Analysis of feature weighting methods based on feature ranking methods for classification

Norbert Jankowski and Krzysztof Usowicz

Department of Informatics, Nicolaus Copernicus University, Toruń, Poland

Abstract. We propose and analyze new fast feature weighting algorithms based on different types of feature ranking. Feature weighting may be much faster than feature selection because there is no need to find cut-threshold in the raking. Presented weighting schemes may be combined with several distance based classifiers like SVM, kNN or RBF network (and not only). Results shows that such method can be successfully used with classifiers.

Keywords: Feature weighting, feature selection, computational intelligence

1 Introduction

Data used in classification problems consists of instances which typically are described by features (sometimes called attributes). The feature relevance (or irrelevance) differs between data benchmarks. Sometimes the relevance depends even on the classifier model, not only on data. Also the magnitude of feature may provide stronger or weaker influence on the usage of a given metric. What's more the values of feature may be represented in different units (keeping theoretically the same information) what may provide another source of problems (for example milligrams, kilograms, erythrocytes) for classifier learning process. This shows that feature selection must not be enough to solve a hidden problem. Obligatory usage of data standardization also must not be equivalent to the best way which can be done at all. It may happen that subset of features are for example counters of word frequencies. Then in case of normal data standardization will loose (almost) completely the information which was in a subset of features.

This is why we propose and investigate several methods of automated weighting of features instead of feature selection. Additional advantage of feature weighting over feature selection is that in case of feature selection there is not only the problem of choosing the ranking method but also of choosing the cut-threshold which must be validated what generates computational costs which are not in case of feature weighting. But not all feature weighting algorithms are really fast. The feature weightings which are wrappers (so adjust weights and validate in a long loop) [21, 18, 1, 19, 17] are rather slow (even slower than feature selection), however may be accurate. This provided us to propose several feature weighting methods based on feature ranking methods. Previously rankings were used to build feature weighting in [9] were values of mutual information were used directly as weights and in [24] used χ^2 distribution values for weighting. In this article we also present selection of appropriate weighting schemes which are used on values of rankings.

Below section presents chosen feature ranking methods which will be combined with designed weighting schemes that are described in the next section (3). Testing methodology and results of analysis of weighting methods are presented in section 4.

2 Selection of rankings

The feature ranking selection is composed of methods which computation costs are relatively small. The computation costs of ranking should never exceed the computation costs of training and testing of final classifier (the kNN, SVM or another one) on average data stream. To make the tests more trustful we have selected ranking methods of different types as in [7]: based on correlation, based on information theory, based on decision trees and based on distance between probability distributions.

Some of the ranking methods are supervised and some are not. However all of them shown here are supervised. Computation of ranking values for features may be independent or dependent. What means that computation of next rank value may (but must not) depend on previously computed ranking values. For example Pearson correlation coefficient is independent while ranking based on decision trees or Battiti ranking are dependant.

Feature ranking may assign high values for relevant features and small for irrelevant ones or vice versa. First type will be called *positive feature ranking* and second *negative feature ranking*. Depending on this type the method of weighting will change its tactic.

For further descriptions assume that the data is represented by a matrix X which has m rows (the instances or vectors) and n columns called features. Let the $\mathbf{x}$ mean a single instance, $\mathbf{x}_i$ being i-th instance of X. And let's X_j means the j-th feature of X. In addition to X we have vector $\mathbf{c}$ of class labels.

Below we describe shortly selected ranking methods.

Pearson correlation coefficient ranking (CC): The Pearson's correlation coefficient:

$$CC(X_j, c) = \left[\sum_{i=1}^{m} (x_i^j - \bar{X}^j)(c_i - \bar{c}) \right] \Big/ (\sigma_{X^j} \cdot \sigma_c) \tag{1}$$

is really useful as feature selection [14, 12]. $\bar{X}^j$ and σ_{X^j} means average value and standard deviation of j-th feature (and the same for vector $\mathbf{c}$ of class labels). Indeed the ranking values are absolute values of CC:

$$J_{CC}(X_j) = |CC(X^j, c)| \tag{2}$$

because correlation equal to -1 is indeed as informative as value 1. This ranking is simple to implement and its complexity is low $O(mn)$. However some difficulties arise when used for nominal features (with more then 2 values).

Fisher coefficient: Next ranking is based on the idea of Fisher linear discriminant and is represented as coefficient:

$$J_{FSC}(X_j) = \left[\bar{X}^{j,1} - \bar{X}^{j,2} \right] / \left[\sigma_{X^{j,1}} + \sigma_{X^{j,2}} \right], \tag{3}$$

where indices $j,1$ and $j,2$ mean that average (or standard deviation) is defined for j-th feature but only for either vectors of first or second class respectively. Performance

of feature selection using Fisher coefficient was studied in [11]. This criterion may be simply extended to multiclass problems.

χ^2 coefficient: The last ranking in the group of correlation based method is the χ^2 coefficient:

$$J_{\chi^2}(X_j) = \sum_{i=1}^{m}\sum_{k=1}^{l} \frac{\left[p(X_j = x_i^j, C = c_k) - p(X_j = x_i^j)p(C = c_k)\right]^2}{p(X_j = x_i^j)p(C = c_k)}. \tag{4}$$

Using this method in context of feature selection was discussed in [8]. This method was also proposed for feature weighting with the kNN classifier in [24].

2.1 Information theory based feature rankings

Mutual information ranking (MI): Shannon [23] described the concept of entropy and mutual information. Now the concept of entropy and mutual information is widely used in several domains. The entropy in context of feature may be defined by:

$$H(X_j) = -\sum_{i=1}^{m} p(X_j = x_i^j)\log_2 p(X_j = x_i^j) \tag{5}$$

and in similar way for class vector: $H(\mathbf{c}) = -\sum_{i=1}^{m} p(C = c_i)\log_2 p(C = c_i)$.

The *mutual information (MI)* may be used as a base of feature ranking:

$$J_{MI}(X_j) = I(X_j, \mathbf{c}) = H(X_j) + H(\mathbf{c}) - H(X_j, \mathbf{c}), \tag{6}$$

where $H(X_j, \mathbf{c})$ is joint entropy. Mutual information was investigated as ranking method several times [3, 14, 8, 13, 16]. The MI was also used for feature weighting in [9].

Asymmetric Dependency Coefficient (ADC) is defined as mutual information normalized by entropy of classes:

$$J_{ADC}(X_j) = I(X_j, \mathbf{c})/H(\mathbf{c}). \tag{7}$$

These and next criterions which base on MI were investigated in context of feature ranking in [8, 7].

Normalized Information Gain (US) proposed in [22] is defined by the MI normalized by the entropy of feature:

$$J_{ADC}(X_j) = I(X_j, \mathbf{c})/H(X_j). \tag{8}$$

Normalized Information Gain (UH) is the third possibility of normalizing, this time by the joint entropy of feature and class:

$$J_{UH}(X_j) = I(X_j, \mathbf{c})/H(X_j, \mathbf{c}). \tag{9}$$

Symmetrical Uncertainty Coefficient (SUC): this time the MI is normalized by the sum of entropies [15]:

$$J_{SUC}(X_j) = I(X_j, \mathbf{c})/(H(X_j, \mathbf{c}) + H(\mathbf{c})). \tag{10}$$

It can be simply seen that the normalization is like weight modification factor which has influence in the order of ranking and in pre-weights for further weighting calculation.

Except the DML all above MI-based coefficients compose positive rankings.

2.2 Decision tree rankings

Decision trees may be used in a few ways for feature selection or ranking building. The simplest way of feature selection is to select features which were used to build the given decision tree to play the role of the classifier. But it is possible to compose not only a binary ranking, the criterion used for the tree node selection can be used to build the ranking.

The selected decision trees are: CART [4], C4.5 [20] and SSV [10]. Each of those decision trees uses its own split criterion, for example CART use the GINI or SSV use the separability split value. For using SSV in feature selection please see [11].

The feature ranking is constructed basing on the nodes of decision tree and features used to build this tree. Each node is assigned to a split point on a given feature which has appropriate value of the split criterion. These values will be used to compute ranking according to:

$$J(X_j) = \sum_{n \in Q_j} split(n), \tag{11}$$

where Q_j is a set of nodes which split point uses feature j, and $split(n)$ is the value of given split criterion for the node n (depend on tree type). Note that features not used in tree are not in the ranking and in consequence will have weight 0.

2.3 Feature rankings based on probability distribution distance

Kolmogorov distribution distance (KOL) based ranking was presented in [7]:

$$J_{KOL}(X_j) = \sum_{i=1}^{m} \sum_{k=1}^{l} \left| p(X_j = x_i^j, C = c_k) - p(X_j = x_i^j) p(C = c_k) \right| \tag{12}$$

Jeffreys-Matusita Distance (JM) is defined similarly to the above ranking:

$$J_{JM}(X_j) = \sum_{i=1}^{m} \sum_{k=1}^{l} \left| \sqrt{p(X_j = x_i^j, C = c_k)} - \sqrt{p(X_j = x_i^j) p(C = c_k)} \right|^2 \tag{13}$$

MIFS ranking. Battiti [3] proposed another ranking which bases on MI. In general it is defined by:

$$J_{MIFS}(X_j|S) = I((X_j, \mathbf{c})|S) = I(X_j, \mathbf{c}) - \beta \cdot \sum_{s \in S} I(X_j, X_s). \tag{14}$$

This ranking is computed iteratively basing on previously established ranking values. First, as the best feature, the j-th feature which maximizes $I(X_J, \mathbf{c})$ (for empty S) is chosen. Next the set S consists of index of first feature. Now the second winner feature has to maximize right side of Eq. 14 with the sum over non-empty S. Next ranking values are computer in the same way.

To eliminate the parameter β Huang et. al [16] proposed a changed version of Eq.14:

$$J_{SMI}(X_j|S) = I(X_j, \mathbf{c}) - \sum_{s \in S} \left[\frac{I(X_j, X_s)}{H(X_s)} - \frac{1}{2} \sum_{s' \in S, s' \neq s} \frac{I(X_j, X_{s'})}{H(X_{s'})} \cdot \frac{I(X_{s'}, X_s)}{H(X_s)} \right] \cdot I(X_s, \mathbf{c}). \tag{15}$$

The computation of J_{SMI} is done in the same way as J_{MIFS}. Please note that computation of J_{MIFS} and J_{SMI} is more complex then computation of previously presented rankings that base on MI.

Fusion ranking (FUS). Resulting feature rankings may be combined to another ranking in *fusion* [25]. In experiments we combine six rankings (NMF, NRF, NLF, NSF, MDF, SRW[1]) as their sum. However an different operator may replace the sum (median, max, min). Before calculation of fusion ranking each ranking used in fusion has to be normalized.

3 Methods of feature weighting for ranking vectors

Direct use of ranking values to feature weighting is sometimes even impossible because we have positive and negative rankings. However in case of some rankings it is possible [9, 6, 5]. Also the character of magnitude of ranking values may change significantly between kinds of ranking methods[2]. This is why we decided to check performance of a few weighting schemes while using every single one with each feature ranking method.

Below we propose methods which work in one of two types of weighting schemes: first use the ranking values to construct the weight vector while second scheme uses the order of features to compose weight vector.

Let's assume that we have to weight vector of feature ranking $\mathbf{J} = [j_1, \ldots, J_n]$. Additionally define $J_{min} = \min_{i=1,\ldots,n} J_i$ and $J_{max} = \max_{i=1,\ldots,n} J_i$.

Normalized max filter (NMF) is defined by

$$\mathscr{W}_{NMF}(J) = \begin{cases} |J|/J_{max} & \text{for} \quad J+ \\ [J_{max} + J_{min} - |J|]/J_{max} & \text{for} \quad J- \end{cases}, \tag{16}$$

where J is ranking element of $\mathbf{J}$. $J+$ means that the feature ranking is positive and $J-$ means negative ranking. After such transformation the weights lie in $[J_{min}, J_{max}, 1]$.

Normalizing Range Filter (NRF) is a bit similar to previous weighting function:

$$\mathscr{W}_{NRF}(J) = \begin{cases} (|J| + J_{min})/(J_{max} + J_{min}) & \text{for} \quad J+ \\ (J_{max} + 2J_{min} - |J|)/(J_{max} + J_{min}) & \text{for} \quad J- \end{cases}. \tag{17}$$

In such case weights will lie in $[2J_{min}/(J_{max} + J_{min}), 1]$.

Normalizing Linear Filter (NLF) is another a linear transformation defined by:

$$\mathscr{W}_{NLF}(J) = \begin{cases} \frac{[1-\varepsilon]J + [\varepsilon - 1]J_{max}}{J_{max} - J_{min}} & \text{for} \quad J+ \\ \frac{[\varepsilon - 1]J + [1-\varepsilon]J_{max}}{J_{max} - J_{min}} & \text{for} \quad J- \end{cases}, \tag{18}$$

where $\varepsilon = -(\varepsilon_{max} - \varepsilon_{min})v^p + \varepsilon_{max}$ depends on feature. Parameters has typically values: $\varepsilon_{min} = 0.1$ and $\varepsilon_{max} = 0.9$, and p may be 0.25 or 0.5. And $v = \sigma_{\mathbf{J}}/\bar{\mathbf{J}}$ is a variability index.

[1] See Eq. 21.

[2] Compare sequence $1, 2, 3, 4$ with $11, 12, 13, 14$ further influence in metric is significantly different

Normalizing Sigmoid Filter (NSF) is a nonlinear transformation of ranking values:

$$W_{NSF}(J) = \left[1 + e^{-[\mathscr{W}(J)-0.5]\log((1-\varepsilon')/\varepsilon')^2}\right]^{-1} + \varepsilon' \tag{19}$$

where $\varepsilon' = \varepsilon/2$. This weighting function increases the strength of strong features and decreases weak features.

Monotonically Decreasing Function (MDF) defines weights basing on the order of the features, not on the ranking values:

$$W_{MDF}(j) = e^{\log\varepsilon\cdot[(j-1)/(n-1)]^{\log(n_s-1)/(n-1)\,\log_\varepsilon\tau}} \tag{20}$$

where j is the position of the given feature in order. τ may be 0.5. Roughly it means the n_s/n fraction of features will have weights not greater than tau.

Sequential Ranking Weighting (SRW) is a simple threshold weighting via feature order:

$$W_{SRW}(j) = [n+1-j]/n, \tag{21}$$

where j is again the position in the order.

4 Testing methodology and results analysis

The test were done on several benchmarks from UCI machine learning repository [2]: appendicitis, Australian credit approval, balance scale, Wisconsin breast cancer, car evaluation, churn, flags, glass identification, heart disease, congressional voting records, ionosphere, iris flowers, sonar, thyroid disease, Telugu vowel, wine.

Each single test configuration of a weighting scheme and a ranking method was tested using 10 times repeater 10 fold cross-validation (CV). Only the accuracies from testing parts of CV were used in further test processing. In place of presenting averaged accuracies over several benchmarks the paired t-tests were used to count how many times had the given test configuration won, defeated or drawn. t-test is used to compare efficiency of a classifiers without weighting and with weighting (a selected ranking method plus selected weighting scheme). For example efficiency of 1NNE classifier (one nearest neighbour with Euclidean metric) is compared to 1NNE with weighting by CC ranking and NMF weighting scheme. And this is repeated for each combination of rankings and weighting schemes. CV tests of different configurations were using the same random seed to make the test more trustful (it enables the use of paired t-test).

Table 1 presents results averaged for different configurations of k nearest neighbors kNN and SVM: 1NNE, 5NNE, AutoNNE, SVME, AutoSVME, 1NNM, 5NNM, AutoNNM, SVMM, AutoSVMM. Were suffix ‘E’ or ‘M’ means Euclidean or Manhattan respectively. Prefix ‘auto’ means that kNN chose the ‘k’ automatically or SVM chose the ‘C’ and spread of Gaussian function automatically.

Tables 1(a)–(c) presents counts of winnings, defeats and draws. Is can be seen that the best choice of ranking method were US, UH and SUC while the best weighting schemes were NSF and MDF in average. Smaller number of defeats were obtained for

KOL and FUS rankings and for NSF and MDF weighting schemes. Over all best configuration is combination of US ranking with NSF weighting scheme. The worst performance characterize feature rankings based on decision trees. Note that the weighting with a classifier must not be used obligatory. With a help of CV validation it may be simply verified whether the using of feature weighting method for given problem (data) can be recommended or not. Table 1(d) presents counts of winnings, defeats and draws per classification configuration. The highest number of winnings were obtained for SVME, 1NNE, 5NNE. The weighting turned out useless for AutoSVM[E|M]. This means that weighting does not help in case of internally optimized configurations of SVM. But note that optimization of SVM is much more costly (around 100 times—costs of grid validation) than SVM with feature weighting!

Tables 2(a)–(d) describe results for SVME classifier used with all combinations of weighting as before. Weighting for SVM is very effective even with different rankings (JM, MI, ADC, US,CHI, SUC or SMI) and with weighting schemes: NSF, NMF, NRF.

5 Summary

Presented feature weighting methods are fast and accurate. In most cases performance of the classifier may be increased without significant growth of computational costs. The best weighting methods are not difficult to implement. Some combinations of ranking and weighting schemes are often better than other, for example combination of normalized information gain (US) and NSF. Presented feature weighting methods may compete with slower feature selection or adjustment methods of classifier metaparameters (AutokNN or AutoSVM which needs slow parameters tuning). By simple validation we may decide whether to weight or not to weight features before using the chosen classifier for given data (problem) keeping the final decision model more accurate.

References

1. Aha, D.W., Goldstone, R.: Concept learning and flexible weighting. In: Proceedings of the 14th Annual Conference of the Cognitive Science Society. pp. 534–539 (1992)
2. Asuncion, A., Newman, D.: UCI machine learning repository (2007), http://www.ics.uci.edu/∼mlearn/MLRepository.html
3. Battiti, R.: Using mutual information for selecting features in supervised neural net learning. IEEE Transactions on Neural Networks 5(4), 537–550 (1994)
4. Breiman, L., Friedman, J.H., Olshen, A., Stone, C.J.: Classification and regression trees. Wadsworth, Belmont, CA (1984)
5. Creecy, R.H., Masand, B.M., Smith, S.J., Waltz, D.L.: Trading mips and memory for knowledge engineering. Communications of the ACM 35, 48–64 (1992)
6. Daelemans, W., van den Bosch, A.: Generalization performance of backpropagation learning on a syllabification task. In: Proceedings of TWLT3: Connectionism and Natural Language Processing. pp. 27–37 (1992)
7. Duch, W.: Filter methods. In: Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L. (eds.) Feature extraction, foundations and Applications, pp. 89–117. Studies in fuzziness and soft computing, Springer (2006)

40.53%	NMF	NRF	NLF	**NSF**	**MDF**	SRW	*Sum*
CC	55	59	53	59	55	52	*333*
FSC	61	58	63	72	64	64	*382*
CHI	67	66	**78**	73	66	72	*422*
MI	66	70	74	76	67	73	*426*
ADC	66	70	74	76	67	73	*426*
US	75	71	69	**79**	76	71	***441***
UH	73	71	**77**	**77**	71	71	***440***
DML	58	54	59	58	58	49	*336*
SUC	71	70	**79**	**77**	70	71	***438***
CRT	46	46	44	43	66	66	*311*
C45	46	48	41	42	64	63	*304*
SSV	50	48	48	48	63	62	*319*
KOL	62	68	66	64	68	67	*395*
JM	72	70	74	76	69	69	*430*
SMI	57	59	64	68	71	69	*388*
FUS	**78**	73	76	74	68	65	*434*
Sum	1003	1001	1039	**1062**	**1063**	1057	*6225*

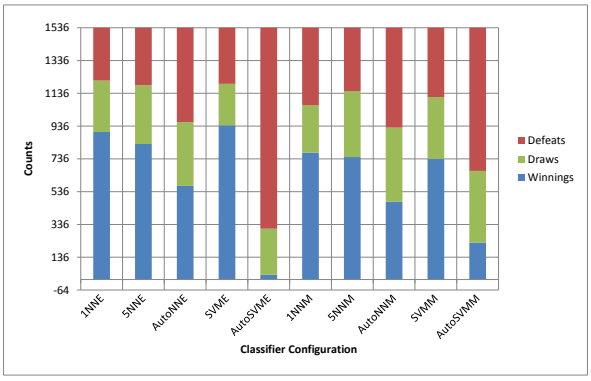
(a) Cumulative count of winnings

36.33%	NMF	NRF	NLF	**NSF**	**MDF**	SRW	*Sum*
CC	67	61	61	53	66	76	*384*
FSC	68	63	61	56	65	66	*379*
CHI	55	55	59	51	52	58	*330*
MI	53	52	59	51	52	57	*324*
ADC	53	52	59	51	52	57	*324*
US	55	47	58	45	50	60	*315*
UH	53	53	54	51	51	57	*319*
DML	41	36	56	46	58	66	*303*
SUC	52	47	57	49	51	57	*313*
CRT	92	83	94	89	52	59	*469*
C45	87	83	86	83	60	63	*462*
SSV	79	72	82	76	55	60	*424*
KOL	50	42	52	39	46	51	***280***
JM	55	52	56	50	55	56	*324*
SMI	69	68	63	56	55	56	*367*
FUS	35	33	51	36	50	58	***263***
Sum	964	899	1008	**882**	**870**	957	*5580*

(b) Cumulative count of defeats

23.14%	NMF	NRF	NLF	NSF	MDF	SRW	*Sum*
CC	38	40	46	48	39	32	*243*
FSC	31	39	36	32	31	30	*199*
CHI	38	39	23	36	42	30	*208*
MI	41	38	27	33	41	30	*210*
ADC	41	38	27	33	41	30	*210*
US	30	42	33	36	34	29	*204*
UH	34	36	29	32	38	32	*201*
DML	61	70	45	56	44	45	*321*
SUC	37	43	24	34	39	32	*209*
CRT	22	31	22	28	42	35	*180*
C45	27	29	33	35	36	34	*194*
SSV	31	40	30	36	42	38	*217*
KOL	48	50	42	57	46	42	*285*
JM	33	38	30	34	36	35	*206*
SMI	34	33	33	36	34	35	*205*
FUS	47	54	33	50	42	37	*263*
Sum	593	660	513	616	627	546	*3555*

(c) Cumulative count of draws



(d) Cumulative counts per classifier configuration

Table 1: Cumulative counts over feature ranking methods and feature weighting schemes (averaged over kNN's and SVM's configurations).

61,26%	**NMF**	**NRF**	NLF	**NSF**	MDF	SRW	*Sum*
CC	9	9	9	11	11	11	***60***
FSC	9	7	8	9	10	10	*53*
CHI	10	10	11	11	9	10	***61***
MI	11	11	11	11	9	9	***62***
ADC	11	11	11	11	9	9	***62***
US	11	12	10	10	10	9	***62***
UH	10	11	10	11	9	8	*59*
DML	8	9	9	9	9	8	*52*
SUC	11	12	10	10	9	8	***60***
CRT	10	9	10	9	10	9	*57*
C45	10	9	9	9	9	8	*54*
SSV	10	8	9	10	9	10	*56*
KOL	9	9	10	11	10	9	*58*
JM	11	11	11	12	10	11	***66***
SMI	10	10	9	10	10	11	***60***
FUS	10	11	10	10	10	8	*59*
Sum	**160**	**159**	157	**164**	153	148	*941*

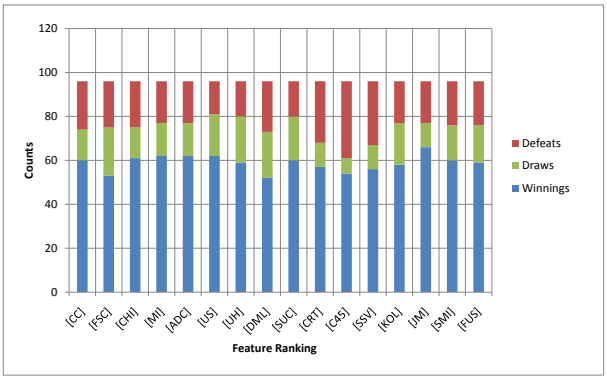
(a) Cumulative count of winnings

22.27%	**NMF**	**NRF**	NLF	**NSF**	MDF	SRW	*Sum*
CC	4	3	4	3	4	4	*22*
FSC	3	3	4	3	4	4	*21*
CHI	3	3	4	3	4	4	*21*
MI	2	2	4	3	4	4	*19*
ADC	2	2	4	3	4	4	*19*
US	2	1	3	2	3	4	***15***
UH	2	2	3	3	3	3	***16***
DML	3	3	5	3	4	5	*23*
SUC	2	2	3	3	3	3	***16***
CRT	4	5	4	5	5	5	*28*
C45	6	6	6	6	6	5	*35*
SSV	5	5	5	5	4	5	*29*
KOL	2	3	4	3	4	3	*19*
JM	2	2	4	3	4	4	*19*
SMI	4	4	3	2	4	3	*20*
FUS	3	3	4	3	4	3	*20*
Sum	**49**	**49**	64	**53**	64	63	*342*

(b) Cumulative count of defeats

16.47%	**NMF**	**NRF**	NLF	NSF	MDF	**SRW**	*Sum*
CC	3	4	3	2	1	1	*14*
FSC	4	6	4	4	2	2	***22***
CHI	3	3	1	2	3	2	*14*
MI	3	3	1	2	3	3	*15*
ADC	3	3	1	2	3	3	*15*
US	3	3	3	4	3	3	***19***
UH	4	3	3	2	4	5	***21***
DML	5	4	2	4	3	3	***21***
SUC	3	2	3	3	4	5	*20*
CRT	2	2	2	2	1	2	*11*
C45	0	1	1	1	1	3	*7*
SSV	1	3	2	1	3	1	*11*
KOL	5	4	2	2	2	4	*19*
JM	3	3	1	1	2	1	*11*
SMI	2	2	4	4	2	2	*16*
FUS	3	2	2	3	2	5	*17*
Sum	**47**	**48**	35	39	39	**45**	*253*

(c) Cumulative count of draws



(d) Cumulative counts per classifier configuration

Table 2: Cumulative counts over feature ranking methods and feature weighting schemes for SVM classifier.

8. Duch, W., Biesiada, T.W.J., Blachnik, M.: Comparison of feature ranking methods based on information entropy. In: Proceedings of International Joint Conference on Neural Networks. pp. 1415–1419. IEEE Press (2004)
9. D.Wettschereck, Aha, D., Mohri, T.: A review of empirical evaluation of feature weighting methods for a class of lazy learning algorithms. Artificial Intelligence Review Journal 11, 273–314 (1997)
10. Grąbczewski, K., Duch, W.: The separability of split value criterion. In: Rutkowski, L., Tadeusiewicz, R. (eds.) Neural Networks and Soft Computing. pp. 202–208. Zakopane, Poland (Jun 2000)
11. Grąbczewski, K., Jankowski, N.: Feature selection with decision tree criterion. In: Nedjah, N., Mourelle, L., Vellasco, M., Abraham, A., Köppen, M. (eds.) Fifth International conference on Hybrid Intelligent Systems. pp. 212–217. IEEE, Computer Society, Brasil, Rio de Janeiro (Nov 2005)
12. Grąbczewski, K., Jankowski, N.: Mining for complex models comprising feature selection and classification. In: Guyon, I., Gunn, S., Nikravesh, M., Zadeh, L. (eds.) Feature extraction, foundations and Applications, pp. 473–489. Studies in fuzziness and soft computing, Springer (2006)
13. Guyon, I.: Practical feature selection: from correlation to causality. http://eprints.pascal-network.org/archive/00004038/01/PracticalFS.pdf (2008), 955 Creston Road, Berkeley, CA 94708, USA
14. Guyon, I., Elisseef, A.: An introduction to variable and feature selection. Journal of Machine Learning Research pp. 1157–1182 (2003)
15. Hall, M.A.: Correlation-based feature subset selection for machine learning. Ph.D. thesis, Department of Computer Science, University of Waikato, Waikato, New Zealand (1999)
16. Huang, J.J., Cai, Y.Z., Xu, X.M.: A parameterless feature ranking algorithm based on MI. Neurocomputing 71, 1656–1668 (2007)
17. Jankowski, N.: Discrete quasi-gradient features weighting algorithm. In: Rutkowski, L., Kacprzyk, J. (eds.) Neural Networks and Soft Computing, pp. 194–199. Advances in Soft Computing, Springer-Verlag, Zakopane, Poland (Jun 2002)
18. Kelly, J.D., Davis, L.: A hybrid genetic algorithm for classification. In: Proceedings of the 12th International Joint Conference on Artificial Intelligence. pp. 645–650 (1991)
19. Kira, K., Rendell, L.A.: The feature selection problem: Traditional methods and a new algorithm. In: Proceedings of the 10th International Joint Conference on Artificial Intelligence. pp. 129–134 (1992)
20. Quinlan, J.R.: C4.5: Programs for machine learning. San Mateo, CA: Morgan Kaufmann (1993)
21. Salzberg, S.L.: A nearest hyperrectangle learning method. Machine Learning Journal 6(3), 251–276 (1991)
22. Setiono, R., Liu, H.: Improving backpropagation learning with feature selection. Applied Intelligence 6, 129–139 (1996)
23. Shannon, C.E.: A mathematical theory of communication. The Bell System Technical Journal 27, 379–423,623–656 (1948)
24. Vivencio, D.P., E. R. Hruschka, J., Nicoletti, M., Santos, E., Galvao, S.: Feature-weighted k-nearest neigbor classifier. In: Proceedings of IEEE Symposium on Foundations of Computational Intelligence (2007)
25. Yan, W.: Fusion in multi-criterion feature ranking. In: 10th International Conference on Information Fusion. pp. 1–6 (2007)