A R T I C L E  I N F O

Article history:
Received 25 October 2012
Received in revised form
8 August 2013
Accepted 9 August 2013
Available online 4 January 2014

Keywords:
Cost-sensitive learning approach
Selective naive Bayes
h-index
Neurosciences journals

A B S T R A C T

Machine learning community is not only interested in maximizing classification accuracy, but also in minimizing the distances between the actual and the predicted class. Some ideas, like the cost-sensitive learning approach, are proposed to face this problem. In this paper, we propose two greedy wrapper forward cost-sensitive selective naive Bayes approaches. Both approaches readjust the probability thresholds of each class to select the class with the minimum-expected cost. The first algorithm (CS-SNB-Accuracy) considers adding each variable to the model and measures the performance of the resulting model on the training data. The variable that most improves the accuracy, that is, the percentage of well classified instances between the readjusted class and actual class, is permanently added to the model. In contrast, the second algorithm (CS-SNB-Cost) considers adding variables that reduce the misclassification cost, that is, the distance between the readjusted class and actual class. We have tested our algorithms on the bibliometric indices prediction area. Considering the popularity of the well-known h-index, we have researched and built several prediction models to forecast the annual increase of the h-index for Neurosciences journals in a four-year time horizon. Results show that our approaches, particularly CS-SNB-Accuracy, achieved higher accuracy values than the analyzed cost-sensitive classifiers and Bayesian classifiers. Furthermore, we also noted that the CS-SNB-Cost always achieved a lower average cost than all analyzed cost-sensitive and cost-insensitive classifiers. These cost-sensitive selective naive Bayes approaches outperform the selective naive Bayes in terms of accuracy and average cost, so the cost-sensitive learning approach could be also applied in different probabilistic classification approaches.

1. Introduction

Classification problems commonly assume that the class values are unordered. But, these values have a natural order in many practical applications. Given ordered classes, we are not only interested in maximizing classification accuracy, but also in minimizing the distances between the actual and the predicted class. Some fields, like statistics, have faced this problem for many years, developing several approaches [37,38], whereas other fields, like machine learning, have only recently started to look at the problem [7,8,18,19,28,34,40,43].

This paper is focused on solving the above problem using the cost-sensitive learning approach. Cost-sensitive learning algorithms take into account matrices of misclassification cost to express relative distances between classes. This approach incorporates decision-making costs to define fixed and unequal misclassification costs between classes. The cost model takes the form of a cost matrix, where the cost of classifying a sample from a true class \( j \) in class \( i \) corresponds to the matrix entry \( m_{ij} \). The diagonal elements of this matrix are usually set to zero, meaning correct classification has no cost. The theory of cost-sensitive learning is summarized in [16,51], describing how the misclassification cost plays a key role in different situations. Cost-sensitive algorithms can be divided into several categories. Algorithms belonging to the first category (direct methods) design classifiers that are naturally cost-sensitive, using directly the misclassification costs in the learning algorithms. In contrast, the second category (indirect methods) converts existing cost-insensitive classifiers into cost-sensitive classifiers.

We incorporate cost-sensitive learning and feature subset selection into the well-known naive Bayes [39], which is the most straightforward and widely tested method for probabilistic induction and has long been used within the field of pattern recognition [11]. For this reason, we develop new cost-sensitive algorithms based on the selective naive Bayes notions [33]. Specifically, we develop two direct algorithms that add misclassification costs to the learning algorithm, and use wrapper approaches to select...
relevant variables that maximize the accuracy (CS-SNB-Accuracy algorithm) and minimize the cost (CS-SNB-Cost algorithm). The objective of these approaches is to build parsimonious models. These models will not include features that are irrelevant and redundant. Some benefits of applying variable selection are better classification performance, faster classification models, smaller databases, and the ability to gain more insight into the process that is being modeled.

We have tested our algorithms on the bibliometric indices prediction area. Bibliometric indices (see reviews [1,14]) are quantitative metrics for evaluating and comparing the research activity of researchers, journals, institutions or countries according to their output. Bibliometric indices are an increasingly important topic for the scientific community nowadays. In fact, many funding agencies and promotion committees use them to assess research projects, recruit researchers and so on.

The interest and originality of our study is two-fold. First, we develop two new classifiers (CS-SNB-Accuracy and CS-SNB-Cost) that bring together the advantages of using the cost-sensitive learning approach and the feature subset selection. Second, both classifiers are used to predict the annual increase of the h-index for scientific journals belonging to the Journal Citation Report Neurosciences category across a 4-year time horizon using bibliometric indices.

The remainder of the paper is organized as follows. The next section reviews some related work. Section 3 explains some concepts related to Bayesian classifiers and cost-sensitive Bayesian classifiers, focusing on our new cost-sensitive selective naive Bayes approaches. Also, we review some standard classifiers and statistical tests. Section 4 presents our results, including dataset construction, data distribution, accuracy and average cost of models and some examples. Finally, Section 5 outlines some conclusions emphasizing the original contribution of the paper and future research on the topic.

2. Related work

Researchers have tried to solve the problem of not only maximizing classification accuracy, but also minimizing the distances between the actual and the predicted class. Different approaches have been proposed in the literature. For example, Kramer et al. [30] transformed the ordinal scales into numeric values, and then solve the problem as a standard regression problem. Herbrich et al. [21,22] applied the principle of structural risk minimization used in support vector machines to learn an algorithm based on large margin rank boundaries. Finally, Frank and Hall [17] used binary decomposition techniques, transforming the original problem involving \( k \) classes into \( k - 1 \) binary problems.

The cost-sensitive learning approach, which is analyzed in this paper, is also used for the above purpose. Direct cost-sensitive algorithms design classifiers that are naturally cost sensitive. The main idea is for misclassification costs to be entered and used directly in the learning algorithms. Several researchers have proposed direct cost-sensitive learning algorithms, such as inexpensive classification with expensive tests [48], which use misclassification costs in the fitness function of genetic algorithms and cost-sensitive decision trees, which use misclassification costs in the tree building process [35], and in the tree pruning process [10]. In contrast, indirect cost-sensitive algorithms convert existing cost-insensitive classifiers into cost-sensitive classifiers. These classifiers can be further categorized into relabeling methods, weighting methods and sampling methods. Relabeling methods [9,49] relabel the classes of instances by applying the minimum expected cost criterion [29]. This criterion is defined by fixed misclassification costs and posterior probabilities. These methods can be further divided into two branches: relabeling training instances (e.g. MetaCost [9]) and relabeling test instances (e.g. CostSensitiveClassifier [49]). Weighting methods [47] assign a weight to each instance in terms of its class according to misclassification costs, that is, instances, which carries a higher misclassification cost, are assigned proportionally high weights. These methods induce cost-sensitivity by directly integrating instance weights. They work whenever the original cost-insensitive classifiers can accept example weights. In this case, the learning algorithm favors the class with high weight/cost. Sampling methods [44,52] modify the class distribution of training data according to their costs and then directly apply cost-insensitive classifiers on the sampled data. Costing [52] uses rejection sampling, whereas CSRoulette [44] uses a cost-proportional roulette sampling technique to change the distribution of the training set according to the cost matrix. The difference between the above methods is that Costing generates much smaller samples than the original training set, whereas CSRoulette generates samples with the same size as the original training set.

According to the bibliometric indices prediction area, some studies have been proposed in the literature for this purpose (e.g. [3,31]). These predictions used time series modeled by exponential and exponential smoothing functions. Other methods, like Bayesian networks, logistic regression, decision trees and the K-NN algorithm, were also used to make predictions [25]. Focusing on the h-index, we noted that not many papers have tried to predict this bibliometric index. The power law model [15] was used to analyze the h-index as a function of time [12]. Nonlinear regression was also used to predict the h-index of authors, journals and universities [50]. Most research concerned with predicting the h-index used only h-index sequences to indicate by extrapolation what the value of the h-index would be in the near future. In a previous study [26], we developed several prediction models to forecast the h-index of Spanish professors for a 4-year time horizon using a cost-sensitive naive Bayes approach. Although the above papers all have a similar aim, the dataset, class variable, predictive features and methods are different.

3. Methods

3.1. Bayesian classifiers

Naive Bayes is one of the simplest models for supervised classification. It is one of the most efficient and effective inductive learning algorithms for machine learning and data mining. A naive Bayes classifier has two types of variables: the class variable \( C \) and a set of predictive features \( X = \{X_1, X_2, \ldots, X_n\} \). Fig. 1 represents the
naive Bayes structure. The class variable \( C \) is discrete and takes values in the set \( \Omega(C) \). The predictive features can be divided into two sets: the set of discrete features \( \{X_1, \ldots, X_m\} \) and the set of continuous features \( \{X_{m+1}, \ldots, X_n\} \). This classifier is based on Bayes’ theorem under the assumption of conditional independence of predictor features given the class variable:

\[
c^* = \arg \max_{c \in \Omega(C)} \sum_{c \in \Omega(C)} p(c|x) \times p(x) \prod_{i=1}^{m} p(x_i|c) \prod_{j=m+1}^{n} \mathcal{N}(x_j, \mu_{cj}, \sigma_{cj}^2)
\]

(1)

Selective naive Bayes is a variant of naive Bayes that uses only a subset of the given variables to make predictions. It improves accuracy in domains with redundant and irrelevant variables. The learning component adds the capability to exclude attributes that introduce dependencies to the original naive Bayes classifier. This process consists of searching the space of attribute subsets. The direction of search could be forward or backward. A forward selection method would start with the empty set and successively add variables, whereas a backward elimination process would begin with the full set and remove unwanted variables. The search process stops adding or eliminating attributes when none of the alternatives improves classification accuracy.

3.2. Cost-sensitive Bayesian classifiers

The objective of cost-sensitive methods is to take into account misclassification costs different from 0 (hit) and 1 (miss). These methods are concerned with classification accuracy and classification costs. We develop two forward cost-sensitive selective naive Bayes approaches. The search process of the first approach (CS-SNB-Accuracy) is based on maximizing classification accuracy, that is, it includes variables that improve classification accuracy, whereas the search process of the second approach (CS-SNB-Cost) is based on minimizing misclassification costs, that is, it includes variables that reduce the distances between the actual and the predicted class.

Given a cost matrix and a set of predicted class probabilities for each instance \( x \), both approaches readjust the probability thresholds of each class to select the class with the minimum-expected cost. The expected cost of each prediction is obtained by multiplying the associated costs by the predicted class probabilities. Unlike selective naive Bayes, these approaches do not select the most likely class value of the posterior distribution, they select the class \( c^* \) that minimizes the expected cost of predictions given a new instance \( x \):

\[
c^* = \arg \min_{c \in \Omega(C)} \sum_{c \in \Omega(C)} \text{cost}(c|x) \ p(c|x)
\]

(2)

where

\[
p(c|x) = p(c) \prod_{i=1}^{m} p(x_i|c) \prod_{j=m+1}^{n} \mathcal{N}(x_j, \mu_{cj}, \sigma_{cj}^2)
\]

(3)

and \( \text{cost}(c|x) \) is the associated misclassification cost.

In short, the first approach (CS-SNB-Accuracy) considers adding each variable to the model and measures the performance of the resulting model on the training data. The variable that most improves the accuracy, that is, the percentage of well-classified instances in the predicted class \( c^* \) over the actual class, is permanently added to the model. In contrast, the second approach (CS-SNB-Cost) considers adding variables that reduce the misclassification cost between the predicted and the actual class.

**Algorithm 1.** Cost-sensitive selective naive Bayes – Accuracy model.

**Input:** Dataset (feature variables and class variable) and cost matrix.

**Output:** Accuracy and cost of the cost-sensitive selective naive Bayes - Accuracy model.

For \( k = 1 \) to folds do

```
// k-fold cross validation
trainingSet ← trainingSetGeneration(dataset, k);
testSet ← testSetGeneration(dataset, k);
// Training phase
model ← {class};
initialProbability ← estimateClassProb(trainingSet, model);
thresholdAccuracy ← max(initialProbability);
accuracyVector ← {};
continue ← true
while continue do
for variable ← 1 to size(numVariables) do
SW ← inModelVariable(variable);
if SW then
    for case ← 1 to size(trainingSet) do
        actualClass ← getActualClass(case);
predictedClass ← predictClass(case, model);
readjustedClass ← readjustClass(predictedClass, costMatrix);
if (actualClass = = readjustedClass) then hit = 1;
else hit = 0;
modelAccuracy ← calculateAccuracy(hit);
end
end
accuracyVector[variable] ← modelAccuracy
end
accuracy ← max(accuracyVector);
if (accuracy > thresholdAccuracy) then
    bestVariable ← selectBestVariable(accuracyVector);
    model ← addToModel(model, bestVariable);
    thresholdAccuracy ← accuracy;
else
    continue ← false;
end
end
// Test phase
for case ← 1 to size(testSet) do
    actualClass ← getActualClass(testSet, case);
predictedClass ← predictClass(testSet, model);
readjustedClass ← readjustClass(predictedClass, costMatrix);
if (actualClass = = readjustedClass) then hit = 1;
else hit = 0;
accuracy ← calculateAccuracy(hit);
cost ← calculateCost(actualClass, readjustedClass);
end
finalAccuracyVector ← addToVector(accuracy);
finalCostVector ← addToVector(cost);
end
finalAccuracy ← mean(finalAccuracyVector);
finalCost ← mean(finalCostVector);
```
Algorithm 2. Cost-sensitive selective naive Bayes – Cost model.

**Input**: Dataset (feature variables and class variable) and cost matrix

**Output**: Accuracy and cost of the cost-sensitive selective naive Bayes – Cost model

for $k \leftarrow 1$ to $folds$ do
  // $k$-fold cross validation
  trainingSet ← trainingSetGeneration(Dataset, $k$);
  testSet ← testSetGeneration(Dataset, $k$);
  // Training phase
  model← (class);
  initialProbability ← estimateClassProb(trainingSet, model);
  initialCost ← classCostEstimation(initialProbability, costMatrix);
  thresholdCost ← min(initialCost);
  accuracyVector ← [];
  continue ← true;
  while continue do
    for variable ← 1 to size(numVariables) do
      if sw = isModelVariable(variable) then
        for case ← 1 to size(trainingSet) do
          actualClass ← getActualClass(case);
          predictedClass ← predictClass(case, model);
          readjustedClass ← readjustClass(predictedClass, costMatrix);
          modelCost ← calculateCost(actualClass, readjustedClass);
        end
        costVector[variable] ← modelCost;
      end
      cost ← min(costVector);
      if (cost < thresholdCost) then
        bestVariable ← selectBestVariable(costVector);
        model ← addToModel(model, bestVariable);
        thresholdCost ← cost;
      else
        continue ← false;
      end
    end
    // Test phase
    for case ← 1 to size(testSet) do
      actualClass ← getActualClass(testSet, case);
      predictedClass ← predictClass(testSet, model);
      readjustedClass ← readjustClass(predictedClass, costMatrix);
      if (actualClass = readjustedClass) then
        hit = 1;
      else
        hit = 0;
      end
      accuracy ← calculateAccuracy(hit);
      cost ← calculateCost(actualClass, readjustedClass);
    end
    finalAccuracyVector ← addToVector(accuracy);
    finalCostVector ← addToVector(cost);
  end
  finalAccuracy ← mean(finalAccuracyVector);
  finalCost ← mean(finalCostVector);

3.2.2. Cost-sensitive selective naive Bayes – Cost

Algorithm 2 shows the pseudocode of the cost-sensitive selective naive Bayes – Cost model. This algorithm also chooses $k$-fold cross-validation as the procedure for estimating the accuracy and cost of the models. This algorithm initializes the model to the class variable, that is, there is not predictable variables in the model yet. After that, the algorithm saves the accuracy of the resulting model (thresholdAccuracy) for subsequent comparisons. The accuracy threshold is computed by means of estimateClassProb and max functions, which, respectively, compute the initial class probabilities given the training set and return the highest probability value, that is, the probability of the most frequent class. In each iteration, the algorithm checks if a specific variable belongs to the model. The isModelVariable function returns true or false according to the current model. The algorithm considers adding each unused variable to the model on a trial basis and measures the performance of the resulting model on the training data. First, the predictClass function computes the predicted class, that is, the most likely class value of the posterior distribution given a case of the training set (see Eq. (1)). Then, the readjustClass function readjusts the probability thresholds of each class to select the class with the minimum-expected cost (see Eq. (2)). Finally, the readjustClass and the actual class are used to calculate the model’s accuracy (calculateAccuracy) and cost (calculateCost) of the model classifying the cases belonging to the test set. Finally, the $k$ percentages of well-classified cases and the $k$ misclassification costs are averaged to output the estimated values of the model learned from all cases to classify new cases.

3.2.1. Cost-sensitive selective naive Bayes – Accuracy

Algorithm 1 shows the pseudocode of the cost-sensitive selective naive Bayes – Accuracy model. This algorithm chooses $k$-fold cross-validation [46] as the procedure for estimating the accuracy and cost of models classifying new cases according to the value of the predictive features. This method is stratified, that is, it divides all cases into $k$ disjoint subsets of approximately equal proportion of class values and equal size. Each subset is used to test a model that is learned from the other $k-1$ subsets. The trainingSet and testSet functions provide the required subsets of cases in each iteration. This algorithm initializes the model to the class variable, that is, there is no predictive variables in the model yet. After that, the algorithm saves the accuracy of the resulting model (thresholdAccuracy) for subsequent comparisons. The accuracy threshold is computed by means of estimateClassProb and max functions, which, respectively, compute the initial class probabilities given the training set and return the highest probability value, that is, the probability of the most frequent class. In each iteration, the algorithm checks if a specific variable belongs to the model. The isModelVariable function returns true or false according to the current model. The algorithm considers adding each unused variable to the model on a trial basis and measures the performance of the resulting model on the training data. First, the predictClass function computes the predicted class, that is, the most likely class value of the posterior distribution given a case of the training set (see Eq. (1)). Then, the readjustClass function readjusts the probability thresholds of each class to select the class with the minimum-expected cost (see Eq. (2)). Finally, the readjustClass and the actual class are used to calculate the model’s accuracy (calculateAccuracy) and cost (calculateCost) of the model classifying the cases belonging to the test set. Finally, the $k$ percentages of well-classified cases and the $k$ misclassification costs are averaged to output the estimated values of the model learned from all cases to classify new cases.
the model (addToModel). The algorithm terminates when the addition of any variable results in a higher cost. During the test phase, the algorithm computes the accuracy (calculateAccuracy) and cost (calculateCost) of the model classifying the cases belonging to the test set. Finally, the k percentages of well-classified cases and the k misclassification costs are averaged to output the estimated values of the model learned from all cases to classify new cases.

3.3. Other classification methods for comparisons

3.3.1. Decision tree

The C4.5 algorithm aims at inducing a decision tree that represents the knowledge of the problem with a tree structure by a recursive division of the predictors’ space. This algorithm is an improvement of the ID3 algorithm [41].

3.3.2. K-nearest neighbour

The basic idea of the K-nearest neighbors method is that a new case will be classified as the most frequent class among its K nearest neighbors. Euclidean distance is used to estimate the nearest neighbors of a given case [20].

3.3.3. Logistic regression

The probability of an event is assumed to be a logistic function of certain variables that are considered potentially influential. The parameters of the model are estimated using the method of maximum likelihood and describe the size of the contribution of each variable to the model [24].

3.4. Statistical tests

Statistical tests determine whether there is enough evidence to reject a conjecture about the data. The conjecture is called the null hypothesis. Not rejecting the conjecture may be a good result if we want to continue to act as if we believe the null hypothesis is true. Or it may be a disappointing result, possibly indicating that we may not yet have enough information to reject the null hypothesis.

Tests that do not make assumptions about the population distribution are referred to as non-parametric tests. All commonly used non-parametric tests rank the outcome variable from low to high and then analyze the ranks.

In this paper, we use two non-parametric tests: Kruskal–Wallis test [32] and Mann–Whitney test [36]. The Kruskal–Wallis test analyzes whether three or more samples could have come from the same distribution. The null hypothesis is that the populations from which the samples originate have the same distribution. When the Kruskal–Wallis test leads to significant results, then at least one of the samples is different from the others. The test does not identify where the differences occur or how many differences actually occur. In contrast, the Mann–Whitney test analyzes whether two samples could have come from the same distribution. It is helpful for analyzing the specific sample pairs for significant differences. The significance level of these tests was 0.05 in all cases.

4. Results

4.1. Dataset construction

We have selected the Neurosciences category for our case study. We have used Thomson Reuters’ Web of Science (WoS) and Journal Citation Reports (JCR) platform to download publication and citation data. In the following, we illustrate the different phases of dataset construction and explain each dataset variable, that is, the predictive features.

First, we selected all journals belonging to the JCR Neurosciences category from 2000 to 2011. There were 269 journals in this category during the analyzed period. Then we obtained the publication list and citation data for these journals from the WoS. We downloaded all documents (1,044,811 papers) published by the 269 journals until 2011. Using the above information, we calculated some scientific impact indices associated with the selected journals for each journal from 2000 to 2011. We also downloaded other specific journal indices values from JCR. Finally, we stored all information in a database designed for this purpose.

The bibliometric indices used in this case study were documents, citations, the h-index, the g-index, the hg-index, the a-index, the m-index, the q^2-index, the h_c-index, the h_i-index, the h_i-index, impact factor, immediacy index, cited half-life, eigenfactor and article influence. In the following, we describe each bibliometric index.

X_1: Documents is an index associated with the number of papers published by each journal. It represents the productivity of each specific journal.

X_2: Citations is an index associated with the number of citations received by each journal. It represents the visibility of each specific journal.

X_3: One of the most successful indices was proposed by Jorge Hirsch and is called the h-index [23]. It quantifies the scientific output of a single researcher as a single-number criterion. It is a simple new measure incorporating both the quantity and visibility of publications. The h-index is based on a list of publications ranked in descending order by number of citations. The value of h is equal to the number of papers (N) in the list that have N or more citations.

X_4: Since the h-index tends to underestimate the achievement of journals that have a “selective publication strategy", that is, journals that do not publish a lot of documents but have a major international impact, the g-index, proposed in [13], is defined as the highest rank such that the cumulative sum of the number of citations received is greater than or equal to the square of this rank. Unlike the h-index, the g-index takes into account the exact number of citations received by highly cited papers, favoring journals with a selective publication strategy.

X_5: The hg-index, which is based on the h-index and the g-index, is presented in [2]. It intends to provide a more balanced view of the scientific production of journals. The hg-index of a journal is computed as the geometric mean of its h-index and g-index, that is,

\[ hg-index = \sqrt{h \cdot g}, \]

where h corresponds to the value of the h-index, and g corresponds to the value of the g-index.

X_6: The a-index was proposed in [27]. This index is calculated for papers that are in the h-core only, that is, the first h papers. It is defined as the average number of citations received by the articles included in the h-core. This index measures the citation intensity in the h-core. The a-index can be very sensitive to just a very few papers receiving extremely high citation counts.

X_7: The m-index is proposed in [5] as a variation on the a-index. As the distribution of citation counts is usually skewed, the median and not the arithmetic mean should be used as the measure of central tendency. This index, which was designed to illustrate the impact of the papers in the h-core, is the median number of citations received by papers in the h-core.

X_8: The q^2-index is developed in [6] to provide a more global view of scientific production. This index is based on the geometric mean of the h-index, describing the number of the papers (quantitative dimension), and the m-index, depicting the impact
of the papers (qualitative dimension), that is,
\[ q^{\text{2-index}} = \sqrt{h \cdot m}, \]
where \( h \) corresponds to the value of the \( h \)-index, and \( m \) corresponds to the value of the \( m \)-index.

\( X_{10} \): The \( h \)-index, which is an extension of the original \( h \)-index, was proposed in [42]. This index takes into account the number of citations needed to increase the \( h \)-index by one unit. It measures the distance to the next value of the \( h \)-index. Mathematically, this is
\[ h_{10} - \text{index} = (h + 1) - \frac{\text{Cits}(h + 1)}{2h + 1}, \]
where \( h \) is the value of the \( h \)-index, and \( \text{Cits}(h + 1) \) is the number of citations received by article \( h + 1 \).

\( X_{10} \): The \( h \)-index, proposed in [4], is complementary to the \( h \)-index and indicates the effective individual average productivity. Mathematically, it is calculated as
\[ h_{10} - \text{index} = \frac{h}{N_a}, \]
where \( h \) is the value of the \( h \)-index, and \( N_a \) is the mean number of authors in the \( h \) papers.

\( X_{11} \): The original \( h \)-index cannot distinguish between inactive journals and young journals and senior journals that are still publishing nowadays. For this reason, there is a need to define a new index that takes into account the “age” of papers. A novel score \( \text{Sc}(i) \) is defined for a paper \( i \) based on citation counting:
\[ \text{Sc}(i) = \gamma \cdot \left( Y(\text{now}) - Y(i) + 1 \right)^{-\delta} \cdot \text{Cits}(i), \]
where \( Y(\text{now}) \) is the current year, \( Y(i) \) is the publication year of paper \( i \); \( \text{Cits}(i) \) is the number of citations received by paper \( i \); \( \gamma \) and \( \delta \) are arbitrary parameters.

Using the above score, the value of old papers gradually declines, even if they still receive citations. Therefore, a new \( h_{10} \)-index is defined in [45]. Its definition states that a journal has index \( h_{10} \), if \( h_{10} \) of its published papers get a score of \( \text{Sc}(i) \geq h_{10} \) each, and the other papers get a score of \( \text{Sc}(i) < h_{10} \).

\( X_{12} \): The impact factor (IF) for a given journal in the year \( y \) is the average number of times the articles that it published in the past 2 years were cited in year \( y \). The impact factor is calculated by dividing the number of citations during year \( y \) by the total number of articles published by the journal in the previous 2 years. Mathematically, this is
\[ \text{IF}(y) = \frac{\text{Cites in year (y) to items published in years (y-1) and (y-2})}{\text{Number of items published in years (y-1) and (y-2})}. \]

\( X_{13} \): The immediacy-index is the average number of times an article is cited in the year that it is published. This index indicates how quickly articles in a journal are cited. It is calculated by dividing the number of citations to articles published in a given year by the number of articles published in that year. Mathematically, this is
\[ \text{immediacy-index (y)} = \frac{\text{Cites in year (y) to items published in year (y)}}{\text{Number of items published in year (y)}}. \]

\( X_{14} \): The cited half-life for a journal is the median age of its documents cited in the current JCR year. Half of the citations to the journal are to documents published within the cited half-life. The cited half-life calculation finds the number of publication years from the current JCR year that account for 50% of citations received by the journal.

\( X_{15} \): The eigenfactor calculation is calculated from the number of times articles from the journal published in the past 5 years which have been cited in the JCR year, but it also considers which journals have contributed these citations so that highly cited journals will influence the network more than lesser cited journals. References from one article in a journal to another article from the same journal are removed, so that the eigenfactor is not influenced by journal self-citation.

\( X_{16} \): The article influence determines the average influence of a journal’s articles over the first 5 years after publication. It is calculated by dividing a journal’s eigenfactor by the number of articles in the journal, normalized as a fraction of all articles in all publications. This measure is roughly analogous to the 5-year impact factor in that it is a ratio of a journal’s citation influence to the size of the journal’s article output over a period of 5 years.

4.2. Data distribution

After collecting the publication list and citation data of all journals, we observed that the number of cases selected to build the predictive models varied depending on the year. We used journal data from 2000 to 2010 (2305 cases) to construct the models assigned to the first-year. On the other hand, the models for the second-year used journal data from 2000 to 2009 (2037 cases). Finally, the predictive models for the third- and fourth-year used journal data from 2000 to 2008 (1785 cases) and from 2000 to 2007 (1449 cases), respectively. Clearly, the longer the prediction horizon was the fewer the cases were used to induce the models.

Fig. 2 shows the distribution of the journals selected according to the annual increase of their \( h \)-index value within the first 4 years. Taking the first year as an example, we observed that the lowest and the highest increment of the \( h \)-index was \( \Delta h = 0 \) (128 journals) and \( \Delta h = 24 \) (1 journal), respectively. We also noted that 457 journals had an increase of \( \Delta h = 3 \), which was the mode value for the first year. Regarding the second year, the minimum value was \( \Delta h = 0 \) (42 journals), the maximum value was \( \Delta h = 45 \) (1 journal) and the mode value was \( \Delta h = 6 \) (235 journals). Finally, we also noted that the \( h \)-index value increased from \( \Delta h = 0 \) to \( \Delta h = 65 \) for third-year models and from \( \Delta h = 0 \) to \( \Delta h = 81 \) for fourth-year models. Their mode values were \( \Delta h = 8 \) (163 journals) and \( \Delta h = 12 \) (110 journals).

We discretized the class variable values into four intervals with equal frequency. The increment of the \( h \)-index values was assigned to one of the four possible class values (low, medium-low, medium-high and high). In this way, first-year models were discretized as low (\( \Delta h = [0–1] \)), medium-low (\( \Delta h = [2] \)), medium-high (\( \Delta h = [3–4] \)) and high (\( \Delta h = [5] \)), whereas fourth-year models were discretized as low (\( \Delta h = [0–8] \)), medium-low (\( \Delta h = [9–12] \)), medium-high (\( \Delta h = [13–18] \)) and high (\( \Delta h = [19] \)). The correspondence between \( \Delta h \) values and class labels for all models is shown in Table 1.

4.3. Accuracy and average cost

We compared our approaches with the standard formulation of selective naive Bayes in order to determine if their accuracy and average cost values were reasonable. Table 2 shows the estimated accuracy and the average cost for each model. Numbers in boldface represent the highest accuracy value and lowest cost value for each model.

We tested our methods with different cost matrices (\( C(0, n) \), \( C(0, n^2) \), \( C(0, 2^n) \) and \( C(0, n^3) \)). The cost matrix \( C(0, n) \) represents costs where the correct classification has no costs and the incorrect classification has linear costs. Similarly, \( C(0, n^2) \), \( C(0, 2^n) \) and \( C(0, n^3) \) represents costs where the correct classification has no costs and the incorrect classification has quadratic and exponential costs.

Using the cost matrix \( C(0, n^3) \), for example, we noted that our models almost always outperform the selective naive Bayes models in higher accuracy and lower cost. Although these models achieved the highest accuracy (0.504) in the first year, our two
new algorithms, specifically the CS-SNB-Accuracy, achieved the highest accuracy in the second-year (0.518), third-year (0.542) and fourth-year (0.532). By average cost, we observed that our models, specifically the CS-SNB-Cost, always achieve a lower cost than the selective naive Bayes. Taking the first year as an example, we noted that cost associated with the selective naive Bayes is 1.227, whereas the cost related to the CS-SNB-Cost is 0.753.

Focusing on cost matrices, we observed that accuracy varied across models and prediction years, but we did not find a general pattern. The selective naive Bayes model achieved the highest accuracy value for first year (0.507) using the cost matrix $C(0, n^2)$. In contrast, the CS-SNB-Accuracy model obtained the highest accuracy value in the second year (0.519) and third year (0.542) with the cost matrices $C(0, 2^3)$ and $C(0, n^2)$, respectively. Finally, the CS-SNB-Cost achieved the highest accuracy values in the fourth year (0.533) using the cost matrix $C(0, n^2)$. By costs, we found that the lowest and the highest average cost were achieved by $C(0, n)$ and $C(0, 2^3)$. CS-SNB-Cost almost always achieved the lowest average cost with the $C(0, n)$ matrix.

Regarding each algorithm, we noted that selective naive Bayes achieved the highest accuracy values for first-year models no matter which cost matrix was used. In contrast, CS-SNB-Accuracy and CS-SNB-Cost predicted almost all the values more accurately than selective naive Bayes for the other prediction years. Given the cost matrix $C(0, 2^3)$, for example, we noted that selective naive Bayes achieved the highest accuracy (0.507) for first-year models, CS-SNB-Accuracy achieved the highest accuracy for second- (0.519) and third-year (0.538) models, and CS-SNB-Cost achieved the highest accuracy (0.532) for fourth-year models. Analyzing average cost, we found that the selective naive Bayes value was never the lowest. The lowest average cost values were always achieved by cost-sensitive models. Specially, we noted that CS-SNB-Cost usually obtained the lowest value. Given the cost matrix $C(0, n^2)$, for example, we noted that CS-SNB-Accuracy achieved the lowest average cost (0.721) for first-year models, whereas CS-SNB-Cost achieved the lowest values for second- (0.775), third- (0.708) and fourth-year (0.706) models. To summarize, we found that our cost-sensitive approaches, particularly CS-SNB-Cost, almost always achieved a lower average cost than selective naive Bayes. Also, our approaches, specially CS-SNB-Accuracy, often obtained higher accuracy values than selective naive Bayes.

Table 3 shows the accuracy and average cost of a set of classifiers (naive Bayes, selective naive Bayes, cost-sensitive selective naive Bayes – Accuracy, cost-sensitive selective naive Bayes – Cost, C4.5 decision tree, K-nearest neighbour and logistic regression) which are learned using the cost matrix $C(0, n^2)$ for all prediction years.

Analyzing the accuracy values, we distinguished three different groups (low, medium and high values). The first group is composed of the naive Bayes classifier which achieved the lowest values. In contrast, the second group is composed of three classifiers (selective naive Bayes, cost-sensitive selective naive Bayes – Accuracy, and cost-sensitive selective naive Bayes – Cost) that achieve medium values, whereas the third group is composed of non-Bayesian classifiers (C4.5 decision tree, K-nearest neighbour and logistic regression), having the highest values. We noted the above behavior no matter which prediction year was used. The results of the Kruskal–Wallis test showed that there were significant differences among the seven classifiers on the basis of the accuracy. So, we run Mann–Whitney tests in order to find out which classifiers rank better according to this criterion. We compared the benchmark classifier, which had the highest average value, with the other classifiers. Classifiers marked in Table 3 with the symbol † had statistically significant differences with respect to the benchmark classifier (highlighted in boldface). Taking the second-year model as an example, results show that there were significant differences between K-nearest neighbour (benchmark classifier) and naive Bayes, selective naive Bayes, CS-SNB-Accuracy, CS-SNB-Cost and logistic regression. In contrast, results do not show statistically significant differences between K-nearest neighbour and C4.5 decision tree.

Regarding the cost values, we also differentiated three groups. In this case, naive Bayes and selective naive Bayes achieved higher
Table 2
Accuracy and average cost of models which are learned using different selective naive Bayes approaches and cost matrices.

<table>
<thead>
<tr>
<th>Methods</th>
<th>First year</th>
<th>Second year</th>
<th>Third year</th>
<th>Fourth year</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accur</td>
<td>Cost</td>
<td>Accur</td>
<td>Cost</td>
</tr>
<tr>
<td>Cost matrix: C(0, n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective naive Bayes</td>
<td>0.502</td>
<td>0.608</td>
<td>0.506</td>
<td>0.644</td>
</tr>
<tr>
<td>CS-SNB-Accuracy</td>
<td>0.477</td>
<td>0.610</td>
<td>0.513</td>
<td>0.579</td>
</tr>
<tr>
<td>CS-SNB-Cost</td>
<td>0.458</td>
<td>0.526</td>
<td>0.519</td>
<td>0.577</td>
</tr>
<tr>
<td>Cost matrix: C(0, n^2)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective naive Bayes</td>
<td>0.503</td>
<td>0.828</td>
<td>0.501</td>
<td>1.005</td>
</tr>
<tr>
<td>CS-SNB-Accuracy</td>
<td>0.460</td>
<td>0.721</td>
<td>0.498</td>
<td>0.873</td>
</tr>
<tr>
<td>CS-SNB-Cost</td>
<td>0.451</td>
<td>0.735</td>
<td>0.514</td>
<td>0.775</td>
</tr>
<tr>
<td>Cost matrix: C(0, 2^n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Selective naive Bayes</td>
<td>0.507</td>
<td>1.211</td>
<td>0.509</td>
<td>1.299</td>
</tr>
<tr>
<td>CS-SNB-Accuracy</td>
<td>0.480</td>
<td>1.221</td>
<td>0.519</td>
<td>1.156</td>
</tr>
<tr>
<td>CS-SNB-Cost</td>
<td>0.460</td>
<td>1.187</td>
<td>0.507</td>
<td>1.168</td>
</tr>
</tbody>
</table>

Table 3
Accuracy and average cost of models which are learned using different classification methods. Results are achieved using the cost matrix C(0, n^n) for all prediction years.

<table>
<thead>
<tr>
<th>Methods</th>
<th>First year</th>
<th>Second year</th>
<th>Third year</th>
<th>Fourth year</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accur</td>
<td>Cost</td>
<td>Accur</td>
<td>Cost</td>
</tr>
<tr>
<td>Naive Bayes (NB)</td>
<td>0.892</td>
<td>0.842</td>
<td>0.857</td>
<td>0.879†</td>
</tr>
<tr>
<td>Selective naive Bayes (SNB)</td>
<td>0.532</td>
<td>0.725</td>
<td>0.528</td>
<td>0.713</td>
</tr>
<tr>
<td>C4.5 decision tree (C4.5)</td>
<td>0.504</td>
<td>0.753</td>
<td>0.516</td>
<td>0.772</td>
</tr>
<tr>
<td>K-nearest neighbour (K-NN)</td>
<td>0.525†</td>
<td>1.204†</td>
<td>0.598†</td>
<td>1.073†</td>
</tr>
<tr>
<td>Logistic regression (Logistic)</td>
<td>0.587</td>
<td>0.978†</td>
<td>0.587†</td>
<td>1.058†</td>
</tr>
</tbody>
</table>

Table 4
Accuracy and average cost of models which are learned using different cost-sensitive approaches. Values achieved using the cost matrix C(0, 2^n) for all prediction years.

<table>
<thead>
<tr>
<th>Methods</th>
<th>First year</th>
<th>Second year</th>
<th>Third year</th>
<th>Fourth year</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Accur</td>
<td>Cost</td>
<td>Accur</td>
<td>Cost</td>
</tr>
<tr>
<td>MetaCost</td>
<td>0.116†</td>
<td>1.770†</td>
<td>0.315†</td>
<td>1.597†</td>
</tr>
<tr>
<td>CostSensitiveClassifier</td>
<td>0.253†</td>
<td>1.866†</td>
<td>0.329†</td>
<td>2.382†</td>
</tr>
<tr>
<td>CSRoulette</td>
<td>0.432†</td>
<td>2.878†</td>
<td>0.517†</td>
<td>2.923†</td>
</tr>
<tr>
<td>CS-SNB-Accuracy</td>
<td>0.480</td>
<td>1.224†</td>
<td>0.519</td>
<td>1.156</td>
</tr>
<tr>
<td>CS-SNB-Cost</td>
<td>0.460†</td>
<td>1.187†</td>
<td>0.507†</td>
<td>1.168</td>
</tr>
</tbody>
</table>

costs, whereas C4.5 decision tree, K-nearest neighbour and logistic regression achieved medium costs. Finally, our proposed classifiers, CS-SNB-Accuracy and CS-SNB-Cost, achieved the lowest costs. We also performed a Kruskal–Wallis test in order to compare classifiers on the basis of the average cost. Taking the second-year model as an example, results show that there were significant differences between CS-SNB-Cost and naive Bayes, selective naive Bayes, C4.5 decision tree, K-nearest neighbour and logistic regression. In contrast, results do not show statistically significant differences between our cost-sensitive algorithms.

Analyzing different cost-sensitive approaches, we compare our algorithms with MetaCost, CostSensitiveClassifier and CSRoulette. These classifiers convert existing cost-insensitive classifiers (e.g. naive Bayes) into cost-sensitive ones. Table 4 shows the accuracy and average cost of the above classifiers which are learned using the cost matrix C(0, 2^n) for all prediction years.

Focusing on accuracy and cost values, we observed in Table 4 that our models outperform other cost-sensitive classifiers no matter which prediction year was used. Taking the first-year as an example, we noted that the CS-SNB-Accuracy achieved the highest accuracy (0.480) whereas the CS-SNB-Cost achieved the lowest cost (1.187). The results of the Kruskal–Wallis test showed that there were significant differences among the classifiers on the basis of accuracy and cost. So, we run Mann–Whitney tests in order to find out which classifiers rank better according to these criteria. We compared the benchmark classifier, which had the best average value, with the other classifiers. Classifiers marked in Table 4 with the symbol † had statistically significant differences with respect to the benchmark classifier (highlighted in boldface). Results show that there were significant differences between CS-SNB-Accuracy (benchmark classifier) and MetaCost, CostSensitiveClassifier, CSRoulette and CS-SNB-Cost in terms of accuracy.
Results also show that there were significant differences between CS-SNB-Cost (benchmark classifier) and MetaCost, CostSensitiveClassifier, CSRoulette and CS-SNBAccuracy in terms of costs. To summarize, we found that our cost-sensitive approaches, particularly CS-SNB-Cost, achieved a lower average cost than other cost-sensitive classifiers. Also, our approaches, specially CS-SNB-Accuracy, obtained higher accuracy values than other cost-sensitive classifiers.

Let us now analyze the models in more detail. Table 5 shows the specific variables, accuracy and cost for the CS-SNB-Cost model by each fold of the cross-validation process. These values were achieved using the cost matrix C(0, n²) for all prediction years. Analyzing Table 5, we found that the models always include the impact factor (variable 12). The models also usually include other variables like the h-index (variable 11), the cited half-life (variable 14) and the article influence (variable 16). We also noted that fewer models include the g-index (variable 4), the a-index (variable 6) and the m-index (variable 7). We noted that first-year models always had two variables, whereas second-, third-, and fourth-year models almost always had three variables. Finally, we found that the feature variables of the model that was most often induced included impact factor, cited half-life and article influence. Not all the models included these variables in all situations. This depends on the cost matrix and prediction year. So, other models were formed by different variables, although the impact factor, the cited half-life and the article influence were also present.

4.4. Example

We predict the increase of the h-index value of a Neurosciences journal in the first year using the cost matrix C(0, n). Table 6 shows the parameters that define the model. All features are described by means of the mean (µ) and the standard deviation (σ).

Given a journal (x) with the following values: impact factor = 2.582, cited half-life = 5.6, and article influence = 0.852, the Δh values can be predicted using the formulation of cost-sensitive selective naive Bayes (Algorithms 1 and 2) and the parameters listed in Table 6.

After propagating the above evidence, the results predicted by CS-SNBAccuracy were p(Δh = low|x) = 0.076, p(Δh = medium−low|x) = 0.391, p(Δh = medium−high|x) = 0.473 and p(Δh = high|x) = 0.060. Similarly, the results predicted by CS-SNB-Cost were p(Δh = low|x) = 0.070, p(Δh = medium−low|x) = 0.291, p(Δh = medium−high|x) = 0.504 and p(Δh = high|x) = 0.135. According to both approaches the h-index of the above journal is likely to increase by three or four units (medium-high) in the next year.

Table 7 shows other prediction years using the above conditions. We found that both models predicted the same class for all years. Note

---

Table 5

<table>
<thead>
<tr>
<th>k</th>
<th>Variables</th>
<th>Accuracy</th>
<th>Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>First year</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>12,11</td>
<td>0.456</td>
<td>0.721</td>
</tr>
<tr>
<td>2</td>
<td>12,11</td>
<td>0.478</td>
<td>0.756</td>
</tr>
<tr>
<td>3</td>
<td>12,11</td>
<td>0.465</td>
<td>0.713</td>
</tr>
<tr>
<td>4</td>
<td>12,14</td>
<td>0.391</td>
<td>0.834</td>
</tr>
<tr>
<td>5</td>
<td>12,14</td>
<td>0.521</td>
<td>0.647</td>
</tr>
<tr>
<td>6</td>
<td>12,11</td>
<td>0.456</td>
<td>0.713</td>
</tr>
<tr>
<td>7</td>
<td>12,11</td>
<td>0.419</td>
<td>0.730</td>
</tr>
<tr>
<td>8</td>
<td>12,11</td>
<td>0.447</td>
<td>0.682</td>
</tr>
<tr>
<td>9</td>
<td>12,14</td>
<td>0.426</td>
<td>0.765</td>
</tr>
<tr>
<td>10</td>
<td>12,11</td>
<td>0.426</td>
<td>0.782</td>
</tr>
<tr>
<td>Mean values</td>
<td></td>
<td>0.451</td>
<td>0.735</td>
</tr>
<tr>
<td>k</td>
<td>Variables</td>
<td>Accuracy</td>
<td>Cost</td>
</tr>
<tr>
<td>---</td>
<td>-----------</td>
<td>----------</td>
<td>------</td>
</tr>
<tr>
<td></td>
<td>Second year</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>12,11</td>
<td>0.456</td>
<td>0.721</td>
</tr>
<tr>
<td>2</td>
<td>12,11</td>
<td>0.478</td>
<td>0.756</td>
</tr>
<tr>
<td>3</td>
<td>12,11</td>
<td>0.465</td>
<td>0.713</td>
</tr>
<tr>
<td>4</td>
<td>12,14</td>
<td>0.391</td>
<td>0.834</td>
</tr>
<tr>
<td>5</td>
<td>12,14</td>
<td>0.521</td>
<td>0.647</td>
</tr>
<tr>
<td>6</td>
<td>12,11</td>
<td>0.456</td>
<td>0.713</td>
</tr>
<tr>
<td>7</td>
<td>12,11</td>
<td>0.419</td>
<td>0.730</td>
</tr>
<tr>
<td>8</td>
<td>12,11</td>
<td>0.447</td>
<td>0.682</td>
</tr>
<tr>
<td>9</td>
<td>12,14</td>
<td>0.426</td>
<td>0.765</td>
</tr>
<tr>
<td>10</td>
<td>12,11</td>
<td>0.426</td>
<td>0.782</td>
</tr>
<tr>
<td>Mean values</td>
<td></td>
<td>0.514</td>
<td>0.775</td>
</tr>
</tbody>
</table>

Table 6

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Impact factor</th>
<th>Cited half-life</th>
<th>Article influence</th>
</tr>
</thead>
<tbody>
<tr>
<td>Δh = low</td>
<td>µ = 0.977 σ = 0.906</td>
<td>µ = 5.573 σ = 3.328</td>
<td>µ = 0.318 σ = 0.354</td>
</tr>
<tr>
<td>Δh = medium−low</td>
<td>µ = 1.781 σ = 1.021</td>
<td>µ = 6.335 σ = 2.394</td>
<td>µ = 0.608 σ = 0.414</td>
</tr>
<tr>
<td>Δh = medium−high</td>
<td>µ = 2.728 σ = 1.724</td>
<td>µ = 5.985 σ = 2.050</td>
<td>µ = 0.950 σ = 0.830</td>
</tr>
<tr>
<td>Δh = high</td>
<td>µ = 5.774 σ = 4.802</td>
<td>µ = 5.643 σ = 2.060</td>
<td>µ = 2.660 σ = 3.202</td>
</tr>
</tbody>
</table>
that accuracy is different for first- and second-year models, but the same for third- and fourth-year models is equal. This is because the induced first- and second-year models were different. Results show that the increase of $h$-index for the above journal (x) will be medium-high ($\Delta h = [3–4]$) in the first year, and medium-low in the second ($\Delta h = [5–6]$); third ($\Delta h = [7–9]$) and fourth ($\Delta h = [9–12]$) years.

5. Conclusion

This paper presents new algorithms for predicting the increase of the $h$-index for scientific journals based on the cost-sensitive approach and the feature subset selection. We developed different cost-sensitive methods, where the learning algorithm includes the misclassification costs. These approaches take into account misclassification costs different from 0 (hit) and 1 (miss). These algorithms are concerned with classification accuracy and classification costs. Specially, we develop two forward cost-sensitive selective naive Bayes approaches. The search process of the first approach (CS-SNB-Accuracy) includes variables that improve classification accuracy, whereas the search process of the second approach (CS-SNB-Cost) includes variables that reduce the distances between the actual and the predicted class.

The main objective of the proposed algorithms is to predict the annual increase of the $h$-index for scientific journals. Models capable of predicting the $h$-index that a scientific journal is likely to have in coming years can be a useful tool for the scientific community. Results show that our approaches, specially the CS-SNB-Accuracy, achieved higher accuracy values than the analyzed cost-sensitive classifiers and Bayesian classifiers. Furthermore, we also noted that CS-SNB-Cost achieved a lower average cost than all analyzed classifiers. These cost-sensitive selective naive Bayes approaches outperform the original selective naive Bayes in terms of accuracy and average cost, so the cost-sensitive learning approach could be used in different probabilistic classification approaches.

In the future, we aim to build new cost-sensitive Bayesian classifiers like the selective tree augmented naive Bayes. These models could include other journal-based features e.g. 5-year impact factor, percentage of documents published by a single author, percentage of documents published in international collaboration and so on. Finally, the $h$-index value could vary depending on the source consulted (Google Scholar, Scopus, ISI WoK, etc.), which is a point to be taken into account.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (MINECO) projects TIN2010-20900-C04-04 and Cajal Blue Brain.

Table 7

<table>
<thead>
<tr>
<th>Class labels</th>
<th>First year</th>
<th>Second year</th>
<th>Third year</th>
<th>Fourth year</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS-SNB-Accuracy</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low</td>
<td>0.076</td>
<td>0.212</td>
<td>0.126</td>
<td>0.101</td>
</tr>
<tr>
<td>Medium-low</td>
<td>0.391</td>
<td>0.387</td>
<td>0.454</td>
<td>0.477</td>
</tr>
<tr>
<td>Medium-high</td>
<td>0.473</td>
<td>0.317</td>
<td>0.338</td>
<td>0.324</td>
</tr>
<tr>
<td>High</td>
<td>0.060</td>
<td>0.084</td>
<td>0.082</td>
<td>0.098</td>
</tr>
<tr>
<td>CS-SNB-Cost</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Low</td>
<td>0.070</td>
<td>0.159</td>
<td>0.126</td>
<td>0.101</td>
</tr>
<tr>
<td>Medium-low</td>
<td>0.291</td>
<td>0.410</td>
<td>0.454</td>
<td>0.477</td>
</tr>
<tr>
<td>Medium-high</td>
<td>0.504</td>
<td>0.330</td>
<td>0.338</td>
<td>0.324</td>
</tr>
<tr>
<td>High</td>
<td>0.135</td>
<td>0.101</td>
<td>0.082</td>
<td>0.098</td>
</tr>
</tbody>
</table>

References
