arXiv:1407.4668v1 [cs.LG] 17 Jul 2014

A feature construction framework based on outlier detection and discriminative pattern mining

Albrecht Zimmermann
albrecht.zimmermann@insa-lyon.fr

INSA Lyon, France

July 18, 2014

Abstract

No matter the expressive power and sophistication of supervised learning algorithms, their effectiveness is restricted by the features describing the data. This is not a new insight in ML and many methods for feature selection, transformation, and construction have been developed. But while this is on-going for general techniques for feature selection and transformation, i.e. dimensionality *reduction*, work on feature construction, i.e. *enriching* the data, is by now mainly the domain of image, particularly character, recognition, and NLP.

In this work, we propose a new general framework for feature construction. The need for feature construction in a data set is indicated by *class outliers* and discriminative pattern mining used to derive features on their k-neighborhoods. We instantiate the framework with LOF and C4.5-Rules, and evaluate the usefulness of the derived features on a diverse collection of UCI data sets. The derived features are more often useful than ones derived by DC-Fringe, and our approach is much less likely to overfit. But while a weak learner, Naive Bayes, benefits strongly from the feature construction, the effect is less pronounced for C4.5, and almost vanishes for an SVM leaner.

Keywords: feature construction, classification, outlier detection

1 Introduction

Supervised learning – concept, classifier, and regression learning – is a core Machine Learning topic and decades of research have resulted in lots of algorithms. No matter the expressiveness or sophistication of a technique, however, its effectiveness is restricted by the data, and its representation.

As a result, researchers have also worked from early on to create useful representations. The resulting techniques can be grouped into three subfields:

1. *Feature selection* [MBN02, GE03, EA13, BH13] deals with the question of *removing* irrelevant or redundant features from the data representation. Its goals are alleviating the "curse of dimensionality" – the phenomenon that high-dimensional descriptor data points all seem to be equally similar to each other – reducing running times of learning techniques, and preventing over-fitting.

2. *Feature transformation* or *dimensionality reduction* [Fod02] has the same goals but achieves them by mathematical transformations of the matrix representation of the data, replacing the original representation.

3. *Feature construction* or *constructive induction* [YRB91, SP05], finally, aims at combining the existing features into new ones, and enriching the data with them, with the goal of making harder problems easier to model and increasing accuracy.

This subdivision is not as clear-cut as we make it appear here: [GE03], for example, also discusses feature transformation techniques, referring to them as "feature construction". What is striking, however, is that after a flurry of early work on symbolic feature construction [MR89, ACS+91, YRB91, Paz98], recent works employ genetic algorithms for feature construction [Kra02, SP05], focusing on constructing features from raw data [SHGHP13], and for the purpose of NLP [GM05, DR12] and image recognition [YKR+08].

For a data miner, this is somewhat puzzling, given that much recent DM research has addressed the problem of mining useful features for classification

from complex data via discriminative pattern mining [BZDRN06, ZBR10, TCG+10]. Hence the motivation for this work. Our contribution is two-fold:

1. We propose a general framework for feature construction, employing outlier detection methods for identifying hard-to-model instances, and discriminative pattern mining for deriving useful features from their neighborhoods. In particular is this framework *independent* of the learning algorithm employed.

2. We instantiate our framework with LOF as outlier detection technique and C4.5-RULES as discriminative pattern miner, and evaluate and compare the usefulness of the derived features on a diverse collection of UCI data sets, using three learners of different strength. We show that our model-independent approach is far less likely to overfit and generates useful features.

The paper is structured as follows: in the following section, we illustrate the problem setting and briefly discuss the four aspects to feature construction [MR89]. In Section 3, we discuss related work. In Section 4, we introduce our framework, and show how it addresses the four feature construction aspects. In Section 5, we describe the concrete instantiation used in the experiments and report on the experimental results in Section 6, before concluding in Section 7.

2 Problem Illustration and Aspects of Feature Construction

As an illustration of the problem addressed in our work, consider Figure 1. Both classes are characterized by large clusters of typical instances, as well as several outlying points. There can be different reasons for this distribution: the training set could, e.g., *not* be a representative sample – a possible explanation for $o_\circ$ or o_{+_1}. An instance like o_{+_2}, however, either is mis-labeled or has incorrect attribute values (in other words, it is noise), or indicates that the description space is incomplete w.r.t. the underlying concepts.

A linear discriminative learner, inducing the decision surface d_l, or a concept learner characterizing the attribute-value space enclosed by d_d, will be unable to classify certain instances correctly. A more powerful learner, such as an SVM or an artificial neural network (ANN), could classify $o_{+,2}$ correctly but the representation would prevent it from separating o_{svm} out.

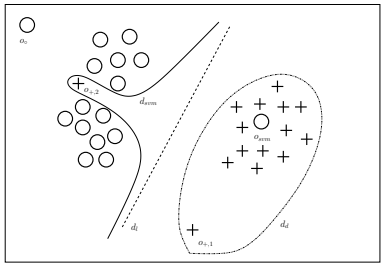

Figure 1: Class outliers in a two-class problem

Finding a feature that takes the value 1 for o_{svm} and 0 for its immediate neighborhood, as shown in Figure 2, would make it possible to learn a decision surface for that instance. Hence, in this setting there is need for additional dimensions.

2.1 Aspects of feature construction

Some data points are clearly not in need of augmentation: points belonging to the two clusters will be reliably modeled, and $o_\circ$ and $o_{+,1}$ can still be separated easily from the other class. There is also the question if a locally discriminating feature might not overfit the data too much and become useless outside of its local context (as in the case of o_+ in Figure 2). The authors of [MR89] turned this intuition systematic when they identified four aspects of feature construction:

1. detection of when construction is required
2. selection of constructors
3. generalization of selected constructors
4. evaluation of the new features

The first point is characterized by the authors writing: "if the original feature set is sufficient for the [...] induction algorithm to acquire the target concept, feature construction is unnecessary.". They identify three possible treatments: i) always perform construction, ii) analysis of the initial data set, iii) analysis of a model.

The second aspect has to do with the fact that the space of all attribute-value combinations can be too large to traverse, especially if arbitrary operators are allowed. They identify two approaches: i) initially limiting allowed operators, and ii) run time decisions that select one constructor over another based on algorithmic, data, concept, or domain knowledge biases.

The third aspect addresses over-fitting since new features might be highly specific to certain training in-

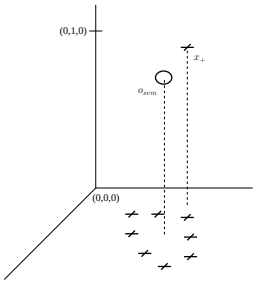

Figure 2: Enriching data with an additional feature

stances. This includes removal of conjuncts, variable introduction etc.

The fourth aspect, finally, comes into play if the number of features becomes too large. The authors identify at least three approaches: i) keeping all features, ii) call on the user, or iii) order the features and keep the best ones.

3 Related Work

Much early work on feature construction used decision trees as models, and based the newly constructed features on them. The setting for FRINGE, the technique developed in [Pag89], is binary classification, with instances described by boolean attributes. The goal was the avoidance of the replication of partial paths in the decision tree. Detection consists of checking for the existence of positively-labeled leaf nodes at depth of at least two in the tree (model analysis), constructors are selected by choosing the tests in the last two decision nodes before the leaf (concept-based), which are combined using logical AND. The features are added to the representation, data instances are re-encoded, and the algorithm re-iterated until either no new features are generated or a pre-set number of iterations have been performed. The authors extended their approach to be able to deal with additional class labels, called SYMMETRIC FRINGE. Matheus *et al.* [MR89] introduced the four aspects of feature construction discussed in the preceding section, and extended FRINGE by considering tests in other regions of the tree, not just near the leaves. They keep the same detection and constructor selection, but add a generalization step – in which constants can be replaced by variables – and only keep the top-27 features, according to information gain. The resulting technique is called CITRE [MR89] and allows the use of background knowledge in the selection, and evaluation steps. DC-FRINGE [YRB91], finally, considers not only leaves but also the siblings of their parents. The authors claim that CITRE does not profit from using other tests than the ones at the fringe, and show that DC-FRINGE outperforms the earlier two techniques on a number of artificial data sets.

Other works from this period chose different classification models. Aha [ACS+91] used an incremental instance-based learner, using classification failure for detection. Classification is performed based on similarity to already encountered instances, and the new feature is chosen to be the conjunction of any pair of base attributes that maximizes the dissimilarity between the misclassified instance, and the differently labeled instances closest to it. They point towards the need for background knowledge for this approach to outperform CITRE. Pazzani [Paz98] proposed constructing features from the Cartesian product of existing attributes. Features are constructed from pairs of attributes, features can be deleted, only one such operation is performed per iteration, and the evaluation takes the form of accuracy estimation of a learner. In all these works, feature generation iterates repeatedly, using the same classification model.

At the same time, genetic algorithms were first used for feature construction to replace the local search for good features [BK96], and most recent work comes from this direction [SP05, SHGHP13]. These approaches allow for arbitrary combinations of atribute tests but pay with increased running times (especially since they typically iterate over a model inducer), which they address by limiting the number of base attributes used in feature construction.

The last reference also points to another trend in feature construction: recent works have mostly moved away from attribute-value settings and aim to construct features for text categorization [GM05], NLP [DR12], and image recognition [YKR+08, RVL13].

Somewhat separate from this are works in the data mining community, which aim to construct features useful for representing structured data, e.g. trees or graphs, for processing by machine learning techniques [BZDRN06, ZBR10, TCG+10]. The typical approach in those works consists of mining substructures that discriminate between two classes, and performing classification by SVM or decision tree learners.

4 A Novel Feature Construction Framework

Based on the problem illustration given in Section 2, we propose a novel, general framework for feature construction. We aim at a *model independent* approach,

i.e. one that can be run as a pre-processing step independent of the learner used. This also means that our approach does *not* reiterate model learning. In the following, we lay out how we address the four aspects of feature construction.

4.1 Detection – Identifying class outliers

We first need to address whether to construct features at all, and from which instances. For this purpose, we propose to use outlier detection techniques to identify atypical points in the data. Identifying outliers in the full data will not be useful since a point that appears as an outlier from the perspective of a cluster of differently-labeled instances could be easy to model for a classifier learner. Hence, we identify outliers from the perspective of *individual* classes. Assuming a data set $\mathcal{D}$, an *outlier oracle* $OO(d, \mathcal{D}') \mapsto \{\text{true}, \text{false}\}$ that can decide whether a instance d is an outlier w.r.t. $\mathcal{D}'$, we can define the set of class outliers:

Definition 1 *Given a set of labeled data* $\mathcal{D} = \{(\vec{x}, y)\}$*, the set of outliers of class* $\mathcal{D}_c = \{d = (\vec{x}, c) \in \mathcal{D}\}$ *is defined as* $\mathcal{O}_{c,\mathcal{D}} = \{d \in \mathcal{D}_c \mid OO(d, \mathcal{D}_c) = true\}$*. The union over all the outliers of all classes* $\mathcal{O}_{C,\mathcal{D}} = \bigcup_c \mathcal{O}_{c,\mathcal{D}}$ *is called the set of* class outliers*.*

While we use the class label information in the detection step, we are *not* making use of a particular classification model to detect the need for feature construction. This has the advantage that our approach can be used as a pre-processing step for arbitrary classifier learners, but the potential disadvantage that we overestimate the need for feature construction.[1]

4.2 Constructor selection – Assembling outliers' k-neighborhoods

We intend to perform feature construction itself by mining discriminative patterns. The choice of pattern miner will impose some restrictions on the form new features can take, depending on the pattern language $\mathcal{L}_p$ employed.[2] In addition to this, we employ a data-biased selection mechanism, collecting the k-neighborhoods of class outliers:

[1]Note that this outlier detection is not limited to the space of attribute-valued data. Any instance space in which a distance measure between instances is defined allows for the identification of outliers.

[2]Again, this step is not limited to the space of attribute-valued data. Any instance space in which discrimative patterns can be mined can be worked on.

Definition 2 *Given an instance* d *and a distance measure* $\delta : \mathcal{D} \times \mathcal{D} \mapsto \mathbb{R}$*, its* k-neighborhood *is defined as:* $\mathcal{N}_k(d) = \{d' \in \mathcal{D} \setminus \{d\} \text{ s.t. } |\{d'' \mid \delta(d'', d) \leq \delta(d', d)\}| \leq k - 1\}$

and, if they belong to different classes, mining discriminative patterns from them. This condition – that different classes have to be present – can be considered part of the detection step: class outliers like $o_\circ$ in Figure 1 do not indicate a need for feature construction.

This data selection step reduces the space of possible constructors to those present in this k-neighborhood, making the feature construction step tractable. Additionally, they can be expected to be relatively small, aiding further in tractability.

4.3 Generalization – Merging k-neighborhoods

According to [MR89], generalization of features occurs *after* they have been mined. In contrast to this, we address the generalization aspect *before* feature construction by comparing the k-neighborhoods of all pairs of class outliers, and merging them if at least 50% of the instances of the smaller neighborhood are included in the larger one. This is transitive: if an outlier's neighborhood has at least half its instances contained in a second neighborhood, those instances will also be contained in any neighborhood the second one was merged with.

4.4 Evaluation – Removing inconsistent and insufficiently supported features

Since we intend our feature construction approach to be used in a pre-processing step, we do not use any model's classification accuracy to evaluate features during construction. Since we *do* intend to mine features that discriminate outliers from their differently-labeled neighbors, however, we only select *consistent* patterns, i.e. ones that cover instances of only a single class within a k-neighborhood:

Definition 3 *Given a matching function* match: $\mathcal{L}_p \times \mathcal{D} \mapsto \{0, 1\}$*, the* cover *of a pattern* $\pi \in \mathcal{L}_p$ *in a data set* $\mathcal{D}'$ *is defined as* $cov(\pi, \mathcal{D}') = \{d \in \mathcal{D}' \mid match(\pi, d) = 1\}$*.*

Definition 4 *Given a pattern* π*,* k*-neighborhood* $\mathcal{N}_k$ *it has been mined from, we call a pattern* consistent *iff* $\forall d_i = (\vec{x}_i, y_i), d_j = (\vec{x_j}, y_j) \in cov(\pi, \mathcal{N}_k) : y_i = y_j$*.*

Since in that case features could overfit, we also impose a minimum support constraint on a feature.

Definition 5 *Given a pattern π, its* support *on a data set $\mathcal{D}'$ is defined as $supp(\pi, \mathcal{D}') = |cov(\pi, \mathcal{D}')|$.*

Finally, if a feature appears several times, it is selected only once.

4.5 Algorithm

The meta-algorithm describing the workings of our approach is given as Algorithm 1. We will refer to our framework as CobFC – for class-outlier based feature construction – in the rest of the paper.

Algorithm 1 The CobFC algorithm

Given: data set $\mathcal{D}$, set of class labels C, support threshold θ
Return: set of consistent discriminative patterns $\mathbb{F}$

$\mathcal{O}_{C,\mathcal{D}} = \emptyset$
for $c \in C$ **do** {Collecting class outliers}
  $\mathcal{O}_{C,\mathcal{D}} = \mathcal{O}_{C,\mathcal{D}} \cup \mathcal{O}_{c,\mathcal{D}}$
$\mathbb{N} = \emptyset$
for $o \in \mathcal{O}_{C,\mathcal{D}}$ **do**
  if $\exists d_i = (\vec{x}_i, y_i), d_j = (\vec{x}_j, y_j) \in \mathcal{N}_k(o) : y_i \neq y_j$ **then** {Different classes in the neighborhood}
    $\mathbb{N} = \mathbb{N} \cup \{\mathcal{N}_k(o)\}$
while $\exists \mathcal{N}_k(o), \mathcal{N}_k(o') \in \mathbb{N} : (\frac{|\mathcal{N}_k(o') \cap \mathcal{N}_k(o)|}{\min\{|\mathcal{N}_k(o')|, |\mathcal{N}_k(o)|\}} \geq 0.5)$ **do** {Merging overlapping neighborhoods}
  $\mathbb{N} = ((\mathbb{N} \cup \{\mathcal{N}_k(o') \cup \mathcal{N}_k(o)\}) \setminus \mathcal{N}_k(o')) \setminus \mathcal{N}_k(o)$
for $\mathcal{N}_k(o) \in \mathbb{N}$ **do** {Mining discrimative patterns}
  for all $f \in \mathrm{PM}(\mathcal{N}_k(o))$ **do**
    if f is consistent $\wedge\, \mathrm{supp}(f, \mathcal{D}) \geq \theta$ **then**
      $\mathbb{F} = \mathbb{F} \cup \{f\}$
return $\mathbb{F}$

5 A Framework Instantiation

We aim to give an impression of our framework's performance compared to other approaches, and will use a proof-of-concept instantation of CobFC.

In the experimental study, we work in the usual space of data described by a vector of attribute values: given a set of attributes $\mathcal{A} = \{A_1, \ldots, A_d\}$, having domains $dom(A_i)$, each instance d is a tuple $(\vec{x}, y)$ with $\vec{x} = \langle x_1, \ldots, x_d \rangle, x_i \in dom(A_i)$, $y \in dom(C) = \{c_1, \ldots, c_m\}$ a class label.

As outlier oracle, we choose the LOF algorithm [BKNS00] in the version available for download by its authors as part of the ELKI package.[3]

To mine discriminative patterns, we use the C4.5 implementation of WEKA [FW99] (J48), with pruning turned off and minimum number of instances set to 1. We transform the resulting tree into conjunctive rules and use the left-hand side of these rules as features for augmentation. This results in binary features. Notably, this means that we can use two off-the-shelf solutions for instantiating our framework and do not have to write tailor-made code, making our approach highly modular. The ELKI package offers numerous outlier detection methods, which we plan to explore in more detail in the future.

6 Experimental Setup

We expect the CobFC features to improve the modeling of classification problems and therefore the estimated classification accuracy. To evaluate classification performance, we perform a ten-fold cross-validation on a number of UCI data sets [BM98]. We aimed for data sets of different dimensionality, size, and distribution and number of classes to allow us to evaluate the behavior of class outlier detection and the effects of derived features thoroughly.[4] For each fold, class outliers and their k-neighborhoods are extracted from the combined training folds, and conjunctive patterns mined on these subsets. To evaluate the effect on learners of different strength, we used LibSVM, as well as the J48 decision tree and the Naïve Bayes classifier implementations contained in the WEKA workbench [FW99].

6.1 Comparison techniques

We compare our approach to the feature construction technique DC-Fringe [YRB91], which had been shown to outperform the other decision tree based feature construction methods and alternatives using other models. That method shows some differences to our approach:

- Detection is handled by inspecting leaf nodes of depth at least two in a learned decision tree. As in our approach, this will lead to a focus on

[3] At `http://www.dbs.ifi.lmu.de/research/KDD/ELKI/`.

[4] We do not list their characteristics. Those can be found in supplementary material at `http://www.scientific-data-mining.org/supplementary-material.html` or accessed at the UCI repository, however.

small subsets that lie near decision boundaries but whether different classes are present in such leaves has no effect.

- Individual features are constructed as conjunction or disjunction of the last two tests leading to a given leaf. For details, we refer the reader to the original publication. Thus, while restricted in size, features are constructed using more powerful operators.

- The process is reiterated, learning a new decision tree on the augmented data, until a stopping criterion is reached, in contrast to our method, which constructs features only once.

DC-FRINGE was proposed in the context of concept learning and binary attributes. While most of the method can be transferred to a multi-class setting, and multi-valued or numerical attributes in a straightforward manner, the stopping criterion cannot. DC-FRINGE stops when there are no more leaves having at least depth two but we noticed in our experiments that this case is typically not reached. Instead, decision trees stabilize, having the same form, and therefore leading to the same features, iteration after iteration. Hence, we modify the stopping criterion such that feature construction stops when the decision tree does not change from one iteration to the next. We used the J48 decision tree learner.

As a baseline comparison, we chose a method suggested by an anonymous reviewer of an earlier version of this work: we remove the class outliers from the training data, and train on the reduced data set. While this is somewhat counter to our purpose, it might prevent over-fitting effects caused by attempts to model unaugmented class outliers and therefore improve testing accuracies.

6.2 Experimental results outlier detection and feature generation

We used Euclidean distance for identifying outliers in data sets with numerical attributes, normalizing attribute values. We used Manhattan distance for identifying outliers in data sets with nominal and ordinal attributes, binarizing attributes to that end.

In this first section of the experimental evaluation, we report on the quantitative characteristics of the feature construction step: number of class outliers identified, number of merged k-neighborhoods, and number of features derived.

As Table 1 shows, there is a significant number of class outliers for most data sets (column 2) but in some data sets some classes do not have outliers at all (column 3), often because they are too small for the concept of class outliers to have meaning. Depending on the data set, merging outliers' k-neighborhoods can reduce their number by half (e.g. for the *Spambase* data set), or leave them unchanged (*Nursery*). The number of features, finally, can range from below ten to thousands.

It can also be seen that DC-FRINGE constructs far more features (with the notable exception of *Nursery*), and performs multiple iterations, each of which will be costlier due to the higher dimensionality of the data.

6.3 Experimental evaluation: Classification accuracy

We chose the RBF kernel for LibSVM, selecting the λ and C-parameters by grid search via internal five-fold cross-validation from the ranges $[2^{-15}, 2^3]$, and $[2^{-5}, 2^{15}]$, respectively, doubling the value in each step. J48 and Naïve Bayes were run using the standard settings predefined in WEKA. We want to stress again that these are proof-of-concept experiments, and we did not fine-tune the aspects of our framework. But, anticipating reviewer comments, we have used the Friedman-Nemenyi procedure as stipulated in [GH08], finding that none of the three techniques leads to significantly better results than any other.

An obvious risk with feature construction lies in over-fitting. We therefore report testing accuracies in the following way. In each table, for NAIVE BAYES, SVM, and J48, we show training and testing accuracies on the original, unaugmented data, as well as testing accuracies for data that has been pre-processed, either having been augmented by features or having had the class outliers removed from the training data.

If the *testing* accuracy for a pre-processing approach is higher than for the unaugmented data, the value is shown in **bold**, whereas if the *training* accuracy is higher, but the testing accuracy equal or lower – i.e., when we observe over-fitting – the value is underlined.[5]

Table 2 shows the results for NAIVE BAYES. Obviously, NAIVE BAYES, a classifier challenged when trying to learn concepts that are not linearly separable, benefits strongly from a pre-processing step of the data. This is in line with other results from the feature

[5] Full tables with training/testing accuracies can be downloaded at `http://www.scientific-data-mining.org/supplementary-material.html`.

Data Set	Outliers Avg. #	Outliers Classes w/o	Outliers Folds w/o	Merged neighborhoods	Number of features	DC-Fringe Features	DC-Fringe Iterations
Breast Cancer (Wisc.)	44.30	0.00	0	30.40	64.50	255.00	15.80
Diabetes (Pima)	47.60	0.00	0	40.60	119.20	2572.90	47.10
Ecoli	20.50	1.00	0	15.40	26.10	506.50	16.80
Glass	35.30	1.30	0	28.30	56.00	418.40	21.40
Heart Statlog	17.50	0.00	0	16.40	56.60	307.10	15.30
Ionosphere	27.30	0.00	0	19.00	23.80	81.00	8.20
Iris	8.90	0.00	0	6.30	7.00	18.70	5.20
Liver Disorders	27.50	0.00	0	14.40	60.50	1045.40	33.30
Optdigits	220.80	0.00	0	203.80	246.60	6463.00	25.80
Page blocks	528.40	0.00	0	244.60	329.70	1247.00	16.56
Pendigits	757.40	0.00	0	587.90	414.20	5035.10	22.60
Segment	163.60	0.00	0	79.60	149.90	538.30	12.20
Segnoise	105.40	0.00	0	98.00	410.60	349.70	10.40
Sonar	7.80	0.00	0	7.30	15.10	108.30	10.90
Spambase	354.20	0.00	0	126.60	408.40	8102.50	66.50
Spectrometer	39.80	8.30	0	33.70	126.50	3585.30	32.20
Vehicle	43.00	0.00	0	28.70	111.30	2971.80	23.30
Audiology	16.70	4.20	0	10.80	31.50	350.10	12.00
Breast Cancer	20.30	0.00	0	18.90	85.10	854.60	22.40
Car	247.90	1.00	0	246.70	410.00	236.80	13.90
Dermatology	20.70	0.40	0	19.70	35.20	84.80	9.70
Kr-vs-kp (Chess)	184.70	0.00	0	27.60	50.50	370.10	13.50
Lung Cancer	1.22	2.22	1	1.22	2.89	44.80	8.70
Lymph	9.30	0.60	0	8.70	26.60	138.70	9.10
Mol. Biol. (Prom.)	47.70	1.00	0	37.20	40.40	30.60	6.10
Nursery	904.90	0.20	0	904.90	2500.60	423.80	14.00
Postop. Patient Data	6.50	0.00	0	6.10	17.50	171.00	10.70
Primary Tumor	20.20	5.80	0	18.90	80.70	1509.60	19.30
Soybean	34.10	6.70	0	29.20	56.50	495.70	13.40
Splice	97.10	1.00	0	75.00	126.10	1659.22	20.00
Tic-tac-toe	62.50	0.00	0	62.50	168.20	317.60	12.10
Voting Record	47.40	0.00	0	25.50	28.80	105.80	8.00

Table 1: Quantitative characteristics of the feature construction step on a selection of UCI data sets

construction literature, as well as theoretical considerations.

CobFC helps Naive Bayes improve on two-thirds (21/32) of the data sets, compared to 13 for DC-Fringe (which uses many more features), and even the baseline leads to improvements in one-third of the cases. *When* DC-Fringe leads to improvements, the gain is occasionally much higher than for CobFC but this is paid for by a many cases of over-fitting. This phenomenon is even more pronounced for the baseline, which removes all troublesome points from the training data, rewarding simpler hypotheses.

Tuning the SVM's parameters takes so much time for the larger data sets (*Nursery*, *Optdigits*, *Pendigits*), especially in combination with the expensive DC-Fringe feature construction, that the experiments had not finished by the time of writing. We do not expect that those data sets would have made a difference in the general trend of the results, however.

The SVM is such a strong learner that it benefits only in few cases from feature construction (see Table 3), no matter the technique. Instead, the addition of attributes seems to allow too fine-grained modeling of the training data, leading to many more cases of over-fitting than for Naive Bayes, even though CobFC still performs better than DC-Fringe.

Finally, we would expect the decision tree learner to benefit greatly from DC-Fringe's features since it is its model that drives feature construction and as Table 4 shows, this is indeed the case, DC-Fringe showing its best performance. It is only marginally better than CobFC, however, and we can also observe that on all data sets for which it does not lead to an improvement, DC-Fringe leads to over-fitting. Interestingly, this is also the learner that suffers most from removal of class outliers.

For each classifier, there are data sets for which the CobFC features overfit, and Table 5 shows the effects of controlling this via a minimum support constraint. Naive Bayes results are shown in the two top rows, SVM in the eleven underneath, and J48 in the seven at the bottom. The emphasis is the same as before – **bold** for improvement over unaugmented, underlined for over-fitting. As can be seen, such over-fitting control can be beneficial, increasing the number of data sets for which the SVM can improve by six. It also shows, however, that J48 cannot be helped in this manner, and that – as too often in pattern mining – there is no clear-cut mininum support threshold.

Data set	Original Train	Original Test	CobFC Test	DC-Fringe Test	Baseline Test
Breast Cancer (Wisc.)	96.11 ± 0.26	96.14 ± 2.24	$\mathbf{96.43 \pm 2.80}$	96.00 ± 2.59	95.43 ± 3.21
Diabetes (Pima)	76.27 ± 0.73	76.18 ± 4.83	74.62 ± 4.52	70.83 ± 3.20	74.62 ± 5.92
Ecoli	88.33 ± 1.43	85.43 ± 3.82	83.65 ± 6.25	78.24 ± 7.09	84.53 ± 4.76
Glass	54.77 ± 2.79	47.36 ± 11.52	$\mathbf{65.06 \pm 12.77}$	$\mathbf{72.94 \pm 10.37}$	40.28 ± 7.83
Heart Statlog	85.76 ± 1.12	83.70 ± 7.45	$\mathbf{84.07 \pm 8.74}$	81.85 ± 6.86	83.33 ± 7.04
Ionosphere	83.57 ± 0.86	82.63 ± 5.76	$\mathbf{87.75 \pm 3.80}$	$\mathbf{89.46 \pm 4.67}$	$\mathbf{84.90 \pm 5.86}$
Iris	95.85 ± 0.87	96.00 ± 4.66	91.33 ± 6.32	94.67 ± 5.26	96.00 ± 6.44
Liver Disorders	56.88 ± 2.75	53.34 ± 5.82	$\mathbf{63.18 \pm 6.60}$	$\mathbf{66.99 \pm 6.50}$	$\mathbf{63.53 \pm 5.46}$
Optdigits	91.82 ± 0.16	91.30 ± 1.03	$\mathbf{92.17 \pm 1.32}$	$\mathbf{93.10 \pm 1.25}$	90.87 ± 1.07
Pendigits	85.86 ± 0.20	85.72 ± 0.92	84.42 ± 1.47	$\mathbf{88.24 \pm 1.08}$	85.32 ± 1.10
Spambase	79.51 ± 0.17	81.17 ± 1.17	$\mathbf{91.64 \pm 0.75}$	$\mathbf{93.61 \pm 0.82}$	78.85 ± 1.80
Page blocks	90.31 ± 0.92	90.28 ± 1.77	90.01 ± 1.90	89.56 ± 2.32	87.96 ± 5.93
Segment	80.49 ± 0.24	80.17 ± 2.23	$\mathbf{89.70 \pm 1.82}$	$\mathbf{91.69 \pm 1.55}$	$\mathbf{83.03 \pm 1.08}$
Segnoise	81.88 ± 0.30	80.87 ± 1.78	$\mathbf{89.20 \pm 3.20}$	$\mathbf{89.20 \pm 2.45}$	80.33 ± 3.27
Sonar	72.28 ± 1.35	67.83 ± 7.99	$\mathbf{69.26 \pm 6.73}$	$\mathbf{77.45 \pm 10.97}$	65.95 ± 8.65
Spectrometer	62.08 ± 1.08	42.74 ± 5.63	$\mathbf{43.30 \pm 5.91}$	42.55 ± 4.51	41.61 ± 7.48
Vehicle	46.93 ± 0.76	45.27 ± 2.26	$\mathbf{49.75 \pm 3.86}$	$\mathbf{52.01 \pm 4.11}$	$\mathbf{49.63 \pm 3.39}$
Audiology	93.76 ± 0.90	75.24 ± 5.85	$\mathbf{76.52 \pm 7.77}$	74.82 ± 6.06	65.08 ± 7.19
Breast Cancer	74.98 ± 1.27	72.38 ± 8.80	70.18 ± 10.72	67.49 ± 10.90	72.34 ± 7.57
Car	88.22 ± 0.26	86.86 ± 2.64	$\mathbf{87.03 \pm 3.73}$	$\mathbf{91.44 \pm 8.10}$	81.60 ± 2.77
Dermatology	99.33 ± 0.24	98.92 ± 1.40	97.82 ± 2.79	95.11 ± 2.47	98.38 ± 2.61
Kr-vs-kp (Chess)	85.79 ± 0.37	85.39 ± 2.13	$\mathbf{87.20 \pm 2.92}$	81.70 ± 5.30	$\mathbf{87.58 \pm 2.66}$
Lung Cancer	91.31 ± 2.50	52.50 ± 32.41	49.17 ± 28.45	36.67 ± 23.31	51.67 ± 27.72
Lymph	88.29 ± 1.11	82.57 ± 9.93	$\mathbf{83.14 \pm 6.38}$	$\mathbf{84.48 \pm 8.30}$	$\mathbf{83.81 \pm 9.03}$
Mol. Biol. (Prom.)	98.95 ± 0.49	91.36 ± 7.34	91.27 ± 8.51	87.64 ± 6.50	50.00 ± 3.71
Nursery	90.40 ± 0.13	90.31 ± 1.11	$\mathbf{93.83 \pm 0.88}$	$\mathbf{90.47 \pm 2.94}$	$\mathbf{92.52 \pm 0.82}$
Postop. Patient Data	67.04 ± 3.08	54.44 ± 12.23	$\mathbf{56.67 \pm 13.30}$	53.33 ± 17.21	$\mathbf{58.89 \pm 15.76}$
Primary Tumor	56.11 ± 0.83	43.98 ± 5.12	42.48 ± 7.61	38.95 ± 5.41	$\mathbf{44.27 \pm 7.09}$
Soybean	93.80 ± 0.41	93.26 ± 3.47	$\mathbf{93.56 \pm 2.87}$	91.51 ± 3.23	90.19 ± 2.94
Splice	96.12 ± 0.19	95.89 ± 1.17	94.98 ± 1.17	92.07 ± 1.74	95.80 ± 1.17
Tic-tac-toe	69.38 ± 1.05	67.74 ± 3.90	$\mathbf{68.48 \pm 5.16}$	$\mathbf{89.35 \pm 2.72}$	$\mathbf{68.37 \pm 3.94}$
Voting Record	91.55 ± 0.65	91.50 ± 4.16	$\mathbf{94.25 \pm 3.45}$	90.33 ± 5.06	90.58 ± 4.73
Improvements			21	13	10
Over-fitting			2	12	14

Table 2: Results of Naive Bayes for data sets pre-processed with CobFC, DC-Fringe, and the baseline approach, respectively

7 Summary and Conclusions

In this paper, we have introduced a new general framework for feature construction. We propose to use outlier detection techniques to identify class outliers, and discriminative pattern mining techniques for deriving features from the k-neighborhoods surrounding them.

We have discussed how our framework addresses the four aspects of feature construction identified in [MR89], and instantiated our framework for attribute-valued data using LOF as outlier detector, and C4.5-Rules as discriminative pattern miner.

In the experimental evaluation, we have demonstrated the usefulness of the framework for Naive Bayes and J48, while an SVM learner could not benefit much. In addition, we have shown that it is far less prone to over-fitting than feature construction that reiterates on the training data, or removing class outliers from the data.

Combined with the cheaper pre-processing cost compared to iterative feature construction, and the independence from a particular classifier, we consider this to be evidence for the promise of our new framework.

Since the experimental evaluation used only a proof-of-concept instantiation of the framework, however, there is need for further evaluation to identify effective outlier detection/pattern mining combinations. Also, while over-fitting control by minimum support showed some beneficial effects, the difficulty of choosing a good threshold means that alternative methods would be preferable. We will work on improving this aspect of the framework. Finally, we intend to explore the use of our framework for more complex representations, i.e. structured data.

References

[ACS+91] David W. Aha, Peter Clark, Steven Salzberg, Gunnar Blix, and Robin Boswell Newld. Incremental constructive induction: An instance-based approach, 1991.

[BH13] Khalid Benabdeslem and Mohammed Hindawi. Efficient semi-supervised feature selection: Constraint, Relevance

Data set	Original Train	Original Test	CobFC Test	DC-Fringe Test	Baseline Test
Breast Cancer (Wisc.)	97.12 ± 0.23	96.57 ± 1.80	96.43 ± 1.81	95.57 ± 2.37	95.14 ± 1.80
Diabetes	77.79 ± 1.29	74.62 ± 6.48	74.49 ± 5.99	69.14 ± 4.81	$\mathbf{75.66 \pm 5.18}$
Ecoli	89.58 ± 1.54	87.82 ± 5.41	84.22 ± 4.30	76.18 ± 8.91	86.33 ± 5.37
Glass	82.29 ± 4.44	70.15 ± 7.41	$\mathbf{75.30 \pm 8.52}$	$\mathbf{73.42 \pm 8.29}$	68.70 ± 8.95
Heart Statlog	86.42 ± 1.21	80.00 ± 8.76	$\mathbf{82.22 \pm 8.69}$	75.56 ± 8.04	77.41 ± 4.77
Ionosphere	98.26 ± 1.24	93.74 ± 3.75	$\mathbf{95.16 \pm 3.82}$	91.74 ± 4.14	92.02 ± 3.51
Iris	98.37 ± 1.20	96.67 ± 3.51	93.33 ± 6.29	95.33 ± 4.50	96.00 ± 6.44
Liver Disorders	77.46 ± 0.76	73.08 ± 8.01	72.24 ± 9.29	64.36 ± 7.68	69.32 ± 7.73
Spambase	95.83 ± 0.97	93.92 ± 1.42	93.38 ± 0.75	75.61 ± 42.28	91.52 ± 2.20
Page Blocks	98.21 ± 0.12	96.02 ± 0.68	95.89 ± 0.79	83.89 ± 33.91	94.90 ± 0.83
Segment	99.32 ± 0.22	97.36 ± 0.75	$\mathbf{97.88 \pm 0.75}$	$\mathbf{97.88 \pm 1.07}$	95.67 ± 1.34
Segnoise	99.73 ± 0.31	96.33 ± 1.14	$\mathbf{96.80 \pm 1.21}$	$\mathbf{97.20 \pm 0.98}$	95.40 ± 1.39
Sonar	99.73 ± 0.84	86.57 ± 6.25	79.76 ± 8.20	78.40 ± 9.14	86.07 ± 8.28
Spectrometer	10.36 ± 0.11	10.36 ± 1.01	10.36 ± 1.01	10.36 ± 1.01	10.36 ± 1.01
Vehicle	94.77 ± 1.36	83.45 ± 3.55	83.09 ± 2.51	80.84 ± 2.83	82.14 ± 2.64
Audiology	99.80 ± 0.34	78.72 ± 5.98	76.94 ± 7.55	$\mathbf{80.08 \pm 9.71}$	77.43 ± 5.62
Breast Cancer	82.86 ± 6.14	72.38 ± 7.11	72.02 ± 6.39	66.10 ± 8.56	71.66 ± 5.94
Car	100.00 ± 0.00	99.71 ± 0.41	99.25 ± 0.72	$\mathbf{99.83 \pm 0.39}$	95.25 ± 1.77
Dermatology	99.24 ± 0.43	98.10 ± 2.87	97.55 ± 2.69	96.19 ± 2.62	97.00 ± 2.69
Kr-vs-kp (Chess)	99.98 ± 0.02	99.72 ± 0.31	99.47 ± 0.33	99.62 ± 0.29	99.19 ± 0.49
Lung Cancer	76.00 ± 26.65	40.00 ± 19.56	36.67 ± 17.21	40.00 ± 25.09	36.67 ± 17.21
Lymph	96.02 ± 4.89	82.57 ± 8.88	81.05 ± 8.36	79.71 ± 10.97	$\mathbf{84.52 \pm 8.27}$
Mol. Biol. (Prom.)	99.68 ± 0.51	90.36 ± 6.68	90.36 ± 7.94	88.55 ± 7.56	50.00 ± 3.71
Postop. Patient Data	71.11 ± 0.64	71.11 ± 5.74	71.11 ± 5.74	56.67 ± 22.50	67.78 ± 11.05
Primary Tumor	63.68 ± 4.50	48.09 ± 6.73	$\mathbf{49.27 \pm 7.88}$	44.29 ± 7.90	46.61 ± 7.97
Soybean	97.48 ± 1.47	93.12 ± 2.86	93.12 ± 3.60	91.80 ± 3.93	92.53 ± 3.42
Splice	98.74 ± 1.24	96.83 ± 0.76	96.05 ± 0.89	81.95 ± 36.15	96.36 ± 0.81
Tic-tac-toe	100.00 ± 0.00	100.00 ± 0.00	98.12 ± 1.61	97.91 ± 1.77	97.92 ± 1.55
Voting	97.93 ± 0.66	96.55 ± 2.69	96.10 ± 3.87	94.01 ± 3.30	$\mathbf{97.25 \pm 2.80}$
Improvements			6	5	3
Over-fitting			11	18	17

Table 3: Classification accuracies for the SVM on data sets pre-processed with CobFC, DC-Fringe, and the baseline approach

and Redundancy. *IEEE Transactions on Knowledge and Data Engineering*, July 2013.

[BK96] Hilan Bensusan and Ibrahim Kuscu. Constructive induction using genetic programming. In *Proceedings of International Conference on Machine Learning, Evolutionary Computing and Machine Learning Workshop, Fogarty, T. and Venturini, G.(Eds)*, 1996.

[BKNS00] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. In Weidong Chen, Jeffrey F. Naughton, and Philip A. Bernstein, editors, *SIGMOD Conference*, pages 93–104. ACM, 2000.

[BM98] C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.

[BZDRN06] Björn Bringmann, Albrecht Zimmermann, Luc De Raedt, and Siegfried Nijssen. Don't be afraid of simpler patterns. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *10th European Conference on Principles and Practice of Knowledge Discovery in Databases*, pages 55–66. Springer, 2006.

[DR12] K Dhanasekaran and R Rajeswari. A research-oriented survey and current status on feature extraction, ontology construction towards natural language processing. *International Journal of Computer Science Issues (IJCSI)*, 9(3), 2012.

[EA13] Haytham Elghazel and Alexandre Aussem. Unsupervised Feature Selection with Ensemble Learning. *Machine Learning*, pages 1–24, August 2013.

[Fod02] Imola Fodor. A survey of dimension reduction techniques. Technical report, 2002.

[FW99] Eibe Frank and Ian H. Witten. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

[GE03] Isabelle Guyon and André Elisseeff. An introduction to variable and feature se-

Data set	Original Train	Original Test	CobFC Test	DC-Fringe Test	Baseline Test
Breast Cancer (Wisc.)	97.73 ± 0.47	94.57 ± 2.84	$\mathbf{96.00 \pm 2.50}$	94.14 ± 2.89	93.99 ± 3.35
Diabetes	83.41 ± 2.04	73.57 ± 4.92	$\mathbf{75.27 \pm 5.50}$	70.96 ± 4.42	$\mathbf{74.22 \pm 3.85}$
Ecoli	93.09 ± 1.26	81.84 ± 5.38	$\mathbf{83.65 \pm 6.00}$	78.26 ± 6.01	$\mathbf{82.74 \pm 6.25}$
Glass	93.10 ± 1.19	66.88 ± 11.23	$\mathbf{67.86 \pm 11.93}$	$\mathbf{72.01 \pm 10.06}$	60.30 ± 10.40
Heart Statlog	92.72 ± 1.53	76.30 ± 5.58	$\mathbf{81.48 \pm 6.76}$	74.81 ± 9.04	$\mathbf{77.78 \pm 6.98}$
Ionosphere	98.42 ± 0.39	90.02 ± 4.53	88.03 ± 2.64	$\mathbf{90.30 \pm 5.09}$	88.33 ± 5.74
Iris	98.00 ± 0.78	94.00 ± 6.63	93.33 ± 6.29	$\mathbf{94.67 \pm 4.22}$	92.00 ± 8.78
Liver Disorders	85.57 ± 3.32	64.93 ± 5.96	$\mathbf{65.83 \pm 7.26}$	63.23 ± 9.08	64.37 ± 7.55
Optdigits	98.00 ± 0.10	90.39 ± 1.01	90.09 ± 1.73	$\mathbf{92.05 \pm 0.99}$	90.23 ± 0.98
Pendigits	99.26 ± 0.06	96.37 ± 0.75	$\mathbf{96.42 \pm 0.69}$	$\mathbf{96.96 \pm 0.72}$	95.53 ± 0.66
Spambase	97.29 ± 0.22	92.25 ± 1.68	$\mathbf{92.94 \pm 0.10}$	$\mathbf{92.66 \pm 1.24}$	$\mathbf{92.76 \pm 1.42}$
Page Blocks	98.59 ± 0.13	97.02 ± 0.60	$\mathbf{97.17 \pm 0.61}$	96.37 ± 0.83	95.91 ± 0.91
Segment	99.14 ± 0.14	96.54 ± 1.40	96.41 ± 1.51	$\mathbf{96.80 \pm 1.39}$	95.97 ± 1.29
Segnoise	99.02 ± 0.34	94.87 ± 1.66	93.87 ± 2.01	94.27 ± 1.76	93.80 ± 1.48
Sonar	98.13 ± 0.88	71.17 ± 10.30	$\mathbf{74.52 \pm 7.89}$	$\mathbf{78.88 \pm 8.56}$	$\mathbf{73.52 \pm 7.79}$
Spectrometer	90.56 ± 0.58	48.96 ± 4.04	48.79 ± 5.04	42.74 ± 3.22	48.21 ± 3.51
Vehicle	91.53 ± 3.29	74.60 ± 4.24	73.76 ± 3.20	68.32 ± 4.21	70.09 ± 3.60
Audiology	91.15 ± 1.24	77.02 ± 5.65	75.26 ± 5.35	$\mathbf{77.91 \pm 8.37}$	73.93 ± 6.19
Breast Cancer	81.89 ± 1.77	72.09 ± 7.10	70.96 ± 5.58	63.29 ± 6.73	70.63 ± 4.64
Car	98.97 ± 0.21	96.24 ± 1.14	96.06 ± 3.02	$\mathbf{99.71 \pm 0.41}$	93.92 ± 1.77
Dermatology	98.03 ± 0.36	96.19 ± 3.89	96.19 ± 3.89	93.46 ± 3.64	95.63 ± 4.16
Kr-vs-kp (Chess)	99.59 ± 0.11	99.37 ± 0.55	99.25 ± 0.49	$\mathbf{99.47 \pm 0.47}$	99.16 ± 0.61
Lung Cancer	90.23 ± 4.98	35.00 ± 29.61	35.00 ± 29.61	$\mathbf{36.67 \pm 23.31}$	$\mathbf{35.83 \pm 28.88}$
Lymph	94.30 ± 1.42	75.57 ± 13.77	$\mathbf{76.29 \pm 15.46}$	$\mathbf{80.29 \pm 11.02}$	$\mathbf{77.67 \pm 13.38}$
Mol. Biol. (Prom.)	96.96 ± 0.92	74.73 ± 19.04	$\mathbf{75.64 \pm 17.66}$	$\mathbf{81.09 \pm 7.74}$	50.00 ± 3.71
Nursery	99.83 ± 0.02	99.44 ± 0.25	$\mathbf{99.84 \pm 0.09}$	$\mathbf{99.98 \pm 0.03}$	96.59 ± 0.80
Postop. Patient Data	73.21 ± 2.02	68.89 ± 10.21	68.89 ± 10.21	57.78 ± 22.10	66.67 ± 11.71
Primary Tumor	62.41 ± 1.47	40.73 ± 9.09	$\mathbf{43.07 \pm 5.09}$	38.35 ± 7.35	36.88 ± 9.95
Soybean	96.88 ± 0.68	92.83 ± 3.62	91.80 ± 4.22	91.81 ± 3.46	89.75 ± 4.68
Splice	98.18 ± 0.11	94.17 ± 1.17	94.04 ± 1.22	92.61 ± 0.81	94.08 ± 0.96
Tic-tac-toe	97.68 ± 0.48	93.21 ± 4.08	89.98 ± 4.50	$\mathbf{97.18 \pm 2.03}$	90.71 ± 2.32
Voting	97.24 ± 0.31	95.87 ± 2.36	$\mathbf{96.32 \pm 2.22}$	94.72 ± 4.33	94.94 ± 2.82
Improvements			15	16	7
Over-fitting			7	16	20

Table 4: Classification accuracies of J48 on data sets pre-processed with CobFC, DC-Fringe, and the baseline approach, respectively

Data set	Minimum Support 1%	2%	3%	4%	5%
Breast Cancer	70.18 ± 10.72	71.24 ± 11.52	71.24 ± 11.52	71.24 ± 11.52	71.24 ± 11.52
Lung Cancer			49.17 ± 28.45		49.17 ± 28.45
Breast Cancer (Wisc.)	$\mathbf{96.71 \pm 1.65}$	$\mathbf{96.71 \pm 1.65}$	$\mathbf{96.86 \pm 1.47}$	$\mathbf{96.86 \pm 1.47}$	$\mathbf{96.86 \pm 1.47}$
Diabetes	74.36 ± 5.73	74.23 ± 5.59	74.36 ± 5.73	74.09 ± 4.97	74.22 ± 4.91
Liver Disorders	$\mathbf{73.09 \pm 8.70}$	$\mathbf{73.09 \pm 9.14}$	72.80 ± 9.06	72.52 ± 9.13	72.23 ± 8.60
Spambase	$\mathbf{94.24 \pm 1.23}$	$\mathbf{94.30 \pm 1.04}$	$\mathbf{94.08 \pm 0.80}$	$\mathbf{94.30 \pm 0.75}$	$\mathbf{94.13 \pm 0.79}$
Page Blocks	95.83 ± 0.79	95.91 ± 0.81	95.93 ± 0.80	95.91 ± 0.81	$\mathbf{96.05 \pm 0.66}$
Breast Cancer	71.66 ± 6.34	$\mathbf{73.07 \pm 7.08}$	$\mathbf{72.71 \pm 7.03}$	$\mathbf{73.41 \pm 6.28}$	$\mathbf{73.08 \pm 6.15}$
Dermatology	97.55 ± 2.69	97.55 ± 2.69	97.55 ± 2.69	97.55 ± 2.69	97.55 ± 2.69
Lymph	81.05 ± 8.36	80.38 ± 8.71	79.05 ± 9.73	80.38 ± 8.71	79.71 ± 8.99
Mol. Biol. (Prom.)	90.36 ± 7.94	90.36 ± 7.94	90.36 ± 7.94	90.36 ± 7.94	90.36 ± 7.94
Soybean	92.82 ± 3.94	92.97 ± 3.65	$\mathbf{93.12 \pm 3.60}$	93.11 ± 3.47	$\mathbf{93.26 \pm 3.27}$
Splice	96.02 ± 0.94	96.02 ± 0.94	96.02 ± 0.94	96.02 ± 0.94	96.02 ± 0.94
Vehicle	73.76 ± 3.20	73.76 ± 3.20	73.16 ± 3.71	73.28 ± 3.68	73.40 ± 3.67
Audiology	75.26 ± 5.35	75.26 ± 5.35	75.26 ± 5.35	75.69 ± 4.56	75.69 ± 4.56
Breast Cancer	70.96 ± 5.58	67.87 ± 6.93	68.90 ± 5.19	70.28 ± 7.24	70.97 ± 6.26
Dermatology	96.19 ± 3.89	96.19 ± 3.89	96.19 ± 3.89	96.19 ± 3.89	96.19 ± 3.89
Postop. Patient Data	68.89 ± 10.21	68.89 ± 10.21	65.56 ± 15.23	65.56 ± 15.23	68.89 ± 10.21
Soybean	91.80 ± 4.22	91.80 ± 4.22	91.80 ± 3.80	91.80 ± 3.80	91.80 ± 3.80
Splice	94.04 ± 1.22	94.04 ± 1.22	94.04 ± 1.22	94.04 ± 1.22	94.04 ± 1.22

Table 5: Classification accuracies when using overfitting control via minimum support

lection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[GH08] Salvador Garcia and Francisco Herrera. An extension on" statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. *Journal of Machine Learning Research*, 9(12), 2008.

[GM05] Evgeniy Gabrilovich and Shaul Markovitch. Feature generation for text categorization using world knowledge. In Leslie Pack Kaelbling and Alessandro Saffiotti, editors, *IJCAI*, pages 1048–1053. Professional Book Center, 2005.

[Kra02] Krzysztof Krawiec. Genetic programming-based construction of features for machine learning and knowledge discovery tasks. *Genetic Programming and Evolvable Machines*, 3(4):329–343, 2002.

[MBN02] Luis Carlos Molina, Lluís Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *ICDM*, pages 306–313. IEEE Computer Society, 2002.

[MR89] Christopher J. Matheus and Larry A. Rendell. Constructive induction on decision trees. In N. S. Sridharan, editor, *IJCAI*, pages 645–650. Morgan Kaufmann, 1989.

[Pag89] Giulia Pagallo. Learning dnf by decision trees. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 639–644. Morgan Kaufmann, 1989.

[Paz98] Michael J Pazzani. Constructive induction of cartesian product attributes. In *Feature Extraction, Construction and Selection*, pages 341–354. Springer, 1998.

[RVL13] Marian-Andrei Rizoiu, Julien Velcin, and Stéphane Lallich. Unsupervised feature construction for improving data representation and semantics. *Journal of Intelligent Information Systems*, 40(3):501–527, 2013.

[SHGHP13] Leila Shila Shafti, Pablo A. Haya, Manuel García-Herranz, and Eduardo Pérez. Inferring eca-based rules for ambient intelligence using evolutionary feature extraction. *JAISE*, 5(6):563–587, 2013.

[SP05] Leila Shila Shafti and Eduardo Pérez. Constructive induction and genetic algorithms for learning concepts with complex interaction. In Hans-Georg Beyer and Una-May O'Reilly, editors, *GECCO*, pages 1811–1818. ACM, 2005.

[TCG+10] Marisa Thoma, Hong Cheng, Arthur Gretton, Jiawei Han, Hans-Peter Kriegel, Alexander J. Smola, Le Song, Philip S. Yu, Xifeng Yan, and Karsten M. Borgwardt. Discriminative frequent subgraph mining with optimality guarantees. *Statistical Analysis and Data Mining*, 3(5):302–318, 2010.

[YKR+08] Mingqiang Yang, Kidiyo Kpalma, Joseph Ronsin, et al. A survey of shape feature extraction techniques. *Pattern recognition*, pages 43–90, 2008.

[YRB91] Der-Shung Yang, Larry A Rendell, and Gunnar Blix. A scheme for feature construction and a comparison of empirical methods. In *IJCAI*, pages 699–704. Citeseer, 1991.

[ZBR10] Albrecht Zimmermann, Björn Bringmann, and Ulrich Rückert. Fast, effective molecular feature mining by local optimization. In José L. Balcázar, Francesco Bonchi, Aristides Gionis, and Michèle Sebag, editors, *ECML/PKDD (3)*, volume 6323 of *Lecture Notes in Computer Science*, pages 563–578. Springer, 2010.