Polynomial Association Rules with Applications to Logistic Regression

Szymon Jaroszewicz
Szczecin University of Technology
Zołnierska 49, 71-210 Szczecin, Poland
National Institute of Telecommunications
Szachowa 1, 04-894, Warsaw, Poland
sjaroszewicz@wi.ps.pl

ABSTRACT

A new class of associations (polynomial itemsets and polynomial association rules) is presented which allows for discovering nonlinear relationships between numeric attributes without discretization. For binary attributes, proposed associations reduce to classic itemsets and association rules. Many standard association rule mining algorithms can be adapted to finding polynomial itemsets and association rules. We applied polynomial associations to add non-linear terms to logistic regression models. Significant performance improvement was achieved over stepwise methods, traditionally used in statistics, with comparable accuracy.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-Data Mining

General Terms

Algorithms, Experimentation, Performance

Keywords

Association rules, continuous attributes

1. INTRODUCTION AND RELATED WORK

Association rule mining is a well established data mining approach [1, 8]. Almost all association rule mining algorithms require binary or categorical attributes. The standard approach to continuous attributes is discretization [11]. This approach can lead to significant information loss and incurs problems such as choosing correct interval widths.

We present polynomial itemsets and polynomial association rules which allow for discovering complex nonlinear relationships between attributes without the need for discretization. An itemset is defined simply as a polynomial (more strictly a monomial) on a set of the attributes, and its support, as a fraction of records in which all its attributes (in their respective powers) are close to their maximum.

KDD'06, August 20–23, 2006, Philadelphia, Pennsylvania, USA.

An application to nonlinear logistic regression is presented. Despite many new classification algorithms available, logistic regression is still the major classification workhorse in natural and social sciences. While nonlinear regression models are possible, methods for building them are very inefficient on large datasets. This part of the paper is based on [10], where a similar approach has been presented in the context of subgroup discovery. The method is motivated by boosting [5, 7], where weights are chosen for examples in such a way that the currently built classifier loses all its predictive power. New terms are then found in the reweighted dataset, making it likely that they will explain part of class variability not explained before.

As itemsets are simply monomials, they are identical to nonlinear terms used in regression models [2], and are thus naturally suited to finding terms for logistic regression. An algorithm is presented for updating logistic regression models using polynomial associations, which achieves accuracy comparable to standard stepwise methods, with dramatically better performance on large data.

There has been little work on association rules for continuous attributes, not requiring discretization. A notable exception is [12] where a general framework for defining support measures has been presented, and a measure of support for continuous attributes, not requiring discretization, defined within this framework. Our approach differs by allowing different powers of attributes significantly extending the range of nonlinear relationships which can be discovered. Also, the author believes that the measure of support introduced in this paper has a more intuitive interpretation.

Nonlinear relationships have been used in the context of equation discovery [15, 4], and classification (Support Vector Machines, spline regression). Those methods are not directly applicable to association pattern mining.

A method for using association rules to guide adding terms to logistic regression models has been presented in [6]. There has been some amount of work on using association rules to construct or update classification models, see e.g. [14]. All those methods are presented for categorical attributes only.

2. DEFINITIONS AND NOTATION

Let D be a dataset. Let $H = \{x_1, \ldots, x_n\}$ be the header of D. Let $t[x]$ denote the value of x in a record $t \in D$. We assume that all attributes $x_1, \ldots, x_n$ are continuous. Non-continuous attributes, are converted as follows. Binary attributes are replaced with continuous attributes taking values in $\{0, 1\}$ in the obvious way. Categorical attributes are replaced with binary attributes (one for each category).

2.1 Polynomial itemsets

DEFINITION 2.1. *A* polynomial itemset *is an expression of the form*

$$x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2} \dots x_{i_r}^{\alpha_r},$$

where x_{i_j} *are distinct attributes (elements of* H*), and* $\alpha_j \in \{1, 2, \dots\}$ *for all* $j \in \{1, \dots, r\}$. □

The *degree* of an itemset $I = x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2} \dots x_{i_r}^{\alpha_r}$ is defined as $\deg(I) = \sum_{j=1}^{r} \alpha_j$.

We now define support of polynomial itemsets. Intuitively we want to define the support as the proportion of records in which all factors $x_{i_j}^{\alpha_j}$ have *simultaneously* high (*i.e.* close to their respective maximums) absolute values. The reason is that when the itemset is used for example as a term in polynomial regression it should significantly influence the result in a large number of records. A formal definition is given below.

DEFINITION 2.2. *For each* $x_i \in H$*, let* $c_i \in \mathbb{R}, c_i > 0$ *be a constant such that* $c_i \cdot \max_{t \in D} |t[x_i]| = 1$*. The* support *of a polynomial itemset* $x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2} \dots x_{i_r}^{\alpha_r}$ *in a dataset* D *is defined as*

$$\text{supp}_D(x_{i_1}^{\alpha_1} x_{i_2}^{\alpha_2} \dots x_{i_r}^{\alpha_r}) = \frac{\sum_{t \in D} \prod_{j=1}^{r} |c_{i_j} t[x_{i_j}]|^{\alpha_j}}{|D|}. \quad (1)$$

□

The constants c_i always exist provided that no attribute is equal to 0 in all records.

The amount of support, a record t gives to an itemset is a value in the range $[0, 1]$. When $|t[x_{i_j}]|^{\alpha_j}$ is close to its maximum, $|c_{i_j} t[x_{i_j}]|^{\alpha_j}$ will be close to 1. If this is true for all attributes in the itemset the record will contribute highly to the support. When some attributes are close to zero, t will contribute only marginally.

Let us now look at the properties of polynomial itemsets. An itemset $I_1 = x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r}$ is a *subset* of an itemset $I_2 = x_{j_1}^{\beta_1} \dots x_{j_s}^{\beta_s}$, denoted $I_1 \sqsubseteq I_2$, if for all $k = 1, \dots, r$, there is an $l \in \{1, \dots, s\}$ such that $i_k = j_l$ and $\alpha_k \leq \beta_l$.

THEOREM 2.3. *For any two polynomial itemsets* I_1*,* I_2*;* $I_1 \sqsubseteq I_2$ *implies* $\text{supp}_D(I_2) \leq \text{supp}_D(I_1)$.

THEOREM 2.4. *For binary attributes and an itemset* $I = x_1^{\alpha_1} \dots x_r^{\alpha_r}$*, Definition 2.2 of* $\text{supp}_D(I)$ *is equivalent to classical definition of support, regardless of the values of* α_i*.*

The first proof follows since the c_i factors guarantee the absolute values of attributes never exceed 1. The second is obtained by replacing all binary attributes with continuous attributes taking values in $\{0, 1\}$.

Support can be generalized to infinite populations as follows: $\text{supp}(x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r}) = E(\prod_{j=1}^{r} |c_{i_j} t[x_{i_j}]|^{\alpha_j})$, where E is the expected value over all possible vectors t. Suppose all attributes $x_1, \dots, x_n$ are independent and uniformly distributed on $[0, 1]$. All c_i's in Definition 2.2 are 1 and thus omitted. Let us now look at properties of support of polynomial itemsets on this population. Notice first that

$$\text{supp}(x_i^{\alpha}) = \int_0^1 |x_i|^{\alpha} dx_i = \int_0^1 x_i^{\alpha} dx_i = \frac{1}{\alpha + 1}. \quad (2)$$

This shows that the support of an attribute decreases inversely proportionally to its exponent. This is desirable since it favors polynomials with low degrees.

Take now an itemset $x_{i_1}^1 \dots x_{i_r}^1$. Since all variables are independent and uniformly distributed on $[0, 1]$, we have

$$\text{supp}(x_{i_1}^1 \cdots x_{i_r}^1) = \prod_{j=1}^{r} \int_0^1 |x_{i_j}| dx_{i_j} = \frac{1}{2^r}. \quad (3)$$

It can be concluded that for independent, uniformly distributed attributes the support of polynomial itemsets with all exponents 1 behaves exactly like the support of independent binary attributes, each with support $\frac{1}{2}$. This provides further justification for the presented support measure.

Let x_1 and x_2 be two attributes such that $c_1 = c_2 = 1$, it can be shown that $\text{supp}(x_1^1 x_2^1) \geq |\text{cov}(x_1, x_2) + \overline{x}_1 \overline{x}_2|$, where $\overline{x}$ denotes the sample mean of attribute x. The inequality becomes most useful when $\overline{x}_1 = \overline{x}_2 = 0$, stating that a 2-itemset with low support necessarily has low covariance.

Outliers. Requiring that every attribute be scaled so that the maximum of its absolute value is 1 may seem very harsh; even a single outlier with a very high absolute value may cause the support of an itemset to be close to zero.

The positive side of this property is that itemsets containing attributes with outliers are naturally eliminated. If this is not desirable, outlier removal can be applied prior to polynomial itemset mining.

2.2 Polynomial association rules

For two polynomial itemsets $I = x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r}$ and $J = x_{j_1}^{\beta_1} \dots x_{j_s}^{\beta_s}$ over disjoint sets of attributes, their *union*, IJ is defined as a polynomial itemset $IJ = x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r} x_{j_1}^{\beta_1} \dots x_{j_s}^{\beta_s}$.

DEFINITION 2.5. *A* polynomial association rule *is a pair* $I \to J$*, of polynomial itemsets* I*,* J *with disjoint sets of attributes. Define its* confidence *as* $\text{conf}_D(I \to J) = \frac{\text{supp}_D(IJ)}{\text{supp}_D(I)}$. □

Intuitively we want the absolute value of J to be high in those records where the absolute value of I high. The above definition loosely corresponds to the proportion of records with high value of I where the value of J is also high. For binary attributes the definition becomes classical confidence.

3. IMPLEMENTATION

Due to Theorem 2.3 many Apriori style frequent itemset mining algorithms can be applied to mining polynomial itemsets after only minimal changes.

Figure 1 shows the adaptation of the Apriori algorithm [1]. Itemsets are generated in the order of increasing degree, support counting in step 2 is done by a simple database scan.

Candidate generation is done in steps 5 to 8. Note step 6, which increases the exponent of the last attribute in the itemset. This corresponds to adding to the itemset an attribute which is already present in it, which is the main difference from standard Apriori candidate generation. Steps 7 and 8 generate more candidates than Apriori for the sake of simplification, extra candidates are removed in step 11.

3.1 Performance analysis

Unfortunately support does not decrease fast enough with the increase of degrees of attributes in the itemset (compare Equations 3 and 2). Maximum degree of itemsets was thus limited to 5. In practice this limit is not a problem since polynomials of very high degree are rarely useful.

Input: dataset D with header H, min. support $\min_{\text{supp}}$
Output: all polynomial itemsets with support $\geq \min_{\text{supp}}$

1: $k \leftarrow 1$; $C_1 \leftarrow \{x_i^1 : x_i \in H\}$
2: compute support of all itemsets in C_k
3: $F_k \leftarrow \{I \in C_k : \text{supp}_D(I) \geq \min_{\text{supp}}\}$
4: $C_{k+1} \leftarrow \emptyset$
5: **for all** $I = x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r} \in F_k$ **do**
6: add $x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r+1}$ to C_{k+1}
7: **for all** $j > i_r$ **do**
8: add $x_{i_1}^{\alpha_1} \dots x_{i_r}^{\alpha_r} x_j^1$ to C_{k+1}
9: **end for**
10: **end for**
11: remove from C_{k+1} all itemsets with a subset ($\sqsubseteq$) of degree k not in F_k
12: $k \leftarrow k+1$; **goto** 2

Figure 1: The `PolyApriori` algorithm

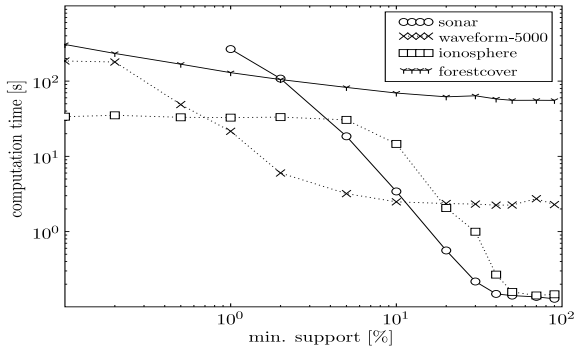

Figure 2: Performance of `PolyApriori` on benchmark datasets for different values of $\min_{\text{supp}}$

We implemented a depth first version of the algorithm in Python on a 1.7GHz Pentium4 machine. Figure 2 shows the performance of the mining algorithm for a number of datasets. A mixture of small and large datasets was used (see Table 1). The class attribute was treated like any other categorical attribute. Times for the `spambase` and `iris` datasets were dominated by startup time and are omitted for clarity. It can be seen that the algorithm performs well, even for large datasets, until the minimum support becomes too low. This is consistent with frequent set mining algorithms for binary data.

Short running time for `spambase` dataset is explained by small number of frequent itemsets it generates. This is a result of very skewed distributions of the items, when effects of outliers come into play.

4. ILLUSTRATIVE EXAMPLES

We now give some examples of polynomial association rules found in an artificial dataset and the `sonar` data from the UCI repository. The artificial dataset `sin` consists of 10000 points drawn uniformly at random from the interval $[-1,1] \times [-1,1]$. To each point (x_1, x_2) we assign a class y as follows:

$$y(x_1, x_2) = \begin{cases} 1 & \text{if } |x_2| > |sin(\pi x_1)|, \\ 0 & \text{otherwise.} \end{cases}$$

Figure 3a depicts the situation. It can be seen that the relationship between x_1, x_2 and y is highly nonlinear. Table 2

dataset	records	classes	attributes
sin	10000	2	2
sonar	208	2	61
spambase	4601	2	58
iris	150	3	4
segment	2310	7	10
glass	214	7	10
ionosphere	351	2	35
waveform-5000	5000	3	41
forest-cover	100000*	7	58

*) random sample from the full dataset

Table 1: Parameters of datasets used

rule			support	confidence
x_2^3	$\rightarrow$	y	0.1556	0.6253
x_2^2	$\rightarrow$	y	0.1906	0.5744
x_2^1	$\rightarrow$	y	0.2478	0.4974
$x_1^1 x_2^1$	$\rightarrow$	y	0.1234	0.4962
$\emptyset$	$\rightarrow$	y	0.3595	0.3595

Table 2: Association rules with highest confidence for the artificial `sin` dataset.

shows four association rules with the highest confidence for consequent y, as well as the rule $\emptyset \rightarrow y$ for comparison. Rules have been mined with $\min_{\text{supp}}$ of 10% and maximum itemset degree of 4 allowed.

It can be seen in Figure 3a that the class y is equal to 1 mostly for values of x_2 close to -1 or 1. Similarly, the polynomial itemset x_2^3 has high absolute value for x_2 close to -1 or 1 and value close to 0 otherwise. This explains high confidence of the rule $x_2^3 \rightarrow y$. Similar argument holds for $x_2^2 \rightarrow y$ and $x_2 \rightarrow y$ but since absolute values of x_2^2 and x_2 are high over a larger area their confidences are lower. The itemset $x_1^1 x_2^1$ has high absolute value close to the 'corners' of the domain. This is also where the class is 1, thus the high confidence of $x_1^1 x_2^1 \rightarrow y$, see the contour plot in Figure 3b.

The confidence of $\emptyset \rightarrow y$ is much lower than that of the most confident rules, which shows that they correctly identify areas where $y = 1$. The example also shows some limitations of expressiveness of polynomial association rules. No single rule is able to separate the area of $y = 1$ near the $x_1 = 0$ axis from the areas of $y = 0$ to the left and to the right. However linear combinations of polynomial association rules can be much more expressive; a classifier based on polynomial itemsets achieves very high accuracy on the dataset, see Section 6.

Table 3 shows the most confident rules found in the `sonar` data to predict *class='rock'*. Several non-linear rules with high confidence have been discovered. The confidence of the top rules is significantly higher than that of the rule $\emptyset \rightarrow$*class='rock'*.

rule			support	confidence
x_{44}^2	$\rightarrow$	*class='rock'*	0.0972	0.7788
$x_{27}^2 x_{44}^1$	$\rightarrow$	*class='rock'*	0.1145	0.7341
$x_{27}^2 x_{48}^1$	$\rightarrow$	*class='rock'*	0.1024	0.7302
$x_{26}^1 x_{27}^1 x_{44}^1$	$\rightarrow$	*class='rock'*	0.1101	0.7281
$\emptyset$	$\rightarrow$	*class='rock'*	0.5337	0.5337

Table 3: Association rules with highest confidence for the `sonar` dataset.

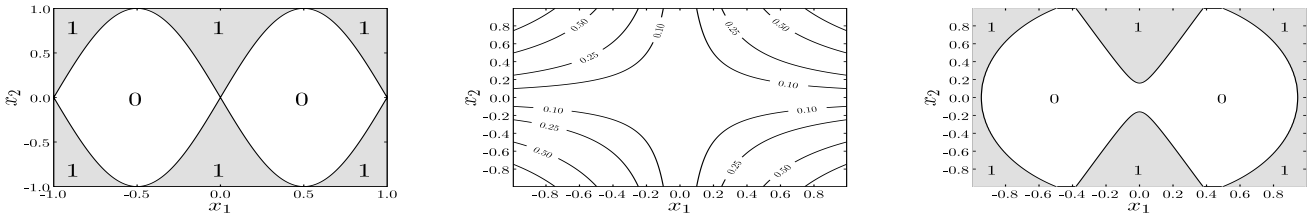

Figure 3: The artificial `sin` dataset (a), contour plot of a polynomial itemset $x_1^1 x_2^1$ (b), and the decision boundary of the classifier given in Equation 4 (c).

5. APPLICATIONS TO BUILDING LOGISTIC REGRESSION MODELS

Let us extend the header of D with a binary attribute y which will be our *target* attribute. Logistic regression models the probability of $y = 1$ in terms of $x_1, \ldots, x_n$ using the following equation

$$\text{logit}(\Pr\{y = 1\}) = \beta_0 + \beta_1 x_1 + \ldots + \beta_n x_n,$$

where $\text{logit}(p) = \log\left(\frac{p}{1-p}\right)$ is the *logit link function* which maps the predicted probability onto $(-\infty, \infty)$, such that it can be predicted by a linear model, see [2, 9] for details. Coefficients $\beta_0, \beta_1, \ldots, \beta_n$ are estimated using the maximum likelihood method. The model can be used to predict the value of $\Pr\{y = 1\}$ (by inverting the logit function).

The method is not limited to linear models and continuous data. Categorical variables can be incorporated by replacing each of them with a number of real zero-one variables. Nonlinearity is usually handled by including monomials in $x_1, \ldots, x_n$ as terms in the model. Those monomials are in fact identical to our polynomial itemsets. See [2] for details.

Most statistical packages include procedures for automatic selection of such non-linear terms. Most procedures work in a stepwise fashion [2, 9]. In forward steps the improvement (measured *e.g.* using AIC score) made to the model by adding each selected new term is computed. The best term is selected and added to the model. The backward step deletes the term whose removal gives the maximum model improvement. In the forward step not all possible terms are checked. Usually, new terms are obtained by adding every possible new variable to terms already in the model.

Since the coefficients of the model need to be recalculated for each possible term, the procedure can be very inefficient. Also, only a small fraction of the search space can effectively be checked, despite very long computation times. Another problem is that, most implementations do not take into account for the fact that statistical tests are repeated *many* times during the procedure, which may result in overfitting.

Despite those shortcomings, logistic regression is still the major classification workhorse in natural and social sciences. This is due to well established statistical procedures for model verification, availability in major statistical packages, as well as a long tradition.

5.1 Using Polynomial Itemsets to Build Logistic Regression Models

We use polynomial association rules to improve logistic regression models, making it possible to obtain small, simple models which can be constructed efficiently and nevertheless provide high classification accuracy. This is achieved with a *standard* logistic regression model to which all standard statistical tools can be applied, and which, as long as the number of terms is not too large, is human understandable.

Input: $\min_{\text{supp}}$ – minimum support, $\max_{\text{deg}}$ – maximum itemset degree, dataset D with header $H = \{x_1, \ldots, x_n, y\}$
Output: T_{best}, β_i – terms and coefficients of the best model

1: Split D into training set D_t and validation set D_v
2: $T \leftarrow \emptyset$ ▷ set of terms of the logistic regression model
3: $T_{best} \leftarrow T$; $\text{acc}_{best} \leftarrow -\infty$
4: Fit the model $\text{logit}(\Pr\{y = 1\}) = \beta_0 + \sum_{I \in T} \beta_I I$ on the (unweighted) training set D_t
5: $\text{acc}_v \leftarrow$ accuracy of fitted model on (unweighted) D_v
6: **if** $\text{acc}_v > \text{acc}_{best}$ **then**
7: $\quad T_{best} \leftarrow T$; $\text{acc}_{best} \leftarrow \text{acc}_v$
8: **end if**
9: $\text{acc}_t \leftarrow$ accuracy of fitted model on (unweighted) D_t
10: **for all** $t \in D_t$ **do**

$$t[w] \leftarrow \begin{cases} 1 & \text{if } t[y] \text{ is correctly predicted,} \\ \frac{\text{acc}_t}{1-\text{acc}_t} & \text{otherwise;} \end{cases}$$

11: **end for**
12: re-scale the weights of records in D_t to add up to 1
13: $F \leftarrow$ frequent polynomial itemsets in D_t (weighted). Class attribute y is ignored.
14: Pick $I^* = x_{i_1}^{\alpha_1} \ldots x_{i_r}^{\alpha_r}$ with highest weighted linear correlation with y on D_t
15: $T \leftarrow T \cup \{I^*\}$
16: **goto** 4 ▷ see text for the stopping criterion
17: Re-fit the best model on full dataset D

Figure 4: The `PolyForward` algorithm for building logistic regression models based on polynomial itemsets.

Figure 4 shows the `PolyForward` algorithm for updating logistic regression models using polynomial itemsets. The name comes from the fact that the algorithm implements the forward stage of a stepwise algorithm. In the Figure, $t[w]$ denotes the weight of record t.

Overall, the algorithm works as follows: at each iteration the training set is reweighted so that the current model loses all predictive power [5, 7] on it. Frequent polynomial itemsets are the mined from reweighted data. The frequent itemset most correlated with the class y is picked and used as the new model term. The rationale is that polynomial itemsets provide a source of nonlinear terms of correlated variables, each term explaining part of variability of y not explained by previous terms due to the reweighting in step 12.

To avoid overfitting, the initial dataset D is split into training (D_t) and validation (D_v) parts. D_t is used to fit logistic regression models, and D_v for selecting the best model. We use two thirds for training and one third for validation.

In step 4 current models coefficients are found on unweighted training set. The model is tested on validation set (step 5) and its accuracy compared with current winner. In step 12 the training dataset is reweighted such that the current model has no predictive power; under the new weights, its accuracy is exactly 0.5. This is exactly the type of reweighting which is done in AdaBoost and the reader is referred to boosting literature for details [5, 7]. If we want to predict probabilities, not just the class, re-weighting in step 12 should be replaced with the method from [10].

Steps 13 and 14 are the core part of the algorithm. First (weighted) frequent polynomial itemsets are found, then the one with highest weighted sample correlation coefficient with y is chosen as the next term in the model.

The use of linear correlation is not entirely legitimate as it does not coincide with maximum likelihood in case of logistic regression [2]. Ideally we should retrain the model for every new candidate term using maximum likelihood and pick the candidate which gave the biggest improvement. Such a procedure is used by stepwise regression algorithms in most statistical packages but is much to slow to apply to all frequent itemsets, so we decided to use linear correlation anyway and count on the maximum likelihood procedure to assign totally non-predictive terms weights close to zero. The biggest risk is the performance loss due to adding useless terms. We hope that the inaccuracy during term selection will be offset by much larger size of the searchspace.

Terms which did not cause an increase in accuracy are not removed from the model in hope that combined with terms added in the future they may become useful.

It has been shown in [7], that boosting algorithms are in fact building logistic regression models. However after adding each term, the coefficients remain constant. This is the main difference from our approach (apart from using polynomial terms), where *all* weights are recomputed after adding each term. We use polynomial itemsets only to *select* terms for the model, not to find the coefficients.

There are three stopping criteria (applied after step 7): achieving perfect accuracy on the training set (non-random search based on D_t is no longer possible), lack of frequent itemsets after reweighting, and an arbitrary limit of 30 on the number of iterations.

6. EXPERIMENTAL EVALUATION

Unless otherwise stated all experiments are performed with 5% minimum support and maximum itemset degree of 4.

Let us first show an illustrative example of a model found for the artificial `sin` dataset. The model contains 5 terms and is given by the equation:

$$y(x_1, x_2) = \begin{cases} 1 & \text{if } \text{logit}^{-1}(\eta) > 0.5, \\ 0 & \text{otherwise,} \end{cases} \quad (4)$$

where $\eta = -0.61 + 23.87x_2^2 + 175.81x_1^4 - 157.83x_1^2 + 87.98x_1^2x_2^2 - 16.82x_2^4$. Figure 3c shows contours of the predicted class. Despite the model's simplicity, the nonlinear relationship between x_1, x_2 and y is modelled very well. Prediction accuracy (5-fold cross-validation) is above 96%, much higher than for simple logistic regression and comparable to leading classification algorithms such as boosted decision trees.

dataset	`PolyForw.`	logistic	stepwise	AdaBoost
sin	96.00 7.4/22.6 29.89	64.05 2.0/2.0 2.56	95.96 4.0/6.4 25.16	98.14 2392/1201 21.81
sonar	77.44 5.4/18.4 598.02	75.9 60.0/60.0 1.2	78.81† 20.4/24† 377.01†	81.66 177/92 2.604
spambase	91.17* 26.0/28.2* 59.28*	91.6 57.0/57.0 5.2	—† —† $>$ 6 hours†	94.58 1869/940 90.48
segment	96.10 77.4/245.4 153.31	95.0 399/399 26.80	97.10† 63/79† 190.886†	98 748/379 11.372
glass	64.51 35.8/137.8 33.65	61.7 135/135 3.42	67.74 65.2/75.8 32.86	74.74 380/195 0.942
ionosph.	89.18 10.6/38.8 109.71	88.3 34.0/34.0 1.14	87.76 18.2/120.4 2100.4	91.46 193/102 2.56
waveform	85.20 62/135 291.22	86.5 120/120 32.78	85.78† 90/135† 1920.0†	81.6 3823/1916 87.56
forest‡	80.01 256/830 8821	— — $>$ 6 hours	—† —† $>$ 6 hours†	out of mem.

each cell: accuracy [%]; model size; computation time [s]
*) $\min_{\text{supp}}$ lowered to 1% (no itemsets frequent at 5%)
†) max. degree 2 used (performance of stepwise regression)
‡) sample of 50000 used for memory consumption reasons

Table 4: Performance comparison of `PolyForward` with other classification algorithms

6.1 Performance Evaluation

In this section we evaluate the performance and accuracy of the `PolyForward` algorithm and compare it to boosted decision trees, a leading classification algorithm. It seems natural to also compare with Support Vector Machines, esp. with polynomial kernels. Unfortunately we found SVMs to require individual tuning of parameters for each dataset, and using default parameters gave poor results. We thus compared only with boosted decision trees. It should be noted that both boosted decision trees and SVMs produce huge models, involving large trees and large numbers of support vectors (often over half of the training set), while our models stay simple most of the time and can at least be inspected, if not understood, by the user.

For boosted decision trees, the `AdaBoostM1` algorithm implemented in the `Weka` [13] package was used. The boosted classifiers were Quinlan's J4.8 decision trees. To compare with standard statistical logistic regression procedures, we used the stepwise regression implemented in the `R` package [9]. For stepwise regression we started from an empty model, used only forward direction with at most 30 iterations, just like in the case of `PolyForward`. For multiclass problems we use all-pairs method [3].

To assess prediction accuracy, 5-fold cross-validation was used. The same folds were used to test all algorithms. All reported quantities have been averaged over the five folds.

Results of comparison are shown in Table 4. Characteristics of datasets used were presented in Table 1. The columns of the table describe the dataset used, and respective performance of: the `PolyForward` algorithm, simple linear logistic regression model on all variables, nonlinear stepwise model built using `R` and boosted J4.8.

Every cell of the table shows the cross-validation accuracy, model size and computation time. For logistic models, size is

the total number of terms / the sum of degrees of all terms. For multiclass problems, both this values are summed over all classifiers produced, so the size can be large, even though individual classifiers are usually simple and understandable. For boosted decision trees the total number of nodes and leaf nodes (summed over all tress) is reported. Large sizes are rounded to the nearest integer.

All parameters were tuned on `sin`, `sonar` and `spambase` datasets, other datasets were tested with default parameters in order to avoid overfitting due to parameter tuning.

Boosted decision trees almost always achieved higher accuracy than logistic regression models, but the price for it is paid in model size, *much* larger then for regression models, comprising several trees, hundreds of nodes each. It is hard to imagine that the user could use such a classifier to gain understanding of the data. Note, that boosted decision trees did not use the pairwise multiclass method, which gives size disadvantage to logistic regression approach.

It can be seen that the standard stepwise procedure often gives better results for small datasets. We believe that this is primarily due to the fact that splitting into training and validation sets causes our method to lose a lot of predictive power. We were not able to complete the stepwise procedure for all large datasets, but for `sin`, `segment` and `waveform-5000` the difference was not larger than 1%, suggesting that for large datasets both methods achieve comparable accuracy.

For performance reasons the maximum term degree has often been lowered to 2 for the stepwise method. Apparently this did not adversely affect the accuracy (except for the `sin` dataset) which remained high. Thus, even though it is not entirely correct, we decided to compare stepwise regression's accuracy with accuracy of `PolyForward` with maximum term degree 4 in order to better highlight performance differences.

The performance of the `PolyForward` algorithm was vastly superior to traditional stepwise approach. This was true even after lowering the maximum degree to 2 in the stepwise approach. For `spambase` and `forest-cover` we abandoned the stepwise method after several hours. For the `forest-cover` data, `PolyForward` was the only algorithm which completed within a reasonable amount of time. Even simple logistic regression failed in this case. Even though `PolyForward` computes logistic regression models repeatedly, the models involve few terms. With a large number of terms, maximum likelihood computation becomes very slow, thus poor performance of simple logistic regression.

To assess the effects of discretization, we discretized the x_1 and x_2 attributes of the `sin` dataset into 3, 5 and 10 buckets and compared the accuracy of our approach and of boosted decision trees:

no. of buckets	`PolyForward`	AdaBoostM1.J48
3	72.81%	72.81%
5	82.63%	82.63%
10	89.44%	89.79%
undiscretized	96.00%	98.14%

It can be seen that after discretization, accuracy achieved by both methods decreases significantly in comparison to the undiscretized case.

7. CONCLUSIONS AND FUTURE WORK

In the paper a new kind of associations: polynomial itemsets and association rules have been introduced, which allow for discovery of nonlinear relationships between numerical attributes without discretization. Polynomial itemsets can be efficiently discovered using modified association rule mining algorithms. An application to adding nonlinear terms to logistic regression models was presented, and shown experimentally to offer accuracy comparable to standard methods and a dramatic performance improvement for large datasets.

There are several alternatives and modifications to the definition of support for polynomial itemsets which we are planning to investigate in the future.

8. REFERENCES

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD*, pages 207–216, 1993.

[2] A. Agresti. *An Introduction to Categorical Data Analysis*. John Wiley & Sons, Inc., New York, 1996.

[3] K. Duan and S. Keerthi. Which is the best multiclass SVM method? An empirical study. In *Multiple Classifier Systems, 6th International Workshop (MCS'05)*, pages 278–285, Seaside, CA, 2005.

[4] S. Dzeroski and L. Todorovski. Discovering dynamics. In *ICML*, pages 97–103, 1993.

[5] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[6] J. Freyberger, N.T. Heffernan, and C. Ruiz. Using association rules to guide a search for best fitting transfer models of student learning. In *Workshop on Analyzing Student-Tutor Interaction Logs to Improve Educational Outcomes at the 7th Annual Intelligent Tutoring Systems Conference*, Maceio, Brazil, 2004.

[7] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: a statistical view of boosting. Technical report, Dept. of Statistics, Stanford University, 1998.

[8] B. Goethals. Survey on frequent pattern mining. Manuscript, 2003.

[9] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2005. `http://www.R-project.org`.

[10] M. Scholz. Sampling based sequential subgroup mining. In *Proc. of the 11th International Conference on Knowledge Discovery and Data Mining (KDD'05)*, pages 265–274, Chicago, IL, August 2005.

[11] R. Srikant and R. Agrawal. Mining quantitative association rules in large relational tables. In *ACM SIGMOD*, pages 1–12, Montreal, Canada, 1996.

[12] M. Steinbach, P.-N. Tan, H. Xiong, and V. Kumar. Generalizing the notion of support. In *KDD*, pages 689–694, Seattle, WA, August 2004.

[13] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

[14] X. Yin and J. Han. CPAR: Classification based on predictive association rules. In *SIAM International Conference on Data Mining (SDM)*, 2003.

[15] R. Zembowicz and J. M. Zytkow. Discovery of equations: Experimental evaluation of convergence. In *10th National Conference on Artificial Intelligence (AAAI)*, pages 70–75, San Jose, CA, 1992.