On the application of multi-class classification in physical therapy recommendation

Jing Zhang [1], Douglas Gross [2], and Osmar R. Zaiane [1]

[1]Department of Computing Science, [2]Department of Physical Therapy,
University of Alberta, Edmonton, Alberta, Canada
{jzhang14,dgross,zaiane}@ualberta.ca

Abstract. Recommending optimal rehabilitation intervention for injured workers that would lead to successful return-to-work (RTW) is a challenge for clinicians. Currently, the clinicians are unable to identify with complete confidence which intervention is best for a patient and the referral is often made in trial and error fashion. Only 58% recommendations are successful in our dataset. We aim to develop an interpretable decision support system using machine learning to assist the clinicians. We use various re-sampling techniques to tackle the multi-class imbalance and class overlap problem in real world application data. The final model has shown promising potential in classification compared to human baseline and has been integrated into a web-based decision-support tool that requires additional validation in a clinical sample.

Keywords: multi-class imbalance; re-sampling; clinical decision-support; rule-based learning

1 Introduction

Decision support systems (DSS) in clinical prognosis have received increased attention from researchers. In this paper, we develop a system to help clinicians categorize injured workers and recommend appropriate rehabilitation programs based on the unique characteristics of individual worker.

Our system is a web application consisting of a user interface and a knowledge base. Unlike many DSS using knowledge bases developed manually by domain experts, we use rule-based machine learning algorithms to learn a set of rules from data. The rules can be further modified and tuned by the experts. By doing so, the experts can inject their own knowledge into the discovered rule set.

The major challenge of generating the knowledge base is the presence of multi-class imbalance and class overlap in our real clinical dataset. Directly using off-the-shelf classifiers is not a solution since these classifiers would be biased with class imbalance. Typically, since most classifiers assume a balanced training set, data distribution is altered before training by over-sampling a minority class and under-sampling a majority class. However, it is not realistic to simply balancing the dataset with complex class overlap. We compare and analyze various data re-sampling and cleaning methods to tackle these problems. We find that the combination of SMOTE [4] with Tomek Link [1] and RIPPER [6] can produce meaningful recommendation rules as evaluated by our domain expert. Moreover, the combination of class decomposition and data processing method can help the classification on the minority class examples.

The rest of the paper is organized as follows: We provide the background of the project in Section 2, and describe the system design methods, model and implementation in Section 3. In Section 4, we discuss the evaluation of our model and conclude in Section 5 with a summary and future study.

2 Background

Work-related musculoskeletal conditions are some of the most burdensome health conditions in terms of personal, societal and economic costs[8, 7, 3]. Low back pain is a leading cause of work disability and was recently identified as the sixth most disabling health condition worldwide in terms of overall disease burden[5].

In general, each injured worker receives a return-to-work (RTW) assessment and a following rehabilitation treatment. This is a classification process which involves assigning patients to appropriate rehabilitation programs that lead to successful return-to-work (RTW) based on their clinical and work-related characteristics (obtained from the assessment). There are five types of rehabilitation programs in total labeled as prog0, prog3, prog4, prog5 and prog6.

Each rehabilitation program has two possible outcomes:

- The program leads to successful return-to-work at a pre-determined time.
- An unsuccessful result at that pre-determined time followed by subsequent rehabilitation programs.

Although it is possible that multiple rehabilitation programs can lead to return-to-work for a patient, we cannot determine them since we cannot possibly let a patient go through multiple programs at once to observe the outcomes. Therefore, an important assumption is that for each patient there exists only one appropriate program. If patients are correctly categorized into the true appropriate program, they return to work. Otherwise, there will be no successful RTW. Under the assumption above, we could determine a patient's return-to-work status in advance based on the classification result. The main idea here is to build a classification model that categorizes an injured worker into the appropriate rehabilitation program leading to RTW.

3 System Design and Implementation

3.1 System Requirements

The system we are developing has the following requirements:

- The classification model should be interpretable. The users should be able to see the evidence supporting the recommendations made by the system. Rule-based algorithms are more desirable.

- The system should provide multiple predictions with support evidence (e.g., supporting rules or guidelines) so the users can choose the most appropriate one under different considerations.
- The system should include a limited number of variables.

3.2 Data Analysis

The dataset is from an outcome evaluation database managed by the Workers' Compensation Board Alberta. This includes data on workers in the province of Alberta who filed compensation claims for musculoskeletal injuries and who were referred to rehabilitation facilities for Return-to-Work assessment. WCB-Alberta's administrative database was augmented by clinical data from rehabilitation providers who are contracted to file reports at time of claimants' admission and discharge from rehabilitation programs.

The dataset of mainly the year of 2010 contains 14484 cases of injured workers, of which 8611 were unique cases and included in further analysis. To train a classification model that predicts successful interventions, we extract only the successful cases. The successful outcome is when the injured worker receives no compensation at 30 days after the assessment admission. In total, 4859 cases are extracted. The new dataset is highly-skewed as shown in Table 1. The rest of the data consisting of unsuccessful cases is used to train a negative model. The dataset includes 200 features. We consulted the experts from the Department of Physical Therapy to check each variable and eliminate those that are absolutely irrelevant from the perspective of clinical practice. 59 features are selected for further investigation.

Class	prog0	prog3	prog4	prog5	prog6
num of records	1828	84	2286	96	582

Table 1. Class Distribution of the Final Dataset

3.3 Method

The class distribution in the dataset is severely imbalanced as shown in Table 1. With class imbalance, the classifier becomes biased towards the majority classes over-shadowing the minority classes. Possible solutions for this problem are cost-sensitive learning, attaching costs to misclassifications, and re-sampling techniques, adjusting the data distribution. Since we cannot obtain the costs of misclassifications, we focus on the re-sampling techniques.

We use a variety of known re-sampling techniques including 1) over-sampling: SMOTE [4] and 2) data cleaning (under-sampling) methods: Tomek Link [1], Edited Nearest Neighbor (ENN) [11] and Neighbor Cleaning Rule (NCR) [9]. These are used to mitigate the imbalance and to weed out noise in the data.

However, these methods are only validated in the literature on binary datasets (2 classes). Their effectiveness is unknown for multi-class imbalance problems. We apply the re-sampling techniques directly on the dataset as well as combining class decomposition with the re-sampling techniques The details of each method are described below:

SMOTE: SMOTE stands for Synthetic Minority Over-sampling Technique. It manipulates the feature values of data examples that are nearest neighbors to each other in order to form new synthetic minority class examples. It generalizes the data space and thus can avoid overfitting to some extent.

Tomek Links: If two data examples from different classes are the 1 nearest neighbors to each other, they form a Tomek Link. Either both of them are borderline points, or one of them is noise invading the data space of the other class. Generally, we can remove both points in a Tomek Link.

Neighborhood Cleaning Rule(NRC): Unlike ENN, NCR finds each data example whose class label differs from the class of at least two of its three nearest neighbors. If this example belongs to the majority class, remove it. Otherwise, remove its nearest neighbors which belong to the majority class.

SMOTE + Tomek Link: The main reason for this combination is that the synthetic data from a minority class might invade the majority class too deeply and with the cleaning of Tomek Links, we could avoid potential overfitting.

3.4 Algorithms

Naive Bayes, C4.5 and RIPPER classifiers were all investigated; however, the best results were obtained with RIPPER. We briefly present these methods. **Naive Bayes**: The Naive Bayes algorithm applies the Bayes theorem to compute the likelihood that an unseen object belongs to a certain class C_i $(i = 1, 2, ..., k)$ given the attribute values in the object. Then the algorithm assigns the object to the class with the maximum likelihood. The algorithm relies on a naive assumption that given the class label, all attributes are mutually independent. Although the assumption seems over-simplified, it has shown its competitiveness in many practical situations.

Equation 1 shows that given a test case t with values of f_1 to f_n, we can use Bayes rule to calculate the probability of class C_i $(i = 1, 2, ..., k)$.

$$p(C_i|f_1, f_2, f_3, ..., f_n) = \frac{p(C_i)p(f_1, f_2, f_3, ..., f_n|C_i)}{p(f_1, f_2, f_3, ..., f_n)} \tag{1}$$

Based on the assumption of conditional independence, Equation 1 can be represented by Equation 2 as followed.

$$p(C_i|f_1, f_2, f_3, ..., f_n) = \frac{p(C_i)\prod_{j=1}^{n} p(f_j|C_i)}{p(f_1, f_2, f_3, ..., f_n)} \tag{2}$$

Then according to the maximum a posteriori (MAP) decision rule, one can classify a test instance t using Equation 3 as followed.

$$classification(t) = arg\, max\, p(C_i) \prod_{j=1}^{n} p(f_j|C_i) \tag{3}$$

C4.5: C4.5 [10] is a well-known decision tree induction algorithm. Each node in the tree represents a selected feature and the tree branches out based on the values in the node. The leaf node represents a class. A new instance is classified by testing against the feature at each node and following the branch of the tree based on the observed value in the instance. This process repeats until the instance reaches the leaf node and is assigned to the class of the leaf.

Associative Classification: Associative classification is based on the association rule mining. It only discovers associations between a set of features and a class label. We use the Associative Rule Classification-By Category (ARC-BC) [12] algorithm. ARC-BC mines rules in each class separately and is shown to be less affected by class imbalance. The rules that pass through a local threshold are grouped together to build a final classifier. The classifier may assign multiple class labels to a new instance, the final decision is made by measuring and comparing the quality of each prediction.

Repeated Incremental Pruning to Produce Error Reduction (RIPPER): RIPPER [6] is an inductive rule-based learner that generates Disjunctive Normal Form (DNF) rules to identify classes while minimising the error defined by the number of misclassified training example by the rule. Each classification rule has a conjunction of attribute values as antecedent and a class as consequent.

$$r : (Rule\ Antecedent) \to y \tag{4}$$

In a multi-class situation, the rules generated from the RIPPER algorithm are ranked in ascending order based on the number of examples in the class. An unknown instance is tested against the rules in that order. The first rule that covers the test instance fires and the testing phase ends.

The default RIPPER algorithm makes only one prediction. It fires the first rule that covers the test instance. To make multiple predictions, we make the following modifications and refer to the modified algorithm as ARIPPER (Alternate RIPPER) in the rest of the paper: for each test instance, we gather all the rules covering it and group the rules together that predict the same program and rank these predictions based on their quality. Such quality can be computed by measuring the quality of the underlying supporting rules. We consider four types of measurements:

- **Highest Average Rule Confidence (HAvgRCF)**: calculate the average rule confidence of all rules supporting each recommendation. The one with the highest average rule confidence is the final prediction.
- **Single Rule with Highest Confidence (SRHCF)**: the rule with the highest confidence makes the final prediction.
- **Highest Average Weighted Chi-Square (HAvgCS)**: HAvgCS is a measurement adopted from CMAR, i.e., Classification based on Multiple Association Rules [2]. It calculates the weighted rule Chi-Square value of all rules supporting each recommendation.
- **Single Rule with Highest Weighted Chi-Square (SRHCS)**: SRHCS looks at the Chi-Square of each single rule and uses the one with the highest Chi-Square value as the quality of a recommendation.

3.5 Evaluation Measurements

To evaluate the performance of the physicians (human baseline), we use the successful rate as our measurement. It is the only measurement we can use for the human baseline. The successful rate of the physicians is defined as the number of successful recommendations (patient returns to work by receiving this recommendation) made by a physician over the number of all cases in the dataset. This is similar to the overall classification accuracy measurement. With class imbalance, this is not a good measurement. However, since we do not know the true class label of unsuccessful cases, it is neither possible to obtain the confusion matrix to use other measurements like Precision and F-measure, nor to know the measurements of each class. Therefore, we can only use overall classification accuracy in comparison with the human baseline, however, we do include other measurements for completeness.

- **Sensitivity**: sensitivity describes the proportion of actual positive examples that are correctly identified.
- **Specificity**: specificity measures the proportion of actual negative examples that are correctly identified.
- **Geometric Mean (G-mean)**: There is a trade-off between sensitivity and specificity. G-mean is a more harmonic measurement defined as

$$\sqrt{sensitivity * specifity}. \tag{5}$$

3.6 Experiment Design

We conducted a variety of experiments and only those giving meaningful results are presented here. We explain one experiment in detail. The rest of them are described briefly since most experiments use the same strategy with the only difference being the underlying method for sampling.

SMOTE + Tomek Link + DataSet1 (direct approach): To mitigate the class imbalance, we use a progressive sampling approach to change the class distribution:

1. Choose one minority class and fix the rest.
2. Increase the size of the selected class by a certain percentage P.
3. Train an ARIPPER classifier on the sampled dataset. If the true positive rate of the selected class increases significantly, undo the sampling and repeat step 2 with a larger percentage P and step 3. However, if the size of the sampled class is greater than that of the largest class in the dataset or the increase is less than 2%, stop the sampling process.
4. Choose P as the final sampling percentage.

The final sampling percentage obtained for each minority class is 900%, 900% and 300% respectively. Figure 1 shows the class distribution before and after the sampling. We can visualize the sampled dataset using Principal Component Analysis (PCA) with the first two components as shown in Figure 2-a. We can

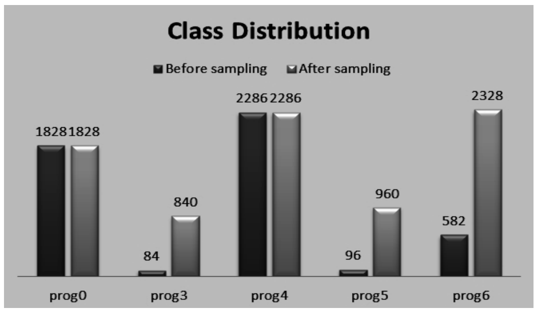

Fig. 1. Class Distribution Before and After Sampling-Final Dataset

see that on the left side class 3 has a minor overlap with class 6 while class 5 has invaded class 6. On the right side, class 0 and 4 are mixed together.

To make it easier for any algorithm to build a good classification model, a data cleaning stage is desirable to clean up the borders between each class. We apply the Tomek Link Cleaning method to weed out noise. Data points from different classes that form a Tomek Link are considered as borderline or noisy points, and generally can be removed. The details of the cleaning process are stated as follows:

1. Extract each pair of classes.
2. Identify the Tomek links between these two classes.
3. Remove noise or borderline points. If such cleaning improves the overall performance of the model, merge the cleaned up classes back to the whole dataset. Otherwise, undo the cleaning.
4. Repeat step 1 to 3 until all possible pairs of classes is processed.

Figure 2-b visualizes the dataset after Tomek Link cleaning. We can see that data points from Class 0 are completely mixed with Class 4. It is possible that the current selected features cannot separate these two classes effectively. Since feature selection is data dependent, we further sampled on prog0 as a possible solution for the class overlap. It is possible that we can select new and effective features to separate Class 0 and 4. Sampling on Class 0 may cause further overlapping between Class 0 and 4. But those points will be removed later as noise while the useful examples will be reinforced. We choose to sample 60% on class 0 and apply the same procedures above. 19 features are selected and the visualization using PCA is shown in Figure 2-c.

Clearly, we can see that part of Class 0 is now separable from Class 4. We then apply the Tomek Link cleaning on the new dataset. Figure 2-d visualizes the dataset after the cleaning. Those points from class 0 mixing with class 4 are removed while those separable remain in the data space. We then build a model using both ARIPPER and associative classification learner on this dataset. The evaluation is detailed in the next section.

Class decomposition + SMOTE + NCR (OVA: One-vs-All): In this approach we first decompose the dataset into 5 binary datasets. Each binary

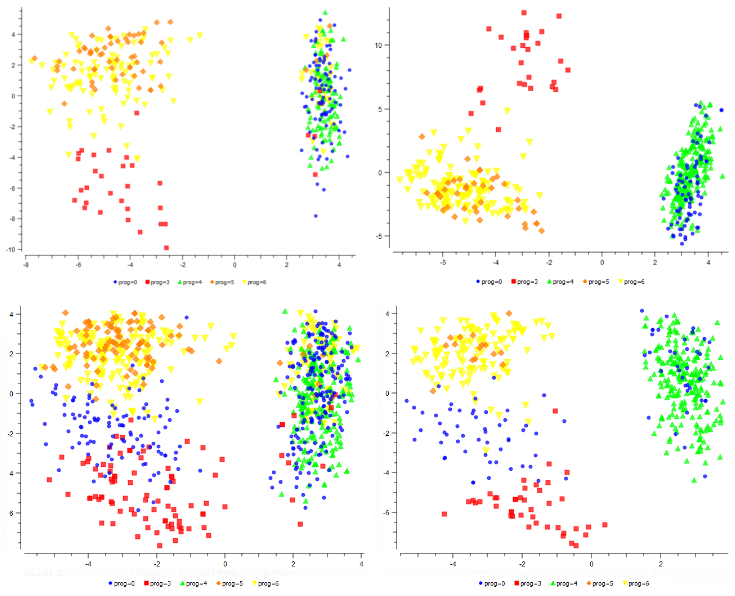

Fig. 2. Dataset Visualization a)left top: after SMOTE, b)right top: after Tomek Link cleaning, c)left bottom: after resampling, d)right bottom: after cleaning in second round

dataset contains the data from one positive class and all other classes are considered as one negative class. We use SMOTE to sample on the minority class in each binary dataset. The size of the minority class should be close to but smaller than that of the majority class. Then we use Neighborhood Cleaning Rule (NCR) as a data cleaning method to clean the data space. After the cleaning, five binary classifiers are created using different learning algorithms. To make a prediction for an unknown instance, each classifier generates a probability of that instance belonging to the positive class. We use the imbalance rate to combine the probability prediction of all 5 classifiers: we take the product of prediction probability and the imbalance rate of its corresponding class as a final weight. The test instance belongs to the class with the highest weight.

4 Evaluation and Discussion

4.1 Experiment Evaluation

As mentioned in the System Requirement Section, our system should make multiple recommendations for the users to choose from. Since this is a Decision **Support** System, our goal is to help the physicians but not to replace them. The physicians can view the rules supporting the recommendations and make their own decisions from the recommendation pool.

However, from a computer science perspective, this is not sufficient. For a multi-class classification problem, the model has to finalize its prediction. Therefore, for each dataset obtained by using different data preprocessing strategies, we train a model from it using different algorithms and then evaluate their performance on the test set.

SMOTE + Tomek Link + DataSet1 (direct approach): We first train a model using the ARIPPER algorithm. The rules obtained from this model were evaluated by experts from the Department of Physical Therapy and considered as meaningful rule sets. Our prototype system is implemented based on these rules. Table 2 shows the prediction evaluation on the test set using these four measurements: The potential means that if any of the predictions matches with the true label, we count it as a correct prediction. Note that in rules generated from the RIPPER algorithm, there is a default rule with empty rule body and neither confidence nor Chi Square is applicable. So for each test instance, we assign to it both the selected prediction and the default prediction. If either of them matches with the true label, we count it as a correct prediction.

Criterion	HAvgRCF	SRHCF	HAvgCS	SRHCS	Potential
Accuracy	0.73	0.72	0.48	0.48	0.78

Table 2. Evaluation On the Test Set (ARIPPER)

We then train three other classifiers using the Naive Bayes algorithm, C4.5 algorithm and ARC-BC respectively. Table 3 to 5 show the evaluation of each algorithm on the test set. The overall accuracy is 0.385, 0.478, and 0.470 respectively.

	Sensitivity	Specificity	G-Mean
Prog0	0.071	0.961	0.261
Prog3	0.000	1.000	0.000
Prog4	0.969	0.069	0.260
Prog5	0.000	1.000	0.000
Prog6	0.000	1.000	0.000
Overall	0.482	0.870	0.647

Table 3. Evaluation On the Test Set (Naive Bayes)

Class decomposition + SMOTE + NCR (OVA): for the decomposition approach, we are using two base learners for each binary classifier Naive Bayes and RIPPER (original RIPPER). Table 6 and 7 show the confusion matrix of the evaluation on the test set using base learner Naive Bayes and RIPPER.

	Sensitivity	Specificity	G- Mean
Prog0	0.05	0.98	0.22
Prog3	0	1	0
Prog4	0.98	0.04	0.199
Prog5	0	1	0
Prog6	0	1	0
Overall	0.478	0.869	0.645

Table 4. Evaluation On the Test Set (C4.5)

	Sensitivity	Specificity	G- Mean
Prog0	0.132	0.958	0.355
Prog3	0.588	0.458	0.519
Prog4	0.856	0.274	0.484
Prog5	0.105	0.969	0.319
Prog6	0.069	0.952	0.256
Overall	0.471	0.747	0.593

Table 5. Evaluation On the Test Set (ARC-BC)

4.2 Discussion

In this section, we discuss the evaluation in the former section. Note that the evaluation here has its own limitations as we mentioned in the earlier section.

For the direct approach with ARIPPER algorithm, we can see that by choosing the prediction with rule confidence measurement, the prediction accuracy reaches around 72%. Unfortunately, since each instance has two predictions, we cannot analyze other measurement using the confusion matrix. The accuracy on test set using Naive Bayes, C4.5 and ARC-BC algorithms is lower than the human baseline. However, ARC-BC does slightly better than the other two on predicting minority class examples. For the class decomposition approach, the overall accuracy is also lower than the human baseline. However, one thing worth noticing is that by using Naive Bayes as the base learner, the model makes good predictions of the minority classes. But it is difficult to ensure good classification

	Sensitivity	Specificity	G- Mean
Prog0	0.08	0.931	0.276
Prog3	1	0.732	0.856
Prog4	0.284	0.849	0.491
Prog5	0.556	0.666	0.608
Prog6	0.06	0.902	0.249
Overall	0.201	0.8	0.401

Table 6. Evaluation On the Test Set (Naive Bayes)

	Sensitivity	Specificity	G- Mean
Prog0	0.846	0.308	0.511
Prog3	0.444	0.987	0.662
Prog4	0.314	0.857	0.519
Prog5	0	0.994	0
Prog6	0	1	0
Overall	0.473	0.868	0.641

Table 7. Evaluation On the Test Set (RIPPER)

results on both the majority and minority classes at the same time. This is a common tradeoff with the presence of class imbalance.

The nature of the rehabilitation program is also somewhat responsible for the misclassification between the majority classes. As we are informed by the experts, prog0 and prog4 are similar to each other. A large portion of people receiving prog4 actually does not need prog4. But in order to make sure that people do return to work, they are assigned with prog4 in the end. Additionally, prog6 is a hybrid program of prog4 and prog5, which makes it even more complicated in classification. The data visualization in earlier section also confirms this issue as we can see the overlap between these classes. Currently our prototype system implements the ARIPPER model trained from the first experiment since the rules are considered to be very meaningful from clinical perspective. This rule set shows a high potential on the test evaluation. As a decision *support* system, this should be sufficient since the clinician is the one who makes the final decision. To further evaluate the system, we need to do additional validations in real clinical settings.

5 Summary

In this work we build a decision support system with a knowledge base generated by machine learning algorithms. To tackle the multi-class imbalance and class overlap due to the nature of this clinical dataset, we apply several data re-sampling techniques to make it easier for the learning stage. Our results show that the direct approach SMOTE + Tomek Link + ARIPPER generates a meaningful rule-based model whose prediction ability is comparable to the clinicians. Moreover, combining class decomposition with data re-sampling is a better way to effectively classify minority class examples than applying the data re-sampling directly.

Since our system provides human readable rules and presents these rules as evidence of any recommendation, a feedback loop is conceivable allowing an expert user to change these rules by directly injecting domain knowledge in the model initially automatically derived from the data.

As for future study, we plan to find a solution to determine the right prediction between the default prediction and the other one as discussed in experiment

1. Building a binary classifier between these two predictions would be a good start. Another extension to our work is to integrate the negative model into the evaluation of the positive model such as canceling conflicting predictions under certain circumstances. To evaluate the system from a clinical perspective, additional validation in random clinical trials is required.

References

1. Two modifications of cnn. *Systems, Man and Cybernetics, IEEE Transactions on*, SMC-6(11):769 –772, nov. 1976.
2. *CMAR: accurate and efficient classification based on multiple class-association rules*, 2001.
3. Martin BI, Deyo RA, Mirza SK, Turner JA, Comstock BA, Hollingworth W, and Sullivan SD. Expenditures and health status among adults with back and neck problems. *JAMA*, 299:656–64, 2008.
4. Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *J. Artif. Int. Res.*, 16(1):321–357, June 2002.
5. CJ. Murray CJ, T. Vos, R. Lozano, M. Naghavi, AD. Flaxman, C. Michaud, and et al. Disability-adjusted life years (dalys) for 291 diseases and injuries in 21 regions, 1990-2010: a systematic analysis for the global burden of disease study 2010. *Lancet*, 380(9859):2197–223, 2012.
6. William W. Cohen. Fast effective rule induction. In *In Proceedings of the Twelfth International Conference on Machine Learning*, pages 115–123, 1995.
7. N.M. Hadler. *Occupational musculoskeletal disorders.* 3rd ed. Philadelphia: Lippincott Williams & Wilkins, 2005.
8. R. Lane and S. Desjardins. Canada. population and public health branch. Strategic policy directorate. Policy research division. Economic burden of illness in Canada [Ottawa]: Health Canada, 2002.
9. Jorma Laurikkala. Improving identification of difficult small classes by balancing class distribution. In *Proceedings of the 8th Conference on AI in Medicine in Europe: Artificial Intelligence Medicine*, AIME '01, pages 63–66, London, UK, UK, 2001. Springer-Verlag.
10. J. Ross Quinlan. *C4.5: programs for machine learning.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1993.
11. Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *Systems, Man and Cybernetics, IEEE Transactions on*, 2(3):408–421, July 1972.
12. Osmar R. Zaïane and Maria-Luiza Antonie. Classifying text documents by associating terms with text categories. In *Proceedings of the 13th Australasian database conference - Volume 5*, ADC '02, pages 215–222, Darlinghurst, Australia, Australia, 2002. Australian Computer Society, Inc.