Credit Client Classification: Models Using Information from Class Boundaries and from Cluster Representatives

Klaus B. Schebesch[1] and Ralf Stecking[2]

[1] Faculty of Economics, Vasile Goldiş Western University, Arad, RO-310396 Arad, Romania, kbschebesch@uvvg.ro
[2] Department of Economics, Carl von Ossietzky University Oldenburg, D-26111 Oldenburg, Germany, ralf.w.stecking@uni-oldenburg.de

Summary. Real life credit client data containing well in excess of hundred thousand records are used nowadays in order to predict the expected defaulting behavior of new clients. Such credit client feature data is also rather high dimensional and it may contain a hidden cluster structure, which is exploitable for prediction model building and becomes even more so with very large numbers of client records. Some preliminary modeling success concerning clustering as a preprocessing step in prediction model building by means of classification models was reported by the present authors and others. This approach is based on separately extracting cluster representatives from class-wise data subsets. A powerful classification method like non-linear SVM is then trained on these labeled cluster representatives. Hence, clustering is here used in order to more compactly represent the training data sets of the defaulting and of the non-defaulting credit clients, respectively. In order to accomplish this for large data sets a variant of centroid clustering like the k-means algorithm is used. Therefore, in this modeling approach, cluster formation is in fact not actively connected to the task of class separation: few large clusters may form at class boundaries, potentially oversimplifying the resulting separation function. However, somewhat surprisingly, upon extensively validating the combined cluster-classification approach for different cluster numbers and alternative data encodings, we find that expected forecasting performance measured by the ROC criterion is competitive if not superior for relatively small to medium sized cluster numbers and levels off or even decreases when using many more clusters.

1 Motivation and Background

Forecasting defaulting behavior of credit clients is an important request from modern financial industries to data based statistical modeling. In a world with very many credit applicants this seems reasonable in order to improve judgment concerning credit-worthiness of newly arriving credit clients. Modeling defaulting behavior of credit clients assumes that there exists sufficient statistical regularity relating credit client features to defaulting behavior, and hence, that in a statistical sense, one can learn from considering past credit client feature-behavior pairs. Such learning has been shown to be indeed the case to occur for small to medium sized data sets (Thomas et al. (2005), Schebesch and Stecking (2005), Baesens et al. (2003)) and for a broad spectrum of statistical models, ranging from traditional Linear Discriminant Analysis (LDA) to the more recently developed Support Vector Machines (SVM) and especially their nonlinear kernels variants. Being naturally equipped to deal with nonlinearity and class asymmetries and being more robust with regard to problem data distributions, SVM seem to slightly outperform more traditional methods in credit scoring.

Classification of such more involved data representations requires an adequate data coding to be combined with appropriate multivariate analysis tools. We show how to construct good and bad credit client clusters that can be classified with SVM. In this way, owing to the reduced number of effective training examples, very large data sets can be made accessible to SVM with non linear kernels and to SVM which handle non standard situations of classification (with unequal class sizes and asymmetric misclassification costs), which are very important in practice. In using SVM we can also profit from information inherent to support vectors, deriving region information from different types of such support vectors, which in turn helps to get more insight into the data clusters. This can be used in order to compare alternative

cluster solutions or to learn a more appropriate distance function for the process of clustering. Finally, we evaluate the expected forecasting performance of our classification models trained on cluster representatives instead of all original data points.

There are at least two particular situations, where such a model approach is worthwhile investigating. One is where the more powerful forecasting model variants of a model spectrum are too expensive (or even impossible) to train and validate. In this case they are trained on the much smaller set of compressed data instead. The other one is where a modeler cannot obtain access to the full case wise data for different reasons concerning information disclosure, but can nevertheless express an option for or even provide the compression procedure to be applied by the data owner before the latter will hand the compressed data over. Here the concern is to recommend or to configure the compression procedure and to choose a forecasting model, which can maximally exploit these compressed data.

Several comparative studies of credit scoring and rating approaches with SVM have been published. Baesens et al. (2003) report on benchmarking SVM with other classifiers using several different data sets. They find SVM performing well, but with traditional methods being competitive, especially when there is weak non-linearity in credit scoring data sets. Li et al. (2006) compare SVM with traditional neural networks for consumer credit data, with SVM being superior in terms of generalization performance. Huang et al. (2004) also observe slight improvement of SVM over neural networks when used for credit rating analysis. Bellotti and Crook (2008) report that SVM with linear or RBF kernel are successful when compared to more traditional methods. Using a very large data set of approximately 25000 credit card customers, however, they found no overwhelming superiority of SVM.

Different aspects of combining clustering with SVM are treated in the more recent literature: Li et al. (2007) introduce a Support Cluster Machine (SCM) as a general extension of the SVM with the RBF kernel, where cluster size and cluster covariance information is incorporated into the kernel function. Evgeniou and Pontil (2002) propose a special clustering algorithm for binary class data, that tends to produce large clusters of training examples which are far away from the boundary between the two classes and small clusters near the boundary. Boley and Cao (2004) use Adaptive Clustering to reduce the number of training data. After partitioning the training set into several clusters, the cluster representatives are used to train an initial SVM. Yuan et al. (2006) concentrate on large *imbalanced* data sets. They partition the examples from the bigger negative class into disjoint clusters and train an initial SVM with RBF kernel using positive examples and cluster representatives of the negative examples.

2 What are Large Data Sets in Credit Scoring ?

Our data set consists of information from 139951 clients for a building and loan credit. Within a time period of one year 3692 clients refused to repay the loan. Thus, the *default rate*, based on a definition of the building and loan association, is 2.6%. There are twelve variables per client of which eight are categorical with two up to five categories and four are quantitative. Input variables include loan related attributes like *interest rate* and *credit amount*, personal attributes like *employment status* and object related attributes like *house type*. The data set is large and in general not (or hardly) tractable for standard SVM. Furthermore, extremely unequal class sizes will make classification difficult. Past own experiments suggest using a small sample with equal sized classes for SVM model building. However this leads merely to average out–of–sample forecasting performance for the large data set (Stecking and Schebesch (2006)). In the present work we use an alternative way of dealing with such large data sets: First, a cluster analysis is carried out and then the SVM is estimated on the respective cluster representatives.

The advantage of a preprocessing type cluster analysis is massive down-sizing of the large data set of N cases into a much smaller set of $n \ll N$ cluster representatives. In order to avoid difficulties that are typical for imbalanced data sets (a survey is given in Weiss (2004)), these cluster solutions may include a weighting scheme, which uses for instance a balanced number of "good" and "bad" credit client clusters. As a convenient by-product of clustering, possible noise in the data will be averaged out. In some applications like credit scoring, the number of cases or clients can be expected to further grow even further in the future, whereas the number of easily accessible features per client can be expected to remain approximately the

same (barring the use of some new and controversial information on health records, on social networking, etc) making the present approach of combining clustering and classification even more relevant.

3 Cluster-based Classification of Credit Client Data

Given the data from section 2 we proceed with evaluating the combined cluster and classification procedure by evaluating a large number of cluster based nonlinear SVM models using RBF kernels. To this end we concisely describe the most important aspects of classification and cluster modeling used. A set of $N > 0$ data points relating to individual credit clients $\{x_i, y_i\}$, $i = 1, ..., N$, with $x_i \in \mathbb{R}^m$ (m client input features) and associated labels $y_i \in \{-1, 1\}$ (for non-defaulting and defaulting behavior, say) will used in both the clustering and the classification step. If one would use the data directly in a SVM classification model as training examples, then the resulting *support vectors* are those training examples which are near the *class boundaries* of a SVM solution. These special data points permit training of a classifier with the same out-of-sample performance as a classifier trained on all the other (redundant, etc.) training examples as well. For our exposition it suffices to know that the SVM finally produces a decision rule (a separating function) of the type $y^{pred} = \mathbf{sign}(s(x)) = \mathbf{sign}\Big(\sum_{i=1}^{N} y_i \alpha_i^* k(x_i, x) + b^*\Big)$, with $0 \leq \alpha_i^* \leq C$ and b^* the result of the SVM optimization. Important is to note that $\alpha_i^* > 0$ and a support vector i is referred to as "essential" if $\alpha_i^* < C$ and as "bounded" if $\alpha_i^* = C$. The first user selected *hyper-parameter* $C > 0$ controls the amount of misclassification (or "softness") of the SVM model by means of which of the learning examples to activate via $\alpha_i^* > 0$ in $s(x)$. Hence, C implicitly selects the effective functional form of $s(x)$. The decision function invokes the ith training example via a *kernel function* $k(,)$ which in our cases is selected to be the RBF kernel defined by $k(u, v) = \exp\Big(-\sigma||u - v||^2\Big)$ for two commensurable feature vectors u and v, with $||.||^2$ being the (e.g. Euclidean) distance between u and v, and with $\sigma > 0$ being the second *hyper-parameter* of the SVM model. Changing σ changes the locality of the function $s(x)$, that is the distance–dependent contribution of training examples to forecasting of the label of x. Programs in classification software packages can often account for certain data biases like asymmetric class membership frequencies, which are very important in practice. For instance, the software SVM_Light (Joachims (2002, 2008)) which we use in order to compute our SVM classification models can account for different class and case wise misclassification costs, which may change the resulting model behavior substantially.

Now our goal is to replace the full or "extensive" training data set $\{x_i, y_i\}$, $i = 1, ..., N$ by cluster representatives of clusters computed on that data. Obviously, in order to achieve useful compression, the number of such cluster representatives n should be much less than N. While, in general, apart from being a compressed image of some characteristics of the cluster members, cluster representatives can assume a large variety of forms. A clustering procedure, which solves such a problem in the sense of grouping similar points into non-overlapping clusters (consult e.g. Jain et al. (1999)) and which does not require the computation of all mutual client distances is the k-means algorithm, readily available in many popular statistical computer packages like for instance R-Cran and SPSS.

By now there is a substantial literature available (including some very recent research, e.g. Bubeck et al. (2009) which analyzes the interplay of initialization and stability of k-means clustering solutions, especially when using very few clusters. Again, this is obviously important when clustering is intended to identify "true" clusters in data but may be much less so when clustering is used to compress data of each class separately, in the hope of nevertheless pertaining a good class separation. In the latter situation using as many *cluster centers* as required or as can be processed with reasonable effort in very large applications is recommended (von Luxburg (2009)).

4 Experimental Setup

Evaluating the comparative expected forecasting performance of models trained on cluster centers and to ease their comparison with "conventional" models trained on the full data, we

have to use a procedure, which is statistically safe and which also enables a fair comparison of the models from structurally different model classes. In order for our model building to be statistically acceptable, we use a ten-fold *crossvalidation procedure*. To this end we randomly subdivide the data into $\frac{2}{3}N$ examples used for training while the remaining $\frac{1}{3}N$ examples are held out for validation. This is repeated ten times generating ten different training-validation sets. The expected forecasting performance of a model is compared on these respective ten validation sets, returning thus the average, the best and the worst measured *forecasting performance* computed for a model.

For the clustering part the distance measure between two data points is chosen to be the Euclidian distance function. The Euclidian distance is also used within the RBF kernels of the SVM classification models. The expected forecasting performance of the cluster based classification models is measured by the *cutoff-independent receiver operator characteristics curve* (ROC) mapping "hit rate" against "false alarm rate" resulting in a 45 degree diagonal for models no better than "pure chance" and in a curve above the diagonal, if both rates are not equal for most cut-off values of class separation. For the comparison between different models the area under (the) ROC curve, conventionally denoted by AUC, is used.

Finally some consideration is in order about how and when to adapt SVM hyper-parameters to new training sets resulting from each new clustering. As discussed earlier in this sections new clusterings are in general the result of starting the k-means algorithm with different random initial cluster centers and owing to trying out different numbers of such clusters $1 < k < N$. Adapting the SVM hyper-parameters to each new modeling instance would be very time consuming.

Fortunately enough, the outcome of preliminary exploratory experiments with random data sampling, random overlapping and non-overlapping clusterings, random cluster numbers and random cluster algorithm restarts seems to indicate that expected SVM forecasting performance is not very sensitive with regard to the choice of the hyperparameters as long as a "reasonable" combination of these parameter values is found. In order to find such values a grid search over $\{C, \sigma\}$ around values determined for several models trained on small subsamples of the data and validated by the built-in leave one out procedure of SVM_Light is conducted.

5 Classification Results

We now proceed to the description of the results of the combined cluster based classification model evaluation. Figure 1 depicts an experiment with a small number of non-overlapping clusters enforced for each class by the k-means algorithm producing (for each class) five training points in order to be used by the SVM classification model. The different models result from consecutive initializations of the cluster algorithm by random "cluster centers". The variation of AUC values for each initialization is produced by using different $\frac{1}{3}$ holdout sets in the ten-fold validation procedure. Variation owing to changing validation sets is bigger than variation due to cluster algorithm initialization, although the latter seems to influence the distribution of validation set produced AUC values. Note also that, somewhat surprisingly, average and maximal AUC assume rather high values for these very small SVM classification models.

Figure 2 depicts a similar situation to that from figure 1, except that owing to 50 cluster representatives (training points) for each class the resulting SVM classification models can now be much bigger (much more expressive in terms of separating functions). The average AUC values for each initialization are individually superior to those of figure 1. Also the smallest minimum of the bigger models is bigger than the biggest maximum of the smaller models indicating that there is indeed some exploitable finer grained structure in the data.

In order to push the limits towards large models, which can potentially use almost all detail from the effective ($\frac{1}{3}$) training sets of the smaller class presented to the classification models, figure 3 shows evaluation results analogous to figures 1 - 2, which clearly shows a leveling off effect concerning AUC both values and their variations. A much "weaker" difference seems to be found in the shape of the distribution of the respective AUC values. This result indicates that using classification models trained on many cluster centers does not pay off, at least for these quite common clustering procedures and for the standard distance measures used.

Figure 4 displays the evolution of AUC values with increasing numbers of clusters used for the eventual classification model building. Note the logarithmic scale over the class wise

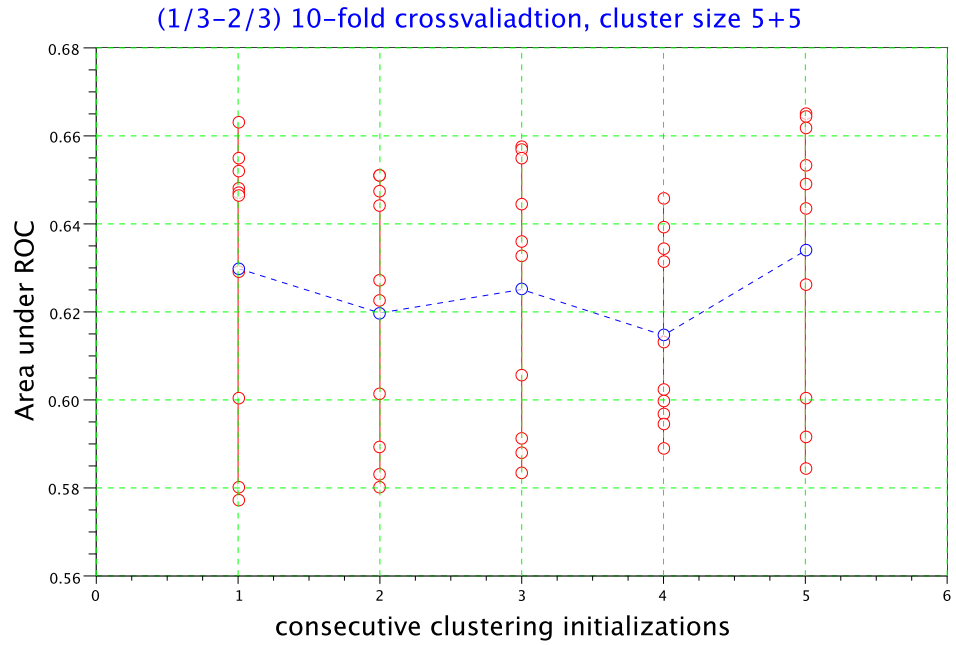

Fig. 1. AUC values of ROC curves computed on holdout data with models trained on five cluster representatives for each credit client class. Results for consecutive initializations of the cluster algorithm are shown. For every initialization ten validations over different holdout sets ($\frac{1}{3}$ of data held out randomly) are computed.

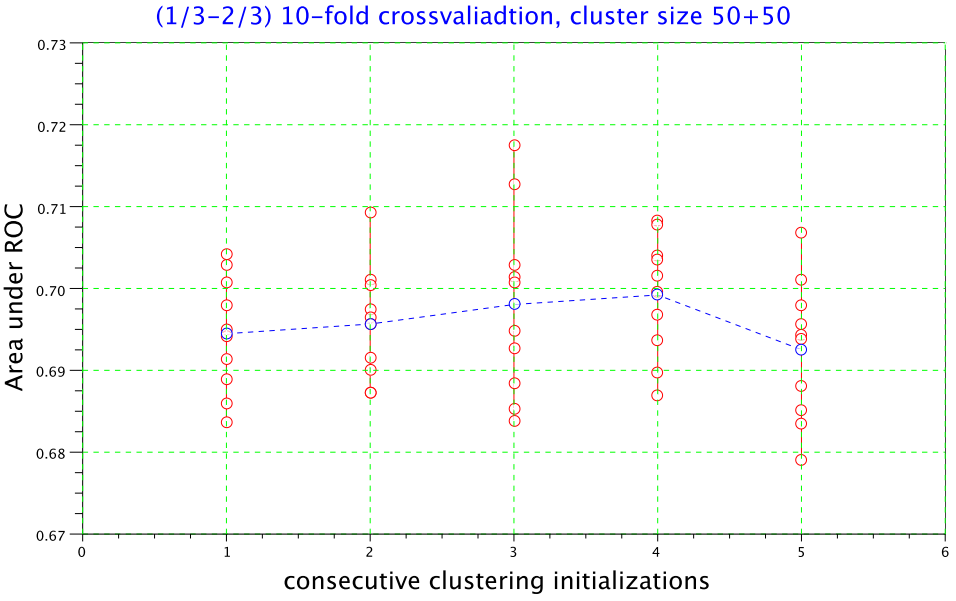

Fig. 2. AUC values of ROC curves computed on holdout data with models trained on 50 cluster representatives for each credit client class. Other conditions and remarks are the same to those from figure 1.

cluster representatives. From the figure, one is tempted to assert that at about model size approximately needed to specify a linear model ($m \approx 50$) AUC variances start to reduce and progress in terms of AUC increase slows down. Level off is clearly seen starting latest with 200 clusters per class (corresponding to $m = 400$).

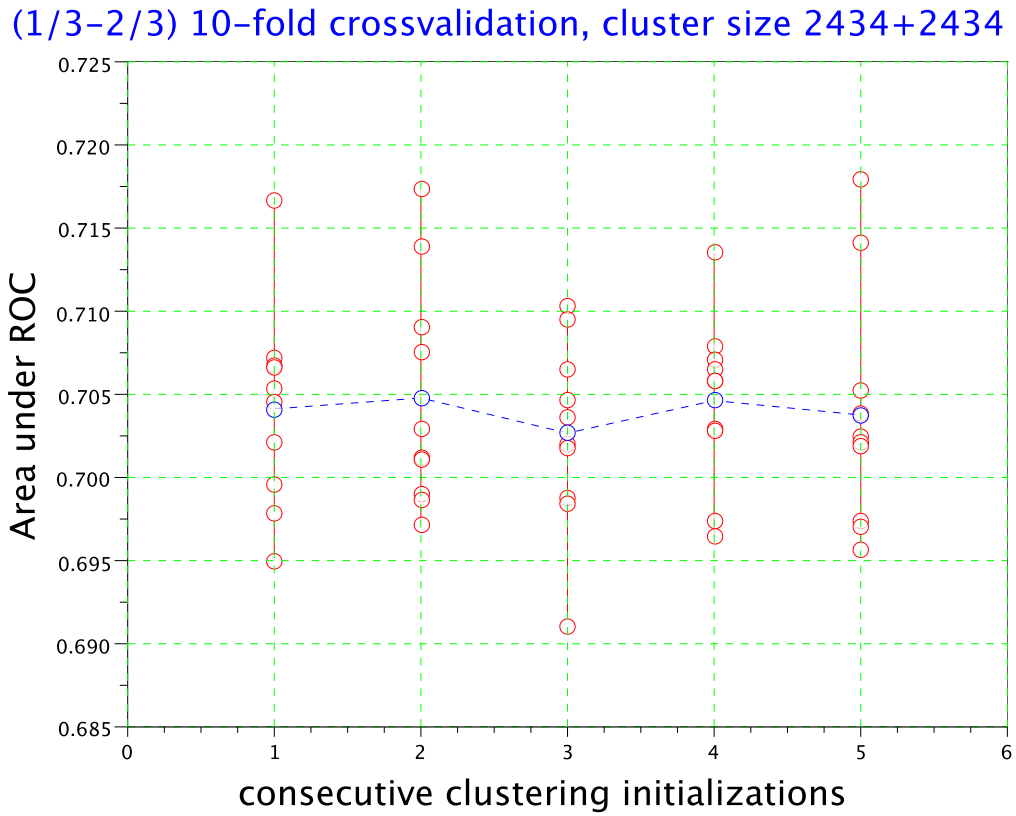

Fig. 3. AUC values of ROC curves computed on holdout data with models trained on 2434 cluster representatives for each credit client class. This number of training points is approaching that maximally possible for using $\frac{1}{3}$ of the data from the smaller class for training, while all other remarks are again the same to those from figure 1.

Finally, figure 5 depicts a smaller scale experiment, similar to that from figure 4 but with the important difference that instead of the conventional numeric encoding with $m = 25$ a symbolic encoding with $m = 43$ of the data is used. The encoding produces cluster centers which lead to much better small classification models. Variance also seems to reduce in the vicinity of model sizes capable of specifying a linear model. The leveling off effect with increasing cluster numbers is even more pronounced than with traditionally encoded data and there even seems to be a degradation of performance for larger models. Here one may be tempted to ask to which extend one can "trade" additional data dimension against the number of data clusters. These results are preliminary. They are also expected to further depend both on the symbolic encoding chosen and on the distance metric used in the consecutive modeling steps.

6 Conclusions

Our classification models trained on cluster representatives from a large credit client data set tend to produce competitive if not superior out of sample performance when compared to models trained on the entire data. In spite of our credit scoring data being highly asymmetric in terms of their respective class representatives we use equally sized training sets for each class and we compare out of sample performance by means of receiver operating characteristics (using AUC).

However, some features of this combined model building approach are leading to a certain level of uncertainty concerning the expected results. Clustering the original data varies with the starting points (e.g. random starting clusters) used and the training of powerful classification procedures themselves like SVM also depends on the suitable setting of hyper-parameters.

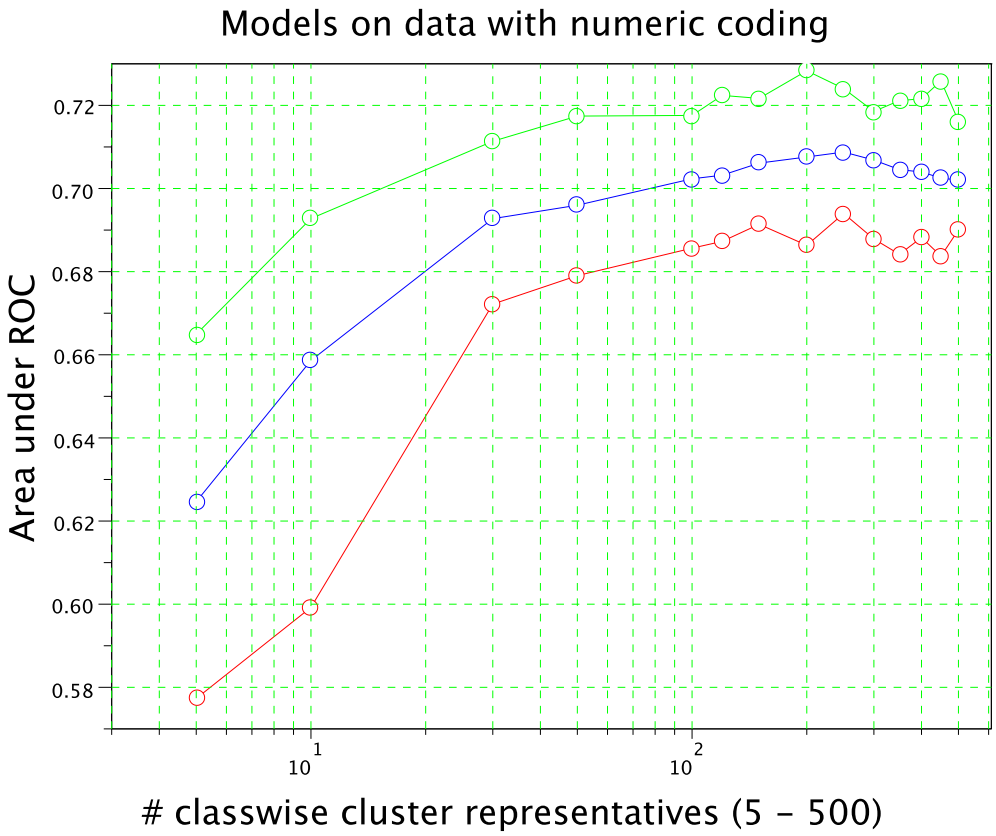

Fig. 4. AUC values of ROC curves computed on holdout data with models over models trained on an increasing number of cluster representatives beginning with 5 and ending with 500 for each class of credit clients, respectively.

Before recommending the combined model building approach to practical credit client scoring, an extensive evaluation of model performance stability is in order. This is not restricted the parameters mentioned above, but it evaluates the influence of the number of clusters used, the data encoding, and, hence, the resulting cluster representatives as well as cluster based procedures intended to improve the separation of classes. The further result of such an extended evaluation is a better data description, which enables a better understanding of the high volume, high dimensional credit scoring data. Concerning the case of traditional numeric data encoding, the computational results point towards an average out of sample performance improvement with increasing number of clusters (also averaged over repeated clustering restarts), which levels off at a moderate number of clusters way below the number of data points in the smaller class of defaulting credit applicants. A residual uncertainly still remains with regard to the method's ability of finding SVM hyper-parameters to the mainly used classification model, which are better adapted to the individual clusterings, although other tests reveal that the sensitivity of the SVM to hyper-parameters changes is generally rather low in that different SVM solutions occur occasionally by means of "jumping" from one set of support vectors to others, hereby forming the different separating functions between the classes. Furthermore, an as yet less extensive evaluation of some symbolic coding of the data and the resulting clusters points towards a somewhat different situation in terms of an increasing number of clusters used: the increase in average expected model performance is much flatter than that with classical numeric encoding and it even decreases in still higher cluster numbers. While the same residual uncertainly concerning SVM hyper-parameters applies here as well, symbolic encoding and the associated distance measure used here may be themselves suboptimal choices. While the latter results are still preliminary, symbolic encoding of the original data offers more complex representations of clustering information concerning the data and are therefore an attractive

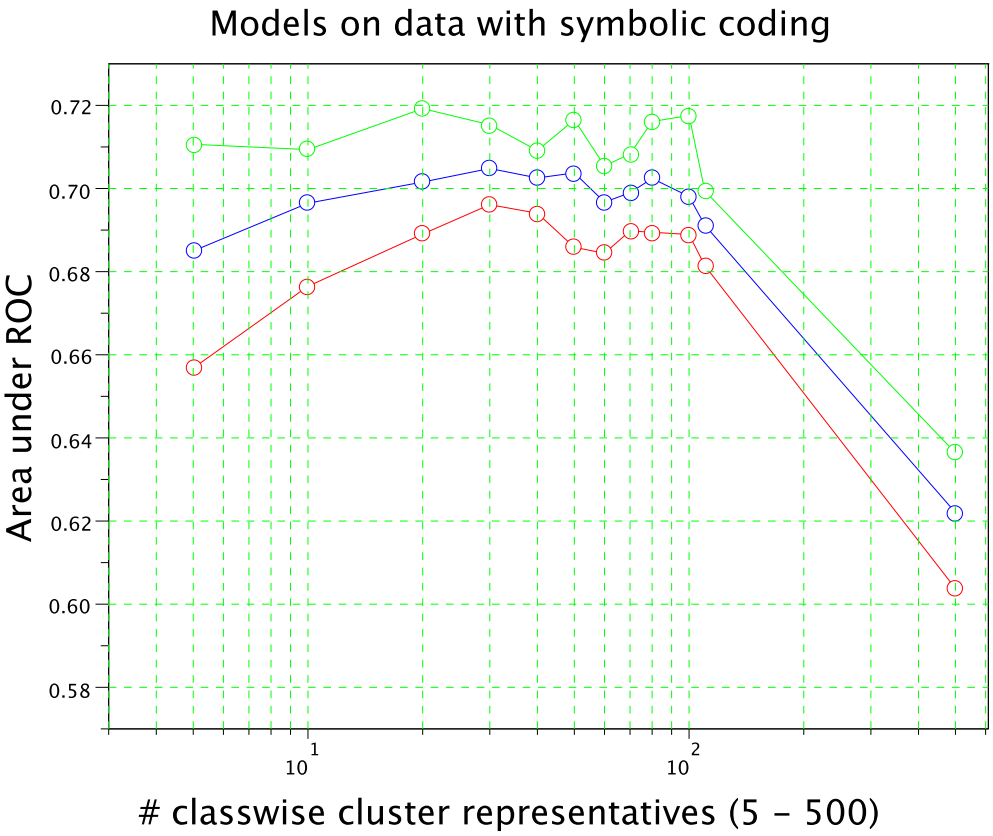

Fig. 5. Similar to figure 4 except for using a symbolic encoding of the data over $m = 43$ input feature dimensions as opposed to the traditional numeric encoding used for all models from the figures 1 - 4. Here, to the day of editing this report, AUC values of ROC curves computed on holdout for models trained on a smaller number of consecutive cluster representatives as those from figure 4.

alternative to using the raw data directly, especially when a suitable encoding would be learned during the modeling process. Research on this subject is part of future work.

Further work will also focus on combining different types of credit client data or information sources for model building and to explore procedures for designing multi-label and constrained cluster-based training approaches for credit client classification. We also plan to search for more alternative and more appropriate validation mechanisms for the contribution of the consecutively employed modeling components of producing clusterings and classification to the expected prediction performance and to model stability.

References

1. BAESENS, B., VAN GESTEL, T., VIAENE, S., STEPANOVA, M., SUYKENS, J. and VANTHIENEN, J. (2003): Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society*, 54, 627–635
2. BELLOTTI, T. and CROOK, J. (2008): Support vector machines for credit scoring and discovery of significant features. *Expert Systems with Applications, to appear.*
3. BOLEY, D. and CAO, D. (2004): Training Support Vector Machine using Adaptive Clustering. *Proceeding of 2004 SIAM International Conference on Data Mining, April 2004.*
4. BUBECK, S., MEILĂ, M. and von LUXBURG, U. (2009): How the initialization affects the stability of the k-means algorithm, `arXiv:0907.5494v1 [stat.ML] 31 Jul 2009`, 23pp
5. EVGENIOU, T. and PONTIL, M. (2002): Support Vector Machines with Clustering for Training with Very Large Datasets. *Lectures Notes in Artificial Intelligence, vol. 2308, 346-354.*
6. HUANG, Z., CHEN, H., HSU, C.-J., CHEN, W.-H. and WU, S. (2004): Credit Rating Analysis with Support Vector Machines and Neural Networks: A Market Comparative Study, in: *Decision Support Systems (DSS)*, 37(4), 543-558

7. JAIN, A.K., MURTY, M.N. and FLYNN, P.J. (1999): *Data Clustering: A Review*, in: ACM Computing Surveys, Vol. 31, No. 3, S. 264-323
8. JOACHIMS, T. (2002): *Learning to classify text using support vector machines: methods, theory, and algorithms*. Kluwer, Boston.
9. JOACHIMS, T. (2008): SVM^{light} – *Support Vector Machine Vers. 6.02*, accessed after October 2009 at `http://svmlight.joachims.org`
10. LI, S., SHIUE, W., HUANG, M.H. (2006): The evaluation of consumer loans using support vector machines. *Expert Systems with Applications, 30, 4, 772–782.*
11. LI, B., CHI, M., FAN, J. and XUE, X. (2007): Support Cluster Machine. *Proceedings of the 24th International Conference on Machine Learning, 505-512.*
12. von LUXBURG, U. (2009): High-level Summary of Theoretical Results on Clustering Stability, Max Planck Institute for Biological Cybernetics, Tübingen, 43pp
13. SCHEBESCH, K.B. and STECKING, R. (2005): Support vector machines for credit applicants: detecting typical and critical regions. *Journal of the Operational Research Society, 56(9), 1082-1088.*
14. STECKING, R. and SCHEBESCH, K.B. (2006): Variable Subset Selection for Credit Scoring with Support Vector Machines. In: Haasis, H.-D., Kopfer, H. and Schönberger, J. (Eds.): *Operations Research Proceedings 2005.* Springer, Berlin 251–256.
15. THOMAS, L.C., OLIVER, R.W. and HAND, D.J. (2005): A survey of the issues in consumer credit modelling research. *Journal of the Operational Research Society, 56(9), 1006–1015.*
16. WEISS, G.M. (2004): Mining with Rarity: A Unifying Framework. *SIGKDD Explorations, 6(1), 7–19*
17. YUAN, J., LI, J. and ZHANG, B. (2006): Learning Concepts from Large Scale Imbalanced Data Sets Using Support Cluster Machines. *Proceedings of the ACM International Conference on Multimedia, 441-450.*