinations and their boundaries. Parameter value combinations from outside the optimal region lead to a lower classification precision with either lower TP or lower TN values. In the first case with lower TP values the precision decreases due to an overlap of feasible and infeasible training examples. This overlap causes an under estimation of the feasible class. In the second case low TN values near the class boundary are due to a large gap between feasible and infeasible examples. This gap leads to an over estimation of the feasible class.

Overall we recommend a careful parametrization of the data preprocessing methods with some pre-test to increase the cascade classifier precision.

We intend to repeat the sensitivity analysis on the *Hyperbanana* data set to study the behavior of infeasible examples near the class boundary, because the analyzed  $\mu$ CHP data set does not have infeasible examples near the class boundary. Furthermore we plan a comparison of traditional one class classifiers with the cascade classification model with preprocessing.

**Acknowledgement.** This work was funded by the Ministry for Science and Culture of Lower Saxony with the PhD program System Integration of Renewable Energy (SEE).

## References

1. Bagnall, A., Davis, L.M., Hills, J., Lines, J.: Transformation based ensembles for time series classification. In: Proceedings of the 12th SIAM International Conference on Data Mining, pp. 307–318 (2012)
2. Bánhalmi, A., Kocsor, A., Busa-Fekete, R.: Counter-example generation-based one-class classification. In: Kok, J.N., Koronacki, J., Mantaras, R.L., Matwin, S., Mladenich, D., Skowron, A. (eds.) ECML 2007. LNCS (LNAI), vol. 4701, pp. 543–550. Springer, Heidelberg (2007). doi:[10.1007/978-3-540-74958-5\\_51](https://doi.org/10.1007/978-3-540-74958-5_51)
3. Bellinger, C., Sharma, S., Japkowicz, N.: One-class versus binary classification: which and when? In: 11th International Conference on Machine Learning and Applications, ICMLA 2012, vol. 2, pp. 102–106, December 2012
4. Blachnik, M.: Ensembles of instance selection methods based on feature subset. *Procedia Comput. Sci.* **35**, 388–396 (2014). Knowledge-Based and Intelligent Information and Engineering Systems 18th Annual Conference, KES-2014 Gdynia, Poland, September 2014 Proceedings
5. Borgonovo, E., Plischke, E.: Sensitivity analysis: a review of recent advances. *Eur. J. Oper. Res.* **248**(3), 869–887 (2016)
6. Bremer, J., Rapp, B., Sonnenschein, M.: Support vector based encoding of distributed energy resources' feasible load spaces. In: Innovative Smart Grid Technologies Conference Europe IEEE PES (2010)
7. Cortez, P., Embrechts, M.: Opening black box data mining models using sensitivity analysis. In: 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM), pp. 341–348, April 2011
8. Cortez, P., Embrechts, M.J.: Using sensitivity analysis and visualization techniques to open black box data mining models. *Inf. Sci.* **225**, 1–17 (2013)
9. Garcia, S., Derrac, J., Cano, J., Herrera, F.: Prototype selection for nearest neighbor classification: taxonomy and empirical study. *IEEE Trans. Pattern Anal. Mach. Intell.* **34**(3), 417–435 (2012)
10. Hamby, D.M.: A review of techniques for parameter sensitivity analysis of environmental models. *Environ. Monit. Assess.* **32**, 135–154 (1994)
11. He, H., Garcia, E.: Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **21**(9), 1263–1284 (2009)
12. Heiselberg, P., Brohus, H., Hesselholt, A., Rasmussen, H., Seinre, E., Thomas, S.: Application of sensitivity analysis in design of sustainable buildings. *Renew. Energy* **34**(9), 2030–2036 (2009). Special Issue: Building and Urban Sustainability
13. Jankowski, N., Grochowski, M.: Comparison of instances selection algorithms I. Algorithms survey. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 598–603. Springer, Heidelberg (2004). doi:[10.1007/978-3-540-24844-6\\_90](https://doi.org/10.1007/978-3-540-24844-6_90)
14. Japkowicz, N.: *Assessment Metrics for Imbalanced Learning*, pp. 187–206. Wiley, Hoboken (2013)
15. Kleijnen, J.P.C.: *Design and Analysis of Simulation Experiments*. International Series in Operations Research and Management Science. Springer, Heidelberg (2015)
16. Lin, W.J., Chen, J.J.: Class-imbalanced classifiers for high-dimensional data. *Brief. Bioinform.* **14**(1), 13–26 (2013)
17. Liu, H., Motoda, H., Gu, B., Hu, F., Reeves, C.R., Bush, D.R.: *Instance Selection and Construction for Data Mining*. The Springer International Series in Engineering and Computer Science, vol. 608, 1st edn. Springer US, New York (2001)

18. Neugebauer, J., Kramer, O., Sonnenschein, M.: Classification cascades of overlapping feature ensembles for energy time series data. In: Woon, W.L., Aung, Z., Madnick, S. (eds.) DARE 2015. LNCS (LNAI), vol. 9518, pp. 76–93. Springer, Heidelberg (2015). doi:[10.1007/978-3-319-27430-0\\_6](https://doi.org/10.1007/978-3-319-27430-0_6)
19. Neugebauer, J., Kramer, O., Sonnenschein, M.: Improving cascade classifier precision by instance selection and outlier generation. In: ICAART, vol. 8 (2016, in print)
20. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
21. Shang, Y.W., Qiu, Y.H.: A note on the extended Rosenbrock function. *Evol. Comput.* **14**(1), 119–126 (2006)
22. Tax, D.M.J., Duin, R.P.W.: Uniform object generation for optimizing one-class classifiers. *J. Mach. Learn. Res.* **2**, 155–173 (2002)
23. Tomašev, N., Buza, K., Marussy, K., Kis, P.B.: Hubness-aware classification, instance selection and feature construction: survey and extensions to time-series. In: Stańczyk, U., Jain, L.C. (eds.) *Feature Selection for Data and Pattern Recognition*. SCI, vol. 584, pp. 231–262. Springer, Heidelberg (2015). doi:[10.1007/978-3-662-45620-0\\_11](https://doi.org/10.1007/978-3-662-45620-0_11)
24. Tsai, C.F., Eberle, W., Chu, C.Y.: Genetic algorithms in feature and instance selection. *Knowl.-Based Syst.* **39**, 240–247 (2013)
25. Wilson, D., Martinez, T.: Reduction techniques for instance-based learning algorithms. *Mach. Learn.* **38**(3), 257–286 (2000)
26. Wu, J., Dhingra, R., Gambhir, M., Remais, J.V.: Sensitivity analysis of infectious disease models: methods, advances and their application. *J. R. Soc. Interface* **10**(86), 1–14 (2013)
27. Zhuang, L., Dai, H.: Parameter optimization of kernel-based one-class classifier on imbalance text learning. In: Yang, Q., Webb, G. (eds.) *PRICAI 2006*. LNCS (LNAI), vol. 4099, pp. 434–443. Springer, Heidelberg (2006). doi:[10.1007/978-3-540-36668-3\\_47](https://doi.org/10.1007/978-3-540-36668-3_47)