JOURNAL OF
CHEMICAL INFORMATION
AND MODELING

ARTICLE

pubs.acs.org/jcim

Classifying Molecules Using a Sparse Probabilistic Kernel Binary Classifier

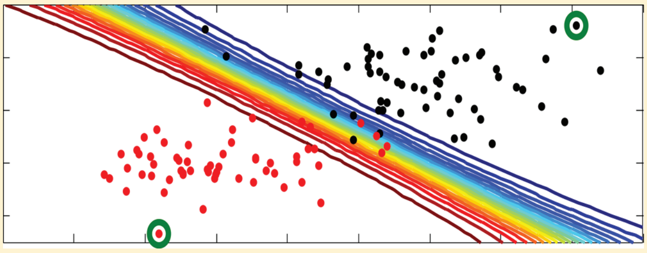
Robert Lowe,*[†] Hamse Y. Mussa,*[†] John B. O. Mitchell,[‡] and Robert C. Glen[†]

[†]Unilever Centre for Molecular Sciences Informatics, Department of Chemistry, University of Cambridge, Lensfield Road, Cambridge CB2 1EW, United Kingdom

[‡]EaStCHEM School of Chemistry and Biomedical Sciences Research Complex, University of St. Andrews, North Haugh, St. Andrews, Scotland KY16 9ST, United Kingdom

S *Supporting Information*

ABSTRACT: The central idea of supervised classification in chemoinformatics is to design a classifying algorithm that accurately assigns a new molecule to one of a set of predefined classes. Tipping has devised a classifying scheme, the Relevance Vector Machine (RVM), which is in terms of sparsity equivalent to the Support Vector Machine (SVM). However, unlike SVM classifiers, the RVM classifiers are probabilistic in nature, which is crucial in the field of decision making and risk taking. In this work, we investigate the performance of RVM binary classifiers on classifying a subset of the MDDR data set, a standard molecular benchmark data set, into active and inactive compounds. Additionally, we present results that compare the performance of SVM and RVM binary classifiers.



INTRODUCTION

Virtual screening, in particular the similarity search approach, plays a pivotal role in discovering potential lead molecules for known drug targets. In the chemoinformatics context, a similarity search is basically a supervised pattern recognition (classification) problem. The work presented in this paper is confined to binary classification problems. Given a priori a data set of size N, i.e., $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$, the essence of supervised classification is to design a classifying algorithm that optimally assigns a new molecule to one of two classes. $\mathbf{x}_i$, an l-dimensional vector commonly known as the descriptor vector, represents the information that we have about the i^{th} molecule, and y_i is the response variable denoting the class to which the i^{th} molecule belongs. Mathematically, the descriptor vector $\mathbf{x}_i$ resides in an l-dimensional space (henceforth) denoted by $\mathscr{M}$, and $y_i \in \{\omega_1, \omega_2\}$ with $\omega_1 = 1$ and $\omega_2 = 0$ because y_i is just a class label in the binary classification problem. From now on, unless stated otherwise, "classification" means "supervised classification".

In broad terms, there are two main categories of binary classifiers: "hard point" (HP) and probabilistic classifiers (PC). HP classifiers denoted here as $\mathrm{d}(\mathbf{x})$ directly map, each molecule $\mathbf{x}$ to the appropriate class[1] as follows

$$\begin{aligned} \mathrm{d}(\mathbf{x}) &> \rho \Longrightarrow \mathbf{x} \in \omega_1 \\ &< \rho \Longrightarrow \mathbf{x} \in \omega_2 \end{aligned} \qquad (1)$$

i.e., for an appropriately set threshold value ρ, the molecule $\mathbf{x}$ is assigned to ω_1 or to ω_2 if $\mathrm{d}(\mathbf{x}) > \rho$, or $\mathrm{d}(\mathbf{x}) < \rho$, respectively. In the case of $\mathrm{d}(\mathbf{x}) = \rho$, $\mathbf{x}$ is assigned arbitrarily to one of the two classes.[1,2] A Support Vector Machines (SVM) classifier[3] is a classic example of a HP classifier. The HP classifiers do not consider the inevitable uncertainties in their outputs (predictions), instead they yield hard point outputs, i.e., a "yes" or "no" response. Thus, HP classifiers are quite often not useful in (decision making) applications that involve minimizing classification risks and penalties[4]—two concepts that are fundamental in, for example, drug development and toxicology.

Ideally, PC capture the uncertainty in our classification predictions. For an optimal decision maker, the probabilities on the prediction can be germane. Moreover, unlike HP, PC can with relative ease accommodate the minimum risk and penalty decision criterion risk.[4]

In the PC group, given S, one estimates a posteriori probabilities, $p(\omega_1|\mathbf{x})$ and $p(\omega_2|\mathbf{x})$ directly (discriminative scheme) or indirectly (generative scheme), that predict the probabilities of molecule $\mathbf{x}$ belonging to class ω_1 or ω_2, respectively.[1,5,6]

The Naive Bayesian (NB) algorithm[1,7,8] and the Binary Kernel Discrimination (BKD) classifier recently introduced by Harper et al.,[9] both of which are widely used in the chemoinformatics domain, are classic examples of binary probabilistic classifiers. In NB and BKD, a new molecule $\mathbf{x}$ is assigned to class ω_1 if $p(\omega_1|\mathbf{x}) > \rho \times p(\omega_2|\mathbf{x})$; otherwise $\mathbf{x}$ is considered to belong to class ω_2. ρ is a threshold value to be set accordingly.

In the Naive Bayesian (NB) case, $p(\omega_n|\mathbf{x})$, with $n = 1, 2$, is estimated by assuming that the individual l elements of the

Received: March 17, 2011
Published: June 22, 2011

descriptor vector, $\mathbf{x} = (x_1, x_2, ..., x_l)$, are statistically independent. Although this is a severe restriction, it has been reported that Naive Bayesian classifiers perform well on classifying realistic chemical data sets.[8,10−12] Nonetheless, computing $p(\omega_n|\mathbf{x})$ can become computationally intractable if the descriptor values are continuous.[13] Usually this computational problem is circumvented by discretizing the continuous descriptors or assuming that the descriptors follow a normal distribution. However, the effectiveness of these two solutions is dependent on the accuracy of the discretization scheme and the validity of the normal distribution assumption.

The BKD method indirectly estimates $p(\omega_n|\mathbf{x})$ through[11,14]

$$p(\omega_n|\mathbf{x}) = c_n \sum_{\mathbf{x}_i \in \omega_n}^{M_n} K(\mathbf{x}_i, \mathbf{x}; \lambda) \tag{2}$$

with

$$c_n = \frac{p(\omega_n)}{M_n p(\mathbf{x})}; \; n = 1, 2$$

where $p(\mathbf{x})$, $p(\omega_n)$, and M_n denote the evidence distribution, the ω_n class prior probability, and the number of molecules known to be in class ω_n, respectively. $K(\mathbf{x}_i;\mathbf{x};\lambda)$ is a kernel function, and λ is a tunable parameter.[1,7,8]

In BKD, classifiers store all of the training patterns so there is no training phase to speak of. That is, there is little (or nothing) in the way of training data set reduction. Thus, the BKD classifier can be slow in the predicting phase because all the computation is delayed to that phase. Furthermore, in these types of classification schemes, according to Duda and Hart,[2] the size of training data set demanded for estimating the underlying distribution of the given data set can grow exponentially with the number of dimensions of the input space, which may lead to severe CPU time and storage requirements in classifying a new molecule.[2,11]

In short, the BKD classifiers are nonsparse probabilistic classifiers: they are nonsparse in the sense that all (or a significant amount) of the training data set is required in the predicting phase.

Sparsity is a very useful concept in pattern recognition: sparse classifiers can generalize well when tested on data points outside the training data set; and they can also be core-memory efficient and fast computationally in the predicting phase. In this regard, using Bayesian learning techniques, Tipping has devised an arguably computationally efficient algorithm that can generate sparse probabilistic classifiers.[15] He has demonstrated that with this technique, unlike BKD, a significant training data reduction is achievable, i.e., only a tiny proportion of the training data set is essential for approximating $p(\omega_n|\mathbf{x})$. Besides, in his approach, the elements of the descriptor vector $\mathbf{x}$ do not have to be discrete variables nor is it required to make an assumption about the form of their distribution function.

In passing, we note that estimates of $p(\omega_n|\mathbf{x})$ have been coerced from SVM via post processing.[16] However, in a highly cited paper,[15] it was argued that these estimates are unreliable, and as of yet we have not come across a counter argument to that asserted in ref 15 even though, in the past few years, there has been a plethora of papers inspired by the work reported in ref 16. Of course, the powerful Bayesian neural networks scheme can also provide estimates of $p(\omega_n|\mathbf{x})$.[17] However, this approach is computationally involved.[18]

The main objective of this paper is to investigate the performance of binary probabilistic classifiers based on this "new" approach, which is termed the Relevance Vector Machine (RVM),[15] on classifying a large MDDR data set into active and inactive compounds. In the following section, we briefly describe the essence of the RVM method for binary classification problems. Because this method is, in terms of the sparsity spirit, equivalent to SVM that is highly popular and widely used in chemoinformatics, the third section of the paper presents results that compare the performance of SVM and RVM classifiers on classifying the same data set. The final section gives our concluding remarks.

METHOD: RELEVANCE VECTOR MACHINE

As was stated before, in this paper, attention is confined to the RVM algorithm for binary classification, i.e., RVM is discussed and presented in the context of binary classification.

RVM is another algorithm that can estimate $p(\omega_n|\mathbf{x})$ given S. In simple terms (conceptually), the RVM approach finds the data points in the training data set, which seem to be more representative of the two classes, "prototypical" patterns/molecules of the two classes; and in agreement with intuition, it turns out that generally these patterns, which are termed the Relevance Vectors (hence, the name Relevance Vector Machine), are located away from the decision boundary. For an illustrative purpose, see the simple 2D case in Figure 1. The red and black circles denote training data sets from class ω_1 and ω_2, respectively. The green circles represent the Relevance Vectors. The surface in the top panel demonstrates the $p(\omega_{n=1}|\mathbf{x})$ yielded by the RVM algorithm, whereas the bottom panel shows the location of the decision boundary of the classifier. Contrast this with SVM, where the Support Vectors are in the vicinity of the decision boundary.

More formally, albeit briefly, in RVM, $p(\omega_n|\mathbf{x})$ is viewed as a conditional class distribution given by a Bernoulli distribution

$$p(\omega_n|\mathbf{x}) = [[1 + \exp(-f(\mathbf{x};\boldsymbol{\theta}))]^{-1}]^{\omega_n}[1 - [1 + \exp(-f(\mathbf{x};\boldsymbol{\theta}))]^{-1}]^{1-\omega_n} \tag{3}$$

where the weights θ_i in $f(\mathbf{x};\boldsymbol{\theta})$ defined as

$$f(\mathbf{x};\boldsymbol{\theta}) = \sum_{i=1}^{N} \theta_i K(\mathbf{x}_i, \mathbf{x}; \lambda) + \theta_0 \tag{4}$$

are computed/estimated (from the given data set) in a Bayesian framework.[4,15] In this framework, an explicit zero-mean Gaussian prior probability distribution is imposed over each one of the weights θ_i. These prior constraints over the weight vector coupled with a Bayesian learning approach drive a significant number of the weights θ_i to zero, i.e., a large number of these weights, θ_i, become sharply distributed about zero, which naturally leads to a substantial training data reduction (training data sparsification). In other words, in practice, a small fraction of the N kernel $K(\mathbf{x}_i;\mathbf{x};\lambda)$ terms enter eq 4 and are relevant to the estimation of $p(\omega_n|\mathbf{x})$, eq 3: those $K(\mathbf{x}_i;\mathbf{x};\lambda)$ associated with the nonzero coefficients θ_i. Tipping termed the patterns $\mathbf{x}_i$, corresponding to the relevant kernel terms the Relevance Vectors. (Note in eqs 3 and 4, $\mathbf{x}$, $\mathbf{x}_i$, $K(\mathbf{x}_i;\mathbf{x};\lambda)$, and N are as defined before; θ_0 denotes the bias term.)

In the prediction (testing) phase, the class a posteriori probability $p(\omega_1|\mathbf{x}_{new})$ for a new molecule $\mathbf{x}_{\text{new}}$ is obtained by simply substituting nonzero weights θ_i and their corresponding training data points $\mathbf{x}_i$ (the Relevance Vectors) into eq 3 via eq 4.

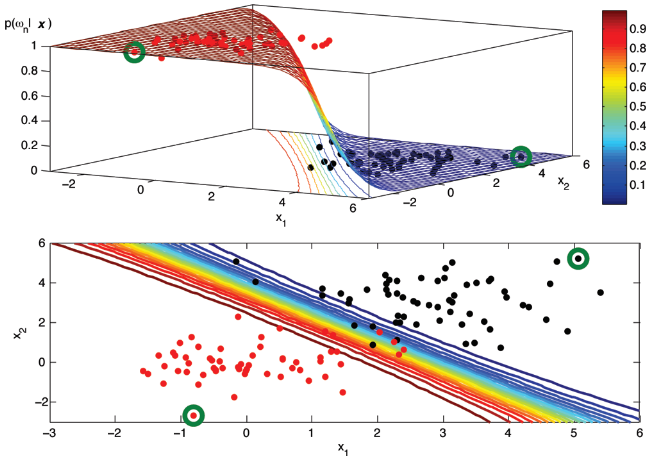


Figure 1. Plot of a two-dimensional toy data set with two different class labels ω_1 = red and ω_2 = black. The points highlighted in green are the so-called Relevance Vectors. In comparison to the Support Vectors (not shown here) that are in the vicinity of the decision boundary, these points are generally the furthest points from the decision boundary. The colors represent the probability, $p(\omega_{n=1}|\theta)$, where the boundary for classes is 0.5 (colored in green). Unlike SVM that contains a single line for the decision boundary, there is a gradient going from one class to the other representing the fact that the classifier is probabilistic.

The original implementation of the RVM algorithm, which was provided by Tipping himself,[19] requires one to store and invert a dense $(N+1) \times (N+1)$ matrices, i.e., memory and CPU time requirements (for this realization of the algorithm) scale quadratically and cubically with $(N + 1)$, respectively. With current conventional computer power, the algorithm can become less practical for large (say $N \gg 10{,}000$) data sets. In order to address this issue, Tipping proposed a new approach that he called "fast algorithm-based on marginal likelihood maximization".[20] In this work, we employed our own realization of this new algorithm in C++ code.

A detailed yet accessible description and motivation of the RVM method can be found in refs 15 and 20.

There are numerous metrics that one can employ to evaluate the generalization ability of the classifiers generated via application of the RVM method. In this work, we employ Precision (P), Recall (R), and Fmeasure which are calculated according to

$$P = \frac{t_p}{(t_p + f_p)};\ R = \frac{t_p}{(t_p + f_n)};\ Fmeasure = \frac{2 \times R \times P}{R + P} \qquad (5)$$

where t_p is the number of true positives; f_p is number of false positives; and f_n is the number of false negatives. The Recall, Precision, and Fmeasure metrics were chosen because our data is biased toward the biologically inactive molecules class.

■ RESULTS AND DISCUSSIONS

Data Set and Attributes. RVM binary classifiers were tested on classifying a bioactivity data set that was taken from the MDL Drug Data Report (MDDR) database.[21] The data set consisted of 8293 compounds and 11 activity classes reported in a number of classification studies.[10,22,23] The 8293 compounds, 11 activity classes, and their MDDR codes are summarized in Table 1. Columns 1 and 3 of the table show the activity class and how many compounds (out of the 8293) are active against the given target. Thus, the classification problem we would like to solve was the one−against−all type two-category classification problem.

Table 1. 8293 Structures to Classify According to Their Biological Activity

activity class	MDDR activity code	no. of active compounds
5HT3 antagonist	06233	752
5HT1A agonists	06235	827
5HT reuptake inhibitors	06245	359
D2 antagonist	07701	395
renin inhibitors	31420	1130
angiotensin II AT1 antagonist	31432	943
thrombin inhibitors	37110	803
substance P antagonist	42731	1246
HIV protease inhibitors	71523	750
cyclooxygenase inhibitors	78331	636
protein kinase C inhibitors	78374	452

In this work, the relevant properties of the compounds were represented as MOLPRINT 2D[8] descriptors. MOLPRINT 2D descriptors are circular fingerprints where each atomic environment is described by the Sybyl mol2 atom types of the central atom and of all neighbors up to a path length of two bonds away. The features are then ranked by information gain and the top 40 are selected. The employed kernel was the Aitchison−Aitken (AA)[24] kernel function, defined over the discrete descriptor space and given as

$$K(\mathbf{x}_i, \mathbf{x}; \lambda) = \lambda^{l-r(\mathbf{x}_i - \mathbf{x})}(1-\lambda)^{r(\mathbf{x}_i - \mathbf{x})} \qquad (6)$$

where $\mathbf{x}$ and $\mathbf{x}_i$ are as defined before; and λ is a smoothing variable ($0.5 < \lambda \leq 1$). In the case of $\lambda = 1$, there is no smoothing. $r(\mathbf{x}_i - \mathbf{x}) = (\mathbf{x}_i - \mathbf{x})^T(\mathbf{x}_i - \mathbf{x})$ is the number of disagreements in corresponding components of $\mathbf{x}_i$ and $\mathbf{x}$.

In this work, l was set to 40, and the class labels were $\omega_1 = 1$ and $\omega_2 = 0$ for the active and inactive activity classes, respectively.

It is worth noting that in our input space $\mathcal{M}$, it was found empirically that all the employed kernel matrices (with components $\{K(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1}^{N}$) were positive definite. Meeting this condition was important only in the SVM case because in the RVM method the kernel $K(\mathbf{x}_i, \mathbf{x})$ does not have to be positive definite.[15] This flexibility is another crucial advantage that the RVM has over SVM.

In this work, we just used the estimated probability via $p(\omega_1|\mathbf{x})$ as a score, such that (see eq 3)

$$\mathbf{x}\ \textit{is assigned to} \begin{cases} \omega_1 & \text{if } p(\omega_1|\boldsymbol{\theta}) > 0.5 \\ \omega_2 & \text{if } p(\omega_1|\boldsymbol{\theta}) \leq 0.5 \end{cases} \qquad (7)$$

In each activity class, the data set was randomly partitioned into training, validation, and test sets. On the basis of the work of Kearns,[25] 70%, 10%, and 20% of the sample data set were allotted to become the training, internal validation, and external test data sets, respectively. In each activity class, the data set is biased, in the worst case an imbalance on the order of about 1 to 22 toward the inactive compounds. Instead of "artificially" addressing the apparent class imbalance problem by oversampling and undersampling the active and inactive classes, respectively, the class imbalance was maintained. In this regard, a data set stratification was used: 70% of the inactive molecules and

Table 2. Precision, Recall, and Fmeasure reported on the different activity classes using RVM with AA Kernel [a]

data set	λ	Precision	Recall	Fmeasure	no. of RV	time(s)
06235	0.590	0.946	0.915	0.930	56	0.080
06245	0.570	0.876	0.794	0.833	52	0.080
07701	0.550	0.881	0.814	0.846	61	0.090
31432	0.582	0.978	0.957	0.968	45	0.080
42731	0.600	0.949	0.957	0.953	68	0.090
71523	0.550	0.933	0.929	0.931	51	0.080
31420	0.570	0.982	0.991	0.987	31	0.070
06233	0.600	0.947	0.880	0.912	54	0.080
78331	0.570	0.934	0.816	0.871	84	0.100
78374	0.600	0.911	0.911	0.911	44	0.080
37110	0.590	0.969	0.917	0.942	67	0.090

[a] The number of Relevance Vectors used to predict is shown in column 6.

Table 3. Precision, Recall, and Fmeasure reported on the different activity classes using SVM with AA Kernel [a]

data set	λ	C	ε	Precision	Recall	Fmeasure	no. of SV	time(s)
06235	0.7	64.000	1.0	0.958	0.919	0.938	222	0.141
06245	0.8	32.000	1.0	0.864	0.888	0.876	251	0.161
07701	0.7	168.897	1.0	0.955	0.898	0.926	274	0.171
31432	0.8	97.006	1.0	0.933	0.982	0.957	328	0.211
42731	0.8	42.224	1.0	0.932	0.957	0.944	460	0.271
71523	0.7	222.861	1.0	0.924	0.969	0.946	363	0.221
31420	0.9	73.517	1.0	0.977	0.979	0.978	171	0.131
06233	0.8	32.000	1.0	0.921	0.929	0.925	303	0.181
78331	0.7	147.033	1.0	0.953	0.858	0.903	438	0.232
78374	0.9	256.000	1.0	0.915	0.882	0.898	205	0.131
37110	0.9	294.067	1.0	0.941	0.925	0.933	298	0.191

[a] The number of Support Vectors used to predict is shown in column 8.

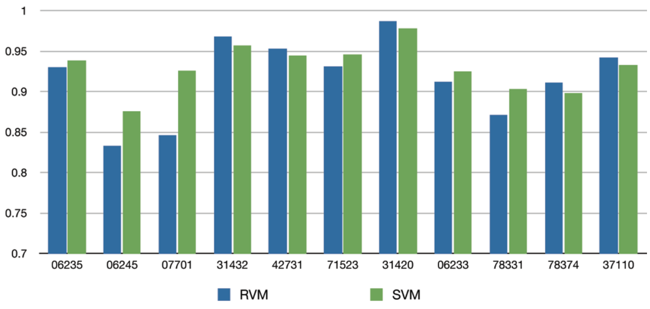


Figure 2. Plot showing the Fmeasure for RVM and SVM across the different activity classes.

70% of the active molecules were chosen as the training set, and 10% of the inactive molecules and 10% of the active molecules were selected as the validation set. The test set consisted of the remaining 20% of the inactive molecules and 20% of the active molecules. For each activity class, a random integer number generator was employed to yield a random index for each entry of the sample data. This allowed us to use a multiple random three-way sample splits scheme.[26]

RESULTS

The Fmeasure metric and the validation set were used to "optimally" tune the only adjustable parameter (λ) in the chosen kernel to construct binary classifiers based on the RVM algorithms. Although for each activity class, we have not exhaustively searched for the "optimal" value of λ, which by definition lies between 0.5 and 1.0; the values shown in Table 2 were deemed to be "optimal" as we employed a substantial grid search over the range [0.5,1.0].

The classification performances of the generated binary classifiers on the test sets are shown in columns 3−5 of Table 2.

For comparison, the SVM-*Light* Package[27] was used. Barring the so-called control parameter C and the slack variable ε, the adjustable parameters in SVM-*Light* were set to their default values. Because the package does not include the AA kernel, we used our own realization of the kernel. To generate a SVM-*Light* binary classifier, it was therefore necessary to optimize three parameters: λ, C, and ε. Again the Fmeasure metric and the validation set were used to tune these three parameters. The "optimal" values obtained are shown in columns 2−4 of Table 3.

Columns 5−7 of Table 3 illustrate the classification performances of the generated SVM-*Light* classifiers on the 11 test sets on which the RVM classifiers were tested.

As the Fmeasure metric is the harmonic mean of the Recall and Precision metrics, in the following discussion, attention is confined to the Fmeasure values. The arithmetic mean of the Fmeasure values for RVM and SVM-*Light* classifiers across the 11 activity classes are 0.916 and 0.929, respectively.

Because the two sets of the Fmeasure values were clearly paired and we could not assume a Gaussian distribution for the Fmeasure values, a nonparametric test was used. At the 0.05 significance level, a Wilcoxon signed rank test performed on the two sets of values for the 11 different activity classes gave a *p*-value of 0.298. We concluded that the prediction powers of classifiers based on the two approaches, RVM and SVM, were not statistically significantly different, and this is illustrated in Figure 2.

However, there is a notable difference between the two methods. In the RVM case, the number of training data points, the Relevance Vectors, that are required to estimate the binary classifiers ranged between 31 and 84. In comparison, the number of training data points or in the parlance of the machine learning community the number of Support Vectors, ranged from 171 to 460. The number of Relevance Vectors and their corresponding number of Support Vectors are shown in columns 6 and 8 of Tables 2 and 3, respectively. Generally, for a given activity class, the number of Support Vectors was about five times larger than its corresponding number of Relevance Vectors. This was a significant (about 80%) improvement of RVM over SVM with respect to their memory requirements.

It is all-important to note that theoretically the generalization performance of the RVM classifiers does not degrade as the number of Relevance Vectors becomes significantly large.[4] Contrast this with the SVM approach, where the classification performance is expected to degrade as the number of Support Vectors gets large (see sction 5.10 in ref 28).

The time shown in columns 7 and 8 of Tables 2 and 3, respectively, is the CPU time used to predict the test set using the generated binary classifiers via application of the RVM and SVM algorithms. The CPU time was calculated by running the prediction 10000 times, and the reported values are the mean times.

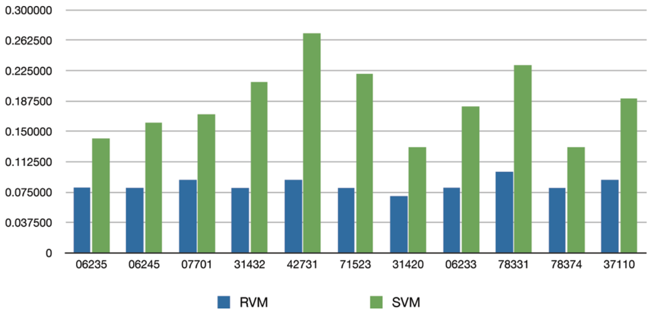


Figure 3. Plot of the CPU time (in seconds) of the runs for SVM compared to RVM.

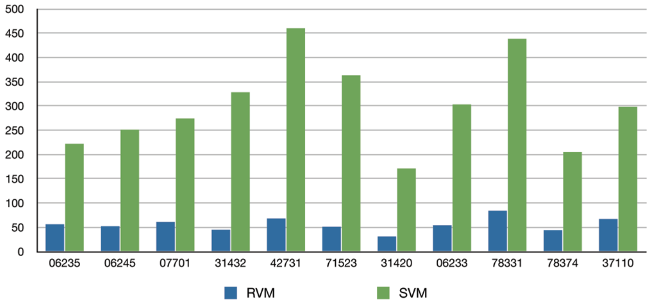


Figure 4. Plot of the number of vectors used in the prediction model for RVM and SVM.

The inbuilt Linux function *time* was used for the timing. Figure 3 is a visual representation of the times for the 11 different test sets. Evidently, RVM consistently outperforms SVM in terms of speed, as one would expect. As is shown in comparison to Figure 4, the more vectors a model contains, the larger the amount of time required to make a prediction. This is true for all test sets except activity class 06233 for SVM in which a shorter time is used than in activity class 37110. This could be explained as the number of Support Vectors (303 and 298) is very close, and unlike our implementation of RVM, SVM-*Light* uses a file type that ignores 0 valued descriptors.

The RVM algorithm was written in C++ and is based on the Matlab version supplied by Tipping.[19] The program utilized the Accelerate Framework that provides optimized versions of the LAPACK and BLAS[29] routines required. Both SVM and RVM were run on a Mac with 2.4 GHz Intel Core 2 Duo processor and 4 GB 667 MHz DDR2 SDRAM.

CONCLUSION

In this paper, we have demonstrated that the Relevance Vector Machine can be used for classifying a large chemoinformatics data set. In line with data analysis results obtained in other disciplines, such as engineering, economics, etc., the results presented indicate that the generalization abilities of the generated RVM and SVM classifiers are similar. The RVM classifier, however, systematically required substantially fewer training data points, that is, less requirement for computation time and memory than its corresponding SVM classifier. We noted that, in contrast to SVM, RVM yields probabilistic classifiers, and that it is not important for the kernel functions to be positive definite. Thus, one may, albeit cautiously, conclude that the RVM approach can become a valuable addition to the classification toolkit employed by the chemoinformatics community.

ASSOCIATED CONTENT

Supporting Information. The MDDR ID's for our data set and the C++ code used for RVM binary classification, and a description on how to obtain the code to calculate MOLPRINT 2D descriptors. This information is available free of charge via the Internet at http://pubs.acs.org/.

AUTHOR INFORMATION

Corresponding Authors

*E-mail: ral64@cam.ac.uk (R.L.); hym21@cam.ac.uk (H.Y.M).

ACKNOWLEDGMENT

H.Y.M. acknowledges the Unilever Centre for their support. R.C.G. thanks Unilever for financial support. R.L. and J.B.O.M. thank the EPSRC (Grant EP/F049102/1) and the Scottish Universities Life Sciences Alliance (SULSA) for funding.

REFERENCES

(1) Webb, A. R. *Statistical Pattern Recognition*, 2nd ed.; John Wiley & Sons, Ltd: United Kingdom, 2002.

(2) Duda, R. O.; Hart, P. E. *Pattern Classification and Scene Analysis*, 1st ed.; John Wiley & Sons, Ltd: New York, 1973.

(3) Vapnik, V. N. In *The Nature of Statistical Learning Theory*, 1st ed.; Jordan, M., Lauritzen, S. L., Eds.; Springer: New York, 1995.

(4) Bishop, C. M. *Pattern Recognition and Machine Learning*, 1st ed.; Springer: New York, 2006.

(5) Ripley, B. *Pattern Recognition and Neural Networks*, 1st ed.; Cambridge University Press: United Kingdom, 1996.

(6) Fukunaga, K. *Introduction to Statistical Pattern Recognition*, 2nd ed.; Academic Press: San Diego, CA, 1990.

(7) Mitchell, T. M. *Machine Learning*, 1st ed.; McGraw-Hill International Editions: New York, 1997.

(8) Bender, A.; Mussa, H. Y.; Glen, R. C.; Reiling, S. Molecular similarity searching using atom environments, information-based feature selection, and a Naïve Bayesian classifier. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 170–178.

(9) Harper, G.; Bradshaw, J.; Gittins, J. C.; Green, D. V. S.; Leach, A. R. Prediction of biological activity for high-throughput screening using binary kernel discrimination. *J. Chem. Inf. Comput. Sci.* **2001**, *41*, 1295–1300.

(10) Nigsch, F.; Mitchell, J. B. O. How to winnow actives from inactives: Introducing Molecular Orthogonal Sparse Bigrams (MOSBs) and multiclass winnow. *J. Chem. Inf. Model.* **2008**, *48*, 306–318.

(11) Mussa, H. Y.; Hawizy, L.; Nigsch, F.; Glen, R. C. Classifying large chemical data sets: Using a regularized potential function method. *J. Chem. Inf. Model.* **2011**, *51*, 4–14.

(12) Chen, B.; Harrison, R. F.; Pasupa, K.; Willett, P.; Wilton, D. J.; Wood, D. J.; Lewell, X. Q. Virtual screening using binary kernel discrimination: Effect of noisy training data and the optimization of performance. *J. Chem. Inf. Model.* **2006**, *46*, 478—486, PMID: 16562975.

(13) Bouckaert, R. Naive Bayes Classifiers That Perform Well with Continuous Variables. In *AI 2004: Advances in Artificial Intelligence*; Webb, G., Yu, X., Eds.; Springer Berlin/Heidelberg: Germany, 2005; Vol. 3339, pp 85—116.

(14) Parzen, E. On estimation of a probability density function and mode. *Ann. Math. Statist.* **1962**, *33*, 1065–1076.

(15) Tipping, M. E. Sparse Bayesian learning and the relevance vector machine. *J. Mach. Learn. Res.* **2001**, *1*, 211–244.

(16) Platt, J. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *AI 2004: Advances in Large Margin Classifiers*; Smola, A. J., Bartlett, P., Scholkopf, B., Schuurman, D., Eds.; MIT Press: Cambridge, MA, 2000; pp 1−11.

(17) Neal, R. M. *Bayesian Learning for Neural Networks*, 1st ed.; Springer: New York, 1996.

(18) Muller, P.; Insua, D. R. Issues in Bayesian analysis of neural network models. *Neural. Comput.* **1998**, *10*, 749–770.

(19) Sparse Bayesian Models (& the RVM). http://www.miketipping.com/index.php?page=rvm (accessed June 1, 2011).

(20) Tipping, M. E.; Faul, A. Fast Marginal Likelihood Maximisation for Sparse Bayesian Models. In Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics, Key West, FL, January 3−6, 2003; Bishop, C. M., Frey, B. J., Eds.

(21) Accelrys. http://accelrys.com/products/databases/bioactivity/mddr.html (accessed June 1, 2011).

(22) Wilton, D. J.; Harrison, R. F.; Willett, P.; Delaney, J.; Lawson, K.; Mullier, G. Virtual screening using binary kernel discrimination: Analysis of pesticide data. *J. Chem. Inf. Model.* **2006**, *46*, 471–477.

(23) Hert, J.; Willett, P.; Wilton, D. J.; Acklin, P.; Azzaoui, K.; Jacoby, E.; Schuffenhauer, A. Comparison of topological descriptors for similarity-based virtual screening using multiple bioactive reference structures. *Org. Biomol. Chem.* **2004**, *2*, 3256–3266.

(24) Aitchison, J.; Aitken, C. G. G. Multivariate binary discrimination by the kernel method. *Biometrika* **1976**, *63*, 413–420.

(25) Kearns, M. A bound on the error of cross validation using the approximation and estimation rates, with consequences for the training-test split. *Neural. Comput.* **1997**, *9*, 1143–1161.

(26) Haykin, S. *Neural Networks: A Comprehensive Foundation*, 2nd ed.; Prentice Hall: Upper Saddle River, New Jersey , 1998.

(27) Joachims, T. In *Advances in Kernel Methods: Support Vector Learning*; Schölkopf, B., Burges, C., Smola, A., Eds.; MIT Press: Cambridge, MA, 1999; Chapter 11, pp 169−184.

(28) Theodoridis, S.; Koutroumbas, K. *Pattern Recognition*, 4th ed.; Academic Press: Burlington, MA, 2009.

(29) Anderson, E.; Bai, Z.; Bischof, C.; Blackford, S.; Demmel, J.; Dongarra, J.; Du Croz, J.; Greenbaum, A.; Hammarling, S.; McKenney, A.; Sorensen, D. *LAPACK Users' Guide*, 3rd ed.; Society for Industrial and Applied Mathematics: Philadelphia, PA, 1999.