

Teaching machine learning from examples

Isabelle Guyon
ETH Zürich, Switzerland
isabelle@clopinnet.com

Jiwen Li
University of Zürich, Switzerland
li@ifi.unizh.ch

**Theodor Mader, Patrick A. Pletscher,
Georg Schneider, Markus Uhr**
ETH Zürich, Switzerland
{*tmader,patrickp,scgeorg,uhrm*}@student.ethz.ch

June, 2006

Abstract

We used the datasets of the NIPS 2003 challenge on feature selection as part of the practical work of an undergraduate course on feature extraction. The students were provided with a toolkit implemented in Matlab. Part of the course requirements was that they should outperform given baseline methods. The results were beyond expectations: the student matched or exceeded the performance of the best challenge entries and achieved very effective feature selection with simple methods. We make available to the community the results of this experiment and the corresponding teaching material [16].

1 Introduction

In the recent years, it has been recognized by the machine learning and neural network communities that competitions are key to stimulate research and bring improvement. Several large conferences are now regularly organizing competitions. For NIPS 2003, a competition on the theme of feature selection, which attracted 75 participants, was organized [14]. The outcomes of that effort were compiled in a book including tutorial chapters and papers from the proceedings of that workshop [13]. The website of the challenge remains open for post-challenge submissions [1]. Meanwhile, another challenge on the theme of model selection has taken place [2], in which the participants were provided with a Matlab toolkit based on the Spider package [17]. All this material constitute a great teaching resource that we have exploited in a course on feature extraction [16]. We are reporting on our teaching experience with two intentions:

- encouraging other teachers to use challenge platforms in their curricula, and
- providing to graduate students simple competitive baseline methods to attack problems in machine learning.

The particular theme of the class is feature extraction, which we define as the combination of feature construction and feature selection. In the past few years,

feature extraction and space dimensionality reduction problems have drawn a lot of interest. More than a passing fancy, this trend in the research community is driven by applications: bioinformatics, chemistry (drug design, cheminformatics), text processing, pattern recognition, speech processing, and machine vision provide machine learning problems in very high dimensional spaces, but often with comparably few examples. A lot of attention was given in class to feature selection because many successful applications of machine learning have been built upon a large number of very low level features (e.g. the “bag-of-word” representation in text processing, gene expression coefficients in cancer diagnosis, and QSAR features in cheminformatics). Our teaching strategy is to make students gain hands on experience by working on large real world datasets (those of the NIPS 2003 challenge [10]), rather than providing them with toy problems.

2 Datasets and synopsis of the challenge

The NIPS 2003 challenge included five datasets (Table 1) from various application domains. All datasets are two-class classification problems. The data were split into three subsets: a training set, a validation set, and a test set. All three subsets were made available at the beginning of the challenge. The class labels for the validation set and the test set were withheld. The challenge participants could submit prediction results on the validation set and get their performance results and ranking on-line during a development period. The validation set labels were then revealed and the participants could make submissions of test set predictions, after having trained on both the training and the validation set. For details on the benchmark design, see [14].

The identity of the datasets and of the features (some of which were random features artificially generated) were kept secret during the challenge, but have been revealed since then. The datasets were chosen to span a variety of domains and difficulties (the input variables are continuous or binary, sparse or dense; one dataset has unbalanced classes.) One dataset (MADELON) was artificially constructed to illustrate a particular difficulty: selecting a feature set when no feature is informative by itself. To facilitate the assessment of feature selection methods, a number of artificial features called **probes** were drawn at random from a distribution resembling that of the real features, but carrying no information about the class labels. A good feature selection algorithm should eliminate most of the probes. The details of data preparation can be found in a technical memorandum [10].

The distribution of the results of the challenge participants for the various datasets are represented in Figure 1. Using these graphs, we introduced in class the datasets in order of task difficulty. We provided the students with baseline methods approximately in the best tenth percentile of the challenge entries. For the class, the students had access to the training and validation set labels, but not the test labels. The test set labels have not been released to the public to keep an on-going benchmark. The students could thus make post-challenge submissions to the web site of the challenge [1] to obtain their performance on the test set. We asked students to try to outperform the baseline method and gave them extra credit for outperforming the best challenge entry (see Section 4).

3 Learning object package

The machine learning package we used for the class called CLOP (Challenge Learning Object Package) is available from the website of the “performance prediction

Table 1: **NIPS 2003 challenge datasets.** For each dataset we show its domain of application, its type T (d=dense, s=sparse, or sb=sparse binary), the total number of features N_{feat} , the fraction of probes in the original feature set F_{probe}^+ , the number of examples in the training, validation, and test sets, the fraction of examples in the positive class $\%_{[+]}$, and the ratio number of training examples to number of features Tr/\mathbf{F} . All problems are two-class classification problems.

Dataset	Domain	T	N_{feat}	F_{probe}^+	#Tr	#Va	#Te	$\%_{[+]}$	Tr/F
ARCENE	Mass Spectrometry	d	10^4	0.30	100	100	700	44	0.01
DEXTER	Text classification	s	$2 \cdot 10^4$	0.50	300	300	2000	50	0.015
DOROTHEA	Drug discovery	sb	10^5	0.50	800	350	800	10	0.008
GISETTE	Digit recognition	d	5000	0.50	6000	1000	6500	50	1.2
MADOLON	Artificial	d	500	0.96	2000	600	1800	50	4

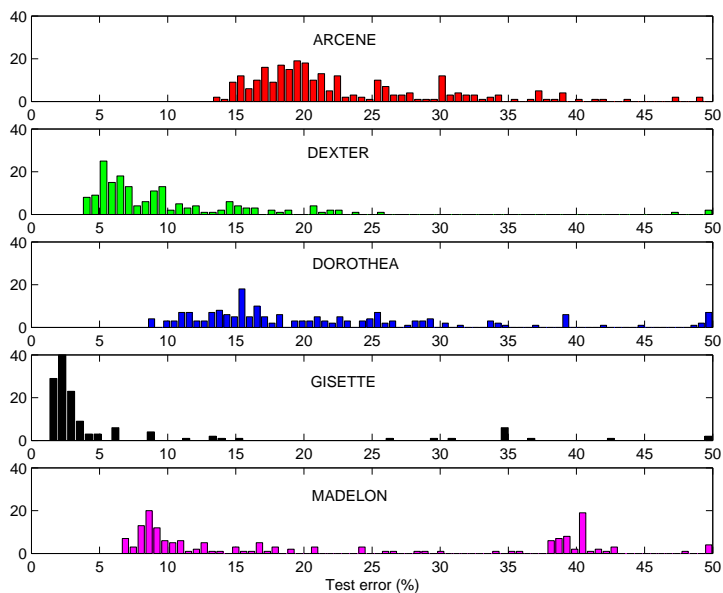


Figure 1: **Distribution of the challenge participant results.** We show histograms of the balanced error rate (BER) for the five tasks.

challenge” [2]. For an introduction, see the QuickStart guide [3]. We present in this section a high level overview of the package.

Data and algorithm objects

The Spider package [17] on top of which CLOP is built, uses Matlab[®] objects. Two simple abstractions are used:

- **data:** Data objects include two members X and Y, X being the input matrix (patterns in lines and features in columns), Y being the target matrix (i.e. one column of \pm for binary classification problems).
- **algorithms:** Algorithm objects represent learning machines (e.g. neural networks, kernel methods, decision trees) or preprocessors (for feature con-

struction, data normalization or feature selection). They are constructed from a set of hyper-parameters and have at least two methods: train and test. The train method adjusts the parameters of the model. The test method processes data using a trained model.

Trained models are simply saved and reloaded with the `save/load` Matlab commands. This feature is convenient to verify results and enforce reproducibility.

The Spider (with some CLOP extensions) provides ways of building more complex “compound” models from the basic algorithms with two abstractions: **chains** and **ensembles**. Chains combine models serially while ensembles combine them in parallel. The Spider provides several objects for cross-validation and other model selection methods.

Learning objects provided for the class

It is easy to be overwhelmed when starting to use a machine learning package. CLOP is an extremely simplified package, limited to a few key methods, each having just a few hyper-parameter values with good default values. This makes it suitable for teaching a machine learning class. More advanced students can venture to using other methods provided in the Spider package, on top of which CLOP is built.

The CLOP modules correspond to methods having performed well in the feature selection challenge [14]. There are five classifier objects (we reference the implementations used): Kernel ridge regression [11], naive Bayes [12], neural networks [4], Random Forest [6], and Support Vector Machines [7]. We limited the number of hyperparameters of the algorithms to simplify model selection. For instance, the kernel methods use a single kernel: $k(\mathbf{x}, \mathbf{x}') = (coef_0 + \mathbf{x} \cdot \mathbf{x}')^d \exp(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2)$ with three hyperparameters $coef_0$, d , and γ . The CLOP modules also include five preprocessors performing standard normalization and feature extraction steps, and five feature selection methods.

We gave to the students other examples of learning objects which can be found in the homework instructions [16]. These include methods of feature extraction for image data (e.g. Fourier transform, and two-dimensional convolution), and statistical tests for feature selection. Details are found in a longer technical report [8].

We provided the students with baseline methods, which are compound models combining the basic learning objects with “chains” and “ensembles” (see [8] for details). We chose the simplest possible baseline methods, which attain performances approximately in the best tenth percentile of the challenge entries (Table 2).

4 Performance assessment and class requisites

As part of the class requirements, the student had to submit results on all five datasets of the challenge and include their corresponding CLOP models. Performances were assessed using several metrics:

- *BER*: The balanced error rate, that is the average of the error rate of the positive class and the error rate of the negative class. This metric was used rather than the simple error rate because some datasets (particularly DOROTHEA) are unbalanced in class cardinality. Unless otherwise stated, the *BER* is computed with test examples.
- *F_{feat}*: The fraction of features selected.

Table 2: **Performance comparison.** The table shows the balanced error rate and the corresponding fraction of features used for the baseline method, the best challenge entry, and the best student method. The students earned one point for $BER < BER^0$ or $\{BER = BER^0 \text{ and } F_{feat} < F_{feat}^0\}$. They earned 2 points for $BER < BER^* + \delta BER$. We also show the training time of the best student models on a 1.5GHz Pentium.

Dataset	Baseline		Challenge best		Student best		
	BER^0	F_{feat}^0	$BER^* \pm \delta BER$	F_{feat}^*	BER	F_{feat}	Train. time (s)
ARCENE	0.1470	0.11	0.1073 ± 0.0117	1.00	0.1048	0.14	3.8
DEXTER	0.0500	0.02	0.0330 ± 0.0040	0.19	0.0325	0.23	1.2
DOROTHEA	0.1237	0.01	0.0854 ± 0.0099	1.00	0.0930	0.01	0.7
GISETTE	0.0180	0.20	0.0126 ± 0.0014	1.00	0.0111	0.20	11.2
GISETTE (pixels)	0.0106	1.00	NA	NA	0.0078	1.00	127.9
MADÉLON	0.0733	0.04	0.0622 ± 0.0057	1.00	0.0622	0.04	7.8

- F_{probe} : The fraction of probes found in the feature set selected.¹ By normalizing F_{probe} by F_{probe}^+ , the fraction of probes in the whole feature set, we obtain an estimate of the False Discovery Rate (FDR). The FDR is indicative of Type I errors (irrelevant features wrongly selected).
- Z_{BER} : To assess the effectiveness of feature selection, we can compare a reduced model built with the selected features with the full model built using all the features. A large balanced error rate of the reduced model BER , compared to BER^+ obtained using all the features, is indicative of Type II errors (false negative, i.e. relevant features wrongly discarded). To assess the incidence of Type II errors we use the statistic $Z_{BER} = (BER - BER^+) / \sigma_{\Delta}$, where σ_{Δ} is the standard deviation of $BER - BER^+$.²

For each dataset, the students would earn one point if they obtained a better BER than the provided baseline method, or a smaller F_{feat} for the same BER . They would earn two points if they matched the BER of the best challenge entry (within the statistical error bar). The overall grade also included points for a poster presentation of their results and for the presentation in class of a research paper from the book [13]. The other performance metrics (F_{probe} and Z_{BER}) are used in our analysis (Section 5), but were not used for grading.

5 Student work

During the curriculum, the datasets were introduced one at a time to illustrate topics addressed in class, based on progressive difficulty. A description of the homework assignments associated with each dataset is provided in [8]. Briefly: GISETTE (the handwritten digit dataset) was used to illustrate feature construction. The student could experiment with transforms such as Fourier transform and convolution using the original pixel representation and create their own feature extraction

¹We remind the reader that “probes” are meaningless features purposely added to the feature set to assess the effectiveness of feature selection.

²For small training sets, the benefit of reducing the dimensionality may outweigh the loss of information by discarding useful features, making Z_{BER} an imperfect indicator of Type II errors. The BER may actually be better than BER^+ , in which case Z_{BER} will be negative.

Table 3: Methods employed by the best student entries.

Dataset	Code of the methods employed
ARCENE	<code>my_svc=svc({'coef0=2','degree=3','gamma=0','shrinkage=0.1'}); my_model=chain({'relief('f_max=1400'),'normalize,my_svc'})</code>
DEXTER	<code>my_svc=svc({'coef0=1','degree=1','gamma=0','shrinkage=0.5'}); my_model=chain({'s2n('f_max=4500'),'normalize,my_svc'})</code>
DOROTHEA	<code>my_model=chain({'TP(f_max=15000),normalize, ... relief(f_max=700'),naive,bias'})</code>
GISETTE	<code>my_svc=svc({'coef0=1','degree=3','gamma=0','shrinkage=1'}); my_model=chain({'s2n('f_max=1000'),'normalize,my_svc'})</code>
GISETTE (pixels)	<code>my_svc=svc({'coef0=1','degree=4','gamma=0','shrinkage=0.1'}); my_model=chain({'convolve(gauss_ker({'dim1=5','dim2=5'})), ... normalize,my_svc'})</code>
MADOLON	<code>my_svc=svc({'coef0=1','degree=0','gamma=0.3','shrinkage=0.3'}); my_model=chain({'relief('f_max=20'),'standardize,my_svc'})</code>

learning objects. DEXTER (the text classification dataset) was used to learn about univariate filter methods and how to optimize the number of features with statistical tests or cross-validation. MADOLON (the artificial dataset) gave an opportunity to the students to experiment with multivariate filters such as Relief, or try wrappers or embedded methods with non-linear multivariate classifiers. ARCENE (the mass-spectrometry dataset) included two difficulties: a small training set and heterogeneous data (coming from 3 sources). Therefore, it was a good dataset to further study multivariate feature selection algorithms, and experiment with mixtures of experts or ensemble methods. DOROTHEA (the drug discovery dataset) is the hardest because of its 100,000 features for only 800 training examples, with only ten percent of positive examples. The students learned about problems of overfitting and unbalanced data.

The best student entries are shown in Table 3. The training set and validation set of the challenge were merged for training. The hyper-parameters were adjusted by cross-validation (usually 5-fold cross-validation). The final model is trained on all the available labeled data. As indicated in Table 2, the training times are modest (of the order of 1-10 seconds per model, except for GISETTE when the pixel representation is used, because of a time consuming preprocessing). Most students matched the performances of the best challenge entries (within the statistical error bar) and therefore earned the maximum number of points (Table 2 and Figure 2). Some student entries even exceeded the performance of the best challenge entries. We think that the achievements of the students are remarkable. They exceeded our expectations. Of course, it can be argued that they had several advantages over the competitors: knowledge of the application domain, of the nature of the features and fraction of probes, and access to test performance. However, this does not diminish significantly their achievement since many other post-challenge entries were made since the end of the challenge and hardly any matched or outperformed the best results of the challengers.³

³We noted that, except in the case of GISETTE, where they were asked to take advantage of the knowledge of the features to outperform the challengers, the students did not take advantage of the knowledge of the nature of the features. Since they did not have access to the test labels, they could only submit results to the web site to get feed-back on their performance, which limited the possibility of tuning their method to the test set.

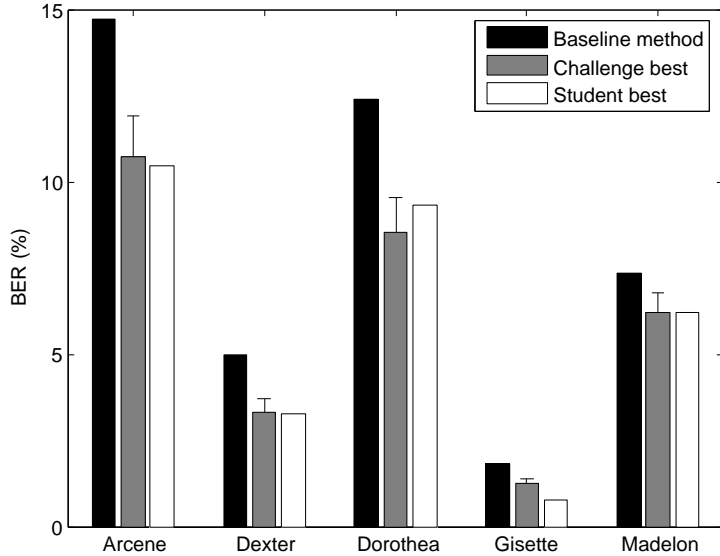


Figure 2: **Result comparison.** The student performances matched (within the statistical error bar) or exceeded the performances of the best challengers, using comparable or smaller feature set sizes (see table 2)

General observations can be made about the best models. All of them use simple classifiers linear in their parameters (`naive` = naïve Bayes [12] and `svc` = Support Vector Classifier [5]). Both the naïve Bayes and SVC are known to be robust against overfitting. Feature selection is performed with only three simple filters: `TP`, a filter useful for highly unbalanced classes like DOROTHEA to pre-select “true positive” features [18], `s2n`, a simple univariate filter, which ranks features according to the “signal-to-noise” (`s2n`) ratio [9], and `relief` a filter which ranks features according to their separating power “in the context of other features” [15]. The strategy adopted is to reduce space dimensionality while retaining as many relevant features as possible, with no attempt to remove feature redundancy. Additional regularization is obtained by tuning the classifier “shrinkage” hyperparameters.

By analyzing the results of the student work, stronger conclusions can be drawn about the effectiveness of feature selection than by analyzing the results of the challenge itself, because for four datasets out of five the best challenge entries used 100% of the features. In contrast, the student entries use a small subset of features, yet outperform the best challenge entries or match their performances within the statistical error bar.

To further evaluate the impact of feature selection, we reran the best models without feature selection, keeping the same hyper-parameters. We summarize the results in Table 4. In all cases, the reduced model (built with the selected features) has a smaller test BER than the full model (built with all the features) and consequently Z_{BER} is negative: There is no significant loss of information by discarding useful features and/or the benefit of space dimensionality reduction outweighs that loss. For the three last datasets, the reduced model is significantly better than the full model at the 1% risk level ($Z_{BER} < -2.33$) and the False Discovery Rate (FDR) is zero or near zero. This means that we have both very few type I and type II errors (few irrelevant features wrongly selected and few useful features discarded). At the 5% risk level ($Z_{BER} < -1.65$), feature selection yields significant performance

Table 4: **Effectiveness of feature selection.** The table shows the fraction of probes in the original data F_{probe}^+ and the BER of the full model BER^+ , the fraction of selected features F_{feat} , the fraction of probes in the selected feature sets F_{probe} , and the BER of the model built with the reduced feature set BER , the false discovery rate $FDR \simeq F_{probe}/F_{probe}^+$, and the Z statistic $Z_{BER} = (BER - BER^+)/\sigma_{\Delta}$.

Dataset	Full model		Reduced model			FDR	Z_{BER}
	F_{probe}^+	BER^+	F_{feat}	F_{probe}	BER		
ARCENE	0.30	0.1186	0.14	0.04	0.1048	0.14	-1.09
DEXTER	0.50	0.0410	0.23	0.55	0.0325	1.1	-1.75
DOROTHEA	0.50	0.3094	0.01	0.03	0.0930	0.05	-5.82
GISETTE	0.50	0.0180	0.20	0.00	0.0111	0	-3.49
MADELON	0.96	0.4872	0.04	0.00	0.0622	0	-35.53

improvement for one more dataset (DEXTER), even though the FDR of the selected features is very high. This shows that space dimensionality reduction plays an important role in performance improvement, regardless of the effectiveness of the method to filter out irrelevant features. For ARCENE, the performances of the reduced model and the full model are not significantly different. Yet, the feature set is significantly reduced without performance degradation and the FDR is relatively low.

Although it is difficult to predict ahead of time which method will work best on a given dataset, in retrospect, some justifications can be given. A simple rule-of-thumb for the choice of a classifier is that its complexity should be proportional to the ratio Tr/F of number of training examples over the number of features. If we sort the classifiers according to that ratio, we obtain: DOROTHEA < ARCENE < DEXTER < GISETTE < MADELON. Accordingly, it makes good sense that the naïve Bayes method (the simplest) was used for DOROTHEA, while the non-linear SVM (the most complex) was used for GISETTE and MADELON. DEXTER in the middle, uses the linear SVM of middle range complexity. It easily understood why **relief** performs well for MADELON and ARCENE: both datasets have classes containing multiple clusters. So we need a filter that performs a “local” feature selection. For the same reason, these two datasets benefit from using a non-linear classifier (particularly MADELON). For DOROTHEA, the problem of class imbalance has been addressed by using a special feature filter (TP) and a special post-processor to adjust the bias. For a more detailed analysis dataset by dataset, see [8].

6 Conclusions and future work

A challenge can be more than a one-time event. It can become an on-going life benchmark and a teaching tool. Leaving the website of the NIPS2003 feature selection challenge open for post-challenge submissions has given to graduate students and researchers the opportunity to compare their algorithms to well established baseline results. Since the end of the challenge, the number of entrants has almost doubled.

In this paper, we have demonstrated that even undergraduate students can get their hands dirty and “learn machine learning from examples”, with a success that exceeded our expectations. All of them easily outperformed the baseline methods we provided them and most of them matched the performances of the best challengers (within the statistical error bar) or even exceeded them. We hope that this

experience will be followed by similar other attempts. In the mean time, we make available all of our teaching material, data and code.

The results obtained mark also a victory of simple methods. All the models used to match or outperform the best challenge entries use a combination of simple normalization, feature selection filters (signal-to-noise ratio, Relief, or fraction of true positive), and a naïve Bayes or a support vector machine classifier. There was no need to use ensemble methods or transduction. However, univariate feature selection and linear classifiers did not always suffice.

With this study, we could reach more conclusive results regarding the effectiveness of feature selection than by analyzing the results of the challenge. The best challenge entries significantly outperformed other entries without using feature selection. In contrast, the student entries used a reduced feature set, while matching or outperforming the performance of the best challenge entries. For three datasets the reduced model using a small fraction of the original feature set significantly outperformed the full model and the false discovery rate approached zero.

This paper by no means marks an end point to the problem of feature selection or even to solving the tasks of the NIPS2003 feature selection challenge. Our explorations indicate that there is still much room for improvement. In particular, since we have released the identity of the features, it is now possible to introduce domain knowledge in the feature construction process. To a limited extent we have seen that this strategy show promises on the GISETTE datasets: a simple smoothing of the pixel image allowed us to boost performances.

Acknowledgments

We are very thankful to the institutions that have contributed data (see [10]), to the developers of the Spider who make their code available, and to the co-developers of the CLOP package Amir Reza Saffari Azar and Gideon Dror. The kind support of Prof. Joachim Buhmann and the encouragements and advices of Peter Orbanz are gratefully acknowledged. The CLOP project is partially supported by the National Science Foundation under Grant N0. ECS-0424142. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

- [1] Feature selection challenge. NIPS 2003. <http://www.nipsfsc.ecs.soton.ac.uk>.
- [2] Performance prediction challenge. WCCI 2006. <http://www.modelselect.inf.ethz.ch>.
- [3] Amir Reza Saffari Azar Alamdari and Isabelle Guyon. Quick start guide for CLOP. Technical report, May 2006. <http://ymer.org/research/files/clop/QuickStartV1.0.pdf>.
- [4] Christopher M. Bishop. *Neural networks for pattern recognition*. Oxford University Press, Oxford, UK, 1996.
- [5] Bernhard Boser, Isabelle Guyon, and Vladimir Vapnik. A training algorithm for optimal margin classifiers. In *Fifth Annual Workshop on Computational Learning Theory*, pages 144–152, Pittsburgh, 1992. ACM.
- [6] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001. <http://citeseer.ist.psu.edu/breiman01random.html>.
- [7] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>.
- [8] Isabelle Guyon, et al. Feature selection with the CLOP package. Technical report, 2006, <http://clonet.com/isabelle/Projects/ETH/TM-fextract-class.pdf>.

- [9] T. R. Golub et al. Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
- [10] Isabelle Guyon. Design of experiments of the NIPS 2003 variable selection benchmark. Technical report, 2003. <http://www.nipsfsc.ecs.soton.ac.uk/papers/Datasets.pdf>.
- [11] Isabelle Guyon. Kernel ridge regression tutorial. Technical report, 2005, <http://clopinnet.com/isabelle/Projects/ETH/KernelRidge.pdf>.
- [12] Isabelle Guyon. Naïve bayes algorithm in CLOP. Technical report, 2005, <http://clopinnet.com/isabelle/Projects/ETH/NaiveBayesAlgorithm.pdf>.
- [13] Isabelle Guyon, Steve Gunn, Masoud Nikravesh, and Lofti Zadeh, Editors. *Feature Extraction, Foundations and Applications*. Studies in Fuzziness and Soft Computing. Physica-Verlag, Springer, 2006.
- [14] Isabelle Guyon, Steve R. Gunn, Asa Ben-Hur, and Gideon Dror. Result analysis of the NIPS 2003 feature selection challenge. In *NIPS*, 2004. http://books.nips.cc/papers/files/nips17/NIPS2004_0194.pdf.
- [15] K. Kira and L. Rendell. A practical approach to feature selection. In D. Sleeman and P. Edwards, editors, *International Conference on Machine Learning*, pages 368–377, Aberdeen, July 1992. Morgan Kaufmann.
- [16] Anonymous reference. Feature extraction course, ETH WS 2005/2006. <http://clopinnet.com/isabelle/Projects/ETH/>.
- [17] J. Weston, A. Elisseeff, G. BakIr, and F. Sinz. The Spider machine learning toolbox. 2005. <http://www.kyb.tuebingen.mpg.de/bs/people/spider/>.
- [18] Jason Weston, Fernando Pérez-Cruz, Olivier Bousquet, Olivier Chapelle, André Elisseeff, and Bernhard Schölkopf. Feature selection and transduction for prediction of molecular bioactivity for drug design. *Bioinformatics*, 19(6):764–771, 2003.