A Classification Learning Algorithm Robust to Irrelevant Features

H. Altay Güvenir

Bilkent University,
Department of Computer Engineering and Information Science
06533 Ankara, Turkey
guvenir@cs.bilkent.edu.tr
http://www.cs.bilkent.edu.tr/~guvenir/

Abstract. Presence of irrelevant features is a fact of life in many real-world applications of classification learning. Although nearest-neighbor classification algorithms have emerged as a promising approach to machine learning tasks with their high predictive accuracy, they are adversely affected by the presence of such irrelevant features. In this paper, we describe a recently proposed classification algorithm called VFI5, which achieves comparable accuracy to nearest-neighbor classifiers while it is robust with respect to irrelevant features. The paper compares both the nearest-neighbor classifier and the VFI5 algorithms in the presence of irrelevant features on both artificially generated and real-world data sets selected from the UCI repository.

1 Introduction

Inductive classification or concept learning algorithms derive some form of classification knowledge from a set of training examples. In most real-world applications of classification learning, it is common to include all available information about the domain in the training data, and expect the learning algorithm somehow select the relevant portions [2]. This is a valid assumption since exactly which features are relevant to the target concept being learned may be unknown.

In recent years, instance-based nearest-neighbor (NN) classification algorithms have emerged as a promising approach to machine learning, with researchers reporting excellent results on many real-world induction tasks [1]. The nearest neighbor algorithm normally represents instances as feature-value pairs. In order to predict the class of a novel instance, first its distance to each of the training instances is computed. Then the class value of the test instance is predicted to be the class of the training example with shortest distance, that is the nearest neighbor. Learning in nearest-neighbor classifiers consists of simply storing the training instances in memory, leaving all the computation to the classification phase. For that reason, these kind of algorithms are called lazy learners [8]. The kNN algorithm is a generalization of the NN algorithm, where the classification is based on a majority voting of the nearest k neighbors.

One solution to the problem of irrelevant features is to separately learn weights for features so that the irrelevant ones are assigned low weight values and therefore their effect on the distance measure is reduced. Feature selection is the extreme case of feature weighting, where only zero and one are used as weight values. The nearest-neighbor classifier is then run with only these selected features that have one as their weight value. Although feature selection is a special case of feature weighting, Kohavi et al. reported that increasing number of possible weights beyond two (zero and one) has very little benefit and sometimes degrades performance [12]. Wettschereck et al. provide a good review and an empirical evaluation of feature weighting methods for a class of lazy learning algorithms [16]. Some researchers have developed algorithms just for the selection of relevant features [3, 13–15].

In this paper we present a classification learning algorithm that achieves high accuracy, comparable to nearest-neighbor classifier, and is not adversely affected by the presence of irrelevant features. The VFI5 (Voting Feature Intervals) algorithm described here is quite robust with respect to irrelevant features, yet achieves good performance on existing real-world datasets. The VFI5 algorithms eliminates the adverse effect of irrelevant features by its inherent voting mechanism.

The rest of the paper is organized as follows. Section 2 explains the VFI5 classification learning algorithm in detail. Section 3 presents an evaluation of the VFI5 algorithm on artificially generated data sets that contain a varying number of irrelevant features. Section 4 evaluates the VFI5 algorithm on some existing data sets with artificially added irrelevant features. Section 5 concludes the paper.

2 VFI5 Classification Learning Algorithm

The VFI5 classification algorithm is an improved version of the early VFI1 algorithm [5,7], which is a descendent of the CFP algorithm [11]. It has been applied to the problem of differential diagnosis of Erythemato-Squamous diseases [6] and arrhythmia analysis of ECG signals [9]; and very promising results were obtained. Here, the VFI5 algorithm is described in detail.

The VFI5 classification learning algorithm represents a concept in terms of feature value intervals, and makes a classification based on feature votes. It is a non-incremental learning algorithm; that is, all training examples are processed at once. Each training example is represented as a vector of feature values plus a label that represents the class of the example. From the training examples, the VFI5 algorithm constructs feature value intervals for each feature. The term *interval* is used for feature value intervals throughout the paper. An interval represents a set of values of a given feature where the same set of class values are observed. Two neighboring intervals represent a different set of classes. For each interval, a lower bound of the values and the number of examples of each class in that interval are maintained. Thus, an interval may represent several classes by storing the number of examples for each class.

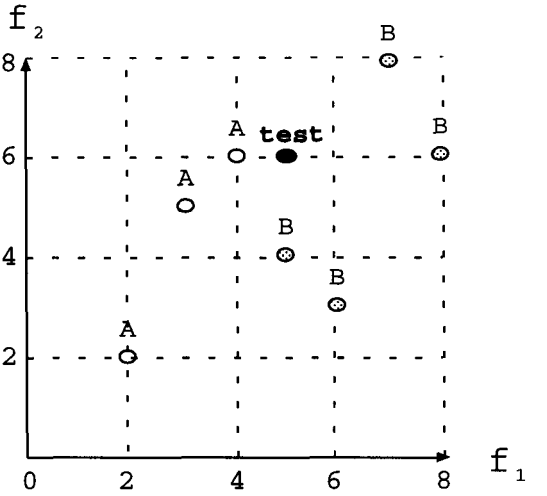

Fig. 1. A sample training dataset with two features and two classes.

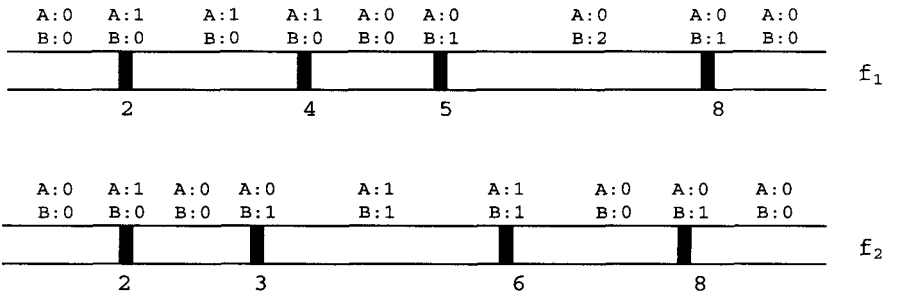

Fig. 2. Intervals constructed by VFI5 with their class counts for the sample dataset.

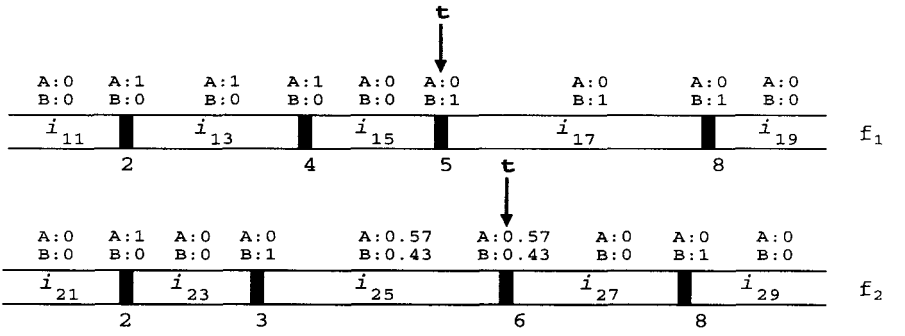

Fig. 3. Votes of intervals for the sample dataset.

In order to describe the VFI5 algorithm, consider the sample training dataset in Figure 1. In this dataset, we have two linear features f_1 and f_2, and there are 3 examples of class A and 4 examples of class B. There are 9 intervals for each feature. The intervals formed in the training phase of the VFI5 algorithm are shown in Figure 2.

The training process in the VFI5 algorithm is given in Figure 4. The lower bounds of intervals are learned by finding the *end points* for each feature and for each class. The procedure *find_end_points*(*TrainingSet*, *f*, *c*) finds the lowest and the highest values for feature *f* from the examples of class *c* in the *TrainingSet*. The lowest and highest values are called the *end points*, and for each feature there are $2C$ end points where C is the number of distinct classes in the domain. VFI5 constructs a *point interval* at each distinct end point. Further, for linear features a *range interval* is constructed between every consecutive end points. These range intervals do not cover the end point values. Maximum number of intervals constructed for linear features is $4C + 1$.

Each interval is represented by a vector of $< lower, vote_1, \ldots, vote_C >$ where *lower* is the lower bound of that interval, $vote_i$ is the vote given to class i by that interval. These votes are computed as

$$interval\_class\_vote[f, i, c] = \frac{interval\_class\_count[f,\ i,\ c]}{class\_count[c]}$$

where $interval\_class\_count[f, i, c]$ is the number of examples of class c which fall into interval i of feature f. The individual vote of feature f for class c, $interval\_class\_vote[f, i, c]$, is then normalized to have the sum of votes of feature f equal to 1. Hence, the vote of feature f is a real-valued vote in [0,1]. This normalization guarantees that, unless otherwise specified, each feature has the same weight in the voting. Class votes of the intervals for the data set given in Figure 1 are shown in Figure 3.

Note that since each feature is processed separately, no normalization of feature values is required.

The VFI5 classifier is shown in Figure 5. The process starts by initializing the votes of each class to zero. The classification operation includes a separate preclassification step on each feature. The preclassification of feature f involves a search for the interval on feature f into which e_f falls, where e_f is the value test example e for feature f. This search is performed by the *find_interval* function in Figure 5. If that value is unknown (missing), that feature does not participate in the classification process. Hence, the features containing missing values are simply ignored. Ignoring the feature about which nothing is known is a very natural and plausible approach.

If the value for feature f of example e is known, the interval i into which e_f falls is determined. An interval may contain training examples of several classes. The classes in an interval are represented by their normalized votes. The votes of an interval are already stored as part of its representation. These votes of the interval is used as the vote vector of the corresponding feature. After every feature completes their preclassification process, the individual vote vectors are

```
train(TrainingSet):
begin
   for each feature f
      if f is linear
        for each class c
            EndPoints[f] = EndPoints[f] ∪ find_end_points(TrainingSet, f, c);
        sort(EndPoints[f]);

        for each end point p in EndPoints[f]
            form a point interval from end point p
            form a range interval between p and the next endpoint ≠ p
      else /* f is nominal */
        form a point interval for each value of f

      for each interval i on feature f
          for each class c
              interval_class_count[f, i, c] = 0
      count_instances(f, TrainingSet);
      for each interval i on feature f
          for each class c
              interval_class_vote[f, i, c] = interval_class_count[f, i, c] / class_count[c]
              normalize interval_class_vote[f, i, c];
                        /* such that Σ_c interval_class_vote[f, i, c] = 1 */
end.
```

Fig. 4. Training in the VFI5 Algorithm.

```
classify(e):
/* e: example to be classified */
begin
      for each class c
          vote[c] = 0
      for each feature f
          for each class c
              feature_vote[f, c] = 0 /* vote of feature f for class c */
          if e_f value is known
              i = find_interval(f, e_f)
              feature_vote[f, c] = interval_class_vote[f, i, c]
              for each class c
                  vote[c] = vote[c] + feature_vote[f, c];
      return class c with highest vote[c];
end.
```

Fig. 5. Classification in the VFI5 Algorithm.

summed up to get a total vote vector $< vote_1, \ldots, vote_C >$. Finally, the class with the highest vote from the total vote vector is predicted to be the class of the test instance.

3 Empirical Evaluation on Artificial Data Sets

In order for an empirical comparison of kNN and VFI5 algorithms, we have artificially generated data sets with varying number of relevant and irrelevant features and measured the predictive accuracies of these algorithms.

We have generated data sets where the number of relevant features ranges from 1 to 6. We call these data sets Rn, where n represents the number of relevant features. These artificial data sets contain two classes. The instance space is divided into two regions of equal volume. 50 randomly generated instances are distributed to each of the regions uniformly. Therefore such a data set contains 100 instances. Once an artificial data set Rn with n relevant features is generated, we further added varying number of irrelevant features to the data set. The number of irrelevant features ranged from 0 to 20. For each such a data set, we computed the 5-fold cross-validation accuracies of both NN and VFI5 algorithms. We have repeated this process for 100 times and reported the results in Figure 6. We have run the kNN algorithm for k values of 1, 3 and 5.

It is clear from Figure 6 that VFI5 is much less affected by the existence of irrelevant features in the data set. On the other hand, the predictive accuracy of the kNN algorithm almost linearly drops as the number of irrelevant features increases. Also the slope of the accuracy plot decreases as the number of relevant features increases, as expected.

4 Empirical Evaluation on Existing Data Sets

In order to compare the kNN and VFI5 classifiers we also tested them on six existing data sets selected from the UCI repository [4]. Since most of the datasets in the UCI repository are carefully constructed by eliminating irrelevant features, we modified the data sets by artificially adding increasing number of irrelevant features. We used 1, 3 and 5 as the values of k in the kNN algorithm. The comparison of the classification accuracies kNN and VFI5 algorithms on six UCI-Repository data sets with increasing number of artificially added irrelevant features is depicted in Figure 7.

The experiments indicate that, although, both algorithms achieve about the same predictive accuracy without relevant features, the accuracy of the nearest-neighbors classifier drops quickly when irrelevant features are added. On the other hand, the accuracy of the VFI5 classifier remains at about the same level as the case without the irrelevant features.

This shows that the VFI5 algorithm is robust with respect to the existence of irrelevant features. The robustness of the VFI5 algorithm is due to the voting mechanism used in the classification. Since the votes of an interval, in turn a feature, are normalized, an irrelevant feature gives about the same vote to all

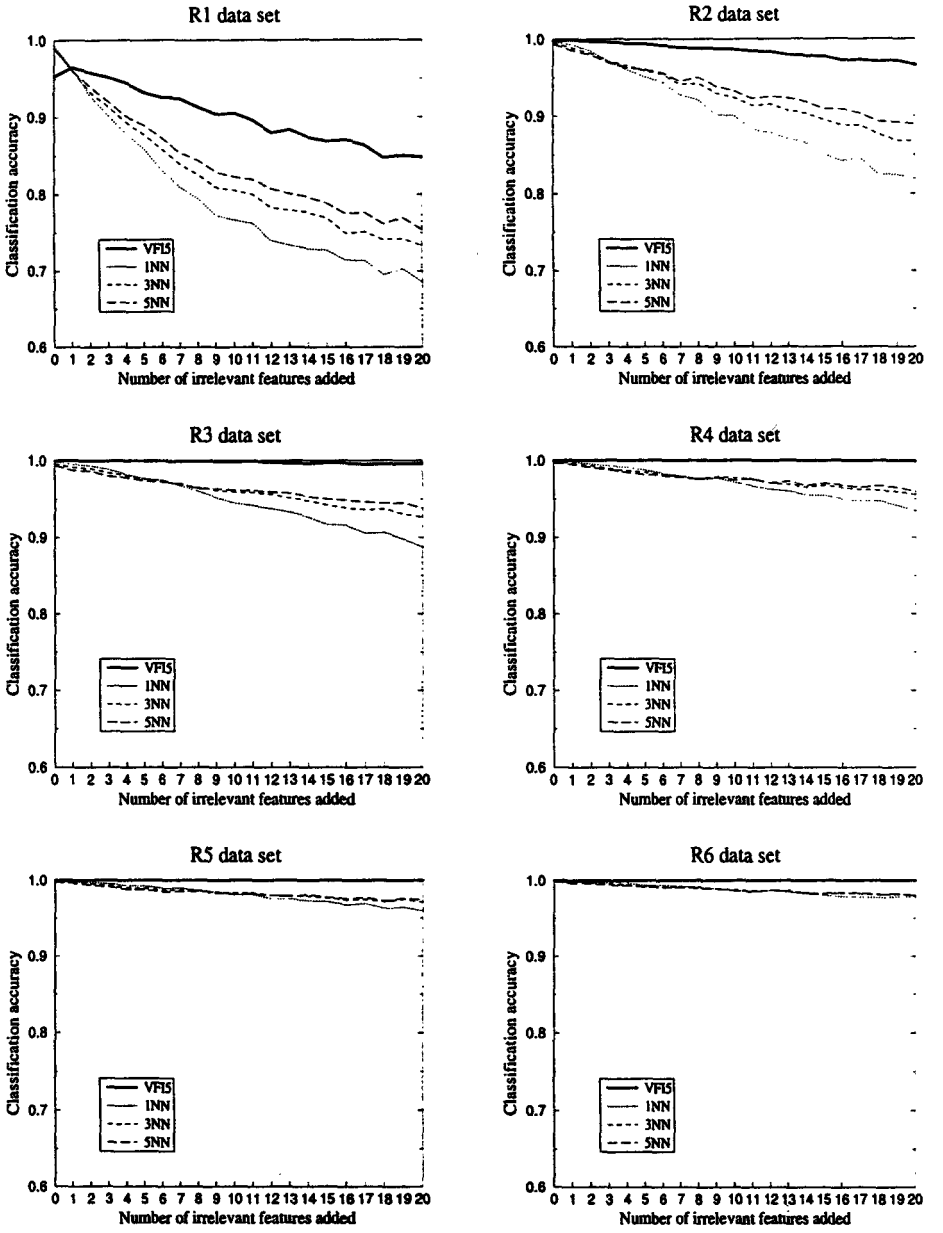

Fig. 6. The comparison of the average classification accuracies for kNN and VFI5 on some artificially generated data sets. Rn represents a data set with n relevant features and two classes. Accuracy value is the average of 100 5-fold cross-validation accuracies.

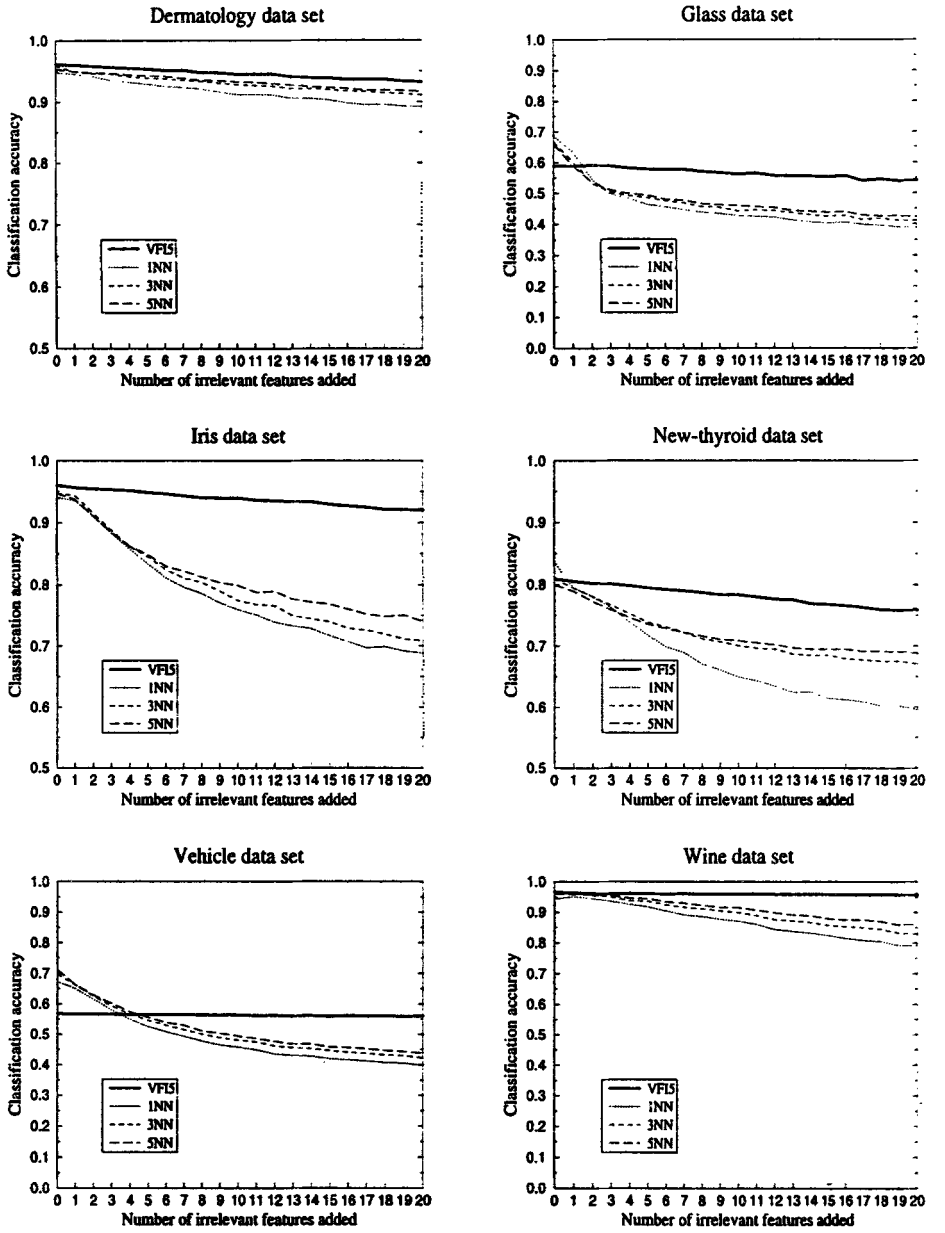

Fig. 7. The comparison of the average classification accuracies for kNN and VFI5 on some of UCI-Repository data sets with increasing number of artificially added irrelevant features. Accuracy given is the average of 100 5-fold cross-validation accuracies.

the classes in the domain. Therefore have no effect on the outcome of the voting. The main advantage of the VFI5 algorithm is that it achieves this robustness without requiring an external help for feature selection.

These experiments also indicate that, for higher values of k, the kNN algorithm becomes more robust to irrelevant features.

5 Conclusion

In this paper, a voting based classification algorithm called VFI5 is described. The VFI5 algorithm is compared with the nearest-neighbor algorithm which has been reported to achieve high accuracy values. These algorithms were tested on both artificially generated and existing data sets with increasing number of artificially added irrelevant features. Our experiments showed that, in most data sets, both algorithms achieve about the similar predictive accuracy without relevant features. However, when irrelevant features are added, the accuracy of VFI5 algorithm remains at about the same level or exhibit very small amount of decrease, while the accuracy of the nearest neighbor classifier drops quickly. This shows that the VFI5 algorithm is robust with respect to the existence of irrelevant features. The VFI5 algorithm achieves this by the voting mechanism used in the classification, where the votes of an irrelevant feature are about the same for all classes, and therefore have no effect on the outcome. The main advantage of the VFI5 algorithm is that it achieves this robustness without requiring an external help for feature selection.

References

1. Aha, D., Kibler, D., Albert, M.: Instance-based Learning Algorithms. Machine Learning. **6** (1991) 37–66
2. Almuallim, H., Dietterich, T.G.: Learning with many irrelevant features. In: Proceedings of the 9th National Conference on Artificial Intelligence: AAAI Press, Menlo Park (1991) 547–552
3. Cardie, C.: Automating Feature Set Selection for Case-Based Learning of Linguistic Knowledge. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing, University of Pennsylvania (1996) 113–126
4. Christopher, J.M., Murphy, P.M.: UCI repository of machine learning databases. At `http://www.ics.uci.edu/~mlearn/MLRepository.html` (1998)
5. Demiröz, G.: Non-Incremental Classification Learning Algorithms based on Voting Feature Intervals. MSc. Thesis. Bilkent University, Dept. of Computer Engineering and Information Science. Ankara, Turkey (1997)
6. Demiröz, G., Güvenir, H.A., İlter, N.: Differential Diagnosis of Erythemato-Squamous Diseases using Voting Feature Intervals. In: Ciftcibasi, T., Karaman, M., Atalay, V. (Eds.): New Trends in Artificial Intelligence and Neural Networks (TAINN'97), Kızılcahamam, Turkey, (May 22-23, 1997), 190–194
7. Demiröz, G., Güvenir, H.A.: Classification by Voting Feature Intervals. In: van Someren, M., Widmer, G. (Eds.): Machine Learning: ECML-97. Lecture Notes in Computer Science, Vol. 1224. Springer-Verlag, Berlin (1997) 85–92

8. Domingos, P.: Context-sensitive feature selection for lazy learners. Artificial Intelligence Review **11** (1997) 227–253
9. Güvenir, H.A., Acar, B., Demiröz, G., Çekin, A.: A Supervised Machine Learning Algorithm for Arrhythmia Analysis. In: Computers in Cardiology 1997, **24** Lund, Sweden (1997) 433–436
10. Güvenir, H.A., Akkuş, A.: Weighted K Nearest Neighbor Classification on Feature Projections. In: Kuru, S., Çağlayan, M.U., Akın, H.L. (Eds.): Proceedings of the Twelfth International Symposium on Computer and Information Sciences (ISCIS XII). Antalya, Turkey (1997) 44–51
11. Güvenir, H.A., Şirin, İ.: Classification by Feature Partitioning. Machine Learning **23** (1996) 47–67
12. Kohavi, R., Langley, P., Yun, Y.: The Utility of Feature Weighting in Nearest-Neighbor Algorithms. In: van Someren, M., Widmer, G. (Eds.): Machine Learning: ECML-97. Lecture Notes in Computer Science, Vol. 1224. Springer-Verlag, Berlin (1997) 85–92
13. Langley, P.: Selection of Relevant Features in Machine Learning. In: Proceedings of the AAAI Fall Symposium on Relevance. New Orleans, USA, AAAI Press, (1994)
14. Liu, H., Setiono, R.: A probabilistic approach to feature selection - A filter solution. In: Saitta, L. (Ed.): Proceedings of the Thirteenth International Conference on Machine Learning (ICML'96) Italy (1996) 319-327
15. Skalak, D.: Prototype and feature selection by sampling and random mutation hill climbing algorithms. In: Proceedings of the Eleventh International Machine Learning Conference (ICML-94). Morgan Kauffmann, New Brunswick (1994) 293–301
16. Wettschereck, D., Aha,D.W., Mohri, T.: Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. Artificial Intelligence Review **11** (1997) 273–314.