Chapter 23

Mining for Complex Models Comprising Feature Selection and Classification

Krzysztof Grąbczewski and Norbert Jankowski

Department of Informatics, Nicolaus Copernicus University, Toruń, Poland
`kgrabcze@phys.uni.torun.pl, norbert@phys.uni.torun.pl`

Summary. Different classification tasks require different learning schemes to be satisfactorily solved. Most real-world datasets can be modeled only by complex structures resulting from deep data exploration with a number of different classification and data transformation methods. The search through the space of complex structures must be augmented with reliable validation strategies. All these techniques were necessary to build accurate models for the five high-dimensional datasets of the NIPS 2003 Feature Selection Challenge. Several feature selection algorithms (e.g. based on variance, correlation coefficient, decision trees) and several classification schemes (e.g. nearest neighbors, Normalized RBF, Support Vector Machines) were used to build complex models which transform the data and then classify. Committees of feature selection models and ensemble classifiers were also very helpful to construct models of high generalization abilities.

23.1 Introduction

Solving classification problems includes both classifiers' learning and relevant preparation of the training data. In numerous domains the stage of data preprocessing can significantly improve the performance of final classification models. A successful data mining system must be able to combine the two analysis stages and take their interaction into account. Each classification method may need differently prepared inputs to build an accurate and reliable model. Therefore we need to search for a robust combination of methods and their parameters.

Using complex model structures implies the necessity of adequate validation. It is very important to validate the whole sequences of transformations and classifiers instead of performing data preprocessing first and then validating the classification algorithms only. Otherwise we are sentenced to overoptimistic results estimates, which do not reflect real generalization abilities.

To build and validate complex models it is very important to have general data analysis tools, which facilitate combining different algorithms and testing their interaction. In the NIPS 2003 Feature Selection Challenge efforts we have been supported by our object oriented data mining technology of the GhostMiner[1] system. All the algorithms we have used and describe below are components of the package. Recently, we have developed some new functionality of the system to comply with the needs of feature selection, balanced error minimization etc. Thanks to the general, object oriented philosophy of the tool all the components could be easily combined and validated.

It is worth pointing out that all our computations have been run on personal computers including notebooks – thanks to the algorithms no supercomputers or clusters are necessary to obtain interesting results in data mining.

23.2 Fundamental algorithms

There is no single model architecture, which could be recommended for all the possible applications. To solve different classification problems we need different kinds of models and different data transformation techniques. The search for the final combined model must be based on a set of fundamental algorithms, possibly of different inherent structures and methodologies.

23.2.1 Classification

In our exploration of the datasets we have tested a variety of methods, which implement different ways of cost function minimization. This broadens the search area in the model space. Final models for the five datasets were based on Support Vector Machines, Normalized Radial Basis Functions and Nearest Neighbors approaches. Apart from these we have also tested SSV decision tree, IncNet (Jankowski and Kadirkamanathan, 1997) and Feature Space Mapping (Adamczak et al., 1997) classification algorithms. The SSV is presented here because it was useful for building the feature selection parts of the models.

A special treatment was necessary in the case of the Dorothea dataset (and to a lower extent of Arcene), because the minimization of the standard classification error or MSE leads to completely different models than the optimization of the balanced error rate. The latter is in fact a special case of the classification error defined by a cost matrix, but not all algorithms support it.

Due to the space limitations we are unable to present the methods in full detail. Please refer to the bibliography for more information on the algorithms of interest.

Support Vector Machines (SVMs)

We have used Support Vector Machines for several reasons. One of them is that SVMs optimize margins between class regions. Another one is that with different kernel functions the SVM changes from simple linear model to a nonlinear one. Yet another reason is that the SVM model may be implemented really effectively and can deal with high-dimensional data.

[1]GhostMiner is a trademark of FQS Poland

SVMs were proposed initially by Boser et al. (1992) and thoroughly examined by Vapnik (1995, 1998). They are often very accurate and efficient. The idea is applicable to both data classification and function approximation tasks. The statement of the SVM optimization for classification problems may be the following:

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C\sum_{i=1}^{m}\xi_i \tag{23.1}$$

with constraints:

$$y_i(\boldsymbol{w}^T\boldsymbol{\phi}_i + b) \geq 1 - \xi_i \tag{23.2}$$

$$\xi_i \geq 0, \quad i = 1, \ldots, m \tag{23.3}$$

where m is the number of vectors, $\mathbf{x}_i$ is the ith data vector and $\boldsymbol{y}_i$ is its class label (1 or -1 – the binary classification). The dual problem definition is:

$$\min_{\boldsymbol{\alpha}} \quad \frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}\boldsymbol{\alpha} + \mathbf{1}^T\boldsymbol{\alpha} \tag{23.4}$$

with constraints:

$$0\mathbf{1}\alpha_i\mathbf{1}C, \quad i = 1, \ldots, m \tag{23.5}$$

$$\mathbf{y}^T\boldsymbol{\alpha} = 0 \tag{23.6}$$

where $C > 0$ defines the upper bound for α_i factors, and $\boldsymbol{Q}$ is a matrix defined by:

$$Q_{ij} = y_i y_j k(\boldsymbol{x}_i, \boldsymbol{x}_j). \tag{23.7}$$

The $k(\cdot)$ function is called a kernel and $k(\boldsymbol{x}_i, \boldsymbol{x}_j) \equiv \boldsymbol{\phi}_i^T\boldsymbol{\phi}_j$.

Most often used kernels are: gaussian, linear, sigmoidal and polynomial. With the exception of the linear kernel all the others have some parameters of free choice. Although in our framework we have implemented all the kernels, we recommend only the most useful ones: linear, Gaussian and exponential inverse of distance. In the simplest case $k(\cdot)$ is defined by:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \boldsymbol{x}^T\boldsymbol{x}'. \tag{23.8}$$

To add a nonlinear behavior, the Gaussian kernel can be used instead:

$$k(\boldsymbol{x}, \boldsymbol{x}') = G(\boldsymbol{x}, \boldsymbol{x}'; \sigma) = \exp(-||\boldsymbol{x} - \boldsymbol{x}'||^2/\sigma). \tag{23.9}$$

Interesting results may also be obtained using the following kernel (exponential inverse of distance) which is quite similar to the Gaussian one:

$$k(\boldsymbol{x}, \boldsymbol{x}') = \exp(-||\boldsymbol{x} - \boldsymbol{x}'||/\sigma). \tag{23.10}$$

The main problem with the original definition of SVM was that its learning procedure, the quadratic programming (QP), converged very slowly. Recent years have brought a few novel methods of acceleration of the QP procedure for SVM learning. The most attention deserve the methods proposed by Osuna et al. (1997b), Joachims (1998), Saunders et al. (1998), Platt (1998, 1999) and Chang et al. (2000). Platt's Sequential Minimal Optimization (SMO) algorithm for the QP problems is very fast and provides an analytical solution. Further improvements to the QP procedure were made by Shevade et al. (2000).

The SMO algorithm augmented by the ideas presented in (Shevade et al., 2000) yields very fast and accurate solution of SVM learning problem. We have used such a version of SVM in our research.

The common point of acceleration of the QP procedure is the **decomposition** of $\boldsymbol{\alpha}$ to a working part ($\boldsymbol{\alpha}_B$) and a fixed part ($\boldsymbol{\alpha}_R$):

$$\max_{\boldsymbol{\alpha}_B} \quad W(\boldsymbol{\alpha}_B) = (\mathbf{1} - \boldsymbol{Q}_{BR}\boldsymbol{\alpha}_R)^T \boldsymbol{\alpha}_B - \frac{1}{2}\boldsymbol{\alpha}_B^T \boldsymbol{Q}_{BB}\boldsymbol{\alpha}_B \tag{23.11}$$

with constraints:

$$0 \le \alpha_{B,i} \le C \quad \forall\, i \in B, \tag{23.12}$$

$$\mathbf{y}_B^T \boldsymbol{\alpha}_B + \mathbf{y}_R^T \boldsymbol{\alpha}_R = 0, \tag{23.13}$$

where $\begin{bmatrix} \boldsymbol{Q}_{BB} & \boldsymbol{Q}_{BR} \\ \boldsymbol{Q}_{RB} & \boldsymbol{Q}_{RR} \end{bmatrix}$ is a permutation of matrix $\boldsymbol{Q}$.

The decomposition scheme consists of two steps: selection of $\boldsymbol{\alpha}_B$ and optimization of Eq. 23.11. These two steps are repeated as long as the stop criterion is not satisfied. The SMO selects only two $\boldsymbol{\alpha}$ scalars to put them to $\boldsymbol{\alpha}_B$. This is equivalent to the optimization of two (potential) support vectors in a single optimization step. The $\boldsymbol{\alpha}_B$ selection procedure introduced in (Platt, 1998) was optimized by Shevade et al. (2000). Keerthy proposed to optimize Equation 23.11 for the two indices ($B = \{i, j\}$) which violate the KKT conditions Equation 23.2 and Equation 23.3) the most:

$$i = \ \arg\ \max_t(\ \{-\nabla f(\boldsymbol{\alpha})_t \mid y_t = 1 \wedge \alpha_t < C\} \ \cup \tag{23.14}$$

$$\{\nabla f(\boldsymbol{\alpha})_t \mid y_t = -1 \wedge \alpha_t > 0\}), \tag{23.15}$$

$$j = \ \arg\ \min_t(\ \{\nabla f(\boldsymbol{\alpha})_t \mid y_t = -1 \wedge \alpha_t < C\} \ \cup$$

$$\{-\nabla f(\boldsymbol{\alpha})_t \mid y_t = 1 \wedge \alpha_t > 0\})$$

where $f(\boldsymbol{\alpha}) = \frac{1}{2}\boldsymbol{\alpha}^T\boldsymbol{Q}\boldsymbol{\alpha} + \mathbf{1}^T\boldsymbol{\alpha}$, and $\nabla f(\cdot)$ defines the gradient. For details on the stopping criterion see (Shevade et al., 2000).

When B consists of two indices the QP optimization defined by Equation 23.11 may be solved analytically. This was proposed in the SMO algorithm (Platt, 1998).

In the case of unbalanced data (large difference of the numbers of representatives of different classes) Osuna et al. (1997a) proposed to use a separate C parameter for each class. This changes the goal described by Equation 23.1 to:

$$\min_{\boldsymbol{w},b,\boldsymbol{\xi}} \quad \frac{1}{2}||\boldsymbol{w}||^2 + C^+ \sum_{y_i=1} \xi_i + C^- \sum_{y_i=-1} \xi_i \tag{23.16}$$

A method of automatic selection of C (Equation 23.1) and σ (Equation 23.9) parameters can be found in (Jankowski and Grabczewki, 2003).

Normalized Radial Basis Function (NRBF) Networks

The NRBF is a Radial Basis Function network with normalized Gaussian transfer functions. It resembles the concept of Parzen windows. The number of basis functions in the NRBF is equal to the number of vectors in the training dataset. Each basis function is placed exactly at the place defined by given input vector. The NRBF may be seen as lazy learning algorithm because there are no adaptation parameters.

Let $X = \{\boldsymbol{x}_i : i = 1, \ldots, m\}$ be a set of input patterns and $Y = \{y_i : i = 1, \ldots, m\}$ a set of class labels. The final class label (decision) on unseen vector $\boldsymbol{x}$ is computed as a conditional probability of class c given vector $\boldsymbol{x}$:

$$P(c|\mathbf{x}, X, Y) = \sum_{i \in I^c} k(\mathbf{x}; \mathbf{x}_i), \tag{23.17}$$

where $I^c = \{i : \mathbf{x}_i \in X \wedge y_i \in Y \wedge y_i = c\}$ and

$$k(\mathbf{x}; \mathbf{x}_i) = \frac{G(\mathbf{x}, \mathbf{x}_i; \sigma)}{\sum_{j=1}^{m} G(\mathbf{x}, \mathbf{x}_j; \sigma)}, \tag{23.18}$$

where $G(\mathbf{x}, \mathbf{x}_i; \sigma)$ is the Gaussian kernel (Equation 23.9) with σ parameter. It can be seen that $\sum_{i=1}^{K} P(i|\mathbf{x}, X, Y) = 1$, where K is the number of classes.

The behavior of NRBF is similar (but not equivalent) to the k nearest neighbors model (see Section 23.2.1) – the classification decision of given vector $\mathbf{x}$ depends on the neighborhood region of $\mathbf{x}$ (on which basis function is the nearest). The biggest difference is that the NRBF decision $(P(c|\mathbf{x}, X, Y))$ changes continuously while for kNN it is discrete.

If the training dataset consists of a large number of vectors the Learning Vector Quantization (Kohonen, 1986) or prototype selection methods (Grochowski and Jankowski, 2004) can be used to reduce the number of vectors appropriately.

k Nearest Neighbors (kNN)

k Nearest Neighbors models were proposed by Cover and Hart (1967) and are designed to classify unseen vectors on the basis of the class labels observed for neighboring reference vectors (typically the training set vectors). The kNN is parameterized by k, the number of nearest neighbors considered during classification. The winner class for a given vector $\mathbf{x}$ may be defined as the majority class within the set $NN(\mathbf{x}; k)$ of its k nearest neighbors.

Typically k is chosen manually. Sub-optimal value of k may be estimated quite effectively via cross-validation based learning – since each fold may estimate a different optimum for k, the sub-optimal value may be estimated by the k for which the average test accuracy (counted for the submodels of the CV based learning) is maximal.

The set $NN(\mathbf{x}; k)$ of *nearest neighbors* depends on the measure used to compute distances between $\mathbf{x}$ and the reference vectors. In most cases the Euclidean measure is used. The Euclidean measure can be simply generalized to the Minkovsky measure:

$$D_M^{\alpha}(\mathbf{x}, \mathbf{x}') = \sqrt[\alpha]{\sum_{i=1}^{n} |x_i - x_i'|^{\alpha}} \tag{23.19}$$

The Euclidean metric corresponds to $\alpha = 2$, which is completely isotropic, and Manhattan metric to $\alpha = 1$.

Sometimes good results can be obtained using the Canberra measure:

$$D_{Ca}(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{n} \frac{|x_i - x_i'|}{|x_i + x_i'|}. \tag{23.20}$$

The Chebychev function corresponds to the infinite Minkovsky exponent:

$$D_{Ch}(\mathbf{x}, \mathbf{x}') = \max_{i=1,\ldots,n} |x_i - x_i'|. \tag{23.21}$$

Please note that in the case of symbolic attributes a special metric or a data transformation (see (Grabczewski and Jankowski, 2003)) should be used.

SSV tree

The Separability of Split Value (SSV) criterion is one of the most efficient heuristic used for decision tree construction (Grabczewski and Duch, 1999, 2000). Its basic advantage is that it can be applied to both continuous and discrete features in such a manner that the estimates of *separability* can be compared regardless the substantial difference in types.

The *split* value (or *cut-off point*) is defined differently for continuous and symbolic features. For continuous features it is a real number and for symbolic ones it is a subset of the set of alternative values of the feature. The *left side* (LS) and *right side* (RS) of a split value s of feature f for a given dataset D is defined as:

$$\begin{aligned} \mathsf{LS}(s,f,D) &= \begin{cases} \{x \in D : f(x) < s\} & \text{if } f \text{ is continuous} \\ \{x \in D : f(x) \notin s\} & \text{otherwise} \end{cases} \\ \mathsf{RS}(s,f,D) &= D - \mathsf{LS}(s,f,D) \end{aligned} \tag{23.22}$$

where $f(x)$ is the f's feature value for the data vector x. The definition of the *separability of split value* s is:

$$\begin{aligned} \mathsf{SSV}(s,f,D) = 2 * \textstyle\sum_{c \in C} |\mathsf{LS}(s,f,D_c)| * |\mathsf{RS}(s,f,D-D_c)| \\ - \textstyle\sum_{c \in C} \min(|\mathsf{LS}(s,f,D_c)|, |\mathsf{RS}(s,f,D_c)|) \end{aligned} \tag{23.23}$$

where C is the set of classes and D_c is the set of data vectors from D assigned to class $c \in C$.

Among all the split values which separate the maximum number of pairs of vectors from different classes the most preferred is the one that separates the smallest number of pairs of vectors belonging to the same class. For every dataset containing vectors, which belong to at least two different classes, for each feature represented in the data by at least two different values, there exists a non-trivial split value with maximum separability. When the feature being examined is continuous and there are several different split values of maximum separability, close to each other, the split value closest to their average is selected. To avoid such ties and to eliminate unnecessary computations, the analysis should be restricted to the split values that are natural for the given dataset (i.e. centered between adjacent feature values that occur in the data vectors). If there are non-maximal (regarding separability) split values between two maxima or if the feature is discrete, then the average is not a reasonable choice – the winner split value should be selected randomly from those of maximum separability.

Decision trees are constructed recursively by searching for best splits among all the splits for all the features. At each stage when the best split is found and the subsets of data resulting from the split are not completely pure (i.e. contain data belonging to more than one class) each of the subsets is analyzed in the same way as the whole data. The decision tree built this way gives maximum possible accuracy (100% if there are no contradictory examples in the data), which usually means that the created model overfits the data. To remedy this a cross validation training is performed to find the optimal parameters for pruning the tree. The optimal pruning produces a tree capable of good generalization of the patterns used in the tree construction process.

Like most decision tree algorithms the SSV based method is independent on the scaling of the feature values, so in particular it is normalization and standardization invariant. The decision borders are perpendicular to the feature space axes and can be described by logical formulae, however in some cases it is a restriction, which limits accuracy. Nevertheless its ability to find informative features can be helpful in feature selection for other classification methods.

The SSV criterion has been successfully used not only for building classification trees, but also for feature selection (Duch et al., 2002, 2003) and data type conversion (from continuous to discrete and in the opposite direction (Grabczewski and Jankowski, 2003)).

23.2.2 Feature extraction

Providing classifiers with feature spaces, which help obtaining the best possible accuracy is a very complex task. Feature selection and construction play a very important role here. There is no single recipe for good data transformation and no unarguable method to compare the performance of different feature selection algorithms. Moreover each classifier may require different data preparation. To obtain an accurate and stable final model with a particular classifier one must validate a number of data preparation methods.

The term *feature extraction* encompasses both selection and construction of features. Thorough analysis includes testing filters (which are independent on the classifier) and wrappers (which use external classifiers to estimate feature importance). Feature selection strategies either produce a ranking (each feature is assessed separately) or perform *full-featured* selection (select/deselect with respect to the interaction between the features).

CC based feature ranking

The correlation coefficient (CC) is a simple but very robust tool in statistics. It is very helpful also in the task of feature selection. For two random variables X and Y it is defined as

$$\varrho(X,Y) = \frac{E(XY) - E(X)E(Y)}{\sqrt{D^2(X)D^2(Y)}}, \tag{23.24}$$

where E and D^2 stand for the expected value and variance respectively. $\varrho(X,Y)$ is equal to 0 if X and Y are independent and is equal to 1 when the variables are linearly dependent ($Y = aX + b$).

The correlation coefficient calculated for a feature (treated as a random variable) and the class labels (in fact the integer codes of the labels) is a good measure of feature usefulness for the purpose of classification. The feature list ordered by decreasing absolute values of the CC may serve as feature ranking.

SSV based feature selection

Decision tree algorithms are known to have the ability of detecting the features that are important for classification. Feature selection is inherent there, so they do not need any feature selection at the data preparation phase. Inversely: their capabilities can be used for feature selection.

Feature selection based on the SSV criterion can be designed in different ways. The most efficient (from the computational point of view) one is to create **feature ranking** on the basis of the maximum SSV criterion values calculated for each of the features and for the whole training dataset. The cost is the same as when creating decision stubs (single-split decision trees).

Another way is to create a **single** decision **tree** and *read* feature importance from it. The filter we have used for this type of SSV based feature selection is the algorithm 12.

```
Feature selection filter based on the SSV criterion

▶ Input: A sample $X$ of input patterns and their labels $Y$ (training data)
◀ Output: List of features ordered by decreasing importance.

• $T \leftarrow$ the SSV decision tree built for $\langle X, Y \rangle$.
• For each non-final (i.e. which is not a leaf) node $N$ of $T$,
  $G(N) \leftarrow E(N) - E(N_1) - E(N_2)$, where $N_1$ and $N_2$ are the subnodes of
  $N$, and $E(N)$ is the number of vectors in $X$ falling into $N$ but incorrectly
  classified by $N$.
• $\mathcal{F} \leftarrow$ the set of all the features of the input space.
• $i \leftarrow 0$
• While $\mathcal{F} \neq \emptyset$ do:
    – For each feature $f \in \mathcal{F}$ not used by $T$ define its rank $\mathcal{R}(f) \leftarrow i$.
      Remove these features from $\mathcal{F}$.
    – Prune $T$ by deleting all the final splits of nodes $N$ for which $G(N)$ is
      minimal.
    – Prune $T$ by deleting all the final splits of nodes $N$ for which $G(N) = 0$.
    – $i \leftarrow i + 1$
• The result is the list of features in decreasing order of $\mathcal{R}(f)$.
```

This implements a full-featured filter – the decision tree building algorithm selects the splits locally, i.e. with respect to the splits selected in earlier stages, so that the features occurring in the tree, are complementary. The selection can be done by dropping all the features of rank equal to 0 or by picking a given number of top ranked features.

In some cases the full classification trees use only a small part of the features. It does not allow to select any number of features – the maximum is the number of features used by the tree. To remedy this the Sequential Feature Selection technique (described below) can be used.

The SSV criterion is defined to reflect class separability and has no parameters to adjust it to standard or balanced classification error. Thus we have also used the SSV framework to construct trees with balanced classification error as split eligibility criterion. It was especially useful for the exploration of the Dorothea dataset.

Feature selection wrapper

Wrapper methods use external algorithms and search techniques to determine the best (from the point of view of some particular criterion) values of some parameters. The technique may also be helpful in feature selection. A wrapper method available in the GhostMiner package simply adds the features one by one to some initial set of (base) features and estimates the performance of a classifier in so extended feature spaces. If the initial set of features is empty then the classifier is trained on one-dimensional datasets, and the results for all the features are collected – the feature ranking is built according to the accuracies. When started with some base features the method searches for additional features, which extending the preselected set yield a satisfactory improvement in submodel accuracy.

The basic advantage of the wrapper method of feature selection is that its applications are dedicated to some particular models. The major drawback of wrappers is that they require multiple training of their submodels.

Feature selection committee

The task of feature selection committees is to combine different feature selection methods and select features which seem attractive from different points of view. Several feature selection models are constructed independently and their selection results collected. The committee selects the features most often selected by its members. If we assign the value of 1 to each selected feature and 0 to not selected then we may sum up the scores obtained from the committee members to get an integer value for each of the features. The committee scores are integer values in the range of 0 to the number of committee members. Setting a threshold value for the scores gives a criterion of final selection. The threshold equal to the committee size selects the features selected by each of the committee members while the value of 1 results in a weak rejection committee. The two border values correspond respectively to the intersection and the sum of the sets of features determined by the members.

Sequential feature selection

Full-featured filters select the features providing different (complementary) information about the classification task. They are likely to reject informative features, which although valuable do not introduce anything new to already selected ones. If we want neither simple rankings, which do not reflect features dependencies nor filters that deselect informative but not independent features, the sequential feature selection technique may be of interest. The idea is to select just a number of top ranked features and repeat filtering in the feature space reduced by the selected features. The parameters of this method are the filter algorithm, the number of filter runs and the number of features to select after each step.

The method is helpful in the case of full-featured filters, especially like the ones based on decision trees, which in some circumstances can select only a small number of features. Running them repetitively facilitates selection of any number of features.

Information theory based filters

There is a number of feature filters based on information theory. Unfortunately they usually suffer from the necessity of data discretization by external methods. The equal-width and equal-frequency discretizations are not very robust. Much more interesting results can be obtained with SSV based discretization (Duch et al., 2003, Grabczewski, 2004) but in most cases they are not better than those of SSV feature selection while being more computationally expensive. Some successful methods which employ information gain or mutual information were tried by us on the NIPS FSC datasets. The results were very similar to those obtained with CC or SSV based feature selection. The lack of information theory models inside GhostMiner significantly reduced our validation possibilities for these models – this is the major reason why we have not used the methods in our final models.

PCA

The Principal Components Analysis is a well known technique of data transformation. In its standard formulation it finds linear combinations of features which show the directions of largest variance of the data. Viewing the data in two dimensional plots, where the axes are the first and the second principal components is often very informative. Using the largest variance directions as features may significantly lower dimensionality of the space (where most classification models are more effective) without a substantial reduction of information. This feature extraction method constructs valuable features, but it can not be treated as a feature selection technique because all the feature values are still necessary to calculate the new features.

23.3 Fully operational complex models

It can be seen in Section 31.4 that single classification algorithms are often not sufficient to solve a given problem with high accuracy and confidence. It is much more successful to examine different combinations of data transformation and classification models presented in the previous sections (23.2.1 and 23.2.2). Sometimes it is recommended to use even more than one transformation before classification (compare section 23.4.3), but searching for a suitable model sequence and configuration is far from trivial. Sometimes, default parameters (commonly used as starting configuration) are completely inadequate for a specific dataset (compare section 23.4.4). Also, a combination of transformation and classifier useful for one dataset may be useless for another dataset.

It is recommended to search for proper parameters of transformations and classifiers with a meta-learning. Meta-learning should perform internal validation of meta-parameters (the parameters the values of which are searched for), otherwise the learning process is very likely to end up with an overfitted model.

The task of searching for complex models can be viewed as a number of subproblems: testing of base classifiers, testing of feature selection algorithms in the context of their suitability for base classifiers, parameters tuning for models, which seem promising solutions, branching of promising solutions (substitution of parts in the complex models), further search for configuration of complex models, testing of feature selection committees and classification committees. The whole process must switch from one subproblem to another with repetitions and backtracking.

Reliable validation is extremely important in the case of complex models (combinations of transformations and classifiers). If a transformation is supervised (dependent on the class labels) then it is insufficient to validate the classifier – instead, the cross-validation (or other random validation procedure) should run over the whole combination of transformation and classifier. Otherwise the prediction of accuracy and their variance is overoptimistic and falsifies the real generalization possibility. In the case of unsupervised transformations (like PCA, standardization or selection of high variance features), they may be applied before the validation, however if a combination of supervised and unsupervised transformations is used to prepare data for a classification model, then the combination is supervised and as such, must be nested in the validation.

The cross-validation test may be easily changed into a cross-validation committee. It means that all the models built in the cross-validation test can compose a committee. The final decision of the CV committee are based on the voting scheme. To reduce the probability of impasse, it is recommended to use an odd number of CV folds (especially in the case of two–class problems). CV committees have several important advantages: the first is that committee decisions are more stable than those of single models, the second is that the estimated accuracy and variance of the submodels are known directly from the CV, another one is that they avoid the problem of configuration parameters, which although validated may be suitable only for a particular dataset size (the numbers of features, vectors, etc.) and applied to the whole training dataset may produce less successful models.

If for a given dataset we had a number of interesting (complex) models then we would choose the best one according to the following criterion:

$$\text{best-model} = \arg\max_{M} \; [\, \mathit{accuracy}(M) - \alpha \cdot \mathit{standard\text{-}deviation}(M) \,] \tag{23.25}$$

with the values of α close to 1. The aim is to prefer not only accurate but also confident models.

23.4 Challenge data exploration

The challenge datasets differ in many aspects (their size – the number of features and vectors, the source, the representation type, etc.). As a result, the final classification models are also significantly different.

Below, an overview of the most interesting models for the challenge datasets is presented. For each dataset there is a table depicting the structure of our best model and its error rate calculated by the challenge organizers for the test part of the dataset.

Table 23.1. NIPS 2003 challenge results.

Dec. 8th	Our best challenge entry					The winning challenge entry					
Dataset	Score	BER	AUC	Feat[2]	Probe[2]	Score	BER	AUC	Feat	Probe	Test
Overall	37.14	7.98	92.11	26.8	—	71.43	6.48	97.20	80.3	47.8	0.8
Arcene	14.29	13.53	86.47	75	—	94.29	11.86	95.47	10.7	1.0	1
Dexter	71.43	3.50	96.50	40	—	100.00	3.30	96.70	18.6	42.1	0
Dorothea	17.14	13.11	86.89	2	—	97.14	8.61	95.92	100.0	50.0	1
Gisette	57.14	1.31	98.69	14	—	97.14	1.35	98.71	18.3	0.0	0
Madelon	65.71	7.44	92.56	3	0.0	94.29	7.11	96.95	1.6	0.0	1

The models were submitted to the second part of the contest (December 8th). We assessed the second stage as the most important. Thus, for real, we did not take part in the stage of December 1st. In the final contest we reached the **group rank of 3** and **best entry rank of 7**. Table 23.1 presents a summary of the results.

23.4.1 Arcene (spectroscopic data)

Arcene and Dorothea are characterized by high quotient of the number of attributes and the number of input vectors (≈ 100). In such spaces looking for accurate and certain classification is a very hard problem. If a supervised preprocessing is used and then the validation (such as the CV test) performed, the classification accuracy estimates are overoptimistic – real accuracy on unseen data is dramatically higher (the generalization is very poor).

The best model we have found is a CV Committee of combinations of SSV based feature selection and the SVM classifier with linear kernel and class balance. The CV test error was $7.8\% \pm 5\%$.

CV Committee 9-fold [SSV 7000 $\rightarrow$ SVM linear]	Test error: 13.5%

The training set was standardized before feature selection. Quite similar results can be obtained with the correlation coefficient based feature selection. The use of CV Committee increases the CV test accuracy by 1-1.5% It was a surprise, that using SVM with linear kernel and no feature selection the CV test accuracy was only 2% lower (than the best score) with slightly higher variance.

The CV test accuracy goes down (quite monotonically) when SSV or correlation coefficient selection is used to extract less than 7000 features.

It can be observed that SVMs with linear kernel work very well in high-dimensional spaces while with gaussian kernel rather do not.

[2]We did not submit the lists of selected features to the contest (except for Madelon). Hence the fractions of features differ from the ones calculated by the organizers and some probe information is missing.

23.4.2 Dexter (corporate acquisitions)

In the case of Dexter the first step of the analysis was to remove the zero–variance features. After that the number of features reduced from 20 000 to 11 035.

The Dexter dataset is sparse. It is important to treat undefined values as zeros, not like a missing value – otherwise the classification is much more difficult. Data standardization also makes the task harder, so we have used the original data. Alternatively, a standardization using the mean and the standard deviation over the whole training set (not *per* feature) can be used. The best model we have found, consists of the correlation coefficient feature selector and SVM with linear kernel. The CV test error was around $4.8\% \pm 2\%$.

CC 8000 → SVM linear	Test error: 3.5%

Very similar results were obtained using CV Committee of the above combinations (2 more vectors misclassified on the test set). Another model with very similar certainty was a CV Committee of the same combinations but with 5000 features selected.

23.4.3 Dorothea (which compounds bind to Thrombin)

The Dorothea dataset is binary and strongly sparse. As it was already mentioned, it has (over) 100 times more features than vectors.

In the first step unsupervised feature selection was used. A given feature was selected only if it had a sufficient variance (high variance selection – HVS): in practice, more than p 1s per feature were required. There are no features with high number of 1s in the training data, so the values of the p parameter we have used are: 8, 10 and 11.

After the preprocessing the best combinations of models use both supervised and unsupervised data transformations before final classification. One of the best models starts with selection of features with high variance, next the SSV selects 2000 features (respecting the class balance), then two first principal components are extracted (see Fig. 23.1), and finally the SVM classifier with gaussian kernel is used (also respecting the class balance). The CV test of this model estimated the error of $12.3\% \pm 4.4\%$.

HVS p=11 → SSV 2000+balance → PCA 2 → SVM Gaussian C=50	Test error: 13.1%

Similar results can be obtained with $p = 8$ for the high variance selection or with SSV based selection of 1500 features.

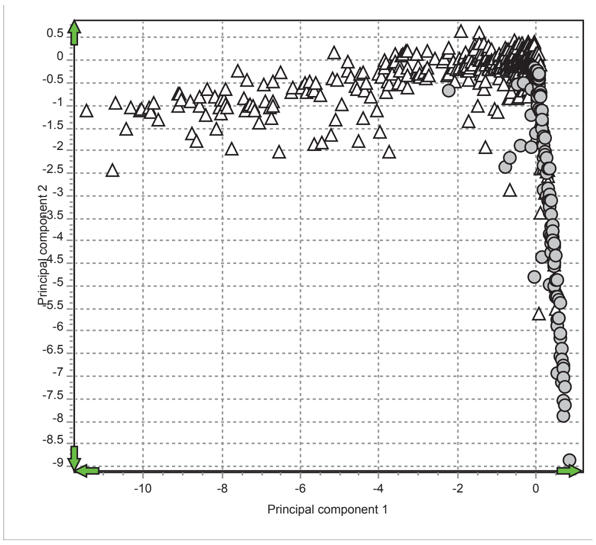

Fig. 23.1. First two principal components of DOROTHEA after SSV feature selection.

23.4.4 GISETTE (handwritten digits)

The GISETTE dataset has nearly balanced numbers of instances and attributes. The major problem of this data was not only to find a proper combination of models but also to tune the parameters of the CC selector and SVM classifier. Our best model uses 700 features, and gives the CV test error of 1.5% ± 0.5%:

CC 700 → SVM Gauss C=1000 bias=0.002	Test error: 1.31%

Another interesting model is the combination of correlation coefficient based feature selector (with just 200 features) and kNN with the number of neighbors (k) equal to 5. The CV test error of such configuration is 3.6%, and the standard deviation is smaller than 0.6%. When the correlation coefficient selector is used to select 50 features the CV test error increases to 5%.

23.4.5 MADELON (random data)

MADELON is a dataset of its own kind. In comparison to ARCENE and DOROTHEA, it has a small quotient of the numbers of features and instances.

Atypically, to select relevant features the feature selection committee was used. The committee members were a SSV ranking, a SSV single tree (ST) selection and a correlation coefficient selector.

Selection Committee [SSV, SSV ST, Correlation coefficient] → NRBF	Test error: 7.44%

The CV test error of the above combination of models is $9\% \pm 0.5\%$ (very stable model). Although the selection committee looks complicated, it selects just 15 features. The validation techniques showed that both selecting more features and reducing the number of features led to a decrease of the accuracy on unseen data. In the cases of MADELON and GISETTE the subtask of finding an accurate and stable classifier was much harder than the feature selection stage.

Slightly worse results can be obtained with kNN model instead of NRBF – the CV test accuracy reduction is close to 1%.

23.5 Conclusions

The challenge efforts of our group brought a number of conclusions which in general are compatible with our earlier experience, however some aspects deserve to be emphasized and some are a surprise:

- It has been confirmed in practice that there is no single architecture (neither learning algorithm nor model combination scheme) of best performance for all the tasks. Although the SVM method was used most often in the final models, its internal structure was not the same each time – some models were based on linear and some on gaussian kernels.
- The models created must be properly validated – all the supervised data preprocessing (like most feature selection methods) must be included in a complex model validated with a CV or similar technique. Otherwise there is a serious danger of overfitting the training data.
- It is advantageous to build committees of models, which proved their generalization abilities in a CV test.
- It is surprising that a feature ranking method as simple as the one based on the correlation coefficient is a very valuable component of complex models.

There is still a lot of work to be done in the area of feature selection and building efficient model combinations. The problem is NP-complete, and the needs are growing due to the bioinformatics and text mining applications. Among other things we should: look for stronger and still effective feature selection algorithms; construct powerful aggregation or aggregation–selection algorithms to help the classifiers by supplying more informative features; develop intelligent meta-learning techniques which could help in automating the search for adequate complex models.

References

R. Adamczak, W. Duch, and N. Jankowski. New developments in the feature space mapping model. In *Third Conference on Neural Networks and Their Applications*, pages 65–70, Kule, Poland, 1997. Polish Neural Networks Society.

B.E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Fifth Annual Workshop on Computational Learning Theory*, pages 144–152. ACM, 1992.

C.-C. Chang, C.-W. Hsu, and C.-J. Lin. The analysis of decomposition methods for support vector machines. *IEEE Transaction on Neural Networks*, 4:1003–1008, 2000.

T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

W. Duch, J. Biesiada, T. Winiarski, K. Grudziński, and K. Grabczewski. Feature selection based on information theory filters and feature elimination wrapper methods. In *Proceedings of the International Conference on Neural Networks and Soft Computing (ICNNSC 2002)*, Advances in Soft Computing, pages 173–176, Zakopane, 2002. Physica-Verlag (Springer).

W. Duch, T. Winiarski, J. Biesiada, and A. Kachel. Feature selection and ranking filters. In *Artificial Neural Networks and Neural Information Processing – ICANN/ICONIP 2003*, pages 251–254, Istanbul, 2003.

K. Grabczewski. SSV criterion based discretization for Naive Bayes classifiers. In *Proceedings of the 7th International Conference on Artificial Intelligence and Soft Computing*, Zakopane, Poland, June 2004.

K. Grabczewski and W. Duch. A general purpose separability criterion for classification systems. In *Proceedings of the 4th Conference on Neural Networks and Their Applications*, pages 203–208, Zakopane, Poland, June 1999.

K. Grabczewski and W. Duch. The Separability of Split Value criterion. In *Proceedings of the 5th Conference on Neural Networks and Their Applications*, pages 201–208, Zakopane, Poland, June 2000.

K. Grabczewski and N. Jankowski. Transformations of symbolic data for continuous data oriented models. In *Artificial Neural Networks and Neural Information Processing – ICANN/ICONIP 2003*, pages 359–366. Springer, 2003.

M. Grochowski and N. Jankowski. Comparison of instances seletion algorithms II: Algorithms survey. In *Artificial Intelligence and Soft Computing*, pages 598–603, 2004.

N. Jankowski and K. Grabczewki. Toward optimal SVM. In *The Third IASTED International Conference on Artificial Intelligence and Applications*, pages 451–456. ACTA Press, 2003.

N. Jankowski and V. Kadirkamanathan. Statistical control of RBF-like networks for classification. In *7th International Conference on Artificial Neural Networks*, pages 385–390, Lausanne, Switzerland, October 1997. Springer-Verlag.

T. Joachims. *Advances in kernel methods — support vector learning*, chapter Making large-scale SVM learning practical. MIT Press, Cambridge, MA, 1998.

T. Kohonen. Learning vector quantization for pattern recognition. Technical Report TKK-F-A601, Helsinki University of Technology, Espoo, Finland, 1986.

E. Osuna, R. Freund, and F. Girosi. Support vector machines: Training and applications. AI Memo 1602, Massachusetts Institute of Technology, 1997a.

E. Osuna, R. Freund, and F. Girosi. Training support vector machines: An application to face detection. In *CVPR'97*, pages 130–136, New York, NY, 1997b. IEEE.

J. C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*. MIT Press, Cambridge, MA., 1998.

J.C. Platt. Using analytic QP and sparseness to speed training of support vector machines. In M.S. Kearns, S.A. Solla, and D.A. Cohn, editors, *Advances in Neural Information Processing Systems*, volume 11, 1999.

C. Saunders, M.O. Stitson, J. Weston, L. Bottou, B. Schoelkopf, and A. Smola. Support vector machine reference manual. Technical Report CSD-TR-98-03, Royal Holloway, University of London, Egham, UK, 1998.

S.K. Shevade, S.S. Keerthi, C. Bhattacharyya, and K.R.K. Murthy. Improvements to the SMO algorithm for SVM regression. *IEEE Transactions on Neural Networks*, 11:1188–1194, Sept. 2000.

V. Vapnik. *The Nature of Statistical Learning Theory.* Springer-Verlag, New York, 1995.

V. Vapnik. *Statistical Learning Theory.* Wiley, New York, NY, 1998.